Report

Biometric Template Protection using Fuzzy Vault

~ *Sagnik Chakraborty* [22101106020]

1. Introduction

The fuzzy vault is a cryptographic technique designed to secure biometric templates such as fingerprints and palmprints. These templates are unique to individuals and provide a reliable means of authentication. However, protecting this sensitive biometric data from unauthorized access is crucial. The fuzzy vault provides a mechanism to encode biometric features with a secret key, ensuring that only authorized users can access the stored information.

The fuzzy vault is built on principles of error-correcting codes, where the biometric data is encoded into a vault using a polynomial equation. By adding false or "chaff" points, the genuine biometric points are hidden, enhancing security. The vault can only be decoded with the correct key, ensuring that sensitive biometric information is protected from adversaries.

2. Fuzzy Vault: Concepts and Workflow

2.1 Key Concepts

1. **Biometric Template**: A digital representation of the unique features of an individual's fingerprint or palmprint.
2. **Minutiae Points**: Features extracted from biometric data (e.g., ridge endings, bifurcations in fingerprints) that are used for template encoding.
3. **Chaff Points**: Randomly added points to obscure the real biometric points, improving security.
4. **Galois Field (GF)**: A mathematical field used for encoding biometric data to ensure secure polynomial evaluations.

2.2 Workflow of Fuzzy Vault Implementation

The fuzzy vault follows a series of steps to secure biometric templates:

5. **Feature Extraction**:

   - Biometric features, such as minutiae points and orientation fields, are extracted from the raw biometric data (fingerprint or palmprint). These points are essential for generating the vault.

6. **Feature Selection**:

   - A subset of the extracted minutiae points is chosen based on the Fuzzy Vault parameter r (number of minutiae points considered for vault generation).

7. **Normalization**:

    - The selected points are normalized using parameters such as B_u, B_v, and B_θ, corresponding to the x, y, and angular orientations.

8. **Key Generation**:

    - A unique user-specific key of length 16×16 bits is generated. A CRC checksum is appended, forming a 272-bit key.

9. **Polynomial Construction**:

    - The 272-bit key is divided into 16-bit segments and used as coefficients of a polynomial of degree 16. The polynomial forms the basis for vault encoding.

10. **Point Evaluation and Encoding**:

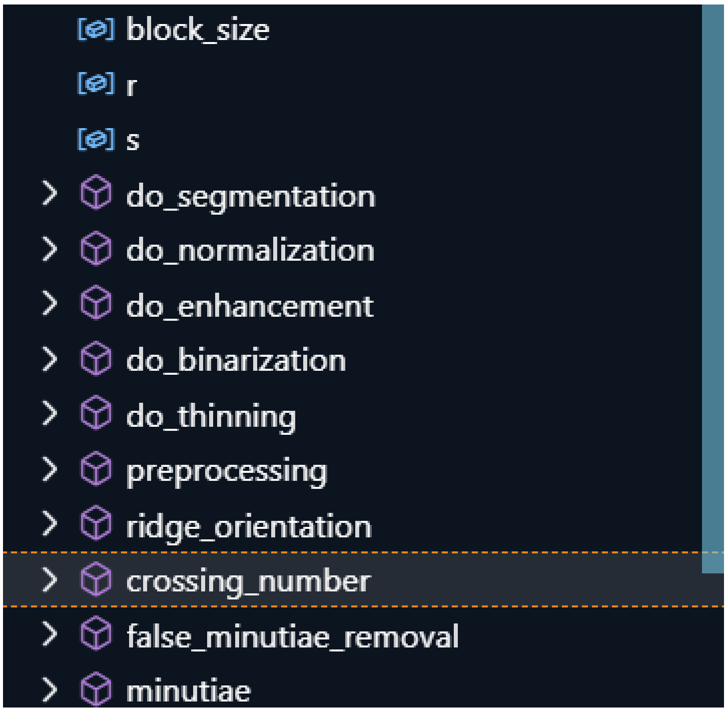
    - The biometric features are encoded and evaluated in the Galois Field $GF(2^{16})$, ensuring secure transformation of minutiae points into vault points.

11. **Adding Chaff Points**:

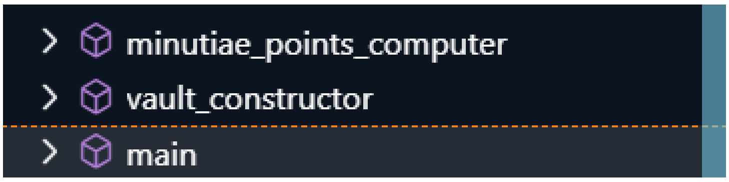
    - To provide additional security, chaff points (random points) are added. These points must not overlap with the genuine minutiae points or their polynomial evaluations.

12. **Vault Construction**:

    - The genuine minutiae points and chaff points are combined to form the final Fuzzy Vault, hiding the real biometric information among false points.



block_size
r
s
do_segmentation
do_normalization
do_enhancement
do_binarization
do_thinning
preprocessing
ridge_orientation
crossing_number
false_minutiae_removal
minutiae



3. Implementation Details

3.1 Feature Extraction

In this implementation, fingerprint and palmprint images are used as inputs. The minutiae points are extracted using functions that apply techniques like binarization, thinning, and orientation field analysis. The block size of 16 is considered optimal for minutiae extraction. The key stages of preprocessing include segmentation, normalization, enhancement, and feature thinning.

3.2 Minutiae Selection and Encoding

After minutiae extraction, a subset of 20 minutiae points (`r=20`) is selected. These points are normalized and encoded using MinMax scaling for x, y coordinates (B_u=`6 bits`, B_v=`6 bits`) and angular orientations (B_θ= `4 bits`), resulting in 16-bit representations of each point.

3.3 Key and Polynomial Generation

The user-specific key is 272 bits long, including a CRC checksum for error detection. The key is divided into segments to form the coefficients of a degree-16 polynomial, which is evaluated using the Galois Field *GF(216)GF(2^{16})*GF(216). This ensures secure encoding of the biometric points.

3.4 Vault Construction and Security

To enhance security, 180 chaff points (`s=180`) are randomly generated. These chaff points are indistinguishable from the genuine minutiae points, making it difficult for an attacker to isolate the real points. The genuine and chaff points together form the final fuzzy vault.

4. Code Overview

4.1 Functionality

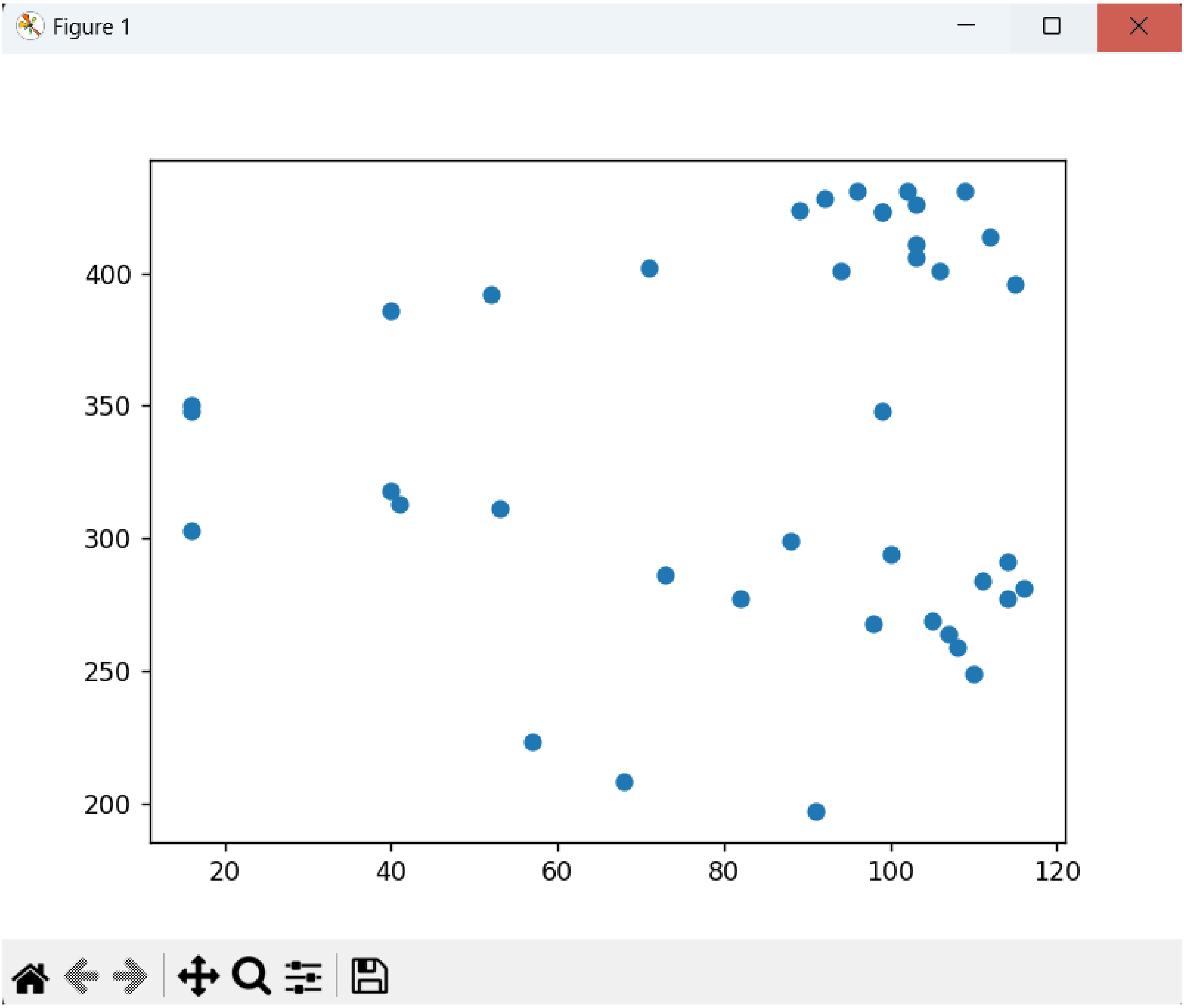
- **Minutiae Extraction**: The function `minutiae_points_computer` processes fingerprint and palmprint images to extract minutiae points using image processing techniques like skeletonization and orientation mapping.
- **Vault Construction**: The `vault_constructor` function takes the extracted minutiae and user-specific key to generate a polynomial and evaluate feature points using Galois Field encoding. Chaff points are added to conceal the genuine minutiae.
- **Visualization**: A scatter plot is generated to visualize the vault, showing both genuine and chaff points, using `matplotlib`.

4.2 Libraries Used

- `numpy`: For matrix operations and numerical computation.
- `cv2`: For image processing (OpenCV).
- `math`: Standard mathematical functions for computations.
- `fingerprint_enhancer`: For enhancing fingerprint images to make minutiae extraction more reliable.
- `scikit-learn`: For scaling and normalization of extracted biometric features.
- `galois`: For Galois Field operations used in polynomial encoding and evaluation.
- `crc`: For generating and verifying CRC checksums in key generation.
- `matplotlib`: For generating scatter plots to visualize genuine and chaff points.
- `warnings`: For managing warnings during execution.
- `os`: For handling file operations, including reading and saving images.
- `random`: For generating random values and chaff points to hide genuine biometric points.

5. Parameters and Results

- **Fuzzy Vault Parameters**:
  - `r = 20`: Number of minutiae points selected.
  - `s = 180`: Number of chaff points added for security.
  - $B_u = 6$ bits, $B_v = 6$ bits, $B_\theta = 4$ bits: Parameters for minutiae point normalization.
  - **Total Sum**: 16 bits (6 + 6 + 4) per minutiae point.



Result:

The output of the implementation is a scatter plot displaying both genuine and chaff points. The genuine minutiae points are securely hidden among the chaff points, making it difficult for an unauthorized user to extract the real biometric data. This multimodal fuzzy vault (combining fingerprint and palmprint) enhances the security of biometric templates.

6. Conclusion

The fuzzy vault is a powerful method for securing biometric templates by encoding them with a secret key. By hiding genuine biometric data among false points and using polynomial-based encoding, the fuzzy vault ensures that sensitive biometric information is protected. The multimodal approach, using both fingerprint and palmprint data, further strengthens security. This implementation demonstrates the effectiveness of the fuzzy vault in protecting biometric data from unauthorized access.

7. References

13. K. Nandakumar, A. K. Jain, and S. Pankanti, “Fingerprint-based Fuzzy Vault: Implementation and Performance,” *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 4, pp. 744–757, 2007.
14. A. Juels and M. Sudan, “A Fuzzy Vault Scheme,” *Proceedings of IEEE International Symposium on Information Theory*, vol. 6, no. 3, pp. 408, 2002.
15. K. Nandakumar, “Multibiometric System: Fusion Strategies and Template Security,” *PhD Thesis*, Department of Computer Science and Engineering, Michigan State University, January 2008.
16. V. Evelyn Brindha, “Biometric Template Security using Fuzzy Vault,” *IEEE 15th International Symposium on Consumer Electronics*, 2011.

This report provides a comprehensive overview of the fuzzy vault technique for biometric template protection, including the key steps and logic behind the implementation. The references listed are seminal works that provide deeper insights into the underlying concepts.