Combining Local and Global Visual Feature Similarity using a Text Search Engine [*]

Giuseppe Amato, Paolo Bolettieri, Fabrizio Falchi, Claudio Gennaro, Fausto Rabitti
{giuseppe.amato, fabrizio.falchi, paolo.bolettieri, claudio.gennaro, fausto.rabitti}@isti.cnr.it
ISTI - CNR, Pisa, Italy

Abstract

In this paper we propose a novel approach that allows processing image content based queries expressed as arbitrary combinations of local and global visual features, by using a single index realized as an inverted file. The index was implemented on top of the Lucene retrieval engine.

This is particularly useful to allow people to efficiently and interactively check the quality of the retrieval result by exploiting combinations of various features when using content based retrieval systems.

Categories and Subject Descriptors: H.3 [Information Storage and Retrieval]: H.3.3 Information Search and Retrieval;

Keywords: Approximate Similarity Search, Access Methods, Lucene

1 Introduction

Applications of image content based retrieval techniques are increasingly becoming popular for accessing cultural heritage information, as a complement to metadata based search.

In fact, in some cases metadata associated with images do not describe the content with sufficient details to satisfy the user queries, or metadata are completely missing. Images containing reproductions of work of art, contain a lot of implicit information that is not generally described in manually generated metadata.

Furthermore, in some cases users prefer, or are obliged, to use visual queries rather than keyword based queries. Consider for instance a scenario where a user, visiting a tourist site, is in front of a monument (a building or a statue for instance), which he/she does not even know the name, and wants to have information on it. An easy thing to do, in that case would be that of taking a picture, by using a smartphone, and use it as a query to some image content based retrieval engine.

Image content based retrieval is typically obtained by extracting visual features from images and by comparing the visual features rather the original images. Various types of visual features are available, which offer different performance depending on the type of applications and type of queries. In the cultural heritage domain itself effectively processing different types of queries requires also choosing the right type of visual features. Just as an example, identification of a specific work of art, as for instance a specific building or a specific statue, might require the use of local features, as for instance SIFT [17] or SURF [5]. On the other hand, recognizing specific art styles, or type of subjects, can be better obtained by using combinations of global features, as for instance combinations of MPEG-7 descriptors [2, 22]. However, in many cases higher performance can be obtained by combining together also global and local features at the same time.

Providing users (or applications) with the possibility of combining different type of features poses some efficiency problems, especially when dealing with a very large amount of images (as for instance the web itself). Unless a predefined combination of features is imposed beforehand, each available type of feature should be indexed separately from the other, and queries expressed as combinations of features should processed by accessing separately to the various index and by combining the results.

In this paper we propose a novel approach that allows processing image content based queries expressed as arbitrary combinations of local and global visual features, by using a single index realized as an inverted file. More in detail the index was implemented on top of the Lucene text retrieval engine.

The article is organized as follows. Section 2 introduces the main type of visual features. Section 3 presents other works related to this. Section Section 4 presents the proposed approach. Section 5 describes a prototype that we have realized. Section 6 concludes.

[*]This work was partially supported by the ASSETS project, funded by the European Commission.

2 Local and Global Features

As reported in [9], a feature captures a certain visual property of an image, either globally for the entire image or locally for a small group of pixels. The most commonly used features include those reflecting color, texture, shape, and interest (or salient) points in an image. In global extraction, features are computed to capture the overall characteristics of an image. The advantage of global extraction is its high speed for both extracting features and computing similarity.

In the MPEG-7 standard [14] various global features have been defined. In particular: *Scalable Color* is an histogram of the colors of the pixel in an image, when colors are represented in the Hue Saturation Value (HSV) space. *Color Structure* expresses local color structure in an image by use of a structuring element that is comprised of several image samples. *Color Layout* is a compact description of the spatial distribution of colors in an image. *Edge Histogram* descriptor describes edge distribution with a histogram based on local edge distribution in an image, using five types of edges *Homogeneous Texture* descriptor characterizes the properties of the texture in an image. For extracting the MPEG-7 visual descriptors we made use of the MPEG-7 eXperimental Model (XM) Reference Software [15].

Unfortunately global features can be oversensitive to location and hence fail to identify important visual characteristics. To increase the robustness to spatial transformation, local features can be used to describe images. Feature based on local invariants (e.g. corner points or interest points), have been originally presented for stereo matching but are nowadays used also for multimedia information retrieval. Local features are low level descriptions of keypoints in an image. The most famous local feature is probably Scale Invariant Feature Transformation (SIFT) [17]. In SIFT, keypoints are interest points in an image that are invariant to scale and orientation. Each keypoint in an image is associated with one or more orientations, based on local image gradients. Image matching is performed by comparing the description of the keypoints in images.

Another widely used local feature is Speeded Up Robust Features (SURF) [5] which is quite similar to SIFT. SURF detects some keypoints in an image and describes these keypoints using orientation information. However, the SURF definition uses a new method for both detection of keypoints and their description that is much faster still guaranteeing a performance comparable or even better than SIFT. Specifically, keypoint detection relies on a technique based on a approximation of the Hessian Matrix. The descriptor of a keypoint is built considering the distortion of Haar-wavelet responses around the keypoint itself. For both, detecting keypoints and extracting the SURF features, we used the public available noncommercial software developed by the authors [1].

3 Related Work

3.1 Indexes for Global Features

Global features consist of visual descriptors that are obtained by computing statistics of the entire image content. In order to compare two images and assess their similarity, each global feature is associated to a similarity function, or equivalently to a distance function (a dissimilarity function). The assumption is that the more closer the features are, the more similar the images are.

To efficiently process similarity queries based on global features various index structures and search methods were proposed in literature that organize data in such a way that k nearest neighbors search and range search can be executed efficiently. In case of k nearest neighbors search, the objective is to retrieve the k features closest to the query feature. In case of range search, the aim is to retrieve the features whose similarity to the query is above a certain threshold.

An extensive literature surveys on this topic can be found in [25]. Here we introduce some of the most significant techniques classifying them as sequential scan techniques, hash based techniques, and tree based techniques.

The basic technique to efficiently process similarity search queries is to perform a single scan of the entire database, provided that data are stored on contiguous blocks of disk. This technique might be faster than accessing randomly small blocks spread widely on the secondary storage. A relevant approach based on sequential scan, that can be used when features are represented in a vector space, is the VA-File (Vector Approximation File) [24]. It reduces the size of the data set using a coarse approximate representation of data objects. Sequential scan is performed on the reduced data set containing the coarse representation.

Hashing techniques for similarity search, use hash functions that preserve the closeness of data objects. A significant hashing approach applicable to data represented in vector spaces, is the *grid-file* [20]. It partitions the search space symmetrically in all dimensions. Each cell of the obtained grid is associated with data buckets. Another interesting hashing approach for similarity search in metric spaces is the *D-Index* [12]. It is a multilevel hash structure that takes advantage of the idea of the *excluded middle partitioning*, for building an access method based on excluded middle vantage point. An interesting approach, recently proposed, is to represent a data object as a permutation of a set of reference objects ordered according to the their distance from the data object being represented [7]. Similarity between permutations is used in place of similarity between data objects. In [3] it is shown how to use this technique with inverted files.

Tree based access methods relies mainly on a hierarchical decomposition of the data space. One of the first proposed approaches is the *K-d-Trees* [6], that can be used with features represented in vector spaces. Let us suppose that vectors have dim dimensions. In every level of the tree a value (key) is used as discriminator for branching decision of a specific dimension of the corresponding vector space. The root node (level 0) discriminates for the first dimension. Nodes pointed by the root (level one) discriminate for the second dimension. And so on, up to the level $dim - 1$. If more than dim levels are needed, the process starts again from the first dimension.

R-Trees were originally proposed in [13] and can be used when features are represented in vector spaces. The tree is built hierarchically grouping vectors with bounding boxes. Leaf nodes represent bounding boxes including vectors. Internal node represent bounding boxes including other bounding boxes.

M-Trees [8] are secondary storage height balanced access methods that can be used when features are represented in metric spaces. They organize data objects by creating partitions where set of objects are bounded by ball regions. As before, leaf nodes represent ball regions containing features. Internal nodes represent ball regions containing other ball regions.

In [18] authors propose a similar approach of adopting the Lucene text search engine. However, to the best of our knowledge, in this approach the scoring function of Lucene was extended, while in our implementation we exploit the cosine similarity scoring, in this way any full text search engine can be adopted.

3.2 Indexes for Local Features

Comparing two images on the basis of their local features requires matching of their interest points. Typically, as a first step, each interest point in the query image is used as query for a nearest neighbor search between the points of the other image. A different approach recently proposed is to assign each interest point to a visual word of a predefined vocabulary. At search time, two local features assigned to the same visual words will be considered as matching. In the following, we report the most important works related to both the approaches.

Searching for the nearest neighbor given a local features is a very similar problem to the one described in the previous section. While many similarity searches will be performed in order to select the matching points for any local feature describing the image query, each search can be efficiently performed using vector or metric access structures as they would single global features. In fact, a very popular index used for local features matching is *K-d-Trees* [6], discussed also in next section. However, the computer vision community has also developed some specific indexes. The most important is probably the algorithm that applies priority search on hierarchical *k-means* trees proposed in [19].

The bag of visual words model was initially proposed in [21]. The first step to describe images using visual words is to select some visual words creating a vocabulary. The visual vocabulary is typically built grouping local descriptors of the dataset using a clustering algorithm such as *k-means*. The second step is to describe each image using the words of the vocabulary that occur in it. At the end of the process, each image is described as a set of visual words.

In [16], Hamming embedding and weak geometric consistency constraints were proposed to improve bag-of-features efficacy. The proposed approach can be used in combination with traditional inverted files when high accuracy is needed. In [23] the use of term weighting techniques and classical distances from text retrieval in the case of images has been explored. The experiments show that the effectiveness of a given weighting scheme or distance is strongly linked to the dataset used. In the case of large and varied image collections, the noise in descriptor assignation and the need to use larger vocabularies tend to make all distances and weights equivalent.

4 Approximate Content Based Retrieval on Global and Local Features Using Text-Based Indexing Techniques

The technique that we present here leverages on a special representation, based on a text encoding, of local and global features such that any text retrieval engine can be used to perform image similarity search based on local and global visual features. As discussed later, we implemented these ideas on top of the Lucene text retrieval engine.

Two different approaches are used to generate a convenient text encoding for local and global visual features.

For *encoding Local Features* we use the state-of-the-art technique of Bag of Features Approach discussed in Section 3. In particular, we selected 20,000 visual words out of the local descriptors of the dataset using the *k-means*++ clustering method proposed in [4]. Then each image has been described using a subset of the vocabulary. In particular, for each local features in each image, the first nearest neighbor between the visual words is selected and added to the set of words describing the images. Eventually, the images are converted into a textual form using a textual representation for each visual word of the vocabulary.

The approach to *encode Global Features* leverages on the perspective based space transformation developed in [3]. The idea at the basis of this technique is that when two global descriptors (GDs) are very similar, with respect to a given similarity function, they 'see' the 'world around' them in the same way. In the following, we will see that the 'world around' can be encoded as a *surrogate text representation* (STR), which can be managed with an inverted index by means of a standard text-based search. The conversion of the GDs in textual form allows us to employ the search engine off-the-shelf indexing and searching abilities with a little implementation effort.

Let $\mathcal{D}$ be a domain of GDs, and a dataset of $X \in \mathcal{D}$, we define a ranking function $f^k(o, X)$ as a function that takes a GD o as input and returns an ordered subset of k GDs from X in order of decreasing similarity to o. f^k is also known as a *top-k query*.

Let $R \subset \mathcal{D}$ be a set of m reference GDs chosen from $\mathcal{D}$. $f^k(o, R)$ returns the top-k elements of R in order of decreasing similarity with o. Let $p_i^k(o, R)$ the position that $r_i \in R$ assumes in $f^k(o, R)$, assuming $p_i^k(o, R) = k + 1$ when r_i is not present in $f^k(o, R)$. We define the distance function $d(o, q)$ between two GDs o and q as follows:

$$d(o,q) = \sqrt{\sum_{i=1}^{m} \left(p_i^{k_x}(o,R) - p_i^{k_q}(q,R) \right)^2}$$

where k_x and k_q are two integers such that $k_x \geq k_q$. $d(o, q)$ is a generalization of the Spearman Rho Distance with location parameter for the special case $l = k_x = k_q$ [10], which evaluates the distance (or dissimilarity) of two top-k ranked lists. By using this distance we can produce an approximate ranking function $\widetilde{f^k}$. Note, in fact, that d measures the discrepancy between the ordering of the reference descriptors in R, from o and q respectively. These two orderings, can be seen as the representation of the view of the 'world around' o and q. According to [3], the more similar o and q, the more similar their view of the world around, so $\widetilde{f^k}(q, X)$ is an approximation of $f^k(q, X)$.

In order to implement the function $d(o, q)$ and $\widetilde{f^k}(q, X)$ in an efficient way and leveraging on the search functionality offered by a text retrieval engine, we associate each element $r_i \in R$ with a unique alphanumeric keyword τ_i. Then we use the function $t^{\bar{k}}(o)$, defined in the following, to obtain a space-separated concatenation of zero or more repetitions of τ_i words:

$$t^{\bar{k}}(o) = \bigcup_{i=1}^{i} \bigcup_{j=1}^{k+1-p_i^{\bar{k}}(o,R)} \tau_i$$

where, by abuse of notation, we denote the space-separated concatenation of words with the union operator $\bigcup$. The function $t^{\bar{k}}(o)$ returns a text representation of o such that, if the reference descriptor r_i appears in position s in $f^k(o, R)$, then the term τ_i is repeated $(k+1) - s$ times in the text. The function $t^{\bar{k}}(o)$ is used to generate the SRT to be used for both indexing and querying purposes. Specifically we use $\bar{k} = k_x$ for indexing and $\bar{k} = k_q$ for querying.

Most text search engine, including Lucene, use the Vector Space model to represent text. In this representation, a text document is represented as a vector of terms each associated with the number of occurrences of the term in the document. In our case, this means that, if for instance term τ_i corresponding to the reference descriptor r_i ($1 \leq i \leq m$) appears n times, the i-th element of the vector will contain the number n, and whenever τ_i does not appear it will contain 0. Let us refers to these vectors of size m as $\overline{o}^{k_x}$ and $\overline{o}^{k_q}$, which correspond to $t^{k_x}(o)$ and $t^{k_q}(o)$, respectively. The cosine similarity is typically adopted to determine the similarity of the query vector and a vector in the database of the text search engine, and it is defined as:

$$sim_{cos}(o,q) = \frac{\overline{o}^{k_x} * \overline{q}^{k_q}}{\|\overline{o}^{k_x}\| \| \overline{q}^{k_q} \|},$$

where $*$ is the scalar product. sim_{cos} can be used as a function that evaluates the similarity of the two ranked lists similarly as $d(o, q)$ (although it is defined as a distance), and it is possible to prove that the first one is an order reversing monotonic transformation of the second one, and then that they are equivalent for practical aspects [1]. This means that if we use $d(o, q)$ and we take the first k nearest GDs from X (i.e, from the shortest distance to the highest) we obtain exactly the same descriptors in the same order if we use $sim_{cos}(o, q)$ and take the first k similar GDs (i.e., the greater values to the smaller ones).

To summarize, given a dataset of objects $X \subset \mathcal{D}$ and a ranking function f^k, we are able to produce an approximate ranking function $\widetilde{f^k}$ starting from a random selection of m reference descriptors R from $\mathcal{D}$. The approximate ranking function $\widetilde{f^k}$ can be obtained using the text generator $t^k(o)$ and indexing all documents corresponding to $t^{k_x}(o)$ for all $o \in X$ with a standard text-based search engine and using $t^{k_q}(q)$ text for querying the object q.

5 Prototype Description

In order to index and search a collection of images for visual features, the following operations have to be performed, as the overview of Figure 1 depicts:

- *Feature Extraction* – Starting from image files, the system extracts visual descriptors. In particular, we adopted SURF descriptor [5] as local feature and five MPEG-7 descriptors [14] (*CS*, *CL*, *SC*, *HT*, *EH*) as global features.

[1]To be precise, it is possible to prove that $sim_{cos}(o, q)$ is an order reversing monotonic transformation of $d^2(o, q)$. However, since $d(o, q)$ is monotonous this does not affect the ordering. The prove of this statement is given in [11] due to lack of space.

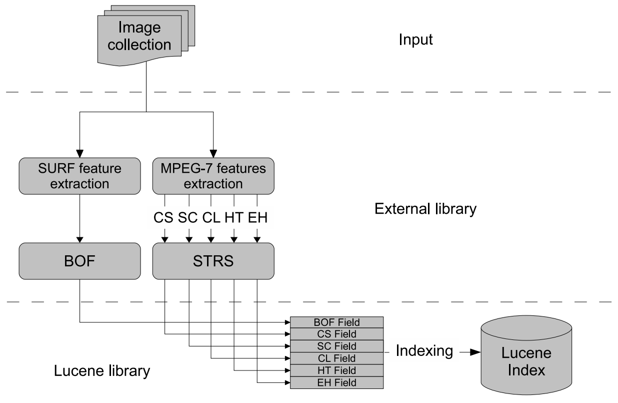

Figure 1. Overview of the indexing architecture.

- *Indexing* – Each images is associated with six text fields, one corresponding to the BOF obtained from the SURF descriptor and five STRs, one for each MPEG-7 descriptor. These segments, which form the basic units on which search is performed, are stored in the Lucene inverted index for fast processing. We use the *WhitespaceAnalyzer* in order to avoid stemming and stop words analysis.
- *Querying* – the similarity computation between a query image and the images in the collection is based on a standard full text facilities of Lucene. In particular, we use the multi field query feature of Lucene, which allow us to combined different descriptors in the same query.

For both testing and demonstration, we developed a web user interface to search among the indexed images. In the following we briefly describe the web user interface which is public available at the address `http://melampo.isti.cnr.it/`
Starting from the home page the user can perform an image similarity search beginning from one of the random selected images, or from an image uploaded through the upload form (the visual features will be automatically extracted by the system). For each visual searching it is possible to set, by some checkboxes placed above the search bar, the combination of visual descriptors to use to perform the query. The available checkboxes allow to combine 5 MPEG-7 descriptors (*CS*, *SC*, *CL*, *HT*, *EH*) with Bag of Features (*BoF*). A typical results page will show the most similar images to the query, and by some links and the visual descriptor checkboxes a user can:

- perform a similarity search with the given result as query by the *similar* link available on top of each query result
- select a new visual descriptors combination
- show the image info clicking the link at the bottom of each image result
- see the next/previous results in order of relevance to the query using the navigation buttons at the bottom of the page
- go back to the home page

In the results page, for each descriptors combination changing, the last query will be automatic performed with the new settings.

6 Conclusions and future work

Good performance of a image retrieval system is typically obtained by the right choice of the visual features. In many cases combinations of various features is needed to obtain the best performance for a query. In this paper we propose a novel approach to index different types of features at the same time and to allow the user to interactively choice the right combination for the current query. The technique was implemented on top a the Lucene retrieval engine.

References

[1] SURF detector. `http://www.vision.ee.ethz.ch/~surf/`. last accessed on 10-Jan-2011.

[2] G. Amato, F. Falchi, C. Gennaro, F. Rabitti, P. Savino, and P. Stanchev. Improving image similarity search effectiveness in a multimedia content management system. In *Proc. of Workshop on Multimedia Information System (MIS)*, pages 139–146, 2004.

[3] G. Amato and P. Savino. Approximate similarity search in metric spaces using inverted files. In *Proceedings of the 3rd international conference on Scalable information systems (InfoScale '08)*, pages 1–10. ICST, 2008.

[4] D. Arthur and S. Vassilvitskii. k-means++: the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '07, pages 1027–1035, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[5] H. Bay, T. Tuytelaars, and L. J. V. Gool. Surf: Speeded up robust features. In *ECCV (1)*, pages 404–417, 2006.

[6] J. Bentley. Multidimensional binary search trees in database applications. *IEEE Transactions on Software Engineering*, 5(4):333–340, 1979.

[7] E. Chavez Gonzalez, K. Figueroa, and G. Navarro. Effective proximity retrieval by ordering permutations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(9):1647–1658, 2008.

[8] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In M. Jarke, M. J. Carey, K. R. Dittrich, F. H. Lochovsky, P. Loucopoulos, and M. A. Jeusfeld, editors, *VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*, pages 426–435. Morgan Kaufmann, 1997.

[9] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40:5:1–5:60, May 2008.

[10] R. Fagin, R. Kumar, and D. Sivakumar. Comparing top-k lists. *SIAM J. of Discrete Math.*, 17(1):134–160, 2003.

[11] C. Gennaro, G. Amato, P. Bolettieri, and P. Savino. An approach to content-based image retrieval based on the lucene search engine library. In M. Lalmas, J. Jose, A. Rauber, F. Sebastiani, and I. Frommholz, editors, *Research and Advanced Technology for Digital Libraries*, volume 6273 of *Lecture Notes in Computer Science*, pages 55–66. Springer Berlin / Heidelberg, 2010.

[12] C. Gennaro, P. Savino, and P. Zezula. Similarity search in metric databases through hashing. In *Proceedings of MIR 2001 - 3rd Intl Workshop on Multimedia Information Retrieval October 5, 2001. Ottawa, Canada*, 2001. In conjunction with ACM Multimedia 2001.

[13] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data, Boston, MA*, pages 47–57, 1984.

[14] ISO/IEC. Information technology - Multimedia content description interfaces, 2003. 15938.

[15] ISO/IEC. Information technology - Multimedia content description interfaces. Part 6: Reference Software, 2003. 15938-6:2003.

[16] H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *Int. J. Comput. Vision*, 87:316–336, May 2010.

[17] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[18] M. Lux and S. A. Chatzichristofis. Lire: lucene image retrieval: an extensible java cbir library. In *MM '08: Proceeding of the 16th ACM international conference on Multimedia*, pages 1085–1088, New York, NY, USA, 2008. ACM.

[19] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP 2009 - Proceedings of the Fourth International Conference on Computer Vision Theory and Applications, Lisboa, Portugal, February 5-8, 2009 - Volume 1*, pages 331–340, 2009.

[20] J. Nievergelt, H. Hinterberger, and K. C. Sevcik. The grid file: An adaptable, symmetric multikey file structure. *TODS*, 9(1):38–71, 1984.

[21] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, pages 1470–, Washington, DC, USA, 2003. IEEE Computer Society.

[22] E. Spyrou, H. Le Borgne, T. Mailis, E. Cooke, Y. Avrithis, and N. O'Connor. Fusing mpeg-7 visual descriptors for image classification. In *Proceedings of the 15th International Conference on Artificial Neural Networks (ICANN'05)*, pages 847–852, 2005.

[23] P. Tirilly, V. Claveau, and P. Gros. Distances and weighting schemes for bag of visual words image retrieval. In *Proceedings of the international conference on Multimedia information retrieval*, MIR '10, pages 323–332, New York, NY, USA, 2010. ACM.

[24] R. Weber, H.-J. Schek, and S. Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In A. Gupta, O. Shmueli, and J. Widom, editors, *VLDB'98, Proceedings of 24rd International Conference on Very Large Data Bases, August 24-27, 1998, New York City, New York, USA*, pages 194–205. Morgan Kaufmann, 1998.

[25] P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search - The Metric Space Approach*, volume 32 of *Advances in Database Systems*. Springer, 2006.