Large-Scale Image Retrieval Based on Compressed Camera Identification

Diego Valsesia, *Student Member, IEEE*, Giulio Coluccia, *Member, IEEE*, Tiziano Bianchi, *Member, IEEE*, and Enrico Magli, *Senior Member, IEEE*

***Abstract*—Retrieving pictures from large collections according to a specific criterion is an increasingly relevant task. An important, but so far overlooked, such criterion is the retrieval of pictures acquired by a specific camera. Instead of relying on metadata, which can be absent or easily manipulated, a forensic tool is exploited, namely the photo response non-uniformity (PRNU) of the camera sensor. Recent works showed that random projections can be used to significantly compress the PRNU, enabling operation on very large scales, previously impossible due to the size of the PRNU and to the complexity of the matching operations. In this paper, we propose efficient techniques for management and retrieval of images employing the PRNU, and test them on a database of 1174 cameras and half a million pictures downloaded from the Internet.**

***Index Terms*—Image forensics, image search and retrieval, photo response non-uniformity (PRNU), random projections.**

I. Introduction

EVERY day, millions of pictures are uploaded, shared, and browsed by Internet users, resulting in very large collections of images that call for efficient solutions for their management. An important task is the retrieval of pictures matching a specific criterion, that can be used for searching pictures of interest or for classifying similar photos. In this sense, content-based image retrieval is a technology that has received a lot of attention in recent years [1]. In a nutshell, it consists in extracting a set of representative features that allow to find pictures having similar content with respect to a query image.

However, the content of a picture is not the sole criterion for performing image retrieval. Another very interesting task consists in looking for pictures that have been acquired by a specific device. Let us imagine a search engine that, given a specific camera as a query, returns all the web pages containing photos acquired by that camera. Such a technology could be very useful for detecting improper usage of images. For example, professional photographers could use it to prevent improper diffusion of their photos, large web sites could avoid being sued for redistributing unlicensed pictures, police investigators who have come across a digital camera or even just pictures linked to an unlawful act, e.g., child pornography, could look for other pictures taken by the same camera in either public databases (e.g., social networks) or large internal databases managed by the police.

To this day, a similar technology has not yet been developed, at least on a large scale. A simple solution could be to rely on metadata, such as Exif data. However, this information can be easily removed from the image file, as it happens, e.g., when sharing images on Facebook, or altered by an image editing software. In order to have a reliable camera-based retrieval technique, it is essential that the acquisition device information be indissolubly linked to the picture.

When a picture is acquired by a digital sensor, slight imperfections in the manufacturing of individual detectors produce a unique fingerprint, usually referred to as photo-response non-uniformity (PRNU) [2]. The PRNU can be considered as a noise-like, yet deterministic, pattern affecting every image taken by a sensor, and can be used to determine if a picture has been acquired by a given sensor, or if two or more pictures have been taken by the same camera. Several works demonstrate that the PRNU is a robust fingerprint, usually surviving processing like lossy compression and image resizing [3], [4].

The use of PRNU as a fingerprint for camera identification has so far focused on forensic tasks involving a small number of cameras or photographs, as it is the case when it has to be used as evidence in a trial. In such scenarios, one has to verify whether a picture, or a small set of pictures, has been taken by a specific camera and the main requirements are high matching accuracy and low probability of false alarm. Another interesting scenario that has received some attention in recent literature is using the PRNU to classify a set of images according to the device that acquired them [5]–[7]. However, the proposed technologies have not been applied to very large databases so far [8].

Using the PRNU for large-scale image retrieval is an extremely challenging task that, to the best of our knowledge, has not yet been addressed in the literature. A large-scale database could contain several millions of fingerprints, and PRNU patterns have the same size as the imaging sensor, which typically counts tens of millions of pixels. Moreover, due to the specific properties of PRNU patterns, namely the fact that they are noise-like, traditional approaches, e.g., based on color and

Manuscript received February 27, 2015; revised May 22, 2015; accepted June 17, 2015. Date of publication July 10, 2015; date of current version August 10, 2015. This work was supported by the European Research Council under the European Community's Seventh Framework Programme under Grant FP7/2007-2013 and ERC Grant Agreement 279848. The guest editor coordinating the review of this manuscript and approving it for publication was Dr. Ramesh Jain.

The authors are with Department of Electronics and Telecommunications, Politecnico di Torino, Torino 2410129, Italy (e-mail: diego.valsesia@polito.it; giulio.coluccia@polito.it; tiziano.bianchi@polito.it; enrico.magli@polito.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2015.2455417

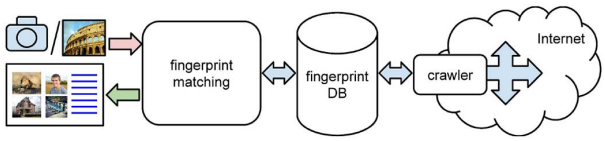

Fig. 1. High-level block diagram of the proposed system.

texture [9] or feature descriptors [10] do not work for the presented problem.

In this paper, we address the following problem of large-scale image retrieval based on PRNU fingerprints. A large collection of pictures is obtained by scanning portions of the web, using a crawler software that downloads photos from publicly available repositories. From this collection, a large fingerprint database is automatically generated by extracting the PRNU pattern of each individual photo. A query is presented to the system in the form of a camera fingerprint, and the goal is to retrieve all the photos acquired by the same device. The proposed system, visualized in Fig. 1, is a realistic example of the envisioned search engine, and can be used to demonstrate that the above task is indeed feasible. Notice that this image retrieval problem is significantly different from the problems addressed by the well-researched area of content-based image retrieval where the *content* of an image is the query and the user searches for images with similar content.

Due to the characteristics of PRNU, a technique for obtaining a compact representation of PRNU fingerprints is essential to deal with such a big data scenario. For example, assuming a collection of 6 billion photos, which is the number of photos hosted by Flickr in 2011 [11], and an average sensor resolution of 12 megapixels, the whole uncompressed fingerprint database, represented in single precision, would require about 250 petabytes of storage. In this paper, we propose to solve the above problem by using compressed fingerprints obtained via proper quantization of random projections. A recent work [12] has shown that random projections permit to obtain very compact representations with limited performance loss in terms of matching accuracy. Moreover, this scheme is flexible, meaning that the number of projections can be tuned according to the desired trade-off between compression and accuracy. Hence, this technique appears to be one of the best candidates for performing camera-based image retrieval in very large scale scenarios.

In this paper we propose technical solutions for the large-scale PRNU problem based on compressed fingerprints. The performance of random projections in comparison to other state-of-the-art methods for PRNU compression was investigated in [12]. On the other hand, the focus of this paper is the optimization of such method to address the retrieval problem on large scales. In particular, several techniques for solving the retrieval problem are proposed and compared in terms of speed and matching accuracy. We first consider to perform a linear search, comparing the compressed query fingerprint with all the compressed fingerprints in the database. Then, we propose an alternative solution based on a two-step approach, in which a first search is performed over the entire database using a coarse version of the compressed fingerprint returning a subset of the database; a second search is then performed on this subset using a refined fingerprint. An improved version of the above technique is then considered, in which the coarse version of the compressed fingerprint is obtained by adaptively choosing the random projections with the largest magnitude.

The paper is organized as follows. Section II introduces the notation and provides background material on PRNU patterns. Section III describes the proposed image retrieval problem and discusses in detail how different efficient implementations can be achieved using fingerprints compressed via random projections. Section IV focuses on numerical experiments and comparisons, while Section V draws some conclusions.

II. Background

A. Notation and Definitions

We denote (column-) vectors and matrices by lowercase and uppercase boldface characters, respectively. The ℓ-th element of column vector $\mathbf{v}$ is v_ℓ. The i-th column of the matrix $\mathbf{A}$ is $\mathbf{a}_i$.

The notation $\mathbf{A}\cdot\mathbf{B}$ denotes the elementwise product between matrices $\mathbf{A}$ and $\mathbf{B}$, while $\mathbf{A}/\mathbf{B}$ denotes elementwise division.

The notation $\langle \mathbf{a}, \mathbf{b}\rangle$ denotes the scalar product between vectors $\mathbf{a}$ and $\mathbf{b}$, and $\|\mathbf{a}\|_2 = \sqrt{\langle \mathbf{a}, \mathbf{a}\rangle}$.

The notation $d_H(\mathbf{a}, \mathbf{b})$ denotes the Hamming distance between $\mathbf{a}, \mathbf{b} \in \{0,1\}^m$, where $d_H(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^m a_i \oplus b_i$ and $\oplus$ denotes the XOR operator.

The notation $\mathbf{a} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ means that the random vector $\mathbf{a}$ is Gaussian distributed, its mean is $\boldsymbol{\mu}$, and its covariance matrix is $\boldsymbol{\Sigma}$.

B. PRNU and Compression via Random Projections

The PRNU of a digital imaging sensor is a noise-like pattern of the same size as the sensor, generated by imperfections in its manufacturing process. Such imperfections cause slight variations in the properties of individual pixels, which hence respond in a different manner to the incident light field, thus causing pixel-dependent gain variations. These random imperfections are peculiar to each manufactured sensor, therefore making the PRNU a highly informative pattern, able to uniquely identify a photographic device. Although this pattern is deterministic, it shows noise-like characteristics and it is unique to each sensor. Indeed, its entries are often modeled as realizations of i.i.d. Gaussian random variables. It is proved to be a robust fingerprint, thanks of its stability in time, and resilience to standard processing operations like image compression, resizing, or many enhancement operations.

The PRNU pattern of a particular sensor can be extracted from one or more photos taken by that sensor. In formulas, an acquired image $\mathbf{o}$ can be modeled as

$$\mathbf{o} = \mathbf{o}^{\mathsf{id}} + \mathbf{o}^{\mathsf{id}} \cdot \mathbf{k} + \mathbf{e} \tag{1}$$

where $\mathbf{o}^{\mathsf{id}}$ is the ideal sensor output, $\mathbf{k}$ is the PRNU term and $\mathbf{e}$ collects other sources of noise. Calling $\mathbf{o}^{\mathsf{dn}}$ a denoised version of $\mathbf{o}$ obtained through proper filtering, this can be used as an approximation of the ideal sensor output and subtracted from each side of (1) to obtain the so-called *noise residual*, which

can be modeled as

$$\mathbf{w} = \mathbf{o} - \mathbf{o}^{\mathsf{dn}} = \mathbf{o} \cdot \mathbf{k} + \tilde{\mathbf{q}} \tag{2}$$

where $\tilde{\mathbf{q}}$ accounts for $\mathbf{e}$ and for other non-idealities of the model. The interested reader can refer to [13] for more details on the model. It can be noticed that the noise residual is a *modulated version* of the PRNU, where the modulating term is the current image. For this reason, an estimate of the fingerprint $\mathbf{k}$ can be obtained by pixel-wise dividing the noise residual by the image. Supposing that C photos acquired by the same camera are available, the maximum likelihood estimate can be obtained as

$$\hat{\mathbf{k}} = \sum_{\ell=1}^{C} \left(\mathbf{w}^{(\ell)} \cdot \mathbf{o}^{(\ell)} \right) \Big/ \sum_{\ell=1}^{C} (\mathbf{o}^{(\ell)})^2 \ . \tag{3}$$

PRNU patterns are often used in forensic tasks such as determining whether a photo has been acquired by a specific camera. Recently, variants of this problem have been studied in terms of classification or clustering problems, such as associating n photos to k cameras. As an example, a system is presented with n photos of unknown origin and it is told they come form k cameras, so that the task is determining the k clusters of photos according to the similarity of their PRNU patterns [5], [7], [8].

The main drawback of PRNU patterns is their size. In fact, due to their noise-like nature, they are hard to compress using standard image compression techniques. This problem arises especially when it is required to build a large database of fingerprints. Recently, a certain number of works started considering the problem of compression of PRNU fingerprints. In [14], [15], the authors introduced a so-called *fingerprint digest* obtained by keeping only a fixed number of the largest fingerprint values, along with their positions, which enables a fast search strategy with constant size fingerprints [16]. On the other hand, in [17] the authors proposed to represent sensor fingerprints in binary-quantized form: even though the size of binary fingerprints scales with sensor resolution, binarization can considerably speed-up the fingerprint matching process and reduce the storage requirements.

A novel compression strategy was adopted in [12]. Based on the Johnson-Lindenstrauss (JL) lemma [18], in fact, the authors proposed a compact representation of PRNU fingerprints based on a fixed number of random projections. In particular, a collection of N n–dimensional fingerprints, $\mathbf{D} \in \mathbb{R}^{n \times N}$, is reduced to a m–dimensional subspace $\mathbf{A} \in \mathbb{R}^{m \times N}$ by

$$\mathbf{A} = \mathbf{\Phi D}. \tag{4}$$

A single query fingerprint $\hat{\mathbf{k}} \in \mathbb{R}^n$ to be matched with the database can be first compressed using the same sensing matrix $\mathbf{\Phi}$ used to compress the database, namely

$$\mathbf{y} = \mathbf{\Phi} \hat{\mathbf{k}}. \tag{5}$$

Then, the compressed query fingerprint is compared to each column of the compressed database $\mathbf{A}$ to find the most correlated by computing the correlation coefficient. Several technical problems may arise in this framework. The first issue is the complexity related to the generation and storage of a fully random sensing matrix. The second is the complexity related to the matrix-matrix product of (4). Both can be mitigated by using a partial circulant sensing matrix, which allows to generate a lower number of random coefficients and to efficiently perform the product using the FFT, maintaining the distance-preserving properties in the compressed domain [19], [20]. Moreover, further compression and complexity reduction can be obtained with binary quantization of the compressed database, i.e., reducing the compressed database to the matrix of its signs only

$$\mathbf{A} = \operatorname{sign}(\mathbf{\Phi D}) \qquad \mathbf{y} = \operatorname{sign}(\mathbf{\Phi} \hat{\mathbf{k}}). \tag{6}$$

In the case of binary measurements the correlation coefficient is replaced by the Hamming distance as test metric.

Results in [12] indicate that randomly-projected compressed fingerprints achieve identification performance close to the uncompressed fingerprints at a fraction of the storage space and considerably reduced computational cost for the matching operation, outperforming both the digest and the binarization methods. Moreover, they offer a very flexible scheme, in the sense that they allow to tune the number of projections to the desired tradeoff between speed and accuracy, and can also be adapted to deal with image resizing [21]. They are thus the most promising technique to handle large scale scenarios.

The retrieval problem presented in this paper is essentially an approximate nearest neighbor problem on a large dataset. Some works such as Locality Sensitive Hashing (LSH) [22], [23] focused on avoiding exhaustive search over the N points in a database. LSH uses locality sensitive hash functions and multiple hash tables to achieve a $\mathcal{O}(N^{\rho})$ retrieval complexity, being $\rho < 1$ the ratio between the radius of the ball enclosing the true neighbors and the distance beyond which the other points are supposed to lie. The peculiar characteristics of PRNU patterns do not allow an effective use of LSH and call for other techniques. In fact, fingerprint estimates extracted from one or even multiple photos acquired by the same device present very low correlation [17], thus making the factor ρ very close to 1, and allowing LSH to have only a marginal gain with respect to linear complexity. In this paper, we use random projections for dimensionality reduction purposes, with the aim of relaxing the storage requirements. Thanks to reduced dimensionality, the matching process is indeed faster than using uncompressed fingerprints, but it still requires exhaustive search over the N entries. Moreover, we propose novel techniques, with respect to the methods in the LSH literature, to avoid exhaustive search and increase search speed by specifically tailoring them to the low correlation between fingerprints.

III. Proposed technique

This section discusses techniques to address the image retrieval problem using PRNU fingerprints compressed with a fixed number of random projections.

The system is modeled as in Fig. 1. A collection of N photos is gathered, e.g., by means of an automatic web crawler, and an estimate of the fingerprint is extracted from each photo. This estimate is compressed by means of binary random projections

and stored (additional steps are required for the method presented in III-C). We refer to this collection as the *fingerprint database*, namely the set

$$\mathcal{Y} = \{\mathbf{y}_i : \mathbf{y}_i = \mathcal{Q}(\mathbf{\Phi}\mathbf{k}_i)\}, i = 1, \ldots, N \tag{7}$$

being $\mathbf{k}_i$ the fingerprint estimate of photo i, $\mathbf{\Phi}$ the sensing matrix and $\mathcal{Q}(\mathbf{x}) = \text{sign}(\mathbf{x})$.

A query to the system is composed by a camera identity, which is a fingerprint estimate $\mathbf{k}^q$ extracted from multiple photos of the same camera, as detailed in (3).[1] Its random projections are $\mathbf{a}^q = \mathbf{\Phi}\mathbf{k}^q$ and their binary-quantized version is $\mathbf{y}^q = \mathcal{Q}(\mathbf{a}^q)$.

This scenario well models a camera search engine, in which a camera owner queries the system in order to find webpages containing photos acquired by the presented device. In the following, we address methods to retrieve the photos matching the query, discussing in detail their complexity in terms of storage requirements and matching speed. The ultimate goal is to design a system that can respond to a query in real time, requiring modest hardware resources, even for large values of N, and providing scalability to huge data collections.

A. Linear Search

A straightforward technique to solve the retrieval problem is *linear search*, consisting in comparing the compressed query fingerprint with all N compressed fingerprint estimates in $\mathcal{Y}$. When binary random projections are used to compress the fingerprints, this amounts to evaluating N Hamming distances between binary vectors with m entries. A threshold τ is used for detection purposes, so that the retrieved pictures are those whose Hamming distance is lower than τ.

The threshold can be set according to various criteria. A typical criterion is to model the distribution of the Hamming distance for non-matching fingerprints, and set τ to achieve a predefined false alarm rate. In other words, this method presets a desired average precision, i.e., the average number of photos from a different camera that are erroneously retrieved. As an example, if we model two non-matching compressed fingerprints as realizations of two independent Bernoulli processes with equiprobable outcomes, then their Hamming distance follows a binomial distribution, well approximated by a Gaussian with mean $\frac{m}{2}$ and variance $\frac{m}{4}$. Hence, the false-alarm probability is readily obtained as a function of the threshold τ

$$P_{FA} = \mathbb{P}\left(d_H(\mathbf{y}^q, \mathbf{y}_i) < \tau\right) \approx \frac{1}{2} + \frac{1}{2}\text{erf}\left(\sqrt{\frac{2}{m}}\left(\tau - \frac{m}{2}\right)\right). \tag{8}$$

Another method that can be used to set the threshold is to determine which threshold yields the best F-score in order to balance precision and recall. Calling the recall R (the ratio between the number of retrieved photos acquired by the same camera of the query fingerprint and the overall number of photos of that camera in the database), and the precision P (the ratio between the number of retrieved photos acquired by the same camera of the query fingerprint and the total number of retrieved photos), the F-score is defined as

$$F = 2\frac{PR}{P+R}. \tag{9}$$

The choice of a suitable threshold to maximize the F-score can be performed in a system design phase by means of cross-validation techniques. A reference dataset with available ground truth is partitioned into a training set to extract the high quality fingerprints, a test set of photos used to extract fingerprint estimates from single images to create the database, and a cross-validation set also made of fingerprint estimates but used exclusively to optimize the threshold according to the F-score achieved on it.

Even though random projections are extremely efficient at compressing PRNU patterns, outperforming other methods as shown in [12], it has been observed that a significant number of projections (typically $m \approx 2^{19}$) is required for reliable detection performance. This implies that for a big data scenario, where N can be in excess of hundreds of millions, this kind of exhaustive search is still too complex in terms of number of XOR operations to be performed and in terms of RAM requirements. For example, for a 6 billion database the number of XOR operations is about $3 \cdot 10^{15}$, while the required RAM is about 350 TiB.

B. Hierarchical Search

As a first improvement over linear search we propose the following two-step matching process. A first matching operation selects a small set of candidate entries by means of a coarse but very fast operation. This set of candidates is then analyzed in detail during the second stage. This strategy is inspired by indexing mechanisms used in content-based image retrieval problems (see for example [24]). However, in this case we exploit the scalability of compressed fingerprints [12], since classical indexing techniques are not efficient due to low fingerprint correlation. The first stage of the hierarchical search essentially implements a linear search, using a threshold τ_1, over the full database of N entries, but with a compact descriptor made by a reduced number of binary random projections ($m_1 \ll m$). The second stage works on the reduced subset of N' fingerprints produced by the first stage ($N' \ll N$) but using all the m random projections. This system reduces the number of matching operations to be performed and also relaxes the RAM requirements since it does not require all the mN database bits to be eventually loaded in main memory, but only $m_1 N + mN'$. However, its detection performance will typically be suboptimal with respect to linear search, and, in fact, it hinges on the ability to create a meaningful compact descriptor of m_1 measurements. In the following, we propose two strategies to create such compact descriptor.

1) Non-adaptive hierarchical search: A first strategy to create such compact descriptors would be to simply retain m_1 measurements out of m in fixed positions of $\mathbf{y}_i$ (e.g., and without loss of generality, the first m_1). This method has the advantage of requiring no overhead to encode locations, since the descriptor is generated from fixed positions in the original measurement vectors, thus allowing to preload such descriptors in RAM. Unfortunately, as will be shown in Section IV-C2, this

[1] A lower quality fingerprint extracted from a single photo may be used as well, with an inevitable degradation in the system performance.

method is not robust enough and requires a very high number of measurements in order to have detection performance close to linear search, thus being only marginally faster and marginally less RAM-demanding.

2) Adaptive hierarchical search: A second strategy is to create the descriptors adaptively in order to have better dimensionality reduction properties. When a query fingerprint is presented to the system, its random projections are computed. Before quantizing them to the binary format, m_1 outliers, i.e., the entries with largest magnitude, are identified. The short descriptors are generated by the binary-quantized outliers of the query, to be matched with the measurements in the database in the corresponding locations. This method can better preserve the distances between the fingerprints, achieving better performance than the non-adaptive method with smaller m_1, hence with faster matching speed.

An intuitive explanation of why the adaptive descriptor is more robust than the non-adaptive one is that, under the assumption of zero-mean noise affecting an ideal measurement, the probability that the corrupted measurement is quantized to the opposite bit is given by the tail of the noise distribution, which is smaller if the measurement is far from zero. More formally, suppose that a test measurement a^t is modeled as $a^t = a^q + z$, where a^q is the query measurement and $z \sim \mathcal{N}(0, \sigma^2)$ is the additive Gaussian noise. Then, the probability of a bit flip after binary quantization $\mathcal{Q}(\cdot)$ is

$$\mathbb{P}\left(\mathcal{Q}(a^t) \neq \mathcal{Q}(a^q)\right) = \frac{1}{2}\operatorname{erfc}\left(\frac{|a^q|}{\sigma\sqrt{2}}\right) \tag{10}$$

which is clearly minimized by choosing the largest available $|a^q|$.

The main drawback of this method is that the locations of the outliers are only known once the query has been presented. This implies that, if the database cannot be entirely loaded in RAM, $m_1 N$ bits need to be loaded from disk at each query, thus possibly creating a significant delay which contrasts with our goal of providing real-time responses. In the following sections, we propose two possible strategies to face this problem.

Finally, note that the adaptive method resembles in some way the digest method described in [25], that can be used for compression of the fingerprints. However, if outliers are used for storage purposes, one faces the problem of storing location information as well, which causes a significant overhead. It was shown in [12] that random projections outperform the digest method for this very reason.

C. Fast Adaptive Hierarchical Search

In this section, we generalize the adaptive hierarchical method presented in the previous section, in order to provide a flexible framework that can be optimized by a system designer, according to the specific constraints on RAM usage and read throughput from disk. In the following, we assume that the database is too large to be entirely loaded into the main memory.

As previously mentioned, the main drawback of creating short descriptors using locations selected from the outliers of the query is that the $m_1 N$ bits of the descriptors from the database must be loaded from disk at query-time, causing a delay in the response. The alternative could be keeping in RAM the outliers of the database entries together with their positions, so that the corresponding locations in the query can be chosen accordingly. In this case, no load from disk is needed, but the overhead due to location information may be significant and enforce very large RAM requirements.

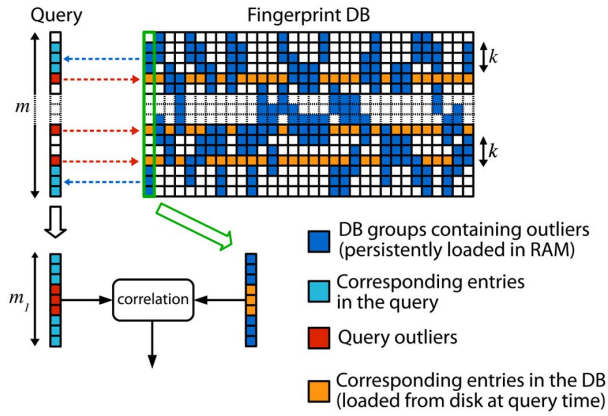

Fig. 2. First-stage matching employs short descriptors created from outliers or group-outliers of the query and of the database entries according to the desired tradeoff between RAM usage and disk load.

A whole spectrum of choices lies between these two extremes, by considering that short descriptors may use some locations derived from outliers of database fingerprints and some other locations derived from outliers of the query, thus creating several trade-offs between how many data are to be read from disk and how many can be stored in RAM.

Furthermore, an additional technique can be used in order to reduce the overhead of location information when using database outliers. Fig. 2 graphically summarizes how the short descriptors are assembled. Instead of selecting single measurements as outliers, we can select *group outliers*, i.e., groups of k measurements likely to contain outliers. This can be done in multiple ways, e.g., choosing the groups with higher variance or using higher order moments like kurtosis. The grouping can be predefined (e.g., groups of contiguous and non-overlapping entries) so that $\lceil \log_2 \frac{m}{k} \rceil$ bits are needed to identify a group using a trivial code. Moreover, since each group contains k measurements, we need fewer groups to reach a predefined quota of measurements to be used in the descriptors.

Let us describe the proposed method more formally. Short descriptors are vectors of m_1 binary measurements. The Hamming distance between the short descriptor of the query and the short descriptor of each and every fingerprint in the database is computed for the first stage of matching. A percentage $r \in [0, 1]$ of the measurements in a short descriptor is obtained from group-outliers of a database fingerprint, thus $m_r = \lceil r m_1 \rceil$ measurements. Each group contains k measurements, which means that $\lceil \frac{m_r}{k} \rceil$ groups are chosen in order to obtain m_r measurements. The remaining $m_1 - m_r$ measurements of the short descriptor are obtained from the locations of the outliers of the query. Overall, the first stage of matching requires to persistently store in RAM

$$R = m_r N + \frac{m_r}{k}\left\lceil \log_2\left(\frac{m}{k}\right)\right\rceil N \tag{11}$$

bits and to load from disk at each query

$$D = (m_1 - m_r)N \tag{12}$$

bits. Additionally, the second stage of matching requires to load the full compressed fingerprints for the N' candidates selected by the first stage, thus requiring to load from disk

$$D' = mN' \tag{13}$$

bits.

D. Index Compression

In the previous section, we remarked that preloading database outliers or group-outliers in RAM has a significant overhead due to the need to store the locations of such outliers. A trivial code requires a total of $\frac{m_r}{k}\lceil \log_2(\frac{m}{k})\rceil$ bits per fingerprint to store location information. Such trivial code is fast in the sense that it immediately gives the position of the outliers in the vector, but it is quite inefficient in terms of space. The entropy of the location information is in fact much lower. When m_r outliers ($k = 1$) have to be identified and their locations stored, we have a total of $\binom{m}{m_r}$ equiprobable ways of selecting them. Generalizing to the use of group-outliers, if the grouping is predefined (e.g., groups of k contiguous measurements) there are $\binom{m/k}{m_r/k}$ equiprobable ways of selecting them. Thus, the entropy of location information is

$$H = \log_2 \left[\binom{\frac{m}{k}}{\frac{m_r}{k}} \right] \tag{14}$$

bits, which is significantly smaller than $\frac{m_r}{k}\lceil \log_2(\frac{m}{k})\rceil$. However, reaching such entropy could be complicated and would require a decoding operation for every fingerprint in the database. Such decoding operation could significantly slow the matching down, so that it might not be worth the savings in memory. Nevertheless, we studied a simple code that gets quite close to the entropy and has a rather fast implementation. The idea is to sort the locations vector by increasing values, encode the first value using a trivial code ($\log_2(\frac{m}{k})$ bits) and then compute the difference between successive location values. Finally, a universal code such as the Exp-Golomb code of order κ [26] is used to encode the differences. Notice that decoding the Exp-Golomb code could be done via a lookup table, so it is rather fast, although $\lceil \frac{m_r}{k} \rceil$ additions are still required to unwrap the differential code. We tested the method on measurements of synthetic Gaussian fingerprints, using $m = 2^{19}$, $m_r = 4096$, various values of k and an Exp-Golomb code of order 6. Table I shows the number of bits needed to encode location information for one fingerprint and compares the value obtained by differential Exp-Golomb coding against the entropy and against the bits needed by the trivial fixed length code. We notice that this simple and rather fast scheme achieves values quite close to the entropy. The good performance can be qualitatively explained by the fact that the distribution of the differences between the sorted locations (normalized by m) is well approximated by a Beta distribution Beta $(1, \frac{m_r}{k})$ (this is a known result on the spacings of order statistics of a continuous uniform distribution, refer to [27]), which has a power law decay well suited for the implicit distribution assumed by universal codes.

TABLE I
INDEX OVERHEAD (BITS)

	k				
	1	**2**	**4**	**8**	**16**
Entropy	34,551	17,272	8,633	4,314	2,154
Diff. Exp-Golomb	35,760	17,868	8,934	4,457	2,222
Trivial	77,824	36,864	17,408	8,192	3,840

E. Scaling to Billion-Entry Datasets

The scalability of the proposed techniques to huge databases of photos, where potentially billions of fingerprints have to be stored, is crucial. We emphasize that the retrieval problem solved by the fast hierarchical search scheme of Section III-C still admits a high degree of parallelization, and is suitable for implementation with standard frameworks for distributed computing on big data, such as MapReduce [28]. In MapReduce, a *Map* function splits a complex task into many individual smaller tasks working in parellel on subproblems. Then, the results of those tasks are gathered and jointly processed by a *Reduce* function. Algorithm 1 reports a high-level pseudocode to implement the photo retrieval problem with the proposed fast hierarchical search using MapReduce. Essentially, the database of N fingerprints is partitioned into J chunks of N_j fingerprints, which can be stored locally on J cluster nodes. The search problem is independently solved for each chunk in parallel by returning J lists of retrieved photos. The final response to the query is provided by simply collating the lists.

Algorithm 1: Hierarchical Search using MapReduce

procedure MAP
  Input: database chunk identifier j; query fingerprint $\mathbf{y}^q$
  Find $m_1 - m_r$ entries of $\mathbf{a}^q$ with largest magnitude and their locations $\mathbf{l}^q$
  Quantize and keep them in $\mathbf{d}^q$
  for $i = 1, \ldots, N_j$ **do**
    Read locations $\mathbf{l}_i$ of group-outliers in RAM
    Quantize $\mathbf{a}^q$ at locations $\mathbf{l}_i$ and append to $\mathbf{d}^q$
    Load $\mathbf{y}_i$ at locations $\mathbf{l}^q$ and keep them in $\mathbf{d}_i$
    Append the m_r group outliers to $\mathbf{d}_i$
    Compute $\delta_i = d_H(\mathbf{d}^q, \mathbf{d}_i)$
  end for
  Find set of candidates $C = \{i : \delta_i < \tau_1\}$
  for $t \in C$ **do**
    Compute $\Delta_t = d_H(\mathbf{y}^q, \mathbf{y}_t)$
  end for
return $Z_j \subseteq C$, $Z_j = \{t : \Delta_t < \tau\}$
end procedure

procedure REDUCE
  Input: Z_j, $j = 1, \ldots, J$; $\zeta = \emptyset$
  for $j = 1, \ldots, J$ **do**
    $\zeta \leftarrow Z_j \cup \zeta$
  end for
return Set of matching fingerprints ζ
end procedure

IV. Experimental Results

The goal of this section is to test the presented techniques to solve the retrieval problem on a large scale database of photos. In the retrieval scenario a large number N of photos is available without any prior knowledge about them (it is not known if some of the photos are from the same camera). A user queries the system with a known fingerprint (*e.g.*, because they own the camera and can extract a high quality fingerprint). The goal of the system is to present to the user all the photos among the ones in the database acquired by the same device as the one having the fingerprint presented by the user. To do it, a fingerprint estimate is independently extracted from each of the N photos and stored in a compressed way. Matching uses the techniques presented in Section III. We also remark that comparisons with other state-of-the-art techniques for fingerprint compression are not present in this paper but can be found in [12], where exhaustive tests show that random projections achieve the best performance.

A. The Flickr Database

A database of cameras and test photos has been assembled by downloading publicly available photos from the photo-sharing website Flickr.[2] A similar experiment [29] also used Flickr to provide source material, but the objective was different. In fact, the article by Goljan *et al.* was mainly concerned with proving that PRNU is an effective fingerprint for camera identification. Thus, a large-scale experiment was needed in order to ascertain the robustness of the fingerprint across a wide range of commercially-available cameras and ordinary photos instead of photos acquired in a lab. Their results showed that in most of the cases non-unique artifacts (i.e., artifacts in the PRNU estimate that are common to all cameras of a particular model) can be effectively suppressed and that the PRNU survives JPEG compression. They also tested a fingerprint matching scenario in which a given photo has to be tested against a database of known fingerprints. Notice that this scenario is exactly dual to the retrieval problem we address in this paper, where the database is composed of a large number of images (and relative fingerprint estimates) while the query is a high-quality fingerprint.

In order to have some control over the type of material, we only selected photos tagged to be at their original resolution and in landscape format. Notice that the landscape format assumption is just for convenience of testing. In fact, ongoing research [21] suggests that random projections can efficiently deal with scale and rotation transformations as well. Whenever Exif data reported the ISO speed, we capped it to 800 to avoid excessively noisy photos. Finally, we assumed that a user only possessed a single camera of a specific model. A total of 549135 images, shot by 1174 cameras, have been processed. The photos from each camera are partitioned into two sets. A first set typically contains between 50 and 100 photos per camera and is used to extract a "high quality" fingerprint estimate, which will be used to simulate a query. We remark that here, with the term "high quality" we mean that the fingerprint is extracted from a set of pictures rather than from a single one. However, there is no guarantee that the content of those pictures is ideal for extracting a fingerprint (e.g., uniform content with unsaturated pixels). The remaining photos constitute the database which has size $N = 421618$.

Even if this test database is smaller than the expected size of a real system, which can be in excess of 10^9 fingerprints, its management is already a big data problem that would be hard to solve without compression, or with other PRNU compression methods not based on random projections. In fact, the database would require approximately 15 TiB of storage if the fingerprints were kept uncompressed and responding to a query would take a very long time to perform the exhaustive matching. Among other proposed compression techniques, the binarization method of Bayram *et al.* [17] would need about 500 GiB and the matching complexity would still scale with the number of pixels of the sensor. Finally, the digest method [25] is quite efficient, but based on the results in [12] we expect about a 10% penalty in terms of memory requirements with respect to random projections. Binary random projections ($m = 524288$, 64 KiB per fingerprint) require about 25 GiB of storage for the database. The costs in terms of RAM occupation and amount of data to be loaded from secondary storage are detailed in the following, depending on the matching technique that is used. Finally, we remark that the size of our Flickr dataset is significantly larger than what is typically used in the literature (*e.g.,* [17] uses 3 cameras and 300 images; [30] studies 70 cameras and 13696 photos; [14] has about 3000 cameras and 200000 images but only some of them are used for tests; the aforementioned [29] is an exception with about one million photos). Indeed, previous works did not consider the retrieval problem, but since we want to scale the use of PRNU fingerprints up to address photo retrieval on large scales, it would not make sense to use just a few cameras and a few hundreds of pictures. Half a million photos are easily managed by a non-optimized implementation of the presented methods and allow experimenting with several choices of system parameters. Since previous works have not studied the retrieval problem, the contributions of this paper are twofold: experimental verification that the problem can be solved at all in a robust manner by using a large real dataset, and that it admits an efficient solution thanks to the proposed techniques.

B. Test Methodology

Each of the 1174 cameras has a training set of approximately 50 pictures. Those pictures are used jointly to extract a high-quality fingerprint according to (3). This fingerprint is then compressed using 2^{19} random projections calculated using a circulant sensing matrix. Both the real-valued measurements and their binary-quantized versions are kept in order to be able to test the methods that require detection of the outliers. The $N = 421618$ test photos are used to assemble the fingerprint database. The noise residual [see (2)] is extracted from each of them and compressed using random projections. Notice that we actually use the noise residual instead of the fingerprint estimate since it was shown in [12] that this choice has practically no impact on the final performance while it allows to save one computation. Camera fingerprints are extracted using the Camera Fingerprint toolbox [29], [31] and all the tests are performed in MATLAB, on an implementation not optimized for speed.

[2][Online]. Available: www.flickr.com

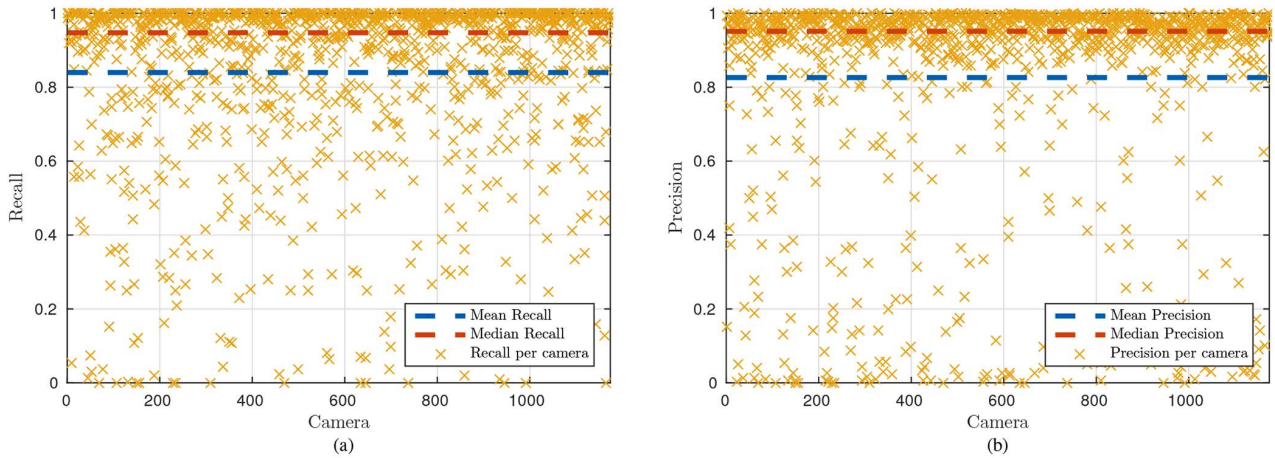

Fig. 3. Recall and Precision of linear search as function of the camera under test. Best F-score threshold: $\tau = 260497$. (a) Recall per camera. (b) Precision per camera.

C. System Performance

In this section we analyze the performance of the proposed compression and matching techniques presented in the previous sections. The chosen performance metrics are *recall* and *precision*. The recall is the number of retrieved photos acquired by the same camera of the query fingerprint, divided by the total number of photos of that camera stored in the database. The precision is the number of retrieved photos acquired by the same camera of the query fingerprint, divided by the total number of retrieved photos. Furthermore, we study the impact of the various system parameters introduced in Section III in terms of primary and secondary storage requirements, as well as their impact on recall and precision. Index compression is not included in the following results because its fast implementation is the subject of future work.

1) Performance of linear search: Linear search has the highest complexity in terms of RAM requirements (the whole database of 25 GiB has to be loaded in RAM at some point), and in terms of amount of calculations (mN XOR operations). However, it offers the best performance in terms of recall and precision. Fig. 3 shows the recall and precision obtained by linear search as a function of the camera used for the query. The threshold τ is selected as the threshold yielding the best F-score. It can be noticed that the system retrieves a very large percentage of photos for most of the query cameras and returning very few false positives. Few cameras display rather low recall and precision. Upon closer inspection of the corresponding photos, they typically present heavy post processing which likely degrades the PRNU estimate. It should be remarked that the database was assembled in an unsupervised way (except for the few rules explained in the previous section), so some troublesome photos were expected. Moreover, it has to be stressed that the "high-quality" query fingerprints are also collected in an unsupervised way from ordinary pictures, possibly heavily post-processed too. It is also possible that some of the cameras exhibiting low performance might be affected by uncommon non-unique artifacts such as the ones discovered in [32]. Nevertheless, the median recall and precision are around 95% meaning that the system is able to achieve such high performance half of the time.

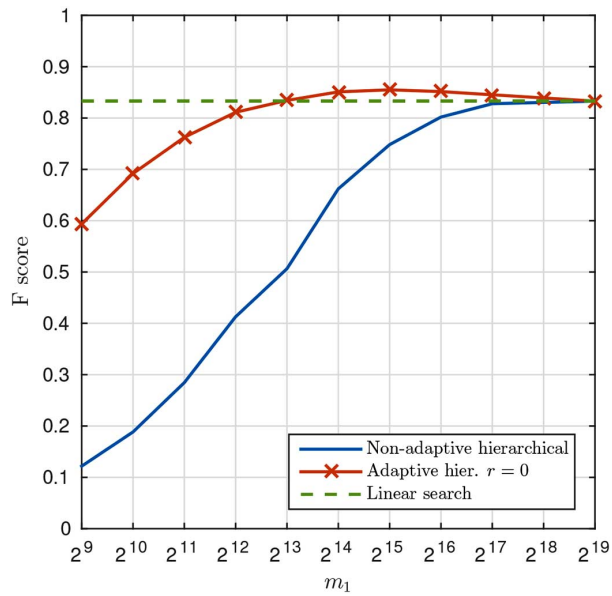

Fig. 4. F-score as function of first stage descriptor length. Non-adaptive chooses first m_1 entries. Adaptive uses $r = 0$. $\tau_1 = [230, 475, 971, 1973, 3979, 8039, 16167, 32458, 65094, 130415, 261091]$ to ensure retained entries are less than 1%.

2) Non-adaptive versus adaptive hierarchical search: A two-stage search was introduced to speed up the matching operation. We are now assessing two methods to create the short descriptors used in the first stage to generate a short list of candidate fingerprints. We compare a non-adaptive method choosing m_1 measurements from fixed predefined locations, and an adaptive method that chooses the m_1 locations from the outliers of the query fingerprint. The first stage threshold τ_1 is chosen to let at most 1% of the database through. Fig. 4 plots the final F-score as a function of m_1 and shows that the non-adaptive method fails to generate descriptors that are short and effective. On the other hand, adaptively choosing the measurements provides a

TABLE II
TOTAL RAM OCCUPATION (MIB)

m_r \ k	1	2	4	8	16
0	412	412	412	412	412
2048	2367	1338	849	618	508
4096	4323	2265	1287	823	605
6144	6341	3238	1782	1086	755
8192	8395	4267	2316	1388	948

TABLE III
DATA LOADED FROM DISK AT FIRST STAGE (MIB)

m_r \ k	1	2	4	8	16
0	412	412	412	412	412
2048	309	309	309	309	309
4096	206	206	206	206	206
6144	103	103	103	103	103
8192	0	0	0	0	0

superior way of dimensionality reduction, quickly reaching performance close to linear search. Notice that the performance of linear search is actually slightly exceeded for some choices of m_1. This is due to the robust embedding properties of adaptively chosen measurements which allow to slightly improve precision by discarding some non-matching fingerprints at the first stage. Once established the importance of an adaptive selection of measurements, we can investigate the problems mentioned in Section III-C of excessive delay due to disk load or excessive RAM consumption.

3) Trading off primary and secondary storage: Using outlier locations from the query fingerprint requires to load part of the database from disk at query-time, because such locations are not known in advance and the full database does not typically fit the main memory. On the other hand, using outliers (or group outliers) of the database fingerprints (found before storing them in binary-quantized form) allows to preload them in RAM, but requires to store their locations as well, which can be a significant overhead. The parameter r trades off the two solutions by specifying that m_r measurements are kept in RAM together with their locations (or group locations if $k > 1$), while $m_1 - m_r$ measurements are loaded from disk at query-time depending on the outliers of the query.

For the following experiments, we fixed $m_1 = 8192$, a value that we deemed satisfactory from the experiment of Fig. 4, a first stage threshold $\tau_1 = 3979$ and a second stage threshold of $\tau = 260497$.

Table II shows the total RAM usage for the Flickr database, for various choices of k and r. The figures include both persistent data, such as the quantized measurements at database group outlier locations, as well as dynamic data, such as the quantized measurements at query outlier locations and the full fingerprints, loaded from disk during the first stage and second stage, respectively. Referring to equations (11), (12) and (13), the figures in Table II show $R + \max(D, D')$ (the maximum is due to the fact that the D bits loaded for the first stage are not needed anymore for the second and can be overwritten).

Table III reports only the amount of data to be loaded from disk at first stage, i.e., D. The choice we made for the threshold τ_1 implies an average disk load at the second stage

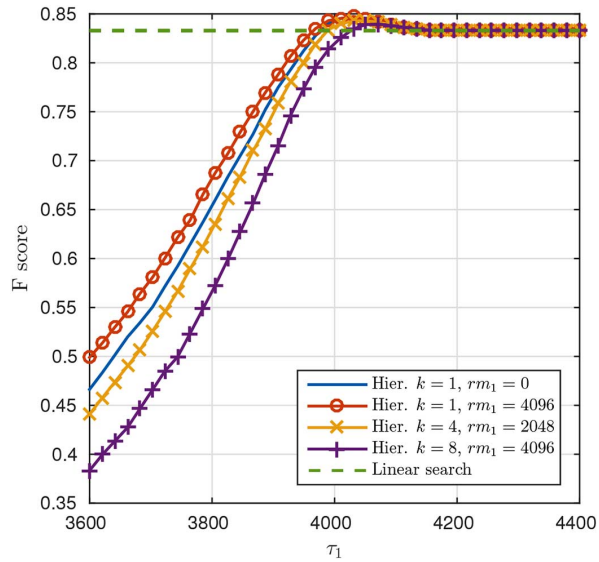

Fig. 5. F-score as function of the first stage threshold τ_1.

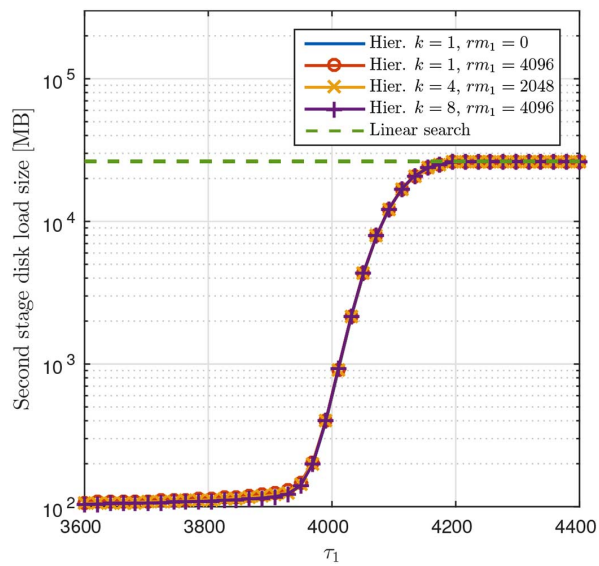

Fig. 6. Size of disk load (no. of retained fingerprints $\times 64$ KiB) for the second stage.

$D' \approx 150$ MiB. Notice that the difference between the figures in Table III and D' may seem small (suggesting that loading from disk at first stage is not a big penalty), but this is only due to the relatively small size of the Flick dataset we use for testing. In fact, it has to be considered that if τ_1 is chosen to retain a fixed maximum number of photos, then D' would be constant, while the figures in Table III would scale as $\mathcal{O}(N)$. The importance of first-stage threshold τ_1 is highlighted in Figs. 5 and 6, where we report the dependence of F-score and of D' on τ_1. Notice that τ_1 should be chosen to yield an F-score as close as possible to the one of linear search, under a constraint on the value of D'. We also observe the same phenomenon as in Fig. 4 that the F-score of the hierarchical system slightly exceeds that of linear search for a small range of τ_1 for certain values of k and r. This is due to a small improvement in precision allowed by the robustness of the short descriptors

TABLE IV
MEAN/MEDIAN RECALL

m_r \ k	1	2	4	8	16
0	0.78 / 0.89	0.78 / 0.89	0.78 / 0.89	0.78 / 0.89	0.78 / 0.89
2048	0.79 / 0.90	0.78 / 0.89	0.77 / 0.87	0.76 / 0.86	0.76 / 0.86
4096	0.79 / 0.90	0.77 / 0.87	0.74 / 0.83	0.73 / 0.82	0.72 / 0.80
6144	0.79 / 0.90	0.75 / 0.84	0.70 / 0.78	0.68 / 0.75	0.66 / 0.73
8192	0.78 / 0.88	0.70 / 0.78	0.62 / 0.67	0.57 / 0.60	0.52 / 0.52

TABLE V
MEAN/MEDIAN PRECISION

m_r \ k	1	2	4	8	16
0	0.90 / 1.00	0.90 / 1.00	0.90 / 1.00	0.90 / 1.00	0.90 / 1.00
2048	0.89 / 1.00	0.90 / 1.00	0.90 / 1.00	0.89 / 1.00	0.90 / 1.00
4096	0.89 / 1.00	0.90 / 1.00	0.90 / 1.00	0.90 / 1.00	0.90 / 1.00
6144	0.89 / 1.00	0.90 / 1.00	0.90 / 1.00	0.90 / 1.00	0.91 / 1.00
8192	0.90 / 1.00	0.90 / 1.00	0.91 / 1.00	0.91 / 1.00	0.91 / 1.00

which discard some non-matching fingerprints at the first stage. Finally, Tables IV and V show the mean and median values of precision and recall for various choices of k and r.

D. System Design Example

In this section, we briefly show how the reported tables can help a system designer in choosing the appropriate values of the parameters. The design objective will be the maximization of the system F-score, under a constraint on the total RAM occupation and on the delay between the submitted user query and the response. For the sake of simplicity, we assume that disk load time significantly exceeds computation time, so that the latter can be considered negligible. For example, let us suppose that our Flickr database is used for a retrieval service hosted on a system with 4 GiB of RAM and 250 MiB/s read throughput from disk and we want a maximum response time of 2 seconds. We can determine the admissible values of k and r depending on the 4 GiB RAM constraint using Table II, while Table III and $D' \approx 150$ MiB allow to determine the admissible values subject to the 2 sec. delay constraint, which in turn means that $D + D' < 500$ MiB. The choice maximizing the F-score is therefore $k = 1$, $m_r = 2048$ providing a median recall of 90% and a median precision of 100%. As a rule of thumb, once determined which values of k and m_r are admitted by the constraints, seeking smaller groups and few measurements in RAM typically provides better F-score.

V. CONCLUSION

In this paper, we addressed an image retrieval problem where the user wants to find all the photos in a collection acquired by a specific device. PRNU fingerprints are an established method for robust camera identification, but their size makes them impractical to use on large scales. We showed how random projections can effectively compress the fingerprints and enable fast matching techniques that are suitable for huge collections of photos at a small cost in terms of detection performance. Moreover, the matching techniques presented in this paper are very flexible and allow several tradeoffs in terms of primary and secondary storage as well as computational complexity. All the techniques were validated with real data on a test dataset of half a million photos from 1174 cameras, assembled in an highly unsupervised way by downloading pictures from the image sharing website Flickr. The experimental results are interesting and novel by themselves, because they show for the first time that the retrieval problem can indeed be solved with good precision and recall on a database of non-trivial size composed by real photos, so that the experiment effectively studies a realistic use case. Moreover, the results suggest that the proposed techniques enable image retrieval based on camera identities on unprecedented scales, effectively paving the way to the realization of a camera search engine spanning huge image collections available on the Internet.

REFERENCES

[1] R. Datta, D. Joshi, J. Li, and J. Z. Wang, "Image retrieval: Ideas, influences, and trends of the new age," *ACM Comput. Surveys*, vol. 40, no. 2, pp. 5:1–5:60, May 2008.

[2] J. Lukáš, J. Fridrich, and M. Goljan, "Determining digital image origin using sensor imperfections," in *Proc. SPIE Electron. Imag., Image Video Commun. Process.*, 2005, vol. 5685, pp. 249–260.

[3] J. Lukas, J. Fridrich, and M. Goljan, "Digital camera identification from sensor pattern noise," *IEEE Trans. Inf. Forensics Security*, vol. 1, no. 2, pp. 205–214, Jun. 2006.

[4] M. Chen, J. Fridrich, M. Goljan, and J. Lukas, "Determining image origin and integrity using sensor noise," *IEEE Trans. Inf. Forensics Security*, vol. 3, no. 1, pp. 74–90, Mar. 2008.

[5] C.-T. Li, "Unsupervised classification of digital images using enhanced sensor pattern noise," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2010, pp. 3429–3432.

[6] R. Caldelli, I. Amerini, F. Picchioni, and M. Innocenti, "Fast image clustering of unknown source images," in *Proc. IEEE Int. Workshop Inform. Forensics Security*, Dec. 2010, pp. 1–5.

[7] B. Bei Liu, H.-K. Lee, Y. Hu, and C.-H. Choi, "On classification of source cameras: A graph based approach," in *Proc. IEEE Int. Workshop Inform. Forensics Security*, Dec. 2010, pp. 1–5.

[8] I. Amerini, R. Caldelli, P. Crescenzi, A. D. Mastio, and A. Marino, "Blind image clustering based on the normalized cuts criterion for camera identification," *Signal Process.: Image Commun.*, vol. 29, no. 8, pp. 831–843, 2014.

[9] O. A. Penatti, E. Valle, R. da, and S. Torres, "Comparative study of global color and texture descriptors for web image retrieval," *J. Vis. Commun. Image Represent.*, vol. 23, no. 2, pp. 359–380, 2012.

[10] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[11] K. Kremerskothen, "6,000,000,000," Aug. 2011 [Online]. Available: http://blog.flickr.net/en/2011/08/04/6000000000/

[12] D. Valsesia, G. Coluccia, T. Bianchi, and E. Magli, "Compressed fingerprint matching and camera identification via random projections," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 7, pp. 1472–1485, Jul. 2015.

[13] J. Fridrich, "Digital image forensics," *IEEE Signal Process. Mag.*, vol. 26, no. 2, pp. 26–37, Mar. 2009.

[14] M. Goljan, J. Fridrich, and T. Filler, "Managing a large database of camera fingerprints," in *Proc. SPIE, Media Forensics Security II*, 2010, vol. 7541, pp. 754 108–754 108–12.

[15] Y. Hu, B. Yu, and C. Jian, "Source camera identification using large components of sensor pattern noise," in *Proc. 2nd Int. Conf. Comput. Sci. Appl.*, Dec. 2009, pp. 1–5.

[16] Y. Hu, C.-T. Li, Z. Lai, and S. Zhang, "Fast camera fingerprint search algorithm for source camera identification," in *Proc. 5th Int. Symp. Commun. Control Signal Process.*, May 2012, pp. 1–5.

[17] S. Bayram, H. Sencar, and N. Memon, "Efficient sensor fingerprint matching through fingerprint binarization," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 4, pp. 1404–1413, Aug. 2012.

[18] W. B. Johnson and J. Lindenstrauss, "Extensions of Lipschitz mappings into a Hilbert space," *Contemporary Math.*, vol. 26, pp. 189–206, 1984.

[19] A. Hinrichs and J. Vybral, "Johnson-Lindenstrauss lemma for circulant matrices," *Random Strucures. Algorithms*, vol. 39, no. 3, pp. 391–398, Oct. 2011.

[20] H. Rauhut, "Circulant and Toeplitz matrices in compressed sensing," in *Proc. Signal Process. Adapt. Sparse Structured Representations*, 2009, pp. 1–6.

[21] D. Valsesia, G. Coluccia, T. Bianchi, and E. Magli, "Scale-robust compressive camera fingerprint matching with random projections," in *Proc. Int. Conf. Acoust., Speech Signal Process.*, Apr. 2015, pp. 1697–1701.
[22] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proc. 30th Annu. ACM Symp. Theory Comput.*, 1998, pp. 604–613.
[23] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proc. 20th Annu. Symp. Comput. Geometry*, 2004, pp. 253–262.
[24] J. Wang, H. Zha, and R. Cipolla, "Coarse-to-fine vision-based localization by indexing scale-invariant features," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 36, no. 2, pp. 413–422, Apr. 2006.
[25] M. Goljan and J. Fridrich, "Sensor fingerprint digests for fast camera identification from geometrically distorted images," in *Proc. SPIE, Media Watermark., Security, Forensics*, 2013, vol. 8665, pp. 86 650B–86 650B–10.
[26] J. Teuhola, "A compression method for clustered bit-vectors," *Inf. Process. Lett.*, vol. 7, no. 6, pp. 308–311, 1978.
[27] H. A. David and H. N. Nagaraja, *Order Statistics*. New York, NY, USA: Wiley, 1970.
[28] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008.
[29] M. Goljan, J. Fridrich, and T. Filler, "Large scale test of sensor fingerprint camera identification," in *Proc. SPIE, Media Forensics Security*, 2009, vol. 7254, pp. 72 540I–72 540I–12.
[30] Y. Hu, C.-T. Li, and Z. Lai, "Fast source camera identification using matching signs between query and reference fingerprints," *Multimedia Tools Appl.*, pp. 1–24, 2014.
[31] *Camera Fingerprint Software*. Univ. Binghamton Digit. Data Embedding Lab., Binghamton, NY, USA, Jan. 2012 [Online]. Available: http://dde.binghamton.edu/download/camera_fingerprint/
[32] T. Gloe, S. Pfennig, and M. Kirchner, "Unexpected artefacts in PRNU-based camera identification: A 'Dresden Image Database' case-study," in *Proc. Multimedia Security*, 2012, pp. 109–114.

Diego Valsesia (S'12) received the M.Sc. degree in telecommunications engineering from the Politecnico di Torino, Torino, Italy, and the M.Sc. in electrical and computer engineering from the University of Illinois at Chicago, Chicago, IL, USA, both in 2012. He is currently working toward the Ph.D. degree in electronics and telecommunications at the Politecnico di Torino.

His current research interests include compression of remote sensing images, compressed sensing, and sparse representations.

Giulio Coluccia (S'06–M'09) received the B.Sc. degree in 2003 and the M.Sc. degree in telecommunications engineering in 2005, both from the Politecnico di Torino, Torino, Italy, and the Ph.D. degree in electronic and communications engineering from the Electronics Department of the Politecnico di Torino, Torino, Italy, in 2009.

He is currently a Post Doctoral Researcher with the Image Processing Lab, Politecnico di Torino, Torino, Italy. He is currently involved in the 5-year ERC project "CRISP - Towards compressive information processing systems," funded by the European Research Council. His research interests include compressed sensing, with particular interest in its application to image processing and forensics, multidimensional signals, and to distributed source coding and wireless sensor networks.

Tiziano Bianchi (S'03–M'05) received the M.Sc. degree (Laurea) in electronic engineering and the Ph.D. degree in information and telecommunication engineering from the University of Florence, Florence, Italy, in 2001 and 2005, respectively.

From 2005 to 2012, he was with the Department of Electronics and Telecommunications, University of Florence, Florence, Italy, as a Research Assistant. Since December 2012, he has been an Assistant Professor with the Department of Electronics and Telecommunications, Politecnico di Torino, Torino, Italy. He has authored or coauthored over 100 papers in international journals and conference proceedings. His research interests include signal processing in communications and processing of SAR images, multimedia security technologies, signal processing in the encrypted domain, and security aspects of compressed sensing.

Enrico Magli (S'98–M'01–SM'07) received the M.Sc. and Ph.D. degrees from the Politecnico di Torino, Torino, Italy, in 1997 and 2001, respectively.

He is currently an Associate Professor with the Politecnico di Torino, Torino, Italy. His research interests include compressive sensing, image and video coding, and vision.

Dr. Magli is an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, the IEEE TRANSACTIONS ON MULTIMEDIA, and the *EURASIP Journal on Image and Video Processing*. He is an IEEE Distinguished Lecturer for 2015–2016, and a corecipient of the IEEE Geoscience and Remote Sensing Society 2011 Transactions Prize Paper Award.