December 2006

A Novel Shape and Ontological Indexing for VIR Systems

Starostenko Oleg
Universidad de las Américas-Puebla

Alberto Chávez-Aragón
Universidad de las Américas-Puebla

Zehe Alfred
Universidad de las Américas-Puebla

Burlak Gennadiy
Center for Research on Engineering and Applied Sciences- UAEM

A Novel Shape and Ontological Indexing for VIR Systems

Starostenko Oleg
CENTIA, Universidad de las Américas-Puebla,
Pue., 72820, Mexico
oleg.starostenko@udlap.mx

Chávez-Aragón Alberto
CENTIA, Universidad de las Américas-Puebla,
Pue., 72820, Mexico
j_chavez_mx@yahoo.com.mx

Zehe Alfred
Department of Physics and Mathematics, BUAP,
72000, Puebla, Mexico
azehe@prodigy.net.mx

Burlak Gennadiy
Center for Research on Engineering and Applied
Sciences, UAEM, Cuernavaca, Mor. Mexico
gburlak@uaem.mx

ABSTRACT

This paper presents a novel hybrid method for visual information retrieval (VIR) that combines shape analysis of objects in image with their automatic indexing by textual descriptions. The principal goal of proposed method is the applying semantic Web approaches for visual information description in systems which use the low-level image characteristics. In the proposed method the user-oriented textual queries are converted to image characteristics which are used for visual information seeking and matching analysis. A decision about similarity between a retrieved image and user queries is taken by computing the shape convergence star field or two-segment turning functions combining them with matching of ontological annotations of objects in image providing in this way the machine-understandable semantics. For analysis of proposed method the image retrieval IRONS (Image Retrieval by Ontological Description of Shapes) system has been designed and evaluated in some specific image-restricted domains.

Keywords (Required)

Semantic web, ontology, image retrieval, shape analysis, object indexing.

INTRODUCTION

A typical approach to automatic indexing and classification of images is based on analysis of the low-level image characteristics, such as color, texture or shape (Gevers, 2000; Starostenko, 2001; Fensel, 2000) but this type of systems does not provide semantics associated with the content of each image. There are some well-known systems for visual information retrieval (VIR) which may be used as prototypes for a novel approach. One of them is Query by Image Content system (QBIC) provides retrieval of images, graphics and video data from online collections using image features such as color, texture, and shape for computing the similarity between images (QBIC™, 2006). AMORE (Advanced Multimedia Oriented Retrieval Engine) and SQUID systems provide image retrieval from the Web using queries formed by keywords specifying similar images, sketches, and SQL predicates (Amore, 2006). Although the contributions of these systems to field of VIR systems design were important, they do not provide mechanisms to represent the meaning of objects in images. In order to overcome this problem, we propose to apply the machine-understandable semantics for search, access, and retrieval of multimedia data using ontology (Fensel, 2004). The widely used Grubber's definition permits to describe semantics that establishes a common and shared understanding of a domain and facilitates the implementation of user-oriented vocabulary of terms and their relationship with objects in image (Gruber, 1993). The potential applications of the proposed image retrieval facilities include systems for supporting digital image processing services, high performance exchange of multimedia data in distributed collaborative and learning environments, digital libraries, etc.

PROPOSED SHAPE ANALYSIS APPROACHES

The proposed method may be described as a combination of specific descriptors based on shape preprocessing for extraction of sub-regions (objects) invariant to scale, rotation, and illumination, and application of ontology concepts for definition of machine-understandable semantics for retrieved images.

Shape indexing with two-segment turning function approach

Traditionally, a shape is described as a closed polygon which may be extracted by any well-known method for border estimation. Usually obtained shape may be represented by polygon with a great number of vertices; that require a lot of time for its processing (Chávez-Aragón, 2002; Lew, 2001). In order to reduce a quantity of polygon vertices to a subset of vertices containing relevant information about the original outline the discrete curve evolution process is proposed. It is achieved by assigning a relevance measure to each vertex, so that the least important vertex may be removed. Once a vertex is removed, its neighboring vertices must be connected. This process is repeated until we obtain the desired shape simplification. The relevance measure K is defined as follows.

$$\boldsymbol{K}(s_1, s_2) = \frac{\beta(s_1, s_2) l(s_1) l(s_2)}{l(s_1) + l(s_2)} \tag{1}$$

where $\beta(S_1,S_2)$ is the turning angle at the common vertex of the segments $S_1;S_2$, and $l(S_1)$, $l(S_2)$ are the length functions for segments normalized with respect to the total length of the polygonal curve C. The lower value of $K(S_1; S_2)$ corresponds to the less contribution to the curve C of arc $S_1 \cup S_2$. The algorithm for curve evolution is presented below.

Algorithm 1 Curve evolution algorithm

Input: a closed polygon P_m, where m is the number of segments, n is the number of segments of output polygon, $n < m$
Output: a closed polygon P_n

1. Find in P_m a pair of segments $S_i; S_{i+1}$ such that $K(S_i; S_{i+1})$ is minimum
2. Replace $S_i; S_{i+1}$ by the line S_0 that joins the endpoints of arc $S_1 \cup S_2$.
3. $m = m - 1$.
4. Repeat steps from 1 to 4 until m is equal to n.

Figure 1 shows the results of applying this technique that keeps the main visual parts of the original polygonal curve and obviously the amount of information has decreased drastically.

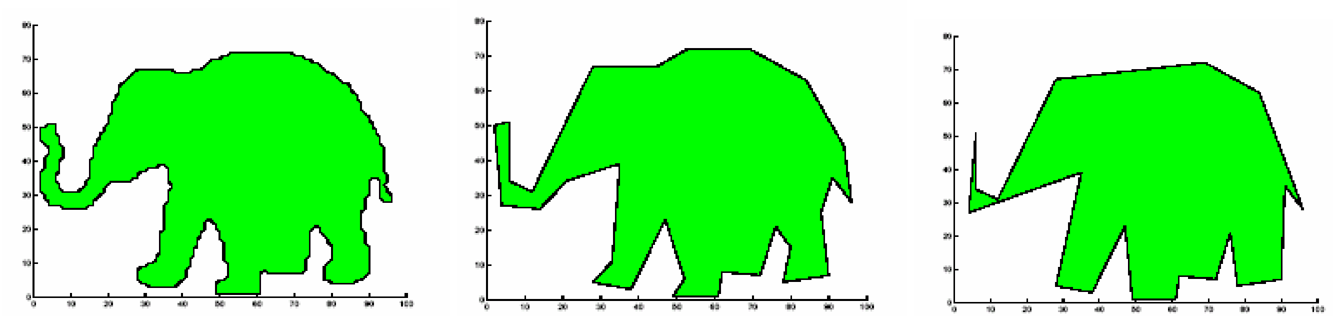

Figure 2. Evolution of the original shape (left) to the polygon of 30 segments (middle) and the polygon of 20 segments (right)

However, the polygonal representation of a shape is not a convenient form for calculation how similar is that shape to another. We propose to compute a matching using so called two-segment turning function or 2STF. Our approach for representing a curve is related to proposal of (Arkin, 1991), where a step function calculated from a silhouette of object is used. This function is called tangent function; however, it has some disadvantages regarding to invariant features. 2STF solves these problems by a simple strategy which operates with turn angle between two consecutive segments for computing steps in so-called step function (Chávez 2006). Using 2STF a polygonal curve P is represented by a step function, the steps on x - axis represents the normalized arc length of each segment in P, and the y - axis represents the turn angle between two consecutive segments in P. 2SFT has some advantages for shape matching, because this approach is invariant to translation, scaling, reflection, and rotation. 2STF is built taking into account the relative position between consecutive segments. That allows getting the same representation for a set of shapes even though they are placed in different positions or has been reflected or rotated. The turning angle obviously is not affected by size of a shape because the angle is the same. Regarding the length of segments, 2STF uses the normalized length with respect to the total perimeter of the shape that means the sum of the lengths of the whole curve is equal to 1. As a consequence, this approach is invariant to scaling too. In Figure 2 the computing of 2STF is presented for the same polygonal curve. The polygon on the right has been reflected and scaled by a factor of *0.8*. It is clear that both 2STFs have similar characteristics. The similarity between two shapes is computed by computing the differences between obtained 2STFs as it shown in the following algorithm:

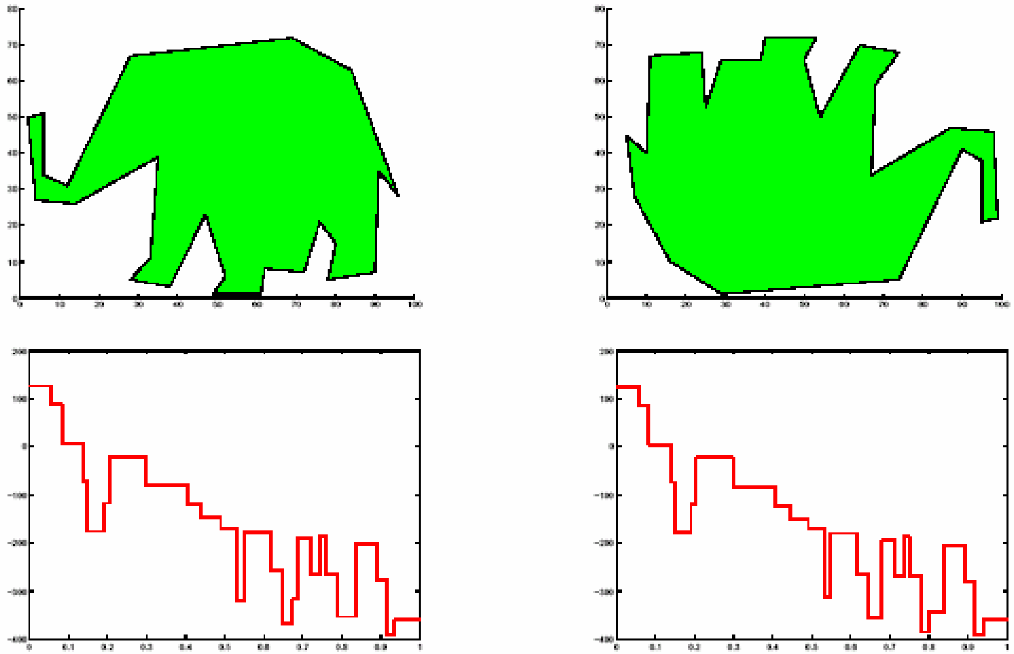

Figure 2. The polygon with rotation and scaling and their 2STFs

Algorithm 2 Matching strategy for comparison of 2STFs

Input: a polygon transposed into curve of 2STF, Output: compared value

1. Scale two curves to the same length, the scaling factor *sf* is $sf = \frac{l(a_1)}{l(a_2)}$ where $l(a)$ is the length of curve a and $l(a_1) > l(a_2)$
2. Curve a_2 is shifted, equalizing the weighted average of the angle values of a_1 and a_2
3. Compute the area between the two 2*STF* curves called difference D
4. The difference value D is obtained by multiplying the scaling factor *sf* by the length of the mayor curve

Figure 3 shows the proposed matching strategy. The shaded area represents how similar two shapes are.

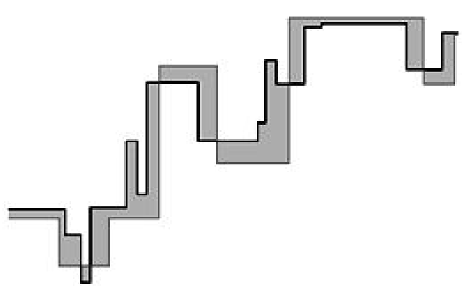

Figure 3. Matching strategy for computing similarity between two polygons.

The disadvantage of 2STF representation is significant time that it takes to find the best correspondence between two curves which may be reduced by decomposition of a curve P into groups G in order to obtain the same number of arcs of both curves. This process consists in grouping consecutive largest arcs to form groups of segments covering the whole curve P. The idea of joining together largest arcs has the following reasons: 1) curves can have different number of arcs; 2) small arcs can be joined together so they can be compared with a bigger arc of another curve. Formally, a polygonal curve P is made up of a set of segments S_i, so we can denote a curve P as, $P = (S_1; S_2:::S_{ns})$. Therefore, a curve representing the shape consists of *ns* line segments, where *ns* is the number of segments in *P*. The decomposition of the curve into the largest arcs can be stated this way.

We denote the largest arc as Ma_i which can be either convex or concave arc. It is a set of segments whose turning angles have the same sense except the first one. The largest arcs are defined as follows. $Ma_i = (Sstart; :::; Send); start >= 1$ and $end <= ns$; as a result, two consecutive Ma_i and Ma_{i+1} intersect in one segment, so $Send(Ma_i) = Sstart(Ma_{i+1})$. The number of group operations for each curve is $NG = (2^{Ma})^2$, where *Ma* - number of the largest arcs of the curve, but there are restrictions for grouping either concave or convex arcs. That reduces the number of combination to valid ones. Table 1 shows the time that takes the calculation of valid combinations using a personal computer with processor of 2GHz and RAM of 1GB.

Maximal arcs	Combinations	Valid Combinations	Time (seconds)
2	16	2	0.031
4	256	18	0.047
6	4096	166	0.078
8	65536	1634	0.375
9	262144	5198	4.375
10	1048576	16646	53.781
11	4194304	53594	581.75
12	16777216	173318	6474.3

Table 1. Time for computing the valid combination of either concave or convex arcs

Experimentally, we determined that the best correspondence between more than ten-largest-arc polygon takes a lot of time (more than 10 seconds). Therefore, it is possible to conclude that advantage of this approach is independence from scale, reflection, translation, and rotation, but it requires a significant time for computing of similarity between shapes. This problem may be solved by another technique called Star Field representation of shapes.

Shapes matching with Star Field

Star Field (SF) is an alternative representation for shapes that allows obtaining more precise comparison because it is not necessary to apply a great grade of evolution of polygonal curves. It means that we able to compare polygons with more than ten arcs (max value acceptable for 2STF). As a result, SF gives an easier and faster matching process. Our Star Field method combines approach for computing the similarity among shapes proposed by (Mokhtarian 1992) and proposed earlier 2STF. Mokhtarian proposed using the maxima of curvature zero-crossing contours of Curvature Scale Space (CSS) as a feature vector to represent shapes. However, computing CSS is a expensive process and we propose to use 2STF which is easier, faster and more effective (Chávez 2006).

Formally, a SF representation is a set of marks or stars $M_1;M_2:::M_{nm}$, where *nm* is the number of vertices of the polygonal curve that it represents and this number is equivalent to the number of steps in its 2STF. *Mn* is defined by means of two coordinates $(x; y)$. The *x* - coordinates indicate the normalized distance from the starting point to the corresponding vertex, making sure that in the middle of the SF plane is the star that corresponds to the most important vertex of the polygon. The *y*-coordinate is the normalized angle between two consecutive segments that share the corresponding point. In other words, *y*-coordinate of stars correspond to the height of each step in its equivalent 2STF in the range [0,1] where a value of zero represents a $-\pi$ angle and one corresponds to $+\pi$. The principal difference between 2STFs and SF is the grade of evolution of the digital curves they work with. In contrast with the use of a parameter that indicates the number of final vertices of the simplified curve in 2STF, the SF is able to work with a larger number of maximal arcs, consequently with a larger number of vertices without increasing the time for their processing. The curve is simplified until the significance of each vertex is over a parameter φ. Thus, the digital curve must be simplified until the least important vertices have been disappeared using threshold φ as it shown in algorithm 3.

Algorithm 3 Curve evolution algorithm with threshold φ

Input: a closed polygon P_m, *m* is the number of segments; a parameter *n*, where *n* is the least number of segments of the output polygon that is acceptable; and a threshold φ; Output: a closed polygon P_n

1. Find in P_m a pair of segments $S_i; S_{i+1}$ such that $K(S_i; S_{i+1})$ is minimum.
2. Replace $S_i; S_{i+1}$ by the line S_0 that joins the endpoints of arc $S_1 \cup S_2$.
3. $m = m - 1$.
4. Repeat steps from 1 to 4 until $m > n$ and $abs(K(S_i; S_{i+1})) < \varphi$

In the algorithm 3 the threshold φ is used in order to stop the simplification process. Threshold φ is in the interval $[0;¼\pi]$; however, experimentally we have obtained good results using the threshold φ in the interval $[0.4;0.8]$. The SF representation of a curve is made up of a set of points placed on a 2*D* plane. Each star or point, in the SF represents the vertex that is shared by two consecutive steps from the equivalent 2STF. A particular SF as well as its equivalent 2STF illustration depends on a first point to be drawn. The initial point of the SF diagram is the one in the extreme left. The way we determine this point is by means of rotating the stars until the star that represents the most important vertex is in the center of the diagram. A SF diagram as 2*D* plane is divided horizontally into two sections. The upper section holds the stars that represent vertices of concave arcs. On the other hand, lower part holds vertices of convex arcs. So, a curve can be converted into a star field representation, and this novel representation can be seen as a cloud of points. In the Figure 4 the original apple-like image a) and 15-segments polygon b) obtained from the original image using curve evolution algorithm are presented. The image c) shows the two-segment turning function of polygon and d) shows the star field representation of the same polygon.

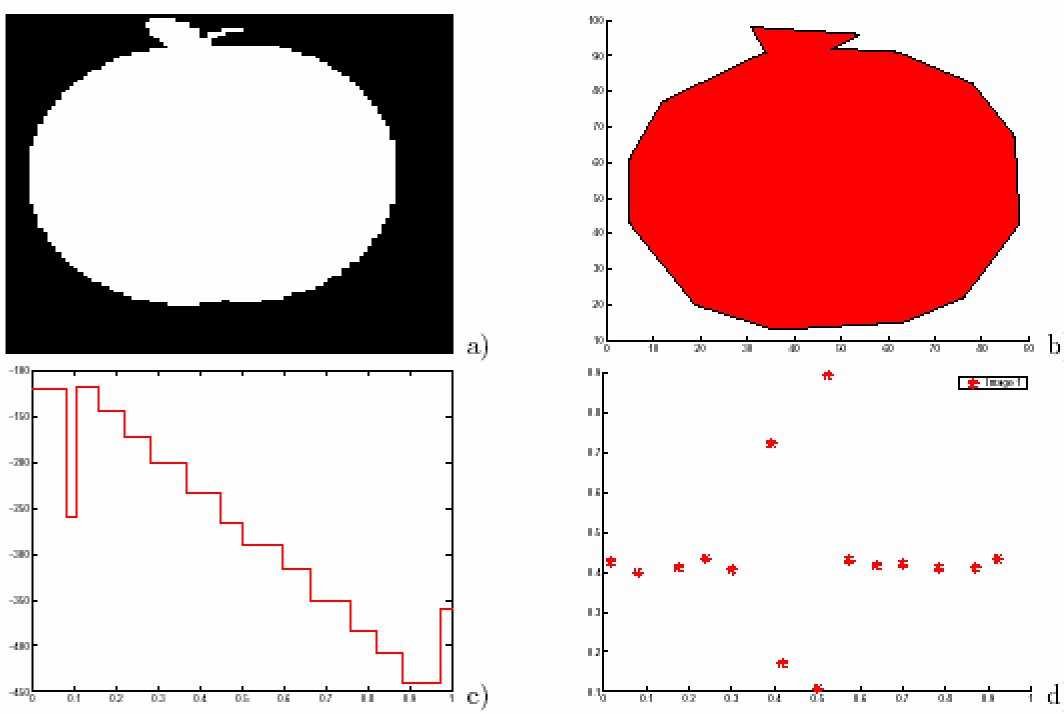

Figure 4. Original image, its 15-segments polygon, and its 2STF and SF representations

So far, a new convenient way for representing a polygonal curve has been presented. For SF a new similarity measure using graph and adjacency matrix will be introduced. Given two polygonal curves P_1 and P_2 and their star field representations SF_1 and SF_2, the graph G that allows us to compute their similarity is defined as $G = (V;E)$ where V and E are disjoint finite sets. We call V the vertex set and E the edge set of G. Our particular graph G has a set V that consists of two smaller subset of vertices υ_1 and υ_2. Set $V = \upsilon_1 \cup \upsilon_2$., where υ_1 is the set of point of SF_1 and υ_2 is the set of points of SF_2. On the other hand, E is the set of pairs $(r; s)$, where $r \in \upsilon_1$ and $s \in \upsilon_2$. Then we propose to use the adjacency matrix for representing the graph, where each cell of that matrix contains the cost for traveling from one column to each row and vice versa. The main idea behind the construction of the matching graph consists in building a connected weighted graph so that an algorithm to find the minimum spanning tree is applied. The minimum spanning tree is a subset of edges that forms a tree that includes every vertex, where the total weight of all the edges in the tree is minimized. Thus, for the more similar shapes we obtain the lower value of corresponding total weight. But, in order to get the desired result the matching must be constructed in a very particular way as it shown in following algorithm.

Algorithm 4 Matching graph construction

Input: two set of points SF_1 and SF_2 that define two star field representations and an increment Δ
Output: a connected weighted graph

1. Rotate SF_1 and SF_2 so that, the most import star of each *SF* coincide in the center of the window
2. For each point *sp* from the SF_1 do:
3. Look for those points which belong to SF_2, which stay at most a distance *d* in all directions from *sp*, and which have not been connected previously
4. Connect *sp* with each point found in previous step and assign a weigh equal to the Euclidian distance of two vertices of each edge
5. If there was not any connection, increase *d* in a value Δ and go to step 3
6. Select one point of SF_1 and connect the rest of the points from SF_1 with it; finally assign to each edge generated in this step a weigh equal to zero.

The Figure 5 shows a 3D representation of the graph which is the result of applying the Matching graph construction algorithm. All the star-like marks are connected with a weight zero; on the other hand, the star-like marks and the cross-like marks are connected with a weight equal to the Euclidian distance between them.

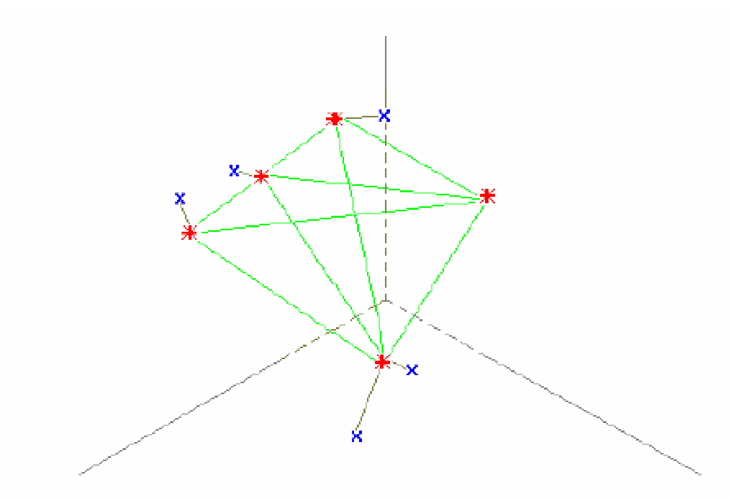

Figure 5. Te result of Matching graph construction algorithm

Given two identical shapes with the same number of steps, the total weight of the spanning tree is equal to zero. This is, because each star is connected with the corresponding one and since they have the same value of *x*- coordinate and *y* –coordinate, the Euclidian distance is equal to zero. Additionally, we have mentioned that all the stars from the first shape are connected with a weight equal to zero. As a result, the values of the path through the spanning tree are equal zero that means they are identical. That may be obtained by applying the Prim's algorithm for finding the minimum spanning tree

Algorithm 5 Prim's algorithm

Input: a connected weighted graph, Output: a minimum spanning tree

1. Create a tree containing a single vertex, chosen arbitrarily from the graph
2. Create a set containing all the edges in the graph
3. Loop until every edge in the set connects two vertices in the tree
a) remove from the set an edge with minimum weight that connects a vertex in the tree with a vertex not in the tree
b) add that edge to the tree

Finally, we can define how to calculate the similarity among shapes. The most important part of this calculation is the value of the cumulative weight of the edges that make up the spanning tree. However, the similarity value is also affected by a penalty quantity. It is possible that some stars of the first shape never connected with a star from the second one, a penalty value is added to the final similarity measure. The additional cost is computed as it shown in Figure 6. There dotted lines show the distances that are added to the cumulative length obtained from the minimum spanning tree, this is because two stars have not been connected with the corresponding ones.

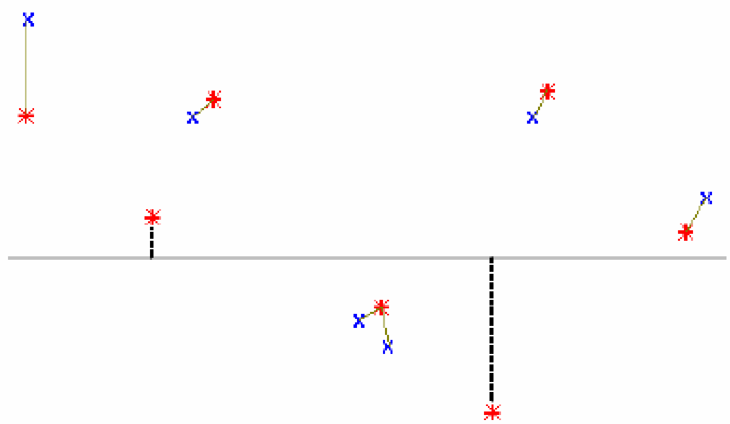

Figure 6. The additional cost defines by distances that are added to the cumulative length

This new SF approach based on 2STF maintains its advantages and due to its simplicity allows to work with complex polygons. SF permits to define a similarity measure based on calculation of a minimum spanning tree from a connected weighted graph which is more fast and accurate.

Query preprocessing module

The IRONS system consists of four principal modules: query preprocessing, indexing module, feature vector comparison and feedback GUI. The query preprocessing module provides the selection of sub-region containing the relevant objects. Once the sub-region is extracted, the object within that sub-region is found by the CORPAI algorithm. The Smallest Univalue Segment Assimilating Nucleus (SUSAN) method (Smith, 1997) for corner detection and Convex Regions Preprocessing Algorithm in Images (CORPAI) (Medina, 2004) have been used for extension of types of queries. The result of applying SUSAN and CORPAI algorithms is a convex polygon that may be simplified by discrete curve evolution process described early. The algorithm for query preprocessing is show as it follows.

IRONS IMAGE RETRIEVAL SYSTEM

After presentation of two approaches for shape analysis and matching the Image Retrieval by Ontological Description of Shapes (IRONS) system has been implemented. Its block diagram is shown in Figure 7. The input for the system may be an image, its shape, or a keyword, which describes the object with a certain degree of similarity. The retrieved images will be ones with more similarity to the low-level features of a query and will have a high degree of matching with the ontological annotations defining the content of the image. Once the user draws a query, the system uses the SF shape indexing algorithms in order to generate the feature vector for comparison with the other ones in the image database (Starostenko, 2005). Then the content-based recognition process is applied to shapes in order to find similar ones in the ontology namespace.

Algorithm 6 Query preprocessing algorithm

Input: A color image with luminance of pixels *Ic*; Output: the feature vector described a shape

1. *Ig* ⇐ ComputeLuminance (using *Ic*) // it converts color into gray level image
2. Principal corners ⇐ SUSAN operator (*Ig*) // detection of object's corners
3. *Scs* ⇐ SpatialSampling (*Ic*) // reduction of image size to an 8x8 pixels window
4. ColorDescriptor ⇐ ComputeColorDescriptor I1I2I3 *(Scs*) // descriptors based on I1I2I3 color system model
5. FeaturesVector⇐ ComputeDescriptor (Principal Corners, ColorDescriptor) // the sub-region descriptor includes a color vector and the principal corner's position.
6. Subregion ⇐ CORPAI(*Ic*,*Sp*) // applying the CORPAI algorithm over regions
ConvexHulls (points*[ ]*) // compute the convex hull
{ if (query_sub-region(image *[][]*)) // apply boundary detection operator to sub-region (operator(image *[][]*))}
7. Ic_{NEW} ⇐ TransformationFromSubregionToImage (Subregion) // transformation of the irregular convex sub-region of the

original image to a new normalized one
8. FeaturesVector ⇐ ComputeDescriptor (Principal Corners, ColorDescriptor, ConvexRegions) // the convex region descriptor is obtained.
9. FeaturesVector ⇐ DiscreteCurveEvolution (Simplified Polygon) // removal of the least important polygon vertexes.
If the query is a keyword, the preprocessing step is not applied.

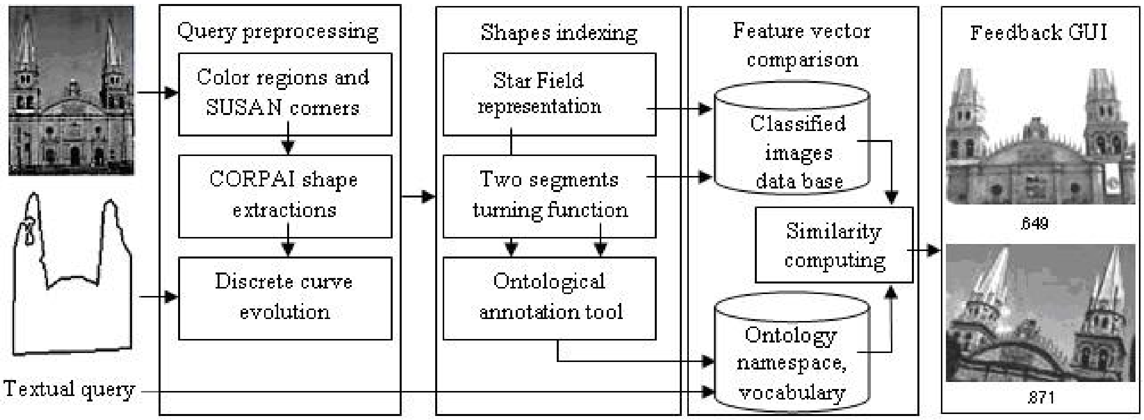

Figure 7. Block diagram of the proposed IRONS system

Indexing module

The indexing module generates a feature vector describing shape of objects in image and content-based annotations. The preprocessed polygon is represented by SF using 2STF invariant to scaling, rotation, reflection, and translation.

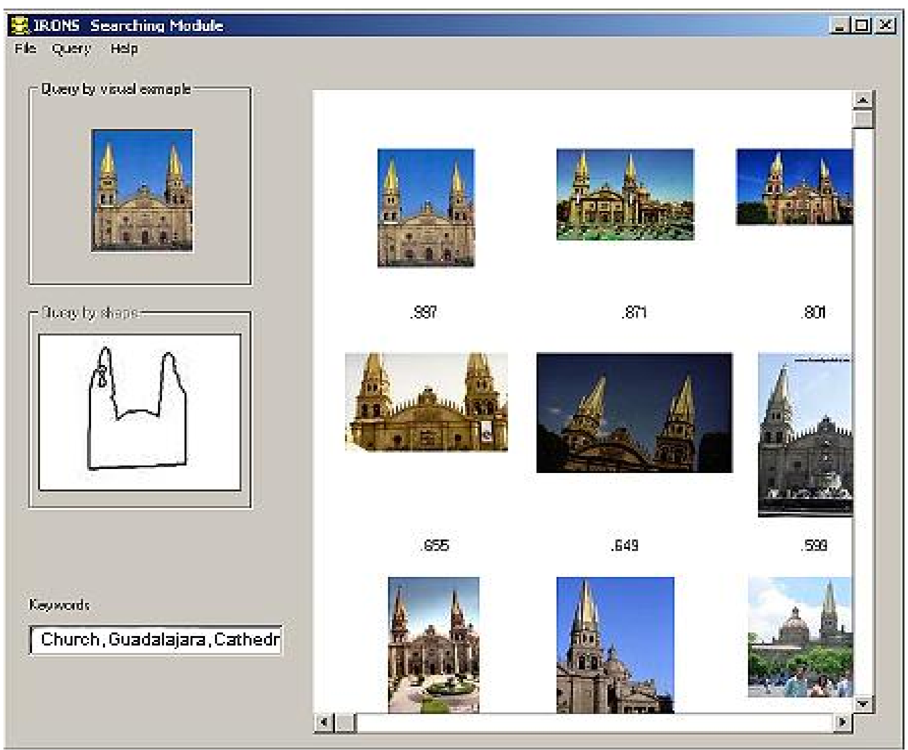

Figure 8. Image retrieval GUI of the IRONS system

The ontological annotation tool is used for searching matches in the ontology name space. The images with higher matching are retrieved and visualized on GUI with a certain degree of similarity. In this work we use hybrid feature vector which defines such low-level image characteristics (shape) and semantic descriptions. This permits to speed up the matching process as well as reduce the number of iterations with nonsense results. Second vector is formed by ontological description tool which establishes the relationship between the object and its formal explicit definition. In such a way, the meaning of an image may be obtained in textual form as a set of descriptions for each sub-region related to a particular ontology. The Resource Description Framework (RDF) language to support the ontology management has been used in this approach that defines a syntactic convention and a simple data model to implement machine-readable semantics (Fensel, 2000). Using RDF it is possible to describe each web resource with relations to its object-attributes-value based on metadata standard developed by the World Wide Web Consortium (Backett, 2001). The ontology is described by a directed acyclic graph; each node has a feature vector that represents the concept associated with that node. Concept inclusion is represented by the IS-A inter-relationship. For example, particular elements of buildings, churches, etc. correspond to specific concepts of shapes defining these buildings, churches. If the query describes an object using this ontology, the system would recover shapes that contain windows, columns, façades, etc. even though, those images have not been labeled as geometric figures for the retrieved object. The feature vectors of each node in the ontology name space consist of keywords linking the previously classified images to the characteristics of the new shape extracted by the SF or 2STF. The indexing and the ontology annotation processes may be described now as:

1. *FeaturesVector* $\Leftarrow$ *Shape*$_i$ *(Pentagon, Pi,Ci) // Pi* is its SF or 2STF representation and *Ci* is the compactness of the shape computed as a ratio: square root (*RegionBorderLength* divided by *ShapeArea*)..
2. *SaveRelationInOntology(Ic, FeaturesVector of Ic*$_{NEW}$*, Td) //* update the ontology namespace

As it has been mentioned, two kinds of vector comparison are used: matching the shapes and definition of similarity in ontological annotations. The computing of similarity is additionally provided by computing the Euclidean distance d_E to compare feature vectors according to the equation:

$$d_E(\mu,\sigma)=\sqrt{\sum(\mu-\sigma)^2} \quad (2)$$

where μ and σ denote two feature vectors. The query interface of the IRONS system is shown in Figure 8 where the images with high degree of matching are shown in downward order. The user may submit a visual example, a sketch, a keyword or a combination of the above.

Experiments and discussion

We tested the proposed method using the image collection CE-Shape-1.

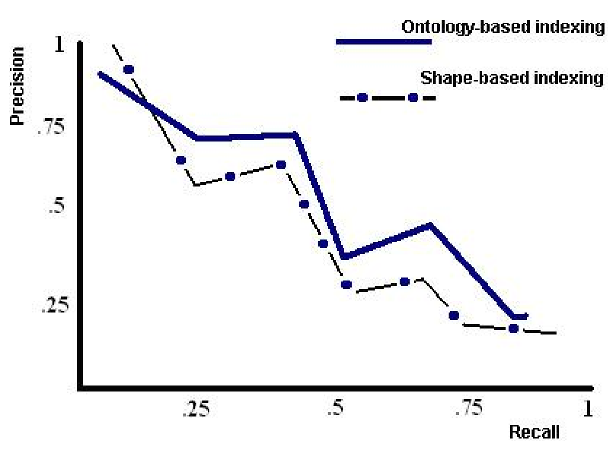

Figure 9. Precision-recall curve for IRONS

This database has about 1400 images divided into 60 different categories. Each category has about 20 images. We performed the experiments to verify the role of shape-based and ontology-based indexing in the retrieval process. The performance of the method was evaluated using the precision and recall metrics. For example, the IRONS responds with 6 from 10 relevant images in database. In this case the precision is 60%. The precision of ontology-based indexing was higher as it shown in Figure 9 where comparison with only shape indexing approach has been used.

CONCLUSION

The evaluation of the proposed method and the testing of the implemented system have been done by means of two metrics, precision and recall. In order to use these metrics it necessary to build a reference database which has a set of preprocessed image descriptions. During the comparison of the characteristic vectors of input and preprocessed images the reference database for particular restricted domain is used. The system performance is better when the image is processed in sub-regions; excessive subdivision does not produce good results. Satisfactory retrieval of expected images is achieved faster due to the lower number of iterations in the search process with ontology. The analysis of the indexing approaches shows that SF is in order as fast as 2STF. This occurs because the typical data structures used in indexing tools are hashing tables, which are manipulated with specific keys or signatures representing a shape. The disadvantages of the system are errors in spatial sampling during generation of the image feature vector as well as the required amount of system memory. Factors like tolerance to occlusion and deformation, robustness against noise, and feasibility of indexing are also considered in our approach.

The most important contribution of this research is the proposed hybrid method combining the advantages of low-level image characteristics extraction with textual description of image semantics. The use of ontological annotations allows simple and fast estimation of the meaning of a sub-region and of the whole image. The proposed image retrieval method is robust to partial occlusion and to small changes in the position of the objects. From the obtained experimental results, we can conclude that the method could be considered as an alternative way for the development of visual information retrieval facilities.

REFERENCES

1. Amore (2006) Advance multimedia oriented retrieval engine, http://www.ariadne.ac.uk/issue9/web-focus/
2. Arkin, M., Chew, L.P. (1991) An efficiently computable metric for comparing polygonal shapes. *IEEE trans*, 13, 209-206.
3. Beckett, D. (2001) The design and implementation of the Redland RDF application framework, *Proceedings of the 10-th International WWW Conference*, 120-125.
4. Chávez, J., Starostenko, O. (2001) Motion estimation algorithms of image processing services for wide community, Baba N, Jain L.C. (Eds.) *Proceedings of Knowledge/Based Intelligent Information Engineering Systems Conference*, September, Japan IOS Press, 758-763.
5. Chávez-Aragón A. (2006) A Semantic Image Information Retrieval Model, Ph.D Technical report, UDLA-P
6. Fensel, D. (2000) The semantic web and its languages, *IEEE Computer Society* 15, 6, 67-73.
7. Fensel, D. (2004) Ontologies: a silver bullet for knowledge management and electronic commerce, 2-nd ed. Springer, USA.
8. Gevers, T., Smeulders, A.W. (2000) Combining color and shape invariant features for image retrieval, *IEEE Trans. on Image Processing*, 9, 1, 102-119.
9. Gruber, T.R. (1993) A translation approach to portable ontology specifications, Knowledge Acquisition, 199-220.
10. Lew, M.S. (2001) Principles of visual information retrieval, Advances in pattern recognition, Springer-Verlag, USA.
11. Medina, M. A., Chávez-Aragón, J.A., Starostenko, O. (2004) A Novel hybrid method for image retrieval by ontological descriptions of sub-regions, *Journal WSEAS Transactions on Systems*, 4, 3, 1301-1306.
12. Mokhtarian, F. (1992) A theory of multiscale, curvature-based shape representation for planar curves, *IEEE Trans. On Pattern Anal. Mach Intell.*, 14, 8, 789-805.
13. QBIC ™, (2006) IBM's Query by image content, http://wwwqbic.almaden.ibm.com/.
14. Smith, S.M., Brady, J.M. (1997) A new approach to low-level image processing, *Journal of Computer Vision*, 23, 1, 45-78.
15. Starostenko, O., Neme J. (2001) Novel advanced complex pattern recognition and motion characteristics estimation algorithms, Nabeel Murshed (Ed.) *Proceedings of VI Iber - American Symposium on Pattern recognition*, October 12-14, Brazil, IAPR Press, 7-13.
16. Starostenko, O., Chávez-Aragón A. (2005) A Hybrid Approach for Image Retrieval with Ontological Content-Based Indexing, Lecture Notes, Progress in Pattern recognition, Springer-Verlag, Germany, 3773, 997-1004.