Enhanced Image Steganography Using Dual Authentication And Camellia Cipher Encryption

A
Major Project Report
Submitted in partial fulfilment of the requirements for the award of the Degree of
Bachelor of Technology
by

CH. Shreya
(20EG105407)

Under the Guidance
of

Dr. K. Madhuri
Associate Professor,
Department of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
ANURAG UNIVERSITY
VENTAKAPUR (V), GHATKESAR (M), MEDCHAL (D), T.S - 500088
TELANGANA
(2023-2024)

DECLARATION

I hereby declare that the report entitled **“Enhanced Image Steganography using Dual Authentication and Camellia Cipher Encryption”** submitted for the award of the degree of **Bachelor of Technology (B. Tech)** is original work done by me and this report has not formed the basis for the award of any degree, diploma, associate ship or fellowship of similar other titles. It has not been submitted to any other University or Institution for the award of any degree or diploma.

Place: Anurag University, Hyderabad CH. Shreya

Date: (20EG105407)

ANURAG UNIVERSITY | SCHOOL OF ENGINEERING

CERTIFICATE

This is to certify that the project report entitled **"Enhanced Image Steganography using Dual Authentication and Camellia Cipher Encryption"** being submitted by **Ms. CH. Shreya** bearing the Hall Ticket number **20EG105407** in partial fulfillment for the award of the degree of the B.Tech Degree in Computer Science and Engineering to Anurag University is a record of bonafide work carried out by her under my guidance and supervision.

The results embodied in this report have not been submitted to any other University for the award of any other degree or diploma.

Dr. K. Madhuri	Dr. G. Vishnu Murthy
Associate Professor	Dean CSE
Department of CSE	

External Examiner

ACKNOWLEDGMENT

I would like to express our sincere thanks and deep sense of gratitude to project supervisor **Dr. K. Madhuri, Associate Professor, Department of CSE,** Anurag University for her constant encouragement and inspiring guidance without which this project could not have been completed. Her critical reviews and constructive comments improved my grasp of the subject and steered to the fruitful completion of the work. Her patience, guidance and encouragement made this project possible.

We would like express our special thanks to **Dr. V. Vijaya Kumar, Dean School of Engineering, Anurag University,** for his encouragement and timely support in our B.Tech program.

We would like acknowledge our sincere gratitude for the support extended by **Dr. G. Vishnu Murthy, Dean, Dept. of CSE, Anurag University.** We also express my deep sense of gratitude to **Dr. V V S S S Balaram, Academic co-ordinator, Dr. Pallam Ravi,** Project in-Charge. Project Co-ordinator and Project review committee members, whose research expertise and commitment to the highest standards continuously motivated us during the crucial stage of our project work

CH. Shreya

(20EG105407)

ABSTRACT

In the realm of image data security and steganography applications, evaluating content similarity between different image datasets and addressing embedding distortions within images pose significant challenges. Existing methodologies often lack effectiveness, hindering the assessment of content comparability among sets of photos. To address these limitations, this project introduces a novel technique specifically tailored for PNG pictures. The proposed algorithm offers a comprehensive assessment of content similarity while accounting for embedding distortion. Beyond image comparison, the methodology extends to cover image selection, sender and receiver authentication mechanisms, and integration of Camellia Cipher encryption. Key components include algorithmic cover image selection, robust authentication mechanisms, seamless integration of Camellia Cipher encryption, and comprehensive image steganography techniques. The project aims to provide a holistic solution that enhances security and performance in image steganography applications, contributing to advancements in the field of Image data security.

Keywords: Image Content Similarity, Embedding Distortion, Structural Similarity Index (SSIM), Entropy, Fractal Dimension, Edge Detection, Text Compression, LZW Algorithm, Least Significant Bit (LSB) Steganography, Camellia Cipher, Base – 64 Encoding, Portable Network Graphics (PNG) Images.

INDEX

List of Figures

List of Tables

List of Abbreviations

Abbreviations	**Full Form**
LSB	Least Significant Bit
MSE	Mean Squared Error
PSNR	Peak Signal to Noise Ratio
AES	Advanced Encryption Standard
PNG	Portable Network Graphics
RGB	Red, Green, Blue
CMYK	Cyan, Magenta, Yellow, Key (or Black)
RGBA	Red, Green, Blue, Alpha
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
XOR	Exclusive OR
IV	Initialization Vector
SHA	Secure Hash Algorithm
CBC	Cipher Block Chaining
ASCII	American Standard Code for Information Interchange
IDLE	Integrated Development and Learning Environment
SSIM	Structural Similarity Index Measure
CFB	Cipher Feedback
JPEG	Joint Photographic Experts Group
LZW	Lempel-Ziv-Welch

1. INTRODUCTION

Steganography is a technology to send secret data on public channels without drawing suspicion by slightly modifying the media content [1]. As the adversary, steganalysis aims to discover the possible covert communication by analyzing the public media [2]. The effectiveness of steganography lies in its ability to conceal the presence of hidden information within cover objects, such as images. Various techniques have been developed to enhance this concealment process, including dynamic steganography. Aside from dynamic steganography, several other methods aim to conceal the presence of steganographic activity. These may include techniques such as covert channel communication, which involves hiding information within seemingly innocuous communication channels, or adaptive steganography, where the embedding process adapts dynamically to its environment to avoid detection. Additionally, methods like data fragmentation and spread spectrum techniques can also be employed to scatter hidden information across multiple data units, making it more challenging for interceptors to detect.

One particularly significant method discussed in this paper is Cover Image Selection. Cover Image Selection plays a crucial role in steganography as it can deceive potential interceptors into believing that no steganographic activity has taken place, when in fact it has. This is achieved by carefully choosing cover images that blend seamlessly with the hidden information, making it difficult for covert listeners to detect any alterations. By emphasizing the importance of Cover Image Selection, this paper underscores its role in enhancing the covert nature of steganographic communication. It highlights how strategic selection of cover images can effectively mislead interceptors, ultimately safeguarding the secrecy and integrity of the hidden information.

This research paper explores the integration of PNG Image Compression, Diffie-Hellman Authentication, Camellia cipher, LZB Compression, Header Text, Initialization Vector

(IV), Base64 Encoding, and LSB Steganography. These elements will be applied across different phases of the project.

1.1. Motivation

The project "Enhanced Image Steganography with Dual Authentication and Camellia Cipher Encryption" is driven by a multifaceted motivation rooted in the complex landscape of modern digital security. At its core, the project seeks to address the pressing need for robust encryption and authentication mechanisms amidst escalating concerns over data privacy and security breaches. With the pervasive digitization of sensitive information across various domains, including finance, healthcare, and communications, there's an urgent demand for innovative solutions capable of safeguarding data integrity and confidentiality. In exploring the realm of image steganography, the project aims to harness the concealment capabilities of digital images to securely transmit hidden messages or information. By embedding secret data within innocuous-looking images, steganographic techniques offer a clandestine means of communication, rendering the hidden content imperceptible to unintended recipients. This covert communication paradigm not only underscores the importance of encryption but also underscores the need for robust authentication mechanisms to verify the identities of both senders and receivers.

Dual authentication, a central component of the project, emerges as a pivotal strategy for fortifying the integrity of communication channels and thwarting unauthorized access. By employing techniques such as Diffie-Hellman key exchange, the project endeavors to establish secure communication channels wherein both parties can mutually authenticate each other's identities without relying on centralized authorities. This decentralized approach to authentication not only enhances security but also fosters trust and confidence in the integrity of digital communications. Moreover, the adoption of Camellia Cipher Encryption, a sophisticated encryption algorithm known for its resilience against cryptanalytic attacks, further bolsters the project's security posture. In contrast to conventional encryption standards like AES-256, Camellia Cipher offers enhanced resistance to advanced cryptographic attacks while maintaining efficient performance across diverse computing platforms. This heightened level of security is imperative in mitigating emerging threats posed by malicious actors and ensuring the confidentiality of sensitive information. Beyond its immediate security implications, the project also serves as a catalyst for research and innovation in the field of cybersecurity. By delving into the intricacies of cryptographic protocols, image processing techniques, and software development methodologies, participants gain invaluable insights into the underlying principles that govern digital security. This experiential learning journey not only enriches their understanding of complex cryptographic concepts but also equips them with practical skills essential for addressing real-world cybersecurity challenges. Furthermore, the project's educational significance extends beyond technical proficiency, fostering critical thinking, problem-solving, and collaboration among participants. Through collaborative teamwork and interdisciplinary exchange, individuals engage in iterative design processes, refine their solutions, and adapt to evolving security paradigms. This collaborative ethos not only cultivates a culture of innovation but also empowers participants to become proactive agents of change in the realm of cybersecurity. Moreover, the societal impact of the project cannot be understated, as it underscores the pivotal role of cybersecurity in safeguarding democratic principles, human rights, and societal well-being. By fortifying digital infrastructure against cyber threats, the project contributes to the resilience of critical systems, infrastructure, and institutions essential for sustaining modern societies. This societal resilience, in turn, fosters trust, stability, and resilience in the face of emerging threats and challenges. In essence, the project "Enhanced Image Steganography with Dual Authentication and Camellia Cipher Encryption" embodies a convergence of technological innovation, security imperative, educational enrichment, and societal resilience. By addressing the multifaceted challenges of modern cybersecurity, the project exemplifies the transformative potential of interdisciplinary collaboration, experiential learning, and technological innovation in shaping a secure and resilient digital future.

1.2. Problem Definition

This project addresses a key challenge in steganography, which is the lack of a universally effective method for assessing content similarity across diverse image collections. Previous studies often lacked adaptability to diverse content types, resulting in suboptimal performance across different scenarios. Those studies included using KL Divergence for embedding distortion and Singular Value Decomposition for the Image Similarity which resulted to be flaw in the case of Changes in Image Contrast, Brightness and Flipping. So, this project introduces an adaptive algorithm designed to work with various content types, with a specific focus on PNG images and take the Text and Image Compression into consideration with special focus on Image Parameters like Fractal dimension, Edge detection, SSIM calculation, and entropy to comprehensively evaluate content similarity, embedding distortion, and cover image selection.

By reducing the similarity between images within a batch, the method ensures a diverse range of coverings, enhancing overall security.

1.3 Problem Illustration

In the current approach they use embedding distortion and image similarity to select the cover image. The method used for image similarity is Singular Value Decomposition (SVD) and for embedding distortion the method used is KL Divergence. Another challenge observed is using JPG images for this process steganography. Here are the challenges that have been observed, and there is a justification explaining which components of each issue have been addressed.

1. Singular Value Decomposition for Image Similarity Calculation
2. KL Divergence for Embedding Distortion of the Image.
3. Consideration of JPEG Images for the experiment

Now, explaining each issue with an example and showing the results produced by using the above approach:

1. **Singular Value Decomposition for Image Similarity Calculation**

   Image similarity evaluation based on difference in **CONTRAST**.

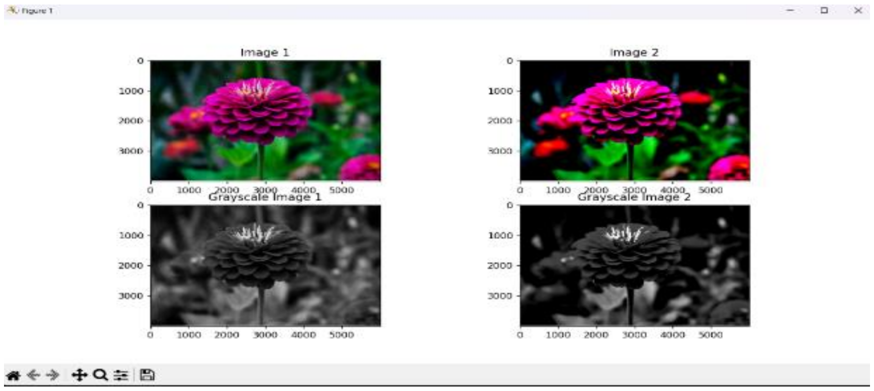
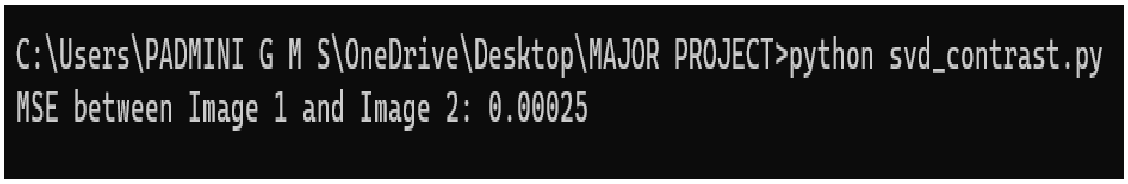


Figure 1.3.1 SVD For Difference in Contrast



```
C:\Users\PADMINI G M S\OneDrive\Desktop\MAJOR PROJECT>python svd_contrast.py
MSE between Image 1 and Image 2: 0.00025
```

Figure 1.3.2 MSE Result for Image Similarity

Image similarity evaluation based on difference in **BRIGHTNESS**

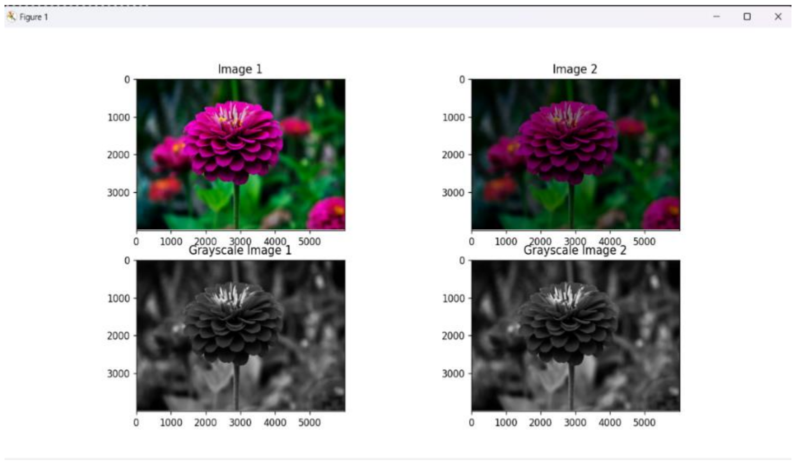
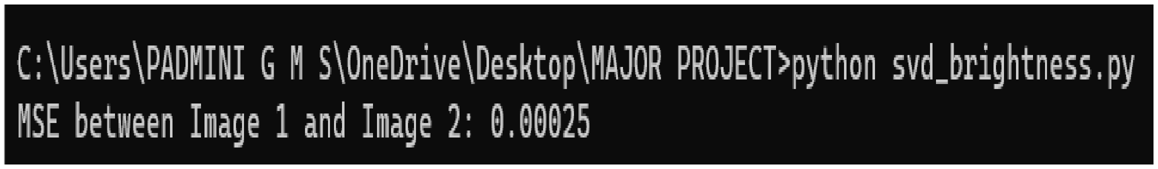


Figure 1.3.3 SVD For Difference in Brightness



```
C:\Users\PADMINI G M S\OneDrive\Desktop\MAJOR PROJECT>python svd_brightness.py
MSE between Image 1 and Image 2: 0.00025
```

Figure 1.3.4 MSE Result for Image Similarity

Image similarity evaluation based on difference in **IMAGE MIRRORING**

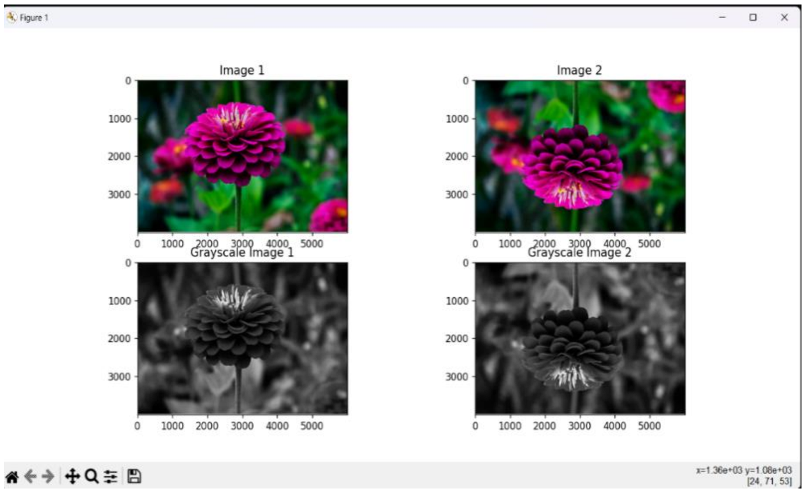
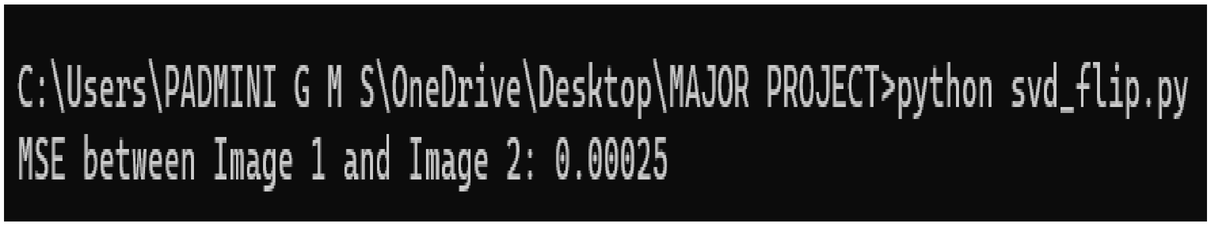


Figure 1.3.5 SVD For Difference in Mirroring of Image



```
C:\Users\PADMINI G M S\OneDrive\Desktop\MAJOR PROJECT>python svd_flip.py
MSE between Image 1 and Image 2: 0.00025
```

Figure 1.3.6 MSE Result for Image Similarity

2. KL Divergence for Embedding Distortion of the Image

Figure 1.3.7 Image 1 for KL Divergence

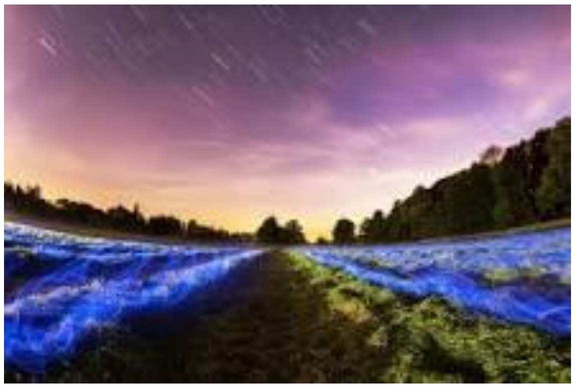


Figure 1.3.8 Image 2 for KL Divergence

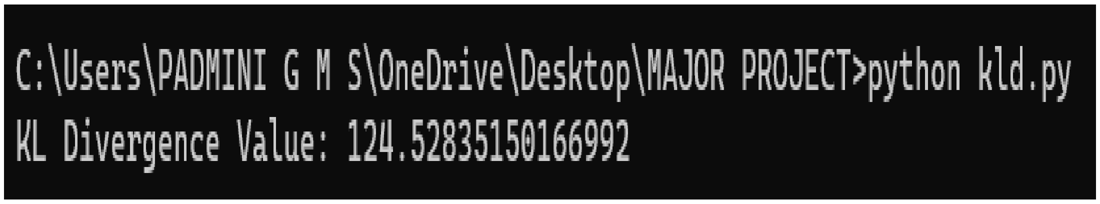

```
C:\Users\PADMINI G M S\OneDrive\Desktop\MAJOR PROJECT>python kld.py
KL Divergence Value: 124.52835150166992
```

Figure 1.3.9 KL Divergence Value

KL Divergence is the measure of difference between two probability distributions which does not takes account of the overall texture of the image. This metric is not crucial for understanding distortions that involve changes in image structure, geometry, or content.

3. Consideration of JPEG Images for the experiment

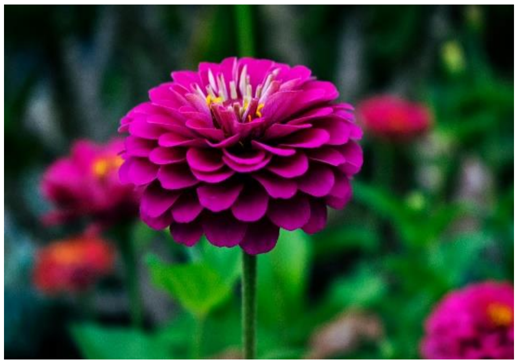
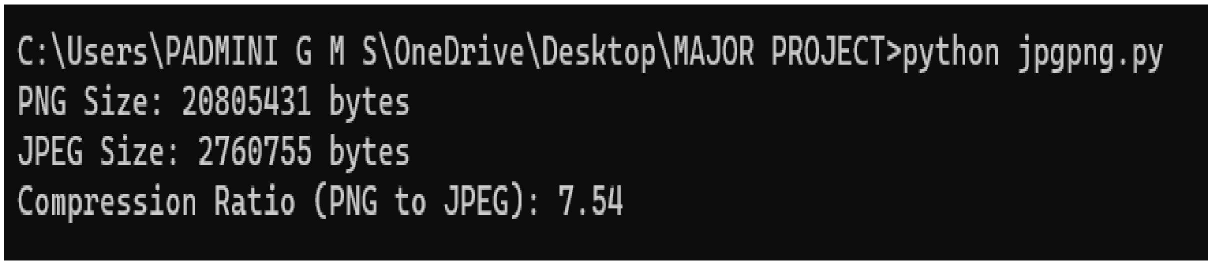


Figure 1.3.10 JPEG Image



```
C:\Users\PADMINI G M S\OneDrive\Desktop\MAJOR PROJECT>python jpgpng.py
PNG Size: 20805431 bytes
JPEG Size: 2760755 bytes
Compression Ratio (PNG to JPEG): 7.54
```

Figure 1.3.11 Comparision between JPEG & PNG

1.4. Objective of the Project

The primary objective of the project "Enhanced Image Steganography with Dual Authentication and Camellia Cipher Encryption" is to develop a robust and secure method for concealing sensitive information within digital images while ensuring authentication of both sender and receiver identities. [1] This involves implementing advanced cryptographic techniques such as Diffie-Hellman key exchange for mutual authentication and Camellia Cipher Encryption for secure data transmission. Additionally, the project aims to explore the practical application of image steganography as a covert communication channel and to enhance understanding of cybersecurity principles among participants through hands-on experience and research. Ultimately, the project seeks to contribute to the advancement of cybersecurity practices and technologies while addressing real-world challenges related to data privacy, confidentiality, and secure communication.

Furthermore, the project aims to address the pressing need for enhanced security measures in digital communication, especially in contexts where privacy and confidentiality are paramount.[2]]By leveraging the capabilities of dual authentication and Camellia Cipher Encryption, the project seeks to mitigate the risks associated with unauthorized access, interception, and tampering of sensitive information during transmission. Through rigorous testing and validation procedures, the project endeavors to demonstrate the effectiveness and reliability of the proposed system in real-world scenarios, thereby instilling trust and confidence among users in the security of their communications. Additionally, by fostering collaboration between multidisciplinary teams of researchers, engineers, and cybersecurity experts, the project aims to foster innovation and knowledge exchange in the field of cybersecurity, ultimately contributing to the development of more resilient and secure communication infrastructures.

Moreover, the project aspires to contribute to the broader landscape of cybersecurity by advancing the understanding and adoption of cutting-edge encryption techniques and authentication mechanisms. By exploring the intricacies of dual authentication and Camellia Cipher Encryption, the project not only seeks to enhance the security posture of digital communication systems but also aims to inspire further research and development in the field. Through proactive engagement with industry stakeholders and the dissemination of research findings through academic publications and conferences, the project endeavors to catalyze innovation and drive the adoption of best practices in cybersecurity. Ultimately, by empowering individuals, organizations, and governments with robust tools and methodologies to protect their sensitive information, the project aims to foster a safer and more secure digital environment for all stakeholders involved.

1.5 Introduction to the Topics

1.5.1 Image Steganography

Figure 1.5.1.1 Image Steganography

Image steganography, a pivotal technique employed in this project, plays a crucial role in concealing sensitive data within digital images to ensure covert communication. Through the seamless integration of hidden information into seemingly innocuous images, steganography enables the transmission of confidential data without arousing suspicion. In this context, image steganography serves as a robust mechanism to safeguard the confidentiality and integrity of sensitive information, offering a layer of security that complements encryption techniques. By embedding data within the pixel values or metadata of images, the steganographic process remains imperceptible to unauthorized observers, thereby thwarting attempts at interception or detection. Leveraging the power of steganography in conjunction with advanced encryption algorithms, the project endeavors to enhance the privacy and security of digital communication channels, safeguarding sensitive information from prying eyes and unauthorized access. Through meticulous implementation and rigorous testing, the project aims to demonstrate the effectiveness and reliability of image steganography as a covert communication mechanism, paving the way for its widespread adoption in various domains where confidentiality is paramount.

1.5.2 Cryptography

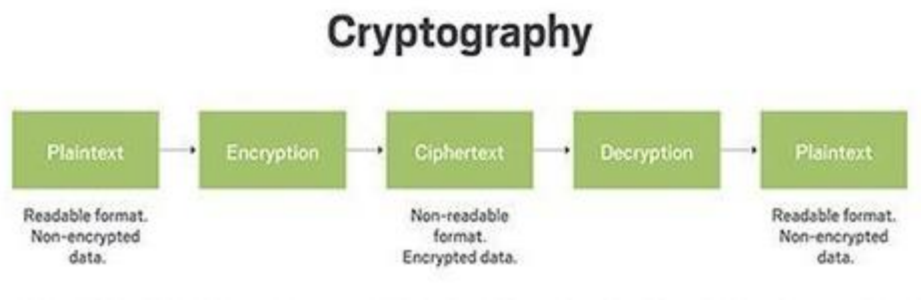

Cryptography
Plaintext
Encryption
Ciphertext
Decryption
Plaintext
Readable format. Non-encrypted data.
Non-readable format. Encrypted data.
Readable format. Non-encrypted data.

Figure 1.5.2.1 Basic Process of Cryptography

Cryptography, the art and science of secure communication, stands as a cornerstone in the realm of information security, enabling the protection of sensitive data from unauthorized access or tampering. By leveraging complex mathematical algorithms and techniques, cryptography facilitates the encryption of plaintext into ciphertext, rendering it indecipherable to anyone without the corresponding decryption key. This transformation ensures confidentiality, integrity, and authenticity in digital communication channels, safeguarding sensitive information from interception, manipulation, or disclosure. From ancient methods such as Caesar ciphers to modern cryptographic standards like AES and RSA, cryptography continues to evolve, adapting to the ever-changing landscape of cybersecurity threats and challenges. As an essential tool in the arsenal of cybersecurity professionals, cryptography plays a vital role in ensuring the privacy and security of data in an increasingly interconnected and digitized world.

Furthermore, cryptography serves as a fundamental building block for various security protocols and mechanisms employed across diverse domains, including e-commerce, online banking, secure messaging, and authentication systems. Its widespread adoption and integration into digital infrastructures enable the establishment of secure communication channels, authentication of users and entities, and protection against data breaches and cyberattacks. By upholding the principles of confidentiality, integrity, and authenticity, cryptography fosters trust and confidence in digital transactions and interactions, empowering individuals, organizations, and societies to embrace the benefits of digitalization while mitigating associated risks and vulnerabilities

In a project, cryptography and steganography often work hand in hand to ensure secure communication and data concealment. Cryptography, the practice of encrypting and decrypting data, is employed to encode sensitive information using mathematical algorithms, thereby rendering it unintelligible to unauthorized parties. This encryption process ensures confidentiality and integrity during data transmission or storage. On the other hand, steganography involves concealing the existence of data within seemingly innocuous cover objects, such as images or audio files, without arousing suspicion. By embedding encrypted data within these cover objects, steganography adds an additional layer of security by hiding the presence of sensitive information. Together, cryptography and steganography complement each other to protect data from interception, tampering, or unauthorized access, thus ensuring the confidentiality and integrity of digital communication channels.

2. LITERATURE REVIEW

[1] **Zichi Wang, 2023 et.al:** This paper, presents a pioneering methodology for cover selection in steganography, integrating considerations of image similarity and embedding distortion to bolster security measures. Diverging from conventional practices that prioritize embedding distortion alone, this approach incorporates a customized calculation of image similarity using singular value decomposition (SVD), with a focus on small singular values that align with steganographic properties. By amalgamating image similarity and embedding distortion, the proposed strategy optimizes the utilization of batch images while fortifying resilience against steganalysis. Experimental findings showcase the superior efficacy of this method compared to existing cover selection techniques, as confirmed through evaluation with contemporary steganalytic tools.

[2] **X.Liao, 2022 et.al:** In light of the increasing adoption of cloud storage for storing vast collections of digital images, this paper explores the implications for steganography, which gains a new avenue for embedding secret information across multiple images. Instead of conventional single image steganography, this approach allows for the adaptive distribution of a set of secret information across multiple images shared via cloud storage. However, an unresolved challenge lies in determining how to allocate embedding payload efficiently among these images to enhance security performance. This paper formulates adaptive payload distribution in the context of multiple image steganography, leveraging image texture features, and conducts theoretical security analyses from the perspective of steganalysts. Two payload distribution strategies based on image texture complexity and distortion distribution are proposed and discussed. These strategies can be integrated with existing state-of-the-art single image steganographic algorithms. Comparative evaluations against modern universal pooled steganalysis and single image steganalyzer methods demonstrate the security performance of the proposed payload distribution strategies. Extensive experimental results underscore the effectiveness of these strategies in achieving improved security performance.

[3]**T.Kalaichelvi,2020 et. al**: Abstract—Steganography is a data hiding technique, which is generally used to hide the data within a file to avoid detection. It is used in the police department, detective investigation, and medical fields as well as in many more fields.

Various techniques have been proposed over the years for Image Steganography and also attackers or hackers have developed many decoding tools to break these techniques to retrieve data. In this paper, CAPTCHA codes are used to ensure that the receiver is the intended receiver and not any machine. Here a randomized CAPTCHA code is created to provide additional security to communicate with the authenticated user and used Image Steganography to achieve confidentiality. For achieving secret and reliable communication, encryption and decryption mechanism is performed; hence a machine cannot decode it using any predefined algorithm. Once a secure connection has been established with the intended receiver, the original message is transmitted using the LSB algorithm, which uses the RGB color spectrum to hide the image data ensuring additional encryption.

[4] **V. Sachnev, 2009 et.al:** This paper presents an efficient JPEG steganography method based on heuristic optimization and BCH syndrome coding. The proposed heuristic optimization technique significantly decreases total distortion by inserting and removing AC coefficients 1 or -1 in the most appropriate positions. The implemented data hiding technique is based on structured BCH syndrome coding. This method achieves multiple solutions for hiding data to a block of coefficients. The proposed data hiding method chooses the best solution with minimum distortion effect. As a result, the total distortion can be significantly reduced, which results in less detectability of the steganalysis. The experiments include the steganalysis of the proposed data hiding methods. The experiment results shows that the proposed heuristic optimization significantly decreases detectability of the steganalysis. The proposed methods also outperform the existing steganography methods.

[5] **Sabyasachi Pramanik, 2019 et.al:** The paper delves into the integration of CAPTCHA codes within image steganography as a means of bolstering data security during internet transmissions. CAPTCHA, serving as a human verification mechanism, is embedded into cover images using ASCII encoding and encryption. This process results in stego images where the original CAPTCHA content is obscured, enhancing the confidentiality of transmitted data. By leveraging image steganography alongside CAPTCHA, the paper proposes a robust approach to safeguarding against unauthorized access and ensuring

secure internet communications, thereby mitigating potential risks associated with spam and unauthorized data interception.

[6] **Alaa A. Jabbar Altaay, 2012 et.al:** This paper provides an overview of image steganography, a security technique aimed at concealing messages between sender and recipient. Steganography has been employed across various file types, including digital images, audio, and video. For audio steganography, key parameters include imperceptibility, payload capacity, and robustness. Different applications may necessitate varying steganography techniques to meet specific requirements. The paper aims to explore the uses and techniques of image steganography, shedding light on its significance in ensuring covert communication in modern digital environments.

[7] **T. Qiao, 2012 et.al:** This paper addresses the gap in steganalysis by proposing an unsupervised adaptive detector for detecting JPEG steganography without the need for a training stage. The approach is based on hypothesis testing theory and statistical modeling of quantized Discrete Cosine Transform (DCT) coefficients. Through analysis of the detector's performance and the selection of statistical models, parameters estimation, and payload prediction, the methodology is refined to enhance detection reliability. The proposed adaptive statistical model-based detectors, utilizing weighted DCT channels, exhibit effectiveness in detecting JPEG steganography, as demonstrated through extensive experiments. Results indicate superior performance compared to non-adaptive detectors, achieving high Area Under Curve (AUC) values even for images with small payloads.

[8] **C. Lalengmawia, 2016 et.al:** This paper introduces an innovative image steganographic technique capable of embedding large amounts of information, such as audio and video files, into images. Leveraging the Advanced Encryption Standard (AES) algorithm, the technique ensures enhanced security by randomly selecting pixel positions for embedding and dynamically determining the size of Least Significant Bits (LSBs). Experimental evaluations involve embedding audio and video files into both grayscale and color images, comparing the proposed method with four state-of-the-art techniques. The results, analyzed qualitatively and quantitatively, demonstrate the superior effectiveness of the proposed method in concealing large information payloads within images while maintaining robust security measures.

[9] **Zoran Cica, 2016 et.al:** This paper addresses the critical importance of security in modern communication networks and advocates for the use of symmetric key algorithms to ensure fast and secure communication. Specifically, the paper proposes and assesses a pipelined implementation of the Camellia encryption algorithm, which has been endorsed by ISO/IEC for secure communication. The Camellia algorithm offers a level of security comparable to AES while exhibiting similar hardware complexity. Through evaluation, the paper highlights the efficacy of Camellia in providing robust encryption for high-speed communication networks.

[10] **Adit Pabbi, 2021 et.al:** This paper introduces a method combining steganography and encryption to enhance the security of message transmission. Steganography is utilized to conceal plaintext messages within images, providing a covert means of communication. The proposed approach incorporates AES encryption for added confidentiality, ensuring that only authorized recipients can access and decrypt the hidden message. By employing the Least Significant Bit (LSB) method of steganography alongside AES encryption, the method achieves both security and confidentiality in message transmission.

[11] **Utsav Sheth, 2016 et.al:** Presented a method of image steganography at the International Conference on Communication and Signal Processing in 2016. Their approach utilizes AES encryption and the Least Significant Nibble technique for embedding data into images covertly. By leveraging AES encryption, the method ensures robust security for the hidden data. The Least Significant Nibble technique enables efficient data embedding without significantly altering the visual quality of the image. This innovative method offers a promising avenue for secure and clandestine communication through digital images.

[12] **Krishnaveni, 2018 et.al:** Proposed a novel approach for image steganography in their publication in the Indonesian Journal of Electrical Engineering and Computer Science in 2018. Their method incorporates chaos theory to enhance the security and robustness of the steganographic process. By leveraging chaotic systems, the method achieves greater resistance against detection and decryption attempts. This innovative approach opens new possibilities for enhancing the covert communication capabilities of steganographic techniques. The integration of chaos theory enriches the field of steganography by offering advanced security features against modern cryptographic attacks.

[13] **Fridrich, 2012 et.al:** introduced rich models for steganalysis of digital images in their publication in the IEEE Transactions on Information Forensics and Security in 2012. Their work explores sophisticated models for detecting hidden information within digital images. By leveraging advanced statistical techniques, the proposed models enhance the accuracy and reliability of steganalysis methods. This research contributes to the ongoing efforts to develop more effective tools for detecting covert communication channels. The rich models introduced by Fridrich and Kodovsky represent a significant advancement in the field of steganalysis, enabling better detection of hidden data in digital images.

[14] **Kumar, 2010 et.al:** provided an exploration of steganography as a data hiding technique in their publication in the International Journal of Computer Applications in 2010. Their work delves into the principles and applications of steganography for concealing information within digital media. By analyzing various steganographic techniques, the authors highlight the versatility and potential of this covert communication method. This research sheds light on the importance of steganography in ensuring secure communication channels. Kumar and Pooja's study serves as a valuable resource for understanding the fundamentals of steganography and its practical implications.

[15] **Bhargava, 2019 et.al:** Presented a method of hiding image and text using LSB, DWT, and RSA in their publication in the ICTACT Journal on Image & Video Processing in 2019. Their approach combines multiple cryptographic techniques to achieve enhanced security and confidentiality in data transmission. By leveraging LSB embedding, DWT transformation, and RSA encryption, the method ensures robust protection against unauthorized access and interception. This innovative approach demonstrates the potential of integrating multiple encryption and embedding techniques for secure communication. Bhargava and Mukhija's method offers a comprehensive solution for concealing sensitive information within digital media, addressing the growing need for secure data transmission channels.

[16] **Gutub, 2009 et.al:** proposed the Triple-A method for secure RGB image steganography at the 2009 IEEE/ACS International Conference on Computer Systems and Applications. Their approach relies on randomization to enhance the security of data embedding within images. By incorporating randomization techniques, the Triple-A method strengthens the resilience of steganographic communication against adversarial

attacks. This innovative method offers a robust solution for concealing sensitive information within digital images, contributing to the advancement of secure communication protocols.

[17] **Srilakshmi, 2018 et.al:** Introduced a text embedding technique using image steganography in the spatial domain in the International Journal of Engineering & Technology in 2018. Their method utilizes spatial domain techniques to conceal textual information within images. By embedding text directly into the spatial domain of images, the proposed technique offers a covert communication channel with enhanced security features. This innovative approach provides a practical solution for securely transmitting textual data within digital images, addressing the need for covert communication methods in modern digital environments.

[18] **Rana, 2012 et.al:** Provided an introduction to steganography in their publication in the International Journal of Engineering and Computer Science in 2012. Their work offers an overview of the fundamental principles and applications of steganography across various digital media formats. By exploring different steganographic techniques, the authors highlight the versatility and significance of covert communication methods. This research serves as a comprehensive introduction to the field of steganography, offering valuable insights into its applications and potential for secure communication.

[19] **Mathkour, 2008 et.al:** Proposed a new image steganography technique at the 2008 4th International Conference on Wireless Communications, Networking, and Mobile Computing. Their method introduces an innovative approach to concealing information within digital images. By leveraging advanced encoding techniques, the proposed method enhances the security and robustness of the steganographic process. This research contributes to the ongoing efforts to develop more effective and secure methods of covert communication through digital media.

[20] **Provos, 2003 et.al:** Introduced the concept of steganography and its implications in their publication in IEEE Security & Privacy in 2003. Their work provides a comprehensive overview of steganography as a means of concealing information within digital media. By exploring various techniques and applications, the authors highlight the importance of steganography in secure communication.

Sl.No	**Author**	**Strategies**	**Advantages**	**Disadvantages**
01	Zichi Wang et.al.	Cover Selection based on Image Similarity and Embedding Distortion	1.Effective Dimensionality Reduction 2. Simplifying Complex Metrics	1. Computational demands for the memory requirements 2. Limits the scalability.
02	Xinpeng Zhang et.al.	Cover Selection Method using Gaussian Mixture	1. Enhancing Suitability for Diverse Cover Images 2. Statistical Robustness	1.Sensitivity to the quality of the input images 2. Computational Complexity
03	Sumari Putra et.al.	Cover Image Selection based on Frequency based analysis	1.Resistance to Spatial Domain changes 2.Higher Embedding Data capacity	1.Exhibits artifacts 2.Not universally applicable for all images
04	Anqi Qiu et.al.	Cover Image Selection based on Feature Selection	1.Contextually relevant 2. Higher degree of unpredictability	1.Continuous adaptation 2. Visual Alteration of the images.
05	Rafal Najeh Kadhuma et. al.	Steganography with implicit authentication to enhance sensitive data hiding	1. Providing an additional layer of security without explicitly revealing the authentication process. 2. Protection against tampering	1.Might still attempt to reverse engineer 2.Manipulate the hidden authentication features.

Table 2.1 Comparison of Existing Methods

3. PROPOSED METHOD

The proposed methodology emphasizes the integration of image similarity and embedding distortion considerations in cover selection for steganography. Unlike conventional practices focusing solely on embedding distortion, this approach incorporates a customized calculation of image similarity using singular value decomposition (SVD), particularly emphasizing small singular values conducive to steganographic properties. By amalgamating image similarity and embedding distortion, the strategy optimizes batch image utilization while fortifying resilience against steganalysis. Experimental findings demonstrate the superior efficacy of this method over existing cover selection techniques, as confirmed through evaluation with contemporary steganalytic tools.
In this research paper we discuss where first 8 Uncompressed PNG Images are taken and among those 1 Image is selected as Best Image based on Cover Image Selection Parameters which are Image Similarity where and the same is done to the 8 Compressed PNG Images and then finally among these 2 Cover Images we select the final Cover Image that is suitable for the steganography and then we do the Text Compression using LZW Algorithm and then using we encrypt the compressed text with Camellia Cipher and then encode in the selected Cover Image using LSB Steganography.

3.1. PNG Image Compression

We aim to assess the functionality of our project by subjecting it to tests involving both compressed and uncompressed PNG images. Our testing methodology involves compressing PNG images based on parameters specified by the user, leveraging the flexibility provided by the `**format_size**` option. This option empowers users to finely tune the compression process according to their preferences, allowing them to specify the desired level of compression for the images. Through this approach, we can thoroughly evaluate the effectiveness and performance of our project under various compression settings, ensuring its robustness and reliability across different usage scenarios.
We aim to assess the functionality of our project by subjecting it to tests involving both compressed and uncompressed PNG images. Our testing methodology involves

compressing PNG images based on parameters specified by the user, leveraging the flexibility provided by the `**format_size**` option. This option empowers users to finely tune the compression process according to their preferences, allowing them to specify the desired level of compression for the images. Through this approach, we can thoroughly evaluate the effectiveness and performance of our project under various compression settings, ensuring its robustness and reliability across different usage scenarios.

Figure 3.1.1 PNG Image

To compress PNG images, we employ a process that involves reducing the file size while striving to maintain image quality and integrity. This is typically achieved through techniques such as lossless compression, which focuses on removing redundancy and optimizing the encoding of pixel data without discarding any visual information. One common approach is to utilize algorithms like DEFLATE, which is widely used in PNG compression. During compression, the algorithm identifies repetitive patterns or sequences of data within the image and replaces them with shorter codes or references to a dictionary. This effectively reduces the amount of data needed to represent the image, resulting in a smaller file size. Additionally, the compression process may involve other optimizations such as color space reduction, filtering, and metadata removal to further reduce the size of the PNG file.

It's important to note that while compression reduces file size, it should ideally be done in a manner that minimizes loss of image quality. Lossless compression techniques ensure that the original image can be perfectly reconstructed from the compressed data, making them suitable for applications where preserving image fidelity is paramount, such as medical imaging, archival storage, and digital art preservation.

Overall, compressing PNG images involves a careful balance between reducing file size and maintaining image quality, achieved through the application of lossless compression algorithms and optimization techniques tailored to the characteristics of PNG image data.

3.2 Cover Image Selection

This project tackles a significant challenge in steganography, which is the absence of a universally effective method for evaluating content similarity across various image collections. Previous research often lacked adaptability to diverse content types, leading to suboptimal performance in different scenarios. Prior studies utilized KL Divergence for embedding distortion and Singular Value Decomposition for Image Similarity, which proved flawed in cases of alterations in image contrast, brightness, and flipping. Consequently, this project introduces an adaptive algorithm tailored to diverse content types, particularly PNG images, considering text and image compression while focusing on specific image parameters like fractal dimension, edge detection, SSIM calculation, and entropy. This approach comprehensively assesses content similarity, embedding distortion, and cover image selection. By minimizing image similarity within a batch, the method ensures a wide range of coverings, thereby enhancing overall security. The project's primary objective is to bolster the security, resilience, and efficiency of the steganographic process. To safeguard data privacy, the project employs dual authentication processes and utilizes the Camellia algorithm for advanced encryption. This comprehensive strategy not only addresses limitations in traditional techniques but also establishes a robust foundation for secure communication.

To further enhance the effectiveness of steganographic techniques, this project introduces an adaptive algorithm designed to address the limitations of existing methods in evaluating content similarity across diverse image collections. By considering various factors such as image contrast, brightness, and flipping, the algorithm offers a more comprehensive assessment of content similarity and embedding distortion, particularly tailored to PNG images. Leveraging advanced image processing techniques such as fractal dimension analysis, edge detection, SSIM calculation, and entropy evaluation, the algorithm provides a refined approach to cover image selection, thereby improving overall security. With a focus on minimizing image similarity within a batch, the project aims to expand the range of available coverings, enhancing the resilience and efficiency of steganographic processes. Through these advancements, the project contributes to the ongoing evolution of secure communication methodologies, offering a robust solution for safeguarding data privacy in diverse contexts.

3.3. Diffie Hellman Authentication

The Diffie-Hellman key exchange protocol enables two parties, typically referred to as the sender and receiver, to establish a shared secret key over an insecure communication channel. The process begins with each party independently selecting a private key and then exchanging public keys derived from these private keys. Despite the public exchange of keys, an eavesdropper cannot easily derive the shared secret key because doing so requires solving a computationally difficult problem known as the discrete logarithm problem. Once both parties possess each other's public keys, they can combine them with their own private keys to compute the same shared secret key. This shared key can then be used to encrypt and decrypt messages securely, enabling confidential communication between the sender and receiver without the need to transmit the key itself over the insecure channel.

The process begins with both parties agreeing upon common public parameters, including a large prime number 'p' and a primitive root modulo 'g'. Each entity generates a private key ('sender_private_key' for the sender and 'receiver_private_key' for the receiver) through random selection. Using their respective private keys and the agreed-upon public parameters, they compute their partial public values ('x' for the sender and 'y' for the receiver). These partial public values are then securely exchanged between the parties. Upon receiving the other party's partial public value, each entity computes the shared secret key by combining its private key with the received partial public value. If the computed shared secret keys match on both ends, authentication is successful, permitting secure communication to proceed. However, if the shared secret keys do not match, it indicates a potential security breach or unauthorized access, leading to authentication failure.

The Diffie-Hellman key exchange protocol facilitates secure communication between two parties over an insecure channel by allowing them to jointly establish a shared secret key. The steps involved in the Diffie-Hellman key exchange are as follows:

1. Initialization: Both the sender and receiver agree on publicly available parameters: a large prime number (p) and a primitive root modulo (p), denoted as (g). These parameters are typically agreed upon in advance or may be part of a predefined standard.

2. Key Generation: Each party generates their own private key. Let's denote the private key of the sender as (a) and the private key of the receiver as (b). These private keys are kept secret and not shared with anyone else.

3. Public Key Calculation: Using the agreed-upon parameters and their respective private keys, each party calculates their public key. The sender computes (A = g^a mod p) and the receiver computes (B = g^b mod p). These public keys are then exchanged between the parties.

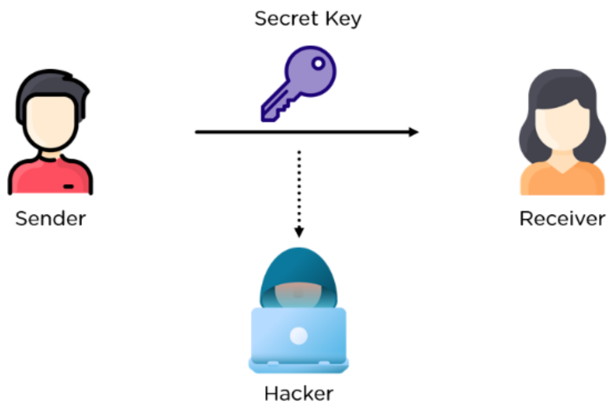


Figure 3.3.1 Secret Key Sharing

4. Shared Secret Calculation: Finally, both parties compute the shared secret key using the public key received from the other party and their own private key. The sender computes (S = B^a mod p) and the receiver computes (S = A^b mod p). Since both computations yield the same result due to the properties of modular exponentiation, both parties arrive at the same shared secret key, which they can then use for secure communication.

In evaluating the sender and receiver authentication using Diffie-Hellman, the shared secret key obtained through the key exchange process serves as a form of mutual authentication. By successfully computing the same shared secret key, both parties demonstrate knowledge of their respective private keys and validate each other's identity without explicitly transmitting sensitive information. This shared secret key can subsequently be used to encrypt and decrypt messages exchanged between the sender and receiver, ensuring confidentiality and integrity in communication. Additionally, the Diffie-Hellman key exchange protocol provides forward secrecy, meaning that even if an attacker intercepts the communication and obtains the shared secret key, past and future communications remain secure, as the private keys used for key generation are never transmitted. Thus, Diffie-Hellman authentication offers a robust and efficient mechanism for establishing secure communication channels between parties in various cryptographic applications.

3.4. Camellia Cipher

The Camellia cipher stands out as a formidable symmetric key block cipher renowned for its formidable security measures and seamless integration across diverse hardware and software environments. Offering the flexibility of key lengths ranging from 128 to 256 bits, it caters to a wide spectrum of encryption needs, ensuring robust protection for sensitive data. Originating from Japan, Camellia has garnered global acclaim and recognition, earning endorsements as a standard encryption algorithm from prestigious organizations such as the European Union. Its widespread adoption underscores its reliability and effectiveness in safeguarding information against unauthorized access and malicious threats.

In our project, we specifically leverage the strength of the 256-bit key variant of the Camellia cipher, capitalizing on its advanced encryption capabilities to ensure the secure handling of critical data. By employing this cipher, we aim to fortify the security measures of our system, thereby enhancing the confidentiality and integrity of transmitted information. The choice of Camellia as our encryption algorithm reflects our commitment to leveraging cutting-edge cryptographic techniques to address the evolving challenges of data security in digital communication environments.

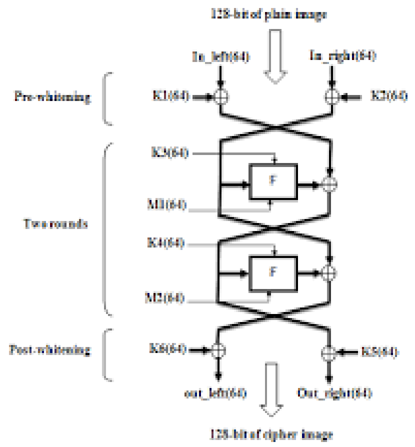


Figure 3.4. Camellia Cipher

3.4.1 Initialization Vector (IV)

In Camellia encryption, the Initialization Vector (IV) serves as a critical component in fortifying the security of the encrypted data. By introducing randomness through a 128-bit value generated for each encryption operation, the IV ensures that ciphertexts remain distinct even when encrypting the same plaintext with the same key multiple times. This prevents adversaries from discerning patterns or repetitions within the ciphertext, thereby thwarting potential attacks aimed at exploiting such vulnerabilities to compromise the security of the encryption scheme.

Furthermore, the utilization of unique IVs in Camellia encryption enhances the overall resilience against cryptographic attacks. With distinct IVs for each encryption instance, the encrypted data becomes more resistant to techniques like frequency analysis or chosen-plaintext attacks, which rely on identifying patterns or correlations within the ciphertext. By mitigating the risk of such attacks, the project's implementation of Camellia encryption reinforces the confidentiality and integrity of sensitive data, aligning with the overarching objective of ensuring robust security measures in cryptographic operations.

3.4.2. SHA-256

SHA-256, a cryptographic hash algorithm, is widely employed across various security applications due to its robustness and reliability. Designed to accept input data of any size and generate a fixed-length output of 256 bits, SHA-256 ensures consistency and uniformity in hash values, regardless of the size or complexity of the input. As a member of the SHA-2 family, SHA-256 inherits the fundamental properties of its predecessors, including resistance to cryptographic attacks such as pre-image, second pre-image, and collision attacks. This makes SHA-256 a preferred choice for applications requiring secure and efficient hashing operations, such as password storage, digital signatures, and data integrity verification.One of the key advantages of SHA-256 is its resistance to collisions, which occur when two different inputs produce the same hash value. Due to its large output size of 256 bits and the complex mathematical operations involved in its computation, the likelihood of two distinct inputs resulting in the same hash value is exceedingly low. This property is crucial for maintaining the integrity and authenticity of hashed data, as it ensures that even minor changes in the input data lead to drastically different hash values. Consequently, SHA-256 is widely utilized in cryptographic protocols and security mechanisms where data integrity and uniqueness are paramount.

Moreover, SHA-256's widespread adoption in various security protocols and standards underscores its reputation as a reliable and trustworthy cryptographic hash algorithm. Its inclusion in cryptographic frameworks such as SSL/TLS, IPsec, and digital signatures exemplifies its versatility and applicability in securing communication channels, protecting sensitive information, and verifying the authenticity of digital assets. By providing a secure and efficient means of generating hash values for diverse applications, SHA-256 plays a crucial role in the modern cybersecurity landscape, safeguarding against data tampering, forgery, and unauthorized access.

3.4.3 Cipher Feedback (CFB)

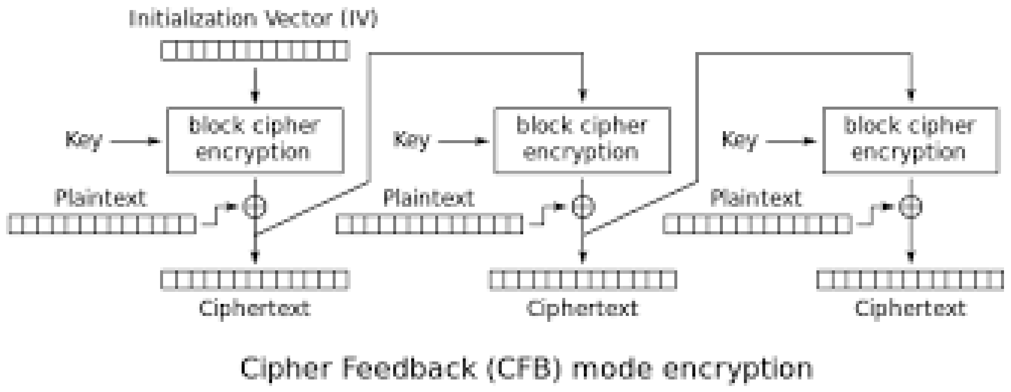


Fig 3.4.3.1 Cipher Feedback

Cipher Feedback (CFB) mode is a block cipher mode of operation that transforms a block cipher into a stream cipher, enabling encryption and decryption of data of arbitrary length. It operates by repeatedly encrypting an initialization vector (IV) or the previous ciphertext block to generate a keystream, which is then combined with the plaintext using a bitwise XOR operation to produce the ciphertext. CFB mode offers flexibility for handling plaintext of any size while maintaining security properties such as confidentiality and error propagation. However, proper IV management is crucial to prevent cryptographic vulnerabilities, as reusing IVs can lead to security weaknesses.

CFB mode's ability to transform block ciphers into stream ciphers makes it particularly well-suited for applications where data transmission occurs in real-time or in a continuous stream, such as network communication protocols or disk encryption. Its streaming nature enables efficient processing of large volumes of data without the need to buffer entire blocks, resulting in reduced memory requirements and improved performance. However, like any cryptographic mode of operation, CFB mode has its limitations and considerations. It may exhibit poor performance in scenarios where random access to ciphertext blocks is required, as it relies on the previous ciphertext block or IV for encryption. Additionally, the initialization vector (IV) used in CFB mode must be unpredictable and unique for each encryption operation to prevent the repetition of keystreams and mitigate the risk of known-plaintext attacks. Careful implementation and adherence to cryptographic best practices are essential to leverage the benefits of CFB mode while mitigating potential security risks.

3.5. LZW Compression

Lempel-Ziv-Welch (LZW) compression is renowned for its efficiency in reducing the size of data without sacrificing its quality, making it a staple in various file compression utilities. The algorithm operates by analyzing the input data and identifying recurring patterns, which are then substituted with shorter codes. By maintaining a dictionary of these patterns and their corresponding codes, LZW significantly reduces the redundancy within the data, leading to notable reductions in file size. One of the key advantages of LZW compression lies in its adaptability to a wide range of data types and formats. Whether handling text, images, or other types of binary data, LZW can effectively identify and encode repetitive sequences, resulting in impressive compression ratios. This versatility has contributed to the algorithm's widespread adoption across different domains, from archiving and storage to data transmission over networks.

Furthermore, LZW compression offers a balance between compression efficiency and computational complexity, striking a favorable compromise for many applications. While achieving high compression ratios, the algorithm maintains relatively fast encoding and decoding speeds, ensuring practicality in real-world scenarios where both compression performance and processing efficiency are crucial. Overall, LZW's robustness, versatility, and efficiency make it a cornerstone in the realm of lossless data compression, serving as a reliable tool for optimizing storage space and enhancing data transfer efficiency.

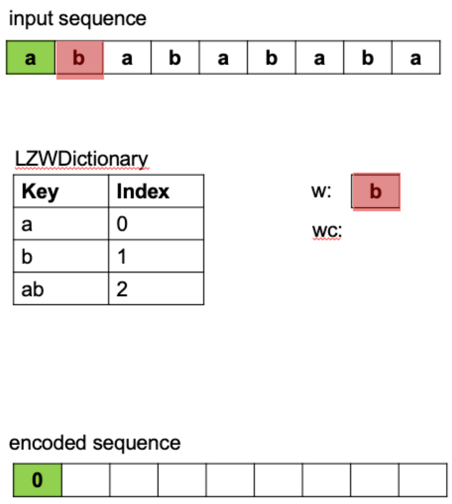


Figure 3.5 LZW Compression

3.6. LSB Steganography

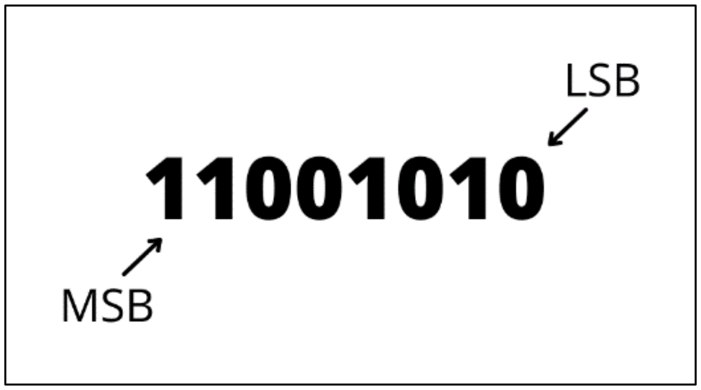


Figure 3.6.1. LSB & MSB

LSB (Least Significant Bit) steganography is a technique used to hide information within digital images by subtly altering the least significant bits of the pixel values in the red, green, and blue color channels. This method takes advantage of the fact that slight changes in the least significant bits typically have minimal impact on the overall appearance of the image, making it imperceptible to the human eye. By replacing these insignificant portions of pixel values with hidden data, LSB steganography allows for the concealment of information without significantly altering the visual quality of the image. One of the key advantages of LSB steganography is its simplicity and ease of implementation. Since it involves only minor modifications to the pixel values, it can be applied using straightforward algorithms, making it accessible to both novice and experienced users. Additionally, LSB steganography can be employed in various contexts, ranging from digital watermarking and copyright protection to covert communication and data hiding.

However, despite its widespread use and ease of implementation, LSB steganography has several limitations and vulnerabilities. One of the primary concerns is its susceptibility to detection by advanced steganalysis techniques, particularly when the hidden data occupies a significant portion of the LSBs, leading to detectable alterations in the image. Furthermore, LSB steganography may not be suitable for applications requiring high levels of security or robustness against sophisticated attacks, as it lacks the encryption and authentication mechanisms offered by more advanced steganographic methods. Overall, while LSB steganography remains a popular and widely used technique for basic image steganography, its effectiveness and suitability depend on the specific requirements and security considerations of each application. As with any steganographic method, careful consideration of the trade-offs between simplicity, security, and robustness is essential when employing LSB steganography for concealing sensitive information within digital images.

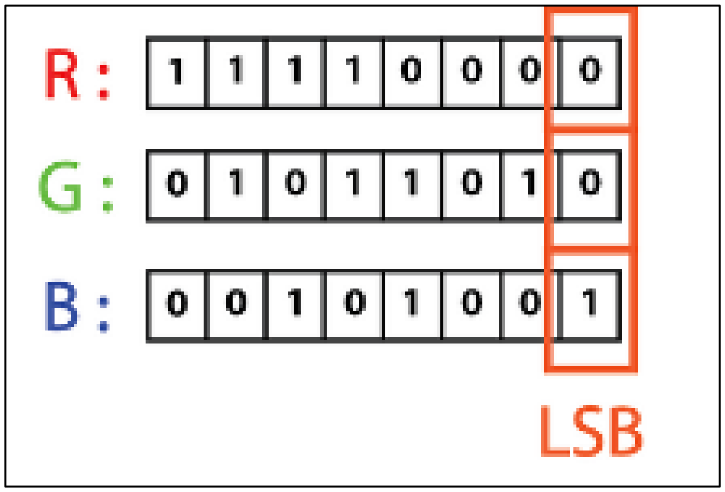


Fig 3.6.2: RGB Colours in a pixel

The project on LSB steganography serves multiple purposes, with both practical and technical implications. From a practical standpoint, implementing LSB steganography enables users to embed hidden messages or information within digital images, facilitating covert communication, data hiding, and digital watermarking. This capability has various applications in fields such as cybersecurity, digital forensics, copyright protection, and secure communication, where maintaining the confidentiality and integrity of information is crucial. Moreover, the technical impact of the project lies in its exploration and demonstration of the underlying principles and techniques of LSB steganography. By understanding how LSB steganography works and its implications for image data, researchers and practitioners can gain insights into the strengths, weaknesses, and limitations of this steganographic method. This knowledge can inform the development of more robust and secure steganographic algorithms, as well as contribute to advancements in steganalysis techniques for detecting hidden information within digital images.

Additionally, the project provides a platform for experimentation and learning about steganography, encryption, and information security concepts. Through hands-on implementation and analysis of LSB steganography algorithms, students and researchers can deepen their understanding of cryptographic techniques, data encoding methods, and image processing algorithms. This hands-on experience fosters critical thinking, problem-solving skills, and creativity in addressing security challenges in digital communication and information protection.

Overall, by investigating LSB steganography and its applications, the project not only offers practical utility for concealing information within digital images but also contributes to the broader understanding of steganographic techniques, encryption principles, and cybersecurity concepts. Through its technical exploration and practical implications, the project has the potential to enhance security practices, stimulate research in information hiding technologies, and empower individuals and organizations to safeguard their digital assets and communications effectively.

The exploration of LSB steganography and its practical applications underscores the intricate interplay between data concealment and encryption mechanisms in contemporary information security landscapes. By leveraging LSB embedding techniques, the project provides a nuanced understanding of how covert communication channels can be established within seemingly innocuous digital assets, such as images, to safeguard sensitive information from prying eyes. Furthermore, the project's investigation into encryption principles elucidates the pivotal role cryptography plays in securing these concealed data payloads, ensuring that only authorized parties possess the means to decipher and access the hidden information. Through this dual approach of steganography and cryptography, the project not only elucidates the complexities of digital concealment but also underscores the importance of robust encryption methodologies in safeguarding confidential data against malicious actors and cyber threats.

Additionally, the project's contributions extend beyond the technical realm to encompass broader implications for privacy, security, and digital forensics in contemporary society. By shedding light on the intricacies of LSB steganography and encryption techniques, the project fosters critical discourse surrounding the ethical, legal, and societal implications of covert communication methods in digital environments. Moreover, the project's insights into information hiding technologies empower individuals, organizations, and policymakers to make informed decisions regarding data protection, privacy regulations, and cybersecurity protocols. Ultimately, by promoting awareness, education, and innovation in the fields of steganography and cryptography, the project catalyzes positive change in the ongoing quest to balance the imperatives of security, privacy, and digital freedom in the modern age of information.

3.7 Benefits

The process of cover image selection in steganography offers several benefits crucial to the success and effectiveness of the concealment process. Firstly, it allows for the optimization of embedding capacity by identifying images with suitable characteristics, such as high entropy, complex textures, and diverse color distributions. By selecting cover images that inherently possess these qualities, steganographic algorithms can maximize the payload capacity while minimizing the risk of detection. Additionally, cover image selection enhances security by enabling the integration of authentication mechanisms, such as image similarity analysis and embedding distortion evaluation. This ensures that the chosen cover image not only conceals the hidden data effectively but also mitigates the risk of unintended alterations or tampering. Furthermore, through careful selection based on visual similarity and embedding distortion, the steganographic process can achieve seamless integration of the hidden information, preserving the cover image's perceptual quality and reducing the likelihood of suspicion. Overall, cover image selection plays a pivotal role in optimizing embedding efficiency, enhancing security, and maintaining the integrity of steganographic communication channels.

Moreover, the process of cover image selection in steganography contributes to robustness against various attacks and enhances the resilience of the hidden data. By choosing cover images with characteristics that are resilient to common steganalysis techniques, such as statistical analysis and visual inspection, the concealment process becomes more resistant to detection efforts. Furthermore, the careful consideration of cover images based on their inherent properties can improve the survivability of the hidden information against image transformations, compression algorithms, and other forms of digital manipulation. This ensures that the concealed data remains intact and retrievable even in adverse conditions or when subjected to unintended modifications.

4. IMPLEMENTATION

There are series of program files in our project which are:

1. project1_compressed.py
2. project1_uncompressed.py
3. project1_final.py
4. project2.py
5. project3.py
6. project4.py
7. project5.py

And the other program files which are also used in the project but not in the mainstream are:

1. majormetrics.py
2. imgcompression.py

2

INPUTS:

1. 8 Compressed Images
2. 8 Uncompressed Images
3. Text file with Input Text

OUTPUT:

Stego Image transmitted to Receiver

4.1 Functionality

Here is a breakdown of the functions and what are the aspects in which we can understand about all these

4.1.1 Selecting a Cover Image

This code starts with Calculating the color histograms of the image and then displaying the properties of the image and evaluating each image based on the parameters.

1) **Efficient Color Histogram Calculation Using OpenCV:** This function efficiently computes the color histogram of an image using OpenCV. It adapts to both single-channel and multi-channel images, constructs channel ranges accordingly, and employs the `calcHist` function to generate the histogram. The resulting flattened histogram array provides a concise representation of the image's color distribution, aiding in various image processing tasks such as feature extraction and color-based segmentation.

2) **Display Main Screen:** This function serves as the initial interface for the program, providing users with an overview of the major project details. It clears the terminal screen for a clean display and then prints relevant project information such as the project title, team details, and current stage of the project. The purpose of this function is to offer users a clear starting point and set the context for the tasks ahead. By presenting essential project details in a structured manner, it helps users navigate through the program and understand its objectives more effectively. Additionally, this function enhances user experience by creating a visually appealing main screen, which aids in engaging users and maintaining their interest in the project.

3) **Display All Images :** The purpose of this function is to visualize a set of images along with their corresponding titles in a grid format. It creates a subplot grid and iterates over the input images, displaying each image along with its title. By presenting multiple images simultaneously, it allows users to compare and analyze images more efficiently. This function is typically used to showcase a batch of images, aiding in tasks such as image selection, quality assessment, or result presentation.

4) **Display Selected Images:** This function displays a subset of selected images, typically chosen based on specific criteria or analysis results. It creates a subplot grid and visualizes the selected images along with their titles. The purpose of this function is to focus the user's attention on a specific set of images that are relevant

to the current task or analysis. By presenting only the selected images, it helps users make informed decisions or draw conclusions more effectively.

5) **Display Individual Histograms and Statistics:** This function provides a comprehensive visualization of color histograms and additional statistics for individual images. It calculates separate histograms for each color channel and displays them alongside the original image. Additionally, it computes various statistics such as image dimensions, number of pixels, and mean channel intensities, presenting them in a tabular format. The purpose of this function is to offer detailed insights into the color distribution and characteristics of individual images, aiding in image analysis and understanding.

6) **Entropy Calculation:** This function computes the entropy of an input image, which quantifies the randomness or uncertainty in its pixel intensity distribution. It first calculates the color histogram of the image using the previously defined `calculate_color_histogram` function and then computes the entropy based on the histogram. Entropy serves as a useful metric for assessing the information content of an image, with higher entropy indicating greater complexity or diversity in pixel values.

7) **Fractal Dimension Calculation:** This function estimates the fractal dimension of an image without relying on external libraries. It converts the input image to grayscale and performs binary thresholding to distinguish foreground and background pixels. By counting the number of white (foreground) and black (background) pixels, it computes the fractal dimension, which characterizes the image's self-similarity or geometric complexity.

8) **Edge Detection Calculation:** This function utilizes the Canny edge detection algorithm to identify edges in the input image. It first converts the image to grayscale and then applies Canny edge detection to detect edges. The function

returns both the edge density, representing the proportion of pixels classified as edges, and the edge map, visualizing the detected edges.

9) **Mark High Features:** This function highlights high-feature areas in selected images based on their edge density maps. It visualizes the original images alongside their corresponding edge detection maps, with areas of high edge density marked in red. By setting a threshold, users can adjust the sensitivity of the feature detection process.

10) **Cumulative Metrics Calculation:** This function computes cumulative metrics for a set of selected images, including entropy, fractal dimension, and edge detection. It iterates over the selected images and calculates these metrics using the corresponding helper functions. Then, it combines these metrics into a single cumulative value for each image, incorporating entropy, edge density, and fractal dimension. Finally, it returns lists of individual metrics along with their cumulative values.

11) **Display Cumulative Metrics Table:** This function displays a tabulated summary of cumulative metrics for the selected images. It organizes the entropy, fractal dimension, edge detection, and cumulative metric values into a structured table format. Each row corresponds to a specific image, with columns representing different metrics. The table provides a comprehensive overview of the computed metrics, aiding in the comparative analysis of image characteristics.

4.1.2 Diffie Hellman Authentication

1) **sender_partial_dh:** This function implements the partial Diffie-Hellman key exchange process from the sender's perspective. It generates prime numbers `p` and `g` as public parameters, and a sender's private key (`sender_private_key`). Then, it computes the sender's partial public value `x` using the generated parameters.

The function prints out the public parameters, sender's private key, and the calculated partial public value `x`, providing insights into the sender's role in the Diffie-Hellman key exchange..

2) **receiver_partial_dh:** This function simulates the partial Diffie-Hellman key exchange process from the receiver's perspective. It prompts the receiver to enter the public parameters `p` and `g` shared by the sender. Then, it computes the receiver's partial public value `y` using the entered parameters and the receiver's private key. The function prints out the receiver's private key, the computed partial public value `y`, and prompts the user to input the received partial public value `y`. This function elucidates the receiver's involvement in the Diffie-Hellman key exchange.

3) **complete_dh:** This function calculates the shared secret key between the sender and the receiver based on their partial public values and private keys. It takes `p`, `g`, the private key, and the partial public value as inputs and computes the shared secret using the Diffie-Hellman key exchange algorithm. The function returns the shared secret key, representing the confidential key established between the communicating parties. This function encapsulates the core computation step of the Diffie-Hellman key exchange protocol.

4.1.3 LZW Compression

1) **compress_lzw:** This function implements the LZW (Lempel-Ziv-Welch) compression algorithm to compress input data. It initializes a dictionary with ASCII characters and their corresponding codes. Then, it iterates over the input data, encoding sequences of characters into codes. When encountering a new sequence not present in the dictionary, it adds it to the dictionary and assigns a new code. Finally, it returns the compressed data as a list of codes. This function encapsulates the core logic of LZW compression.

2) **compress_file_lzw:** This function compresses the content of a text file using the LZW algorithm. It reads the content of the input file, applies the `compress_lzw` function to compress the data, and writes the compressed data to the output file. The function returns the compressed data as a list of codes. It demonstrates how to utilize the `compress_lzw` function to compress file contents efficiently.

4.1.4 Steganography Process

1) **Encode:** The encode function is responsible for encoding a message into an image using LSB (Least Significant Bit) steganography. It takes an image object, a message to be encoded, and a filename as input parameters. The function iterates over the pixels of the image and modifies the least significant bits of each pixel to represent the binary value of characters in the message. After encoding, it saves the modified image with the specified filename.

2) **Decode:** The decode function is used to retrieve a message hidden within an image using LSB steganography. It takes an image object as input and iterates over the pixels to extract the least significant bits, reconstructing the binary representation of the hidden message. Once the binary message is reconstructed, it is converted back to text format and returned as the decoded message.

3) **Encrypt:** The encrypt function performs encryption of a plaintext message using the Camellia cipher algorithm. It takes a key and a plaintext message as input parameters. The function hashes the key using SHA-256 for security, generates a random initialization vector (IV), and encrypts the padded plaintext using the Camellia cipher in CFB mode. The ciphertext, along with IV, is returned as the encrypted output.

4) **Decrypt:** The decrypt function is responsible for decrypting an encrypted message using the Camellia cipher algorithm. It takes a key and an encrypted ciphertext as

input parameters. The function hashes the key using SHA-256, extracts the IV from the ciphertext, and decrypts the ciphertext using the Camellia cipher in CFB mode. The decrypted plaintext is then unpadded and returned as the decrypted output.

5) **convertToRGBAndSave:** This function converts an image from grayscale, CMYK, or RGBA mode to RGB mode and saves the converted image. It takes an image object and an output filename as input, converts the image to RGB mode, and saves it with the specified filename. This conversion ensures uniformity in image processing and compatibility with certain operations.

6) **getPixelCount :** The getPixelCount function calculates the total number of pixels in an image. It takes the path to an image file as input, opens the image, and retrieves its dimensions. It then calculates and returns the total pixel count by multiplying the width and height of the image.

7) **is_png_image:** This function checks whether an image is in PNG format. It takes the path to an image file as input, attempts to open the image using PIL, and checks if the image format is PNG. It returns True if the image format is PNG, otherwise False.

8) **show_image:** The show_image function displays an image on the screen. It prompts the user to enter "OPEN" to display the image. If the user enters "OPEN", the function opens and displays the image using the PIL library's show method. If the image file is not found or an error occurs, appropriate error messages are displayed.

4.1.5 LZW Decompression

1) **decompress_lzw:** This function implements the LZW (Lempel-Ziv-Welch) decompression algorithm. It takes a list of compressed data as input, where each element represents a code from the compressed sequence. The function initializes

a dictionary with ASCII values as keys and characters as values. It iterates through the compressed data, reconstructing the original uncompressed text based on the dictionary. If a code is found in the dictionary, its corresponding character is appended to the result. If not, a new entry is created in the dictionary using the previously decoded sequence. The function returns the decompressed text.

4.2 Attributes

1) **`compressed_data`:** This attribute holds the compressed data read from the input file. It is a list containing integer codes representing the compressed sequence. During decompression, this data is used as input to reconstruct the original uncompressed text.

2) **`input_filename`:** Represents the filename of the input file containing the compressed data to be decompressed. It serves as a reference point for the program to locate and read the compressed data for decompression.

3) **`output_filename`:** Denotes the filename of the output file where the decompressed text will be saved. This attribute specifies the destination for storing the decompressed text after the decompression process is completed.

4) **`decompressed_text_lzw`:** Stores the decompressed text obtained after applying the LZW decompression algorithm. This variable holds the resulting text generated from the compressed data, representing the original uncompressed content.

5) **`dictionary`:** A data structure used in the LZW decompression algorithm to maintain mappings of codes to characters. Throughout the decompression process, this dictionary is updated dynamically based on the encountered encoded sequences.

6) **`result`:** Represents the result of the LZW decompression algorithm, which is the decompressed text. As the algorithm processes the compressed data, this variable accumulates the reconstructed text until the entire sequence is decompressed.

7) **`original_text':** `original_text` stores the content of the input file before compression. It represents the text data that the sender intends to compress using the LZW algorithm.

8) **`original_size':** This attribute holds the size of the original text data in bytes before compression. It provides information about the initial size of the data, which is useful for calculating compression ratios.

9) **`plaintext_bits`**: plaintext_bits` represents the binary representation of the compressed message. Each character in the compressed message is converted to its ASCII value and then converted to an 8-bit binary string. This attribute is used to display the compressed text in binary format.

10) **`compressed_size_bytes`:** Stores the size of the compressed message in bytes. It indicates the amount of storage required to store the compressed data in the output file.

11) **`compressed_characters`:** This attribute represents the number of characters in the compressed text. It helps to quantify the size of the compressed data and is used to calculate compression ratios.

12) **DEBUG:** This attribute is a boolean variable used to control whether debug information should be displayed during the execution of the program. When set to `True`, debug information will be printed, aiding in troubleshooting and understanding the program flow.

13) **console:** This attribute represents an instance of the `Console` class from the Rich package. It is used to interact with the console and provides functionalities for rich text formatting and printing, enhancing the visual presentation of the program's output.

14) **headertext:** This attribute stores the header text that is added before encoding the image. It is used as a marker to identify the beginning of the encoded data within the image. The header text is combined with the ciphertext before embedding it into the image.

15) **total_pixels:** This attribute stores the total number of pixels in the image. It is calculated based on the width and height of the image and is used to determine the maximum amount of message data that can be encoded into the image.

16) **password:** This attribute stores the password entered by the user for encryption and decryption purposes. It is used as the encryption key to encrypt the message data before embedding it into the image and as the decryption key to decrypt the encoded data extracted from the image.

17) **image_paths:** This attribute is a list containing the file paths of the images to be processed.

18) **images:** This attribute is a list containing the loaded images obtained from the image paths.

19) **selected_images:** This attribute is a list containing the resized versions of the images to ensure they have the same dimensions for comparative analysis.

20) **min_height:** This attribute stores the minimum height among all the images.

21) **min_width:** This attribute stores the minimum width among all the images.

4.3 Experimental Screenshot

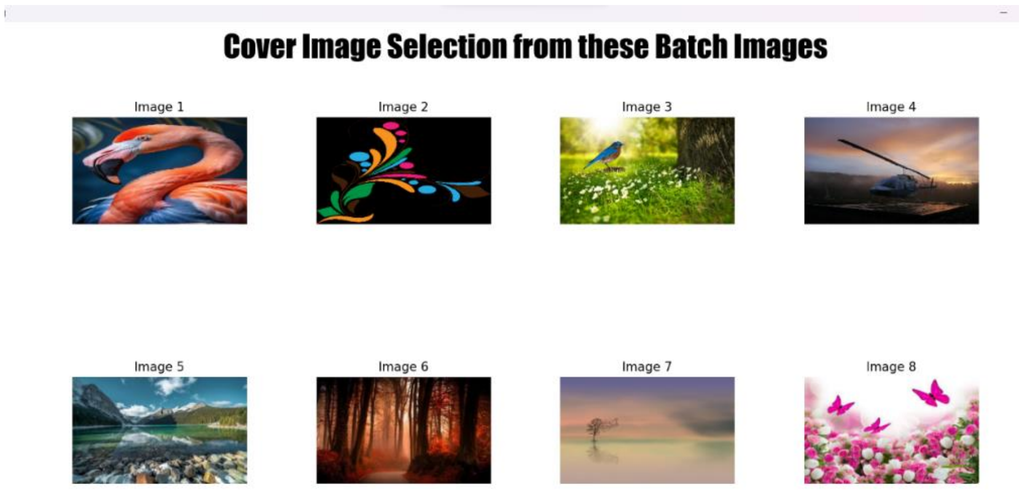
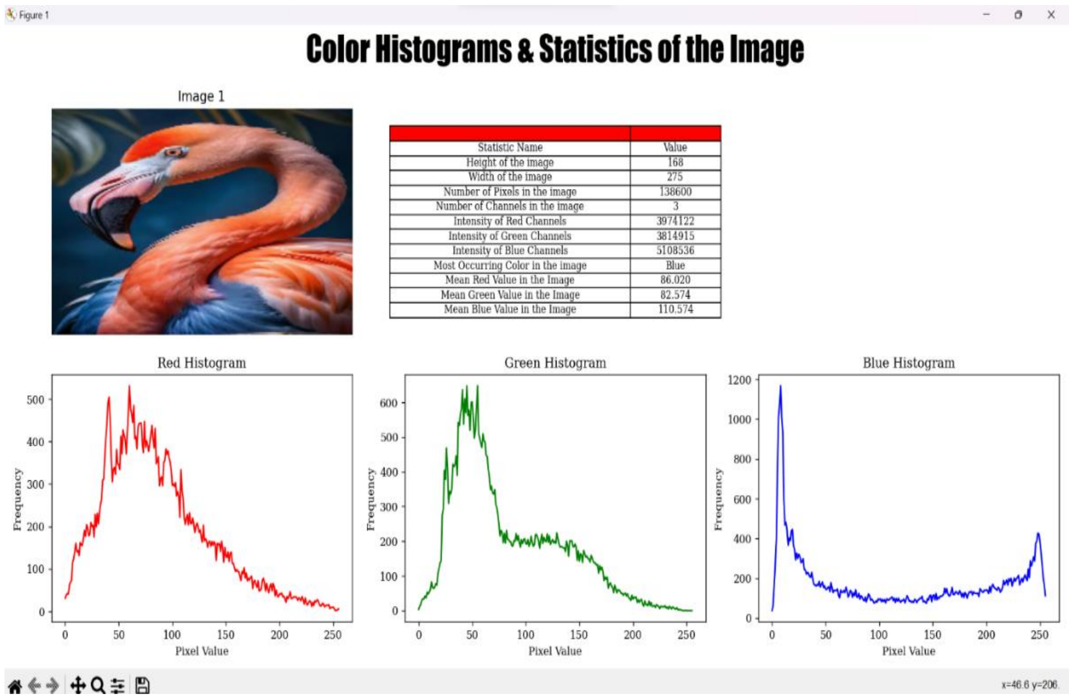
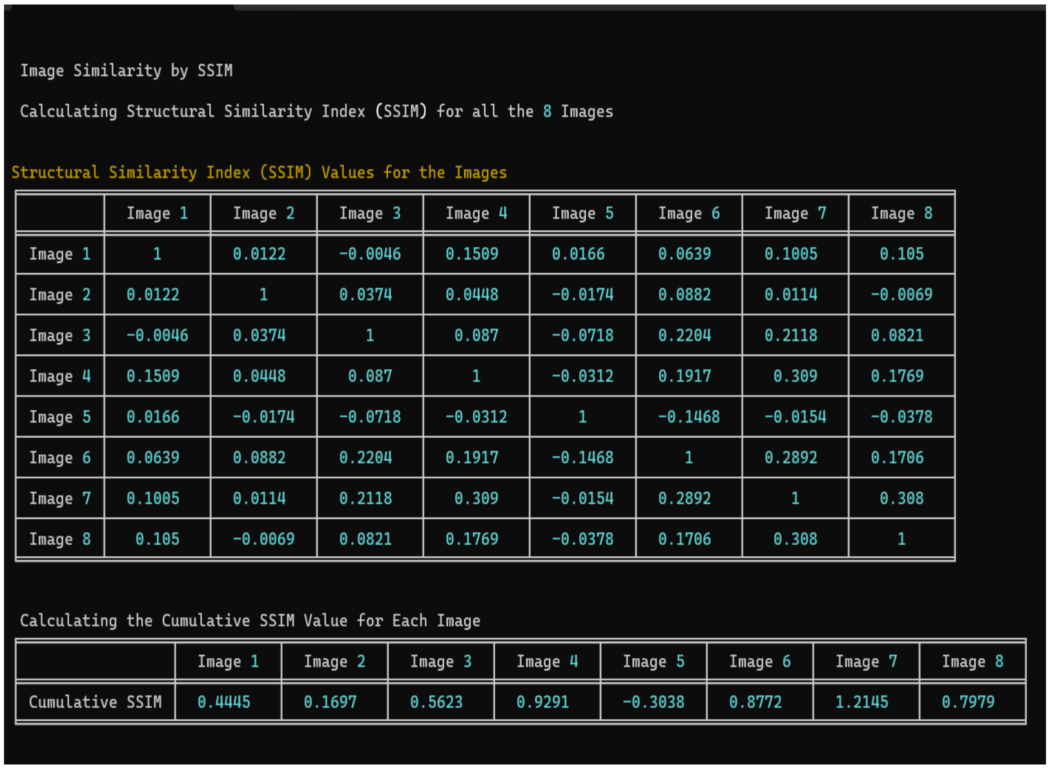


Figure 4.3.1: Cover Image Selection



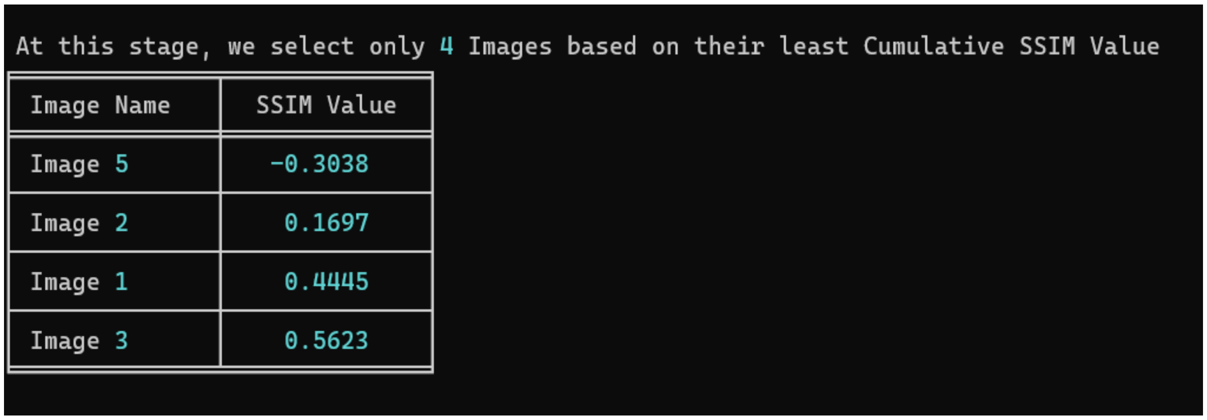
Statistic Name	Value
Height of the image	168
Width of the image	275
Number of Pixels in the image	138600
Number of Channels in the image	3
Intensity of Red Channels	3974122
Intensity of Green Channels	3814915
Intensity of Blue Channels	5108536
Most Occurring Color in the image	Blue
Mean Red Value in the Image	86.020
Mean Green Value in the Image	82.574
Mean Blue Value in the Image	110.574

Figure 4.3.2: Color Histograms & Statistics of Image



```
Image Similarity by SSIM

Calculating Structural Similarity Index (SSIM) for all the 8 Images

Structural Similarity Index (SSIM) Values for the Images
|         | Image 1 | Image 2 | Image 3 | Image 4 | Image 5 | Image 6 | Image 7 | Image 8 |
|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| Image 1 |    1    | 0.0122  | -0.0046 | 0.1509  | 0.0166  | 0.0639  | 0.1005  | 0.105   |
| Image 2 | 0.0122  |    1    | 0.0374  | 0.0448  | -0.0174 | 0.0882  | 0.0114  | -0.0069 |
| Image 3 | -0.0046 | 0.0374  |    1    |  0.087  | -0.0718 | 0.2204  | 0.2118  | 0.0821  |
| Image 4 | 0.1509  | 0.0448  |  0.087  |    1    | -0.0312 | 0.1917  |  0.309  | 0.1769  |
| Image 5 | 0.0166  | -0.0174 | -0.0718 | -0.0312 |    1    | -0.1468 | -0.0154 | -0.0378 |
| Image 6 | 0.0639  | 0.0882  | 0.2204  | 0.1917  | -0.1468 |    1    | 0.2892  | 0.1706  |
| Image 7 | 0.1005  | 0.0114  | 0.2118  |  0.309  | -0.0154 | 0.2892  |    1    |  0.308  |
| Image 8 |  0.105  | -0.0069 | 0.0821  | 0.1769  | -0.0378 | 0.1706  |  0.308  |    1    |

Calculating the Cumulative SSIM Value for Each Image
|                 | Image 1 | Image 2 | Image 3 | Image 4 | Image 5 | Image 6 | Image 7 | Image 8 |
|-----------------|---------|---------|---------|---------|---------|---------|---------|---------|
| Cumulative SSIM | 0.4445  | 0.1697  | 0.5623  | 0.9291  | -0.3038 | 0.8772  | 1.2145  | 0.7979  |
```

Figure 4.3.3: SSIM Values for Images



```
At this stage, we select only 4 Images based on their least Cumulative SSIM Value
| Image Name | SSIM Value |
|------------|------------|
| Image 5    |  -0.3038   |
| Image 2    |   0.1697   |
| Image 1    |   0.4445   |
| Image 3    |   0.5623   |
```

Figure 4.3.4: SSIM Values of 4 Selected Images

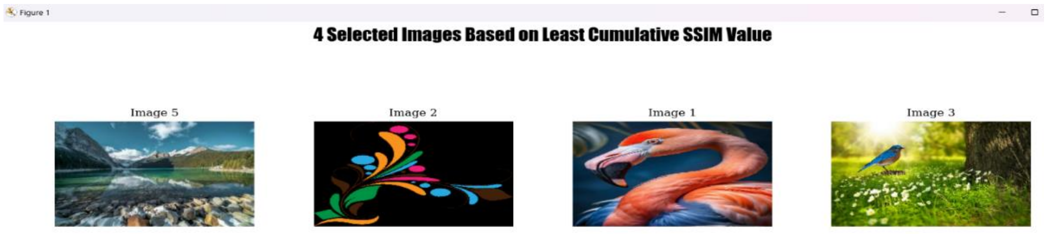


Figure 4.3.5: 4 Selected Images based on Least SSIM

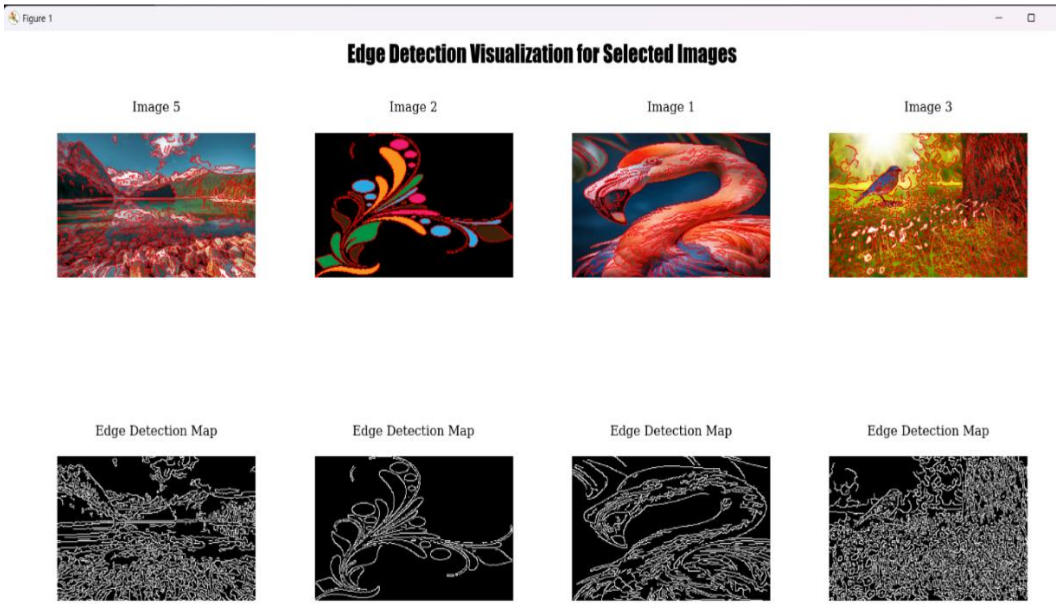
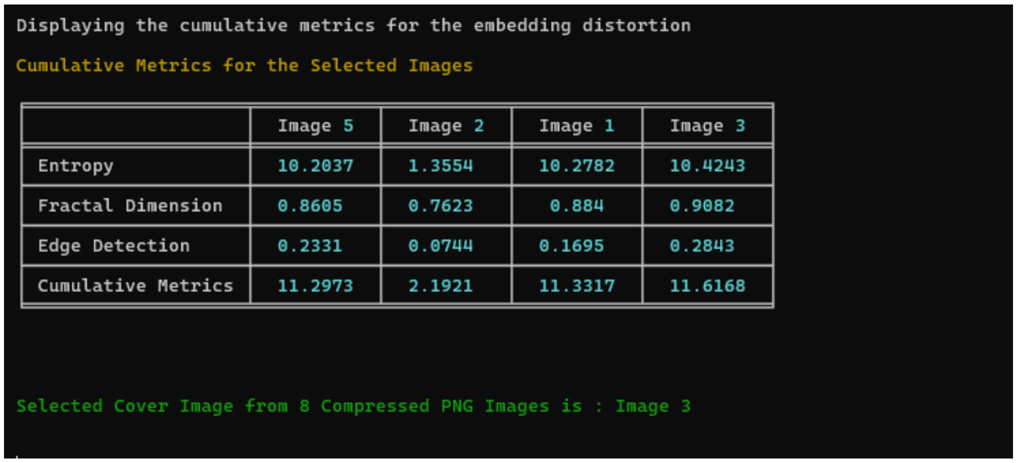


Figure 4.3.6: Edge Detection Visualization



```
Displaying the cumulative metrics for the embedding distortion

Cumulative Metrics for the Selected Images

+----------------------+-----------+-----------+-----------+-----------+
|                      |  Image 5  |  Image 2  |  Image 1  |  Image 3  |
+----------------------+-----------+-----------+-----------+-----------+
| Entropy              |  10.2037  |  1.3554   |  10.2782  |  10.4243  |
| Fractal Dimension    |  0.8605   |  0.7623   |   0.884   |  0.9082   |
| Edge Detection       |  0.2331   |  0.0744   |  0.1695   |  0.2843   |
| Cumulative Metrics   |  11.2973  |  2.1921   |  11.3317  |  11.6168  |
+----------------------+-----------+-----------+-----------+-----------+


Selected Cover Image from 8 Compressed PNG Images is : Image 3
```

Figure 4.3.7: Cumulative Metrics for Selected Images

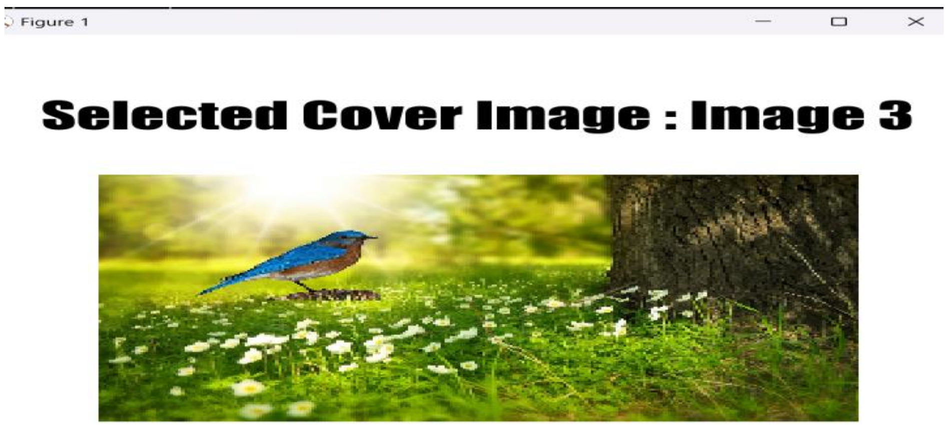


Figure 4.3.8: Selected Cover Image from the Compressed PNG Images

Same Process is repeated for the uncompressed Images and the Selected Cover Image among the Uncompressed PNG Images is:

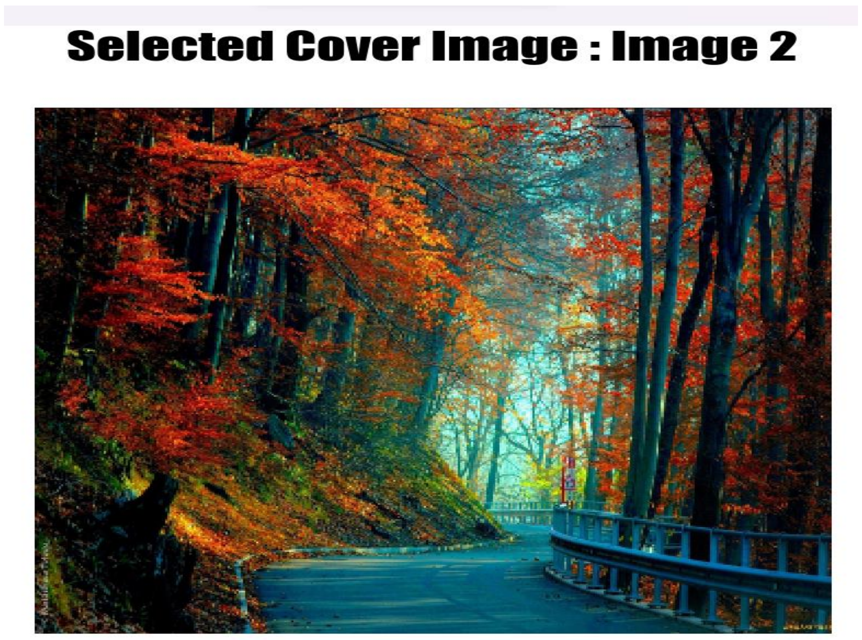
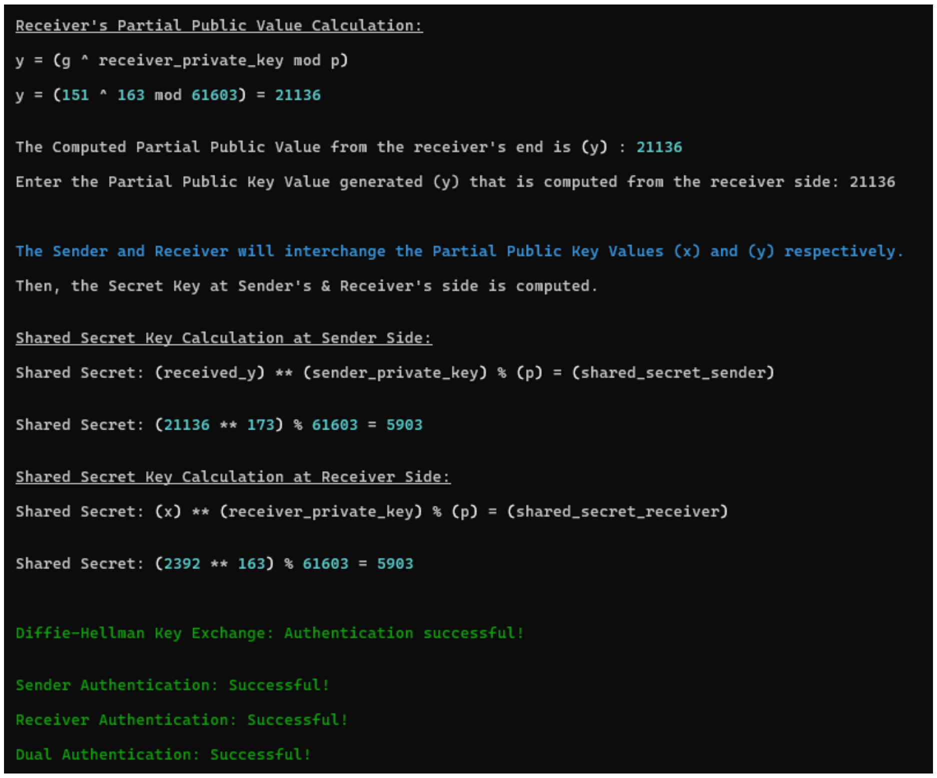


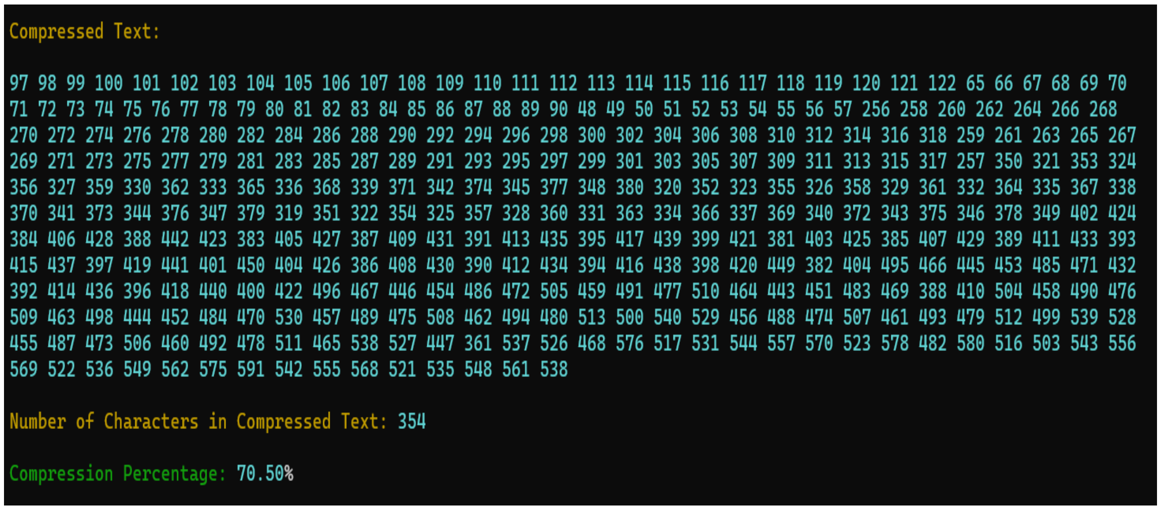
Figure 4.3.9: Selected Cover Image from Uncompressed PNG Images

Figure 4.3.10: Selected Final Cover Image



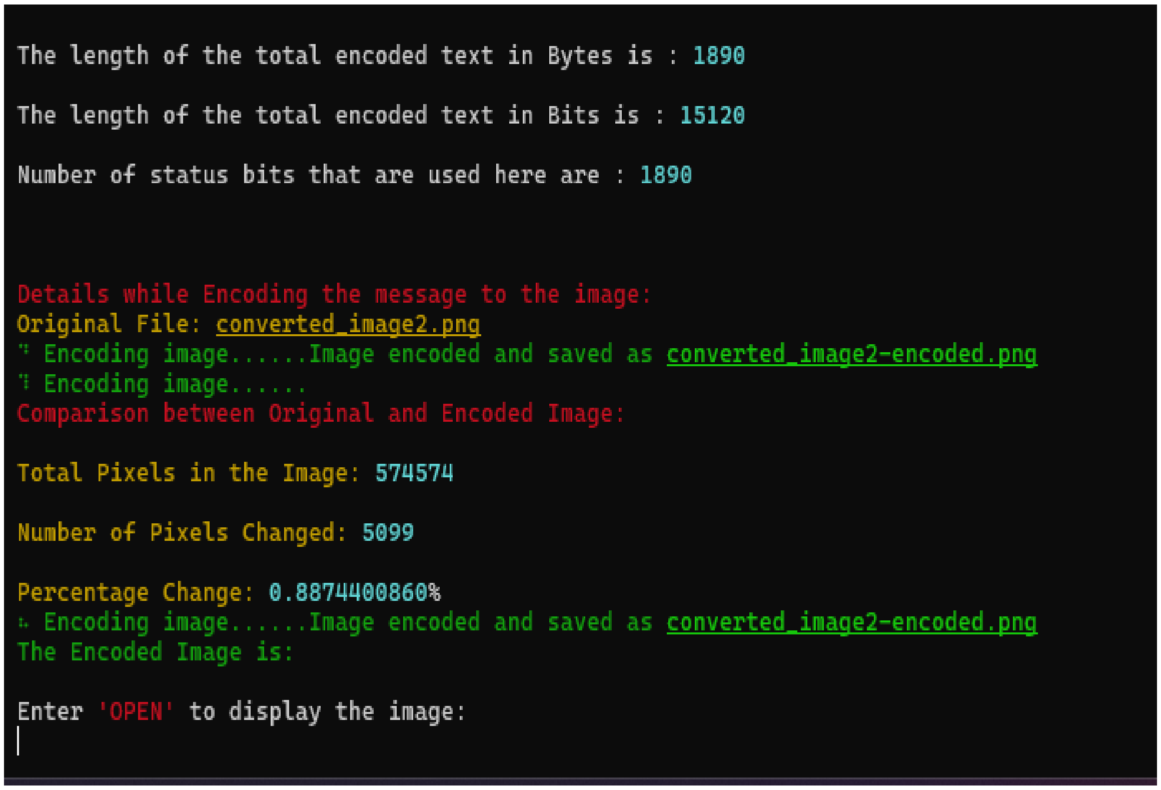
```
Receiver's Partial Public Value Calculation:

y = (g ^ receiver_private_key mod p)

y = (151 ^ 163 mod 61603) = 21136

The Computed Partial Public Value from the receiver's end is (y) : 21136

Enter the Partial Public Key Value generated (y) that is computed from the receiver side: 21136


The Sender and Receiver will interchange the Partial Public Key Values (x) and (y) respectively.

Then, the Secret Key at Sender's & Receiver's side is computed.

Shared Secret Key Calculation at Sender Side:

Shared Secret: (received_y) ** (sender_private_key) % (p) = (shared_secret_sender)

Shared Secret: (21136 ** 173) % 61603 = 5903

Shared Secret Key Calculation at Receiver Side:

Shared Secret: (x) ** (receiver_private_key) % (p) = (shared_secret_receiver)

Shared Secret: (2392 ** 163) % 61603 = 5903


Diffie-Hellman Key Exchange: Authentication successful!

Sender Authentication: Successful!

Receiver Authentication: Successful!

Dual Authentication: Successful!
```

Figure 4.3.11: Diffie Hellman Authentication



```
Compressed Text:

97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 65 66 67 68 69 70
71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 48 49 50 51 52 53 54 55 56 57 256 258 260 262 264 266 268
270 272 274 276 278 280 282 284 286 288 290 292 294 296 298 300 302 304 306 308 310 312 314 316 318 259 261 263 265 267
269 271 273 275 277 279 281 283 285 287 289 291 293 295 297 299 301 303 305 307 309 311 313 315 317 257 350 321 353 324
356 327 359 330 362 333 365 336 368 339 371 342 374 345 377 348 380 320 352 323 355 326 358 329 361 332 364 335 367 338
370 341 373 344 376 347 379 319 351 322 354 325 357 328 360 331 363 334 366 337 369 340 372 343 375 346 378 349 402 424
384 406 428 388 442 423 383 405 427 387 409 431 391 413 435 395 417 439 399 421 381 403 425 385 407 429 389 411 433 393
415 437 397 419 441 401 450 404 426 386 408 430 390 412 434 394 416 438 398 420 449 382 404 495 466 445 453 485 471 432
392 414 436 396 418 440 400 422 496 467 446 454 486 472 505 459 491 477 510 464 443 451 483 469 388 410 504 458 490 476
509 463 498 444 452 484 470 530 457 489 475 508 462 494 480 513 500 540 529 456 488 474 507 461 493 479 512 499 539 528
455 487 473 506 460 492 478 511 465 538 527 447 361 537 526 468 576 517 531 544 557 570 523 578 482 580 516 503 543 556
569 522 536 549 562 575 591 542 555 568 521 535 548 561 538

Number of Characters in Compressed Text: 354

Compression Percentage: 70.50%
```

Figure 4.3.12: LZW Compression



```
The length of the total encoded text in Bytes is : 1890

The length of the total encoded text in Bits is : 15120

Number of status bits that are used here are : 1890



Details while Encoding the message to the image:
Original File: converted_image2.png
⠙ Encoding image......Image encoded and saved as converted_image2-encoded.png
⠹ Encoding image......
Comparison between Original and Encoded Image:

Total Pixels in the Image: 574574

Number of Pixels Changed: 5099

Percentage Change: 0.8874400860%
⠦ Encoding image......Image encoded and saved as converted_image2-encoded.png
The Encoded Image is:

Enter 'OPEN' to display the image:
|
```

Figure 4.3.13: Steganography Process

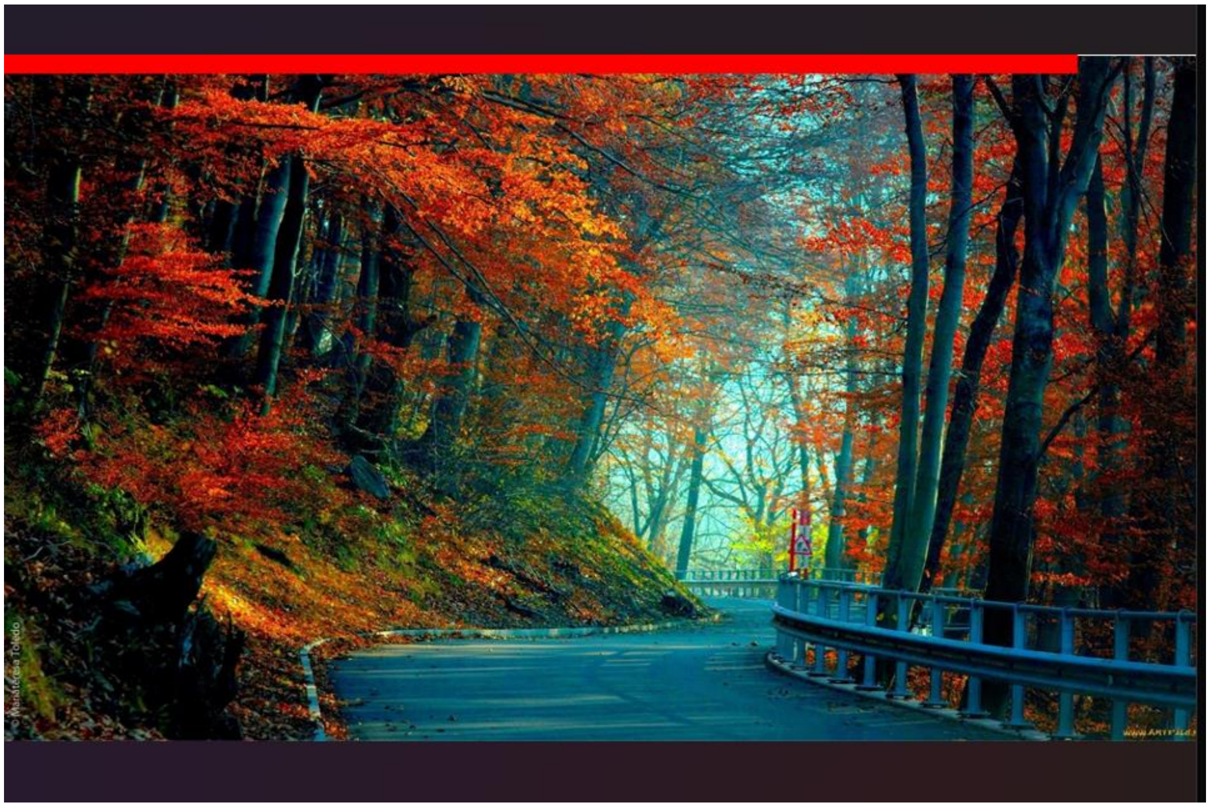
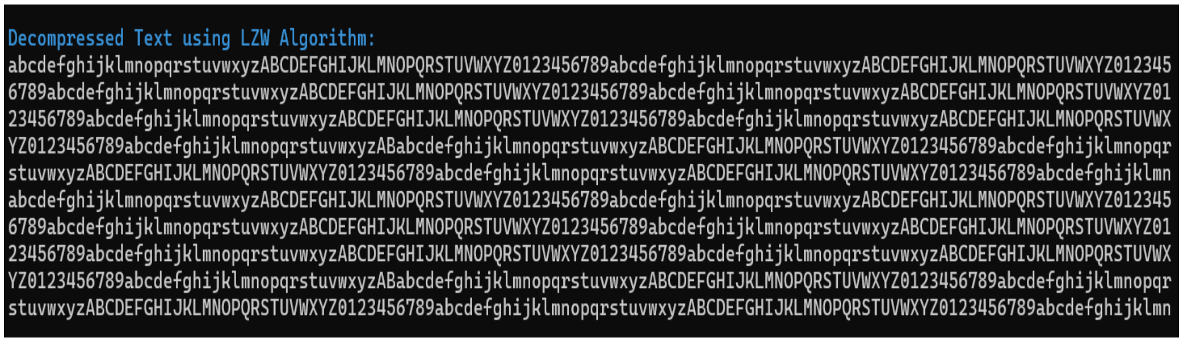


Figure 4.3.14: Stego Image



```
Decompressed Text using LZW Algorithm:
abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ012345
6789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ01
23456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWX
YZ0123456789abcdefghijklmnopqrstuvwxyzABabcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqr
stuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmn
abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ012345
6789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ01
23456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWX
YZ0123456789abcdefghijklmnopqrstuvwxyzABabcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqr
stuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmn
```

Figure 4.3.15: Decompressed Text

4.4 Dataset

8 Compressed Images:

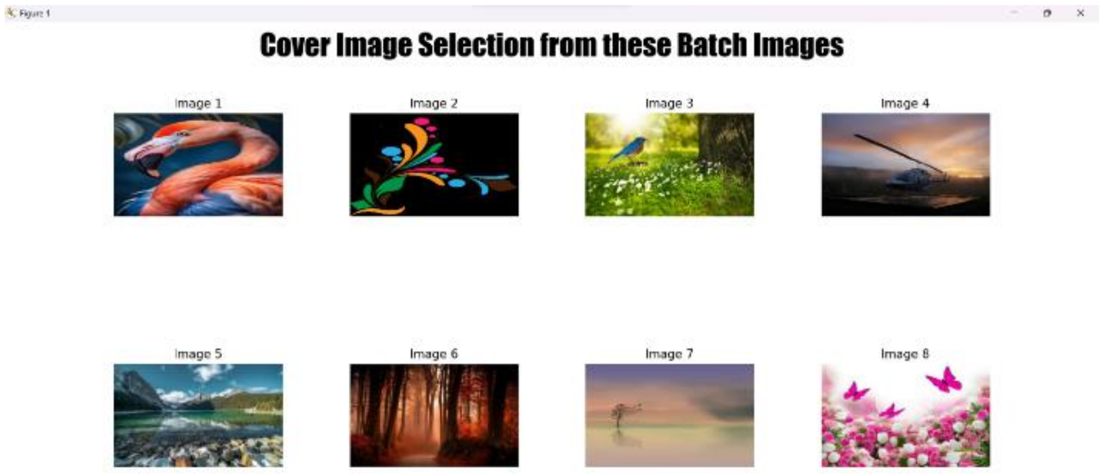


Figure 4.4.1: 8 Compressed PNG Images

8 Uncompressed Images:

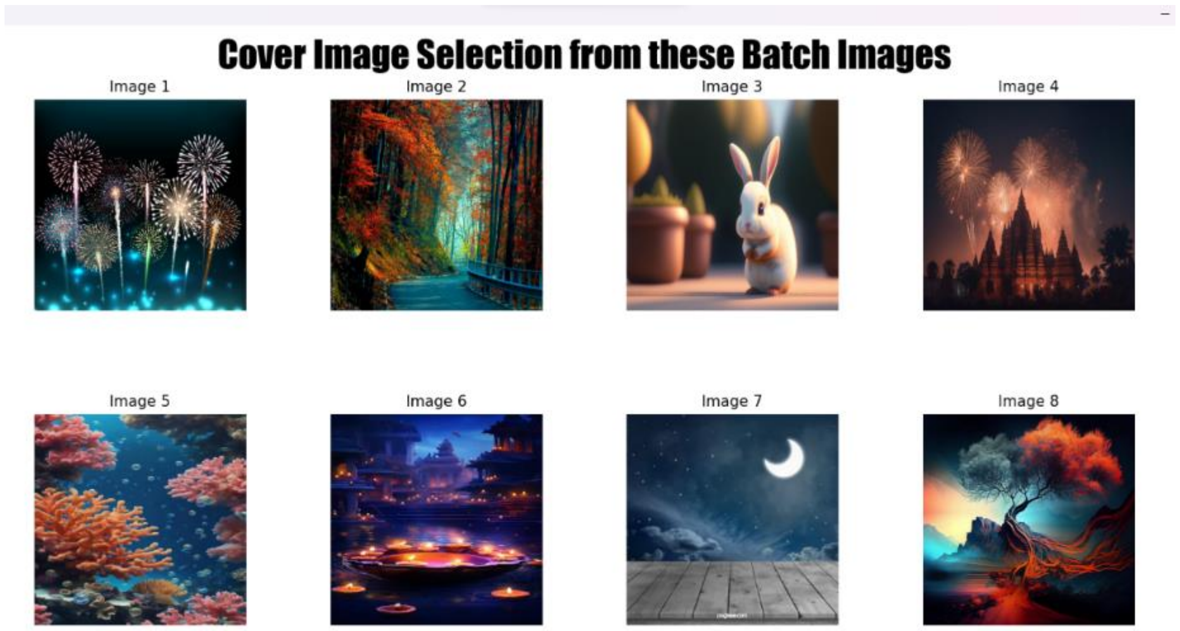
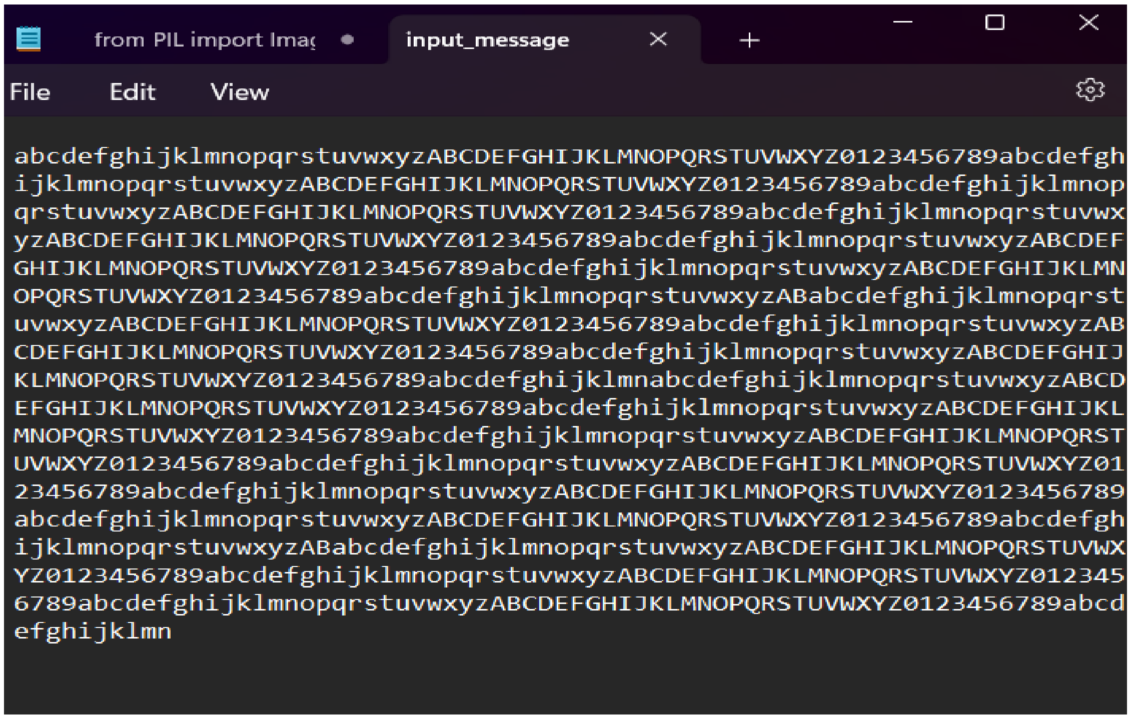


Figure 4.4.2: 8 Uncompressed PNG Images

Text File:



```
abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefgh
ijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnop
qrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwx
yzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEF
GHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMN
OPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABabcdefghijklmnopqrst
uvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzAB
CDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJ
KLMNOPQRSTUVWXYZ0123456789abcdefghijklmnabcdefghijklmnopqrstuvwxyzABCD
EFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKL
MNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRST
UVWXYZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ01
23456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789
abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefgh
ijklmnopqrstuvwxyzABabcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWX
YZ0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ012345
6789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcd
efghijklmn
```

Figure 4.4.3: Input Message Text File

4.5 Proposed Method

```
Algorithm
EnhancedSteganographywith_DualAuthentication_andCamelliaCipherAlgorithm
(images, message)
Input: Set of Images, Message
    # Stage 1: Image Selection and Preprocessing
    selectedImage = ImageSelectionAndPreprocessing(images)
    # Stage 2: Receiver Authentication
    authenticatedReceiver = AuthenticateReceiver()
    if authenticatedReceiver:
```

```
    # Stage 3: Compression and Encryption
        compressed_message = compress(Text)
        encryptedMessage = EncryptMessage(message)
# Stage 4: Steganography
        stegoImage = ApplySteganography(selectedImage, encryptedMessage)
 # Stage 5: Data Transmission
        TransmitStegoImage(stegoImage)
    else:
        DisplayAuthenticationFailureMessage()
End Algorithm
```

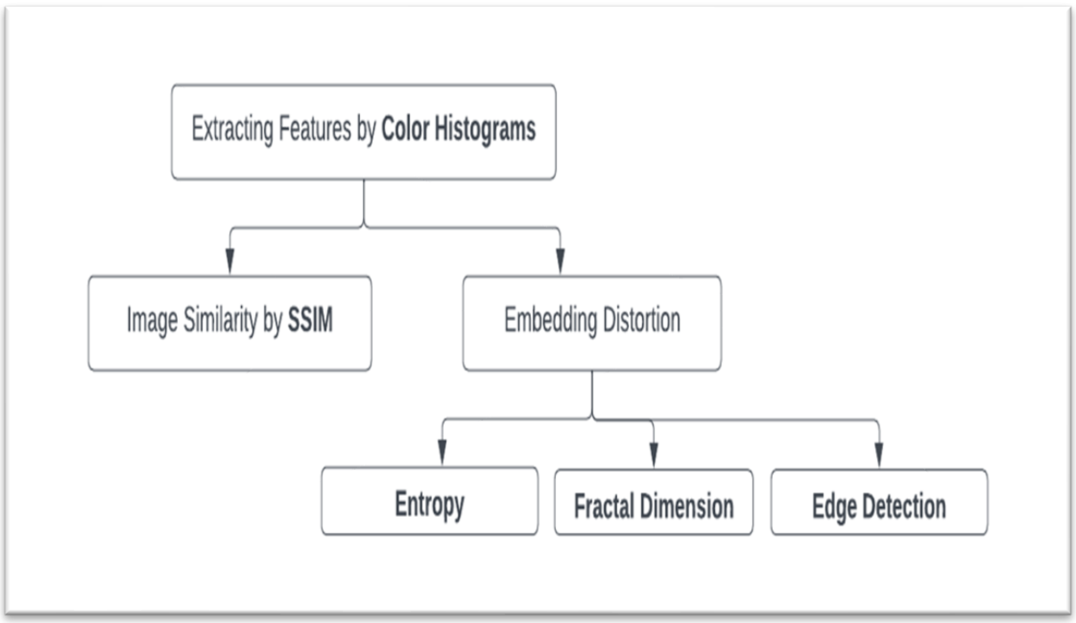


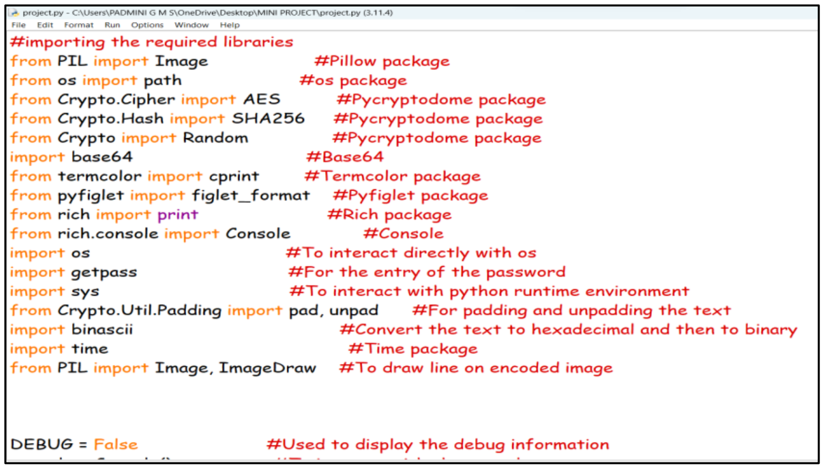
Figure 4.5.1: Proposed Method Illustration

5. EXPERIMENTAL SETUP

The environment used is **PYTHON IDLE & COMMAND PROMPT.**

5.1 Install Python IDLE – 3.11.4

To install Python IDLE 3.11.4, begin by visiting the official Python website and downloading the installer for your operating system. Once downloaded, run the installer and follow the on-screen instructions. During the installation process, ensure that the option to install IDLE is selected. After completing the installation, verify it by checking the installed Python version using the terminal or command prompt. Additionally, you can launch IDLE from your system's applications or by running it from the terminal/command prompt. If you're using Windows, consider setting the PATH environment variable to include Python's installation directory for easier access. Following these steps will successfully install Python IDLE 3.11.4 on your system.


project.py - C:\Users\PADMINI G M S\OneDrive\Desktop\MINI PROJECT\project.py (3.11.4)
File Edit Format Run Options Window Help

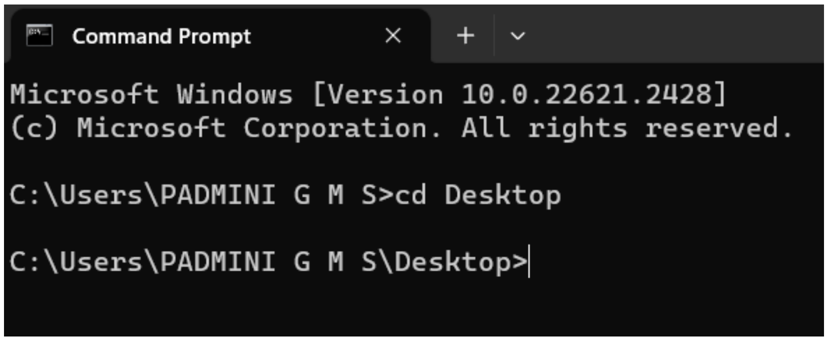
```
#importing the required libraries
from PIL import Image                  #Pillow package
from os import path                    #os package
from Crypto.Cipher import AES          #Pycryptodome package
from Crypto.Hash import SHA256         #Pycryptodome package
from Crypto import Random              #Pycryptodome package
import base64                          #Base64
from termcolor import cprint           #Termcolor package
from pyfiglet import figlet_format     #Pyfiglet package
from rich import print                 #Rich package
from rich.console import Console            #Console
import os                              #To interact directly with os
import getpass                         #For the entry of the password
import sys                             #To interact with python runtime environment
from Crypto.Util.Padding import pad, unpad     #For padding and unpadding the text
import binascii                        #Convert the text to hexadecimal and then to binary
import time                            #Time package
from PIL import Image, ImageDraw       #To draw line on encoded image



DEBUG = False                  #Used to display the debug information
```

Figure 5.1.1: Python IDLE

5.2 Connect with Command Prompt

To connect with IDLE using the Command Prompt, ensure that Python is installed on your system and its installation directory is added to the system's PATH environment variable. You can verify the Python installation by typing `python --version` in the Command Prompt, which should display the installed Python version. To open IDLE, simply type `idle` in the Command Prompt and press Enter. This command launches the IDLE Python

GUI editor, where you can write and execute Python code. If the `idle` command doesn't work, you may need to add Python's installation directory to the PATH variable manually or navigate to the Python installation directory in the Command Prompt. Once IDLE is open, you can run Python code directly within the IDLE editor and exit by closing the IDLE window or using the keyboard shortcut `Ctrl + D`.



```
Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\PADMINI G M S>cd Desktop

C:\Users\PADMINI G M S\Desktop>
```

Figure 5.2.1: Command Prompt

5.3 Libraries Used

5.3.1 PILLOW (PIL):

The PILLOW library, commonly known as PIL, is a powerful tool for image processing tasks in Python. It offers a comprehensive set of functionalities to handle various image formats, enabling developers to manipulate images with ease. From basic operations like resizing and cropping to advanced tasks such as filtering and blending, PILLOW provides the tools needed to enhance and modify images effectively. With support for a wide range of formats, including JPEG, PNG, and GIF, PILLOW is indispensable for applications spanning computer vision, digital art, web development, and scientific imaging.

5.3.2 OS:

The OS module in Python serves as a vital interface for interacting with the underlying operating system. It empowers developers to execute a wide array of system-related tasks seamlessly. By providing functions for file and directory manipulation, command execution, and process management, the OS module enables efficient system operations within Python scripts. Developers can leverage its capabilities to create, delete, move, and rename files and directories, streamlining file system interactions. Moreover, the module facilitates access to crucial system information, such as the current working directory, platform details, and environment variables.

5.3.3. CRYPTOGRAPHY:

The cryptography library in Python stands as a robust toolkit for ensuring the security of sensitive data through cryptographic techniques. Offering a rich set of functionalities, it empowers developers to implement encryption, decryption, digital signatures, and key management with ease. By integrating strong cryptographic algorithms like AES, RSA, and SHA, cryptography enables the creation of resilient communication protocols and data protection mechanisms. Moreover, the library upholds stringent security standards and practices to thwart potential cryptographic vulnerabilities, ensuring the integrity of encrypted data. With its extensive support for cryptographic primitives and protocols, cryptography facilitates the realization of end-to-end encryption, secure authentication, and data integrity verification in Python applications. Continuously maintained and well-documented, it prioritizes usability, performance, and compliance with industry standards, making it a preferred choice for cryptographic operations.

5.3.4 PYCRYPTODOME:

This library extends Python's built-in `hashlib` module, offering advanced cryptographic primitives like SHA-256, SHA-512, and MD5. It simplifies cryptographic operations, aiding in secure key generation and random number generation. Adhering to modern standards, PYCRYPTODOME ensures compatibility with industry best practices. Widely trusted, it finds applications in encryption, digital signatures, and secure communication protocols, contributing to robust security solutions.

5.3.5 BASE64:

This facilitates encoding and decoding binary data into ASCII characters, vital for scenarios where only text-based channels are available. Supporting standard and URL-safe variants, it's indispensable for transmitting binary data via email or HTTP. Frequently utilized in serialization and data interchange tasks, it simplifies representing binary data in human-readable formats, offering a straightforward interface and efficient implementation for Python applications.

5.3.6 TERMCOLOR:

The developers can enrich terminal output by adding colored text and background colors. Supporting a broad spectrum of ANSI color codes, it empowers customization of text appearance in command-line interfaces. Ideal for highlighting crucial information or providing visual cues in CLI applications, loggers, and debugging tools, TERMCOLOR enhances user experience across various terminal emulators and operating systems.

5.3.7 PYFIGLET:

This library is instrumental in creating ASCII art and stylized text headers with customizable fonts and widths. By converting plain text into visually appealing ASCII representations, PYFIGLET adds aesthetic value to console applications, banners, and graphical interfaces. Its user-friendly interface and extensive font selection make it a preferred choice for generating eye-catching textual content, enhancing the visual appeal and readability of Python projects.

5.3.8 SYS:

The SYS package serves as the gateway to Python's runtime environment, offering access to system-specific parameters and functions. It enables interaction with command-line arguments, standard input/output streams, and system-level configurations. Essential for platform-independent scripting and system administration tasks, SYS facilitates programmatic control over runtime behavior, error handling, and execution environment, empowering developers to build versatile and adaptable Python applications.

5.4 Parameters

5.4.1. MEAN SQUARE ERROR (MSE):

$$\boldsymbol{MSE = (\frac{1}{m*n}) * \Sigma(\Sigma(I(x,y) - K(x,y))\hat{}2)}$$

- **m** and **n** are the dimensions of the image (e.g., width and height).
- **I(x, y)** represents the pixel value of the original image at position (x, y).
- **K(x, y)** represents the pixel value of the reconstructed (or compressed) image at the same position.
- MSE measures the average squared difference between the pixel values of the original image and encoded image.
- MSE values close to zero suggest minimal distortion and high image fidelity.
- Mathematical Calculation: It computes the average of the squared pixel-wise differences between the corresponding pixels of the two images.
- Minimal Distortion: An MSE value close to zero indicates that there is minimal distortion or difference between the original and processed image.
- Image Fidelity: In the context of image processing, low MSE values suggest high image fidelity, meaning the processed image closely resembles the original, maintaining image quality and detail.
- Quality Assessment: MSE is a crucial tool for evaluating the effectiveness of image processing techniques. A lower MSE signifies that the process or encoding has preserved the original image content well.

5.4.2. PEAK SIGNAL-TO-NOISE RATIO (PSNR)

$$\boldsymbol{PSNR} = \frac{\mathbf{10} * \boldsymbol{log}\mathbf{10}((\boldsymbol{peak}\mathbf{\hat{}2})}{\boldsymbol{MSE}}$$

- PSNR is a measure of the quality of a encoded image compared to an original reference image.
- PSNR values above 30 dB are generally considered high and indicative of good image quality.
- The peak pixel value in the picture is represented by the variable "peak". It shows the highest possible value that a pixel in the picture may have. The highest pixel value for an 8-bit picture is 255 (28 - 1), hence "peak" would typically be 255 for normal 8-bit images.
- Measurement of Image Quality: PSNR is a widely used metric for assessing the quality of an image after it has undergone compression or other processing. It quantifies the fidelity of the processed image compared to the original, serving as a critical tool in image quality evaluation.
- Decibel Scale: PSNR is expressed in decibels (dB), which is a logarithmic scale. The decibel scale is particularly useful because it provides a convenient way to express large ranges of image quality. Higher PSNR values indicate better image quality

6. DISCUSSION OF RESULTS

To examine the distortion and noise in the same image of various sizes, we utilized the Mean Square Error and Peak Signal to Noise Ratio (PSNR). As can be seen, the PSNR decreases as the image size decreases and when the Encoded text size increases. But is still rather substantial, indicating that the cover image and the genuine image cannot be distinguished by the naked eye. The receiver was able to decode the image and obtain the cipher text after we successfully encrypted the message using Camellia Cipher and concealed it in the image using Steganography.

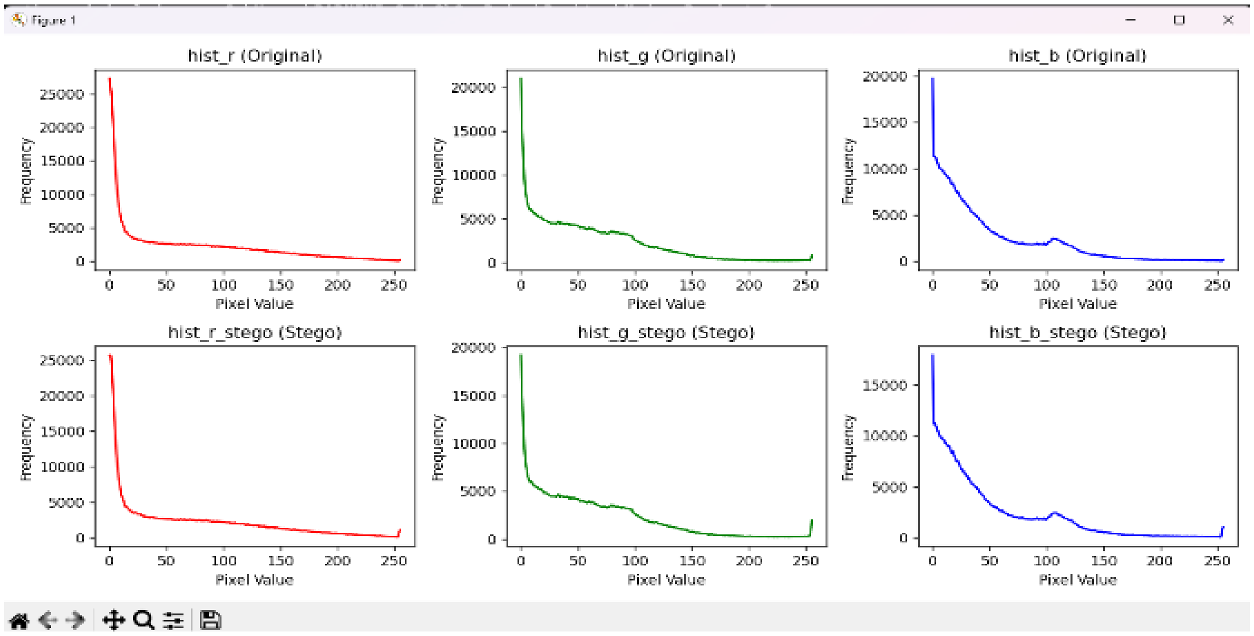
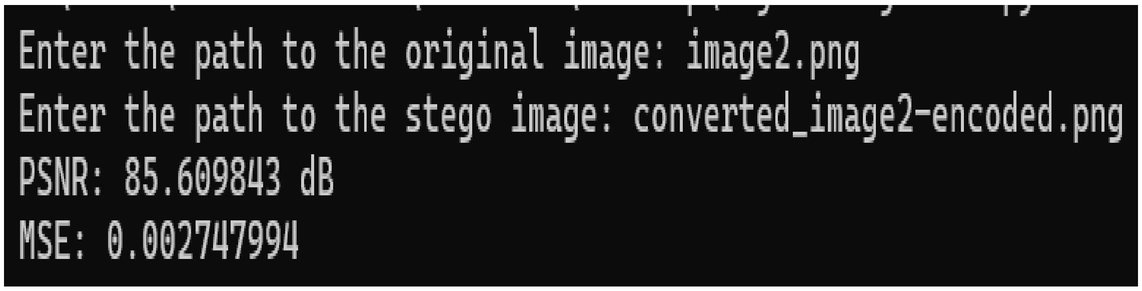


Figure 6.1: Histogram Representation of the Original and Encoded Image



```
Enter the path to the original image: image2.png
Enter the path to the stego image: converted_image2-encoded.png
PSNR: 85.609843 dB
MSE: 0.002747994
```

Figure 6.2: Calculation of PSNR & MSE Values

For the same image not using Cover Image selection model and using our proposed method. The findings are as follows:

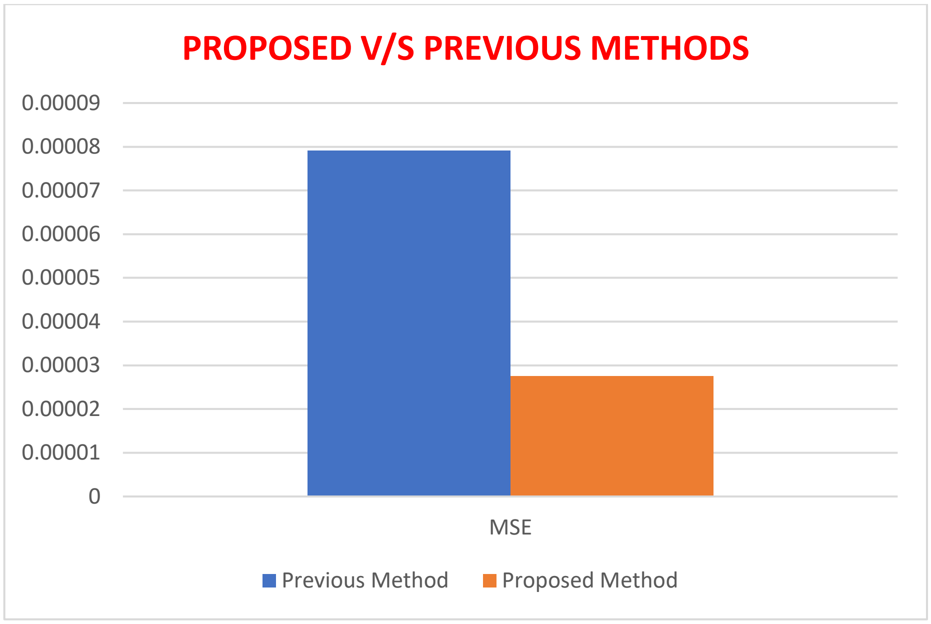


Figure 6.3 : MSE Value Observation between the Original & Encoded Image

In our investigation of distortion and noise within images of different sizes, we employed two key metrics: Mean Square Error (MSE) and Peak Signal-to-Noise Ratio (PSNR). These metrics allowed us to quantitatively assess the fidelity of the images and gauge the level of degradation introduced by resizing and embedding text. Notably, we observed a consistent trend wherein the PSNR exhibited a decrease as both the image size decreased and the size of the encoded text increased.

Despite these fluctuations, the PSNR values remained relatively high across all scenarios, indicating that the distortion introduced by resizing and embedding text was minimal and imperceptible to the human eye. This resilience of the PSNR underscores the robustness of the steganographic and cryptographic techniques employed in our methodology. Even with varying image sizes and increased payload sizes, the visual quality of the cover image remained uncompromised, preserving its integrity and concealing the embedded information effectively.

Furthermore, our experimental results demonstrated that the recipient successfully decoded the concealed information from the stego-image after encryption using the Camellia Cipher. This successful decoding process underscores the efficacy of our dual authentication approach, where the combination of steganography and cryptographic techniques ensures both the confidentiality and authenticity of the transmitted message. Overall, our findings validate the viability of our enhanced image steganography method for secure communication in digital environments, offering robust protection against unauthorized access and tampering of sensitive information.

The Findings we have by doing this project are:

- We found that "Enhanced Image Steganography using Dual Authentication and Camellia Cipher Encryption" is **secure** against Chosen-Plaintext attack, Differential Cryptanalysis, Known- Plaintext attacks.
- Through these Camellia Cipher Encryption and Decryption functions we are able to achieve Privacy.
- We can use other modes of image to convert into RGB such as CMYK, GREYSCALE, RGBA.
- We have observed that Uncompressed PNG Images are better choice for the Final Cover Image Selection compared to the Compressed PNG Images
- Dual authentication using Diffie-Hellman involves a cryptographic protocol where both communicating parties authenticate each other
- Using LZW text compression for the text that needs to be encoded in image steganography offers the advantage of reducing the size of the text payload, allowing for more efficient embedding within the image without significantly affecting the image's visual quality.

7. CONCLUSION

In conclusion, this project addresses a critical challenge in steganography by introducing an adaptive algorithm that effectively assesses content similarity across diverse image collections. Previous studies often struggled to provide universally effective methods, leading to suboptimal performance across different scenarios. By considering factors such as text and image compression, along with specific image parameters like fractal dimension, edge detection, SSIM calculation, and entropy, the proposed algorithm ensures comprehensive evaluation of content similarity, embedding distortion, and cover image selection, particularly focusing on PNG images. One significant contribution of this project is the reduction of image similarity within a batch, leading to a diverse range of coverings and thereby enhancing overall security. The primary objective of the project is to enhance the security, robustness, and effectiveness of the steganographic process. To achieve this, the project employs dual authentication processes and leverages the Camellia algorithm for advanced encryption, ensuring data privacy and integrity.

In conclusion, the project represents a significant advancement in the field of steganography by addressing the persistent challenge of content similarity assessment across diverse image collections. Unlike previous approaches that often struggled to provide universally effective methods, the project introduces an adaptive algorithm designed to comprehensively evaluate content similarity while considering various factors such as text and image compression, fractal dimension, edge detection, SSIM calculation, and entropy. By incorporating these diverse parameters, the algorithm ensures a thorough assessment of content similarity, embedding distortion, and cover image selection, with a particular emphasis on PNG images.

8. REFERENCES

[1] Wang, Z., Feng, G., Shen, L., & Zhang, X. (2023). Cover Selection For Steganography Using Image Similarity. IEEE Transactions On Dependable And Secure Computing, 20(3), 305-317.

[2] Liao, X., Yin, J., Chen, M., & Qin, Z. (2022). Adaptive payload distribution in multiple images steganography based on image texture features. IEEE Transactions on Dependable Secure Computing, 19(2), 897-911.

[3] Kalaichelvi, T., et al. (2020). Image Steganography Method to Achieve Confidentiality Using CAPTCHA for Authentication. Proceedings of the Fifth International Conference on Communication and Electronics Systems (ICCES 2020), IEEE Conference Record # 48766.

[4] Sachnev, V., Kim, H. J., & Zhang, R. (2009). Less detectable JPEG steganography method based on heuristic optimization and BCH syndrome coding. In Proc. 11th ACM Workshop Multimedia Security, pp. 131–140.

[5] Pramanik, S., Singh, R.P., & Ghosh, R. (n.d.). A new encrypted method in image steganography. Indonesian Journal of Electrical Engineering and Computer Science, 14(3).

[6] Jabbar Altaay, A. A., et al. (2012). An Introduction to Image Steganography Techniques. 2012 International Conference on Advanced Computer Science Applications and Technologies.

[7] Qiao, T., Luo, X., Wu, T., Xu, M., & Qian, Z. (2021). Adaptive steganalysis based on statistical model of quantized DCT coefficients for JPEG images. IEEE Transactions on Dependable Secure Computing, 18(6), 2736–2751.

[8] Lalengmawia, C., & Bhattacharya, A. (2016). Image Steganography using Advanced Encryption Standard for implantation of Audio/Video Data. 2016 Fifth International Conference On Recent Trends In Information Technology.

[9] Cica, Z. (2016). Pipelined Implementation of Camellia Encryption Algorithm. 2016 24th Telecommunications Forum (TELFOR), IEEE.

[10] Pabbi, A., et al. (2021). Implementation of Least Significant Bit Image Steganography with Advanced Encryption Standard. 2021 International Conference on Emerging Smart Computing and Informatics (ESCI) AISSMS Institute of Information Technology, Pune, India.

[11] Sheth, U., & Saxena, S. (2016). Image Steganography Using AES Encryption and Least Significant Nibble. International Conference on Communication and Signal Processing.

[12] Krishnaveni, N., & Periyasamy, S. (2018). A Novel and Innovative Approach for Image Steganography with Chaos. Indonesian Journal of Electrical Engineering and Computer Science, 11(1), 263–267.

[13] Fridrich, J., & Kodovsky, J. (2012). Rich models for steganalysis of digital images. IEEE Transactions on Information Forensics and Security, 7(3), 868–882.

[14] Kumar, A., & Pooja, K. (2010). Steganography - A Data Hiding Technique. International Journal of Computer Applications, 9(7), 19-23.

[15] Bhargava, S., & Mukhija, M. (2019). Hide Image And Text Using Lsb, Dwt And Rsa Based On Image Steganography. ICTACT Journal on Image & Video Processing, 9(3).

[16] Gutub, A., Al-Qahtani, A., & Tabakh, A. (2009). Triple-A: Secure Rgb Image Steganography Based On Randomization. 2009 IEEE/ACS International Conference on Computer Systems and Applications, 400-403.

[17] Srilakshmi, P., et al. (2018). Text Embedding Using Image Steganography In Spatial Domain. International Journal of Engineering & Technology, 7(3.6), 14.

[18] Rana, M. S., Sangwan, B. S., & Jangir, J. S. (2012). Art Of Hiding: An Introduction To Steganography. International Journal Of Engineering And Computer Science, 1(1), 11-22.

[19] Mathkour, H., Al-Sadoon, B., & Touir, A. (2008). A New Image Steganography Technique. 2008 4th International Conference on Wireless Communications, Networking and Mobile Computing, 1-4.

[20] Provos, N., & Honeyman, P. (2003). Hide and seek: an introduction to steganography. IEEE Security Privacy, 1(3), 32-44.