

A BIT OF MULTIMEDIA RETRIEVAL ALGORITHMICS

Remco C. Veltkamp

Department of Information and Computing Sciences

Utrecht University

Remco.Veltkamp@cs.uu.nl

Abstract

After text retrieval, the next waves in web searching and multimedia retrieval are the search for and delivery of images, music, video, and 3D scenes. Not only the perceptual and cognitive aspects, but also many of the algorithmic and performance aspects are still badly understood. One relevant issue is the design of dissimilarity measures (distance functions) that have desired properties. Another aspect is the development of algorithms that can compute or approximate these distances efficiently. Indexing data structures and search algorithms are necessary to make the search more efficient than sequential browsing through large collections.

1 Introduction

Multimedia research has been going on since the nineteen-sixties, even if it was not called like that. A key aspect of multimedia research is interfacing: establishing a seamless interaction and communication between the user and the computer. In that respect it represents an important ingredient of current developments which are denoted by buzz phrases such as ubiquitous computing, ambient intelligence, context awareness, the disappearing computer, video at your fingertips, anything, anyone, anywhere, anytime. Multimedia retrieval is essential for coping with the problems of information overload, in production and content management, and in personalized usage. Indeed, the reason that email and web search engines have become so immensely popular are precisely that they cope with these issues with respect to text. However, if perceptually relevant multimedia methods, that guarantee performance, are not invented soon, there is no hope that similar problems are effectively solved with respect to images, music, video, and 3D models.

Since the first pictorial information systems in the early nineteen-eighties, research has come a long way in developing various methods to handle visual information by its content, as opposed to processing by keywords [1]. However, these

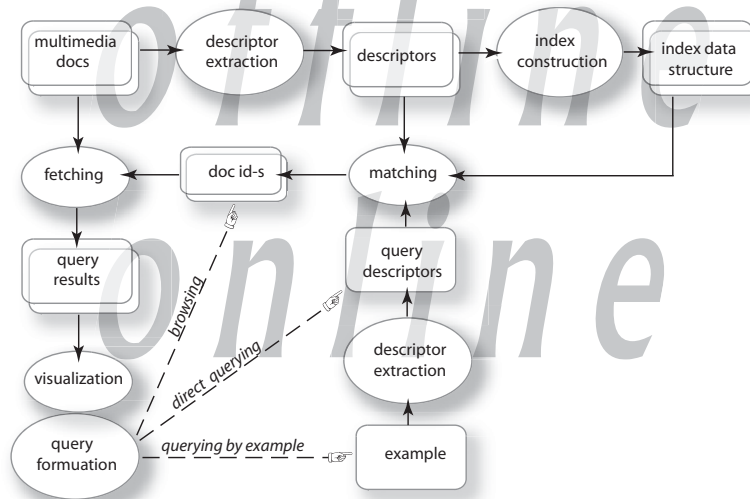


Figure 1: Multimedia retrieval framework.

content descriptions consist of low level color, texture, and shape features [2], and they often miss perceptual relevance. The methods for extracting and comparing these features are primarily heuristic, which, although they are clever themselves, miss guaranteed properties. In contrast, an algorithmic approach is focused on provable properties, see section 1.1.

Looking at a particular multimedia framework as in figure 1, we see that those processes that are of an algorithmic nature are the extraction of features from the multimedia documents, the matching of the query features with the database features, the construction of the indexing data structure to speed up the searching, and the visualization of the resulting retrieved multimedia documents.

The big challenge in multimedia for the next years is the processing of information in a way that is perceptually and semantically relevant. Because of the need for personalized information access and searching, processing should be done in a manner that is guaranteed effective. Because the searching and filtering is performed on very large databases of multimedia information, it must be done with guaranteed efficiency also. The holy grail is not yet within reach. What makes this difficult is the gap between the high level semantic information and the low level features of current multimedia systems. For example, if one is looking for an image of the holy grail on the basis of image content features, one may query for a chalice shape and a star shape. However, simple edge detection yields a set of unconnected lines, not a star. Therefore, low level features will fail miserably for this purpose.

A concrete listing of research issues in multimedia is the following. Firstly, in order to arrive at semantic access, a necessary step is the identification of what is perceptually and cognitively important in the multimedia documents.

Secondly, in order to cope with the data and information overload, it is becoming essential that effective and efficient searching techniques are developed. Indeed, not only company archives contain huge amounts of media. The success of mobile phone with sms (short message service) shows that as soon as consumer groups adopt devices like mobile phones with built-in digital cameras and mms (media message service) via broad band communication like GPRS or UMTS, massive amounts of images and video are produced and stored. Digital music and movies is already causing a very large amount of Internet traffic. The so-called fourth wave in multimedia (after images, video and music), consisting of 3D models and scenes, is showing more and more on the web. Together these media form an enormous amount of data, and it is essential to provide tools to match and filter, and to retrieve personalized information from it.

Thirdly, to make retrieval feasible from such large quantities, efficient searching methods must be invented. In particular, indexing data structures and algorithms must be designed that avoid the need to scan whole collections from front to back, but instead refer the user in a few steps to the right place in the collection.

Fourthly, any successful fully-fledged system needs to provide a combination of image, video, music, and 3D model handling with text capabilities. The integrated system engineering is far from trivial, and challenging in itself.

The algorithmic aspects of these items form an area of research that I have coined *multimedia algorithmics* [3].

1.1 Multimedia Algorithmics

Like all computer systems, all multimedia systems are built on algorithms. Any system in modern information and communication technology is an algorithmic system. When it comes to the design of algorithms for multimedia, this involves for example algorithms for extracting and grouping perceptually relevant patterns, computing the similarity, indexing and searching in large collections, and visualizing retrieval results in a way that is meaningful for relevance feedback. Research issues are the invention of new algorithms that solve problems in an efficient way, guaranteeing provable properties in a rigorous way, taking an axiomatic approach, basing derivations on first principles.

The above-stated aspects are combined into a line of research that is rooted in the discipline of fundamental algorithm design, and applied to the domain of multimedia: multimedia algorithmics. The gap between the high level semantic information and the low level features of current multimedia systems makes it difficult to make a significant step forward. A challenging research agenda for the

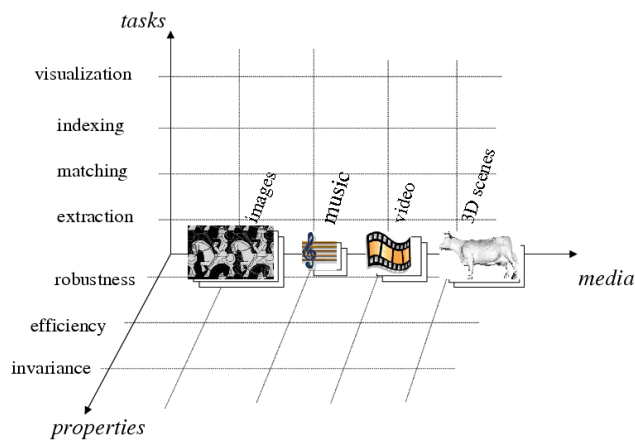


Figure 2: Multimedia research space.

next years is to invent algorithms for multimedia along the following orthogonal axes:

1. The tasks in a typical multimedia framework that are of an algorithmic nature: perceptual feature extraction, pattern matching, indexing, and visualization.
2. The different media (images, music, video, 3D models and scenes) to which these task are applied.
3. The desired properties of algorithms that must be invented: robustness, invariance, and efficiency, etc.

Together, these aspects span a whole research space, as illustrated in figure 2.

This paper discusses some aspects from the field of multimedia algorithmics, and is largely based on [4]. In the next sections we will discuss an algorithm for a particular type of pattern matching, and an algorithm for indexing over large collections to speed up the retrieval.

2 Shape Matching

This section is about one of the aspects mentioned above, matching of patterns. We look at shape matching, in particular matching of multiple polylines to a single polygon.

There is evidence that, for the task of object recognition, the human visual system uses a part-based representation. Biederman [5], for example, suggested that objects are segmented at regions of deep concavity into an arrangement of simple geometric components. For the retrieval of polygonal shapes, we have

therefore developed an algorithm to search for the best matching polygon, given one or more query parts. This dissimilarity measure models partial matching, is translation and rotation invariant, and deformation robust.

Let P_1 be a polyline, and let $P_1(s)$ be the point on P_1 at distance s along the polyline from its beginning. The *turning-angle function* Θ_1 of a polyline P_1 measures the angle of the counterclockwise tangent at $P_1(s)$ with respect to a reference orientation as a function of s . It is a piecewise constant function, with jumps corresponding to the vertices of P_1 . The domain of the function is $[0, \ell_1]$, where ℓ_1 is the length of P_1 . Rotating P_1 by an angle θ corresponds to shifting Θ_1 over a distance θ in the vertical direction.

The turning-angle function Θ_P of a polygon P is defined in the same way, except that the distance s is measured by going counterclockwise around the polygon from an arbitrarily chosen *reference point*. Since P is a closed polyline, we can keep going around the polygon, and the domain of Θ_P can thus be extended to the entire real line, where $\Theta_P(s + \ell_P) = \Theta_P(s) + 2\pi$. Moving the location of the reference point over a distance s along the boundary of P corresponds to shifting Θ_P horizontally over a distance s .

To measure the mismatch between P_1 and the part of P starting at $P(t)$, we align $P_1(0)$ with $P(t)$ by shifting the turning-angle function of P over a distance t and computing the L_2 -distance between the two turning-angle functions, minimized over all possible rotations θ (that is: vertical shiftings of the turning functions). The squared mismatch between P_1 and P , as a function of t , is thus given by:

$$d_1(t) := \min_{\theta \in \mathbb{R}} \int_0^{\ell_1} (\Theta_P(s+t) - \Theta_1(s) + \theta)^2 ds. \quad (1)$$

An ordered set of k polylines $\{P_1, P_2, \dots, P_k\}$ can be represented by concatenating the turning-angle functions of the individual polylines. Thus we get a function $\Theta_{PL} : [0, \ell_k] \rightarrow \mathbb{R}$, where ℓ_j is the cumulative length of polylines P_1 through P_j . For $1 \leq j \leq k$ and $\ell_{j-1} \leq s \leq \ell_j$ we have $\Theta_{PL}(s) := \Theta_j(s - \ell_{j-1})$, so that each polyline P_j is represented by the section of Θ_{PL} on the domain $[\ell_{j-1}, \ell_j]$. The squared mismatch between P_j and P (shifted by t) is now given by:

$$d_j(t) := \min_{\theta \in \mathbb{R}} \int_{\ell_{j-1}}^{\ell_j} (\Theta_P(s+t) - \Theta_{PL}(s) + \theta)^2 ds. \quad (2)$$

We now express the mismatch between the set of polylines $\{P_1, P_2, \dots, P_k\}$ and P as the square root of the sum of squared mismatches between each polyline and P , minimized over all valid shiftings:

$$d(P_1, \dots, P_k; P) := \min_{\text{valid shiftings } t_1 \dots t_k} \left(\sum_{j=1}^k d_j(t_j) \right)^{1/2}. \quad (3)$$

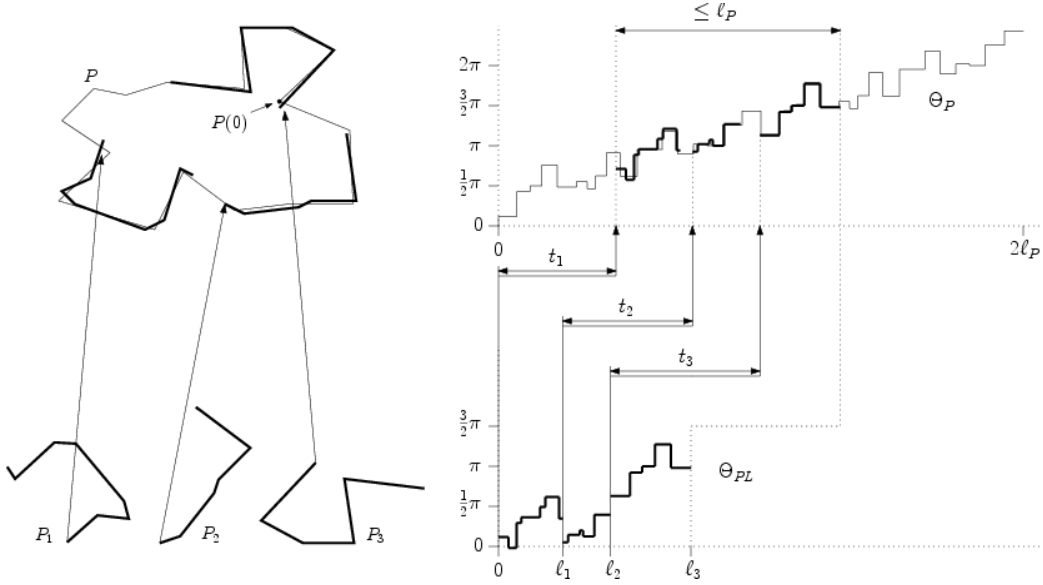


Figure 3: To match polylines P_1, \dots, P_3 to polygon P , we shift the turning functions of the polylines over the turning function of the polygon. To maintain the order of the polylines around the polygon, we need to guarantee $t_1 \leq t_2 \leq t_3$ and $\ell_3 + t_3 \leq t_1 + \ell_P$.

It remains to define what the valid shiftings are. To keep the polylines disjoint (except possibly at their endpoints) and in counterclockwise order around the polygon, each polyline has to be shifted at least as far as the previous one, that is: $t_{j-1} \leq t_j$ for all $1 < j \leq k$. Furthermore, to make sure that P_k does not wrap around the polygon beyond the starting point of P_1 , we have to require that $\ell_k + t_k \leq t_1 + \ell_P$ (see figure 3).

2.1 Algorithm

For $j \in \{2, \dots, k\}$, $1 \leq a \leq |X_1|$, and $b \in \{a, \dots, |X|\}$, with $x_b \leq x_a + \ell_P - \ell_j$, we define:

$$\mathcal{D}[j, b](a) := \min_{\substack{t_2, \dots, t_j \in \{x_a, \dots, x_b\} \\ t_2 \leq \dots \leq t_j}} d_1(x_a) + \sum_{h=2}^j d_h(t_h). \quad (4)$$

It can be shown that there is an optimal solution to the optimization problem in equation (3) such that $t_j \in X$ for $j \in \{1, \dots, k\}$ and $t_1 \in X_1$. Thus, we get $d(P_1, \dots, P_k; P) = \min_{a \in \{1, \dots, |X_1|\}} \mathcal{D}[k, b(a)](a)$, where $b(a)$ is the maximum b such that $x_b \leq x_a + \ell_P - \ell_j$. We now show that an optimal solution of this problem can

Algorithm SimpleCompute $d(P_1, \dots, P_k; P)$

```

1: Compute the sorted set of canonical shiftings  $X = \{x_1, \dots, x_{|X|}\}$ 
2: For all  $j \in \{1, \dots, k\}$  and all  $b \in \{1, \dots, |X|\}$ , evaluate and store  $d_j(x_b)$ 
3:  $minimumMismatch \leftarrow \infty$ 
4: for  $startingPoint \leftarrow 1$  to  $|X_1|$  do
5:    $b \leftarrow startingPoint$ 
6:   for  $j \leftarrow 2$  to  $k$  do
7:      $\mathcal{D}[j, b-1] \leftarrow \infty$ 
8:     while  $x_b + \ell_k \leq x_{startingPoint} + \ell_P$  do
9:        $\mathcal{D}[1, b] \leftarrow d_1(x_{startingPoint})$ 
10:      for  $j \leftarrow 2$  to  $k$  do
11:         $\mathcal{D}[j, b] \leftarrow \min(\mathcal{D}[j-1, b] + d_j(x_b), \mathcal{D}[j, b-1])$ 
12:         $b \leftarrow b+1$ 
13:       $minimumMismatch \leftarrow \min(\mathcal{D}[k, b-1], minimumMismatch)$ 
14: return  $\sqrt{minimumMismatch}$ 

```

Figure 4: Dynamic programming algorithm for optimal placement.

be constructed recursively. Let (t_2, \dots, t_j) be an optimal solution for $\mathcal{D}[j, b](a)$. Regarding the value of t_j we distinguish two cases:

- $t_j = x_b$, in which case (t_1, \dots, t_{j-1}) must be an optimal solution for $\mathcal{D}[j-1, b](a)$, otherwise (t_1, \dots, t_j) would not give a minimum for $\mathcal{D}[j, b](a)$; thus in this case, $\mathcal{D}[j, b](a) = \mathcal{D}[j-1, b](a) + d_j(x_b)$;
- $t_j < x_b$, in which case (t_1, \dots, t_j) must be an optimal solution for $\mathcal{D}[j, b-1](a)$, otherwise (t_1, \dots, t_j) would not give a minimum for $\mathcal{D}[j, b](a)$; thus in this case, $\mathcal{D}[j, b](a) = \mathcal{D}[j, b-1](a)$.

We now conclude for $j \in \{2, \dots, k\}$, $1 \leq a \leq |X_1|$, and $b \in \{a, \dots, |X|\}$, with $x_b \leq x_a + \ell_P - \ell_j$:

$$\mathcal{D}[j, b](a) = \min(\mathcal{D}[j-1, b](a) + d_j(x_b), \mathcal{D}[j, b-1](a)), \quad (5)$$

where the boundary cases are $\mathcal{D}[1, b](a)$, which is $d_1(x_a)$ for all b and all $1 \leq a \leq b$. Note that $\mathcal{D}[j, b](a)$ has no solution when $b < a$. We represent that by defining $\mathcal{D}[j, b](a) = \infty$ in that case.

Equation 5 leads to a straightforward dynamic programming algorithm SimpleCompute for computing the similarity measure $d(P_1, \dots, P_k; P)$, see figure 4. In [6] it is shown that this algorithm takes $O(km^2n^2)$ time and $O(kmn)$ storage. Furthermore, it gives a novel algorithm that runs in $O(kmn \log(mn))$ time and space.

3 Indexing

Proximity searching in multimedia databases has gained more and more interest over the years. In particular searching in dissimilarity spaces (rather than extracting a feature vector for each database object) is an increasing area of research. With growing multimedia databases indexing has become a necessity.

Vantage indexing works as follows: given a multimedia database A and a distance measure $d : A \times A \rightarrow \mathbb{R}$, select from the database a set of m objects $A^* = \{A_1^*, \dots, A_m^*\}$, the so called vantage objects. Compute the distance from each database object A_i to each vantage object, thus creating a point $p_i = (x_1, \dots, x_m)$, such that $x_j = d(A_i, A_j^*)$. Each database object corresponds to a point in the m -dimensional vantage space.

A query on the database now translates to a range-search or a nearest-neighbor search in this m -dimensional vantage space: compute the distance from the query object q to each vantage object (i.e. position q in the vantage space) and retrieve all objects within a certain range around q (in the case of a range query), or retrieve the k nearest neighbors to q (in case of a nearest neighbor query). The distance measure used on the points in vantage space is L_∞ .

Vleugels and Velkamp show [7] that as long as the triangle inequality holds for the distance measure d defined on the database objects, recall (ratio of number of relevant retrieved objects to the total number of relevant objects in the whole data base) is 100%, meaning that there are no false negatives. However, false positives are not excluded from the querying results, so precision (ratio of number of relevant retrieved objects to the total number of retrieved objects) is not necessarily 100%. We claim that by choosing the right vantage objects, precision can increase significantly.

The retrieval performance of a vantage index can improve significantly with a proper choice of vantage objects. This improvement is measured in terms of false positives, as defined below. Let δ be the distance measure in vantage space.

Definition 1. Return set Given $\epsilon > 0$ and query A_q , object A_i is included in the return set of A_q if and only if $\delta(A_q, A_i) \leq \epsilon$.

Definition 2. False positive A_p is a false positive for query A_q if $\delta(A_q, A_p) \leq \epsilon$ and $d(A_q, A_p) > \epsilon$.

We present a new technique for selecting vantage objects that is based on two criteria which address the number of false positives in the retrieval results directly. The first criterion (spacing) concerns the relevance of a single vantage object, the second criterion (correlation) deals with the redundancy of a vantage object with respect to the other vantage objects. We call this method Spacing-based Selection.

The main idea is to keep the number of objects that are returned for a query A_q and range ϵ low. Since false negatives are not possible under the condition that the triangle inequality holds for d , minimization of the number of false positives is achieved by spreading out the database along the vantage space as much as possible. False positives are, intuitively speaking, pushed out of the returned sets.

Spacing In this section we will define a criterion for the relevance of a single vantage object V_j . A priori the query object A_q is unknown, so the distance $d(A_q, V_j)$ between a certain query A_q and vantage object V_j is unknown. The size of the range query (ϵ) is unknown beforehand as well. Optimal performance (achieved by small return sets given a query A_q and range ϵ) should therefore be scored over all possible queries and all possible ranges ϵ .

This is achieved by avoiding clusters on the vantage axis belonging to V_j . Our first criterion therefore concerns the *spacing* between objects on a single vantage axis, which is defined as follows:

Definition 3. *The spacing between two consecutive objects A_i and A_{i+1} on the vantage axis of V_j is $d(A_{i+1}, V_j) - d(A_i, V_j)$.*

Let μ be the average spacing. Then the variance of spacing is given by formula $\frac{1}{n-1} \sum_{i=1}^{n-1} ((d(A_{i+1}, V_j) - d(A_i, V_j)) - \mu)^2$. To ensure that the database objects are evenly spread in vantage space, the variance of spacing has to be as small as possible. A vantage object with a small variance of spacing has a high discriminative power over the database, and is said to be a relevant vantage object.

Correlation It is not sufficient to just select relevant vantage objects, they also should be non-redundant. A low variance of spacing does not guarantee that the database is well spread out in vantage space, since the vantage axes might be strongly correlated.

Therefore, we compute all linear correlation coefficients for all pairs of vantage objects and make sure these coefficients do not exceed a certain threshold. Experiments show that on the MPEG-7 shape images set pairwise correlation is sufficient and that higher order correlations are not an issue.

3.1 Algorithm

Spacing-based Selection selects a set of vantage objects according to the criteria defined above with a randomized incremental algorithm. The key idea is to add the database objects one by one to the index while inspecting the variance of spacing and correlation properties of the vantage objects after each object has been added. As soon as either the variance of spacing of one object or the correlation of a pair

Algorithm SpacingBasedSelection

Input: Database A with objects A_1, \dots, A_n , $d(A, A) \rightarrow \mathbb{R}$, thresholds ϵ_{corr} and ϵ_{spac}

Output: Vantage Index with Vantage objects V_1, V_2, \dots, V_m

- 1: select initial V_1, V_2, \dots, V_m randomly
- 2: **for** All objects A_i **do** in random order
- 3: **for** All objects V_j **do**
- 4: compute $d(A_i, V_j)$
- 5: add A_i to index
- 6: **if** $\text{var}(\text{Spacing})(V_j) > \epsilon_{spac}$ **then**
- 7: remove V_j
- 8: select new vantage object randomly
- 9: **if** for any pair $p(V_k, V_l)$, $\text{Corr}(V_k, V_l) > \epsilon_{corr}$ **then**
- 10: remove p 's worst spaced object
- 11: select new vantage object randomly

Figure 5: Spacing-based Selection.

of objects exceeds a certain threshold, a vantage object is replaced by a randomly chosen new vantage object. These repair steps are typically necessary only at early stages of execution of the algorithm, thus keeping the amount of work that has to be redone small. For details, see the algorithm in figure 5.

The complexity of our algorithm is expressed in terms of distance calculations, since these are by far the most expensive part of the process. The running time complexity is then $O(\sum_{i=0}^n P_i \times i + (1 - P_i) \times k)$ where k is the (in our case constant) number of vantage objects and P_i is the chance that, at iteration i , a vantage object has to be replaced by a new one. This chance depends on the choice for ϵ_{spac} and ϵ_{corr} . There is a clear trade-off here: the stricter these threshold values are, the better the selected vantage objects will perform but also the higher the chance a vantage object has to be replaced, resulting in a longer running time. If we only look at spacing and set ϵ_{spac} such that, for instance, P_i is $(\log n)/i$, the running time would be $O(n \log n)$ since k is a small constant (8 in our experiments).

3.2 Experimental Evaluation

We implemented our algorithm and tested it on the MPEG-7 test set CE-Shape-1 part B, and the distance measure used to calculate the distance between two of these shape images is the Curvature Scale Space (CSS), [8]. To justify our criteria, we manually selected four sets of eight vantage objects that either satisfy both criteria (weakest correlation and lowest variance of spacing: *weak-low*), none (strongest correlation and highest variance of spacing: *strong-high*) or a *strong-*

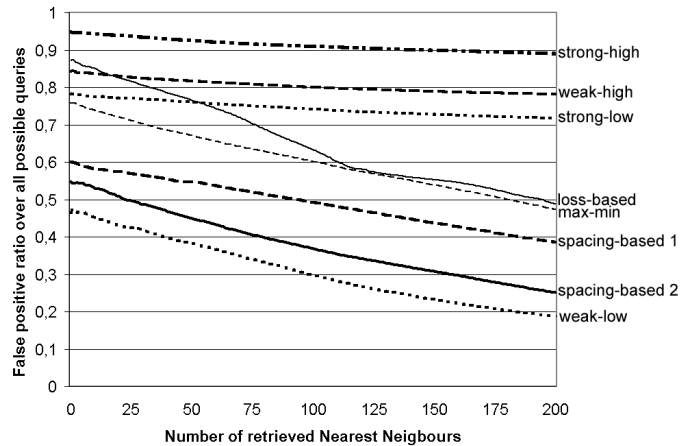


Figure 6: False positive ratios of different algorithms.

low or *weak-high* combination.

The performance of these four sets of vantage objects was evaluated by querying with all 1400 objects. The number of nearest neighbors that was retrieved for each query object varied from 1 to 200. The distance of the furthest nearest neighbor functioned as ϵ , which was used to calculate the number of false positives among these nearest neighbors, see Definition 2. For each vantage index, and all k -NN queries, $k = 1, \dots, 200$, an average ratio of false positives in result was calculated over all 1400 queries. The results are displayed in figure 6, together with some typical runs of our algorithm, the “MaxMin” approach [7] and the “loss-based” approach [9].

These results show that both criteria need to be satisfied in order to achieve good performance (only the set called *weak-low* scores less than 50% false positives for all sizes of nearest neighbor query). Furthermore, it shows that our algorithm can actually select a set of vantage objects in which these criteria are satisfied, since false positive ratios are low for these sets. For more details, see [10].

4 Concluding Remarks

This paper primarily shows multimedia algorithms in the domain of image retrieval, but we have taken a similar approach to music retrieval. As a dissimilarity measure we have designed the Proportional Transportation Distance [11], a normalized version of the Earth Mover’s Distance [12]. It satisfies the triangle inequality, which makes it suitable for indexing with the vantage method. In-

deed, we have used it in combination with the vantage indexing method in our music retrieval systems Muugle (<http://give-lab.cs.uu.nl/muugle>) [13] and Orpheus (<http://give-lab.cs.uu.nl/orpheus/>). The vantage indexing made it possible to identify anonymous incipits (beginnings of pieces, for example twenty notes long) from the RISM A/II collection [14] consisting of about 480,000 incipits [15]. All 80,000 anonymous incipits were compared to the remaining 400,000 ones, giving a total of 32,000,000,000 comparisons. Should a single comparison take 1 ms, this would have taken about 370 days. The vantage indexing made it possible to do this within a day on a 1 GHz PC. A total of 17,895 incipits were identified.

Acknowledgment

I want to thank all persons I have worked with on multimedia retrieval, and with whom the results reported here are obtained. In particular I thank Martijn Bosma, Panos Giannopoulos, Herman Haverkort, Reinier van Leuken, Mirela Tanase, Rainer Typke, and Frans Wiering. This research was supported by the FP6 IST projects 511572-2 PROFI and 506766 AIM@SHAPE.

References

- [1] Smeulders, A.W., Worring, M., Santini, S., Gupta, A., Jain, R.: Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(12) (2000) 1349–1380
- [2] Veltkamp, R.C., Tanase, M.: A survey of content-based image retrieval systems. In Marques, O., Furht, B., eds.: *Content-Based Image and Video Retrieval*, Kluwer (2002) 47–101
- [3] Veltkamp, R.C.: Multimedia algorithmics. *Multimedia Tools and Applications* **27**(2) (2005) 187–193
- [4] Veltkamp, R.C.: Multimedia retrieval algorithmics. In: *Proceedings SOFSEM07*, Springer LNCS 4362. (2007) 138–154
- [5] Biederman, I.: Recognition-by-components: A theory of human image understanding. *Psychological Review* **94**(2) (1987) 115–147
- [6] Tanase, M., Veltkamp, R.C., Haverkort, H.: Multiple polyline to polygon matching. In: *Proceedings 16th Annual Symposium on Algorithms and Computation (ISAAC)*, LNCS 3827. (2005) 60–70
- [7] Vleugels, J., Veltkamp, R.C.: Efficient image retrieval through vantage objects. *Pattern Recognition* (2002) 69–80

- [8] Mokhtarian, F., Abbasi, S., Kittler, J.: Efficient and robust retrieval by shape content through curvature scale space. In: Proceedings of IDB-MMS'96. (1996) 35–42
- [9] Henning, C., Latecki, L.J.: The choice of vantage objects for image retrieval. *Pattern Recognition* (2003) 2187–2196
- [10] van Leuken, R.H., Veltkamp, R.C., Typke, R.: Selecting vantage objects for similarity indexing. In: Proceedings of the 18th International Conference on Pattern Recognition (ICPR). (2006)
- [11] Giannopoulos, P., Veltkamp, R.C.: A pseudo-metric for weighted point sets. In: Proceedings European Conference on Computer Vision (ECCV 2002), LNCS 2352, Springer (2002) 715–730
- [12] Rubner, Y.: Perceptual Metrics for Image Database Navigation. PhD thesis, Stanford University, Department of Computer Science (1999)
- [13] Bosma, M., Veltkamp, R.C., Wiering, F.: Muugle: A music retrieval experimentation framework. In: Proceedings of the 9th International Conference on Music Perception and Cognition. (2006) 1297–1303
- [14] Répertoire International des Sources Musicales (RISM): Serie A/II, manuscrits musicaux après 1600. K. G. Saur Verlag, München, Germany (2002)
- [15] Typke, R., Giannopoulos, P., Veltkamp, R.C., Wiering, F., van Oostrum, R.: Using transportation distances for measuring melodic similarity. In: Proceedings of the Fourth International Conference on Music Information Retrieval (ISMIR 2003). (2003) 107–114