The truth about Corel - evaluation in image retrieval

Henning Müller, Stephane Marchand-Maillet, Thierry Pun

Computer Vision Group, University of Geneva, Henning.Mueller@cui.unige.ch

Abstract. To demonstrate the performance of content-based image retrieval systems (CBIRSs), there is not yet any standard data set that is widely used. The only dataset used by a large number of research groups are the Corel Photo CDs. There are more than 800 of those CDs, each containing 100 pictures roughly similar in theme. Unfortunately, basically every evaluation is done on a different subset of the image sets thus making comparison impossible.
In this article, we compare different ways of evaluating the performance using a subset of the Corel images with the same CBIRS and the same set of evaluation measures. The aim is to show how easy it is to get differing results, even when using the same image collection, the same CBIRS and the same performance measures. This pinpoints the fact that we need a standard database of images with a query set and corresponding relevance judgments (RJs) to really compare systems.
The techniques used in this article to "enhance" the apparent performance of a CBIRS are commonly used, sometimes described, sometimes not. They all have a justification and seem to change the performance of a CBIRS but they do actually not. With a larger subset of images it is of course much easier to generate even bigger differences in performance.
The goal of this article is not to be a guide of how to make the "apparent" performance of systems look good, but rather to make readers aware of CBIRS evaluations and the importance of standardized image databases, queries and RJ.

1 Introduction

The Corel Photo CDs have been used in many publications to demonstrate the performance of content-based image retrieval systems (CBIRSs) and has become a defacto standard in the field [2, 3, 10, 12, 18, 19, 24–27]. Sometimes the Corel images are mixed with other images [6]. Unfortunately, Corel (http://www.corel.com/) does not only offer one set of images but a collection of more than 800 Photo CDs containing 100 images each of a certain theme but also differently compiled collections of smaller-sized images with sets of several thousand images grouped into same-subject groups [27]. Even people of the same research lab presenting at the same conference often use completely different sets of images from the Corel collection to evaluate their systems [10, 13, 19]. This makes it very hard to compare the performance of systems as different groups of images can be either very easy or hard to separate from one another, depending on the images they contain. When having a large number of image sets, it does make sense to compile a collection of images with groups where each contains a distinct feature, so they can be separated easily and the performance of a system looks better (sunsets, planes on a blue sky, divers, deserts images, ...).

Once a collection is compiled, a decision has to be taken as to which images to chose as query images. Here, it is important to chose images that represent a group well and not images that differ from all other images within the group. Sometimes, the first image is taken [15] but mostly nothing is said about the process of selecting a query image [18, 25] or it is selected by hand [5, 12] or randomly [10]. Both these options leave a lot of room for choosing the images as query images that perform best for a given system, which, as we will show, can improve the performance enormously (Section 3.3). Often, more than one image of a class is chosen as query image [5, 12].

Although the Corel sets contain images of the same subject, many groups contain a few images that are visually dissimilar. It does make sense to keep these images out of the relevance sets for a query as they will make the performance look bad, although it was only a few visually dissimilar pictures that were not found. Thus, often a subset of the images is chosen to be in the relevance group [5, 25]. Sometimes the entire grouping of Corel is discarded and a new grouping is done [18, 27] which basically means that the developers can decide upon the performance of the system under evaluation. This can also be done by choosing easy-to-separate groups. Basically all researchers use a small subset of the images. Choosing the easiest subsets can drastically improve the performance as shown in Sections 3.5 and 3.6. In [26], the system developers judge the results themselves after every query step, which makes all results irreproducible.

One of the problems with using the groupings of the Corel collection as relevance judgments (RJs) is that the grouping does not always comply with what a human might chose visually as a good result for this group. What we really would like to evaluate from a CBIRS, is how well its response fits with what a human would expect to be returned as a result. Humans are subjective and have different opinions [23]. Thus, it might be sensible to use real user judgments as a basis for evaluation, as done in [22].

The problem induced by the fact that retrieving images from small sets is easier than from large collections has already been mentioned in [23] where a correction for discounting the chance factor was proposed. Similarly, the use of *generality* in [8] leads to appropriate measures to compare the retrieval quality when the database and relevance sets have a varying size. The *normalized averaged rank* proposed in [16] shows to be effective for comparing the performance when removing unused images (Section 3.6).

In Section 2, we shortly describe the CBIRS we used for our experiments. Section 3 explains the infrastructure of the system evaluation setup and shows the results of the different techniques to prune the performance of a system. As shown beforehand, all these techniques are common practices in CBIR evaluation. This highlights the need for a standard testbed for benchmarking because otherwise results are incomparable and researchers can basically choose the performance of their system by compiling a set of nice images plus friendly query images and corresponding groundtruth.

There is a large need for a common, freely-available image database plus standard queries and RJs for these queries. The Benchathlon (`http://www.benchathlon.net/`) is an iniative that tries to compile an image database and a testbed for evaluation. It is very important to support such an iniative and help to build databases and ground truth to get serious with benchmarking in CBIR. In other areas, such as in text retrieval, standard databases exist since the 60s [4] and the TREC (Text REtrieval Conference) is doing a standard evaluation every year since 1992 [7].

2 The *Viper* system

For the validation of our approach, we use the *Viper* system (`http://viper.unige.ch/`), which has been developed at the University of Geneva. A stable version called *GIFT* (GNU Image Finding Tool) can be downloaded free of charge at (`http://www.gnu.org/software/gift/`). The system is described in more detail in [17, 22].

The main difference between *GIFT* and other systems is the usage of more than $85,000$ possible features. Most images contain between $1,000$ and $2,000$ of theses features. The access method to the features is the *inverted file*, which is the most common access method used in text retrieval (TR). The system implements four different groups of image features:

- A global color histogram based on the HSV color space corresponding roughly to the human color perception [20];
- local color blocks at different scales for fixed-size regions by using the mode color for each of the fixed blocks; the image is successively partitioned into four equally sized blocks and each block is partitioned again four times;
- global texture characteristics are represented by the histograms of the response to Gabor filters of different frequencies and directions; Gabor filters are known to be a good model for the human perception of edges [14];
- local Gabor filters at different scales and in different regions by using the smallest blocks of the local color features and applying Gabor filters with different directions and scales to these blocks.

3 Results

3.1 Technical infrastructure and images used

Unfortunately, we only have access to a small subset of images from the Corel Photo CDs, containing 61 different groups of 100 images each. They do not contain the groups most often used by other systems and also relatively easy to query such as "sunsets", "tigers", "eagles on the sky". We are thus limited with the upper performance we can reach, but in the following sections it will become clear that even with a small subset the apparent performance can be boosted.

For evaluation, we use the automated benchmark described in [15]. Such a benchmark makes it easy to perform a large number of example evaluations to test and optimize the apparent system performance. The performance measures we use are described in [16] which also gives an overview about different methods used in CBIR. These measures are:

- N_{rel}, number of relevant images and t, the time it takes to execute the query;
- rel_1, $\overline{Rank}$ and $\widetilde{Rank}$: rank at which the first relevant image is retrieved, average rank and normalized average rank of relevant images, respectively;
- $P(20)$, $P(50)$ and $P(N_R)$: *precision* after 20, 50 and N_R images are retrieved;
- $R_P(.5)$ and $R(100)$: *recall* at *precision* .5 and after 100 images are retrieved;
- *Precision vs. recall (PR)-graph*.

A simple *average rank* is difficult to interpret, since it depends on both the collection size N and the number of relevant images N_R for a given query. Consequently, we normalize by these numbers and use the *normalized average rank*, $\widetilde{Rank}$:

$$\widetilde{Rank} = \frac{1}{NN_R}\left(\sum_{i=1}^{N_R} R_i - \frac{N_R(N_R-1)}{2}\right) \tag{1}$$

where R_i is the rank at which the ith relevant image is retrieved. This measure is 0 for perfect performance, approaches 1 as performance worsens, and is 0.5 for random retrieval.

3.2 Evaluation of the Corel database

Our initial evaluation procedure using databases with fixed grouping consists in using the first image of a group as a query image and the entire set as RJ for the query.

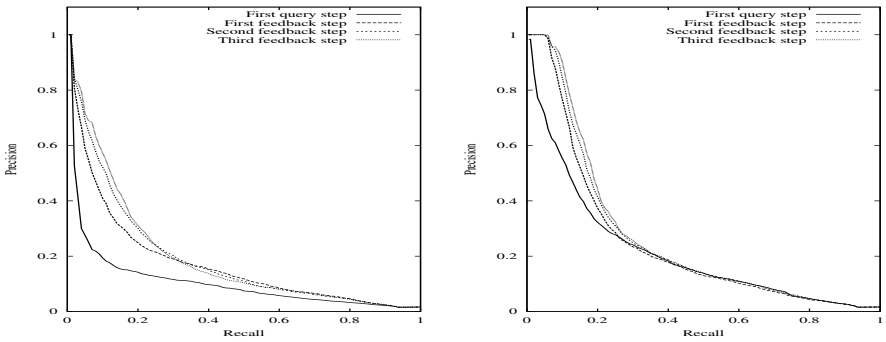

Fig. 1. *PR-graph* using the first image as query image (left) and an optimized query image (right).

The left side of Fig. 1 and Table 1 show that the results do not look very good, although the use of RF does improve sensibly the *precision* in the first $n = 20$ images. Having an average of 5 relevant images in the first 20 returned does not sound too bad. However, we can see that rel_1 is well above 1, which means that there are image sets where no relevant image was retrieved in the top 20.

	1st step	RF 1	RF 2	RF 3		1st step	RF 1	RF 2	RF 3
N_{rel}	100	100	100	100	N_{rel}	100	100	100	100
$timet$	8.53	11.69	12.62	13.19	$timet$	8.35	11.60	13.08	12.89
rel_1	28.79	21.13	24.30	23.56	rel_1	1.49	1	1	1
$R(P(.5))$	.0552	.1336	.1487	.1525	$R(P(.5))$	.1590	.1904	.2074	.2136
$\overline{Rank}$	1713.4	1602.8	1561.0	1543.5	$\overline{Rank}$	1223.9	1205.7	1185.9	1176.1
$\widetilde{Rank}$	.2537	.2368	.2303	.2277	$\widetilde{Rank}$	.1789	.1761	.1731	0.1716
$P(20)$	.2508	.4082	.4795	.5164	$P(20)$	.5508	.6352	.6959	.7344
$P(50)$	.1711	.2603	.2846	.2895	$P(50)$	.3531	.3750	.3892	.4006
$P(N_r)$	.1267	.1821	.1857	.19	$P(N_r)$	.2452	.2449	.2566	0.2549
$R(100)$	.1267	.1821	.1857	.19	$R(100)$	.2452	.2449	.2566	0.2549

Table 1. Performance using the first image (left) and an optimized image (right) as query.

When taking a look at the sets with the worst results we can see that sometimes the first image is visually very different from other images in the same group. An example is a query for images of "Paris" where the first image of the group was a church at night, which perfectly helped retrieving images of churches and other buildings at night, but none of them were from "Paris". Therefore, it sounds logical to chose a query image that represents the class in a better way.

3.3 Optimizing the query image

For not having to chose a "best" query image by hand, we wanted to find the image that represents the class best from our system's point of view. We thus evaluated a query for every image of every class and choose the image with the highest $P(20)$. If there were several images with the same value, then $P(50)$ gave the decision. These two measures correspond to what a normal user would look at on the screen and therefore we regard these measures as the most important ones.

Fig. 1 and Table 1 illustrate how strongly the performance of our system improved, although the same system with the same relevance sets and the same database was used. For every group, we have a relevant image in the top 20 after one step of feedback, we even have an average of 11 relevant images in the top 20 and after three steps of RF even 15 relevant images in the top 20.

Comparing the two data sets in Table 1, we can see clearly that all the performance measures improve strongly. $P(20)$ more than doubles and the rank-based measures improve substantially.

3.4 Removing bad images

We said earlier that there are images in each group that are visually dissimilar from several images in the same group. Although these images are not used as query images anymore, they still worsen the results of retrieval. We do not want to cut too many images and set an upper limit of 20 images to be removed from each set of the database. We assume that an image is really bad for the query and can never be retrieved in one of the top ranks if the image is ranked below one third of the entire database (we used $Rank < 2000$).

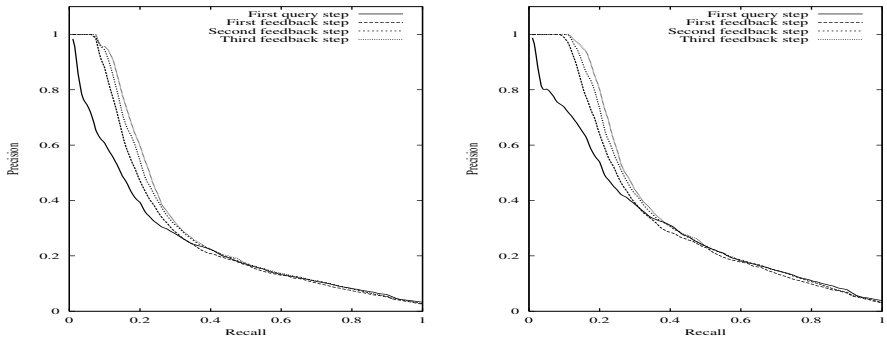

Fig. 2. *PR-graph* when removing bad images from the RJ (left) and removing bad groups (right).

As changing this does not at all change the ranking of the images and neither the *precision* after the first images are retrieved, we can only hope for an improvement in the *recall* measure. Fig. 2 shows that the entire *PR-graph* looks improved as the curve is slightly lifted up. This makes the performance looks quite a bit better although the *precision* measures in Table 2 show that the performance in the first 2000 result images did not change at all. As expected, the average rank measures and the $R(100)$ do improve significantly.

	1st step	RF 1	RF 2	RF 3		1st step	RF 1	RF 2	RF 3
N_{rel}	82.41	82.41	82.41	82.41	N_{rel}	83.68	83.68	83.68	83.68
$timet$	8.39	11.56	12.33	12.77	$timet$	8.66	11.90	12.43	12.46
rel_1	1.49	1	1	1	rel_1	1.43	1	1	1
$R(P(.5))$	.1871	.2268	.2465	.2544	$R(P(.5))$	.2519	.2839	.3058	.3124
$\overline{Rank}$	850.3	883.3	860.0	850.39	$\overline{Rank}$	600.5	651.3	628.2	616.2
$\widetilde{Rank}$	.1327	.1381	.1343	.1327	$\widetilde{Rank}$	.0917	.0999	.0962	.0942
$P(20)$	.5508	.6352	.6959	.7344	$P(20)$	.6463	.7412	.8088	.8463
$P(50)$	.3531	.3751	.3891	.400	$P(50)$	.4365	.4495	.4685	.4845
$P(N_r)$	.2717	.2729	.2829	.2882	$P(N_r)$	.3368	.3290	.3395	.3456
$R(100)$	.2931	.2933	.3077	.3055	$R(100)$	.3609	.3532	.3648	.3650

Table 2. Performance when removing bad images from the RJ (left) and bad groups (right).

3.5 Removing bad image groups

The easiest way of improving the performance is of course to only have a set of easily distinguishable groups of images. With the Corel Photo CDs containing more than $80,000$ images it is easy to choose a small subset of groups that work well. Unfortunately, we do not have access to many of the image groups commonly mentioned in citations, but we can already see that removing the 21 worst relevance sets from the evaluation (leaving us with 40 query images) still leads to a massive improvement in performance. And this with still using the exact same system, exact same database with the exactly same measures for evaluation.

Fig. 2 and Table 2 show that basically all measured results are massively improved by simply removing a few bad groups from the evaluation setup. We now have average of 13 relevant images in the top 20 and 22 in the top 50. This is 13 images more than what we had with our first evaluation. All values basically doubled in result.

3.6 Creating a smaller database

Until now, we always used the exact same database for the queries. Just like we remove image sets or single images from the query process, it makes sense to remove all these unused images completely from the database. Like this, we create a database where every image is part of exactly one relevance group. This database now contains 40 image groups and a total of 3500 images.

We can see clearly in Fig. 3 and Table 3 that this again improves the performance strongly, especially the *average rank* drops to roughly half the value. The query time is strongly reduced and all *precision* and *recall* values go up. Interestingly, the *normalized average rank* keeps almost exactly the same value as before, which shows that the basic

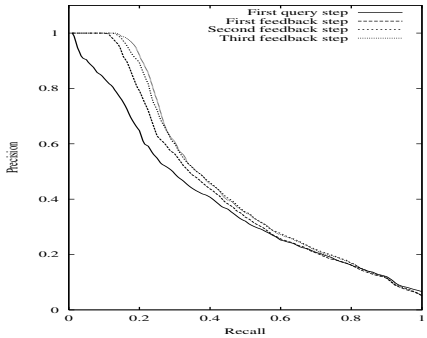

Fig. 3. *PR-graph* when creating an inverted file with only the good images and groups.

performance of the system did not improve and that it simply looks better because of the smaller database size. As the *normalized average rank* is normalized by the database size, it characterizes well the ranks of relevant images.

	1st step	RF 1	RF 2	RF 3
N_{rel}	83.68	83.68	83.68	83.68
$timet$	4.71	5.80	6.51	6.52
rel_1	1.18	1	1	1
$R(P(.5))$	.3228	.3654	.3863	.3923
$\overline{Rank}$	350.3	364.3	347.88	351.89
$\widetilde{Rank}$	.09086	.0950	.0902	.0913
$P(20)$	.7212	.8250	.8813	.9163
$P(50)$	.504	.5345	.5585	0.5650
$P(N_r)$	.3848	.4013	.4094	.4092
$R(100)$	.4162	.4268	0.4387	.4391

Table 3. Performance when creating an inverted file with only the good images and groups.

3.7 Global comparison

As it is very hard to compare graphical values over several graphs, Fig. 4 shows a comparison of the different evaluations in the first query step. The graph shows how strongly the performance of the system may be improved by simply choosing well the image sets, query images and RJs.

Fig. 4 also shows the improvement in performance obtained for the first RF step. The increase in performance between the best and worst graph is often more than 100%.

The same can be seen when we take a look at the other performance measures shown in Tables 1–3. $P(20)$ improved from $.25$ to $.72$, $P(50)$ from $.17$ to $.50$ and $R(100)$ from $.13$ to $.42$. Still, we have to recall that it is exactly the same algorithm with exactly the same results, only the method of evaluation is changing between the different steps.

With having a larger number of Corel image sets we could surely increase the performance even more, and with not only removing the worst 20 images but a larger number, we can as well get a much higher improvement. If we even create the relevance sets on our own such as [18], we can basically select the apparent performance of our system even when using the same metrics and features as before.

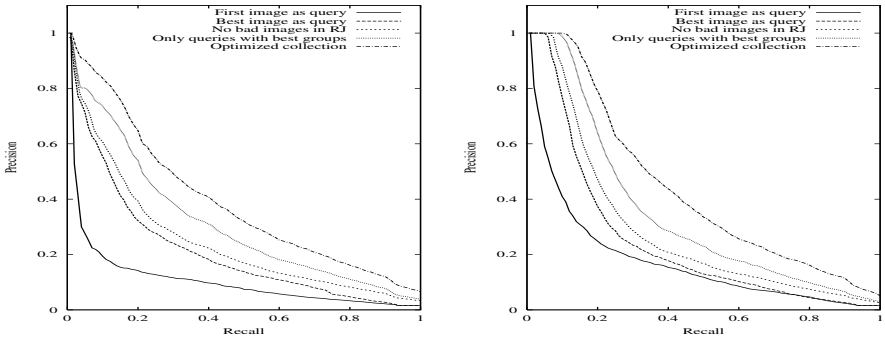

Fig. 4. Comparing the initial query (a) and the first feedback step (b) of all evaluation runs.

3.8 Easy query groups

Finally, to have an idea about the performance differences of single images sets, we list in Table 4 the images groups with the highest $P(50)$, characterizing an easy group. We can see how much easier the very easy image sets are. By having the entire set of more than 800 groups it should be possible to have an almost perfect performance at least for the first 50 results.

$P(50)$	group number	content	$P(50)$	group number	content
0.92	211000	oil paintings	0.66	153000	swimming Canada
0.9	156000	divers and diving	0.6	524000	works of art: engravings
0.9	99000	religious stained glass	0.54	161000	land of the pyramids
0.72	373000	everyday objects	0.54	502000	works of art: landscapes
0.7	125000	coins and currency	0.54	96000	candy backgrounds

Table 4. $P(50)$ for the easiest image groups.

Based on this study, it would be interesting to create a difficulty measure for every Corel image set, but unfortunately the difficulty of a group depends on the other groups present in an image database and the CBIRS used. Not only the intra-group similarity but also inter-group similarities are important to measure the difficulty of an image collection.

4 Conclusions and future work

This article shows how easy it is to change the apparent performance of a CBIRS without changing the system and even by using the same image set or a subset. This demonstrates that it is impossible to objectively compare CBIRS performances unless it is clearly stated which images where used, which images were used as queries and what had been used as RJs. Most of the times, this is not stated in articles on CBIR and only the database used plus a *precision* measure or a *PR-Graph* of the system is shown. This makes it completely impossible to compare any two CBIRS.

The introduction of measures based on *generality* in [8] and the correction of retrieval by chance in [23] do help in some way, but many of the problems still persist when the other factors in evaluation are not clear.

The aim of this article is to highlight the need for standardized evaluation in CBIR and especially the need for a standard image database. In text retrieval, this was already realized many years ago [21] and large efforts were made to create such databases. In CBIR, an effort to create a database exists (`http://www.cs.washington.edu/research/imagedatabase/groundtruth/`) and there is also an iniative to create a benchmark for CBIR, the Benchathlon (`http://www.benchathlon.net/`). It is very important to support these efforts if we want to be able to really compare CBIRSs.

Not only the creation of a standard database for CBIR is important but also the creation of proper RJs. Predefined groups of the same subject such as the Corel Photo CDs are interesting because they are easy to use without any effort. However, the fact that groups contain very dissimilar images gives room and reasons for manipulations. Although real user tests might be the best option to gain RJs, they are costly and have to be repeated once the database gets bigger. Thus, it is important to develop alternatives such as generating RJs based on annotation [11].

References

1. *Proceedings of the ACM Multimedia Workshop on Multimedia Information Retrieval (ACM MIR 2001)*, Ottawa, Canada, October 2001. The Association for Computing Machinery.
2. C. Carson, S. Belongie, H. Greenspan, and J. Malik. Color- and texture-based segmentation using em and its application to image querying and classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1998.
3. K. Chakrabarti, K. Porkaew, and S. Mehrotra. Efficient query refinement in multimedia databases. In *Proceedings of the 16th International Conference on Data Engineering (ICDE2000)*, San Diego, CA, USA, March 1–3 2000. IEEE Computer Society.
4. C. W. Cleverdon, L. Mills, and M. Keen. Factors determining the performance of indexing systems. Technical report, ASLIB Cranfield Research Project, Cranfield, 1966.
5. L. Duan, W. Gao, and J. Ma. A rich get richer strateghy for content-based image retrieval. In R. Laurini, editor, *Fourth International Conference On Visual Information Systems (VISUAL'2000)*, number 1929 in Lecture Notes in Computer Science, pages 290–299, Lyon, France, November 2000. Springer-Verlag.
6. K.-S. Goh, E. Chang, and K.-T. Cheng. Support vector machine pairwise classifiers with error reduction for image classification. In ACMMIR2001 [1], pages 32–37.
7. D. Harman. Overview of the first Text REtrieval Conference (TREC-1). In *Proceedings of the first Text REtrieval Conference (TREC-1)*, pages 1–20, Washington DC, USA, 1992.
8. D. P. Huijsmans and N. Sebe. Extended performance graphs for cluster retrieval. In *Proceedings of the 2000 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2001)*, pages 26–31, Kauai, Hawaii, USA, December 9–14 2001. IEEE Computer Society.
9. IEEE. *Proceedings of the second International Conference on Multimedia and Exposition (ICME'2001)*, Tokyo, Japan, August 2001. IEEE.
10. F. Jing, B. Zhang, F. Lin, W.-Y. Ma, and H.-J. Zhang. A novel region-based image retrieval method using relevance feedback. In ACMMIR2001 [1], pages 28–31.
11. C. Jörgensen and P. Jörgensen. Testing a vocabulary for image indexing and ground truthing. In G. Beretta and R. Schettini, editors, *Internet Imaging III*, volume 4672 of *SPIE Proceedings*, San Jose, California, USA, January 21–22 2002. (SPIE Photonics West Conference).
12. C. S. Lee, W.-Y. Ma, and H. Zhang. Information embedding based on user's relevance feedback in image retrieval. In S. Panchanathan, S.-F. Chang, and C.-C. J. Kuo, editors,

Multimedia Storage and Archiving Systems IV (VV02), volume 3846 of *SPIE Proceedings*, pages 294–304, Boston, Massachusetts, USA, September 20–22 1999. (SPIE Symposium on Voice, Video and Data Communications).

13. M. Li, Z. Chen, L. Wenyin, and H.-J. Zhang. A statistical correlation model for image retrieval. In ACMMIR2001 [1], pages 42–45.
14. W. Y. Ma, Y. Deng, and B. S. Manjunath. Tools for texture- and color-based search of images. In B. E. Rogowitz and T. N. Pappas, editors, *Human Vision and Electronic Imaging II*, volume 3016 of *SPIE Proceedings*, pages 496–507, San Jose, CA, February 1997.
15. H. Müller, W. Müller, S. Marchand-Maillet, D. M. Squire, and T. Pun. Automated benchmarking in content-based image retrieval. In ICME'2001 [9].
16. H. Müller, W. Müller, D. M. Squire, S. Marchand-Maillet, and T. Pun. Performance evaluation in content-based image retrieval: Overview and proposals. *Pattern Recognition Letters*, 22(5), April 2001.
17. H. Müller, W. Müller, D. M. Squire, Z. Pečenović, S. Marchand-Maillet, and T. Pun. An open framework for distributed multimedia retrieval. In *Recherche d'Informations Assiste par Ordinateur (RIAO'2000) Computer-Assisted Information Retrieval*, volume 1, pages 701–712., Paris, France, apr 12-14 2000.
18. M. Ortega, Y. Rui, K. Chakrabarti, K. Porkaew, S. Mehrotra, and T. S. Huang. Supporting ranked boolean similarity queries in MARS. *IEEE Transactions on Knowledge and Data Engineering*, 10(6), December 1998.
19. F. Qian, M. Li, W.-Y. Ma, F. Ling, and B. Zhang. Alternating features spaces in relevance feedback. In ACMMIR2001 [1], pages 14–17.
20. J. R. Smith and S.-F. Chang. VisualSEEk: a fully automated content-based image query system. In *The Fourth ACM International Multimedia Conference and Exhibition*, Boston, MA, USA, November 1996.
21. K. Sparck Jones and C. van Rijsbergen. Report on the need for and provision of an ideal information retrieval test collection. British Library Research and Development Report 5266, Computer Laboratory, University of Cambridge, 1975.
22. D. M. Squire, W. Müller, H. Müller, and J. Raki. Content-based query of image databases, inspirations from text retrieval: inverted files, frequency-based weights and relevance feedback. In *The 11th Scandinavian Conference on Image Analysis (SCIA'99)*, pages 143–149, Kangerlussuaq, Greenland, June 7–11 1999.
23. D. M. Squire and T. Pun. A comparison of human and machine assessments of image similarity for the organization of image databases. In M. Frydrych, J. Parkkinen, and A. Visa, editors, *The 10th Scandinavian Conference on Image Analysis (SCIA'97)*, pages 51–58, Lappeenranta, Finland, June 1997. Pattern Recognition Society of Finland.
24. N. Vasconcelos and A. Lippman. Learning over multiple temporal scales in image databases. In D. Vernon, editor, *6th European Conference on Computer Vision (ECCV2000)*, number 1842 in Lecture Notes in Computer Science, pages 33–47, Dublin, Ireland, June 26–30 2000. Springer-Verlag.
25. N. Vasconcelos and A. Lippman. A propabilistic architecture for content-based image retrieval. In *Proceedings of the 2000 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'2000)*, Hilton Head Island, South Carolina, USA, June 13–15 2000. IEEE Computer Society.
26. J. Z. Wand, J. Li, and G. Wiederhold. Simplicity: Semantics-sensitive integrated matching for picture libraries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23 No 9:1–17, 2001.
27. L. Zhu, C. Tang, A. Rao, and A. Zhang. Using thesaurus to model keyblock-based image retrieval. In ICME'2001 [9], pages 237–240.