# AN IMAGE SIGNATURE FOR ANY KIND OF IMAGE

*H. Chi Wong, Marshall Bern,* and *David Goldberg*

Xerox Palo Alto Research Center
3333 Coyote Hill Rd., Palo Alto, CA 94304

## ABSTRACT

We describe an algorithm for computing an image signature, suitable for first-stage screening for duplicate images. Our signature relies on relative brightness of image regions, and is generally applicable to photographs, text documents, and line art. We give experimental results on the sensitivity and robustness of signatures for actual image collections, and also results on the robustness of signatures under transformations such as resizing, rescanning, and compression.

## 1. BACKGROUND AND MOTIVATION

Massive image databases are becoming increasingly common. Examples include document image databases such as declassified government documents [8], and photo archives such as the New York Times archive. Duplicate removal offers space and bandwidth savings, and more user-friendly search results. Despite some effort to cull duplicates, the image search service of Google [4] often retrieves a number of duplicate and near-duplicate images. Duplicate detection also finds application in copyright protection and authentication.

We believe that duplicate detection is most effectively solved in two distinct stages. A fast first stage reduces images to small *signatures*, with the property that signatures of two different versions of the same image have small vector distance relative to the typical distance between signatures of distinct images. A slow second stage then makes a detailed comparison of candidate duplicates identified in the first stage. Detailed comparison of document images can identify changes as small as moving a decimal point [16].

In this paper we give a fast and simple algorithm for the first stage. Our image signature encodes the relative brightness of different regions of the image; it can be applied quite generally to text document images, line art (such as cartoons), and continuous-tone images. Although there are a number of image signature schemes already in the literature, there is no one signature that applies to such a wide class of images. The main limitation of our signature is that it is not designed to handle large amounts of cropping or rotation. This design choice is appropriate for document image databases and Web search, but not for object recognition or for detecting copyright violations.

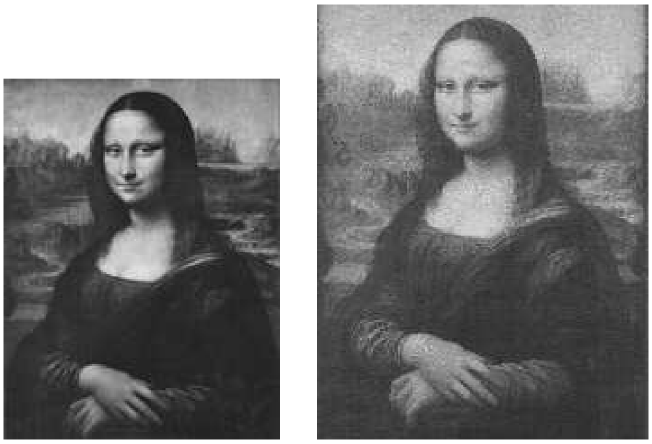
## 2. PREVIOUS WORK

Image signatures have already been used to address three different, but closely related, problems: similarity search, authentication, and duplicate detection. The three problems have slightly different characteristics that affect the design of the signature. Signatures for similarity search [10] (for example, color histograms) must preserve similarity, not just identity, and hence do not necessarily spread out non-duplicates very effectively. Signatures for authentication [1, 11, 13, 15] must be more sensitive—but can be much slower—than signatures for the first stage of duplicate detection.

Nevertheless, the techniques developed for search and authentication can be harnessed for duplicate detection. Our work adapts a method by O'Gorman and Rabinovich [11] for computing a signature for ID card photos. Their method places an $8 \times 10$ grid of points on the ID card photo and computes a vector of 8 bits for each point in the grid; roughly speaking, each bit records whether a neighborhood of that point is darker or lighter than a neighborhood of its 8 diagonal or orthogonal neighbors. In the usage scenario, the image signature computed from the photo is compared with a reference signature written on the ID card and digitally signed with public-key cryptography.

Another technique worth considering lets the image content dictate the neighborhoods used in the signature [1, 14]. Schmid and Mohr [14] use a corner-point detector to define "interesting points" in the image. The signature then includes a statistical abstract of the neighborhood of each interesting point. Compared to grid-based methods such as ours, this approach has the advantage that it can handle large amounts of cropping, and if the statistics at each interesting point are rotation invariant, it can also handle arbitrary amounts of rotation. On the other hand, it seems to take several hundred interesting points to obtain a reliable signature, and this approach is likely to break down entirely for text documents or line art images, which may contain thousands of interesting points with very similar statistics.

Also available in the literature are specialized signature methods for document images, using optical character recognition [5, 8] or document-specific features such as paragraph layout, line lengths, and letter shapes [2, 3]. We expect that these specialized methods are more accurate than our own generic method. Our method, however, should perform quite well for first-stage filtering, reducing the number of possible duplicate pairs by a factor of hundreds or thousands, whereupon still more accurate and detailed full-image matching algorithms [9, 12, 16] can be used for the slow second-stage matching.



(a) A $150 \times 196$ image. (b) A $180 \times 240$ image.

**Fig. 1**. "Naturally occurring" duplicates: two images of the Mona Lisa downloaded from the Web.

## 3. THE SIGNATURE

Ideally, we would like a signature that is small enough to allow efficient nearest-neighbor search, sensitive enough to effectively filter a database for possible duplicates, and yet robust enough to find duplicates that have been resized, rescanned, or lossily compressed. Figure 1 shows black-and-white versions of a typical duplicate pair. These are black-and-white versions of two different JPEG-compressed color images of the Mona Lisa downloaded from the Web. The images have different sizes, gray values, and cropping (look at the fingers).

We now describe the steps of our algorithm in detail. Although inspired by the signature algorithm of O'Gorman et al. [11], our algorithm differs at almost every step. For example, we added something to handle mild cropping, and expanded O'Gorman's two-level relative darkness values—simply "darker" or "lighter"—to five levels to give greater sensitivity and robustness.

**Step 1.** If the image is color, we first convert it to 8-bit grayscale using the standard color-conversion algorithms included in *djpeg* and *ppmtopgm*. Pure white is represented by 255 and pure black by 0.

**Step 2.** We next impose a $9 \times 9$ grid of points on the image. (For large databases, a bigger grid such as $11 \times 11$ would give greater first-stage filtering.) We define the grid in a way that is robust to mild cropping, under the assumption that such cropping usually removes relatively featureless parts of the image, for example, the margins of a document image or the dark bottom of the Mona Lisa picture. For each column of the image, we compute the sum of absolute values of differences between adjacent pixels in that column. We compute the total of all columns, and crop the image at the 5% and 95% columns, that is, the columns such that 5% of the total sum of differences lies on either side of the cropped image. We crop the rows of the image the same way (using the sums of original uncropped rows).

Conceptually, we then divide the cropped image into a $10 \times 10$ grid of blocks. We round each interior grid point to the closest pixel (that is, integer coordinates), thereby setting a $9 \times 9$ grid of points on the image.

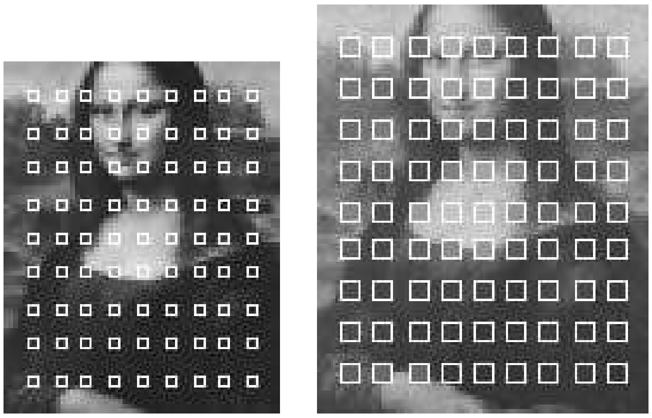
**Step 3.** At each grid point, we compute the average gray level of the $P \times P$ square centered at the grid point. We ran our experiments with

$$P = \max\{2, \lfloor .5 + \min\{n, m\}/20 \rfloor\},$$

where $m$ and $n$ are the dimensions of the image in pixels. The squares are slightly soft-edged, meaning that instead of using the pixel's gray levels themselves, we use an average of a $3 \times 3$ block centered at that pixel. Figure 2 shows the grid of squares for half-size versions of each of the images from Figure 1.

**Step 4.** For each grid point, we compute an 8-element array whose elements give a comparison of the average gray level of the grid point square with those of its eight neighbors. The result of a comparison can be "much darker", "darker", "same", "lighter", or "much lighter", represented numerically as -2, -1, 0, 1 and 2, respectively. The "same" values are those averages that differ by no more than 2 on a scale of 0 to 255. We set the boundary between "much darker" and "darker" so that these two values are equally popular; we do the same for "lighter" and "much lighter". The rationale in this step is that "same" may be very common in images with flat backgrounds (such as text documents), and hence it should not be included in the histogram equalization applied to the other values. Grid points in the first or last rows or column have fewer than 8 neighbors; for simplicity we fill in the missing comparisons—there are 104 of them—in the array with zeros.

**Step 5.** The signature of an image is simply the concatenation of the 8-element arrays corresponding to the grid points, ordered left-to-right, top-to-bottom. Our signatures are thus vectors of length 648. We store them in 648-byte arrays, but because some of the entries for the first and last rows and columns are known to be zeros and because each byte is used to hold only 5 values, signatures could be represented by as few as $\lceil 544 \log_2 5 \rceil = 1264$ bits.



(a) $3 \times 3$ squares. (b) $5 \times 5$ squares.

**Fig. 2**. The grid squares for the Mona Lisa images.

## 4. NEAREST NEIGHBORS

The *normalized distance* $\Delta(u, v)$ between two image signatures $u$ and $v$ is $\|u - v\|/(\|u\| + \|v\|)$, where $\|\cdot\|$ represents the $L_2$ norm of a vector, that is, its Euclidean length. Relative to the $L_1$ norm (or Hamming distance, as used in [11]), the $L_2$ norm gives greater emphasis to larger differences; this is appropriate because a difference of 1 at a coordinate may be due to roundoff, but a difference of 2 is meaningful.

It is necessary to normalize the distance, rather than just using $\|u - v\|$, in order to compare the distance against a fixed threshold. A threshold of .6 seems to be a good choice, giving reasonable robustness to image transformations without admitting very many false matches. For text documents, a slightly lower threshold is better, because their signatures usually have many more zeros than do the signatures of continuous-tone images. An easy fix, which allows the use of a .6 threshold for any type of image, is to change basic arithmetic: count the difference between a 0 and a 2 or $-2$ in $\|u - v\|$ as 3. We used this fix in the experiments reported below.

Given a query signature $u$, we now need a way to find all signatures $v$ in a database within normalized distance of .6 or less. The usual way to accomplish this task involves chopping up the signatures into (possibly non-contiguous and overlapping) *words* of $k$ bytes, and finding all $v$'s that exactly match $u$ on some word. For each of the $N$ word positions, an *index table* points to all the signatures with each given word. The parameters $k$ and $N$ are chosen so that any $v$ within normalized distance of .6 is sure to match on at least one word, and so that the space requirement of the index tables is not too large. This method solves the problem of nearest-neighbor search in high dimensions by finding exact matches in a number of lower-dimensional projections. There are also more sophisticated algorithms and analysis of nearest neighbor search in high dimensions [6, 7].

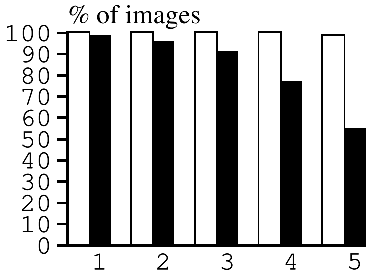
In our case, a normalized distance of .6 allows $u$ and $v$ to differ at every byte, so there is no good choice of word length $k$. Therefore we propose that in the index tables, we lump together $-2$ and $-1$, that is, represent them both by $-1$, and similarly lump together 2 and 1. The table entries, although indexed by a words over 3 possible letters, still point to the more precise 5-letter signatures. Now a reasonable choice would be $k = 10$ and $N = 100$. A signature $v$ within .6 of the query $u$ is very likely to match on some word. If each letter of the lumped version of $v$ has probability .75 of matching the corresponding letter in the lumped version of $u$, then each word has probability $.75^{10} \approx .05$ of matching, and hence the chance that at least one of 100 words match is about $1 - .05^{100}$, which is greater than .993. A random signature $v$, however, is unlikely to match on some word. If each letter of lumped $v$ has probability .5 of matching the corresponding letter in lumped $u$, then each word has probability $.5^{100} \approx .001$ of matching, and hence the chance that at least one of the 81 words match is about $1 - .999^{100} \approx .10$. Hence only about 10% of the database signatures need to have their actual normalized distances to $u$ computed. Because the computation of normalized distances is very fast, this should be quite tolerable for a million-image database. Larger values of $N$ would give greater discriminating power but use more memory space. As $N$ grows larger, signatures within .6 would tend to have many more word matches than random signatures.

Notice that with this method for nearest-neighbor searching, the overall image duplicate detection process becomes a three-level scheme. The indexing on words finds candidate signature pairs, which in turn lead to candidate image pairs.

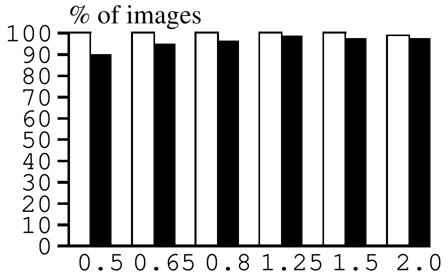
## 5. RESULTS

We tested the robustness (the "recall") of our algorithm on both synthetic and natural data. The synthetic data consist of a set of original images (600 photo CD images, belonging to 6 themes - Michelangelo, transportation, cityscape, etc - and 170 document images - pages from a Ph.D. thesis) and their synthetic duplicates, generated by rotating, resizing, and compressing/decompressing the originals. The bar charts below show the robustness of our algorithm in finding synthetic duplicates. The natural data consist of 10 sets of images downloaded from the Web, each containing a number of visually (close-to-)identical images of different sizes, with different amounts of cropping, and some even with slight color differences. Our algorithm handles well these naturally occurring duplicates, failing only in cases where there is a significant amount of cropping or color difference.

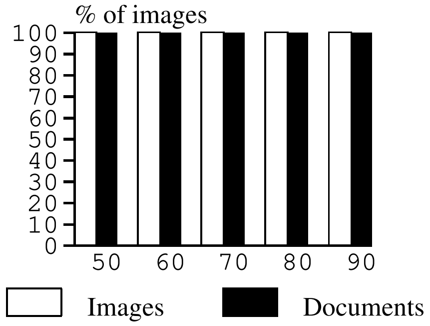
**Rotation (in degrees)**




**Resizing**




**Compression**




We also tested the sensitivity (the "precision") of our algorithm by running it on our original set of photo CD images and document images. The table below shows the average number of false positives returned for each type of images.

| image type | average # of false positives |
|---|---|
| photo CD images | 0.53 (in a total of 600) |
| document images | 4.05 (in a total of 170) |

As future work, we plan to conduc studies comparing our method with other existing methods.

**Acknowledgments.** We would like to thank Paul Stewart and Russel Atkinson for their help with software tools.

## 6. REFERENCES

[1] S. Bhattacharjee and M. Kutter. Compression tolerant image authentication. *Proc. ICIP*, Vol. 1, pp. 435–439, 1998.

[2] V. Chalana, A.G. Bruce, and T. Nguyen. Duplicate document detection in DocBrowse. *Proc. SPIE Document Recognition and Retrieval Conference*, 1999.

[3] D. Doermann, H. Li, and O. Kia. The detection of duplicates in document image databases. *Proc. Int. Conf. Document Analysis and Recognition*, pp. 314–318, 1997. http://documents.cfar.umd.edu/LAMP/

[4] Google Advanced Search. http://www.google.com

[5] J. Hull. Document image similarity and equivalence detection. *Pattern Recognition Letters*, 2001, 545–550.

[6] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. *ACM Symp. Theory of Computing*, 1999.

[7] P. Indyk, R. Motwani, P. Raghavan, S. Vempala. Locality-preserving hashing in multidimensional spaces. *ACM Symp. Theory of Computing*, 1997.

[8] D. Lopresti. A comparison of text-based methods for detecting duplication in document image databases. *Proc. Document Recognition and Retrieval VI, SPIE Electronic Imaging*, Vol. 3967, pp. 210–221, 2000. http://www.bell-labs.com/org/1133/Research/DocumentAnalysis/dup.html

[9] J. Mao and K. Mohiuddin. Form dropout using distance transformation. *Proc. ICIP*, Vol. 3, pp. 328–331, 1995.

[10] W. Niblack, R. Barber, and W. Equitz. The QUBIC project: querying images by content using color, texture, and shape. IBM Research Report #RJ 9203, February 1993.

[11] L. O'Gorman and I. Rabinovich. Secure identification document via pattern recognition and public-key cryptography. *IEEE Trans. Pattern Analysis and Machine Intelligence* 20 (1998), 1097–1102.

[12] H. Peng, F. Long, W.-C. Siu, Z. Chi, and D. Feng. Document image matching based on component blocks. *Proc. ICIP*, Vol. 2, pp. 601–604, 2000.

[13] M. Ruhl, M. Bern, and D. Goldberg. Secure notarization of paper text documents. *Proc. 12th SIAM Symp. Disc. Algorithms*, 2001.

[14] C. Schmid and R. Mohr. Image retrieval using local characterization. *Proc. ICIP*, pp. 781–784, 1996.

[15] R. Venkatesan, S.-M. Koon, M.H. Jakubowski, and P. Moulin. Robust image hashing. *Proc. ICIP*, 2000.

[16] M. Ye, M. Bern, and D Goldberg. Document image matching and annotation lifting. *Proc. Int. Conf. Document Analysis and Recognition*, 2001.