GEO-TAGGING AND PRIVACY-PRESERVATION IN MOBILE CLOUD COMPUTING

A Dissertation

Presented to

The Faculty of the Graduate School

At the University of Missouri

In Partial Fulfillment

Of the Requirements for the Degree

Doctor of Philosophy

By

QIA WANG

Dr. Wenjun Zeng, Dissertation Supervisor

December 2013

The undersigned, appointed by the dean of the Graduate School,

have examined the dissertation entitled

GEO-TAGGING AND PRIVACY-PRESERVATION IN MOBILE CLOUD COMPUTING

Presented by Qia Wang

A candidate for the degree of

Doctor of Philosophy

And hereby certify that, in their opinion, it is worthy of acceptance.

Professor Wenjun Zeng

Professor Yi Shang

Professor Chi-Ren Shyu

Professor Tony Xu Han

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to my advisor, Dr. Wenjun Zeng for his support, guidance and encouragement during my Ph.D. study. I enjoy all the meetings and discussions with him. He is very open-minded and patient, and always elevates me to a higher level with his sharp thoughts and accumulated experiences. He not only trains me to have solid technical skills, but also, more importantly, to be an independent thinker and researcher. His enthusiasm and rigorous attitude toward his work has motivated and will continue to inspire me in my future life and career.

I would like to thank my committee members, Dr. Yi Shang, Dr. Chi-Ren Shyu and Dr. Tony Han. Thanks for their time out of their busy schedules and constructive suggestions during the completion of this dissertation. Also I thank Dr. Hong S. He and the GIS lab members at the Forest Department, I enjoy the projects working with the foresters.

I would like to thank Dr. Heather Yu and Dr. Jun Tian from the Media Networking Lab, Futurewei Technologies in New Jersey. My internships and the collaborative projects with Futurewei broaden my vision and enrich my research experiences and they are invaluable for my future career.

Many thanks go to the fellow members in our lab, Wei Liu, Yingnan Zhu, Lina Dong, Qiuming Yao, Aleksandre Lobzhanidze, Suman Deb Roy. I thank their help and appreciate the friendship established with them. I thank my friend Peng Zhuang and Dan

Wang from our department, it was a great experience working with them together as a team.

I thank the Department of Computer Science at University of Missouri. Thanks for the high quality courses and great efforts offered by the faculty and staff members.

Lastly, the special appreciation and love go to my family. I am eternally grateful to my parents for their endless love and support for my life and work. I thank my girlfriend Qingning Zhou. It is so blessed to meet my special you in the right time, at the right place. Thank you for the company and love during the busiest time of my Ph.D. study.

TABLE OF CONTENTS

ACKNOWLEDGEMENT ii

TABLE OF CONTENTS iv

LIST OF FIGURES viii

LIST OF TABLES xi

ABSTRACT xiii

Chapter 1. Introduction 1

1.1. Motivations 1

1.2. Summary of Contributions 7

1.3. Overview of the Dissertation 9

Chapter 2. Background and General Knowledge 11

2.1. Imaging Geometry 11

2.1.1. Pin-hole Camera Model 11

2.1.2. Multi-View Geometry 13

2.2. Homomorphic Encryption Systems 14

2.2.1. Public Key Encryption Systems 15

2.2.2. Public Key Homomorphism 16

2.2.3. Handling Real Values through Scaling 17

Chapter 3. A Mobile Geo-tagging System 19

3.1. Introduction 19

3.2. Related Works 22

3.2.1. Image based Localization Systems 22

3.2.2. Video based Object Tracking Systems 23

3.3. Single Image-based Remote Target Localization 26

3.3.1. Algorithm 26

3.4. Two Image-based Remote Target Localization 29

3.4.1. Algorithm 29

3.5. Video-based Remote Target Tracking 32

3.5.1. Algorithm 32

3.5.2. Kalman Filtering 37

3.6. Experimental Results 39

3.7. Speed Optimization 44

3.7.1. Two-image based Localization System Optimization 44

3.7.2. Video-based Remote Target Tracking System Optimization 48

3.8. Conclusion and Future Works 51

Chapter 4. A Compressive Sensing based Privacy Preserving Data-mining and Signal Processing Framework 53

4.1. Introduction 53

4.2. Preliminary Knowledge 55

4.2.1. Compressive Sensing 55

4.2.2. Secure Multiparty Computation 56

4.3. Related Works on Privacy-preserving Data-mining 57

4.3.1. Privacy Preserving Data-mining 57

4.3.2. Secure Multiparty Computation for Privacy Preserving Data-mining 58

4.3.3. Compressive Sensing Domain Data-mining 60

4.4. The Proposed Framework 61

4.5. The Proposed Secure Multiparty Computation Protocols 63

4.5.1. Existing MPC Protocols Referenced by the Proposed Framework 63

4.5.2. MPC based CS Transformation Protocol 65

4.5.3. MPC based CS Reconstruction Protocol 65

4.6. Framework Analysis 69

4.6.1. Complexity Analysis 69

4.6.2. Security Analysis 73

4.7. Complexity Comparison to MPC based Data-mining Algorithm 75

4.8. Experimental Results 78

4.8.1. Compressive Sensing Domain Data Processing 78

4.8.2. Experiments of MPC for Compressive Sensing 80

4.9. Conclusion and Future Works 82

Chapter 5. Compressive sensing based privacy preserving storage and secure watermark detection framework 84

5.1. Introduction and Related Works 84

5.2. Correlation-based Watermarking System 86

5.3. Watermark Detection in the Compressive Sensing Domain 88

5.4. The Secure Watermark Detection Framework 92

5.5. Framework Analysis 96

5.5.1. Complexity Analysis 96

5.5.2. Security Analysis 97

5.5.3. Comparison to Previous Works 98

5.6. Experimental Results 100

5.6.1. Experimental Settings and Notations 100

5.6.2. Scaling Floating Point to Integer Error Analysis 101

5.6.3. Secure Watermark Detection in the Compressive Sensing Domain 101

5.6.4. Compressive Sensing Encryption 107

5.6.5. Compressive Sensing Reconstruction 108

5.7. Conclusion and Future Works 109

Chapter 6. Use Case: A Privacy Preserving Collaborative Filtering Recommendation System for Mobile Users 111

6.1. Introduction and Related Works 111

6.2. The System 113

6.3. Experimental Results 115

6.4. Conclusions 117

Chapter 7. Conclusions and Future Works 118

7.1. Summary of the Dissertation 118

7.2. Future Works 121

APPENDIX I 123

BIBLIOGRAPHY 126

VITA 134

LIST OF FIGURES

Figure 1.1. Mobile devices in a vicinity link to each other to create a mobile cloud system. 2

Figure 1.2. Mobile devices offload the data storage and processing to the cloud to create a mobile cloud system. 2

Figure 1.3. Privacy preserving collaborative data mining in the cloud. 5

Figure 2.1. The geometry of the pin-hole camera model: π is the image plane; f is focal length; O^C is the camera coordinate origin. 11

Figure 2.2. Transformation between the world coordinate system and the camera coordinate system (Left: camera coordinate; Right: world coordinate). 12

Figure 2.3. Epipolar Geometry 13

Figure 3.1. The proposed smartphone based system has three major components: 1. Single-image based remote target localization; 2. Two-image based remote target localization; 3. Video based remote moving object tracking. 20

Figure 3.2. (a) Single-image based remote target localization: user takes a picture of a remote target and draws a tight bounding box around it; (b) Two-image based remote target localization: user inputs a rough bounding box on the left and right image enclosing the target. (The camera and the remote target positions are displayed on Google Maps. Blue dots: camera position; Red pin: estimated position of the remote target). 26

Figure 3.3. The 4-point algorithm: WIDTH and HEIGHT are the object's physical size; W_i are four 3D points and Ci are their corresponding points on the image plane. 27

Figure 3.4. The 2-point algorithm 27

Figure 3.5. Two view triangulation: P^W is a remote point; p_L and p_R are the corresponding points on the two image planes. 30

Figure 3.6. Diagram of the two-image based localization algorithm 31

Figure 3.7. Overview of the video tracking algorithm 33

Figure 3.8. Yellow cross: user's tapping point; Black square: centered by user's tapping point with edge size D; Red points: Lucas-Kanade optical flow features of the moving target. 33

Figure 3.9. Black areas are Object patches with high confidence; White areas are Background patches with high confidence. Grey level areas are unclassified patches. 37

Figure 3.10. Target tracking results on the video 41

Figure 3.11. Video tracking results with different backgrounds and remote targets 41

Figure 3.12. (a) Trajectory based on original observations with 4-point algorithm; smoothed trajectory with 4-point and 2-point algorithms after modeling with EKF. Camera position is (0, 0); (b) Estimated velocity of the remote target. 43

Figure 3.13. Progressive random block selection for feature extraction. 45

Figure 3.14. Feature matching results with progressive block selection method. The blackout blocks are not selected for feature extraction. (a) and (b) have textured background, only 30% of the left image are used. (c) has more uniform background and 75% of the left image are used. 47

Figure 3.15. Video temporal down-sampling 51

Figure 3.16. Tracking results with temporal and spatial down-sampling ("$X \times Y$, n" denotes spatial down-sampling to resolution $X \times Y$ and temporal down-sampling by a factor of n). Camera position is (0, 0). 51

Figure 4.1. Architecture of the proposed generic privacy preserving data-mining framework. 61

Figure 5.1. Architecture of the proposed secure watermark detection framework 92

Figure 5.2. Some DCT coefficients (DCT^1) are only used for storage; The other coefficients (DCT^2) serve for both the storage and watermark detection purposes 94

Figure 5.3. 8x8 DCT block zig-zag scan order: (a) an example of selective DCT coefficients for watermark embedding: the bold underlined numbers indicate that the corresponding DCT channels are embedded with watermark signals. (b) top 20 AC coefficients (white areas) are selected for watermark detection in the CS domain (i.e., DCT^2 in Figure 5.2). 94

Figure 5.4. Watermark detection in the CS domain under different CS rates when different DCT components are selected. For the legend of the figure, "real" means the mean of q^{CS} calculated from $\{z_i\}$, while "est" means the estimated mean of q^{CS} based on Equation (5.9). The number "xx" means "Top AC xx". (for the 512x512 'Lenna' image) 103

Figure 5.5. qcs's distribution under $H0$ and $H1$ with CS rate 0.1 using the top 20 AC coefficients as DCT^2 and the top 63 AC coefficients as DCT^2 (for the 512x512 'Lenna' image) 104

Figure 5.6. The mean of qcs at different CS rates under zero-mean Gaussian noise attack (i.e. “20 atk” means Gaussian noise with zero mean and standard deviation 20); The mean of q in the original domain after CS reconstruction (i.e. if top 63 AC coefficients are selected for CS transformation, denoted as “CS recons atk, 63”). (for the 512x512 ‘Lenna’ image) 105

Figure 5.7. Watermark detection performance in the CS domain for other images (i.e. “Goldhill, 63” means Goldhill image with top 63 AC coefficients selected) 107

Figure 5.8: (a) Original image; (b) Image in 8x8 DCT domain; (c) DCT coefficients after CS transformation; (d) Image reconstruction with the wrong CS matrix. (CS rate 1.0 is chosen here, similar effects are observed under other CS rates) 107

Figure 5.9. CS reconstruction distortion when top 63 AC coefficients are transformed to the CS domain 109

Figure 6.1. The architecture of the mobile privacy preserving collaborative filtering system. 114

LIST OF TABLES

Table 3.1. Single-image based localization accuracy (Unit: Yards) 40

Table 3.2. Two-image based localization accuracy (Unit: Yards) 40

Table 3.3. Complexity distribution of two-image based localization on PC and Android smartphones respectively (1: Feature extraction, 2: Feature matching, 3: others.) 45

Table 3.4. Two image based system optimization 47

Table 3.5. Video based tracking complexity distribution (1: Motion estimation; 2: Segmentation; 3: Connected component analysis; 4: Optical flow feature tracking; 5: Edge detection; 6: others) 49

Table 3.6. Video tracking optimization considering user's input (Motion: motion estimation; Optical: optical flow feature tracking; spf: second per frame) 49

Table 3.7. Tracking speed and accuracy with video down-sampling 51

Table 4.1. Protocol 1's computational complexity per vector in number of modular multiplications when the vector length is N. 70

Table 4.2. Protocol 2's computational complexity in number of modular multiplications 70

Table 4.3. Summary of computational complexity in number of modular multiplications 70

Table 4.4. Summary of the communication complexity (in bits) of the fundamental protocols 72

Table 4.5. Relative errors of the Square of the Euclidean distance of Two Attributes 79

Table 4.6. Relative cluster error of K-means clustering of UCI KDD data 80

Table 4.7. Average communication costs of encrypting a vector (data in public key domain only). 81

Table 4.8. Average communication costs of decrypting a vector (data in public key domain only) 82

Table 5.1. Mean square error w.r.t scaling factor 101

Table 5.2. Test results of the covariance term in Equation (5.9) 102

Table 5.3. Validation for the assertion: $\lambda > 0$ (when $m = n$). (for the 512x512 ‘Lenna’ image).. 102

Table 5.4. Watermark detection (q under H1) in original domain with different DCT coefficients. (for the 512x512 ‘Lenna’ image) 103

Table 5.5. Average q^{CS} and β for different images when different Top AC coefficients are chosen (CS rate = 0.1)... 106

Table 6.1. An example of user’s visiting history... 114

Table 6.2. Relative errors of the recommendation results in CS domain w.r.t the original domain.. 116

Table 6.3. Communication cost of the proposed system (data in public key domain only). 117

ABSTRACT

With the emerge of the cloud computing service and the explosive growth of the mobile devices and applications, mobile computing technologies and cloud computing technologies have been drawing significant attentions. Mobile cloud computing, with the synergy between the cloud and mobile technologies, has brought us new opportunities to develop novel and practical systems such as mobile multimedia systems and cloud systems that provide collaborative data-mining services for data from disparate owners (e.g., mobile users). However, it also creates new challenges, e.g., the algorithms deployed in the computationally weak mobile device require higher efficiency, and introduces new problems such as the privacy concern when the private data is shared in the cloud for collaborative data-mining.

The main objectives of this dissertation are: 1. to develop practical systems based on the unique features of mobile devices (i.e., all-in-one computing platform and sensors) and the powerful computing capability of the cloud; 2. to propose solutions protecting the data privacy when the data from disparate owners are shared in the cloud for collaborative data-mining.

We first propose a mobile geo-tagging system. It is a novel, accurate and efficient image and video based remote target localization and tracking system using the Android smartphone. To cope with the smartphones' computational limitation, we design light-weight image/video processing algorithms to achieve a good balance between estimation accuracy and computational complexity. Our system is first of its kind and we provide

first hand real-world experimental results, which demonstrate that our system is feasible and practicable.

To address the privacy concern when data from disparate owners are shared in the cloud for collaborative data-mining, we then propose a generic compressive sensing (CS) based secure multiparty computation (MPC) framework for privacy-preserving collaborative data-mining in which data mining is performed in the CS domain. We perform the CS transformation and reconstruction processes with MPC protocols. We modify the original orthogonal matching pursuit algorithm and develop new MPC protocols so that the CS reconstruction process can be implemented using MPC. Our analysis and experimental results show that our generic framework is capable of enabling privacy preserving collaborative data-mining. The proposed framework can be applied to many privacy preserving collaborative data-mining and signal processing applications in the cloud.

We identify an application scenario that requires simultaneously performing secure watermark detection and privacy preserving multimedia data storage. We further propose a privacy preserving storage and secure watermark detection framework by adopting our generic framework to address such a requirement. In our secure watermark detection framework, the multimedia data and secret watermark pattern are presented to the cloud for secure watermark detection in a compressive sensing domain to protect the privacy. We also give mathematical and statistical analysis to derive the expected watermark detection performance in the compressive sensing domain, based on the target image, watermark pattern and the size of the compressive sensing matrix (but without the actual CS matrix), which means that the watermark detection performance in the CS domain can

be estimated during the watermark embedding process. The correctness of the derived performance has been validated by our experiments. Our theoretical analysis and experimental results show that secure watermark detection in the compressive sensing domain is feasible.

By taking advantage of our mobile geo-tagging system and compressive sensing based privacy preserving data-mining framework, we develop a mobile privacy preserving collaborative filtering system. In our system, mobile users can share their personal data with each other in the cloud and get daily activity recommendations based on the data-mining results generated by the cloud, without leaking the privacy and secrecy of the data to other parties. Experimental results demonstrate that the proposed system is effective in enabling efficient mobile privacy preserving collaborative filtering services.

Chapter 1. Introduction

Mobile devices (e.g., smartphone, tablet, etc) are becoming an essential part of our daily life as the most effective and convenient communication tools. The rapid progress of mobile computing becomes a powerful trend in the development of IT technology as well as commerce and industry fields. Cloud computing has been widely recognized as the next generation's computing infrastructure. Cloud computing offers some advantages by allowing users (e.g., mobile users) to use infrastructure (e.g., servers, networks, and storages), platforms (e.g., middleware services and operating systems), and software (e.g., application programs) [1]. Mobile cloud computing, which takes advantages of both the mobile devices and the cloud, could become the dominant model [2] in the future. In this chapter, we first introduce the motivations of our proposed mobile-cloud systems and frameworks in Section 1.1. Then a summary of the contributions from this dissertation is presented in Section 1.2. We then give an overview of this dissertation in Section 1.3.

1.1. Motivations

There are basically two types of mobile cloud computing systems [2]. The first one is to consider mobile devices them-selves as the cloud making up a mobile peer-to-peer network which provides certain services and applications, as shown in Figure 1.1. There are several application scenarios that are based on the first type [3][4], e.g., video recordings from multiple mobile devices are spliced to construct a single video that covers the entire event from different angles, and perspectives. The second type of mobile cloud system is: mobile devices act as a thin client connecting to a remote cloud through the network, and offloading the data and processing to the cloud as shown in Figure 1.2. Some examples of this type are Facebook's location aware services, Twitter for mobile, mobile weather widgets [2].

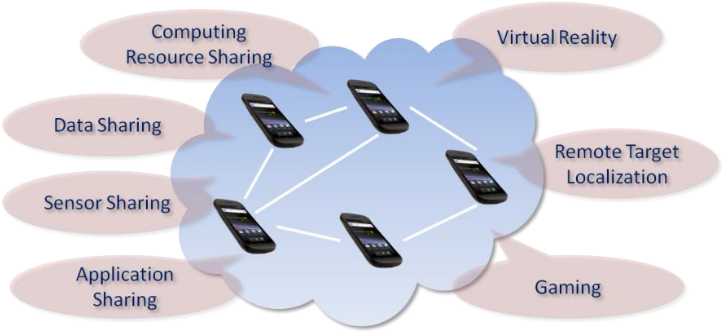


Figure 1.1. Mobile devices in a vicinity link to each other to create a mobile cloud system.

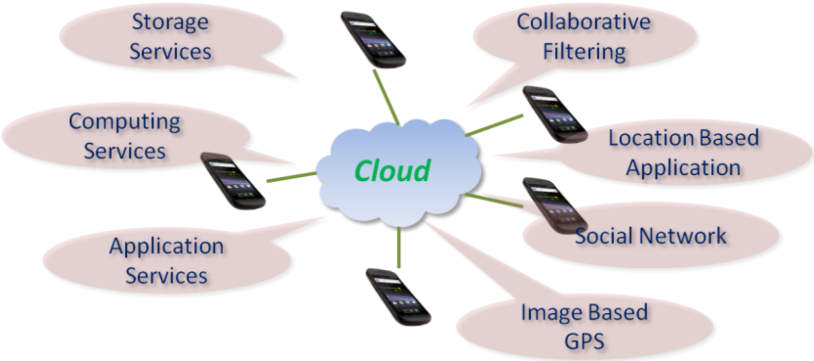


Figure 1.2. Mobile devices offload the data storage and processing to the cloud to create a mobile cloud system.

The first type of mobile cloud system is essentially a system executing on mobile devices without support from a computationally powerful server. In this dissertation, we develop a mobile geo-tagging system based on one or more connected mobile devices, i.e., an image/video based remote target localization and tracking system on the Android smartphones. Our proposed system has three major components: 1. Single-image based remote target localization when the remote target's physical size is known; 2. Two-image based remote target localization when the target size is unknown; 3. Video-based tracking when the remote target of known physical size is moving. Such system will be useful for many scenarios. For example, on the battlefield, a soldier needs a rangefinder, compass, GPS and other tools to do reconnaissance before calling for an air strike. On the golf course, players often would like to estimate the distance to the green. Biologists can document the location of a rare animal without disturbing it. A tourist may be very interested in a remote object's GPS location while taking a picture of it; later on that picture can be associated with a map (such as Google Maps or Google Earth) and be shared with other people with the location information. On the highway, a policeman can use our video tracking system to estimate a moving vehicle's speed while taking video evidence. In previous works, mobile localization systems are all designed for locating the device itself, but not the remote target. Furthermore, the previous smartphone vision based localization systems have to interact with a server and leverage the server's resource. In our geo-tagging system, all the computations are performed on the smartphones without a remote server. We also design optimization methods that are tailored to the unique characteristics of the smartphone's computing platform to improve the efficiency. For the image based localization component, we propose an optimized feature extraction method designed for the smartphone computing platform taking into account its memory I/O and CPU limitations. For the video based tracking component, we develop a light-weight video based remote target tracking system that can accurately track the moving objects in the video on the smartphone. Our experimental results demonstrate that our system is feasible and practicable.

The second type of mobile cloud system (i.e., thin mobile clients and a remote cloud service) can be introduced to support various services including the collaborative data-mining applications. In the collaborative data-mining system, users (e.g., mobile users) can share their data with other parties in the cloud and the cloud could perform data-mining tasks with the aggregated data.

Collaborative data-mining system has received increasing attentions recently. The rapid development of the Internet, storage and database systems has given opportunities for companies, institutions and individuals to collect data by themselves. Mobile devices and smartphones also have a database to store personal data or application data. Analysis merely based on those limited data may generate some biased statistical results. It would be more accurate to aggregate data from different data holders to jointly perform data-mining to better understand the overall data characteristics. For some e-commerce companies, as an example, it will be beneficial to have a better product recommendation system for the customers by integrating with other companies' customer data. Disparate data owners can outsource their data to the cloud for data-mining and retrieve the expected results. However such applications are infeasible without the protection of individual data privacy.

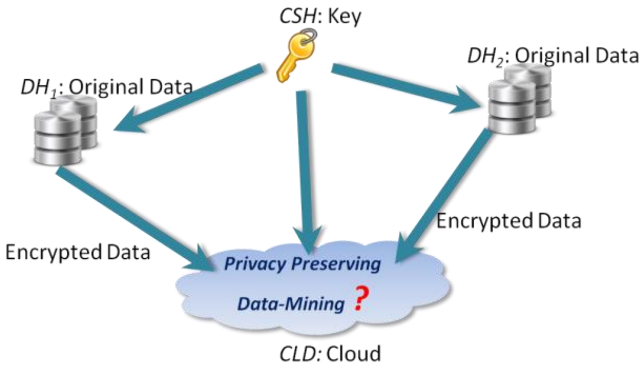


Figure 1.3. Privacy preserving collaborative data mining in the cloud.

In this dissertation, we propose a generic privacy preserving data-mining framework based on compressive sensing using secure multiparty computation. In our framework, the data-mining algorithms are performed in a compressive sensing (CS) domain to protect the data privacy, and the secure multiparty computation (MPC) protocols are used for the CS transformation and reconstruction processes. The scenario we address is illustrated in Figure 1.3: Suppose there are *m* data holders (e.g., mobile users) DH_1, DH_2,…, DH_m. Each of them has a private database and there is a third party computing cloud service provider CLD who performs data-mining tasks and a trusted key manager CSH which generates and maintains a common secret key to be used for the encryption of data (the encryption here refers to CS transformation) from all data holders. Each of the DHs would like to keep its data from being disclosed to any other parties including the CSH; and the CSH would not expose the common key to any other parties for the sake of security. In our framework, all DHs' data are transformed to the same CS domain for collaborative data-mining through MPC based CS transformation before they are outsourced to the CLD. The data-mining results are sent to DH by executing MPC based CS reconstruction process between CSH and CLD. Our analysis and experimental results demonstrate that the proposed generic framework is effective in enabling efficient privacy preserving collaborative data-mining. Our generic framework can also be extended to support many other privacy preserving data-mining and signal processing tasks, e.g., the secure watermark detection framework proposed in Chapter 5 of this dissertation.

Due to the rapid growth of the Internet and social networks, it is very easy for a user to collect a large amount of multimedia data from different sources without knowing the copyright information of those data. The user may want to take advantage of the cloud for storage, and at the same time, work with copyright owners for watermark detection while keeping those self-collected multimedia data private. The watermark pattern owner wants to keep their watermark patterns private during the watermark detection as well. Most of the existing secure watermark detection works assume the watermarked copy are publicly available and focus on the security of the watermark pattern, while the privacy of the target media on which watermark detection is performed has received little attention. But for some applications such as the scenario given above, it is required to protect the multimedia data's privacy in the watermark detection process. We therefore propose a compressive sensing based privacy preserving storage and secure watermark detection framework based on our generic privacy preserving collaborative data-mining framework. Our theoretical analysis and experimental results show that secure watermark detection in the CS domain is viable. We also demonstrate that the expected watermark detection performance in the CS domain can be estimated by the target image, the watermark pattern and the height of the CS matrix (disregarding the actual CS matrix used). This is an important result since the watermark detection performance in the CS domain can be estimated during the watermark embedding process by the content providers.

In a future ubiquitous setting [5], users will be able to routinely record their own locations via smartphones, and their purchases through digital wallets or credit card records. Their data are stored in a remote cloud and then the cloud can perform collaborative filtering so that the users could get recommendations about many of their everyday activities, including restaurants, bars, movies, and interesting sights to see and things to do in a neighborhood. However, the smartphone users may desire to keep their personal data private while using such collaborative filtering recommendation service. In this dissertation, we develop a mobile privacy preserving collaborative filtering system based on our proposed mobile geo-tagging system and generic privacy preserving data-mining framework. Our experimental results demonstrate that our system is capable to provide collaborative filtering recommendation services while protecting the data privacy.

1.2. Summary of Contributions

In this dissertation, significant efforts have been made to improve the efficiency, practicality, feasibility and security for the mobile cloud computing systems. The major contributions of our proposed mobile geo-tagging system can be summarized as:

1. The proposed smartphone based remote target localization and tracking system is first of its kind. We make the initial efforts developing the system on the smartphone and provide first hand real-world experimental results of the system. We demonstrate that

accurate image/video based remote target localization and tracking on the smartphone is feasible and practicable.

2. All the tasks are performed on the smartphone, no remote server is required. We have developed algorithms and optimization methods tailored to the smartphone computing platform by considering its unique characteristics.

The main contributions of our generic privacy preserving collaborative data-mining framework are summarized below:

1. Compared to previous random projection based privacy-preserving data-mining work [39], we improve the security by introducing a trusted third party to manage the common key used for encrypting multiple disparate data sources without revealing the key to individual data holders.
2. Compared to previous MPC based privacy-preserving data-mining, e.g., [69], the proposed framework decouples the encryption/compression process that could be shared from different data mining operations, which significantly reduces the computational and communication complexity often incurred in MPC protocols for privacy preserving data mining.
3. Compared to other MPC based privacy preserving data mining algorithms, the proposed framework offers much better scalability in that it can add more data holders and data sources without incurring additional cost to existing data holders.
4. The proposed compressive sensing encryption based secure multiparty privacy-preserving framework can be extended to develop many other privacy preserving collaborative data-mining and signal processing systems.

The main contributions of our secure watermark detection framework are summarized as follows:

1. Most of the existing secure watermark detection works paid little attention to the privacy of the multimedia data, while our framework protects the privacy of the self-collected data.
2. We derive the expected watermark detection performance in the compressive sensing domain, given the target image, watermark pattern and the size of the compressive sensing matrix (but without the CS matrix itself). The correctness of the derived performance has been validated by our experiments.
3. Our theoretical analysis and experimental results show that secure watermark detection in the compressive sensing domain is feasible.

1.3. Overview of the Dissertation

The rest of the dissertation is organized as follows:

Chapter 2 introduces some background knowledge including imaging geometry and homomorphic encryption systems referenced by this dissertation.

Chapter 3 presents the proposed mobile geo-tagging system. In this chapter, the system is proposed first and then we present the experimental results and the speed optimization methods showing that our system is practicable and feasible.

Chapter 4 introduces our generic compressive sensing based privacy preserving collaborative data-mining framework using secure multiparty computation. We present the proposed framework and the proposed MPC protocols followed by the framework security and complexity analysis. Then we show that compressive sensing domain data-mining (e.g., k-means clustering) is feasible based on our experimental results.

Chapter 5 presents the integrated privacy preserving storage and secure watermark detection framework. We present our framework as well as the security and complexity analysis of the framework. The superiority of our framework to previous secure watermark detection methods is shown quantitatively. We show that compressive sensing domain secure watermark detection is viable based on our analysis and experimental results.

Chapter 6 discusses a use case: a mobile privacy preserving collaborative filtering system by taking advantage of our mobile geo-tagging system and our generic privacy preserving data-mining framework. Our experimental results with real-world data demonstrate that our system is feasible.

Finally, Chapter 7 presents a summary of the contributions and gives several suggestions for the future work.

Chapter 2. Background and General Knowledge

In this chapter, we introduce some backgrounds and general knowledge referenced in this dissertation. Section 2.1 introduces the geometry of image formation and the camera parameters (i.e., Pin-hole camera model and Multi-view geometry), which determine the relationship between a point in the image plane and a scene point. They will be applied in our mobile geo-tagging system in Chapter 3. In Section 2.2, several public key encryption systems and their homomorphism property (i.e., additive and multiplicative homomorphism) are introduced. They are referenced by our privacy-preserving data-mining frameworks and systems from Chapter 4 to Chapter 6.

2.1. Imaging Geometry

2.1.1. Pin-hole Camera Model

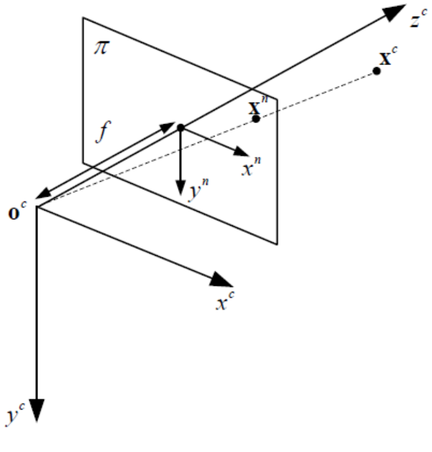


Figure 2.1. The geometry of the pin-hole camera model: π is the image plane; f is focal length; O^C is the camera coordinate origin.

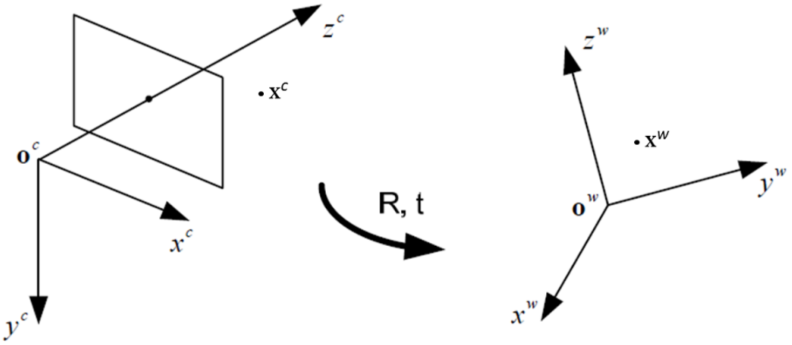


Figure 2.2. Transformation between the world coordinate system and the camera coordinate system (Left: camera coordinate; Right: world coordinate).

The most common and simplest geometric model of a camera is the perspective (pin-hole) model [18] as shown in Figure 2.1. The line through the center of projection (optical center) O^C and perpendicular to the image plane π is the optical axis. The intersection point between π and the optical axis is the principal point. In an actual camera, the image plane π is at a distance of f behind the center of projection and the projection image is inverted. However, in pin-hole camera model, we assume the image plane is in front of the center of projection, so that we can avoid the inversion. The relationship between a world coordinate homogenous representation $\mathrm{p^W([x^W, y^W, z^W, 1]^T)}$ (X^W in Figure 2.2) and its corresponding camera coordinate homogenous representation $\mathrm{p^C([x^C, y^C, z^C, 1]^T)}$ (X^C in Figure 2.2) can be expressed by a transformation which contains the 3×3 rotationmatrix $\mathrm{R = [r_1\, r_2\, r_3]}$ and the 3×1 translation vector v [18]:

$$\mathrm{p^C = [R \quad v]\, p^W}$$

where $[\mathrm{R} \quad \mathrm{v}]$ is called the extrinsic parameters. Then the image coordinate homogenous representation p^{P} (X^n in Figure 2.1) and its corresponding world coordinate homogenous representation can be related by [18]:

$$\mathrm{p}^{\mathrm{P}} = [\mathrm{x}^{\mathrm{p}}\ \mathrm{y}^{\mathrm{p}}\ 1]^{\mathrm{T}} = \mathrm{s}\ \mathrm{K}[\mathrm{r}_1\ \ \mathrm{r}_2\ \ \mathrm{r}_3\ \ \mathrm{v}][\mathrm{x}^{\mathrm{w}}\ \mathrm{y}^{\mathrm{w}}\ \mathrm{z}^{\mathrm{w}}\ 1]^{\mathrm{T}} \quad (2.1)$$

$$\text{where } \mathrm{K} = \begin{bmatrix} \mathrm{f}_{\mathrm{x}} & \gamma & \mathrm{u}_0 \\ 0 & \mathrm{f}_{\mathrm{y}} & \mathrm{v}_0 \\ 0 & 0 & 1 \end{bmatrix}$$

K is called the intrinsic matrix, where f_{x} and f_{y} represent the focal length (in pixels) of the camera in the x and y direction respectively, and γ is called the skew factor and is typically very close to zero. u_0 and v_0 are the coordinates of the principal point [18]. s is a scale factor. In the calibration process [11][18], user takes several checker board images from different perspectives, then the standard calibration algorithms and tools [12][20] can be used to calculate the camera's intrinsic matrix K. The pin-hole camera model will be later referenced in our single-image based localization system in Section 3.3.

2.1.2. Multi-View Geometry

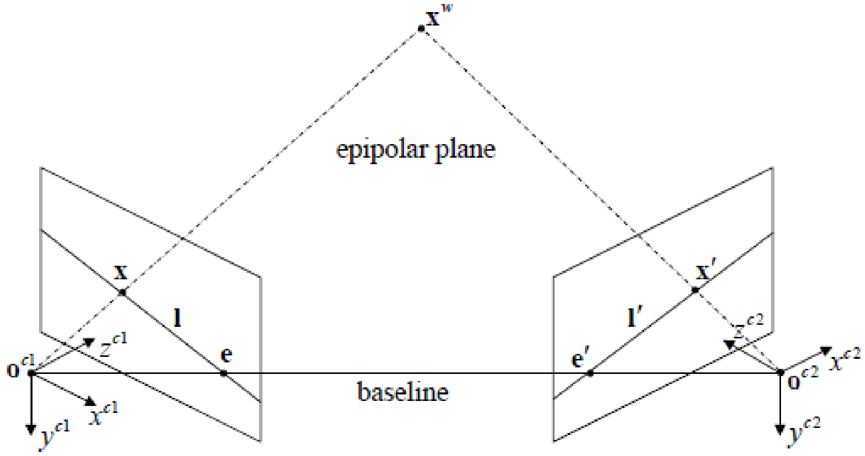


Figure 2.3. Epipolar Geometry

Figure 2.3 shows the Epipolar geometry. Consider two planes with distinct viewpoints, let x = (x, y, 1) and x' = (x', y', 1) be the corresponding points in two image planes, i.e., the projection points of x^w. The epipolar geometry defines the geometric relationship between these corresponding points. The plane passing through the optical center O^{c1} and O^{c2} and the scene point x^w is called an epipolar plane. The projection e (e') of one camera center onto the image plane of the other camera frame is called an epipole. An epipolar line l (l') is the intersection of an epipolar plane for x^w with the image plane. All epipolar lines pass through the epipole.

The fundamental matrix represents the epipolar geometry and contains most of the information about the relative position and orientation between the two views. For all corresponding points x (p_L) and x' (p_R) in two image planes, the fundamental matrix F is the unique 3x3 rank 2 homogenous matrix which satisfies:

$$p_R^T F \, p_L = 0$$

The 8-point algorithm [15] is the simplest method to compute the fundamental matrix. Fundamental matrix will be further utilized to calculate the essential matrix, which has been referenced by our two-image based localization system in Section 3.4.

2.2. Homomorphic Encryption Systems

Homomorphic encryption is a form of encryption which allows certain computations to be performed on ciphertext and generate an encrypted value, and then the decryption result of the generated encrypted value matches certain computation result on the plaintext. There are several public key systems that can perform homomorphic encryption, e.g., Unpadded RSA, ElGamal, Goldwasser-Micali, Paillier and etc. In this section, we introduce ElGamal and Paillier public key systems and their homomorphic properties.

2.2.1. Public Key Encryption Systems

Paillier Cryptosystem [72]:

-Key generation:

Let $N = ps$, where p and s are two large primes. Choose $g \in \mathbb{Z}_{N^2}^*$ (integers less than N^2 but bigger than zero) such that the order of g is divisible by N. Any such g is the form of $g \equiv (1+N)^a b^N mod\ N^2$, for a pair (a, b), where $g \in \mathbb{Z}_N$ and $g \in \mathbb{Z}_N^*$. Let $\lambda = lcm(p - 1, s - 1)$ (lcm means least common multiple). Let $\mu = (L(g^\lambda\ mod\ N^2))^{-1} mod\ N$, where $L(u) = \frac{u-1}{N}$ Then the public key is (g, N) and the private key is (λ, μ).

-Encryption ($E_{paillier}()$):

1. Let m be a message to be encrypted where $m \in \mathbb{Z}_N$
2. Select random r where $r \in \mathbb{Z}_N^*$
3. Compute ciphertext as $c = g^m \cdot r^N mod\ N^2$

-Decryption ($D_{paillier}()$)

1. Ciphertext $c \in \mathbb{Z}_{N^2}^*$
2. Compute $m = L\left(c^\lambda\ mod\ N^2\right) \cdot \mu\ mod\ N$
3.

ElGamal Cryptosystem [71]:

-Key Generation:

Let p be a prime, and g be a generator of $\mathbb{Z}_p$. The private key x is an integer between 1 and $p-2$. Let $y = g^x \, mod \, p$. The public key for ElGamal encryption is the triplet (p, g, y).

-Encryption ($E_{ElGamal}()$):

1. Let m be a message to be encrypted where $m \in \mathbb{Z}_p$
2. Then let $a = g^k \, mod \, p$ and $b = m \, y^k \, mod \, p$, where k is a random integer.
3. (a, b) is the ciphertext.

-Decryption ($D_{ElGamal}()$):

1. Let s = a^x, where x is the private key.
2. Then compute $m = b \cdot s^{-1} \, mod \, p$.

2.2.2. Public Key Homomorphism

Homomorphism Definition:

Given two algebra systems A and B, • and ◦ are the operations in A, B, respectively. If ∀ x, y ∈ A, we have f(x◦y) = f(x) •f(y), then the mapping f: A → B is called A to B’s homomorphism.

The Paillier public key system has the following homomorphic properties:

Additive homomorphism:

$D_{Paillier}(E_{Paillier}(m_1, r_1) \cdot E_{Paillier}(m_2, r_2) \ mod \ N^2) = m_1 + m_2 \ mod \ N$ (a)

Multiplicative homomorphism:

$D_{Paillier}(E_{Paillier}(m_1, r_1)^{m_2} \ mod \ N^2) = m_1 * m_2 \ mod \ N$ (b)

The ElGamal public key system has the following homomorphic properties:

Multiplicative Homomorphism:

$D_{ElGamal}(E_{ElGamal}\ (m_1) \cdot E_{ElGamal}\ (m_2)) \ = m_1 * m_2 \ mod \ p$

The homomorphism of the Paillier and the ElGamal public key systems will be utilized by the MPC protocols in Chapter 4.

2.2.3. Handling Real Values through Scaling

The public key systems introduced in this chapter only take positive integers as input, while practical data might involve real-number values such as in the frameworks and systems presented in Chapter 4 through Chapter 6. We scale the floating point values into integer values with certain scaling factor. Negative integers are represented by the upper half of the range [0, N-1] (N is the modulo) in a modulo field, e.g., -1 is represented as N-1, as suggested in [90]. We show how to handle negative values in the Paillier homomorphic properties (a) and (b) in Section 2.2.2. Given two messages m_1 and m_2 to encrypt, and without loss of generality, assume m_1 is a positive number and m_2 is negative, where $|m_1|, |m_2| \in \mathbb{Z}_{N/2}$. Let $N + m_2$ represent m_2, we have:

$D_{Paillier}(E_{Paillier}(m_1, r_1) \cdot E_{Paillier}(N + m_2, r_2)\ mod\ N^2) = N + m_1 + m_2\ mod\ N\ = \tau$, then:

$$m_1 + m_2 = \begin{cases} \tau - N, & \tau > N/2 \\ m_1 + m_2, & \tau \leq N/2 \end{cases}$$

$D_{Paillier}(E_{Paillier}(m_1, r_1)^{N+m_2}\ mod\ N^2) = m_1 * (N + m_2) mod\ N = \tau$, then:

$$m_1 * m_2 = \begin{cases} \tau - N, & \tau > N/2 \\ m_1 * m_2, & \tau \leq N/2 \end{cases}$$

Chapter 3. A Mobile Geo-tagging System

As summarized in Chapter 1, one type of the mobile cloud systems is formed by several connected mobile devices. In such mobile cloud system, one of the challenges is that the algorithms deployed in the computationally weak mobile device require higher efficiency. In this chapter, we introduce a mobile geo-tagging system which uses one or more connected mobile phones to locate and track a remote target (still or moving) utilizing smartphone's sensors, e.g., GPS, digital compass and camera sensor. To cope with the smartphones' computational limitation, we design light-weight image/video processing algorithms to achieve a good balance between estimation accuracy and computational complexity.

3.1. Introduction

Smartphone is becoming a powerful computing, sensing and communication platform. Due to its ubiquity, low-cost, all-in-one sensors and convenient programming environment, commodity smartphone based system has attracted increasing attention and is becoming an active research field. In particular, the high resolution smartphone cameras which can take both images and videos have created abundant opportunities for mobile multimedia research and led to many interesting works in the literature. For example, Hile *et al.* [28] developed a system on the smartphone that automatically generates landmark-based pedestrian navigation instructions. Yu *et al.* [26] developed an image based user localization system on smartphone for use when the phone's GPS information is lost in big cities. A mobile vision system has also been developed to detect pedestrian lights to help pedestrians with visual impairment cross roads [27]. Smartphones' powerful CPU, camera, accelerometer, GPS, and digital compass have also made them a very promising platform for augmented reality [29][30]. We propose and develop an efficient image and video based real-world remote target localization and tracking system using the Android smartphone, by leveraging its built-in sensors such as camera, digital compass, GPS, etc. The system includes single-image based remote target localization, two-image based remote target localization and video-based remote target tracking, as shown in Figure 3.1.

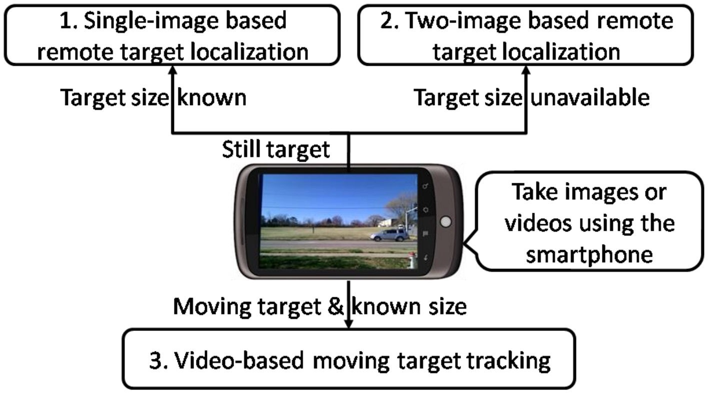

1. Single-image based remote target localization
2. Two-image based remote target localization
Target size known
Target size unavailable
Still target
Take images or videos using the smartphone
Moving target & known size
3. Video-based moving target tracking

Figure 3.1. The proposed smartphone based system has three major components: 1. Single-image based remote target localization; 2. Two-image based remote target localization; 3. Video based remote moving object tracking.

Image based localization algorithms and stereo vision systems have been studied by researchers for decades. Video object tracking is also an active research area for years. However, the smartphone is an emerging computing platform and has computational limitations. New designs and methods of image/video based remote target localization and tracking for use on mobile smartphones are yet to be developed, considering the following unique characteristics and requirements of smartphones:

a) Light-weight computing: due to the still limited computation capability of smartphones, sophisticated algorithms with high complexity and memory requirement may not fit.

b) Trade-off between accuracy and complexity: achieving a good balance between the accuracy and the complexity is a critical consideration on the smartphone platform. For example, to estimate a remote target's trajectory based on the video, an accurate tracking with the object boundary appropriately identified is very important, while detecting the detailed contour of the object may not be necessary.

c) Exclusive interactive user interface: smartphone provides a friendly user interface that can be leveraged to input parameters and improve the accuracy and speed. Explicit and implicit information from user's interaction is a unique feature on smartphones.

d) Memory I/O limitation: the smartphone's computing platform is different from PC, which leads to difference in computational complexity distribution. For example, our experiments show that large memory space allocation is extremely time-consuming on the smartphone.

In this chapter, we not only develop the image/video based remote target localization and tracking system on the smartphone, but also propose optimization methods that are tailored to the unique characteristics of the smartphone's computing platform. The contributions of this chapter can be summarized as below:

1. The proposed smartphone based remote target localization and tracking system is first of its kind. We make the initial efforts developing the system on the smartphone and provide first hand real-world experimental results of the system. We demonstrate that accurate image/video based remote target localization and tracking on the smartphone is feasible and practicable.
2. All the tasks are performed on the smartphone, no remote server is required. We have developed algorithms and optimization methods tailored to the smartphone computing platform by considering its unique characteristics.

The rest of this chapter is organized as follow. Section 3.2 describes the related work. Section 3.3, Section 3.4, and Section 3.5 introduce our proposed system in detail including the algorithms. Section 3.6 presents the system performance, and Section 3.7 proposes and evaluates some complexity optimization methods. Section 3.8 concludes the Chapter and discusses the future work.

3.2. Related Works

We describe the related work in image based localization and video based object tracking, and highlight how our work differs in this section.

3.2.1. Image based Localization Systems

Research on image and vision based localization has been extensively conducted in the context of robot vision and the stereo vision systems, e.g., Zhang and Kosecka [22] studied image based localization in the urban environments and [23] explored image based localization in the indoor environments. Smartphones have also been used for both indoor and outdoor localization purposes [24][25][26], based on the digital compasses, accelerometers and image sensors. However, these localization systems are all designed for locating the device itself, but not the remote target. Furthermore, the smartphone vision based localization systems, such as [24][26], have to interact with a server and leverage the server's resource. In this chapter, *we develop a system to locate a remote target and all the computations are performed on the smartphone without a remote server*.

Typically, there are two technical approaches for image based localization. The first uses image global properties for correlation and the second explores the correlation based on a set of local features. Local-feature based approaches are often considered to be more robust [21]. In our work, we will use SURF [17] local features for feature matching. However, in general, feature extraction and matching are considered computationally complex even on a PC platform. In this chapter, *we propose an optimized feature extraction method designed for the smartphone computing platform taking into account its memory I/O and CPU limitations*.

3.2.2. Video based Object Tracking Systems

Video-based object tracking is an active research field and many works have been proposed in the literature. For example, Weng *et al.* [31] proposed a video tracking algorithm using adaptive Kalman filter based on the object's dominant color assuming the background is static. However in our application scenario, such static background assumption does not hold anymore since users' hands may be shaking while shooting a video. Roh *et al.* [32] proposed a tracking algorithm that detects the detailed contour of the moving target, which is not suitable for the mobile scenario due to its high complexity. Rosales and Sclaroff [33] developed a 3D trajectory tracking system with a fixed uncalibrated camera. However it estimates the 3D trajectory up to a scale factor to help more accurate 2D tracking on the image plane, not the real world 3D trajectory. Xu *et al.* [34] also proposed a 3D-trajectory estimation system based on video sequences. However, its computationally expensive SIFT-based feature matching component is not suitable for video based object tracking on the smartphone platform. The existing video based object tracking algorithms cannot be adopted for our problem for one or more of the following reasons:

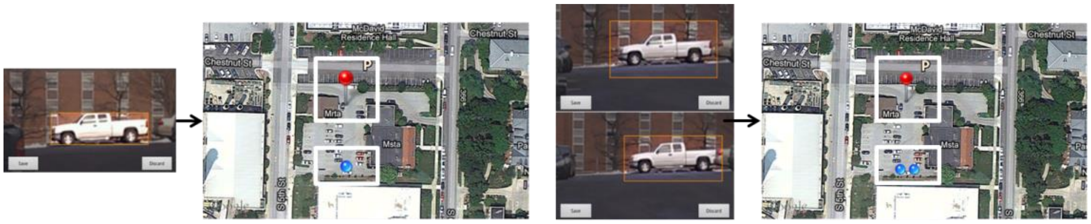
1. Static background assumption: such assumption does not hold for videos taken by the smartphones.
2. Roughly locating the moving object on the image only: such tracking results cannot be used to estimate the object's remote location in the physical world.
3. Detecting object's detailed contour: the detailed contour detection introduces high complexity and is unnecessary and not suitable for our system.

4. Not originally designed for the smartphone platform: Smartphone's computational limitation denies many existing high complexity algorithms; User interaction is another exclusive advantage of the smartphone that has not been considered before.

In this chapter, we develop a light-weight video based remote target tracking system that can accurately track the moving objects in the video on the smartphone, based on which we can estimate the remote objects' moving trajectory in the physical world.

In addition, the extended Kalman Filter (EKF) has proven to be very useful in the recovery of rigid motion and structure from image sequences [33]. Meanwhile, applying EKF to remote target tracking such as vehicle tracking based on some non-image sensor readings has also been widely used [35]. In our work, we adopt the EKF model to track a remote target's trajectory and estimate its velocity based on the observations from the video.

Figure 3.1 shows the components of the proposed smartphone based remote target localization and tracking system. Before using the system, the smartphone camera needs to be calibrated [11]. In order to position or track the remote object, the system collects the compass reading and the GPS coordinates of the phones while taking images/videos of the remote object. Under the circumstances where no GPS coordinates and/or digital compass readings are available on the phone, distance and relative trajectory with respect to the phone can still be estimated. In the following, we present the three major components and the basic algorithms of our system: the single image based localization system, the two-image based localization system and the video based object tracking system, respectively.



(a). Single-image based localization

(b). Two-image based localization

Figure 3.2. (a) Single-image based remote target localization: user takes a picture of a remote target and draws a tight bounding box around it; (b) Two-image based remote target localization: user inputs a rough bounding box on the left and right image enclosing the target. (The camera and the remote target positions are displayed on Google Maps. Blue dots: camera position; Red pin: estimated position of the remote target).

3.3. Single Image-based Remote Target Localization

When estimating the position/distance based on a single image, user is expected to know the object's physical height and width. For example, if the target object is a car and its model can be automatically detected using computer vision techniques, its precise size can be retrieved from the web. The user inputs a tight bounding box around the object in the image (note this condition will be relaxed for moving target objects when we have a video, as discussed in Section 3.5), and then the system will estimate the remote target's relative position to the smartphone based on the correspondences between the object's physical size and its pixel size on the image plane. Putting together the GPS and the compass reading, we can estimate the GPS coordinate of the object, as illustrated in Figure 3.2 (a).

3.3.1. Algorithm

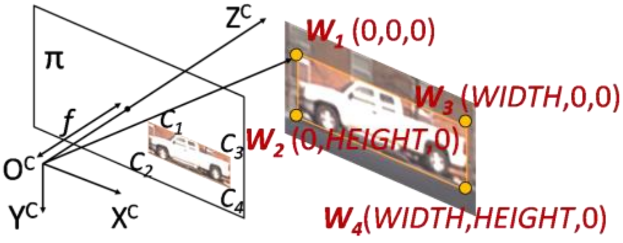


Figure 3.3. The 4-point algorithm: WIDTH and HEIGHT are the object's physical size; W_i are four 3D points and Ci are their corresponding points on the image plane.

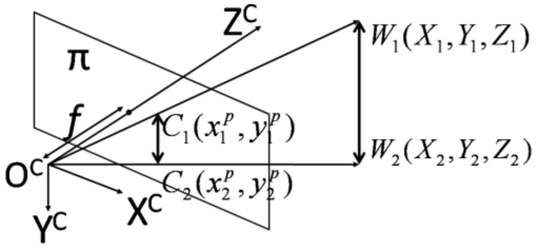


Figure 3.4. The 2-point algorithm

The algorithm's assumption is that the remote target's physical size is known. As shown in Eq.(2.1), we can calculate the relative pose $[r_1 \ r_2 \ r_3 \ v]$ to estimate the remote target's relative position. Note the translation vector v is the relative position of the remote target to the camera. The rotation $[r_1 \ r_2 \ r_3]$ can be described by three angles [12] and the translation v is a 3×1 vector, hence the relative pose includes six unknowns. We need to have at least six equations with the relative pose as the only unknown in order to estimate the relative pose.

Because of the smartphone's exclusive interactive user interface, a simple user's input can be used as the system input parameter to create the equations that we need. We assume the 3-D object is a planar object. This assumption holds in many cases since the target is far away and approximately only one side of the 3-D object appears in the image. The four corners of the bounding rectangular box drawn by the user around the object correspond to four 3-D points (W_i) of the world coordinate system as shown in

Figure 3.3, which demonstrates our 4-points algorithm. The coordinate values of the four 3-D points are also shown in Figure 3.3. Since we assume W_i's are co-planar, their z axis values in the world coordinate system are all zero, we have:

$$[x^p \; y^p \; 1]^T = \; s \, K \, [r_1 \;\; r_2 \;\; r_3 \;\; v][x^w \; y^w \; 0 \;\; 1]^T \quad (3.1)$$

Each of Eq. (3.1) gives two equations, so there are a total of 8 equations. Then the six unknowns could be calculated by solving a linear system.

In fact, in many practical cases, when the user takes image of an object on a level surface, the object's physical height corresponds to the height of the tight bounding box on the image when the camera is held upright. In these cases, the physical height information alone is sufficient for object position estimation. We can assume that the vertical line W_1W_2 and C_1C_2 of Figure 3.3 are in parallel, as shown in Figure 3.4. With such an assumption, we describe our 2-point algorithm for single-image based localization. In Figure 3.4, $W_1$and W_2's camera coordinates ($[X_1, Y_1, Z_1]^T$, $[X_2, Y_2, Z_2]^T$) and C_1 and C_2's image coordinates ($[x_1^p, y_1^p, 1]^T$, $[x_2^p, y_2^p, 1]^T$) are related by:

$$[x_1^p, y_1^p, 1]^T \cong [Z_1x_1^p, Z_1y_1^p, Z_1]^T = K \, [X_1, Y_1, Z_1]^T \quad (3.2)$$

$$[x_2^p, y_2^p, 1]^T \cong [Z_2x_2^p, Z_2y_2^p, Z_2]^T = K \, [X_2, Y_2, Z_2]^T \quad (3.3)$$

According to the assumption, W_1W_2 is in parallel with C_1C_2 and both of them are in parallel with Y^C-axis too, we have:

$$X_1 = X_2, Z_1 = Z_2 \text{ and } x_1^p = x_2^p$$

Then we can derive:

$$Z_1 = Z_2 = f_y * (Y_1 - Y_2)/(y_1 - y_2)$$

$$X_1 = X_2 = Z_1(x_1 - u_0)/f_x$$

$$Y_1 = Z_1(y_1 - v_0)/f_y$$

$$Y_2 = Z_1(y_2 - v_0)/f_y$$

W_1's camera coordinate ($[X_1, Y_1, Z_1]^T$) is the remote target's relative position to the phone. Typically, the value of Y^C axis is close to zero since the camera and the object are on a level surface.

3.4. Two Image-based Remote Target Localization

In case we do not have the information about the object's physical size, we can still estimate the object's position by taking two images of the remote object from two different positions with two different phones or the same phone. The phone's GPS locations for both images are recorded. If the GPS reading is not accurate enough to estimate the two phones' distance, we can use the single-image based system to estimate the distance between the two phones by treating one phone as the remote target of the other phone. The user can identify the remote target using a bounding box roughly enclosing the target on both images, after which the system will perform two-view triangulation to locate the remote target as shown Figure 3.2(b). If the two images are taken on different phones, we utilize Android phone's Bluetooth or WiFi network capabilities to send the image from one to the other.

3.4.1. Algorithm

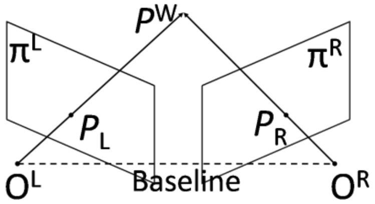


Figure 3.5. Two view triangulation: P^W is a remote point; p_L and p_R are the corresponding points on the two image planes.

We perform triangulation between the two image views as shown in Figure 3.5. As given in Section 2.1.2, epipolar geometry related theories show that if p_L and p_R are two corresponding pixel points between two views, the fundamental matrix F and the two points have the constraint:

$$p_R^T F\, p_L = 0$$

First, we perform SURF [17] feature matching between the two images. We then use the matched features to calculate the fundamental matrix F based on the 8-Point Algorithm [15]. The essential matrix E can be calculated from the fundamental matrix and the two camera's intrinsic matrixes:

$$E = K_R^T F\, K_L$$

The essential matrix E contains the camera relative pose: the rotation R and the translation v:

$$E = [v]_x\, R$$

By decomposing E, one can obtain the rotation matrix R and the translation vector v whose L-2 norm equals one [18]. Actually by decomposing E, we can have four possible R and v combinations. However, with the positive depth constraint, we can identify the camera's true relative pose [18].

After the relative pose is calculated, we then try to identify the corresponding points on the objects as triangulation target points. Based on user's bounding box, we can have two cropped images. We then perform feature matching from the two cropped images. We assume the two objects are planar objects, since they are far away from the camera. We calculate the perspective transformation between the two planes using RANSAC [19] based on the matched features and filter out the outliers which potentially include the matching points in the cropped image but outside of the object. The corresponding points on the objects can be used for triangulation. If we cannot find matching features on the objects, the geometric centers of the two bounding boxes are used as the triangulation target point. The solution to handle noisy triangulation target points proposed by Hartley and Strum [16] is adopted in our system. After v's norm is scaled based on the true distance, we perform Two-View Triangulation [18] to estimate the target's relative position to the phones. Then with the GPS and compass readings of the phone, the remote target's GPS location can be calculated. The diagram of the basic two-image based localization algorithm is illustrated in Figure 3.6.

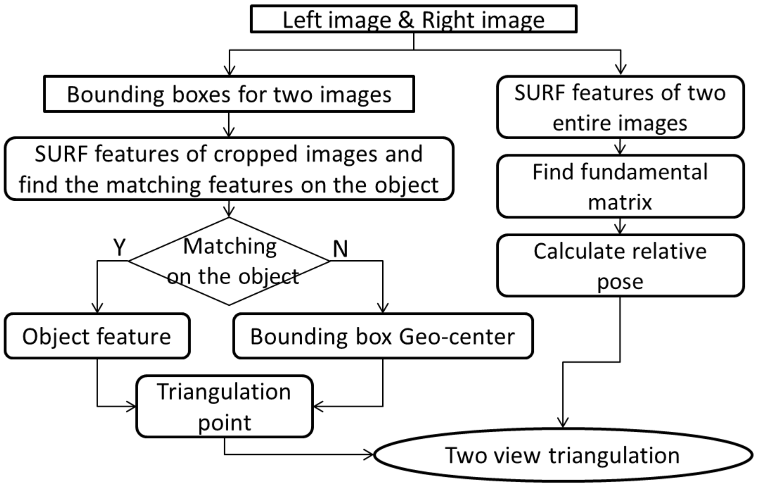


Figure 3.6. Diagram of the two-image based localization algorithm

3.5. Video-based Remote Target Tracking

When using the video tracking system, the user holds the smartphone still while shooting a remote moving object, and simply taps on the target for tracking. Our system then derives tight bounding boxes around the target object in all video frames. Assuming the physical width and height of the remote target is known, we then apply the single-image based localization to project the moving object position on the image plane to the real world coordinates. We then use the Extended Kalman Filtering (EKF) model [13] to estimate a smooth trajectory and the velocity of the moving target. With the smartphone's GPS coordinates and digital compass readings, the moving object's trajectory could be drawn on the map.

3.5.1. Algorithm

An overview of the video tracking algorithm is given in Figure 3.7. An important task is to generate tight bounding boxes around the target for all frames in the video to facilitate subsequent remote target position estimation. In this process, image pixels can be classified into two categories: Background B and Object O, based on different motions of B and O.

Based on the user's touching point in the first frame, we will create a square area centered around the touching point with side size D, as shown in Figure 3.8. The square side size D is pre-determined and large enough to enclose the moving object (in our experiments, the size D is set to half of the height of the video). The operations for the first frame are performed only in the square area.

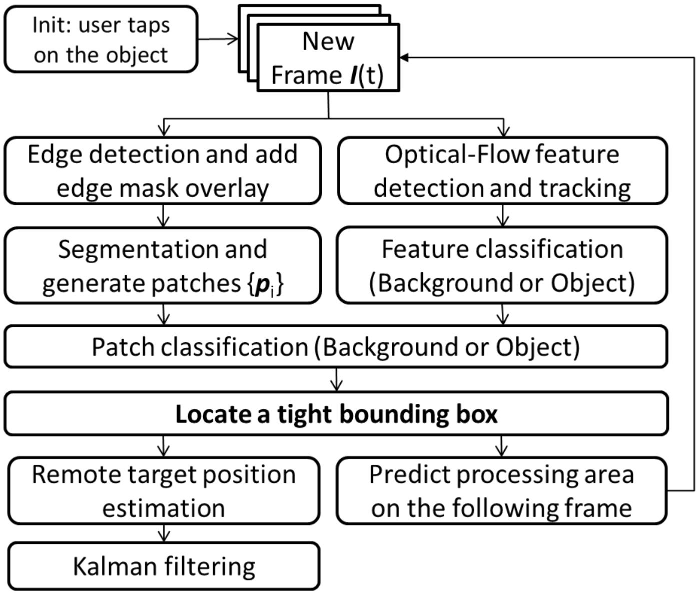

Init: user taps on the object
New Frame $\boldsymbol{I}$(t)
Edge detection and add edge mask overlay
Optical-Flow feature detection and tracking
Segmentation and generate patches {$\boldsymbol{p}_i$}
Feature classification (Background or Object)
Patch classification (Background or Object)
Locate a tight bounding box
Remote target position estimation
Predict processing area on the following frame
Kalman filtering

Figure 3.7. Overview of the video tracking algorithm.

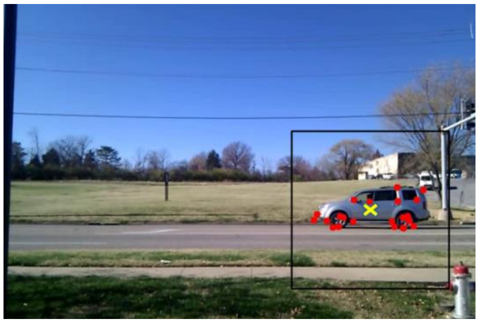


Figure 3.8. Yellow cross: user's tapping point; Black square: centered by user's tapping point with edge size D; Red points: Lucas-Kanade optical flow features of the moving target.

To obtain a tight and accurate bounding box, we need to find the object's canny edges that belong to the object boundary. Canny edges are detected first and added as a mask on top of the cropped image prior to the following segmentation step. Then color similarity based segmentation [12] is performed to get different clusters. After that, connected component analysis is applied to the segmented image to get image patches $\mathcal{P} = \{\mathrm{p_i}\}$, i =

1,…,n. The patches are then categorized into either background patches $\mathcal{P}_B$or object patches $\mathcal{P}_O$, as described below.

Then the patches can be clustered based on their estimated motion vectors with respect to the next frame. However, motion search merely based on minimizing the compensation error over the whole space is very computationally expensive and is not suitable for smartphone applications. Instead, we calculate the motion based on the Lucas-Kanade optical flow features [10]. The optical flow features' motion can serve as references for the motion estimation based on minimizing the compensation error.

It is easy to differentiate between background features and foreground features because their motions are very different. Red points in Figure 3.8 demonstrate the moving object's optical flow features positions. Both of the background and object optical flow feature motions are represented by a motion probability density function (PDF) and are assumed to follow a Gaussian distribution. Let the object motion at time t be $\vec{V}_O(t)$, then:

$$\vec{V}_O(t) \sim N(\vec{\mu}_O, \Sigma_O), \vec{\mu}_O = \begin{bmatrix} \mu_{O,x} \\ \mu_{O,y} \end{bmatrix}, \Sigma_O = \begin{bmatrix} \sigma_{O,x}^2 & 0 \\ 0 & \sigma_{O,y}^2 \end{bmatrix}$$

Similarly, let the background motion be $\vec{V}_B(t)$, then

$$\vec{V}_B(t) \sim N(\vec{\mu}_B, \Sigma_B), \vec{\mu}_B = \begin{bmatrix} \mu_{B,x} \\ \mu_{B,y} \end{bmatrix}, \Sigma_B = \begin{bmatrix} \sigma_{B,x}^2 & 0 \\ 0 & \sigma_{B,y}^2 \end{bmatrix}$$

The Gaussian parameters $\vec{\mu}_O$, $\vec{\mu}_B$, Σ_O and Σ_B are estimated from the Lucas-Kanade optical flow features. Note that for a Gaussian distribution N(μ, σ), we have $P(\mu - 3\sigma < X \leq \mu + 3\sigma) = 99.7\%$. So the motion's variance derived from the optical flow features will help reduce the image patch's search range.

We will perform motion estimation guided by $\vec{V}_O(t)$ and $\vec{V}_B(t)$ for each patch, in which the searching range is constrained by the variance. For a given p_i, we calculate ε_O and ε_B defined as:

$$\varepsilon_O = \min_{\vec{v}_O}\{E(\vec{v}_O)\}, \vec{v}_O = \begin{pmatrix} v_{O,x} \\ v_{O,y} \end{pmatrix}, v_{O,x} \in [\mu_{O,x} - 3\sigma_{O,x}, \mu_{O,x} + 3\sigma_{O,x}],$$

$$v_{O,y} \in [\mu_{O,y} - 3\sigma_{O,y}, \mu_{O,y} + 3\sigma_{O,y}]$$

$$\varepsilon_B = \min_{\vec{v}_B}\{E(\vec{v}_B)\}, \vec{v}_B = \begin{pmatrix} v_{B,x} \\ v_{B,y} \end{pmatrix}, v_{B,x} \in [\mu_{B,x} - 3\sigma_{B,x}, \mu_{B,x} + 3\sigma_{B,x}],$$

$$v_{B,y} \in [\mu_{B,y} - 3\sigma_{B,y}, \mu_{B,y} + 3\sigma_{B,y}]$$

where $E(\vec{v}) = \frac{\sum_{L \in p_i} |I^{t+1}(L+\vec{v}) - I^t(L)|}{\text{total number of pixels of } p_i}$

In the above definition, L is the position of a pixel from patch p_i and $I^t(L)$ is L's gray scale intensity value at frame t.

ε_O and ε_B are the motion compensated prediction residual errors for the object motion and background motion respectively. L1 norm is used instead of L2 norm to reduce the potential bias introduced by some large intensity difference on motion estimation. Then we have:

If $\varepsilon_O \gg \varepsilon_B$ ($\varepsilon_O > \rho \cdot \varepsilon_B$), then p_i is a Background patch;

Else if $\varepsilon_B \gg \varepsilon_O (\varepsilon_B > \rho \cdot \varepsilon_O)$, then p_i is an Object patch;

Else, then p_i is labeled as unclassified.

where ρ is a threshold value. It should be noted that if p_i belongs to a uniform background close to the object, it might also satisfy the object patch condition, creating a misleading scenario. Therefore another criterion is added for the Object patch to address the uniform background problem:

$\varepsilon_\tau > \rho \cdot \varepsilon_O$, where

$$\varepsilon_\tau = \min_{\vec{v}_\tau}\{E_\tau(\vec{v}_\tau)\}, \vec{v}_\tau = \begin{pmatrix} v_{\tau,x} \\ v_{\tau,y} \end{pmatrix}$$

$$v_{\tau,x} \in \left[\mu_{\tau,x} - 3\sigma_{\tau,x}, \mu_{\tau,x} + 3\sigma_{\tau,x}\right]$$

$$v_{\tau,y} \in \left[\mu_{\tau,y} - 3\sigma_{\tau,y}, \mu_{\tau,y} + 3\sigma_{\tau,y}\right]$$

where $E_\tau(\vec{v}) = \frac{\sum_{L \in p_i} |I^t(L+\vec{v}) - I^t(L)|}{\text{total number of pixels of } p_i}$ and ε_τ is the compensation error under object motion on the same frame. In summary, we have:

Object patches = $\{p_i | \varepsilon_B > \rho \cdot \varepsilon_O , \varepsilon_t > \rho \cdot \varepsilon_O\}$

Background patches = $\{p_i | \varepsilon_O > \rho \cdot \varepsilon_B\}$.

Figure 3.9 shows an example of the image patch classification result. There are some very small unclassified patches shown as grey level areas in the figure. They will be ignored since most of the patches will be classified and classified patches are sufficient for locating the tight bounding box. To obtain the tight bounding box, all we need is to detect the canny edges belonging to the object. The K-Nearest Neighbor algorithm is used. For each canny edge pixel p_{canny}, find its neighborhood pixels whose distance to p_{canny} is less than R (R specifies the neighborhood range). If the following condition holds:

$$\text{Num}(\mathcal{O}) > \theta \cdot \text{Num}(\mathcal{B})$$

where $\text{Num}(*)$ denotes the number of pixels belonging to *, θ is a threshold value, then p_{canny} will be treated as a canny edge pixel belonging to the object. Then a tight bounding box could be drawn whose sides pass through the most outside object edge pixels, as shown in Figure 3.10. Given the optical flow feature speed of the object, the position and size of the tight bounding box on the current frame, a predicted bounding box for the next frame in which the object is enclosed with a high probability can be generated. Then the same operations for the current frame will be applied to the next frame. The same process will be repeated for the consecutive frames as represented by the loop shown in Figure 3.7. After the tight bounding boxes are obtained for every frame, we can estimate the remote moving target's positions for each frame based on the single image based algorithm. Currently, our system only tracks a moving object on the image plane without considering the scenarios with occlusion. Algorithms tracking multiple moving objects with occlusion have been proposed in many works such as [31][33], which will be included into our system in the future work.

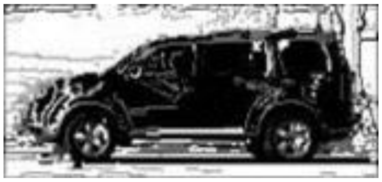


Figure 3.9. Black areas are Object patches with high confidence; White areas are Background patches with high confidence. Grey level areas are unclassified patches.

3.5.2. Kalman Filtering

The remote target position estimation will have errors introduced potentially by camera shaking, tight bounding box estimation error, as well as camera intrinsic parameter estimation errors. To mitigate these potential errors, Extended Kalman Filtering (EKF) [13] is applied. Since in most application scenarios, the camera is level with the remote target, the YC axis value in the coordinate system OC as shown in Figure 3.3 is almost zero. We therefore only introduce the XC and ZC axis values into our EKF model.

The state vector is defined as:

$$\vec{s} = [\mathrm{pos}_x \quad \mathrm{pos}_z \quad \mathrm{vel}_x \quad \mathrm{vel}_z]^T$$

where (pos_x, pos_z) represents the object position and (vel_x, vel_z) represents the object instant velocity. The dynamic equation relates the target states $\vec{s}$ at time t+1 and t:

$$\vec{s}_{t+1} = F\vec{s}_t + A\vec{a}_{t+1}$$

$$\text{where } F = \begin{bmatrix} 1 & 0 & \Delta T & 0 \\ 0 & 1 & 0 & \Delta T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, A = \begin{bmatrix} 0.5\Delta T^2 & 0 \\ 0 & 0.5\Delta T^2 \\ \Delta T & 0 \\ 0 & \Delta T \end{bmatrix}$$

and ΔT is the time between two consecutive frames(i.e., 1/25 sec if the video frame rate is 25fps). $\vec{a}_{t+1} \sim N(\vec{0}, \sigma_v^2 I)$ represents the acceleration of the target as a statistical perturbation. The process noise level σ_v^2 controls the tradeoff between tracking convergence for a constant velocity target and the flexibility to track a maneuvering (e.g., accelerating) target. The observation matrix is:

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Two major steps in each iteration of EKF are the prediction step and the correction step:

Prediction:

$$\hat{\vec{s}}_{t+1} = F\,\vec{s}_t; \;\; \hat{\Sigma}_{t+1} = F\Sigma_t F^T + \sigma_v^2 AA^T$$

Correction:

$$K_{t+1} = \hat{\Sigma}_{t+1} C^T \left(C\hat{\Sigma}_{t+1} C^T + Q\right)^{-1};$$

$$\vec{s}_{t+1} = \hat{\vec{s}}_{t+1} + K_{t+1}\left(Z_t - C\hat{\vec{s}}_{t+1}\right);$$

$$\Sigma_{t+1} = (I - K_{t+1}C)\hat{\Sigma}_{t+1}$$

where Z_t is the observation obtained from the image based position estimation,Q is the measurement error and is based on our distance estimation accuracy presented in Section 3.6.

3.6. Experimental Results

Google Nexus One with 1GHz processor and 512 MB memory is chosen as our test phone. Our image/video based localization and tracking system is implemented using JAVA and C++ native package on the smartphone. For image processing purposes we integrate OpenCV [20] into our smartphone application. Different from an in-door, well set up stereo vision system, our system's accuracy may be affected by various factors, e.g., blurred image caused by human hand shaking for both single-image and two-image based systems; bounding box input by user that fails to enclose the remote target tightly for the single-image based system; only a few SURF features extracted due to the less textured background views, and inaccurate relative pose caused by larger distance between two phones for the two-image based system. In the real world applications, all of these heterogeneous factors for accuracy make the error analysis very difficult. In our experiments, users try to take images steadily without blurriness visually (in the cases a blurred image is taken, users re-take another image for further testing). In the experiments of the single-image based localization, we assume the user can locate a tight bounding box around the target with our handy user interface. Our experiments for two-image based localization are conducted given the follow settings: the distances between two phones are 5-7 yards (remote targets are more than 30 yards away); the two phones are about the same height; some background scenes are shown in Figure 3.14. The single-image based localization uses the image resolution 2592×1944 and the two-image based localization uses the resolution 1600×1200. The average accuracy of our image based localization system for the single image based and two-image based cases are listed in Table 3.1 and Table 3.2 respectively. The accuracy is presented for the estimated target distance to the phone for the single image based localization and the target distance to the left phone for the two-image based localization. The ground truth distances are obtained from a Laser Range Finder. It can be observed from Table 3.1 and Table 3.2 that the single image based approach can achieve an average accuracy of 94% with the 4-point algorithm and slightly worse with the 2-point algorithm. The two-image based localization can achieve an average accuracy of 83%. The performance of the two-image based algorithm looks slightly worse for a couple of reasons: first the distance between the two cameras is an estimated value based on the single image based method, so the error will propagate during the two-image based localization process. Second, the camera relative pose estimation is constrained by the feature detection and matching accuracy. Third, if there is no identified feature matching on the object, we have to rely on the bounding box's geometric center as the triangulation point which might reduce the accuracy. In addition, the highest resolution image (2592×1944) is not chosen due to the smartphone's computing power limitation. Note that we can achieve an average accuracy of 89% on the two-image based approach with image size (2592×1944) on the PC platform.

Table 3.1. Single-image based localization accuracy (Unit: Yards)

Ground Truth	11	14	21	32	41	80
4-point	12.26	15.71	22.10	33.68	41.94	76
2-point	12.10	16.21	22.34	34.76	43.22	77

Table 3.2. Two-image based localization accuracy (Unit: Yards)

Ground Truth	35	40	49	50	96
Estimated	30	34	45	42	111

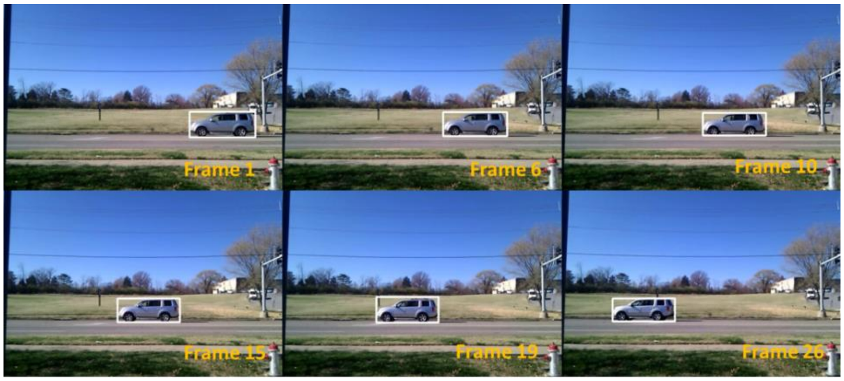


Figure 3.10. Target tracking results on the video

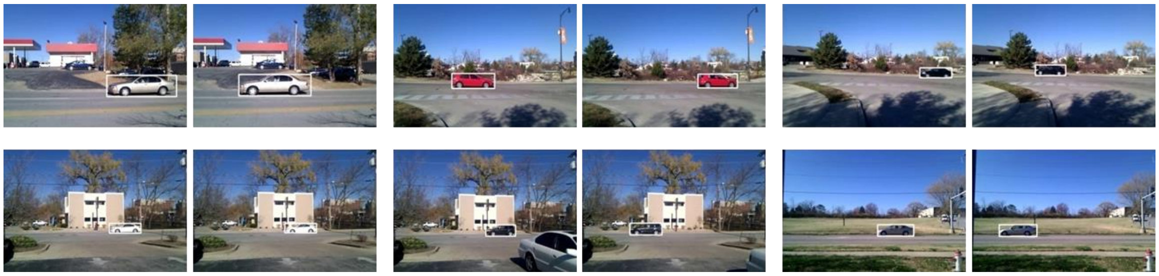


Figure 3.11. Video tracking results with different backgrounds and remote targets

Figure 3.10 shows the video based target tracking results with the example given in Figure 3.8 (video resolution: 720x480). More real-world video tracking results with different backgrounds and remote targets are shown in Figure 3.11, which shows our video tracking algorithm is robust. In our experiments, we set the video tracking algorithm parameters ρ as 1.6, θ as 0.3 and R as 3 empirically. It shows that the tight bounding boxes enclosing the moving vehicle could be located accurately. We have more video tracking results with similar performance. When the tight bounding boxes are located accurately on the video frames, the single-image based algorithm (4-point and 2-point algorithm) can be used for the remote target localization for each frame, and the accuracy is almost the same as what is presented in Table 3.1 even though the video frame resolution is lower. Figure 3.12 shows the estimated relative trajectory and velocity of the example given in Figure 3.10. With the GPS and the compass sensor reading, we can also plot a moving trajectory on a map. Figure 3.12 (a) shows that the original trajectory estimation based on the 4-point algorithm has some abrupt changes which are introduced by either the bounding box estimation errors or human hands shaking, while our EKF model provides a smooth trajectory. In many real world scenarios, the 3-D remote target's projection on the image plane includes more than one side of the target. So the pixel size of the bounding box does not correspond to the physical size of the target. Then the 4-point algorithm will not fit. However, in most practical cases, when the user shoots a video of a moving target on a level surface, the object's physical height corresponds to the height of the tight bounding box on the image if the camera is held upright. Then we can use our 2-point algorithm to estimate the remote target's position. Figure 3.12 (a) also shows the tracking results with the 2-point algorithm after EKF is applied. We can see that the trajectories based on the two algorithms are very close to each other, but the 4-point algorithm is a little bit more stable than the 2-point algorithm since the 4-point algorithm has two more corresponding point constraints. However, the advantage of using the 2-point algorithm in the video tracking is that it is more practical and can be applied for more general scenarios. Figure 3.12 (b) shows the moving vehicle's velocity. It concurs with our observation that the target was accelerating slowly after waiting for the "green" traffic light. It is worth pointing out that, in our real world tests, we found that holding the camera roughly upright is sufficient and the tracking result is not very sensitive to it.

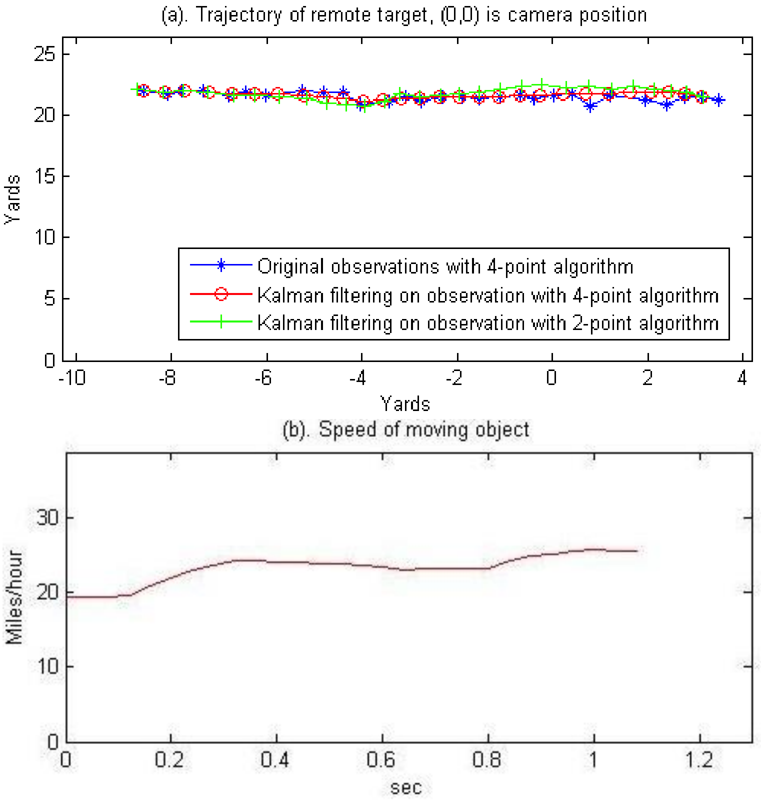


Figure 3.12. (a) Trajectory based on original observations with 4-point algorithm; smoothed trajectory with 4-point and 2-point algorithms after modeling with EKF. Camera position is (0, 0); (b) Estimated velocity of the remote target.

Regarding the complexity of the three components, it takes less than a second for the single-image based localization system for any image resolution, and 30-40 seconds for the two-image based localization with image size 1600×1200. Our video based target tracking system can achieve a position estimation speed of about 3~4 fps (frames per second). In Section 3.7, we propose some optimization methods to reduce the complexities of the two-image based localization and the video based target tracking respectively.

3.7. Speed Optimization

In this section we propose some optimization schemes to improve the speed of the system to make it more practical.

3.7.1. Two-image based Localization System Optimization

3.7.1.1. Block based feature extraction

As shown in the second and the third row of Table 3.3, the running time distributions over different sub-tasks are different between the PC and smartphone platform when feature extraction is performed on the entire image. Feature extraction is extremely computationally heavy on the smartphone. The reason is that SURF builds image pyramids and does filtering for each layer with Gaussian of increasing variance by taking the difference between layers, which involves large memory allocation and de-allocation operations. Our experiment shows that allocating such large memory space is responsible for the majority of the complexity in the feature extraction step on the smartphone. To deal with such an I/O limitation, we do not perform feature extraction on the entire image. Instead, we cut the image into blocks (100×100) and perform feature extraction on individual blocks. The fourth row of Table 3.3 shows that the speed of feature extraction has been improved significantly after performing feature extraction on the 100×100 blocks (50% overall speed improvement). Experiments show that the feature extraction and the final triangulation results based on blocks are almost the same as when processing on the entire image.

Table 3.3. Complexity distribution of two-image based localization on PC and Android smartphones respectively (1: Feature extraction, 2: Feature matching, 3: others.)

Components (sec&percentage)	1	2	3	Total
PC	0.956 (57%)	0.174 (10.4%)	0.547 (32.6%)	1.677 (100%)
Smartphone (entire image)	39.7 (90.3%)	1.99 (4.5%)	2.25 (5.2%)	43.94 (100%)
Smartphone (100×100 block)	18.17 (84.2%)	1.3 (6.0%)	2.1 (9.8%)	21.57 (100%)

3.7.1.2. *Progressive random block selection*

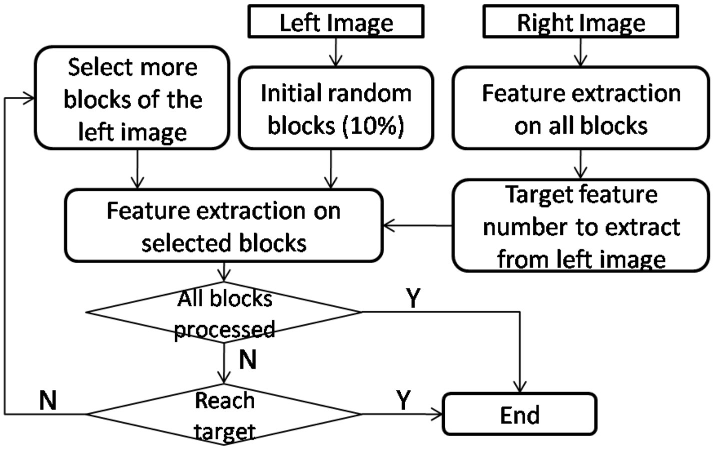
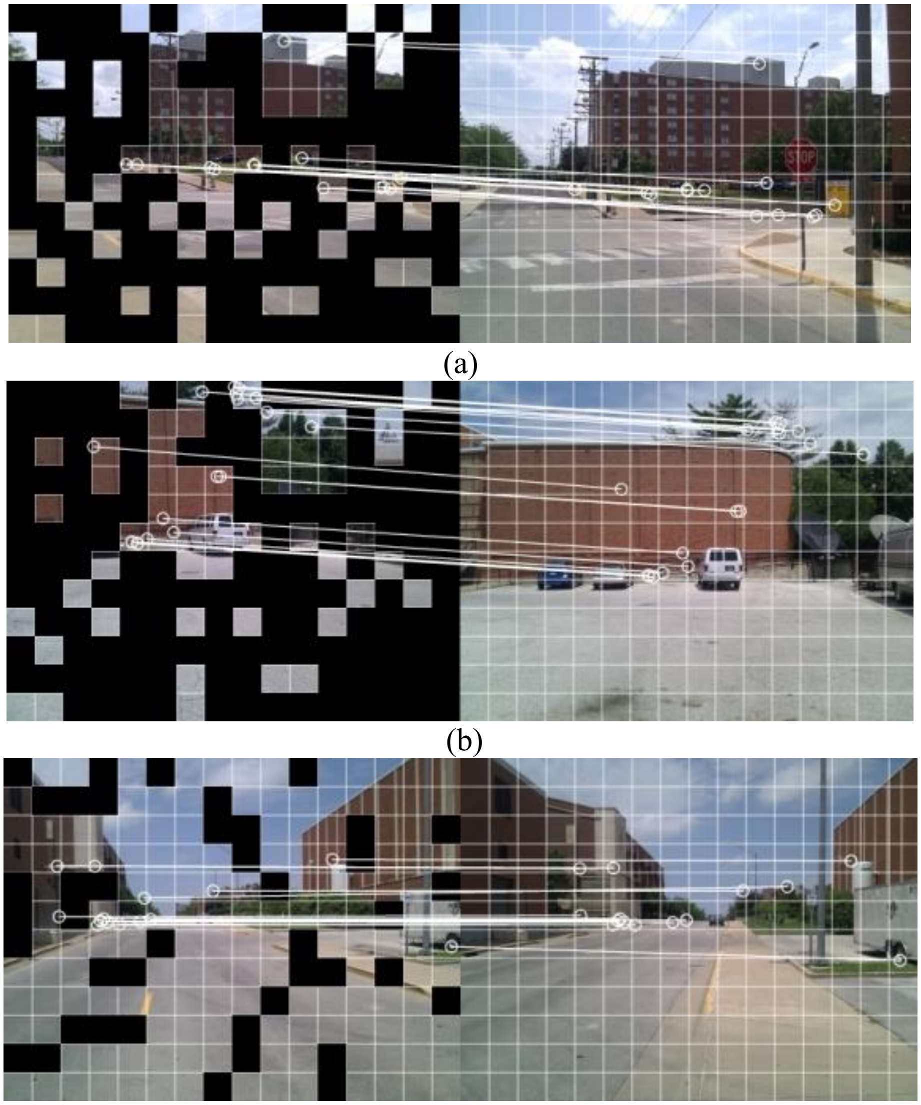

Left Image
Right Image
Select more blocks of the left image
Initial random blocks (10%)
Feature extraction on all blocks
Feature extraction on selected blocks
Target feature number to extract from left image
All blocks processed
Y
N
N
Reach target
Y
End

Figure 3.13. Progressive random block selection for feature extraction.

In our application scenario, the left images and right images are very similar to each other. As long as we can find sufficient amount of features for matching, the relative pose between the two image planes can be calculated. So it is unnecessary to extract features from the blocks covering the entire image. Instead, we only perform feature extraction on a subset of blocks. We propose a progressive random block selection method in this section.

For the sake of robustness, the entire right image (or the left) has to be used for feature extraction. Based on the number of features of the right image, we can expect that the left image also has about the same amount of features since they have the same scene just with different perspectives. The number of features in the right image also tells us the texture of the images. If the right image is a textured image with many feature points (in our experiment we set 6 feature points per 100×100 block as the threshold), the target number of features we will extract from the left image is 40% of the total number of features of the right image; otherwise it will be 70% of the total number of features of the right image. The above threshold numbers are determined empirically.

We progressively select the blocks on the left image for feature extraction until we are able to reach the target number of features for matching. Firstly, we uniformly randomly select 10% of all the blocks and extract SURF features on these blocks. If we have extracted enough feature points, the feature extraction terminates. If not, we continue to process the neighboring blocks of the processed blocks. Typically, features often concentrate in the textured areas of an image and have lower density in the uniform areas (such as the road) as shown in Figure 3.14. Neighboring blocks of the blocks that have more features are given higher priority for feature extraction, in that we can have more features by processing fewer blocks. The algorithm selects the unprocessed blocks for feature extraction iteratively and terminates until the target number of features has been extracted. Figure 3.13 shows the algorithm diagram of our progressive random block selection. Figure 3.14 gives the feature matching results with the progressive block selection method. We can see from Figure 3.14 (a) and (b) that with feature-rich images, processing about 30% of the left image's blocks has given us enough matches for estimating the fundamental matrix (8-point algorithm requires at least 8 correspondences between the two images). But with the image like Figure 3.14 (c) that has more uniform areas without many features, about 70% of the left image's blocks are used for feature extraction.



(a)

(b)

(c)

Figure 3.14. Feature matching results with progressive block selection method. The blackout blocks are not selected for feature extraction. (a) and (b) have textured background, only 30% of the left image are used. (c) has more uniform background and 75% of the left image are used.

Table 3.4. Two image based system optimization

Feature extraction method	Speed (sec)	Relative error
Entire image	38.1	0%
Block based (All the blocks)	18.4	3.9%
Progressive block selection	14.2	6.7%

Based on our tests with 5 sets of real world images, Table 3.4 shows the average complexity and average accuracy of the two-image based localization before and after speed optimization. The final triangulation results are slightly worse with block-based feature extraction. This is because we might have slightly fewer features with block based feature extraction since the features of the original image close to the block edges may be lost. The progressive random block selection does not affect the triangulation results too much, while saving a lot of complexity. Our experiments show that the two-image based localization speed is improved by 60% on average with the proposed optimization method.

3.7.2. Video-based Remote Target Tracking System Optimization

3.7.2.1. Feature tracking area prediction and earlier patch classification

The video based target tracking system has several computing components: optical flow feature tracking, segmentation, motion estimation, connected component analysis, as well as I/O operations. The complexity distribution can be found in Table 3.5. We can see that the four most significant computing components are programming overhead (others), segmentation, optical flow tracking and motion estimation. We have tried several existing fast segmentation algorithms including quickshift [37], superpixel [38] and etc. The color similarity based segmentation [12] which we adopted is the fastest and provides satisfactory results for our system. We improve the speed by reducing the complexity of the optical flow feature tracking and the motion estimation.

When the object is moving in the video frames, its position on the following frames could be predicted by the previous motions. Instead of searching the whole next frame for the optical flow feature tracking, we limit the search area within the predicted box which is indicated in Figure 3.7. So the optical flow feature tracking is optimized by limiting the tracking area of the following frames. Further, in the patch classification process, the patches along the sides of the square box or the predicted bounding box have a high probability of being background and the patches in which motion features can be found have a high probability of being object. For these patches, their class (Background or Object) can be earlier determined so that the motion estimation process for them is bypassed to reduce the computational cost.

Table 3.6 shows the video tracking complexity after the optimization based on user's input. Nearly 50% speed improvement can be obtained for the patch motion estimation component and 70% speed improvements for the optical flow feature tracking. Overall speed improvement is 20%.

Table 3.5. Video based tracking complexity distribution (1: Motion estimation; 2: Segmentation; 3: Connected component analysis; 4: Optical flow feature tracking; 5: Edge detection; 6: others)

Components	1	2	3	4	5	6
Percentage	6.3%	34%	1.3%	19%	0.94%	38%

Table 3.6. Video tracking optimization considering user's input (Motion: motion estimation; Optical: optical flow feature tracking; spf: second per frame)

Video	Motion (spf)	Optical (spf)	Processing rate(fps)
original	0.0158	0.0473	3.98
optimization	0.0078	0.0152	4.8

3.7.2.2. Spatial and temporal video down-sampling

Both spatial and temporal down-sampling are adopted in order to reduce the complexity of video based target tracking. For the temporal down-sampling, all the frames are used for optical feature tracking in order to detect the motion accurately and robustly. But we do not use every frame for remote target position estimation. Instead, we only use one out of every *n* frames, as shown in Figure 3.15. The original video spatial resolution is 720×480, the lowest resolution we use is 540×360 for the spatial down-sampling.

Figure 3.16 gives the tracking results with different spatial and temporal down-sampling parameters compared to the results without down-sampling by using the experiment shown in Figure 3.10, when the 4-point algorithm is used. Figure 3.16 shows that the tracking trajectory with down-sampling is close to the trajectory based on the original video. Table 3.7 summarizes the complexity and the accuracy (accuracy is represented by relative error compared to the tracking results of the original video) of the example of Figure 3.10. We can see from the table that with the spatial resolution of 540×360 and the temporal down-sampling factor of 3. The processing frame rate can be increased to 14 fps with small relative error. The average relative error after down-sampling is about 0.5 yards even when the objects are more than 30 yards away. Based on our tests with 10 real world videos, our video based target tracking can obtain near real-time processing speed (on average 16 fps).

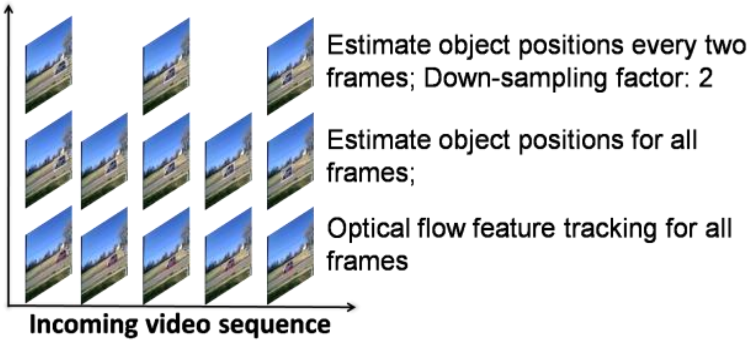

Estimate object positions every two frames; Down-sampling factor: 2
Estimate object positions for all frames;
Optical flow feature tracking for all frames
Incoming video sequence

Figure 3.15. Video temporal down-sampling

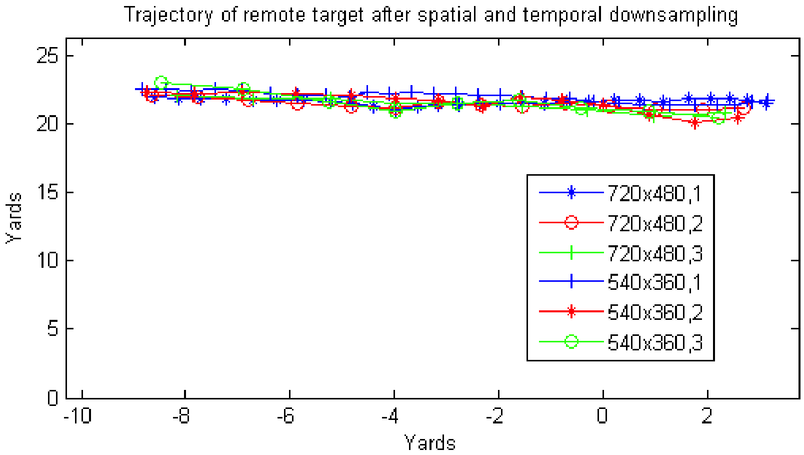


Figure 3.16. Tracking results with temporal and spatial down-sampling ("*X*×*Y*, *n*" denotes spatial down-sampling to resolution *X*×*Y* and temporal down-sampling by a factor of *n*). Camera position is (0, 0).

Table 3.7. Tracking speed and accuracy with video down-sampling

Video	Processing rate (fps)	Relative err(yards)
720×480,1	4.8	0
720×480,2	7.08	0.273
720×480,3	8.29	0.435
540×360,1	8.87	0.54
540×360,2	12.31	0.62
540×360,3	14	0.57

3.8. Conclusion and Future Works

In this chapter, a novel, easy-to-use, image/video-based remote target localization and tracking system on commodity smartphones is presented. We demonstrate that remote target position, trajectory and speed estimation using commodity smartphone is feasible, by utilizing image processing and computer vision techniques combined with the smartphone's sensors. Our system can be used to measure coordinate values of any scene point with respect to the camera, so that it can be extended for applications such as gaming, 3-D measurement, augmented reality applications and etc. It can also combine with image based GPS systems to provide mobile user better location-based services. Furthermore, gyroscope (which accurately measures the phone's self 3D rotation) has now been integrated into smartphones. We will integrate gyroscope into our system, such that users could rotate the cell phone when tracking video of a moving object. Our future work also includes further system accuracy improvement and complexity reduction. We believe our system could be applied in a number of application scenarios such as military operation, law enforcement, and commercial products.

Chapter 4. A Compressive Sensing based Privacy Preserving Data-mining and Signal Processing Framework

As discussed in Chapter 1, the second type of the mobile cloud systems is: mobile devices act as a thin client connecting to a remote cloud, and offloading the data and processing to the cloud, in which privacy is a significant concern. To solve the privacy issue when private data are shared in mobile cloud computing and other collaborative data-mining scenarios, we proposed a generic privacy preserving collaborative data-mining framework in this chapter.

4.1. Introduction

The problem of privacy preserving is becoming more and more important in many joint data-mining applications that deal with health care, e-commerce, security, financial, personal data and other types of sensitive data as the examples given in Chapter 1. In the meantime, the cloud computing technologies are growing, and it is more economical for data holder institutions to shift data storage or data-mining computations to the cloud instead of purchasing hardware and software by themselves. Ideally, the cloud will only store the data or perform data-mining in the encrypted or perturbed domain in order to preserve the data holder's data privacy. We propose a generic privacy preserving data-mining collaborative data-mining framework based on compressive sensing theory and the secure multiparty computation. In our framework, the data are stored and processed in the compressive sensing domain in the cloud simultaneously.

Earlier attempts to address privacy preserving data mining from disparate data sources suffer from one or more of the following limitations: low level of security (e.g., the risk of exposing the common key to individual data holders [39]), high computational and communication complexity, and no/low scalability, as will be discussed in more details in Section 4.7. Our framework not only supports privacy preserving collaborative data-mining, but also offers a great deal of flexibility and scalability including fully utilizing the cloud infrastructure, bringing no additional cost to existing data holders when introducing new data holders, and supporting interpretation of the data mining results in the original data domain.

The rest of this chapter is organized as follows. Section 4.2 introduces preliminary knowledge for the proposed framework including the compressive sensing theory and the secure multiparty computation. In Section 4.3, related works are discussed and how the proposed framework differs is highlighted. In Section 4.4 and Section 4.5, details of the proposed framework, including relevant algorithms and protocols are presented. In Section 4.6, security analysis and complexity analysis are given. Then advantages of the proposed framework are demonstrated in Section 4.7. Section 4.8 presents the experimental results with real-world data. The chapter concludes in Section 4.9.

4.2. Preliminary Knowledge

4.2.1. Compressive Sensing

Restricted Isometry Property (RIP) is a required condition for the perfect reconstruction in the compressive sensing theory. Before presenting the compressive sensing theory, we first introduce the Restricted Isometry Property:

Restricted Isometry Property (RIP): A vector $x \in \mathbb{R}^n$ is k-sparse if $|\{j: |x_j|>0\}| \leq k$. A matrix $\Phi \in \mathbb{R}^{m\times n}$ is said to have the Restricted Isometry Property of order k and level $\delta \in (0,1)$ (equivalently, (k, δ)-RIP) if

$$(1-\delta)\|x\|_2^2 \leq \|\Phi x\|_2^2 \leq (1+\delta)\|x\|_2^2 \quad (4.1)$$

for all k-sparse $x \in \mathbb{R}^n$. The restricted isometry constant δ_k is defined as the smallest value of δ for which the above inequality holds.

The compressive sensing theory [64] asserts that when a signal can be represented by a small number of non-zero coefficients, it can be perfectly recovered after being transformed by a limited number of incoherent, non-adaptive linear measurements. Suppose a signal $s \in \mathbb{R}^n$ is a k-sparse vector (only k out of the n elements of s are nonzero) and can be transformed to $x \in \mathbb{R}^m, m < n$, where $x = \Phi_{m\times n}s$. If $\Phi_{m\times n}$ satisfies RIP, it can be shown [64] that solving the below optimization problem:

$$\min\|s\|_1 \text{ s.t. } x = \Phi_{m\times n}s \quad (4.2)$$

is equivalent to finding the sparsest solution s to $x = \Phi_{m\times n}s$, provided $m \geq Cklog(n/k)$, where C is a small constant. Eq. (4.2) presents a L1 minimization problem which can be solved by orthogonal matching pursuit (OMP) algorithms [68]. It has been shown [67] that there are many ways to construct a matrix $\Phi_{m\times n}$ that meets the RIP property, e.g., if the entries of matrix $\Phi_{m\times n}$ are generated from a Gaussian distribution with zero mean and variance $1/m$, $\Phi_{m\times n}$ is a RIP matrix with overwhelming probability. In our framework, the compressive sensing matrix is generated from such a Gaussian distribution.

4.2.2. Secure Multiparty Computation

Secure multiparty computation (MPC) is a subfield of cryptography. The goal of MPC is to create methods that enable parties to jointly compute a function over their inputs, while at the same time keeping these inputs private. Here we introduce secure scalar product protocol and secure comparison protocol, which will be applied in our framework. Most of the MPC protocols in the literature are developed based on the homomorphic public key encryption systems introduced in Section 2.2.

In MPC, an important security model is ***Semi-honest security model*** [82]: all parties comply with the protocol's procedure strictly, and none of them will actively withdraw midway or incorporate false or malicious data. No two parties will collude to attack a third one. But during the computing process, they may try to keep all the intermediate information, so that they can infer others' input after the process. Semi-honest model is a reasonable assumption for adversaries such as third-party service providers [84].

4.3. Related Works on Privacy-preserving Data-mining

4.3.1. Privacy Preserving Data-mining

Techniques proposed for privacy preserving data-mining can be grouped into six categories [39]: data perturbation, data swapping, k-Anonymity, secure multiparty computation (MPC), distributed data mining, and rule hiding. Data perturbation includes the probability distribution approach and the value distortion approach [65]. Our proposed privacy preserving work is a kind of value distortion approach which adopts multiplicative perturbation using a key-based random compressive sensing transformation.

Random projection [39] refers to the technique of projecting a set of data points from a high-dimensional space to a randomly chosen lower-dimensional subspace. The idea of random projection arises from the Johnson-Linenstrauss Lemma. The lemma says that any set of *s* points in a high dimensional Euclidean space can be embedded (under a Lipschitz map [65]) into a lower dimensional space such that the pair-wise distance of any two points is maintained within an arbitrarily small factor. This nice property implies that it is possible to change the data's original form by reducing its dimensionality but still maintains its statistical characteristics. Random projection is applied as a multiplicative perturbation in [39] for privacy preserving collaborative data-mining. In [39], the random projection matrix is shared between and known by all the data holders, which results in a very vulnerable situation. When the shared key and some encrypted data are known to the adversary (e.g., when some of the data holders are compromised), the system security relies on that an underdetermined linear system does not have a unique solution. However, by exploring the properties of the random projection matrix, an approximation of the original data could be found. In addition, when the input data is restricted to Boolean, the protocol will be at a high disclosure risk. Furthermore, this method does not provide an accurate decryption method to reconstruct the data from the encrypted domain to the original domain. In our proposed framework, we address the significant concern of shared key being exposed to all data holders by introducing an independent trusted key provider and performing compressive sensing key based encryption/decryption using an efficient secure multiparty computation scheme without revealing the key to individual data holders. We assume that this independent trusted key provider is much more secure and trustable than individual data holders who might have limited capability to protect their data. In addition, the encrypted data is not known and possessed by the data holders, further reducing the opportunity of encrypted data being attacked.

Canny [5] proposes an algorithm that allows a community of users to compute a public aggregate of the data without exposing individual users' data. The method uses SVD to find an orthogonal space based on the aggregated data and projects the original data to such space for collaborative filtering. The similarity between users is determined by the representation of the users in the reduced space. Using such method requires continuously performing SVD for a large dimensional matrix first, which is computationally expensive and also introduces significant latency for the whole system.

4.3.2. Secure Multiparty Computation for Privacy Preserving Data-mining

Secure multiparty computation naturally provides an interesting direction for distributed privacy preserving data-mining since MPC provides protocols that allow different parties to jointly perform a computational task without revealing their own data to the others. Lindell and Pinkas [40] adopt a secure multiparty computation protocol to solve a decision tree learning problem with the popular ID3 algorithm. [57] presents some early attempts toward building a secure multiparty computation toolkit of privacy-preserving distributed computation techniques that can be assembled to solve specific real-world problems. It presented protocols for secure sum, secure set union, secure size of set intersection and secure scalar product. [45][69]developed protocols for privacy preserving k-means clustering in a distributed scenario. A general concern about MPC is its computational and communication complexity that make it difficult to use for large data sets and large number of parties (e.g., data holders). For example, in the MPC algorithms in [45][69], the k-means clustering is an iterative process, so the computational and communication costs highly depend on the dataset and the number of data holders. Another problem of the MPC based data-mining protocols is their low level of scalability. In [45][69], when there are new data holders join the multiparty group, the whole MPC process involving all the other data holders needs to be repeated. To address these problems, our proposed framework takes a two-step approach - developing new MPC protocols in performing simultaneous encryption and compression of data, followed by data mining in the compressed and encrypted domain. This separation approach significantly reduces the computational and communication complexity introduced by the MPC based data-mining approaches, while allowing various data mining operations to be applied to the compressed and encrypted data that needs to be generated only once. Note that the computational complexity and communication complexity of our MPC protocols for encryption have a linear relationship with the number of data holders and the amount of data. In addition, in designing the new MPC protocols, effort has been made to move the complexity from data holders to the cloud.

4.3.3. Compressive Sensing Domain Data-mining

Most of the literature of compressive sensing has focused on improving the speed and accuracy of compressive sensing reconstruction. Davenport *et al* [85] take some initial steps towards a more general framework called compressive signal processing (CSP), which shows fundamental signal processing problems such as detection, classification, estimation, and filtering can be solved in the compressive sensing domain. Hsu *et al* [51] use compressive sensing to learn to predict compressed label vectors and then reconstruct the learned compressed label vectors. It provides mathematical proof and experimental results that show prediction of sparse vectors could be done in the compressive sensing domain. Calderbank *et al* [47] give some theoretical results and show that compressed learning, learning directly in the compressed domain, is possible. It gives the tight bounds demonstrating that the linear kernel SVM's classifier in the measurement domain, with high probability, has an accuracy close to the accuracy of the best linear threshold classifier in the original data domain. Earlier than the birth of the compressive sensing theory, random projection using the Johnson-Lindenstrauss Lemma [39] was also used for privacy preserving data-mining. The paper by Liu *et al* [39] gave the following lemma about linear correlation in the compressive sensing domain:

Lemma 1. [39]: Let $X = \{x_i\}$, $Y = \{y_i\}$ be two $n - dimensional$ vectors and $\Phi_{m\times n}$ be an $m \times n - dimensional$ random matrix. Each entry of $\Phi_{m\times n}$ is independent and identically chosen from Gaussian distribution with mean zero and variance $1/m$. Further, let $P = \Phi_{m\times n}X$; and $R = \Phi_{m\times n}Y$.

Then:

$$E[X^TY - P^TR] = 0$$

$$Var[X^TY - P^TR] = \frac{1}{m}\left(\sum_1^n x_i^2 \sum_1^n y_i^2 + \left(\sum_1^n x_i y_i\right)^2\right)$$

4.4. The Proposed Framework

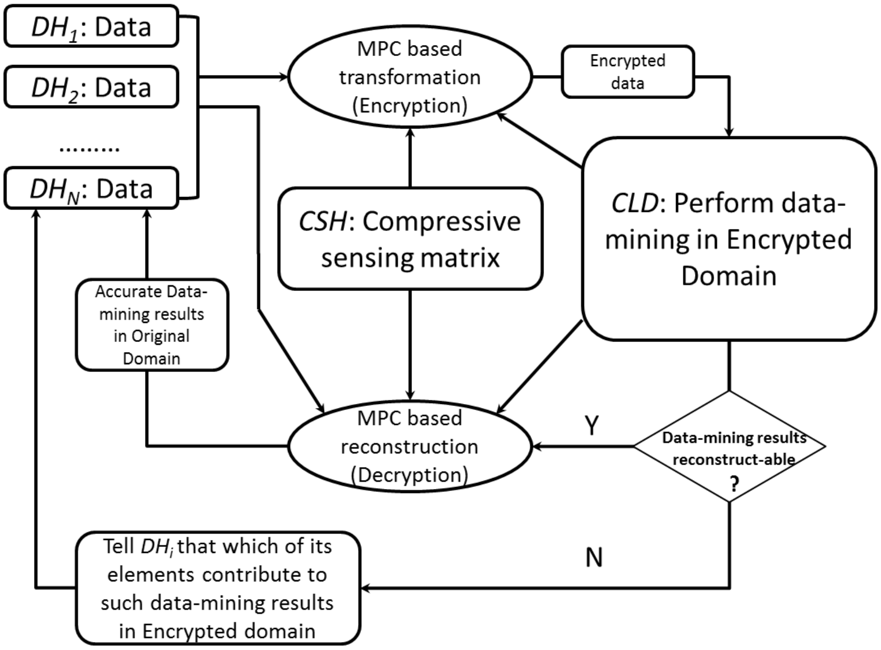

DH_1: Data
DH_2: Data
....
DH_N: Data
MPC based transformation (Encryption)
Encrypted data
CSH: Compressive sensing matrix
CLD: Perform data-mining in Encrypted Domain
Accurate Data-mining results in Original Domain
MPC based reconstruction (Decryption)
Y
Data-mining results reconstruct-able ?
N
Tell DH_i that which of its elements contribute to such data-mining results in Encrypted domain

Figure 4.1. Architecture of the proposed generic privacy preserving data-mining framework.

Figure 4.1 presents the architecture of the framework. The framework is developed under the semi-honest assumption. Our framework considers three major parties: the compressive sensing matrix holder (CSH) serving as the independent trusted key provider, the data holders (DH's) and the cloud computing service provider (CLD). In our framework, the compressive sensing encryption process and decryption process are implemented using a secure multiparty computation scheme, so that:

1. The compressive sensing matrix is held and known by CSH only. If such matrix is leaked to CLD, it could reconstruct the data from the data holders. If it is leaked to some data holders, it is risky that other data holder's privacy may be compromised. We assume the CSH is very secure and trustable.
2. Data holder does not want to expose its own private data to any other parties.
3. CLD needs to prevent any other parties from having access to the encrypted data, since if they are leaked to CSH, it could be reconstructed by CSH and the data holder's privacy is compromised. If it is leaked to DH, the compressive sensing matrix is vulnerable under known-plaintext attack. We assume CLD is very secure.

First, the DHs will encrypt all of their data using the compressive sensing matrix as the key to the compressive sensing domain by working with the CSH and CLD through MPC. Then CLD will have all the data in the encrypted domain. CLD runs some data-mining algorithms such as K-means clustering to generate the data-mining results in the compressive sensing domain. If certain data-mining result is reconstruct-able, the result could be sent to DHs after compressive sensing reconstruction through MPC. If it is not reconstruct-able, CLD can let DHs know the data-mining results by telling them how their data points contribute to such result. In Section 4.5, we introduce our MPC based CS transformation and CS reconstruction protocols that are used by the framework.

4.5. The Proposed Secure Multiparty Computation Protocols

Here we first give the secure scalar product protocol and the secure comparison protocol in Section 4.5.1, which lay the foundation for the MPC based CS transformation and reconstruction protocols which are later introduced in Section 4.5.2 and Section 4.5.3 respectively.

4.5.1. Existing MPC Protocols Referenced by the Proposed Framework

4.5.1.1. Secure Scalar Product Protocol

There are many existing secure scalar protocols such as homomorphism based, commodity server based, secret sharing based techniques as summarized in [91]. Homomorphism based techniques only require two parties to be involved in the computation process and let the third party have the final results, which is the best fit for our scenario. In this chapter, we adopt the protocol proposed by Goethals *et al* (as shown in Protocol 1) based on the Paillier public key system and its homomorphism properties.

4.5.1.2. Solution to the Millionaire's Problem

Yao's Millionaires' ("Greater than" or "GT") problem [54] is to determine who is richer between two parties such that no information about the amount of assets a party has is leaked to the other. There are many GT solutions, and here in this chapter we adopt the solution proposed by Lin *et al* [53] as shown in Protocol 2, because of its lower complexity.

Protocol 1. Private secure scalar protocol

Input: Alice owns private vector $\vec{x} \in \mathbb{Z}^M$ and Bob owns private vector $\vec{y} \in \mathbb{Z}^M$.

Output: Only Charlie gets product $\vec{x} \cdot \vec{y}$.

1. Setup phase. Alice does: Generate a Paillier key pair (sk, pk). Send pk to Bob
2. Alice does for $i \in \{1, \dots, M\}$: Generate a random new number r_i. Send $c_i = E_{pk}(\vec{x}_i, r_i)$ to Bob.
3. Bob does: Set $w \leftarrow \prod_{i=1}^{M} c_i^{\vec{y}_i}$.
   Generate a random plaintext $s_B \in \mathbb{Z}$ and a random number $r' \in \mathbb{Z}$.

   Send $w' = w \cdot E_{pk}(-s_B, r')$ to Alice. Send s_B to Charlie.
4. Alice does: Computes $s_A = D_{sk}(w') = \vec{x} \cdot \vec{y} - s_B$ and sends $s_A \in \mathbb{Z}$ to Charlie.
5. Charlie has $\vec{x} \cdot \vec{y} = s_A + s_B$.

Protocol 2. Solution to the Millionaire's Problem

Input: Alice has positive private integer x and Bob has positive private integer y.

Output: they want to find who has the bigger value without revealing the private values to the other.

1. Alice has private value x and denotes it as a binary sequence: $x_L x_{L-1} \dots x_1$, where L is the bit length. Alice has an ElGamal public key pair and share the public key with Bob.
2. Alice prepares a 2× L-table T[i,j] [53], i∈ {0,1}, $1 \le j \le L$, such that:
   T[x_i, i] = $E_{ElGamal}(1)$ and T[$\bar{x}_i, i$] = $E_{ElGamal}(r_i)$ for some random r_i.

   And send T to Bob.
3. Bob denotes y as $y_L y_{L-1} \dots y_1$ and generates set S_y^0 definition of S_y^0 is given in [53], then Bob does the following:
   For each t = $t_L t_{L-1} \dots t_i \in S_y^0$, compute

   $$c_t = T[t_L, L] * T[t_{L-1}, L-1] \dots * T[t_i, i]$$

   Prepare l = $L - \left|S_y^0\right|$ random encryptions z_j, $1 \le j \le l$.

   Scalarize c_t's and permutate c_t's and z_j's randomly as $c_1, c_2, \dots, c_L$. And send all $c_i, 1 \le i \le L$ to Alice.
4. Alice decrypts $c_i, 1 \le i \le L$ and get m_i= $D_{ElGamal}(c_i)$, and determine $x > y$ if and only if some $m_i = 1$.

4.5.2. MPC based CS Transformation Protocol

Based on Protocol 1, it is straightforward to give the secure CS transformation protocol (Protocol 3).

Protocol 3. Secure CS transformation

Input: DH has CS matrix $\mathit{\Phi}_{m\times n}$, WO has $\vec{v}$, an $n\times1$ vector.

Output: CLD has $\vec{k} = \mathit{\Phi}_{m\times n} * \vec{v}$.

Between DH and WO, for all $\vec{\theta}_j$, where $1 \leq j \leq m$, a row of $\mathit{\Phi}_{m\times n}$, apply Protocol 1, let CLD have $\theta_j^T \cdot \vec{v}$. Finally, CLD will have $\vec{k}_{m\times1} = \mathit{\Phi}_{m\times n} * \vec{v}$.

4.5.3. MPC based CS Reconstruction Protocol

Before we introduce the MPC based CS reconstruction protocol, we design an OMP algorithm (based on the original orthogonal matching pursuit algorithm [86]) which could fit the MPC framework as given in Algorithm 1. In Algorithm 1, $Value1$ and $Value2$ are private values of CSH; h is held by CLD only, $\hat{y}_J$ and $|r|_2$ should only be known by DH. In Algorithm 1, $value1$ and $value2$ are computed based on the selected columns of $\mathit{\Phi}_{m\times n}$ and the selection criterion is based on the correlation results between columns of $\mathit{\Phi}_{m\times n}$ and h. The final reconstructed $\hat{y}$ is calculated from $value1$, h and the sequence J. Since in the framework, $\mathit{\Phi}_{m\times n}$ is the secret of CSH, h is only owned by CLD and we will give the computed r and $\hat{y}$ only to DH. So we only need to perform Step 1, Step 4 and Step 6 using MPC protocols. It is easy to see that Step 6 only requires the secure scalar product protocol. In the following, we present the MPC protocols for Step1 and Step 4.

Algorithm 1. OMP algorithm designed for MPC

```
Parameters: original vector sparsity level $k$, CS matrix:$\Phi_{m\times n} = \{a_1|a_2|a_3|a_4...|a_n\}$, $h$ is the vector in the CS domain.
Init: set $J \leftarrow \emptyset$, residual error $r \leftarrow h$, $\hat{y} \leftarrow \vec{0}$, $A_J \leftarrow \emptyset$, $value2 \leftarrow I_{m\times m}$ ($I_{m\times m}$ $is\ an\ identity\ matrix$)
For $iter$ = 0,....,$2k$ do
    Step1.$j_{max} \leftarrow argmax_{1\le i\le n, i\notin J}|(a_i^T * value2) * h|$
    Step2. $J \leftarrow J \cup \{j_{max}\}$,$A_J \leftarrow A_J \cup a_{j_{max}}$ and set $a_{j_{max}} as\ \vec{0}$
    Step3. $value1 = (A_J^T * A_J)^{-1} * A_J^T$
           $value2 = \left(I_{m\times m} - A_J * \left(A_J^T * A_J\right)^{-1} * A_J^T\right)$
    Step4. $r^2 \leftarrow |value2 * h|^2$
    Step5. If $r^2$ is smaller than a small threshold End For
        End if
End For
If $r^2$ is smaller than a threshold
Step6. $W \leftarrow value1 * h$
Given the set: $J = \{J_1, J_2, ...\}$( $J_i$ is $j_{max}$ from loop i); and the vector: $W = \{w_1|w_2|...\}$:
Construct $\hat{y}$ as $\hat{y}_{(J_i)} \leftarrow w_i, where\ i = \{1,2,...,|J|\}$
Output: Reconstructed value: $\hat{y}$
Else No reconstruction results
End if
```

In Step 1 of Algorithm 1, $(a_i^T * value2)\ where\ 1 \le i \le n$ are vectors owned by CSH and h belongs to CLD. We apply Protocol 1 between $(a_i^T * value2)$ and h. Then the CSH will have $\{\alpha_1 + \beta_1, \alpha_2 + \beta_2, ...\}$(Let α_i represent $a_i^T * value2 * h$) and CLD has the random value sequence$\{\beta_1, \beta_2, ...\}$. The Step 1 requires locating the maximum $a_i^T * value2 * h$, while all the $a_i^T * value2 * h$ need to be hidden from both CSH and CLD for the privacy purpose. We present the hidden maximum value selection protocol (as shown in Protocol 4) based on Protocol 2. The protocol can be easily extended for use in the hidden selection process in Step 1 of the *Algorithm 1*. In the Step 4 of the Algorithm 1, CSH and CLD need to tell DH r^2, which is the loop termination criterion. We give a secure L2 norm protocol for Step 4 of the Algorithm 1 based on the Protocol 1 (as shown in Protocol 5). Then, given the MPC protocols of Step 1, Step 4 and Step 6 of Algorithm 1, the MPC CS reconstruction protocol (as shown in Protocol 6) can be built easily.

Protocol 4. Hidden maximum value selection protocol

Input: CSH has $\{\alpha_i + \beta_i\}$ and CLD has $\{\beta_i\}$.

Output: CSH knows which is larger between $|\alpha_i|$and $|\alpha_j|$, but $|\alpha_i|$ and $|\alpha_j|$ need to be hidden from both CSH and CLD.

```
1. CSH computes $A_1$= $\alpha_i + \alpha_j + \beta_i + \beta_j$,$A_2 = \alpha_i - \alpha_j + \beta_i - \beta_j$. CLD computes $B_1 = \beta_i + \beta_j$,$B_2 = \beta_i - \beta_j$.
2. CSH and CLD find the bigger one between $A_1$ and $B_1$; and the bigger one between $A_2$ and $B_2$ using $SCP$.
3. CLD will have: $\alpha_i \pm \alpha_j \le or \ge 0$. Then
     If $\alpha_i + \alpha_j \ge 0, \alpha_i - \alpha_j \ge 0$, then $|\alpha_i| \ge |\alpha_j|$.
       Else if $\alpha_i + \alpha_j \le 0, \alpha_i - \alpha_j \ge 0$, then $|\alpha_i| \le |\alpha_j|$.
       Else if $\alpha_i + \alpha_j \ge 0, \alpha_i - \alpha_j \le 0$, then $|\alpha_i| \le |\alpha_j|$.
       Else if $\alpha_i + \alpha_j \le 0, \alpha_i - \alpha_j \le 0$, then $|\alpha_i| \ge |\alpha_j|$.
     End if
4. CLD tells CSH which is larger between $|\alpha_i|$ and $|\alpha_j|$.
```

Protocol 5. Secure L2 norm protocol

Input: CSH has a secret matrix $value2$ (in $value2$ every row is a vector t_i^T $where$ $1 \leq$ i $\leq$ n) and CLD has a secret vector h.

Output: DH has $r^2 = |value2 * h|^2$.

1. CSH and CLD apply Protocol 1 between each row of $value2$ and h. Then CSH can get a vector V $=(\rho_1 + \beta_1, \ldots, \rho_m + \beta_m)^T = (t_1 \cdot h + \beta_1, \ldots, t_m \cdot h + \beta_m)^T$, and R = $\{\beta_i\}$, $where$ $1 \leq$ i $\leq$ m, are random numbers known by CLD only.
2. CSH and CLD apply Protocol 1 between V and 2*(-R). Then CSH has x $=\sum_{i=1}^{m}(-2 * \rho_i * \beta_i - 2 * \beta_i^2)$ + H, where H is a random value holds by CLD only. CLD computes z = $\sum_{i=1}^{m}(r_i^2)$.
3. CSH computes y = $|V|_2^2 = \sum_{i=1}^{m}(2 * \rho_i * \beta_i + \beta_i^2 + \rho_i^2)$
4. CSH sends x, y to DH; CLD sends z and H to DH.
5. DH computes $r^2 = x + y + z -$ H = $|value2 * h|^2$.

Protocol 6. MPC Orthogonal Matching Pursuit Protocol

Input: CSH has Compressive sensing matrix $\Phi_{m \times n}$, CLD has a data vector in compressive sensing domain $h_{m \times 1}$.

Output: DH obtains $\hat{y}$ after applying Algorithm 1. $\hat{y}$ satisfies $h = \Phi_{m \times n} * \hat{y}$, where h is private with CLD, $\Phi_{m \times n}$ is private with CSH and $\hat{y}$ is known by DH only.

Init: $value2 = I_{m \times m}$, A_J = ∅, J = ∅

For i = 0,….,$2k$ **do**

1. Between every column of $\Phi_{m \times n}$: μ_j(where $1 \leq$ j $\leq$ n) and $h_{m \times 1}$, CSH and CLD apply Protocol 1, let CSH have $product$ = $\{\mu_1^T * value2 \cdot h + R_1^{CLD}, \ldots, \mu_n^T * value2 \cdot h + R_n^{CLD}\}$. $\{R_1^{CLD}, \ldots, R_n^{CLD}\}$ is a random sequence hold by CLD only.
2. CSH and CLD apply Protocol 4 between every pair of elements and CSH can have index pos that $|\mu_{pos}^T * value2 * h|$ is maximum among all elements of *product*.
3. CSH updates A_J, J, $Value1$ and $Value2$ as shown in Algorithm 1.
4. CSH has $Value2$ and CLD holds h, they apply Protocol 5 to let DH have $|r|_2$, where $|r|_2 = |Value2 *$ h$|_2$.
5. DH decides if $|r|_2$ is smaller than a small threshold:
   If yes, **End Loop**. If no**, continue.**

End for

If DH finds $|r|_2$ is smaller than a threshold

6. CSH and CLD apply Protocol 1, let DH have $\hat{y}_J$ = $Value_1 * h$ CSH sends J to DH and DH construct $\hat{y}$ as shown in Algorithm 1.
   Else No reconstruction

End if

4.6. Framework Analysis

4.6.1. Complexity Analysis

4.6.1.1. *Computational Complexity*

The proposed protocols use the public key systems. Both of the ElGamal and Paillier public key systems are computationally expensive because of the modular exponentiation. We therefore present the computational analysis based on the modular multiplications. The computational complexities of the random number generation and linear operations can be ignored compared to public key operations. ElGamal encryption requires two modular exponentiations, while decryption takes one modular exponentiation in the modified scheme proposed by [53]. Paillier encryption needs one modular exponentiation and decryption takes one modular exponentiation. ElGamal's multiplication of ciphertexts takes two modular multiplications and Paillier's multiplication of ciphertexts takes one modular multiplication.

Let b denote the base, e denote the exponent and a denote the modular, modular exponentiation is represented as:

$$b^e \ mod \ a$$

The complexity of calculating the above is $log\, e$ modular multiplications [61]. When e is bounded by a, the complexity of modular exponentiation is bounded by $O(log\, a)$ modular multiplications.

Assume the ElGamal and Paillier use the modular n1 and n2 respectively, with the same bit length. So each modular exponentiation will take $O(log\, n1) \approx O(log\, n2)$

modular multiplications, and we use $O(log\,n)$ to represent both of them. Assume the plaintext's bit length is L. In Table 4.1, the first column gives the operations that need modular multiplications. Given vector length is N, the second column is the computational complexity in modular multiplication. Protocol 2's computational complexity is given in [53] and summarized in Table 4.2. L is the plaintext's bit length. Protocol 2 is used in the decryption protocol Protocol 6 only. When the vector length is N and the matrix size is M × N, the computational complexities from Protocol 1 to Protocol 5 are given in Table 4.3.

Table 4.1. Protocol 1's computational complexity per vector in number of modular multiplications when the vector length is N.

Encryption	N* $O(log\,n)$
Ciphertexts Exponentiation	N* $O(log\,L)$
Ciphertexts Multiplication	N
Decryption	1
Total	N* $O(log\,n)$+ N* $O(log\,L)$+N+1

Table 4.2. Protocol 2's computational complexity in number of modular multiplications

Encryption	$L \times 2O(log\,n)$
Scalaring	$L \times 2O(log\,n)$
Ciphertexts Multiplication	$L \times O(log\,n)$
Decryption	$2 \times (2L - 3)$
Total	$L \times 5O(log\,n) + 4L - 6$

Table 4.3. Summary of computational complexity in number of modular multiplications

Protocol	Computational complexity (modular multiplications)
Protocol 1 (Secure scalar product)	N* $O(log\,n)$+ N* $O(log\,L)$+N+1
Protocol 2 (Millionaire problem's solution)	$5L$*$O(log\,n)$ +4L −6
Protocol 3 (CS transformation)	M*(N* $O(log\,n)$+ N* $O(log\,L)$+N+1)
Protocol 4 (Hidden maximum value selection)	10L*$O(log\,n)$ +8L −12
Protocol 5 (Secure L2 norm)	M*(N* $O(log\,n)$+ N* $O(log\,L)$+N+1)+ M* $O(log\,n)$+ M* $O(log\,L)$+M+1

Computational complexity of the compressive sensing encryption protocol (Protocol 3):

CSH could send the encrypted compressive sensing matrix as part of the setup stage since it is used repeatedly for each data vector. Then the secure scalar protocol for a data vector will take N* $O(log\, L)$+N+1 modular multiplications. Given that the compressive sensing matrix size is M × N, to encrypt one data vector with length N, the complexity is M*(N* $O(log\, L)$+N+1). Since $O(log\, L)$ is the dominant factor, we can simplify the complexity to be M*N*($O(log\, L)$) modular multiplications. This shows that the computational complexity per vector of the compressive sensing encryption has a linear relationship with the size of the compressive sensing matrix.

Computational complexity of the compressive sensing decryption protocol (Protocol 6):

Protocol 6 is an iterative process. Assume the compressive sensing matrix size is M × N. From the above analysis, it is easy to derive the computational complexity of all the steps within a loop when the loop index number is l, where $l \in \{0,1,2, \dots 2 \times k\}$($k$ is the sparsity level of the original sparse vector). Given Table 4.3, it is easy to derive each step's complexity. Step 1's complexity is (N$-l$)*(M* $O(log\, n)$+M* $O(log\, L)$+M+1). Step 2's complexity is (N $-l$ $-$1)* (10L*$O(log\, n)$ +8L $-$12). Step 5 uses Protocol 5, so its complexity is M*(N* $O(log\, n)$+N* $O(log\, L)$+N+1) + M* $O(log\, n)$ + M* $O(log\, L)$ +M+1. Finally if there is a reconstruction, Step 7 is based on Protocol 1, so it takes (l+1)*(N* $O(log\, n)$+ N* $O(log\, L)$+N+1) modular multiplications. Protocol 6 will have at most $2k$ loops and most of the complexities are contributed by steps within the loop. Since $O(log\, n)$ is much larger than the other factors, we can simplify the complexity to be $2k$*(2*M*N+10*L*N+M)* $O(log\, n)$ modular multiplications per vector. It has a linear relationship with the size of the compressive sensing matrix.

4.6.1.2. Communication Complexity

Both ElGamal's ciphertext and Paillier's ciphertext will take $O(log\,n)$ bits which is the dominant factor for the communication cost. Table 4.4 summarizes the communication costs of the protocols from Protocol 1 to Protocol 5.

Table 4.4. Summary of the communication complexity (in bits) of the fundamental protocols

Protocol	Communication complexity (bits)
Protocol 1 (Secure scalar product)	(N+1) * $O(log\,n)$
Protocol 2 (Millionaire problem's solution)	6(L $O(log\,n)$)
Protocol 3 (CS transformation)	M*(N+1) * $O(log\,n)$
Protocol 4 (Hidden maximum value selection)	12(L $O(log\,n)$)
Protocol 5 (Secure L2 norm)	M*((N+1) * $O(log\,n)$)+ (M+1) * $O(log\,n)$

Communication complexity of the compressive sensing encryption (Protocol 3):

CSH could send the encrypted compressive sensing matrix in the setup stage. So each round of using Protocol 1 in Protocol 3 to encrypt a vector will have less communication cost. Given that the compressive sensing matrix is M by N: transmitting the encrypted compressive sensing matrix in the setup stage will take: M*(N+1) * $O(log\,n)$ bits. For each new data vector, DH only needs sending M*$O(log\,n)$ bits to CSH. The communication cost of CS transformation has linear relationship with the CS matrix size.

Communication complexity of the compressive sensing decryption (Protocol 6):

Protocol 6 is an iterative process. Given the compressive sensing matrix with a size of M ×N, and when the iteration loop index number is l, where $l \in \{0,1,2,\ldots 2 \times k\}$. Step 1 costs (N-$l$)*M*((N+1) $* O(log\, n)$) bits and Step 2 costs (N-l-1)*12*($L\ O(log\, n)$) bits. Step 5 uses Protocol 5 and its communication cost is M*((N+1) $* O(log\, n)$)+ (M+1) $* O(log\, n)$ bits. If there is a reconstruction, finally Step 7 costs (l+1)*M*((N+1) $* O(log\, n)$) bits. Protocol 6 will have at most $2k$ loops and most of the complexities are from the steps within the loop. We can simplify the complexity as $2k$*(M*N+12*L*N+M)* $O(log\, n)$ bits. The communication cost of CS reconstruction has linear relationship with the CS matrix size.

In our framework, different DHs are independent of each other in both the encryption and decryption process. Different data entries are also independent of each other in the MPC framework, so the total computational and communication complexity of our MPC encryption/compression protocols will have linear relationship with the number of DHs as well as the amount of data.

4.6.2. Security Analysis

4.6.2.1. MPC based CS transformation

It has been proven in the original paper [92] that Goethals's secure scalar protocol is secure under the semi-honest model. It is straightforward to see that the MPC protocols (Protocol 1 and Protocol 3) are also secure under the semi-honest assumption that all the parties follow the protocol strictly and no two parties will collude to attack a third party. After running the secure CS transformation protocol, DH and CSH do not leak their private values to other parties. Only the CLD has the data in the CS domain.

4.6.2.2. MPC based CS Reconstruction

As for the security of the CS reconstruction process, it is easy to see that Step 6 of the Protocol 6 only utilizes the Protocol 1 which has proven to be secure in [92] under the semi-honest model. The information leak, however, might happen in Step 2 of Protocol 6 (Protocol 1). In Step 3 of Protocol 1, CLD will have the information: $\alpha_i \pm \alpha_j \ \leq or \geq \ 0$, which may help CLD infer CSH's private value $a_i^T * value2$. But CSH can avoid such vulnerability in two ways: First, randomly permuting the order of $a_i^T * value2$ for every loop so that CLD cannot track $a_i^T * value2$; Second, in every loop, CSH randomly generates a random value $\gamma \in \mathbb{R}^*$, then $a_i^T * value2$ can be disguised by multiplying the value γ before executing Protocol 1, so that the sign of $\alpha_i \pm \alpha_j$ will appear to be random for CLD. Note that the Step 3 of Protocol 1 shows that when the signs of $\alpha_i + \alpha_j$ and $\alpha_i - \alpha_j$ are flipped at the same time, the decision for the relationship between $|\alpha_i|$ and $|\alpha_j|$ will not be affected. Another information leak to CSH is j_{max} in each loop at Step 1 of *Algorithm 1*, and CSH will have sequence J (step 2 of *algorithm 1*). Even though CSH cannot get the absolute values of the entries of h, CSH can guess the non-zero entry positions and the order of the magnitudes of non-zero entries. Solutions for this issue will be studied in the future work.

4.6.2.3. CS domain data processing

It has been shown that random projection based on transformation is viable for protecting privacy [39], even though sharing the transformation matrix incurs a security issue. We address such security issue by introducing a semi-trusted party managing the CS matrix. Further, our framework presents a scenario which is similar to the system proposed by Lu *et al* [83][84]. Lu *et al* prove that when the original vectors are normalized to have the same constant norm and their correspondences in the same random projection domain (ciphertext only) are obtained by a third party, the third party gains no additional information other than the correlations between different data under the semi-honest security model. The correlation between data will be inevitably leaked to the cloud since the other parties desire the cloud to provide such services.

4.7. Complexity Comparison to MPC based Data-mining Algorithm

We compare the complexity of our proposed MPC framework for k-means clustering to the complexity of the MPC algorithm for the k-means proposed in [69]. We continue to use the notations used in the above complexity analysis.

Assume there are x data holders, y data entities and let c be the number of clusters. K-means is an iterative process, and we assume I is the number of iterations for the clustering algorithm to converge. Most of the k-means computational complexity will be spent on computing vector distances. Let Θ(M) represent the costs of one such operation, where M is the compressive sensing measurement length. The *finding closest center* step of k-means computes c*y distances, so its overall complexity is Θ(c*y*M). In the *center recomputation* step of k-means, each vector gets added to a centroid once, so the complexity is Θ(y*M). The overall complexity of k-means is therefore Θ(I*c*y*M). Then the computational complexity per vector of k-means is Θ(I*c*M). So our framework's computational complexity for k-means is M*(N+1)*($O(\log n)$)+ Θ(I*c*M) per vector and communication complexity is M*N* $O(\log n)$ bits per vector. The MPC computational and communication complexity per data entry of our framework has nothing to do with the number of data holders. It only depends on the vector length N and the compressive sensing measurement size M.

The MPC algorithm in [69] uses additive homomorphic public key encryption system in step 4-5 of Algorithm 3 in [69], which finds the closest cluster for each data entry. The step 4-5 of that algorithm alone costs I*x*c*($3O(\log n)$) modular multiplications per data entry and each data holder has to share certain amount of public key operation burdens. [69] gives the algorithm's communication cost analysis, and shows that step 4-5 of Algorithm 3 alone will cost 2*I*$(x-1)$*c* $O(\log n)$ bits per data entry.

Since the modular multiplication is more computational expensive than the vector distance computing, it is obvious that the computational complexity (I*x*c*($3O(\log n)$)) of [69] is much higher than the complexity for clustering of our framework (Θ(I*c*M)). When there are a lot of data holders, the number of clusters is large, and I is big. The computational and communication complexity of only two steps of the algorithm in [69] may be larger than the overall computational and communication complexity of our framework. In our framework, the k-means clustering will be performed in a cloud environment with the compressed data vector dimension, which will make the k-means perform much more efficiently.

More importantly, the complexity of our MPC protocols is shared by many data mining tasks that potentially will be applied to the same set of encrypted data. Furthermore, our framework is designed to reduce the data holders' computational and communication complexity, which the algorithm of [69] cannot offer. To summarize, our proposed framework outperforms the MPC algorithm proposed in [69] in several aspects:

- The overall computational and communication complexity is much lower in our framework, because we decouple the MPC encryption process from the data-mining process to avoid the high complexity incurred by the MPC processing for the iterative data-mining algorithms. In addition, our framework creates opportunities to take advantage of cloud computing technologies as well as lower data dimension to expedite the data-mining process.
- The data holders are only responsible for a limited amount of computational and communication burden in our framework, while the MPC based k-means clustering in [69] requires all data holders to perform public key operations and bear significant communication cost.
- Our framework achieves a higher scalability and flexibility than other MPC protocols such as the one in [69]. For example, in [69], if new data holders want to join the group, the whole MPC process has to start over. But in our framework, the MPC encryption step makes the data holders independent of each other. Furthermore,

previous MPC algorithms such as the one in [69] are limited to specific data-mining methods, while in our framework, after the MPC based encryption, different kinds of data-mining analysis can be performed simultaneously on the same set of compressed and encrypted data. Even though [45] proposes methods that can outperform the one in [69], they still suffer from iterative MPC processing as well as scalability and security problems.

4.8. Experimental Results

To confirm that data-mining in the compressive sensing domain is valid, this section provides some further experimental results for data processing in the encrypted compressive sensing domain based on the MovieLens 1M Dataset [62], UCI KDD Dataset [70] as well as self-generated data. In addition, simulation of the proposed MPC framework is also conducted to demonstrate that our framework works efficiently and effectively, especially for DH.

4.8.1. Compressive Sensing Domain Data Processing

4.8.1.1. Distance preservation

The MovieLens 1M dataset consists of 1 million ratings from more than 6000 users on 4000 movies. It was originally designed for collaborative filtering to recommend some movies that a user may be interested in based on the users' rating history. Most of the user ratings are sparse in the MovieLens 1M dataset, since on average a user only rates 4.19% of 4000 movies. If a movie is rated by a user, the rating value is an integer and ranges from 1 to 10. We measure the error of pair-wise distance after compressive sensing transformation. For all the data vectors V_i where i$\in \{1,2,\dots,n\}$ of the MovieLens Dataset, their corresponding values in the compressive sensing domain are K_i where i$\in \{1,2,\dots,n\}$. The relative error is given by:

$$error = \frac{\left| \left| V_i - V_j \right|_2^2 - \left| K_i - K_j \right|_2^2 \right|}{\left| V_i - V_j \right|_2^2}$$

where i, j $\in \{1,2,\dots,n\}$. The compressive sensing matrix $\Phi_{m\times N}$ is constructed in a way that all the entries follow the Gaussian distribution with mean 0 and variance $1/m$, which promises the JL-lemma embeddings. m/N will be the compressive sensing rate. Table 4.5 gives the statistics of the relative error under different compressive sensing rates. It shows that even with a compressive sensing rate of 0.2, the average error is still as low as 4.05% and variance is 0.0937%. This confirms that the compressive sensing transformation can preserve the Euclidean distance very well and data-mining techniques based on Euclidean distance could be performed in such compressive sensing domain without sacrificing too much of the accuracy.

Table 4.5. Relative errors of the Square of the Euclidean distance of Two Attributes

Compressive sensing rate	min	Max (%)	Avg (%)	Var (%)
0.9	0.0	13.0319	1.8572	0.0198
0.8	0.0	14.1061	2.0662	0.0241
0.7	0.0	16.9437	2.1534	0.0265
0.6	0.0	16.8986	2.3098	0.0301
0.5	0.0	17.1858	2.5091	0.0361
0.4	0.0	19.8011	2.8274	0.0455
0.3	0.0	22.9261	3.2342	0.0604
0.2	0.0	26.9828	4.0505	0.0937

4.8.1.2. K-means clustering

We use the Synthetic Control Chart Time Series data set from the UCI KDD Archive for K-means clustering algorithm testing. This data contains 600 examples of control charts, each has 60 attributes. It is designed as a common dataset for testing clustering algorithms. There are six different classes of control charts: normal, cyclic, increasing trend, decreasing trend, upward shift and downward shift. The K-Means clustering is conducted in the original domain as well as compressive sensing domain with different compressive sensing rates. We calculate the classification error of clustering in the compressive sensing domain compared to clustering in the original data domain. The compressive sensing matrix is constructed in the same way as in section 5.1.1. Table 4.6 shows that the classification error is small even when compressive sensing rate is 20%.

Table 4.6. Relative cluster error of K-means clustering of UCI KDD data

compressive sensing rate	Classification Error rate (%)
0.9	2.3333
0.8	3.0
0.7	3.3333
0.6	3.3333
0.5	4.1667
0.4	5.1667
0.3	6.3333
0.2	8.1667

4.8.2. Experiments of MPC for Compressive Sensing

We use our self-generated dataset to measure the communication cost for both of the MPC encrytion and decryption. In this dataset, the dimension of the original data is 100. The sparsity level (number of non-zero entries of a data vector) k is less than or equal to

5. Each element of the data vector is a floating point ranging from 1 to 10. Theoretically, for a compressive sensing matrix $\Phi_{m\times N}$ with i.i.d. Gaussian entries, m = $O(k\ log\ N/k)$ is sufficient for perfect reconstruction. In our experiments, we set m to 30 which not only promises sufficient measurements for perfect reconstruction but also the Euclidean distance preserving properties. Paillier and Elgamal systems use ciphertexts with 2048 bits.

4.8.2.1. MPC based compressive sensing encryption

The total communication cost of the setup stage is 768000 bytes, where CSH needs to send DH the encrypted compressive sensing matrix. Such communication cost depends exclusively on the size of the compressive sensing matrix of CSH. Table 4.7 shows the average pair-wise communication costs (bytes to encrypt a single vector with length 100) between parties.

Table 4.7. Average communication costs of encrypting a vector (data in public key domain only).

From	To	Traffic (Bytes)
CSH	DH	0
DH	CSH	7680
DH	CLD	240
CLD	DH	0
CLD	CSH	0
CSH	CLD	240

4.8.2.2. MPC based compressive sensing reconstruction

The computational cost for DH in the decryption step could be ignored, since all DH needs to do is to receive a norm and execute an if-else operation as shown in Protocol 6. The decryption is an iterative process, and is typically performed only for the data mining results (e.g., cluster center in the encrypted domain). Table 4.8 shows the pair-wise average communication cost (Bytes needs to decrypt an encrypted vector with length 30). It shows that the communication cost for DH is very small, most of the communications occur between CSH and CLD, which helps save the computational power for DHs with limited computing resources such as mobile users.

Table 4.8. Average communication costs of decrypting a vector (data in public key domain only)

From	To	Traffic (Bytes)
CSH	DH	64
DH	CSH	0
DH	CLD	0
CLD	DH	64
CLD	CSH	1976243
CSH	CLD	2910463

4.9. Conclusion and Future Works

This chapter proposes a compressive sensing based secure multiparty computation framework that enables efficient privacy preserving collaborative data-mining and privacy preserving storage. Compared to prior works, our framework improves in the aspect of either security or scalability. We also developed a MPC adapted orthogonal matching pursuit algorithm and its MPC protocol. Our framework is also designed taking into account the future trend of cloud computing where cloud will provide computational services as well as storage services, in which scenario privacy is a big concern. Our framework requires scaling a floating point value to an integer in order to use the homomorphism encryption based MPC protocols. Further work will include analyzing how the scaling factor affects the accuracy of the MPC framework. Future work also includes developing efficient distributed data-mining algorithms for cloud computing and further computational and communication complexity reduction.

Chapter 5. Compressive sensing based privacy preserving storage and secure watermark detection framework

In this chapter, we identify a cloud computing application scenario that requires simultaneously performing secure watermark detection and privacy preserving multimedia data storage. Then we propose a compressive sensing based privacy preserving storage and secure watermark detection framework based on the generic privacy preserving data-mining framework proposed in Chapter 4.

5.1. Introduction and Related Works

Due to the rapid growth of the Internet and social networks, it is very easy for a user to collect a large amount of multimedia data from different sources without knowing the copyright information of those data. The user may want to take advantage of the cloud for storage, and at the same time, work with copyright owners for watermark detection while keeping those self-collected multimedia data private. A legal cloud offering storage services may also desire to participate in watermark detection initiated by the users, or initiate watermark detection itself without the involvement of the users, to check if the uploaded multimedia data is copyright protected. Another benefit of storing the encrypted multimedia data and facilitating encrypted domain watermark detection in the cloud is that those encrypted data can be reused if the image data holder (or the cloud) needs to work with other watermark owners later for secure watermark detection.

Traditional secure watermark detection techniques are designed to convince a verifier whether or not a watermark is embedded without disclosing the watermark pattern so that an untrusted verifier cannot remove the watermark from the watermark protected copy [73][74]. Two types of approaches have been proposed for secure watermark detection: asymmetric watermarking [75][76] and zero-knowledge watermark detection [77][78][79]. However, most of the existing secure watermark detection works assume the watermarked copy are publicly available and focus on the security of the watermark pattern, while the privacy of the target media on which watermark detection is performed has received little attention. But for some applications such as the scenario given above, it is required to protect the multimedia data's privacy in the watermark detection process. Performing privacy preserving storage and secure watermark detection simultaneously is possible by using the existing secure watermark detection technologies such as zero-knowledge proof protocols [77][78][79] that transform the multimedia data to a public key encryption domain. However, their limitations, such as complicated algorithms, high computational and communication complexity [73], and large storage consumption in the public key encryption domain, may impede their practical applications.

We propose a compressive sensing based privacy preserving watermark detection framework that leverages secure multiparty computation and the cloud. In our framework, the target image/multimedia data is possessed by the image holder only. A compressive sensing matrix is issued by a certificate authority (CA) server to the image holder. The image holder transforms the DCT coefficients of the image data to a compressive sensing domain before outsources it to the cloud. For secure watermark detection, the watermark is transformed to the same compressive sensing domain using a secure multiparty computation (MPC) protocol and then sent to the cloud. The cloud only has the data in the compressive sensing domain. Without the compressive sensing matrix, the cloud cannot reveal the original multimedia data and the watermark pattern. The cloud will perform watermark detection in the compressive sensing domain. The image data in the compressive sensing domain can be stored in the cloud and reused for detection of watermark from many other watermark owners.

Our system is secure under the semi-honest [82] assumption that all parties comply with the protocol's procedure strictly, and none of them will actively withdraw midway or incorporate false or malicious data. No two parties will collude to attack a third one. But during the computing process, they may try to keep all the intermediate information, so that they can infer others' input after the process. Semi-honest model is a reasonable assumption for adversaries such as third-party service providers [84].

The rest of this chapter is organized as follows. In Section 5.2 and Section 5.3, we show that watermark detection of the correlation-based watermarking system in the CS domain is viable theoretically, and the expected performance is derived. Section 5.5 gives the details of the proposed framework together with some analysis. Section 5.6 presents the experimental results. The chapter concludes in Section 5.7.

5.2. Correlation-based Watermarking System

In Zeng and Liu's work [80], watermarks are embedded in the discrete cosine transform (DCT) domain. Let the feature set $\{d_i\}$ be the set of selective DCT coefficients (i.e., those with absolute values larger than certain perceptual thresholds) excluding the

DCs. The embedding process is $d_i' = d_i + w_i$, where $\{w_i\}$ is the inserted watermark signals which is derived from the watermark pattern $\{y_i\}$. In the detection process, the test feature set $\{x_i\}$ is correlated with the watermark pattern $\{y_i\}$. Detection of the watermarks is accomplished via the hypothesis testing:

H_0: x_i=d_i+n_i, without watermark

H_1: x_i=d_i+w_i+n_i, with watermark

where n_i is the noise. The normalized correlating detector outputs the test statistic q, which is compared to a threshold T to determine if the test image contains the claimed watermarks:

$$q = \frac{\sum_{i=1}^{n} z_i}{V_{z_i} * \sqrt{n}} = \frac{M_{z_i} * \sqrt{n}}{V_{z_i}} \qquad (5.1)$$

where $z_i = x_i y_i$, n is the size of the feature set $\{z_i\}$. M_{z_i} and $V_{z_i}^2$ are the sample mean and sample variance of z_i, given by:

$$M_{z_i} = \frac{\sum_{i=1}^{n} z_i}{n};\ V_{z_i}^2 = \frac{\sum_{i=1}^{n} (z_i - M_{z_i})^2}{n-1} \quad (5.2)$$

Then under hypothesis H_0, for large n, q is approximately a normal distribution q ~N(0,1). Under H_1, q ~N(μ,1), where μ >0.

5.3. Watermark Detection in the Compressive Sensing Domain

Compressive sensing or random sampling domain linear correlation hypothesis test has been proven feasible and the error bound analysis has been given in works such as [85][89]. In [85][89], the analysis is based on the *Neyman-Pearson* detector and the statistic for detection is computed *given that the random projection matrix Φ is known*. However, in the scenarios addressed in our work, the entity who performs watermark detection does not have access to Φ, as described in Section 5.4. In our work, the statistic is calculated based only on the observation in the compressive sensing domain. We not only demonstrate that the statistical correlation hypothesis test based watermark detection is feasible by using only the data in the CS domain, but also show specifically what parameters will affect the watermark detection performance.

Let vector X and Y represent the sets $\{x_i\}$ and $\{y_i\}$ in Section 5.2 respectively:

$$X = [x_1, x_2, \dots, x_n]^T; Y = [y_1, y_2, \dots, y_n]^T$$

Let the compressive sensing matrix be $\Phi_{m\times n}$ whose entries $\Phi_{i,j}$ have the distribution given in Lemma 1, then after the projection we have:

$$P = [p_1, p_2, \dots, p_m]^T = \Phi_{m\times n}X$$

$$R = [r_1, r_2, \dots, r_m]^T = \Phi_{m\times n}Y$$

where given X and Y, $\{p_i\}$ and $\{r_i\}$ are both i.i.d. Gaussian.

Similarly, we define the statistic q^{cs} in the CS domain as:

$q^{cs} = \frac{\sum_{i=1}^{m} z_i^{cs}}{V_{z_i^{cs}} * \sqrt{m}}$ (5.3)

where $z_i^{cs} = p_i r_i$; Then under H_0, for large m, q^{cs} is approximately a normal distribution with unit variance, $q^{cs} \sim N(0,1)$. Under H_1, $q^{cs} \sim N(\mu^{cs}, 1)$, where $\mu^{cs} > 0$. To analyze the relationship between μ and μ^{cs}, let us look at the conditional expectation of q^{cs}, given X, Y:

$$E(q^{cs}|X,Y) = E\left[\frac{\sum_{i=1}^{m} z_i^{cs}}{V_{z_i^{cs}} * \sqrt{m}} |X,Y\right]$$

$$= E(\sum_{i=1}^{m} z_i^{cs} |X,Y) * E\left(\frac{1}{V_{z_i^{cs}} * \sqrt{m}}\middle|X,Y\right) + Cov(\sum_{i=1}^{m} z_i^{cs}, \frac{1}{V_{z_i^{cs}} * \sqrt{m}} |X,Y) \qquad (5.4)$$

Since $\{f(x) = \frac{1}{x}, where\ x > 0\}$ is a convex function, according to the Jensen's inequality: $E(f(x)) \geq f(E(x))$. Then we have:

$$E\left(\frac{1}{V_{z_i^{cs}} * \sqrt{m}}\middle|X,Y\right) \geq \frac{1}{E(V_{z_i^{cs}} * \sqrt{m}|X,Y)} \qquad (5.5)$$

Since $\{f(x) = \sqrt{x}, where\ x > 0\ and\ f(x) > 0\}$ is a concave function, according to the Jensen's inequality: $E(f(x)) \leq f(E(x))$.:

$$\frac{1}{E(V_{z_i^{cs}} * \sqrt{m}|X,Y)} \geq \frac{1}{\sqrt{E\left(V_{z_i^{cs}}^2|X,Y\right) * m}} \qquad (5.6)$$

Then:

$$E(q^{cs}|X,Y) \geq Cov(\sum_{i=1}^{m} z_i^{cs}, \frac{1}{V_{z_i^{cs}} * \sqrt{m}} |X,Y) + \frac{E(\sum_{i=1}^{m} z_i^{cs}|X,Y)}{\sqrt{E\left(V_{z_i^{cs}}^2|X,Y\right) * m}} \qquad (5.7)$$

Lemma 2. Let the compressive sensing matrix $\Phi_{m\times n}$ be an $m \times n\text{–}\,dimensional$ random matrix. Each entry of $\Phi_{m\times n}$ is independently and identically chosen from a Gaussian distribution with mean zero and variance $1/m$. Then, given X, Y,

$$E\left(V^2_{z_i^{cs}}|X,Y\right) = \frac{1}{m^2}\left((\textstyle\sum_{a=1}^{n} x_a y_a)^2 + \sum_{a=1}^{n} x_a^2 \sum_{a=1}^{n} y_a^2\right) \quad (5.8)$$

Proof of Lemma 2: Please see Appendix I for the proof. ■

Based on Lemma 1 (in Section 4.3.3) and Lemma 2, we have:

$$\frac{E(\sum_{i=1}^{m} z_i^{cs}|X,Y)}{\sqrt{E\left(V^2_{z_i^{cs}}|X,Y\right)*m}} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\frac{1}{m}\left((\sum_{a=1}^{n} x_a y_a)^2 + \sum_{a=1}^{n} x_a^2 \sum_{a=1}^{n} y_a^2\right)}} = \frac{M_{z_i}}{\sqrt{\frac{1}{mn^2}\left((\sum_{a=1}^{n} x_a y_a)^2 + \sum_{a=1}^{n} x_a^2 \sum_{a=1}^{n} y_a^2\right)}} = \beta$$

Then, from Eq. (5.7), we have:

$$E(q^{cs}|X,Y) \geq Cov\left(\sum_{i=1}^{m} z_i^{cs}, \frac{1}{V_{z_i^{cs}} * \sqrt{m}}\middle| X,Y\right) + \beta \approx \beta \qquad (5.9)$$

It is well known in statistics that when a random variable is i.i.d Gaussian, its sample mean and sample variance are independent [93]. $\{z_i^{cs}\}$ is not i.i.d Gaussian, however, our experimental results (see Section 4.3) validate that the covariance term in Eq. (5.9) is very small (close to zero). From the β above, we can see that given the original input X and Y for watermark detection, the statistic q^{cs}'s mean value μ^{cs} is affected by the CS matrix height m: if m is smaller, the μ^{cs} will be smaller. In the experimental results section, we show that β is validated to be very close to the mean of the statistic q^{cs}. *This is important since the expected watermark detection performance q^{cs} can be estimated by β (for example during the watermark embedding process) disregard the actual CS matrix used later by other parties for secure watermark detection*.

The statistic q of the original domain is defined as $M_{z_i}/\sqrt{V_{z_i}^2/n}$. In order to find the relationship between β and q, we need to compare the relationship between $V_{z_i}^2/n$ and $\frac{1}{mn^2}\left((\sum_{a=1}^{n} x_a y_a)^2 + \sum_{a=1}^{n} x_a^2 \sum_{a=1}^{n} y_a^2\right)$ as:

$$\lambda = \frac{1}{mn^2}\left((\sum_{a=1}^{n} x_a y_a)^2 + \sum_{a=1}^{n} x_a^2 \sum_{a=1}^{n} y_a^2\right) - \frac{\sum_{a=1}^{n}(x_a y_a)^2 - nM_{z_i}^2}{(n-1)n}$$

$$\geq \frac{1}{nn^2}\left(\sum_1^n x_a^2 \sum_1^n y_a^2 + n^2 M_{z_i}^2\right) - \frac{\sum_{i=1}^{n}(x_a y_a)^2 - nM_{z_i}^2}{(n-1)n} \qquad \text{(Since } m \leq n\text{)}$$

$$= \frac{1}{nn^2}\left(n^2\overline{(x_a y_b)^2} + n^2 M_{z_i}^2\right) - \frac{\overline{(x_a y_a)^2} - M_{z_i}^2}{n-1}$$

$$\approx \frac{1}{n}\left(\overline{(x_a y_b)^2} + M_{z_i}^2\right) - \frac{\overline{(x_a y_a)^2} - M_{z_i}^2}{n} \qquad \text{(Since } n \approx n-1\text{)}$$

$$= \frac{1}{n}\left(\overline{(x_a y_b)^2} - \overline{(x_a y_a)^2} + 2M_{z_i}^2\right) \qquad (5.10)$$

where $\overline{(x_a y_b)^2}$ is the average of all possible $(x_a y_b)^2, (where\ 1 \leq a, b \leq n)$; $\overline{(x_a y_a)^2}$ is the average of all possible $(x_a y_a)^2, (where\ 1 \leq a \leq n)$.

Note that it is expected that the difference between $\frac{1}{n}\overline{(x_a y_b)^2}$ and $\frac{1}{n}\overline{(x_a y_a)^2}$ is very small, as $(x_a y_b)^2$ and $(x_a y_a)^2$ are statistically similar. Our experiment shows that $\frac{1}{n}\overline{(x_a y_b)^2}$and $\frac{1}{n}\overline{(x_a y_a)^2}$ are very close, and $\lambda > 0$, which means that the watermark detection in the CS domain will be inferior to that in the original domain.

Furthermore, based on Eq. (5.10), we have:

$$\gamma = \frac{q}{\beta} = \sqrt{1 + \frac{\lambda}{V_{z_i}^2/n}} \geq \sqrt{1 + \frac{\left(\overline{(x_a y_b)^2} - \overline{(x_a y_a)^2} + 2M_{z_i}^2\right)}{V_{z_i}^2}} \qquad (5.11)$$

A larger γ means larger watermark detection distortion caused by the CS transformation.

5.4. The Secure Watermark Detection Framework

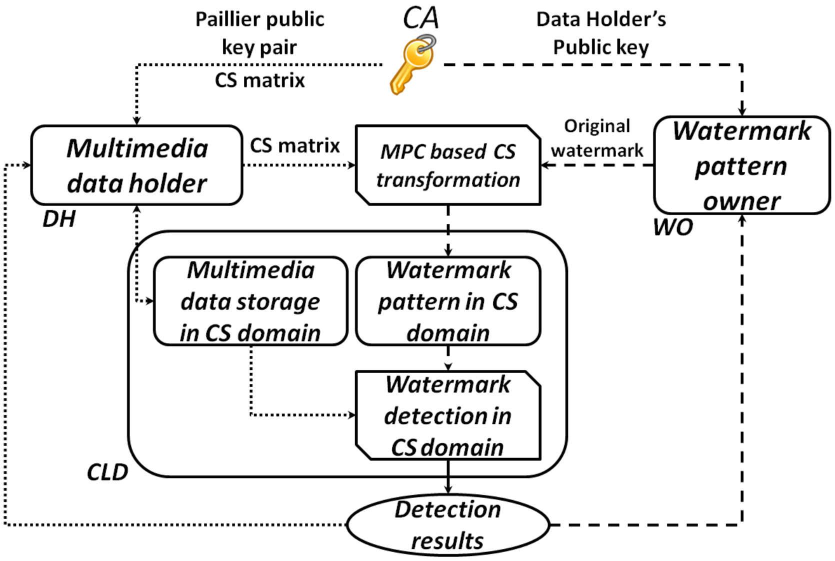


Figure 5.1. Architecture of the proposed secure watermark detection framework

Similarly with the generic framework shown in Figure 4.1 of Chapter 4, there are three parties in the proposed framework, the data holders (DH) of the potentially watermarked images, the watermark owners (WO) and the cloud (CLD) as illustrated in Figure 5.1. The secure watermark detection framework is developed under the semi-honest assumption as well. The framework also requires a certificate authority (CA) to issue the public keys and CS matrix keys to certain parties of the framework. For DH (e.g., media agencies), when it collects a large volume of multimedia data from the Internet and stores their encrypted versions in the CLD, it wants to make sure those multimedia can be edited and republished legally. Watermark owners (WOs) are also the content providers who distribute their watermarked content (the watermark embedding is performed by WO before the contents are published). WOs always want to know if their contents are legally used and republished.

In some scenarios, not only DH and WO care about the copyright of the multimedia data, certain CLD who offers storage services may also desire to initiate the watermark detection to check if the uploaded multimedia data is copyright protected. For example, a CLD may choose not to provide storage services to copyright protected data illegally owned. If DH would like to use a CLD for storage or migrate the encrypted multimedia data from another cloud to this CLD, it will require the CLD to perform watermark detection on the encrypted multimedia data before providing the storage services.

In our framework, initially, the CA needs to issue CS matrix suites to the DH. The CS matrix suites include the seeds and the random function used to generate the Gaussian CS matrix. We use the CA to issue the random function to guarantee the randomness of the generated Gaussian CS matrix (Our generic framework in Chapter 4 does not require CA because the CS matrix is hold by CSH only). The CA also needs to issue a Paillier public key pair to the DH and the DH's public key to the WO. The public key is used for the MPC based CS transformation protocol discussed in Protocol 3 from Section 4.5.2.

In general, the DH *also has a different private compressive sensing matrix (derived from the seed) for each image.* DH transforms the image's DCT coefficients to the compressive sensing domain and let CLD have the CS domain data for storage. If watermark detection with WO is required, we need to let CLD have the watermark in the same CS domain, which is achieved through running a secure multiparty protocol (Protocol 3: Secure CS transformation protocol introduced in Section 4.5.2) by DH, WO and CLD collaboratively under the semi-honest model [82]. Then CLD can detect if the watermark exists in the CS domain and let both DH and WO know the detection results. After Protocol 3 is executed, the compressive sensing matrix and the watermark pattern are still the secret values of the image holder and the watermark pattern owner respectively. In the framework, each CS matrix is used **only once** to encrypt the images' DCT coefficients, which is proven to be computationally secure in [88]. Note, however, that *a CS encrypted image will be stored and reused for secure detection of multiple watermarks on the same image.*

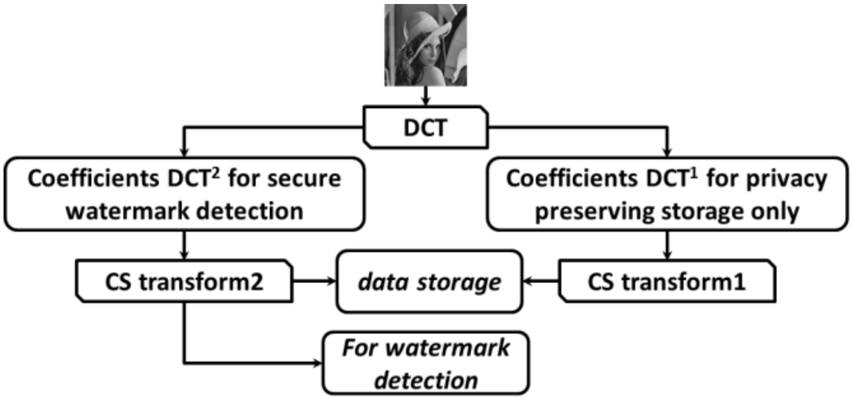
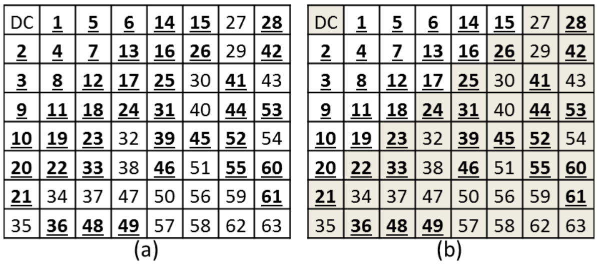

DCT
Coefficients DCT² for secure watermark detection
Coefficients DCT¹ for privacy preserving storage only
CS transform2
data storage
CS transform1
For watermark detection

Figure 5.2. Some DCT coefficients (DCT^1) are only used for storage; The other coefficients (DCT^2) serve for both the storage and watermark detection purposes



DC	**1**	**5**	**6**	**14**	**15**	27	**28**
2	**4**	**7**	**13**	**16**	**26**	29	**42**
3	**8**	**12**	**17**	**25**	30	**41**	43
9	**11**	**18**	**24**	**31**	40	**44**	**53**
10	**19**	**23**	32	**39**	**45**	**52**	54
20	**22**	**33**	38	**46**	51	**55**	**60**
21	34	37	47	50	56	59	**61**
35	**36**	**48**	**49**	57	58	62	63

(a)

DC	**1**	**5**	**6**	**14**	**15**	27	**28**
2	**4**	**7**	**13**	**16**	**26**	29	**42**
3	**8**	**12**	**17**	**25**	30	**41**	43
9	**11**	**18**	**24**	**31**	40	**44**	**53**
10	**19**	**23**	32	**39**	**45**	**52**	54
20	**22**	**33**	38	**46**	51	**55**	**60**
21	34	37	47	50	56	59	**61**
35	**36**	**48**	**49**	57	58	62	63

(b)

Figure 5.3. 8x8 DCT block zig-zag scan order: (a) an example of selective DCT coefficients for watermark embedding: the bold underlined numbers indicate that the corresponding DCT channels are embedded with watermark signals. (b) top 20 AC coefficients (white areas) are selected for watermark detection in the CS domain (i.e., DCT^2 in Figure 5.2).

In Zeng and Liu's work [80], the best watermark detection performance is achieved by using selective DCT coefficients for detection (i.e., select the potentially watermark embedded DCT channels for watermark detection as shown in Figure 5.3 (a)). However, the DH in our framework may not have the information about the criteria for the selection process. Even if we assume that the DH has such information, the DH needs to let the WO know which DCT channels are used for watermark detection. This will cause two issues for the framework. Firstly, the privacy issue: the image information of the DH might be leaked to the WO. Secondly, DH needs to send WO a large amount of data describing the selected DCT channels. To ensure the watermark detection performance, in our framework, DCT coefficients in the zig-zag order are split into two groups DCT^1 and DCT^2, as shown in Figure 5.2. DCT^1 (e.g., grey areas in Figure 5.3 (b)) includes the DC coefficients and potentially higher frequency AC coefficients, and DCT^2 (e.g., white areas in Figure 5.3 (b)) includes the lower frequency AC coefficients. The CS transformation of DCT^2 serves for both secure watermark detection and privacy preserving storage while DCT^1 serves for privacy preserving storage only. This is because the watermark detection performance in the CS domain will be penalized if the coefficients from DCT^1 are included. Note that in Zeng and Liu's work [80], most watermarks are embedded in the low- and mid- frequency AC coefficients, while DC coefficients and most of the higher frequency AC coefficients are not even selected for watermark embedding (as demonstrated in Figure 5.3 (a)). Including DC and higher frequency AC coefficients (none-watermark-carriers) will introduce noises for the watermark detection in the CS domain, as will be shown in our experimental results section. The DH needs to synchronize with WO about DCT^2 (e.g., Top 20 AC coefficients in the zig-zag order).

5.5. Framework Analysis

5.5.1. Complexity Analysis

Please refer to Section 4.6.1 for the complexity analysis of MPC based CS transformation protocol (Protocol 3). Note that the DH might be a computationally weak party such as a mobile device. In our framework, the complexity of DH is reduced when there are multiple watermark owners who are interested in performing watermark detection. When there are multiple watermark owners who are involved in performing watermark detection on an image, the data holder can send the public key encrypted CS matrix to the cloud. The watermark owners can get the public key encrypted CS matrix from the cloud to continue the secure CS transformation protocol. Then DH only needs to receive m public key encrypted values and decrypt them (Step 4) for every run of Protocol 3.

In a practical system, some trade-offs between complexity and security could be considered. When a user has many images, the same CS matrix could be used multiple times for multiple images so that a CS encrypted watermark pattern could be stored and reused for secure watermark detection on multiple images. However, the security level of the CS transformation encryption might be sacrificed due to multi-time use of the same CS matrix on different images, as discussed in Section 5.5.2.

5.5.2. Security Analysis

The security analysis of the MPC protocols has been given in Section 4.6.2 that the secure scalar protocol used in the framework is secure under the semi-honest security model. After running the secure CS transformation protocol, DH and WO do not leak their private values to other parties. Only the CLD has the image data and watermark pattern in the CS domain.

The security of using compressive sensing transformation as an encryption has been explored in [87][88] and it was concluded that it is computationally secure under the brute force and structured attacks when each CS matrix is used only one time. So if the data holder encrypts different images with different CS matrix keys, the CS domain data are secure in the cloud.

The security analysis of data-mining in CS domain has presented in Section 4.6.2.3: when multiple data in the same random projection domain (ciphertext only) are obtained by a third party, it gains no additional information other than the correlations between different data. The leakage of correlation between the image and watermark patterns is inevitable since we desire the CLD to provide such watermark detection services. Since the watermark patterns we use are i.i.d Gaussian, their corresponding CS domain versions are uncorrelated (but dependent on the same CS matrix). So the CLD cannot infer anything about the watermarks.

As discussed in Section 5.5.1, if a practical system chooses to use the same CS matrix for multiple images to reduce the secure watermark detection complexity, multiple image data in the same CS domain will be presented to the CLD simultaneously, in which case their dependencies on the same CS matrix may reveal the correlations and Euclidean distances ([39] shows the distance preserving property of the CS transformation) between the images, and may be used to reduce the attacking complexity. In addition, statistical information of the multimedia data is common sense knowledge [88], which may also be used to reduce the attacking complexity. The RIP property in Section 4.2.1 suggests that the CS transformation can preserve the energy of the original data, which means the overall energy of the image's DCT coefficients is leaked to the CLD. But this problem can be addressed, since the data holder can normalize all the original data vectors before outsourcing the CS encrypted data to the CLD, and the normalization will not affect the performance of the watermark detection. More analysis considering the above security concerns will be studied in our future work.

5.5.3. Comparison to Previous Works

When compared to previous works, our framework has the following advantages:

1. Our framework utilizes the computing and storage resource of the cloud simultaneously and provides better efficiency and flexibility as the encrypted image data (and the encrypted watermark pattern under some circumstances, if so chosen) can be reused for multiple watermark detections in the cloud.
2. Most of the existing secure watermark detection works paid little attention to the privacy of the multimedia data, while our framework protects the privacy of the self-collected data.

We compare the communication cost of our framework to other secure watermarking systems such as [78] under the following setting (chosen for the sake of complexity evaluation and comparison): the Paillier public key domain value takes 2048 bits; the image size is 1000x1; the secure CS transformation protocol is executed with a CS matrix size of 1000x1000 and a watermark size of 1000x1. We focus our evaluation on the communication cost of the data in the public key encryption domain, since it is the dominant factor over other communication cost. The watermark pattern is transformed to the CS domain in the initial step: DH needs to send WO around 256 MB of data (i.e., public key encrypted CS matrix) and WO needs to send 0.256 MB of data (i.e., public key encrypted watermark pattern) to DH. As analyzed above, when multiple images can be transformed to the same CS domain and the CS encrypted watermark pattern exists in the CLD, there is no communication cost between DH and WO for secure watermark detection. Note that for the previous secure watermarking systems such as [78], each time when secure watermark detection is performed on a new image, DH (corresponding to the verifier in [78]) has to work with WO (corresponding to the prover in [78]), which introduces computational and communication overhead for both DH and WO. As given in [78], the communication overhead to execute the zero-knowledge watermark detection protocol once is in the order of 1MB when the length of the watermark signal reaches 1000. Even though our framework has the semi-honest assumption that [78] is not constrained to, our framework has much better scalability and higher efficiency when performing secure watermark detection on a large number (e.g., thousands) of images. It is also very important to note that previous secure watermark detection methods [75][76][77][78][79] assume the multimedia data is available to all the parties and do not consider the privacy of the multimedia data. It might be possible to adapt those methods to protect the privacy of the multimedia data by encrypting it into a public key encryption domain. Then the computational and communication overhead will be increased significantly.

5.6. Experimental Results

5.6.1. Experimental Settings and Notations

We tested the proposed system using some standard 512x512 images. For the watermark detection, there are several detection methods proposed in [80]. We choose the one in which the watermark pattern used for watermark detection is directly generated from a Normal distribution $N(0, 1)$. Given a CS matrix $\Phi_{m\times n}$, m/n will be referred to as the compressive sensing rate ($CS\ rate$). Since the CS matrix size will be extremely large if we convert the 512x512 image to a vector for CS transformation. Instead, we cut the image into pieces and each piece contains 64 8x8 DCT blocks. Selective DCT coefficients of each piece will form a vector and be transformed to a CS domain with the same CS rate but using different CS matrixes. The data in the CS domain from all pieces is treated as $\{p_i\}$. Similarly, we get $\{r_i\}$ from the 512x512 original watermark pattern. We test the watermark detection performance when different numbers of DCT components are transformed to the CS domain as DCT^2 in Figure 5.2. In the rest of this section, “Top AC 20” means top 20 AC coefficients in the zigzag order are selected as DCT^2.

5.6.2. Scaling Floating Point to Integer Error Analysis

Since the MPC protocol is based on the Paillier public key system which requires integers as input, we scale the floating point values to integers with certain scaling factors. We test the error introduced by the conversion by comparing the result from secure CS transformation protocol to CS transformation with the original CS matrix and the watermark pattern. As shown in Table 5.1, the MSE decreases significantly as the scaling factor increases. In the following experiments, the scaling factor is set to 1.0e8.

Table 5.1. Mean square error w.r.t scaling factor

Scaling factor	1.E3	1.E4	1.E5	1.E6	1.E7	1.E8
MSE	2.24E-5	3.48E-7	3.12E-9	3.05E-11	3.31E-13	3.52E-15

5.6.3. Secure Watermark Detection in the Compressive Sensing Domain

5.6.3.1. Assertions validation

Table 5.2 summarizes the mean and variance of the sample covariance term in Equation (5.9) with different CS rates, under H_1 and H_0. The test result is based on several images including 512x512 'Lenna', 'Baboon', 'Barbara', 'Goldhill', 'Peppers' and etc. We can see that the covariance is very small and close to zero. However, it is interesting to see that under H1, the covariance term is concentrated around a very small *negative* value. This may suggest that the expected watermark detection output q^{cs} might be slightly lower than the β in Eq. (5.9). Table 5.3 summarizes some values for λ in Eq. (5.10) and γ in Eq.(5.11) using 'Lenna', when different DCT components are selected. It shows that the assertion $\boldsymbol{\lambda} > 0$ is true. Furthermore, if "Top AC 10" is chosen as DCT^2, γ is close to one, meaning that the CS transformation of such DCT channel coefficients will introduce nearly no distortion to the watermark detection. This is because almost all of the top 10 AC coefficients are selected for watermark embedding for the 'Lenna' image, while the distortions are mainly introduced by none-watermark-carriers that are mixed with watermark-carriers in the CS domain.

Table 5.2. Test results of the covariance term in Equation (5.9)

CS rate	1.0	0.7	0.4	0.1
H1(mean/ variance)	-3.3E-03/ 1.98E-06	-3.36E-03/ 2.5E-06	-3.71E-03/ 3.64E-06	-5.51E-03/ 7.4E-06
H0(mean/ variance)	1.69E-04/ 2.39E-06	8.59E-04/ 2.99E-06	-3.78E-03/ 5.09E-06	-1.63E-03/ 9.79E-06

Table 5.3. Validation for the assertion: $\lambda > 0$ (when $m = n$). (for the 512x512 'Lenna' image)

Coefficients	$\frac{1}{n}\overline{(x_a y_a)^2}$	$\frac{1}{n}\overline{(x_a y_b)^2}$	λ	γ
Top AC 63	0.0179	0.0879	0.0704	2.24
Top AC 40	0.0301	0.089	0.0602	1.75
Top AC 30	0.0367	0.0926	0.0577	1.62
Top AC 20	0.0486	0.0707	0.0243	1.21
Top AC 10	0.1582	0.1627	0.0092	1.02

5.6.3.2. Watermark detection in the CS domain

Figure 5.4 shows the watermark detection performance in the CS domain with different CS rates when different DCT components are selected. We give the watermark detection results (q under H1) in the original domain in Table 5.4. Table 5.4 and Figure 5.4 show that watermark detection in the CS domain has lower performance than in the original domain. The distortion is introduced by the CS transformation. Figure 5.4 also presents the estimated q^{cs} based on Eq. (5.9). We can see that the estimated q^{cs} and the tested real q^{cs} agree with each other very well. The estimated q^{cs} is calculated based only on the original signals and the CS rate, but not on the CS matrix used. It can be used as a reference to set a certain CS rate and achieve desired watermark detection performance in that CS domain. From Figure 5.4, we can see that when top 20 AC's are selected, the watermark detection performance is the best for the 512x512 'Lenna' image. This is because most of the watermarks are embedded in the top 20 AC coefficients and γ is relatively smaller as seen from Table 5.3. The one with all the 63 AC coefficients selected has lower q^{cs} value because the higher frequency DCT coefficients without watermark embedded will introduce noise to the watermark detection.

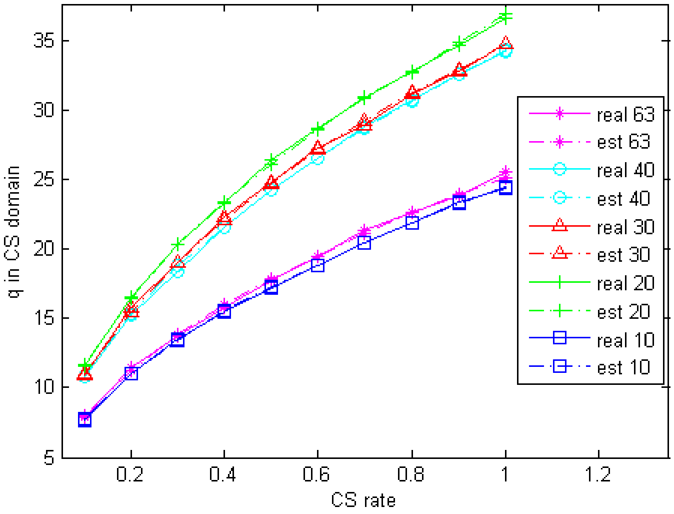


Figure 5.4. Watermark detection in the CS domain under different CS rates when different DCT components are selected. For the legend of the figure, "real" means the mean of q^{CS} calculated from $\{z_i\}$, while "est" means the estimated mean of q^{CS} based on Equation (5.9). The number "xx" means "Top AC xx". (for the 512x512 'Lenna' image)

Table 5.4. Watermark detection (q under H1) in original domain with different DCT coefficients. (for the 512x512 'Lenna' image)

Top AC 63	Top AC 40	Top AC 30	Top AC 20	Top AC 10
5.6E+01	6.1E+01	5.6E+01	4.5E+01	2.5E+01

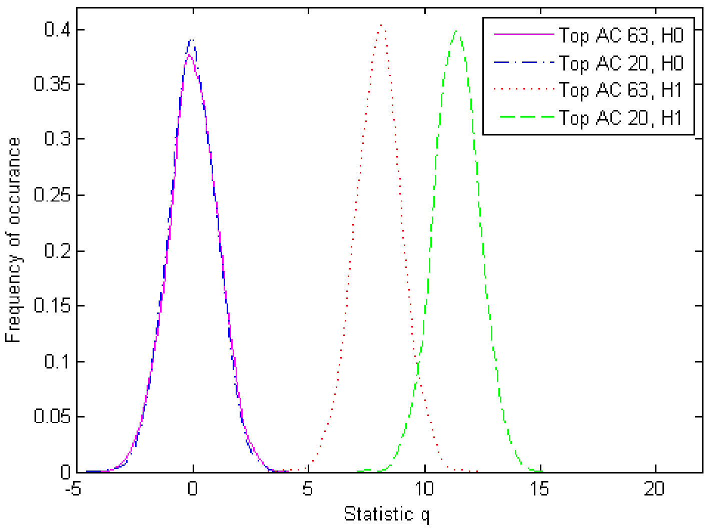


Figure 5.5. q^{cs}'s distribution under H_0 and H_1 with CS rate 0.1 using the top 20 AC coefficients as DCT^2 and the top 63 AC coefficients as DCT^2 (for the 512x512 'Lenna' image)

Figure 5.5 shows the distribution of the statistic q^{cs} with CS rate 0.1 by using the top 20 and 63 AC coefficients as DCT^2 for watermark detection for the 512x512 'Lenna' image. The figure shows that even with high dimension reduction, the watermark can still be detected.

We evaluate the watermark detection performance in the CS domain when both the watermark signals and certain noises are transformed to the CS domain simultaneously. Figure 5.6 shows the watermark detection performance in the CS domain when Gaussian noise (generated by the Gaussian random value generator in Matlab) is inserted into the test image. The figure shows that the watermark detection performance decreases only slightly even when the zero-mean Gaussian noise has a standard deviation of 40. The CS reconstruction will introduce distortion to the test image, which is referred to as CS reconstruction attack (e.g., labeled as "CS recons atk, 63" in Figure 5.6) for the watermark detection. We transform the top 63 (and 20) AC coefficients to a CS domain and perform watermark detection in the original domain after CS reconstruction. Figure 5.6 shows that when the CS rate is very low, the performance could be inferior to CS domain watermark detection, due to significant loss of information in the CS reconstruction process. Compared with “CS recons atk, 63”, “CS recons atk, 20” of Figure 5.6 shows that the watermark detection in the original domain after CS reconstruction is even lower than the CS domain across most CS rates. This is because most of the top 20 AC DCT channels are selected for watermark embedding and the CS reconstruction distortion to any of those channels will affect the watermark detection performance. However, the CS reconstruction distortion for “CS recons atk, 63” goes to the higher frequency DCT coefficients, most of which are not selected for watermark embedding.

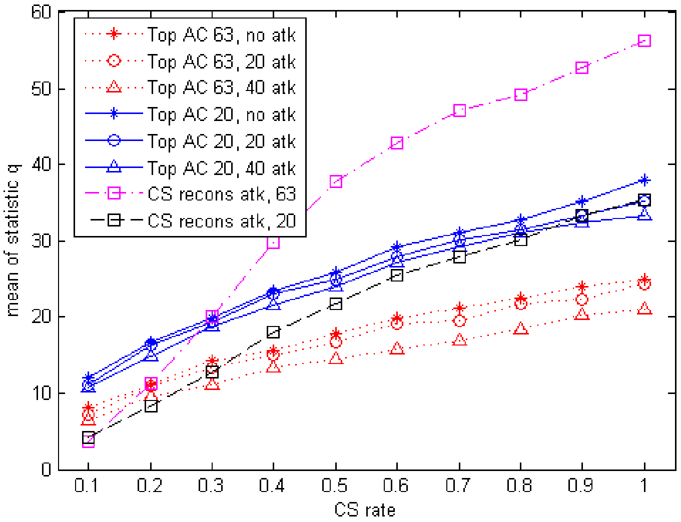


Figure 5.6. The mean of q^{cs} at different CS rates under zero-mean Gaussian noise attack (i.e. “20 atk” means Gaussian noise with zero mean and standard deviation 20); The mean of q in the original domain after CS reconstruction (i.e. if top 63 AC coefficients are selected for CS transformation, denoted as “CS recons atk, 63”). (for the 512x512 ‘Lenna’ image)

Table 5.5. Average q^{CS} and β for different images when different Top AC coefficients are chosen (CS rate = 0.1).

Image/q^{orig}		TopAC 63	TopAC 20	TopAC 10
‘Baboon’/ 1.16E+02	q^{cs}	3.26E+01	1.70E+01	9.93E+00
	β	3.29E+01	1.70E+01	9.97E+00
‘Barbara’/ 6.01E+01	q^{cs}	1.45E+01	1.24E+01	8.48E+00
	β	1.45E+01	1.25E+01	8.52E+00
‘Goldhill’/ 7.68E+01	q^{cs}	1.43E+01	1.56E+01	9.54E+00
	β	1.42E+01	1.56E+01	9.54E+00
‘Peppers’/ 5.57E+01	q^{cs}	8.49E+00	1.04E+01	6.68E+00
	β	8.41E+00	1.04E+01	6.64E+00

Table 5.5 and Figure 5.7 present the test results with 512x512 ‘Baboon’ image, ‘Peppers’ image and etc. Table 5.5 gives the watermark detection results q^{cs} in the CS domain and the estimated watermark detection performance β, when different images and different AC coefficients are selected with CS rate being 0.1. The first column also gives the watermark detection q in the original domain. The table shows that even with a CS rate of 0.1, the watermark can still be successfully detected for all the images when different AC coefficients are selected. Furthermore, the table also confirms our analysis in Section 2.3 that β in Eq. (5.9) can be used for accurately estimating the expected watermark detection performance in the CS domain. Figure 5.7 summarizes the watermark detection performance under different CS rates. The watermark detection performance in the CS domain for the ‘Baboon’ image is very good when top 63 AC coefficients are selected. This is because ‘Baboon’ is a highly textured image and many of its higher frequency DCT coefficients are also selected for watermark embedding.

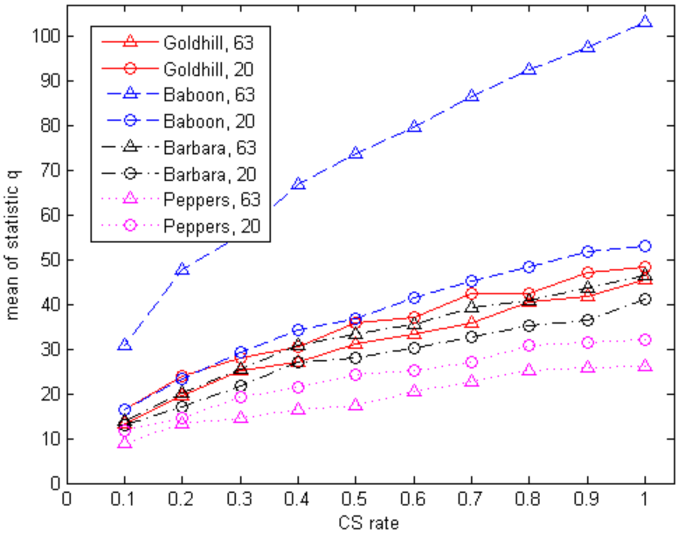


Figure 5.7. Watermark detection performance in the CS domain for other images (i.e. "Goldhill, 63" means Goldhill image with top 63 AC coefficients selected)

5.6.4. Compressive Sensing Encryption

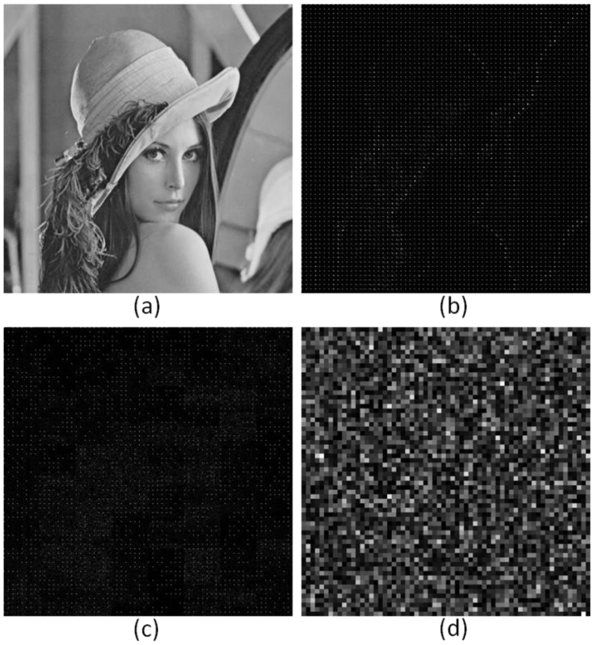


Figure 5.8: (a) Original image; (b) Image in 8x8 DCT domain; (c) DCT coefficients after CS transformation; (d) Image reconstruction with the wrong CS matrix. (CS rate 1.0 is chosen here, similar effects are observed under other CS rates)

Figure 5.8 shows the encryption results using the CS matrix as the encryption key. Figure 5.8 (d) shows that if a different CS matrix is used for the CS reconstruction, the reconstructed image is totally random. The block effect is due to the inverse-DCT operation on the 8x8 DCT block. If watch closely, it can be observed that Figure 5.8 (c) still preserves the spatial contour of Figure 5.8 (b) roughly. The reason is that the CS transformation in our experiment is performed piece-wisely as mentioned in Section 5.6.1 instead of treating the whole image as a single vector. As the RIP property given in Section 4.2.1 suggests, the CS transformation can preserve the energy of the original data. Such spatial contour similarity between DCT coefficients in the original domain and the CS domain can be removed by permuting the order of the pieces or by treating the whole image as a single vector.

5.6.5. Compressive Sensing Reconstruction

For privacy preserving storage, since the DCT coefficients are not perfectly sparse, the CS reconstruction will introduce distortion to the reconstructed image, especially when CS rate is low. The CS reconstruction error has been studied in many other works. Here we present our CS reconstruction experimental results when all AC components are transformed to a CS domain. In order to have a good quality image after the CS reconstruction, the CS rate needs to be high. Our experiments show that the PSNR (Peak Signal-to-Noise Ratio) is around 35 after the CS transformation/reconstruction process when the CS rate is 0.8, as shown in Figure 5.9. Even when the CS rate is set to 1.0, the

CS reconstruction algorithm (Orthogonal Matching Pursuit) still introduces distortion as we can see the PSNR is around 38. However, it should be noted that when the CS rate equals 1.0, the original DCT coefficients can be recovered perfectly given the inverse of the CS matrix, in which case CS reconstruction is not necessary.

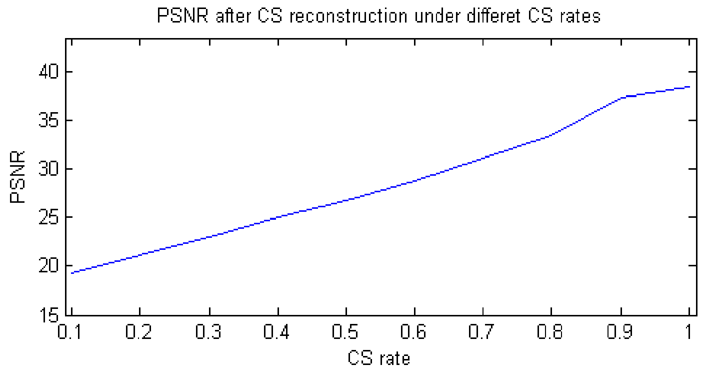


Figure 5.9. CS reconstruction distortion when top 63 AC coefficients are transformed to the CS domain

5.7. Conclusion and Future Works

This chapter proposes a compressive sensing based secure signal processing framework that enables simultaneous secure watermark detection and privacy preserving storage. Our framework is secure under the semi-honest adversary model to protect the private data. Note that without the semi-honest assumption, our framework will fail to protect the secret values. For example, collusion between WO and CLD will cause the leakage of DH's CS matrix. When compared to previous secure watermark detection protocols, our framework offers better efficiency and flexibility, and protects the privacy of the multimedia data that has not yet been considered in the previous works. We have demonstrated that secure watermark detection in the CS domain is feasible theoretically and experimentally. More theoretical analysis of the covariance term in Eq. (5.9) will be conducted in the future work. In addition to watermark detection, it would be interesting to explore the possibility of secure watermark embedding in the compressive domain. We show that the performance of the watermark detection in the compressive sensing domain could be estimated without the actual CS matrix used. Such an idea could be extended for other data-mining and signal processing algorithms in the CS domain as well. Future work also includes further evaluation of the robustness of the watermark detection in the CS domain under some other attacks.

Chapter 6. Use Case: A Privacy Preserving Collaborative Filtering Recommendation System for Mobile Users

In this chapter, we develop a privacy preserving collaborative filtering recommendation system for mobile users based on our mobile geo-tagging system proposed in Chapter 3 and the privacy preserving collaborative data-mining framework proposed in Chapter 4.

6.1. Introduction and Related Works

Recommender systems have become extremely common in recent years. Good personalized recommendations can add another dimension to the user experience, some of the largest e-commerce web sites have invested in high-quality recommender systems. Two well-known examples are the web merchant Amazon.com and the online movie rental company Netflix, which make the recommender system a salient part of their web sites. A majority of the recommender systems is based on the collaborative filtering techniques [44], which predict user preferences for products or services by learning past user-item relationships. The underlying assumption of the collaborative filtering approach, as given in [94], is that if a person A has the same opinion as a person B on an issue, A is more likely to have B's opinion on a different issue x than to have the opinion on x of a person chosen randomly.

Based on the collaborative filtering literature review article given by Su and Koshgoftaar [44], there are three types of collaborative filtering (CF) techniques: 1. Memory-based CF techniques, 2. Model-based CF techniques, and 3. Hybrid CF techniques.

Memory-based CF techniques systems use the user rating data to calculate the similarity or weight between users or items and make predictions or recommendations according to those calculated similarity values.

Model-based CF techniques use the pure rating data to estimate or learn a model to make predictions. The model can be a data mining or machine learning algorithm. Well-known model-based *CF* techniques include Bayesian belief nets (*BNs*) *CF* models, clustering *CF* models, and latent semantic *CF* models. An *MDP* (Markov decision process)-based *CF* system produces a much higher profit than a system that has not deployed the recommender.

Hybrid *CF* techniques combine *CF* and content-based techniques, hoping to avoid the limitations of either approach and thereby improve recommendation performance. Content-based filtering, besides collaborative filtering, is another important class of recommender systems. Content-based recommender systems make recommendations by analyzing the content of textual information and finding regularities in the content. The major difference between *CF* and content-based recommender systems is that *CF* only uses the user-item ratings data to make predictions and recommendations, while content-based recommender systems rely on the features of users and items for predictions.

One of the challenges [44] of the CF systems is the protection of the privacy when sharing the personal data. Canny [5] proposed solutions to protect users' privacy for *CF* recommendation tasks (i.e., mobile users who share their activity history to receive daily activity recommendations). Canny's method uses SVD to find an orthogonal space based on the aggregated data and projects the original data to such space for collaborative filtering. The similarity between users is determined by the representation of the users in the reduced space. Using such method requires continuously performing SVD for a large dimensional matrix first, which is computationally expensive and also introduces significant latency for the whole system.

In this chapter, we developed a mobile privacy preserving collaborative filtering system. In our system, mobile users can share their personal data (visit history or preference, e.g., restaurants, bars, landmarks, and etc.) with each other in the cloud and get daily activity recommendations based on the data-mining results generated by the cloud, without leaking the data privacy to other parties. Our mobile geo-tagging system could help improve the geo-tagging accuracy, i.e., users may record their visited or preferred places or landmarks by taking pictures of them from a remote spot. In such case, our mobile geo-tagging system could better reflect the user's intention, while using GPS reading only is biased.

The rest of this chapter is organized as follows. In Section 6.2, the detailed of the system is presented. Section 6.3 gives experimental results showing that our system is practicable and feasible. The chapter concludes in Section 6.4.

6.2. The System

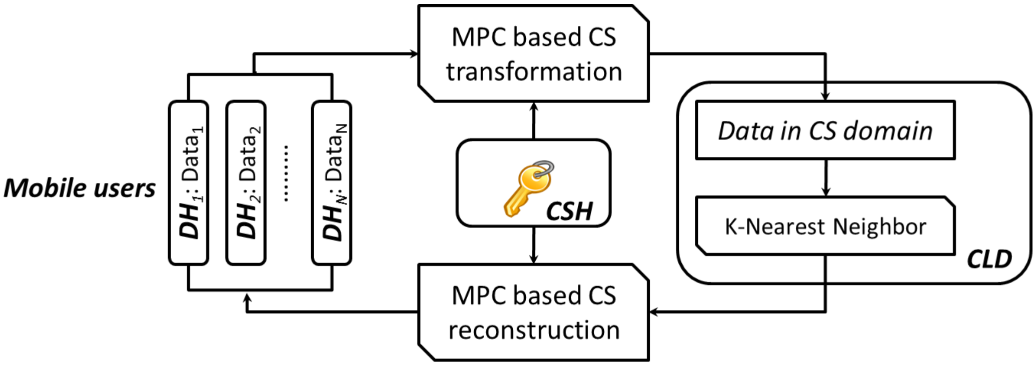


Figure 6.1. The architecture of the mobile privacy preserving collaborative filtering system.

Table 6.1. An example of user's visiting history.

Users	Location1	Location2	Location3	……	LocationN
User1	0	3	8	……	0
User2	1	0	0	……	0
User3	0	5	3	…....	0

The architecture of the system is given in Figure 6.1, which is generated from our generic framework in Figure 4.1. All the DHs (data holders) are mobile users who use their smartphone's sensors (GPS) or our proposed geo-tagging system to get their visited histories or the locations they like. The data will be formed as a vector including all the places, an example of which is shown in Table 6.1. In Table 6.1, each user's data vector entries record how many times such user visits the corresponding location. DHs work with CSH (compressive sensing matrix holder) to transform their personal data vector to the CLD (cloud) through MPC protocols given in Section 4.5. CLD will perform data-mining algorithm to the data to generate the recommendation in the CS domain. Then the CSH and CLD will send the recommendation in original domain to the DH through MPC based CS reconstruction protocol as given in Section 4.5. In our system, the K-nearest neighbor algorithm is chosen to generate recommendations, because it is one of the pre-dominant approaches used for collaborative filtering [96].

The system's security is based on the security of our generic framework, the security analysis of which has been given in Section 5.5. The analysis shows that the potential information leak will happen when the MPC based CS reconstruction protocol is executed, i.e., CSH may be able to guess the structure (the ranking of the entries based on the entry values) of the original vector. The complexity of the system can be easily derived from the complexity analysis of the generic framework in Section 5.5 as well.

6.3. Experimental Results

In our experiment, MovieLens 1M Dataset [62] (a benchmark dataset used for testing collaborative filtering techniques) is used for our simulation. Since its data structure is close to the data vector structure required in our system and its way of collecting users' ratings is similar with how our system will get users' preferences. We select 1578 sparse vectors from MovieLens Dataset with vector size 1000x1. Sparse data vectors are chosen in that the mobile users' vector data is sparse (because user's visited place is typically a small portion of an area). In the experiments, the scaling factor is set to 1.0e8 for the MPC protocols. A recommendation for a certain user is calculated based on its 10 nearest neighbors' centroid. We give the recommendations based on the data-mining results in the CS domain and compare them with the results in the original domain as shown in Table 6.2. In Table 6.2, the relative error is given as:

$$error = \frac{||V|_2^2 - |K|_2^2|}{|V|_2^2}$$

where both V and K are in the original domain. V is the recommendation result generated in the original domain, and K is the CS reconstruction result from the recommendation generated in the CS domain under difference CS rates. We can see that the recommendation results in the CS domain are very close to the original domain. The communication cost for our system is given in Table 6.3 under the CS rate 1.0 (chosen because it generates the highest communication cost). The table only gives the dominant communication cost introduced by public key domain values. The setting for public key systems is the same with Section 4.8.2. The initial step in which CSH sends DH the public key encrypted CS matrix (i.e., 256MB with the current setting) is not included in the table since it could be reused. From Table 6.3, we can see that during the CS transformation, DH only needs to send out 0.255MB data. During the reconstruction process, the communication cost between CSH and CLD is high while the cost for DH is negligible (no communication cost in the public-key domain). In our system, we assume CSH and CLD are computationally powerful parties but DHs are mobile users who have lower computational power. We can see from Table 6.2 and Table 6.3 that our system is feasible and practicable in enabling privacy preserving collaborative filtering services for mobile users.

Table 6.2. Relative errors of the recommendation results in CS domain w.r.t the original domain.

Compressive sensing rate	Relative error (%)
1.0	0.6
0.9	0.6
0.8	0.8
0.7	0.9
0.6	1.2
0.5	1.6
0.4	1.8

Table 6.3. Communication cost of the proposed system (data in public key domain only).

From	To	CS transformation (MB)	CS reconstruction (MB)
CSH	DH	0	0
DH	CSH	0.255	0
DH	CLD	0	0
CLD	DH	0	0
CLD	CSH	0	197
CSH	CLD	0	292

6.4. Conclusions

In this chapter, a privacy preserving collaborative filtering system for mobile users is proposed. Such system is enabled by both of our mobile geo-tagging system and our generic privacy preserving data-mining framework. Our experimental results based on the simulation with real-world data show that our system is feasible and practicable.

Chapter 7. Conclusions and Future Works

In recent years, with the off-the-shelve mobile devices and the cloud computing services getting popular, mobile cloud computing has drawn increasing attentions from the industrial and academic communities. There are a great deal of opportunities to be explored and problems to be solved in the mobile cloud computing area. Our research focuses on developing practical systems by taking advantage of the on-board computing capability and sensors of the mobile devices and proposing solutions to protect the privacy when personal data is shared in the cloud for collaborative data-mining. In this dissertation, we have developed mobile geo-tagging system and privacy preserving collaborative data-mining frameworks and systems [6][7][8][9]. In this chapter, we conclude with a summary of our contributions and discuss future works.

7.1. Summary of the Dissertation

Our proposed frameworks and systems including their contributions are summarized in the following.

- Mobile geo-tagging system

Mobile multimedia and vision systems and mobile location based systems have received significant attentions in recent years. Current mobile vision-based geo-tagging systems in the literature are used to locate the device it-self but not a remote target and they have to rely on a remote server for computationally heavy tasks.

In this dissertation, we propose an image/video based remote target localization and tracking system. In our system, all the computational tasks are performed on the mobile devices only. To cope with smartphone's computationally weak limitation, we propose optimization methods that are tailored to the unique characteristics of the smartphone's computing platform.

Our system is first of its kind and we provide the first hand real-world experimental results showing that our system is feasible and practicable. Our system can be applied in various scenarios including military and commercial applications.

- Generic privacy preserving collaborative data-mining framework

Due to the necessity of collaborative data-mining applications and services, privacy is becoming a critical concern when users share their data with other parties. Previous privacy preserving data-mining methods or systems suffer from either lower security level or lower scalability and flexibility.

In this dissertation, we propose a generic compressive sensing based privacy preserving collaborative data-mining framework using secure multiparty computation. In our framework, the secure multiparty computation is applied to the compressive sensing transformation and reconstruction processes and the data-mining is performed in the compressive sensing domain, so that the CS transformed data could be stored in the cloud securely for reuse for various types of data-mining tasks, e.g., k-means clustering, k-nearest neighbor, and etc. Our analysis and experimental results have demonstrated that our system is viable in enabling privacy preserving data-mining tasks.

- Privacy preserving storage and secure watermark detection framework

Secure watermark detection is an important field, however there are no new approaches and solutions to such problem for years. Previous secure watermark detection methods only protect the watermark pattern but not the target image and they are constrained by the significant computational and communication complexity.

In this dissertation, we identify an application scenario that also requires protecting the privacy of the target image during the secure watermark detection process. We propose a secure watermark detection framework based on our generic privacy preserving data-mining framework. Our framework provides privacy protection for both watermark pattern and the target image with better flexibility and efficiency compared to previous works.

We show that secure watermark detection in the compressive sensing domain is feasible based on our theoretical analysis and experimental results. We derive the expected watermark detection performance in the compressive sensing domain, given the target image, watermark pattern and the size of the compressive sensing matrix (but without the CS matrix used). This is important since the performance can be estimated during the watermark embedding process by the content provider.

- Mobile privacy preserving collaborative filtering system

Based on our mobile geo-tagging system and our generic privacy preserving framework, in this dissertation, we propose a mobile privacy preserving collaborative filtering system. Our mobile geo-tagging system is introduced to facilitate more accurate user personal data for better collaborative filtering recommendations. Our generic privacy preserving framework is used for protecting the data privacy of mobile users while providing data-mining services (i.e., k-nearest neighbor algorithm based recommendation service) through cloud. Our experimental results based on the simulation with real-world data show that our system is feasible and practicable.

7.2. Future Works

Details of the future works of our proposed systems and frameworks have been discussed in each corresponding chapter respectively. In this section, we summarize the future works in two aspects: developing more practical mobile geo-tagging systems and designing more efficient and secure privacy preserving data-mining frameworks.

- Developing more practical mobile geo-tagging systems

Future work could include further system accuracy improvement and complexity reduction, e.g., 1. the accuracy of the two-image based localization system could be improved by introducing more images of the remote target from different perspectives; 2. computationally heavy tasks (video tracking and image feature extraction & matching) can be dispatched to several smartphones and get processed in parallel; 3. on-board GPU programming of mobile devices can also facilitate higher computational efficiency for our system. Providing more user-friendly interface opens a direction for our system's future work as well, e.g., in the single-image based localization, drawing bounding boxes enclosing the object tightly by a user could be eased by taking advantage of image segmentation techniques so that the *tight* bounding box could be generated automatically based on the image segmentation results (i.e., the side of the box could be aligned automatically to the object's segment boundary).

- Designing more efficient and secure privacy preserving data-mining frameworks

We have already demonstrated that many data-mining and signal processing tasks can be securely performed in the compressive sensing domain. Future work could include developing efficient distributed data-mining algorithms in the cloud. The public key based MPC protocols introduce significant computational and communication overheads. Future work could include complexity reduction for the MPC protocols, especially for the computationally weak data holders (e.g., mobile users). For example, the MPC protocols could be designed in a way that some public key operations of the data holders can be shifted to the cloud. Future work also includes security analysis of the MPC CS reconstruction process in which the ordering information is leaked to the CS matrix holder. Two approaches might be helpful to resolve such potential security issue: introducing an additional party to the MPC CS reconstruction protocol or eliminating the security leak step of the protocol by modifying the CS reconstruction algorithm.

APPENDIX I

Proof of Lemma 2 in Section 5.3:

An equivalent way to interpret the second formula from Lemma 1 [39] is:

$$Var[P^T R|X,Y] = \frac{1}{m}(\sum_1^n x_i^2 \sum_1^n y_i^2 + (\sum_1^n x_i y_i)^2)$$

Since given X and Y, $\{p_i\}$ and $\{r_i\}$ are both i.i.d. Gaussian, then $z_i^{cs} = p_i r_i$ are i.i.d., therefore:

$$Var(z_i^{cs}|X,Y) = \frac{1}{m} Var[P^T R|X,Y] = \frac{1}{m^2}\ ((\sum_{i=1}^n x_i y_i)^2 + \sum_{i=1}^n x_i^2 \sum_{i=1}^n y_i^2)$$

■

The proof above is based on Lemma 1 in Section 4.3.3. However, the link of the proof for Lemma 1's second equation is invalid in [39]. We give our proof of Lemma 2 in the following.

$$E\left(V_{z_i^{cs}}^2|X,Y\right) = Var(z_i^{cs}|X,Y)$$

$$= Var[\left(\sum_{j=1}^n \Phi_{i,j}\, x_j\right) * (\sum_{j=1}^n \Phi_{i,j}\, y_j)]$$

$$= \sum_{a=1}^n \sum_{b=1}^n \sum_{c=1}^n \sum_{d=1}^n Cov[\Phi_{i,a}\Phi_{i,b} x_a y_b, \Phi_{i,c}\Phi_{i,d} x_c y_d]$$

$$= \sum_{a=1}^n \sum_{b=1}^n \sum_{c=1}^n \sum_{d=1}^n x_a y_b x_c y_d Cov\left[\Phi_{i,a}\Phi_{i,b}, \Phi_{i,c}\Phi_{i,d}\right]$$

There are three different values for $Cov\left[\Phi_{i,a}\Phi_{i,b}, \Phi_{i,c}\Phi_{i,d}\right]$:

1. When $a = b = c = d$:

Based on the moment-generating function of a Gaussian distribution, it is easy to derive the second and fourth moments of $\Phi_{i,a} \sim N(0, \frac{1}{m})$ as:

$$E\left(\Phi_{i,a}^2\right) = \frac{1}{m},\ E\left(\Phi_{i,a}^4\right) = \frac{3}{m^2}$$

Then:

$$Cov\left[\Phi_{i,a}\Phi_{i,a},\Phi_{i,a}\Phi_{i,a}\right]=Var\left(\Phi_{i,a}^2\right)=E\left(\Phi_{i,a}^4\right)-E^2\left(\Phi_{i,a}^2\right)=\frac{2}{m^2}$$

2. When $(a=c,b=d,a\neq b)$ or $(a=d,b=c,a\neq b)$:

$$Cov\left[\Phi_{i,a}\Phi_{i,b},\Phi_{i,a}\Phi_{i,b}\right]=Var\left(\Phi_{i,a}\Phi_{i,b}\right)$$

$$=Var\left(\Phi_{i,a}\right)*Var\left(\Phi_{i,b}\right)=\frac{1}{m^2}$$

3 Without loss of generality, when a is not equal with any of b,c,d:

$$Cov\left(\Phi_{i,a}\Phi_{i,b},\Phi_{i,c}\Phi_{i,d}\right)$$

$$=E\left(\Phi_{i,a}\Phi_{i,b}\Phi_{i,c}\Phi_{i,d}\right)-E\left(\Phi_{i,a}\Phi_{i,b}\right)E\left(\Phi_{i,c}\Phi_{i,d}\right)$$

$$=E\left(\Phi_{i,a}\right)E\left(\Phi_{i,b}\Phi_{i,c}\Phi_{i,d}\right)-E\left(\Phi_{i,a}\right)E\left(\Phi_{i,b}\right)E\left(\Phi_{i,c}\Phi_{i,d}\right)=0$$

Then we have:

$$x_ay_bx_cy_dCov\left[\Phi_{i,a}\Phi_{i,b},\Phi_{i,c}\Phi_{i,d}\right]=$$

$$\begin{cases}\frac{2*x_a^2y_a^2}{m^2}, & if\ a=b=c=d\\ \frac{x_ay_bx_ay_b}{m^2}, & if\ (a=c,b=d,a\neq b)\\ \frac{x_ay_bx_by_a}{m^2}, & if\ (a=d,b=c,a\neq b)\\ 0, & otherwise\end{cases}$$

Then:

$$\sum_{a=1}^{n}\sum_{b=1}^{n}\sum_{c=1}^{n}\sum_{d=1}^{n}x_ay_bx_cy_dCov\left[\Phi_{i,a}\Phi_{i,b},\Phi_{i,c}\Phi_{i,d}\right]$$

$$=\frac{1}{m^2}(2\sum_{a=1}^{n}(x_ay_a)^2+\sum_{b=1,b\neq a}^{n}\sum_{a=1}^{n}x_ay_bx_ay_b\ +\sum_{b=1,b\neq a}^{n}\sum_{a=1}^{n}x_ay_ax_by_b)$$

$=$

$\frac{1}{m^2}\big(\,\big(\sum_{a=1}^{n}(x_a y_a)^2 +$

$\sum_{b=1,b\neq a}^{n}\sum_{a=1}^{n} x_a y_a x_b y_b\big) + \big(\sum_{a=1}^{n}(x_a y_a)^2 + \sum_{b=1,b\neq a}^{n}\sum_{a=1}^{n} x_a y_b x_a y_b\big)\,\big)$

$= \frac{1}{m^2}\ \big(\sum_{b=1}^{n}\sum_{a=1}^{n} x_a y_a x_b y_b + \sum_{b=1}^{n}\sum_{a=1}^{n} x_a y_b x_a y_b\big)$

$= \frac{1}{m^2}\ \big(\sum_{a=1}^{n} x_a y_a \sum_{a=1}^{n} x_a y_a + \sum_{a=1}^{n} x_a x_a \sum_{a=1}^{n} y_a y_a\big)$

Therefore:

$E\left(V_{z_i^{cs}}^2 | X, Y\right) = \frac{1}{m^2}\ \big((\sum_{i=1}^{n} x_i y_i)^2 + \sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i^2\big)$ ■

BIBLIOGRAPHY

[1] H. Dinh, C. Lee, D. Niyato and P. Wang. “A Survey of Mobile Cloud Computing: Architecture, Applications, and Approaches”, in proceedings of *Wireless Communications and Mobile Computing Conference,* 2012.

[2] N. Fernando, S. W. Loke and W. Rahayu. “Mobile cloud computing: A survey”, *Future Generation Computer Systems*, Vol. 29, 2013, pp. 84-106.

[3] G. Huerta-Canepa, D. Lee. “A virtual cloud computing provider for mobile devices”, in *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing&Services:Social Networks and Beyond*. MCS’10, ACM, New York, NY, USA, 2010, pp. 6:1-6:5.

[4] E.E. Marinelli, “Hyrax: cloud computing on mobile devices using MapReduce”, *Master Thesis*, Carnegie Mellon University, 2009.

[5] J. Canny. “Collaborative Filtering with Privacy”, *IEEE Symposium on Security and Privacy*, pp. 45-57, 2002.

[6] Q. Wang, W. Zeng, and A. G. Lobzhanidze, “Mobile media in action: remote target localization and tracking,” *IEEE Multimedia Magazine*, vol. 19, no. 3, July-Sept, 2012.

[7] Q. Wang, A. Lobzhanidze, H.I. Jang, W. Zeng, and Y. Shang, “Video based Real-world Remote Target Tracking on Smartphones,” *IEEE Inter. Conf. on Multimedia and Expo main conference*, July 2012.

[8] Q. Wang, A. Lobzhanidze, S. Roy, W. Zeng and Y. Shang, “PositionIt – An Image-based Remote Target Localization System on Smartphones”, demo, *ACM Multimedia Conference*, Nov 2011.

[9] Q. Wang, W. Zeng and J. Tian, “Integrated Secure Watermark Detection and Privacy Preserving Storage Framework in the Compressive Sensing Domain,” *IEEE International Workshop on Information Forensics and Security*, Nov 2013.

[10] B. D. Lucas and T. Kanade, “An interactive image registration technique with an application to stereo vision”. *Proc. of Image Understanding Workshop*, pp. 121-130, 1981.

[11] Z. Zhang, “A flexible new technique for camera calibration, ” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1330-1334, 2000.

[12] Gary Bradski, Adrian Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media. ISBN-13: 978-0596516130, 2008.

[13] L. M. Kaplan, “Global Node Selection for Localization in a Distributed Sensor Network,” *IEEE Transactions on Aerospace and Electronic Systems*, vol.42, pp.113-135, 2006.

[14] T.M. Cover and P.E. Hart, “Nearest neighbor pattern classification”. *IEEE Transactions on Information Theory*, vol.13 (1): pp 21–27, 1967.

[15] R. I. Hartley, “In Defense of the Eight-Point Algorithm”, *IEEE Transaction on Pattern Recognition and Machine Intelligence*, vol.19, pp. 580–593, 1997.

[16] R. I. Hartley and P. Sturm, “Triangulation”, *Comput. Vision Image Understand.*, 68-2 (1997-11),146–157.

[17] H. Bay, A. Ess, T. Tuytelaars, L. V. Gool, “SURF: Speeded Up Robust Features”, *Computer Vision and Image Understanding (CVIU)*, vol. 110, No. 3, pp. 346-359, 2008.

[18] R. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. *Cambridge University Press*. ISBN 978-0-521-54051-3, 2003.

[19] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for modeling fitting with applications to image analysis and automated cartography”, *Comm. of ACM*, vol. 24, Issue 6, 1981, pp. 381-395.

[20] http://sourceforge.net/projects/opencvlibrary/files/opencv-android/2.3.1

[21] R. Sim and G. Dudek, “Comparing image-based localization methods”, in *Proceedings of the 18th international joint conference on Artificial intelligence*, pp. 1560-1562, 2003.

[22] W. Zhang and J. Kosecka, “Image Based Localization in Urban Environments”, in *Proceedings of the Third International Symposium on 3D Data Processing, Visualization, and Transmission*, pp. 33-40, 2006.

[23] J. Kosecka, L. Zhou, P. Barber and Z. Duric, “Qualitative Image Based Localization in Indoors Environments”, in *Proceedings of Computer Vision and Pattern Recognition*, vol. 2, pp. 3-8, 2003.

[24] M. Werner, M. Kessel and C. Marouane, “Indoor Positioning Using Smartphone Camera”, *International conference on Indoor Positioning and Indoor Navigation*, 2011.

[25] W. Waqar, Y. Chen and A. Vardy, “Exploiting Smartphone Sensors for Indoor Positioning: A Survey”, the *20th Annual Newfoundland Electrical and Computer Engineering Conference (NECEC)*, 2011.

[26] F. X. Yu, R. Ji, and S.-F. Chang. "Active query sensing for mobile location search". In *Proc. of ACM International Conference on Multimedia (ACM MM)*, 2011.

[27] J. Roters, X. Jiang and K. Rothaus, "Recognition of traffic lights in live video streams on mobile devices". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, pp. 1497-1511, 2011.

[28] H. Hile, R. Grzeszczuk, A. Liu, R. Vedantham, J. Kosecka, and G. Borriello. "Landmark-Based Pedestrian Navigation with Enhanced Spatial Reasoning". In *7th International Conference on Pervasive Computing*. Springer, 2009.

[29] G. Takacs, et. al, "Outdoors augmented reality on mobile phone using loxel-based visual feature organization", in *Proceedings of the 1st ACM Inter. Conf. on Multimedia information retrieval*, pp. 427-434, 2008.

[30] J. Carmigniani, B. Furht, M. Anisetti, P. Ceravolo, E. Damiani and M. Ivkovic, "Augmented reality technologies, systems and applications", *Multimedia Tools and Applications*, vol.51, pp. 341-377, 2011.

[31] S. Weng, C. Kuo and S. Tu, "Video object tracking using adaptive Kalman filter," *Journal of Visual Communication and Image Retrieval*, vol. 17, pp. 1190-1208, 2006.

[32] M. Roh, T. Kim, J. Park and S. Lee, "Accurate object contour tracking based on boundary edge selection", *Pattern Recognition*, vol. 40, pp. 931-943, 2007.

[33] R. Rosales and S. Sclaroff, "3D Trajectory Recovery for Tracking Multiple Objects and Trajectory Guided Recognition of Actions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog*, vol.2, pp. 117-123, 1999.

[34] F. Xu, K. Lam and Q. Dai, "Video-Object segmentation and 3D-trajectory estimation for monocular video sequences," *Journal of Image and Vision Computing*, vol. 29, pp.190-205, 2011.

[35] L. M. Kaplan, "Global Node Selection for Localization in a Distributed Sensor Network," *IEEE Transactions on Aerospace and Electronic Systems*, vol.42, pp.113-135, 2006.

[36] D. F. Dementhon and L. S. Davis, "Model-based object pose in 25 lines of code," *Inter. Journal of Computer Vision*, vol. 15, pp. 123-141, 1995.

[37] A. Vedaldi and S. Soatto, "Quick shift and kernel methods for mode seeking", in *Proceeding of European Conference on Computer Vision*, 2008.

[38] X. Ren and J. Malik, "Learning a classification model for segmentation", in *Proc. 9th Int. Conf. Computer Vision*, volume 1, pages 10-17, 2003.

[39] K. Liu, H. Kargupta and J. Ryan. “Random Projection-Based Multiplicative Data Perturbation for Privacy Preserving Distributed Data Mining”, *IEEE Transactions on Knowledge and Data Engineering*, vol.18, No.1, pp. 92-106, 2006.

[40] Y. Lindell and B. Pinkas. “Privacy Preserving Data Mining”, in Proceedings of the *ACM SIGMOD international conference on Management of Data*, Vol. 29, Issue 2, pp.439-450, 2000.

[41] S. Dasgupta. “Experiments with Random Projection”, in *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pp.143-151, 2000.

[42] R. Agrawl and R. Srikant. “Privacy-Preserving Data Mining”, in *Proceedings of the ACM SIGMOD international conference on Management of data*, vol. 29, Issue 2, pp. 439-450, 2000.

[43] B. Pinkas. “Cryptographic techniques for privacy-preserving data mining”, *ACM SIGKDD Explorations Newsletter*, Vol. 4 Issue 2. pp. 12-19, 2002.

[44] X. Su and T. M. Khoshgoftaar. “A Survey of Collaborative Filtering Techniques”, *Journal of Advances in Artificial Intelligence*, Vol. 2009, No.4, 2009.

[45] M. Doganay, T. Pedersen, Y. Saygin, E. Savas and L. Levi. “Distributed Privacy Preserving k-Means Clustering with Additive Secret Sharing”, in *Proceedings of international workshop on Privacy and anonymity in information society*, pp. 3-11, 2008.

[46] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. “A Simple Proof of the Restricted Isometry Property for Random Matrices”, *Constr. Approx.*, 28(3):253-263, 2008.

[47] R. Calderbank, S. Jafarpour and R. Schapire. “Compressed Learning: Universal Sparse Dimensionality Reduction and Learning in the Measurement Domain”. *Preprint*, 2009.

[48] Y. Rachlin and D. Baron. “The Secrecy of Compressed Sensing Measurement”, *46th Annual Allerton Conference on Communication, Control, and Computing*, pp. 813-817, 2008.

[49] M. Mayiami, B. Seyfe and H. Bafghi. “Perfect Secrecy Using Compressed Sensing”, *arXiv:1011.3985v1*, 2010.

[50] Y. Weiss, H. S. Chang and W. T. Freeman. “Learning Compressed Sensing”. in *Proc. Allerton Conf. Commun. Control Comput.*, 2007.

[51] D. Hsu, S. M. Kakade, J. Langford and T. Zhang. “Multi-Label Prediction via Compressed Sensing”, In *Neural Information Processing Systems (NIPS)*, 2009.

[52] W. Du and M. Atallah. “Secure Multi-Party Computation Problems and Their Applications: A Review and Open Problems”, in *Proceedings of workshop on New security paradigms*, pp.13-22, 2001.

[53] H. Lin and W. Tzeng. “An Efficient Solution to The Millionaires’ Problem Based on Homomorphic Encryption”, in *Proceedings of the Third international conference on Applied Cryptography and Network Security*, pp. 456-466, 2005.

[54] A. C. Yao. “Protocols for Secure Computations”, In *Proceedings of 23th Annual Symposium on Foundations of Computer Science*, pp. 160-164, 1982.

[55] K. Peng, C. Boyd, E. Dawson, and B.Lee. “An Efficient and Verifiable Solution to the Millionaire Problem”, *Lecture notes in computer science*, Vol. 3506, pp. 51-66,2004.

[56] F. Xu, S. Zeng, S. Luo, C. Wang, Y. Xin and Y. Guo. “Research on Secure Scalar Product Protocol and its Application”, *International Conference on Wireless Communications Networking and Mobile Computing*, 2010.

[57] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin and M. Zhu, “Tools for Privacy Preserving Distributed Data Mining”, *ACM SIGKDD Explorations Newsletter*, vol.4, Issue 2, pp. 28-34, December 2002.

[58] F. Krahmer, R. Ward. “New and improved Johnson-Lindenstrauss embeddings via the Restricted Isometry Property”, *SIAM J. Math. Anal*. Vol.43, pp. 1269-1281, 2010.

[59] G. Nayak and S. Devi. “A Survey on Privacy Preserving Data Mining: Approaches and Techniques”, *International Workshop on Database Technology and Applications*, 2009.

[60] J. Yang, A. Bouzerdoum, F. Tivive and S. Phung. “Dimensionality Reduction using Compressed Sensing and its Application to a Large-Scale Visual Recognition Task”, *International Joint Conference on Neural Networks (IJCNN)*, 2010.

[61] Bruce Schneier, Applied Cryptography, *John Wiley & Sons*, ISBN: 0-471-11709-9, 1996.

[62] MovieLens data, http://www.grouplens.org/node/73.

[63] E. Candes, J. Romberg, and T. Tao. “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information,” *IEEE Transactions on Information Theory* 52 (2006), no. 2, pp. 489-509.

[64] D. Donoho, “Compressed Sensing”, *IEEE Transaction on Information Theory* 52 (2006), no. 4, pp.1289-1306.

[65] W. Johnson and J. Lindenstrauss, "Extensions of Lipschitz mappings into a Hilbert space", *Contemp. Math*, 26:189-206, 1984.

[66] S. Mendelson, A. Pajor and N.Tomczak-Jaegermann. "Uniform uncertainty principle for Bernoulli and subgaussian ensembles," *Constructive Approximation*, 28(3):277-289, 2008.

[67] M. Rudelson and R. Vershynin. "Sparse reconstructions by convex relaxation: Fourier and Gaussian measurements," in *Proc. Conference on Information Sciences and Systems*, 2006.

[68] S. Mallat and Z. Zhang. "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, 41(12):3397-3415, 1993.

[69] J. Vaidya and C. Clifton. "Privacy-preserving k-means clustering over vertically partitioned data." In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 206-215. 2003.

[70] UCI KDD Dataset, http://kdd.ics.uci.edu/.

[71] T. ElGamal. "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," *IEEE Trans. on Information Theory*, 31: 469-472, 1985.

[72] P. Paillier. "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes," In *Advances in Cryptology-Eurocrypt*, pp. 223-238, 1999.

[73] T. Bianchi and A. Piva, "Secure watermarking for multimedia content protection: A review of its benefits and open issues," *IEEE Signal Processing Magazine*, vol. 30, issue: 2, pp. 87-96, 2013.

[74] Z. Erkin, A. Piva, S. katzenbeisser, R. Lagendijk, J. Shokrollhi, G. Neven, and M. Barni, "Protection and retrieval of encrypted multimedia content: when cryptography meets signal processing," *EURASIP Journal on Information Security*, vol.7, no. 2, pp.1-20, 2007.

[75] J. Eggers, J. Su and B. Girod, "Public key watermarking by eigenvectors of linear transforms," in *Proc. of Euro. Signal Processing Conf.*, 2000.

[76] S. Craver and S. Katzenbeisser, "Security analysis of public-key watermarking schemes," in *Mathematics of Data/Image Coding, Compression, and Encryption IV, with Applications*, vol.4475, pp.172-182, 2001.

[77] A. Adelsbach and A. Sadeghi, "Zero-knowledge watermark detection and proof of ownership," in *Proceedings of the 4th Inter. Workshop on Info. Hiding*, vol.2137, pp.273-288, 2001.

[78] J. R. Troncoso-Pastoriza and F. Perez-Gonzales, “Zero-knowledge watermark detector robust to sensitivity attacks,” in *Proc. ACM Multimedia Security Workshop*, 2006.

[79] M. Malkin and T. Kalker, “A cryptographic method for secure watermark detection,” in *Proc. 8th Int. Workshop on Information Hiding*, 2006.

[80] W. Zeng and B. Liu, “A statistical watermark detection technique without using original images for resolving rightful ownerships of digital images,” *IEEE Transactions on Image Processing*, vol. 8, NO. 11, pp.1534-1548, 1999.

[81] N. A. Weiss*, A Course in Probability*, Addison-Wesley, pp. 385-386, 2005.

[82] O. Goldreich, *The Foundations of Cryptography*, Basic Applications, Cambridge University Press. 2004.

[83] W. Lu, A. L. Varna, A. Swaminathan, and M. Wu, “Secure image retrieval through feature protection,” in *IEEE Conf. on Acoustics, Speech and Signal Proc.*, pp. 1533-1536. 2009.

[84] W. Lu, A. L. Varna and M. Wu, “Security analysis for privacy preserving search for multimedia,” in *Proceedings of IEEE 17th Inter. Conf. on Image Processing*, 2010.

[85] M. Davenport, P. Boufounos, M. Wakin, and R. Baraniuk, “Signal processing with compressive measurements,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, No.2, pp. 445-460, 2010.

[86] J. Tropp and A. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit,” *IEEE Transactions on Information Theory*, Vol. 53, No.12, pp.4655-4666, 2007.

[87] Y. Rachlin and D. Baron, “The secrecy of compressed sensing measurement”, *46th Annual Allerton Conference on Communication, Control, and Computing*, pp. 813-817, 2008.

[88] A. Orsdemir, H.O. Altun, G. Sharma and M.F. Bocko, "On the security and robustness of encryption via compressed sensing", *IEEE Military Communications Conference*, pp.1040- 1046, 2008.

[89] S. Voloshynovskiy, O. Koval, F. Beekhof and T. Pun, “Random projection based item authentication”, in *Proceedings of SPIE Photonics West, Electronic Imaging/Media Forensics and Security XI*, San Jose, USA, 2009.

[90] F. Kerschbaum, D. Biswas and S. Hoogh, “Peformance comparison of secure comparison protocols,” in *Proceedings of 20th International Workshop on Database and Expert System Application*, pp.133-136, 2009.

[91] I-C. Wang, C. Shen, J. Zhan, T. Hsu, C. Liau and D. Wang, "Toward empirical aspects of secure scalar product," *IEEE Trans. on Sys., Man, and Cybernetics*, vol.39, No. 4, pp. 440-447, 2009.

[92] B. Goethals, S.Laur, H. Lipmaa and T. Mielikainen, "On private scalar product computation for privacy-preserving data-mining," in *Proceedings of the 7th Inter. Conf. on Information Security and Cryptology*, pp. 104-120, 2004.

[93] L. Zhang, "Sample mean and sample variance: Their covariance and their (in) dependence," *Amer. Statist*. Vol.61, pp. 159-160, 2007.

[94] http://en.wikipedia.org/wiki/Collaborative_filtering

[95] G. Adomavicius and A. Tuzhilin, "Towards the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions", *IEEE Transactions on Knowledge and Data Engineering* 17 (2005), 634–749.

[96] R. Bell and Y. Koren, "Improved neighborhood-based collaborative filtering," in *Proceedings of KDD Cup and Workshop*, 2007.

[97] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, "Itembased collaborative filtering recommendation algorithms," in *Proceedings of the 10th International Conference on World Wide Web (WWW '01)*, pp. 285–295, May 2001.

VITA

Qia Wang was born in Shenyang, China in 1985. He received his B.E. degree from the Dept. of Computer Science, Tongji University, Shanghai, China in 2007. Since then, he has been working toward his Ph.D. degree in the Computer Science Department, University of Missouri, under the supervision of Dr. Wenjun Zeng. Qia's research interest includes mobile computing, mobile multimedia, computer vision, compressive sensing, digital watermarking, encrypted domain signal processing and secure multiparty computation. He was an intern with the Media Networking Lab, Futurewei Technologies in New Jersey, in 2010 and 2013 respectively. Now he is a senior engineer with Samsung's Mobile Lab, Bellevue, WA.