CONTENT-BASED IMAGE RETRIEVAL USING JOINT CORRELOGRAMS

Adam Williams and Peter Yoon

Department of Computer Science
Trinity College
Hartford, CT 06106 USA

ABSTRACT

The comparison of digital images to determine their degree of similarity is one of the fundamental problems of computer vision. Many techniques exist which accomplish this with a certain level of success, most of which involve either the analysis of pixel-level features or the segmentation of images into sub-objects that can be geometrically compared. In this paper we develop and evaluate a new variation of the pixel feature and analysis technique known as the color correlogram in the context of a content-based image retrieval (CBIR) system. Our approach is to extend the autocorrelogram by using multiple image features in the same way the joint histogram extends the color histogram. The experiment shows that the joint correlogram indexing method gives a significant improvement over histogram or color-only correlogram indexing, and it is also memory-efficient.

1. INTRODUCTION

The problem addressed by a CBIR system is simple: given a query image, find the target images which appear most similar to the input image, based solely on the content of the images, that is, with no text information involved. We will use a combination of the join histogram and color correlogram techniques described in [8] for image comparison and ranking.

The applications of CBIR systems are very broad, including everything from law enforcement to medical diagnostics. In this paper we focus on developing a new technique for general CBIR that is suitable for use with a heterogeneous set of images. For testing the system, 24,000 jpeg images of varying content have been obtained from the Berkeley Digital Library Project.

The main contribution of this paper is the development and evaluation of a new technique for image feature analysis and comparison called the joint correlogram. This is done in the context of a CBIR system which ideally performs well enough to be used in real-world applications. A CBIR system is generally composed of two main components: the indexer and the query engine.

Effective indexing is the most important part of any CBIR system. Care must be taken so that each image's index entry captures the distinguishing features of an image while using only a small amount of storage. In this system, the common approach of storing a feature vector for each image is adopted since it allows for the use of standard matrix structures and measures of distance in comparisons. In this paper, we are interested in developing a new technique for feature analysis which is an extension of two older techniques known generally as histograms and correlograms.

The color histogram is one of the first feature analysis techniques used in image retrieval. Color histograms record the distribution of colors in an image, and their effectiveness is based on the fact that similar images will have similar distributions. They are also tolerant to rotation and movement of objects within an image. This technique is moderately effective in practice. An extension of this method, known as a joint histogram, uses other pixel features such as texturedness and gradient intensity in addition to color when computing histograms. Joint histograms are generally two to three times more effective than simple color histograms. The reason for the improvement is clear: while dissimilar images may have similar color distributions, it is less likely that the joint distributions of multiple features will be similar. This is an important concept which motivates the new approach taken in this paper.

The color correlogram technique is similar in spirit to the color histogram. The difference is that the correlogram also includes spatial information about the colors. Specifically it records the probability of finding a pixel of a certain color at a certain distance from a given pixel of another color. Using this general approach with all possible color combinations is computationally expensive, so in practice just autocorrelograms are used [10]. Autocorrelograms only involve distances from pixels to those of the same color. This technique has been shown to be superior in recall to color histograms and color coherence vectors although it is not compared to joint histograms. The autocorrelogram is

This material is based upon work supported by the Trinity College Research Grant Program and NASA EPSCoR Core Funding Program.

the other piece our approach.

2. APPROACH

Our approach is to extend the autocorrelogram by using multiple image features in the same way the joint histogram extends the color histogram (hence "joint correlogram"). Specifically, the joint correlograms will be computed using color, texturedness, gradient magnitude, and rank, all as defined in [8]. It is expected that using the additional feature information will increase the recall accuracy just as joint histograms improve over color histograms, although not necessarily in as dramatic as fashion. This technique will be compared to the standard correlogram and joint histogram techniques in a manner described later.

2.1. Indexing

Before the images can be indexed, the features must be quantized into smaller spaces. For example, a 24-bit image in the RGB color space has over 16 million color possibilities, which would require an unnecessarily large amount of storage for the histogram. The solution is to map each color into a smaller space; this must be done in such a way that each discrete value appears equally as often as the others [3], that is, based on the probability distribution of possible values. This must be done for each feature to be indexed. Also, a small distance value must be chosen for the correlogram computation. Small values are most efficient and effective.

The first step in indexing an image is the preprocessing phase, which will be described in the next section. This involves decompressing the image from whatever format is is stored in and loading it into memory so that it can be operated on. Next, the image must be normalized or scaled to a predetermined size so that different sized images can be accurately compared.

Once an image is in memory and has been scaled, its features can be analyzed. For the sake of comparison and also for a querying efficiency reason which will be described later, both the joint histograms and joint correlograms of the images will be computed and stored on disk. Methods for computing joint histograms and color correlograms are described in [9]; joint correlograms are computed using similar methods. While the size of the histograms and correlograms are not particularly large, a sparse matrix storage structure can be used to reduce storage requirements significantly. It is generally preferable to store histogram entries in some kind of sorted order for querying purposes [9]. The histograms and correlograms are computed once for each image, generally in a large batch job. New images can be added by performing the computation on just the new images. Once the images have been indexed, they are ready for querying.

2.2. Preprocessing

Let X be an $m \times n$ input image data matrix. We assume that $m < n$. Before an image is stored, it must be normalized and decompressed. This can be achieved by singular value decomposition (SVD) of the data matrix [6]. The SVD of X is given by

$$X = U \Sigma V^T \tag{2.1}$$

where $U \in \mathcal{R}^{m \times m}$ and $V \in \mathcal{R}^{n \times n}$ are orthogonal matrices, and $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_m) \in \mathcal{R}^{m \times n}$. Then we can partition U and V as follows:

$$U = [U_r \; \bar{U}_r], \quad U_r \in \mathcal{R}^{m \times r}, \bar{U}_r \in \mathcal{R}^{m \times (m-r)} \tag{2.2}$$

$$V = [V_r \; \bar{V}_r], \quad V_r \in \mathcal{R}^{n \times r}, \bar{V}_r \in \mathcal{R}^{n \times (n-r)} \tag{2.3}$$

where r is approximate rank of X. The preprocessing transformation T is then given by

$$T = \Sigma^{+} [U_r \; 0]^T \tag{2.4}$$

where Σ^{+} denotes an $n \times m$ pseudo inverse matrix

$$\Sigma^{+} = \text{diag}(\sigma_1^{-1}, \ldots, \sigma_r^{-1}) \tag{2.5}$$

Then the preprocessed data is then given by

$$\tilde{X} \overset{\text{def}}{=} TX = \Sigma^{+} [U_r \; 0]^T [U_r \; \bar{U}_r] \Sigma [V_r \bar{V}_r]^T = [V_r \; 0]^T. \tag{2.6}$$

As (2.5) shows, the SVD normalizes and the data and reduces dimensionality of data by using only left singular vectors and correspond to r largest singular values.

These preprocessed data are further processed by subsequent indexing algorithms. The SVD preprocessor accomplishes the objectives of normalization and dimensionality reduction in one step. This preprocessing is necessary for theoretical reasons and significantly speeds up the indexing phase. The SVD also provides most complete analysis of the input data matrix X. In spite of many advantages of SVD, it has several computational shortcomings. First of all, it is a computationally intensive, requiring $\mathcal{O}(n^3)$ operations to compute and modify the existing decomposition. In addition the block processing algorithms do not have stable updating algorithms when successive changes in the image collection are required. To overcome these shortcomings we propose to apply the ULV decomposition (ULVD) to the preprocessing step. ULVD has been used previously in signal and image processing [7]. However, it is worthwhile to exploit recent developments of ULVD methods for image processing applications.

Let X have the factorization

$$X = U_0 C V_0^T, \tag{2.7}$$

where $U_0 \in \mathcal{R}^{m\times m}, V_0 \in \mathcal{R}^{n\times n}$ orthogonal,

$$C = \begin{array}{c} \\ r \\ p-r \\ m-p \end{array} \begin{array}{c} \begin{array}{ccc} r & p-r & n-p \end{array} \\ \begin{pmatrix} L & 0 & 0 \\ F & G & 0 \\ 0 & 0 & 0 \end{pmatrix} \end{array} \tag{2.8}$$

$$\|L^{-1}\|_2^{-1} \geq \epsilon \geq \|G\|_2, \;\; \|F\|_F\|L^{-1}\|_2 = \eta \ll 1. \tag{2.9}$$

Here L and G are lower triangular matrices, and $\|\cdot\|_2$ and $\|\cdot\|_F$ denote the Euclidean and the Frobenius norms, respectively. The value ϵ is the threshold value. The value r is usually referred to as ϵ-rank or the rank "revealed" by the ULVD. It is not numerical rank unless ϵ is very close to machine precision times the norm of X, and, in practice, ϵ will often be larger than that value. Thus the conditions in (2.9) assume a significant gap between the r^{th} singular value of C and its $(r+1)^{st}$ singular value. Both L and G are mathematically nonsingular and p is the mathematical rank of X. That is, the last $n-p$ columns and the last $m-p$ rows of L are known to be zero.

The ULVD of X can be obtained by computing its QL factorization $X = Q[L_0 \; 0]$ where $Q \in \mathcal{R}^{m\times m}$ is orthogonal and $L_0 \in \mathcal{R}^{n\times n}$ lower triangular, followed by computing the ULVD of L_0 using the deflation technique described in [11].

Now we partition V_0 as follows:

$$V_0 = [V_1, V_2], V_1 \in \mathcal{R}^{n\times r}, V_2 \in \mathcal{R}^{n\times(n-r)} \tag{2.10}$$

Fierro and Bunch [5] showed how close the subspace spanned by the columns of V_1 is to that of V_r in (2.3). They showed that the subspaces spanned by V_r and V_1 are close as long as we can keep $\|F\|_2$ small enough. A number of strategies to do that have been proposed in [4, 11].

We now parameterize the matrix X with the time step t. Then for each time step of t, the ULVD of $X(t+1)$ can be formed from that of $X(t)$ by adding a new image to the existing collection, thus performing an *update* operation on C, and deleting an old image, thus performing a *downdate*.

An algorithm for updating is described in [11]. A stable algorithm for downdating is given in [1, 2, 12], that does not require keeping U_0. The authors of [2] also give a thorough discussion of the stability and regularity issues for downdating the ULVD. Both procedures require only $O(p^2)$ operations and are thus considerably faster than modifying the SVD. The algorithms described in [1, 12] are as robust as the ones in [2, 11], but are much more automatic.

2.3. Querying

The querying approach that the system uses is a two-pass approach described in [10]. Also known as *filtering*, it involves retrieving a subset of the database using a more efficient but perhaps less effective algorithm (joint histogram comparison in this case), and then using the joint correlograms to improve the ranking accuracy of the query. A similar approach has been shown to be effective when first using color histograms and then color correlograms for refinement [9]. When evaluating the joint correlogram, filtering will not be used in order to make comparison with other methods more direct. Another important feature of the query system is the distance measure. There are several choices presented in [8].

The system is accessible via a simple Web interface. An example image can be selected from the database and queried upon, and query results can be browsed. The interface will also allow the user to select which indexing method to use in the query. This will make it easier to evaluate the joint correlogram's performance.

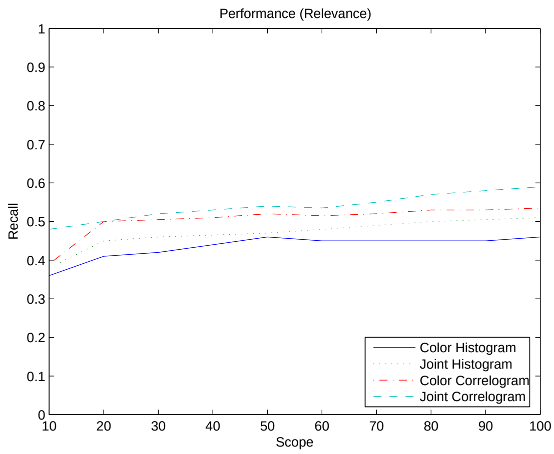

Fig. 1. Performance Evaluation of Four CBIR systems

3. PERFORMANCE EVALUATION AND CONCLUSION

In evaluating and comparing CBIR systems, the standard measure is scope vs. recall. Score refers to how many images deep into the results on has to look before finding the "correct" result (the correct image is determined by human comparison). Thus in scope vs. recall analysis, one is interested in what percentage of correct images appear in a given scope for several different test queries. Ideally, 70%+ of correct images should appear in a scope of 10 (that is, within the first 10 images returned by a query). This analysis allows for a simple and practical comparison of several image summary techniques. It will be used to evaluate the performance of the joint correlogram and compare it to joint histograms and color correlograms.

The joint correlogram method was tested against the color and joint histogram and the color correlogram methods using scope vs. recall analysis. Approximately 30 images were used to query the system using each of the four methods; the best matches for these images were determined manually beforehand. As shown in Figure 1, approximately 63% of the best matching images appeared within the first 100 results returned by the system when using the joint correlogram method. Recall percentages of 61%, 56%, and 53% were achieved by the color correlogram, joint histogram, and color histogram methods, respectively. The speed of each query is nearly identical regardless of what method is used, however the memory requirements for the joint correlogram are somewhat greater than for the color histogram (65 MB versus 5 MB for 24,000 iamges). Naturally, it also takes longer to compute the joint correlogram index, although dynamic programming techniques can be used to speed it up. Overall, we have determined that the joint correlogram indexing method gives a noticeable improvement over histogram or color-only correlogram indexing, and its time and space requirements are still more than practical.

4. REFERENCES

[1] J.L. Barlow, H. Erbay, and Z. Zhang. A modified Gram-Schmidt based downdating technique for the ULV decomposition with applications to recursive TLS problems. In F.T. Luk, editor, *Proceedings of Adv. Sig. Proc. Alg., Arch. and Impl. IX,*, volume 3807, pages 247–257, Bellingham, WA, August 1999. SPIE.

[2] J.L. Barlow, P.A. Yoon, and H. Zha. An algorithm and a stability theory for downdating the ULV decomposition. *BIT*, 36:14–40, 1996.

[3] C.-Y. Chang, A.A. Maciejewski, and V. Balakrishnan. Fast eigenspace decomposition of correlated images. *IEEE Transactions on Image Processing*, 9:1937–1949, 2000.

[4] H. Erbay and J.L. Barlow. Recursive ULV decomposition. In F.T. Luk, editor, *Proceedings of Adv. Sig. Proc. Alg., Arch. and Impl. X*, volume 4116, pages 157–166, Bellingham, WA, August 2000. SPIE.

[5] R.D. Fierro and J.R. Bunch. Bounding the subspaces from rank revealing two-sided orthogonal decompositions. *SIAM Journal on Matrix Analysis and Applications*, 16:743–759, 1995.

[6] G.H. Golub and C.F. Van Loan. *Matrix Computations, Third Edition*. The Johns Hopkins Press, Baltimore, MD, 1996.

[7] S. Hosur, A.H. Tewfik, and D. Boley. ULV and generalized ULV subspace tracking adaptive algorithms. *IEEE Transactions on Signal Processing*, 46:1282–1297, 1998.

[8] J. Huang, S.R. Kumar, M. Mitra, W-J Zhu, and R. Zabih. Image indexing using color correlograms. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition*, pages 762–768, 1997.

[9] G. Pass and R. Zabih. Histogram refinement for content-based image retrieval. In *Proceedings of the 3rd IEEE Workshop on Applications of Computer Vision*, pages 96–102, 1996.

[10] G. Pass, R. Zabih, and J. Miller. Comparing images using color coherence vectors. In *Proc. ACM Intern. Conf. Multimedia*, pages 65–73, 1996.

[11] G.W. Stewart. Updating a rank-revealing ULV decomposition. *SIAM Journal on Matrix Analysis and Applications*, 14:494–499, 1993.

[12] P.A. Yoon and J.L. Barlow. An efficient rank detection procedure for modifying the ULV decomposition. *BIT*, 38:781 – 801, 1998.