# On a Hierarchical Indexing Fuzzy Content-based Image Retrieval Approach

Krisztián Veréb

University of Debrecen, Institute of Informatics,
Department of Information Technology
P.O.B. 12
H-4010 Debrecen
Hungary
sparrow@math.klte.hu

## Abstract

In this paper a two-layer content-based image retrieval model will be described. The model has a prefilter indexing function, based on a possible class hierarchy of the images to be stored. Based on this hierarchical indexing some typified queries can be introduced. The $\delta$ distance function, used in the queries can be defined by a fuzzy algorithm based on the cut operation and the fuzzy logical connectives. This aprroach is called cut-and-or-not.

## 1 Introduction

The content-based image retrieval from image databases has large literature. One can study the main approaches in [2] or in [4]. The main indexing techniques can be found in [1]. Based on these basics and our researches [6] [7] a new approach has been developed for content-based image retrieval from databases.

Generally the existing approaches do not support the OR and the NOT relationships between the spatial (image part) matchings. In most cases only the AND relationship is supported. By the way, the spatial matchings with OR and NOT relationships (in case of cut operations) induce more matchings. That is the reason why an indexing technique has to be introduced to cut down the time of queries. The use of the OR and the NOT connectives can provide an efficient solution to the problems of complex query formalisation.

The approach uses Object-Oriented concepts in the scope of indexing, and logical approaches in the scope of query formalisation. Using the indexes a prefiltering can be performed to determine which images will be used in the retrieval. Then a query image (or set of query images) can be specified. The query images and the stored images can be disintegrated into small pieces, and using more matching algorithms and fuzzy logical connectives a complex query can be formalized.

## 2 On the hierarchical indexing

The basic scheme of image retrieval from image database is as it follows: given an image database containing images and given a query image (pattern). The question is if there are any images similar to the query image. As the number of images identical to the query image is probably very few, a $\delta : \mathrm{Obj} \times \mathrm{Obj} \to \mathbb{R}_0^+$ distance should be introduced, where Obj denotes the objects storeable in the database ($\mathbb{R}_0^+$ denotes the non-negative real numbers). We have to introduce a feature vector extraction mapping $F : \mathrm{Obj} \to \mathbb{R}^k$, where $\mathbb{R}^k$ denotes the space of $k$-dimensional vectors (in more details see next section). So the $\delta$ distance can be formed as it follows: $\delta(\mathrm{obj}_1, \mathrm{obj}_2) = \delta_{vectors}(F(\mathrm{obj}_1), F(\mathrm{obj}_2))$, where $\delta_{vectors}$ is a distance (e.g., Euclidean distance) of the vectors.

Thus the following matchings can be identified:

- Identical, totally exact matching, when $\delta(\mathrm{obj}_1, \mathrm{obj}_2) = 0$.
- $\epsilon$-similarity, when $\delta(\mathrm{obj}_1, \mathrm{obj}_2) < \epsilon$, where $\epsilon \in \mathbb{R}^+$.
- NN-similarity (Nearest Neighbour), if $\forall \mathrm{obj} \in \mathrm{DB}$, $\mathrm{obj} \neq \mathrm{obj}_2$, $\delta(\mathrm{obj}_1, \mathrm{obj}_2) \leq \delta(\mathrm{obj}_1, \mathrm{obj})$.

The DB stands for the database.

The images and image elements are all objects. At first it means that an image object contains both its own charactersistics and its own processing and manipulating methods as well. This mechanism can be referred as encapsulation. Using many separate objects the problem of incompatibility may appear. To have this problem eliminated the image objects must have a common interface or some standard representation. The solution is the inheritance. Every specialized image is an image (ISA association). It means they have common operations that can be inherited and selfspecifically changed. The construction of parent classes give us an interface and unified manageability.

The inheritance tree of the objects has the same structure as that of the tree indexes. Building an index on each element stored in database tables, we get a special multi-level association indexing technique.

The technique is based on the role 'semantics' of the objects played in the object hierarchy. The 'heart' of the indexing is an ordering of the images (feature vectors) using some ordering technique in the multidimensional space. We have to divide the multidimensional vector space (by data partitioning), and with using this classification we can build up a multi-level indexing.

The first and most important thing is the type definition of the images to be stored. This typing is a hierarchical classification based on non-measureable (called associative) features of the images. It could be based on motives depicted by images. The second important thing is that these types have to be known when images are inserted into the database or retrieved. As we mentioned above this typing is a hierarchical classification, so we have to get ready for all image types storeable in the database. Their types can be represented in a tree. By the Object-Oriented terminology they can be represented in an inheritance tree.

Let us consider the class hierarchy as a type tree. We have to note that a type tree is such a class hierarchy where the associations between the classes are only inheritances, or to be more precise only simple inheritances (i.e., there is no composites, aggregations, etc.).

Let all images $\mathrm{obj} \in \mathrm{Obj}$ could be stored in the database be given. We have to create an object hierarchy and every image has to be assigned to the appropriate class in the hierarchy. Let $M$ be the number of the classes in the hierarchy $\mathcal{CH}$.

Let the classes in the hierarchy $\mathcal{CH}$ be denoted by $C_j$, where $j = 1, \ldots, M$. As the result of the classification, for every obj, $\mathrm{obj} \in \mathrm{DB}$ there exists a class $C_j,\ j \in 1, \ldots, M$, so that obj is an instance (element) of the class $C_j$. It is denoted by $\mathrm{obj} \in C_j$, so $\forall \mathrm{obj}, \mathrm{obj} \in \mathrm{DB}, \exists C_j, j \in 1, \ldots, M, \mathrm{obj} \in C_j$. It can be realized that there is no declared class for every element of Obj, only for the objects in the database. It is only a theoretical consideration. If such database exists, which could store all the elements of Obj, then there will be a declared class for all of them, so by the above mentioned consideration, the speculation will not be restricted. We have to note that because of the practical implementation of the image databases, the query image has to be considered as a stored image, so it has a class as well.

Note: A class $C_2$ is a direct descendant class (child) of $C_1$, if there is a direct line between them in the type tree, i.e., this is a direct inheritance. It is denoted by $C_1 \to C_2$. In this case the class $C_1$ is the (direct) parent of the class $C_2$. We can say, that the class $C_p$ is at a lower level of the tree than a class $C_q$, if $\exists C_1, \ldots, C_n$, where $C_p = C_1$, $C_q = C_n$ and $C_i \to C_{i+1}$, $i = 1, \ldots, n-1$. It is denoted by $C_p < C_q$. It is a symmetric relation, so one can use the notation $C_q > C_p$ as well. We can use the terminology, that if $C_p$ is at a lower level of the tree than $C_q$, then $C_q$ is at a higher level than $C_p$. (If $C_1 < C_2$, then $C_1$ is a (direct or indirect) parent of $C_2$, and $C_2$ is a (direct or indirect) descendant (child) of $C_1$.)

$\mathcal{CH}$ has to fulfill the followings:

(1) $\mathcal{CH}$ has only one root, i.e., $\exists C_k \forall C_i,\ i, k \in \{1, \ldots, M\},\ i \neq k,\ C_k < C_i$, and $\not\exists C_j,\ C_j < C_k$, $j \in \{1, \ldots, M\}$. It is denoted by $C_0$.

(2) Every class has only one direct parent, except the root, so $\forall C_i \exists C_k, i, k \in \{1, \ldots, M\}, i \neq k, C_i \neq C_0$, $C_k \to C_i$, and $\forall C_l,\ C_l \to C_i,\ C_l = C_k$.

With these conditions the $\mathcal{CH}$ is a general tree.

If we search for images stored in an image database with our type-tree, a searching criterion (specialization) is the definition of the type of the searched image. Because the tree is built up from ISA associations, the root level type is the level of the general searching, i.e., every image in the database could be matched to the query image. As we march on the leaf elements the set of the possible matched images become more and more restricted. (It comes from the Object-Oriented approach, i.e., every child inherits their parents' characteristics, so children could appear instead of their parents.) So we introduce a query, where the type (class) of the query image determines the set of the matched image classes. It is called typified query.

So three typified queries can be identified with the following result sets:

$$\text{identical typified query:}$$
$$\{\mathrm{obj} \in \textstyle\bigcup_{C_i < C_j} C_j | \mathrm{obj}_q \in C_i, \delta(\mathrm{obj}, \mathrm{obj}_q) = 0\},$$

$$\epsilon \text{ typified query:}$$
$$\{\mathrm{obj} \in \textstyle\bigcup_{C_i < C_j} C_j | \mathrm{obj}_q \in C_i, \delta(\mathrm{obj}, \mathrm{obj}_q) < \epsilon\},$$

$$\text{NN typified query:}$$
$$\{\mathrm{obj} \in \textstyle\bigcup_{C_i < C_j} C_j | \mathrm{obj}_q \in C_i, \forall \mathrm{obj}_p \in \mathrm{DB}, \mathrm{obj} \neq \mathrm{obj}_p, \delta(\mathrm{obj}, \mathrm{obj}_q) \leq \delta(\mathrm{obj}_p, \mathrm{obj}_q)\}.$$

Let some indexing technique be assigned for every class $C_i,\ i = 1, \ldots, M$. The techniques can be different as well. So theoretically we have $M$ indexes $(I_1 \ldots, I_M)$. It is an important fact that these indexes index not only the elements of the appropriate class $C_i$, but the elements of the descendant classes as well (it comes from the inheritance tree and the ISA associations). Thus at the level of $C_0$ every image is indexed. As we go towards the leaf elements, images become more specified and the number of images indexed by the given indexing technique at the given level decreases. Finally it can be declared, that an index $I_i$, $i = 1, \dots, M$ indexes the images where $\forall \text{obj}, \text{obj} \in \bigcup_{C_i < C_j} C_j$.

If the resulted query set is an empty set, the query could be generalized — step by step —, because a unique path can be determined from class $C_i$ to the root $C_0$, where $I_0$ indexes the whole database. If we also get an empty set at the level of $C_0$, with the given identical, NN or $\epsilon$ feature, the image is not in the database.

## 3 On the fuzzy approach of the $\delta$ function

Now we will describe, how can the $\delta$ distance function be calculated in a fuzzy content-based image retrieval approach, called cut-and-or-not.

Let us see some definitions [5]. We shall use the following notations: let $\mathbb{Z}$ be the set of integers, $\mathbb{Z} = \{0, \pm 1, \pm 2, \dots\}$, let $\mathbb{N} = \{0, 1, 2, \dots\}$ be the set of natural numbers, and let $\mathbb{R}$ be the set of real numbers.

If $X \subset \mathbb{Z} \times \mathbb{Z}$, $X$ is called *digital set*, and $x \in X$ is a *point*. Let $X$ be a digital set and given a function $f : X \to \{0, \dots, m\}$ where $m \in \mathbb{N}$. Then, the function $f$ is an $m$-level *digital image*. ($m$ is the number of pixelintensities.)

Now, we define the *cut* operation $\mathcal{C}$.

**Definition 1** *Let $f : X \to \{0, \dots, m\}$ be a digital image. Then*

$$\mathcal{C}_{x_1,y_1,x_2,y_2}(f)(x,y) := \begin{cases} f(x,y), \text{ if } \begin{cases} (x,y) \in X, \\ x_1 \le x \le x_2, \\ y_1 \le y \le y_2 \end{cases} \\ \textit{undefined, otherwise} \end{cases}$$

*where $x_1, y_1, x_2, y_2 \in \mathbb{Z}$.*

It can be seen, that $\mathcal{C}_{x_1,y_1,x_2,y_2}(f)$ is simply a digital image $q : Y \to \{0, \dots, m\}$ where $Y \subseteq X$ and it is given by the parameters $x_1, y_1, x_2, y_2$.

For every image $f$ there exists a finite set of features $F_i$, where $i = 1, \dots, l$. These features have finite feature domains $D_{F_i}$. The feature vectors are $\underline{d} = (d_1, \dots, d_l)$, where $d_i \in D_{F_i}$. These feature vectors can be mapped into a $k$-dimensional vector space $\mathbb{R}^k$, with elements $\underline{x} \in \mathbb{R}^k$, $\underline{x} = (x_1 \dots, x_k)$, with a feature vector mapping $\mathcal{F} : D_{F_1} \times \dots \times D_{F_l} \to \mathbb{R}^k$. This is very important because matchings in most cases are distances. How is it possible to interpret the distance?

The elements of the $\mathbb{R}^k$ are vecrtors. We can use the vector addition in every vector space, so we can define the norm $\|\underline{x}\| = \sqrt{x_1^2 + \dots + x_k^2}$, as well. If the norm is defined by the inner product, we can use the triangle inequality $\|\underline{x} - \underline{y}\| \le \|\underline{x} - \underline{z}\| + \|\underline{z} - \underline{y}\|$, where $\underline{x}, \underline{y}, \underline{z} \in \mathbb{R}^k$.

If the mapping from $\underline{d}$ to $\underline{x}$ uses an *a priori* knowledge about the distribution of vectors, or the original feature vectors cannot ensure the interpretation of the triangle inequality, the inner product cannot be used. In this case, we cannot use the triangle inequality and we have to define the norm in an other way.

The features and the feature domains are all finite. The possible feature vector mappings from these domains make up finite vector spaces. Thus, it can be proven, that these vector spaces are bounded (because they contain finite vectors with finite elements), i.e., there exist maximum distances between elements—called boundaries—in the spaces. This is the maximum norm value, i.e., $N = \max_{\underline{x},\underline{y} \in \mathbb{R}^k} \{\|\underline{x} - \underline{y}\|\}$.

The mapping $\mathcal{F}$ is very important, because we have to establish the existence of $N$, and we have to establish its finite feature. Thus, a lot of norms may have to be studied in accordance with the extracted features, i.e., the norm used by information theory or Banach spaces, etc. For the fuzzy approach of the method, we can use a weighted norm $\|\underline{x} - \underline{y}\| = \sum_{j=1}^k w_j |x_j - y_j|$, in accordance with the *a priori* distribution of the values. The $\underline{w} = (w_1, \dots, w_k) \in \mathbb{R}^k$, $w_i \ge 0$, $i = 1, \dots, k$, is a weight vector, which can represent the *a priori* knowledge of the distribution.

Now, we define the matching itself.

**Definition 2** *Let $Q_N(f,g)$ be a norm between two given digital images $f : X \to \{0, \dots, m\}$ and $g : Y \to \{0, \dots, n\}$ based on an above mentioned vector space, where $m, n \in \mathbb{N}$, and let $N$ be its finite boundary, such that $0 \le Q_N(f,g) \le N$, and $N > 0$ for every image $f$ and $g$. If $Q_N(f,g) = 0$, the digital images $f$ and $g$ are identical, so the distance between them is zero, and $N$ is the maximum distance between two images. This norm function $Q_N$ is called* matching.

The above mentioned definition does not detail which metrics has to be used to interpret the distance. We do not care about either the matching $Q$ is good or not, its technical parameters or the applied metrics. We have only one condition. In case of identical images, their results have to be zero according to the definition. In particular implementations any kind of matchings (e.g. statistical, syntactical) can be used.

To define the cut-and-or-not approach with the tools of fuzzy logic, we have to define the fuzzy logical connectives and their operations. Let us mention that the applied fuzzy logic is not part of the cut-and-or-not approach, any kind of fuzzy logic can be applied. We have to transform the result of matchings into coverable by fuzzy connectives.

The evaluated value of matching $Q_N$ according to fuzzy logic is $\frac{N - Q_N}{N} \in [0, 1]$, if $Q_N$ has uniform distribution. If we use *a priori* knowledge about the distribution of the values of $Q_N$, we can use any other mapping as well. The best solution is to transform the values of $Q_N$ as a fuzzy set into a $[0, 1]$ fuzzy interval, which has uniform distribution. (It is known from the literature that the fuzzy sets have to be comparable. If the incomparability of two fuzzy sets is minimal, they have the same distribution.)

Then we can get the following definitions:

**Definition 3** *Let $q_1$ and $q_2$ be evaluated matching values, then $q_1 \wedge q_2 = \max\{0, q_1 + q_2 - 1\}$ is a fuzzy conjunction.*

**Definition 4** *Let $q_1$ and $q_2$ be evaluated matching values, then $q_1 \vee q_2 = \min\{1, q_1 + q_2\}$ is a fuzzy disjunction.*

**Definition 5** *Let $q$ be an evaluated matching value, then $\neg q = 1 - q$ is a fuzzy negation.*

It is noticeable that the evaluation of the matching $Q_N$ according to fuzzy logic is a one to one correspondence to the set $[0, 1]$. Thus the fuzzy cut-and-or-not algorithm can be defined as it follows:

(i) Given $X_1, \dots, X_n$ and $Y_1, \dots, Y_m$ digital images, and $Q_{N_1}, \dots, Q_{N_k}$ matchings.

(ii) Compose from these the required image parts $\mathcal{C}_{x_{1i}, y_{1i}, x_{2i}, y_{2i}} X_i$ and $\mathcal{C}_{x_{1j}, y_{1j}, x_{2j}, y_{2j}} Y_j$ where $i = 1, \dots, n$, $j = 1, \dots, m$.

(iii) Doing the matchings on the respective images we get $p$ matching results $Q_{N_l}(\mathcal{C}_{x_{1i}, y_{1i}, x_{2i}, y_{2i}} X_i, \mathcal{C}_{x_{1j}, y_{1j}, x_{2j}, y_{2j}} Y_j)$ where $l \in \{1, \dots, k\}$, $i \in \{1, \dots, n\}$ and $j \in \{1, \dots, m\}$.

(iv) Evaluating these matchings according to fuzzy logic we get $q_1, \dots q_p$ evaluated matching values.

(v) Make a fuzzy logical formula from these values $q_i$, $i \in \{1, \dots, p\}$ by the help of fuzzy connectives and evaluate it.

If the evaluated formula according to fuzzy logic is true, the answer to the question formulated by the cut-and-or-not formalism is yes, otherwise not. If this fuzzy cut-and-or-not approach is only executed on two images ($X$ and $Y$, where $X, Y \in \mathrm{Obj}$), it can be used for the fuzzy approach of the $\delta$ distance function of the content-based image retrieval as well.

Note: in most cases the values $q_i$, $i \in \{1, \dots, p\}$ are weighted with some weights $w_1, \dots, w_p$, where $w_i \geq 0$, $w_i \in \mathbb{R}$, $i = 1, \dots, n$. If there exists $i$ such that $w_i > 1$, thus it is possible that the weighted value will be greater than one. In this case the above mentioned fuzzy connectives cannot be applied. We mentioned before, that the applied fuzzy logic is not part of the cut-and-or-not approach, any kind of fuzzy logic can be applied. Thus one can use the following generalized connectives as well:

**Definition 6** *Let $w_1 q_1$ and $w_2 q_2$ evaluated weighted matching values, then $w_1 q_1 \wedge w_2 q_2 = \min\{w_1 q_1, w_2 q_2\}$ is a generalized fuzzy conjunction.*

**Definition 7** *Let $w_1 q_1$ and $w_2 q_2$ evaluated weighted matching values, then $w_1 q_1 \vee w_2 q_2 = \max\{w_1 q_1, w_2 q_2\}$ is a generalized fuzzy disjunction.*

**Definition 8** *Let $wq$ evaluated weighted matching value, then $\neg wq = \max\{0, 1 - wq\}$ is a generalized fuzzy negation.*

## 4 On the evaluation

We use an Oracle9*i* ORDBMS for the implementation of the approach, because the Oracle *inter*Media is its basic feature. The methods are coded in PL/SQL. The database contains approximately 160 images with dimensions $1500 \times 1500$. The size of them is about 1 GByte. The used matching algorithms are the native Oracle *inter*Media Visual Information Retrieval [3] algorithms (namely the Color, the Texture and the Shape matchings).

The use of the cut opeartion in real-time is not too time-effective, that is the reason why the cut part of the images are stored. So, the database contains the original images and disintegrated versions of them, where 9 image parts build up the original image. They are cut and stored at the insertion of the original image. Thus additionaly 1440 image parts are in the database with dimensions $500 \times 500$.

The system has a query by example HTML (with the Oracle PSP technology) interface, where a query image can be determined with the help of (about 350) auxiliary images. The fuzzy connectives can be set up via this interface as well. A three level hierarchical classification was the base of the indexing.

The stored images are from general image archives as from

- NSSDC Image Catalog, NASA, `http://nssdc.gsfc.nasa.gov/imgcat`,
- VRoma Image Archive, `http://www.vroma.org/images /image_search.html`,
- Mayang's Free Textures v8.1, `http://www.mayang.com/textures`

Based on the tests we can say that the hierarchical indexing can cut down the time of the queries while the cut-and-or-not approach ensures major flexibility in the complex queries. The average time of query is about 6.29 ms without the use of hierarchical indexing. With its help the average query time is about 1.92 ms.

Because our system is a web-based system, we examined the following web-based systems as well:

- 1: Amore (NEC), Advanced Multimedia Oriented Retrieve Engine, `http://www.ccrl.com/amore/`
- 2: Blobworld, `http://elib.cs.berkeley.edu /photos/blobworld/start.html`

- 3: CIRES, Content-based Image REtrieval System,
`http://amazon.ece.utexas.edu/~qasim/cires.htm`
- 4: NETRA,
`http://maya.ece.ucsb.edu/Netra/netra.html`
- 5: SIMPLIcity, PennState University, Multimedia Information Technology Research Group,
`http://jzw.stanford.edu/IMAGE/simp_java/`
- 6: PicToSeek (Zomax),
`http://zomax.wins.uva.nl:5345/ret_user/`

These systems can be classified by the following criteria: Whether they use Color, Shape or Texture matching or not (Color, Shape and Texture columns), spatial (image part) matchings or not(Parts column), and whether the result values can be weighted or not. (In the table, the system No. 7 stands for our system).

| No. | Color | Shape | Texture | Parts | Weights |
|---|---|---|---|---|---|
| 1. | + | - | - | - | + |
| 2. | + | + | + | + | + |
| 3. | + | - | - | - | + |
| 4. | + | + | + | + | + |
| 5. | + | + | - | - | - |
| 6. | + | - | - | - | - |
| 7. | + | + | + | + | + |

It can be seen, that only the systems No. 2 and No. 4 can use the image part matchings with Color, Shape and Texture matchings and weights. But neither of them supports the NOT and the OR connectives in the image part matchings. It is only supported by the fuzzy cut-and-or-not approach.

## References


[1] Elmasri, R., Navathe, S. B., *Fundamentals of Database Systems*, The Benjamin/Cummings Publishing Company, Inc. (1994)

[2] Lew, M. S., *Principles of Visual Information Retrieval*, (ed.), Springer, (2001),

[3] Oracle interMedia User's Guide and Reference, Release 9.0.1, Part Number A88786-01

[4] Santini, S., *Exploratory Image Databases, Content-Based Image Retrieval*, Academic Press, (2001),

[5] Sonka, M., Hlavac, V., Boyle, R., *Image Processing Analysis and Machine Vision*, PWS Publishing, 2nd. ed. (1998)

[6] Veréb, K., *Image Objects in Object Relational Database Management Systems*, European Conference in Object Oriented Programming, Poster, (2001),

[7] Veréb, K., *Research Trends on Objedt-Oriented Image Databases*, Informatika a felsőoktatásban, in hungarian, (2002), 975–981