www.ijecs.in
International Journal Of Engineering And Computer Science
Volume 13 Issue 08 August 2024, Page No. 26325-26337
ISSN: 2319-7242 DOI: 10.18535/ijecs/v13i08.48474

Image Steganography based on Fernet Symmetric Encryption and Odd-Even Pixel Modification

Deena Faria.[1], Habiba Sultana.[1], A.H.M. Kamal

[1] Department of Computer Science and Engineering, Jatiya Kabi Kazi Nazrul Islam University, Mymensingh 2220, Bangladesh

Abstract

The combination of steganography and cryptographic approaches for information concealing has piqued the curiosity of numerous researchers. We also integrate those two methods in our research. Using Fernet symmetric encryption and odd-even pixel alteration, this research presents a novel method for image steganography that distributes the data equally throughout the image. To create a stego image, the sender uses a password to encrypt a secret message that is then embedded into an image's red channel. The stego image bears a visual resemblance to the original cover image, while the underlying secret data is concealed from human view. The main goals of the system—encryption, consistent data concealing, and message retrieval—were all successfully met. Before integrating the secret data into the image, it can be easily and successfully secured by using symmetric encryption using the Fernet cipher. By adding another layer of security to the system, the password feature makes it harder for unauthorized users to access hidden data. By altering image pixels according to whether they are odd or even, the odd-even pixel modification-based steganography approach makes it possible to conceal data. Based on the experiment's results, it can be concluded that this approach embeds concise messages with a high visual quality.

Key Words: Odd-Even pixel modification, RGB image, Steganography, Uniform data distribution.

1. Introduction

For secured communication, information must be protected. Among various techniques information hiding and cryptography are the two main branches for invulnerable communication [1-5]. Cryptography is a technique where sender transforming or encrypting the plaintext or secret message into an unreadable format called ciphertext [2]. In information hiding, sender implants the secret messages into a cover media then form as stego media. This stego media is sent to the receiver. Information hiding can be classified as watermarking, fingerprinting and steganography [2-4]. Steganography is a Greek word which means ‘covered writing’ that is to hide any secret messages within a cover media. The most commonly used cover media are image, audio, video and text etc. Among them image is the most popular cover media.

The performance of steganography system can be measured by some parameters such as imperceptibility, payload and robustness. Steganography system can be classified as reversible and irreversible technique. In reversible scheme, sender extracts both secret message and cover media from stego media. On the other hand, irreversible schemes only extract the secret messages. This research is focused on irreversible scheme.

Least Significant Bit (LSB) substitution is the most popular technique in the field of image steganography for data implanting [3,6-14]. In LSB steganography, secret data is hidden in the least significant bit of the cover image pixels. Since the least significant bit is altered, there is no significant amount of changes in the image. While LSB is simple and efficient, it has the disadvantage of low embedding capacity and vulnerability to detection. There have been many works done on LSB based steganography as well as its steganalysis. Thangadurai (2014) provides an overall analysis on LSB based image steganography and its application to various file formats. Transform domain techniques, for example, Discrete Cosine Transform (DCT) and

Discrete Wavelet Transform (DWT), offer increased capacity for embedding and improved security. Goel (2013) conducted a research to demonstrate the results against LSB based steganography, DCT based steganography, and DWT based steganograph. LSB-based approaches are comparatively easier to detect than the other methods because of their simple nature. On the other hand, transform domain techniques offer better security, but they can suffer from increased computational complexity. Again deep learning methods seem to overcome these difficulties in terms of security and capacity but they have the disadvantage of requiring large datasets for training [15]. Although image steganography provides a secure means of communication, it still has vulnerability issues. Different methods also have been developed for steganalysis purposes. Deep learning is used for steganalysis with the advent of machine learning in recent time. Such as, Qian (2015) proposed a new method for steganalysis to learn the features of stego-media automatically through deep learning models that can detect suspicious content with high accuracy successfully. Therefore, the development of countermeasures to improve the security of steganography against detection is still an active research area [16-20]. Aggarwal et al. (2019) proposed an LSB based technique where secret message is embedded using a secret key on the image's red channel. Mahdi et al. (2019) proposed a technique based on P-Even/P-Odd on spatial domain and multilevel encryption. They compressed the secret data using Huffman coding before embedding into the cover image to increase the capacity. Fateh et al. (2021) proposed a new LSB matching revisited scheme for hiding image data. In this method, the secret data was first divided into groups and then these groups were hidden inside the image to increase the capacity. Ali (2019) proposed a technique of LSB steganography based on random bit substitution. They embedded the secret message at random positions of the image pixels and used a reference to identify those positions.

Our proposed method embeds data using a secret stego-key on image's red channel by distributing the data uniformly based on the secret message's length. It encrypts the data, calculates the data length and computes a special parameter called 'pixel_jump' based on the stego-key, number of image pixels, and the data length. Then based on this 'pixel_jump' parameter, it distributes the data on image's red channel using odd-even pixel modification technique. Experimental results suggest significant improvement over the other existing systems.

The key contributions of this paper are:

1. Significant improvement in the visual quality parameters, for example, PSNR, Payload, and, SSIM.

2. A new approach for distributing data across the image to acquire less statistical detectability.

3. Improved robustness by incorporating Fernet Encryption to the secret message.

The rest of the article contains several sections elaborating on the proposed method further. Section 2 contains related works in this area. Section 3 demonstrates the proposed method. Section 3 shows the experimental results. And finally, section 4 concludes the article.

2. Literature Review

Uniform data distribution refers to the embedding of the secret message evenly within the image to exploit the cover image's full potential. It can be performed by different techniques, such as, uniform embedding distortion function (UED), etc. Guo (2015) showed a method of embedding data uniformly using UED in JPEG images. (12) The goal of uniform distribution is to reduce the statistical susceptibility and thus enhance security. Moreover, the entire image can be exploited to embed the image leading to the reduction of the wastage created by unutilized image pixels.

The Odd-Even Pixel Modification approach basically works by modifying the cover image pixel based on whether it is odd or even. Based on the combination of the data bit and the image pixel value, the cover image pixels are changed accordingly. Mahdi (2019) applied this idea of p-even/p-odd modification on their improved system.

Aggarwal (2019) showed an LSB based technique where secret data has been embedded using a secret stego-key. In this existing system, the secret data was embedded into a cover image which was in bmp format. At first, each character of the secret data and each pixel of the cover image were transformed into their corresponding binary equivalents. The user had to enter the password and the stego key to the system. Stego-

key was applied to define the starting point of the cover image to embed the secret message. After embedding the secret data into the cover image, the stego-image was sent to the receiver party via the communication channel. The average of the ASCII characters of the stego-key value was calculated. Then the seed LSB position was defined according to the computed mean value of the user provided stego-key letters. Then the embedding process had been continued til the end of secret data.

Mahdi et al. (2019) proposed a technique based on P-Even/P-Odd on spatial domain and multilevel encryption. This improved steganography method was based on two random parameters and encryption system. P-Even/P-Odd classification based LSB system was used in this scheme. This method had been implemented deploying the idea of matching the secret bits with the LSB while embedding to determine 0 (i.e., P-Even) and 1 (i.e., P-Odd).

The encoding phase includes two processes: (i) Block Selection and (ii) Data Embedding. To hide the data into the cover image, these two processes ran simultaneously. At first, the whole cover image was divided into 8×8 blocks of 64×64 pixels. Then, using a Henon map function, the block and the pixel selection were done. Then the secret message was embedded using P-Even/P-Odd classification scheme in the cover image's LSBs. This method ensured a minimal distortion of the stego image by modifying cover image's LSBs.

The extracting algorithm retrieved the data from the stego image's LSB by using the secret key that was taken from the receiver side. It extracts the secret message from the lsb bits by determining the P-Even/P-odd pixel values which was reverse of the embedding process.

Fateh et al. (2021) proposed a method of LSB matching revisited for hiding image data. At first, they divided the secret data into groups of n bit. Then they took 2n−1 pixels of the cover image to conceal these groups. Their experiments showed that this method required some changes in the basic LSB method.

Ali (2019) proposed a technique of LSB steganography based random bit substitution. This scheme involved concealing the secret information at random postions of the cover image pixels to deceive the attackers or to prevent third party access. They used a pseudo random number generator (PRNG) to generate those random bit positions and embedded the secret message accordingly. Using this method, they achieved a significant improvement in PSNR values but it lacks consistency with respect to different cover images, meaning this method did not provide similar high PSNR for different cover images. Additionally, they showed experimental results for only three images.

3. Methods

We proposed a system that conceals data effectively for different images which also includes improved PSNR and other statistical parameters. In this research, we focused on building a simpler yet effective system to embed information that can work efficiently with different images consistently.

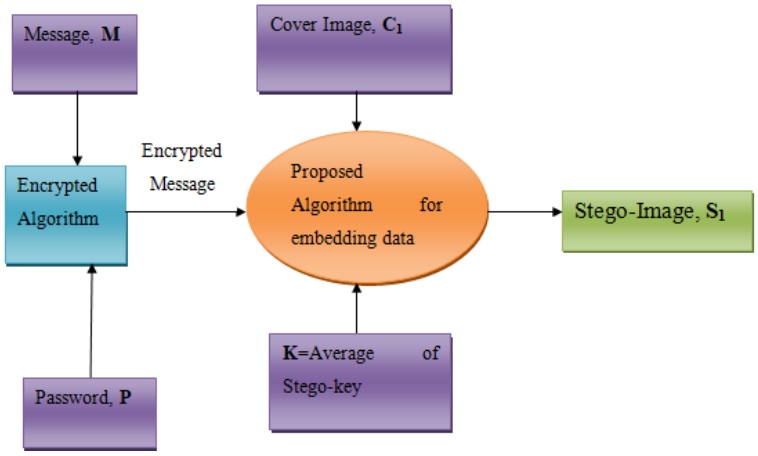

Figure1. Simplified Block Diagram of the proposed technique

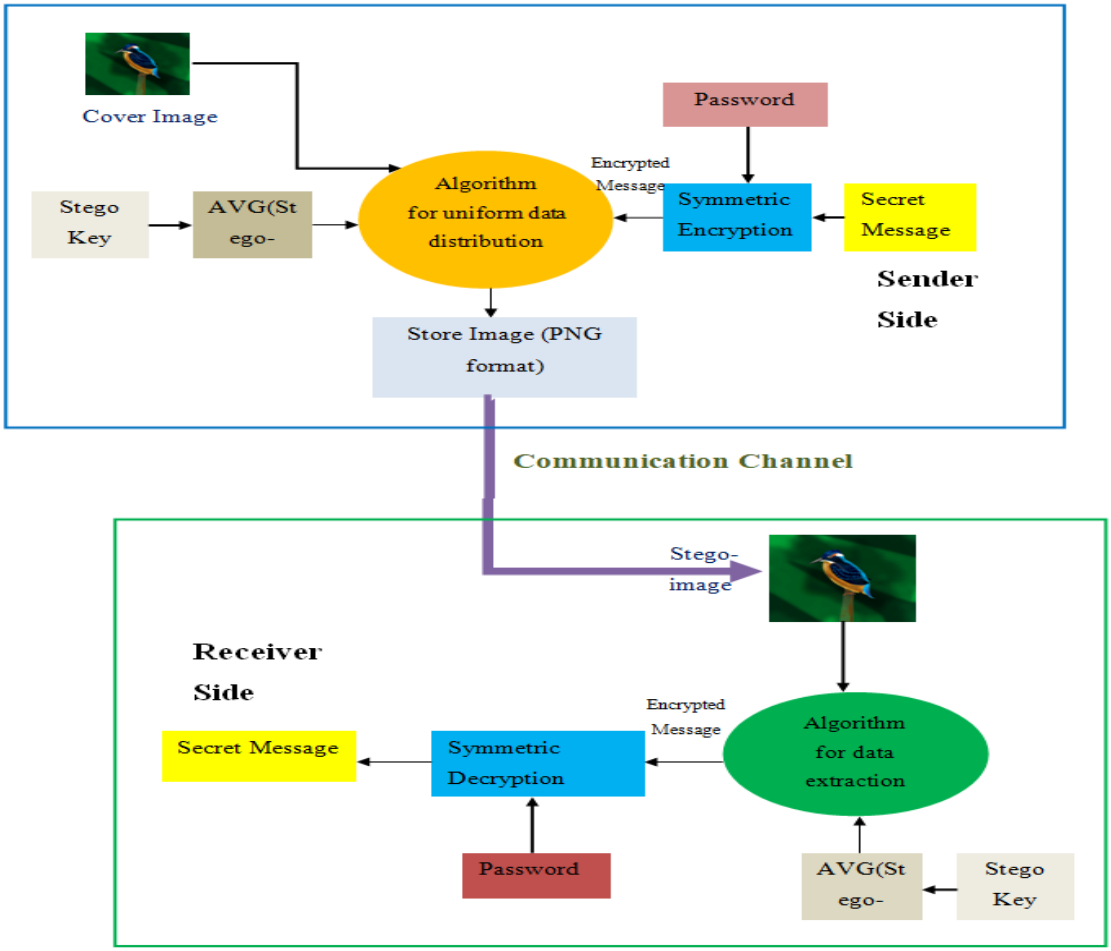

Figure2. Architectural design for the proposed system

In our proposed method, the system accepts four parameters from the user; which are (i) Secret message, (ii) Cover image, (iii) Password and (iv) Stego-key (as a string). Then, it encrypts the secret message using the password applying fernet symmetric encryption and calculates the average of the stego-key after converting it to its corresponding ASCII values. Then, it calculates the available pixels by subtracting this average value from the encoding capacity of the image (red channel). Then, if the length of the secret message is greater than the available pixels, it will create a new stego-key by performing modulo operation on the available pixels and the length of the message as "available pixels MOD length of the message". And this new stego key will be used as the seed or starting point for embedding the encrypted secret message. Otherwise, the first stego-key will be used. Then, based on the data length, a new parameter named "pixel_jump" is defined. Finally, using this "pixel_jump" parameter, the secret data is distributed uniformly across the image's red channel. Data length is hidden within the image's green channel. Blue channel remains untouched. In the receiver side, the reverse algorithm is performed to retrieve the secret message. The advantage of uniform distribution is it can help avoiding noticeable patterns that makes the hidden information less detectable both visually and statistically.

3.1. Secret Message Encryption Technique

1. Password Encryption: The user provides a password as input to the encode function.The function takes this password and applies a hash function (MD5) to it to generate a fixed-size hash value which is represented as a hexadecimal string. Hashing is a one-way process. That means that one cannot reverse it to get the actual password.

2. Generating Encryption Key: This hashed password is then encoded using base64 to make it a suitable format for the generation of the encryption key. This base64-encoded hash is then used as the encryption key for the Fernet symmetric encryption.

3. Fernet Cipher Creation: The function creates a Fernet cipher object with the encryption key generated from the password. The Fernet cipher is designed specifically for symmetric encryption and decryption.

4. Encryption/Decryption function: The encrypt decrypt function accepts three inputs: the string to be encrypted/decrypted, the password, and the mode(encryption/decryption).

5. Encryption for Modifying Secret Data: When the user selects for encrypting the secret data, the input text is passed to the encrypt decrypt function along with the provided password and the mode set to 'enc'. This way, the secret data is encrypted before embedding it into the image.

6. Decryption to Retrieve Secret Data: When the user selects for decrypting the secret data from an encoded image, the function decode is called with the Stego key, image path, password, and mode set to 'dec'. This decrypts the embedded secret data using the provided password and returns the original data.

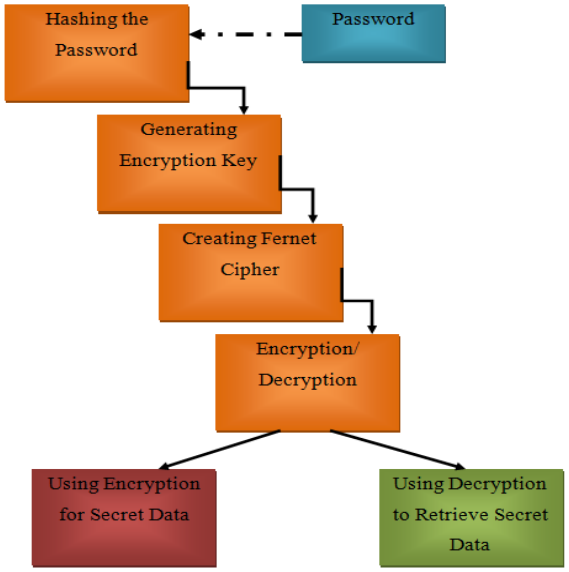

Figure3. Encryption/Decryption of the secret data

3.2 Secret Message Distribution Technique

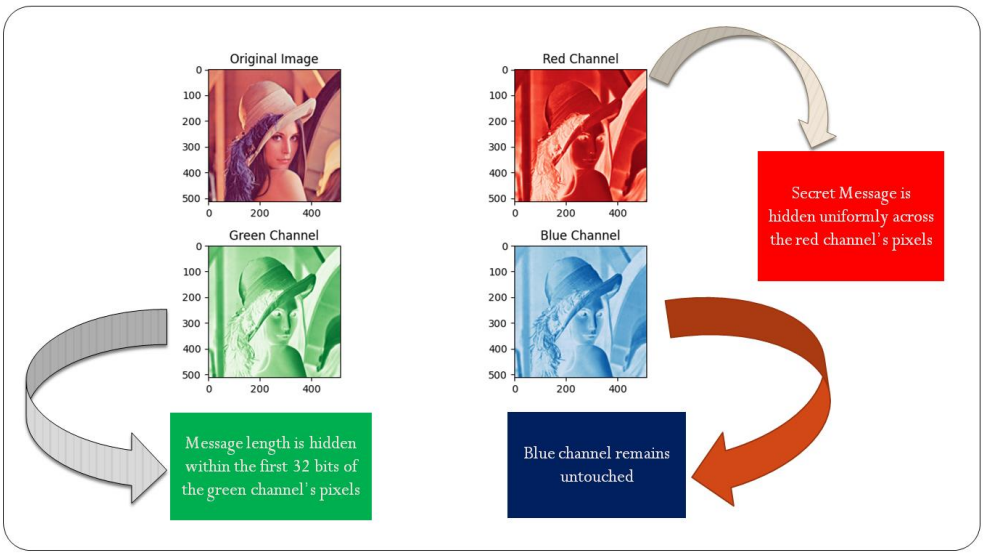

Figure4. Distribution of secret data in RGB image

1. Algorithm 1 for calculating stego-key
2. Calculate the size of the cover image C1, Encoding capacity = height*width
3. Calculate the length of the message M, LM
4. Compute K1=avg(K), where K is the user provided stego key
5. Calculate available pixels for encoding, LP=Encoding capacity-K1
6. If LM > LP, go to the next step.
7. Define a new key, K2 = LP Mod LM
8. Use K2 as the initial point for the secret message bit embedding position
9. Else use K1 as the initial point

Algorithm 2 for uniformly distributing the message across the image

1. Calculate the stego key using algorithm 1
2. Recalculate LP=Encoding capacity- new stego key
3. Compute pixel jump=LP/LM
4. Start modifying from the seed pixel (defined by the stego key)
5. Jump to the next pixel by an amount of pixel jump
6. Continue step 5 until the end of the data or the image pixels.

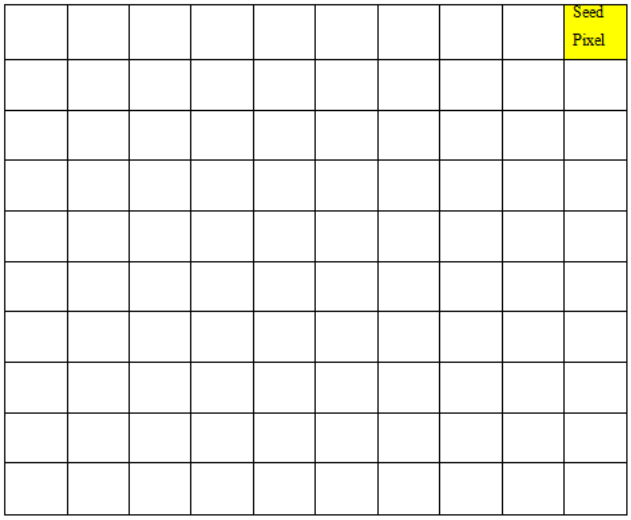

Figure5. Initializing the seed pixel using stego-key

Example: Let's assume the height and the width of the image are 10 and 10 pixels respectively. Hence, the encoding capacity will be, Encoding-capacity=height*width=10*10. User provides a stego key as a string. Our algorithm will convert each character to its corresponding ASCII value and then calculate the average of these values which will be used as the initial stego key. Let's assume the value is 10. Now, the available pixels, LP= Encoding-capacity-avg Therefore, LP=100-10=90 Let's assume the length of the message LM is, 5. 5 is less than 90, in this case we can retain the original stego key and calculate the values of the iterators accordingly. Therefore, pixel jump=LP/LM =90/5=18. In this way, the entire message will be uniformly distributed across the image's red channel.

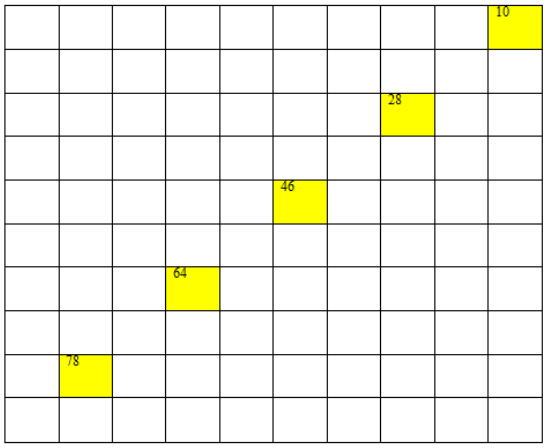

Figure6. Uniform distribution of the data

3.3. Secret Message Embedding Technique

Different embedding techniques have been evolved to ensure secure data hiding and transmission. Odd-Even pixel modification technique is one of them. In this technique, the pixel value of the image where data is to be hidden, is extracted and compared with the data bit. Then, based on certain combinations, the pixel value is modified. At the receiver end, the modified pixel values are retrieved and the hidden message is extracted. In figure 3, the entire process is shown.

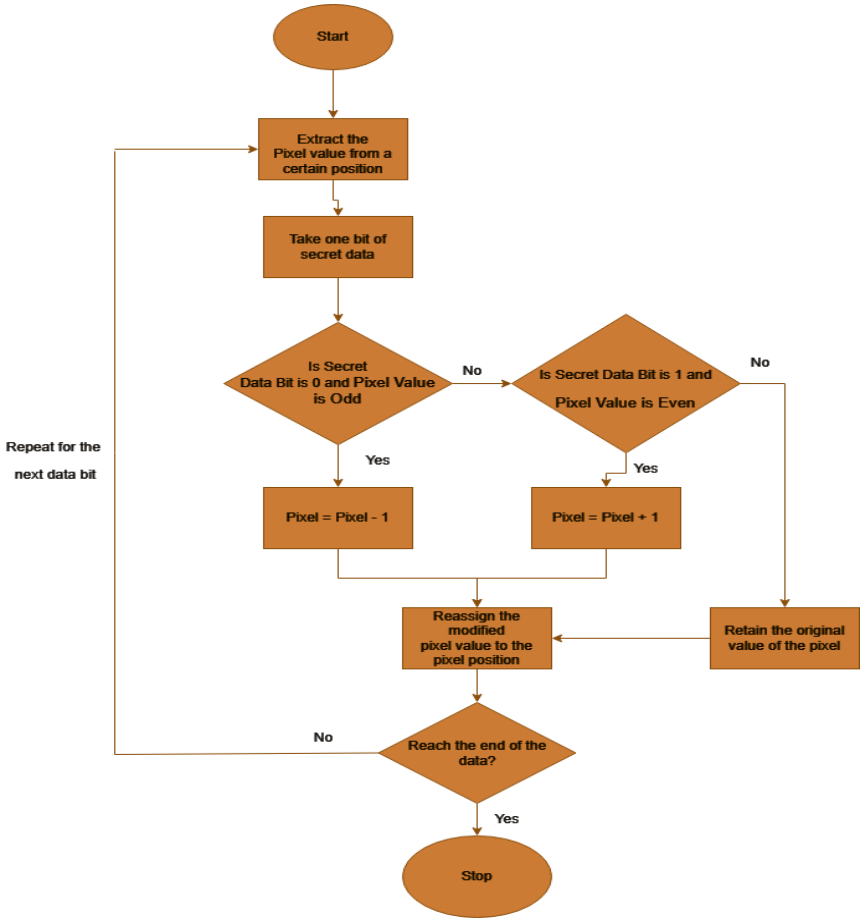

Figure7. Embedding Technique using Odd-Even Pixel Modification.

3.4. Secret Message Extraction Process

Reverse algorithm is performed to retrieve the data. First, the data length is extracted from the image's green channel and then the pixels where the secret data is embedded, is extracted using pixel_jump from the image's red channel. Then, symmetric decryption is applied to recover the original message.

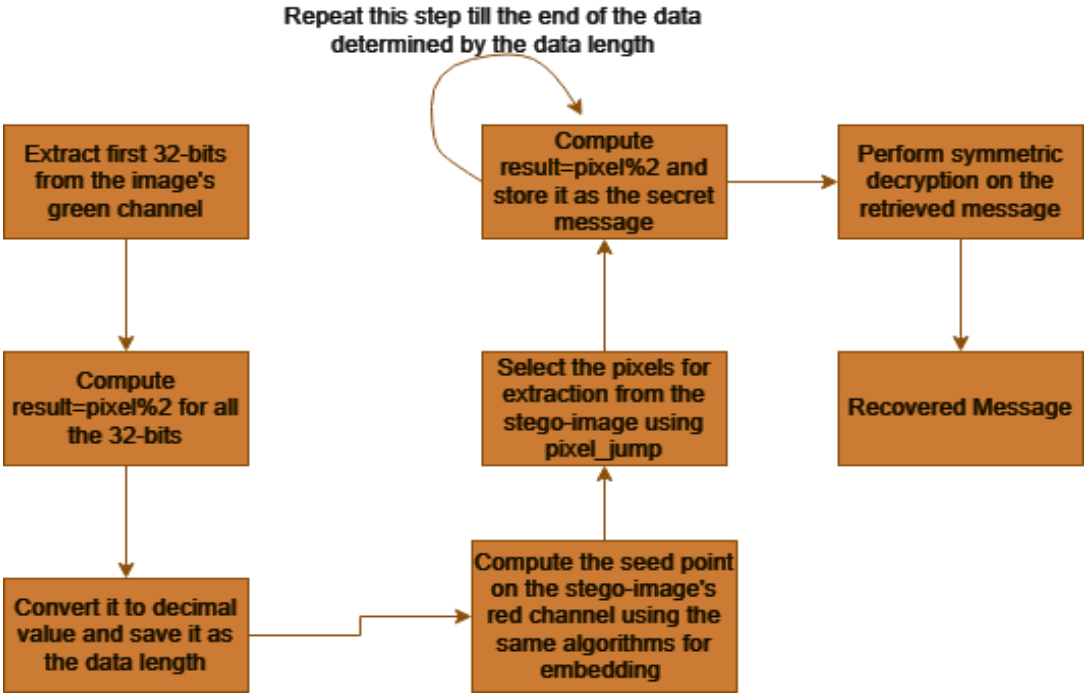

Figure8. Extraction of the secret data

4. Results and Discussion

This section shows the experimental results conducted to evaluate the performance of the proposed scheme compared with the works of Mahdi et al. (2019), Fateh et al. (2021), and Ali et al. (2019). The output we provided shows the progress that we have made in this research. Users have to provide a cover image, a secret data, a password, and a stego key. The system encrypts the data using fernet encryption using the password and the stego key is used to determine the initial point on the cover image for embedding. Then the encrypted data is embedded uniformly across the cover image's red channel. Data length is hidden in the image's green channel.

Test Data: Secret Message: We used following text as the secret data:

> Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Figure9. Test data

Stego Key: We used the following stego-key for all of the test images: cat

Test Images: we used following set of images for testing purpose. We have collected the images from various datasets like scikit-image and LLSD (Large Scale Steganalysis Database) [21-24].

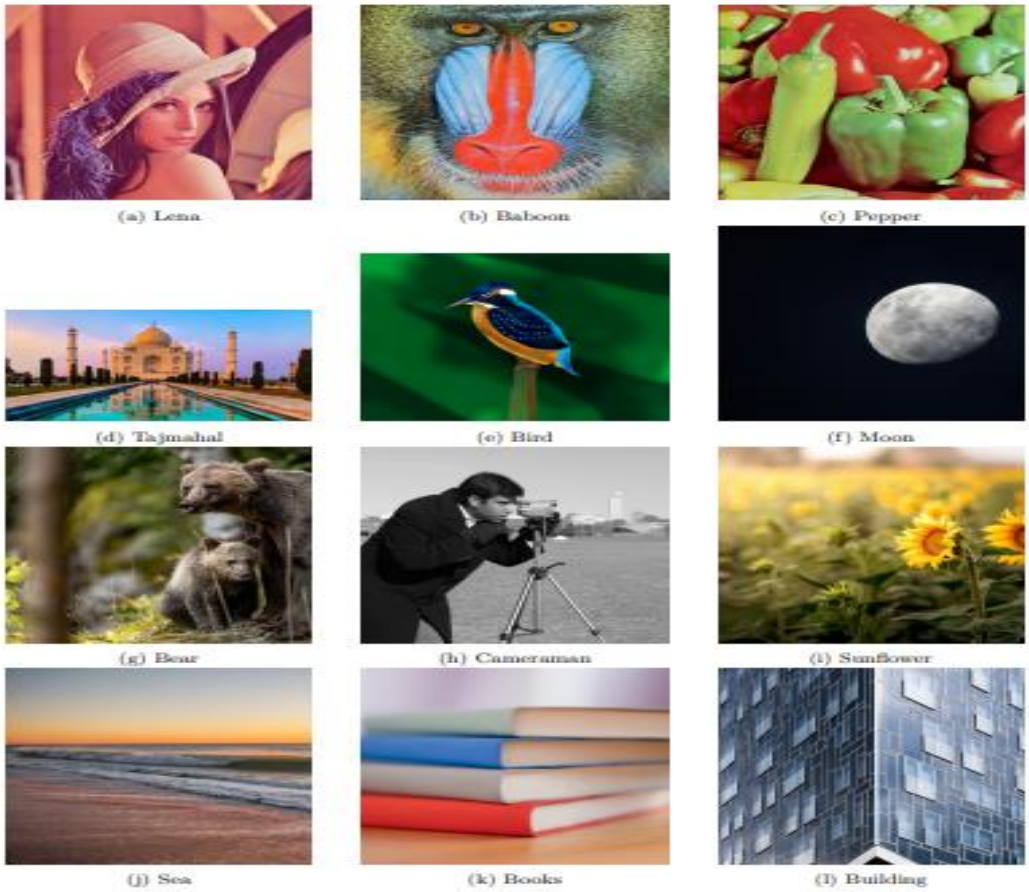

Figure10. Set of images used for testing purpose

Now we have discussed about various performance metrics such as payload, Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), Entropy, Standard Deviation and histogram analysis etc.

Capacity: Capacity is the measure of the amount of information that can be stored in the image. It is an important measure since the communication overhead depends on the maximum capacity of the image.

Following formula is used to calculate the capacity in Bits Per Pixel (BPP):

$$BPP = \frac{Total\ Number\ of\ bits\ to\ be\ embedded}{Encoding\ Capacity(Number\ of\ pixels)} \ldots\ldots\ldots\ldots (1)$$

Table 1 shows the capacity of different methods.

ImageName	Payload			
	Proposed	Mahdi	Fateh	Ali
Lena.png	0.75	1	0.5	0.33
Baboon.png	0.95	1	0.5	0.70
Pepper.png	0.93	1	0.5	0.31
Cameraman.png	0.85	1	0.5	0.65
Tajmahal.png	0.91	1	0.5	0.82
Bird.png	0.98	1	0.5	0.77
Bear.png	0.78	1	0.5	0.65
Books.png	0.75	1	0.5	0.69
Sunflower.png	0.93	1	0.5	0.84
Moon.png	0.78	1	0.5	0.89

PSNR (Peak Signal-to-Noise Ratio): PSNR measures the quality of reconstructions of a compressed or a modified image compared to the original image. It calculates the ratio between the maximum possible power of an image and the power of corrupting noise that affects the fidelity of its representation. The PSNR is calculated using the following formula:

$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right) \dots\dots\dots\dots (2)$$

Where, $MSE = (\sum_{i=1}^{N} \frac{(Original\ Pixel_i - Modified\ Pixel_i)^2}{Total\ Number\ of\ Pixels})$

Table 2 shows the PSNR of different methods.

ImageName	PSNR values			
	Proposed	Mahdi	Fateh	Ali
Lena.png	72.32	66.62	63.34	90.18
Baboon.png	72.22	67.68	62.04	61.48
Pepper.png	68.21	65.84	63.28	90.09
Cameraman.png	72.43	65.84	60.23	82.56
Tajmahal.png	75.72	67.89	55.67	65.89
Bird.png	80.40	69.45	65.12	88.67
Bear.png	78.30	66.23	45.56	65.30
Books.png	78.36	65.45	64.45	80.36
Sunflower.png	78.19	66.48	60.50	73.91
Moon.png	78.30	68.45	62.67	68.06

SSIM (Structural Similarity Index Measure): SSIM is a metric used to measure the similarity between two images. It evaluates the perceived quality of an image by comparing its structural information such as luminance, contrast, and structure against a reference image. SSIM can help detect changes in these structural elements of an image caused by the embedding of hidden information. (24)

The formula for SSIM can be expressed as follows:

$$SSIM(X,Y) = \left(\frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + 2)}\right) \dots\dots\dots (3)$$

Where, x and y are the reference and distorted images, μx and μy are the means of x and y, σ_x^2 and σ_y^2 are the variances of x and y, σ_xy is the covariance between x and y, C1 and C2 are constants.

Table 3 shows the SSIM values of different methods.

ImageName	SSIM values			
	Proposed	Mahdi	Fateh	Ali
Lena.png	0.9998	0.9992	0.9988	0.9888
Baboon.png	0.9997	0.9989	0.9996	0.9990
Pepper.png	0.9999	0.9985	0.9982	0.9889
Cameraman.png	0.9994	0.9976	0.9978	0.9965

Cosine Similarity Index: Cosine similarity is used to compare feature vectors or histograms of images to detect any significant differences caused by the presence of hidden data. Lahitani et al. (2016) Formula for measuring cosine similarity index is: $Cosine\ Similarity = \frac{A.B}{||A||||B||} \dots\dots\dots (4)$

Table 4 shows the Cosine similarity values of different methods.

ImageName	Cosine similarity values			
	Proposed	Mahdi	Fateh	Ali
Lena.png	0.9987	0.9970	0.9981	0.9998
Baboon.png	0.9985	0.9981	0.9979	0.9970
Pepper.png	0.9995	0.9923	0.9988	0.9880
Cameraman.png	0.9992	0.9990	0.9985	0.9989

Standard Deviation: Standard deviation measures the amount of variation or dispersion of pixel values in an image.

Formula for measuring standard deviation is:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2} \dots\dots\dots\dots (5)$$

Table 5 shows the Standard deviation values of different methods.

Image Name	Standard deviation values			
	Proposed	Mahdi	Fateh	Ali
Lena.png	58.98	39.80	52.47	49.80
Baboon.png	56.19	52.90	73.64	52.90
Pepper.png	65.29	52.47	46.66	62.47
Cameraman.png	73.79	63.64	53.80	53.64

5. Conclusions

The provided system demonstrates a new approach of image steganography to distribute the data uniformly across the image using the Fernet symmetric encryption and odd-even pixel modification-based steganography techniques. The system allows users to encrypt a secret message using a password and embed it within an image's red channel leads to the creation of a stego image. The stego image seems visually similar to the original cover image, and the embedded secret data remains hidden to the bare human eyes. The system successfully achieved its primary objectives, which are encryption, uniform data hiding, and image retrieval. Although, the high value of standard deviation indicates a statistical detectability of alteration, high PSNR suggests a high quality of the stego image indicating that our method has succeeded to retain a high level of image fidelity. The use of symmetric encryption with the Fernet cipher provides a simple yet effective way to secure the secret data before embedding it into the image. The added option to use a password enhances the security of the system, making it more challenging for unauthorized users to access the concealed data. The odd-even pixel modification-based steganography technique, allows for the hiding of data by modifying the image pixels based on their being odd or even. We successfully embedded the data uniformly across the image according to our proposed algorithm. We also calculated the Peak Signal-to-Noise Ratio (PSNR) to assess the image quality degradation caused by data embedding. The achieved PSNR value is a reasonable indicator of the image fidelity, showing that the steganography process resulted in minimal visible distortion to the cover image. The system provides a practical demonstration of effective techniques of steganography that can be served as a tool to understand the basic concepts of image steganography. We learned about symmetric encryption, different techniques of steganography, PSNR calculation, and the importance of secure password management. For practical use in real-world scenarios, we will work on the system's security and capabilities to be further enhanced and scrutinized in the future.

References

1. Kadhim, I.J., Premaratne, P., Vial, P.J. and Halloran, B. Comprehensive survey of image steganography: Techniques, Evaluations, and trends in future research. Neurocomputing 2019, 335, 299–326.

2. Sultana, H.; Kamal, A. An Edge Detection Based Reversible Data Hiding Scheme. In Proceedings of the 2022 IEEE Delhi Section Conference (DELCON), New Delhi, India, 11–13 February 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–6.
3. Reddy, V. L., Subramanyam, A. and Reddy, P. C. Implementation of LSB steganography and its evaluation for various file formats. Int. J. Advanced Networking and Applications, 2011, pp. 868--872.
4. Thangadurai, K and Devi, G. S. An analysis of LSB based image steganography techniques. 2014 international conference on computer communication and informatics. 2014, pp. 1-4.
5. Goel, S., Rana, A. and Kaur, M. A review of comparison techniques of image steganography. Global Journal of Computer Science and Technology, 2013, pp. 9-14.
6. Hashim, M.M., Abdulrazzaq, A.A., Rahim, M.S.M, Taha, M.S., Khalid, H.N., and lafta, S.A.S. Improvement of image steganography scheme based on LSB value with two control random parameters and multi-level encryption. IOP Conference Series: Materials Science and Engineering. s.l. : IOP Publishing, 2019, p. 052002.
7. Fateh, M., Rezvani, M. and Irani, Y. A new method of coding for steganography based on LSB matching revisited. Security and Communication Networks, 2021, pp. 1--15.
8. Ali, U., Sohrawordi, M. and Uddin, M. P. A robust and secured image steganography using LSB and random bit substitution. American Journal of Engineering Research (AJER), 2019, pp. 39--44.
9. Neeta, D., Snehal, K. and Jacobs, D. Implementation of LSB steganography and its evaluation for various bits. 2006 1st international conference on digital information management, 2006, pp. 173--178.
10. Fridrich, J., Goljan, M. and Du, R. Detecting LSB steganography in color, and gray-scale images. IEEE multimedia, 2001, pp. 22--28.
11. Fkirin, A., Attiya, G. and El-Sayed, A. Steganography literature survey, classification and comparative study. Communications on Applied Electronics, 2016, pp. 13--22.
12. Djebbar, Fatiha, A., Beghdad, M., Karim, A. and Hamam, H. Comparative study of digital audio steganography techniques. EURASIP Journal on Audio, Speech, and Music Processing, 2012, pp. 1--16.
13. Liu, Y., Liu, S., Wang, Y., Zhao, H. and Liu, S. Video steganography: A review. Neurocomputing, 2019, pp. 238--250.
14. Krishnan, Thandra,R.B., P Kumar, P. and Baba, M.S. steganography, An overview of text. 2017 Fourth International Conference on Signal Processing, Communication and Networking (ICSCN), 2017, pp. 1--6.
15. Setiadi, De Rosal Igantius Moses. PSNR vs SSIM: imperceptibility quality assessment for image steganography. Multimedia Tools and Applications, 2021, pp. 8423--8444.
16. Mukherjee, S. and Sanyal, G. Edge based image steganography with variable threshold. Multimedia Tools and Applications, 2019, pp. 16363--16388.
17. Qian, Y., Dong, J., Wang, W. and Tan, T. Deep learning for steganalysis via convolutional neural networks. Media Watermarking, Security, and Forensics, 2015, pp. 171--180.
18. Reddy, H. M., Sathisha, N., Kumari, A. and Raja, KB. Secure steganography using hybrid domain technique. 2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12), 2012, pp. 1--11.
19. Guo, L., Ni, J., Su, W., Tang, C. and Shi, Y. Using statistical image model for JPEG steganography: Uniform embedding revisited. IEEE Transactions on Information Forensics and Security, 2015, pp. 2669--2680.
20. Cogranne, R., Giboulot, Q. and Bas, P. The ALASKA Steganalysis Challenge: A First Step Towards Steganalysis. Proceedings of the ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec'2019), 2019, pp. 125--137.
21. Dang-Nguyen, D.T., Pasquini, C., Conotter, V. and Boato, G. RAISE – A Raw Images Dataset for Digital Image Forensics. Proceedings of ACM Multimedia Systems, 2015.
22. Gloe, T. and Bohme, R. The 'Dresden Image Database' for Benchmarking Digital Image Forensics. Proceedings of the 25th Symposium on Applied Computing (ACM SAC 2010). 2010, pp. 1585--1591.

23. Newman, J. and Lin, L. and Chen, W. and Reinders, S. and Wang, Y. and Wu, M. and Guan, Y. StegoAppDB: a Steganography Apps Forensics Image Database. Proceedings of Media Watermarking, Security, and Forensics (MWSF'2019), Part of IS&T International Symposium on Electronic Imaging (EI'2019). 2019.
24. Subsol, H. R., Mehdi, Y., Marc, C., Frédéric, C. and Gérard. LSSD: A Controlled Large JPEG Image Database for Deep-Learning-based Steganalysis "into the Wild". Proceedings of the 25th International Conference on Pattern Recognition (ICPR'2021), Workshop on Multi Media FOR ensics in the WILD (MMForWILD'2021). s.l.: Springer, 2021, pp. 470--483.