Content-based Retrieval of Compressed Images

Gerald Schaefer

Department of Computer Science
Loughborough University
Loughborough, U.K.
gerald.schaefer@ieee.org

Abstract. Content-based image retrieval allows search for pictures in large image databases without keyword or text annotations. Much progress has been made in deriving useful image features with most of these features being extracted from (uncompressed) pixel data. However, the vast majority of images today are stored in compressed form due to limitations in terms of storage and bandwidth resources. In this paper, we therefore investigate a different approach, namely that of compressed-domain image retrieval, and present some compressed-domain image retrieval techniques that we have developed over the past years. In particular, a method for retrieving images compressed by vector quantisation, that uses codebook information as image features, is presented. Retrieval of losslessly compressed images obtained using lossless JPEG, can be retrieved using information derived from the Huffman coding tables of the compressed files. Finally, CVPIC, a 4-th criterion image compression technique is introduced and it is demonstrated that compressed-domain image retrieval based on CVPIC is not only able to match the performance of common retrieval techniques on uncompressed images, but even clearly outperforms these.

Keywords: content-based image retrieval (CBIR), image compression, compressed-domain image retrieval, vector quantisation, lossless JPEG, CVPIC

1 Introduction

With the recent explosion in availability of digital imagery the need for content-based image retrieval (CBIR) is ever increasing. While many methods have been suggested in the literature, only few take into account the fact that - due to limited resources such as disk space and bandwidth - virtually all images are stored in compressed form. In order to process them for CBIR they first need to be uncompressed and the features calculated in the pixel domain. The desire for techniques that operate directly in the compressed domain providing, so-called midstream content access, is therefore evident [18].

In this paper, we introduce several techniques that perform compressed-domain image retrieval. In general there are two approaches. The first is based on existing compression techniques and tries to extract useful information from

J. Pokorný, V. Snášel, K. Richta (Eds.): Dateso 2010, pp. 175–185, ISBN 978-80-7378-116-3.

the compressed data streams produced by these. The second approach is to develop so-called 4-th criterion compression algorithms where the data in compressed form is directly visually meaningful and can hence be exploited for image retrieval. We will cover both approaches in this paper.

2 Content-based image retrieval

Since textual annotations are not available for most images, searching for particular pictures becomes an inherently difficult task. Luckily, a lot of research has been conducted over the last two decades leading to various approaches for content-based image retrieval [30, 2]. Content-based image retrieval (CBIR) does not rely on textual attributes but allows search based on features that are directly extracted from the images [30]. This however is, not surprisingly, rather challenging and often relies on the notion of visual similarity between images or image regions.

While humans are capable of effortlessly matching similar images or objects, machine vision research still has a long way to go before it will reach a similar performance for computers. Currently, many retrieval approaches are based on low-level features such as colour, texture, and shape features, leaving a 'semantic gap' to the high-level understanding of users [30].

3 Image compression and compressed-domain CBIR

Despite continuous advances in technology both storage space and bandwidth are still limited. In terms of the storage and transmission of images (e.g. through the Internet) this means that images have to be stored in compressed form. However, to achieve this compression some of the original image information needs to be sacrificed; that is, the compressed image will differ from the original image. Consequently, image descriptors obtained from compressed images will also be somewhat different from those derived from their uncompressed counterparts. In one of our studies [28], we investigated the effect image compression has on the performance of several popular CBIR techniques [33, 4, 32, 17, 6, 1]. We found that the resulting drop in retrieval performance is small yet not negligible [28].

Although most images exist only in compressed form, almost all CBIR techniques operate in the pixel domain. In contrast, compressed domain techniques operate directly on the compressed data without the need for decompression [13]. Compressed domain CBIR can be performed either based on existing compression formats such as JPEG [19] or vector quantisation [25], or employing so-called 4-th criterion compression techniques where the compressed information is directly visually meaningful [18].

4 Compressed-domain CBIR based on vector quantisation

4.1 Vector quantisation

Vector quantisation (VQ [5]) represents a mapping that assigns to each input vector a codebook vector achieving compression by setting the size of the codebook small relative to the possible gamut of input vectors. In particular, in terms of VQ image compression, an image is divided into a set of L-dimensional vectors I by splitting it into image blocks where each block forms a vector. A codebook C with N entries is then found. There are many ways how this can be achieved. In our approach we start with one codebook entry (the mean of the distribution) and then iteratively add new entries by identifying and splitting the cluster which has the largest variance. After the desired codebook size has been reached we apply the LBG algorithm [12] to optimise the generated codebook. Once a codebook is defined the input vectors can be mapped to vectors (codewords) of the codebook according to a nearest neighbour rule:

$$I_i \rightarrow C_j \text{ iff } d(I_i, C_j) \leq d(I_i, C_k) \; \forall \; C_k \in C \tag{1}$$

The respective image block can then be represented by an index to the closest codeword only.

4.2 CBIR through VQ codebook matching

Even though information is lost due to the compression, image retrieval based on VQ data not only provides information on the colour content, as do colour histograms for example, but also on the spatial information (encompassing textural and shape attributes) of the image, which is due to the image being divided into blocks and the blocks coded as a whole.

In our algorithm we use block sizes of 4×4 pixels thus giving vectors of length 48 for colour images. In contrast to previous methods which use a universal codebook for all images, and then base their retrieval technique on histograms [10] or binarised histograms [9] of codebook indices, codebooks were generated on a per image basis ensuring that image quality is high, even for a small number of codes. Also, this not only makes image distribution easier (the there is no need for codebook negotiation between encoder and decoder) but also guarantees that the information stored by the codevectors is optimally adapted for each image. Indeed, this is the key feature that is used in our technique. Because prototype codes encompass precise information about an image, images can be compared by using the content stored in their respective codebooks.

There are several ways to compare 2 L-dimensional point sets C_A and C_B. One choice would be to use the Hausdorff distance [8]. However, as this is based on a max-min operator, the original Hausdorff distance can become highly dependent on outliers and so is statistically not very robust. A better way to compare two VQ codebooks would therefore be to use a variant of the Hausdorff distance which shows more robustness. In our approach, we use a Modified Hausdorff distance $\mathrm{HD}_{\mathrm{mod}}$ defined as

$$\mathrm{HD}_{\mathrm{mod}} = \max(\mathrm{hd}_{\mathrm{mod}}(C_A, C_B), \mathrm{hd}_{\mathrm{mod}}(C_B, C_A)) \tag{2}$$

with

$$\mathrm{hd}_{\mathrm{mod}}(C_A, C_B) = \frac{1}{N} \sum_{i=1}^{N} \min_j \|C_A(i) - C_B(j)\| \tag{3}$$

where $\|.\|$ denotes some underlying norm, in our case the L_2 norm. Rather than taking the maximum of the minima as in the original Hausdorff distance we use the average of the minimum which makes the distance measure less sensitive to outliers [3, 26].

After calculating the distances to all images in the database, the images can be ranked in order of their similarity to a given query image.

4.3 Experimental results

We performed VQ image retrieval on the MPEG-7 Common colour dataset [15]. This database consists of 5466 images and a set of 50 queries with predefined ground truth images. We compressed the images with codebooks of size 64, and performed image retrieval based on the Modified Hausdorff distances between the VQ codebooks. We use the MPEG-7 Normalised Modified Retrieval Rank (NMRR) [15] as the standard performance measure for this data set. The NMRR is defined as

$$\mathrm{NMRR} = \frac{\mathrm{MRR}(q)}{K + 1/2 - N_G/2} \tag{4}$$

where $\mathrm{MRR}(q) = \mu(q) - 1/2 - N_G(q)/2$ and $\mu(q) = \sum_{i=1}^{N_G(q)} r_i / N_G(q)$. $N_G(q)$ is the number of ground truth images for the qth query image and r_i denotes the retrieved rank. For K we use the MPEG-7 recommendation $K = \min(4N_G(q), 2\max_q(N_G(q)))$.

The results achieved give an average NMRR of 0.1196. In comparison, image retrieval based on colour histograms [33] results in an average NMRR of 0.1075. The slight drop in performance can be explained with the fact that we are essentially compressing the already severely (JPEG) compressed images of the MPEG-7 dataset again and hence part of the information stored in the VQ codebooks can be attributed to compression artefacts rather than to image content.

5 Compressed-domain CBIR based on lossless JPEG

5.1 Lossless JPEG compression

Predictive image coders work on the basis that images tend to change slowly over most areas of an image. Consequently, most neighbouring pixels will have similar values. A pixel at location (i, j) is predicted, based on the values of its neighbouring pixels as

$$P'_{(i,j)} = \sum_{k<i,l<j} \omega_{(k,l)} P_{(k,l)} \tag{5}$$

where P' represents the prediction, P are the actual pixel values of the neighbouring pixels, and ω describe weights used for the prediction. In this paper we adapt one of the predictors of the lossless JPEG [34] scheme, in particular, the JPEG-7 predictor where pixels are predicted as the the average of their top and left neighbours (i.e. $\omega_{(i-1,j)} = \omega_{(i,j-1)} = 0.5$).

Once a pixel has been predicted it is encoded as the difference between its actual value and its prediction:

$$D_{(i,j)} = P_{(i,j)} - P'_{(i,j)} \tag{6}$$

This has the advantage that now differences close to 0 are much more likely than higher differences. Consequently, an entropy encoding stage, which assigns shorter codewords to more frequent codes and longer codewords to rarer events, is then applied. In our framework we use a Huffman coder [7] for performing the entropy coding. Huffman coders are optimal in the sense that they allow encoding data using the minimal number of bits (with the restriction that each codeword has an integer number of bits).

The losslessly compressed image then comprises two parts: the Huffman table, and the differences now represented as indices into the Huffman table.

5.2 CBIR in the losslessly compressed domain

Difference histogram In order to find a way to index the compressed images directly in the encoded domain, we first reverse the entropy coding stage. This is also being done by all other methods that operate in the compressed domain where entropy coding is part of the compression algorithm [13]. After this, we naturally end up with the difference data $D_{(i,j)}$ for each pixel. We now want to make explicit what this data actually means. The prediction of each pixel is essentially a statistical description of its neighbourhood. By calculating the difference to the actual pixel value, the resulting descriptor $D_{(i,j)}$ represents the change of the pixel compared to its neighbourhood. Texture can be defined as a property that pixels exhibit in comparison to their neighbourhood. Therefore, the differences between the predictions and the actual pixel values also define a description of the textural properties of the image.

Hence, we propose to use the difference data directly as a description of the image content. Building histograms of the differences seems to provide a good choice. However, one has to be aware that the distribution of the prediction differences is not uniform. Differences close to 0 are much more likely than higher values. To rectify this we first apply a non-linear transformation to the predictor differences:

$$D'_{(i,j)} = \begin{cases} -M - \log(-D_{(i,j)}) & \text{if } D_{(i,j)} < 0 \\ 0 & \text{if } D_{(i,j)} = 0 \\ M + \log(D_{(i,j)}) & \text{if } D_{(i,j)} > 0 \end{cases} \tag{7}$$

where $M = \log(1/255)$, i.e. the transformed value of the smallest prediction difference possible.

After this transformation we build a uniformly quantised histogram of the D's. Once histograms H_i are built, they can be compared using histogram intersection, as described in [33]

$$d(H_1, H_2) = \sum_k \min\{H_1(k), H_2(k)\} \tag{8}$$

where H_1 and H_2 are the histograms of the D' coefficients. Image retrieval is performed by calculating the distances between a query image and all images in the database, and returning the closest matches.

Codebook matching As we have mentioned above, all algorithms to date that operate in the compressed domain of images need to reverse the entropy coding as a first step. We will now introduce a technique that, based on the predictive coding framework outlined in Section 5.1, allows for image indexing directly in the compressed image data without the need to undo the entropy coding. In particular, we will use the Huffman codebook itself as the index.

The Huffman codebook contains one codeword for each possible difference in the interval $[-255; 255]$. Shorter codewords are assigned to events that are more probable. Consequently, the length of a Huffman codeword is indirectly proportional to its frequency in the image. That is, the codebook contains approximately the same information as a histogram of the data! Hence to compare two images, one can compare their codebooks. To do this we calculate the cumulative difference of codeword lengths

$$d(B_1, B_2) = \sum_{-255<k<255} |B_1(k) - B_2(k)| \tag{9}$$

where B_1 and B_2 are the Huffman codebooks of two images, and $|.|$ is the L_1 norm. Codewords that are not present in a codebook are assigned the maximum length that is to be found in the respective codebook before the comparison.

5.3 Experimental Results

We took 80 images of the VisTex [14] image set, a collection of colour texture images from MIT, and extracted from each of the 512×512 pixel images two 256×256 non-overlapping regions. One of each was assigned model while the others represent the query images. In order to acquire the following results, each query image was compared to each model image, and as we know which one is the corresponding picture the rank in which the correct image is retrieved can be recorded. As performance measure, we use the match percentile [33] defined as

$$MP = \frac{N - R}{N - 1} \tag{10}$$

or rather the average match percentile over all query images as a measure of goodness for assessing retrieval performance. Here N is the number of model images in the database, and R is the rank, i.e. the position of the correct match in the retrieval list.

We encoded all images using the JPEG-7 predictor and Huffman compression as explained in Section 5.1. As we deal with RGB images, each channel was coded separately. Image retrieval was then performed by computing the difference histograms ($35 \times 35 \times 35$ bins) as defined in Section 5.2 and calculating the histogram intersection (Equation (8)) of each query image to each of the models, before the returned models were ranked according to their distance to the query. The average match percentile achieved over the whole dataset is 98.20 with 88.75% of the correct images retrieved in 1st rank.

In order to compare this performance, we also applied the rotation invariant version of the LBP operator [16] to the images. LBP has been shown to represent a powerful texture classification technique that outperforms most other standard texture algorithms [16]. The average match percentile, based on the resulting $36 \times 36 \times 36$ LBP histograms is 98.50 (92.50% 1st rank retrievals). Hence we see that our proposed algorithm performs comparable to current state-of-the-art techniques.

Finally, we also evaluated the performance of our codebook matching algorithm from Section 5.2 on the VisTex dataset. The result is an average match percentile of 96.36 with 67.50% first rank retrievals. We see confirmed what we suspected, namely that the performance drops due to the inexactness of the representation, and also due to the lack of measurement of correlation between the channels (which for 3-dimensional histogram is preserved). However, a match percentile of more than 96 is still a very good basis to reject most of the images and leave only those that are close to the query.

6 Compressed-domain CBIR based on CVPIC

6.1 Colour Visual Pattern Image Coding

Colour Visual Pattern Image Coding (CVPIC) divides an image into small 4×4 pixel blocks and then matches each block to one of a pre-defined classes of patterns (14 edge patterns shown in Figure 1, plus a uniform block, i.e. a block without an edge), followed by quantisation of the colour information. The compressed data stream contains direct information about colour and shape information of the image, and we have introduced various techniques for performing compressed domain CBIR in the CVPIC domain [21, 20, 22, 24, 23]. In here, we focus on the approach presented in [23].

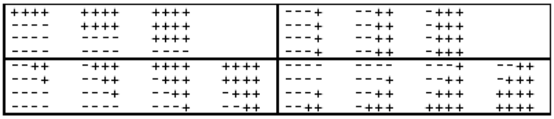

Fig. 1. The 14 edge patterns used in CVPIC [27].

6.2 CBIR based on CVPIC data

The data that is readily available in CVPIC compressed images is the colour information of each of the 4×4 image blocks, and information on the spatial characteristics of each block, in particular on whether a given block is identified as a uniform block (a block with no or little variation) or a pattern block (a block where an edge or gradient has been detected). Furthermore, each pattern block is assigned to one of 14 universally predefined classes according to the orientation and position of the edge within the block. We make direct use of this information to derive an image retrieval algorithm that utilises both colour and shape information. The colour information is summarised similar to colour coherence vectors introduced in [17] and the border/interior pixel approach in [31] which both show that dividing the pixels of an image into those that are part of a uniform area and those that are not can improve retrieval performance.

In essence we create two colour histograms, one for uniform blocks and one for non-uniform (pattern) blocks. Shape descriptors are often calculated as statistical summaries of local edge information such as in [11] where the edge orientation and magnitude is determined at each pixel location and an edge histogram calculated. Exploiting the CVPIC image structure, an effective shape descriptor can be determined very efficiently. Since each (pattern) block contains exactly one (pre-calculated) edge and there are 14 different patterns we simply build a 1×14 histogram of the edge indices. CVPIC image retrieval based on both colour and shape features can then be performed by calculating the combined difference (a weighted L_1 norm) between all three histograms.

Table 1. Retrieval results obtained on the UCID dataset in terms of modified average match percentile [29].

	MP
Colour histograms [33]	90.47
Colour coherence vectors [17]	91.03
Border/interior pixel histograms [31]	91.27
Colour correlograms [6]	89.96
CVPIC	94.24

Fig. 2. Sample query together with top five ranked images returned by (from top to bottom) colour histograms, colour coherence vectors, border/interior pixel histograms, colour correlograms, and CVPIC retrieval.

6.3 Experimental results

Results on the UCID dataset [29], shown in Table 1 in terms of match percentile, confirm that this approach is not only capable of performing efficient and effective compressed domain image retrieval but that our algorithm also outperforms various popular CBIR techniques, even when these are run on uncompressed images. An example query with the top five retrieved images obtained from several pixel domain CBIR methods and our CVPIC techniques is shown in Figure 2.

7 Conclusions

Compressed-domain image retrieval provides an interesting alternative to common image retrieval algorithms as it provides the advantage that image features are extracted directly from the compressed data stream of images. In this paper we have presented several compressed domain techniques that allow efficient and effective querying of large image databases. In particular, we have presented techniques based on vector quantisation, lossless JPEG compression, and CVPIC, a 4-th criterion compression algorithm. Experimental results have shown that the introduced techniques are able to match or even exceed the performance of common pixel-based retrieval algorithms.

References

1. L. Cinque, S. Levialdi, and A. Pellicano. Color-based image retrieval using spatial-chromatic histograms. In *IEEE Int. Conf. Multimedia Computing and Systems*, pages 969–973, 1999.
2. R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40(2):1–60, 2008.
3. M-P. Dubiusson and A.K. Jain. A modified Hausdorff distance for object matching. In *12th IEEE Int. Conference on Pattern Recognition*, pages 566–568, 1994.
4. C. Faloutsos, W. Equitz, M. Flickner, W. Niblack, D. Petkovic, and R. Barber. Efficient and effective querying by image content. *journal of Intelligent Information Retrieval*, 3(3/4):231–262, 1994.
5. R.M. Gray. Vector quantization. *IEEE ASSP Magazine*, 1(2):4–29, 1984.
6. J. Huang, S.R. Kumar, M. Mitra, W-J. Zhu, and R. Zabih. Image indexing using color correlograms. In *IEEE Int. Conference Computer Vision and Pattern Recognition*, pages 762–768, 1997.
7. D.A. Huffman. A method for the construction of minimum redundancy codes. *Proceedings of the Institute of Radio Engineers*, 40:1098–1101, 1952.
8. D.P. Huttenlocher, G.A. Klanderman, and W.J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993.
9. F. Idris and S. Panchanathan. Storage and retrieval of compressed images. *IEEE Trans. Consumer Electronics*, 41(3):937–941, 1995.
10. F. Idris and S. Panchanathan. Image and video indexing using vector quantization. *Machine Vision and Applications*, 10:43–50, 1997.
11. A.K. Jain and A. Vailaya. Image retrieval using color and shape. *Pattern Recognition*, 29(8):1233–1244, 1996.
12. Y. Linde, A. Buzo, and R.M. Gray. An algorithm for vector quantizer design. *IEEE Trans. Communications*, 28:84–95, 1980.
13. M. Mandal, F. Idris, and S. Panchanathan. A critical evaluation of image and video indexing techniques in the compressed domain. *Image and Vision Computing*, 17(7):513–529, 1999.
14. MIT. VisTex Vision Texture database. `http://www-white.media.mit.edu/vismod/imagery/VisionTexture/vistex.html`.
15. Moving Picture Experts Group. Description of core experiments for MPEG-7 color/texture descriptors. Technical Report ISO/IEC JTC1/SC29/WG11/ N2929, 1999.
16. T. Ojala, M. Pietikäinen, and T. Menpää. Gray scale and rotation invariant texture classification with local binary patterns. In *6th European Conference on Computer Vision*, pages 404–420, 2000.

17. G. Pass and R. Zabih. Histogram refinement for content-based image retrieval. In *3rd IEEE Workshop on Applications of Computer Vision*, pages 96–102, 1996.
18. R.W. Picard. Content access for image/video coding: The fourth criterion. Technical Report 195, MIT Media Lab, 1994.
19. G. Schaefer. JPEG image retrieval by simple operators. In *2nd International Workshop on Content-Based Multimedia Indexing*, pages 207–214, 2001.
20. G. Schaefer. Mistream content access of visual pattern coded imagery. In *4th Int. Workshop on Multimedia Data and Document Engineering*, 2004.
21. G. Schaefer and S. Lieutaud. Colour and shape based image retrieval for CVPIC coded images. In *Int. Conference on Imaging Science, Systems, and Technology*, pages 456–461, 2004.
22. G. Schaefer and S. Lieutaud. CVPIC based uniform/non-uniform colour histograms for compressed domain image retrieval. In *7th Int. Conference on VISual Information Systems*, pages 344–348, 2004.
23. G. Schaefer and S. Lieutaud. CVPIC compressed domain image retrieval by colour and shape. In *Int. Conference on Image Analysis and Recognition*, volume 3211 of *Springer Lecture Notes on Computer Science*, pages 778–786, 2004.
24. G. Schaefer, S. Lieutaud, and G. Qiu. CVPIC image retrieval based on block colour co-occurance matrix and pattern histogram. In *IEEE Int. Conference on Image Processing*, pages 413–416, 2004.
25. G. Schaefer and W. Naumienko. Midstream content access by VQ codebook matching. In *Int. Conference on Imaging Science, Systems, and Technology*, pages 170–176, 2003.
26. G. Schaefer, G. Qiu, and G. Finlayson. Retrieval of palettised colour images. In *Storage and Retrieval for Image and Video Databases VIII*, volume 3972 of *Proceedings of SPIE*, pages 483–493, 2000.
27. G. Schaefer, G. Qiu, and M.R. Luo. Visual pattern based colour image compression. In *Visual Communication and Image Processing '99*, volume 3653 of *Proceedings of SPIE*, pages 989–997, 1999.
28. G. Schaefer and S. Ruszala. Effective and efficient browsing of image databases. *Int. Journal of Imaging Systems and Technology*, 18:137–145, 2008.
29. G. Schaefer and M. Stich. UCID - An Uncompressed Colour Image Database. In *Storage and Retrieval Methods and Applications for Multimedia 2004*, volume 5307 of *Proceedings of SPIE*, pages 472–480, 2004.
30. A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(12):1249–1380, 2000.
31. R.O. Stehling, M.A. Nascimento, and A.X. Falcao. A compact and efficient image retrieval approach based on border/interior pixel classification. In *Proc. 11th Int. Conf. on Information and Knowledge Management*, pages 102–109, 2002.
32. M. Stricker and M. Orengo. Similarity of color images. In *Conf. on Storage and Retrieval for Image and Video Databases III*, volume 2420 of *Proceedings of SPIE*, pages 381–392, 1995.
33. M.J. Swain and D.H. Ballard. Color indexing. *Int. Journal of Computer Vision*, 7(11):11–32, 1991.
34. G.K. Wallace. The JPEG still picture compression standard. *Communications of the ACM*, 34(4):30–44, 1991.