Catch Me in the Dark: Effective Privacy-preserving Outsourcing of Feature Extractions over Image Data

***Abstract*—Advances in cloud computing have greatly motivated data owners to outsource their huge amount of personal multimedia data and/or computationally expensive tasks onto the semi-trusted cloud by leveraging its abundant resources for cost saving and flexibility. From the privacy perspective, however, the outsourced multimedia data and its originated applications may reveal the data owner's private information, such as the personal identity, locations or even financial profiles. This observation has recently aroused new research interest on privacy-preserving computations over outsourced multimedia data. In this paper, we propose an effective privacy-preserving computation outsourcing protocol for the prevailing scale-invariant feature transform (SIFT) over massive encrypted image data. We first show that previous solutions to this problem have either efficiency/security or practicality issues, and none can well preserve the important characteristics of the original SIFT in terms of distinctiveness and robustness. We for the first time present a new privacy-preserving outsourcing protocol for SIFT with the preservation of its key characteristics, by randomly splitting the original image data, carefully distributing the feature extraction computations to two independent cloud servers and further leveraging the garbled circuit for secure keypoints comparisons. We both carefully analyze and extensively evaluate the security and effectiveness of our design. The results show that our solution is practically secure, outperforms the state-of-the-art and performs comparably to the original SIFT in terms of various characteristics, including rotation invariance, image scale invariance, robust matching across affine distortion and change in 3D viewpoint.**

I. Introduction

With the increasing popularity of cloud-based data services, data owners are highly motivated to store their huge amount of (potentially sensitive) personal multimedia data files and computationally expensive tasks onto remote cloud servers. While enjoying the abundant storage and computation resources for cost saving and flexibility, the outsourcing of data storage and computation to the cloud also raises great security and privacy concerns due to the different trust domains the data owner and the cloud belong to [1]. More specifically, popular social network providers like Facebook and Flickr commonly exploit the outsourced personal image data to conduct behavioral advertising and preference analytics for improving user experience. To do this, instead of directly working on the massive data sets, they extract image features as inputs to well-defined data mining models. On the other side, from the data owner's perspective, to relieve the heavy computation workload in image feature extraction and utilization on the "Big Data" (e.g., massive satellite images) for building versatile user-defined applications such as similarity search indexes, more and more users are inclined to outsource the computation of image feature extractions to the cloud. Obviously, the expose of original image data to the semi-trusted cloud service provider may inevitably reveal the data owner's private information, such as the personal identity, locations or even financial profiles etc. To provide privacy guarantees for sensitive data, a straightforward approach is to encrypt the sensitive multimedia data locally before outsourcing. While providing strong end-to-end privacy, encryption also becomes a hindrance to data computation or utilization. Now the challenging problem is *how to enable privacy-preserving image feature extractions over massive image data while apparently relieving the database owner of its high computation burden and relying on the cloud for providing fast and effective image feature extraction service.*

In the existing literature, the problem of privacy-preserving outsourcing of computations (e.g., modular exponentiation [2], linear programming [3], and keyword-based search [4] etc.) over various data types, such as text data, numerical data, spatial data etc., has been extensively investigated in recent years. However, the problem of privacy-preserving computation outsourcing over multimedia data has received limited research attention so far. Only in recent years, privacy-preserving data search in the ciphertext domain has been extended to content-based multimedia retrieval [5] and face recognition [6]. Due to the importance of image feature extraction in multimedia data processing and its heavy operations on massive data, especially for satellite data set for its tremendous size and feature points, the extraction or detection of image features from the ciphertext domain has began to attract more and more research interest.

To the best of our knowledge, Hsu et al. [7] was the first to investigate privacy-preserving SIFT in the encrypted domain by utilizing homomorphic encryption. However, their solution is either computationally intractable or insecure from the privacy perspective. In [8], Wang et al. considered the problem of secure and private outsourcing of shape-based feature extraction and proposed two approaches with different levels of security by using homomorphic encryption and the garbled circuit protocol, respectively. Following [9], Qin et al. revisited the privacy-preserving SIFT outsourcing problem and proposed an efficient scheme by utilizing splitting based encryption with three independent servers, order preserving encryption (OPE) and random permutation. While putting great effort on the privacy or efficiency aspect, one common limitation of the previous solutions is that they all lack of comprehensive evaluations with respect to the preservation of the key characteristics of its corresponding original image feature extraction algorithm. In other words, whether these solutions can well preserve the important characteristics of

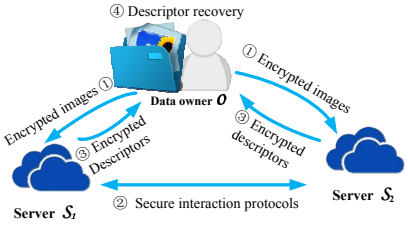


Fig. 1: System model.

the original SIFT in terms of distinctiveness and robustness or not remains in question.

Motivated by the above observations, in this paper, we propose an effective privacy-preserving computation outsourcing protocol for the prevailing scale-invariant feature transform (SIFT) over massive encrypted image data. We additively split the image randomly into two shares and distribute them to two independent cloud servers which can perform the accumulation process locally. We then design a new protocol by leveraging the garbled circuit, letting the two servers collaboratively detect real locations of keypoints via the difference-of-Gaussian (DoG) scale space built on encrypted image. After analyzing the structure of the orientation histogram, we also design a protocol to build encrypted versions of it through computing the orientation range instead of their specific values, which enables the servers to detect real main orientations. Finally, by exploiting the additive property of the encrypted image, the data owner can recover the real feature descriptors from the encrypted descriptors generated by the two servers while preventing them from seeing any information about the original image. Our main contributions can be summarized as follows.

1. We examine the state-of-the-art solutions and show their insufficiencies and efficiency/security weaknesses when meeting the practical needs of privacy-preserving outsourcing computations and applying to real-world image feature extraction-based applications.

2. We for the first time propose a new and effective privacy-preserving outsourcing protocol for SIFT with the preservation of its key characteristics, by randomly splitting the original image data, carefully distributing the feature extraction computations to two independent cloud servers and further leveraging the garbled circuit for secure keypoints comparisons.

3. We carefully analyze our protocol by showing why it can preserve much well the important characteristics of the original SIFT in terms of distinctiveness and robustness as compared to the existing solutions. We also provide both a detailed security analysis and an extensive privacy evaluation to demonstrate the privacy-preserving guarantee of our design.

4. We evaluate our protocol comprehensively with real-world massive image dataset. The results show that our solution is practically secure, outperforms the state-of-the-art and performs comparably to the original SIFT in terms of various characteristics, including rotation invariance, image scale invariance, robust matching across affine distortion and change in 3D viewpoint.

II. Problem Statement

A. System Model

In our work, we consider a cloud-based image feature extraction outsourcing computation system involving three parties: the data owner $\mathcal{O}$, the cloud server $\mathcal{S}_1$, the cloud server $\mathcal{S}_2$ (as illustrated in Fig. 1). In this application scenario, we assume that the data owner $\mathcal{O}$ holding a large volume of sensitive image files (e.g., satellite images that are usually of extremely large size and each has plenty of keypoints) is resource-constrained. Thus, $\mathcal{O}$ would like to outsource both the image data set and the computation-intensive SIFT task to the cloud by leveraging its abundant storage and computation resources. We assume $\mathcal{S}_1$ and $\mathcal{S}_2$ are independent with each other and can be considered to belong to two independent cloud service providers (Note that for ease of exposition, we use $\mathcal{S}_1$ and $\mathcal{S}_2$ to represent two independent entities, but each entity may comprise a cluster of servers, e.g., representing multiple computing instances in Amazon EC2 Instance Configuration). To protect the privacy, $\mathcal{O}$ will first encrypt each image set and then distribute the ciphertexts to $\mathcal{S}_1$ and $\mathcal{S}_2$. After running SIFT algorithm in the ciphertext domain via a sequence of secure interaction protocols, $\mathcal{S}_1$ and $\mathcal{S}_2$ will return the encrypted feature descriptors to the data owner, who can eventually recover the real feature descriptors from their encrypted versions.

B. Threat Model

As discussed above, $\mathcal{S}_1$ and $\mathcal{S}_2$ are assumed to be two independent or say non-colluding servers. Note that, as shown in [9], delegating all image feature extraction tasks to only one cloud entity will obviously lead to the leakage of pixel values of an image. Hence, it is necessary to have at least two independent entities for achieving privacy-preserving image feature extraction outsourcing. To this end, three independent servers are strictly required in [9] while our protocol is implemented with only two ones, which can be more practical.

Following all existing privacy-preserving data/computation outsourcing protocols [2], [4], [7]–[9], we also assume the cloud is semi-trusted (which can be seen as the adversary in our application scenario), i.e., $\mathcal{S}_1$ and $\mathcal{S}_2$ will execute the protocol as specified but may try to learn additional private information from the encrypted image data and all the intermediate results generated during the protocol execution. In addition, instead of knowing the pixels' specific values, the adversary may try to deduce the image content by leveraging i) relationships between pixels (e.g., keypoint locations) and/or ii) encrypted feature descriptors.

C. Background

Scale-invariant feature transform (**SIFT**). In the area of computer vision and pattern recognition, scale-invariant feature transform (SIFT) is an important feature extraction algorithm that has been widely used due to its distinctiveness and robust matching across a substantial range of affine distortion, addition of noise, and change in illumination [10]. According

to SIFT [10], image features are extracted by the following three stages:

Scale-space extrema detection. For an initial image $\mathcal{I}$, a difference-of-Gaussian scale space is built by convolving $\mathcal{I}$ with a difference-of-Gaussian (DoG) function as

$$\begin{aligned}\mathcal{D}(x,y,\sigma) =&(\mathcal{G}(x,y,k\sigma) - \mathcal{G}(x,y,\sigma)) * \mathcal{I}(x,y)\\ =&\mathcal{L}(x,y,k\sigma) - \mathcal{L}(x,y,\sigma),\end{aligned} \tag{1}$$

where $\mathcal{L}(x,y,\sigma) = \mathcal{G}(x,y,\sigma) * \mathcal{I}(x,y)$ is the smoothed image and $*$ is the convolution operation in x and y. In order to detect the local extrema of $\mathcal{D}(x,y,\sigma)$, each sample point is compared to its 26 neighbors in the current and adjacent scales. If it is larger or smaller than all of them, it will be selected as a candidate keypoint.

Orientation assignment. One or more orientations will be assigned to each keypoint based on the local image gradient direction. In this way, the keypoint can be represented relative to its orientation and therefore achieve invariance to image rotation. Specifically, for the Gaussian smoothed image $\mathcal{L}(x,y)$ that a keypoint lies in, if we denote $\mathsf{Diff}_x = \mathcal{L}(x+1,y) - \mathcal{L}(x-1,y), \mathsf{Diff}_y = \mathcal{L}(x,y+1) - \mathcal{L}(x,y-1)$, the gradient magnitude $\mathsf{m}(x,y)$ and orientation θ are computed as $\mathsf{m}(x,y) = \sqrt{\mathsf{Diff}_x^2 + \mathsf{Diff}_y^2}$ and $\theta(x,y) = \tan^{-1}\frac{\mathsf{Diff}_y}{\mathsf{Diff}_x}$.

Then, an orientation histogram $\mathcal{H}$ with 36 bins covering 360-degree range is formed from the gradient orientations of sample points within a region around the keypoint. Each sample added to the histogram is weighted by its gradient magnitude and by a Gaussian-weighted circular window. Finally, the highest peak in the histogram is detected and any other local peaks that are within the 80% of the highest peak are also used to create a keypoint with that orientation.

Keypoint descriptor. After being rotated relative to the keypoint orientation, a keypoint descriptor is created by computing the gradient magnitude and the orientation for each sample point in a region around the keypoint. These samples are weighted by a Gaussian window and then accumulated into an orientation histogram with 8 bins covering 360-degree range over 4×4 subregions. Finally, a 128-dimensional feature descriptor is formed by using a vector to represent its 16 histograms.

Garbled circuits. Yao for the first time introduced the idea of using garbled circuits for securely evaluating two-party functionalities [11]. Garbled circuits enable two parties P_1 and P_2, holding their private inputs x_1 and x_2, respectively, to compute $f(x_1, x_2)$ for an arbitrary function f. After the evaluation of the function, neither of the parties will learn anything about the other party's input other than what it can infer from his own input and the outcome. The key idea of garbled circuits is to let one party represent boolean wire values on each wire with a cryptographic key (called that wire's *wire label*) and replace each gate's truth table with a corresponding garble gate, which is constructed by encrypting the gate's outgoing wire labels using an appropriate combination of the two input wire labels. On the other side, the second party obtains the input wire labels using oblivious transfer, based on which it can evaluate the rest of the circuit without any further communication. For each garbled gate, the second party can decrypt exactly one entry for the outgoing wire based on the wire labels it knows for the two input wires. The first party also sends the mapping from wire labels to boolean values for any output wires, so the second party can map its final out-wire labels to boolean values. Due to the space limitation, readers can refer to [12] for a detailed review of Yao's protocol and a complete and rigorous security proof.

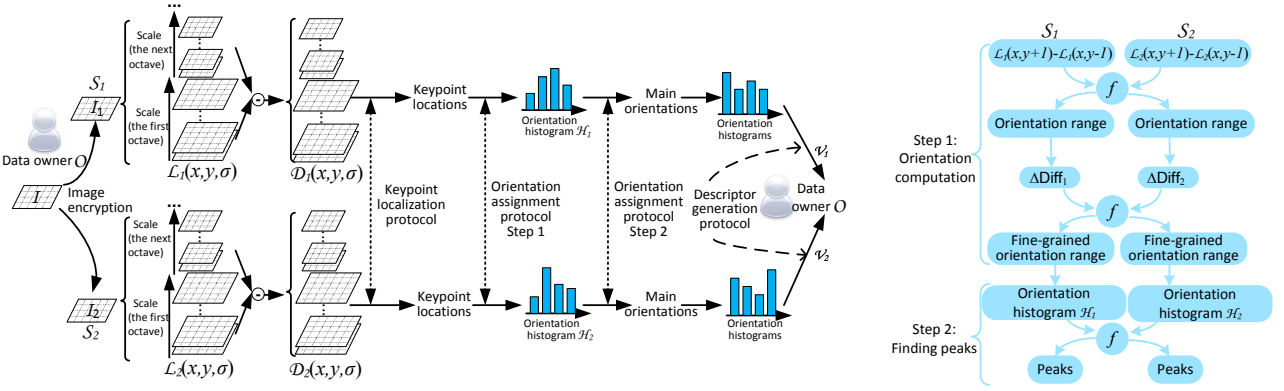
In this paper, we define f to be a comparison function, that is f will output 1 if $x_1 \geqslant x_2$; otherwise output 0. The outcome of f will finally be revealed to both parties. Our motivation to use garbled circuit is two-fold. First, it is provably secure [12]. Second, garbled circuit has been shown for its efficiency to securely evaluate simple comparison functions in existing literature [13]. In this work, we will adopt the approach in [13] as a building block and fully leverage the powerful cloud to implement our secure outsourcing of SIFT algorithm.

III. Privacy-Preserving Outsourcing of SIFT: An Examination of the State-of-the-Art

In this section, we provide a detailed analysis of two most closely-related works, which reflects the most recent research progress and results in privacy-preserving outsourcing of SIFT algorithm. A careful analysis of these solutions (in terms of security/efficiency and the effectiveness of outsourcing in preservation of original SIFT's key characteristics) motivates us to seek a new solution.

1) HE-SIFT in [7]: Hsu et al. [7] was the first to investigate the problem of secure SIFT outsourcing and proposed a solution that enables the cloud server to run SIFT algorithm in the encrypted domain. Their main idea is to encrypt the initial image by using Paillier cryptosystem [14], which provides additive homomorphism and plaintext multiplication. By leveraging the above two properties, all the operations of original SIFT, including common addition, multiplication and comparison, can be done over ciphertexts. However, a careful analysis of this solution shows that their scheme has two main problems that make it inapplicable to real-world applications.

First, it introduces unrealistic computation complexity on the encrypted data comparison. In the comparison protocol of [7], each comparison needs to traverse half of the plaintext space between any two thresholds (Note that the plaintexts and thresholds are transformed to or chosen as integers due to the use of Paillier cryptosystem). Take the following application scenario as an example: assuming that 10 thresholds are chosen and the secret keys p and q are 1000-bit primes (to ensure the security of the Paillier cryptosystem), the plaintext space $\mathbb{Z}_N$ ($N = pq$) has about 2^{2000} elements, and thus it will require $2^{2000}/(2 \times 10) > 2^{1995}$ comparison operations. Apparently, it puts an infeasible computation effort on the cloud even if today's fastest supercomputer is employed. Besides, as the user needs to encrypt and decrypt each pixel in the image, by using a local machine with an Intel Xeon CPU running at 2.0GHz and a memory of 4GB, each encryption will cost at least 0.02 seconds for a 512-bit modulus of the Paillier cryptosystem. For an image with size of 300×300,



(a) System Overview. (b) A sketch of orientation assignment protocol.

Fig. 2: System and OSP overviews.

the total encryption time will be about 30 minutes. So, it also puts huge intolerable workload for the data owner, which only has limited computation resource. On the other hand, if the complexity of the comparison protocol is reduced by choosing a smaller plaintext domain, the security will be broken. For example, if the secret keys p and q are 16-bit primes, thus the plaintext space will have 2^{32} elements which makes the comparison protocol computationally feasible. However, due to the small sizes of p and q, it can factor N and break the whole system by successfully launching a brute-force attack.

Second, the content of the image can be deduced by the cloud server. This is because the server has the ability to compare two arbitrary cipertexts by the proposed comparison strategy. Thus, it can order all the pixels and then traverse all the possible values in the range $[0, 255]$ (assuming that each pixel has 8 bits) to recover the image content. For an image with size of $n \times n$ that has n^2 elements in total, a quick sorting algorithm has time complexity $O(n^2 \log n^2)$ and the traversing step is only $O(n^2)$. For the cloud server, the time complexity of launching this attack is polynomial-time.

2) Privacy-Preserving Outsourcing of SIFT in [9]: As a following work, Qin et al. [9] proposed a new and efficient secure SIFT outsourcing protocol. The authors argued that the locations of keypoints are protected from revealing to the cloud. However, we show that this is not true and it violates their security statements. In [9], to protect the locations of keypoints, it generates a set of dummy keypoints and mixes them with the real ones. To measure its security level, a probability bound that the adversary can choose a real keypoint from the mixed keypoint group is given, i.e., $P = \frac{|r|}{|r|+|d|}$, where $|r|$ denotes number of real keypoints and $|d|$ denotes number of dummy keypoints. In practice, the maximum value of $|d|$ is about 100 times larger than the value of $|r|$, which leads to $P \approx 1\%$. For an typical image of size 400×400 pixels, about 2000 keypoints will be generated, but the number of dummy keypoints is certainly less than the number of total pixels in the image, say about 160000. Thus, the probability P is approximately $\frac{2000}{2000+160000} \approx 1.2\%$. Strictly speaking, however, this probability is *non-negligible* at all! That implies that the cloud still has a good probability of finding the real locations of keypoints.

In addition to the above limitations, the most serious problem with both of the two schemes is that their protocol designs accidentally destroy the robustness and distinctiveness of original SIFT, whose original steps are intentionally modified in order to operate over the ciphertexts. As described in Section II-C, in the original SIFT algorithm, each keypoint is assigned with the image location, the scale and the orientation to provide invariance to these parameters, and the computation of the keypoint descriptor within a local region of the image makes it highly distinctive. In fact, these properties explain why SIFT outperforms other feature extraction methods and prevails in object recognition. In [7] and [9], however, the authors have both eliminated the orientation assignment in the stage 2 (*Orientation assignment*) and created descriptors without sampling in a region in the stage 3 (*Keypoint descriptor generation*). These modifications render their systems useless in image matching, as will be shown by our experimental results in Section VI.

IV. Privacy-Preserving Outsourcing of SIFT: Our Construction

In this work, our ultimate goal is to *enable privacy-preserving SIFT outsourcing computation while preserving all its key characteristics*. To this end, we have the following specifical goals. First, *security*, i.e., the privacy of the original image content should be protected as much as possible from the adversary. Second, *effectiveness*, i.e., the extracted feature vectors generated by the outsourced SIFT should preserve the key characteristics as much as possible. Third, *efficiency*, i.e., the computations after outsourcing on the data owner side should be substantially reduced compared with performing SIFT locally, and the computation burden on the cloud side should be practically reasonable considering its capabilties. Our implementation consists of a series of protocols as illustrated in Fig. 2 (a).

A. Image Encryption

For an original image $\mathcal{I}$ with size of $n \times n$ pixels, we use the matrix $\mathcal{I}(x, y)$ $(0 \leqslant x, y < n)$ with each entry of 8-bit length.

The data owner randomly selects n^2 integers from $[0, 255]$ to generate a random matrix $\mathcal{I}_2(x, y)$ and then encrypt $\mathcal{I}(x, y)$ by computing $\mathcal{I}_1(x, y)$ as: $\mathcal{I}_1(x, y) = \mathcal{I}(x, y) + \mathcal{I}_2(x, y)$. The ciphertexts $\mathcal{I}_1$ and $\mathcal{I}_2$ are sent to $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively.

B. Keypoint Localization Protocol

After receiving ciphertexts $\mathcal{I}_i$ from $\mathcal{O}$, $\mathcal{S}_i$ creates Gaussian spaces $\mathcal{L}_i(x, y, \sigma)$ $(i \in \{1, 2\})$ by conducting convolution and downsampling operations on it with the variable-scale Gaussian $\mathcal{G}(x, y, \sigma)$. Specifically, $\mathcal{S}_1$ will compute

$$\mathcal{L}_1(x, y, \sigma) \quad = \quad \mathcal{G}(x, y, \sigma) * \mathcal{I}_1(x, y), \tag{2}$$

where $*$ denotes the convolution operation in x and y, and $\mathcal{G}(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$. As shown in Fig. 2, for each octave of the scale space, the original image is repeatedly convolved with $\mathcal{G}(x, y, \sigma)$ by Eq. (2) to produce the set of images in the scale space. After generating each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeats. Note that the number of octaves (which determine the repetition times) and number of scales per octave are pre-defined by the data owner.

Then, $\mathcal{S}_1$ further compute the difference-of-Gaussian function $\mathcal{D}_1(x, y, \sigma)$ from the substraction of adjacent Gaussian images as

$$\mathcal{D}_1(x, y, \sigma) = \mathcal{L}_1(x, y, r\sigma) - \mathcal{L}_1(x, y, \sigma),$$

where r is a constant multiplicative factor pre-defined by the data owner. In the meantime, $\mathcal{S}_2$ will do the same operations for $\mathcal{I}_2(x, y)$ and obtain $\mathcal{D}_2(x, y, \sigma)$. Note that both servers operate in the same order to generate a homologous scale space.

Next, in the process of extremum detection, each sample point is compared with its 26 neighbors in the current and adjacent scales in the original difference-of-Gaussian scale space $\mathcal{D}(x, y, \sigma)$. If it is larger or smaller than all of them, it will be selected as a candidate keypoint. To determine which is larger between points $\mathcal{D}(x, y, \sigma)$ and $\mathcal{D}(x, y + 1, \sigma)$, $\mathcal{S}_1$ and $\mathcal{S}_2$ will compute $\Delta\mathcal{D}_1 = \mathcal{D}_1(x, y, \sigma) - \mathcal{D}_1(x, y + 1, \sigma)$ and $\Delta\mathcal{D}_2 = \mathcal{D}_2(x, y, \sigma) - \mathcal{D}_2(x, y + 1, \sigma)$, respectively. Then, they will work together to compute the comparison function $f(\Delta\mathcal{D}_1, \Delta\mathcal{D}_2)$ by using the garbled circuit [13]. If the output of f is 1, which means $\Delta\mathcal{D}_1 \geqslant \Delta\mathcal{D}_2$, we will claim that $\mathcal{D}(x, y, \sigma) \geqslant \mathcal{D}(x, y + 1, \sigma)$, otherwise $\mathcal{D}(x, y, \sigma) < \mathcal{D}(x, y + 1, \sigma)$.

Remarks. To reduce the rounds of interactions between them, $\mathcal{S}_1$ and $\mathcal{S}_2$ can compute $\Delta\mathcal{D}_1$ and $\Delta\mathcal{D}_2$ between all neighboring two points in the scale space simultaneously. Then $\mathcal{S}_1$ can encrypt and transmit all $\Delta\mathcal{D}_1$'s to $\mathcal{S}_2$ (using garbled circuits) in a one-time transmission. Consequently, the rounds of interactions required in detecting all the keypoints are a constant.

C. Orientation Assignment Protocol

To give a better illustration of this protocol, we first provide a sketch that depicts the workflow in Fig. 2 (b).

1) Orientation Computation: Once the location of a keypoint is determined, the gradient magnitude and the orientation should be computed according to pixel differences so as to build an orientation histogram. The scale of a keypoint $\mathcal{D}(x, y, \sigma)$, associated with a parameter σ, is used to select the Gaussian smoothed image $\mathcal{L}$ that has the closest scale, so all the following computations are performed in a scale-invariant manner. Thus, for ease of expression we will omit parameter σ in the following discussions.

For the orientation computation on the keypoint $\mathcal{L}(x, y)$ (which is mapped back from $\mathcal{D}(x, y)$), $\mathcal{S}_1$ and $\mathcal{S}_2$ will first determine which is larger between $\mathcal{L}(x, y+1)$ and $\mathcal{L}(x, y-1)$, and between $\mathcal{L}(x + 1, y)$ and $\mathcal{L}(x - 1, y)$, respectively. Here, we set two variables α and β as: $\alpha = 1$ if $\mathcal{L}(x, y + 1) \geqslant \mathcal{L}(x, y - 1)$; otherwise $\alpha = -1$, and $\beta = 1$ if $\mathcal{L}(x + 1, y) \geqslant \mathcal{L}(x - 1, y)$; otherwise $\beta = -1$. Secure comparisons of the two pairs can be achieved by computing the functions $f(\mathcal{L}_1(x, y + 1) - \mathcal{L}_1(x, y - 1), \mathcal{L}_2(x, y + 1) - \mathcal{L}_2(x, y - 1))$ and $f(\mathcal{L}_1(x+1, y) - \mathcal{L}_1(x-1, y), \mathcal{L}_2(x+1, y) - \mathcal{L}_2(x-1, y))$ by using the garbled circuit. As a result, which range the orientation lies in can be obtained, i.e., $0^\circ \sim 90^\circ, 90^\circ \sim 180^\circ, 180^\circ \sim 270^\circ, 270^\circ \sim 360^\circ$. To get a more fine-grained direction range, $\mathcal{S}_1$ computes

$$\begin{aligned} \Delta\mathsf{Diff}_1 &= \mathsf{Diff}_{1y} - k\mathsf{Diff}_{1x} \\ &= \alpha(\mathcal{L}_1(x, y + 1) - \mathcal{L}_1(x, y - 1)) \\ &\quad - k\beta(\mathcal{L}_1(x + 1, y) - \mathcal{L}_1(x - 1, y)). \end{aligned}$$

On the other side, $\mathcal{S}_2$ will get $\Delta\mathsf{Diff}_2 = \mathsf{Diff}_{2y} - k\mathsf{Diff}_{2x}$, where k is a constant used to determine the orientation interval. Then, $\mathcal{S}_1$ and $\mathcal{S}_2$ will involve in garbled circuit computation with $\Delta\mathsf{Diff}_1$ and $\Delta\mathsf{Diff}_2$ as the two inputs to the function f. If the outcome of f is 1, which means $\Delta\mathsf{Diff}_1 \geqslant \Delta\mathsf{Diff}_2$, we say that the direction degree is larger than $\arctan k$. Repeatedly executing the above steps for a series of pre-defined k's, a fine-grained direction degree range can be obtained. In our system, we select k to be eight values respectively, *i.e.* $\tan 10^\circ, \tan 20^\circ, \ldots, \tan 80^\circ$. Therefore, A 360-degree range of orientations can be fully covered with each span of the interval to be 10°.

2) Gradient Magnitude Computation: In our outsourcing model, we define our gradient magnitude as

$$\mathsf{m}(x, y) = |\mathsf{Diff}_x| + |\mathsf{Diff}_y|, \tag{3}$$

where $\mathsf{Diff}_x = \mathcal{L}(x + 1, y) - \mathcal{L}(x - 1, y), \mathsf{Diff}_y = \mathcal{L}(x, y + 1) - \mathcal{L}(x, y - 1)$. Accordingly, $\mathcal{S}_1$ and $\mathcal{S}_2$ compute gradient magnitudes $\mathsf{m}_1(x, y) = \mathsf{Diff}_{1x} + \mathsf{Diff}_{1y}$ and $\mathsf{m}_2(x, y) = \mathsf{Diff}_{2x} + \mathsf{Diff}_{2y}$, respectively. Then, they respectively accumulate m_1 and m_2 according to their orientations, after being weighted by the same Gaussian window, to build their own encrypted orientation histograms $\mathcal{H}_1$ and $\mathcal{H}_2$.

3) Finding Peaks in the Orientation Histogram: To detect peaks in the original orientation histogram $\mathcal{H}$, all we need is to find whose accumulated gradient magnitude is larger between two bins. Let $\sum_{10^\circ} \mathsf{m}_1$ and $\sum_{20^\circ} \mathsf{m}_1$ denote the accumulated gradient in $\mathcal{H}_1$ whose orientations lie in $0^\circ \sim 10^\circ$ and $10^\circ \sim 20^\circ$, respectively. Let $\sum_{10^\circ} \mathsf{m}_2$ and $\sum_{20^\circ} \mathsf{m}_2$

denote those computed in $\mathcal{H}_2$. To determine which of these two bins is higher in the orientation histogram $\mathcal{H}$, we can use the garbled circuit to compute the function $f(\sum_{10^\circ} \mathsf{m}_1 - \sum_{20^\circ} \mathsf{m}_1, \sum_{10^\circ} \mathsf{m}_2 - \sum_{20^\circ} \mathsf{m}_2)$. If the outcome is 1, we say the bin of $0^\circ \sim 10^\circ$ is higher; lower otherwise. After repeating this comparison between any two bins, we are able to find the highest peak and all the local peaks. To detect any local peak within 80% of the highest peak in $\mathcal{H}$, $\mathcal{S}_1$ and $\mathcal{S}_2$ will compute function $f(\sum_{LP} \mathsf{m}_1 - 0.8\sum_{HP} \mathsf{m}_1, \sum_{LP} \mathsf{m}_2 - 0.8\sum_{HP} \mathsf{m}_2)$ where $\sum_{LP} \mathsf{m}_i$ corresponds to the bin with local peak and $\sum_{HP} \mathsf{m}_i$ $(i \in 1, 2)$ corresponds to the bin with the highest peak. As a result, for locations with multiple peaks of similar magnitudes, there exists multiple keypoints created at the same location and scale, but with different orientations as the original SIFT does. *Note that*, in the above protocol, constant-round interactions can also be achieved by computing all the inputs simultaneously and transmitting them at one time.

D. Descriptor Generation Protocol

Once $\mathcal{S}_1$ and $\mathcal{S}_2$ have detected dominant orientations on a keypoint, they will first generate their respective descriptors as follows.

First, the coordinates and the gradient orientations are rotated relative to a keypoint orientation. Note that the specific value of the gradient orientation is needed in the process of rotation, so we propose to use the median of the interval its orientation lies in to replace the actual orientation value. For example, if the orientation of one point lies in the interval $10^\circ \sim 20^\circ$, we use 15° as its orientation to perform rotation. Our subsequent theoretical analysis and experiments show that this approximate orientation substitution has a negligible impact on the final results.

Second, in the 4×4 subregions around the keypoint, gradient magnitudes of sample points (as shown in Eq. (3)) are weighted by a Gaussian window and accumulated into 4×4 orientation histograms. Then, each histogram consists of 8 bins and the span of each bin is 45° so that a 360-degree range of orientations can be covered. To improve the system efficiency, all the gradients are pre-computed in the phase of orientation assignment.

Finally, a 128-dimensional feature vector is used to represent 16 histograms for a keypoint. Let $\mathcal{V}_1$ and $\mathcal{V}_2$ denote the feature vectors generated by $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively. They will be sent to $\mathcal{O}$, who can recover the actual feature vector by computing $\mathcal{V} = \mathcal{V}_1 - \mathcal{V}_2$, which will be normalized to unit length by $\mathcal{O}$ in order to achieve invariance to affine changes in illumination.

V. System Analysis

A. Effectiveness Analysis

In this subsection, we will provide a theoretical examination of the effectiveness of our scheme step by step in the sense that the extracted features are very close to the outputs generated by the original SIFT.

Keypoint Localization Protocol: In the keypoint localization, when comparing two neighbors to detect the extrema after building the difference-of-Gaussian space, we have

$$\begin{aligned}
&\Delta\mathcal{D}_1 - \Delta\mathcal{D}_2 \\
&= [\mathcal{D}_1(x, y, \sigma) - \mathcal{D}_1(x, y+1, \sigma)] - [\mathcal{D}_2(x, y, \sigma) - \mathcal{D}_2(x, y+1, \sigma)] \\
&= [\mathcal{D}_1(x, y, \sigma) - \mathcal{D}_2(x, y, \sigma)] - [\mathcal{D}_1(x, y+1, \sigma) - \mathcal{D}_2(x, y+1, \sigma)] \\
&= \mathcal{D}(x, y, \sigma) - \mathcal{D}(x, y+1, \sigma),
\end{aligned}$$

which means that the comparison of ΔD_1 and ΔD_2 is equivalent to determining which is larger between $D(x, y, \sigma)$ and $D(x, y+1, \sigma)$. Thus, all keypoints can be correctly localized by the cloud servers.

Orientation Assignment Protocol: In the computation of orientations, we have

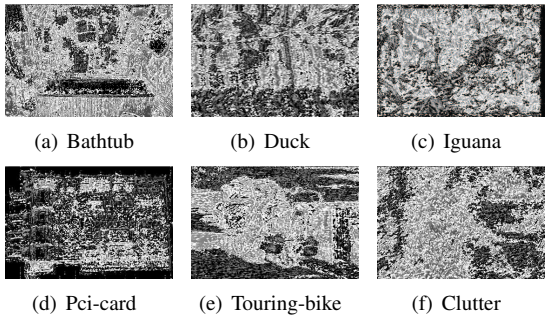
$$\begin{aligned}
&\Delta\mathsf{Diff}_1 - \Delta\mathsf{Diff}_2 \\
&= [\mathsf{Diff}_{1y} - k\mathsf{Diff}_{1x}] - [\mathsf{Diff}_{2y} - k\mathsf{Diff}_{2x}] \\
&= [\alpha(\mathcal{L}_1(x, y+1) - \mathcal{L}_1(x, y-1)) - k\beta(\mathcal{L}_1(x+1, y) - \mathcal{L}_1(x-1, y))] \\
&- [\alpha(\mathcal{L}_2(x, y+1) - \mathcal{L}_2(x, y-1)) - k\beta(\mathcal{L}_2(x+1, y) - \mathcal{L}_2(x-1, y))] \\
&= \alpha[(\mathcal{L}_1(x, y+1) - \mathcal{L}_2(x, y+1)) - (\mathcal{L}_1(x, y-1) - \mathcal{L}_2(x, y-1))] \\
&- k\beta[(\mathcal{L}_1(x+1, y) - \mathcal{L}_2(x+1, y)) - (\mathcal{L}_1(x-1, y) - \mathcal{L}_2(x-1, y))] \\
&= \alpha[\mathcal{L}(x, y+1) - \mathcal{L}(x, y-1)] - k\beta[\mathcal{L}(x+1, y) - \mathcal{L}(x-1, y)] \\
&= |\mathsf{Diff}_y| - k|\mathsf{Diff}_x|
\end{aligned} \tag{4}$$

In the last but one line of Eq. (4), we use the property $\mathcal{L}(x, y) = \mathcal{G} * \mathcal{I}(x, y) = \mathcal{G} * (\mathcal{I}_1(x, y) - \mathcal{I}_2(x, y)) = \mathcal{L}_1(x, y) - \mathcal{L}_2(x, y)$. When $\Delta\mathsf{Diff}_1 \geqslant \Delta\mathsf{Diff}_2$, we will have $|\mathsf{Diff}_y| \geqslant k|\mathsf{Diff}_x|$, which leads to $\frac{|\mathsf{Diff}_y|}{|\mathsf{Diff}_x|} \geqslant k$. Therefore, we can claim that the direction degree of this keypoint is larger than $\arctan k$. As discussed in Section IV-C, by choosing a series of k's, the orientation of each keypoint can be located in a interval whose span is 10°. So it is enough to construct an orientation histogram with 36 bins covering the 360-degree range as the original SIFT does, though their specific values of orientation still remain unknown.

In the process of finding peaks, our goal is to determine which is larger between two bins in the original histogram $\mathcal{H}$. For ease of analysis, we assume $\sum_{10^\circ} \mathsf{m}_1 = w_1\mathsf{m}_1(x, y)$ and $\sum_{20^\circ} \mathsf{m}_1 = w_2\mathsf{m}_1(x', y')$, where w_1 and w_2 denote the corresponding Gaussian weight values. It means that the bins of $0^\circ \sim 10^\circ$ and $10^\circ \sim 20^\circ$ in the histogram built by $\mathcal{S}_1$ consist of only one point. Accordingly, we can get $\sum_{10^\circ} \mathsf{m}_2 = w_1\mathsf{m}_2(x, y)$ and $\sum_{20^\circ} \mathsf{m}_2 = w_2\mathsf{m}_2(x', y')$ computed on $\mathcal{S}_2$. Then, when computing $f(\sum_{10^\circ} \mathsf{m}_1 - \sum_{20^\circ} \mathsf{m}_1, \sum_{10^\circ} \mathsf{m}_2 - \sum_{20^\circ} \mathsf{m}_2)$, we have

$$\begin{aligned}
&(\sum_{10^\circ} \mathsf{m}_1 - \sum_{20^\circ} \mathsf{m}_1) - (\sum_{10^\circ} \mathsf{m}_2 - \sum_{20^\circ} \mathsf{m}_2) \\
=&w_1[\mathsf{m}_1(x, y) - \mathsf{m}_2(x, y)] - w_2[\mathsf{m}_1(x', y') - \mathsf{m}_2(x', y')] \\
=&w_1[(\mathsf{Diff}_{1x} + \mathsf{Diff}_{1y}) - (\mathsf{Diff}_{2x} + \mathsf{Diff}_{2y})] \\
&- w_2[(\mathsf{Diff}_{1x'} + \mathsf{Diff}_{1y'}) - (\mathsf{Diff}_{2x'} + \mathsf{Diff}_{2y'})] \\
=&w_1(|\mathsf{Diff}_x| + |\mathsf{Diff}_y|) - w_2(|\mathsf{Diff}'_x| + |\mathsf{Diff}'_y|) \\
=&w_1\mathsf{m}(x, y) - w_2\mathsf{m}(x', y') \\
=&\sum_{10^\circ} \mathsf{m} - \sum_{20^\circ} \mathsf{m},
\end{aligned} \tag{5}$$

which explains that this comparison function f is able to find out which bin has the largest accumulated gradient magnitude in the histogram $\mathcal{H}$. Similarly, local peaks within 80% of the highest peak can also be detected. To sum up, the main



(a) Bathtub (b) Duck (c) Iguana

(d) Pci-card (e) Touring-bike (f) Clutter

Fig. 3: Recovered images by the cloud server.

orientations can be correctly computed by the *Orientation Assignment Protocol*.

Descriptor Generation Protocol: First, we will show that the approximate orientation substitution in the image rotation has a very low error probability, which can be ignored. In the orientation assignment, each direction is located in an interval, and we use the median to represent its value. If one point, for example, lies in the interval $0^\circ \sim 10^\circ$, we will assume its direction is 5°. Since the rotation degree are $0^\circ, 10^\circ, 20^\circ, \ldots, 350^\circ$, and it needs to determine which new interval (*i.e.* , $0^\circ \sim 45^\circ, 45^\circ \sim 90^\circ, \ldots, 315^\circ \sim 360^\circ$) the rotated direction lies in, the error that the rotated direction is distributed into a wrong interval will happen with probability $\frac{1}{2}$ only when the rotated interval contains $45^\circ, 135^\circ, 225^\circ$ and 315°, such as $40^\circ \sim 50^\circ, 130^\circ \sim 140^\circ, 220^\circ \sim 230^\circ$ and $310^\circ \sim 320^\circ$. Hence, the total error probability is $\frac{4}{36} \times \frac{1}{2} = \frac{1}{18} \approx 6\%$. In Section VI, our experimental results will further demonstrate that this probability is low enough to be ignored in practice.

Finally, in the descriptor recovery process, for the entry corresponding to the bin of $0^\circ \sim 10^\circ$ in the descriptor, the data owner $\mathcal{O}$ has

$$\sum_{10^\circ} \mathsf{m}_1 - \sum_{10^\circ} \mathsf{m}_2 = w_1 \mathsf{m}_1(x,y) - w_1 \mathsf{m}_2(x,y)$$
$$= w_1 \mathsf{m}(x,y) = \sum_{10^\circ} \mathsf{m}.$$

Note that the above equation holds by Eq. (5), and it has the same property for all the other entries in the descriptor. This means that each entry $\mathcal{O}$ derives by computing $\mathcal{V} = \mathcal{V}_1 - \mathcal{V}_2$ is the accumulated gradient magnitude in that orientation of the original histogram. Therefore, $\mathcal{O}$ is able to recover the real descriptor correctly in this way.

B. Security Analysis

1) Confidentiality of Pixel Values: In the *Image Encryption* step, each image $\mathcal{I}$ represented by a matrix is disguised by adding a random matrix. After splitting, each cloud server receives a random matrix. If the image has size of 300×300 and each pixel has 8 bits, the server has to use brute-force approach which needs $256^{300\times300}$ operations to recover the image, and it is computationally-infeasible in practice. As for the *Keypoint Localization Protocol*, a secure comparison is implemented by using garble circuit which is provably secure under our semi-honest model [12], so both servers cannot learn anything about each other's input but the comparison result. Since the servers only know large or small of pixel values between any two neighbors, they are unable to compute the specific value of each pixel at all. Similarly, in the *Orientation Assignment Protocol*, the servers can obtain nothing but some more complex but unuseful relationships between pixels in a small region around keypoints. For example, they may get $\mathcal{L}(x, y+1) \geqslant \mathcal{L}(x, y-1)$ and $|\mathsf{Diff}_y| \geqslant k|\mathsf{Diff}_x|$ in orientation computation, and $\sum_{10^\circ} \mathsf{m} \geqslant \sum_{20^\circ} \mathsf{m}$ in finding peaks. These available inequalities are still insufficient to compute the exact value of $\mathcal{L}(x, y)$. Therefore, we claim that the confidential of pixel values are well protected against the servers.

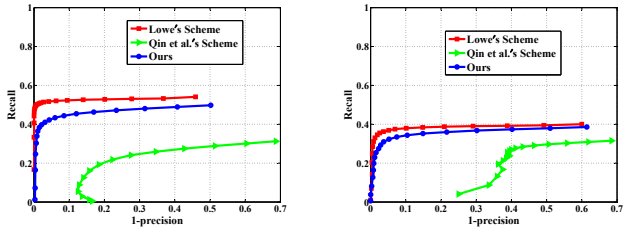
2) Privacy of Image Content: We next show that the servers are unable to deduce the original image content by what they have obtained. As illustrated above, the servers can only get some inequalities about the pixel values, so they have to traverse all the possibilities for each pixel that satisfy these inequalities. We argue that in order to reconstruct the whole image, the most useful information is the knowledge of the relationship between any two neighboring pixels while other inequality constraints have very limited contribution in recovering pixels. Fig. 3 shows the cloud server's possible reconstructions of six images that are randomly chosen from six categories of Caltech256 Dataset [15]. Obviously, it is very difficult to recognize the original content in the recovered images. Formally, we show that there exist exponential number of possible reconstructions that satisfy the given constraints between any two neighboring pixels. For an image of size $n \times n$ with each pixel of 8-bit length, as shown in Fig. 5 (b) with $n = 8$, the values of pixels in each square have $\binom{256}{4}$ different possibilities. The rest pixels are supposed to have only one possibility so that all the neighbors can meet the given constraints. Since there exist at least $(\frac{n}{4})^2$ squares in the image, the total number of different possibilities is at least $(\binom{256}{4})^{(\frac{n}{4})^2}$. Hence, there has an exponential number of possible reconstructions of the original image, which is impractical for the servers to launch an exhaustive search.

VI. Experimental Results

In this section, we conduct various experiments to evaluate the effectiveness and the efficiency of our secure SIFT outsourcing scheme using representative real-world image datasets. For each cloud service provider $\mathcal{S}$, we utilize up to 1000 instances each with Intel Core 3.1GHz CPU and 12GB memory. We set up the data owner $\mathcal{O}$ on a local machine with the same hardware configuration.

A. Effectiveness Evaluation

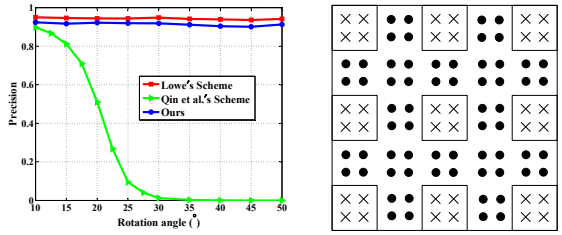
In the following experiments, since Qin et al.'s scheme [9] and Hsu et al.'s scheme [7] always generate the same feature vectors, we will only compare our scheme with Qin et al.'s and also the original SIFT proposed by Lowe [10]. To have a better evaluation on the distinctiveness and robustness of feature vectors, we choose to measure the performance on keypoint matching experiment as did in [16]. That is, given



(a) Matching experiments where target images are rotated by 20° and scaled by 20%.

(b) Matching experiments where target images have a 3D viewpoint change of 20°.

Fig. 4: *Recall* versus 1-*precision*.



(a) *Precision* versus rotation angle.

(b) Structure analysis for image recovery.

Fig. 5: Precision and structure analysis.

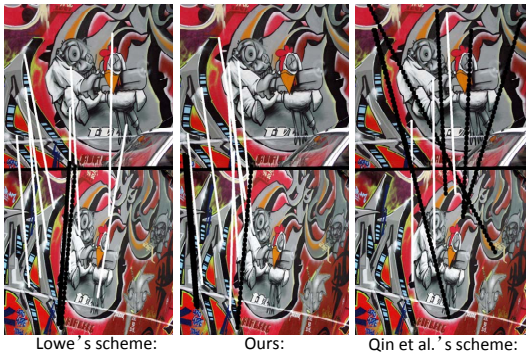


Fig. 6: A performance comparison of the visualized matching results. Here, real-world images are selected from Graffiti dataset taken from different viewpoints. The top ten matches are shown for each algorithm: solid white lines represent correct matches while dotted black lines represent false ones.

an interest point, finding all matches of that interest point in a dataset. More specifically, we will compute Euclidean distances between feature vectors to find the nearest point and the second nearest one in the dataset. A *match* between the interest point and its nearest point occurs if the distance ratio between these two points is below a threshold t. We conduct experiments on real image dataset: INRIA Graffiti [17], which contains images of graffiti-covered walls taken from different camera viewpoints, and we use well-acknowledged evaluation metrics *recall* and 1-*precision* defined in [18] to quantify our results: $recall = \frac{\text{number of } \textit{correct-matches}}{\text{total number of positives}}$ and 1-$precision = \frac{\text{number of } \textit{false-matches}}{\text{total number of matches}}$, where a *correct-match* is a match where the two keypoints correspond to the same physical location while a *false-match* is a match where the two keypoints come from different physical locations, and the *total number of positives* is known as a *priori* for a given dataset.

In Fig. 4, we plot *recall* VS. 1-*precision* for the image matching experiment by varying the distance ratio threshold t ($0 \leqslant t \leqslant 1$). In Fig. 4 (a), the target images are rotated by 20° and scaled by 20%. It is shown that when t is rather small, our scheme and Lowe's [10] both perform better in finding the correct match, i.e., achieving a high *recall* that is getting close to the maximum 0.5 and a low 1-*precision* that is much smaller than 0.1, while the recall of Qin et al.'s scheme [9] is low (i.e., smaller than 0.2) and its 1-*precision* is high (i.e., larger than 0.1). Note that when t approaches to 1, the 1-*precision* in our scheme and Lowe's remains around 0.5 but that of Qin et al.'s scheme has quickly increased to 0.7. The results indicate that Qin et al.'s scheme will generate much more false matches than our scheme and Lowe's under the same threshold. As shown in Fig. 4 (b), when the target images have a 3D viewpoint change of 20°, it is shown that our scheme and Lowe's almost have the same performance and both of them outperform Qin et al.'s scheme.

To further demonstrate the rotation-invariant property, Fig. 5 (a) plots the relationship between *precision* and rotation angle. Still, the *precision* in Lowe's scheme [10] and that of our scheme both maintain a high level when the target images are rotated from 10° to 50°, while the *precision* of Qin et al.'s scheme [9] decreases sharply to 0 when the targets are only rotated less than 30°. In Fig. 6, we provide a comparison of three schemes' matching results visualized for illustration purpose. For a better illustration, we manually set the thresholds to have each scheme return 10 matches. It clearly shows that our scheme approaches Lowe's original SIFT [10] and outperforms Qin et al.'s scheme [9] in finding correct matches. Fig. 7 illustrates the top-k results when applying the three algorithm to image matching. We randomly select 11 categories from Oxford database [17], each of which contains 10 images and shares one common object, as the dataset. Still, ours and Lowe's original SIFT [10] have almost the same performance while Qin et al.'s scheme [9] may generate some mismatches.

B. Efficiency Evaluation

In Fig. 8, we evaluate the computation efficiency of privacy-preserving SIFT outsourcing with the increase of image size. Fig. 8 (a) shows the time cost of performing SIFT algorithm locally and that of outsourcing of SIFT scheme on the data owner side. As can be seen, our scheme has an extreme low time cost on the data owner, which is nearly a constant value. On the contrary, without outsourcing the time cost on the data owner increases linearly with the image size. In addition, Fig. 8 (b) depicts the computation burden on the cloud server. It is shown that the computation cost of Qin et al.'s scheme [9] is much lower than that of our scheme. However, we **emphasize** that our scheme preserves more key characteristics of the original SIFT [10] and requires less number of independent cloud servers, which are the key to the successful application

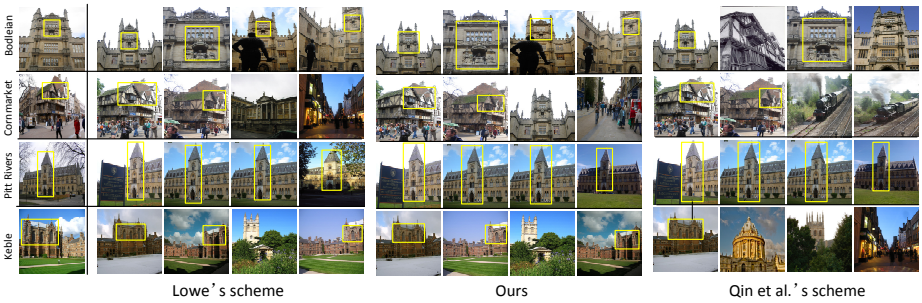
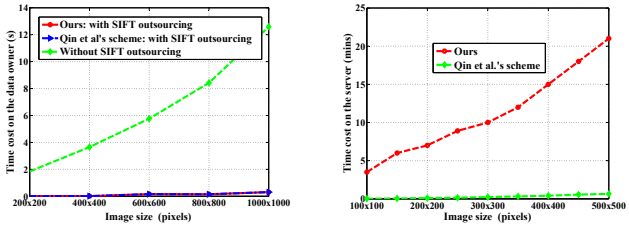


Fig. 7: A comparison of the three schemes for the image matching application. The leftmost column consists of four query images while the right column images are the corresponding top-k (k=4) matching results of each scheme. Note that in each image the yellow rectangle depicts the matched object.



(a) A comparison of time costs of the SIFT algorithm on the data owner with and without outsourcing.

(b) A comparison of time costs on the cloud server.

Fig. 8: Computation efficiency evaluation.

of privacy-preserving SIFT outsourcing in practice. From the perspective of utilizing cloud, we argue that the computation cost on the cloud in our scheme remains to be practical, and it can be significantly reduced by adopt more cloud computing instances to run the encryption steps in parallel.

Communication efficiency. As discussed before, our scheme achieves constant-round interactions. Now we analyze the specific communication cost. The communication cost of our scheme mainly lies in transmitting the garbled circuits and executing *OT* protocol. Note that garbled circuit has been shown for its efficiency to securely evaluate simple functions, such as comparison. By adopting the fast garbled circuit approach [13], we can implement our privacy-preserving SIFT outsourcing protocol in a more efficient way. Take a typical image with 300×300 8-bit pixels as an example, our experimental results show that it will cost approximately 2403693 comparison functions f over all rounds of interactions (including the running of *OT* protocols). Each comparison function with two 8-bit inputs requires one gate in the circuit, with each gate requiring 32 bytes and costing 11 bytes in executing the *OT* protocol [13]. So, the total communication cost is $2403693 \times (32+11) \approx 98$MB. The time that takes to transmit 98MB of message is significantly less than the computation time. Therefore, this communication cost is practical for message transmission between the two cloud servers.

VII. Conclusion

In this work, we presented a new and novel privacy-preserving SIFT outsourcing protocol. We both carefully analyze and extensively evaluate the security and effectiveness of our design. Our experimental results show that our protocol outperforms the state-of-the-art and performs comparably to the original SIFT and is practical for real-world applications.

References

[1] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," *IEEE Internet Computing*, no. 1, pp. 69–73, 2012.
[2] S. Hohenberger and A. Lysyanskaya, "How to securely outsource cryptographic computations," in *Proc. of TCC'05*. Springer, 2005, pp. 264–282.
[3] M. J. Atallah and K. B. Frikken, "Securely outsourcing linear algebra computations," in *Proc. of AsiaCCS'10*. ACM, 2010, pp. 48–59.
[4] F. Hahn and F. Kerschbaum, "Searchable encryption with secure and efficient updates," in *Proc. of ACM CCS'14*, 2014, pp. 310–320.
[5] L. Weng, L. Amsaleg, A. Morton, and S. Marchand-Maillet, "A privacy-preserving framework for large-scale content-based information retrieval," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 1, pp. 152–167, 2015.
[6] M. Osadchy, B. Pinkas, A. Jarrous, and B. Moskovich, "Scifi-a system for secure face identification," in *Proc. of S&P'10*. IEEE, 2010, pp. 239–254.
[7] C.-Y. Hsu, C.-S. Lu, and S.-C. Pei, "Image feature extraction in encrypted domain with privacy-preserving sift," *IEEE Transactions on Image Processing*, vol. 21, no. 11, pp. 4593–4607, 2012.
[8] S. Wang, M. Nassar, M. Atallah, and Q. Malluhi, "Secure and private outsourcing of shape-based feature extraction," in *Proc. of ICICS'13*. Springer, 2013, pp. 90–99.
[9] Z. Qin, J. Yan, K. Ren, C. W. Chen, and C. Wang, "Towards efficient privacy-preserving image feature extraction in cloud computing," in *Proc. of ACM MM'14*. ACM, 2014, pp. 497–506.
[10] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
[11] A. Yao, "How to generate and exchange secrets," in *Proc. of FOCS'86*. IEEE, 1986, pp. 162–167.
[12] Y. Lindell and B. Pinkas, "A proof of security of yaos protocol for two-party computation," *Journal of Cryptology*, vol. 22, no. 2, pp. 161–188, 2009.
[13] Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster secure two-party computation using garbled circuits," in *Proc. of USENIX Security'11*, vol. 201, no. 1, 2011.
[14] P. Paillier and D. Pointcheval, "Efficient public-key cryptosystems provably secure against active adversaries," in *Proc. of ASIACRYPT'99*, 1999, pp. 165–179.
[15] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," 2007.
[16] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, 2005.
[17] http://www.robots.ox.ac.uk/~vgg/.
[18] Y. Ke and R. Sukthankar, "Pca-sift: A more distinctive representation for local image descriptors," in *Proc. of CVPR'04*, vol. 2. IEEE, 2004, pp. II–506.