ISTANBUL COMMERCE UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEVELOPMENT OF AN OPTIMUM BIOMETRIC HASHING FOR DEEP LEARNING APPLICATIONS

Abdulrahim Mohamed IBRAHIM

Supervisor
Assist. Prof. Dr. M. Alper ÖZPINAR

MASTER'S THESIS
COMPUTER ENGINEERING DEPARTMENT
ISTANBUL – 2022

ACEPTANCE AND APPROVAL PAGE

On 30/06/2022 **Abdulrahim Mohamed IBRAHIM** successfully defended the thesis, entitled "**Development of an Optimum Biometric Hashing for Deep Learning Applications**" which he prepared after fulfilling the requirements specified in the associated legislation, before the jury members whose signatures are listed below. This thesis is accepted as a **MASTER'S THESIS** by Istanbul Commerce University, Graduate School of Natural and Applied Science **Department of Computer Engineering.**

Supervisor	**Assist. Prof. Dr. M. Alper ÖZPINAR** İstanbul Commerce University	.....
Jury Member	**Assist. Prof. Dr. Ali BOYACI** İstanbul Commerce University	.....
Jury Member	**Assoc. Prof. Dr. Buket DOĞAN** Marmara University	.....

Approval Date: 08.07.2022

In accordance with Article 2 of the Board of Directors Decision No. 2022/351 dated 08.07.2022 of the Institute of Natural and Applied Sciences, Istanbul Commerce University, it was unanimously decided to graduate the student named " Abdulrahim Mohamed IBRAHIM" who was determined to have fulfilled his course loads and thesis obligation.

Prof. Dr. Necip ŞİMŞEK
Head of Graduate School of Natural and Applied Science

ACADEMIC AND ETHICAL RULES
DECLARATION OF CONFORMITY

In this proect I prepared in accordance with the rules of thesis writing, Istanbul Commerce University, Institute of Science,

- I obtained all the information and documents in the project within the framework of academic rules.
- I present all visual, audio and written information and results in accordance with scientific moral rules.
- I refer to the related works in accordance with scientific norms in case of using others' works.
- I cited all the works I cited as a source.
- I did not make any distortions in the data used.
- and that I do not present any part of this project as another thesis study at this university or another university.

I declare.

08.07.2022

Abdulrahim Mohamed IBRAHIM

CONTENTS

Page

ABSTRACT

M.Sc. Thesis

DEVELOPMENT OF AN OPTIMUM BIOMETRIC HASHING FOR DEEP LEARNING APPLICATIONS

Abdulrahim Mohamed IBRAHIM

Istanbul Commerce University
Graduate School of Natural and Applied Sciences
Department of Computer Engineering

Supervisor: Assist. Prof. Dr. M. Alper ÖZPINAR
2022, 46 pages

One of the most important approaches to securing digital and physical access to protected systems is biometric identification. This approach can be combined with other identification methods such as passwords and various 2FA tools to improve the security level. Whatever the security policy and governance approach, none of these systems can provide a fully secure system unless the assets are fully physically and digitally isolated. The literature and history are replete with examples of security breaches and data leaks. Regardless of the damage from leak, if the leaked and exposed data can be easily replaced/changed and updated with more secure and unique data, the problems can be more easily overcome. However, if the biometric data is breached or leaked, that biometric identification data cannot be used for the rest of the person's life. Since biometric-based authentication approaches are easy to implement, disseminate and use, there is a growing intended to use them in most common daily applications, even for cloud computing. Concerns about security and ownership of data are much greater in cloud computing due to its nature and notoriety. In this paper, the possible use, system performance and protection of biometric data with homomorphic encryption combined with matching the biometric data by distance-based machine learning approaches. A real use case also implemented by performing all computations in the encrypted domain is explained using an example set of hand recognition sample image datasets.

Keywords: Biometrics technology, homomorphic encryption, SIFT, template protection.

ÖZET

Yüksek Lisans Tezi

DERIN ÖĞRENME UYGULAMALARI IÇIN OPTIMUM BIYOMETRIK HASHING GELIŞTIRME

Abdulrahim Mohamed IBRAHIM

Istanbul Ticaret Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Alper ÖZPINAR
2022, 46 sayfa

Korunan sistemlere dijital ve fiziksel erişimi güvence altına almaya yönelik en önemli yaklaşımlardan biri biyometrik tanımlamadır. Bu yaklaşım, güvenlik düzeyini iyileştirmek için parolalar ve çeşitli 2FA araçları gibi diğer tanımlama yöntemleriyle birleştirilebilir. Güvenlik politikası ve yönetişim yaklaşımı ne olursa olsun, varlıklar tamamen fiziksel ve dijital olarak izole edilmedikçe bu sistemlerin hiçbiri tam olarak güvenli bir sistem sağlayamaz. Literatür ve tarih, güvenlik ihlalleri ve veri sızıntıları örnekleriyle doludur. Sızıntıdan kaynaklanan hasar ne olursa olsun, sızdırılan ve açığa çıkan veriler daha güvenli ve benzersiz verilerle kolayca değiştirilebilir/değiştirilebilir ve güncellenebilirse sorunların üstesinden daha kolay gelinebilir. Ancak, biyometrik veriler ihlal edilirse veya sızdırılırsa, bu biyometrik kimlik verileri, kişinin hayatının geri kalanında kullanılamaz. Biyometrik tabanlı kimlik doğrulama yaklaşımlarının uygulanması, yaygınlaştırılması ve kullanılması kolay olduğundan, bunların en yaygın günlük uygulamalarda, hatta bulut bilişim için bile kullanılmasına yönelik bir büyüme vardır. Bulut bilişimde verinin güvenliği ve sahipliği ile ilgili endişeler, doğası ve kötü ünü nedeniyle çok daha fazladır. Bu bildiride, homomorfik şifreleme ile biyometrik verilerin olası kullanımı, sistem performansı ve korunması, biyometrik verilerin uzaktan temelli makine öğrenmesi yaklaşımları ile eşleştirilmesi ile birleştirilmiştir. Şifrelenmiş alandaki tüm hesaplamaların gerçekleştirilmesiyle de uygulanan gerçek bir kullanım durumu, bir örnek el tanıma örnek görüntü veri kümeleri kümesi kullanılarak açıklanmıştır.

Anahtar Kelimeler: Biyometri teknolojisi, homomorfik şifreleme, şablon koruması.

ACKNOWLEDGEMENT

Throughout the writing of this thesis, I have received knowledge support, ideas and advice combined all of that guided me in this journey of completing the thesis which I am deeply grateful to my supervisor, Dr. Öğr. Üyesi Mustafa Alper ÖZPINAR, his expertise is uncountable in structuring the research in steps by steps by questioning and reviewing the methods. Dr. Alper your feedbacks always made me do much research and look where I have made a mistake and recover from it.

Finally, I would like to thank my family for standing with me all this time.

Abdulrahim Mohamed IBRAHIM
Istanbul, 2022

LIST OF FIGURES

Page

LIST OF TABLES

Page

SYMBOLS AND ABBREVIATIONS LIST

FLANN	Fast library for Approximate Nearest Neighbors
SEAL	Simple Encrypted Arithmetic Library
SIFT	Scale-Invariant Feature Transformation
TENSEAL	Tensor SEAL

1. INTRODUCTION

The movement toward security and securing data has increased in the daily and widespread use of biometric technologies; it has become a critical problem to recognize the security of our biometric identity in case of a breach or leak. The increase in protection and its improvement through biometric technology has drawn attention to conduct studies.

1.1. Background of The Study

Within the recent years, cloud computing has advanced from being a promising business theory to one of the fastest-growing parts of the IT industry. Now, recession-hit corporations progressively recognize that simply by tapping into the cloud, they can obtain fast passage to best-of-breed business applications or drastically increase their infrastructure resources, all at negligible value. However, as more information on individuals and companies are placed in the cloud, concerns are beginning to rise about how safe an environment is.

Arguments about the security of cloud-stored data such as identity can be breached in an easy way. Other ideas are mentioning that cloud providers own strong inducement to stabilize trust and generate a higher score of security; industrious hackers can breach virtually any server and one-third of attacks are lost or stolen devices which exposing data on the internet nearly 16 over hundred percent because of inside theft (Yang et al, 2010).

1.2. Problem Statement

Traditionally, digital security measures were secured using passwords and early identification methods (e.g., 2FA). With the recent technological advancements, securing digital systems becomes a challenging task due to the complexity of the digital systems, the plethora of software (or applications) (i.e., different security requirements based on the specific application), and different security policies and governance approaches. One of the robust security identification methods is Biometric

identification, in which a bio-related (e.g., a Palm) metric is used to identify a person or grant access to a protected digital system. Biometric identification provides a greater security level compared to traditional security methods. However, Biometric identification cannot tolerate a security breach (or leak) due to the uniqueness of the identification metric used (e.g., an iris). Unlike traditional methods, whereby a password can be easily replaced with a stronger one. In this research, we address the issue of security breaches (or leaks) on Biometric identification and propose to use state-of-the-art homomorphic encryption methods combined with matched biometric data by a distance-based machine learning approach. Therefore, this research investigates the applicability of using homomorphic encryption with Biometric identification on empirical data to improve the security levels of Biometric identification methods and reduce security breaches (or leaks).

2. LITERATURE REVIEW

On this specific chapter will do a deep preview of the literature review about the related project of our choice, which is Homomorphic Encryption in Templet Protection for Signature Data; first, brief talk about the general history of Encryption and associated matters with Biometric, the latest ideas on the system and existing system.

2.1. Introduction

On the first hand, let us understand the meaning of encryption; in actual point of view, it can be seen as a securing data with all needed matters, in a result that you can see the necessary needs for the data to be secure, most of the studies they do focus on that deeply, on the other hand, we can assume that its importance in all level of data integrity, from that side we can calculate the significance of encryption that also needed plus we should know how many systems made for the biometric with encryptions, According to (Kr et al, 2020) There are so many security issues which are:

- A data leak has a status of illegal actions of obtaining data which is unauthorized or re-gain the data from applications or individuals acting like the actual owner of the data, hacker attacks using methods to get a secure database (Yang et al, 2010).
- Accounts breaking has a target for hackers to dominate the privacy of emails, personal information connected with technologies (Singh, 2014).
- Attacks at any organization from their resources or information might be supported by inside theft or outsider with the support of inside tips (Yang et al, 2010).
- Using Trojan or Malware, Hackers can breach into sensitive and private information (Singh, 2014).
- Misuse of cloud services leads to major threats (Omar, 2015).
- Leaking or Data loss is where hackers succeed in accessing sensitive data by using multiples of attacks. Those attacks generate a lot of data loss in the time of communication and processing specific data in cloud servers (Subashini et al, 2011).

2.2. Cryptography

According to (Dube, 2008) defined it is the process of translating an information-conduct message to completely purposeless “nonsense" or turning back of the process (decoding a protected cryptographed message). mainly there is two primary cryptographic processes; symmetric and asymmetric, where symmetric is in brief using same key to encrypt and decrypt a message, and asymmetric refer to different keys to encrypt and decrypt.

- Symmetric Cryptography: (Dube, 2008) they clarify in a comment that “it has two issues.”
  - The first fundamental issue to implement the method, it requires transmission between parties securely by the key encrypted; necessarily, the key cannot transfer among the parties without security (Dube, 2008).
  - The Second Fundamental that they also mentioned in detail (Dube, 2008) was that the issue is that a various key is required for each possible party. The presentation of keys separately, the control of systems showing each recipient has various keys immediately increases challenging tasks as the number of recipients increases; In short, it ultimately restricts the value of symmetric key.
- Asymmetric Cryptography: (Dube, 2008) also describe in asymmetric cryptography, equal keys are applied, one key for encryption and the second is for decryption. To describe more; the entire process is about the receiver should be holding the decrypting key after encrypting an information.

2.3. Passwords and Keys

A password is a private that must be recognized only by the inventor of the password. In this ability, it is known as (something you know). As it should be difficult to guess the password, it may have some value in preserving data when linked to cryptography (Dube, 2008).

- From (Dube, 2008) prospective they explain that the Password and authentication are not only meant for encryption, but It also gives strong

background in identifying the user by the techniques of checking the secrets displayed against what was previously shared by the user, So in short authentication of a person is not like the authentication of a computer for one major reason, authenticating a person requires extract numerical data by the support of devices such fingerprint, where it should be unique to the person, Authentication may contain one or more “factors” that are evaluated by the receiving party in determining whether a person is in fact who he claims to be, these factors may include:

- The user knows the information (a secret not possibly known by anyone else).
- The user has the information (something unique in his possession).
- The user “is” the information/Collected (such as a fingerprint).

- On the other hand (Dube, 2008) they talked about Hardware-based authentication, and they show that the authentication of parties over the internet can be categorized into one of two categories:
  - Technologies interact with the person and does the authentication.
  - Technologies authenticates the device consequent to which he is operating.
- Further details on Biometric authentication for a person: identifying the person's identity. As explained, "something you know" is a secret PIN or Password granted to a user/Person, so the browser recognizes it; writing the password down is a vulnerable technique where can be breached. For this issue, hardware-based authentication is much secure than software-based authentication, where the hardware interacts with you as identity "something you are", such as a fingerprint (Dube, 2008).
- In details of famously used hardware-based person authentication tech (Dube, 2008), they classified it in:
  1. Fingerprint Scanner: The uniqueness of a fingerprints person is measured by a tool that creates extracted digital signature, which should not be changed, and that extracted digital signature should be saved in a database where can be used later to recognize the person and avoid hacker’s attacks.

2. Voiceprints: The voiceprint identifies voice pattern representations of a user/person by recording the voice at first and registering it in the database, supported by tools that extract the different patterns of a word's that the person says and recorded.
3. Iris Scans: Iris scans extracted and unique features from the person's eye, in comparison over 200 measurement points applied. Extracted features like rings, furrows, spots, and color are what gives high identification security; hence, some methods use pupil to differentiate from live data to targeted iris picture. The positive advantage of the iris is retinal scans.
4. Palm prints: Hand palm uses unique and unchanging characteristics in the hand shape and the signs/marks or scars and make it an identifier, with the ability to save it for the comparison status and identifying the person's identity.

2.4. Biometric Concept of Security

According to (Rafie, 2016) Biometrics is the Analyzing individuals physical and behavioral form, Bio originated meaning is life, and metric is the measure. Biometric is identifying and detecting a person's physiological elements, using finger, face, hand, voice, eyes etc. The behavior of the individual can be as typing movement, gait or gestures. It's much unique to each individual and helps to make identification and access control for security purposes. In a detailed explanation, biometrics are divided into two factors, identification, and verification, where verification uses the captured or extracted features stored to match the pattern and approves the access and identification are used to detect the unique features out of the device used to store it in database.

In (Tripathi, 2011) they explain how does biometrics work as shown in Figure 2.1:

- Enrollment: The method Information of a person is collected, stored, and prepared as a template to re-use it in the biometric system called enrollment; in the stage of verification and identification trail are conducted against the templates.

- Presentation: Presentation is the method by which the user displays biometric data into the acquisition device; for example, it may require looking directly into the camera, or putting the finger on a finger scan device or speaking into the device a name or secret word to pass.
- Biometric data: Biometric users submit in a raw image or record tracking of feature; the raw data is also described as raw biometric data or representation.
- Feature extraction: It is the process of encountering and encoding unique characteristics from biometric data for generating a template name feature extraction. In detail, this process takes place throughout enrollment and verify any time a template is created, and this involves locating features by filtering and optimizing image and data accurately.

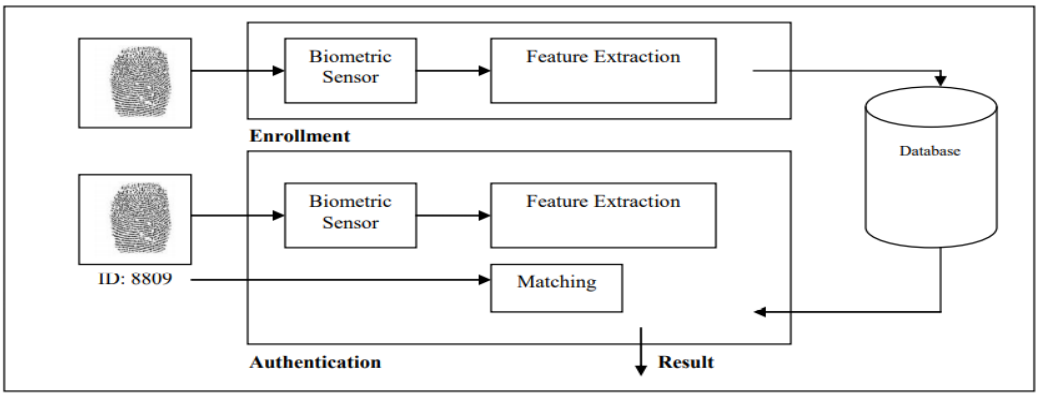

Figure 2.1 Comparison between biometric techniques (Tripathi, 2011).

- From (Rafie, 2016) perspective, they classify it as category of biometrics: Many biometrics types for an individual use biometric in the form of access control and identification, it covers many types of features which stored from the individual to identify the person, such DNA matching or form of ear shape called ear shape recognition or eye pattern recognition or face recognition and many more.
    1. Fingerprint Recognition: Each person has a unique finger identity pattern, and it requires at first a device to extract the lines and if there are any scars can be recorded then enroll into the database to store with the help of algorithms to encode the image into numerical, so when it comes to matching the pattern/template that was created for the person,

it gives an accurate response from the system. (Paderes, 2016). Figure 2.2 shows the process of reading fingerprint biometric.

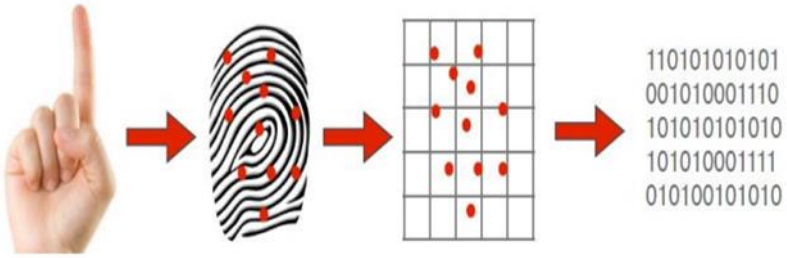

Figure 2.2 An example of how a fingerprint biometric is being read (Paderes, 2016).

2. Facial Recognition: According to (Rafie, 2016) It uses the same ideology of the fingerprint by detecting the face shape using a device like a camera, and store the unique figures in the shape and structure of the parts of the face measurement such as nose, forehead and mouth, with the support of the algorithm to encode it into numerical and store it in the database where it uses to identify and verify in the matching template process. The process of facial recognition is shown in Figure 2.3.

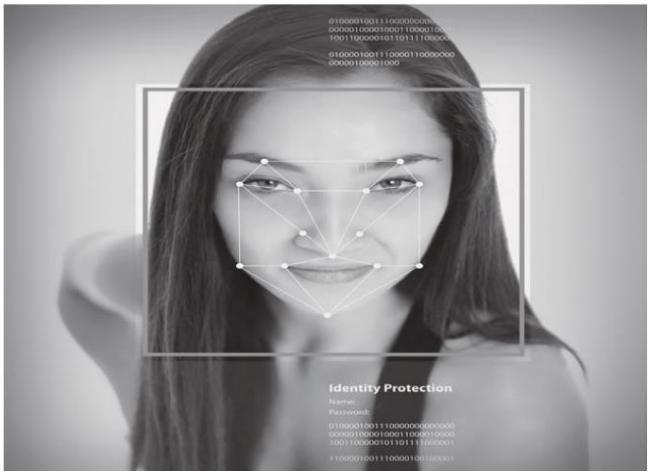

Figure 2.3 Two-dimensional facial recognition image (Das, 2018).

3. Eye pattern recognition: Iris recognition systems (see Figure 2.4), according to (Rafie, 2016) It's one of the most complex systems; it requires nearly 200 identification points on an individual's iris, and it requires devices that give an accurate detection to the iris/retina and

store it in a template by algorithms that encode it for template matching as shown in Figure 2.4.

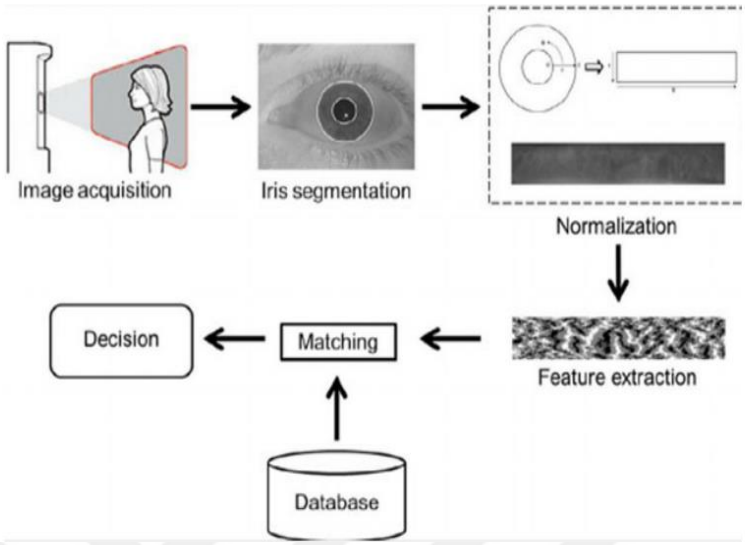

Figure 2.4 Major steps in iris biometrics processing (Dargan et al, 2020).

4. Voice Pattern Recognition: Voice biometrics is unique digital representation of a voice of a person with a specific pattern is different from one to other individuals, and it's hard to duplicate for biometric identification and authentication. In detailed voice recognition requires a clear speech pattern which is unique for template storing and matching same as a fingerprint (Paderes, 2016). Figure 2.5 shows an illustration of voice recognition system with human voice patterns.

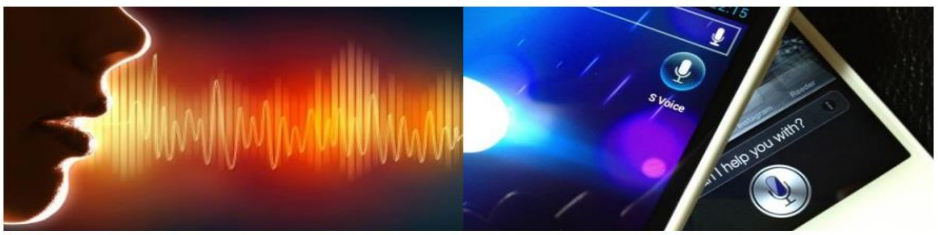

Figure 2.5 Sample on how the voice recognition system works (Paderes, 2016).

- In further explanation (Rafie, 2016) Biometric Authentication Systems: The revolution toward security made the importance of biometric authentication

and verification much common in big security companies; in addition to the value of biometric is the comfort of use and support for the individuals who implement this technology in their daily work performance.

- From (Rafie, 2016) perspective of view, they classify the Advantages and Disadvantages of Biometric systems:
    1. Improved Security: Biometric recognition systems can achieve a more elevated degree of security in contrast to traditional models of ordinary authentication systems.
    2. Improved Convenience: Biometrics considerably clarifies the entire method that is required in authentication and in accomplishment also minimize the pressure load on the user of the system.
    3. Improved Accountability: By set up biometrics to ensure access to computers and other facilities fulfils the event such as buddy punching and therefore gives a higher point of confidence.
    4. Improved Cost: Yes, Cost is a major issue which is a disadvantage after knowing the level of security that provides a biometric method.

According to (Patel et al, 2015), The advantage of using biometrics levelled up security and privacy concern as outlined below:

- Biometrics is no secret: Knowledge based authentication process relay on privacy; for Example, password keys are only recognized by the user; hence privacy can be stabilized; on the opposite side, biometrics techniques can be registered and misused without the individual permission.
- Biometrics can’t be withdrawn or cancelled: biometric based authentication method usually would not interact with spoofed data on the system, but if hackers’ success an entrance to biometric samples and can display it then the system is vulnerable and endangered forever, where password keys once an attack happen can be replaced, in explanation views, the biometric information is permanently related to the user no change or replace can happen if endangered.
- Cross application invariance and cross-matching: It's important to have different passwords and tokens in common authentication procedures. However, biometric authentication depends on identical matching; once it is

displayed, it's breached forever. Moreover, since the identical biometric process to overall schemes and places the user can follow if one or more organizations collude and share their respective biometrics databases.

- Persistence: While relative improvement over time is helpful for biometrics, it can also be a considerable challenge from a privacy case of view while it requires to be changed. The uniqueness included in them is still the same even though the signal and the template can look different.

To overcome all these vulnerabilities, researchers studied the template protection method in-order to get an accurate solution. Figure 2.6 shows the categories of template protection schemes, in which, in the opinion of (Rathgeb et al, 2011), Cancelable biometrics (CB) It is a method to prevent a loss in biometric data and can be breached and re-used again throughout the templates in the domain.

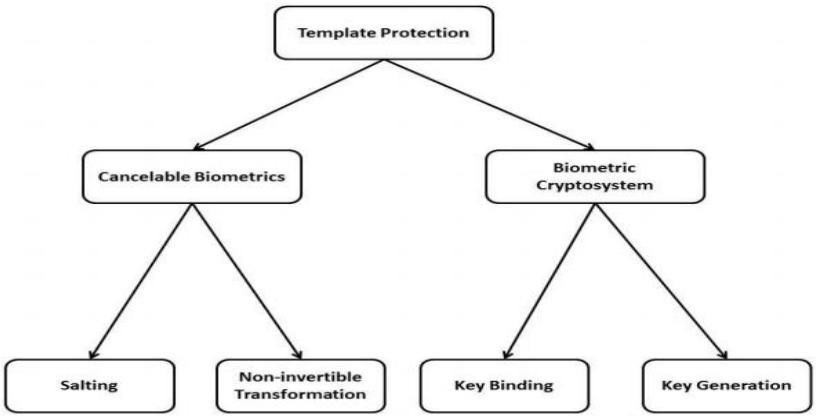

Figure 2.6 Diagram of different categories of template protection schemes (Choudhury et al,2018).

2.5. Encryption Concepts

Encryption applies to a form of algorithms that transform the plain text to a form of code or unclear text and present privacy to decrypt the text; the receiver uses the “key” for the encrypted text, which can be grouped into symmetric and asymmetric keys as shown in Figure 2.7. It has been the traditional system of securing the data, which is essential for the military and the government operations which now it becomes more to civilian’s day-to-day life too, also the online activities of banks, the data transfer via

networks, transfer of necessary personal information that requires the form of encryption for security purposes (Onwutalobi, 2016).

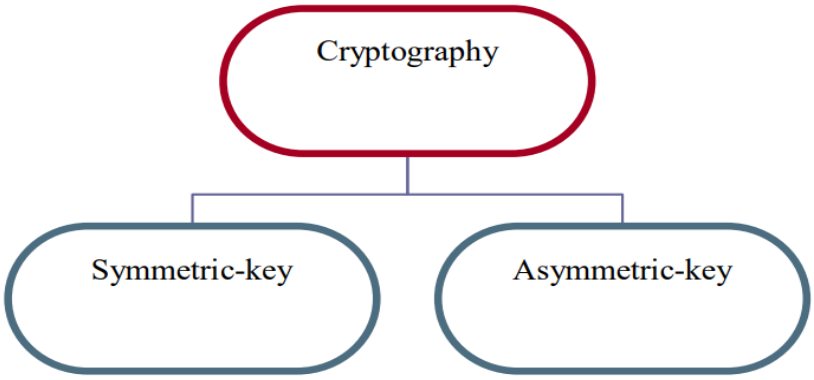

Figure 2.7 Overview of the cryptographic encryption algorithms (Reza et al, 2013).

2.5.1. Symmetric key encryption

According to (Faheem et al, 2017) is used related key for the encryption and decryption of a message. Encryption and decryption keys are saving secrets and are known by authorized senders and the recipient required to transmit. Allocating several keys to the different parties raises the overall message security. The power of symmetric key encryption relies on the privacy of encryption and decryption keys. The symmetric encryption algorithm is classified into block and stream cipher based on the grouping of message bits.

On (Sharma et al, 2012) part of view, symmetric encryption scenario has five methods as shown in Figure 2.8:

1. Plaintext: This is the first understandable information or data that is provided into the algorithm as input.
2. Encryption Algorithm: The encryption algorithm works different substitutions and transformations on the plaintext.
3. Secret Key: The key is a value separated of plaintext and the algorithm. The algorithm will deliver a separate output depending on the specific key being used at the time.
4. Ciphertext: This is the composite information created as output, and it relies on the plaintext and secret key.

5. Decryption Algorithm: It's practically the encryption algorithm flow in contra, and it takes the ciphertext and the secret key and delivers the actual plaintext.

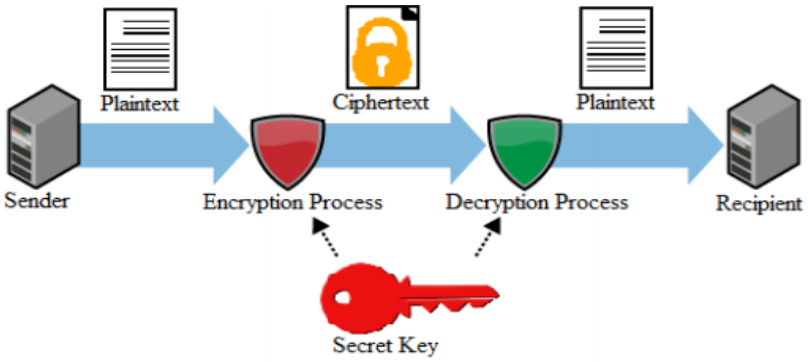

Figure 2.8 Components of the symmetric block cipher (Faheem et al, 2017).

2.5.2. Asymmetric key encryption

According to (Faheem et al, 2017) The asymmetric key encryption (see Figure 2.9) is generally guided to as public-key encryption in which different keys are used for the encryption and decryption of the message. The encryption key is also stated as the public key and can be operated to encrypt the message with the key. The decryption key is stated to be a secret or private key and can be operated to decrypt the message. The power of the asymmetric key encryption is utilized with a digital signature, and then it delivers to the users via message authentication detection.

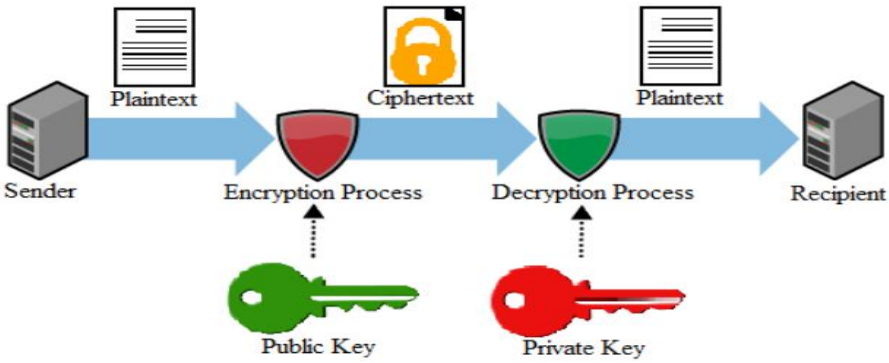

Figure 2.9 Components of asymmetric block cipher (Faheem et al, 2017)

2.5.3. Classification of classical encryption techniques

According to (Soni et al, 2013) Classical Encryption techniques can be broadly classified into Substitution and Transposition techniques as shown in Figure 2.10:

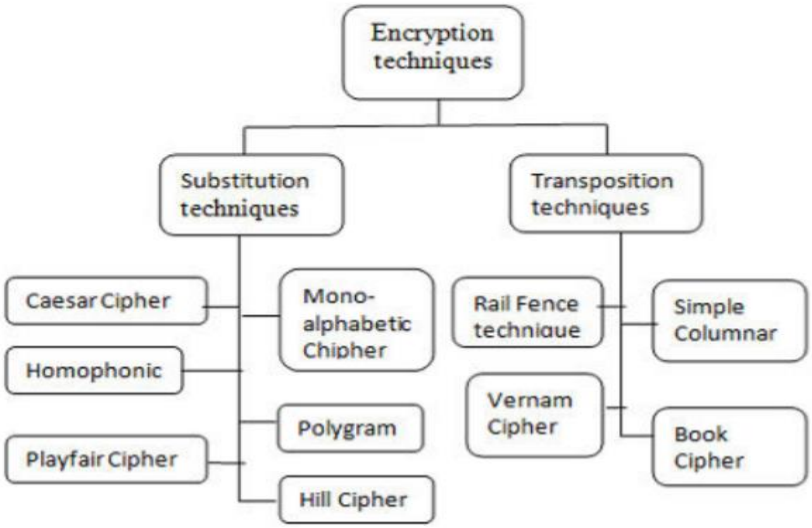

Figure 2.10 Various encryption techniques (Soni et al, 2013)

- Substitution Techniques: The letter of plain text is exchanged by different letters or by numbers or symbols, such:
  - Caesar Cipher: This is the simplest substitution cipher by Julius Caesar; in this substitution technique, to encrypt the plain text, each letter of the plain text is substituted by the letter three places further it. And to decrypt the ciphertext, each alphabet of ciphertext is replaced by the letter three that places before, As an example:

    Plain Text: meet me tomorrow.

    Cipher Text: phhw ph wrpruurz

    See the example above; we have substituted ‘m’ with ‘p’, which occur three places after ‘m’. Similarly, ‘e’ is substituted with ‘h’, which occurs in three places after ‘e’. If we must substitute the letter ‘z’, then the following three letters included next to ‘z’ will be ‘a’ ‘b’ ‘c’. So, while computing additional three letters, if ‘z’ happens, it circularly follows ‘a’. There are also some of disadvantages to this simple substitution method. If the hacker understands that the Caesar cipher is

applied, he must try 25 potential keys to decrypt the plain text to implement brute force cryptanalysis, and the hacker is also aware of the encryption and decryption algorithm.

- Mono-alphabetic Cipher: Monoalphabetic cipher it's a substitution cipher, where the cipher alphabet for each plaintext alphabet is fixed for the whole encryption. In simple words, if the alphabet 'p' in the plain text is replaced by the cipher alphabet 'd'. Then in the entire plaintext, wherever alphabet 'p' is used, it will be replaced by the alphabet 'd' to form the ciphertext.
- Homophonic Cipher: Homophonic Cipher applies the substitution of one plain text character with a ciphertext character at a time; however, the ciphertext character can be any of the selected set. For example, replace A with D, B with E.
- Polygram Cipher: Polygram Substitution Cipher method substitutes one block of plain ciphertext with a block of cipher text-it does not work on a character-by-character basis.
- Playfair Cipher: Playfair cipher is a substitution cipher that involves a 5X5 matrix.

- Transposition techniques: Transposition cipher utilize some type of permutation on plaintext letters to perform ciphertext, such:
  - Rail Fence Technique: It is the straightforward transposition cipher. The process to obtain ciphertext are as:

    The first step is the plain text is written as a series of diagonals; the second process then, to achieve the ciphertext, the text should read as a series of rows and to clarify it see example:

    Plaintext: meet me Tomorrow.

    First, should write this plain text series wise in a diagonal set, see below:

    reading the above example, you would get it named by that because it seems a rail fence; once you have written the message as a series of

diagonals, to achieve the ciphertext out of it, so reading the first row, the first half of ciphertext:

m e m t m r o

computing the second row will get the second half ciphertext:

e t e o o r w

next, to achieve the complete ciphertext, merge both the halves of ciphertext, and the final ciphertext can be:

Cipher Text: M E M T M R O E T E O O R W

Rail fence cipher is simple to process and implement.

- Simple Columnar: The simple Columnar Transposition Technique simply arranges the plain text as a sequence of rows of a rectangle that are read in columns randomly, the columnar transposition cipher is not the easiest of inversion ciphers to break, but there are statistical features of language that can be utilized to recover the key. To significantly increase the security, a substitution cipher could be applied as well as the transposition. Vernam Cipher (One Time Pad):

  The vernam cipher is implemented using a random set of non – repeating characters as the input ciphertext. Once an input ciphertext for transposition is used, it is never used again for any other message. The length of the input plaintext is equal to the length of the ciphertext.
- Book Cipher (Running Cipher): The concept used in Book Cipher is completely simple and is similar in principle to the Vernam Cipher. For creating ciphertext, some part of text from a book is operated, which suits the purpose of a one-time pad.

2.6. Homomorphic Encryption

In the opinion of (Ogburn et al, 2013) describe Homomorphic encryptions are a process that allows particular types of calculations to be taken out on ciphertext, which generates an encrypted outcome which is also in ciphertext; its point is the outcome of operations created on the plaintext. Case in point, one person, could add two encrypted numbers and then another person could decrypt the result without both being able to retrieve the value of the original numbers.

Encryption refers to algorithms applied to convert plaintext into a code or incoherent form of content and shop security. The recipient uses the "key" to the encrypted content to decrypt the content. Since the old technique was used to secure and protect the source of information, that is extremely essential for high classified operations by the government, it has extended to the more lives of civilians. This includes the online routines of banks, the exchange of data through frameworks, the exchange of important private data that is necessary for the application of encryption for security (Bozduman et al, 2020)

The evolution of ICT systems from On-premises and disconnected resources to cloud-based storage systems such as OneDrive, The Box, Dropbox, Google Drive expands the daily use of these systems. Although they take advantage of these services, the conceivable downsides of operating cloud services are the casualty of privacy and the business significance of personal information. A practical way to solve these problems exists by encrypting most of the data reserved in the cloud and operating on the encrypted data (Bozduman et al, 2020)

According to (Bozduman et al, 2020) statement it is a knowledge of how to hold and protect significant data using encryption which gives permission on applying addition over the ciphertext even applying multiplication. Figure 2.11 shows an example on calculation applied through plaintext using HE. While Figure 2.12 shows A simple client-server scenario using HE.

As clearly known, Homomorphic Encryption has five stages of passing through:

- Setup: scheme, a security parameter, Functionality parameter.
- Key generation: a secret key, public key, re-linearization key, Galois key.
- Encryption: number or vector of number (ciphertext).
- Evaluation Then Decryption.

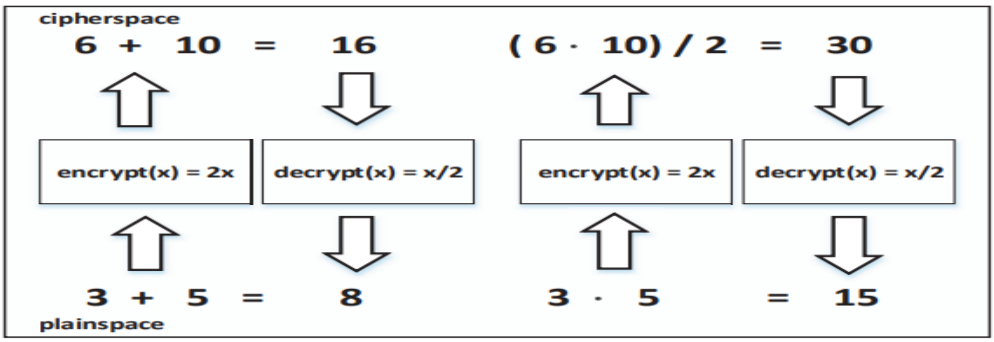

Figure 2.11 Example on calculation applied through plaintext using HE (Ogburn et al, 2013).

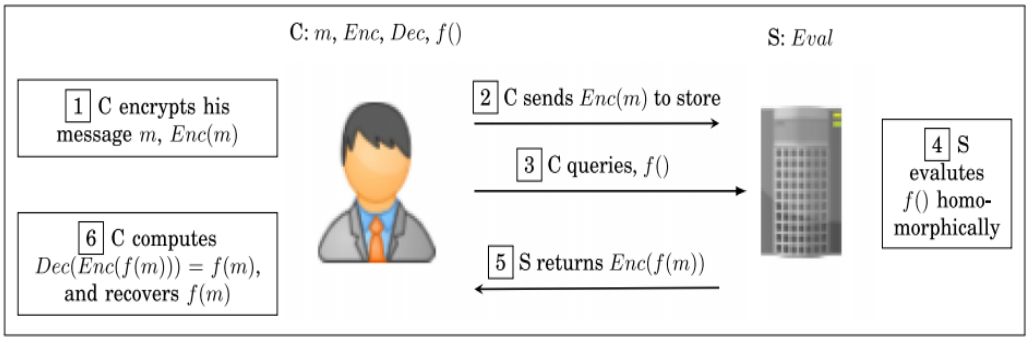

Figure 2.12 A simple client-server HE scenario (Acar et al, 2018).

Classification of Homomorphic Encryption is shown in Figure 2.13 and are broadly divided into:

1. Partially Homomorphic Encryption (PHE)
2. Somewhat Homomorphic Encryption (SWHE)
3. Fully Homomorphic Encryption (FHE)

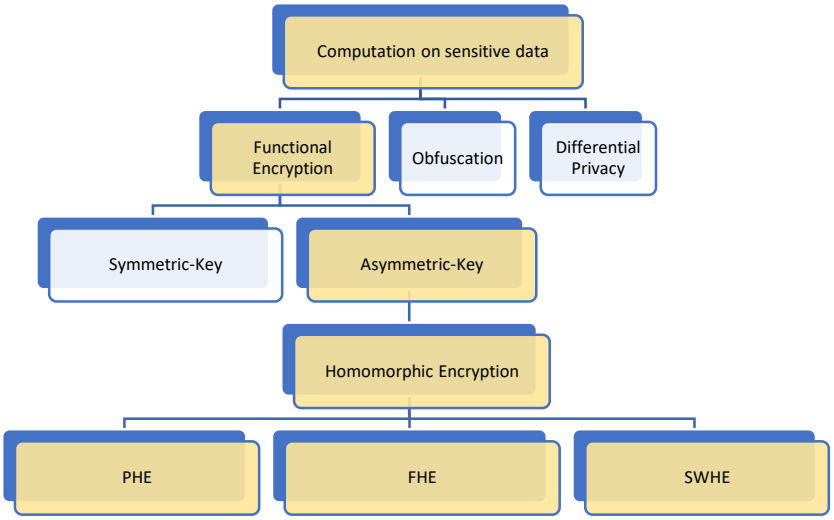

Figure 2.13 Classification of homomorphic encryption

2.6.1. Partially homomorphic encryption

In PHE, exclusively some calculations can be applied to the encrypted data. in one process, only one operation can be handled, such as addition or multiplication; this scheme works explicitly on the voting scheme; the encrypted voted ballots can be publicly kept direct in the cloud, and the public can compute the votes together with securing their votes per candidate without having any proof to the third party (Devi, 2017). According to (Morris, 2013) Some examples of partially homomorphic cryptosystems are:

1. RSA - multiplicative homomorphism: According to (Ora et al, 2016), This algorithm was designated for Rivest, Shamir and Adleman, who applied their public-key cryptosystem; in short RSA method is implemented by multiplying two RSA ciphertexts with the decrypting the outcome also comparing to the multiplication part of the plaintext values.
2. ElGamal multiplicative homomorphism: On comparison prospective (Imran et al, 2020), they announced and described that El-Gamal is an asymmetric key algorithm focused on key exchange for the public key, in additional description the security based on computing the discrete logs of large prime modulus, as follow the representation of El-Gamal:
    - Generating Key.
    - Encryption.

- Decryption.

3. Paillier additive homomorphism: Paillier cryptosystem is probabilistic encryption and has an additive homomorphic character; created in 1999 by Pascal , Paillier cryptosystem, it has public key n and g, which is the RSA modulus (Anggriane et al, 2017).

2.6.2. Somewhat homomorphic encryption

SWHE can meet the criteria in operating within limited times of addition and a small number of multiplications on ciphertexts. It is established on ring -LWE homomorphic cryptosystem and parametrized by the ring $R_q \triangleq Z_q$ [x]/ $\langle x^n + 1 \rangle$. In this ring, the dimension n is a power of 2, a modulus prime number q, and an error parameter σ that makes a definition of a discrete Gaussian error distribution $\chi = D_{Z^n}\Sigma$ with standard deviation σ. According to a prime t < q, the message space of the scheme can be defined as $R_t = Z_t$ [x] /$\langle f(x) \rangle$. The goal of choosing these parameters (depending on the security parameter κ) in such a way is to guarantee the correctness and security of SHE (Xiong et al, 2018).

2.6.3. Fully homomorphic encryption

FHE is an encryption method that enables analytical procedures to be run directly on encrypted data while delivering the same encrypted results as if the functions were run on plaintext; the purpose behind fully homomorphic encryption is to enable anyone to use encrypted data to implement useful procedures without access to the encryption key (Zhi-gang et al, 2014).

2.7. Scale Invariant Feature Transform

SIFT is an approach applied in feature extraction and matching/identification, The scale-invariant feature transform (SIFT) is under computer vision used algorithm to extract, detect and describe features in images and use matching techniques that are also is under SIFT based matches; David Lowe invented it in 1999, and the most

application that was used in object recognition, robotic mapping,3D modeling and image stitching.

According to (Yanbin et al, 2008) Image feature extracted by SIFT advantages as scale invariability, affine or rotation invariance and that could reduce mismatching because of confusion, noise and provides high matching rate. Figure 2.14 shows matching diagram using SIFT algorithm.

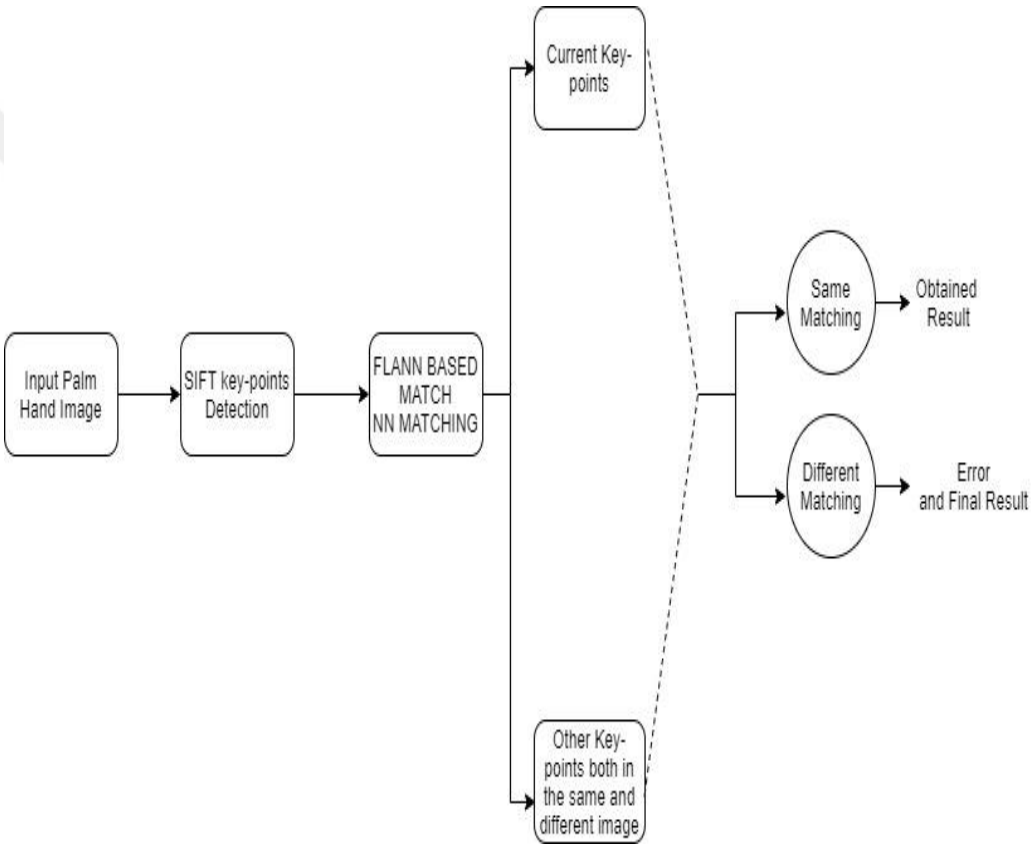

Figure 2.14 Matching diagram using SIFT algorithm

2.7.1. Justification of using SIFT

SIFT has already been widely used in generic object detection problems. Recently, its application in face authentication, fingerprint verification and palmprint images. SIFT helps locate the local features in an image, commonly known as the 'keypoints' of the image. These keypoints are scale and rotation invariant that can be used for

various computer vision applications, like image matching, object detection, scene detection. Figure 2.15 shows the keypoints obtained from an image.

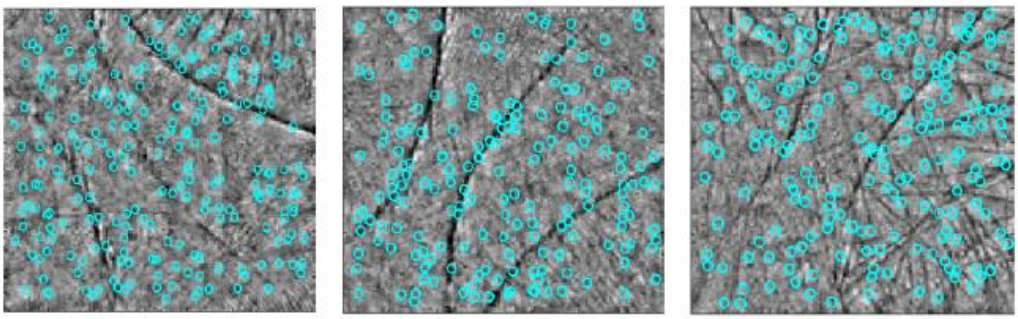

Figure 2.15: Keypoints obtained from an image shown circled in Cyan color

3. METHODOLOGY

3.1. Biometric Identification

According to (Jorgensen et al, 2011) The identification mode involves building the character of a person depending on biometric evaluations. The comparator identifies the captured biometric data with those stored in the database using a 1:N matching algorithm for recognition; in the identification mode, the framework perceives a person looking through the formats of the multitude of clients in the information base for a match. Subsequently, the system directs a one-to-numerous correlations with building up a singular's character (or falls flat if the subject isn't enrolled in the database) without the issue asserting an identity. Identification is an important part of negative recognition applications where the system produces up whether the individual is who (verifiably or expressly) denies being. The reason for negative recognition is to keep a solitary individual from utilizing various characters. Recognizable proof may likewise be used in specific recognition for convenience (the client not needed to guarantee a character). While traditional individual recognition strategies such as passwords, PINs, keys, and tokens might work for positive recognition, negative recognition must be established through biometrics.

The identification system will ask you to classify yourself by submitting a biometric measurement. The measurement you enter is then compared to the measurement you provided when you registered with the system to determine if it matches. Biometric tests are always fuzzy to some degree and change over time and with the level of capture. If the biometric measures presented and captured are sufficient, you are assumed to be the person registered under the name you provided with certainty. If the characteristics presented and captured are insufficient, you will usually be allowed one repeat attempt. With multiple attempts, the number of incorrectly rejected users may be less than one percent.

3.2. SEAL Homomorphic Encryption

Simple Encrypted Arithmetic Library SEAL is a homomorphic encryption library made by Microsoft which allows additions and multiplications to be performed on encrypted data. SEAL presents a plain way to operate for homomorphic encryption library and offers academia and industry functional use of homomorphic operations. An illustration of Microsoft SEAL cloud storage and computation is shown in Figure 3.1. Seal involves two different encryption schemes: BFV and Cheon Kim- Kim-Song, known as CKKS. (Mert et al, 2020).

Only certain privacy-critical parts of programs computed in the cloud should be implemented using Microsoft SEAL. There are two different homomorphic encryption schemes such as the Brakerski/Fan-Vercauteren BFV (Fan et al, 2012) and Cheon, Kim, Kim, and Song CKKS (Cheon et al, 2017) with completely various effects. BFV permits modular arithmetic to be carried on encrypted data. The CKKS scheme (Mert et al, 2019) allows addition and multiplication with encrypted real or complex numbers, but only gives approximate results (Cheon et al, 2017).

BFV permits the client to induct significant computation over encrypted data without access to the decryption key, this can happen for the reason of the homomorphism property that offers a map (or function) between the plaintext and ciphertext spaces that maintains the operations in certain two spaces.

CKKS method allows calculation on encrypted natural or even complex integers, but the outcomes are inaccurate results. Also, CKK operates computations on vectors of complex values as well as real values.

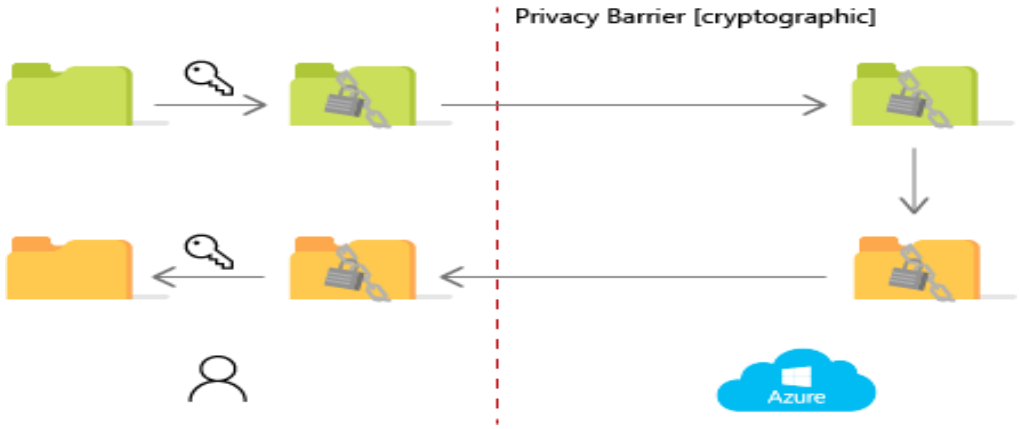

Figure 3.1 Microsoft SEAL cloud storage and computation

3.2.1. TENSEAL

According to (Benaissa et al, 2021) TENSEAL is a library that combines classical machine learning frameworks with homomorphic encryption functions. It controls all complexities of executing tensor methods on encrypted data. The core API is designed around three main components: the context, the simple tensors, and the encrypted tensors.

TENSEAL is a library for conducting homomorphic encryption operations on tensors, constructed on top of Microsoft SEAL, easy to use through Python API while maintaining performance by executing most of its procedures using C++ (Mert et al, 2019).

3.3. Dataset

The experiments have been conducted on the 11K Hands dataset, which is publicly available for research experiments. According to (Afifi, 2019) this dataset has 11,076 full hand from each side and each angle. The subjects were different in age,18 to 75 years old. To vary the shapes of the photographed hands, we asked each subject to open and close the fingers of their right side and left side hands at random. Each hand was photo'd from both the dorsal and palmar sides. Sample images can be seen from Figure 3.2 There is a precise description of metadata combined with each hand image:

- 190 Subjects

- (1600 x 1200 pixels)
- Subject identification
- Gender details
- Age information
- Skin color of the hand

Figure 3.2 Proposed dataset

3.4. Proposed Methodology and Algorithm

In Figure 3.3, a detailed flowchart for the proposed methodology and algorithm has been provided. The proposed framework focuses on security and the use of HE and assumes that clean and unbiased biometric images are already available to the system. Many sampling methods and devices for biometric feature extraction have already been explained in the literature and in the initial parts of this paper, the algorithm can be separated in to two main stages “Feature Extraction from Biometric Data” and working with this data with HE.

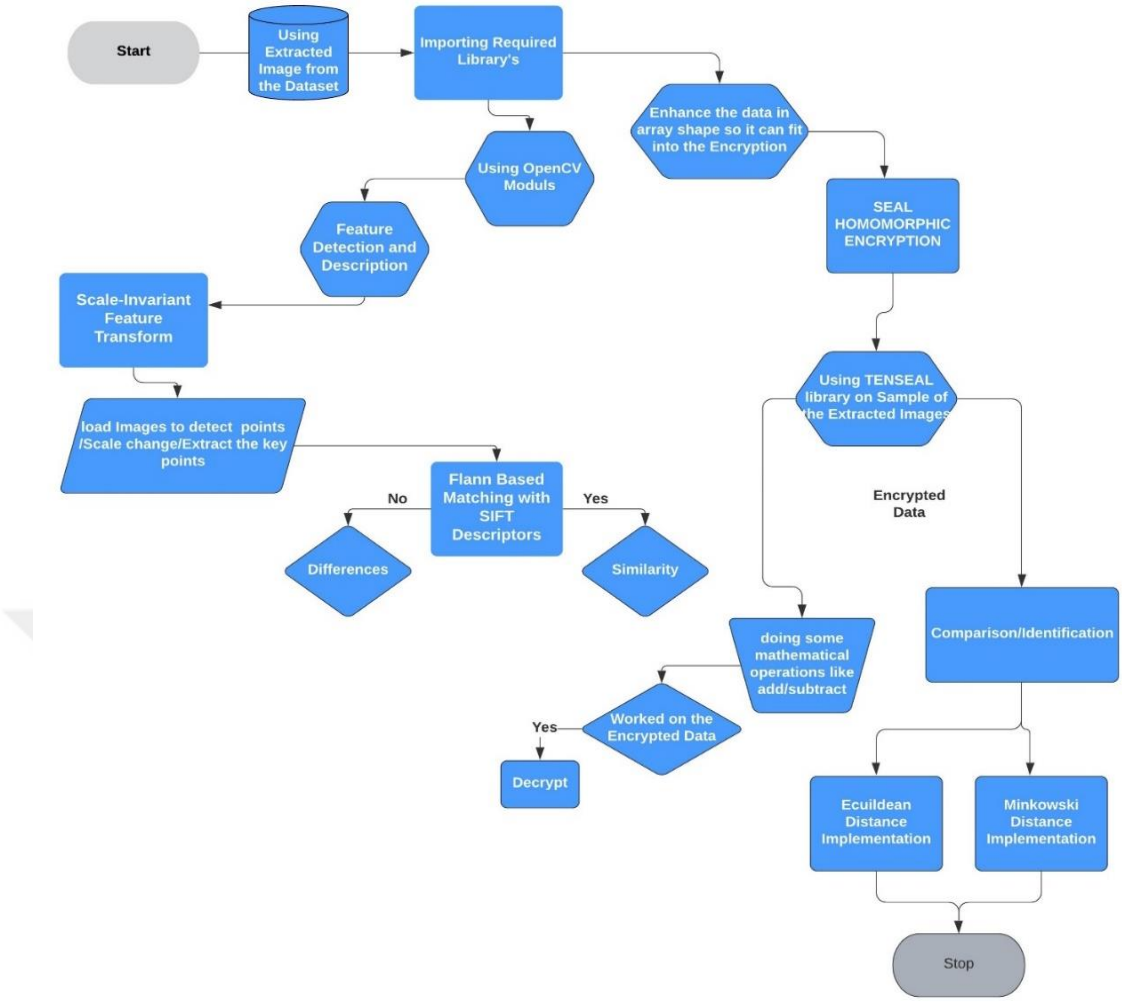

Figure 3.3 Proposed method

4. RESEARCH FINDINGS AND DISCUSSION

The aim of this study is to acquire a way to enhance and secure biometric data without compromising the data itself, implementing two approaches one in the sequence of the identification for the data using SIFT, and the second is after identifying the samples turn it to implement it with homomorphic encryption to obtain results.

Colaboratory, or “Colab” from Google Research used for computations with Python 3 with standard free setup. The first stage for the study is to prove that chosen SIFT algorithm can be used for hand images (Palm).

4.1. First Experimental

The idea is to check if different people have different hand images, and they have a limited similarity. For both left and right hands similarity between different people is around 20% as can be seen from Table 4.1.

Table 4.1 Implementing SIFT on peoples palm vs other people Palm

Samples	Type	Time (second)	Min	Max	Good Match	Average % Similarity
P vs P	Left	35.8807	3.81	20.69	439	18.29
P vs P	Right	34.3043	10	31	338	14.08

As a result, different people also have different SIFT extracted image similarity levels. So, the next part of the study is to see if this approach works for different poses of the same people. The dataset was searched for different samples of the same people and a new dataset was created with 2 images of each person with left and right hand. The processing time is not important as it is a relative approach depending on the machine configuration. The results are shown in Table 4.2, and as expected, the similarities are above 75%.

Table 4.2 Implementing SIFT on people vs themselves (verification)

Samples	Type	Time (second)	Min	Max	Good Match's	Average % Similarity
Verification	Left	18.8149	35	141	910	75.83
Verification	Right	15.5564	29	154	1041	86.75

In summary, the results have shown that identification using SIFT works well in matching and identification as well as in securing and is incomparable can be seen in Table 4.3.

Table 4.3 Results from total samples match

Hands Samples	Type	Total Sample	Average Similarity (%)
Verification	Left &Right	50	81.29
Different People	Left &Right	50	16.185

As part of our experiment, another comparison was also made to see if our SIFT implementation performed similarly with reference studies. For this purpose, a comparison was made with the recent approach of (Karami et al., 2017) for different types of images from different domains. As can be seen from Table 4.4, using SIFT for handheld images has better matching rates than comparing different images.

Table 4.4 Comparison between our proposed work and related work

Approach	Algorithm	Application	Match Rate (%)
Proposed Work	SIFT	Paper's Dataset	81.29
Reference Study(Karami, Prasad, and Shehata 2017)	SIFT	Image Matching	62.89

4.2. Approach 2

The next research section focuses on processing hand images with extracted features to compute homomorphic encryption. Since basic mathematical operations can be

performed with, HE in vectors, distance-based similarity approaches such as Euclidean and Minkowski are implemented with SEAL. Thus, encrypted feature vectors are compared based on distances to see how similar they are, as in neighborhood- based classification approaches. The results of these studies can be seen in Table 4.5.

By using homomorphic encryption TENSEAL to encrypt the extracted samples that passed through SIFT, the new size of the extracted sample was 1039x1384 which was smaller than the original and running it through HE allowed us to encrypt the outputs to secure them and process a distance measure to evaluate the result with the time taken for processing, As can be seen in Table 4.5, we implemented a distance measure like Euclidean over our data sample which provided improved security. The result of homomorphic encryption over Euclidean to measure the distances shows that it consumes more time than direct Euclidean encryption with the original 1600x1200 data.

As expected, computation time has increased by working with HE, but this is a scaling problem that can be improved in the cloud without sacrificing security. And the average distances of the data that are above a certain threshold can be used for identification.

Table 4.5 Comparison Euclidean timing with Homomorphic encryption and without

People vs People Differences		
	Time (sec)	**Average Distance**
Euclidean Distance	0.368	0.71
Euclidean over HE	23.441	1.04
Self-Verification		
	Time(sec)	**Average Distance**
Euclidean Distance	0.364	0.52
Euclidean over HE	23.434	0.86

5. CONCLUSION

As a result of the study, our approach has shown that optimal security through homomorphic encryption and identification management with SIFT is required to secure biometric identity. In our approach, we address the use of the patterns without changing the actual data by implementing the homomorphic encryption, Firstly we implemented SIFT to seek for rate of accurate similarity between samples of the datasets, our findings was the different samples to tested was lower rate of similarity and for the same people's sample displayed a higher rate of similarity and identification, Secondly since we obtained that result we moved forward to the second approach which was processing and measuring the distance using Euclidean and Minkowski by implementing throughout TENSEAL encrypted samples, our findings here also shows that using homomorphic with the measurement of the distance gives expected result for average distance and more higher than the samples that uses without homomorphic encryption as normal Euclidean distance measurement for direct samples .Lastly, In future studies, there is a gap in improving the timing in the performance of homomorphic encryption, and we will study and improve the timing of homomorphic encryption when using biometric data. The authors also plan to extend this study with other biometric identification data such as vain, palm and iris identification.

REFERENCES

Acar, A., Aksu, H., Uluagac, A. S., & Conti, M. (2018). A Survey on Homomorphic Encryption Schemes. ACM Computing Surveys, 51(4), 1–35. https://doi.org/10.1145/3214303

Afifi, M. (2019). 11K Hands: Gender recognition and biometric identification using a large dataset of hand images. Multimedia Tools and Applications, 78(15), 20835–20854.

Anggriane, S. M., Nasution, S. M., & Azmi, F. (2017). Advanced e-voting system using Paillier homomorphic encryption algorithm. 2016 International Conference on Informatics and Computing, ICIC 2016, Icic, 338–342. https://doi.org/10.1109/IAC.2016.7905741

Benaissa, A., Retiat, B., Cebere, B., & Belfedhal, A. E. (2021). TenSEAL: A Library for Encrypted Tensor Operations Using Homomorphic Encryption. 1–12. http://arxiv.org/abs/2104.03152

Bozduman, H. Ç., & Afacan, E. (2020). Simulation of a Homomorphic Encryption System. Applied Mathematics and Nonlinear Sciences, 5(1), 479–484. https://doi.org/10.2478/amns.2020.1.00046

Cheon, J. H., Kim, A., Kim, M., & Song, Y. (2017). Homomorphic encryption for arithmetic of approximate numbers. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10624 LNCS, 409–437. https://doi.org/10.1007/978-3-319-70694-8_15

Choudhury, B., Then, P., Issac, B., Raman, V., & Haldar, M. K. (2018). A Survey on Biometrics and Cancelable Biometrics Systems. International Journal of Image and Graphics, 18(1). https://doi.org/10.1142/S0219467818500067

Dargan, S., & Kumar, M. (2020). A comprehensive survey on the biometric recognition systems based on physiological and behavioral modalities. Expert Systems with Applications, 143, 113114. https://doi.org/10.1016/j.eswa.2019.113114

Das, R. (2018). The Science of Biometrics. In The Science of Biometrics. https://doi.org/10.4324/9780429487583

Devi, M. K. K. (2017). Homomorphic Encryption-State of the Art.

Dube, R. (2008). Hardware-Based Computer Security Techniques to Defeat Hackers. 254. http://doi.wiley.com/10.1002/9780470425497

Faheem, M., Jamel, S., Hassan, A., A., Z., Shafinaz, N., & Mat, M. (2017). A Survey on the Cryptographic Encryption Algorithms. International Journal of Advanced Computer Science and Applications, 8(11). https://doi.org/10.14569/ijacsa.2017.081141

Fan, J., & Vercauteren, F. (2012). Somewhat practical fully homomorphic encryption. IACR Cryptol. EPrint Arch., 2012, 144.

Imran, O. A., Yousif, S. F., Hameed, I. S., Al-Din Abed, W. N., & Hammid, A. T. (2020). Implementation of El-Gamal algorithm for speech signals encryption and decryption. Procedia Computer Science, 167(Iccids 2019), 1028–1037. https://doi.org/10.1016/j.procs.2020.03.402

Jorgensen, Z., & Yu, T. (2011). On mouse dynamics as a behavioral biometric for authentication. Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, 476–482.

Karami, E., Prasad, S., & Shehata, M. (2017). Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images.

Kr, D., Kr, V., & Trivedi, M. C. (2020). Materials Today : Proceedings Encryption algorithm in cloud computing. Materials Today: Proceedings, xxxx. https://doi.org/10.1016/j.matpr.2020.07.452

Mert, A. C., Ozturk, E., & Savas, E. (2020). Design and Implementation of Encryption/Decryption Architectures for BFV Homomorphic Encryption Scheme. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 28(2), 353–362. https://doi.org/10.1109/TVLSI.2019.2943127

Mert, A. C., Öztürk, E., & Savaş, E. (2019). Design and implementation of encryption/decryption architectures for bfv homomorphic encryption scheme. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 28(2), 353–362.

Morris, L. (2013). Analysis of partially and fully homomorphic encryption. Rochester Institute of Technology, 1–5.

Ogburn, M., Turner, C., & Dahal, P. (2013). Homomorphic encryption. Procedia Computer Science, 20, 502–509. https://doi.org/10.1016/j.procs.2013.09.310

Omar, M. (2015). Cloud computing security: Abuse and nefarious use of cloud computing. Handbook of Research on Security Considerations in Cloud Computing, 30–38. https://doi.org/10.4018/978-1-4666-8387-7.ch002

Onwutalobi, A.-C. (2016). Overview of Cryptography. SSRN Electronic Journal, December. https://doi.org/10.2139/ssrn.2741776

Ora, P., & Pal, P. R. (2016). Data security and integrity in cloud computing based on RSA partial homomorphic and MD5 cryptography. IEEE International Conference on Computer Communication and Control, IC4 2015. https://doi.org/10.1109/IC4.2015.7375655

Paderes, R. E. O. (2016). A Comparative Review of Biometric Security Systems. Proceedings - 8th International Conference on Bio-Science and Bio-Technology, BSBT 2015, 8–11. https://doi.org/10.1109/BSBT.2015.12

Patel, V. M., Ratha, N. K., & Chellappa, R. (n.d.). IEEE SIGNAL PROCESSING MAGAZINE, VOL. X, NO. X, MONTH 20XX 1 Cancelable Biometrics: A Review. X(X), 1–25.

Rafie, R. (2016). Implementation of biometric authentication methods for home based systems. April.

Rathgeb, C., & Uhl, A. (2011). A Survey on Biometric Cryptosystems. 1–25.

Reza, S. M., & Rahad, K. (2013). A Study on Network Security Services with Cryptography and an Implementation of Vigenere-Multiplicative Cipher. Proceedings of the Workshop on Information and Communication Technology, December 2019.

Sharma, R., Sharma, R., & Singh, H. (2012). Classical Encryption Techniques. International Journal of Computers & Technology, 3(1), 84–90. https://doi.org/10.24297/ijct.v3i1b.2745

Singh, J. (2014). Cyber-Attacks in Cloud Computing: A Case Study. International Journal of Electronics and Information Engineering, 1(2), 78–87.

Soni, G., Gupta, U., & Singh, N. (2013). Analysis of Modified Substitution Encryption Techniques.

Subashini, S., & Kavitha, V. (2011). A survey on security issues in service delivery models of cloud computing. Journal of Network and Computer Applications, 34(1), 1–11. https://doi.org/10.1016/j.jnca.2010.07.006

Tripathi, P.K. (2011). A Comparative Study of Biometric Technologies with Reference to Human Interface. International Journal of Computer Applications, 14(5), 10–15. https://doi.org/10.5120/1842-2493

Xiong, L., Dong, D., Xia, Z., & Chen, X. (2018). High-Capacity Reversible Data Hiding for Encrypted Multimedia Data with Somewhat Homomorphic Encryption. IEEE Access, 6, 60635–60644. https://doi.org/10.1109/ACCESS.2018.2876036

Yanbin, H., Jianqin, Y., & Jinping, L. (2008). Human face feature extraction and recognition base on SIFT. Proceedings - International Symposium on Computer Science and Computational Technology, ISCSCT 2008, 1, 719–722. https://doi.org/10.1109/ISCSCT.2008.249

Yang, J., & Chen, Z. (2010). Cloud computing research and security issues. 2010 International Conference on Computational Intelligence and Software Engineering, CiSE 2010. https://doi.org/10.1109/CISE.2010.5677076

Zhi-gang, C., Jian, W., Liqun, C., & Xin-xia, S. (2014). Review of how to construct a fully homomorphic encryption scheme. International Journal of Security and Its Applications, 8(2), 221–230. https://doi.org/10.14257/ijsia.2014.8.2.23

APPENDIX

Appendix A - Implementation of SIFT using FLANN matcher on the dataset

Appendix B - Extracting Key points, Detecting Descriptors

Appendix C - Implementation of TENSEAL with distance measurements

Appendix A - Implementation of SIFT Using FLANN Matcher on The Dataset

```
def Flann_match(img1, img2, des1, des2, kp1, kp2):
    #find matches using FLANN
    FLANN_INDEX_KDTREE = 0
    index_params = dict(algorithm = FLANN_INDEX_KDTREE, trees = 5)
    search_params = dict(checks = 50)

    flann = cv2.FlannBasedMatcher(index_params, search_params)
    matches = flann.knnMatch(des1,des2,k=2)

#apply ratio test to find best matches (values from 0.7-1 made sense here)
    good = []
    for m,n in matches:
        if m.distance < 0.8*n.distance:
            good.append(m)

#find homography to transform the edges of the query image and draw them on the
train image
#This is also used to mask all keypoints that aren't inside this box further below.
    src_pts = np.float32([ kp1[m.queryIdx].pt for m in good]).reshape(-1,1,2)
    dst_pts = np.float32([ kp2[m.trainIdx].pt for m in good]).reshape(-1,1,2)

    M, mask = cv2.findHomography(src_pts, dst_pts, cv2.RANSAC,5.0)
    matchesMask = mask.ravel().tolist()

    h,w = img1.shape
    pts = np.float32([ [0,0],[0,h-1],[w-1,h-1],[w-1,0] ]).reshape(-1,1,2)
    dst = cv2.perspectiveTransform(pts,M)
    img2 = cv2.polylines(img2,[np.int32(dst)],True,255,3, cv2.LINE_AA)
  return(img2, matchesMask, good)
```

Appendix B - Extracting Key points, Detecting Descriptors

```
# start time
start_time = time.perf_counter()
MIN_MATCH_COUNT = 100
img1 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(1).jpg",0)
img2 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(2).jpg",0)
img3 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(3).jpg",0)
img4 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(4).jpg",0)
img5 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(5).jpg",0)
img6 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(6).jpg",0)
img7 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(7).jpg",0)
img8 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(8).jpg",0)
img9 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(9).jpg",0)
img10 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(10).jpg",0)
img11 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(11).jpg",0)
img12 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(12).jpg",0)
img13 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(13).jpg",0)
img14 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(14).jpg",0)
img15 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(15).jpg",0)
img16 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(16).jpg",0)
img17 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(17).jpg",0)
img18 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(18).jpg",0)
img19 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(19).jpg",0)
img20 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(20).jpg",0)
img21 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(21).jpg",0)
img22 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(22).jpg",0)
```

```
img23 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(23).jpg",0)
img24 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPLH/LeftDiffPos(24).jpg",0)
#img25 = cv2.imread(r"C:\Users\abdi_\Desktop\Trail
Dataset\Hands\Hands\Preprocessed\SDPRH/RightDiffPos(25).jpg",0)

# Initiate SIFT detector
sift =  cv2.xfeatures2d.SIFT_create()

# find the keypoints and descriptors with SIFT
kp1, des1 = sift.detectAndCompute(img1,None)
kp2, des2 = sift.detectAndCompute(img2,None)

kp3, des3 = sift.detectAndCompute(img3,None)
kp4, des4 = sift.detectAndCompute(img4,None)

kp5, des5 = sift.detectAndCompute(img5,None)
kp6, des6 = sift.detectAndCompute(img6,None)

kp7, des7 = sift.detectAndCompute(img7,None)
kp8, des8 = sift.detectAndCompute(img8,None)

kp9, des9 = sift.detectAndCompute(img9,None)
kp10, des10 = sift.detectAndCompute(img10,None)

kp11, des11 = sift.detectAndCompute(img11,None)
kp12, des12 = sift.detectAndCompute(img12,None)

kp13, des13 = sift.detectAndCompute(img13,None)
kp14, des14 = sift.detectAndCompute(img14,None)

kp15, des15 = sift.detectAndCompute(img15,None)
kp16, des16 = sift.detectAndCompute(img16,None)

kp17, des17 = sift.detectAndCompute(img17,None)
kp18, des18 = sift.detectAndCompute(img18,None)

kp19, des19 = sift.detectAndCompute(img19,None)
kp20, des20 = sift.detectAndCompute(img20,None)

kp21, des21 = sift.detectAndCompute(img21,None)
kp22, des22 = sift.detectAndCompute(img22,None)

kp23, des23 = sift.detectAndCompute(img23,None)
kp24, des24 = sift.detectAndCompute(img24,None)

#kp25, des25 = sift.detectAndCompute(img25,None)

# Flann points for Image 1 and 2
```

```
result= Flann_match(img1, img2, des1, des2, kp1, kp2)
img2, matchesMask, good = result

#draw the good matched key points
draw_params = dict(matchColor = None, # draw matches in green color
                   singlePointColor = None,
                   matchesMask = matchesMask[:100], # draw only inliers
                   flags = 2)

# points for Image 3 and 4
result1= Flann_match(img3, img4, des3, des4, kp3, kp4)
img4, matchesMask1, good1 = result1

#draw the good matched key points
draw_params1 = dict(matchColor = None, # draw matches in green color
                   singlePointColor = None,
                   matchesMask = matchesMask1[:100], # draw only inliers
                   flags = 2)

# points for Image 5 and 6
result2= Flann_match(img5, img6, des5, des6, kp5, kp6)
img6, matchesMask2, good2 = result2

#draw the good matched key points
draw_params2 = dict(matchColor = None, # draw matches in green color
                   singlePointColor = None,
                   matchesMask = matchesMask2[:100], # draw only inliers
                   flags = 2)

# points for Image 7 and 8
result3= Flann_match(img7, img8, des7, des8, kp7, kp8)
img8, matchesMask3, good3 = result3

#draw the good matched key points
draw_params3 = dict(matchColor = None, # draw matches in green color
                   singlePointColor = None,
                   matchesMask = matchesMask3[:100], # draw only inliers
                   flags = 2)

# points for Image 9 and 10
result4= Flann_match(img9, img10, des9, des10, kp9, kp10)
img10, matchesMask4, good4 = result4

#draw the good matched key points
draw_params4 = dict(matchColor = None, # draw matches in green color
                   singlePointColor = None,
                   matchesMask = matchesMask4[:100], # draw only inliers
                   flags = 2)
```

```
# points for Image 11 and 12
result5= Flann_match(img11, img12, des11, des12, kp11, kp12)
img11, matchesMask5, good5 = result5

#draw the good matched key points
draw_params5 = dict(matchColor = None, # draw matches in green color
                    singlePointColor = None,
                    matchesMask = matchesMask5[:100], # draw only inliers
                    flags = 2)

# points for Image 13 and 14
result6= Flann_match(img13, img14, des13, des14, kp13, kp14)
img13, matchesMask6, good6 = result6

#draw the good matched key points
draw_params6 = dict(matchColor = None, # draw matches in green color
                    singlePointColor = None,
                    matchesMask = matchesMask6[:100], # draw only inliers
                    flags = 2)

# points for Image 15 and 16
result7= Flann_match(img15, img16, des15, des16, kp15, kp16)
img15, matchesMask7, good7 = result7

#draw the good matched key points
draw_params7 = dict(matchColor = None, # draw matches in green color
                    singlePointColor = None,
                    matchesMask = matchesMask7[:100], # draw only inliers
                    flags = 2)

# points for Image 17 and 18
result8= Flann_match(img17, img18, des17, des18, kp17, kp18)
img17, matchesMask8, good8 = result8

#draw the good matched key points
draw_params8 = dict(matchColor = None, # draw matches in green color
                    singlePointColor = None,
                    matchesMask = matchesMask8[:100], # draw only inliers
                    flags = 2)

# points for Image 19 and 20
result9= Flann_match(img19, img20, des19, des20, kp19, kp20)
img19, matchesMask9, good9 = result9

#draw the good matched key points
draw_params9 = dict(matchColor = None, # draw matches in green color
                    singlePointColor = None,
                    matchesMask = matchesMask9[:100], # draw only inliers
                    flags = 2)
```

```
# points for Image 21 and 22
result10= Flann_match(img21, img22, des21, des22, kp21, kp22)
img21, matchesMask10, good10 = result10

#draw the good matched key points
draw_params10 = dict(matchColor = None, # draw matches in green color
                singlePointColor = None,
                matchesMask = matchesMask10[:100], # draw only inliers
                flags = 2)

# points for Image 23 and 24
result11= Flann_match(img23, img24, des23, des24, kp23, kp24)
img23, matchesMask11, good11 = result11

#draw the good matched key points
draw_params11 = dict(matchColor = None, # draw matches in green color
                singlePointColor = None,
                matchesMask = matchesMask11[:100], # draw only inliers
                flags = 2)

output1 = cv2.drawMatches(img1,kp1,img2,kp2,good[:100],None,**draw_params)
output2 =
cv2.drawMatches(img3,kp3,img4,kp4,good1[:100],None,**draw_params1)
output3 =
cv2.drawMatches(img5,kp5,img6,kp6,good2[:100],None,**draw_params2)
output4 = cv2.drawMatches(img7,kp7,img8,kp8,good3[:100],None,
**draw_params3)
output5 =
cv2.drawMatches(img9,kp9,img10,kp10,good4[:100],None,**draw_params4)
output6 =
cv2.drawMatches(img11,kp11,img12,kp12,good5[:100],None,**draw_params5)
output7 =
cv2.drawMatches(img13,kp13,img14,kp14,good6[:100],None,**draw_params6)
output8 = cv2.drawMatches(img15,kp15,img16,kp16,good7[:100],None,
**draw_params7)
output9 =
cv2.drawMatches(img17,kp17,img18,kp18,good8[:100],None,**draw_params8)
output10 =
cv2.drawMatches(img19,kp19,img20,kp20,good9[:100],None,**draw_params9)
output11 =
cv2.drawMatches(img21,kp21,img22,kp22,good10[:100],None,**draw_params10)
output12 = cv2.drawMatches(img23,kp23,img24,kp24,good11[:100],None,
**draw_params11)

plt.figure(figsize=(20, 20))
plt.title('Matched Points 1 to 2')
plt.imshow(output1, 'gray'),
plt.show()
print("Detected Keypoints in same a: " + str(len(kp1)))
print("Detected Keypoints in same a: " + str(len(kp2)))
```

```
print("GOOD Matches:", len(good))
print("\nNumber of Matching Keypoints Between The Training and Query Images: ",
len(good)/ len(kp2)*100, "%")

plt.figure(figsize=(20, 20))
plt.title('Matched Points 3 to 4')
plt.imshow(output2, 'gray'),
plt.show()
print("Detected Keypoints in same a: " + str(len(kp3)))
print("Detected Keypoints in same a: " + str(len(kp4)))
print("GOOD Matches:", len(good1))
print("\nNumber of Matching Keypoints Between The Training and Query Images: ",
len(good1)/ len(kp3)*100, "%")

plt.figure(figsize=(20, 20))
plt.title('Matched Points 5 to 6')
plt.imshow(output3, 'gray'),
plt.show()
print("Detected Keypoints in same a: " + str(len(kp5)))
print("Detected Keypoints in same a: " + str(len(kp6)))
print("GOOD Matches:", len(good2))
print("\nNumber of Matching Keypoints Between The Training and Query Images: ",
len(good2)/ len(kp4)*100, "%")

plt.figure(figsize=(20, 20))
plt.title('Matched Points 7 to 8')
plt.imshow(output4, 'gray'),
plt.show()
print("Detected Keypoints in same a: " + str(len(kp7)))
print("Detected Keypoints in same a: " + str(len(kp8)))
print("GOOD Matches:", len(good3))
print("\nNumber of Matching Keypoints Between The Training and Query Images: ",
len(good3)/ len(kp5)*100, "%")

plt.figure(figsize=(20, 20))
plt.title('Matched Points 9 to 10')
plt.imshow(output5, 'gray'),
plt.show()
print("Detected Keypoints in same a: " + str(len(kp9)))
print("Detected Keypoints in same a: " + str(len(kp10)))
print("GOOD Matches:", len(good4))
print("\nNumber of Matching Keypoints Between The Training and Query Images: ",
len(good4)/ len(kp6)*100, "%")

plt.figure(figsize=(20, 20))
plt.title('Matched Points 11 to 12')
plt.imshow(output6, 'gray'),
plt.show()
print("Detected Keypoints in same a: " + str(len(kp11)))
print("Detected Keypoints in same a: " + str(len(kp12)))
```

```
print("GOOD Matches:", len(good5))
print("\nNumber of Matching Keypoints Between The Training and Query Images: ",
len(good5)/ len(kp7)*100, "%")

plt.figure(figsize=(20, 20))
plt.title('Matched Points 13 to 14')
plt.imshow(output7, 'gray'),
plt.show()
print("Detected Keypoints in same a: " + str(len(kp13)))
print("Detected Keypoints in same a: " + str(len(kp14)))
print("GOOD Matches:", len(good6))
print("\nNumber of Matching Keypoints Between The Training and Query Images: ",
len(good6)/ len(kp8)*100, "%")

plt.figure(figsize=(20, 20))
plt.title('Matched Points 15 to 16')
plt.imshow(output8, 'gray'),
plt.show()
print("Detected Keypoints in same a: " + str(len(kp15)))
print("Detected Keypoints in same a: " + str(len(kp16)))
print("GOOD Matches:", len(good7))
print("\nNumber of Matching Keypoints Between The Training and Query Images: ",
len(good7)/ len(kp9)*100, "%")

plt.figure(figsize=(20, 20))
plt.title('Matched Points 17 to 18')
plt.imshow(output9, 'gray'),
plt.show()
print("Detected Keypoints in same a: " + str(len(kp17)))
print("Detected Keypoints in same a: " + str(len(kp18)))
print("GOOD Matches:", len(good8))
print("\nNumber of Matching Keypoints Between The Training and Query Images: ",
len(good8)/ len(kp10)*100, "%")

plt.figure(figsize=(20, 20))
plt.title('Matched Points 19 to 20')
plt.imshow(output10, 'gray'),
plt.show()
print("Detected Keypoints in same a: " + str(len(kp19)))
print("Detected Keypoints in same a: " + str(len(kp20)))
print("GOOD Matches:", len(good9))
print("\nNumber of Matching Keypoints Between The Training and Query Images: ",
len(good9)/ len(kp11)*100, "%")

plt.figure(figsize=(20, 20))
plt.title('Matched Points 21 to 22')
plt.imshow(output11, 'gray'),
plt.show()
print("Detected Keypoints in same a: " + str(len(kp21)))
print("Detected Keypoints in same a: " + str(len(kp22)))
```

```
print("GOOD Matches:", len(good10))
print("\nNumber of Matching Keypoints Between The Training and Query Images: ",
len(good10)/ len(kp12)*100, "%")

plt.figure(figsize=(20, 20))
plt.title('Matched Points 23 to 24')
plt.imshow(output12, 'gray'),
plt.show()
print("Detected Keypoints in same a: " + str(len(kp23)))
print("Detected Keypoints in same a: " + str(len(kp24)))
print("GOOD Matches:", len(good11))
print("\nNumber of Matching Keypoints Between The Training and Query Images: ",
len(good11)/ len(kp13)*100, "%")


Total_Execution_Time = time.perf_counter() - start_time
print("Total Excution Time : ",Total_Execution_Time ,"Seconds")
```

Appendix C - Implementation of TENSEAL with Distance Measurements

```
# start time
start_time = time.perf_counter()
import tenseal as ts
context = ts.context(ts.SCHEME_TYPE.BFV, poly_modulus_degree=8192,
plain_modulus=1032193)
public_context = ts.context(ts.SCHEME_TYPE.BFV, poly_modulus_degree=8192,
plain_modulus=1032193)

sk = public_context.secret_key()
public_context.make_context_public()


encrypted_a_vector1= ts.bfv_vector(context,a_vector1)
encrypted_a_vector2= ts.bfv_vector(context,a_vector2)
encrypted_a_vector1.size()
encrypted_a_vector2.size()


encrypted_a_vector1
encrypted_a_vector2


from scipy.spatial import distance
import numpy as np
encrypted_a_vector1 = np.random.rand(1,3)
encrypted_a_vector2 = np.random.rand(3,1)
dst = distance.euclidean(encrypted_a_vector1, encrypted_a_vector2)
Total_Execution_Time = time.perf_counter() - start_time
print("Total Excution Time : ",Total_Execution_Time ,"Seconds")
print("Euclidean distance : ",dst)
```

BIBLIOGRAPHY

Full Name : Abdulrahim Mohamed IBRAHIM

Educational Status

Computer Science BSc - SIMAD UNIVERSITY 2018

Computer Engineering MSc – TICARET UNIVERSITY ongoing

Publications

Ozpinar, A., Ibrahim, A.M., 2021. Securing and Processing Biometric Data with Homomorphic Encryption for Cloud Computing, International Conference on Artificial Intelligence and Applied Mathematics in Engineering (ICAIAME 2021). 1-2-3 October, Antalya, Turkey.