Wavelet Transform Application to Fast Search by Content in Database of Images

Dimo T. Dimov

***Abstract*—The paper presents a fast access method to images in a conventional DB (DataBase). The method is based on two-dimensional wavelet transform of images preliminary normalized by size, orientation and intensity. The image content for search is considered the normalized image graphics that should be well localizable into the input query picture. The most essential image data are represented as a key of fixed length, on which the fast access is performed using the index access methods of a conventional DB management system. The method is experimented on a DB of about 4000 images of trademarks.**

***Index Terms*—image DB-s, image retrieval by content, fast data access method, wavelet transform application.**

I. Introduction

Recent advance in image retrieval can be summarized in two approaches – representation and learning [1]. Machine learning is hardly to consider a fast approach in the case [1]. A good amount of image DBs now relay on representation approaches [2]. Color, tone and texture features are preferred to image shape because of segmentation troubles [1, 2, 3]. Besides, most of them rely on sequential search into DB.

II. Aim Description

The basic idea hereinafter is to develop a fast access method for searching images by their content, however the method to be easily implemented into conventional DB-s for efficient image retrieval. At this stage, images will be considered most of all graphic images (gray scaled or colored), and the image content will be considered the image itself perhaps being preliminary suitably normalized and/or simplified.

It is well known practice of using conventional DB for keeping images recently. There are generally two classical approaches for that [3, 4, 5, 6, 7, 8, 9], namely:

- by keeping a table of pointers to (or of the names of) the image files that are written anywhere outside the DB, or
- by keeping the images themselves into a table of BLOB (Binary Large OBject) fields.

Both the approaches rely on the convenience of searching the images using conventional access methods of a DBMS (DB Management System), namely – sequential access methods, relative sequential ones, and especially the faster ones – indexed-sequential (or parallel) ones, direct hash-access ones, associative ones, etc. [3, 10, 11, 12]. All fast access methods work using the convention of a "key" and give the search result as follows:

$$\left\langle \text{the key value of the retrieved object (record)} \right\rangle \prec \quad (1)$$

$$\prec \left\langle \text{the key value of the searched object (record)} \right\rangle ,$$

where "(i) $\prec$ (ii)" means "(i) is the greatest value less than or equal to (ii)". The key usually corresponds to a feature (or a combination of features) defined for all DB objects of a given type. In the simplest case, key values are preliminary written in a field of a table (for the objects of given type). As a rule, key fields should be simple fields of fixed and not very long length.

As well as the contemporary object-oriented DBs inherit the well spread relational DBs, which in their turn inherit the old-fashioned DBs of network type, and all of them posses as a minimum a kind of indexed-sequential access method, the choice for experiments has been done on:

- a DB of relational type to keep images, as well as on
- an indexed-sequential access method to realize the search strategy here proposed.

A simple logical scheme of similar DB for keeping images is illustrated on Fig.1.

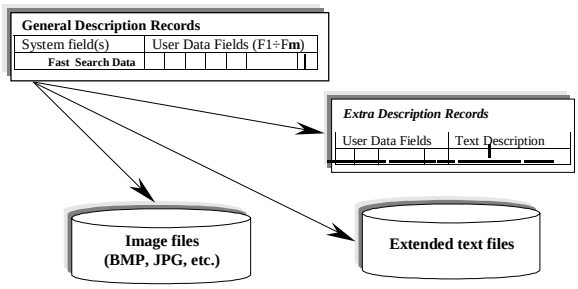

Fig. 1. Simple logical scheme of an image DB.

Most of practical cases of efficient (fast) DB image access rely on simple description of the searched content that is relatively easy to be derived automatically from the input query image, and because of that suffer of low recognition ability. More sophisticated methods for content derivation are, as a rule, not well suitable to basic access methods of conventional DBMS, and consequently are high computer power consuming, i.e. not enough effective ones.

In this work, an effective access method to DB images is proposed that well corresponds to the conventional DBMS cannon, namely – DB objects to be accessed as a whole, and

This research has been elaborated in the frames of the project "Access Methods for Image Databases" supported by research funds of the Bulgarian Academy of Sciences (BAS) under the Institute of Information Technologies' control (IIT No. 010043).

D. T. Dimov is with the Institute of Information Technologies, BAS, Sofia, Bulgaria (telephone: +(359 2) 70 64 93, e-mail: dtdim@iinf.bas.bg).

eventual small differences to be considered admissible noise, i.e. in the DB tolerance of sensitivity.

II. Image Search Strategy

A. Basic type images of interest

Graphic images that will be of main interest hereinafter can be considered as reproduced by a small number of colors or half tones (i.e. gray intensities), and the areas for filling are of relatively not very small size. Besides, these images should be well capable for centralized localization, i.e. the essential graphics to be in the "middle" of the image frame, while the area closed to the frame to be filled only with background (also color or half-tone). Examples of similar images are illustrated on Fig. 2. Like images are intensively used in patent offices' activities, e.g. for registration of companies, firms, etc., and are popular by the name "trademarks", or simply "marks" [13, 14].

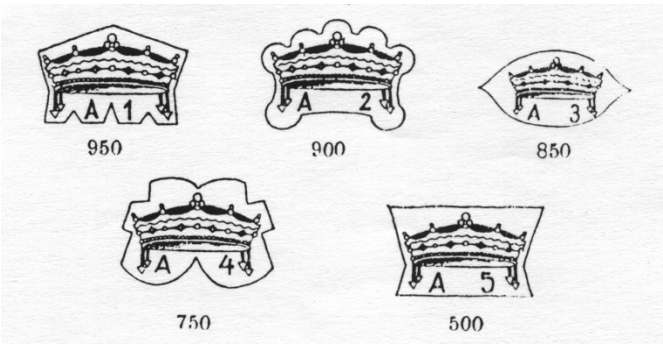

Fig. 2. Image examples suitable to the method proposed (Bulgarian hallmarks on silver registered 1903-1910, according to [13]).

B. Requirements to the image key definition

- The key should consist of the (most) essential information for the image.
- This key information should be of significantly smaller volume than the image itself.
- The key information volume should be written in a 1/D array of fixed length.
- The key information should be structured in a way that essential parts to appear in frontier positions (in the array). In this way, it will be possible simultaneously to reduce the key length of all images into DB. Besides, the length of DB image keys can be chosen corresponding to the quantity of the structured image information.
- The image key is not obligatory to be unique; i.e. there can be more than one image to correspond for a given key of length L.
- All DB images of one and the same key should be considered as equivalent images. For instance, if the length L increases, then the number of equivalent images for a key decreases and v.v., i.e.,

- The length L should be chosen in such a way that the image key to be enough informative one.

Thus, the following intuitive *definition* about keys can be written: The "*image key*", or simply the "*key*" should consist of the most essential information of given image, structured in descending order into a one-dimensional (1/D) array of fixed length.

C. Wavelet transform approach to represent image content

A large number of pyramidal representations of images are well known recently [15, 16, 17, 18]. Most of them can be considered trees of sub-images. Each sub-image at given level of the tree (i.e. the pyramid) is decomposed into a number of, usually 4 (for quad-tree), or 2 (for binary tree) sub-images (successors), most of all of equal frame, and each of them being modified in a way emphasizing the information essential for the respective method. Besides, the pyramid (the tree) can be formally represented (often without loss of information) in a pixel matrix of size equal to the one of the initial image (at the root). Consequently as higher we climb the pyramid so more compressive or more essential is the image information at the respective level there. In opposite we go down in the case of so-called 'inverse' pyramid [15].

By many reasons (accenting on (iii) and (vi), cf. the end of this section), the image decomposition approach for the key performance herein has been chosen among the pyramidal representations that use Wavelet transform (WT) to converge sub-images from level to level of the tree, [17, 18, 19].

Being one more new attraction till the end of 90-s, WT became a very important tool of signal processing area in 1989, when Mallat published his fast wavelet decomposition and reconstruction's algorithm for discrete WT [17], now well-known as a two channel subband coder using conjugate (or mirror) quadrature filters [18, 19], cf. also Fig. 3 and 4.

The 2/D variant of Mallat's algorithm can transform the image into four sub-images – an approximated one and thee detailed ones, each of them of four time smaller area. Applying multiply this scheme just for each next approximated sub-image, the classical WT decomposition chain-tree (cf. Fig. 5) of given image can be produced [17, 18]. Applying the same scheme, but for all sub-images at given current level of decomposition, the so-called "wavelet package" (cf. Fig. 6) can also be produced [18, 19].

This wavelet package that is in fact a quad-tree representation of image, has been chosen for the necessities herein. Besides of two-dimensionality it will be called hereinafter a "wavelet image package" (WIP).

The following features of WT are very akin to our intuitive requirements to DB keys, namely:

(i) WT allows performing a local analysis, i.e. on localized areas of given large image, in difference, for instance to the harmonic analysis of Fourier [20].

(ii) The original image can be decomposed on approximations corresponding to high scale, i.e. low frequency components, and details – to low scale, i.e. high frequency components, what is very convenient for suppressing the statistically regular noise.

(iii) WIP represents the given image into statistically independent sub-images that allow interpreting them in several ways of enumeration, by outer criteria.

(iv) Each level of WIP can be easily enriched by a function of sub-images' significance; in particular a similar function at the last level of WIP will be very suitable to our case of image key creation.

(v) WIP is fully convertible; i.e. whatever sub-image can be reconstructed by the set of its successors of a lower level.

(vi) Nevertheless that WT is strongly defined in the space of continuos functions, it exists an effective algorithm for computer calculus of WT [17, 18, 19].

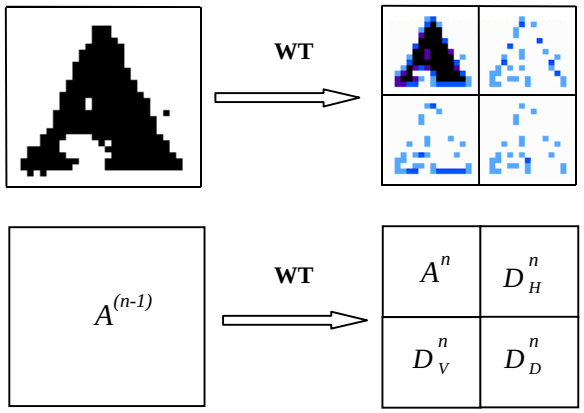

Fig. 3. The 2/D wavelet transform - a visual interpretation.

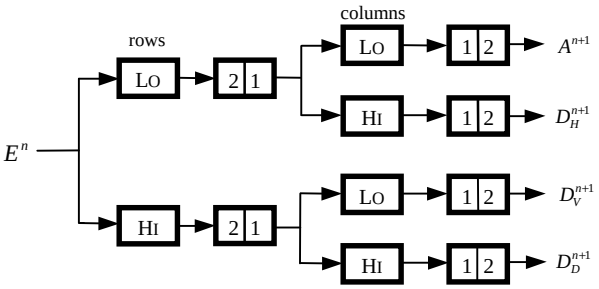

Fig. 4. The 2/D WT decomposition scheme at given level *n*.

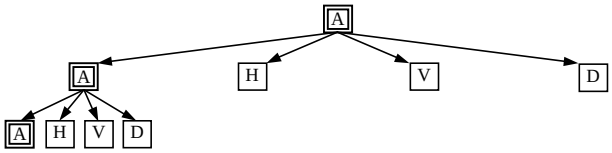

Fig. 5. Wavelet chain-tree of 3 levels.

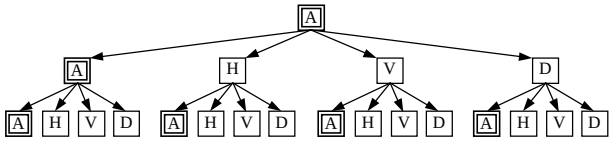

Fig. 6. Wavelet image package of 3 levels.

D. Strategy for creation of the image key

The creation of image keys can be defined on the last level leafs of the image WIP. For this aim, the structure like DWT2 of the respective approximated and detailed maps of wavelets' coefficients [19] has been considered as basic inner representation of the input image, cf. Fig. 3, 4 and 6.

The input image to key creation should be preliminary normalized on size and orientation. For some difficulties met by orientation see section IV-A.

It has been considered as noise all the differences between the original input image and the reverse image obtained after respective reverse WT over the image approximation represented by the key. Besides, in cases of graphic images, the shape is considered much more essential than the color. That's why the original images (true color or gray scale ones) are preliminary converted to black-and-white (B/W), i.e. to binarized images, cf. [21, 22] for details. The binarization herein should be caught as a mean against possible sensitivity of the WT to intensity differences among images of equal shape, cf. Fig. 8, 10 and 12, vs. Fig. 9, 11 and 13.

The DWT2 structure represents a quad-tree, where the normalized image is decomposed on its WT-parts, i.e. sub-maps of wavelet coefficients, cf. Fig.5 and Fig.6. No special attention is put on the WT filters; at this stage, they are chosen Daubeshies' ones *db*(*n*), *n*=1÷6, as given in [19], and because of the binarization the *db*(1) is used for experiments.

Leafs of the tree, i.e. the sub-maps situated on the bottom level of the tree, are of main importance for key creation. These sub-maps are of minimal volumes. The quadruples of them, each containing of one approximated and 3 detailed sub-maps, are successors of their summarizing sub-map of the upper level, each of them being also an approximated or a detailed sub-map, and so on to the root, i.e. to the initial whole image (considered an approximated map).

In this way each leaf-sub-map can be represented by its path from the root, and its importance for the key creation primary depends as on the number of approximated sub-maps in the path as of how close are they to the root. Another peculiarity of this criterion to key creation is sorting of leafs using a specific recursion (sorting scheme) to visit only leafs, and preferably the most essential of them. Each next leaf content following the recursion is averaged to a byte size before entering its place in the key.

The sorting scheme can be the well-known Z-scheme, cf. [3]. But any way, the 'optimal' Π-scheme on Hilbert curves [3] is not recommended in the considered case, because of our accent first of all on approximation, and secondly on horizontal and vertical details instead of diagonal ones.

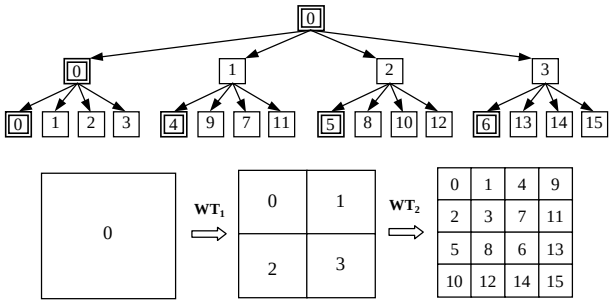

Fig. 7. Possible sorting scheme of the leaves of WIP.

On Fig. 7, a heuristic idea is illustrated that is currently investigated for rearranging the canonic Z-scheme. The front part of the leafs-order following the proposed sorting scheme has been finally used making the image keys of experiment.

E. Primary Algorithm for Image Search

All DB images should be sorted by their key values that well corresponds to the essence of the conventional indexed access methods of now-a-days DBMS-s.

As far as various DBMS-s presume different restrictions for search performance by a key field of a relatively large length (*Key_Lng*, in our case it equals 510 bytes), an extra accompanying field to the image key field was necessary to be implemented in the respective DB table here. The accompanying field is to keep the image sequential number in the way that images are ordered by their key values. In difference to the key field the accompanying field can be a simple numeric field on which a conventional indexed access can be easily applied using whatever DBMS known.

In this way, the image search algorithm (e.g. called ***LocateImage***) can be reduced to a classical binary search on this accompanying field, where the total access number to DB is around $\log_2(DB\_size)$, *DB_size* – the number of images into DB. At each DB access by the accompanying field, the respective image key is read from DB, for being compared with the key of input image, which location into DB is looked for. Besides of the found image ID (its identification number into DB), *LocateImage* should give an extra output of Boolean type, defining both cases: (i) the *exact match* case, when a same key image already exists into DB, and (ii) the *best match* – if it does not.

Next, the visualization module can derive a number of the most similar images in the DB, counting back and/or forward from the found *Best/Exact_match* location in accordance with the distance to the input image.

Of course, *LocateImage* can also check the inner memory resource and if there is enough, it might attempt to load the keys of all DB images herein to perform the binary search speedier. By the way, for each computer, it can be looked for an optimum by time between both variants – directly into DB or by means of an inner memory buffer.

F. Append, Delete and Reindex

AppendImage can be the name of the procedure realizing a new image entering into DB. The peculiarity here comes with the necessity to keep the correspondence between key fields and their accompanying fields, which are especially designed for the procedures of *LocateImage* type. For this reason, a special *ReindexDB* operation should succeed each operation of *AppendImage* type.

On the contrary, the other standard operation, namely ***DeleteImage*** do not need any re-indexing after completion.

ReindexDB should be considered as specific procedure for keeping the DB consistency, because of the accompanying fields. Practically, *ReindexDB* should read the whole DB and actualize the accompanying fields' content in accordance with the changed key order after appending of new image into DB as well as in other cases: e.g. after possible batch loading of new patch of images, or at the start up check of DB consistency if necessary, etc.

Various versions of *ReindexDB* procedures can be realized, most of all being attempts to accelerate the operation. By principle, the accompanying fields can be avoided using DBMS that maintains indexed access methods on long fields (longer than 255 bytes). But, as usual, this feature is not kept by the simpler DBMSs as Paradox and MS-Access, which has been preferred at the stage of research.

G. Distance function

The distance function for DB images has been defined canonically by definition of a norm over the space of keys, independently from the key type and the DB size, namely:

$$0 \le \quad \boldsymbol{DBNorm}(Key) = M^{-1} \sum_{i=0}^{I-1} \boldsymbol{k}_i b^{(I-i-1)} \quad \le 1 \qquad (2)$$

where $I = Key\_Lng = 510$ is the byte length of a ***Key*** = (k_0, k_1,… k_i,… k_I), $0 \le k_i < b$, b=256, and the normalization coefficient M equals the possible maximum for *DBNorm* summation, $M = 2^{4080}$-1. Obviously, the minimal *DBNorm* value corresponds to $Key(k_i \equiv 0,\ i = 1, 2,\ldots I)$ and the maximal one – to $Key(k_i \equiv 1,\ i = 1, 2,\ldots I)$.

In other words, if we imagine a key as a binary number, 4080 bits long, then the *DBNorm* represents the key as a number in a 256-based calculus system. And the "tautology" here is only in a theoretic aspect; because, practically a natural loss of information appears representing 510 bytes long array into 8 bytes variable (of type *double* in C++), i.e. DB-norm is an approximated representation of key. Nevertheless, this single variable representation of keys is very useful performing rough and easy calculations for the visualization necessities.

As generalization, the following relation can be written for images, keys and DB-norms:

$$Image \succ Key \succ DBNorm\,, \qquad (3)$$

where $\succ$ means "is represented approximately by".

The distance between two images can be represented by their norm (2), as follows: (4)

$$\boldsymbol{DBDistance}(\boldsymbol{Img}_1, \boldsymbol{Img}_2) = \left| DBNorm(\boldsymbol{Key}(\boldsymbol{Img}_1)) - DBNorm(\boldsymbol{Key}(\boldsymbol{Img}_2)) \right|$$

Of course, the images' order by norms corresponds to the order by keys in the frames of the norm precision. I.e., the cases of images of equal norms should be expected much more often that the ones of equal keys.

In this way, the DB images can be also considered as positioned in a 1/D Euclidean space, i.e. on a line-cut (0,1) defined by the image norm.

III. EXPERIMENTS

The proposed method is experimented on a DB of about 4000 images of hallmarks, scanned from [13].

The experimental image retrieval system is written on Borland C/C++ Builder v.5.0. Respectively, the DBMS used is Paradox. An IBM compatible PC has been used for development and experiments – CPU Intel Celleron 330MHz, MM 128MB, and HDD 15GB.

Figures 8÷13 are produced using the tools of experimental system. A French "bigorne" [13] is chosen here as a query image of average complexity. Figures 8, 10 and 12 illustrate the processing of the original query, while 9, 11 and 13 correspond to the binarization option preliminary chosen.

The DB access time is averaged to 8 sec. per simple query (*LocateImage*) and to 15 sec. per a query with extra fault tolerance measures (*LocateImageSpecial* - not presented herein). This timing is reducing to about 5 times if all the DB keys are preliminary read in the main memory (that is considered similar to index access as timing). The processing

time for a key creation (it does not depend on DB) is assessed to 0.6 sec., where the WT filter is db1 [19] and the normalized image size is 128x128 pixels, what leads to maximal depth of WIP equal to 6 levels.

The recognition extent is difficult to evaluate at this stage because of not solved problems generally with the WT sensitivity to rotation jet (cf. next section IV-A). Of course, the recognition rate tops 100% for images which exact copies exist into DB. But the final aim is a plausible retrieval of noised images that still cumbers the proposed WT approach. Meanwhile, experimental results of another approach to key creation (a heuristics on the tree of image contours) [24], show to about 98% of recognition rate. The WT approach discussed here is expected to approve this percentage.

IV. Problems to Solve

A. Image normalization

Because of WIP is very sensitive to image size and position, and especially to image orientation, a preliminary normalization to each image entering the DB should be performed. For the time being, the normalization has been organized as follows:

(1) the most probable symmetry axis of the image is computed on the base of inertial moments of second order over the binarized image;

(2) a back rotation of the image to vertical position of the symmetry axis is performed as well as a scale of the resulting image to a fixed size *NORMSIZE*.

The *NORMSIZE* is chosen equal to 128 by reasons of not very large volume of DWT2 structure for the images.

This approach works not very stable each time because of a great number of images of potential interest have shapes of manifold symmetry, higher than twofold, like equilateral triangles, tetragons, pentagons, hexagons, etc. An interesting method to evaluate the angle of occasional rotation of manifold symmetry figures is known by Reiss [23]. Unfortunately, the experiments following him do not succeed to this end and the reason about is under investigation now.

B. Sorting of leafs and recursion depth

The improvement of sorting scheme for WIP leafs is the second direction of future research and development of the method proposed. Besides the intuitive scheme realized in the experimental system, as illustrated of Fig. 7, it should be soon experimented another one emphasizing only on approximated entities of the WIP.

Fig. 8. An original mark image.

Fig. 9. The binarized image.

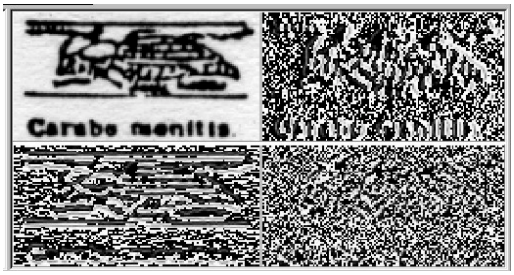

Fig. 10. First WT of the original image.

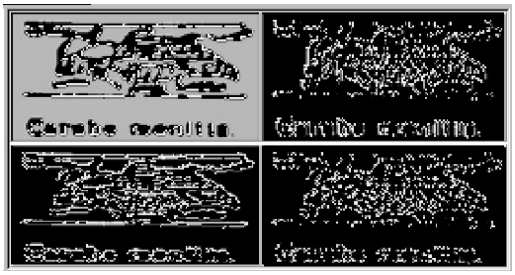

Fig. 11. First WT of the *binarized* image.

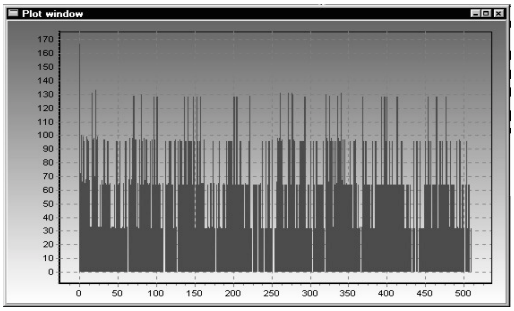

Fig. 12. The key of original image.

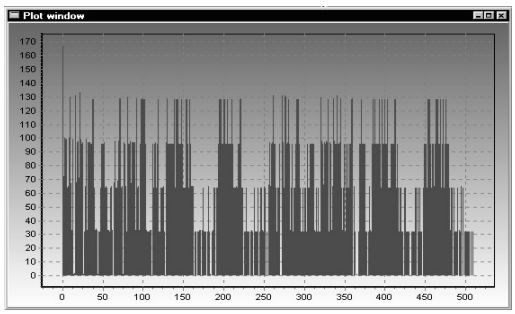

Fig. 13. The key of binarized image.

V. Conclusion

A fast access method to images in a conventional DB has been proposed. The method is based on two-dimensional wavelet transform of images preliminary normalized by size, position, orientation and intensity. The image content for search is considered the normalized image graphics that should be well localizable into the query picture. The most essential image data are represented as a key of fixed length, on which the fast access is performed using the index access methods of a conventional DB management system. The method is experimented on a DB of about 4000 image of trademarks. Development problems of a near future solution are also discussed.

From the viewpoint of pattern recognition theory, the proposed method can be considered as a recognition method by closeness to the centers of the classes for recognition. And the centers here are the images themselves already entered into image DB, while the classes should be defined by the half distances between each couple of DB images, on the line of images sorted by their keys. In this way of thinking, a distance function (4) has been introduced herein only for a better comprehensiveness, besides that it can be effectively used for better visualization of results of a user system developed by the proposed method.

The proposed method accents on image shape instead of other image features as intensity, color and texture. By reasons of traditional difficulties with image segmentation this approach is usually considered hard to develop [1]. But the proposed interpretation of WT seems to escape the necessities of direct segmentation. And no grate troubles are expected with the current problems mentioned in section IV.

On the contrary to the known methods performing sequential search by distance (or similarity) functions, the proposed method searches by key information about images, i.e. it can naturally use the well known fast access methods of conventional DBMS to speed-up the search procedure. Consequently, as for time effectiveness, the method preserves the well-known "good" limit for search time that is proportional to $\log_2(DB\_size)$. As opposite, the known methods based on image shape rarely drop out the limit of 1/2 *DB_size*, generally because of sophisticated definition of image similarity there that tends to a sequential search.

The method proposed can be applied as image search engine in a conventional DB of images, for instance in information and image retrieval systems for marks, hallmarks, trademarks, postmarks, etc. The method can be also implemented as sub-system for preliminary identification of the necessary standard into a systems for precise verification of replicas of this standard, like the system described in [25], even in another applications, for instance in forensic expertise, banking, etc.

At this stage of research and development, the proposed method power is restricted to the set of pictures (images) of well localizable essential graphics (gray scale or colored) vs. the picture background. Besides, the essential graphics should not contain any great level of noise, and especially an "artificial" noise (artifacts). The improvement of the method efficiency against such kind of noise will be the matter of near future work.

References

[1] N. Vasconselos and M. Kunt, "Content-based retrieval from databases: current solutions and future directions", in *Proceedings of ICIP'2001*, Oct. 7-10, 2001, Thessaloniki, Greece: IEEE, Vol. 3, pp. 6-9.

[2] P. L. Stanchev, "Content based image retrieval systems", in *Proceed. of CompSysTech'2001*, June 22-23, 2001, Sofia, pp. P.1.1-6.

[3] C. Zaniolo, S. Ceri, C. Faloutsos, R. T. Snodgrass, V. S. Subrahmanian, and R. Zicari, *Advanced Database Systems*, San Francisco: Morgan Kaufmann Pub., Inc., 1997.

[4] D. Y-M. Chan and I. King, "Genetic algorithm for weights assignment in dissimilarity function for trademark retrieval", in *Visual Information and Information Systems*, D. Huijsmans, A. Smeulders, Eds. Lecture Notes in Comp. Sci. 1614, Berlin: Springer-Verlag, 1999, pp. 557-565.

[5] J. Vleugels and R Veltkamp, "Efficient image retrieval through vantage objects", in *Visual Information and Information systems*, in D. Huijsmans, A. Smeulders, Eds. Lecture Notes in Computer Science 1614, 1999, Berlin: Springer-Verlag, pp. 574-584.

[6] A. Kutics, N. Nikajima, and T. Ieki, "An object-based approach to similar image retrieval", in *IEEE-EURASIP'1999 Workshop on Nonlinear Signal and Image Processing (NSIP)*, July 20-23, 1999, Antalya, Turkey, paper No.161.

[7] K. V. Chandrinos, J. Immerker, and P. E. Trahanias, "ARHON: a multimedia database design for image documents", in *Proceed. of EUSIPCO'1998*, Sept. 8-11, 1998, Rhodes, Greece, pp. 1705-1708.

[8] J. J. Hull, "A database for handwritten text recognition research", *IEEE Trans. on PAMI*, vol. 16, No. 5, 1994, pp. 550-554.

[9] K. Reinsdorph, *Teach Yourself Borland C/C++ Builder in 14 Days*, Indianapolis: Borland Press, Sams' Pub., 1998.

[10] J. Martin, *Computer Database Organization*, Englewood Cliffs NJ: Prentice-Hall, Inc., 1975, Russian translation "MIR", Moscow, 1978.

[11] E. Ozkarahan, *Database Machines and Database Management*, Englewood Cliffs, NJ: Prentice-Hall, Inc., 1986, Russian translation "MIR", Moscow, 1989.

[12] S. Khoshafian, *Object Oriented Databases*, New York: John Wiley & Sons, Inc., 1993.

[13] International Hallmarks on Silver, collected by TARDY, Paris: TARDY, 1981, pp. 429-523.

[14] Official bulletin of the Bulgarian Patent Office, ISSN 1310-179X, Index 20311, Vol.10, Sofia, 2001, (in Bulgarian).

[15] R. Kountchev and J. Ronsin, "Inverse pyramidal image decomposition with 'oriented' surfaces", in *Proceed. of Picture Coding Symposium*, April 1999, Portland, USA, pp. 395-398.

[16] K. H. Tan and M. Ghanbari, "Layered image coding using the DCT pyramid", *IEEE Trans. on Image Processing*, Vol.4, No.4, pp. 512-516, April 1995.

[17] S. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation", *IEEE Trans. on Pattern and Machine Intelligence*, vol. II, No. 7, pp. 674-693, July 1989.

[18] S. Mallat, *A Wavelet Tour of Signal Processing*, 2-d ed., San Diego: Academic Press, 1998.

[19] M. Misiti, Y. Misiti, G. Oppenheim, and J.-M. Poggi, *Wavelet Toolbox: Computation, Visualization*, in *Programming User's Guide*, Ver. 1, MATLAB® v.5.2, ..\MatLab\Help\PDF_doc\wavelet\wavelet_ug.pdf.

[20] J. Benedetto and M Frazier, "Introduction", in *Wavelets: Mathematics and Applications*, J. Benedetto and M. Frazier, Eds. Boca Raton, FL: CRC Press, 1994, pp. 1-20.

[21] N. Otsu, "A threshold selection method from gray-level histograms", *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 9, No. 1, pp. 62-66, 1979.

[22] D. T. Dimov and G. Y. Gluhchev, "A locally adaptive binary-tree method for binarization of text images", in *Progress in Handwriting Recognition*, A. C. Downton and S. Impedovo, Eds. London: World Scientific, 1997, pp. 575-580.

[23] T. H. Reiss, *Recognizing Planar Objects Using Invariant Image Features*, in Lecture Notes in Comp. Sci., 676, Berlin: Springer, 1993.

[24] D. Dimov, "Fast Retrieval of Images by Graphics Content", 10th IC on Artificial Intelligence: Methodology, Systems, Applications (AIMSA' 2002), Varna, Bulgaria, Sept. 4-6, 2002 (submitted for presentation)

[25] D. Dimov, G. Gluhchev, N. Nikolov, I. Burov, E. Kalcheva, S. Bontchev, et al., "The computer system STEMB for verification of replicas of the Bulgarian state emblem", *Cybernetics and Information Technologies*, vol. 1, No. 1, Sofia: Bulgarian Academy of Sciences, pp. 95-107, 2001.