

# IMPROVING THE PERFORMANCE OF CONTENT-BASED IMAGE RETRIEVAL SYSTEMS WITH COLOR IMAGE PROCESSING TOOLS

Fabio Costa  
Advanced Technology & Strategy (CGISS)  
Motorola  
8000 West Sunrise Blvd.  
Plantation, FL 33322  
email:fabio.costa@motorola.com

Oge Marques and Borko Furht  
Department of Computer Science and Engineering  
Florida Atlantic University  
777 Glades Road  
Boca Raton, FL 33431-0991  
email:{oge,borko}@cse.fau.edu

## ABSTRACT

Most of existing Content-Based Image Retrieval (CBIR) systems operate under the query-by-example (QBE) paradigm, by which an example image is presented to the system and the user queries for images that are similar to the given example. Performance of these systems is highly dependent on the properties of the example image. In many cases, the user has an image that could be used as an example if she could quickly retouch it before submitting the query.

This paper describes the development of a tool, Mirage, that encapsulates several useful color image processing and manipulation operations. Mirage is available as a research prototype and can be used either stand-alone or integrated into a CBIR system, MUSE.

A summary of results of experiments using Mirage to improve the performance of MUSE under the QBE paradigm is presented. These results show that the addition of Mirage to MUSE improves the retrieval performance, both from the point of view of precision as well as recall, without posing significant additional burden to the user.

## KEY WORDS

Content-based image retrieval, Digital image processing

## 1 Introduction

The problem of searching and retrieving an image from a large, distributed, unstructured repository based solely on the image's contents has attracted the attention of researchers from the Image Processing and Computer Vision community over the past few years. Research in the field of Content-Based Image Retrieval (CBIR) is likely to continue to be very active during the first decade of the 21<sup>st</sup> century.

The most widely used paradigm for CBIR is query-by-example (QBE). Under this paradigm, the CBIR sys-

tem extracts some key features from an image presented as an example and then retrieves those images from the database with similar features. The selection of image features and similarity measurements plays an important role on retrieval success. The most widely used features rely on color, texture, and shape[1]. For similarity measurements, different distance metrics (e.g., Euclidean, Manhattan) may be used.

The success rates of a CBIR system operating under the QBE paradigm are highly dependent on the image presented to the system as an example. Under the same feature extraction and similarity calculation algorithms, if the user provides an example whose visual properties are close to the desired image, the system will probably display the intended image ranked among the first best candidates. Conversely, if the user chooses an example that is not quite comparable to the desired target, the system may take longer to return meaningful results. The problem of returning too many false positives becomes even more serious when the databases are very large.

There are several possible ways of minimizing this problem. One of them is to explicitly include the user in the loop, in the form of relevance feedback. Results of our work on CBIR using relevance feedback are reported in [4]. The work described in this paper aims at providing the user a set of simple, yet useful, image processing tools, to help improve the search results in those cases where the example image at hand does not bear too much resemblance with the desired target. In practice this not-so-good query image can occur in several different ways, such as: the object is shown on a different color or an unwanted size, the image contains distracting colors or objects that may deceive the retrieval system, or simply because the image is too bright or dark. In those and many other cases, the auxiliary tools we have developed allow the user to easily edit the example image before pressing the 'Search' button.

We have encapsulated the implemented tools into a fully functional application (Mirage) that can be demonstrated as an independent prototype as well as integrated into the MUSE (MULTImedia SEArch and Retrieval Using Relevance Feedback)[4] system. The expected im-

provement obtained by plugging in these tools into MUSE (working in QBE mode) is twofold:

- improve the average rank of the desired target image, hence reducing the time the user spends browsing the system.
- improve the subjective quality of the best ranked images when performing searching by similarity.

The remainder of this paper is organized as follows: in Section 2 we provide background information on CBIR systems paying special attention to MUSE, which is used as a basis to our work. Section 3 presents the implemented image processing algorithms. In Section 4, we summarize the results of our experiments. Finally, in Section 5 we present our conclusions and directions for future work.

## 2 Background and Related Work

Visual Information Retrieval (VIR) is a relatively new field of research in Computer Science and Engineering. As in conventional information retrieval, the purpose of a VIR system is to retrieve all the images that are relevant to a user query while retrieving as few non-relevant images as possible.

First-generation VIR systems use query by text. Their performance depend on the quality of the metadata, which can very often be incomplete, inaccurate, biased by the user's knowledge, ambiguous, or a combination of these. Second-generation (CB)VIR systems support query by content, where the notion of content, for still images, includes, in increasing level of complexity: *perceptual properties* (e.g., color, shape, texture), *semantic primitives* (abstractions such as objects, roles, and scenes), and *subjective attributes* (such as impressions, emotions and meaning associated to the perceptual properties).

The most widely used paradigm for content-based visual search and retrieval is the Query-by-Example (QBE). Several systems allow the user to specify an image as an example and search for the images that are most similar to it, presented in decreasing order of similarity score. The success of a query-by-example operation is limited by the quality and relative importance of the visual features, the dissimilarity measurement used, the example image presented to the system, and the size of the database, among other factors.

Numerous CBIR systems, both commercial and research, have been developed in recent years. Most of these systems support operation under the QBE paradigm, but very few of them offer the user any options to pre-process the query image before submitting the query.

### 2.1 MUSE

MUSE is a complete, fully functional, image search and retrieval system with relevance feedback capabilities and

it was the result of a Ph.D. dissertation by Marques[3]. MUSE supports three different types of access to the visual contents of an image database:

- **Interactive browsing:** Users can browse the database in three different ways:
  - free browsing: where images are sorted by file name.
  - random browsing: where the images are displayed in random order.
  - cluster browsing: where images are displayed based on the color cluster they have been assigned during the clustering stage.
- **Query by example (QBE):** Users can open an image file and use it as an example of the images(s) they are searching for. Technical users can also select which features and distance measurements should be used in the search. The options for features are color, shape, texture, or any combination of those. The options for distance calculations are:

- Manhattan distance, also known as the  $L_1$  norm:

$$d_M(\mathbf{x}_1, \mathbf{x}_2) = \sum_{i=1}^{i=n} |\mathbf{x}_1[i] - \mathbf{x}_2[i]| \quad (1)$$

- Euclidean distance, also known as the  $L_2$  norm:

$$d_E(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\sum_{i=1}^{i=n} (\mathbf{x}_1[i] - \mathbf{x}_2[i])^2} \quad (2)$$

- Histogram intersection, originally proposed by Swain and Ballard[6]:

$$H(\mathbf{x}_1, \mathbf{x}_2) = \frac{\sum_{i=1}^{i=n} \min(\mathbf{x}_1[i], \mathbf{x}_2[i])}{\sum_{i=1}^{i=n} \mathbf{x}_1[i]} \quad (3)$$

- $d_1$  distance, first proposed by Huang[2], and defined as:

$$d_1(\mathbf{x}_1, \mathbf{x}_2) = \sum_{i=1}^{i=n} \frac{|\mathbf{x}_1[i] - \mathbf{x}_2[i]|}{1 + \mathbf{x}_1[i] + \mathbf{x}_2[i]} \quad (4)$$

- **Relevance Feedback:** Users can search for a target image by providing feedback as to how good or bad the intermediate images are. By means of this feedback the system can define the search patterns the user is interested in and the ones she wants to avoid.

For the purpose of this work, only the QBE mode with its color-based features will be relevant.

### 3 The implemented image processing tools

These are the image processing operations currently implemented in Mirage:

- **Negative:** This function computes the negative of a given image. As each color component on a given pixel is represented by an integer in a range of 0 to 255 (255 is the brightest color), the negative of the image is obtained by subtracting each color component value of the pixel from 255.

- **Brighten:** To brighten an image, i.e., to shift all the pixels values toward the brightest value (255), we implemented the following expression:

$$\text{new pixel} = 255 \cdot \left( \frac{\text{old pixel}}{255} \right)^p, p < 0 \quad (5)$$

- **Darken:** The darken function was implemented using the expression:

$$\text{new pixel} = 255 \cdot \left( \frac{\text{old pixel}}{255} \right)^p, p > 0 \quad (6)$$

- **Blur:** We implemented a simple 3 x 3 smoothing filter, whose convolution kernel is given by:

$$\frac{1}{9} \cdot \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix} \quad (7)$$

- **Sharpen:** To implement a sharpen algorithm we use the following convolution mask:

$$\begin{vmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{vmatrix} \quad (8)$$

- **Gray-scale:** We also implemented an option to convert an image into its gray-scale equivalent.

- **Painting Pencil:** This operation is useful to roughly paint an image with a selectable color.

When the Painting Pencil tool is selected, the mouse cursor will be used to ‘paint’ a pixel of the current color on the current image.

- **Automatic Color Change:** Besides offering the user the possibility to manually paint pixels, we decided to implement a tool that can be used to automatically replace all occurrences in the image of a particular color by any other color. Both original and resulting colors can be selected from a palette using the mouse (Figure 1). A user defined threshold is used to determine how similar two colors must be in order to be considered the same by this operation.

- **Zooming:** This operation is intended to magnify/reduce the size of an image by a pre-selected factor.

- **Crop:** This useful operation allows the user to select a rectangular region on the image to be cropped, typically with the intent of isolating a particular object of interest from the undesirable background.



Figure 1. Automatic color change in Mirage.

## 4 Experiments and Results

This Section presents the results of experiments to which Mirage has been submitted. These experiments are similar to tests performed on MUSE in QBE mode[3] and they allow a fair comparison between a CBIR system with and without the aid of an image pre-processing tool.

### 4.1 Performance Measures

In order to measure the performance of the QBE search with and without Mirage we adopted the same performance measures used to test MUSE[3], namely  $r$ -measure and a high  $p_1$ -measure. They are formally defined next.

Let  $\mathcal{D}$  be an image database and  $\mathcal{Q}$  be the query image. Obtain a permutation of the images in  $\mathcal{D}$  based on  $\mathcal{Q}$ , i.e., assign  $rank(\mathcal{I}) \in [|\mathcal{D}|]$  for each  $\mathcal{I} \in \mathcal{D}$ , using some notion of similarity to  $\mathcal{Q}$ . This problem is usually solved by sorting the images  $\mathcal{I} \in \mathcal{D}$  according to  $|f(\mathcal{I}) - f(\mathcal{Q})|$ , where  $f(\cdot)$  is a function computing feature vectors of images and  $|\cdot|_f$  is some distance metric defined on feature vectors.

Let  $\{\mathcal{Q}_1, \dots, \mathcal{Q}_q\}$  be the set of query images. For a query  $\mathcal{Q}_i$ , let  $\mathcal{I}_i$  be the unique correct answer. We use two measures to evaluate the overall performance of a feature vector-distance metric combination:

1.  $r$ -measure of a method which sums up over all queries, the rank of the correct answer, i.e.,

$$r = \sum_{i=1}^q rank(\mathcal{I}_i) \quad (9)$$

We also use the average  $r$ -measure,  $\bar{r}$  which is the  $r$ -measure divided by the number of queries  $q$ :

$$\bar{r} = r/q \quad (10)$$

2.  $p_1$ -measure of a method which is the sum (over all queries) of the precision at recall equal to 1:

$$p_1 = \sum_{i=1}^q 1/\text{rank}(\mathcal{I}_i) \quad (11)$$

The average  $p_1$ -measure is the  $p_1$ -measure divided by  $q$ :

$$\bar{p}_1 = p_1/q \quad (12)$$

Images ranked at the top contribute more to the  $p_1$ -measure. A method is good if it has a low  $r$ -measure and a high  $p_1$ -measure. From a VIR point of view, a low  $r$ -measure means that in average the desired image will be ranked among the best (good recall) while a high  $p_1$ -measure can be interpreted as “either the desired image is found and ranked among the very top or it is missed by many positions” (good precision).

## 4.2 Tests and Results

For our tests we used a simplified version of MUSE as our Content-Based Image Retrieval (CBIR) in Query-by-Example (QBE) mode system together with Mirage in order to test MUSE’s overall performance when given a pre-processed image as an example. The main reason why MUSE was chosen as the preferred CBIR for our tests was because of the easy code-wise integration with Mirage<sup>1</sup>.

For these tests we used the database created for MUSE experiments that had 11,150 images that includes 11,000 images from Corel Gallery — divided into 110 semantic categories as diverse as “Alaskan wildlife” or “Marble textures” — and 150 other images. It is a very heterogeneous database, which should account for a fair evaluation of the techniques implemented.

In this experiment we performed the same testing and over the same database and 20 image pairs to which MUSE was submitted[3], in order to make a fair and practical comparison on performance.

Our experiments with this database consisted in two steps:

- Without Mirage: we first simulated the performance of a regular QBE system.
- With Mirage: based on the results obtained by the first step, we used the operations available on Mirage to improve the ranking of the correct answers. It was decided to take action towards improving the rank of the correct answer only if the desired (ground truth) image was not ranked among the nine first results without Mirage.

Dissimilarity between images was evaluated by calculating the intersection between their color-based feature vectors. Two different feature vectors were tested separately:

<sup>1</sup>This integration is facilitated due to the fact that both MUSE and Mirage were simultaneously implemented in Java 1.3.

Features	Histogram		Correlogram	
	avg r	avg $p_1$	avg r	avg $p_1$
<b>Without Mirage</b>	742.2	0.396	784.5	0.025
<b>With Mirage</b>	202.8	0.488	273.5	0.033

Table 1. Performance improvement of MUSE when Mirage is utilized.

- Color histogram[5], considering RGB color model and 216 bins.
- Color correlogram[2], considering correlogram distance equal to 8.

For the image pairs used on this experiments we concentrated on the following image processing operations:

- Color change (both automatic as well as using the Painting Pencil): in some cases, because of some color discrepancies between the query and the answer, it was convenient to adjust the colors on the query to better represent the correct answer.
- Crop: some other queries got some distracting objects and/or background that were confusing the searching process.

Table 1 summarizes this experiment. There is a considerable improvement on the average ranking (given by avg  $r$ ) of the desired correct answer for both distance measures used. That means that the user is able to retrieve the image he or she is looking for in fewer steps when browsing through the results. There is also a slight improvement on the avg  $p_1$  values when Mirage is used, meaning the correct answers were ranked closer to the ‘very top’. For histogram distance measures, there is a 72.68% improvement on the avg  $r$  value and 23.23% improvement on the avg  $p_1$ ; for correlogram intersection the improvement of the avg  $r$  value is 64.14% and 32.00% for the avg  $p_1$ .

Performance improvement was not even more noticeable because a particular pair (Figure 2) performed very badly even with some query pre-processing. This is due to the fact that the correct answer is a cropped brighter version of the original query image. The cropped issue was solved by cropping the original query but the bright difference on the answer was sufficient to mislead the color constancy algorithm used.

For all the other pairs where the example image was processed by Mirage, the ranking of the correct answer was greatly improved. Figure 3 and Figure 4 show examples where the ‘crop’ tool helped achieve better ranking results. Figure 5 shows an image pair where retouching using color manipulation tools led to a better result.



Figure 2. Worst pair overall for QBE search. See color plate.



Figure 3. (a) Correct answer (ground truth); (b) Example image without Mirage; (c) Example image with Mirage. The use of Mirage has improved the rank of the desired image from 4442 to 3 (out of 11,150).

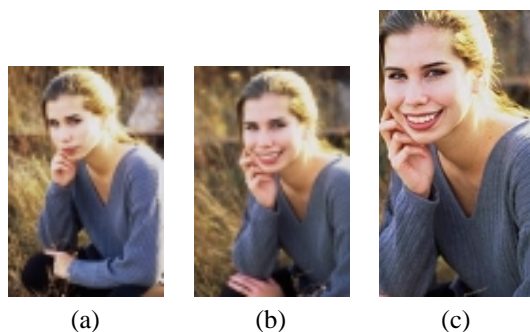


Figure 4. (a) Correct answer (ground truth); (b) Example image without Mirage; (c) Example image with Mirage. The use of Mirage has improved the rank of the desired image from 104 to 1 (out of 11,150).



Figure 5. (a) Correct answer (ground truth); (b) Example image without Mirage; (c) Example image with Mirage. The use of Mirage has improved the rank of the desired image from 261 to 2 (out of 11,150).

## 5 Conclusions and Future Work

The main contribution of this work was to demonstrate the usefulness of color image processing tools and how they can improve the performance of Content-Based Image Retrieval (CBIR) systems.

Our tests show that with those operations we could improve the performance of an existing CBIR system (MUSE) by about 72% when histogram intersection is used as distance measure, and about 65% when correlogram intersection is used.

Ongoing and future work include:

- Extension of the image cropping procedure to allow regions-of-interest of complex shapes.
- Extension of the existing 1-step Undo feature to several steps.
- Addition of object location and recognition capabilities to MUSE, using a variant of the histogram backprojection[6] and correlogram backprojection[2] techniques proposed in the literature.

## References

- [1] G. Duffing. An alternative image retrieval system based on visual and thematic corpus organisation. In *Proceedings of the International Conference on Multimedia Computing and Systems*, volume 2, pages 189–193, 1999.
- [2] J. Huang. *Color-Spatial Image Indexing and Applications*. PhD thesis, Cornell University, August 1998.
- [3] O. Marques. *Content-Based Image Retrieval Using Relevance Feedback*. PhD thesis, Florida Atlantic University, August 2001.
- [4] O. Marques and B. Furht. MUSE: A content-based image search and retrieval system using relevance feedback. *International Journal of Multimedia Tools and Applications*, 17(1), May 2002.

- [5] M. Swain and D. Ballard. Indexing via color histograms. In *Proc. Third International Conference on Computer Vision*, pages 390–393, 1990.
- [6] M. Swain and D. Ballard. Color indexing. *International Journal of Computer Vision*, 7:11–32, 1991.