Partial Shape Retrieval by M-Tree and a Bayesian Approach

Saïd Mahmoudi and Mohamed Daoudi
MIIRE Research Group (INT / LIFL UMR CNRS 8022)
Cité Scientifique, Rue Guglielmo MARCONI
59658 Villeneuve d'ascq, France
mahmoudi@enic.fr, daoudi@enic.fr

Abstract

An important problem in accessing and retrieving visual information is to provide efficient similarity machining in large databases. In this paper we propose partial retrieval by the shape similarity using Curvature Scale Space and an effective indexing using metric trees and a Bayesian approach. Each shape is partitioned into token attributes. Tokens are organized in a tree structure called M-tree. We focus on the study of the partial search ef ciency.

1. Introduction

The visual information retrieval (VIR) systems are concerned with efficient storage and record retrieval. Recently, several systems combine heterogeneous attributes to improve discrimination and classification. These systems use color, texture and shape features for image queries. The shape can be used to represent local image information. In this case, the problem is complicated because a shape does not have a mathematical definition that exactly matches what the user feels as a shape. This concept is not typically present in the spirit of the person tempting to send a request. The well-known distances measures commonly used in mathematics are not suitable to represent a shape similarity as perceived by humans in reality.

The shape similarity based on local descriptors and effective indexing has been proposed by S. Berreti et al. [1]. Each contour is represented by a set of token (partition of contour). Each token is represented by two features, the angle and the orientation. Two shapes are considered similar if they share tokens with similar curvature and orientation according to a distance measure. Tokens are organized as *M-tree*. Mehrota and Gary [7] have developed a retrieval technique known as Feature Index-Based Similar Shape Retrieval. Boundary features are organized as kdb-tree.

In [8], maximum of curvature scale space (CSS) as new shape representation for planar curves has been used.

A similar solution has been proposed by Daoudi and Matusiak [3]. This approach was included in MPEG-7 standard [6]. Each maximum represents the convexity or the concavity of the contour.

It was shown in [8] and [3] that the CSS representation has some properties as:
1. The CSS representation is invariant under the similarity group (the composition of a translation, a rotation, and a scale factor).
2. Completeness: This property ensures that two contours will have the same shape if and only if all their CSS are equal.
3. Stability gives robustness under small distortions caused by quantization, noise.
4. Simplicity and real time computation: This property is very important in database applications. However, no indexing schema was proposed.

In this paper we propose retrieval by the shape similarity using in CSS and an effective indexing using metric trees. Each shape is partitioned into tokens attributes. Tokens are organized in a tree structure called *M-tree* [2].

We present a new algorithm of partial shape indexing, which uses the *M-tree* structure combined with curvature scale space. We propose also to combine this partial search method with a bayesian approach.

This paper is organized in the following way. Section 2 describes the way of representing a curve as a set of parts named tokens by using curvature scale space description (CSS). In Section 3 we present the shape and Token distances used. Section 4 present how to combine the shape description and the distance proposed with the tree structure named *M-tree*. Section 5 describe a probabilistic approach combined to the M-tree partial search. In section 6, we present some experimental results. A comparative analysis between the *M-tree* and *R-tree* structure for shape retrieval is also carried out.

2. Curvature scale space (CSS) description

An important problem in computer vision is the representation of shape. This applies both to the features extracted from image and to the components of object models. We describe the representation of image curves γ, which correspond to the contours of objects, as they appear in the image, this is useful for matching and recognition task.

The curve γ is parameterized by the arc-length parameter. The smoothing curve γ, at multiple scales is done by using low-pass Gaussian filter, applied to the original boundary. Inflexion points of this curve at each scale are corresponding to the Curvature Scale Space (CSS) descriptor [3].

The result of this process can be represented in (u, σ) plane as a binary image called CSS image of the curve. For every u we have a certain curve which in turn, has some curvature zero crossing points. As u increases, the curve becomes smoother and the number of zeros crossings decreases.

The curvature extrema and zeros are often used as breakpoints for segmenting the curve into sections corresponding to shape primitives. The zeros of curvature are points of inflection between positive and negative curvatures. Simply the breaking of every zero of curvature provides the simplest primitives, namely convex and concave sections.

Figure (1) shows an original boundary, and its CSS, which is composed of a set of sub-curves called Peaks P_i.

The X and Y axis respectively represent the normalized arc length (u) and the standard deviation (σ). For each u, we represented the values of the various arc length corresponding to the various zero crossing.

Each Contour is decomposed into parts called tokens, figure (1), using CSS. Each token is delimited by two inflexion points. We notice that each token is corresponding to one Peak of CSS, figure (1). Each peak (P_i) describes a corresponding region of the curve.

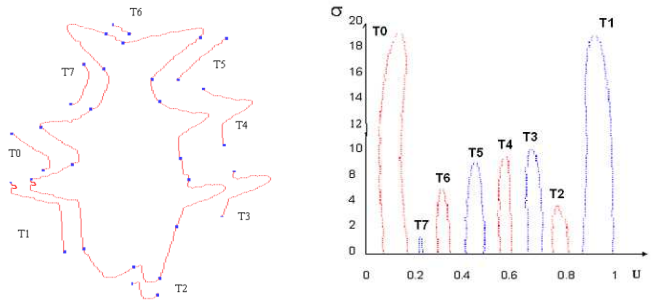

Figure 1. Parts shape with CSS descriptor

3. Token and shape distance

In order to compare the index based on CSS, the geodesic distance has been used by Eberly [4]. Given two points (u_1, σ_1) and (u_2, σ_2), with $u_1 < u_2$, their distance D can be defined in the following way:

$$D((u_1, \sigma_1), (u_2, \sigma_2)) = \log\left(\frac{\sigma_2}{\sigma_1}\frac{\left(1+\sqrt{1-(\varphi\sigma_1)^2}\right)}{\left(1+\sqrt{1-(\varphi\sigma_1)^2}-\varphi L\right)}\right)$$

where

$$\varphi = \frac{2L}{\sqrt{\left(\sigma_1^2-\sigma_2^2\right)^2 + L^2(L^2+2(\sigma_1^2+\sigma_2^2))}}$$

$$L = \|u_1 - u_2\|,$$

L is the euclidean distance.

To calculate distance between two tokens we have to calculate the distances between there relative Peaks. We propose to calculate the distance between two peaks belonging to two different CSS by using a set points uniformly distributed on the two peaks, with step of 10.
The distance between two peaks P_i and P_j is defined by the function d:

$$d(P_i, P_j) = \sum_{\substack{(u_p,\sigma_p) \in P_i \\ (u_q,\sigma_q) \in P_j}} D\left((u_p, \sigma_p)(u_q, \sigma_q)\right)$$

with: $p = (n * step)\ n = 0 \ldots\ max_i/step$
and $q = (n * step)\ n = 0 \ldots\ max_j/step$
max_i : the max number points of P_i
max_j : the max number points of P_j

Given two shapes A and B, the similarity measurement S, between them is defined as the sum of distances d between their relative tokens. This measurement is not necessary a metric.

4. Shape indexing using CSS and *M-tree*

We propose to structure the peaks corresponding to the CSS of each shape in a tree structure known as *M-tree* "Metric Tree" [1][2]. The *M-tree* structure supposes the use of a metric distance between the components "tokens" of the tree. This tree structure stores the set of peaks of each CSS on the basis of their relative distances, and that in order to avoid the computation of some distances on the search process. In this method we propose to index shape using all the information contained on the CSS, we define a similarity measure using trees.

Each shape is decomposed into a set of tokens using CSS, figure (1). The tree structure *M-Tree* organizes tokens as hierarchical set of clusters [2], figure (3).

Entries in leaf nodes of the tree contain sets of couples (u,σ) corresponding to each token. Figure (2) shows the organization of the tokens of figure (1) into an *M-tree* structure. Clusters size is equal to 3.

Each node entry has the following format:

Entry$(t_i) = [t_i,\ ptr(T(t_i)),\ r(t_i),\ d(t_i, P(t_i))]$

Where:

t_i: is the feature vector of the indexed token if the node is a leaf, the Token identifier otherwise.

Example: for a given Token t_0:
If t_0 is in leaf node:
$t_0 = ((u_1, \sigma_1), (u_2, \sigma_2), \ldots, (u_n, \sigma_n))$,
else $t_0 = 0$, (token identifier of t_0).

$Ptr(T(t_i))$: is a pointer to the root of the sub-tree $T(t_i)$, which includes all t_j such that their distance t_i is less then the covering radius $r(t_i)$: $\forall\, t_j \in T(t_i),\ d(t_i, t_j) \leq r(t_i)$.

In the leaf nodes, $ptr(.)$ is replaced with the identifier of the shape to which the Token belongs.

Distances $d(t_i, P(t_i))$ and $r(t_i)$ are precomputed so that the number of accessed nodes and the number of distances computations are reduced at search time. Given a query token t_q and a range r, a range query selects all the database tokens t_i, such that: $d(t_i, t_q) \leq r$.

Triangle inequality is used to prune a sub-tree from a search path. In fact, we can easily proof these two properties: [2]

Property -1-

$$\text{if}\ \ d(t_i, t_q) > r + r(t_i)$$

$$\text{then}\quad d(t_j, t_q) > r\ \ \forall\ \ t_j\ \in\ T(t_i)$$

Property -2-

$$\text{if}\quad |\, d(P(t_i), t_q) - d(t_i, P(t_i))\, | > r + r(t_i),$$

$$\text{then}\ \ d(t_i, t_q) > r + r(t_i).$$

Since traversing tree is from root to leaf nodes, for an entry t_r at a given level:
$d(P(t_r), t_q)$: was already been computed in previous stages when traversing tree.
$d(t_r, P(t_r))$: is stored in the node corresponding to t_r.
Indeed, if condition of property 2 is verified, it is possible to prune the sub-tree $T(t_r)$ without having to compute any new distance at all.

5 Probabilistic approach for partial search

In the following, each contour of a query image is denoted by Q, and the set of model images in our database by $I = I_m$, with $\widehat{m}$ being the number of model images.

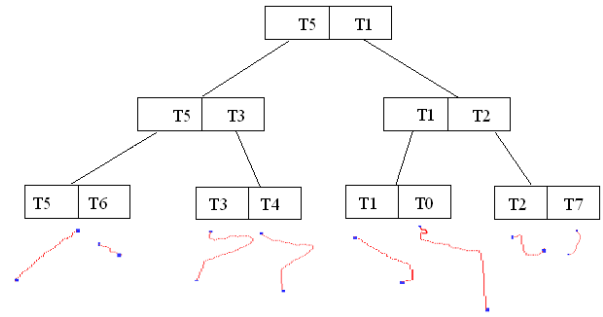

Figure 2. M-tree corresponding to figure (1)

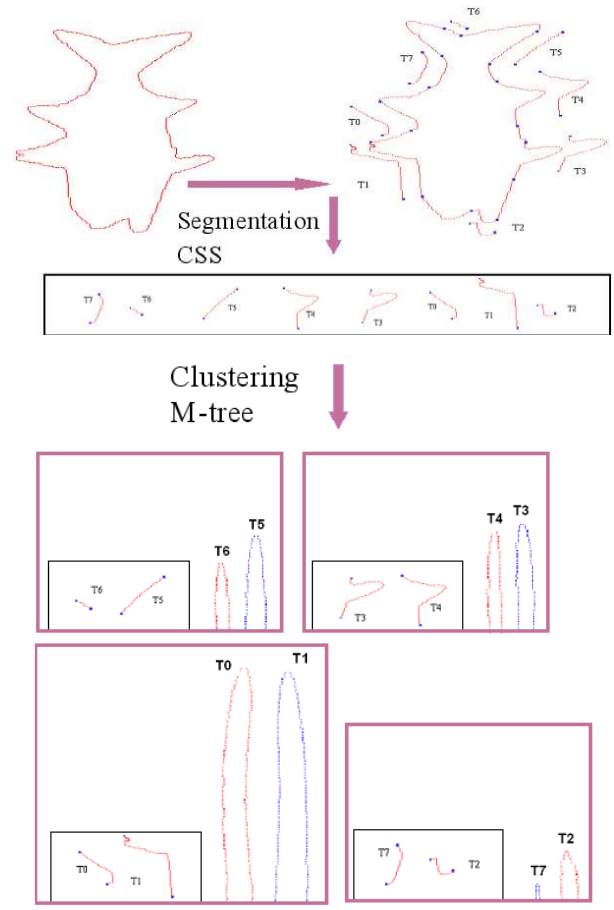

Figure 3. Parts shape clustering with CSS and M-tree

Each contour I (either a query or model image) is characterized by a CSS descriptor $C(I) = P_i$ with P_i being peaks of CSS and $\widehat{P}$ the number of peaks of the CSS relative to the image.

Given a query image Q, we want to find similar model images, i.e. the model with the highest probability.

Since use of CSS as shape descriptor allows to retrieve the parts of shape, figure (3), we use a Bayesian voting approach based on parts appearance. Thus, given a scene part,

the goal is to identify the model candidate which most likely accounts for the presence of this scene part. To achieve this goal, we adopt a Bayesian framework where the measure of discriminatory power of part for a model is defined in terms of posterior probability [9].

We assign that the prior probability to parts of the same model are based on their relative visual relevance calculated according to the distance between shape and parts.

Let $\mathrm{P}(I_k/p_i)$ be the posterior probability that reflect the upload belief that model I_k appears in the scene after the part p_i is observed.

In order to derive the posterior probability $\mathrm{P}(I_k/p_i)$ we need to estimate prior probabilities $\mathrm{P}(p_i/I_k)$ and $\mathrm{P}(I_k)$, we assume that $\mathrm{P}(I_k)$ is the same for every image or view in I. Once certain part, p_i, is decided to be used for indexing, we compute the likelihood $\mathrm{P}(p_i/I_k)$.

$$\mathrm{P}(p_i/I_k) = \frac{e^{-d(p_i,I_k)}}{\sum\limits_{\forall\ p_j \in\ Q} d(p_j, I_k)} \tag{1}$$

We notice that for each part $p_j \in Q$:

$$\sum_{\forall\ p_j \in\ Q} \mathrm{P}(p_j/I_k) = 1$$

where $d(p_i, I_k)$ denotes the distance between I_k and part p_i corresponding to a peak of the CSS belonging to the query image Q, the posterior probability is then computed as:

$$\mathrm{P}(I_k/p_i) = \frac{\mathrm{P}(I_k)\quad \mathrm{P}(p_i/I_k)}{\sum\limits_{\forall\ I_k \in\ I} \mathrm{P}(p_i/I_k)} \tag{2}$$

This formula is used to determine the distribution of the posterior probabilities among the models I_k in I indexed by p_i.

Once the posterior probability $\mathrm{P}(I_k/p_i)$ is computed for every model $I_k \in I$ and every peak p_i , we can rank models by:

$$\mathrm{R}_{bvo}(I_k, Q) = \sum_{\forall\ p_i \in\ Q} \mathrm{P}(I_k/p_i) \tag{3}$$

where R_{bvo} denote the rank of the image, I_k, with Bayesian voting.

6. Experiments and results

We use the Vision, Speech, and Signal Processing Surrey University database, which contain about 1000 images of marine creatures. Each image shows one distinct species on uniform background. Each image is processed in order to recover the boundary contour. An index is associated to each contour. This index is composed of a set of couples, which is uniformly distributed on each peak of the CSS related to the image. The final phase of indexing consists of the construction of the *M-tree* corresponding to this index.

Figure(4) shows the first seven results corresponding of the search for the image query on figure (4), in our database.

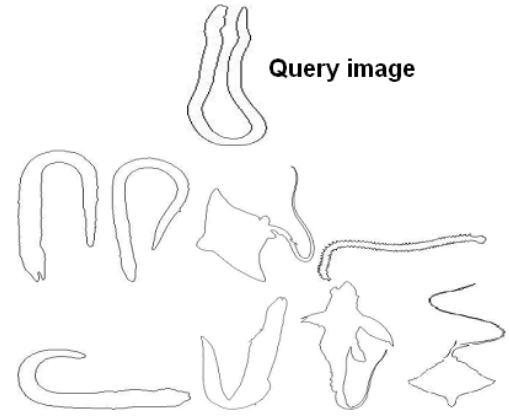

Figure 4. Results of search

We use the curves recall/precision on figure (5) to compare results of seach process using *M-tree* and the sequential method without *M-tree*. Results shows that the first results are most relevant, and that the use of full information CSS combined with partial search method with *M-tree*, improve accuracy of search.

Figure (6) shows curves recall/precision comparing results of search process using M-tree and sequential method both combined with the bayesian proposed approach. We notice that the probabilistic approach increase accuracy of search, since for this method more first results are similar to the request.

Indexing efficiency is evaluated by comparing the modified *M-tree* with respect to the R-tree [5]. Figures (7) compare the number of distances computations d and running time, between the method using *M-tree*, the sequential method and R-tree structure. The results obtained (distance computations and running time) show that *M-Tree* outperforms *R-tree*.

On figure (8), we show that the M-tree and sequential method combined with the probabilistic approach doesn't change calculating time. We can deduce that the *M-Tree* method with the bayesian approach give very good search satisfactory without increasing computing time.

Partial search efficiency is evaluated by comparing search results of the total query in figure (9) with the re-

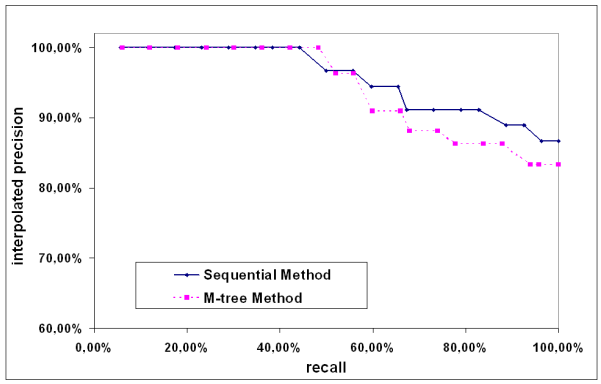

Figure 5. Curve recall/precision comparison between M-tree method and sequential method

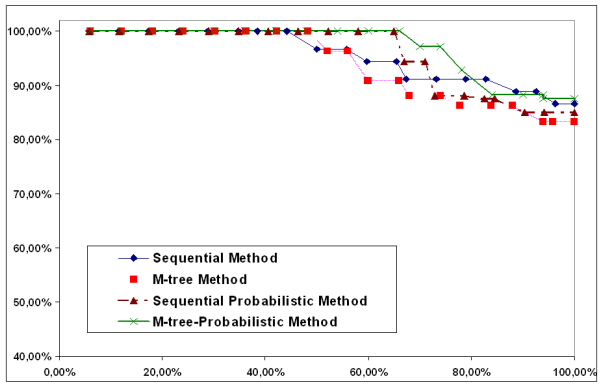

Figure 6. Curve recall/precision comparison between M-tree method and sequential method combined with probabilistic approach

sults of the same search using three partial queries, figure (9). The curve recall/precision in figure (10) shows a little decrease of search accuracy for first, second and third partial queries. But search accuray for the three partial queries still very good. We can deduce that the method using *M-tree* combined with the bayesian approach increase partial search efficiency.

7. Conclusion

In this paper we proposed retrieval by the shape similarity using in Curvature Scale Space and an effective indexing using metric trees combined with a bayesian voting approach. Two distances have been proposed which model respectively token similarity (which defined the *M-Tree*) and shape similarity.

Experimental results obtained show the retrieval effectiveness and indexing efficiency. Test results show that the use of an *M-Tree* based index permits to increase partial accuracy of search and outperform traditional indexing on the R-Tree structure. Results shows that the Bayesian approach combined with the proposed *M-Tree* method increase considerabely the accuracy of search results.

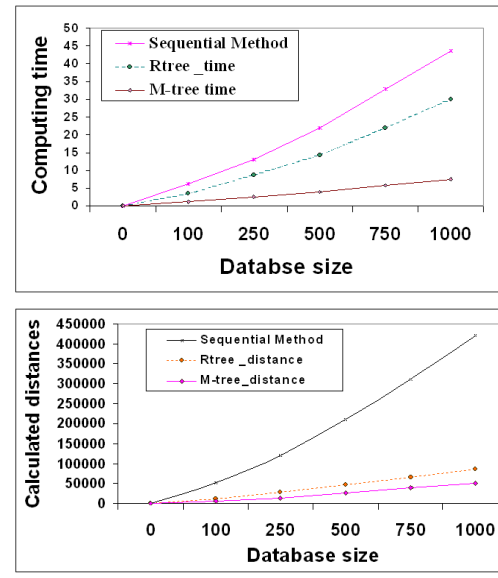

Figure 7. Computing time and average distances number as function of the data set size

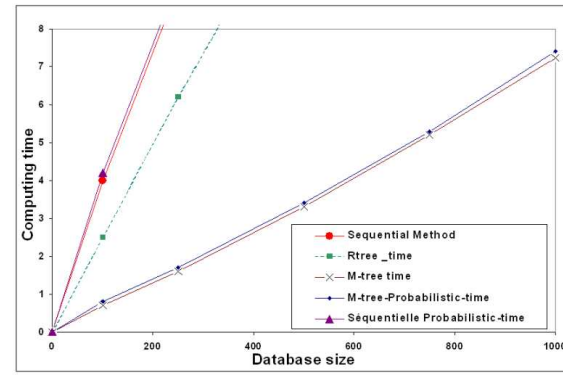

Figure 8. Computing time as function of the data set size for all proposed method

References

[1] S. Berreti, A. D. Bimbo, and P. Pala. Retrieval by shape similarity with perceptual distance and effective indexing. *IEEE Transaction On Multimedia*, 2(4), December 2000.

[2] P. Ciaccia, M. Patella, and P. Zezulan. Indexing metric spaces with m-tree. In *SEBD 97*, pages 67–86, 1997.

[3] M. Daoudi and S. Matusiak. Visual image retrieval by multiscale description of user sketches. *Journal of Visual Comput-*

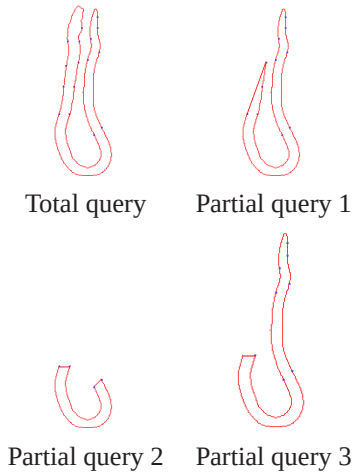

Figure 9. Partial queries

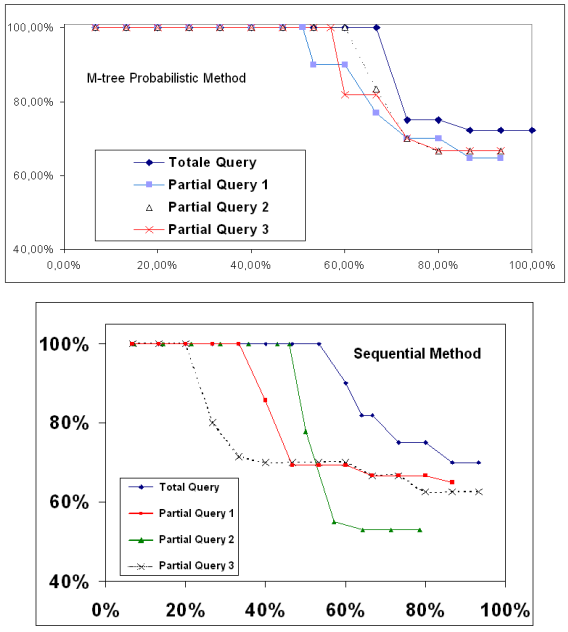

Figure 10. recall/precision for partial queries

ing and languages, 11:pages 287–301, 2000. special issue on image database visual querying and retrieval.

[4] D. Eberly. *Geometric Methods for analysis of Ridges in N-dimensional Images*. Phd thesis, University of North Carolina at Chapell Hill, 1994.

[5] A. Guttman. R-tree: A dynamic index structure for spacial searching. In *The 1984 ACM SIGMOD International Conference on Management of Data*, pages 47–57, Boston, MA, June 1984.

[6] B. Manjunath, P. Salembier, and T. Sikora. Introduction to mpeg-7 multimedia content description interface. Chichester, 2002.

[7] R. Mehrota and J. E. Gary. A similar-shape retrieval in shape data management. *IEEE Computer*, 18(9):57–62, December 1995.

[8] F. Mokhtarian, S.Abbasi, and J.Kittler. Robust and efficient shape indexing through curvature scale space. *Proceedings of the sixth British Machine Vision Conference, BMVC 96*, pages 53–62, 10-12 September 1996.

[9] J. Yi and S.Chelberg. Rapid object recognition from a large model database. In *IEEE 2nd CAD-Based Vision Workshop, Champion, PA*, pages 252–265, 1994.