Combining color descriptors for improved codebook model-based image retrieval

Aitor Álvarez Gila[1,2,3], *Guanqun Cao*[1,2,3], *Sheikh Faridul Hasan*[1,2,4], *Yu Hu*[1,4]
[1] *University of Granada; Granada, Spain.* [2] *University of Saint-Etienne; Saint-Etienne, France*
[3] *Gjøvik University College; Gjøvik, Norway.* [4] *University of Eastern Finland; Joensuu, Finland*

Abstract

In this paper we present the design of an image Content-Based Indexing and Retrieval (CBIR) system which, based upon existing implementations of a number of well-known color descriptors, makes use of the bag-of-words or codebook model in order to construct a robust approach to the retrieval of images from a database in a query-by-example context. A new object image database was constructed specifically for this task, in an attempt to challenge the invariance properties of the system under controlled conditions of illumination, point of view and scale. The system permits the combined use of up to two of the different color descriptors considered. The experiments run over a subset of the image database show an improvement of the obtained results under some of the tested combinations, as well as the effect of the variation of the employed codebook size.

Introduction

Recent years have seen handful of attempts to define effective low level visual features to be extracted from images, aiming to accurately describe their content in the context of object and scene recognition or similarity-based retrieval problems. The very first approaches to this problem dealt with global features extracting color, shape or texture information, while latter methods have been more focused on the detection and description of local features at salient points of the images, first based on intensity level information, and then extending this to include color information in several different ways.

Indexing images based on the color histograms was one of those first attempts. Color histogram can be used as a representation of the color content of an image if the color pattern is unique compared with the rest of the considered data set. It is easy to compute and effective in characterizing both the global and local distribution of colors in an image. In addition, it is robust to translation and rotation about the view axis and changes only slowly with the scale, occlusion and viewing angle. However, when an image database contains a large number of images, histogram comparison will saturate the discrimination and drop its distinctiveness[1, 2].

Texture information can be used to describe an image too. Quite many texture representations have been researched and implemented in the application of image recognition, such as the Tamura features [3], Wold decomposition [4], Gabor filters [5], or wavelet transform [6].

Shape information of images or regions has also been widely used in many content-based image retrieval systems, and been part of content description standards like MPEG-7. Going one step further, regions or objects with similar color and texture properties can be easily distinguished by imposing spatial constraints over them; the spatial location of regions (or objects) or the spatial relationship between multiple regions/objects in an image has been demonstrated to be useful for searching. The most widely used representation of spatial relationship is the 2D strings proposed by Chang et al.

Finally, intensity level-based descriptors that focus on local features around interest points, have gained popularity in the last few years. Examples of these are David Lowe's Scale-Invariant Feature Transform (SIFT) [7], or Herbert Bay et al.'s Speeded Up Robust Features [8], which have inspired a number of extensions or revisions to use color information to help their purpose. Some of them will be discussed in the next section, along with a few variations of color histograms and statistical color descriptors, and their performance for similarity-based image retrieval applications will be tested and shown in the *Results* section. They all compete to offer the best trade-off possible between distinctiveness and invariance to different variations in the conditions under which the pictures are taken, such as light source color, light intensity change, point of view, etc.

The present study was conducted in the context of the *Project Contest* course of the 2008-2010 CIMET master [9]. This paper presents the approach followed by the authors' team in the construction of the contest-winning solution.

The rest of the paper is structured as follows: in the next section we introduce the set of color descriptors to be considered in the testing stage; the *Solution design* section deals with the implementation details, while in the *Image database* part some insight on the construction of the testing database is given. In the last sections, the experimental procedure followed for the tests is described, and the obtained results are presented. Finally, we conclude with a brief discussion and point out some possible improvements in the *Conclusions and future work* section.

Diagonal model and color descriptors

[10] presents a structured approach to the performance analysis of a set of color descriptors, including a taxonomy of their different photometric invariance properties. This is done by introducing the *diagonal model*, a diagonal von Kries model-like mapping that can describe illumination changes, and completing it with the inclusion of an offset term, as defined in [11], to account for additional invariance types:

$$f^c = D^{u,c} f^u + O\,, \tag{1}$$

where f^u are the RGB components of the image taken under a test light source, f^u is the same image after the transformation, so it appears as if it was taken under the reference light, $D^{u,c}$ is a diagonal matrix that defines the mapping of colors captured under a given test illuminant to their corresponding colors under the canonical illuminant, and O is a three-component offset term that represents the deviations from the diagonal model. By defining it in this way, the model can be used to represent invariance to light intensity and light color change and shift.

Based on this, a number of color descriptors' invariance characteristics are analytically derived and evaluated. In our solution, we will consider the following subset of them, selected on the basis of their different invariance properties and reported distinctiveness:

- *RGB-SIFT*: it is created by computing SIFT independently for each of the RGB channels, featuring invariance to all the possible factors considered in the diagonal-offset model.
- *rg-SIFT*: conformed by applying SIFT to the *r* and *g* chromaticity components of the normalized RGB color model, its invariance properties limited to light intensity changes.
- *C-SIFT*: uses the C invariant (a sort of normalized opponent color space) to construct a descriptor invariant to light intensity changes, and which, along with rg-SIFT, has shown to be, in practice (but not analytically), also largely invariant to light color changes in some experimental setups [12].
- *OpponentSIFT*: it is based on the computation of SIFT descriptors independently over each of the three channels of the opponent color space (composed of two chromatic channels and one containing intensity information), which are defined as linear combinations of the R, G and B components. The resulting descriptor is invariant to light intensity change and shift.
- *Color Moment Invariants*: a total of 24 moment invariants are created by appropriate combinations of the generalized color moments in order to normalize them against photometric changes. It presents invariance to all the properties included in the model.
- *Hue histogram*: the well-known instability of the hue channel in the HSV color space near the grey axis is overcome by normalizing each sample by its saturation, yielding a light intensity change and shift invariant descriptor.
- *rg-histogram*: based on the computation of the *r* and *g* chromaticity components of the normalized RGB color model; it shows invariance to light intensity changes, shadows and shading.
- *Transformed color histogram*: the independent normalization of each of the RGB channels to achieve distributions with zero mean and $\sigma = 1$ converts an RGB histogram in a light intensity and light color change and shift invariant descriptor.

Solution design

This section describes the design alternative that was implemented in the presented solution.

Overview

Figure 1 shows the complete workflow diagram of the proposed solution. The application is built, making use of Matlab, upon the implementation available in [13] of the different color descriptors commented in the *Diagonal model and color descriptors* section, more or less closely following the pipeline described in [12, 14, 15]. In such papers, the descriptors are used for object and scene recognition; in this case, the same principles are being applied to a content similarity-based image retrieval problem.

The mentioned software (only the binaries are provided) follows the steps shown below when executed:

Point sampling

Detection of the points over which the color descriptors are going to be computed. Different options are available, such as Harris-Laplace detector, or dense point sampling. Due to computational power limitations, the Harris-Laplace detector was the only one used for testing, given the extremely large number of interest points yielded by the dense sampling approach. The output is a collection of circular areas defined by their x and y coordinates, scale, orientation and cornerness.

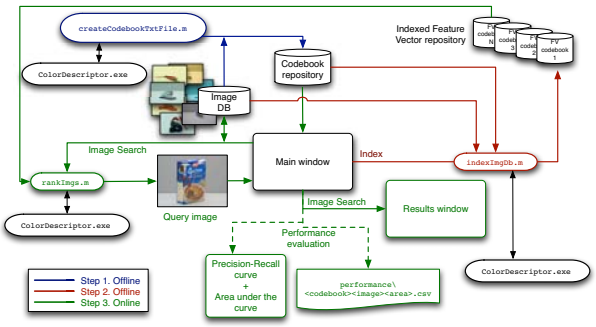

Figure 1. *Design of the solution*

Color descriptor computation

The indicated color descriptor is computed over each of the points (areas) sampled in the previous step, and an output descriptor file is generated, in text or binary format, containing information about the sampled points and about their corresponding feature values. A set of 14 different descriptors is available, from which only a subset of eight of them has been tested, given the characteristics of invariance required by our project specifications: RGB-SIFT, rg-SIFT, C-SIFT, Opponent-SIFT, Color moment invariants (CMI), Hue histogram, rg-histogram and Transformed color histogram (TCH).

Bag of words model

The problem of this approach up to the previous step is the existence of a huge set of feature vectors (one per interest point) in each image (and different for each of them), which makes it difficult to apply similarity measures over them for comparison. Instead, there is the option of, rather than yielding the whole set of feature vectors for every interest point, loading a previously generated codebook against which each feature of the query image will be evaluated and assigned to the closest bin or partition in the Euclidean feature space, as it would happen when constructing a histogram. This approach, which has shown to be effective in recognition and classification [16] problems, is known as the *bag-of-words*, *textons*, or *object-parts* model, in addition to the *codebook* denomination, and basically consists on a vector quantization of the feature vectors of each sampled point in the image against such codebook (the *quantizer*). The result is one single fixed length feature vector representing each image, easily comparable with others. It is also the approach that has been followed in the present implementation.

As shown in Figure 1, there are three main steps in the application execution. The first two are performed offline and require a considerable amount of time and processing power. The last one corresponds to the user querying online (in real time). The following subsections describe the whole process:

Codebook generation

In order to later execute the user query against a codebook, it is first necessary to create it. This is done as follows:

- For each image in the database, we compute the descriptors for every sampled point and store them in one file per image.

- The descriptors corresponding to all the images in the database are loaded into memory. Due to memory constraints, a random sampling of the descriptors of each image is performed, yielding only half of them. This is preferred over the alternative of not using some of the images in the database in the codebook generation.
- The inference of the codewords (each of the entries of the codebook, or representative points of the partitioned feature space) is done by applying k-means clustering over the descriptors in the Euclidean space. This method is employed rather than histogram-like homogeneous bin definition, because, since the descriptor points are not likely to be uniformly distributed across the multidimensional feature space, clustering allows the creation of more compact codebooks with no empty bins. The drawback is the fact that k-means is based in variance, and thus more clusters will be assigned to the more populated areas [17]. But since the most frequent features are not necessarily the most informative ones, this is not the most expressive achievable codebook either, and other alternatives such as radius-based ones could also be explored as a future work.
  As for the implementation, values of k = 500, 1000 and 2000 were selected for comparative purpose (higher values were discarded, again, due to memory constraints), and the computation was performed over around 56000 sampled descriptors from 565 files. Given the extremely high dimensionality of the data, Matlab's native implementation of k-means failed to produce results in acceptable time, so other faster alternatives were considered, tested and made available in the solution's code, which take advantage of Matlab's vectorization properties, or properties of triangle inequalities [18], or even explore a Maximization Expectation-based approach [19] for automatic determination of the optimal value of k fed by the whole set of color descriptors. The triangle inequality-based solution was the one yielding better (faster) results, and was thus the one adopted by default in the testing stage.
- The codebook is generated in the same format as the individual descriptors (but not assigned to any specific sampled point), being each of the values the centroids of the clusters resulting from the previous step.

Indexing

Once we have a codebook generated (a set of codebooks, should we say, since one separate codebook must be used per value of k, detector and descriptor), the next step would consist on running an indexing function. This process will again compute the indicated color descriptors for each image, but will complete the last step too, and thus yield a fixed length feature vector per image, after having assigned all its descriptors to their corresponding *bins* in the codebook. This assignment has traditionally been made as a hard assignment to the bin closest in similarity, but the fact that this model was taken from text retrieval, where, unlike in the case of image features, there is no ambiguity in the assignment, makes of it not the most suitable method. Instead, in [20] a soft assignment approach is proposed, which improves the state of the art, and this is also implemented as an option in the colorDescriptor software. However, this advanced assignment method is left apart as a future parameter to be taken into consideration in future tests.

Image search

Finally, the image search is performed online by the user making use of the GUI constructed with such aim. After selecting the query image, the same codebook-based fixed length feature vector extraction process is performed over it (applying the same codebook as in the indexing), and χ^2 distance (see equation 2) is computed between this feature vector and those corresponding to all the other images in the database, so as to rank them, sort them in ascending order, and retrieve the results.

$$dist_{\chi^2}(\vec{F},\vec{F'}) = \frac{1}{2}\sum_{i=1}^{n}\frac{(\vec{F}_i-\vec{F}'_i)^2}{\vec{F}_i+\vec{F}'_i}\ , \tag{2}$$

where $\vec{F}$, $\vec{F'}$ are the feature vectors and n is their length. The 0/0 cases are taken as 0.

The image search module includes a performance evaluation submodule that optionally computes the precision-recall curve after each query, along with the area below such curve. More details on performance evaluation are given in the *Experimental procedure* section.

Combined descriptors

In addition to the standard use of feature vectors corresponding to one unique color descriptor, the implemented system includes the option of combining up to two different codebooks, which might have been created under a variety of parameter variations (sampling method, codebook size, descriptor, codebook construction method, etc.). In the *Results* section we will show how combining different descriptors (maintaining the rest of the parameters constant) can lead to an improved performance over the simple, single descriptor approach. This makes sense, given the fact that the different presented descriptors are not totally redundant. The combination process is simple: a fixed length feature vector is extracted for the indexed and query images against each of the considered codebooks, and these are concatenated and compared exactly in the same way as in the simple case. The weight of each of the codebooks is controlled by means of a weighting factor, w, selectable by the user, which can take values between 0 and 1, and multiplies all the distance values of the first vector components, while $(1-w)$ does the same for the second feature vector components. Equation 3 shows this approach.

$$\begin{aligned} &dist_{\chi^2}(\vec{F}_c,\vec{F}'_c) = \\ &= \frac{1}{2}\left(w\sum_{i=1}^{n}\frac{(\vec{F}_{c,i}-\vec{F}'_{c,i})^2}{\vec{F}_{c,i}+\vec{F}'_{c,i}} + (1-w)\sum_{i=n+1}^{n+m}\frac{(\vec{F}_{c,i}-\vec{F}'_{c,i})^2}{\vec{F}_{c,i}+\vec{F}'_{c,i}}\right) \end{aligned} \tag{3}$$

where $\vec{F}_c$, $\vec{F}'_c$ are two combined feature vectors, w is the weighting factor, n is the length of the first simple feature vector and m is the length of the second concatenated simple feature vector. The 0/0 cases are taken as 0.

Image database

A necessary step prior to the execution of the performance evaluation experiments was the creation of an image database to be used during the testing process. According to the project requirements, it should contain a reduced number of objects depicted under some variations in the conditions, but against simple backgrounds with little or no clutter (the inclusion of complex scenes was left out of the scope of this project). [21] constitutes a good example of such a kind of database. It was decided to divide the task in two separate subtasks:

- The creation of a **basic image database** that would contain the images corresponding to a given set of single objects from different points of view and scales under controlled uniform illumination. In this case the aim is to get a complete database: each variable (light source color, scale, object, etc.) to modify means a new dimension in a multidimensional matrix. The database should contain an image at each cell of such matrix.
- An extension of such database, which would include single objects or pairs of objects (so as to add clutter) under non-uniform illumination. In this case the idea was to add some additional non-standard variations to some of the photographed objects. Given the practical intractability of a fully complete database as in the previous case, just a few random pictures were taken of selected objects at diverse positions.

Acquisition and processing workflow

The following are some common characteristics of both image databases and the process followed for their creation:

All the images were shot in RAW format using a Canon Powershot G9 camera. In the post-processing stage, the CR2 (RAW) formatted images were all converted into Adobe's digital negative format (.DNG), and further processed to generate six different JPEG compressed outputs for different white balance settings (those corresponding to auto white balance, daylight, flash, fluorescent, shade and tungsten), for each of the originals.

The resolution of the raw images was 4000x3000 pixels. The process of generation of JPEG images for different white balance settings yielded images of 1365x1024 pixels, so each of them was further downsampled with Imagemagick's *mogrify -resize* command, in order to keep both the processing time and database size within reasonable limits, yielding final images with their longest side set to 480 pixels (480x360). This size was decided after studying the composition of several other existing image databases, such as [22, 23].

All the pictures were taken in a GretagMcBeth Spectralight III light cabinet so as to have a uniform simple background and a controlled lighting environment, and they were all taken with similar exposure settings, i.e:

- *Sensibility*: ISO 100
- *Aperture*: f/5.6. Intermediate value to have a wide enough depth of field while maintaining the best quality.
- *Shutter speed*: 1/6s

These were the basic settings. In some cases, no more than -1EV variations were made (changing the shutter speed to 1/13s) over them in order to avoid clipping of the highlight areas in non-metallic objects, since this is one of the assumptions made for the analytical derivations applied to the different color descriptors under the diagonal model [12]. Metallic objects were allowed to show some clipping, given the fact that their almost pure specular reflection directly reflects the light sources.

Basic database

The basic database was designed, with the completeness in mind, to account for the following variations in the viewing conditions:

Lighting set-up

Uniform illumination from the light cabinet.

View point

Two different vertical angles were fixed for shooting (changing the height of the camera): 10°and 30°. In addition, each object was photographed from different points of view, by rotating the object around 3 different axis:

- In the horizontal plane, 8 different angle values were considered always (0°, 45°, 90°, 135°, 180°, 225°, 270°and 315°)
- The number of rotation steps in the other two planes was dependent on the morphology of the object itself and its stability.

Scale

Two different focal lengths were used in order to test the scale invariance of the solution. No object cropping was included. The corresponding focal lengths are 16.8mm (wide angle) and 44.4mm (tele), for a 1/1.7" sensor (7.60 x 5.70 mm, 0.43 cm^2).

Table 1: Characteristics of the different white balance settings

WB Setting	Color Temp.	Tint ($-150 \rightarrow 150$)
Auto	4350K	55
Daylight	5500K	10
Flash	5500K	0
Fluorescent	3800K	21
Shade	7500K	10
Tungsten	2850K	0

Light source color

All the pictures were taken under the Coolwhite illuminant (with a color temperature of about 4300K) provided by the cabinet. In the post-processing stage 6 variants were generated from each of the originals, for the following white balance settings: auto white balance, daylight, flash, fluorescent, shade and tungsten. Table 1 shows the corresponding color temperature and tint values assigned to each of the pre-defined settings, and Figure 2 shows the whole workflow, including the results of the conversion for a sample image.

Figure 2. *Database construction workflow for a sample image*

11 different objects (see Figure 3) have been considered in the basic database: a stapler, 2 different plugs, a mask (with some moving parts), a perfume bottle, a knife (photographed with and without the blades open), a deodorant, a water bottle, a cereal box, a milk tetra-brik, and a plastic toy gun. Both the stapler and the perfume bottle exhibit a metallic surface, the color of which is highly dependent on the color of the light source and the surrounding environment.

Totally, the basic database contains 9306 pictures, divided in six subsets of 1551 photographs, one per white balance setting.

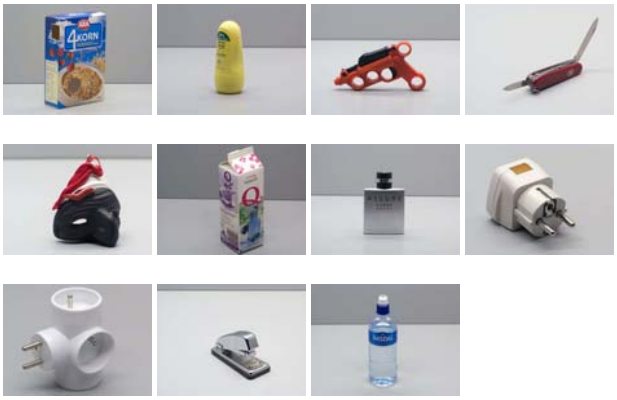

Figure 3. *View of all the included objects with auto white balance setting*

Extended database

The extended database complements the basic one by adding variational elements so that no completeness is aimed, thus allowing a greater flexibility in the shooting. Some of the objects were also photographed under hard light: an external strobe was used in order to produce hard illumination, both with on-camera and off-camera, lateral set up. The goal was to introduce new variations in the set of images, such as casted shadows or reflections that would not be achievable with the uniform illumination. In addition, a series of pictures was taken with object pairs rather than single objects, in order to add some controlled clutter. As in the previous case, six output images were generated from one single digital negative, but in this case the auto white balance setting corresponds to a color temperature of 6550K and a tint value of +6. This yields a total of 732 images in the extended database, 122 per white balance setting.

Combined database

For practical reasons, in order to facilitate the testing procedure a reduced database was created, composed of 563 images randomly sampled from both team's basic and extended databases and other teams' repositories. For homogeneity and performance reasons, all the pictures were resized to a maximum of 480 pixels for the longest side.

Experimental procedure

As mentioned in the *Image search* subsection, a performance evaluation submodule was added to the solution, in order to be able to more efficiently collect experimental results. This module is fed by an input file, which, for every image contained in the combined database, includes an annotation with the group or class to which the depicted object or objects pertain. Therefore, it constitutes a ground truth against which it is possible to check the retrieval results and determine the goodness of the method in use. Note that a retrieved image is considered relevant to this respect if it contains the same object as the one in the query image, independently of any other viewing or capturing condition. This approach, somehow imported from the object recognition and classification field, might not constitute the most accurate relevance feedback gathering solution when applied to a content similarity-based system (although semantically similar, two given images of the same object might greatly differ in appearance, due to the multiple variation factors explained in the *Image database* section), but it constitutes the most straightforward way to automatize the performance evaluation procedure and thus avoid the need of human subjective intervention at this step.

Therefore, in order to adequately characterize the performance of the solution, for each individual query and codebook used, a precision-recall curve is computed and shown, as defined in [24], and the area under the curve (AUC) is computed as in equation 4 for N images in the database.

$$AUC = \sum_{i=1}^{N} P_i \Delta R_i \ , \tag{4}$$

where $\Delta R_i = R_i - R_{i-1}$ and $R_0 = 0$.

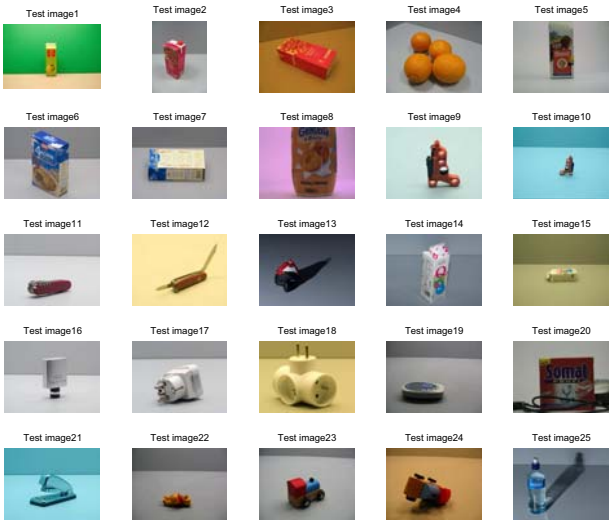

Figure 4. *Subset of 25 test images used in the experiments*

Built upon such functionality, an exhaustive experimental testing procedure was planned, in an attempt to objectively determine which of the different parameter configurations yields best performance (in terms of resulting area) and to observe the effect of a variation of some of these parameters, such as the size of the codebook (k), or the descriptor (or combination of descriptors) used. A small set of 25 query images (see Figure 4), comprising different objects, illuminant and white balance settings, points of view, scales, lighting complexites, and levels of clutter and cropping, were selected from the combined database, and tested against the different created codebooks, including both single color descriptors and combinations of these. Three different values of the parameter k (codebook size, in number of codewords) were considered (k = 500, 1000 and 2000), and, in the combined case, for each pair of different descriptors the weighting factor adopted 5 different values: w = 0.15, 0.3, 0.5, 0.7 and 0.85. The rest of the parameters were kept constant (hard assignment, Harris-Laplace detector, number of descriptors and files used for the creation of the codebook) and their study postponed for future work. The *Results* section shows the performance values obtained in this manner.

Results

The present section shows the results achieved with the different parametrization options detailed in the *Experimental procedure* section. As mentioned, two different experiments were conducted, and the area under the Precision-Recall curve was taken as the only indicator of performance.

Experiment 1: single descriptors

Figure 5 shows the result of averaging across the 25 test images the areas obtained by applying each of the considered

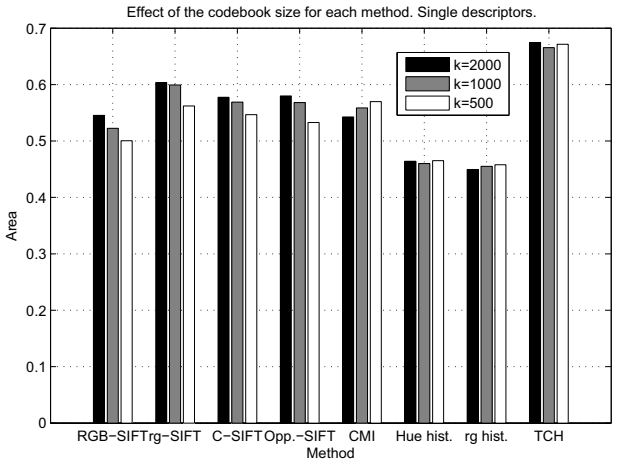

Figure 5. *Average area per individual descriptor and value of k*

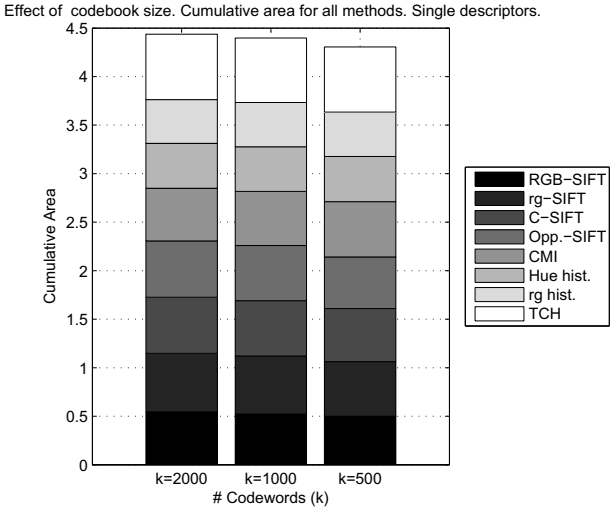

Figure 6. *Effect of the variation of the codebook size, k, in the cumulative area computed for all the individual color descriptors*

single color descriptors, each of them for codebooks of 2000, 1000 and 500 codewords. The Transformed Color Histogram clearly achieves the best results for any of the codebook lengths, followed by rg-SIFT and the rest of the SIFT based descriptors (being RGB-SIFT the worse between them), along with the Color Moment Invariants, while the hue and *rg* histograms achieve considerably worse results. Table 2 shows a relation of the amount of images in the testing set for which each single descriptor performed best, again evidencing the prevalence of the Transformed Color Histogram and yielding results that are consistent with the exposed ones.

Figure 6 focuses on the effect of the codebook size, k, in the results by comparing the sum of the areas yielded by each descriptor. It suggests that, in the range of values of k considered in this study, an increase of such parameter produces a performance increase as well. Concretely, the average improvement of using k = 2000 over 1000 is 0.88%, and over 500 is 3.03%. Higher values of k could not be tested due to memory constraints, but it would be interesting to check at which point this tendency is inverted. If we go back to figure 5, we can see that this conclusion holds for all the methods based on the SIFT descriptors, while CMI shows the opposite tendency, and for the rest of them the results show little variation.

Table 2: Best descriptors by images with best performance

Descriptor	k	# images as best descriptor
TCH	2000	5
TCH	500	5
rg-SIFT	2000	3
rg-SIFT	1000	3
CMI	2000	3
CMI	500	3
Opponent-SIFT	2000	3
TCH	1000	3
rgb-SIFT	2000	2
rgb-SIFT	500	2
rg-SIFT	500	2

Experiment 2: combined descriptors

When combinations of the individual descriptors are considered, the obtained results show a significant performance improvement as compared to the single descriptor case. Table 3 shows a comparison between the average and maximum area means obtained across all the test images for the single and combined approaches, and the unfolded results per value of k are also shown. The data reveal an increase of 9.83% for the averaged area between single and combined descriptors (considering all the possible combinations between the individual descriptors, with values of $w = 0.15, 0.3, 0.5, 0.7, 0.85$), which reduces to 5.52% when considering the improvement achieved between the single and combined descriptors that obtain the maximum performance when averaged across all the images. These are TCH with $k = 2000$ for the individuals and the combination *0.5 x Transformed Color Histogram, 0.5 x rg-SIFT with* $k = 2000$ when considering all the options. The same table also shows how the tendency of getting better performance on average for higher values of k is also maintained in the combined case. The second and third best descriptors, according to the average area yielded, were *0.5 x Transformed Color Histogram, 0.5 x C-SIFT with* $k = 2000$ (average area = 0.7094) and *0.7 x Transformed Color Histogram, 0.3 x rg-SIFT with* $k = 2000$ (average area = 0.7078). All these results point out the remarkable performance of the Transformed Color Histogram for our test database and the considered variations, and how it can be complemented with additional descriptors (best with SIFT-based ones) in order to get it improved.

Table 3: Maximum and average AUC for individual and combined descriptors

Scope	k	Avg.	Max.	Descr. Max.
Comb.	2000	0.6094	0.7116	0.5TCH/0.5rg-SIFT/2000
Comb.	1000	0.6040	0.6984	0.5TCH/0.5rg-SIFT/2000
Comb.	500	0.5907	0.7043	0.5TCH/0.5CMI/500
Comb.	All	0.6014	0.7116	0.5TCH/0.5rg-SIFT/2000
Single	2000	0.5546	0.6744	TCH/2000
Single	1000	0.5498	0.6655	TCH/1000
Single	500	0.5383	0.6715	TCH/500
Single	All	0.5476	0.6744	TCH/2000
ALL	ALL	0.5985	0.7116	0.5TCH/0.5rg-SIFT/2000

If we unfold the data for the different images (figure 7), we can see that the best results are obtained, for our winning descriptor, for images 14, 4 and 8, and the worst are achieved for images

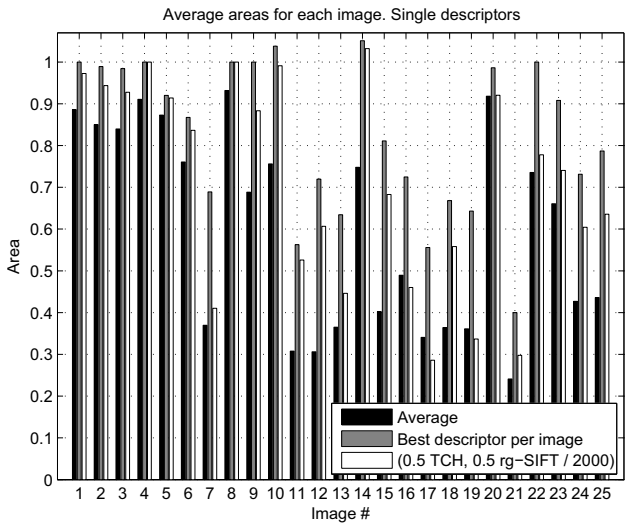

Figure 7. *Areas per image: average for all descriptors, best descriptor for each image and for the best performing descriptor on average: 0.5 TCH, 0.5 rg-SIFT, $k = 2000$*

17, 21 and 19. The results and Precision-Recall curve for such cases are shown in figures 8 and 9, respectively. If we check the retrieved images for the latter, we can suggest that the lack of a large number of salient points (in the case of the remote), the existence of similar but distinct objects of equal neutral color (different plugs, remote) or the metallic nature of the stapler, with the consequent reflections, etc. may be some of the causes that most negatively affect the performance.

Conclusions and future work

In this paper we have presented a working content-based image retrieval system running over a test database created *ad hoc* in order to challenge the invariance and distinctiveness properties of the solution. It was built on the top of an existing implementation of different color descriptors, the performance of which have been tested by applying them as single or combined descriptors over a reduced test-set of images, making use of the bag-of-words or codebook model. The results show that, for the considered options, larger codebooks yield better results in terms of area under the precision-recall curve (for all the SIFT-based descriptors), and that the Transformed Color Histogram exhibits a good trade-off between invariance and distinctiveness, that can be improved by using it in conjunction with other complementary (and preferably SIFT-based) descriptors.

There are, though, a number of design and implementation alternatives that were left out of the presented solution, that might be worth pursuing in future revisions. Some of these are:

- The inclusion of additional point sampling methods, apart from Harris-Laplace detector. The core software allows the use of dense sampling methods and the construction of spatial pyramids that can be combined between them in the same way as we did before.
- The use of soft assignment of the samples to the different codewords (rather than current hard assignment) have shown to correlate well with the ambiguity inherent to the visual world [20].
- An alternative to k-means clustering that does not concentrate most of the centroids where most of the samples are, such as radius-based clustering, might be adequate in this field when constructing a more expresive codebook.
- It might be interesting to reproduce the tests for higher values of k (codebook size), in order to study at which point we get an inversion of the shown tendency to obtain better results for larger codebooks. [12], for example, use codebooks of a constant length of $k = 4096$ in their tests. Memory needs are considerable, though, and the process of codebook creation and indexing is also expensive for large databases.
- Human subjective feedback could also be considered when evaluating the performance, as well as using this information to online and permanently modify the results so as to more closely resemble user expectations. This would imply a change of criteria in the evaluation, though, passing from object-centric assessment to subjective semantic content-based one.

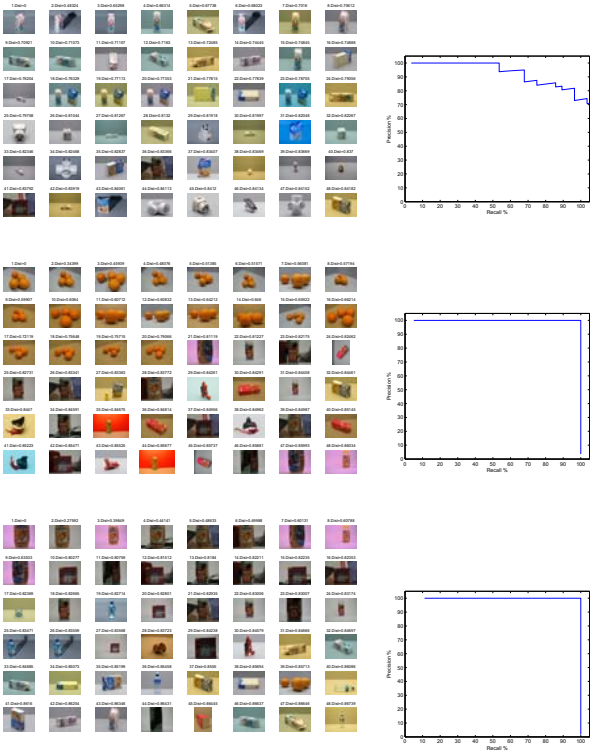

Figure 8. *Results and Precision-Recall curve for the images yielding best results for the winning combination (images 14, 4, and 8. 0.5TCH / 0.5rg-SIFT / 2000)*

Acknowledgments

Aitor Álvarez Gila wishes to thank *Fundación "la Caixa"* for its financial support in the realization of this work.

References

[1] Greg Pass and Ramin Zabih, Comparing images using joint histograms, Multimedia Systems, 7(3):234240 (1999).

[2] Michael J. Swain and Dana H. Ballard, Color indexing, International Journal of Computer Vision, 7:1132 (1991).

[3] Nicu Sebe and Michael S Lew, Texture features for content-based retrieval, In Principles of visual information retrieval, page 5185. Springer-Verlag, London, UK. (2001).

[4] Anil K. Jain and Farshid Farrokhnia, Unsupervised texture segmentation using gabor filters, Pattern Recogn., 24(12):11671186 (1991).

[5] I. Daubechies, The wavelet transform, time-frequency localization

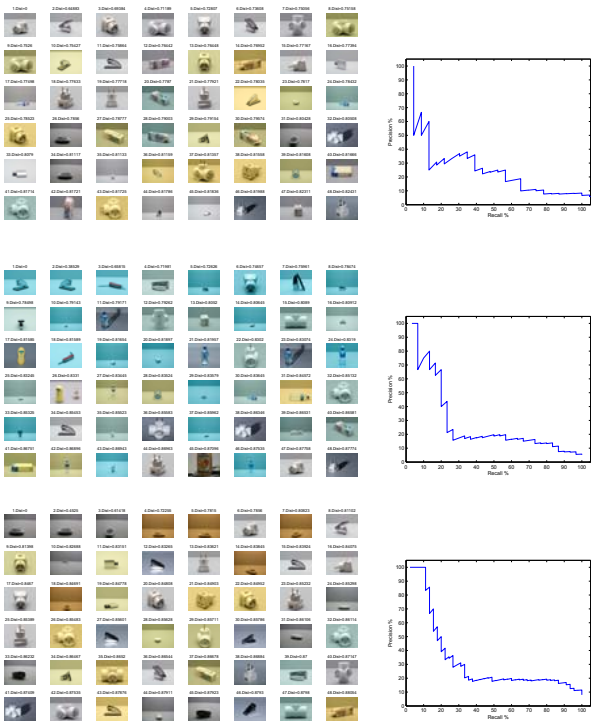

Figure 9. *Results and Precision-Recall curve for the images yielding worst results for the winning combination (images 17, 21 and 19. 0.5TCH / 0.5rg-SIFT / 2000)*

and signal analysis, IEEE transactions on information theory, vol. 36, pg. 9611005 (1990).

[6] Q. Tian, N. Sebe, M. S. Lew, E. Loupias, and T. S. Huang, Image retrieval using wavelet-based salient points, Journal of Electronic Imaging, 10:835 (2001).

[7] D. G Lowe, Object recognition from local scale-invariant features, The Proceedings of the Seventh IEEE International Conference on Computer Vision, volume 2, pages 1150 1157. (1999).

[8] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, Speeded-up robust features (SURF), Computer Vision and Image Understanding, 110(3):346359 (2008).

[9] Master erasmus mundus CIMET (Color in informatics and MEdia technology). http://www.master-erasmusmundus-color.eu/.

[10] K. van de Sande, T. Gevers, and C. Snoek, Evaluation of color descriptors for object and scene recognition, IEEE Conference on Computer Vision and Pattern Recognition, 2008, pg. 18. (2008).

[11] G. Finlayson, S. Hordley, and R. Xu. Convex programming colour constancy with a diagonal-offset model. IEEE International Conference on Image Processing, volume 3. (2005).

[12] K.E.A. van de Sande, T. Gevers, and C. G. M Snoek, Evaluating color descriptors for object and scene recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence, (in press). (2010).

[13] K.E.A. van de Sande, ColorDescriptor software (including color SIFT). http://staff.science.uva.nl/?ksande/research/colordescriptors/readme.

[14] K.E.A. van de Sande, T. Gevers, and C.G.M. Snoek, A comparison of color features for visual concept classification, Proceedings of the 2008 international conference on Content-based image and video retrieval, pg. 141150. (2008).

[15] K.E.A. van de Sande, T. Gevers, and C.G.M. Snoek, Color Descriptors for Object Category Recognition, European Conference on Color in Graphics, Imaging and Vision, pg. 278381. (2008).

[16] J.C. Gemert, J.M. Geusebroek, C.J. Veenman, and A.W. Smeulders, Kernel Codebooks for Scene Categorization, Proceedings of the 10th European Conference on Computer Vision: Part III, pg. 709. (2008).

[17] Jan C. van Gemert, Cees G.M. Snoek, Cor J. Veenman, Arnold W.M. Smeulders, and Jan-Mark Geusebroek, Comparing compact codebooks for visual categorization, Computer Vision and Image Understanding (Special issue on Image and Video Retrieval Evaluation), 114(4):450 462 (2010).

[18] C. Elkan, Using the triangle inequality to accelerate k-means, Machine Learning-International Workshop then Conference-, volume 20, pg 147. (2003).

[19] M. Welling and K. Kurihara, Bayesian K-means as a maximization-expectation algorithm, Sixth SIAM International Conference on Data Mining, volume 22. (2006).

[20] J. C. van Gemert, C. J. Veenman, A. W. M. Smeulders, and J. M. Geusebroek, Visual word ambiguity, IEEE Transactions on Pattern Analysis and Machine Intelligence, (in press). (2010).

[21] J. M. Geusebroek, G. J. Burghouts, and A. W. M. Smeulders, The Amsterdam library of object images, Int. J. Comput. Vis., 61(1):103112 (2005).

[22] M. Grubinger, P. Clough, H. Muller, and T. Deselaers, The IAPR TC-12 benchmark: A new evaluation resource for visual information systems, International Workshop OntoImage, pages 1323. (2006).

[23] IAPR TC-12 benchmark — ImageCLEF - image retrieval in CLEF. http://imageclef.org/photodata.

[24] T. Gevers, Image segmentation and similarity of color-texture objects, IEEE Transactions on Multimedia, 4(4):509516. (2002).

Author Biography

Aitor Álvarez Gila *received his MSc in Electrical Engineering in 2005 from the University of the Basque Country (Spain), where he focused his research on network security. He then worked as a technology consultant for Accenture and Management Solutions in Madrid and London until 2008. He is currently pursuing a master in Color in Informatics and MEdia Technology (CIMET) at the Universities of Granada (Spain), Saint-Etienne (France), Gjøvik University College (Norway), and University of Amsterdam (The Netherlands).*

Guanqun Cao *received his BEng in Electronic and Information Engineering in 2008 from Huazhong University of Science and Technology, P.R. China and he is currently pursuing a master in Color in Informatics and MEdia Technology (CIMET).*

Sheikh Faridul Hasan *received his BS in Computer Science & Information Technology from the Islamic University of Technology, Dhaka, Bangladesh (2006) and his Masters under CIMET Erasmus Mundus program (2010). Now, He is working as an internee (Masters Thesis Student) in THOMSON corporate research center, Rennes, France.*

Yu Hu *received his B.S. in Optoelectronic Information Engineering in 2008 at Tianjin University, P.R. China, and he is currently a master student in Color in Informatics and MEdia Technology (CIMET).*