A Novel Histogram-biasing Factor for Fast Sorted Histogram-based Measurement in Large Image Database Retrieval System

Chun-Ho Cheung and Lai-Man Po

terence@ieee.org and eelmpo@cityu.edu.hk
Department of Electronic Engineering, City University of Hong Kong,
83 Tat Chee Avenue, Kowloon, Hong Kong SAR, China.

Abstract
The exhaustive histogram matching is usually the most computational intensive part for any query in most large image database retrieval systems. In this paper, we introduce a histogram-biasing factor (HBF) to measure the biased-behavior of ordered-bins in a sorted histogram. The proposed HBF can be used to increase the early rejection rate of unreliable or impossible candidate reference images based on one of the sorted histograms. Moreover, it can be treated as a color-histogram descriptor. Only images with very closed HBFs are taken into account, searching speed can thus be increased without loss of accuracy. Experimental results show that the proposed factor results in up to 13 times speedup meanwhile providing the exhaustive retrieval performance.

1. Introduction

With the daily increasing of digital multimedia content from different sources and channels, such as digital pictures and photos from scanners and digital cameras, audio or/and video from CDs / DVDs, recorders, broadcasting channels and digital libraries on the Internet, there is a strong demand on managing and making use of such explosion in available content. To have an efficient indexing and effective retrieving method for multimedia content, keyword annotation [1] is not the best method even though text-based search engines work efficiently in today's Internet and digital libraries. In addition, a high-level semantic description perceived from a query image usually has a large variation from its corresponding low-level one, such as color, texture and shape. Among these features, color is usually employed in many works [2, 10] in the literature since it is the most straightforward information obtainable from an image, without or with lesser pre-processing and extra storage.

Color histogram [2] is one of the popular descriptors that characterize the color distribution in an image. It has also been exploited in various domains, such as DCT or other color spaces (HSV, YUV, CIE*luv*, etc) [3, 4], and is very applicable to content-based image retrieval (CBIR) system [5, 10], which looks for similar or feature-relevant images from a large image database. However, the computational cost for matching a query image from a large database, which usually includes over hundred thousand of images, makes the search severely slowed down even using a promising histogram-based algorithm. Recently, some fast algorithms [6-8] are proposed to address such problem. In [6], Hafner *et al*. propose to eliminate unnecessary full-resolution histogram matching. They employ singular value decomposition (SVD) to generalize a low-resolution feature for speeding up the exhaustive search. Berman *et al*. [7] and Song *et al*. [8] employ the concept of triangular inequality and successive elimination algorithm [9] to eliminate unnecessary matching operations from the search procedures. In addition, Song et al. propose a multi-resolution approach by using sum pyramid structure of color histogram. In [6-8], measurement in lower resolutions is advantageous of earlier eliminating of unreliable candidates. However, computations are still required to justify even at the lowest resolution. In this paper, a histogram-biasing factor (HBF) is proposed as a color-histogram descriptor. It is used for identifying either sorted histogram as the distance cumulating reference. In addition, it can imply the images in comparison are dissimilar if their HBFs are far from each others. Speed is thus gained without further investigations. This advantage of avoiding inevitable computations can also be gained if full or multi-resolution approach [6-8] is adopted.

2. Histogram Intersection and Dissimilarity

A. Histogram Intersection Approach

Normalized histogram of an image, $H(I)=\{\tilde{I}_i\}$, is acquired from the statistical color distribution in particular color space (C) of interest. To measure the degree of similarity between two images Q and R, their histograms are constructed and intersected $I(R, Q)$ [2] as shown below:

$$I(R,Q) = \sum_{i=1}^{n} \min\left(\tilde{R}_i, \tilde{Q}_i\right) \quad (1)$$

where $\tilde{R}_i$ and $\tilde{Q}_i$ are the normalized counts of a particular color c, $c \in C$, of n-color image R and Q, respectively, and $\sum_{i=1}^{n} \tilde{R}_i = \sum_{i=1}^{n} \tilde{Q}_i = 1$. Such histogram intersection method (HIM) is very robust as color histograms are independent of geometrical content such as translation, rotation, etc, and thus is widely employed in color image retrieval applications [10]. Larger value of $I(R, Q)$, $0 \le I(R, Q) \le 1$, indicates the two images are more similar.

B. Dissimilarity Metric for Sorted Histogram

In most images, some colors exist much more than the others, especially in natural images. Furthermore, not all the colors are representatives and perceived or annotated from human beings' point of view. Such dominant and key color concepts are the philosophy behind the color-indexed images, such as GIF-coded ones, and vector-quantized (VQ) [11] images. For similarity measure, comparison using such dominant key-colors in the beginning of matching can significantly avoid unnecessary computations by early rejection of unreliable candidates. For histogram-based method, either histogram is thus sorted with dominant colors being compared initially by using *dissimilarity or distance* metric D, instead of using intersected area I. Dissimilarity between images R and Q, i.e. $D(R, Q)$, is defined as the total differences of every color count as shown below:

$$D(R,Q) = \sum_{i=1}^{n} \left| \tilde{R}_i - \tilde{Q}_i \right| \quad (2)$$

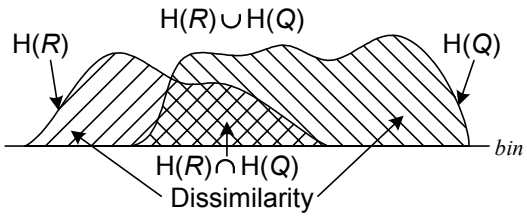


Fig. 1 Dissimilarity of two color histograms.

If the intersected area $I(R, Q)$ is equal to $|H(R)\cap H(Q)|$, the dissimilarity can be regarded as $D(R,Q) = |H(R)\cup H(Q)| - I(R,Q)$, and $0 \le D(R,Q) \le 2$, as depicted in Fig. 1. In contrast to I, the smaller the value D is, the more similar the two images are. Thus, the partial sum of dissimilarity D involves many dominant colors cumulated in the beginning of every match, and compared against the current lowest ranked image's. Early elimination of unreliable candidate images is feasible if the partial sum has been greater than the lowest ranked image's. Therefore, the order of partial sum of D is computed based on the descending ordered-bin of either histogram acquired from the two images in comparison.

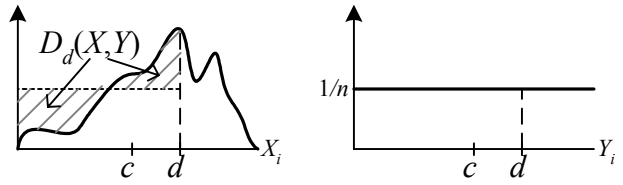
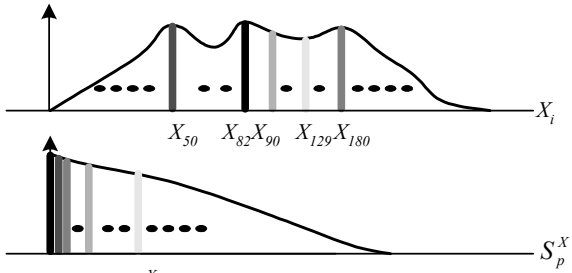
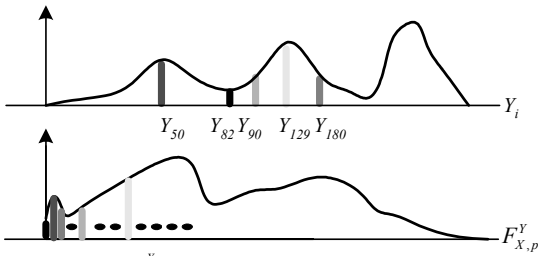


Fig. 2 Dissimilarity measure using PDS between image X and Y without any histogram sorted for ordering reference.



Ordered bin : $\{S_p^X\} = \{ X_{82}, X_{50}, X_{180}, ..., X_{90}, ..., X_{129}, ... \}, 1 \le p \le n$.

(a)



Followed bin : $\{F_{X,p}^Y\} = \{ Y_{82}, Y_{50}, Y_{180}, ..., Y_{90}, ..., Y_{129}, ... \}, 1 \le p \le n$.

(b)

Fig. 3(a) Sorted histogram S^X of image X. (b) Followed histogram F_X^Y of image Y with bin ordering based on S^X.

C. Histogram-biasing Factor

Typically, the partial distance search approach (PDS) [12] widely used in encoding process of VQ can be directly applied to HIM. It can reduce the computations required in the full search (FS) matching style on every bin of image histograms throughout the whole database. Consider the normalized histograms $H(X)$ and $H(Y)$ shown in Fig. 2, early rejection exists when partial dissimilarity distance $D_d(X,Y)$ cumulated at color bin d in both histograms. It is noted that the current minimum dissimilarity distance D_{MIN} is a monotonically decreasing function as the search proceeds. In this case, the shaded area in gray is assumed just larger than the D_{MIN}, i.e. $D_d(X,Y) > D_{MIN}$.

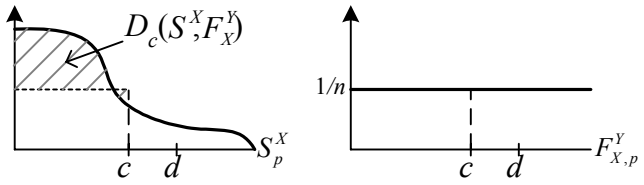


Fig.4 Dissimilarity measure using PDS on sorted histograms.

In order to speed up the early rejection rate, one of the histograms is sorted, $S^X = \{S_p^X\}$. The other will be re-arranged by following the ordered-bin of the sorted one, and is called *followed histogram*, $F_X^Y = \{F_{X,p}^Y\}$. They are illustrated in Fig. 3. By sorting the histograms based on dominant colors in X, a possible sorted version of Fig. 2 is shown in Fig. 4. For the same D_{MIN}, partial distance $D_c(S^X, F_X^Y)$ is just larger than the D_{MIN} at c-th iteration of distance accumulation, where $c<d$, i.e. $D_c(S^X, F_X^Y) > D_d(X,Y) > D_{MIN}$. This results in earlier rejection of unreliable candidate. X is a typical image histogram but Y contains uniform color distribution, which is rarely found in our real world. Fig. 5 shows another image Z with a narrow-spread histogram and with smaller variance than X's. To start dissimilarity measure between X and Z, a histogram-biasing factor (HBF) is proposed to measure their tendency of colors towards the most dominant one (the mode) in their sorted versions. The HBF of a sorted histogram $S = \{S_p\}$ is defined as shown in Eq. (3) and normalized between 0 and 1.

$$HBF = \sum_{i=0}^{n-1} iS_i \Big/ \sum_{i=0}^{n-1} i\left(\frac{1}{n}\right) = \frac{2\sum_{i=0}^{n-1} iS_i}{n-1} \qquad (3)$$

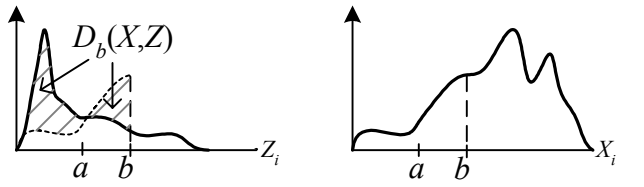


Fig. 5 Compare X against Z, which has a narrow-spread histogram.

For the two sorted histogram S^X and S^Z, the one with colors having higher tendency or biasing behavior towards the dominant colors will be sorted, i.e. HBF_Z is smaller. The other will be used as followed histogram, i.e. F_Z^X. Fig. 6 illustrates that image X will follow the order of sorted histogram of Z. The partial dissimilarity distance $D_a(S_Z, F_Z^X)$ is just greater than the D_{MIN} at earlier ordered-bin a, where $a < b$, i.e. $D_a(S_Z, F_Z^X) > D_b(X,Z) > D_{MIN}$.

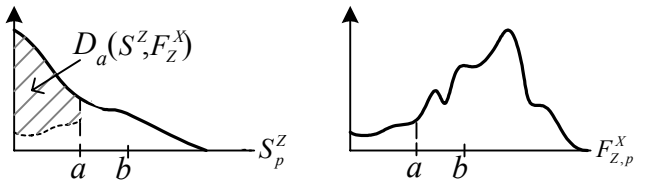


Fig. 6. Dissimilarity measure using PDS with more mode-biased sorted histogram as reference.

3. Dominance-biased Partial Distance Search Algorithm

A. Details of the proposed algorithm

With the use of the histogram-biasing factor (HBF), dominance-biased partial distance search algorithm (DBPDS) is proposed. After submitting a query from a database with N images $\{R(j)\}$, $1 \le j \le N$, dissimilarity measure $D(R(j), Q)$ on the query image Q against every candidate image $R(j)$ is performed. For each pair of comparison, their HBFs are first compared. The one with smaller HBF ($0 \le$ HBF ≤ 1) will then have its histogram sorted, and compared against the other. The sorted histogram is selected as:

$$S = \begin{cases} \{S_p^Q\}, & if\ (HBF_Q \le HBF_{R(j)}) \\ \{S_p^{R(j)}\}, & if\ (HBF_Q > HBF_{R(j)}) \end{cases} \quad (4)$$

and the other one (F_S) is re-ordered as following the S's bins.

Therefore, partial dissimilarity distance $D_p(S, F_S)$ up to p-th ordered bin (a particular i-th bin) is accumulated and compared against the current *minimum dissimilarity distance* D_{MIN}. If $D_p(S, F_S)$, or simply D_p, is greater than the D_{MIN}, the corresponding candidate $R(j)$ is unreliable to be the best match. Otherwise, comparison against the D_{MIN} continues with inclusion of next $(p+1)$-th ordered-bin and performs until all bins in $\{S_p\}$ are checked. Therefore, the proposed DBPDS can eliminate unreliable candidates without comparing the whole histograms. Below summarizes the DBPDS:

Step (1) Set $p = 1$, $j = 1$, $D_0 = 0$, and D_{MIN}=*BIG_TH*.

Step (2) If j is not larger than N, identify S and F_S from Q and $R(j)$ using Eq. (4).

Step (3) (i) If p is equal to n, go to Step (4).

(ii) Else if j is larger than N, go to Step (5).

(iii) Otherwise, get the corresponding $\widetilde{R}_i(j)$ (or $\widetilde{Q}_i$) of the current S_p. Compute $d_p(S, F_S)$ and update $D_p(S, F_S)$.

(a) If $D_p(S,F_S)$ is larger than the D_{MIN}, matching with the current candidate $R(j)$ is stop and this candidate is eliminated from the rest of search , and then replace p and j respectively with 1 and j+1. Repeat Step (2)

(b) Otherwise, replace p with p+1. Repeat Step (3).

Step (4) Get the corresponding $\widetilde{R}_i(j)$ (or $\widetilde{Q}_i$) of the current S_n. Compute $D_n(S, F_S)$.

(i) If $D_n(S, F_S)$ is larger than the D_{MIN}, the current candidate $R(j)$ is finally eliminated at its last chance.

(ii) Otherwise, update the D_{MIN} with $D_n(S, F_S)$.

(iii) Go to Step (2) by replacing p and j with 1 and j+1, respectively.

Step (5) Display the best-matched image, which gets the final smallest D_{MIN}.

Initially, the minimum dissimilarity distance D_{MIN} can be set to infinity, or a threshold (*BIG_TH*) that is large enough for further minimum replacement. In addition, K best matches for Q are usually ranked instead of outputting one best match only. Therefore, a ranked list of $D_{MIN}(k)$, $1 \le k \le K$, is resulted. To achieve such a ranked list, two steps are added to the algorithm. Firstly, comparisons are taken against the lowest ranked $D_{MIN}(k)$, $1 \le k \le K$, i.e. Step(3)(iii)(a) and Step(4)(i). Secondly, the ranked list $D_{MIN}(k)$ is updated by inserting the newly $D_{MIN}(k+1)$ to proper location in an ascending manner of minimum distances, i.e. Step(4)(ii). Such update of ranking can be accomplished by using any sorting algorithm [13], such as insertion sort, or quick sort, etc.

B. Further speed improvement

As partial distance search (PDS) approach, with or without any histogram sorted, results in the same ranked similar images with the same minimum distances as using full search (FS) approach. This can be regarded as *lossless* retrieval. The proposed DBPDS can be adjusted to provide faster matching speed meanwhile without loss of retrieval result. This can be achieved by ignoring the dissimilarity measurement on images $R(j)$, whose HBFs are far away from the query one's. In other words, the search firstly compares the two HBFs of images in comparison and continues if the difference between HBFs is within a particular allowable matching range (AMR), otherwise, the current reference image is treated as impossible and eliminated. This approach may introduce a *lossy* version of different ranked images as compared to FS.

C. Practical implementation and extra storage analysis

In DBPDS, the candidate images with their histograms, their HBFs and sorted histograms are computed beforehand and stored associatively in the database. For sorted histograms, the comparison requirement when using quick sort is about $2n(\ln(n))$ operations in average case, where ln denotes natural log. Nevertheless, sorting of n-color images' histograms is an offline batch job and does not affect the efficiency of a query. The extra storage requirement of DBPDS is just double the size of lookup-table of n entries of counts; typically each entry requires a double floating-point type (for bins) and a long integer type (for ordering). This is equivalent to about $(8+4)n+8$ bytes or about 3KBytes, for a 256-color image.

4. Experimental Results

In our simulations, a database of N=1,000 JPEG images [14] is divided into 10 different categories, for example, dinosaur, African, etc. For each category, it consists of N_R=100 images. The proposed DBPDS approach is compared with the (1) FS approach, and (2) PDS approach without any histogram sorted. To evaluate the speed improvement ratio, the number of operations, such as additions, absolutions and comparisons, are compared. All approaches (App.) employ the same sorting algorithm, such as quick sort, to generate the top K=10 ranked images.

The bin size is set to n and $4n$ for two color domains, and results in two luminance 256-bin (256Y) and 1024-bin (1024Y) histogram matching and one 256-bin (256C) color histogram matching. The retrieval performance is measured by *Precision* and *Recall* [15], and early rejection rate (RR), as shown below:

$$precision = \frac{\text{No. of relevant images retrieved}}{\text{Total no. of images retrieved}}, \quad (5)$$

$$recall = \frac{\text{No. of relevant images retrieved}}{\text{Total no. of relevant image in database}}, \text{ and} \quad (6)$$

$$Rejection\ Rate\ (RR) = \frac{Comp(\text{FS}) - Comp(\text{FastX})}{Comp(\text{FS})} \times 100\%, \quad (7)$$

where $Comp$(FastX) is the total number of partial dissimilarity comparisons against the D_{MIN} using fast method FastX. Relevant images are referred to images in the same category. Precision (Pcn) measures the hit-ratio that retrieved images fall in the same category while recall (Rcl) measures the capability of finding images of the same category inside the database. Below summarize two queries using picture 025 (African) and 285 (scenery).

Bin/C	App.	Operations	RR(%)	Speed	Pcn	Rcl
256Y	FS	768000	0	1	1.0	0.10
	PDS	309889	69.64	2.48	1.0	0.10
	DBPDS	187129	81.63	4.10	1.0	0.10
	DBPDS(0.040)	61100	94.02	12.57	1.0	0.10
	DBPDS(0.039)	64689	93.68	11.87	0.9	0.09
	DBPDS(0.021)	43253	95.77	17.76	0.7	0.07
1024Y	FS	3072000	0	1	1.0	0.10
	PDS	1465660	64.19	2.10	1.0	0.10
	DBPDS	811352	80.17	3.79	1.0	0.10
	DBPDS(0.028)	246799	93.97	12.45	1.0	0.10
	DBPDS(0.027)	258664	93.68	11.88	0.9	0.09
	DBPDS(0.01)	64579	98.42	47.47	0.7	0.07
256C	FS	2304000	0	1	1.0	0.10
	PDS	1033953	66.26	2.23	1.0	0.10
	DBPDS	955145	68.83	2.41	1.0	0.10
	DBPDS(0.077)	245633	92.00	9.38	1.0	0.10
	DBPDS(0.076)	254433	91.71	9.05	0.9	0.09
	DBPDS(0.050)	175120	94.30	13.16	0.7	0.07

Table I. Performance on query picture 025 (African).

Bin/C	App.	Operations	RR(%)	Speed	Pcn	Rcl
256Y	FS	768000	0	1	0.6	0.06
	PDS	532452	47.91	1.44	0.6	0.06
	DBPDS	320340	68.63	2.40	0.6	0.06
	DBPDS(0.026)	96862	90.53	7.93	0.6	0.06
	DBPDS(0.025)	92845	90.92	8.27	0.5	0.05
	DBPDS(0.011)	8192	99.20	93.75	0.5	0.05
1024Y	FS	3072000	0	1	0.5	0.05
	PDS	2368983	42.14	1.30	0.5	0.05
	DBPDS	1544335	62.27	1.99	0.5	0.05
	DBPDS(0.018)	401409	90.20	7.65	0.5	0.05
	DBPDS(0.017)	387418	90.54	7.93	0.4	0.04
	DBPDS(0.008)	214733	94.76	14.31	0.3	0.03
256C	FS	2304000	0	1	0.7	0.07
	PDS	1825291	40.53	1.26	0.7	0.07
	DBPDS	1724711	43.80	1.34	0.7	0.07
	DBPDS(0.086)	561465	81.72	4.10	0.7	0.07
	DBPDS(0.075)	479498	84.39	4.81	0.6	0.06
	DBPDS(0.050)	298180	90.29	7.73	0.4	0.04

Table II. Performance on query picture 285 (scenery).

From Table I-II, the proposed DBPDS algorithm achieves the same retrieval results as using FS approach in both Precision and Recall. It results in up to 4.10 times faster than using FS's. DBPDS(AMR) can even achieve much faster matching speed at different allowable matching range. They are about 3-5 times of its original DBPDS's (i.e. AMR=1.0). For example, DBPDS(0.028) is 12.45 times while DBPDS is 3.79 times for 1024-bin luminance histogram-based matching, i.e. >4 times. When the allowable range is reduced to a very small value, precision and recall rate will drop and traded for much faster matching speed.

5. Conclusion

A novel histogram-biasing factor (HBF) is proposed to measure the biasing-behavior of colors towards the dominant colors in sorted histograms. A dominance-biased partial distance search (DBPDS) is also proposed by selecting image with smaller HBFs as the sorted histogram for the computation order of distance accumulation in partial distance search. In addition, matching can also be limited to images with HBFs very closed. Experimental result shows that the DBPDS can achieve the same retrieval results for a query as in full search meanwhile providing up to 13 times speed improvement.

Acknowledgments

The work described in this paper was substantially supported by Strategic Research Grant from City University of Hong Kong, Hong Kong SAR, China. [Project No. 7001385].

References

[1] M. Flicker et al., "Query by image and video content: The QBIC system", *IEEE Computer*, pp.23-32, vol. 28, Sep 1995.

[2] M. Swain and D. Ballard, "Color indexing", *International Journal of Computer Vision*, vol., 7 no. 1, 1991.

[3] R. W. G. Hunt, "*Measuring Color*", Ellis Horwood Limited, England, 1987.

[4] K. M. Wong, C. H. Cheung and L. M. Po, "Merged-color histogram for color image retrieval", *Proc. in Int. Conf. Image Processing*, vol. III, pp.949-952, Sep 2002.

[5] C. H. Cheung, L. M. Po and K. M. Wong, "Web-based Beowulf-Class Parallel Computing on Image Database Indexing and Retrieval System", *Int. Sym. on Intelligent Multimedia, Video & Speech Processing*, pp.457-460, 2-4 May 2001, Hong Kong.

[6] J. Hafner, H. S. Sawhney, W. Equitz, M. Flicker and W. Niblack, "Efficient color histogram indexing for quadratic form distance functions", *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 729-736, July, 1995.

[7] A. P. Berman and L. G. Shapiro, "Efficient image retrieval with multiple distance measures", *Proc. SPIE Storage and Retrieval for Image and Video Database*, vol. 3022, pp.12-21, Feb. 1997.

[8] B. C. Song, M. J. Kim and J. B. Ra, "A fast multiresolution feature matching algorithm for exhaustive search in large image databases", *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, pp.673-678, May 2001.

[9] W. Li and E. Salari, "Successive elimination algorithm for motion estimation", *IEEE Trans. Image Processing*, vol. 4, pp. 105-107, Jan 1995.

[10] Y. Rui and T. S. Huang, "Image retrieval: current techniques, promising directions and open issues", *Journal of Visual Common. Image Representation*, vol. 10, pp.39-62, 1999.

[11] Y. Linde, A. Buzo and R. M. Gray, "An algorithm for vector quantizer design", *IEEE Trans. Commun.*, vol. COM-28, pp.84-89, Jan 1980.

[12] C. D. Bei and R. M. Gray, "An improvement of the minimum distortion encoding algorithm for vector quantization", *IEEE Trans. Commun.*, vol. COM-33, pp.1132-1133, Oct 1985.

[13] W. A. Martin, "Sorting", *ACM Computing Surveys*, vol. 3, no. 4, pp.147-174, Dec 1971.

[14] J. Wang and G. Wiederhold, "SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture Libraries", *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 23, no. 8, pp.1-17, Sep 2001.

[15] H. Müller, W. Müller, D. Squire, S. Marchand-Maillet and T. Pun, "Performance Evaluation in Content-Based Image Retrieval: Overview and Proposals," *Pattern Recognition Letters*, vol. 22, no. 5, pp. 593-601, 2001.