Percentile Blobs for Image Similarity

Nicholas R. Howe
Cornell University
Department of Computer Science
Ithaca, NY 14853
nihowe@cs.cornell.edu

Abstract

We present a new algorithm called PBSIM for computing image similarity, based upon a novel method of extracting bloblike features from images. In tests on a classification task using a data set of over 1000 images, PBSIM shows significantly higher accuracy than algorithms based upon color histograms, as well as previously reported results for another approach based upon bloblike features.

1 Introduction

As multimedia applications become more commonplace, the need increases for tools to manipulate large collections of visual data. One important tool that has already been the focus of significant research is an algorithmic process for determining the perceptual similarity of images. Such a tool can form the basis of many different image processing systems, including those for automatic classification of video or image data, retrieval of similar images from databases, and many other related and important tasks. For example, many image retrieval systems make use of a suite of similarity algorithms [1, 8, 11].

Broadly speaking, previous work in image similarity falls into one of several categories according to the basic approach used. Some algorithms take a geometric approach to similarity, attempting to construct models of the contents of the images for comparison [5]. Others, including work in face recognition and algorithms based on eigenvalues [11], rely on identifying parts of an image based upon their appearance. A third area of work in image similarity, developed primarily for use with large databases of images, revolves around comparing general properties of entire images. Typical examples of this holistic style are the various methods using color histograms.

Holistic approaches to image similarity have tended to be somewhat simplistic, concentrating only on one property of the image, such as color or texture. With a few exceptions, they ignore the spatial layout of the image entirely. Recently several algorithms have been proposed that attempt to integrate color and texture with spatial information. Color correlograms [7] incorporate both color and geometry. Recent work in blob-based similarity [3] and flexible templates [9] has explicitly attempted to unify all three sources of similarity information. Each of these approaches has specific strengths and weaknesses that are discussed further in Section 3.1.

The algorithm presented here, called *percentile blob-based similarity* or PBSIM, incorporates information about the spatial layout of color and texture in images in an intuitive and straightforward manner. Its relative simplicity means that it is practical for application to large databases. Furthermore, simplicity may be responsible in part for the reliability of the technique as demonstrated in our tests. We evaluate the performance of our algorithm using a data set containing over 1000 images from twelve different categories, based on its ability to select similar images from the same category. The classification accuracy on this data set is significantly higher than previously published results using other methods, including methods based upon color histograms.

In the next section we describe the PBSIM algorithm, including both feature extraction from the images and the comparison process. Section 3 gives the results of tests performed using the sample data set and compares these to the performance of related techniques. Finally we conclude with some observations about the algorithm and directions for future work.

2 Algorithm

The PBSIM algorithm may be divided into two parts. The first stage extracts features from each image for use in comparison by the second stage. The features are characterized and the descriptions stored for later processing. In the second stage, when two images are being compared, their stored feature descriptions are looked up and contrasted using a variant of the L_1 distance measure. The end result is a difference score for the two images, which may be inverted to find the similarity.

2.1 Feature Extraction

The feature extraction algorithm operates on a grayscale rendition of the image. Three such renditions, or *portraits,* come from the YUV components of the image, while a fourth is a simple texture map. This texture map is produced by computing the difference at each pixel between the pixel and a weighted mean of its immediate neighbors in all directions. Each portrait is analyzed to identify large regions of the lowest and highest intensity, and these regions are described according to statistics that characterize the region's size, shape, and other attributes. Figure 1 shows steps in the processing of the luminosity portrait of an image of a sunset.

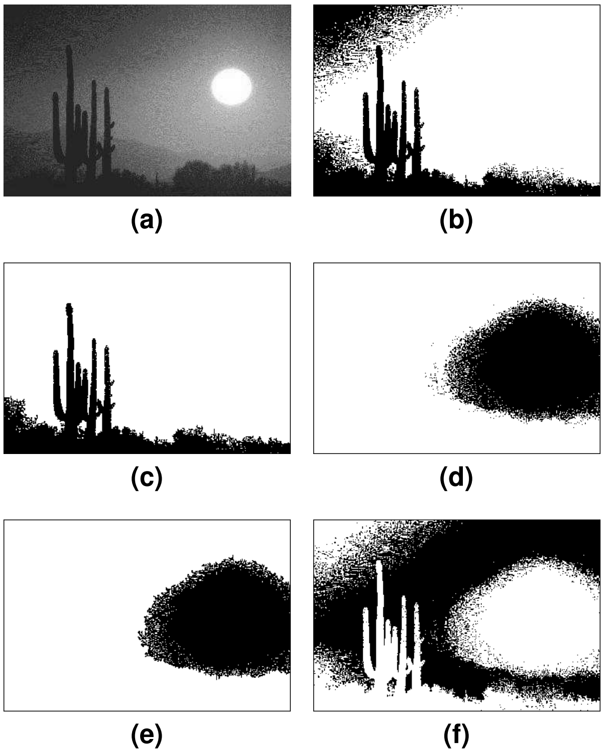

Figure 1. Steps in processing the luminosity portrait of an image. (a) Grayscale portrait; (b) lowest intensity quartile; (c) low-intensity blob; (d) highest intensity quartile; (e) high-intensity blob; (f) middle quartiles (discarded). Original image copyright Corel Corp.; used with permission.

Preliminary tests on smaller data sets indicated that including the middle intensities can actually hurt performance slightly. This corroborates subjective impressions that images are easiest to identify from the intensity extremes. Often the middle intensities contain "negative space" cutouts of other objects in the image, and are thus prone to undergo dramatic changes in shape as a result of relatively simple scene transformations. Figure 1(f) illustrates this phenomenon. Because of it, PBSIM focuses exclusively on the intensity extremes.

In processing a single image component, the feature extraction algorithm first ranks the pixels and divides them into quartiles by intensity. (Although quartiles were used in all the results described in this paper, there is no *a priori* reason not to choose a different percentile threshold.) Pixels in the highest and lowest quartiles are retained for further processing, while those in the middle two quartiles are thrown away. Analyzing the connected components of each remaining quartile gives a list of candidate blobs to serve as image features. Often a single large connected component will dominate, although occasionally there will be several of approximately equal size. All components within a fixed percentage of the size of the largest are retained for further description. The results reported in this paper set this cutoff parameter at 80%. The retained components, referred to as *blob features* or simply blobs, represent large contiguous regions of high or low intensity in the portrait being processed.

2.2 Feature Description

Each blob identified by the procedure described above is characterized for later matching by eleven statistics describing its position, size, and shape. The statistics used, given in Table 1, are designed to be as independent as possible of the size of the image and of each other.

In addition to numbers describing the blobs identified within the image, several general statistics are collected about the entire portrait. In particular, parameters of the blob identification procedure are recorded. The full list of information collected about each portrait is as follows: the mean, standard deviation, skew, and kurtosis of the intensity value distribution, the intensity of the 25th and 75th percentile level, and the range of intensities covered by each percentile band (0-25, 25-75, and 75-100). These general statistics can be highly informative in conjunction with the identified blobs: When the intensity information they contain is added to the blob outlines, images become much easier to identify for human observers. Our preliminary experiments indicated that these statistics also significantly improve the performance of PBSIM.

To summarize, the features describing an image form a hierarchy. Each of the four portraits of a given image is described by the general numeric features described in the preceding paragraph, plus two complex features repre-

Table 1. Statistics Used To Characterize Blobs

Area Percentage of total image area covered by the blob.

Position Mean x and y coordinates, expressed as fractions of the width and height of the image, respectively.

Extent Value of 20th and 80th percentile of x and y pixel coordinates, expressed as a fraction of the width or height of the image.

Aspect Defined as $(\Delta x - \Delta y)/\sqrt{\Delta x \cdot \Delta y}$, where Δx and Δy are the breadths between the extent statistics described above.

Slant Mean of $(x-\bar{x})*(y-\bar{y})$ for all pixels in blob, divided by the mean of $|(x-\bar{x})*(y-\bar{y})|$, where x and y are the pixel coordinates, again expressed as a fraction of the width or height of the image, and $\bar{x}$ and $\bar{y}$ are the position statistics described above.

Density Fraction of pixels lying within the rectangle defined by extent statistics that belong to the blob.

Compactness A comparison of the square of the blob perimeter to its area. Defined as $((N_e/4)^2 - N_b)/N_b$ where N_e is the number of border pixels and N_b is the total number of blob pixels.

senting the portrait's high- and low-intensity blobs. The two blob features consist of a set of one or more individual blob descriptions, comprising the eleven numeric statistics described in Table 1. A full description of an image thus includes 36 simple numeric features and eight complex, set-like features, each of which contains one or more collections of eleven numbers describing a blob. These eleven numbers may be thought of a sub-features that describe a given blob feature.

2.3 Image Comparison

Once a description has been extracted from an image, it can be compared with other image descriptions to determine similarity. Actually, the comparison algorithm computes a difference score, meaning that lower scores indicate greater similarity. It does so by computing a difference score for each feature and summing over all features to arrive at the total difference.

The individual difference scores for the simple numeric features are merely their absolute difference. In contrast, the blob description features require a more complicated treatment. Because the blob extraction step sometimes identifies more than one blob of a particular type, a given blob feature may contain a set of several candidates for matching. In this case, all possible pairwise matches are considered with the corresponding set of blobs for the other image, and the difference for the feature in question is taken to be the difference of the best-matching pair of blobs of that type. Any additional blobs not participating in the best match are simply ignored. Thus, regardless of how many blobs were extracted for the images being compared, exactly eight blob comparisons (two for each portrait, one of low intensity blobs and one of high) are included in the difference score for any two images.

$$D(I_1, I_2) = \sum_{i=1}^{36} |f_i(I_1) - f_i(I_2)| + \sum_{i=37}^{44} \left[\min_{b_1 \in f_i(I_1), b_2 \in f_i(I_2)} \Delta(b_1, b_2) \right] \quad (1)$$

The difference between any two blobs, $\Delta(b_1, b_2)$, is merely the sum of the absolute differences on all eleven blob description statistics, i.e., the L_1 distance between the two blob descriptions. In principle, the step computing all pairwise matches could be computationally expensive, but in the image set we used, multiple blobs are not common enough to add significant extra computation. All eleven statistics describing a blob are treated as a unit for purposes of choosing the best match; thus it is not permitted to take the best match for the shape from one blob and the best match for the position from another.

A few comments are in order about the difference measure described above. It is based on the L_1 distance rather than the L_2 distance, in order to make the measure less sensitive to a bad match on any single feature. Also, because features in one image are matched only against the corresponding feature in the other image certain types of similarity will not be detected. For example, a gray figure against a black background will not match well with the same figure against a white background. Because the lightest areas (high intensity blobs) are matched together, the figure in one area will be compared to the background in another. Although this is a potentially serious failing, it was hoped that such situations would be sufficiently rare to have only a small effect on performance. The results described later in this paper suggest that, at least for the images used here, this is indeed the case.

3 Evaluation

There exists no clearly defined benchmark for assessing a proposed similarity measure on images. Indeed, human notions of similarity may vary, and different definitions are appropriate for different tasks. We chose to evaluate PBSIM in an image classification task previously used to evaluate a different blob-based image similarity measure [3]. The task involves over 1000 images from the Corel photo collection, divided into a dozen categories according to their subject matter. Using leave-one-out cross validation, PBSIM was tested on its ability to correctly predict the category of an image by finding a nearest neighbor in the same category. This procedure involves testing each image once, using the entire remaining image set as possible targets. Each image is labeled with the category of its nearest neighbor in the remainder of the set.

Table 2. Accuracy of PBSIM by Class

	A	B	C	D	E	F	G	H	I	J	K	L
A	**83**	9	3	2	1	1	0	0	0	0	0	0
B	8	**79**	1	5	0	2	0	1	0	0	3	0
C	2	0	**73**	12	3	2	5	0	0	1	0	2
D	0	0	4	**88**	6	0	0	1	1	0	0	0
E	0	0	0	2	**88**	1	4	1	3	0	0	1
F	1	4	4	1	0	**73**	8	1	4	1	1	0
G	0	1	12	4	15	11	**36**	6	8	1	4	1
H	0	1	0	0	3	2	2	**83**	4	0	2	1
I	0	0	3	8	15	9	8	6	**45**	2	5	0
J	0	0	3	6	6	0	0	0	3	**63**	6	14
K	0	0	0	2	0	1	3	1	2	2	**86**	2
L	0	0	2	0	1	0	1	2	0	9	11	**72**

Key: Confusion matrix for the image classification task. Numbers in each row indicate the percentage of images classified into target category at the top of the column. Numbers on the diagonal are correct classifications. A = Sunsets, B = Night Scenes, C = Cheetahs/Leopard/Jaguars, D = Tigers, E = Elephants, F = Deserts, G = Fields, H = Mountains, I = Brown Bears, J = Polar Bears, K = Bald Eagles, L = Airshows.

Table 2 shows the class predicted by the nearest neighbor algorithm using PBSIM. Correct responses, indicated in bold face type, mean that an image's nearest neighbor according to PBSIM was a member of the same class. The classifier produced the correct response in 73.7% of the cases overall, and in several classes the accuracy was over 85%. Only two of the twelve categories proved especially difficult, and the mean accuracy over the other ten was 80.2%. The generally high accuracies are encouraging, as they suggest that PBSIM may be applicable to a wide range of images.

The two problem categories, Brown Bears and Fields, proved markedly more difficult than the others. Although several factors may be involved, the low scores on the Fields category may be explained by the fact that these images are similar to many of the animal categories except for the absence of a single extra object (the animal). This category was also subjectively one of the most diverse in appearance. Brown Bears appear to be a slightly different case. It may be that the bears were not reliably picked out of the background by PBSIM, and thus that background features provided the most reliable matches. However, Belongie et. al.[3] also report low scores on this category, and suggest that its members may also be too disparate to form a strong visual category.

Figure 2 shows some sample retrievals by the system. The target images featured were randomly chosen to ensure a representative mixture.

3.1 Comparison With Previous Results

As noted above, the fact that image similarity is poorly framed makes it difficult to compare different algorithms. Often, they may have been designed with different goals in mind, and been evaluated on different tasks. Nevertheless, some insight into relative merit may be gained by looking at those cases where comparison is possible.

We deliberately chose to test PBSIM on the same set of images used previously by Belongie et. al. for their work on image retrieval. They also use a blob-based approach, but their blob location algorithm is quite complex relative to PBSIM, incorporating a wider range of color and texture variation. Furthermore, they encode the location of a blob in terms of nine subdivisions of the image plane, and use a naive Bayes classifier to determine the class of a query image. They also report results for a classification scheme based on color histograms as a baseline.

In our tests, PBSIM does significantly better than the 54.7% across-the-board accuracy for the blob-based technique reported by Belongie et. al., as well as their histogram baseline (52.3%). Care should be taken in comparing their numbers directly with ours, as they report results based on a test set of only 1/3 of the total image set, whereas we used the entire set for our experiments. (Since we developed and tuned our algorithm using an entirely different set of images, it was unnecessary to divide the test set to create a separate subset for training.) We also performed a test of color histograms, using the same nearest-neighbor, leave-one-out cross validation setup as we used to test PBSIM, and found an average classification accuracy of 65.3%. (Our implementation of color histograms also differed in that Belongie et. al. used more color channels, and compared each image to the mean histogram for each category during classification.) Although still significantly lower than PBSIM, this result shows that a direct comparison between the two studies may underestimate the performance of the Belongie, et. al. approach.

Nevertheless, the discrepancy between the two blob-based approaches invites speculation as to the strengths and weaknesses of each. We suspect that the simplicity of PBSIM makes it more reliable if the differences between the categories is great enough, whereas the system of Belongie et. al. may be more sensitive to finer variations that leave it vulnerable to overfitting in some cases.

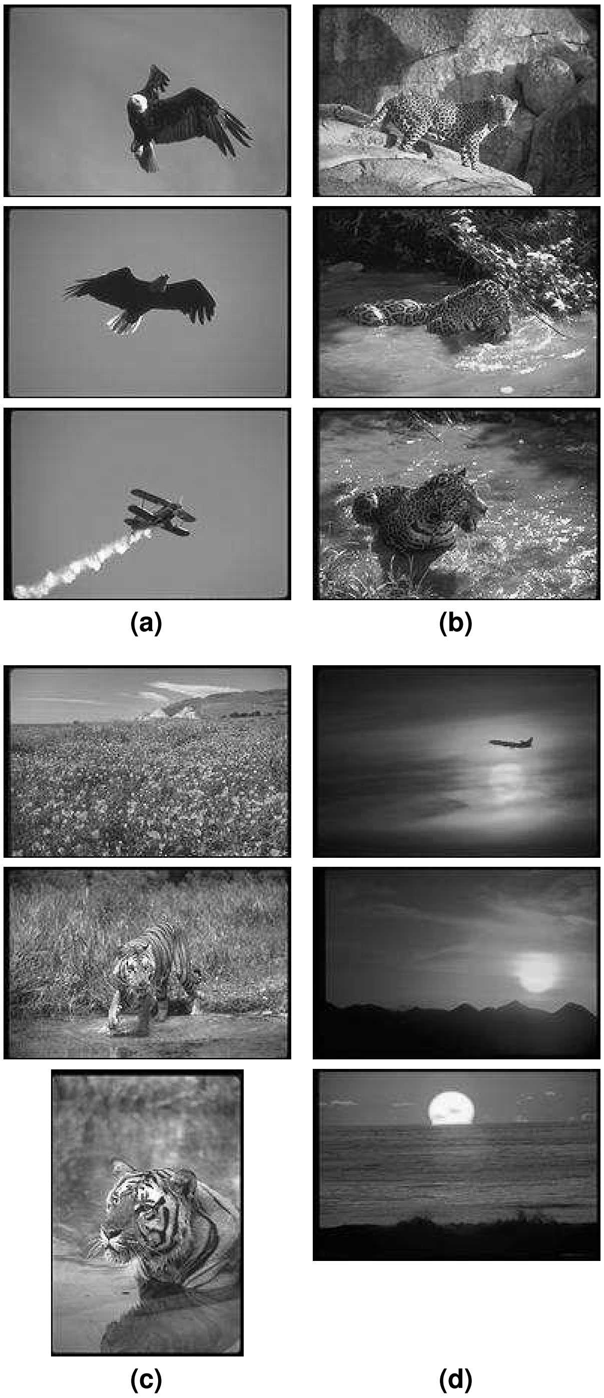

Figure 2. Top two images retrieved for four randomly chosen images from the test set. Query image is at top, closest match is second. The colors match well except in (c). Images copyright Corel Corp.; used with permission.

Other approaches to image similarity that use geometric information present in the images are color correlograms [6] and flexible templates [9]. While not based upon blobs, color correlograms incorporate geometric information about the locations of different hues in the images it is comparing. Huang et. al. report that they outperform color histograms for indexing and comparison. Flexible templates currently rely on a human supervisor to provide templates of desired image categories, which are then used to retrieve images from an image database. The success of this approach is heavily dependent on the quality of the templates provided, but the method can achieve excellent results for some types of images.

One other recent image similarity algorithm that reports better results than color histograms is based primarily on the texture present in images [4]. This approach extracts textural features at multiple resolutions and orientations. De Bonet and Viola show that, for the retrieval tasks they test, their *texture-of-textures* (ToT) method outperforms several implementations based upon color histograms. This suggests that it may be performing at a level comparable to that of PBSIM, although they perform their tests on a different set of images. More importantly, since the ToT method uses no information about the spatial distribution of textures in the image, while PBSIM includes comparatively limited textural information, a combination of the two techniques may prove fruitful.

Several commercial image retrieval systems have been built that incorporate multiple similarity measures. For example, QBIC [1] and Candid [8] allows the user to choose the similarity measure used in a given application, allowing for greater flexibility than a single fixed measure. Photobook [11] also provides an array of tools for retrieving a desired image. In principle, PBSIM could be employed as one of the techniques in such a system. Other image retrieval systems using techniques different from blob matching include Virage [2] and Chabot [10].

4 Conclusion

We have presented PBSIM, an algorithm for measuring the perceptual similarity between two images. PBSIM deliberately adopts a simplistic approach to this problem. Details of images are glossed over in favor of broader characteristics, allowing the algorithm to readily perform high-level generalization. We have seen no signs that PBSIM's performance degrades as the image set size increases – to the contrary, preliminary tests with smaller data sets were producing less impressive results. To some extent, the situation may be analogous to that in the field of text retrieval, where historically, systems with the greatest precision and recall have also frequently been the simplest.

Naturally, because of its general approach, PBSIM is not suited for all types of similarity problems. In a face recognition task, for example, a specially designed algorithm would certainly be better. It will probably also falter on scenes that are cluttered with many small objects, since it is based upon the identification of large regions of importance. But for rapid searches through large sets of natural scenes like the ones tested, PBSIM is a good choice. For some applications, it might also be used as a first pass filter in conjunction with a second more computationally intensive algorithm.

PBSIM produces impressive results on the image classification task we used as a test, classifying images with an accuracy exceeding 85% in several categories tested. These results show the power of incorporating geometric information into the description of an image. Techniques like the standard color histogram approach [12] and the texture-of-textures algorithm [4] fail to take advantage of this rich source of information. The combination of color, texture, and geometry reveals far more about an image than any one of these properties alone.

Although relatively simplistic, PBSIM may be easily improved in a number of ways to make more fine-grained distinctions, while continuing to take advantage of the basic power of the blob description. We expect that continued work on techniques of this kind will produce even greater gains in performance. Furthermore, we expect that more complete image retrieval systems will benefit from adding PBSIM to their collection of tools for determining image similarity.

5 Acknowledgements

The author thanks Dan Huttenlocher for invaluable advice and comments at all stages of this project. Thanks also go to Serge Belongie for providing the list of images used for testing, as well as assistance in locating them. This work was supported in part by Microsoft, Xerox, and NSF Grants GER-9454149 and CDW-9703470.

References

[1] J. Ashley, R. Barber, M. Flickner, J. Hafner, D. Lee, W. Niblack, and D. Petkovic. Automatic and semi-automatic methods for image annotation and retrieval in QBIC. In W. Niblack and R. C. Jain, editors, *Storage and Retrieval for Image and Video Databases III*. SPIE – The International Society for Optical Engineering, 1995.

[2] J. R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. Jain, and C. Shu. The virage image search engine: An open framework for image management. In I. K. Sethi and R. C. Jain, editors, *Storage and Retrieval for Image and Video Databases IV*. SPIE – The International Society for Optical Engineering, 1996.

[3] S. Belongie, C. Carson, H. Greenspan, and J. Malik. Recognition of images in large databases using a learning framework. Technical Report 97-939, UC Berkeley, 1997.

[4] J. S. D. Bonet and P. Viola. Structure driven image database retrieval. *Advances in Neural Information Processing*, 10, 1997.

[5] E. Grimson. *Object Recognition by Computer*. MIT Press, 1990.

[6] J. Huang, S. K. Kumar, M. Mitra, W. Zhu, and R. Zabih. Image indexing using color correlograms. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 1997.

[7] J. Huang, S. Ravi Kumar, M. Mitra, W. J. Zhu, and R. Zabih. Image indexing using color correlograms. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 1997.

[8] P. M. Kelly and M. Cannon. Query by image example: the CANDID approach. In W. Niblack and R. C. Jain, editors, *Storage and Retrieval for Image and Video Databases III*. SPIE – The International Society for Optical Engineering, 1995.

[9] P. Lipson, E. Grimson, and P. Sinha. Configuration based scene classification and image indexing. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 1997.

[10] V. E. Ogle and M. Stonebreaker. Chabot: Retrieval from a relational database of images. *IEEE Computer*, 28(9):40–48, September 1995.

[11] A. Pentland, R. W. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. Technical report, M.I.T. Media Laboratory, 1993.

[12] M. Swain and D. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.