DOI: 10.15439/2018F279 ISSN 2300-5963 ACSIS, Vol. 15

Query by Approximate Shapes Image Retrieval with improved object sketch extraction algorithm

Stanisław Deniziak
Kielce University of Technology
al. Tysiaclecia Panstwa Polskiego 7, 25-314 Kielce, Poland
Email: s.deniziak@tu.kielce.pl

Tomasz Michno
Kielce University of Technology
al. Tysiaclecia Panstwa Polskiego 7, 25-314 Kielce, Poland
Email: t.michno@tu.kielce.pl

***Abstract*—In this paper a new Content Based Image Retrieval based on a sketch method was proposed. The main idea of the algorithm is based on decomposing an object into predefined set of shapes (primitives): line segments, polylines, polygons, arches, polyarches and arc-sided polygons. All primitives are stored as a graph in order to store the mutual relations between them. Graphs are stored in a tree-based structure which allows fast querying. As an improvement to the algorithm, a conversion to the HSL color space was proposed in order to detect primitives more accurately. Moreover, computing all line slopes in relation to the object oriented bounding box was also proposed. Additionally, in order to better detect objects present in images, the usage of Edge Boxes algorithm was proposed.**

I. Introduction

MULTIMEDIA databases are becoming more and more popular. Most often they store huge number of images which causes the need of effective storage, processing and query methods. This is becoming an important problem because they are used more often in everyday life, from searching for information in the internet to authorising, recognizing and monitoring systems. Moreover, most of social media portals stores images as the part of provided content or even are oriented only on providing images, thus they also need effective methods to manage and provide data to users.

There are many different methods in the area of image retrieval from multimedia databases. Most of the algorithms may be grouped into three categories: the Keywords Based Image Retrieval (KBIR) algorithms, the Content Based Image Retrieval (CBIR) algorithms and the Semantic Based Image Retrieval (SBIR) algorithms.

The KBIR algorithms use textual annotations in order to describe objects present in the image. Then, during the query they are compared with keywords typed by the user [1]. The CBIR algorithms use content present in the image in order to perform queries. Most often global statistical features are used (e.g. contrast, entropy [2] or a normalized histogram of colors [3]) or grouping similar pixels into regions in order to construct graphs which are then compared using e.g. Maximum Likelihood [4]. There are also methods which use sketches instead of images as a query(e.g. [5]). The SBIR algorithms tries to minimize the difference between the data present in the image and the information which can be deduced by a human [6], [7]. The text may be classified e.g. by one of the methods described int [8]. There are methods which use textual queries and graphical queries.

This paper presents a new Content Based Image Retrieval method from multimedia database which is based on querying by a sketch. The main idea of the algorithm is based on representation of objects using a set of predefined shapes, called primitives: line segments, arches, polylines, polygons, polyarches (a chain of connected arches) and arc sided polygons (a looped chain of connected arches). For each type of a primitive, there are defined attributes which describes them, e.g. a line slope for a line segment or an angle for an arc. The primitives may be extracted from images using proposed approach based on HSL color space segmentation. All primitives are connected into a graph which stores all relations between them. Graphs are stored in a tree-based database structure which gathers similar objects graphs in the same subtrees, but without fast tree height grow. In comparison to our previous works, in this paper we focused on improving the primitives extraction algorithm and the precision of the results. The proposed approach provides easy graphical queries for users, both using sketches drawn by themselves (without need of high drawing skills) and example images. All similar graphs are stored in congruent nodes, thus the querying is faster than in e.g. linear data structure. Also the structure allows performing queries in parallel which also improves the computation time.

The paper is organised as follows: the first section is an introduction. Next section presents the proposed object representation. The third section describes the shapes extraction algorithm with improvements and the fourth possible usages of Edge Boxes algorithm. The fifth section presents the experimental results. The next section is the summary and future research section. The last one section is a list of references used in this paper.

II. The object representation

During previous stages of our research, we found that most objects and images can be described using sketches. This method is very efficient, because there is no need of example image, but if existent, it can be processed in order to extract a sketch. Moreover, sketch representation of objects may be used when there is no full knowledge about searched matter, e.g. when only a front part of a car is known.

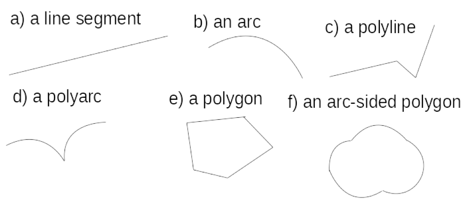

Fig. 1. The proposed primitives: a) a line segment, b) an arc, c) a polyline, d) a polyarc, e) a polygon, f) an arc-sided polygon

The sketch may be represented by an image (e.g. using black pixels as sketch edges [9]), an object outline (e.g. as in [5]) or, as proposed in our research, as a set of predefined shapes. The advantage of our approach is the simplicity to draw a sketch by users, ability to store additional information for each shape (e.g. information about material or color). The predefined, simple shapes are called primitives. During our research we noticed, that line segments (Fig. 1 a)) and arches (Fig. 1 b)) are suitable for describing a sketch of most objects. Moreover, during experiments, we realised that storing the information about connected segments and arches improves the efficiency of the algorithm. Thus, additional primitives are proposed:

- based on line segments: polylines (Fig. 1 c)) and polygons (Fig. 1 e))
- based on arches: polyarches (a chain of connected arches, Fig. 1 d)) and arc-sided polygons (a looped chain of connected arches, Fig. 1 f))

Since all primitives are based on line segments or arches, they are called base primitives, afterwards all primitives created using them are called complex primitives.

Each primitive is defined by its type and an attribute or set of attributes, which are defined as follows: a line segment attribute is its line slope, an arc attribute is its angle, a polyline and a polygon attributes are number of segments and their line slopes, a polyarc and an arc-sided polygon attributes are number of segments and their angles.

During our research we realised that not only the information about primitives is needed but also an information which shapes are connected. Thus, such an information is stored using a graph where nodes are used to store primitives and edges used to store connections between them. This approach is similar to graphs of regions in region-based CBIR algorithms. Moreover storing the information about mutual positions between connected primitives also improves the efficiency of the image retrieval from the multimedia database [10]. Therefore, for each connection, there is stored the information about positions using the geographical windrose (N, S, E and W directions). The example of the graph is shown in the Fig. 2.

Since an image may contain many objects, in order to clearly emphasize that they are separated in a graph, a structure called complex shape was introduced [10]. It may be used optionally or mandatory based on types of stored images in the multimedia database.

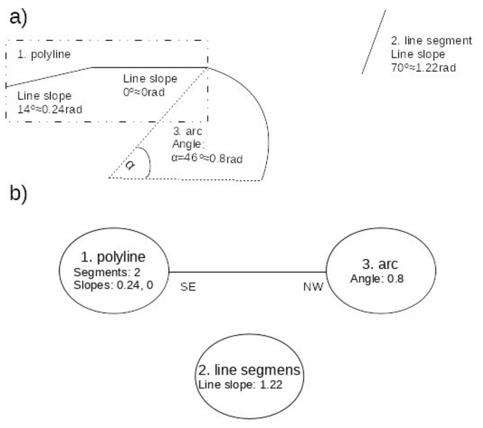

Fig. 2. The example of the graph which contains 3 primitives: a) primitives and their attributes (a polyline, an arc and a line segment), b) the graph: a polyline (1) and an arc (3) are connected, no. 3 lies on the right bottom (SE) of no. 1 and consequently no. 1 lies on the top left (NW) of no. 3; the node no. 2 does not have any connection

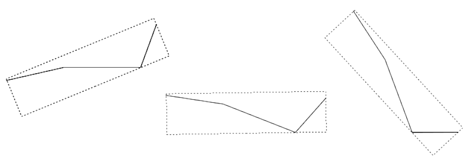

Fig. 3. The example polyline object with its Object Oriented Bounding Box (marked by dotted lines) in three different orientations.

In order to improve the algorithm resistance to different objects orientations, as an addition to the previous works, we propose to store primitives attributes in axis independent manner. During the stage of attributes values computing, line slopes may be computed in relation to the longest side of the Object Oriented Bounding Box built on top of the object. The example bounding box of a polyline is shown in the Fig. 3. Such an approach allows comparisons of differently oriented primitives which are strictly the same or very similar with high precision. Without such solution, there may be some problems with proper image retrieval from the database when users draw a sketch which is e.g. rotated in comparison with sketches stored in the database.

During previous experiments we found that storing the information about mutual positions of connected nodes highly improves the precision of the image retrieval algorithm results. Due to that fact, we propose to store the same information also for each pair of nodes (not only connected), similarly to coincidence matrix used in graphs. In this paper we test if such an approach improves the precision of the results.

III. The image shapes extraction algorithm based on HSL image representation

In this paper we propose to enhance the shapes extraction algorithm in comparison with the previous version used in [1]. The main idea of the improvement is based on converting an image into a HSL color space. Due to the conversion three images are created: first with only hues present in the image,

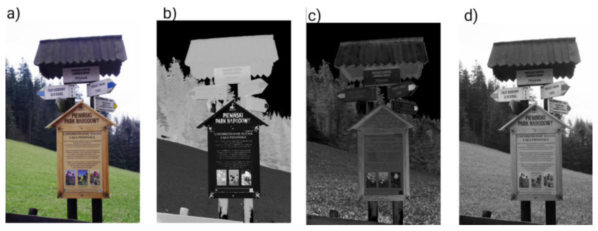

Fig. 4. The representation of image using HSL color space: a) initial image, b) hue channel, c) saturation channel, d) lightness channel which is most often the same as grayscale image representation.

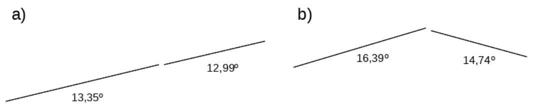

Fig. 5. The example of line segments merging with close endings: a) two segments with line slopes close to 13° which can be merged, b) two segments with completely different line slopes.

the second one with lightness information and the last one which stores saturations of different areas (the example image representation is shown in the Fig. 4). Thanks to that, the algorithm is able to detect line segments and arches with the same hue, the same lightness level and the same saturation.

Similarly to other objects detection algorithms, morphological operations should be performed on the image in order to remove unnecessary details or noise.

The first algorithm step is the detection of base primitives - line segments and arches. Firstly all line segments are detected in hue channel using Line Segment Detector algorithm. Next, for each channel thresholding with different values is performed in order to create uniform areas of similar pixels and reduce the slight differences on the same shapes. After that, line segment detection is performed again for each channel. Thanks to that more line segments are detected. Next all detected line segments are merged if possible, checking if they have the same line slope and connection of very near endings (Fig. 5). As a next step, all segments shorter than defined threshold are removed from the list. The algorithm is shown in the Alg. 1.

When all line segments are found, then the list is searched in order to find the candidates of arches. In order to find line segments which may construct an arc, we check if in the list there is a chain of connected line segments with length equal or greater than 3. Then, if a chain is found, angles between each segments are checked if they are equal or very similar. Moreover a test if their values are in the range $(0°, 180°)$ (Fig. 6) is performed. Additionally, lengths of segments are checked in order to detect if they are very similar. The arc detection algorithm is shown in the Alg. 2. In the practical implementation, the algorithm is assisted with Circular Hough Transform in order to improve the detection of circles.

The last shapes extraction algorithm step is the creation of more complex primitives defined as follows: polylines, polygons, polyarches and arc-sided polygons. Firstly all detected

Algorithm 1 Line segment detection algorithm

```
Ensure: $img$ - the image which has to be processed, $lsList$
    - list which stores line segments
    $img_H \leftarrow$ hue channel of $img$;
 2: detect all line segments in $img_h$ and add them to the
    $lsList$
    for each $img_x \in$ H,L,S channel of $img$ do
 4:     process $img_x$ as follows:
        for each $pixel$ of $img_x$ do
 6:         round $pixel\ value$ to the nearest $threshold$ value
        end for
 8:     detect all line segments in $img_x$ and add them to the
        $lsList$
    end for
10: for each $ls_1 \in lsList$ do
        for each $ls_2 \in \{\{lsList\} \setminus \{ls_1\}\}$ do
12:         if $ls_1$ and $ls_2$ endings are close enough then
                $angle_{LS1} \leftarrow ls_1\ line\ slope$
14:             $angle_{LS2} \leftarrow ls_2\ line\ slope$
                if $angle_{LS1}$ and $angle_{LS2}$ are similar enough
                then
16:                 merge $ls_1$ and $ls_2$
                end if
18:         end if
        end for
20: end for
    for each $ls \in lsList$ do
22:     $len \leftarrow length(ls)$
    end for
```

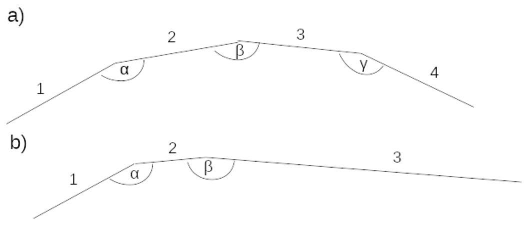

Fig. 6. Different chains of line segments a) a chain from which an arc may be created: segments lengths are very similar (1=146, 2=145, 3=146,8, 4=145,8) and angles α, β, γ are equal to 160° b) the chain from which an arc cannot be created: segments lengths are different (1=138,7, 2=85,4, 3=371) and angles are not equal or very close ($\alpha = 157°$, $\beta = 170°$)

line segments are processed and checked for chains. After chains detection, they are tested in order to detect looped chains from which polygons are created. Then, all others are transformed into polylines. Similarly, founded arches list is processed: firstly chains of arches are detected and then looped ones are transformed into arc-sided polygons and all others into polyarches.

IV. Shapes extraction improvement using Edge Boxes

The shapes extraction algorithm performance could be improved by a usage of object proposal algorithms. One of the

Algorithm 2 Arches detection algorithm

```
Ensure: $lsList$ - list which stores line segments; $chains$ - list of chains of line segments; $archesList$ - list which stores arches
find all connected line segments from $lsList$ longer than 3 segments and add them to $chains$
compare lengths of all segments in each $chain \in chains$
if lengths are similar enough then
    for each $ls \in chain$ do
        compute an angle between $ls$ and next segment from the same chain
    end for
    if angles are similar enough then
        create new arc based on chain
        compute arc angle and center point
        add arc to $archesList$ and remove all $chain$'s line segments from $lsList$
    end if
end if
```

TABLE I
THE COMPARISON OF PRECISION AND RECALL RESULTS FOR BICYCLE IMAGES (NORMAL, ALIGNED TO THE X-AXIS AND SLIGHTLY ROTATED) FOR COMPUTING LINE SLOPES VALUES IN RELATION TO THE X-AXIS AND TO THE LONGEST SIDE OF THE OBJECT ORIENTED BOUNDING BOX (OOB).

object	TOP 10 precision		Recall	
	X-axis	OOB	X-axis	OOB
normal	0.5	0.9	0.26	0.63
rotated	0.3	0.7	0.16	0.42

most suitable algorithm for our approach is the Edge Boxes [11]. It is based on the idea that when there are many contours enclosed by bounding box, there is a high likelihood that it contains an object. As an edge extractor a Structure Edge detector is used. All contours are examined using a sliding window approach. In order to improve the computation time, efficient data structures are used. The result of the algorithm is a set of bounding box objects proposals.

In this paper we propose two approaches using Edge Boxes:

- first - where shapes extraction is only performed for edges in the found bounding boxes
- second - where information about object bounding boxes is used as an assistance to the shapes extraction algorithm

The first approach highly decreases the number of shapes detection areas which should highly improve the computation time. Moreover, the primitives extracted from each bounding box should be stored in a one complex shape which should improve the query comparison time and precision.

The second approach detects primitives in the whole image area and then uses Edge Boxes proposals in order to strength the links between shapes in the same bounding box. This is performed by adding connections between unconnected nodes in a graph. Due to that fact, all shapes which are meant to be in a one object should be connected. Such an approach should also improve the precision of comparisons with graphs in the database but may decrease the computation time.

V. THE DATABASE STRUCTURE

The database structure is based on a tree which stores similar object's graphs in the same part of a subtree. Two types of tree nodes are defined:

- common graph nodes
- data nodes

The common graph nodes are used in order to store the common parts of graphs which are stored in its children. For example, if there are two children with a car and a bicycle graphs, as a common graph, wheels are used. The common graph nodes does not store any image information and are only used to check if a whole subtree should be tested or abandoned during the query.

The data nodes are used to store graphs of objects, images and metadatas connected with them. In order to reduce the height of the tree, all similar graphs are stored in the same node in a structure called a slice. A slice is a vector of very similar graphs where the first element is the most similar to the parent common node graph and the last one the least. There may be more than one slices in the data node in order to provide the ability to process queries in parallel.

The tree database structure and operations which may be performed (adding, deleting and querying graphs) were described more detailed in [1].

VI. EXPERIMENTAL RESULTS

In order to examine the proposed approach, an experimental application was written in C++ language for database implementation and python language with Django, HTML and JavaScript for GUI. The database stores images of three types of objects: cars, bicycles and motorbikes.

Firstly the computation of primitives attributes in relation to the object oriented bounding boxes was examined. In order to compare the results, two commonly used coefficients were used:

$$precision = \frac{number\ of\ relevant\ results\ images}{total\ number\ of\ results\ images} \tag{1}$$

$$recall = \frac{number\ of\ relevant\ results\ images}{total\ number\ of\ relevant\ images\ in\ the\ database} \tag{2}$$

The experimental results are presented in the Table I. As can be seen, when line slopes values are computed in relation to the longest side of the object oriented bounding box, both precision and recall reach much higher values in comparison to the previous version where slopes are computed in relation to the X axis. It can be noted that even if there are used the same images but with rotations, there are still some differences in precision and recall values for them. This problem is caused i.e. by inaccuracies during shapes extraction stage and will be taken into consideration in the future research.

Another tests were performed in order to evaluate the usage of the additional information about mutual positions of all nodes during queries. The results are presented in the Table II. The experiments does not prove that adding such an

TABLE II
THE COMPARISON OF PRECISION AND RECALL RESULTS FOR BICYCLE IMAGES (NORMAL AND ROTATED) AND A CAR IMAGE FOR PREVIOUS ALGORITHM VERSION AND A VERSION WITH INFORMATION ABOUT MUTUAL POSITIONS OF ALL NODES IN THE GRAPH.

object	TOP 10 precision		Recall	
	previous	mutual positions	previous	mutual positions
bike normal	0.5	0.6	0.26	0.79
bike rotated	0.3	0.4	0.16	0.95
car	0.8	0.6	1	1

TABLE III
THE COMPARISON OF NUMBER OF DETECTED PRIMITIVES FOR PREVIOUS VERSION (RGB) AND VERSION WHICH USES HSL COLOR SPACE

primitive	bike		car		motor	
	RGB	HSL	RGB	HSL	RGB	HSL
line segments	24	36	73	54	64	71
arches	2	3	2	2	2	3
polylines	0	3	8	11	2	10

information improves the precision of the query results. For the bicycles images the precision values are only slightly higher, but for the car it is lower than in previous version. Moreover, the computation time was much longer. Contrary, for bicycle images, the recall values are much higher than in previous version. Due to that fact, the usage of additional information about mutual positions of all nodes in a graph is doubtful and more tests should be performed.

Another types of experiments were performed in order to evaluate the shapes detection improvement by a usage of HSL color space. The results are presented in the Table III. There can be noted that for all objects the usage of HSL color space increased the number of detected primitives. The recognition of the third arc in a bicycle is not a mistake, because not only wheels were detected, but also a gearwheel. In the motorbike image also other parts were detected as circles correctly. It can be seen that the number of polylines is much higher when using HSL color space. This is caused by higher number of detected base line segments and more precise detection which allowed to create more complex primitives.

Due to the limited time, practical experiments for usage of the Edge Boxes algorithm will be performed as a future research.

VII. CONCLUSIONS AND FUTURE WORKS

In this paper a new Content Based Image Retrieval algorithm based on a sketch method was proposed. The main idea of the algorithm is based on decomposing an object into predefined set of shapes (primitives): line segments, polylines, polygons, arches, polyarches and arc-sided polygons. All primitives are stored as a graph in order to store the mutual relations between them. As an improvement to the algorithm, a conversion to the HSL color space was proposed in order to detect primitives more accurately. Moreover, computing all line slopes in relation to the object oriented bounding box was proposed which also increased the precision of the results. The experiments which were performed proved that such propositions increased both precision and recall values. Additionally, in order to better detect objects present in images, the usage of Edge Boxes algorithm was proposed which will be tested as a future research. The proposed results improvement by the information about mutual positions of all nodes is doubtful and more tests should be performed.

The future research includes testing the usage of Edge Boxes and deciding which proposed approach would be more suitable for most situations. Moreover, more tests should be performed for all proposed improvements in this paper. Additionally, the computation time should be also tested for different algorithm versions. Another tests may be performed in order to examine different database structure implementations (e.g. using SD2DS data structures and relational databases). Moreover, the primitives detection algorithm could be improved, e.g. using approaches proposed in [12]. Also other graph comparison algorithms may be evaluated, e.g. using the optimization methods with constraints [13].

REFERENCES

[1] S. Deniziak and T. Michno, "New content based image retrieval database structure using query by approximate shapes," in *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*, Sept 2017. doi: 10.15439/2017F457 pp. 613–621.

[2] H. P. Kriegel, P. Kroger, P. Kunath, and A. Pryakhin, "Effective similarity search in multimedia databases using multiple representations," in *2006 12th International Multi-Media Modelling Conference*, 2006. doi: 10.1109/MMMC.2006.1651355. ISSN 1550-5502 pp. 4 pp.–.

[3] M. Mocofan, I. Ermalai, M. Bucos, M. Onita, and B. Dragulescu, "Supervised tree content based search algorithm for multimedia image databases," in *2011 6th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI)*, May 2011. doi: 10.1109/SACI.2011.5873049 pp. 469–472.

[4] C. Y. Li and C. T. Hsu, "Image retrieval with relevance feedback based on graph-theoretic region correspondence estimation," *IEEE Transactions on Multimedia*, vol. 10, no. 3, pp. 447–456, April 2008. doi: 10.1109/TMM.2008.917421

[5] S. Parui and A. Mittal, "Sketch-based image retrieval from millions of images under rotation, translation and scale variations," *CoRR*, vol. abs/1511.00099, 2015. [Online]. Available: http://arxiv.org/abs/1511.00099

[6] H. H. Wang, D. Mohamad, and N. A. Ismail, "Approaches, challenges and future direction of image retrieval," *CoRR*, vol. abs/1006.4568, 2010.

[7] A. Singh, S. Shekhar, and A. Jalal, "Semantic based image retrieval using multi-agent model by searching and filtering replicated web images," in *Information and Communication Technologies (WICT), 2012 World Congress on*, Oct 2012. doi: 10.1109/WICT.2012.6409187 pp. 817–821.

[8] M. M. Mirończuk and J. Protasiewicz, "A recent overview of the state-of-the-art elements of text classification," *Expert Systems with Applications*, 2018.

[9] T. Kato, T. Kurita, N. Otsu, and K. Hirata, "A sketch retrieval method for full color image database-query by visual example," in *11th IAPR International Conference on Pattern Recognition, Vol.I. Conference A: Computer Vision and Applications*, Aug 1992, pp. 530–533.

[10] S. Deniziak and T. Michno, "Content based image retrieval using query by approximate shape," in *2016 Federated Conference on Computer Science and Information Systems (FedCSIS)*, Sept 2016, pp. 807–816.

[11] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014. ISBN 978-3-319-10602-1 pp. 391–405.

[12] M. Woźniak and D. Połap, "Adaptive neuro-heuristic hybrid model for fruit peel defects detection," *Neural Networks*, vol. 98, pp. 16 – 33, 2018. doi: https://doi.org/10.1016/j.neunet.2017.10.009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608017302526

[13] P. Sitek and J. Wikarek, "A hybrid programming framework for modeling and solving constraint satisfaction and optimization problems," *Scientific Programming*, vol. 2016, 2016. doi: 10.1155/2016/5102616