Surfimage: a Flexible Content-Based Image Retrieval System

Chahab Nastar, Matthias Mitschke, Christophe Meilhac, Nozha Boujemaa

INRIA
BP 105
F-78153 Le Chesnay, France
Chahab.Nastar@inria.fr

Abstract

Although powerful image representations have been proposed for content-based image retrieval, most of the current systems are "rigid", i.e. they retrieve a fixed set of images as response to a given query and an image feature. We introduce Surfimage*, a user-friendly, generic and flexible content-based image retrieval system.* Surfimage *uses the query-by-example approach for retrieving images and integrates advanced features such as image signature combination, classification, multiple queries and query refinement. The classic and advanced features of* Surfimage *are detailed in the paper.* Surfimage *has been extensively tested on dozens of databases and produced excellent retrieval results; a sample of retrieval results is presented here.*

1 Introduction

Multimedia documents are dominated by images with respect to bandwidth and complexity. Thus, retrieving images based on their content, commonly called *content-based image retrieval*, has become a major issue in multimedia retrieval. For designing an effective image retrieval system, we find it convenient to divide image databases in two categories.

The first category concerns databases for which a *ground truth* is available. These databases are generally homogeneous – they contain images of the same object class. For querying these databases, the notion of *perceptual similarity* between two images is implicitly obvious (e.g. find more images of this person) or explicitly defined by an expert user (e.g. find more images presenting the same type of tumor). When indexing the database, the designer will consider these ground truths and tune the models or range of parameters accordingly, maximizing the system *efficiency*. The response to the queries will then be "rigid", i.e. an example image will lead to the same set of retrieved images.

The second category includes databases with heterogeneous images where *no ground truth* is available or obvious. Examples include stock photography and the World Wide Web. The user should be assumed to be an average user (not an expert), and the notion of perceptual similarity is *subjective*. It highly depends on the application, the context, and the user. In particular, different users may have dramatically different goals when querying such a database. The goal here is to maximize the system *flexibility*, adapting to and learning from each user in order to satisfy their goal [10, 4, 1, 11].

Unlike most of the earlier systems (e.g. [2]), Surfimage is a flexible image retrieval system that can deal with both categories of image databases. For a specialized database, the ground truth is well-defined and a dedicated image signature will lead to optimal performance [8].

In order to deal with generic databases, Surfimage includes a novel, generic and efficient relevance feedback technique which enables the user to refine their query by specifying over time a set of relevant and a set of non-relevant images. Our idea is to estimate the distribution of relevant images from the examples provided by the user and to simultaneously minimize the probability of retrieving non-relevant images [7]. We can iteratively refine our density estimation through time as the user specifies more positive and negative examples.

Surfimage also offers the possibility of combining image features [7]. Using combined rather than individual features is especially efficient for generic image databases, for which no single feature is outstanding. For highly heterogeneous databases, the combination of features can essentially lead to the classification of the different image types, as shown in this paper.

The various features of Surfimage enabling both efficiency and flexibility are presented in this paper, and several retrieval results in various cases are presented.

2 Surfimage: classic features

For the algorithm designer, the hardest step is image indexing, i.e. the computation of image signatures. This is typically an off-line processing. For doing this, genericity or specificity of the signature and conservation (or non-conservation) of the spatial arrangement of the image are key issues. The corresponding images signature categories with examples are summarized in table 2.

Surfimage offers a large selection of signatures in all the different categories mentioned in table 1. Among the signatures we can mention:

- Low-level signatures capturing color, shape and texture. Examples include color, orientation and texture histograms, Cooccurrence, Fourier and Wavelet transforms.

ACM Multimedia'98, Bristol, UK
© 1998 ACM 1-58113-036-8/98/0008 $5.00

arrangement	generic sig.	specific sig.
encoded	*Fourier spectrum*	*eigenfaces*
lost	*color histogram*	*distance ratios*

Table 1: Image signature categories & examples

- High-level signatures which are derivated from a complex modeling of image content and sometimes a statistical analysis of the database. Examples include eigenimages [13, 5], flexible images [8] or image shape spectrum [6].

The similarity metric is usually defined via a distance measure which will be used for nearest neighbor match in feature space. Various similarity metrics are implemented in Surfimage: the Minkowski L_p distances, the Cosine metric, the Hellinger metric, and M-estimators for outlier rejection. In our experience, the city-block distance L_1 is convenient since it is fast to compute and well-suited to signatures which are histograms.

Surfimage uses the query-by-example approach for querying. In the classic querying scheme, the user (i) loads a database, (ii) selects a signature, (iii) chooses a similarity metric, and clicks on a query image from the database to find more similar images with respect to the chosen signature and metric. Experiments are shown in section 4.

3 Surfimage: advanced features

The specificity of Surfimage is its advanced features which makes it uniquely flexible among image retrieval systems. Advanced features include signature combination and relevance feedback based on density estimation. They are detailed hereafter.

3.1 Signature combination

Combination of different features has been a recent focus of image retrieval [3, 1, 4]. But how do we combine "apples and oranges", i.e. features that have different number of components, different scales etc.?

Simple rescaling of the features is not suitable since it would alter the discrimination properties of each feature. A weighted linear combination of feature vectors is another possible method, and the weights have to be estimated (learned) after various experimentations with the database [3, 1]. We have experimented with two combination methods. Under Gaussian assumption, *the normalized linear combination* method uses the estimated mean μ_i and standard deviation σ_i of the distance measure d for each feature i, providing the normalized distance:

$$d'(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) = \frac{d(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) - (\mu_i - 3\sigma_i)}{6\,\sigma_i}. \qquad (1)$$

where $\mathbf{x}^{(i)}$ and $\mathbf{y}^{(i)}$ are the signature vectors of images X and Y within feature i. The new distance measure d' will essentially have its values in [0...1] and can be linearly combined with the normalized distance measures of the other features (see also [9]).

The voting procedure operates as follows: in response to a query, each feature retrieves images and grades them by increasing order of distance. The ranks of the retrieved images within each feature can then be combined by a weighted sum (e.g. averaging) to output the retrieved images according to the combination of features. Alternative methods include the use of the median rank and the gymnastics-rule (i.e. ignoring the worst and the best ranks of an image) for robustness.

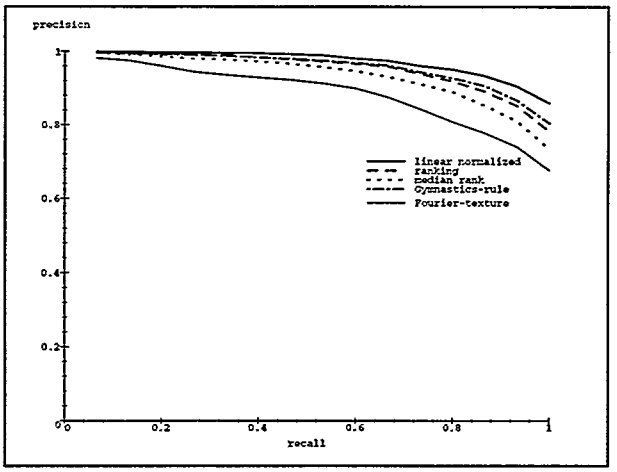

Figure 1: Precision-recall graph for the MIT *Vistex* database. Note that the Fourier texture, the most performant single feature that we have computed, is not as good as any of the combined features.

For evaluating the combinations, we experiment on benchmark databases with ground truth, and draw the precision-recall graphs, with:

$$\text{Precision} = \frac{|\text{retrieved relevant images}|}{|\text{retrieved images}|} \qquad (2)$$

$$\text{Recall} = \frac{|\text{retrieved relevant images}|}{|\text{relevant images}|} \qquad (3)$$

where $|D|$ denotes the number of elements in D.

For most applications, it is impossible to maximize precision and recall simultaneously, but these values should ideally be as large as possible. Figure 1 shows the precision-recall graphs for a database with ground-truth adapted from the MIT *Vistex* database, consisting of 384 homogeneously textured images as used in [11]. From this figure, it is obvious that *any version of feature combination is more performant than a single feature*, as noted elsewhere [10]. We have quantitatively verified the better performance of the combination on a number of other benchmark public-domain databases with ground truth (Columbia database of 3D objects, ORL face database etc.).

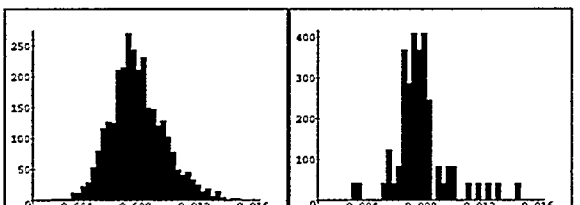

Figure 2: Distribution of a specific feature component. (a) over database (b) over a set of relevant images.

3.2 Relevance feedback

3.2.1 Motivations

We now can combine several features for a single query. Our objective in this section is to enable query refinement by user interaction. Before describing our method, consider the following example that will clarify our motivations. Suppose the user is a customer willing to purchase a shirt. In most cases they have some ideas about the features connected with shirts, like color, texture, size, quality, price. Their idea about these features can either be very precise ("I want a cotton shirt"), other features can vary in a range ("I am ready to spend 30 to 40 dollars on it"), other features might be unimportant ("I don't care about the color"). The salesman's job is to *guess* these different distributions to come up with the ideal shirt.

In statistical terms, each feature has a distribution. In case of parametric distributions like Gaussians, the salesman needs to guess the mean and the standard deviation. In particular, the standard deviation can be zero – or rather narrow – providing a constrained feature, (e.g. cotton), average (e.g. price range), or infinite (e.g. color is unimportant).

What we want to do is similar to the above example. A simple idea is to try to estimate the feature densities of relevant images based on positive examples provided by the user (non-relevant images are not taken into account at this point). More precisely, let X an image, and $\mathbf{x}$ its d-dimensional signature vector: $\mathbf{x} = [x_1 ... x_i ... x_d]$. For a given query, our goal is to estimate the density:

$$P_{\theta}(\mathbf{x}|X \textit{ is relevant}) \tag{4}$$

where θ are the distribution parameters. If we make the simplifying assumption that the x_i are independent[1], then we have:

$$P_{\theta}(\mathbf{x}|X \textit{is relevant}) = \prod_{i=1}^{d} P_{\theta_i}(x_i|X \textit{is relevant}) \tag{5}$$

Our goal is now to estimate the d distributions P_{θ_i} based on the user's examples. Note that each of these distributions concerns an individual feature component.

In this framework, we have to estimate the distribution of the individual feature components for the relevant images from the images labeled 'relevant' by the user, which is only an approximation of the whole set of images relevant to the query. This distribution will be updated through time and user interactions. A new query is then determined by randomly drawing individual feature components according to the corresponding estimated distributions. Based on this query, more relevant images are likely to be retrieved.

More precisely, we assume that the densities of the feature components of the relevant images are Gaussian. This is motivated by looking at the distributions of feature components over the database, and over a defined set of relevant images. Consider for instance the benchmark Columbia database [5]. It contains 1440 images of 20 different objects with a wide variety of properties ranging from uniform reflectance and simple shapes to complex textural properties. The database contains 72 images per object, taken at 5 degrees incremented in pose. Figure 2 shows the distribution of an individual feature component over the entire database and over the set of 72 images of a specific object[2]. We observe that the distributions can be approximated by Gaussians (the parameters are the mean and standard deviation: $\theta_i = (\mu_i, \sigma_i)$).

[1] they are generally not independent, still the independence of index terms is a common assumption in particular in information retrieval theory [12]

3.2.2 Detailed algorithm

In practice, we wish to integrate both the positive (relevant) and the negative (non-relevant) examples of the user. Our idea is to estimate the distribution of relevant images for each feature component (described by θ_i) from the examples provided by the user and to simultaneously minimize the probability of retrieving non-relevant images. Note that the distribution of the non-relevant images cannot be easily modeled since by definition, non-relevant images tend to be multimodal. Nevertheless, we take the user-provided non-relevant examples into account, as explained hereafter.

For the relevant examples, we define the "3σ-surrounding" $V(i)$ as[3]:

$$V_i = [\mu_i - 3\sigma_i, \mu_i + 3\sigma_i] \tag{6}$$

For each feature component i, our goal is to estimate the parameters $\theta_i = (\mu_i, \sigma_i)$ such that more relevant images and less non-relevant images are retrieved.

Hence, the method is different from the maximum likelihood estimation of the relevant images, since it also integrates the non-relevant images. It is detailed in the following pseudo-code, which determines distribution parameters for each feature component s_i, given the sets of values for the images labeled 'relevant' ($\mathcal{D}'_{rel_i}$) and 'non-relevant' ($\mathcal{D}'_{non_i}$) by the user.

1. Let $V_i^{(n)} = [\mu_i^{(n)} - 3\sigma_i^{(n)}, \mu_i^{(n)} + 3\sigma_i^{(n)}]$ where n is the iteration. Initialize $n = 0$ and distribution parameters:

$$\begin{aligned} \mu_i^{(0)} &= \text{mean}\left\{ s_i \in \mathcal{D}'_{rel_i} \right\} \\ \sigma_i^{(0)} &= \text{argmax}_{\sigma} \left| \left\{ s_i \in \mathcal{D}'_{rel_i} \cup \mathcal{D}'_{non_i} | s_i \in V_i^{(0)} \right\} \right| \end{aligned}$$

2. Determine error term $e_i^{(n)}$ based on the percentages of "relevant values not covered" $p_{rel}^{(n)}$ and "non-relevant values covered" $p_{non}^{(n)}$ by $V_i^{(n)}$:

$$\begin{aligned} p_{rel}^{(n)} &= \frac{1}{\left|\mathcal{D}'_{rel_i}\right|} \cdot \left| \left\{ s_i \in \mathcal{D}'_{rel_i} | s_i \notin V_i^{(n)} \right\} \right| \\ p_{non}^{(n)} &= \frac{1}{\left|\mathcal{D}'_{non_i}\right|} \cdot \left| \left\{ s_i \in \mathcal{D}'_{non_i} | s_i \in V_i^{(n)} \right\} \right| \\ e_i^{(n)} &= p_{rel}^{(n)} + p_{non}^{(n)} \end{aligned}$$

3. If $p_{rel}^{(n)}$ exceeds a predefined p_{max}, go to step 5, otherwise update distribution parameters:

$$\begin{aligned} \mu_i^{(n+1)} &= \text{mean}\left\{ s_i \in \mathcal{D}'_{rel_i} | s_i \in V^{(n)} \right\} \\ \sigma_i^{(n+1)} &= \lambda \cdot \sigma_i^{(n)} \qquad \text{with } 0 < \lambda < 1 \end{aligned}$$

4. $n = n + 1$. Go to step 2.

5. Determine distribution parameters with minimum error measure: (in case of an ambiguous minimum, decide for the one with the maximal σ)

$$\begin{aligned} k^* &= \text{argmin}_k \left\{ e_i^{(k)} \right\} \\ \mu_i &= \mu_i^{(k^*)} \\ \sigma_i &= \sigma_i^{(k^*)} \end{aligned}$$

6. Draw random variables from from $\mathcal{N}(\mu_i^{(k^*)}, \sigma_i^{(k^*)})$ with $i = 1...d$. Use the obtained feature vector as a new query to retrieve more relevant images.

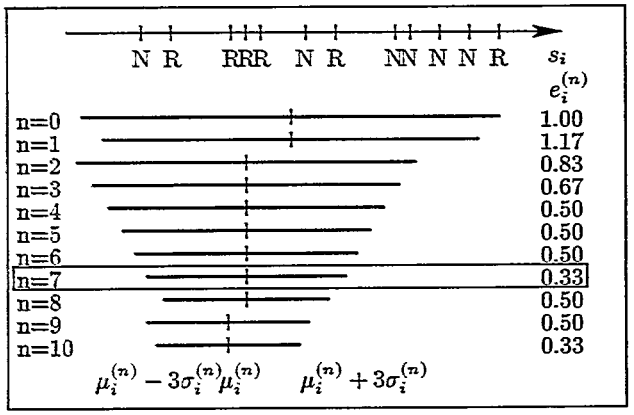

Figure 3: An example to demonstrate the algorithm for estimating μ_i and σ_i

A few comments on the algorithm: we start the procedure with a large $\sigma_i^{(0)}$ covering all user-provided examples – both relevant and non-relevant – thus $p_{rel}^{(0)} = 0$ and $p_{non}^{(0)} = 1$. Over the iterations, we slowly decrease $\sigma_i^{(n)}$ (e.g. $\lambda = 0.9$) until the error $e_i^{(n)}$ is minimized. We ensure that the procedure does not exclude too many relevant images (e.g. $p_{max} = 1/3$). A key issue is that our estimation of the distribution is based on a few data points (less than a dozen) and therefore it is not reliably representative of the true distribution of all the images in the database relevant to the query. Therefore we introduce a randomization (step 6) that will retrieve more "varied" images than the classic maximum likelihood estimator (which will basically retrieve the closest images to μ). Note that the randomization of step 6 can be improved by generating (i.e. randomly drawing) m new query vectors and retrieving the m best matches to each of the new queries.

One of the nice properties of the above algorithm is that, in cases where relevant and non-relevant images are all mixed up within a feature component, the estimated distribution will tend to be flat (large σ_i). This means that the corresponding feature component is not discriminant for the query.

An example illustrating the density estimation part of the algorithm is presented on figure 3. Six 'relevant' (R) and six 'non-relevant' (N) images have been labeled by the user. Their values for a feature component s_i are visualized in the figure. The estimated mean value ($\mu_i^{(n)}$) and 3σ-surrounding ($V_i^{(n)}$) are visualized together with the corresponding error value $e_i^{(n)}$ for the first ten loops of the algorithm. For $n > 10$ the error value does not decrease anymore, because all 'non-relevant' values are already excluded from the 3σ-surrounding, and for smaller values of $\sigma_i^{(n)}$ only more 'relevant' values would be excluded. In this case, the algorithm determines two minima for the error function. In order to favor the parameters such that most relevant values are inside the 3σ-surrounding of the distribution, the algorithm decides for the parameter set $\theta_i = (\mu_i^{(n)}, \sigma_i^{(n)})$ corresponding to $n = 7$.

We now evaluate the performance of our technique on databases with unique ground truth: the Vistex and the Columbia database, using standard image features whose description is not the scope of this paper (see for instance [3, 5]). We measure the precision over a predefined number of retrieved images (e.g. 15). The performance of the standard relevance feedback approach adapted from [12, 11] is also presented. The results are summarized in table 2, illustrating than the proposed technique is better than the standard method.

[2]In this example, the feature vector is the edge orientation histogram

[3]V_i covers 99.7% of the data in case of Gaussian distributions.

	no *rf*	1 *rf*	2 *rf*
Columbia			
std approach	0.752	0.764	0.778
our approach	0.752	0.811	0.830
Vistex			
std approach	0.926	0.927	0.930
our approach	0.926	0.956	0.974

Table 2: Performance of the proposed relevance feedback technique after one (1 *rf*) and 2 sets (2 *rf*) of user interactions. The precision keeps getting better than with no user interaction (no *rf*)

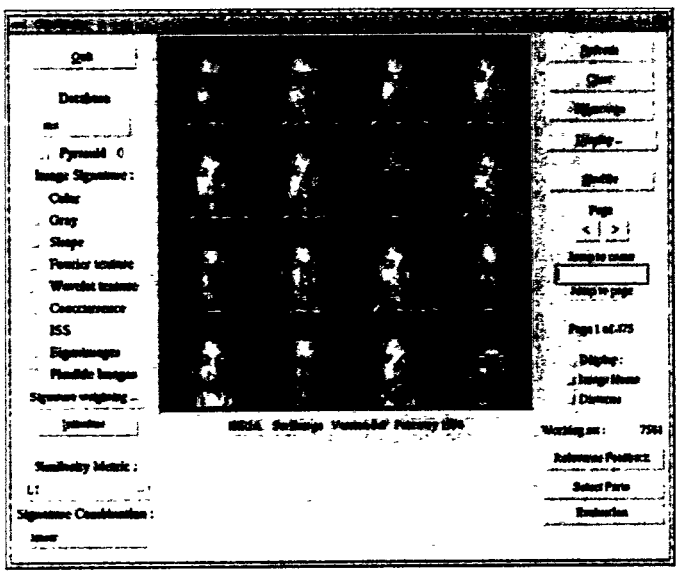

Figure 4: Single specific features: retrieval of the top left face in a database of 7562 images using *flexible images* [8]. Retrieved images are from top left to bottom right in order of best match. Note that the second and third best matches are faces of the query person without sunglasses.

4 Retrieval Results

Figure 4 shows an example of querying with Surfimage on the MIT face database (7652 images). We have developed a specific, arrangement-preserving signature for this database called *flexible images* [8]. With the MIT face database, Surfimage produced a recognition accuracy of 97% based on a nearest neighbor rule. This corresponds to only six mistakes in matching views of 200 people randomly chosen in the database of 7,562 images. This is significantly better than had previously been reported for this dataset [13].

Combining image features (up to a dozen image signatures) increases the discrimination power of the index. We have measured 100% recognition rate using combined features on the Columbia database. We illustrate combined features on our highly heterogenous homebrew *bigdatabase*,

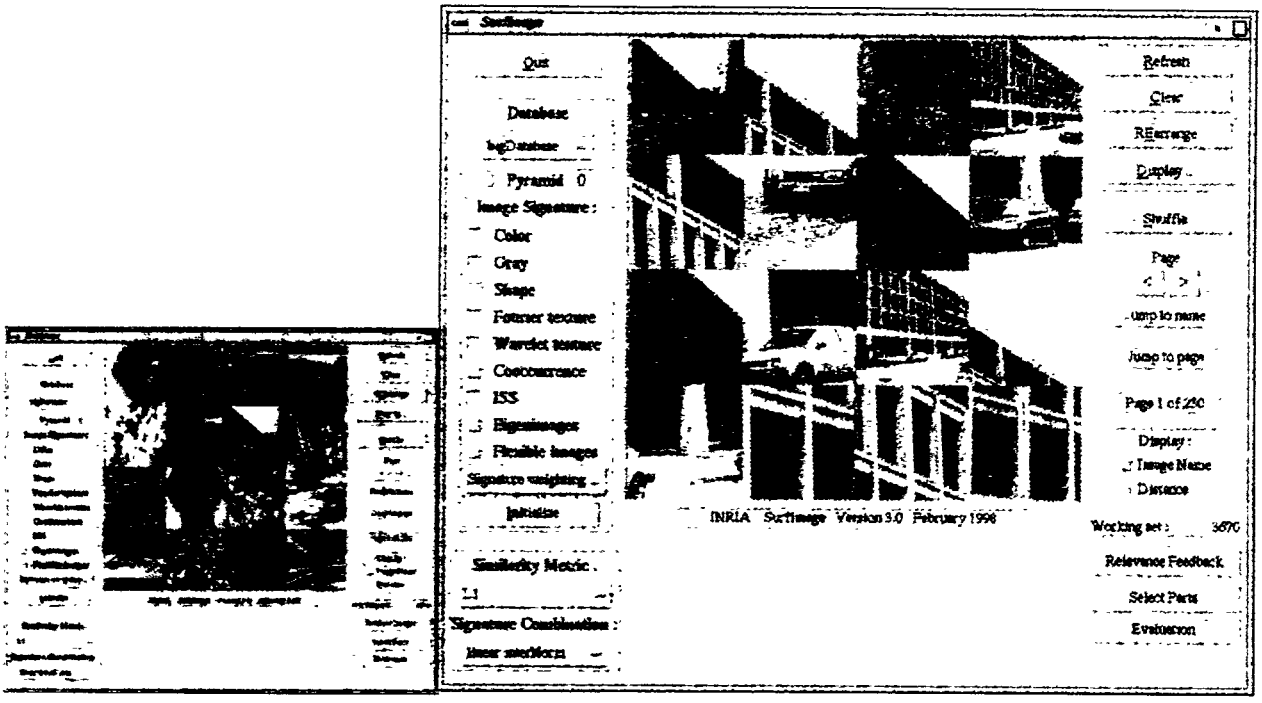

Figure 5: Classification: Left:a sample of the heterogeneous *bigdatabase* of 3670 images. Right: classifying city scenes.

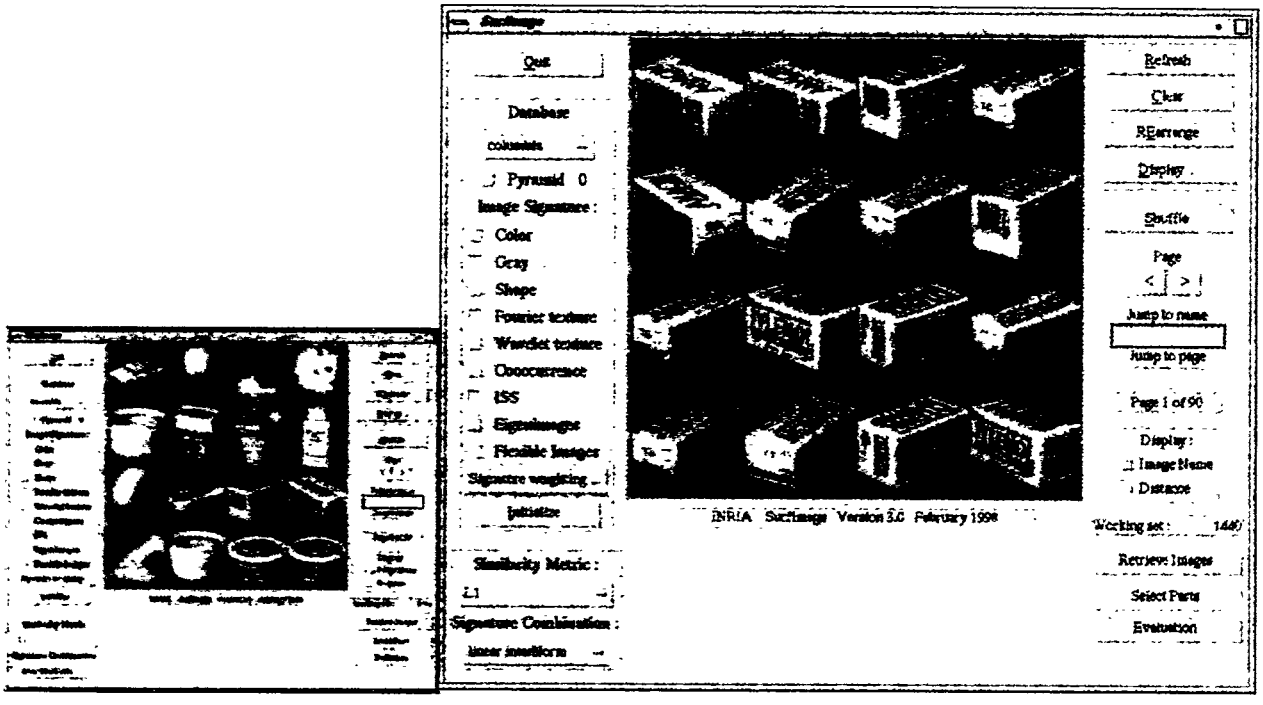

Figure 6: Multiple queries: specifying a couple of images (left) for finding more ANACIN and TYLENOL packs (right).

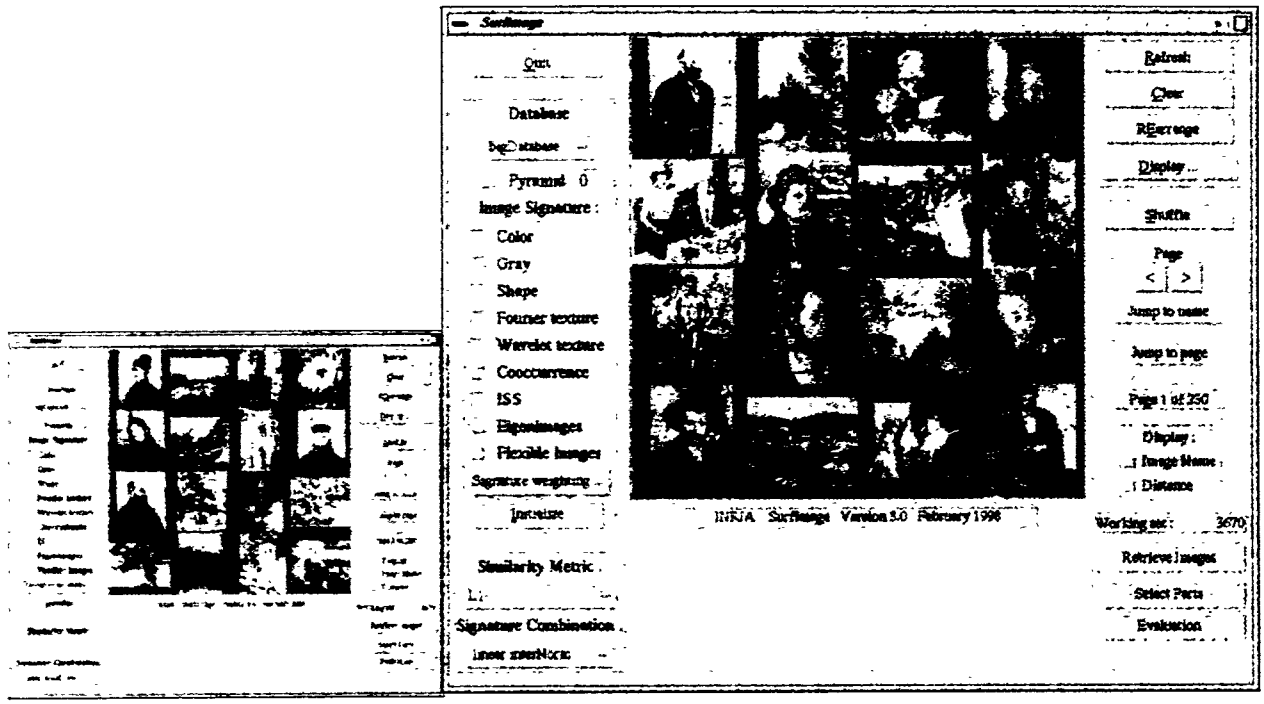

Figure 7: Relevance feedback: Using user's feedback (left) to find more portraits (right).

which was built by merging the MIT *Vistex* database of textures, the *BTphoto* database of city and country scenes, a homebrew *paintings* database, and the *homeface* database of people in the lab. The total number of images in *bigdatabase* is 3670. Figure 5 shows the ability of finding more city scenes from a query, thus essentially performing a classification task. A combination of features were used for increased performance.

Figure 6 presents the results of a multiple query on the Columbia database. The user is shuffling through the database and decides to retrieve more ANACIN and TYLENOL packs. The technique used is the relevance feedback technique detailed above, but in this case no nonrelevant image was specified.

We illustrate query refinement on the *bigdatabase* described above. The user refines their query to obtain more portraits (figure 7). Note that many of the retrieved images can be classified as portraits, although the total number of portraits in the database is small (about 2% of the images in the database are portraits).

5 Conclusion

We introduce Surfimage, a flexible content-based image retrieval system. Surfimage offers a wide range of image signatures, similarity metrics, and a user-friendly interface. Moreover, Surfimage incorporates advanced features for flexibility. Image signatures can be combined for improved performance, and for applications such as classification of a database into classes of scenes. Surfimage is also able to learn from user interaction, allowing multiple queries and query refinement by relevance feedback. The latter technique is based on a density estimation integrating positive and negative examples provided by the user, and it is shown to be more powerful than the standard relevance feedback approach.

References

[1] I. Cox et al. PicHunter: Bayesian relevance feedback for image retrieval. In *Proceedings of 13th International Conference on Pattern Recognition*, Vienna, Austria, 1996.

[2] M. Flickner et al. Query by image and video content: the qbic system. *IEEE Computer*, 28(9), 1995.

[3] A. Jain and A. Vailaya. Image retrieval using color and shape. *Pattern Recognition*, 29(8), 1996.

[4] T. Minka and R. Picard. Interactive learning using a society of models. *Pattern Recognition*, 30(4), 1997.

[5] H. Murase and S. K. Nayar. Visual learning and recognition of 3D objects from appearance. *International Journal of Computer Vision*, 14(5), 1995.

[6] C. Nastar and M. Mitschke. Real-time face recognition using feature combination. In *3rd IEEE International Conference on Automatic Face- and Gesture-Recognition (FG'98)*, Nara, Japan, April 1998.

[7] C. Nastar, M. Mitschke, and C. Meilhac. Efficient query refinement for image retrieval. In *Computer Vision and Pattern Recognition (CVPR '98)*, Santa Barbara, June 1998.

[8] C. Nastar, B. Moghaddam, and A. Pentland. Flexible images: Matching and recognition using learned deformations. *Computer Vision and Image Understanding*, 35(2), February 1997.

[9] M. Ortega, Y. Rui, K. Chakrabarti, S. Mehrotra, and T. Huang. Supporting similarity queries in MARS. In *ACM Multimedia*, Seattle, November 1997.

[10] R. Picard, T. Minka, and M. Szummer. Modeling subjectivity in image libraries. In *IEEE Int. Conf. on Image Proc.*, Lausanne, September 1996.

[11] Y. Rui, T. Huang, S. Mehrotra, and M. Ortega. A relevance feedback architecture for content-based multimedia information systems. In *Workshop on Content Based Access of Image and Video Libraries*, Porto Rico, June 1997.

[12] G. Salton. *Automatic Information Organization and Retrieval.* McGraw-Hill, New York, 1968.

[13] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1), 1991.