Fast Identification of Visual Documents Using Local Descriptors

Eduardo Valle
ETIS UMR CNRS 8051
6, ave du Ponceau BP 44F
95014 Cergy-Pontoise cedex France
+33 1 30 73 66 10
mail@eduardovalle.com

Matthieu Cord
LIP6 — UPMC
4, place Jussieu
75252 Paris cedex 05 France
+33 1 44 27 71 39
matthieu.cord@lip6.fr

Sylvie Philipp-Foliguet
ETIS UMR CNRS 8051
6, ave du Ponceau BP 44F
95014 Cergy-Pontoise cedex France
+33 1 30 73 66 10
sylvie.philipp@ensea.fr

ABSTRACT

In this paper we introduce a system for the identification of visual documents. Since it stems from content-based document indexing and retrieval, our system does not need to rely on textual annotations, watermarks or other metadata, which can be missing or incorrect. Our retrieval system is based on local descriptors, which have been shown to provide accurate and robust description. Because of the high computational costs associated to the matching of local descriptors, we propose **Projection KD-Forest**: an indexing technique which allows efficient approximate k nearest neighbors search. Experiments demonstrate that the Projection KD-Forest allows the system to provide prompt results with negligible loss on accuracy. The Projection KD-Forest also compares well when contrasted to other strategies of k nearest neighbors search.

Categories and Subject Descriptors

H.3.1 [**Information storage and retrieval**]: Content analysis and indexing – *indexing methods.* H.2.2 [**Database management**]: Physical design – *access methods.* H.2.4 [**Database management**]: Systems – *multimedia databases.*

General Terms

Algorithms, Performance, Design, Experimentation.

Keywords

Image retrieval, document identification, copy detection, multidimensional indexing, k nearest neighbors search, local descriptors.

1. INTRODUCTION

"Which image is this?" — this deceivingly simple question is a major source of difficulties for institutions possessing large iconographic collections. Often, users come to libraries and archives with unidentified images taken from newspaper clippings, books and even postcards. When the references are missing or incorrect the visual document itself is the only reliable evidence for the identification of the original and the retrieval of its context — author, title, subject, etc. — which are essential to uncover its meaning. But since the collection consists of hundreds of thousands of items, unassisted search is infeasible. That simple question becomes very difficult to answer, frustrating the users. The absence of immediate means to automatically identify a visual document makes also difficult to curb piracy. Institutions whose profit depends on selling protected material are forced to continuously look for unlicensed uses of their images. This labor intensive work can be greatly streamlined if the most probable misused images can be detected and associated to their originals.

Document identification or *copy detection* can solve those issues and consists in taking a query image and finding the original from where it derives, together with any relevant metadata. In this paper we describe a system for image identification. Our main contribution is a very efficient indexing scheme, which allows to profit from the excellent accuracy provided by local descriptors without the serious performance penalties usually associated to those. The indexing is based on a clever association of multiple moderate-dimensional KD-Trees, which minimizes boundary effects and boosts precision, while keeping access times low.

2. DOCUMENT IDENTIFICATION

A copy is a document resulting from the application of a *transformation* on an *original document*. Document identification can, thus, be defined as finding the set of original images from which a query image derives. The allowed transformations varies from application to application but usually includes translations, rotations, scale changes, photometric and colorimetric transformations, cropping, occlusions, noise, and any combination of those.

Traditional retrieval systems, based on textual keywords, are not up to the job of document identification. They either depend on manual annotation, which is expensive, or the harvesting of contextual text, which is imprecise and depends on the availability of text around the image. In addition, the users may not be knowledgeable enough about the image in order to provide textual descriptors (for example, they may want to find the original of a portrait precisely in order to know whose portrait is that).

A proposed alternative, especially in the context of copyright enforcement, is the use of watermarks. This technique allows the embedding of a unique identification in the visual mesh of each image. Though theoretically very interesting, this technique faces practical challenges: watermarks are not robust to strong modifications and an agent aware of their presence may intentionally

Conference'04, Month 1–2, 2004, City, State, Country.

remove them. But the most serious shortcoming is the fact they only work for images reproduced from the watermarked original. All the copies made outside this controlled chain (e.g., those made before the adoption of the scheme, or those taken from an analogical source, like a painting on a museum) remain unidentifiable.

Content-based image retrieval (CBIR) systems have the advantage of relying only on the visual contents, needing no annotations, tags or watermarks. While those systems are usually tuned to stimulate generalization, exploration and trial-and-error, they can be specialized to take advantage of the exactness of image identification and to tolerate transformations which completely disrupt the appearance of the image (Figure 1).

Content-based systems use the concept of *descriptor* in order to establish the *similarity* between the images, instead of trying to compare directly the raw visual contents. Those descriptors can be either *global*, if the entire image is described by a single descriptor; or *local*, if different features of the image (regions, edges or points of interest) are described individually, and the image is characterized by a set of descriptors.

Systems based on global descriptors [1] have not shown good performance on the task of image identification, except for very slight transformations. In all comparisons, systems based on local descriptors have performed better [2][3].

Descriptor matching is usually performed through the *k nearest neighbors search* (*kNN search*), also known as *similarity search*. This operation consists in finding, on the descriptor space, the *k* descriptors which are the nearest to the query descriptor. An obvious solution to kNN search is sequential comparison, where we compare each element of the database to the query, and keep the *k* most similar. Unfortunately, this brute-force solution is only feasible for small databases. The alternative is using an indexing scheme, in order to accelerate the search.

3. OUR ARCHITECTURE: LOCAL DESCRIPTORS + PROJECTION KD-FORESTS

In order to establish the similarity between images, a local-descriptor based system has to compare sets of descriptors. This is a potentially complex operation, but most systems adopt a criterion of plain vote count, which has the merit of being simple, and of avoiding the expensive pairwise comparison between the query image and all databases images [4]. Our system is also based on vote count: each individual query descriptor is matched with its most similar descriptors in the entire database (using a simple Euclidian distance). Each matched descriptor votes for the image to which it belongs. Then we simply count how many votes each image received and use this number as a criterion of similarity. The method is robust because the descriptors are many: if some get too distorted or are completely lost, enough will remain to guarantee good results. Even if some are matched incorrectly, giving votes for the wrong images, only a correctly identified image will receive a significant amount of votes. The process can be made even more robust by adding geometric consistency constraints which eliminate most spurious matches (Figure 2).

Unfortunately, the multiplicity of descriptors brings a performance penalty, since potentially hundreds, even thousands of matches must be found in order to identify a single image. The system is only feasible if an efficient indexing scheme accelerates the kNN search used to match the descriptors.

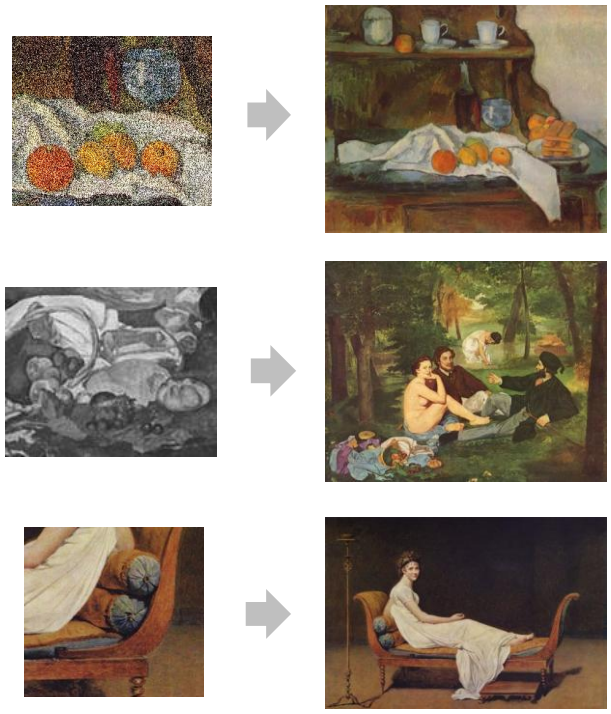

Figure 1: Visual document identification — we should be able to retrieve the original images (right) from the queries (left) even after strong transformations

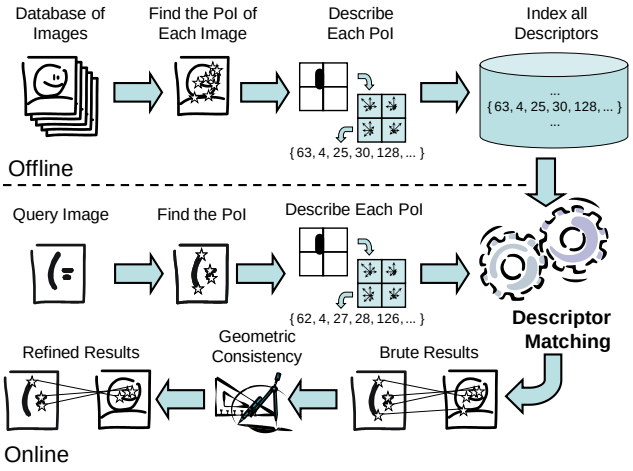

Figure 2: The local-descriptor based image identification system architecture

It is well known, however, that the efficiency of multidimensional indexing depends greatly on the dimensionality of the data. Search time can be made to grow logarithmically with the size of the database, but at the expense of introducing a hidden constant which grows exponentially with the dimensionality. This phenomenon is known as "curse of dimensionality" and expresses the difficulty in partitioning the data or the space efficiently when dimensionality is high [5]. Typically, for over 10 dimensions, it is impossible to perform *exact* kNN search faster than the sequential method, and we are forced to accept approximate solutions, which get progressively coarser as dimensionality grows. Since the

descriptors we want to index have over 100 dimensions, the challenge is considerable.

kNN search is an important topic of research, with many applications outside descriptor matching. Many methods have been proposed to tackle it using techniques as diverse as space partitioning [6], data partitioning [7], clustering [8], projection [9] and hashing [10]. For a comprehensive introduction and up-to-date state-of-the-art of the field the reader is referred to [11].

Fagin et al. proposed a very convenient scheme to perform approximate kNN search using projections of the data [9]. Their algorithm, called MEDRANK, projects all data onto a set of random straight lines. A sorted list is created for each line, with the data sorted accordingly to their position on the line. To answer a query, MEDRANK will project it onto the same straight lines and find, in each sorted lists, the elements nearest to the query projection. At each iteration of the algorithm, the next element nearest to the query (in the line) is chosen and its vote count is incremented. The first element to have a number of votes equivalent to more than half the lines, is considered the *first nearest neighbor*. The search continues until *k* elements have been chosen. The intuitive justification is that if an element is near to the query in the space, it will probably be near to the projections of the query in several lines, and will quickly get many votes (the paper presents a careful probabilistic proof of this intuition).

Unfortunately, MEDRANK performs poorly for large databases in very high dimensionalities. Basically, the basic premise of the algorithm, which is the correlation between proximity in a single dimension and the proximity in the high-dimensional space, becomes weak. But, even worse, the probability that a randomly chosen element will be near to the query in at least one of the dimensions becomes very high, making the algorithm inefficient.

Inspired by MEDRANK, we wanted to devise a method that would perform similarity search in several low-dimensional subspaces and then combine the partial results to obtain the final answer. We wanted, however, to avoid the drawbacks of MEDRANK. The solution we envisaged was to consider more than one dimension at once.

We decided to use several KD-Trees [6] simultaneously, each responsible for a projection of the dataset onto a subspace and called the method *Projection KD-Forest*. The choice of the KD-Tree came mainly from the simplicity of this technique, which allows, when necessary, aggressively optimized implementations.

The idea behind KD-Forests is simple and presents several advantages: a) each tree, being built on a projection of the complete space, has moderate dimensionality — and the less dimensional the trees, the more precise the search; b) in comparison with MEDRANK, there is a greater correlation between proximity in the subspaces and in the original space; c) since we have multiple trees, we can alleviate the boundary effects which plague the KD-Trees, because there is a high probability that we will explore different regions in the different trees. This last factor is very important, since it will allow us to make a radical choice when searching the trees: just the best leaf node (i.e., just the leaf node whose region contains the query) will be explored. This allows us to limit to the bare minimum the number of random accesses to the data.

The construction of the Projection KD-Forests is simple. We determine beforehand how many trees we are going to build and which subspaces of the data space will be associated to each tree. Then, for each tree, we project all data onto its subspace, and create it using the projections.

Naturally, the usual parameters for the component KD-Trees have to be decided (e.g., size of the leaf nodes, criteria for splitting, etc…). We use the interquartile range to choose the splitting dimension and the median as pivot. The size of the leaf node must agree with the number of descriptors which we want to examine in the search, since we examine just one leaf node per tree.

To perform the search, we project the query into the subspaces of the trees and perform a usual KD-Tree search using the projections. Once we determine the leaf node containing the projection on each tree, we examine sequentially all descriptors it contains. We keep the *k* best descriptors found in all trees.

The most important parameter for the construction of the Projection KD-Forests is the number of trees. Once this is set, we may start to worry about how to choose the subspaces. Our approach is much more pragmatic: each leaf access represents a random access to the data, and random accesses on disk are expensive operations. This will prevent any scheme with more than 8–10 trees. At such limited ranges, and for the large dimensionalities we are processing, the rule of thumb is using as many trees as efficiency restrictions will allow. As for the subspace assignment, we opt for a simple partitioning: the dimensions are assigned to the subtrees the most balanced possible way.

4. EXPERIMENTAL RESULTS

For the evaluation of our system, we have used the APM database, composed by SIFT descriptors generated from 1 500 images transformed from 100 originals. The images are old photographs (XIX and first half of XX centuries) from the collection of the Arquivo Público Mineiro, the State Archives of the Brazilian State of Minas Gerais. Each image suffered three rotations, four scale changes, four non-linear photometric changes (gamma corrections), two smoothings and two shearings — a total of 15 transformations. Each transformed image had its SIFT descriptors calculated and aggregate into a database of 2 871 300 descriptors. The queries are the SIFT descriptors calculated from the original images, amounting to 263 968 descriptors. The SIFT (*Scale Invariant Feature Transform*) descriptors [4] are among the most robust under similarity transformations (translations, scale changes and rotations) and photometric distortions. Their dimensionality of 128, however, is very high, creating a challenge to match the descriptors.

The ground truth is the set of the correct nearest neighbors for all query descriptors, according to the Euclidian distance. It was computed using the sequential search, a slow method, but which guarantees exact results. We tested the performance of the descriptor matching by itself, comparing Projection KD-Forests to OMEDRANK (an optimized version of MEDRANK, proposed by Fagin et al. in the same paper [9]) and to the KD-Tree [6]. The results are on Figure 3.

Performance is measured in two axes: efficacy (the capability of returning the correct results) and efficiency (the of doing it fast). To measure the efficacy we use the precision: the fraction of the results which coincides with the ground truth. From the point of view of the user, the most critical efficiency metric is the wall time spent on the search, but using it to compare the methods can

be misleading, since it depends heavily on the machine, the operating system, and even on the current load (concurrent tasks) at the time the experiment is performed. We choose, therefore, to compare the methods by counting, for each method, how many target descriptors were accessed per query descriptor.

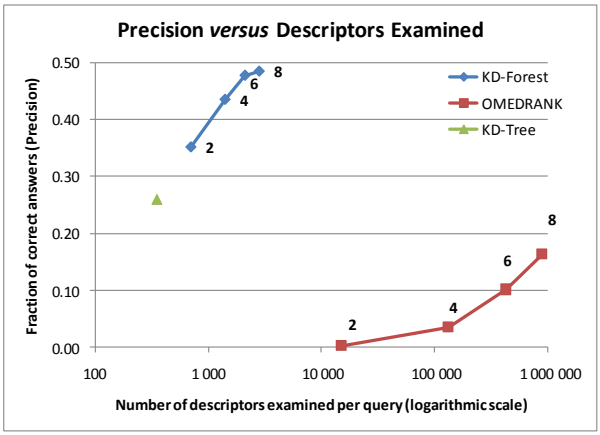

Figure 3: Performance comparison of descriptor matching using Projection KD-Forest, KD-Tree and OMEDRANK. The small number besides the data points indicates the number of trees (KD-Forest) or lines (OMEDRANK)

As we have mentioned in § 3, an important parameter for Projection KD-Forest is the number of trees. This corresponds roughly to the number of lines in MEDRANK (since each additional tree or line means an additional indexing structure and an additional random access to the data). There seems to be a "law of diminishing returns" for this parameter on the KD-Forest, and the sweet spot, on the database analyzed, is around 4 or 6 trees. Note that MEDRANK, even with 8 lines, has worse performance than the KD-Forest with 2 trees. Note also that there is sharp increase in quality from the simple KD-Tree to the 2-tree KD-Forest, indicating the considerable benefit in the multiple subindexes scheme.

We have also tested the performance of the image identification system as whole. We wanted to test the general performance of the system and how much one would gain in speed (and loose in precision) by using the Projection KD-Forest in the place of the exact sequential search. Because of the large amount of local descriptors, the precision of the system is very high. The MAP (mean average precision) of the system using the exact sequential matching is of 0.9626. For the same reason, the impact of the Projection KD-Forest in precision is minimal: though we loose on average 10% of the votes on the correctly matched images, those were so many to begin with, that the MAP of the final results is of 0.9623 — a negligible loss. The gains in efficiency, however, are enormous: running time in the version using the Projection KD-Forest was between 20 and 25 times shorter.

5. CONCLUSION

We have presented an effective and efficient system architecture for image identification, based on local descriptors. At the heart of this architecture, lies Projection KD-Forest a new indexing scheme which, associating multiple moderate-dimensional KD-Trees in a clever way, allows to perform approximate kNN search with a very good compromise between speed and precision.

Using a database issued from a collection of a cultural institution we show that our system has a very good performance, and that the speed-up obtained by Projection KD-Forest (up to 25×) has negligible impact on precision (less than 0.1%).

In our current work we are exploring how the subspace assignment affects the performance of the KD-Forest, and whether there is an optimal way to do this assignment.

6. REFERENCES

[1] Chang, E. Y., Wang, J. Z., Li, C. and Wiederhold, G. 1998. RIME: A Replicated Image Detector for the World-Wide Web. In Multimedia storage and archiving systems III (Boston, MA, USA, November 2–4, 1998). SPIE, Bellingham, USA, pp. 58–67.

[2] Ke, Y., Sukthankar, R. and Huston, L. 2004. An efficient parts-based near-duplicate and sub-image retrieval system. In Proceedings of the 12th ACM international conference on Multimedia (New York, NY, USA, October 10–16, 2004). ACM, New York, NY, USA, pp. 869–876.

[3] Valle, E., Cord, M. and Philipp-Foliguet, S. 2006. Content-Based Retrieval of Images for Cultural Institutions Using Local Descriptors. In Geometric Modeling and Imaging — New Trends (London, UK, July 5–7, 2006). IEEE Computer Society, Washington, DC, USA, pp. 177–182.

[4] Lowe, D. G. 2004. Distinctive Image Features from Scale-Invariant Keypoints International Journal of Computer Vision, 60, 2 (November 2004), pp. 91–110.

[5] Weber, R., Schek, H.-J. and Blott, S. 1998. A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces. In Proceedings of the 24rd International Conference on Very Large Data Bases (New York, NY, USA, August 24–27, 1998). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 194–205.

[6] Friedman, J., Bentley, J. L. and Finkel, R. A. 1976. An Algorithm for Finding Best Matches in Logarithmic Expected Time. Technical Report, Stanford University, Stanford, CA, USA.

[7] Guttman, A. 1984. R-trees: a dynamic index structure for spatial searching. In Proceedings of the 1984 ACM SIGMOD international conference on Management of data (Boston, MA, USA, 1984). ACM, New York, NY, USA, pp. 47–57.

[8] Zhang, T., Ramakrishnan, R. and Livny, M. 1996. BIRCH: an efficient data clustering method for very large databases. ACM SIGMOD Record, 25, 2 (June 1996), pp. 103–114.

[9] Fagin, R., Kumar, R. and Sivakumar, D. 2003. Efficient similarity search and classification via rank aggregation In Proceedings of the 2003 ACM SIGMOD international conference on Management of data (San Diego, California, USA, June 09–12, 2003). ACM, New York, NY, USA.

[10] Indyk, P. and Motwani, R. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In Proceedings of the 13th annual ACM symposium on Theory of computing (Dallas, Texas, USA, May 24–26, 1998). ACM, New York, NY, USA, pp. 604–613.

[11] Samet, H. J. 2006. Foundations of Multidimensional and Metric Data Structures. Morgan Kaufmann, San Francisco, CA, USA.