Securing SIFT: Privacy-preserving Outsourcing Computation of Feature Extractions over Encrypted Image Data

Shengshan Hu, Qian Wang, Member, IEEE, Jingjun Wang, Zhan Qin, and Kui Ren, Fellow, IEEE

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 cloud by leveraging its abundant resources for cost saving and flexibility. Despite the tremendous benefits, 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 and practical 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 then present a new scheme design that achieves efficiency and security requirements simultaneously with the preservation of its key characteristics, by randomly splitting the original image data, designing two novel efficient protocols for secure multiplication and comparison, and carefully distributing the feature extraction computations onto two independent cloud servers. 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, addition of noise and change in 3D viewpoint and illumination.

Index Terms—Scale-invariant feature transform, privacy-preserving, security, homomorphic encryption, cloud computing.

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 [2], [3], [4], [5]. 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 services.

In the existing literature, efforts on privacy-preserving outsourcing computation have been devoted to various mathematical problems including modular exponentiation [6], linear equations [7] and kNN search [8]. These works mainly focus on engineering computation problems over numerical data or text data. Only in recent years, privacy-preserving data search in the ciphertext domain has been extended to content-based multimedia retrieval [9], face recognition [10] and fingerprint identification [11]. The authors in [12], [13] explored how to enable secure image search in the data outsourcing environment. Nevertheless, they all assume that the images have been pre-processed by some feature extraction algorithms to obtain their vector representations. Due to the importance of image feature extraction in multimedia data processing and its heavy operations on massive data, especially for satellite data for its tremendous size and large number of feature points, the extraction or detection of image features from the ciphertext domain has begun to attract more and more research interest.

To the best of our knowledge, Hsu et al. [14] was the first to investigate privacy-preserving SIFT in the encrypted

[1] A preliminary version [1] of this work has been accepted by the IEEE International Conference on Computer Communications (INFOCOM’16).

S. Hu, Q. Wang and J. Wang are are with The State Key Lab of Software Engineering, School of Computer Science, Wuhan University, Wuhan 430072, China. E-mail: {shengshan, qianwang, jingjun}@whu.edu.cn. Q. Wang is the corresponding author.

Z. Qin and K. Ren are with the Department of Computer Science and Engineering, The State University of New York at Buffalo, NY 14260, USA. {zhanqin, kui.ren}@buffalo.edu

A preliminary version [1] of this work has been accepted by the IEEE International Conference on Computer Communications (INFOCOM’16).
domain by utilizing homomorphic encryption Paillier [15]. However, their solution is either computationally intractable, or otherwise insecure from the privacy perspective [16]. As a following work, Qin et al. [17] presented an improved scheme with the aid of order preserving encryption [18] and random permutation. In [19], 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 [20], respectively. While putting great effort on the privacy or efficiency aspect, one common limitation of the previous solutions is that they all lack comprehensive analysis and evaluations with respect to the preservation of the key characteristics of the original image feature extraction algorithm. Recently, we proposed a new scheme [1] to address this problem with the aid of garbled circuits [20]. Although this solution has a good preservation of key characteristics of SIFT, there is still room to further reduce its computation and communication costs. More importantly, it is still unable to eliminate edge responses such that the detected keypoints are unstable to small amount noise.

Motivated by the above observations, in this paper, we propose an effective and practical privacy-preserving computation outsourcing protocol for the prevailing scale-invariant feature transform (SIFT) over massive encrypted image data. In our design, we first additively and randomly split the image into two shares and distribute them to two semi-honest non-colluding cloud servers. Then we design two secure interactive protocols–batched secure multiplication protocol (BSMP) and batched secure comparison protocol (BSCP) based on somewhat homomorphic encryption (SHE) integrated with the latest batching technique single-instruction multi-data (SIMD). The former one allows the two servers to securely compute the products of multiple pairs of their private inputs simultaneously, while the latter one enables them to compare multiple pairs of private inputs at one time with privacy preservation. Based on the two interactive protocols, we then further develop a new approach to let the two servers collaboratively detect real locations of stable keypoints via the difference-of-Gaussian (DoG) scale space built on encrypted image, and main orientations through computing the orientation range instead of their specific values from the encrypted versions of orientation histogram. 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.

- We design two novel secure interactive protocols BSMP and BSCP that enable the two servers to compute the products and make comparisons of multiple pairs of integers simultaneously with privacy preservation by using somewhat homomorphic encryption (SHE) and the batching technique SIMD.
- 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 based on BSMP and BSCP.
- We carefully analyze our protocol to show that 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.
- We evaluate our protocol comprehensively with real-world massive image datasets. 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, change in illumination, addition of noise, 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 $O$, the cloud server $S_1$, the cloud server $S_2$ (as illustrated in Fig. 1). In this application scenario, we assume that the data owner $O$ 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, $O$ 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 $S_1$ and $S_2$ are independent with each other and can be considered to belong to two independent cloud service providers like Amazon EC2 and Microsoft Azure. To protect the privacy, $O$ will first encrypt each image set and then distribute the ciphertexts to $S_1$ and $S_2$. After running SIFT algorithm in the ciphertext domain via a sequence of secure interaction protocols, $S_1$ and $S_2$ 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, $S_1$ and $S_2$ are assumed to be two independent or say non-colluding servers. Note that, as shown
in [17], 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 [17] while our protocol is implemented with only two ones, which is easier to deploy in practice. We emphasize that this non-colluding assumption can be promised in practice because the cloud service providers have to maintain their reputation and take their own financial interests into account.

Following all existing privacy-preserving data/computation outsourcing protocols [6], [14], [19], [17], we also assume the cloud is semi-trusted (which can be seen as the adversary in our application scenario), i.e., $S_1$ and $S_2$ 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) revealed relationships between pixels (e.g., keypoint locations) and/or ii) encrypted feature descriptors.

### III. Background

Before elaborating our solutions, we will first give a brief overview of the Scale-invariant feature transform (SIFT) algorithm to make the paper self-contained. Then we will discuss the cryptographic tools used as building blocks to construct our privacy-preserving protocol for SIFT outsourcing.

#### A. 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 [21]. According to SIFT [21], image features are extracted by the following three stages:

**Keypoint localization.** For an initial image $I$, a difference-of-Gaussian scale space is built by convolving $I$ with a difference-of-Gaussian (DoG) function as

$$D(x, y, σ) = (G(x, y, kσ) - G(x, y, σ)) ∗ I(x, y) = L(x, y, kσ) - L(x, y, σ), \tag{1}$$

where $L(x, y, σ) = G(x, y, σ) ∗ I(x, y)$ is the smoothed image and $∗$ is the convolution operation in $x$ and $y$. In order to detect the local extrema of $D(x, y, σ)$, 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. Then each candidate keypoint will be filtered to eliminate edge responses such that the final detected keypoints are stable.

**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 $L(x, y)$ that a keypoint lies in, if we denote $\text{Diff}_x = L(x + 1, y) - L(x - 1, y)$, $\text{Diff}_y = L(x, y + 1) - L(x, y - 1)$, the gradient magnitude $m(x, y)$ and orientation $θ$ are computed as $m(x, y) = \sqrt{\text{Diff}_x^2 + \text{Diff}_y^2}$ and $\theta(x, y) = \tan^{-1} \frac{\text{Diff}_y}{\text{Diff}_x}$.

Then, an orientation histogram $H$ 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 $4 \times 4$ subregions. Finally, a 128-dimensional feature descriptor is formed by using a vector to represent its 16 histograms.

#### B. Cryptographic Tools

Homomorphic encryption is a public-key encryption scheme that supports meaningful operations over the encrypted data. In particular, somewhat homomorphic encryption (SHE) is such a scheme that enables a limited number of both addition and multiplication operations on the ciphertexts. For our privacy-preserving outsourcing computation system, we propose to use a ring-LWE-based homomorphic cryptosystem [22]. This scheme is parametrized by the ring $R_q = \mathbb{Z}_q[x]/(x^n + 1)$, where $n$ is a power of two, an odd prime number $q$, and an error parameter $σ$ that defines a discrete Gaussian error distribution $χ = D_{\mathbb{Z}_q, σ}$ with standard deviation $σ$. The message space of the scheme is defined by a prime $t$ as $R_t = \mathbb{Z}_t[x]/(x^n + 1)$. We use the tuple $(\text{SH.Gen}, \text{SH.Enc}, \text{SH.Dec})$ to represent the key generation, the encryption and the decryption procedures in a SHE scheme, respectively.

**SH.Gen:** Sample a ring element $s ← χ$ and define the secret key $sk = s$. Sample a uniformly random ring element $a_1 ← R_q$ and an error $e ← χ$ and compute the public key $pk = (a_0 = -(a_1 s + te), a_1)$.

**SH.Enc($m$, $pk$):** Given $pk = (a_0, a_1)$ and a message $m ∈ R_t$, the encryption algorithm samples $u ← χ$, and $f, g ← χ$, and compute the ciphertext $ct = (c_0, c_1) = (a_0 u + tg + m, a_1 u + tf)$.

**SH.Dec($ct$, $sk$):** To decrypt $ct = (c_0, c_1, \ldots, c_5)$, compute $\hat{m} = \sum_{i=0}^{5} c_i s_i$. Output the message as $\hat{m} \mod t$.

The reader may refer to [22], [23] for more detailed parameter analysis and cryptosystem construction.

One of the most appealing advantages of SHE is that the batching technique: single-instruction multiple-data (SIMD) of [24] can be used to reduce the communication cost and speed up the computation on ciphertexts when solving problems on a large scale [25]. SIMD enables one to pack $l$ different plaintext elements (where $l$ is typically in the...
range of $500 \sim 10000$ in a single ciphertext and thus the same operation on these elements (also called plaintext slots) can be computed in parallel. Formally, following [24] we let $I_l$ denote the SIMD-type isomorphism mapping that can pack $l$ integers into a single one, and let $I_{l^{-1}}$ denote the inverse mapping correspondingly. We also use $\oplus$ and $\otimes$ to denote homomorphic addition and multiplication on the ciphertexts respectively. Then for two $l$-dimensional integer vectors $x = (x_1, x_2, \ldots, x_l)$ and $y = (y_1, y_2, \ldots, y_l)$, the SIMD-type homomorphic operations can be expressed as

$$SH.Enc(I_l(x_1 + y_1, x_2 + y_2, \ldots, x_n + y_n)) = SH.Enc(I_l(x_1, x_2, \ldots, x_l)) \oplus SH.Enc(I_l(y_1, y_2, \ldots, y_l)) = \text{SH.Enc}(I_l(x_1, x_2, \ldots, x_l)) \otimes \text{SH.Enc}(I_l(y_1, y_2, \ldots, y_l)).$$

It clearly shows that a single encryption is sufficient to encode the vector, i.e., a single homomorphic operation on the ciphertexts applies the homomorphic operation componentwise on the entire vector. Readers may refer to [25] for detailed constructions and security proof of the SHE scheme and the SIMD mapping.

IV. 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 [14]: Hsu et al. [14] 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 [15], 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 limitations that make it inapplicable to real-world applications as first pointed out by [16].

First, it introduces a very large computation complexity on the encrypted data comparison. In the comparison protocol of [14], 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 $2^{30000}$ elements, and thus it will require $2^{20000} / (2 \times 10) > 2^{1995}$ comparison operations. Apparently, it puts an intolerable computation effort on the cloud. 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 0.07 seconds for a 1024-bit modulus of the Paillier cryptosystem. For an image of size $300 \times 300$ pixels, the total encryption time will be about 105 minutes. So, it also puts huge intolerable workload on the data owner, who 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, then 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 ciphertexts 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 of size $n \times n$, it 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 [17]: As a following work, Qin et al. [17] 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 [17], 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., $P = |d|/|r|$, where $|d|$ denotes number of real keypoints and $|r|$ 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 $400 \times 400$ 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 $2000 / 160000 \approx 1.2\%$. 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 III, 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 [14] and [17], however, the
V. PRIVACY-PRESERVING OUTSOURCING OF SIFT: SYSTEM OVERVIEW

In this work, our ultimate goal is to enable privacy-preserving SIFT outsourcing computation while preserving all its key characteristics as much as possible. To this end, we have the following specific design goals.

Security Constraint: The privacy of the original image content should be protected as much as possible from the adversary.

Effectiveness Constraint: The extracted feature vectors performed by outsourced SIFT algorithm should be close to the results from original SIFT algorithm as much as possible. Meanwhile, these feature vectors should preserve important properties embedded in the original SIFT algorithm.

Efficiency Constraint: The local computations done by the data owner should be substantially reduced compared with performing SIFT algorithm on his own. The computation burden on the cloud servers should be acceptable considering their ability.

To give a better demonstration, we will first provide a system overview of our proposed scheme as illustrated in Fig. 2(a) while presenting the detail constructions for each stage in section VII.

Image Encryption: The data owner $O$, who holds a private image $I$, encrypts it and sends the encrypted sub-images $I_1$ and $I_2$ to servers $S_1$ and $S_2$, respectively. Then, $S_1$ builds their own Gaussian scale spaces $L_i(x, y, \sigma)$ and difference-of-Gaussian spaces $D_i(x, y, \sigma)$ based on $I_i$ ($i = 1, 2$).

Keypoint Localization: $S_1$ and $S_2$ run a keypoint localization protocol based on BSCP to find the candidate keypoint locations without compromising the privacy of their own inputs. Then they will eliminate edge responses to obtain stable keypoints with the aid of BSMP. Finally, the locations are revealed to the two servers.

Orientation Assignment: $S_1$ and $S_2$ run an orientation assignment protocol based on BSCP. By using the orientations of each sample point around the keypoint (computed in Step 1), they will further build their orientation histograms $H_1$ and $H_2$, respectively. Then, the peaks are found in the original orientation histograms $H$ by using $H_1$ and $H_2$, and each peak corresponds to a main orientation (computed in Step 2). Finally, the image data will be rotated relative to the main orientations, and each rotation generates a feature descriptor for a keypoint.

Descriptor Generation: $S_1$ and $S_2$ build orientation histograms with the directions that have been computed in the above steps without interacting with each other. Then, they respectively transform the orientation histograms into encrypted feature descriptors $V_1$ and $V_2$ and send them to the data owner. This phase is called Descriptor Generation Protocol. Finally, the data owner can recover the real feature descriptors from $V_1$ and $V_2$.

In this work, our key idea is to employ two servers to simulate SIFT algorithm by using $I_1$ and $I_2$, such that the final results equal as much as possible to performing SIFT on the original image $I$, while the two servers learn nothing about it. Therefore, each interaction between $S_1$ and $S_2$ is particularly designed to ensure that the returned results well approximate the outputs generated by the original SIFT. This is explicitly shown in our effectiveness analysis in Section VIII-A.

VI. OUR PROPOSED BASIC SECURITY PRIMITIVES

Before elaborating our scheme, we first present two novel protocols that will be used as sub-routines in our privacy-preserving outsourcing scheme for SIFT. The two newly proposed protocols are designed based on SHE integrated with SIMD which are shown to outperform the existing methods in efficiency in terms of both computation and communication overhead. This will be explicitly illustrated in our experiments in section IX.

A. Batched Secure Multiplication Protocol

We propose a new protocol for secure multiplication between two parties, the goal of which is that each party obtains a random share of the product of their private inputs.
Specifically, for the two servers $S_1$ and $S_2$ who own private $k$-bit length integer $x_1$ and $y_1$ respectively, where $x_1, y_1 \in \mathbb{Z}_p$ and $p$ is a pre-defined parameter that determines the plaintext space, after executing the protocol, $S_1$ receives a uniformly distributed random value $s_1 \in \mathbb{Z}_p$ and $S_2$ receives a dependent uniformly distributed random value $r_1 \in \mathbb{Z}_p$ such that $s_1 + r_1 = x_1 \times y_1 \pmod{p}$. During this protocol, no information regarding $x_1$ and $y_1$ is revealed to the other party.

Different from the existing scheme proposed in [26] where the homomorphic cryptosystem Paillier [15] is adopted to encrypt private messages, our batched secure multiplication protocol (BSMP) is built on SHE as described above that enables the two parties to securely compute the products of multiple pairs of private integers simultaneously, with computation and communication costs greatly reduced. Namely, for two $N$-dimensional integer vectors $x = (x_1, x_2, \ldots, x_N)$ and $y = (y_1, y_2, \ldots, y_N)$ owned by $S_1$ and $S_2$ respectively, the outputs of BSMP will be $s = (s_1, s_2, \ldots, s_N)$ and $r = (r_1, r_2, \ldots, r_N)$ such that $s_i + r_i = x_i \times y_i \pmod{p}$ for $1 \leq i \leq N$. Since we only consider 8-bit length inputs and $p$ is unique and usually set to be more than 16bits in length, the bit length of intermediate results are much less than $p$. Thus we will drop this modulo $p$ operations in the following context for ease of exposition. At the beginning of BSMP, $S_1$ will first encrypt $x$ by using SIMD in a packing manner. Here for simplicity, we assume that $N$ is a multiple of $l$. If not, we can append enough 0’s to $x$ to achieve this goal. Then $S_1$ sends the encrypted version of $x$ to $S_2$, who will then multiply it with $y$ by utilizing the homomorphic properties. Finally, $S_2$ will blind the multiplication result with randomly-generated numbers and send it to $S_1$ for decryption. The explicit steps of the protocol BSMP are depicted in Alg. 1.

B. Batched Secure Comparison Protocol

Here we propose a new protocol for secure comparison that enables two servers to compare two integers without revealing the privacy of their own inputs. Compared with the very general secure comparison protocol garbled circuit [20], our subsequent experiments will show that our proposed scheme performs much better in efficiency.

The key idea of our comparison protocol is motivated by [27], where a secure comparison protocol is proposed based on the secure scalar product protocol (SPP) proposed in [26]. However, different from their design, we use SHE combined with SIMD to construct a batched secure comparison protocol (BSCP) instead of using SPP. As a result, we are able to compare multiple pairs of integers at one time with privacy preservation guarantee and substantial communication and computation overhead reduction.

Specifically, if we want to securely compare two $k$-bit length integers $x, y \in \mathbb{Z}_p$, whose binary representations are $x = x_k x_{k-1} \cdots x_1$ and $y = y_k y_{k-1} \cdots y_1$, respectively. Then, for $i = k, k - 1, \cdots, 1$ we compute $a_i$ and $b_i$ as follows:
- If $i = k$, then $a_i = x_i \times (1 - y_i)$, $b_i = y_i \times (1 - x_i)$;
- If $i < k$, then
  
  $a_i = (1 - b_i^{i+1}) \times (a_i^{i+1} + (1 - a_i^{i+1}) \times x_i \times (1 - y_i))$
  
  $b_i = (1 - a_i^{i+1}) \times (b_i^{i+1} + (1 - a_i^{i+1}) \times y_i \times (1 - x_i))$

If the final result $a_0 (b_0)$ equals 1, we claim that $x$ is greater (smaller) than $y$. Based on this result, $S_1$ will first conduct bitwise encryption of $x$ and send the ciphertexts to $S_2$. Here we also assume that the total number of integers $N$ is a multiple of the batched number $l$ for ease of representation as did in BSMP. Then $S_2$ will compute the above equations over ciphertexts based on the homomorphic properties. After $k$ iterations, $S_2$ obtains the final encrypted results, and sends them to $S_1$ for decryption to get the final results in plaintext form. The explicit steps are presented in Alg. 2.

VII. PRIVACY-PRESERVING OUTSOURCING OF SIFT: OUR CONSTRUCTION

In this section, we will describe each protocol explicitly shown in the Fig. 2(a). Each step is carefully designed to satisfy the three constraints mentioned above.

A. Image Encryption

For an original image $I$ with size of $n \times n$ pixels, we use the matrix $I(x, y)$ ($0 \leq x, y < n$) with each entry of 8-bit length. The data owner randomly selects $n^2$ integers from $[0, 255]$ to generate a random matrix $I_2(x, y)$ and then encrypts $I(x, y)$ by computing $I_1(x, y)$ as: $I_1(x, y) = I(x, y) + I_2(x, y)$. The ciphertexts $I_1$ and $I_2$ are sent to $S_1$ and $S_2$, respectively. Note that each element of the matrix $I_1(x, y)$ will be represented with 9 bits due to the potential overflow.
The public binary vector

Require: \( S_1 \) has private \( N \)-dimensional integer vector \( x = (x_1, x_2, \ldots, x_N) \in \mathbb{Z}_p^N \), where \( x_j \)'s binary representation is \( x_j^k \cdot x_j^{k-1} \cdot x_j^1 \cdot x_j^0 \). \( S_2 \) has private \( N \)-dimensional integer vector \( y = (y_1, y_2, \ldots, y_N) \in \mathbb{Z}_p^N \), where \( y_j \)'s binary representation is \( y_j^k \cdot y_j^{k-1} \cdot \ldots \cdot y_j^1 \).

Ensure: The public binary vector \( z = (z_1, z_2, \ldots, z_N) \), where \( z_j = 1 \) iff \( x_j > y_j \).

At \( S_1 \):
1. Generate a private key and public key pair \( (pk, sk) \) and the number \( l \). Compute \( d = \frac{p}{l} \).

At \( S_1 \):
1. for \( i = k, k-1, \ldots, 1 \) do
   2. for \( j = 0, 1, \ldots, d-1 \) do
      3. \( c_j^i = \text{SH.Enc}(T(x_{ij+1}, x_{ij+2}, \ldots, x_{ij+1}), pk) \);
   end for
end for
2. Send ciphertexts \((c_0^d, \ldots, c_{d-1}^d, \ldots, c_0^1, \ldots, c_{d-1}^1)\) and parameters \( pk, d, l \) to the server \( S_2 \);

At \( S_2 \):
1. Randomly select \( d \times k \) integers from \( \mathbb{Z}_p \), that is \((R_0^0, \ldots, R_{k-1}^0, \ldots, R_0^{d-1}, \ldots, R_{k-1}^{d-1})\);
2. Compute \( T = \text{SH.Enc}(T(1, 1, \ldots, 1), pk) \) and \( T' = \text{SH.Enc}(T(-1, -1, \ldots, -1), pk) \);
3. for \( j = 0, 1, \ldots, d-1 \) do
   1. for \( i = k, k-1, \ldots, 1 \) do
      2. \( u_j^i = \text{SH.Enc}(T(y_{ij+1}, y_{ij+2}, \ldots, y_{ij+1}), pk) \);
   end if
end for
4. Send the binary vector \((a_0^d, a_1^d, \ldots, a_{d-1}^d)\) to the server \( S_2 \).

At \( S_1 \):
1. for \( j = 0, 1, \ldots, d-1 \) do
   2. \( (z_{ij+1}, z_{ij+2}, \ldots, z_{ij+l}) = \Gamma^{-1}_l(\text{SH.Dec}(a_j^l)) \);
end for
2. Send \((z_1, z_2, \ldots, z_N)\) back to \( S_2 \).

B. Keypoint Localization Protocol

1) Finding Candidate Keypoint: After receiving ciphertexts \( \mathcal{L}_i \) from \( \mathcal{O} \), \( S_1 \) creates Gaussian spaces \( \mathcal{L}_i(x, y, \sigma) \) by conducting convolution and downsampling operations on it with the variable-scale Gaussian \( \mathcal{G}(x, y, \sigma) \). Specifically, \( S_1 \) will compute

\[
L_1(x, y, \sigma) = \mathcal{G}(x, y, \sigma) \ast I_1(x, y),
\]

where \( \ast \) 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(a), 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, \( S_1 \) further computes the difference-of-Gaussian function \( D_1(x, y, \sigma) \) from the substraction of adjacent Gaussian images as

\[
D_1(x, y, \sigma) = L_1(x, y, r\sigma) - L_1(x, y, \sigma),
\]

where \( r \) is a constant multiplicative factor pre-defined by the data owner. In the meantime, \( S_2 \) will do the same operations for \( L_2(x, y, \sigma) \) and obtain \( D_2(x, y, \sigma) \). 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 \( D(x, y, \sigma) \). 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 \( D(x, y, \sigma) \) and \( D(x, y + 1, \sigma) \), \( S_1 \) and \( S_2 \) will compute \( \Delta D_1 = D_1(x, y, \sigma) - D_1(x, y + 1, \sigma) \) and \( \Delta D_2 = D_2(x, y, \sigma) - D_2(x, y + 1, \sigma) \), respectively. Then, they will work together to compute which is larger between \( \Delta D_1 \) and \( \Delta D_2 \) by using BSCP. If \( \Delta D_1 \geq \Delta D_2 \), we will claim that \( D(x, y, \sigma) \geq D(x, y + 1, \sigma) \), otherwise \( D(x, y, \sigma) < D(x, y + 1, \sigma) \).

2) Eliminating Edge Responses: After the candidate keypoints are located, edge response eliminating will be conducted to exclude the points that are not stable. For a detected candidate keypoint \( D_1(x, y, \sigma) \) on the server \( S_1 \), its derivatives in \( x \)-direction, \( y \)-direction and \( xy \)-direction are first computed as follows to constitute its Hessian matrix

\[
\begin{align*}
R_{1xx} &= D_1(x + 1, y, \sigma) + D_1(x - 1, y, \sigma) - 2D_1(x, y, \sigma) \\
R_{1yy} &= D_1(x, y + 1, \sigma) + D_1(x, y - 1, \sigma) - 2D_1(x, y, \sigma) \\
R_{1xy} &= 0.25(D_1(x + 1, y + 1, \sigma) - D_1(x + 1, y - 1, \sigma) - D_1(x - 1, y + 1, \sigma) + D_1(x - 1, y - 1, \sigma)).
\end{align*}
\]

Let \( R_{2xx}, R_{2yy} \) and \( R_{2xy} \) denote that computed on the server \( S_2 \). By using BSCP with \( R_{1xx} \) and \( R_{2yy} \) as the two private inputs, \( S_1 \) is able to get a share of the product of \( R_{1xx} \) and \( R_{2yy} \) denoted by \( M_1(R_{1xx}R_{2yy}) \), and \( S_2 \) is able to get the other share denoted by \( M_2(R_{1xx}R_{2yy}) \) such that \( M_1(R_{1xx}R_{2yy}) + M_2(R_{1xx}R_{2yy}) = R_{1xx}R_{2yy} \). Similarly, \( S_1 \) can get the shares \( M_1(R_{1yy}R_{2xx}) \) and \( M_1(R_{1xy}R_{2xx}) \), and \( S_2 \) can get the shares \( M_2(R_{1yy}R_{2xx}) \).
and $\mathcal{M}_2(R_{1xy}R_{2xy})$. Then $S_1$ will compute the trace and the determinant of the Hessian matrix as

$$
\text{Tr}_1 = R_{1xx} + R_{1yy} \\
\text{Det}_1 = R_{1xx}R_{1yy} - R_{1xy}^2 - (\mathcal{M}_1(R_{1xx}R_{2yy}) + \mathcal{M}_1(R_{1yy}R_{2xx}) - 2\mathcal{M}_1(R_{1xy}R_{2xy})).
$$

Correspondingly, $S_2$ computes the trace $\text{Tr}_2$ and the determinant $\text{Det}_2$. Finally, to determine whether the point $D_1(x, y, \sigma)$ is a stable one, $S_1$ and $S_2$ will first compare $\text{Det}_1$ and $-\text{Det}_2$. If $\text{Det}_1 \leq \text{Det}_2$, then the point should be rejected immediately, otherwise they will further compute $Z_1 = r\text{Tr}_1^2 - 2r\mathcal{M}_1(\text{Tr}_1\text{Tr}_2) - (r + 1)^2\text{Det}_1$ and $Z_2 = -(r\text{Tr}_2^2 - 2r\mathcal{M}_2(\text{Tr}_1\text{Tr}_2) - (r + 1)^2\text{Det}_2)$ respectively, where $r$ is a pre-defined threshold and $\mathcal{M}_1(\text{Tr}_1\text{Tr}_2)$ and $\mathcal{M}_2(\text{Tr}_1\text{Tr}_2)$ denote the two shares of the product of $\text{Tr}_1$ and $\text{Tr}_2$ that are obtained by using BSMP. Then they will involve in BSCP with $Z_1$ and $Z_2$ as the two private inputs. If $Z_1 < Z_2$, then we claim that the point $D_1(x, y, \sigma)$ is stable, otherwise eliminate this point.

**Remarks.** To reduce the rounds of interactions in the implementation of BSCP and BSMP, $S_1$ can prepare all the data simultaneously and transmit all the encrypted data in a one-time transmission. For example, in detecting the candidate keypoints, $S_1$ and $S_2$ can compute $\Delta D_1$ and $\Delta D_2$ between all neighboring two points in the scale space simultaneously. Then $S_1$ can transmit all encrypted $\Delta D_1$’s to $S_2$ in a one-time transmission. Consequently, the number of rounds of interactions required in detecting all the keypoints is 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 $D(x, y, \sigma)$, associated with a parameter $\sigma$, is used to select the Gaussian smoothed image $L$ 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 $\sigma$ in the following discussions.

For the orientation computation on the keypoint $L(x, y)$ (which is mapped back from $D(x, y))$, $S_1$ and $S_2$ will first determine which is larger between $L(x, y + 1)$ and $L(x, y - 1)$, and between $L(x + 1, y)$ and $L(x - 1, y)$, respectively. Here, we set two variables $\alpha$ and $\beta$ as: $\alpha = 1$ if $L(x, y + 1) \geq L(x, y - 1)$; otherwise $\alpha = -1$, and $\beta = 1$ if $L(x + 1, y) \geq L(x - 1, y)$; otherwise $\beta = -1$. Secure comparisons of the two pairs can be achieved by using the protocol BSCP with $L_1(x, y + 1) - L_1(x, y - 1)$ and $L_2(x, y + 1) - L_2(x, y - 1)$ being the two inputs from $S_1$ and $S_2$ respectively, $L_1(x + 1, y) - L_1(x - 1, y)$ and $L_2(x + 1, y) - L_2(x - 1, y)$ being the other pair of inputs. 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, $S_1$ computes

$$
\Delta \text{Diff}_1 = \text{Diff}_{1y} - k\text{Diff}_{1x} = \alpha(L_1(x, y + 1) - L_1(x, y - 1)) - k\beta(L_1(x + 1, y) - L_1(x - 1, y)).
$$

On the other side, $S_2$ will get $\Delta \text{Diff}_2 = \text{Diff}_{2y} - k\text{Diff}_{2x}$, where $k$ is a constant used to determine the orientation interval. Then, $S_1$ and $S_2$ will involve in BSCP computation with $\Delta \text{Diff}_1$ and $\Delta \text{Diff}_2$ as the two inputs. If $\Delta \text{Diff}_1 \geq \Delta \text{Diff}_2$, 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 $10^\circ$.

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

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

where $\text{Diff}_x = L(x + 1, y) - L(x - 1, y)$, $\text{Diff}_y = L(x, y + 1) - L(x, y - 1)$. Accordingly, $S_1$ and $S_2$ compute gradient magnitudes $m_1(x, y) = \text{Diff}_{1x} + \text{Diff}_{1y}$ and $m_2(x, y) = \text{Diff}_{2x} + \text{Diff}_{2y}$ 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 $H_1$ and $H_2$.

3) **Finding Peaks in the Orientation Histogram:** To detect peaks in the original orientation histogram $H$, all we need is to find whose accumulated gradient magnitude is larger between two bins. Let $\sum_{10} m_1$ and $\sum_{20} m_1$ denote the accumulated gradient in $H_1$ whose orientations lie in $0^\circ \sim 10^\circ$ and $10^\circ \sim 20^\circ$ respectively. Let $\sum_{10} m_2$ and $\sum_{20} m_2$ denote those computed in $H_2$. Then with the aid of BSCP with $\sum_{10} m_1 - \sum_{20} m_1$ and $\sum_{10} m_2 - \sum_{20} m_2$ being the two inputs, if the former is larger, 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 $H_1$, $S_1$ and $S_2$ will compute which is larger between $\sum L_P m_1 - 0.8 \sum H_P m_1$ and $\sum L_P m_2 - 0.8 \sum H_P m_2$ where $\sum L_P m_i$ corresponds to the bin with local peak and $\sum H_P m_i$ ($i = 1, 2$) 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 encrypting all the data simultaneously and transmitting the encrypted data at one time.*

**D. Descriptor Generation Protocol**

Once $S_1$ and $S_2$ 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
O will be sent to O to meet both the effectiveness and the security constraints. We will show that the proposed scheme is carefully constructed to meet both the effectiveness and security constraints described in our design goals in Section V.

### VIII. System Analysis

In this section, we will give a careful analysis of the proposed scheme in terms of effectiveness and security. We will show that the proposed scheme is carefully constructed to meet both the effectiveness and the security constraints described in our design goals in Section V.

### 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

\[ \Delta D_1 - \Delta D_2 = [D_1(x, y, \sigma) - D_1(x, y + 1, \sigma)] - [D_2(x, y, \sigma) - D_2(x, y + 1, \sigma)] \]

\[ = [D_1(x, y, \sigma) - D_2(x, y, \sigma)] - [D_1(x, y + 1, \sigma) - D_2(x, y + 1, \sigma)] \]

\[ = D(x, y, \sigma) - D(x, y + 1, \sigma) \]

which means that the comparison of \( \Delta D_1 \) and \( \Delta D_2 \) is equivalent to determine which is larger between \( D(x, y, \sigma) \) and \( D(x, y + 1, \sigma) \). Thus, all candidate keypoints can be correctly localized by the cloud servers. Note that in the above equations we use the property \( D(x, y, \sigma) = D_1(x, y, \sigma) - D_2(x, y, \sigma) \), which can be easily deduced from the equation \( \mathcal{L}(x, y) = G \ast I(x, y) = (I_1(x, y) - I_2(x, y)) = I_1(x, y) - I_2(x, y) \).

Then in the process of edge response eliminating, when computing derivatives we have

\[ R_{xx} = D(x + 1, y, \sigma) + D(x - 1, y, \sigma) - 2D(x, y, \sigma) \]

\[ = D_1(x + 1, y, \sigma) + D_1(x - 1, y, \sigma) - 2D_1(x, y, \sigma) \]

\[ - [D_2(x + 1, y, \sigma) + D_2(x - 1, y, \sigma) - 2D_2(x, y, \sigma)] \]

\[ = R_{1xx} - R_{2xx} \]

Similarly, we can have \( R_{yy} = R_{1yy} - R_{2yy} \) and \( R_{xy} = R_{1xy} - R_{2xy} \), which will lead to the following equations when computing the Hessian’s determinants,

\[ \text{Det} = R_{xx}R_{yy} - R_{xy}^2 \]

\[ = (R_{1xx} - R_{2xx})(R_{1yy} - R_{2yy}) - (R_{1xy} - R_{2xy})^2 \]

\[ = [R_{1xx}R_{1yy} - R_{1xy}^2 - (M_1R_{1xx}R_{2yy})] \]

\[ + M_1(R_{1yy}R_{2xx} - 2M_1(R_{1xy}R_{2xy})) \]

\[ + [R_{2xx}R_{2yy} - R_{2xy}^2 - (M_2R_{1xx}R_{2yy})] \]

\[ + M_2(R_{1yy}R_{2xx} - 2M_2(R_{1xy}R_{2xy}))] \]

\[ = \text{Det}_1 + \text{Det}_2 \]

Note that the above equations hold due to the property \( M_1(R_{1xx}R_{2yy} + M_2(R_{1xy}R_{2xy}) = R_{1xx}R_{2yy} \). Based on the above results, to determine whether the candidate keypoint is a stable keypoint, if \( \text{Det}_1 + \text{Det}_2 \leq 0 \), we have \( \text{Det} \leq 0 \) which indicates that the candidate keypoint has different signs for curvatures and should be rejected immediately. If \( \text{Det}_1 + \text{Det}_2 > 0 \), we have

\[ Z_1 - Z_2 = rT_r^2 - 2rM_1(Tr_1 Tr_2) - (r + 1)^2 \text{Det}_1 \]

\[ - [-rT_r^2 - 2rM_2(Tr_1 Tr_2) - (r + 1)^2 \text{Det}_2] \]

\[ = rT_r^2 + T_r^2 - 2M_1(Tr_1 Tr_2) - 2M_2(Tr_1 Tr_2) \]

\[ - (r + 1)^2 \text{Det}_1 - \text{Det}_2 \]

\[ = rT_r^2 - (r + 1)^2 \text{Det}_1 + \text{Det}_2 \]

\[ = rT_r^2 - (r + 1)^2 \text{Det} \]

Note that the above equations hold based on \( M_1(Tr_1 Tr_2) \) + \( M_2(Tr_1 Tr_2) = Tr_1 Tr_2 \) and \( Tr = R_{xx} + R_{yy} = Tr_1 - Tr_2 \).
Then if $Z_1 < Z_2$, there has $\frac{T^2}{\text{Diff}} < \frac{(x+1)^2}{T^2}$, which indicates that the candidate keypoint has a low principal curvature and is a stable one.

Note that in our scheme, we cannot remove some unstable keypoints by fitting a 3D quadratic function as the original SIFT did [21]. This is because this filtering step contains division and derivative operations that are difficult to be implemented in the ciphertext domain. Therefore, as shown in Fig. 3, we compare the number of keypoints generated in Lowe’s scheme [21] and ours. As the number of the keypoints varies greatly with the images’ content, the average number of keypoints cannot give a better illustration and thus we only choose three specific images with size of $3000 \times 3000$ for highly structured image, poorly structured image and texture image. Fig. 3 shows that the number of keypoints generated in our scheme is about 1.5 times as much as that of Lowe’s. This is roughly equivalent to the results presented in [21].

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

$$\Delta \text{Diff}_1 - \Delta \text{Diff}_2 = |\text{Diff}_{1y} - k\text{Diff}_{1x}| - |\text{Diff}_{2y} - k\text{Diff}_{2x}|$$

which leads to $|\text{Diff}_{1y} - k\text{Diff}_{1x}| \geq |\text{Diff}_{2y} - k\text{Diff}_{2x}|$, which leads to $k|\text{Diff}_{1y}| \geq k|\text{Diff}_{2y}|$. Therefore, we can claim that the direction degree of this keypoint is larger than $\arctan k$.

As discussed in Section VII-C, by choosing a series of $k$’s, the orientation of each keypoint can be located in an interval whose span is $10^\circ$. 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 $\sum_{10^\circ} m_1 = w_1 m_1(x, y)$ and $\sum_{20^\circ} m_1 = w_2 m_1(x', y')$, 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 $S_1$ consist of only one point. Accordingly, we can get $\sum_{10^\circ} m_2 = w_1 m_2(x, y)$ and $\sum_{20^\circ} m_2 = w_2 m_2(x', y')$ computed on $S_2$. Then, when comparing $\sum_{10^\circ} m_1 - \sum_{20^\circ} m_1$ and $\sum_{10^\circ} m_2 - \sum_{20^\circ} m_2$, we have

$$\sum_{10^\circ} m_1 - \sum_{20^\circ} m_1 - \sum_{10^\circ} m_2 + \sum_{20^\circ} m_2 = w_1 m_1(x, y) - w_2 m_2(x, y)$$

which explains that this comparison 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 orientations can be correctly computed by the Orientation Assignment Protocol.

Descriptor Generation Protocol: First, we will theoretically 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 $5^\circ$. Since the rotation degrees 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 $\frac{1}{36} \times \frac{1}{2} = \frac{1}{36} \approx 6\%$. In order to further demonstrate that this probability is low enough to be ignored in practice, we define the error rate as:

$$\text{Err}_v = \frac{V - V'}{|V'|}$$

where $V$ denotes the feature vector extracted by our scheme and $V'$ denotes the feature descriptor extracted by the original SIFT. As shown in Fig. 5(a), the maximum error rate of the orientation is lower than 5%, which is small enough to be ignored in practice. In Section IX, our experimental results will further validate our theoretical analysis.

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

$$\sum_{10^\circ} m_1 - \sum_{10^\circ} m_2 = w_1 m_1(x, y) - w_2 m_2(x, y)$$

$$= w_1 m_1(x, y) = \sum_{10^\circ} m.$$  

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 $V = V_1 - V_2$ 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 $I$ 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 $300 \times 300$ and each pixel has 8 bits, the server has to use brute-force approach which needs $256^{300 \times 300}$ operations to recover the image, and it is computationally-infeasible in practice. As for the Keypoint Localization Protocol, the two servers interact with each other and may derive some additional information. Since our proposed interaction protocols BSMP and BSCP
are built on semantically-secure SHE and all the intermediate results are generated in the encrypted form, the two servers cannot learn anything but the final encrypted results. Specifically, in BSMP, \( S_1 \) only observes a random encryption of the two private inputs’ product as \( s = xy - r \), where \( r \) is random and held by \( S_2 \). Thus, both \( S_1 \) and \( S_2 \) can derive nothing at all. In BSCP, the servers only know large or small of pixel values between any two neighbors, thus they are unable to compute the specific value of each pixel. Similarly, in the Orientation Assignment Protocol, the servers can obtain nothing but some more complex but useless relationships between pixels in a small region around keypoints. For example, they may get \( L(x, y + 1) \geq L(x, y - 1) \) and \( |\text{Diff}_{xy}| \geq k|\text{Diff}_{xy}| \) in orientation computation, and \( \sum_{100}^{200} m \geq \sum_{200}^{400} m \) in finding peaks. These available inequalities are still insufficient to compute the exact value of \( L(x, y) \). Therefore, we claim that the confidentiality 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. 4 depicts the cloud server’s possible reconstructions of twelve images that fulfill this relationship. These images are randomly chosen from twelve categories of Caltech256 Dataset [28], and correspond to a bat, a clutter, a duck, a iguana, a butterfly, a touring-bike, a centipede, a chimp, a comet, a crab, a toad and a fighter-jet from Fig. 4(a) to Fig. 4(l). Obviously, it is almost impossible to recognize the original content in the recovered images, let alone the details. On the other hand, we show that there exists an 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 \( n = 10 \), the values of pixels in each square have \( \left(\frac{256}{4}\right)^2 \) 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 \( \left(\frac{256}{4}\right)^2 \) squares in the image, the total number of different possibilities is at least \( \left(\frac{256}{4}\right)^2 \). Hence, there has an exponential number of possible reconstructions of the original image, which is impractical for the servers to launch an exhaustive search.

IX. EXPERIMENTAL EVALUATION

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. All our experiments are implemented by C++ programming language on Linux machines, each of which is with an Intel Core CPU 3.1GHz processor and a 12GB RAM running Ubuntu 14.04 LTS.

A. Effectiveness Evaluation

1) Descriptor Evaluation: In the following experiments, we compare our scheme with Qin et al.’s scheme [17], Hsu et al.’s scheme [14] and the original SIFT proposed by Lowe [21] at the same time. 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 [29]. That is, given an interest point, we will find 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 [30], which contains images of graffiti-covered walls taken from different camera viewpoints changes. For each change, it contains 6 images with a gradual geometric or photometric transformation, and all images in this dataset are approximately in \( 800 \times 600 \) pixels. We then use well-acknowledged evaluation metrics recall and 1-precision defined in [31] to quantify our results: \( \text{recall} = \frac{\text{number of correct-matches}}{\text{total number of positives}} \), and \( 1-\text{precision} = \frac{\text{number of false-matches}}{\text{total number of matches}} \), 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. 6, we plot recall VS. 1-precision for the image matching experiment by varying the distance ratio threshold \( t \) \((0 \leq t \leq 1)\). In Fig. 6(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 scheme both perform better in finding the correct matches, 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 recalls of Qin et al.’s scheme and Hsu et al.’s scheme are low (i.e., smaller than 0.2) and their 1-precision are high (i.e., larger than 0.1). Note that when \( t \) approaches to 1, the 1-precision in our scheme and Lowe’s remain around 0.5 but that of Qin et al.’s scheme and Hsu et al.’s scheme have quickly increased to 0.7. The results indicate that Qin et al.’s scheme and Hsu et al.’s scheme will generate much more false matches than our scheme and Lowe’s under the same threshold. The reason why Qin et al.’s scheme and Hsu et al.’s scheme have the same performance is that they always generate the same feature descriptors.

Fig. 6(b) is obtained 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 and Hsu et al.’s scheme.

Fig. 6(c) measures the performance for images with a significant amount blur which is introduced by changing the camera focus. The results show that all descriptors generated by the three schemes are affected by this type of image degradation. But Lowe’s and our scheme clearly dominate Qin et al.’s scheme and Hsu et al.’s scheme especially when the threshold \( t \) is rather low. This is because [17] and [14] are not able to eliminate unstable points that are poorly localized along an edge (and are therefore sensitive to noise).

Fig. 6(d) shows the results for illumination changes which have been obtained by changing the camera setting. It again shows that both our scheme and Lowe’s perform better than the competing approaches. However, [17] and [14] have a low recall due to the reason that they have computed their feature descriptors without sampling in a region.

To further demonstrate the property of invariance to rotation and image scale, Fig. 7 depicts the precision as a function of rotation angle and scale ratio respectively. It shows that both our scheme and Lowe’s scheme maintain a stable and high precision (i.e., always larger than 0.9) no matter how the image rotated. On the contrary, the precision of Qin et al.’s scheme and Hsu et al.’s scheme fall sharply when the image is only rotated by 20°. When it comes to scale change, both our scheme and Lowe’s decrease slowly in precision with the increase of scale ratio, and they will stay above 0.4 at last, while the precision of Qin et al.’s scheme and Hsu et al.’s scheme stay lower than 0.2 all the time. The results indicate that both our scheme and Lowe’s provide a good invariance to rotation change and a certain degree of invariance to image scale, but Qin et al.’s scheme and Hsu et al.’s scheme provide neither of the two properties.

2) Keypoint Matching Experiment: In Fig. 8, we provide a comparison of three scheme’s keypoint matching results visualized for illustration purpose. The images are selected from real-world scenes taken from different viewpoints. For a better illustration, we manually set the thresholds to have each scheme return only 10 matches, which contain correct matches and false positives. It clearly shows that our scheme approaches Lowe’s original SIFT and outperforms Qin et al.’s scheme and Hsu et al.’s scheme in finding correct matches.

3) Image Retrieval Experiment: In Fig. 9, we conduct image retrieval experiments to evaluate the descriptors inspired from [32]. We randomly select 11 categories from Oxford database [33] which consists of 5062 images collected from Flickr as the dataset by searching for particular Oxford landmarks. Given a query image, it returns a ranked list of images that contain the same objects by using the voted based method. The number below each returned image represents how many votes it receives. It clearly shows that our scheme receives much more votes than the original SIFT. We think this may be caused by much more keypoints generated in our scheme. We can also see that Lowe’s scheme and ours generate less false positives than Qin et al.’s and Hsu et al.’s.

B. Efficiency Evaluation

In Fig. 10, we evaluate the computation efficiency of privacy-preserving SIFT outsourcing with the increase of image size on the data owner side. The time cost and communication cost are computed for one image with different sizes. As has been analyzed in Section IV-1, Hsu et al’s scheme incurs an intolerable computation burden on the data owner, so we
Qin et al.'s scheme: 3/10 correct
Ours: 7/10 correct
Lowe's scheme: 8/10 correct
Hsu et al.'s scheme: 3/10 correct

1000 600 400 400 160 (b)
1000 1500 1200 1400 2000 1600 1200 1400 2000 × 200 × 0 40 × 40 80 × 80 120 × 120 160 × 160 Image size (pixels)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2016.2568460, IEEE Transactions on Image Processing

Fig. 9: A comparison of four 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, and the images without it are the false positives.

Fig. 8: 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.

Fig. 10: A comparison of computation and communication costs on the data owner without and with outsourcing in two different schemes.

Fig. 11: A comparison of computation and communication costs on the cloud servers between using BSCP and Garbled Circuit, BSMP and SPP.

only compare our scheme with Qin et al.’s scheme. Fig. 10(a) shows the time cost of performing SIFT algorithm locally and that of outsourcing of SIFT scheme. 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. This is because without outsourcing the data owner needs to perform full extraction algorithm while in our scheme it only needs to do some addition operations. With the increase of image size, the benefit of outsourcing will become more apparent. The side effect of outsourcing is generating some communication overheads on the data owner. In our scheme, however, the communication cost is low enough to be acceptable by the data owner as shown below.

As for the communication cost on the data owner side as illustrated in Fig. 10(b), since the data owner only sends two split images to the two cloud servers and receives two split descriptors with fixed size, our scheme and Qin et al.’s scheme have the same communication overhead which grows linearly with the size of an image.

In Fig. 11, we evaluate the time cost and the communication cost of our scheme on the server side. Note that the servers and the data owner are all simulated on machines of the same configuration without using multithreading or any other parallel techniques. To make a better illustration, we also compare i) the secure product protocol (SPP) proposed in [26] that adopts the Paillier cryptosystem [15] with our BSMP, and ii) a general secure comparison protocol-garbled circuit [20] with our BSCP. Since the protocols BSMP and BSCP take up most of the time costs (because of encryption and decryption operations) and are the only source of communication cost, we only evaluate these two protocols and ignore the other steps such as building gaussian scale spaces, computing descriptors etc. In our experiments, we set the Paillier cryptosystem with a 1024-bit modulus and implement SHE with the aid of SIMD. Garbled circuit is tested using the lastest optimizations [34].

Fig. 11(a) shows the time cost on the servers. With the increase of image size, the time cost in garbled circuit grows much faster than that of BSCP, and SPP also consumes much more than BSMP. According to our experimental results, when the image size grows to 500 × 500, garbled circuit and SPP will cost more than 10854 minutes and 3799 minutes respectively, while the costs in BSCP and BSMP are about 235 minutes and 2.2 minutes respectively. Fig. 11(b) evaluates the communication cost between two servers which is generated by using BSCP and BSMP. It also shows that communication cost of BSMP is much lower than that of SPP, and BSCP also outperforms the garbled circuit greatly. Particularly, when the
image size grows to 500 × 500, garbled circuit costs more than 8458MB while BSCP costs about 2968MB, and SPP needs about 2226MB while BSMP costs about 240MB. The above results indicate that the proposed protocols BSCP and BSMP are more suitable for practical applications. Note that when the image size grows to 3000 × 3000, our scheme will consume about 711 minutes and 8875MB in total. Our future work may focus on further reducing the time and communication costs for large-size images, by exploring some parallel techniques and more packing methods.

As for the interactions between two servers, as we have shown in section VII, a constant number of round can be achieved by sending all the encrypted data in one-time transmission. The specific number of interactions depends on the parameters chosen in somewhat homomorphic encryption scheme. In our experiment, it needs dozens of rounds to complete the entire extraction process.

Finally, for the memory cost on the servers, for a n × n image it at most needs 12 × n × n bytes where 12 is a pre-defined parameter to decide the number of scales in scale space. Thus when n = 3000 the memory cost is about 103MB, which is still acceptable for an off-the-shelf machine, let alone a powerful cloud server.

X. Future Research Directions
In this section, we will give some discussions on the extensions of our scheme.

First, SIFT is now widely computed in a dense manner. Dense SIFT computes a descriptor for every pixel in the image instead of locating points that we are only interested in. The other steps such as orientation assignment and descriptor computation are exactly the same as SIFT. Therefore, we claim that our scheme can be directly applied to dense SIFT outsourcing. This is because the two servers only need to proceed the other steps identically without performing the keypoint localization protocol.

Second, one may want to aggregate SIFT descriptors into a compact representation by using Fish vectors [35] or VLAD [36]. In fact, it is also possible to compute VLAD by using a secure outsourcing protocol. This can be achieved by using techniques in [37] to outsource k-means algorithm to the cloud, and then computing aggregated descriptors which only needs addition operations. As for Fisher vectors, since it contains many derivative and division operations which are difficult to be implemented in the ciphertext domain, we are not sure whether it is suitable for outsourcing yet. We will investigate this issue in our future work.

Finally, feature descriptors are usually used in content-based retrieval such as k-nn search. As far as we know, secure outsourcing of k-nn search has been widely studied and many solutions have been proposed such as [8]. In [8], the data owner only needs to encrypt descriptors and sends ciphertexts to the two cloud servers. The cloud servers will then work cooperatively to complete the k-nn search task and return an accurate search result without obtaining anything private about the descriptors. This method can be directly combined with our scheme such that the data owner can relieve himself from the burden of computationally-intensive search jobs.

XI. Conclusion
Privacy-preserving outsourcing of image feature extraction offers the promise of obtaining feature descriptors dependent on private data without exposing the data owner’s privacy, while enjoying the abundant cloud computation resources. However, previous solutions for secure SIFT outsourcing have either security or efficiency issues, and none can well preserve the important characteristics of the original SIFT in terms of distinctiveness and robustness. In this work, we presented a new and novel privacy-preserving SIFT outsourcing protocol based on newly proposed secure interactive protocols BSMP and BSCP. 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.

Acknowledgment
Qian’s research is supported in part by National Natural Science Foundation of China under Grant No. 61373167, National Basic Research Program of China (973 Program) under Grant No. 2014CB340600, National High Technology Research and Development Program of China (863 Program) under Grant No. 2015AA016004, Wuhan Science and Technology Bureau under Grant No. 2015010101010020, and Fundamental Research Funds for the Central Universities under Grant No. 2042016kf0137. Kui’s research is supported in part by US National Science Foundation under grants CNS-1262277. Qian Wang is the corresponding author.

References