ROBUST HASH FOR DETECTING AND LOCALIZING IMAGE TAMPERING

Sujoy Roy *Qibin Sun*

Institute for Infocomm Research, Singapore

ABSTRACT

An image hash should be (1) robust to allowable operations and (2) sensitive to illegal manipulations and distinct queries. Some applications also require the hash to be able to localize image tampering. This requires the hash to contain both robust content and alignment information to meet the above criterion. Fulfilling this is difficult because of two contradictory requirements. First, the hash should be small and second, to verify authenticity and then localize tampering, the amount of information in the hash about the original required would be large. Hence a tradeoff between these requirements needs to be found. This paper presents an image hashing method that addresses this concern, to not only detect but also localize tampering using a small signature ($< 1kB$). Illustrative experiments bring out the efficacy of the proposed method compared to existing methods.

Index Terms— Locality preserving hashing, edge histogram, local region descriptors.

1. INTRODUCTION

An image hash is an essential component of any signature based approach to verify the authenticity of protected query images. It is a short signature of an image that is robust to allowable modifications (like small rotations, compression, scaling, addition of noise etc) and sensitive to distinct queries and illegal tampering. Figure 1 depicts an image and its manipulated copy (query image). For verification, given the hash of the original and the query, the hash verification algorithm verifies the authenticity of the query. Only allowably modified images (slightly rotated, cropped, JPEG compressed etc.) are declared authentic. Tampered or distinct images are declared non-authentic. For non-authentic images, the application may also require the method to be able to localize any tampering in the image.

Existing image hashing methods can be categorized as (1) exhaustive search based[1] and (2) robust representation based approaches[2, 3, 4, 5, 6]. In an exhaustive search based approach, the difference between the query and the original is modeled by some fixed distortion model (e.g., affine transform) and the hash consist of alignment information about the original. For verification, the right alignment between the original and the query is searched for by trying all possible transformations, reverse applied on the query and comparing the hash of the query with that of the original under a similarity measure. If a very close alignment is indeed found, the query is declared authentic. On the other hand, in a robust representation based approach, robust features are extracted from the image, from which bits are extracted to generate the hash. During verification, the hash of the query is generated and compared with that of the original using a similarity measure. The query is declared authentic based on their similarity.

(a) (b)

Fig. 1. (a) Original Image (b) Illegally tampered image (also rotated by 2^o, cropped, stretched, JPEG compressed (Q=20)).

The above approaches have their advantages and disadvantages. Exhaustive search based methods clearly suffer from impractical levels of search complexity, although in theory they can synchronize the query with the original. Lack of content information as part of the hash also leads to high false positive detection error[7] and does not allow localizing tampering. On the other hand, in a robust representation based method although the hash carries robust content information, desynchronization of the query with respect to the original and lack of alignment information as part of the hash significantly limits the verification performance. Furthermore, the hash generation process (represented in bits) should preserve the performance at the feature representation level[6]. This makes it clear that both alignment and robust content information should be carefully selected to generate an effective hash. This is particularly true for applications that require both detection and tamper localization in images. This implies that the amount of information in the hash about the original should be large.

Moreover, in a signature based approach, the hash is associated with the image as header information and hence must be small. This brings in two contradictory requirements that

have to be met. First, the signature should be small and second, to detect and localize tampering, the amount of information in the hash about the original, should be as large as possible. Therefore, a tradeoff between these two contradictory requirements needs to be found. To resolve this contradiction forms the motivation for this work.

This paper proposes a novel signature based approach for localizing tampering in images, wherein the signature, consisting of a tuple of bit vectors, carries both content and alignment information and is short in size ($< 1kB$). The hash generation process includes a novel locality preserving hashing scheme that reduces the hash size significantly while preserving the robustness-sensitivity performance at the feature level. The hash allows both detecting and localizing image tampering.

2. FORMULATION

Hash Generation An image hashing method consists of two steps: (1) hash generation and (2) verification. For hash generation, a set of features $I = \{F_1, \ldots, F_m\}$, is extracted from the image and a function $f : I \mapsto h$, maps (also called bit extraction process) them to a bit sequence $h \in \{0,1\}^L$, where $F_1 \in \mathbf{R}^n$. $\{\mathbf{R}^n\}$ denotes a set of vectors in n dimensional real space and $\{0,1\}^L$ denotes a bit sequence of size L. If $|I|$ denotes the size of I and $bit(x)$ the bit representation of any real number x, $|I| \times n \times bit(x) \gg L$.

Verification During verification given a query image (hence given $\tilde{I} = \{\tilde{F}_1, \ldots, \tilde{F}_k\}$, as a set of features) and the hash h, the detector decides whether $\tilde{I}$ is authentic or not. The hash $\tilde{h}$ of $\tilde{I}$ is computed and compared with h based on a dissimilarity/similarity measure. Note that I and $\tilde{I}$ are not synchronized and their sizes need not be the same. A query is declared authentic only if it is an allowably modified (rotated, cropped, compressed, scaled etc) version of the original. Under illegal manipulations like localized tampering, the verification routine localizes the tampered region in $\tilde{I}$. The next section describes the proposed hashing method.

3. PROPOSED METHOD

Hash Generation In the proposed method the hash generation process consists of a feature extraction step, followed by a bit extraction process that generates a bit sequence h of fixed size. Unlike existing methods, the bit sequence h is a tuple ($h = \{h_1, h_2\}$) of bit sequences h_1 and h_2, which are generated independently from two different kinds of features. The process of generation of h_1 and h_2 is described as follows.

Generation of h_1: The image is first down-sampled, and a set of SIFT[8] features $I = \{F_1, \ldots, F_m\}$ with $F_i \in \mathbf{R}^{128}$ are extracted from it. SIFT features are well-known to be robust[9] to several geometric transformations. Given I and a set of hyperplanes $\mathcal{H} = \{H_1, H_2, \ldots, H_d\}$ (where $H_j \in \mathbf{R}^{128}$ and each plane passes through the centroid of I), the bit extraction process implements a novel locality preserving hashing[10] algorithm to generate h_1, which is explained herein.

Algorithm 3.1: LOCALITY PRESERVING HASH($\mathcal{H}, I$)

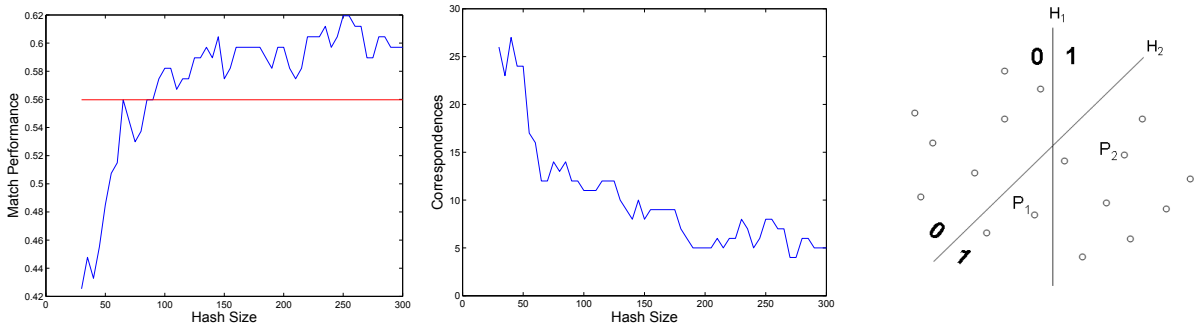
$P \leftarrow$ 2D coordinates of 3-5 stable $F_i \in I$ in bits
for each $F_i \in I, i \leftarrow 1$ **to** m
do
- $t_i \leftarrow []$
- **for each** $H_j \in \mathcal{H}, j \leftarrow 1$ **to** d
  - **do**
    - **if** F_i lies to the left of H_j
      - **then** $t_i \leftarrow t_i \oplus 0$
      - **else** $t_i \leftarrow t_i \oplus 1$

$V \leftarrow \{t_1, t_2, \ldots, t_m\}, t_i \in \{0,1\}^d$
$h_1 \leftarrow \{V, P\}, |P| = \ell$

For each hyperplane in $\mathcal{H}$ concatenate ($\oplus$) a 0 or 1 to t_i, depending on whether F_i lies on its left or right side. Hence for d random hyperplanes, F_i maps to a bit sequence $t_i \in \{0,1\}^d$ and for m features, the total bit sequence is given by the set $V = \{t_1, t_2, \ldots, t_m\}$. Figure 2(c) illustrates the bit extraction process for two hyperplanes. 2D coordinate information for 3-5 feature points in bits (P), are also included to generate the hash $h_1 = \{V, P\}$ of size $md + \ell$, where $|P| = \ell$ is the size of P in bits. Note that the hash h_1 contains content information (V) and alignment information (P). V is used to match correspondences and P helps in synchronizing the query after correspondence matching.

The following intuitive reason explains the efficacy of the locality preserving hashing algorithm. Each F_i can be affected by two kinds of noise, replacement and random perturbation. Under allowable transformations, due to the robustness of SIFT features, with high probability the position of F_i with respect to a hyperplane will not change. Hence there is low probability of any bit flip in t_i, thus preserving the matching performance through bit extraction. Also as d increases, each t_i becomes more stable and distinct. But increasing d, may increase the number of false positives (refer Figure 2(a)). Hence an appropriate value of d must be chosen. Note that for $d = 60$, the matching performance in the feature level is perfectly preserved at the hash level.

Generation of h_2: The hash h_2 complements h_1 by localizing any tampering to the aligned image. The image is first downsampled and filtered using an anisotropic diffusion filter and then edge detection is performed to generate its edge image. The orientation of the edges in the edge image are quantized to 5 directions $[0, 45, 90, 135, 180]$. Next the edge image is divided into non-overlapping blocks and the edge histogram for each block is computed. The edge histograms of each block are concatenated to generate h_2. Each edge histogram is represented by 15 bits. For an image which is



(a) (b) (c)

Fig. 2. **Locality Preserving Hash:** (a) Feature and hash level match performance with increase in hash size. The straight red line gives the match performance between I and $\tilde{I}$ and the blue line gives the match performance between h and $\tilde{h}$. (b) Change in correspondence difference between original and query, with change in hash size. (c) Illustration of the intermediate bit sequence generation process. For two hyperplanes H_1 and H_2, the points P_1 and P_2 are binarized as $\{01\}$ and $\{11\}$ respectively.

divided into 16 blocks, this would generate a hash h_2 of size 240 bits.

Verification Given a query $\tilde{I}$ with hash $\tilde{h} = \{\{\tilde{V}, \tilde{P}\}, \tilde{h_2}\}$, the verification stage uses a similarity measure where a match is declared if the distance between a pair of t_i's between V and $\tilde{V}$ is smaller than a threshold times the distance from their second nearest neighbor. This matches pairs of corresponding 2D points in the original and query. Now P is used to align the query with the original. Next, the hash $\tilde{h_2}$ of the aligned query is extracted and compared (based on a threshold T) with h_2. Image blocks with dissimilarity value greater than T, indicate a probable tampered block. The resolution of tamper localization depends on the block size chosen. A smaller block size will increase the hash size, while improving the tamper localization resolution. In our implementation, the image was first downsampled and then divided into 16 blocks.

4. EXPERIMENTS AND ANALYSIS

For our experiments a collection of 50 distinct images from the USC-SIPI database was used. These images were modified using 9 allowable transformations to generate 450 authentic queries. For checking image tampering, 10 images were spliced, content removed, and content rearranged, to generate perceptually indistinguishable tampered queries.

The hash h_1 aligns the query with the original. To choose the right hash size for h_1, the similarity between the hash ($\tilde{h}$) of the query (rotated by 20^o about a point [100,100] from the center of the image, cropped by 30% and JPEG compressed by QF=10) and the hash h of the original is compared for increasing values of hash size (Figure 2). The number of feature points in the original and the query image is approx. $m = 60$. From Figure 2(a) note that, as the hash size (d) increases, the hash level match performance increases. The straight line marks the feature level match performance and defines a bound on the achievable hash level match performance. Note that as hash size further increases, false positives are introduced, indicating that the hash size cannot be arbitrarily large. Next, given the corresponding features in the original and the query images, we tested how well is the correspondence preserved through the bit extraction process. Figure 2(b) depicts the change in correspondence difference with change in hash size. Note that for $d = 60$, hash level match performance equals feature level match performance and the correspondence mismatch is also low. Since for $d = 60$ feature level performance and point correspondences are preserved, only a few very robust feature points (m), with hash size $d = 60$ are sufficient to represent the query hash. In our implementation, for $m = 10$ and $d = 60$, h_1 requires $md + \ell = 600 + 80 = 680$ bits, where $\ell = 80$ is the 2D coordinate information of 5 points in bits. On the other hand h_2 requires 240 bits. Hence the total size of the hash h is $L = 920$ bits ($< 1kB$).

We compared the performance of our method (just using h_1) with 5 other existing state of the art methods (based on Fourier-Mellin invariants (FMT)[2], radial basis projections(RASH) [3], wavelets[5], SVD[4], structure matching (Feature) [1]). 50 distinct images and 450 allowably modified queries were used to generate true and imposter distance distributions for each method. Figure 3 depicts the ROC curves comparing each method. The proposed method clearly achieves very high discrimination, due to the high discrimination capacity of SIFT features which is preserved through the bit extraction process. Note that this discrimination is between distinct images and allowably modified images. For locally tampered images, h_1 alone in not sufficient; h_2 complements in making a conclusive decision.

Once the query is aligned with the original using h_1, the hash h_2 can be used detect any form of tampering, namely,

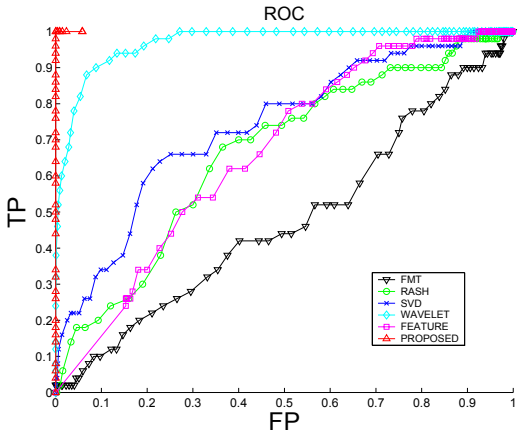


Fig. 3. ROC curve comparing the performance of the proposed method with existing methods for images rotated (20^o), cropped (30%) and JPEG compressed ($QF = 10$) against 50 distinct images from the USC-SIPI database.

Fig. 4. Tamper localization of example in Figure 1.

splicing, content removal, exchange of patches within the same image etc. The idea is that any intentional tampering leaves behind significant addition or deletion of content information, primarily edge boundary information. The block-wise quantized edge histograms captures this information in the hash h_2. The resolution of the patch detection depends on the size of the image blocks considered and hence affects the hash size. Figure 4 depicts the localization of tampering for the image in Figure 1 after aligning.

5. DISCUSSION

The proposed method can be seen as a unified method that combines the advantages of an exhaustive search based hashing and robust representation based hashing methods. The locality preserving projection of region descriptors can be seen as a short robust representation whereas the availability of 2D point location information useful for aligning the original with the query is a component of an exhaustive search based hashing method. The availability of content information as a robust bit representation helps in reducing the search complexity and decreasing false positive error, both of which are drawbacks of an exhaustive search based method. On the other hand, availability of point location information as part of the hash helps in registering the query with the original which in turn addresses the synchronization problem. Once aligned, the use of edge histogram information after some preprocessing of the query allows localizing any tampering.

6. REFERENCES

[1] V. Monga, D. Vats, and B. L. Evans, "Image authentication under geometric attacks via structure matching," in *ICME*, July 2005.

[2] Ashwin Swaminathan, Yinian Mao, and Min Wu, "Robust and secure image hashing," *accepted by IEEE Transactions on Information Forensics and Security*, to appear June 2006.

[3] C. De Roover, C. De Vleeschouwer, F. Lefebvre, and B. Macq, "Robust video hashing based on radial variance projections of key-frames," *IEEE Transactions on Signal Processing*, vol. 53, no. 10, pp. 4020–4037, October 2005.

[4] S. S. Kozat, K. Mihcak, and R. Venkatesan, "Robust perceptual image hashing via matrix invariances," in *Proc. IEEE Conf. on Image Processing*, Oct, 2004.

[5] M. H. Jakubowski R. Venkatesan, S.-M. Koon and P. Moulin, "Robust image hashing," in *Int. Conf. Image Processing, Vancouver, Canada.*, September, 2000.

[6] S. Roy, X. Zhu, J. Yuan, and E-C. Chang, "On preserving robustness false alarm tradeoff in media hashing," in *Proc. SPIE Visual Communication and Image Processing (to appear)*, Jan, 2007.

[7] J. Lichtenauer, I. Setyawan, T. Kalker, and R. Lagendijk, "Exhaustive geometric search and false positive watermark detection probability," in *Proc. SPIE Security and Watermarking Multimedia Contents V*, Jan, 2003, pp. 203–214.

[8] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[9] Krystian Mikolajczyk and Cordelia Schmid, "A performance evaluation of local descriptors," in *International Conference on Computer Vision & Pattern Recognition*, June 2003, vol. 2, pp. 257–263.

[10] P. Indyk, R. Motwani, P. Raghavan, and S. Vempala, "Locality preserving hash for multidimensional spaces," in *Proc. of ACM STOC*, 1997.