View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by International Institute for Science, Technology and Education (IISTE): E-Journals

Computer Engineering and Intelligent Systems
ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online)
Vol.4, No.2, 2013
www.iiste.org

Fast Retrieval Algorithm Using EMD Lower and Upper Bounds and a Search Algorithm in multidimensional index

Bahri abdelkhalak , Hamid zouaki

Department of Mathematics and computer science, Faculty of Science, El Jadida

Modélisation Mathématiques et Informatique Décisionnelle

Bahri_abdelkhalak@yahoo.fr

Hamid_zouaki@yahoo.fr

Abstract. Comparison of images requires a distance metric that is sensitive to the spatial location of objects and features. The Earth Mover's Distance was introduced in Computer Vision to better approach human perceptual similarities. Its computation, however, is too complex for usage in interactive multimedia database scenarios. We develop new upper bounding approximation techniques for the Earth Mover's Distance which satisfy high quality criteria and fast computation. In order to enable efficient query processing in large databases, we propose an index structure LUBMTree (Lower and Upper Bounds MTree), based of using the lower and upper bounds for the EMD to improve the search time.

Experiments show the performance of research in the LUBMTree compared with those obtained by the research in the MTree.

Keywords : indexing, similarity, search, signature, metric EMD, MTree, MAM.

1. Introduction

Any image database, whether a newspaper photo archive, a repository for biomedical images, or a surveillance system, must be able to compare images in order to be more than an expensive file cabinet. Content-based access is key to making use of large image databases, such as a collection of decades' worth of diverse photographs or the torrent of images from new, high-throughput microscopes [1]. Having a notion of distance is also necessary for global analyses of image collections, ranging from general-purpose techniques such as clustering and outlier detection to specialized machine learning applications that attempt to model biological processes. Comparing two images requires a feature extraction method and a distance metric. A feature is a compact representation of the contents of an image. The MPEG-7 standard [2] specifies a number of image features for visual browsing. A distance metric computes a scalar distance between two features: examples are the Euclidean (L2) distance, the Manhattan (L1) distance, and the Mahalanobis distance [3]. The choice of image feature and distance metric depends on the nature of the images, as well as the kind of similarity one hopes to capture. For many classes of images, the spatial location is important for whether two images should be considered similar. For example can be constructed from photographs or surveillance images where one wishes to discount small rotations or translations in defining image similarity. The earth mover's distance (EMD), first proposed by Werman et al. [6], captures the spatial aspect of the different features extracted from the images. The distance between two images measures both the distance in the feature space and the spatial distance.

The EMD has strong theoretical foundations [7] and is robust to small translations and rotations in an image. It is general and flexible, and can be tuned to be having like any Lp-norm with appropriate parameters. It has also been successfully applied to image retrieval based on contours [10] and texture [11], as well as similarity search on melodies [12], graphs [13], and vector fields [14].The complexity of the EMD algorithm is $O(N^3 \log N)$ [1], where N is the number of clusters in the signatures. Some derivation algorithms of the EMD have been proposed to reduce the time computation. Pele and Werman [16]

proposed the fast algorithm of $\hat{EMD}$. Ling and Okada [15] proposed the fast algorithm of EMD_ L1. The $\hat{EMD}$ can be obtained by replacing the ground distance of the EMD with the threshold distance. EMD_L1 is a replacement of the ground distance with the L1 distance. Both $\hat{EMD}$ and EMD_L1 are different from the original EMD. They only indicate high-speed calculation methods in the case that the thresholded distance and L1 distance were used as the ground distance.

To reduce the search time of the query image, we propose a new metric access method called LUBMTree (Lower Upper Bound MTree), which is an extension of the MTree[10] index structure.

The rest of the paper is organized as follows. Section 2 Present the paper contributions. Section 3 Describe the EMD distance measure. Section 4 present the lower bounds for the EMD, and introduces the upper bounds for the EMD. Section 5 Describe the MTree and the proposed index structures LUBMTree . Section 6 evaluates the search in the proposed index structure experimentally. Section 7 discusses related work, before Section 8 concludes the paper. Finally, section 9 the future work.

2. Paper contributions:

This paper presents the upper bounds for the EMD, and the index structure LUBMTree (Lower Upper Bound MTree), that extend the index structure MTree[10]. This our index structure based of using the lower and upper bounds for the EMD. The goal of this index structure is to speed up similarity search.

3. The Earth Mover distance

According to the classification result of pixel images, the signatures of images can be of different size, and hence the Euclidean distance, Bhattacharyya distance…etc are not feasible, hence the choice of EMD (Earth Mover Distance) [8] distance, and that one of the most efficient distance to compare sets of features, see e.g. [17, 18].

The EMD is based on the well-known transportation problem. Suppose some suppliers, each with a given amount of goods, are required to supply some consumers, each with a given limited capacity to accept goods. For each supplier-consumer pair, the cost of transporting a single unit of goods is given. The transportation problem is: Find a minimum expensive flow of goods from the suppliers to the consumers that satisfy the consumers' demand.

The Earth Mover's Distance [Rubner et al., 1998] is one of the distances that can work efficiently on signatures which are not necessarily the same number of bins.

The EMD is the minimum amount of work to change a signature into another. The concept of "work" is based on the quantity of the contents of the signature to be transported from one component to another and the distance chosen by the user to measure the distance between two components. The number of components to compare two signatures may be different, as well as the sum total of the components of the two signatures. Two signatures are compared

$$P = \{(p_1, w_1), \cdots, (p_m, w_m)\} \quad and \quad Q = \{(q_1, u_1), \ldots, (q_n, u_n)\}$$

P is the first signature containing m clusters p_i of weight w_i, and Q the second signature containing n clusters q_j of weight u_j. According to [1], the first step is to find all content portions of f_{ij} to carry the signature of the component i to component j that minimize the cost (work) as follows:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij} f_{ij}$$

Where F={f_{ij}} denotes a set of flows. Each flow f_{ij} represents the amount transported from the i-th supply to the j-th demand.

D=[dij] the ground distance matrix, where d_{ij} is the ground distance between clusters p_i and q_j with the following constraints:

$$f_{ij} \geq 0, 1 \leq i \leq m, 1 \leq j \leq n$$

$$\sum_{j=1}^{n} f_{ij} \leq w_i, 1 \leq i \leq m$$

$$\sum_{i=1}^{m} f_{ij} \leq u_j, 1 \leq j \leq n$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} = \min\left(\sum_{i=1}^{m} w_i, \sum_{j=1}^{n} u_j\right)$$

The first constraint only allows movement of components from P to Q.

The following two constraints limit the amount of displaced components of P, and quantity of components received by Q at their respective weights. The last constraint implies the maximum possible displacement of components.

EMD distance is then defined as

$$EMD(q, p) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij} f_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}}$$

EMD distance is normalized to the minimum sum of weights of signatures to avoid favoring smaller signatures when comparing signatures of different sizes.

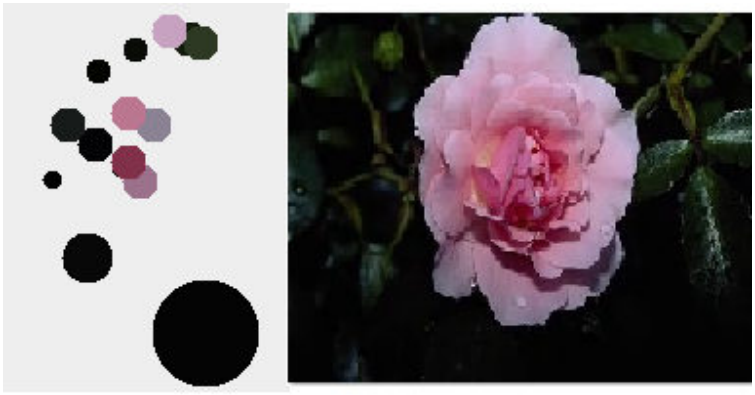
Image Retrieval Using the EMD.

Color distribution data applied for CBIR models is extracted from color segmentation images. As shown in Figure 1, the distribution consists of the weight, flow cost, and image feature. The weight is the number of pixels in each region. The flow cost of each distribution can be obtained by calculating the Euclidean distance between the image features in each region. The image feature of each region consists of the center position (x, y) and the average color signature (Lab).

Weight: The number of pixels in each region

Flow cost: The Euclidean distance between image features

Image feature : The center position and color signature (Lab) in each region



(a) **(b)**

Figure 1. *An example of a color signature. (a) Color signature. (b) Original image.*

4. the bounds for the EMD

4.1 Lower bound for the EMD.

In this section we describe several lower bounds for the EMD:

4.1.1 Rubner and AL. [8] defined the lower bound for the EMD by:

Theorem: EMD Lower Bound (Y. Rubner and C. Tomasi 2001)

Given signatures P and Q, let pi and qj be the coordinates of luster i in the first signature, and cluster j in the second signature respectively.

Then if $\sum_{i=1}^{m} \omega_{pi} = \sum_{j=1}^{n} \omega_{qj}$

Then $EMD(P,Q) \geq \left\| \overline{P} - \overline{Q} \right\|$

Where $\| \ \|$ is the norm that induces the ground distance, and $\overline{P}$ and $\overline{Q}$ are the centers of mass of P and Q respectively: $\overline{P} = \frac{\sum_{i=1}^{m} w_{pi} p_i}{\sum_{i=1}^{m} w_{pi}}$, $\overline{Q} = \frac{\sum_{j=1}^{n} w_{qj} qj}{\sum_{j=1}^{n} w_{qj}}$

Proof: See [8]

4.1.2 I.Assent, A.Wenning, T.Seidl [19] Introduce the L_p-norm-based lower bounds defined are:

The advantages of using (weighted) L_p norms are twofold. First, a single comparison is computed in linear time O(n) in an n dimensional histogram space. Secondly, these distance functions are immediately index enabled and supported by any available multi-dimensional indexing structure.

A. Weighted L_1 lower bound [19]

Theorem Weighted L_1 lower bound LB_{Man}

For any metric ground distance encoded in a cost matrix C = [cij], the EMD between two histograms x = (x_1, . . . , x_n) and y = (y_1, . . . , y_n) with the same total mass, i.e. $m = \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$, is always greater than or equal to the weighted Manhattan distance with weights $w_i = \min_{j, i \neq j} \left\{ \frac{c_{ij}}{2.m} \right\}$. Formally:

$$EMD\ (x, y) \geq \sum_{i=1}^{n} \min_{\substack{j \\ i \neq j}} \left\{ \frac{c_{ij}}{2.m} \right\} |x_i - y_i|$$

Proof: see [19]

B. Weighted L_2 lower bound [19]

Theorem Weighted L_2 lower bound LB_{Eucli}

For any metric ground distance encoded in a cost matrix C = [cij], the EMD between two histograms x = (x_1, . . . , x_n) and y = (y_1, . . . , y_n) with the same total mass, i.e. $m = \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$, we have :

$$EMD(P,Q) \leq \sqrt{\sum_{i=1}^{n} (\min_{\substack{j \\ i \neq j}} \left\{ \frac{c_{ij}}{2.m} \right\})^2 (x_i - y_i)^2}$$

Proof: see [19]

C. Weighted L_∞ lower bound [19]

Theorem Weighted L_∞ lower bound LB_{Max}

For any metric ground distance encoded in a cost matrix C = [cij], the EMD between two histograms x = (x_1, . . . , x_n) and y = (y_1, . . . , y_n) with the same total mass, i.e. $m = \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$, we have :

$$EMD\ (x, y) \geq \max_{i} \left\{ \min_{\substack{j \\ i \neq j}} \left\{ \frac{c_{ij}}{m} \right\} |x_i - y_i| \right\}$$

Proof: see [19]

4.2 Upper bound for the EMD.

In this part we propose the upper bounds of the metric EMD that maximise the distance EMD between two histograms. The upper bounds proposed based of the L_p-norm.

4.2.1 Weight L_1 upper bound

Theorem Weighted L_1 upper bound UB_{Man}

For any metric ground distance encoded in a cost matrix C = [cij], the EMD between two histograms x = (x_1, . . . , x_n) and y = (y_1, . . . , y_n) with the same total mass, i.e. $m = \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$, is always greater than or equal to the weighted Manhattan distance with weights $w_i = \min_{j, i \neq j} \left\{ \frac{c_{ij}}{2.m} \right\}$. Formally:

$$EMD\ (x, y) \leq \sum_{i=1}^{n} \max_{\substack{j \\ i \neq j}} \left\{ \frac{2.c_{ij}}{m} \right\} |x_i - y_i|$$

Proof:

$$EMD\ (x, y) = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ i \neq j}}^{n} \frac{c_{ij}}{m} f_{ij}$$

$$\leq \sum_{i=1}^{m} \max_{\substack{j \\ i \neq j}} \left\{ \frac{c_{ij}}{m} \right\} \sum_{\substack{j=1 \\ i \neq j}}^{n} f_{ij}$$

$$\leq \sum_{i=1}^{n} \max_{\substack{j \\ i \neq j}} \left\{ \frac{2.c_{ij}}{m} \right\} |x_i - y_i|$$

4.2.2 Weighted L_2 upper bound

Theorem Weighted L_2 lower bound UB_{Eucli}

For any metric ground distance encoded in a cost matrix C = [cij], the EMD between two histograms x = (x_1, . . . , x_n) and y = (y_1, . . . , y_n) with the same total mass, i.e. $m = \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i$, we have :

$$EMD(P,Q) \le \sqrt{\sum_{i=1}^{n} (\max_{j} \left\{ \frac{2.c_{ij}}{m} \right\})^2 (x_i - y_i)^2}$$

Proof:

$$EMD(x, y) \le \sum_{i=1}^{n} \max_{\substack{j \\ i \neq j}} \left\{ \frac{2.c_{ij}}{m} \right\} |x_i - y_i|$$

$$= \sqrt{(\sum_{i=1}^{n} \max_{\substack{j \\ i \neq j}} \left\{ \frac{2.c_{ij}}{m} \right\} |x_i - y_i|)^2}$$

$$\le \sqrt{\sum_{i=1}^{n} (\max_{\substack{j \\ i \neq j}} \left\{ \frac{2.c_{ij}}{m} \right\})^2 (x_i - y_i)^2}$$

5. Metric access methods

Metric access methods (MAMs) [20] are index structures designed to perform efficiently similarity queries in metric spaces. They only use especially the triangle inequality, to filter out objects or entire regions of the space during the search, thus avoiding the sequential (or linear) scan over the database.

MAMs can be classified into two main groups: (1) Pivot based MAMs select from the database a number of pivot objects, and classify all the other objects according to their distance from the pivots (2) MAMs based on compact partitions divide the space into regions as compact as possible.

Each region stores a representative point (local pivot) and data that can be used to discard the entire region at query time, without computing the actual distance from the region objects to the query object. Each region can be partitioned recursively into more regions, inducing a search hierarchy.

5.1. M-Tree

A. Basic Principles

The M-tree [10] is a dynamic (meaning easily updatable) index structure that provides good performance in secondary memory. The M-tree is a hierarchical index, where some of the data points are selected as centers (local pivots) of regions and the rest of the objects are assigned to suitable regions in order to build up a balanced and compact hierarchy of data regions. Each region (branch of the tree) is indexed recursively. The

data is stored in the leaves of the M-tree, where each leaf contains ground entries (grnd(Oi), Oi ∈ S). The internal nodes store routing entries (rout(Oi), Oi ∈ S).

The M-tree organizes the objects into fixed-size nodes. Each node can store up to M entries—this is the capacity of M-tree nodes. An entry for a routing object Or also include a pointer, denoted ptr(T(Or)), which references the root of a sub-tree, T(Or), called the covering tree of Or, a covering radius r(Or) > 0, and d(Or, P(Or)), the distance to the parent object P(Or), i.e. the routing object which references the node where the Or entry is stored.

entry (Or) = [Or, ptr(T(Or)), r(Or), d(Or, P(Or))]

For each (ground) DB object, one entry having the format

entry(Oj) = [Oj, oid(Oj), d(Oj, P(Oj))]

is stored in a leaf node, where oid(Oj) is the identifier of the object, which is used to provide access to the whole object resident on a separate data file.

Starting at the root level, a new object Oi is recursively inserted into the best subtree T(Oj), which is defined as the one where the covering radius rOj must increase the least in order to cover the new object. In case of ties, the subtree whose center is closest to Oi is selected. The insertion algorithm proceeds recursively until a leaf is reached and Oi is inserted into that leaf, at each level storing the distance to the routing object of its parent node (so-called to-parent distance). Node overflows are managed in a similar way as in the B-tree. If an insertion produces an overflow, two objects from the node are selected as new centers, the node is split, and the two new centers are promoted to the parent node. If the parent node overflows, the same split procedure is applied. If the root overflows, it is split and a new root is created. Thus, the M-tree is a balanced tree.

The semantics of the covering radius is the following: All the objects stored in the covering tree of Or are within the distance r(Or) from Or, i.e. ∀Oi ∈ T(Or), d(Or,Oi) ≤ r(Or).

A routing object Or, hence, defines a region in the metric space M, centered on Or and with radius r(Or) (see Figure 2).

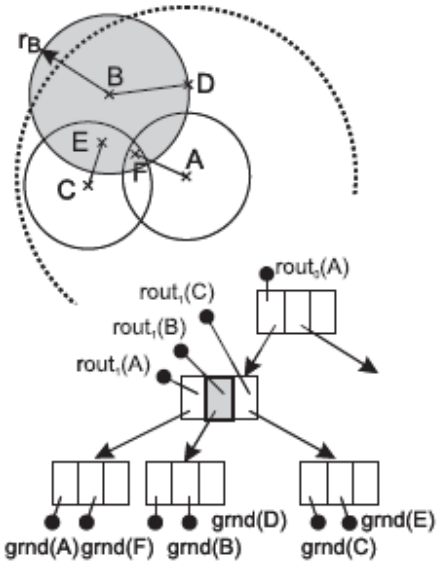


Figure 2. An example of a MTree

The M-tree, therefore, organizes the space into a set of (possibly overlapping) regions, to which the same principle is recursively applied. The covering radius, r(Or), and the distance between the object and its parent object, d(Or, P(Or)), both stored in each entry of the tree, are used to "prune" the search space during the search phase

B. Searching the M-tree

Before presenting specific algorithms for building the M-tree, we show how the information stored in nodes is used for processing similarity queries. Clearly, the performance of search algorithms is largely influenced by the actual construction of the M-tree, even though the correctness and the logic of search are independent of such aspects.

In order to minimize both the number of accessed nodes and computed distances, all the information concerning (pre-computed) distances which are stored in the M-tree nodes, i.e. d(Oi, P(Oi)) and r(Oi), is used to efficiently apply the triangle inequality property.

The query range (Q, rQ) requests for all the database objects Oj , such that $d(Oj,Q) \leq rQ$. Therefore, the algorithm rangeQuery has to follow all such paths in the M-tree which cannot be excluded from leading to objects satisfying the above inequality. Since the query threshold, rQ, does not change during the search process, and provided the response set is required as a unit, the order with which nodes are visited has no effect on the performance of **rangeQuery** algorithm.

Since, when accessing node N, the distance between Q and Op, the parent object of N, has already been computed, it is possible, by applying the triangle inequality, to prune a whole sub-tree without computing any new distance at all. The condition for pruning is as follows.

Lemma 1 *[25]*

If d(Or,Q) > rQ + r(Or), then, for each object Oj in T(Or), it is d(Oj,Q) > rQ. Thus,

T(Or) can be safely pruned from the search.

Proof: Since d(Oj,Or) ≤ r(Or) holds for any object Oj in T(Or), it is

d(Oj,Q) ≥ d(Or,Q) − d(Oj,Or) (triangle inequality)

≥ d(Or,Q) − r(Or) (def. of covering radius)

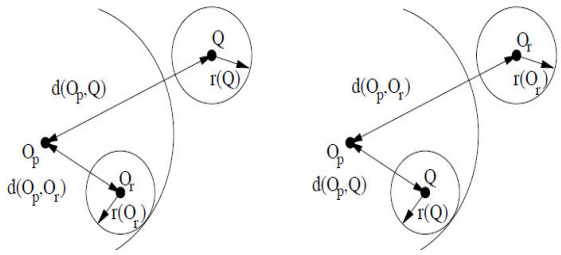
> rQ (by hypothesis)

In order to apply Lemma 1, the distance d(Or,Q) has to be computed. This can be avoided by taking advantage of the following result.

Lemma 2 [25]

If |d(Op,Q) − d(Or,Op)| > rQ + r(Or), then d(Or,Q) > rQ + r(Or).

Proof: This is a direct consequence of the triangle inequality, which guarantees that both d(Or,Q) ≥ d(Op,Q) − d(Or,Op) and d(Or,Q) ≥ d(Or,Op) − d(Op,Q) hold (see Figure 3).



a) b)

Figure 3. How Lemma 2 is used to avoid computing distances. Case a): d(Or,Q) ≥d(Op,Q)−d(Or,Op) > r(Q)+r(Or); Case b): d(Or,Q) ≥ d(Or,Op)−d(Op,Q) > r(Q)+r(Or). In both cases computing d(Or,Q) is not needed.

Range query algorithm in M-tree[25]

Listing 1. (range query algorithm)

```
QueryResult      rangeQuery( Node N, Query (Q,rQ))
{       // if N is root then d(Or,Op)=d(Op,Q)=0
      Let P be the parent rooting object of N
       If   N   is not a leaf then {
     For   each root ( Or) in N do {
            If |d(Op,Q) -   d(Or,Op)|  ≤  r(Or)+rQ      then
```



```
                    Compute   d(Or,Q)
            If d(Or,Q)≤  ≤  r(Or)+rQ          then
                    rangeQuery(ptr(T( Or)),(Q, rQ))
          }
}   /* for each….*/
 } Else{   For   each grnd   (Oj) in N do {
              If | d(Op,Q) -   d(Oj,Op)|  ≤  rQ      then
            Compute   d(Oj,Q)
            If d(Oj,Q)≤   rQ     then
                    Add Oj to the query result
          }
}   /* for each….*/
}
} /* rangeQuery…*/
```

Range queries are implemented by traversing the tree, starting from the root. The nodes which parent region (described by the routing entry) is overlapped by the query ball are accessed (this requires a distance computation). As each node in the tree (except for the root) contains the distances from the routing/ground entries to the center of its parent node (the to-parent distances), some of the non-relevant branches can be further filtered out, without the need of a distance computation, thus avoiding the "more expensive" basic overlap check.

In the MTree structure, the objects in a single node may belong to different classes. During the search if the leaf node is selected, then the distances between the query object and all objects in it are calculated. And therefore we can have additional calculations between the query object and the objects that are not similar to this one. So to reduce the number of calculations we propose a new index structure called LUBMTree

5.2. LUBMTree

The LUBMTree is a metric access method, and it is an extension of MTree. Its goal is to accelerate the search of objects. By using our index structure LUBMTree, the original EMD can be calculated efficiently. The characteristic of the search in our structure is that it does not necessarily calculate all distances with different objects of the same node of the LUBMTree index structure. To limit these calculations, we use the EMD lower and upper bounds.

5.2.1 The structure of LUBMTree

The LUBMTree structure has the same attributes as MTree, in addition two other attributes. The first attribute represent the lower bound of the EMD distance between the routing object and its parent. The second attribute represent the lower bound of the EMD distance of the covering radius of routing object.

The construction of the structure LUBMTree is built in the same way as MTree, while adding two attributes lower bounds, and two other attributes upper bounds for each object.

5.2.2 Searching in LUBMTree

Lemma 3

Let P be the parent object of a data region (R, r). Let

$d^{lb}(\cdot)$ and dub(.) are a lower and upper-bounding distance to EMD(·) respectively.

$$\text{If } d^{ub}(P,Q) - d_p^{ub} > r + \varepsilon_w \vee d_p^{lb} - d^{lb}(P,Q) > r + \varepsilon_w$$

Then the data region is not relevant to the query and can be filtered.

Proof: This is a direct consequence of the triangle inequality, which guarantees that both $d^{ub}(R,Q) \geq r + \varepsilon w$ and $d^{lb}(R,Q) \geq r + \varepsilon w$ hold (see Figure 4).

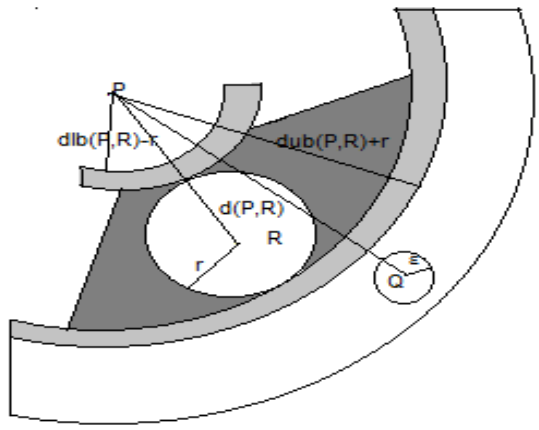


Figure 4. Outer parent filtering in LUBMTree.

5.2.3 Similarity queries

We present the modified range query algorithm (see Listing 2). The k-NN algorithm can be modified the same way (for both original query algorithms on M-tree we refer to [10]).

Listing 2. (range query algorithm)

QueryResult **rangeQueryLUB**(Node N, RQuery (Q,ε))

{

// if N is root then d(R,O)=d(P,Q)=0

Let P be the parent rooting object of N

Let's dlb(R,P)= be the lower bound of the EMD(R,P) distance

```
        If   N   is not a leaf then {
      For   each root ( R) in N do {
              If d^ub(P,Q) -  d^ub(R,P)  ≤  r+ε      and      // lemme 3
                 d^lb(R,P) – d^lb(P,Q)  ≤  r+ ε     then {   // lemme 3
                  If d(P,Q) -   d(R,P)  ≤  r +ε   and // lemme 2
                                 d(R,P)   – d(P,Q)  ≤  r + ε     then {
                 Compute   d(R,Q)
                 If d(R,Q)≤  r_ub + ε     then      // lemme 1
                      rangeQueryLUB(ptr(T( R)),(Q, ε))
            }
  }
          }   /* for each….*/
       }
        Else{
      For   each grnd   ( R) in N do {
                    If d^ub(P,Q) -   d^ub(R,P)  ≤  r+ε      and      // lemme 3
                 d^lb(R,P) – d^lb(P,Q)  ≤  r+ ε     then {   // lemme 3
                  If d(P,Q) -   d(R,P)  ≤  r +ε   and // lemme 2
                                 d(R,P)   – d(P,Q)  ≤  r + ε     then {
                 Compute   d(R,Q)
                 If d(R,Q)≤    ε     then
                          Add R to the query result
            }
    }
     }   /* for each….*/
 }
} /* rangeQuery…*/
```

6. Experimental evaluation

We performed a number of range queries using 50 classes of the Coil-100 database, which each class contains 72 images. We use also the Wang database, composed by 10 classes, which each class contains 100 images.

We implemented our proposed range query search algorithm and compared it against an MTree with bulk loading (best efficiency). The platform on which the experiments were run is a PC with a 5.6 Ghz processor and 4 Gb of main memory.

As efficiency measures, we used the CPU time needed to compute the results of queries.

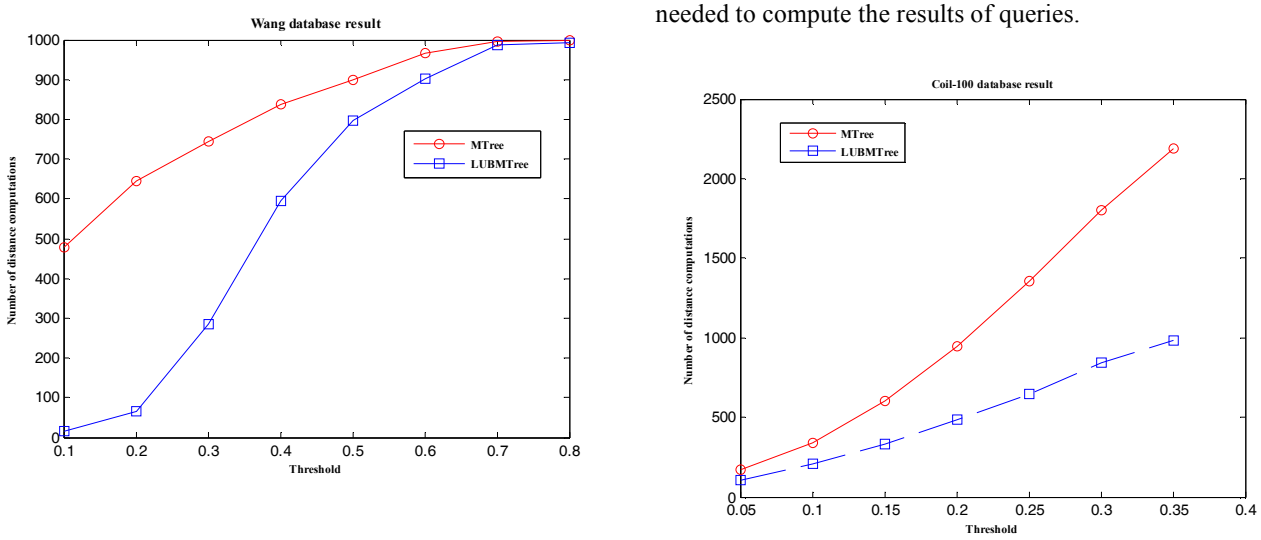


Figure 5 .Number of distance computations calculate in the range query when searching in the LUBMTree compared by the search in in the MTree

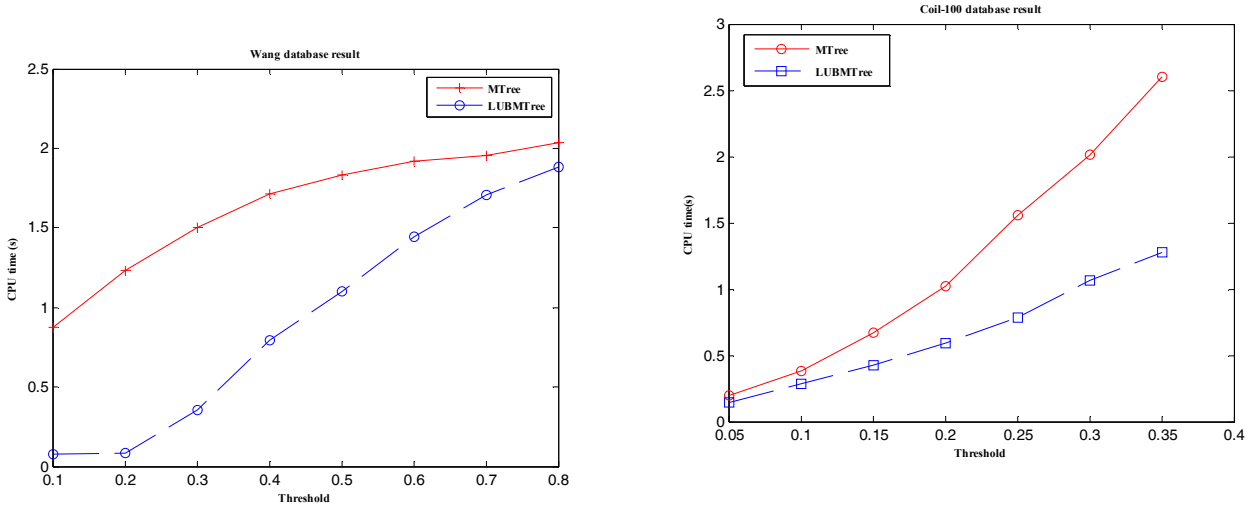


Figure 6. The time computation calculate in the range query when searching in the LUBMTree and in the MTree

Using optimal query processing, we varied the threshold of nearest neighbours from 0.05 to 0.35, we use the signature sizes 16 of images obtained by the classification theirs pixels by KD-Tree [24], and we use our structure describe in the first of this section. In the tow our experiments, we evaluate the performance of our structure by measuring the selectivity. The bounds used in our experiments are the LB_{Euclid} et UB_{Euclid} as

the lower and upper bounds respectively. In the figure 5, the search in the MTree produce more number of distance computations than the search in the LUBMTree, and so produce more image candidates. The figure 6, show the response time obtained when search in the MTree and in our index structure. We see that the search in our structure is faster than the search in the MTree, because the number of the calculated distances in the search algorithm in our structure is less than the one of the MTree. This improvement is explain by the search algorithm in the proposed index uses the lower and the upper bounding to skipping the supplementary calculs.

7. Related Work

Werman et al. [6] define the *match distance* for multidimensional histograms and suggest its application to texture features, shape matching, and image comparison. For the latter, the intensity of pixels is used as the mass. Peleg et al. [7] formulate the match distance as an LP problem. Rubner et al. [5] introduce the name *earth mover's distance*, and study image retrieval using color distributions and texture features.

Rubner et al. [5] derive a lower bound —the distance between the centers of mass (in feature space) of the two images—for the EMD. Their bound disregards position information, as the center of mass of each image lies at the physical center of the image and contributes zero to the bound. I.Assent, A.Wenning, T.Seidl [19] also derive a lower bounds based of the Lpnorm. M.Shishibori, D. Koizumi, and K. Kita [23] use the lower bounding distance of EMD and combines it with a calculation-skipping algorithm.

We implemented their lower bound, and we introduce the upper bound based of Lp-norm. We use the upper and the lower bound for improve the MTree[10] index structure.The characteristic of our algorithms is that unnecessary calculations can be skipped by using the EMD lower and upper bounds. As a consequence, the search in our index structures is significantly faster.

8. Conclusion

Search in large image or other multimedia databases highly dependents on the underlying similarity model. The Earth Mover's Distance, proposed in Computer Vision literature, is an interesting new approach towards achieving high-quality content-based retrieval. Despite its advantages, this distance measure is computed via a linear programming algorithm which is too slow for today's huge and interactive multimedia.

The disadvantage of the EMD metric is the significant response time. To solve this problem, we have proposed a new index structure called LUBMTree. According to the results, we notice that the search in the our proposed index structure show it effectiveness in terms of time compared to the search in the MTree.

The search algorithm in the proposed index uses the lower and the upper bounding to skipping the supplementary calculs.

9. Future work

We plan to adapt the PM-tree [21], a MAM that combines the lower and upper bounds for the EMD with the pivot-based approach, to improve the response time. For this purpose, we will merge the techniques presented in this paper and the ones described in [22] (pivot-based index for multi-metrics). We expect that, by combining all these technique in one index structure, we will be able to further improve the efficiency of the LUBMTree.

Bibliography

1. Swedlow, J.R., Goldberg, I., Brauner, E., Sorger, P.K.: Informatics and quantitative analysis in biological imaging. Science 300 (2003) 100–102

2. Manjunath, B.S., Salembier, P., Sikora, T., eds.: Introduction to MPEG 7: Multimedia Content Description Language. Wiley (2002)

3. Mahalanobis, P.: On the generalised distance in statistics. Proc. Nat. Inst. Sci. India 12 (1936) 49–55

4. Lewis, G.P., Guerin, C.J., Anderson, D.H., to, B.M., Fisher, S.K.: Rapid changes in the expression of glial cell proteins caused by experimental retinal detachment. Am. J. of Ophtalmol. 118 (1994) 368–376

5. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover's distance as a metric for image retrieval. International Journal of Computer Vision 40 (2000) 99–121

6. Werman, M., Peleg, S., Rosenfeld, A.: A distance metric for multi-dimensional histograms. Computer, Vision, Graphics, and Image Proc. 32 (1985) 328–336

7. Peleg, S., Werman, M., Rom, H.: A unified approach to the change of resolution: Space and gray-level. IEEE Trans. PAMI 11 (1989) 739–742

8. Y. Rubner and C. Tomasi, Perceptual Metrics for Image Databases Navigation, Kluwer Academic Publishers, Dordrecht, The Ntherlands, 2001

9. Stricker, M.A., Orengo, M.: Similarity of color images. In Niblack, C.W., Jain, R.C., eds.: Storage and Retrieval for Image and Video Databases III. Volume 2420 of Proceedings of SPIE. (1995) 381–392

10. P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In Proc. 23rd Conference on Very Large Databases (VLDB'97), pages 426–435. Morgan Kaufmann, 1997.

11. Lazebnik, S., Schmid, C., Ponce, J.: Sparse texture representation using affineinvariant neighborhoods. In: Proc. CVPR. (2003)

12. Typke, R., Veltkamp, R., Wiering, F.: Searching notated polyphonic music using transportation distances. In: Proc. Int. Conf. Multimedia. (2004) 128–135

13. Demirci, M.F., Shokoufandeh, A., Dickinson, S., Keselman, Y., Bretzner, L.: Manyto-many feature matching using spherical coding of directed graphs. In: Proc. European Conf. Computer Vision (ECCV). (2004)

14. Lavin, Y., Batra, R., Hesselink, L.: Feature comparisons of vector fields using earth mover's distance. In: Proc. of the Conference on Visualization. (1998) 103–109

15. H. Ling and K. Okada. EMD-L1 : An Efficient and Robust Al-gorithm for Comparing Histogram-Based Descriptors. ECCV, p. 330-343, 2006.

16. O. Pele and M. Werman. Fast and robust earth mover's distances. In ICCV, 2009.

17. Y. Liu, D. Zhang, G. Lu, W.-Y. Ma, "Region-based image retrieval with high-level semantic color names," IEEE Int. Multimedia Modelling Conference, pp. 180–187, 2005.

18. G. Dvir, H. Greenspan, Y. Rubner, "Context-Based image modelling," ICPR, pp. 162–165, 2002.

19. I.Assent, A.Wenning, T.Seidl . “Approximation Techniques for Indexing the Earth Mover’s Distance in Multimedia Databases”, 22nd IEEE international Conference on Data Engineering (ICDE) 2006, Atlanta, USA

20. E. Chávez, G. Navarro, R. Baeza-Yates, and J. Marroqu´ın. Searching in metric spaces. ACM Computing Surveys, 33(3):273–321, 2001.

21. T. Skopal, J. Pokorny, and V. Snasel. Nearest neighbours search using the PM-tree. In Proc. 10th International Conference on Database Systems for Advanced Applications (DASFAA’05), LNCS 3453, pages 803–815. Springer, 2005.

22. B. Bustos, D. Keim, and T. Schreck. A pivot-based index structure for combination of feature vectors. In Proc. 20th Annual ACM Symposium on Applied Computing, Multimedia and Visualization Track (SAC-MV’05), pages 1180–1184. ACM Press, 2005.

23. M.Shishibori, D. Koizumi, and K. Kita , “Fast Retrieval Algorithm for EarthMover’s Distance UsingEMD Lower Bounds and a Skipping Algorithm”, Advances in Multimedia Volume 2011.

24. J. L. Bentley and J. H. Friedman, \Data structures for range searching," ACM Computing Surveys, vol. 11, pp. 397{409, 1979.

25. Tomáš Skopal Ph.D. Thesis, “ Metric Indexing in Information Retrieval”, 2004