Coverless Multi-Image Steganography for RGB Images: Generative Adversarial Networks based Approach & Steganographic Key based Approach

J. K. K. K. Jagoda

2025

Coverless Multi-Image Steganography for RGB Images: Generative Adversarial Networks based Approach & Steganographic Key based Approach

J. K. Kimuthu Kisal Jagoda

Index No: 20000804

Supervisor: Dr. Kasun Gunawardana

Co-Supervisor: Mr. Tharindu Wijethilake

May 2025

Submitted in partial fulfillment of the requirements of the B.Sc in Computer Science Final Year Project (SCS4224)

Declaration

I certify that this dissertation does not incorporate, without acknowledgement, any material previously submitted for a degree or diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, be made available for photocopying and for interlibrary loans, and for the title and abstract to be made available to outside organizations.

Student Name : Kimuthu Kisal Jagoda
Registration Number : 2020/CS/080
Index Number : 20000804

[illegible]

30/06/2025

Signature & Date

This is to certify that this dissertation is based on the work of Mr. J. K. K. K. Jagoda under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Supervisor Name : Dr. Kasun Gunawardana

[illegible]

Signature & Date 30-June-2025

This is to certify that this dissertation is based on the work of Mr. J. K. K. K. Jagoda under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Co-Supervisor Name : Mr. Tharindu Wijethilake

[illegible] 2025-06-30

Signature & Date

Abstract

In recent years, Coverless Image Steganography (CIS) has emerged as a promising alternative to traditional steganographic techniques by eliminating the need for explicit data embedding and instead utilizing naturally occurring or generated images as carriers. However, current CIS methods often suffer from low hiding capacity, limited to concealing a single image, and offer insufficient robustness and security when handling multiple secret images simultaneously.

To address these limitations, we present a novel multi-image coverless steganographic framework tailored for RGB images, capable of securely transmitting and reconstructing up to four secret images using a single fused carrier image. Unlike conventional methods that depend on pre-existing cover images, our model generates an intermediate carrier image through a sophisticated pixel-level fusion technique that merges visual information from multiple RGB secret images while ensuring complete perceptual invisibility of the original content. This fusion process guarantees that the carrier image contains no visually identifiable traces of the concealed images, offering an inherently secure and stealthy representation.

To further strengthen concealment, the fused carrier undergoes multiple rounds of image scrambling transformations, making unauthorized decoding virtually difficult. We also integrate a robust steganographic key mechanism, restricting access and decoding capabilities exclusively to authorized receivers equipped with the correct key. This ensures that even if the carrier is intercepted, the hidden content remains secure and unrecoverable without the proper decryption context.

In the decoding phase, we propose a refinement algorithm designed to restore fine-grained details lost during the fusion and scrambling processes. This algorithm significantly enhances reconstruction fidelity and visual quality, even in the presence of adversarial conditions such as noise, tampering, or geometric transformations.

Comprehensive experiments on benchmark RGB image datasets validate the effectiveness of our proposed model, demonstrating substantially higher hiding capacity, minimal visual distortion, and resilience against statistical and perceptual steganalysis attacks. Compared to existing CIS approaches, our method offers a more secure, high-capacity, and visually imperceptible solution for multi-image coverless steganography, laying the foundation for next-generation covert communication systems.

Keywords : Multi-image Steganography, Coverless Image Steganography, Steganographic Key.

Acknowledgement

First, I would like to express my heartiest gratitude towards my supervisor Dr. Kasun Gunawardana and Mr. Tharindu Wijethilake for guiding me and motivating me towards this interesting research area.
I sincerely thank all the staff members and my colleagues at the University of Colombo School of Computing (UCSC) for their valuable advice, support, and teamwork, which have greatly helped in the success of this research.
I also dedicate this dissertation to my loving family, who have always been a great support throughout my life. Without them, my efforts would have meant nothing. Their love, encouragement, and patience motivate me to overcome challenges and achieve my goals.
Last, I would like to thank everyone who supported me, motivated me and guided me up to this accomplishment.

Preface

This document has been produced for the partial fulfilment of the requirements of the B.Sc. in Computer Science (Hons) Final Year Project in Computer Science (SCS4224). It presents a dissertation that introduces a novel technique built upon advanced pixel-level manipulation, structural transformations and steganographic key-based strategies to address the challenges of coverless multi-image steganography involving RGB secret images. The proposed method focuses on concealing multiple secret images within a single carrier image without the need for a predefined cover, ensuring high hiding capacity, visual feature concealment, and accurate reconstruction, thereby contributing a secure and efficient approach to the field of steganography.

The dissertation is structured to include the problem definition, literature review, methodological design, implementation details, experimental evaluation, critical analysis of results, limitations, future research directions, and a concluding summary of findings.

This dissertation represents original work carried out by myself under the guidance of my supervisor and co-supervisor. All content presented herein, unless explicitly cited or referenced, is the result of our own efforts, ideas, and implementations. Any material not specifically credited to external sources is considered to be our original contribution to the field.

Contents

List of Figures

List of Tables

Acronyms

AMPE Absulute Mean Pixel Error. 34

GAN Generative Adversarial Network. iv, 5–7, 14, 16, 18, 25, 46

LSB Least Significant Bit. v, vi, viii, 2, 3, 11, 42, 53–57, 62, 63, 66

MSB Most Significant Bit. 12, 42, 44, 53–55

MSE Mean Square Error. x, 34, 57, 89

PSNR Peak Signal-to-Noise Ratio. 34

RGB Red Green Blue. 4, 5, 7, 18, 25

RMSE Mean Square Error. viii, ix, 34, 92–100

1 Introduction

1.1 Background

Advancements in technology have facilitated the sharing of multimedia data such as images, audio, and video. As digital media is increasingly used to transmit confidential and sensitive information, the need for highly secure data transmission has become an essential for the sectors which handles critical information. However, threats like criminal wiretapping, interception, data modification, or data deletion during the transmission present significant risks to data security.

To protect such data, several information-hiding techniques have been developed, including cryptography, watermarking, obfuscation, and steganography. Cryptography employs mathematical methods and cryptographic keys to prevent unauthorized access by encrypting data. While this limits access to the data, it does not hide the presence of encrypted content. As a result, unauthorized parties may still detect its existence, potentially leading to attempts at data alteration, deletion or decoding.

Steganography, on the other hand, conceals secret information within another object, making both the content and the existence of the hidden data difficult to detect. Unlike cryptography, where the main goal is to obscure the content of secret data, objective of steganography is to conceal the presence of the hidden information entirely. This technique can be applied across various media, including text, images, audio, video, and network data, providing an additional layer of confidentiality (Duan et al. 2020).

The term "Steganography" originates from the Greek words "Steganos" and "Graph," which mean "hidden writing" or "secret writing" (Semilof & Clark 2023). It is a method that embeds secret data within another object in a way that is undetectable to those unaware of its presence (Tidmarsh 2023). Steganography began in ancient times as a method of hiding secret information in a way that made its very existence hard to detect, and over time, it has evolved into more advanced and sophisticated techniques. In ancient times, people used clever methods to hide secret messages. One such method was to make a person's head bald, tattoo the message on their scalp, and wait for the hair to grow back, hiding the tattooed information. The recipient would then make the head bald again to reveal the hidden message. Another technique involved using invisible ink that would only appear when the paper was heated with a flame. Some also carved messages onto wooden tablets and then covered them with wax, making the writing invisible. To read the message, the wax would be scraped off. These early steganographic techniques show how people have long found creative ways to communicate in secret. This is done by placing the hidden information inside natural looking object, which can be shared openly between people without raising suspicion.

A typical steganographic system includes two primary components: the "Encoder," which hides the secret data, and the "Decoder," which retrieves the hidden information.

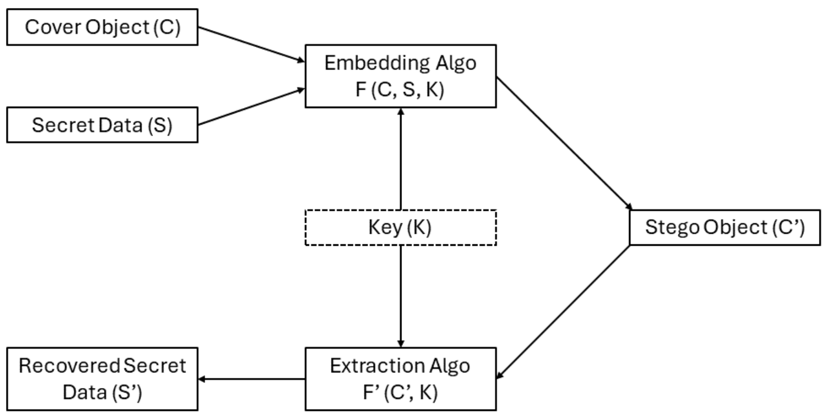

Cover Object (C)
Secret Data (S)
Embedding Algo F (C, S, K)
Key (K)
Stego Object (C')
Recovered Secret Data (S')
Extraction Algo F' (C', K)

Figure 1: The basic steganographic model (Choudary 2023)

Steganography has significantly advanced from its origins, evolving alongside the development of digital technology. What once was a simple method of concealing information has grown into a sophisticated field capable of hiding data across various types of digital media. With the increasing need for secure and undetectable communication, modern steganography now includes techniques for embedding information in text, images, audio, video, and even network traffic. These advancements have made steganography an essential part of digital security, allowing information to be shared secretly and safely in today's interconnected world.

- Text Steganography (Tidmarsh 2023)
  Text steganography is the idea of hiding a secret message inside another text message. A simple form of text steganography is using the first letter of each word to recover the secret message (Tidmarsh 2023). Linguistic Method, Random and Statistical Generation, Format Based Method, are some techniques used for text steganography (Choudary 2023).

- Image Steganography (Tidmarsh 2023)
  Image steganography hides secret information inside a digital image. This approach is based on the fact the small modifications to image features or noise that are very difficult to detect with human observation. The simplest method of image steganography is hiding a low-resolution image inside a high-resolution image by altering the LSBs of the original image (Tidmarsh 2023). Redundant Pattern Encoding, Coding and Cosine Transformation, Encrypt and Scatter, Masking and Filtering are some other common approaches which are used in image stegnography (Choudary 2023).

- Video Steganography (Tidmarsh 2023)
  Video steganography is an advanced version of image steganography that

can encrypt an entire video. As digital videos behave as a series of sequential images, every frame of video can encode as a collection of different images (Tidmarsh 2023). Video Steganography can be considered as a combination of audio steganography and image steganography (Choudary 2023).

- Audio Steganography (Tidmarsh 2023)
  Like in image steganography and video steganography, Audio files also can be hidden using steganography. The simplest from of audio steganography is named as "Backmasking" which the audio information that needed to be hide is plays backwords on a track. Similar to image steganography, more advanced techniques use LSBs of each bite in audio steganography (Tidmarsh 2023). Parity Encoding, Phase Coding, Spread Spectrum are some methods used in audio steganography (Choudary 2023).

- Network Steganography (Tidmarsh 2023)
  Network steganography is a smart digital steganography technique for hiding data inside network traffic. Data can be hidden in network packets such as TCP/IP headers or payloads. The sender may even send information considering the duration between delivering various packets (Tidmarsh 2023).

Steganography system can be considered as a function which takes a carrier object and the secret information as inputs and provides an embedded carrier as the output. In here, the embedded carrier object is revealed to the outside persons, and they are not aware about the existence of the secret information hidden inside the carrier object as it gives no clue. The hidden information can be recovered from the carrier object for the respective receivers (Felix et al. 2020). Steganography should satisfy following constraints.

I. Both the carrier image and the carrier object which embedded with secret information should be similar to evaluation via human vision (Felix et al. 2020).

II. The hidden information should be recoverable with the embedded carrier object and the recovered secret information should be clear and understandable (Felix et al. 2020).

III. Human vision should not be able to detect the existence of the hidden information which embedded in carrier object (Felix et al. 2020).

Image steganography receives significant attention in the field of steganography due to the widespread use of images in digital communication. With the internet and social media platforms heavily relying on image sharing, images have become an ideal carrier for embedding secret data. Their high redundancy and tolerance to slight changes make them suitable for concealing information without noticeably affecting visual quality. Additionally, images are easy to manipulate, transmit,

and store, which allows steganographic techniques to be applied in a seamless and undetectable manner. As a result, image steganography offers a practical and efficient way to protect sensitive information, making it one of the most researched and applied forms of steganography in the digital age. Image steganography can be understood through three main approaches.
The first is the method is cover modification, where an existing image is altered to embed secret information. This information can be either in image or text format, and the embedding is done by changing the pixel values of the cover image. While this method is quite common, it is highly vulnerable to steganalysis attacks, especially with the increasing use of deep learning techniques that can detect even minor changes in image content. Because of this, researchers have shown growing interest in coverless techniques that avoid the use of a traditional cover image.
The second method is known as cover selection, where a suitable image is selected from a large database to represent the secret content in a way that causes minimal distortion. However, this approach requires access to a vast dataset and involves difficulties in identifying the most appropriate image.
The third method is cover synthesis, which involves generating synthetic images from the secret information using generative models. Although this technique reduces the chances of detection, it demands higher computational resources and more time for both encoding and decoding. Among all these approaches, coverless methods have proven to be more resistant to steganalysis compared to traditional modification techniques.
With the help of generative models, coverless image steganography has shown promising potential. Motivated by these advancements, I hope to explore new directions and possibilities within the field of coverless multi image steganography.

1.2 Motivations

Coverless multi-image steganography is a promising field that aims to hide multiple secret images within a synthetic image without requiring a conventional cover image. This approach offers advantages over traditional steganography by reducing reliance on a pre-existing cover and minimizing the chance of detection. Recent work by Dharmawimala (2023) shows promising results in concealing grayscale secret images, achieving effective visual obscurity with minimal detectable features. However, the study also highlights a vulnerability in simpler black-and-white images, which tend to reveal more distinct features in carrier image compared to complex images which are rich in information, where visual details are more successfully obscured in carrier image. In both cases, reconstructed images appeared significantly blurrier than their originals, indicating a trade-off between concealment and image clarity. Inspired by this progress, Lakshan (2024) extended the method to RGB images, making strides toward hiding color-based secrets. However, the RGB adaptation reveals more visible features of the hidden images, leaving the presence of these secret images somewhat detectable to human observers. This limitation hinders the effectiveness of RGB steganography for applications where high security and subtlety are crucial.

The field of Generative Adversarial Network (GAN) has opened up new possibilities for enhancing coverless steganography, particularly by generating realistic synthetic images that could potentially conceal secret content more effectively. GAN-based methods have shown promise in concealing grayscale secret images by significantly reducing visible features and concealing the existence of the hidden content. Yet, applying GAN-based coverless steganography to RGB images remains an unsolved challenge. Current GAN methods struggle to achieve the same level of concealment with color images, where color and texture make visual detection easier. This gap in the literature suggests an opportunity for further research to explore how coverless multi image steganographic approaches can be optimized to enhance visual and contextual concealment for RGB images, creating synthetic images where the hidden content is both visually imperceptible and contextually undetectable.

This research is motivated by the potential to develop a nover method to improve coverless multi-image steganography for RGB images. The aim is to achieve enhanced concealment of the visual features of secret images and to mask the existence of these hidden images entirely. This advancement would not only make it more difficult for observers to visually detect hidden content but also obscure the contextual presence of concealed images. If successful, this method could pave the way for more secure and effective steganographic applications, offering a new level of concealment in color-based steganography that meets the rigorous demands of confidentiality and subtlety in information security.

1.3 Research Gap

Coverless multi-image steganography has utilizes concealing multiple secret images. The method proposed by Dharmawimala has achieved coverless multi-image steganography for Gray scale secret images revealing less visual features of secret images. With the inspiration by Dharmawimala's model, Lakshan's method was able to achieve multi-image steganography for RGB secret images into some context. The major drawback is Lakshan's method is, this method reveals a considerable amount of visual features of secret images.

The utilization of GAN provides some added advantages to the improvement in steganography. Although coverless multi-image steganography based on GANs offers advantages such as enhanced reconstruction quality and increased hiding capacity, it still falls short of fully achieving comprehensive visual feature concealment.

Transforming multiple secret images into a single carrier image demonstrates that high hiding capacity rates can be achieved. Using advanced fusion, diffusion, refinement algorithms we aim to implement a new method to achieve more enhanced security over coverless multi-image steganography for RGB images.

1.4 Problem Statement

Since Lakshan's proposed model was unable to achieve a better concealing of visual features of the secret images, we intend to achieve better hiding performance over the carrier image. Though the proposed model was successfully able to transform multiple images into a single image, human eye could easily detect the features of secret information. So that, it creates the necessity of implementing an enhanced method for achieving more security performance and hiding capability to address this drawbacks. The objective of this research is to develop a novel coverless multi-image steganographic approach aimed at enhancing visual feature concealment while ensuring secure and efficient image reconstruction.

1.5 Research Questions

Through the proposed study, we aim to provide solutions to the following research questions in order to accomplish the requirements of designing an efficient coverless multi-image steganography method, as stated above.

- **How can coverless multi-image steganography methods be improved to achieve higher hiding capacity and visual feature concealment while preserving reconstruction quality and security?**
  This research aims to enhance coverless multi-image steganography by using pixel-level manipulations and structural transformations to fuse multiple RGB secret images into a single intermediate image. The method aims to enhance hiding capacity while enabling accurate reconstruction of the original images with minimal visual degradation, all while preserving high visual quality and robustness against detection.

- **How can coverless multi image steganography methods be improved to reconstruct the original secret images with minimal loss of visual features?**
  The research focuses on developing techniques to hide the visual features of multiple secret images while embedding them into a single intermediate image. A critical aspect of this approach is ensuring that the embedded information can later be accurately reconstructed, enabling the recovery of each original secret image with minimal data loss. The research focuses on maintaining the integrity of visual features in the reconstructed images, preserving essential details as closely as possible to the original content.

- **How GAN based coverless multi image steganography methods can be improved to enhance the visual feature concealment and security performance in GAN based multi-image steganography?**

It emerges that GAN have been improving various steganographic techniques while optimizing visual feature concealment capabilities. So that, our objective is to achieve hiding multiple secret images concealing the features of secret images via GAN based coverless image steganography.

1.6 Project Scope

This proposed research will investigate on implementing a novel steganographic approach to enhance the hiding capabilities on RGB multi-image steganography. During this study, a new GAN based model will be designed and implemented which ensures visual feature concealment of the secret images in the coverless RGB multi image steganographic context. The GAN based model implementation will be limited only for two RGB secret images in this study as the primary purpose of this study is to achieve enhanced concealment of secret images. Due to the limitations in hardware resources and computational power the images will be limited for low resolution images of 256*256.

In addition to enhancing visual concealment, this research will also explore the development and implementation of another model focused on increasing the overall hiding capacity in the coverless RGB multi-image steganographic context based on advanced and novel fusion, diffusion and refinement techniques. This newly designed model will aim to embed a greater volume of secret data while continuing to maintain the concealment of visual features of the secret images. The objective is to investigate how advanced fusion and scrambling techniques can be utilized to support higher embedding capacity without compromising the ability to reconstruct the original secret images.

As the goals of this research are to achieve increased hiding capacity and advanced visual feature concealment, transmission losses during communication through external channels will not be addressed. The study assumes a lossless channel and concentrates solely on embedding the visual features of the secret images into the carrier image and accurately reconstructing the original images.

1.7 Aims and Objectives

This project aims to design a new coverless steganography method for RGB images, including an alternative scrambler algorithm to achieve more security while transferring secret images. In order to do so we achieve the following objectives.

I. To study existing multi-image steganographic models and analyze their strategies for visual feature concealment.

II. To explore alternative techniques, including image scrambling methods, to improve visual feature concealment.

III. To investigate embedding techniques for compressing multiple images into a single representation and examine refinement methods for reconstructing original images while preserving missing or degraded data.

IV. To design and implement a novel steganographic model that achieves enhanced visual feature concealment, higher hiding capacity, and accurate reconstruction.

V. To evaluate the performance of the implemented model in terms of concealment effectiveness, reconstruction quality, and overall robustness.

1.8 Significance Of The Project

While various steganographic models have been developed to hide information effectively, multi-image steganography continues to face challenges in achieving high accuracy in concealment as well as reconstruction. In particular, existing GAN-based coverless image steganography methods have yet to fully conceal the visual features of multiple RGB secret images. There is a clear need to enhance current techniques to ensure stronger visual feature concealment and more accurate reconstruction of the original images. This research aims to address these limitations by exploring suitable embedding techniques, an optimal scrambling algorithm and selecting a suitable GAN model for both image transformation and recovery.
A significant breakthrough in this area would offer considerable benefits to domains that require robust data security and effective information hiding. Furthermore, improvements based on the proposed method could lay the groundwork for more advanced coverless multi-image steganography models specifically tailored for RGB images.
Furthermore, this study assumes that no data loss or modifications occur during the communication process due to technical or network-related issues. By focusing on multi-image steganography, we aim to reduce potential errors per image, ensuring a more robust and reliable method for secure data embedding and retrieval. Since traditional steganographic approaches often suffer from distortions introduced by compression, transmission noise, or lossy channels, our proposed method seeks to mitigate such vulnerabilities.
By designing a model that inherently resists such distortions, we aim to enhance the reliability of steganographic communication. This is particularly crucial in applications requiring high precision, such as confidential data transfer, secure communications, and digital forensics. Future extensions of this research could explore adaptive error correction mechanisms to further strengthen the resilience of coverless multi-image steganographic techniques, ensuring seamless data reconstruction under real-world constraints.

1.9 Research Methodology

The primary goal of this research is to enhance both hiding capacity and security in multi-image steganography using two distinct methodologies: a traditional image fusion-based steganographic approach and a GAN-based coverless steganographic model. The first methodology focuses on compressing multiple secret images into a single intermediate image through pixel-wise fusion, LSB manipulation, and scrambling techniques. This approach achieves high hiding capacity without relying on a predefined cover image, ensuring minimal visual and reconstruction loss while concealing visual features effectively and achieves higher security by using steganographic key approach. The second methodology emphasizes improving security by leveraging advanced GAN-based steganography techniques to enhance concealment and resistance against detection.

The literature review played a crucial role in identifying existing image manipulation techniques and GAN-based image steganographic architectures, highlighting their strengths and limitations. This analysis provided a strong foundation for designing a suitable architecture that effectively supports multi-image steganography.

Following the architecture design phase, the data collection process was carried out, focusing on gathering a dataset comprising secret images and paint art images. The data preprocessing stage involved resizing images, generating fused images, and creating scrambled images to construct two entirely new datasets which is used for the training purposes.

The model implementation was then initiated, followed by setting up an appropriate training environment to ensure optimal performance. Experiments were conducted on existing techniques to benchmark their effectiveness before training the proposed model. The implemented model was tested under diverse conditions during training and evaluation to assess its hiding capacity and security performance.

To further enhance the model, iterative updates were applied based on evaluation results. The improved version of the model was retrained and re-evaluated to ensure better performance. The final results include a complete architectural overview, along with test visualizations and statistical evaluations demonstrating the model's effectiveness in securely concealing multiple secret images.

1.10 Outline of the Dissertation

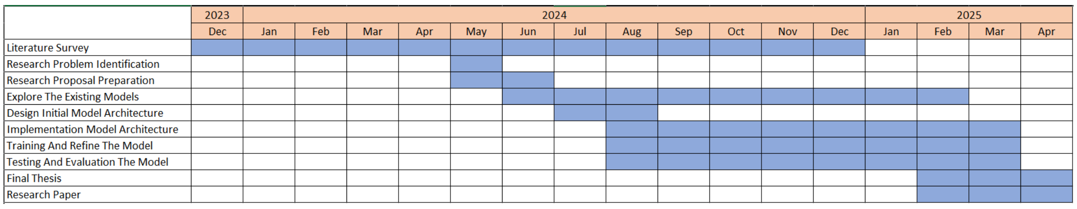
This dissertation presents a novel approaches to coverless multi-image steganography, leveraging Generative Adversarial Networks (GANs) alongside advanced image manipulation techniques and a key-based steganographic framework. The proposed methods are designed to address key challenges such as visual feature concealment, high embedding capacity, and robust reconstruction, without relying on predefined cover images.

The structure of the dissertation is organized as follows:

- **Chapter 1** provides an introduction and background to the study, including the problem statement, research questions, project scope, and the aims and objectives of the research.
- **Chapter 2** presents a comprehensive literature review, covering existing steganographic techniques and related work that form the foundation of this study.
- **Chapter 3** outlines the design of the proposed methodologies and architectural approaches used in the research.
- **Chapter 4** details the implementation process, highlighting key components and system integration.
- **Chapter 5** discusses the experiments conducted, along with the results and analysis of the model's performance.
- **Chapter 6** offers a critical evaluation of the results, discussing strengths, weaknesses, and practical implications.
- **Chapter 7** presents the conclusion with limitations and future possibilities.

1.11 Project Timeline

The gantt chart to illustrate in figure, indicates the timeline of this research project.



	2023	2024												2025			
	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr
Literature Survey	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦				
Research Problem Identification						🟦											
Research Proposal Preparation						🟦	🟦										
Explore The Existing Models							🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦		
Design Initial Model Architecture								🟦	🟦								
Implementation Model Architecture									🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	
Training And Refine The Model									🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	
Testing And Evaluation The Model									🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	
Final Thesis															🟦	🟦	🟦
Research Paper															🟦	🟦	🟦

Figure 2: Gantt Chart of the Research Project

1.12 Chapter Summary

This chapter laid out the foundational elements of the dissertation by providing the necessary background knowledge and motivation in sections 1.1 and 1.2. Research Gap and the Problem State is highlighted in sections 1.3 and 1.4. Section 1.5 presents the research questions. Section 1.6 presents the scope of the project. Aims and Objectives are highlighted in Section 1.7. Section 1.8 presents significance of the project. Methodology is presented in section 1.9. Furthermore in next sections, dissertation outline and project timeline are presented.

2 Literature Review

Steganography has roots that trace back to ancient times. Early examples include writing secret messages on wooden tablets and covering them with wax, using invisible ink for hidden communication, employing letter substitution techniques, and reading letters or words at fixed intervals. These methods all illustrate early forms of steganography (Dickson 2020).
As technology has advanced, the applications of steganography have broadened significantly across various fields, especially in image steganography due to the widespread use of digital images in communication, the high redundancy and capacity of image files, and the ease with which subtle modifications can be made without affecting visual quality.

2.1 Image Steganography

In image steganography, a cover image serves as the carrier for hidden information. The goal is to embed secret data within another image, known as the stego image, providing minimal indication to steganalysis tools about the presence of hidden information (Liu, Ke, Zhang, Lei, Li, Zhang & Yang 2020).
Image steganography is classified into three categories.

I. Cover Modification.
Cover Modification is a technique of altering the cover image in order to hide secret image. LSB substitution method is a more popular method for Cover Modification (Liu, Ke, Zhang, Lei, Li, Zhang & Yang 2020).

II. Cover Selection.
By mapping cover images and secret images, a cover image is chosen from a pool based on pre-defined criteria such as similarity to the original secret image. This technique minimizes perceptual changes and avoids suspicion while embedding the secret image into the cover image (Liu, Ke, Zhang, Lei, Li, Zhang & Yang 2020).

III. Cover Synthesis.
In cover synthesis image steganography, an entirely new image is generated which looks normal and statically similar to the expected cover image to embed the secret image (Liu, Ke, Zhang, Lei, Li, Zhang & Yang 2020).

2.1.1 Cover Modification based Image Steganography

Traditional image steganography primarily relies on cover modification techniques, where the original cover object is slightly altered to embed secret information. One of the most common methods involves modifying the Least Significant Bit (LSB) of the cover image, as these subtle changes are generally imperceptible to the human eye. This technique is based on the principle that small alterations to low-impact image features are difficult to detect through visual observation. A simple example of this approach is embedding a low-resolution secret image into a high-resolution cover image by manipulating its LSBs (Tidmarsh 2023).

These methods emerged alongside advancements in digital technology, with the LSB substitution algorithm becoming one of the most widely adopted techniques. In the LSB substitution approach, the least significant bits in each byte of an image are modified to hide secret data, causing minimal visual distortion. Inspired by this concept, Baluja (2017) introduced a novel technique capable of hiding a full image inside a single cover image. While modifications to the Most Significant Bit (MSB) can significantly alter an image's appearance, changes to the LSBs have a much subtler effect. As a result, LSB substitution enables the embedding of secret information with minimal disruption to the visual quality of the cover image, making it difficult for unintended observers to detect the presence of hidden data. The success of LSB substitution based methods in steganography lies in their ability to conceal information through minor, nearly undetectable modifications, offering a simple yet effective way to secure visual data.

However, while these traditional techniques were effective at obscuring hidden images from human detection, they are more susceptible to exposure by modern steganalysis tools. Advanced analytical methods, including statistical analysis and deep learning-based detection models, can now identify patterns and anomalies introduced by LSB embedding, compromising the secrecy of the hidden content. Additionally, traditional methods often lack robustness against common image processing operations such as compression, resizing, or format conversion, which can distort or destroy the hidden data (Tidmarsh 2023).

These limitations have led to the growing interest in more advanced and secure alternatives, such as coverless steganography and generative approaches using deep learning models. These newer techniques aim to overcome the vulnerabilities of traditional methods by generating carrier image in a way that inherently integrates the secret information, making detection significantly more difficult. As a result, the focus of modern research is shifting towards improving resilience, increasing capacity, and reducing detectability in steganographic systems.

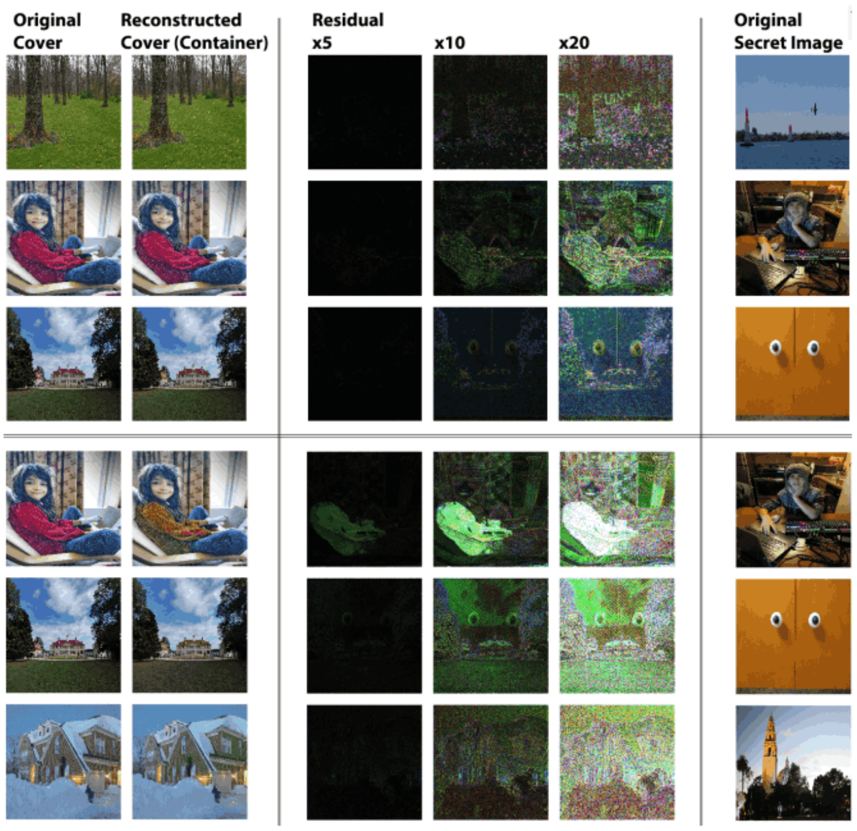


Figure 3: Baluja's Image to Image Hiding

2.1.2 Coverless Image Steganography

In modern steganography, more sophisticated techniques and algorithms are employed to ensure secure and reliable communication of confidential data. Steganographic models generally consist of two main components: the embedding algorithm, which hides the secret information, and the extracting algorithm, which retrieves it. During data transmission, attackers may attempt to use steganalysis tools to intercept, modify, or delete hidden data. This risk highlights the critical need for secure data transfer that provides no indication of the secret data's presence (Liu, Ke, Zhang, Lei, Li, Zhang & Yang 2020).

In response, coverless image steganography was developed, allowing new images to be generated directly from secret data without embedding it into a cover image, thereby reducing the risk of detection.

Coverless image steganography can be broadly classified into two approaches which are cover selection and cover synthesis.

Cover selection involves selecting a suitable image from a predefined database that represents the secret data. Since the chosen image is not altered in any way, this approach offers strong resistance to steganalysis attacks. However, it requires access to a large and diverse image database and selecting the most appropriate image to match the secret data can be difficult. This limitation makes the method less practical in situations where a perfect match is not available.

Cover synthesis uses generative models to create entirely new synthetic images based on the secret data. This technique offers a high level of flexibility and can provide better concealment of visual features since the image is generated specifically to carry the secret information. A major advantage of this method is that it does not depend on an external image database. However, it demands considerable computational resources and time, and the quality of the generative model directly affects the accuracy and reliability of both the concealment and reconstruction processes.
Both cover selection and cover synthesis are important directions in coverless image steganography. They contribute to strengthening data security and minimizing the chances of detection, though each comes with its own set of advantages and limitations.

2.2 Generative Adversarial Network (GAN)

A GAN is a technique that produces high-quality synthetic data that is challenging to distinguish from real data (Goodfellow et al. 2014). With its strong generative capabilities, natural-looking modification abilities, and high recovery accuracy, GANs have been introduced in the field of steganography. This approach has demonstrated improvements in achieving a higher hiding capacity, secure transmission, and accurate recovery of secret data. GANs represent a novel framework for evaluating generative models through computational means, allowing them to generate synthetic data that closely resembles real data, making it nearly indistinguishable (Goodfellow et al. 2014).

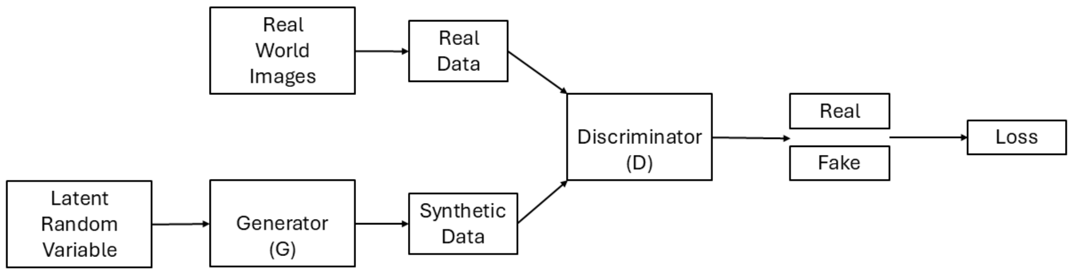


Figure 4: Structure of GAN (Goodfellow et al. 2014)

Figure 4 presented above illustrates the basic structure of a Generative Adversarial Network (GAN), showcasing the adversarial training process between the generator and the discriminator. This architecture represents a breakthrough in generative modeling, where two neural networks are engaged in a competitive framework to iteratively improve one another's performance.
GANs have gained significant traction in recent years, particularly in areas such as image synthesis, super-resolution, image-to-image translation, and data augmentation. In this model, the generator G attempts to generate data samples that mimic the distribution of real-world data, while the discriminator D aims to accurately

classify inputs as real or generated (fake). The success of this adversarial setup lies in its ability to force the generator to produce increasingly realistic outputs by continuously learning from the discriminator's feedback.
The core idea behind GANs can be mathematically formulated as a minimax optimization problem, represented by the following equation:

$$\min_G \max_D L(D, G) = Ex \sim p\text{data}(x)[\log D(x)] + E_{z \sim p_z(z)}[\log(1 - D(G(z)))] \quad (1)$$

Where:

- G is the **Generator**, which learns to generate data samples that resemble real data.
- D is the **Discriminator**, which learns to distinguish between real and generated (fake) samples.
- $x \sim p_{\text{data}}(x)$ denotes that x is a real data sample drawn from the true data distribution p_{data}.
- $z \sim p_z(z)$ denotes that z is a random noise vector sampled from a prior distribution p_z, typically a uniform or Gaussian distribution.
- $D(x)$ is the probability that the discriminator assigns to input x being real.
- $D(G(z))$ is the probability that the discriminator assigns to the generated sample $G(z)$ being real.

In this formulation, the discriminator seeks to maximize the probability of correctly identifying real and fake data, while the generator seeks to minimize this ability by generating increasingly realistic samples. This creates a dynamic equilibrium where both networks compete to outsmart each other.
The loss function of the discriminator is defined as:

$$L_D = -\frac{1}{m} \sum_{i=1}^{m} \left(\log(D(x^{(i)})) + \log(1 - D(G(z^{(i)}))) \right) \quad (2)$$

Where:

- L_D is the **loss of the discriminator**, measuring how well it distinguishes real from generated data.
- m is the **batch size**, i.e., the number of samples used in one training iteration.
- $x^{(i)}$ is the i^{th} real data sample drawn from the true data distribution.
- $z^{(i)}$ is the i^{th} noise vector sampled from a prior distribution (e.g., uniform or Gaussian).
- $G(z^{(i)})$ is the generated sample produced by the generator from noise $z^{(i)}$.

- $D(x^{(i)})$ is the discriminator's estimated probability that $x^{(i)}$ is real.
- $D(G(z^{(i)}))$ is the discriminator's estimated probability that the generated sample $G(z^{(i)})$ is real.

This loss encourages the discriminator to assign high probabilities to real samples and low probabilities to fake ones. It plays a crucial role in evaluating the quality of the samples generated by the generator.
The generator, on the other hand, attempts to maximize the probability that its outputs are classified as real by the discriminator. The generator loss function is given by:

$$L_G = -\frac{1}{m}\sum_{i=1}^{m}\log(D(G(z^{(i)}))) \tag{3}$$

Where:

- L_G is the **loss of the generator**, which measures how well the generator is able to fool the discriminator.
- m is the **batch size**, i.e., the number of noise samples used in one training iteration.
- $z^{(i)}$ is the i^{th} noise vector sampled from the prior noise distribution.
- $G(z^{(i)})$ is the synthetic data (fake sample) generated by the generator using $z^{(i)}$.
- $D(G(z^{(i)}))$ is the discriminator's estimated probability that the generated sample $G(z^{(i)})$ is real.

This loss ensures that the generator learns to produce outputs that are difficult for the discriminator to distinguish from real data.
Despite their powerful capabilities, GANs also come with several training challenges. These include maintaining a balance between the generator and the discriminator to avoid mode collapse or vanishing gradients. Synchronizing both components effectively is crucial for stable training. For instance, overtraining the generator without updating the discriminator can destabilize the entire training process.
Nevertheless, the ability to learn high-dimensional and complex distributions from random noise makes GANs a highly valuable tool in modern AI systems. With advancements in training techniques and architectural improvements, GANs continue to push the boundaries of what is achievable in generative modeling.

2.2.1 CycleGAN

CycleGan (Chu et al. 2017) is a ring network which has two symmetrical GAN models. In CycleGAN, the GAN models have two shared generators and one discriminator for each GAN model in a way that the CycleGAN model has total of two generators and two discriminators. The goal of CycleGAN is to transform images from one domain to another, with the ability to reconstruct the original images when transformed back. This model is particularly useful when paired datasets are not available, as it uses a cycle consistency loss to enforce the reversibility of the transformation.

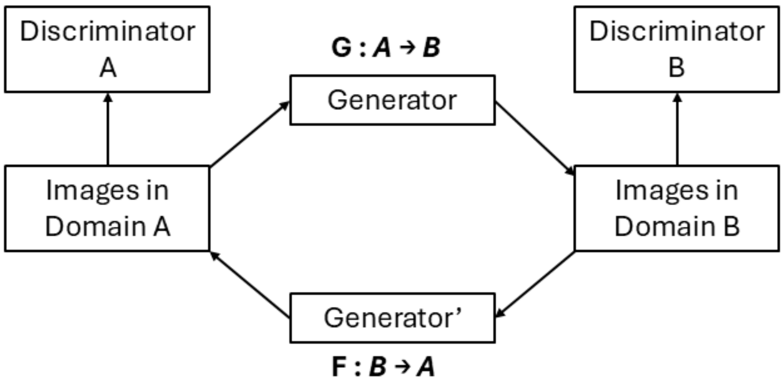
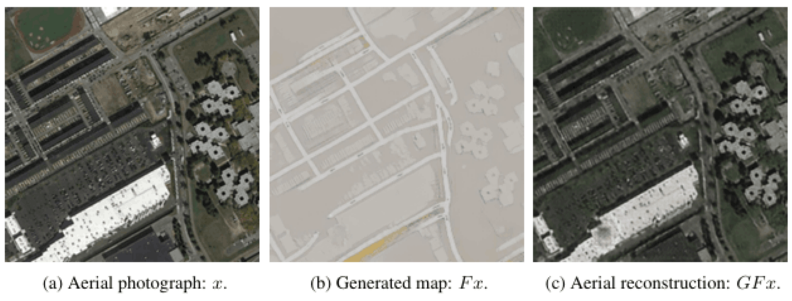


Figure 5: CycleGAN



(a) Aerial photograph: x.
(b) Generated map: Fx.
(c) Aerial reconstruction: GFx.

Figure 6: CycleGAN Transformation on different domains

CycleGAN's architecture offers significant benefits to image steganography by enabling reversible, unsupervised image transformations without requiring paired datasets. Its cycle consistency loss ensures that an image can be transformed into a different style (e.g., painting, map) and then accurately restored, which is ideal for securely embedding and later retrieving hidden information. The complex transformations generated by CycleGAN provide a more secure form of concealment than traditional pixel-based methods, as the hidden data becomes less detectable through basic steganalysis. Furthermore, CycleGAN's flexibility to adapt to diverse visual styles enables dynamic, multi-layered steganographic methods, allowing sensitive information to be embedded in ways that resemble various visual domains, such as artworks or satellite maps, effectively enhancing the security and versatility of data concealment.

2.2.2 Coverless Image Steganography using GAN

In traditional image steganography, cover images are modified during the embedding of secret data, which often results in distortions that make it possible for steganalysis tools to detect discrepancies. To address these limitations, GANs have brought significant advancements to coverless image steganography.
The CycleGAN model (Chu et al. 2017) is a technique designed for image-to-image transformations, and its reversible properties allow for the recovery of the original image.
The Cam-GAN model, introduced by Liu, Ma, Guo, Hou, Schaefer, Wang, Wang & Fang (2020), is capable of hiding a full-sized image in a coverless manner, achieving notable improvements in both concealment capacity and security.

2.2.3 Coverless Multi Image Steganography using GAN

Inspired by CycleGAN, Dharmawimala (2023) developed a novel steganographic method for grayscale images, achieving coverless multi-image steganography. With her approach, she successfully generated a carrier image that minimizes evidence of the presence of secret images and reveals fewer visual features of the hidden content (Dharmawimala 2023). Figure 7 illustrates the experimental results of Dharmawimala's method.
Lakshan (2024) expanded on Dharmawimala's grayscale model by adapting it for RGB images in coverless multi-image steganography. In his architecture, he successfully fused multiple RGB images and used the I^n2I model to recover the secret images from the fused output. However, the extracted features of the secret images were still visible in the stego image to a significant extent, revealing traces of the hidden content (Lakshan 2024). Figure 8 visualizes the experimental results from Lakshan's work. Given these advancements, the need for enhanced security in coverless multi-image steganography for RGB images has become evident. It is essential to transfer RGB secret images using more secure methods that not only ensure the concealment of the secret data but also effectively hide the visual features of the hidden images.

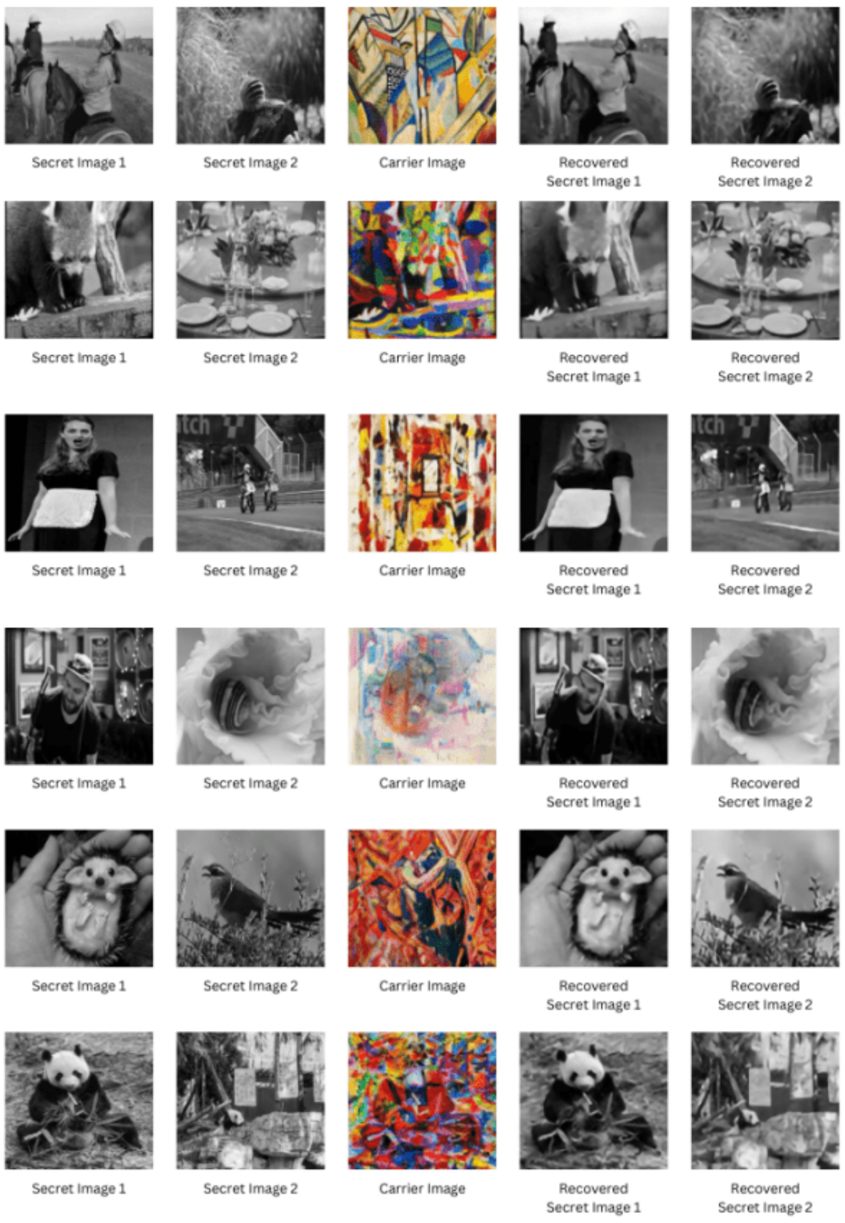


Figure 7: Dharmawimala's Experiment Results (Dharmawimala 2023)

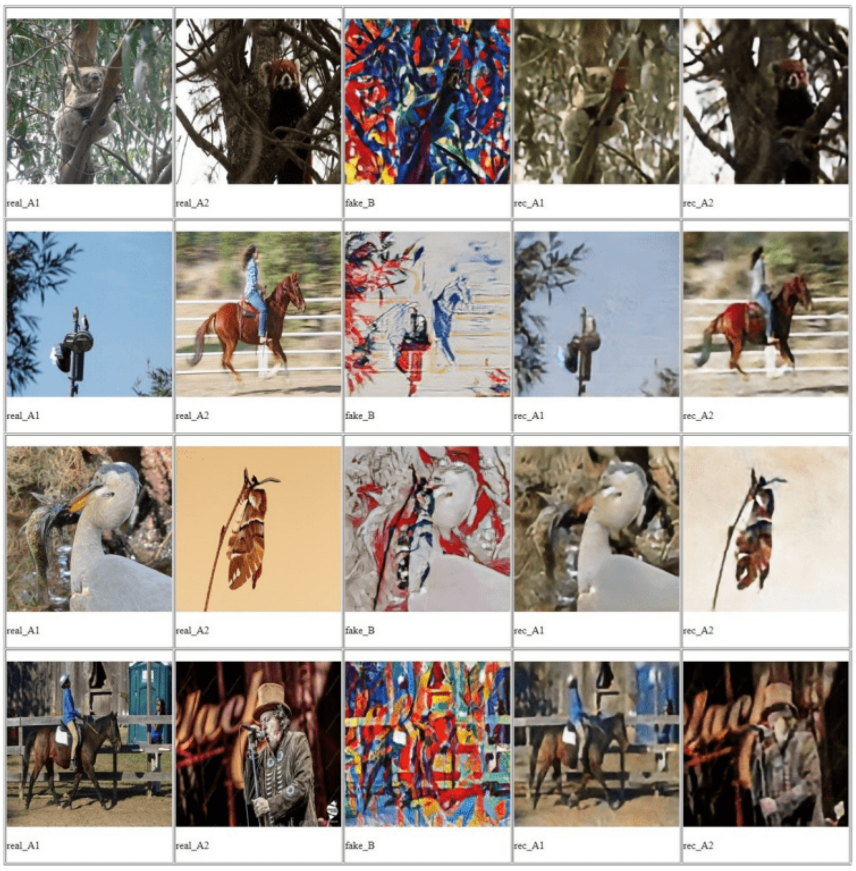


Figure 8: Lakshan's Experiment Results (Lakshan 2024)

2.3 Steganalysis

With the arrival of steganography to the information hiding domain, steganography detection technologies also have been emerged as well (Tidmarsh 2023). Steganalysis is the technique which is used to identify the existence of secret data hidden inside the carrier images. There are three types of steganalysis methods (Duan et al. 2020).

- Active Warden (Duan et al. 2020)
  The main goal is to discover the existence of the secret information and alter or erase the secret information.

- Passive Warden (Duan et al. 2020)
  The delivery of the data is authorized or prevented when the existence of the secret information is discovered. Data can not be destroyed or altered.

- Malicious Steganalysis (Duan et al. 2020)
  When the existence of the secret information is discovered, attempts are done to understand the steganographic method which is used to hide the secret information. Sometimes completely different cover images are generated to deceive both sender and the receiver.

As steganographic methods become more sophisticated, steganalysis has similarly evolved, particularly with the integration of deep learning techniques that offer improved detection capabilities.

Among the state-of-the-art steganalysis models, deep convolutional neural networks have shown promising results. GNCNN, proposed by (Qian et al. 2015) in 2015, was one of the earliest neural network-based models developed specifically for steganalysis. Following that, Xu-Net and Ye-Net introduced more advanced architectures with improved feature extraction layers to enhance classification performance. These models are typically trained on large-scale image datasets and are capable of identifying subtle statistical deviations that may result from data embedding, which are generally invisible to the human eye.

Grayscale images are frequently used during the training and evaluation of steganalysis models to simplify the analysis process by reducing computational complexity and removing the influence of color channels. Once trained, these models output probability vectors indicating whether an image is likely to be a stego image or a clean one.

Modern steganographic models are commonly evaluated using these deep learning-based steganalysis tools to measure their resilience against detection. Tools such as StegExpose, GNCNN, and Xu-Net are often employed to test and validate the robustness of newly proposed models.

2.4 Chapter Summary

This chapter presents a comprehensive Literature Review on Steganography. This provides an overview of the Image Steganography domain, outlining its fundamental requirements and categorizing different approaches to image steganography. This also discusses about the traditional approaches with potential advantages and vulnerabilities. Furthermore, it delves into Coverless Image Steganography, emphasizing its significance in eliminating the dependency on a cover image. The section also explores Coverless Image Steganography using Generative Adversarial Networks (GANs), discussing advancements in leveraging GAN-based models for secure data concealment. Finally, it examines Coverless Multi-Image Steganography using GANs, highlighting its potential in enhancing security and embedding capacity through multiple secret images.

3 Research Approach/ Methodology

The research mainly proposes two distinct methodologies, each highlighting its advantages and disadvantages. The first methodology focuses on increasing the hiding capacity of multi-image steganography by embedding multiple secret images into a single intermediate image, without relying on a predefined cover image, while also preserving the original quality during reconstruction. The second methodology aims to enhance the concealment of visual features and improve the security of hidden content using GAN-based techniques. By addressing both capacity and concealment, the study seeks to build robust and secure coverless steganographic frameworks and open new avenues for future research in GAN-based coverless multi-image steganography for RGB images. Each methodology is carefully designed and evaluated to assess its effectiveness in terms of visual imperceptibility, reconstruction accuracy, and resistance to steganalysis.

This research places a strong emphasis on increasing the hiding capacity of steganography in RGB images by embedding multiple secret images into a single intermediate image without altering a predefined cover. This is accomplished through advanced pixel-level manipulations and structural transformations that fuse information from the RGB channels of each secret image. A critical objective is to ensure the intermediate image retains sufficient data to enable accurate reconstruction and refinement of all original secret images. The study also focuses on improving structural encoding techniques to enhance reconstruction accuracy, increase data concealment, and boost resistance against steganalysis—ultimately contributing to the overall security and effectiveness of coverless steganographic systems.

In parallel with improving capacity and reconstruction, this research also prioritizes the concealment of visual features and the hiding of secret image presence by developing a GAN-based coverless multi-image steganography model specifically tailored for RGB images. Advanced generative techniques such as CycleGAN are explored to improve the model's transformation and concealment capabilities. To further enhance visual obfuscation, scrambling algorithms such as Arnold Scrambling and Reverse Arnold Scrambling are applied strategically. These techniques ensure that embedded features remain visually imperceptible, making it difficult for unauthorized observers to detect the presence of hidden content.

3.1 Proposed Methodology - 1 (Steganographic Key based coverless multi image steganography)

The literature review explores traditional cover modification techniques that manipulate pixel values, particularly the least significant bits of a cover image, to embed secret image information. Inspired by these techniques, this study proposes a novel LSB manipulation method to fuse two secret images into a single intermediate image without relying on a predefined cover. This approach removes the dependency on an external cover while preserving crucial information needed for accurate recovery and refinement of the original secret images. In addition, other compatible techniques are integrated with this method, enabling the full architecture to effectively compress four images into a single intermediate image. This intermediate image reveals no visual clues about the secret images and allows for their reconstruction with minimal visual distortion and low reconstruction loss.

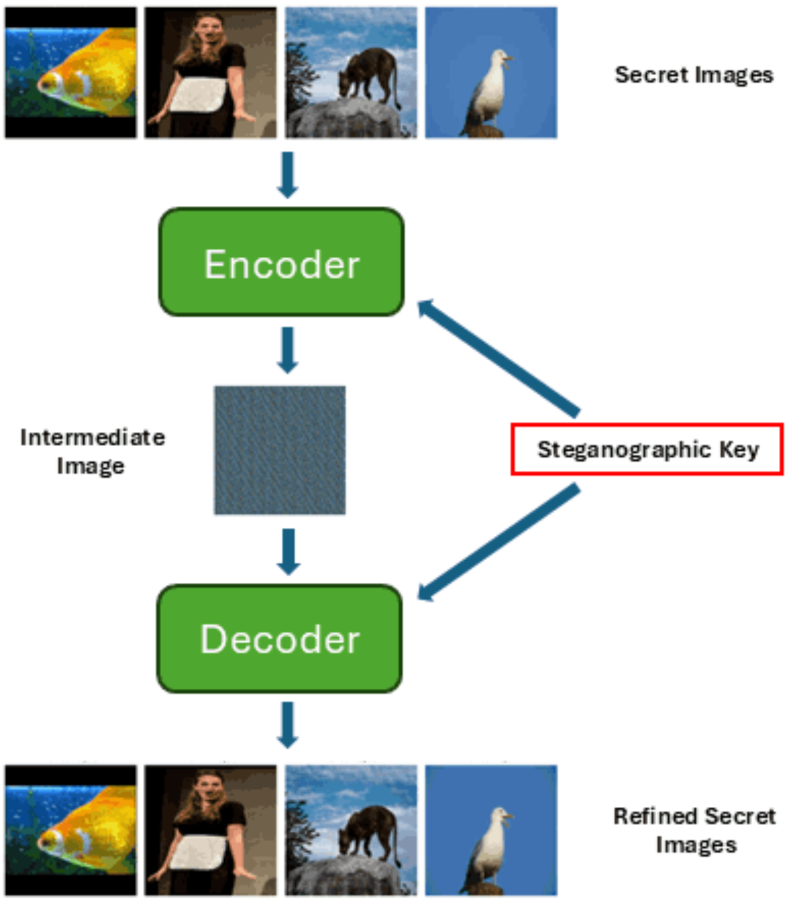


Figure 9: Proposed Methodology of Steganographic key based coverless multi image steganography

The proposed methodology consists of two key phases: Image Fusion and Image Scrambling. Image Fusion plays a crucial role in compressing multiple secret images into a single intermediate representation while preserving essential details required for accurate reconstruction. On the other hand, Image Scrambling enhances security by effectively concealing the visual features of the fused image, making it difficult to interpret or detect hidden content through visual inspection. Figure 9 illustrates the overall architecture.

3.1.1 Image Fusion

Image Fusion is the initial and fundamental phase of the proposed methodology, responsible for compressing multiple secret images into a single intermediate image. This process is designed to retain essential visual and structural information from all input images, enabling accurate and high-quality reconstruction later. By applying specialized fusion algorithms, the method ensures that key features from each secret image are integrated meaningfully into one unified representation. The fusion technique not only maximizes the hiding capacity by encoding multiple images into one but also lays the groundwork for further processing without compromising the integrity of the original data.

3.1.2 Image Scrambling

Image Scrambling aims at enhancing the security of the steganographic system. This phase involves transforming the fused image in a way that conceals its visual content, making it unintelligible to both human observers and automated analysis tools. By applying multiple rounds of scrambling using steganographic keys, the method ensures that the embedded information is not easily traceable or reconstructable without the proper decoding mechanisms. This significantly increases the robustness of the system against steganalysis and unauthorized access, contributing to the stealth and security of the hidden communication process.
To transform multiple images into a single image while preserving the essential details required for accurate reconstruction and achieving effective visual feature concealment, the proposed methodology applies multiple fusion and scrambling phases simultaneously. This integrated approach ensures that the critical visual features from each secret image are meaningfully embedded into a unified intermediate representation, while the scrambling phases obfuscate the visual patterns to enhance security. By combining these processes in parallel, the method achieves high hiding capacity, maintains reconstruction fidelity, and significantly reduces the visibility of embedded features, thereby strengthening the stego image against unauthorized interpretation and detection.
To further strengthen the security of the proposed system, a steganographic key is introduced as an integral part of the scrambling process. This security key governs the behavior and sequence of multiple scrambling operations applied to the fused image. By utilizing the key to dynamically control scrambling parameters, such as the number of iterations and the scrambling pattern at each phase, the process introduces a high level of complexity and unpredictability. As a result, the scrambled image becomes significantly more resistant to visual analysis or reverse engineering attempts. Without access to the correct security key, it becomes nearly impossible to accurately decode or regenerate the original secret images, thereby enhancing the robustness of the steganographic method against unauthorized access and ensuring secure transmission of hidden content.

3.2 Proposed Methodology - 2 (GAN based coverless multi image steganography)

Hense, the research focuses on improving the concealment of visual features and hide the presence of secret images using GAN-based coverless multi-image steganography for RGB images, this study aims to leverage advanced GAN-based techniques to develop an innovative coverless multi-image steganographic model tailored for RGB secret images. To achieve multi-image steganography, several techniques are examined and selected suitable techniques to fuse multiple images into a single image, while recovering the original images from the compressed image with minimum visual loss. In order to ensure further concealment of visual features of secret images, I explore the applicability of more advanced GAN techniques such as CycleGAN. And additional algorthms such as Arnold Scrambling and Reverse Arnold Scrambling are explored and applied in a suitable way to achieve enhanced feature concealment.

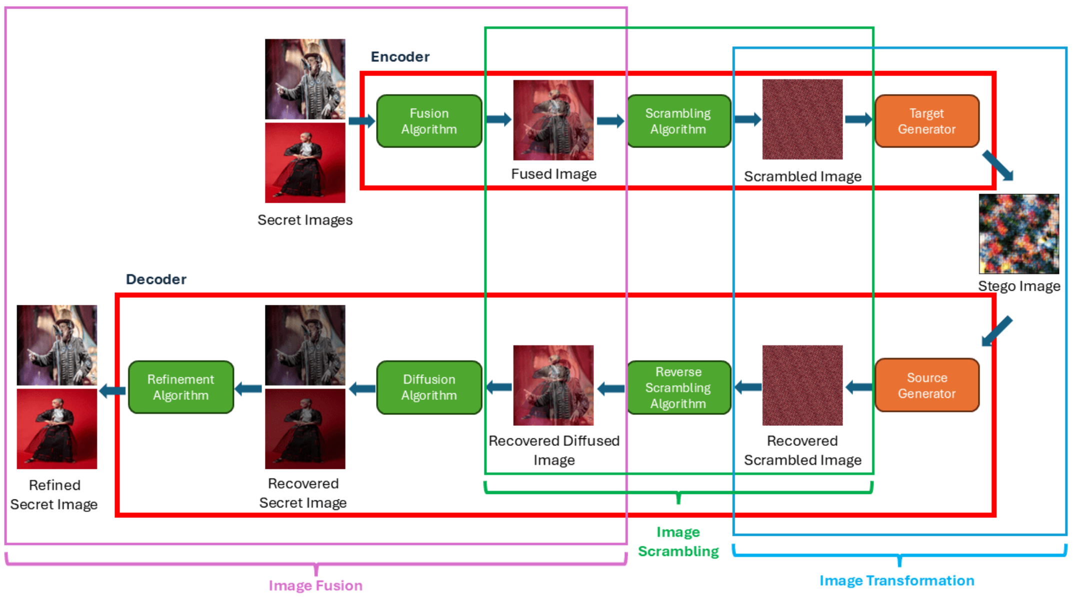


Figure 10: Proposed Methodology of GAN based coverless multi image steganography

The literature review explores current GAN-based coverless image steganographic architectures, highlighting their strengths and limitations. This review will establish a solid foundation for designing an effective architecture to implement GAN-based coverless multi-image steganography.
The proposed methodology consists of three key phases: Image Fusion, Image Scrambling, and Image Transformation. Each phase plays a crucial role in compressing multiple images into a single representation, concealing visual features, and generating synthetic images while maintaining the ability to reconstruct the original inputs with minimal information loss. The overall architecture is illus-

trated in Figure 10.

3.2.1 Image Fusion

The first phase involves compressing multiple secret images into a single intermediate image. The objective of this step is to effectively merge the visual information while preserving essential details for later recovery. The compressed image should allow the retrieval of the original secret images with minimal feature loss. To further enhance the reconstructed images, a refinement process is applied, improving their visual quality and ensuring a high level of similarity to the original inputs.

3.2.2 Image Scrambling

Once the intermediate image is obtained, a scrambling algorithm is applied to obscure visual features and enhance security. This scrambling step ensures that unauthorized access to the compressed image does not reveal the underlying secret images. The scrambling process is designed to be reversible, allowing the recovery of the original intermediate image through a corresponding reverse scrambling transformation. This phase enhances feature concealment, making it more challenging for adversaries to interpret the content.

3.2.3 Image Transformation

In the final phase, a Generative Adversarial Network (GAN) is employed to transform the scrambled image into a synthetic representation, referred to as the stego image. This transformation further obfuscates the original visual features while ensuring the image remains visually realistic. Additionally, another GAN-based generator is utilized to reconstruct the scrambled intermediate image, reversing the transformation and facilitating the recovery of the original compressed representation.

The proposed methodology provides a structured approach to secure multi-image steganography by integrating image fusion, scrambling, and transformation. By leveraging deep learning and scrambling techniques, the approach ensures high security, and effective concealment of secret images.

3.3 Training Architecture in GAN based coverless multi image steganography

The training process for the proposed multi-image steganography model integrates three primary phases: Image Fusion, Image Scrambling, and Image Transformation.

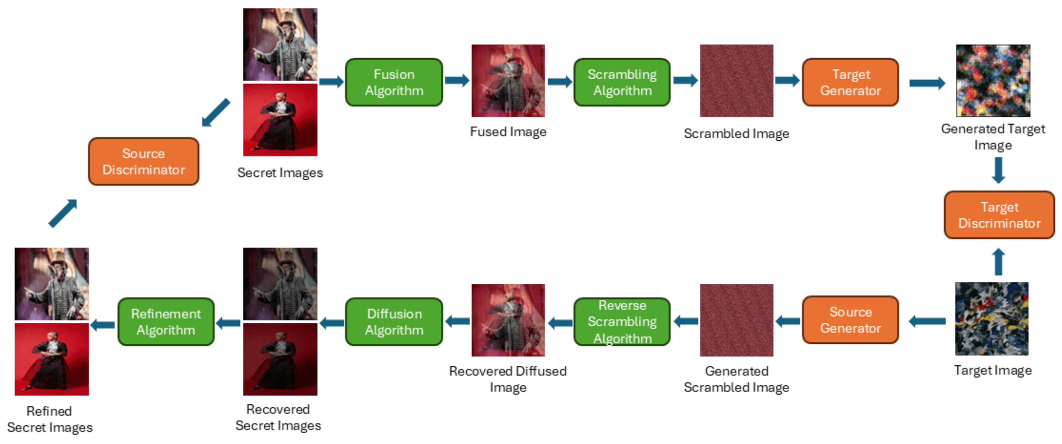


Figure 11: Training Architecture of GAN based coverless multi image steganography

Initially, multiple secret images are fused into a single fused representation using a pixel-wise fusion technique as described in Chapter 4.4.1. This fusion ensures that essential visual features from all input images are retained while forming an intermediate compressed image, reducing distortions and preserving important details necessary for later recovery.
Once the fusion is complete, the resulting intermediate image undergoes Arnold transformation as described in Chapter 4.4.3, a widely used scrambling technique that alters the spatial arrangement of pixels in a chaotic yet reversible manner. This step enhances security by obscuring the visual content, ensuring that unauthorized access to the scrambled image does not reveal the underlying secret information. The scrambling process is carefully designed to be reversible, enabling the retrieval of the fused image in subsequent steps.
Following the scrambling phase, a CycleGAN-based transformation as presented in Chapter 2.2.1, is employed to convert the scrambled image into a synthetic artistic representation, such as a painted artwork. The CycleGAN model plays a crucial role in learning the bidirectional transformation between the scrambled fused images and their corresponding paint-art representations. During training, the generator learns to create visually plausible paint-style transformations while another generator reconstructs the scrambled fused image from the paint-art domain. This transformation not only enhances concealment but also provides an additional layer of security by disguising the true nature of the embedded information.
After the transformation, the reconstructed scrambled fused image is processed through a Reverse Arnold transformation (4.4.3) to recover the original intermediate representation. This recovered image is then diffused back into two separate secret images, ensuring that the original information is fully restored. To refine the reconstructed images further, the four-nearest neighbor refinement (4.4.1) technique is applied, helping to correct any minor distortions that may have arisen during the transformation process.

To optimize this entire training framework, the model employs two generators as described in Chapter 4.4.4 and three discriminators as described in Chapter 4.4.5. The generators focus on learning the transformation between different domains, while the discriminators work to distinguish between real and generated images, ensuring high-quality outputs. By iteratively refining these transformations through adversarial training, the model tries to achieve a balance between secured feature concealment and reconstruction accuracy, making it highly effective for multi-image steganography applications.

3.4 Loss Functions Defined in Training Network in GAN based coverless multi image steganography

The training of the proposed multi-image steganographic model is guided by multiple loss functions, ensuring that the model can effectively encode secret images into target images while preserving reconstruction quality. The key loss functions include the *Adversarial Loss*, *Cycle Consistency Loss*, *Identity Loss* and *Intermediate Loss* which are described below.

3.4.1 Adversarial Loss

Adversarial loss is a fundamental component of the Generative Adversarial Network (GAN) framework. It helps in optimizing the encoder to generate realistic target images that are indistinguishable from real target images. The adversarial training consists of two discriminators:

- **Source Discriminator** (D_S): Evaluates whether the reconstructed secret images are authentic or synthesized.
- **Target Discriminator** (D_T): Distinguishes between the generated target images and real target images.

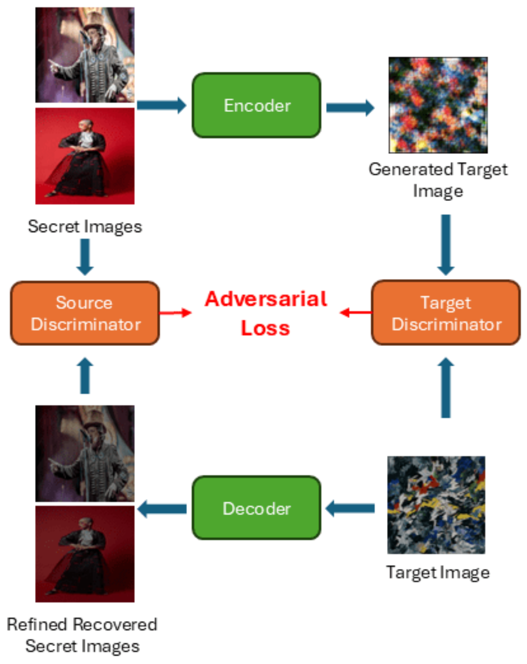


Figure 12: Adversarial Loss

This loss ensures that the generated target images are realistic and visually consistent with real target images. The encoder is optimized to produce images that can successfully fool the target discriminator, improving the quality of the generated outputs.

3.4.2 Cycle Consistency Loss

Cycle consistency loss ensures that the transformations between the secret and target images are reversible, preserving the structural integrity of both the secret and target images. It consists of two components:

- **Cycle Source Loss**: Ensures that the secret images can be accurately reconstructed after encoding and decoding.
- **Cycle Target Loss**: Ensures that the target images maintain consistency when transformed and then reconstructed.

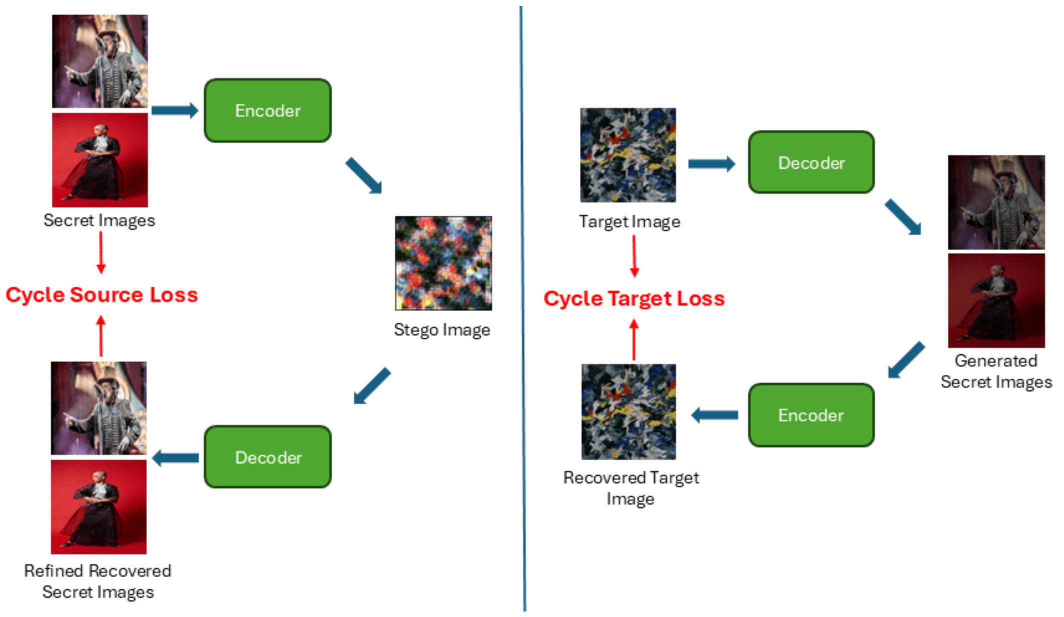


Figure 13: Cycle Consistency Loss

By minimizing the total loss, the model is optimized to generate realistic target images while ensuring accurate reconstruction of the original secret images.

3.4.3 Identity Loss

Identity loss is used to enforce that the generator preserves certain image characteristics when translating images that belong to the target domain. It ensures that if a target image is passed through the generator, it should remain unchanged, thereby improving stability in training.

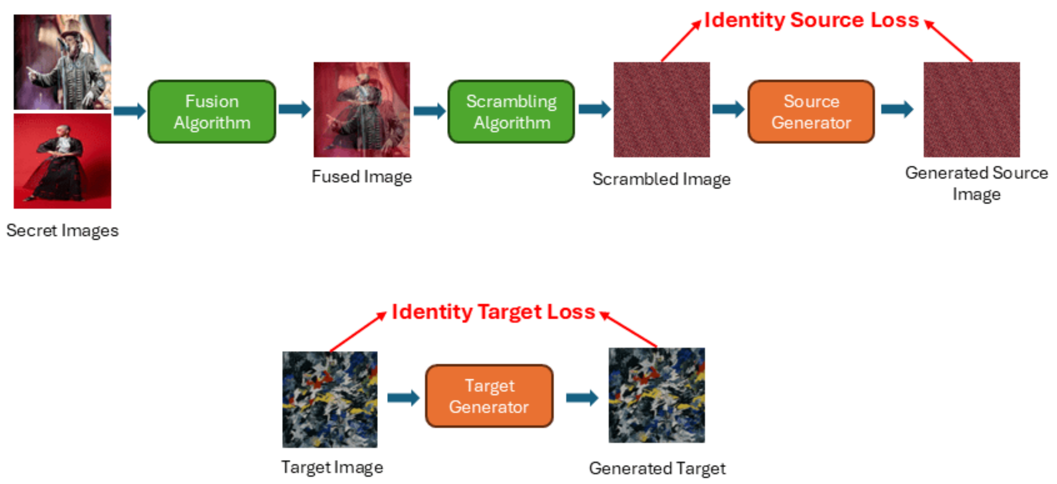


Figure 14: Identity Loss

This loss ensures that the encoder does not alter target images significantly, maintaining their original content. It helps the model learn a mapping that retains the structural and perceptual integrity of target images while preventing unnecessary transformations.

3.4.4 Intermediate Loss

Intermediate loss is introduced to regulate the encoding process and improve the robustness of the model in preserving key features of the secret images. It enforces consistency in the intermediate representations produced by the fusion and scrambling algorithms, before generating the final output.

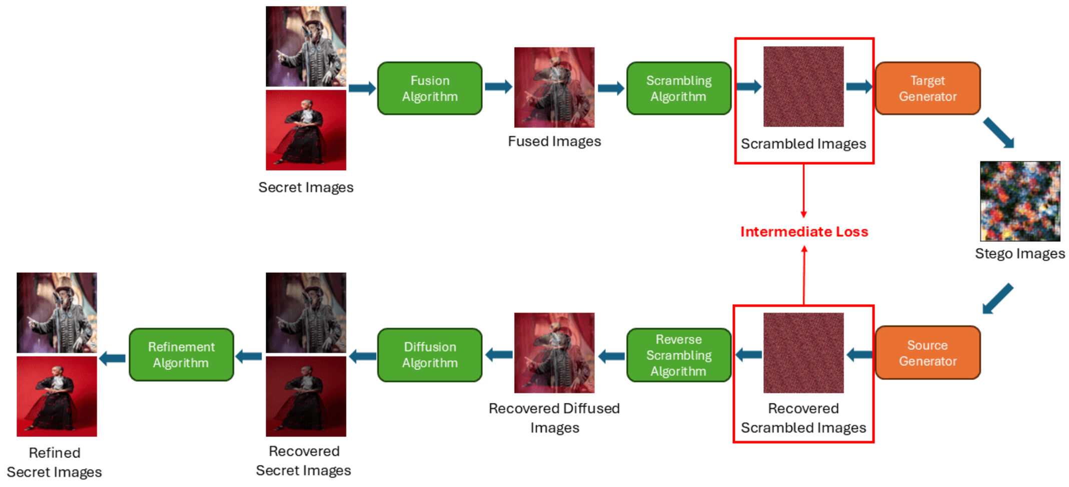


Figure 15: Intermediate Loss

This loss ensures that the fused and scrambled representations of secret images and target images are structurally aligned, improving the model's ability to reconstruct secret images while maintaining the quality of generated target images.

3.4.5 Cumulative Loss

The cumulative generator loss is a composite of all individual loss functions introduced in the model, ensuring the generator learns to produce high-quality, realistic, and structurally consistent outputs. This combined loss guides the model to strike a balance between adversarial realism, reconstruction fidelity, and feature preservation during training.
It incorporates the adversarial losses from both the source and target discriminators, encouraging the generator to produce visually convincing target images that can successfully fool both discriminators. To maintain content consistency, the cycle consistency losses for both the secret and target domains are included, ensuring that transformations are reversible and that original inputs can be accurately reconstructed.
The identity loss helps preserve features when images from the target domain are passed through the generator, promoting stability and reducing unnecessary changes. The intermediate loss enforces alignment between intermediate representations derived during the encoding and decoding processes, improving the model's ability to preserve important structural information from secret images. Additionally, the refine loss contributes by encouraging the generator to produce more

detailed and perceptually coherent outputs.
Each of these losses is scaled by a corresponding lambda value—hyperparameters that control the importance of each component during training. These lambda weights allow for prioritizing certain aspects of the learning process, such as placing more emphasis on reconstruction quality or adversarial realism, depending on the optimization goals. By integrating these components, the cumulative generator loss enables robust training of the model, improving its ability to encode multiple secret images effectively while maintaining visual quality and recoverability.

Generator Loss is defined as follows.

$$\begin{aligned}\mathcal{L}_{gen} = {} & \mathcal{L}_{gen\_source} + \mathcal{L}_{gen\_target} \\ & + \lambda_{cycle}(\mathcal{L}_{cycle\_source} + \mathcal{L}_{cycle\_target}) \\ & + \lambda_{identity}(\mathcal{L}_{identity\_source} + \mathcal{L}_{identity\_target}) \\ & + \lambda_{intermediate}\mathcal{L}_{intermediate} \\ & + \lambda_{refine}\mathcal{L}_{refine} \end{aligned} \tag{4}$$

Discriminator Loss is defined as follows.

$$\mathcal{L}_{disc} = \frac{1}{2}(\mathcal{L}_{disc\_source} + \mathcal{L}_{disc\_target}) \tag{5}$$

Variable Definitions are as follows.

- $\mathcal{L}_{gen}$: Total generator loss used to optimize the generator network.
- $\mathcal{L}_{gen\_source}$: Adversarial loss from the source discriminator evaluating the quality of the reconstructed secret image.
- $\mathcal{L}_{gen\_target}$: Adversarial loss from the target discriminator assessing the realism of the generated target image.
- $\mathcal{L}_{cycle\_source}$: Cycle consistency loss ensuring accurate reconstruction of secret images after transformation and inverse transformation.
- $\mathcal{L}_{cycle\_target}$: Cycle consistency loss ensuring the generated target images retain their original content after round-trip translation.
- $\mathcal{L}_{identity\_source}$: Identity loss encouraging the generator to preserve structural features of secret images when passed directly.
- $\mathcal{L}_{identity\_target}$: Identity loss ensuring that target images remain visually consistent when processed by the generator.
- $\mathcal{L}_{intermediate}$: Intermediate loss enforcing alignment of intermediate feature representations between secret and target domains.

- $\mathcal{L}_{refine}$: Refine loss focusing on enhancing fine-grained image details and improving perceptual quality in reconstructions.
- λ_{cycle}: Weighting coefficient for the cycle consistency loss, controlling its impact on generator optimization.
- $\lambda_{identity}$: Weighting coefficient for the identity loss.
- $\lambda_{intermediate}$: Weighting coefficient for the intermediate loss.
- λ_{refine}: Weighting coefficient for the refine loss.
- $\mathcal{L}_{disc}$: Total discriminator loss used to train both the source and target discriminators.
- $\mathcal{L}_{disc\_source}$: Loss from the source discriminator that distinguishes real vs. reconstructed secret images.
- $\mathcal{L}_{disc\_target}$: Loss from the target discriminator that distinguishes real vs. generated target images.

By aggregating these complementary objectives into a unified loss function, the model is trained to encode multiple secret images within target images while maintaining high visual fidelity and reliable reconstruction. This holistic loss formulation fosters robust steganographic encoding and decoding capabilities.

3.5 Evaluation Plan

The proposed model will be evaluated based on several key performance aspects, including its ability to conceal the visual features of secret images effectively, the perceptual quality of the reconstructed images, and the degree of difference between the original and reconstructed images. Additionally, the security and imperceptibility of the carrier images will be assessed, along with the overall hiding capacity of the method. The robustness of the model under various transformations or processing conditions will also be examined to ensure its reliability and effectiveness in practical applications.

3.5.1 Image Quality Evaluation

The reconstruction quality of images is a critical aspect in multi-image steganography, as it directly influences the effectiveness of the hiding technique. High-quality reconstruction ensures that the original secret images can be accurately retrieved without significant visual loss or distortion. Any degradation in reconstruction can result in incorrect or blurred outputs, compromising the overall objective of secure and reliable steganography. In this study, the evaluation begins with a subjective comparison of the original and reconstructed secret images through visual inspection. To further support this assessment quantitatively, metrics such as

Mean Square Error, Mean Square Error, Peak Signal-to-Noise Ratio and Absulute Mean Pixel Error are employed to measure the accuracy of the reconstruction process. These metrics help determine the most effective fusion and reconstruction strategies for achieving optimal visual fidelity.

- Mean Squared Error (MSE)
  MSE calculates the average of the squared differences between corresponding pixel values in two images. It penalizes larger errors more heavily than AMPE and is sensitive to outliers.

  $$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(I_1(i) - I_2(i))^2$$

  Where:

  - $I_1(i)$ and $I_2(i)$ are the pixel values at position i in the first and second images respectively.
  - N is the total number of pixels in the image.

- RMSE (Root Mean Squared Error)
  RMSE is the square root of MSE and provides error in the same unit as the image pixels, making it more interpretable.

  $$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(I_1(i) - I_2(i))^2}$$

  Where:

  - $I_1(i)$ and $I_2(i)$ are the pixel values at position i in the first and second images respectively.
  - N is the total number of pixels in the image.
  - $(I_1(i) - I_2(i))^2$ is the squared difference between corresponding pixel values.

- PSNR (Peak Signal-to-Noise Ratio)
  PSNR is used to measure the quality of the reconstructed image compared to the original, especially in lossy compression. A higher PSNR generally indicates better reconstruction quality.

  $$\text{PSNR} = 20 \cdot \log_{10}\left(\frac{\text{MAX}_I}{\sqrt{\text{MSE}}}\right)$$

  Where:

  - MAX_I is the maximum possible pixel value of the image (e.g., 1.0 for normalized images or 255 for 8-bit images).

- Absolute Mean Pixel Error (AMPE)
  AMPE measures the average of the absolute differences between corresponding pixel values in two images. It gives an intuitive sense of how much, on average, the pixel values differ, regardless of the direction of the difference (positive or negative).

  $$\text{AMPE} = \frac{1}{N}\sum_{i=1}^{N} |I_1(i) - I_2(i)|$$

  Where:

  - $I_1(i)$ and $I_2(i)$ are the pixel values at position i in the first and second images respectively.
  - N is the total number of pixels in the image.

3.5.2 Steganalysis Evaluation

The proposed models offer a significant advantage in terms of security, primarily because the intermediate image is generated without the use of any predefined cover image. This coverless nature of the steganographic approach reduces the risk of detection through traditional steganalysis tools, which are typically designed to analyze modifications in known cover images. As a result, the employed methodologies inherently enhance the stealth of the communication by making it more resistant to common steganographic attacks and analysis techniques. This design choice supports a more secure and undetectable method for embedding and transmitting secret information.

3.5.3 Hiding Capacity

In image steganographic models, embedding capacity refers to the maximum amount of secret data that can be concealed within an image without noticeably degrading its quality or attracting attention. Traditionally, this is measured in bits or bytes per cover image and is a key metric for evaluating the effectiveness of steganographic techniques. A higher embedding capacity indicates a greater ability to embed secret content while preserving visual integrity
However, in coverless image steganography, where no predefined cover image is used, the concept of embedding capacity is adapted to reflect how much information can be concealed within the generated intermediate image itself. Rather than embedding into an existing cover, secret data is fused and manipulated to produce an entirely new image that inherently carries the hidden content. In this

context, embedding capacity is measured as bits or bytes per intermediate image, emphasizing the model's capability to hide multiple images or a large amount of data while maintaining the security and quality of the output.

3.5.4 Robustness

In steganography, the integrity of the intermediate image is critical to the accurate reconstruction of hidden information. However, during transmission, the intermediate image may be subject to various distortions caused by malicious attacks, communication channel issues, or environmental noise. These distortions can significantly degrade the quality of the reconstructed images and compromise the embedded secret information. To evaluate the robustness of the proposed steganographic method, a series of common attacks are applied to the intermediate image. These attacks simulate real-world conditions and test the model's resilience in preserving and recovering the secret content. The evaluation includes:

- **Scaling Attack** – Resizing the image to a different resolution and then restoring it, which may distort pixel-level data.
- **Rotational Attack** – Rotating the image at various angles to assess recovery consistency after geometric transformations.
- **Cropping and Padding Attack** - Cropping portions of the image and padding them back to original dimensions, leading to spatial content loss or misalignment.
- **Flipping Attack** - Horizontally or vertically flipping the image to simulate basic transformations.
- **Gaussian Noise Attack** – Introducing normally distributed noise to simulate natural channel disturbances.
- **Salt and Pepper Noise Attack** – Randomly replacing pixel values with extreme black or white, mimicking sudden data loss or interference.
- **Speckle Noise Attack** – Applying multiplicative noise that simulates granular distortions often seen in electronic transmission.
- **Median Filtering Attack** – Applying non-linear filtering to reduce noise, which may alter important pixel patterns used in reconstruction.
- **Mean Filtering Attack** – Using averaging filters to smooth the image, which can blur critical details necessary for accurate recovery.
- **Motion Blur Attack** - Simulating camera or object movement to create streaking artifacts that obscure fine image details.

- **Color Jitter Attack** - Randomly altering brightness, contrast, saturation, and hue to simulate variations in lighting conditions.
- **Channel Shuffle Attack** - Randomly permuting the RGB channels to challenge the model's ability to recover based on disrupted color semantics.
- **Random Deletion Attack** - Randomly deleting image patches or pixel groups, imitating partial data loss scenarios.
- **Random Alteration Attack** - Applying arbitrary changes to pixel values to simulate unpredictable tampering or corruption.

By testing the model under these adverse conditions, the robustness and reliability of the proposed method in extracting and reconstructing secret images are rigorously assessed.

3.6 Chapter Summary

This section outlines the proposed methodologies in detail, including the architectural components, algorithmic procedures, and loss functions utilized during experimentation. Additionally, it presents the evaluation strategy adopted to validate the effectiveness and applicability of the proposed approaches.

4 Implementation Details

4.1 Language and Implementation Tools

We selected Python 3 as the programming language for our project due to its extensive ecosystem of libraries tailored for deep learning and computer vision applications. Our study leveraged popular open-source machine learning frameworks, including PyTorch for deep learning, OpenCV for computer vision tasks, and NumPy for numerical computations. Additionally, we utilized ‘tqdm‘ for progress tracking, along with ‘Dataset‘, ‘DataLoader‘, and ‘transforms‘ to streamline data handling. The ‘torch‘ library was also incorporated to enhance deep learning capabilities.
Initially, we trained our model using Google Colab, but later transitioned to AntPC for improved computational resources. To facilitate remote access to AntPC, we employed OpenVPN, which enabled a secure SSH connection to the server. This comprehensive set of tools and libraries provided a robust foundation for the efficient development and implementation of our proposed model.

4.2 Experimental Setup

We used Python 3.6.9 for all our experiments, including model creation and evaluation. These tasks were conducted on the ‘AntPC’ server, equipped with four NVIDIA GeForce RTX 2080 GPUs. The server runs Ubuntu 18.04.1 LTS and is powered by an Intel E5-2620 v4 CPU clocked at 2.10GHz, with 125.65GB of RAM. This system setup provided a robust and reliable foundation for our study, ensuring high computational power and stable, continuous training.

4.3 Data Collection and Preprocessing

The dataset for secret images was sourced from the ImageNet dataset, a large-scale collection widely used in computer vision research. Similarly, the dataset for carrier images was obtained from the Delaunay dataset, which provides diverse natural and synthetic images suitable for use as carrier images in steganography.
For consistency, all images were resized to a uniform dimension of 256×256 pixels. This step was essential to standardize the input format, preventing variations in resolution from affecting model performance. Furthermore, preprocessing involved vertically fusing two secret images to form a single input representation. This method allowed the model to process multiple secret images simultaneously while maintaining a structured input format for batch processing.
To enhance the training process, two additional datasets were created: one containing fused images and another consisting of fused scrambled images. These datasets were generated using specialized fusion algorithm as described in Chapter 4.4.1 and scrambling algorithms as described in Chapter 4.4.3 designed to manipulate and preprocess images in a controlled manner. The fusion algorithm was

applied to fuse multiple images into a single composite image, ensuring that the secret information was effectively embedded. Meanwhile, the scrambling algorithm introduced pixel-wise transformations to obscure identifiable patterns in the secret images.

4.4 Component Implementations, Model Development and Training

The implementation details of the different components of the proposed models are discussed in this section.

4.4.1 Pixel Wise Manipulation

- **Fusion Algorithm**
  In previous studies, generative models based on deep learning networks were often employed for fusing images. However, in this study, a novel fusion technique grounded in image manipulation principles was developed, incorporating tailored enhancements to achieve effective image fusion. The specific technique used for fusing two images follows a pixel-wise alternating pattern. Both images, sized at 3x256x256 (representing the RGB color channels), are fused in a novel way. In the fused image, the first row alternates pixels between the two images: the first pixel is taken from the first image, the second from the second image, and this alternating pattern continues across the entire row. In the second row, the pattern is reversed: the first pixel comes from the second image, the second from the first, and so on. This alternating pixel-wise pattern is maintained across all rows of the image, resulting in a delicate and evenly distributed interleaving of pixel data from both images. This method does lead to a certain loss of information, as each image contributes only half of its original pixel data. However, it creates a finer, more intricate blending of the two images, giving rise to a mosaic-like effect where details from both images intermingle at the pixel level. The technique's flexibility allows it to be extended to various patterns and resolutions, providing an effective solution for artistic or technical applications that require a subtle yet dynamic fusion of image data. It strikes a balance between preserving visual details from both images while enabling a seamless and highly granular blend, making it suitable for scenarios where precise control over information sharing between the images is desired.

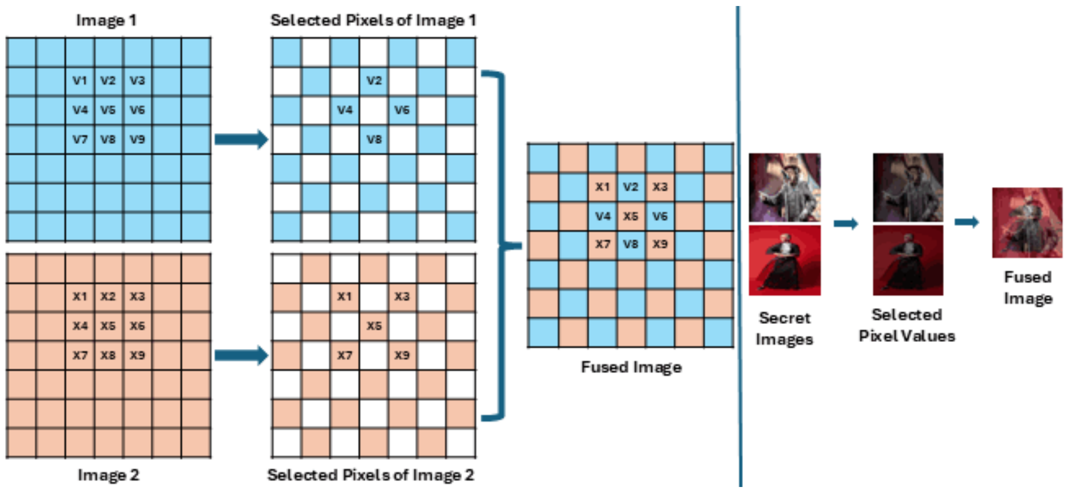


Figure 16: Pixel Wise Fusing Architecture
Left Column – Pixel value representation of Secret Images, Middle Column – Selected Pixels from Each Secret Image Used for Fusion, Right Column – Fused Image Constructed from the Selected Pixels of Multiple Secret Images.

Pixel-wise fusion offers several key advantages, including a finer granularity of blending, which creates a smoother and more cohesive transition between images. This approach interweaves individual pixels, resulting in a more intricate and seamless mix. It ensures a more balanced distribution of information, preserving critical details from both images across the entire image. Additionally, pixel-wise fusion enhances perceptual quality by reducing noticeable boundaries between images, making the final fused image appear more natural and continuous. Furthermore, it improves data recovery, as it retains more detailed pixel information, minimizing data loss and making it ideal for scenarios that require preserving original details and ensuring uniform representation.

- **Diffusion and Refinement Algorithms**
  To reverse the fusion process and recover the original images from the fused one, I employed a method that essentially follows the inverse of the pixel-wise alternating pattern used during fusion. In this approach, the pixels are redistributed back to their original positions in the two images, but due to the nature of pixel interleaving, only half of the data can be recovered. The alternating pixel-wise fusion results in a situation where every other pixel belongs to one image or the other, and hence, reconstructing the original image directly from the fused version leaves some data missing. This recovery is possible but limited by the alternating structure, and only approximately half of the original information can be retrieved using this approach.
  While the basic reversal method can recover half of the image's original data, it doesn't fully restore the details lost during fusion. To address this limitation, a refinement technique was applied to recover the missing data. Specifically, a neighbor-based technique using four nearest neighbors was developed.

In this process, missing pixel values were estimated by calculating the average of the four nearest neighbors of each missing pixel. This method works by relying on the assumption that neighboring pixels have similar values, a reasonable assumption in many images where smooth transitions between neighboring pixels are common. By averaging the values of the four nearest neighbors, the missing data could be approximated more accurately, leading to a significant reduction in the loss of information.

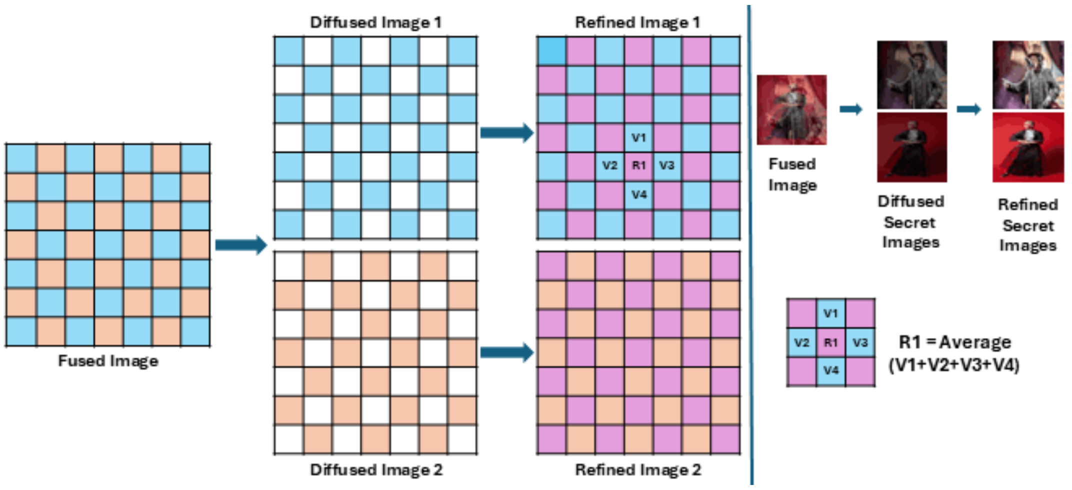


Figure 17: Pixel Wise Diffuse and Refine Architecture
Left Column – Fused Image, Middle Column – Diffused Images with recovered pixel values, Right Column – Refined Secret Images with Four Nearest Neighbours. The refinement step enhances pixel accuracy by averaging neighboring values (e.g., R1 = Average of V1, V2, V3, V4).

This refinement technique offers several key benefits. First, it allows for a more accurate recovery of missing pixel data compared to simple interpolation methods. Since the missing pixels are estimated based on the average of neighboring pixels, the values are distributed more evenly and naturally. This results in a finer restoration of image details, making the recovered image appear much closer to the original than it would with simple interpolation. Moreover, the use of four nearest neighbors provides a better distribution of data, especially in areas where pixels are highly interdependent, such as smooth textures or color gradients. This leads to a more coherent and visually consistent image, which is essential for applications requiring high-quality image reconstruction.

In summary, the reverse of the fusion method recovers approximately half of the original data, but this is complemented by a refinement technique that uses the average of the four nearest neighbors to restore the missing pixels. This technique not only fills in the missing information effectively but also enhances the overall visual quality of the recovered image by providing a better distribution of pixel values. As a result, the recovery process becomes more accurate and faithful to the original images, making it suitable for situations where precise image reconstruction is necessary.

4.4.2 LSB Manipulation

- **Fusion Algorithm**

  The figure 18 illustrates a customized image fusion methodology designed to fuse two RGB secret images (each sized 3×256×256) into a single fused image using a pixel-wise alternating pattern. This novel fusing approach is inspired by the traditional image-to-image hiding technique proposed by Baluja (2017), but with significant adaptations. In this method, only the Most Significant Bit (MSB) values of the secret images are utilized, while the Least Significant Bit (LSB) are entirely discarded. This selective bit-level handling ensures the preservation of perceptually dominant visual features while maximizing concealment.

  The fusion process begins by extracting the MSB values from corresponding pixels in both secret images. These extracted bits are then strategically placed into both the MSB and LSB positions of the resulting fused image using an alternating pixel-wise pattern. Specifically, for each pixel location, the MSB of one secret image is assigned to the MSB position in the fused image, while the MSB of the second secret image is mapped to the LSB position—thereby creating a novel combination that encodes two sets of dominant visual features into a single image.

  In the fused image, the placement of MSB values from the two secret images follows a carefully structured interleaving strategy. For each pixel in the fused image, both the MSB and LSB positions are populated using MSB values derived from the corresponding pixels of the two input secret images, rather than true high- and low-significance data. This is done to maximize visual fidelity while ensuring a secure blend of both inputs.

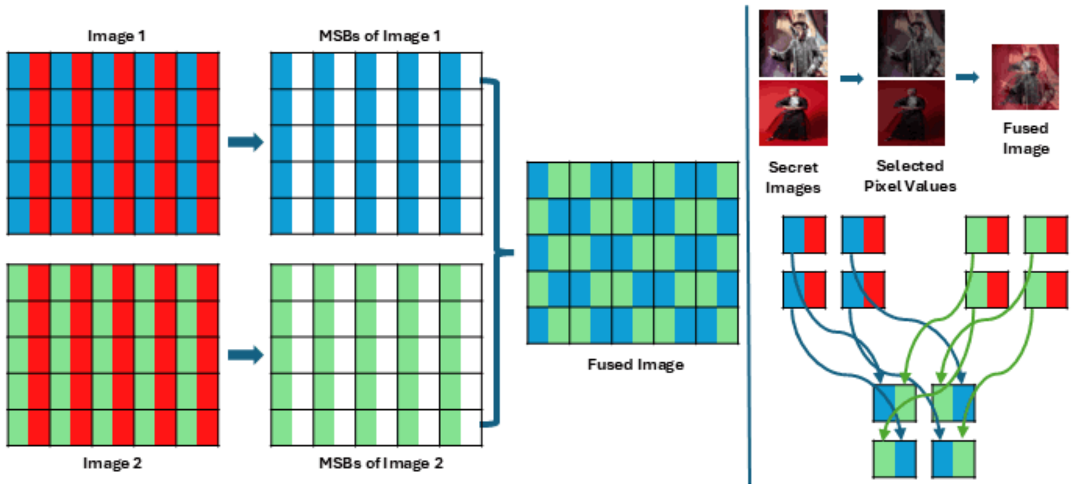


Figure 18: LSB Fuse Architecture
Left Column – Pixel value representation of Secret Images (MSBs: Blue in Image 1 and Green in Image 2; LSBs: Red in both images), Middle Column – Selected MSB values from Each Secret Image Used for Fusion, Right Column – Fused Image Constructed from the Selected MSBs of Multiple Secret Images.

The placement follows an alternating row-wise logic. In the first row, the MSB of the first image is inserted into the MSB position of the fused pixel, while the MSB of the second image is placed into the LSB position. For the next pixel in the same row, this assignment is reversed: the MSB from the second image goes into the MSB position, and the MSB from the first image goes into the LSB position. This alternating MSB-LSB mapping continues across the entire row.

In the second row, the pattern is flipped to maintain balance and distribution: the first pixel takes the MSB from the second image for the MSB position and the MSB from the first image for the LSB position, and this alternation continues across the row.

This interleaved MSB-to-MSB/LSB mapping is maintained across all rows of the fused image, creating a uniform and visually imperceptible blend of both secret images. Although only MSB data is used and LSB information is discarded, the use of interleaved placement within both bit planes (MSB and LSB) retains enough visual structure to reconstruct or analyze the fused content, while keeping the loss undetectable to human vision due to minimal perceptual difference across similar color codes.

Despite this fusion, half of the original information from the secret images is lost due to the absence of LSB data. However, since human perception is primarily sensitive to MSB-level visual information, the resulting fused image appears visually rich and natural. Furthermore, the loss of 16 consecutive color codes during the LSB discarding phase is imperceptible to the human eye, offering no distinguishable clue that any data has been removed—thus enhancing the concealment quality and making detection by unintended observers significantly more difficult.

- **Diffusion Algorithm**
  The image diffusion phase serves as the inverse operation of the image fusion stage, aiming to recover the original secret images from the fused image. This step relies on the same alternating pixel-wise pattern established during fusion to accurately separate the interleaved pixel data. For every row in the fused image, pixel values are selectively extracted in an alternating sequence and assigned back to their respective secret images.

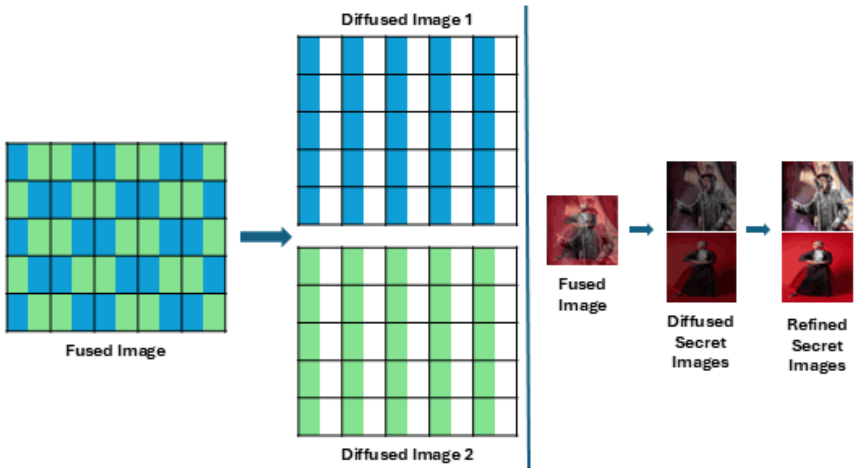


Figure 19: LSB Diffuse Architecture
Left Column – Fused Image, Right Column – Diffused Images with recovered MSB values

During diffusion, only the most significant bits (MSBs) of each pixel are recovered, while the least significant bits (LSBs) are entirely discarded. This selective recovery is intentional and fundamental to the methodology. Although this results in a loss of half of the original data, the MSBs carry the most visually impactful information. As a result, the reconstructed secret images preserve the core structure, dominant colors, and essential visual features of the original images. The human eye, being less sensitive to the absence of fine-grained details carried by LSBs, perceives the reconstructed images as nearly identical to the originals.
In the visual representation of reconstructed images, the areas where LSBs were discarded appear as blank regions. These missing bits, however, do not significantly impact human visual perception, as the preserved Most Significant Bits retain the dominant visual features of the image. Additionally, the reconstruction produces a low error value when compared to the original image, indicating minimal loss of perceptual quality. This unique characteristic allows the system to omit any post-recovery refinement or enhancement processes, reducing computational complexity while maintaining high visual fidelity.

The methodology not only simplifies the recovery pipeline but also ensures that the reconstructed images are visually convincing without requiring the full bit-depth information. This approach strikes a practical balance between data compression, concealment, and perceptual accuracy, making it a lightweight and secure solution for coverless multi-image steganography. The ability to achieve strong visual results with partial data showcases the robustness and efficiency of the proposed diffusion technique.

4.4.3 Arnold Scrambling

- **Arnold Transformation**

  Arnold scrambling is a technique used to alter the pixel positions of an image in a manner that makes the image appear scrambled. It is widely used in image steganography to conceal information within an image, making it difficult to interpret without the proper transformation parameters. The scrambling process involves changing the coordinates of each pixel in the image based on mathematical equations. The pixel coordinates are mapped to new positions using the following transformation formulas:

  $$x_{\text{new}} = (x + y) \mod N$$

  $$y_{\text{new}} = (x + 2y) \mod N$$

  In this method, x and y represent the original pixel coordinates, while x_{new} and y_{new} are the new coordinates after scrambling. The term N denotes the size of the image, typically used in square images (e.g., 256x256). The modulo operation ensures that the coordinates stay within the image boundaries, creating a "wrapped" effect when the calculation exceeds the image dimensions.

  In this method, the transformation equations mix both the x and y coordinates of each pixel, leading to a seemingly random arrangement of pixels. As a result, the image becomes unreadable or encrypted without knowing the specific transformation parameters. This scrambled image can be used in various applications, including hiding secret images or data (Min et al. 2013).

- **Reverse Arnold Transformation**

  To recover the original image from the scrambled image, the Reverse Arnold transformation is applied. This reverse transformation undoes the effects of the scrambling algorithm by applying the inverse of the original transformation. The reverse equations are as follows:

  $$x_{\text{new}} = (2x - y) \mod N$$

$$y_{\text{new}} = (-x + y) \mod N$$

In this case, x and y represent the scrambled pixel coordinates, while x_{new} and y_{new} are the original pixel coordinates prior to the scrambling. The modulo operation ensures the recovery of the pixel positions within the bounds of the image size, just as it did during the scrambling process.

By applying the Reverse Arnold transformation to the scrambled image, the original image is restored, allowing the hidden information to be revealed. The ability to reverse the scrambling process is crucial in steganographic systems, where the aim is to securely embed information within an image and later recover it without revealing the hidden content to unauthorized users (Min et al. 2013).

4.4.4 Generator

The proposed GAN based coverless multi image steganographic strategy uses two generator networks to perform transformations between scrambled images and abstract paint art representations. The Target Generator as illustrated in Figure 10 converts the scrambled image into an abstract paint art image. This generated abstract image functions as the carrier image (stego image) in the GAN based steganographic model. The carrier image is the one that is transmitted from the sender to the receiver and plays a key role in concealing the presence of hidden content because it shows no direct visual similarity to the original secret images.

The Source Generator as illustrated in Figure 10 is used to convert an abstract paint art image into a scrambled image. It takes the carrier image which is the abstract paint art and reconstructs the original scrambled image. This step ensures that the hidden information can be retrieved by the receiver. To make this bidirectional transformation effective the model is trained using CycleGAN. CycleGAN is capable of learning unsupervised image to image translation by training both the Target Generator and the Source Generator together with consistency constraints which helps to maximize the generator performance as much as possible. The complete network architecture used in this approach is shown in Figure 20.

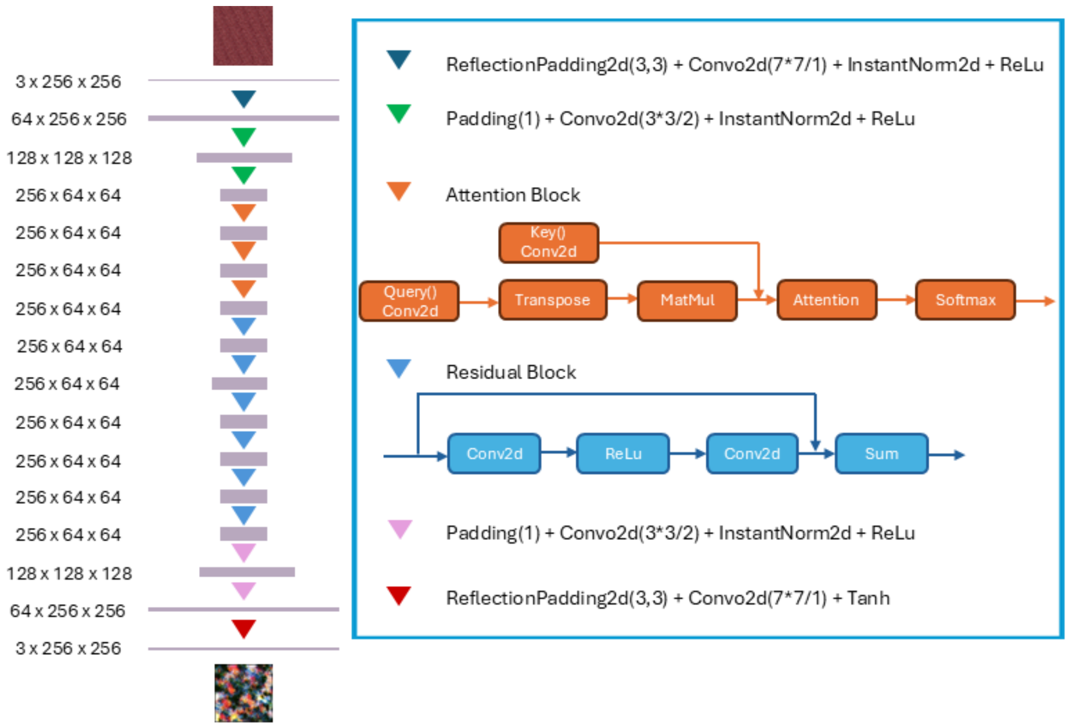


Figure 20: Generator Network

4.4.5 Discriminator

The discriminator networks play a vital role in enhancing the quality and realism of the generated images during the training process. Source Discriminators as illustrated in Figure 10 are designed to differentiate between the original secret images and the reconstructed secret images that are recovered from the scrambled intermediate representations.

In parallel, the Target Discriminator as illustrated in Figure 10 focuses on distinguishing between the original abstract paint art images and those generated by the Target Generator from scrambled inputs. This ensures that the generated abstract images possess the natural characteristics of real abstract artwork, making them suitable to act as inconspicuous carrier images in the steganographic process.

Their primary function of two discriminators is to guide the Source Generator to produce reconstructed outputs that closely resemble the original secret inputs in both structure and visual appearance, while simultaneously directing the Target Generator to generate transformed outputs that effectively emulate the abstract paint art style, preserving structural and visual coherence.

To achieve more localized and detailed feedback during training, the PatchGAN architecture is employed for both the Source and Target Discriminators. Rather than making a single real-or-fake decision for the entire image, PatchGAN classifies each local patch within an image, allowing the network to focus on fine-grained texture and structure. This approach improves the discriminative power of the model and promotes the generation of more realistic and detailed outputs. The full network architecture of the discriminators is depicted in Figure 21.

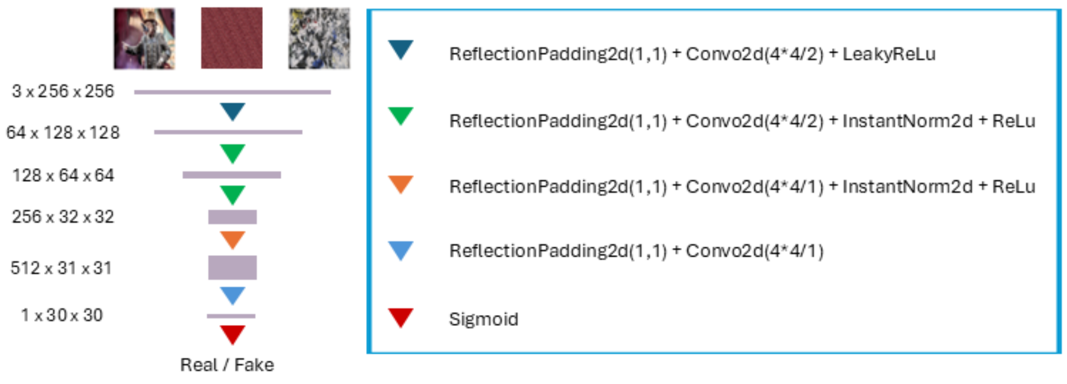

3 x 256 x 256
64 x 128 x 128
128 x 64 x 64
256 x 32 x 32
512 x 31 x 31
1 x 30 x 30
Real / Fake
ReflectionPadding2d(1,1) + Convo2d(4*4/2) + LeakyReLu
ReflectionPadding2d(1,1) + Convo2d(4*4/2) + InstantNorm2d + ReLu
ReflectionPadding2d(1,1) + Convo2d(4*4/1) + InstantNorm2d + ReLu
ReflectionPadding2d(1,1) + Convo2d(4*4/1)
Sigmoid

Figure 21: Discriminator Network

4.5 Chapter Summary

This section provides the implementation details of the proposed methodologies, focusing on the architecture of its components and the algorithms employed at various phases to achieve coverless multi-image steganography. It includes a description of the language and implementation tools used, the experimental setup established for evaluation, and the procedures followed for data collection and preprocessing. These implementation details collectively ensure the practicality, efficiency, and reliability of the proposed approaches in concealing and reconstructing multiple RGB secret images without relying on a predefined cover.

5 Experiments, Preliminary Results and Analysis

5.1 Investigate on compressing multiple images into a single intermediate image with higher reconstruction accuracy with minimum data loss and minimum visual loss

The initial phase of the full steganographic architectures, as illustrated in Figure 9 and Figure 10, involves compressing multiple secret images into a single intermediate image through an image fusion process.

In this phase, two secret images are fused to produce an intermediate image that retains balanced data from both original images. This intermediate image displays visible features from both source images, making it unsuitable as a carrier for secure transmission between two parties, as it reveals substantial information about the secret images. The purpose of this fusion is not to conceal visual features but rather to maximize the information captured from each image within a single intermediate representation. This intermediate image can then be used to facilitate the reconstruction and refinement of the original images.

Once the intermediate image is obtained, it undergoes a reconstruction process to retrieve the original secret images. Each reconstructed image is then refined to minimize data loss and visual feature loss, thereby enhancing the fidelity of the recovered images. For this process, JPEG images with a resolution of 256x256 were used, chosen to balance image quality with computational efficiency due to the high processing power required. This stage sets the foundation for a robust steganographic system that prioritizes image embedding capacity and high-quality reconstruction.

Figure 22 illustrates the Image Fusion, Diffusion and Refinement phases.

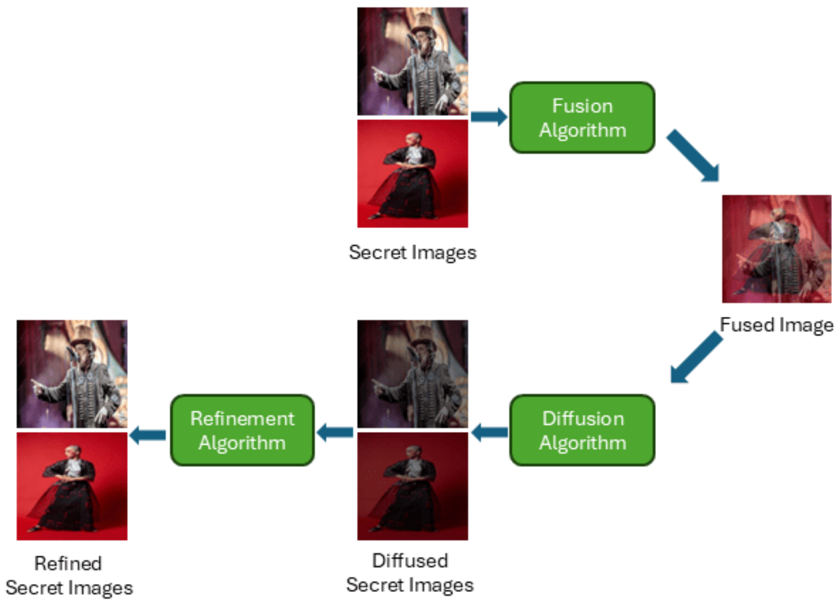


Figure 22: Image Fusion Phase

5.1.1 Pixel Wise Image Fusion

The technique used to fuse two images follows a pixel-wise alternating pattern as described in Chapter 4.4.1. Both images are of size 3×256×256, representing three color channels (RGB). This alternating pixel-wise approach is applied for each pixel across all rows of the image, resulting in an evenly distributed interleaving of pixel data from both input images. Although this method inherently leads to partial data loss—since each image contributes only half of its original pixel information—it enables a fine-grained blending that intermixes visual details from both images at the pixel level. The resulting mosaic-like effect provides a balanced and detailed representation of the source images. This strategy is particularly advantageous in scenarios where it is essential to preserve perceptual features from multiple inputs within a single fused image.

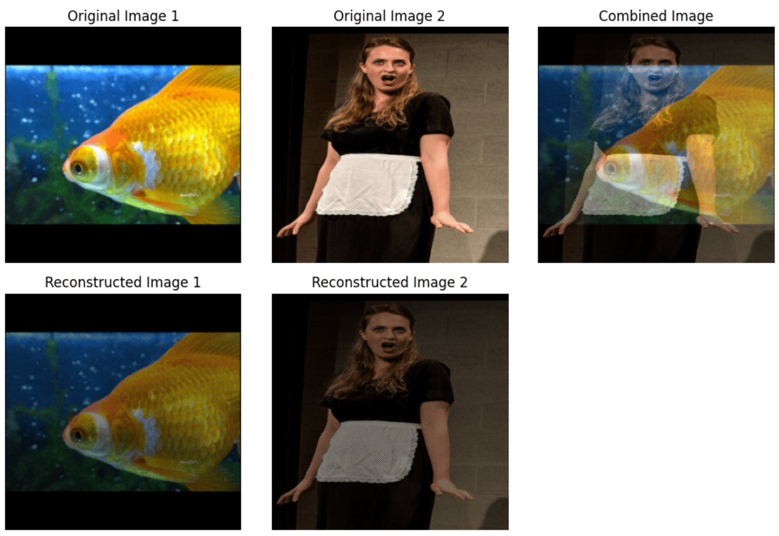


Figure 23: Pixel Wise Manipulated Image Fusion

While pixel-wise fusion may result in some degree of data loss due to the interleaving of two images, it is highly effective in terms of blending quality, perceptual smoothness, and balanced distribution of information. This technique enables finer detail preservation, making it especially suitable for scenarios where content from both images must be merged seamlessly with minimal visual disruption.

- **Granularity of Blending**
  Pixel-wise fusion enhances the granularity of image blending by operating at the level of individual pixels. By alternating pixels within each row and column between the two images, the transition between them becomes subtle and continuous. This fine-grained interleaving results in a more fluid and cohesive visual mix, where both images contribute evenly to the final result. The detailed distribution of pixel information ensures that features from both images are seamlessly integrated, making this approach effective for applications requiring a smooth and balanced fusion of visual data.

- **Perceptual Quality**
  Another major improvement offered by pixel-wise fusion lies in its superior perceptual quality. By interweaving pixels from both images in an alternating pattern, this method produces a smoother and more natural visual appearance. The fine-grained blending minimizes noticeable boundaries and reduces visual distractions, resulting in a cohesive and harmonious fused image. This refined integration is particularly beneficial in applications where seamless visual merging of multiple images is essential.

- **Information Distribution**
  Pixel-wise fusion ensures a balanced and uniform distribution of information across the entire image. By alternating pixel values from both images at a fine-grained level, this technique guarantees that each region of the fused

image contains contributions from both sources. As a result, important visual features from both images are consistently preserved throughout the image, minimizing the risk of dominance or loss of detail from either input. This uniform representation makes pixel-wise fusion particularly effective for applications requiring equal emphasis on multiple source images.

- **Data Recovery**
  Pixel-wise fusion enhances the ability to recover data from the fused image by evenly interleaving pixels from both input images. This fine-grained blending retains a higher level of detail from each image, as no large sections are omitted. The alternating pixel pattern distributes information across the entire image, reducing the impact of localized data loss and enabling more effective reconstruction of the original images. Although the recovery process is slightly more complex due to the interleaving, the overall preservation of detail makes pixel-wise fusion a reliable and effective approach when accurate reconstruction is a priority.

After applying pixel-wise fusion to create fused images, several refinement methods were utilized to recover missing pixel values and enhance the visual quality of the reconstructed secret images. These included Gaussian smoothing, bicubic interpolation, bilinear interpolation, and brightness adjustments. Although these refinements could not fully restore the original visual quality, pixel-wise fusion demonstrated strong effectiveness in preserving visual features, resulting in reconstructions that closely resemble the original images.
To improve the reconstruction of the original secret images and fill in the missing pixel values, I experimented with several image enhancement techniques, including Gaussian smoothing, Bicubic Interpolation and Bilinear Interpolation. Gaussian smoothing, which applies to a Gaussian kernel to reduce noise and blur the image, was intended to refine pixel transitions. Bicubic Interpolation, a more advanced resampling method, estimates new pixel values using a weighted average of the closest 16 pixels, while Bilinear Interpolation considers the nearest 4 pixels for a smoother but less detailed reconstruction. However, despite these efforts, none of these techniques proved effective in accurately recovering and refining the missing pixel values in the original secret images, indicating the need for a more sophisticated approach.

5.1.2 Refine the reconstructed images with Four Nearest Neighbor smoothing

This refinement technique as described in Chapter 4.4.1, fills in missing pixels in a recovered image by selectively processing specific pixels based on secret images and then using the values of the four nearest neighbors to adjust the missing pixel values. For each missing pixel that meets the pattern criteria, the function gathers valid neighboring pixels that have non-zero values and calculates their average.

This average value is then used to replace the missing pixel, smoothing transitions and improving visual coherence.
The 4-nearest neighbor approach is particularly beneficial as it leverages immediate surrounding pixels, allowing the missing pixel to inherit contextual information while preserving local textures and minimizing abrupt transitions. By selectively filling only certain pixels, this approach allows for more controlled and pattern-specific reconstruction, maintaining visual consistency and detail in the refined image.

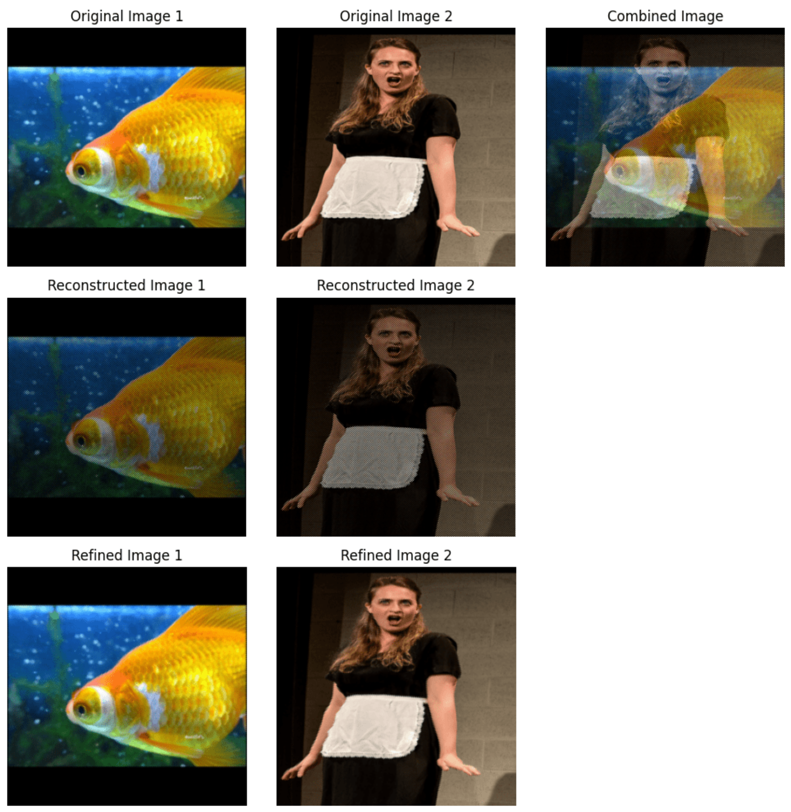


Figure 24: Refine the reconstructed images with Four Nearest Neighbor smoothing

5.1.3 LSB Manipulation Image Fusion

Least Significant Bit manipulation technique constructs a fused image by utilizing the Most Significant Bit from two secret images while entirely disregarding their LSB content. In this approach, each pixel in the fused image is composed by interleaving the MSBs of corresponding pixels from both input images. As a result, each secret image contributes only half of its original bit-level information to the fused output.

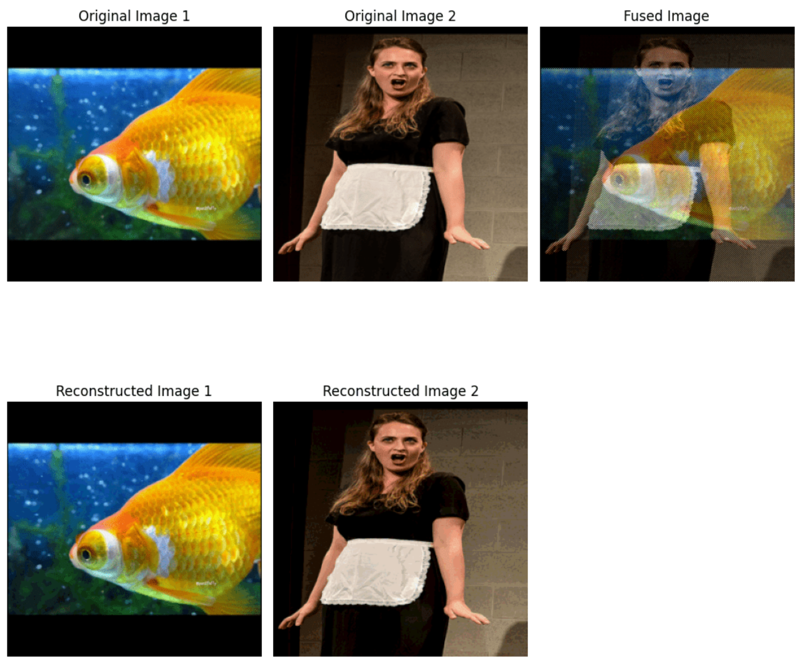


Figure 25: LSB Manipulation Image Fusion

Despite the fact that half the data from each image (namely, the original LSBs) is lost in this process, the technique ensures that the MSBs carrying the most critical visual information are preserved. This results in fused images that, when reconstructed, exhibit minimal to no perceptual difference from the originals when viewed by the human eye. The loss of the LSBs does not significantly impact visual quality, as they are the least noticeable part of a pixel. This method effectively balances data fusion with high visual fidelity, making it a suitable technique for cases where reduced data size or hidden information is necessary, without a noticeable decrease in image clarity.

- **Data Preservation**
  The Least Significant Bit (LSB) manipulation technique excels in data preservation by retaining the Most Significant Bit (MSB) of each image, which are crucial for visual representation. While pixel-wise fusion retains detailed information from both images, it effectively discards half of the original pixel data, leading to a significant loss of information. This can result in important features being underrepresented in certain areas, depending on the pixel interleaving pattern. In contrast, LSB manipulation minimizes data loss by preserving essential details while only discarding the least significant bits, which have a minimal impact on the overall visual quality.

- **Visual Quality**
  In terms of visual quality, LSB manipulation offers a clear advantage by ensuring that the resulting image retains high fidelity and appears seamless and coherent. Although pixel-wise fusion provides a more granular blending of images, it can introduce noticeable transitions or artifacts, particularly

when the two images exhibit distinct features or colors. The interleaving of pixels in pixel-wise fusion may create a mosaic effect, which can detract from the intended visual impact. By focusing on the MSBs, LSB manipulation minimizes visual artifacts, resulting in a fused image that closely resembles the originals and maintains a polished appearance.

- **Perceptual Impact**
  The perceptual impact of LSB manipulation is significantly more favorable than that of pixel-wise fusion. The human eye is less sensitive to changes in the least significant bits, meaning that the loss of LSBs typically does not create a noticeable difference in perceived quality. In contrast, pixel-wise fusion can lead to visible discontinuities between blended pixels, especially in areas with sharp contrasts. By maintaining the essential visual cues intact, LSB manipulation ensures a high level of aesthetic quality in the fused image, making it a superior choice for applications where visual harmony is important.

While both techniques of Pixel wise fusion and LSB manipulation fusion, serve the purpose of image fusion, LSB manipulation offers distinct advantages, especially in contexts where maintaining visual integrity and preserving essential details is crucial. Its simplicity, effectiveness in hiding data, and minimal perceptual impact make it a superior choice for many applications, particularly in secure image processing and scenarios requiring high-quality visual output. In contrast, pixel-wise fusion, despite its granular blending capabilities, can lead to significant data loss and visible artifacts, limiting its effectiveness in certain contexts. Though pixel-wise fusion leads to a data loss, by applying Four Nearest Neighbor Smoothing technique as the refinement technique in reconstruction process, the above mentioned problems will be able to overcome.

5.1.4 Comparison of the statical differences between original images and reconstructed images via different techniques

This subsection presents an analysis of the statistical differences between original and reconstructed images using various refinement techniques, including Bilinear Interpolation, Bicubic Interpolation, Gaussian Smoothing, and Four Nearest Neighbor. Each method aims to refine the reconstructed images to closely match the original images in terms of visual and data integrity.

- **Bilinear Interpolation** - Bilinear interpolation is a simple and efficient technique used to estimate pixel values in an image by averaging the values of the four nearest neighboring pixels. It smooths image transitions and reduces artifacts when resizing or refining low-resolution images.

- **Bicubic Interpolation** - Bicubic interpolation offers higher-quality results than bilinear by considering the closest 16 pixels (a 4x4 grid) around a target pixel. It applies cubic functions to generate smoother and more visually accurate transitions, making it ideal for enhancing image sharpness during upscaling or refinement.
- **Gaussian Smoothing** - Gaussian smoothing reduces noise and softens images by applying a Gaussian filter that weights neighboring pixel values according to a bell-shaped curve. This technique preserves edges better than basic averaging and is widely used to eliminate minor distortions or prepare images for further processing.

The evaluation is conducted using Mean Squared Difference and Mean Absolute Difference metrics, which quantify the pixel-wise discrepancies between the original and reconstructed images. For each technique, the pixel difference tensors and absolute pixel differences are recorded for both images involved in the fusion.
These comparisons highlight the effectiveness of each refinement technique, providing insights into their suitability for minimizing data and visual feature loss in reconstructed images.

	Image 1		**Image 2**	
Image Error	**Mean**	**Absolute**	**Mean**	**Absolute**
4NN Refinement	0.000179	0.004301	0.000319	0.005932
LSB Manipulation	0.000774	0.019443	0.000973	0.024711
Bilinear Refinement	0.048442	0.135292	0.044686	0.138644
Bicubic Refinement	0.078573	0.135267	0.071472	0.138567
Gaussian Refinement	0.042738	0.135785	0.038983	0.139164

Table 1: Mean Squared Error and Absolute Mean Pixel Error with Refinement Techniques

Table 1 provides a comparative analysis of five different refinement techniques of Four Nearest Neighbor (4NN), LSB Manipulation, Bilinear Interpolation, Bicubic Interpolation, and Gaussian Smoothing, based on their ability to preserve pixel similarity between original and reconstructed images which are performed on the same secret images. Among these methods, Four Nearest Neighbor demonstrates the lowest mean and absolute pixel differences for both images, indicating its effectiveness in achieving a close approximation to the originals with minimal deviation. LSB Manipulation, on the other hand, shows little bit higher mean and absolute pixel differences than the Four Nearest Neighbor technique. This result is largely due to its approach of embedding information within the least significant bits, which, while visually indistinguishable to the human eye, but leads to comparatively higher numerical differences since this technique has no ability in preserving the LSB values of original images. Bilinear and Bicubic Interpolation offer moderate pixel difference values, with bicubic slightly outperforming bilinear in terms of accuracy. Gaussian Smoothing yields relatively low Mean Squared differences and is effective in reducing high-frequency noise, though at the expense of some fine details. Overall, each method exhibits unique strengths: 4NN excels in precision, LSB maintains visual fidelity while embedding, and the interpolation and smoothing methods balance accuracy with computational efficiency.

	4NN Refinement	**LSB Manipulation**	**Gaussian Refinement**
Image 1	0.011525	0.028773	0.235576
Image 2	0.011369	0.030166	0.249380

Table 2: Average MSE Comparison on an image dataset

The data presents a comparison of Mean Squared differences across a dataset of 50 image fusions and reconstructions, using three refinement techniques: Four Nearest Neighbor, Gaussian Smoothing, and LSB Manipulation. Each technique was evaluated based on the average pixel differences between the original and reconstructed images over the dataset.
Four Nearest Neighbor exhibits the smallest Mean Squared differences for both images, with values of 0.0115 for Image 1 and 0.0114 for Image 2. This indicates a high level of precision in preserving the original image details, suggesting that this technique is particularly effective at maintaining pixel fidelity.
Gaussian Smoothing shows slightly higher Mean Squared differences, with 0.2356 for Image 1 and 0.2494 for Image 2. This technique focuses on reducing noise and achieving smoother reconstructions, but it introduces moderate pixel discrepancies, likely due to the smoothing effect that sacrifices some sharpness in favor of reducing high-frequency variations
LSB Manipulation, on the other hand, yields significantly larger Mean Squared differences, with values of 0.0288 for Image 1 and 0.0302 for Image 2. This approach, while effective for embedding data, introduces substantial pixel-level alterations in numeric terms. However, these differences are generally not perceptible to the human eye, as LSB manipulation targets manipulating the least significant bits, making it suitable for applications prioritizing data concealment over pixel-level accuracy.
Overall, Four Nearest Neighbor and and LSB Manipulation are the most accurate in preserving pixel values, which offer different benefits in data embedding and data preserving while the reconstruction process respectively.

5.1.5 Combination of Pixel wise manipulation, LSB manipulation and Four Nearest Neighbor Refinement

Throughout the previous explorations, pixel-wise manipulation combined with 4-Nearest Neighbor (4NN) refinement and Least Significant Bit (LSB) manipulation demonstrated strong fusion capabilities and high accuracy in recovering the original images. Since both techniques individually exhibited promising performance, they are integrated into a unified framework to maximize the advantages of each. This combination allows for increased hiding capacity by embedding more image information into a single representation, while still maintaining the high visual quality and recovery accuracy required for effective image steganography.

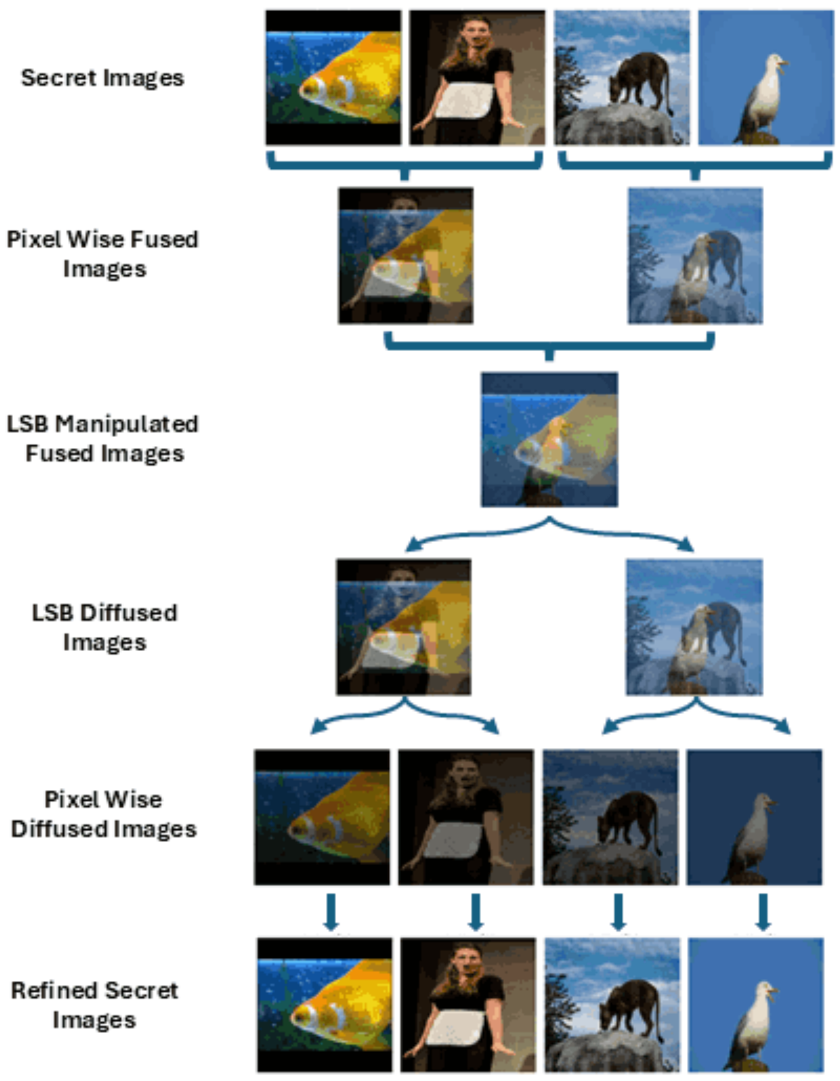


Figure 26: Multi-Image Fusion with LSB manipulation and Pixel wise manipulation

- **Fusion Phase**
  - **Pixel-wise Fusion of Secret Images**
    Two RGB secret images are fused at the pixel level. For example, selected channels or pixel values from each image are interleaved or averaged to create a fused image that conceals both sources while maintaining visual coherence.
  - **LSB Manipulation of Fused Images**
    The fused images are further compressed by embedding them into one

another using Least Significant Bit (LSB) techniques. This subtle embedding hides essential details without causing noticeable distortion.

- **Diffusion Phase**
  - **Reverse LSB Manipulation (Defusion Step 1)**
    The LSB layers are extracted to retrieve the fused images initially combined in Step 2.
  - **Reverse Pixel-wise Defusion (Defusion Step 2)**
    The original secret images are separated from the fused images using the reverse of the pixel-wise fusion algorithm applied in Step 1.
  - **4NN Refinement for Image Recovery**
    The separated images are refined using the 4-Nearest Neighbor (4NN) technique, which smooths pixel-level inconsistencies and restores sharpness, resulting in clearer and more accurate reconstructions of the original secret images.

The separated images are refined using the 4-Nearest Neighbor (4NN) technique, which smooths pixel-level inconsistencies and restores sharpness, resulting in clearer and more accurate reconstructions of the original secret images.
The proposed methodology of Security Key based coverless multi image steganography, involves a multi-phase fusion and diffusion process designed to securely embed and retrieve multiple RGB secret images into a single intermediate image, without relying on a predefined cover image. During the fusion phase, Phase 1 begins with the pixel-wise fusion of two secret images to form a fused representation. In Phase 2, the resulting fused images undergo LSB (Least Significant Bit) manipulation, enabling further embedding while preserving visual fidelity as well as preserving necessary information from all the secret images.
In the diffusion phase, the process is reversed to accurately recover the original secret images. Phase 3 performs reverse LSB manipulation to intermediate image in order to separate the previously fused images. Phase 4 diffuses the pixel-wise fused images back into the individual secret images. Finally, in Phase 5, a 4-Nearest Neighbor (4NN) refinement technique is used to enhance the quality and restore the fine details of the recovered images. This methodology ensures high hiding capacity, secure embedding, and successful reconstruction with minimal visual distortion.

5.2 Explore Scrambling Algorithms

Scrambling algorithms play a vital role in image steganography by disguising or hiding information within an image, making it difficult for unintended viewers to detect or decode. In the context of CycleGAN, scrambling techniques can be applied to alter the pixel arrangement in the input images before they are fed into the network, making the data obfuscation process more secure.

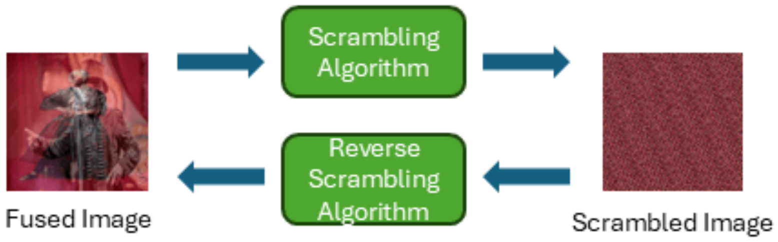


Figure 27: Image Scrambling Phase

5.2.1 Arnold Transformation

Arnold transformation, known for its chaotic nature, is particularly effective for image scrambling as it rearranges pixel positions in a deterministic yet complex manner. In image steganography, applying Arnold transformation to hide messages or patterns within images allows the data to be securely concealed. When used with proposed methodologies, the transformation ensures that even during transformations, hidden content remains obfuscated. This makes Arnold transformation a powerful tool in the field of image steganography, as it enhances both security and recoverability of hidden data.

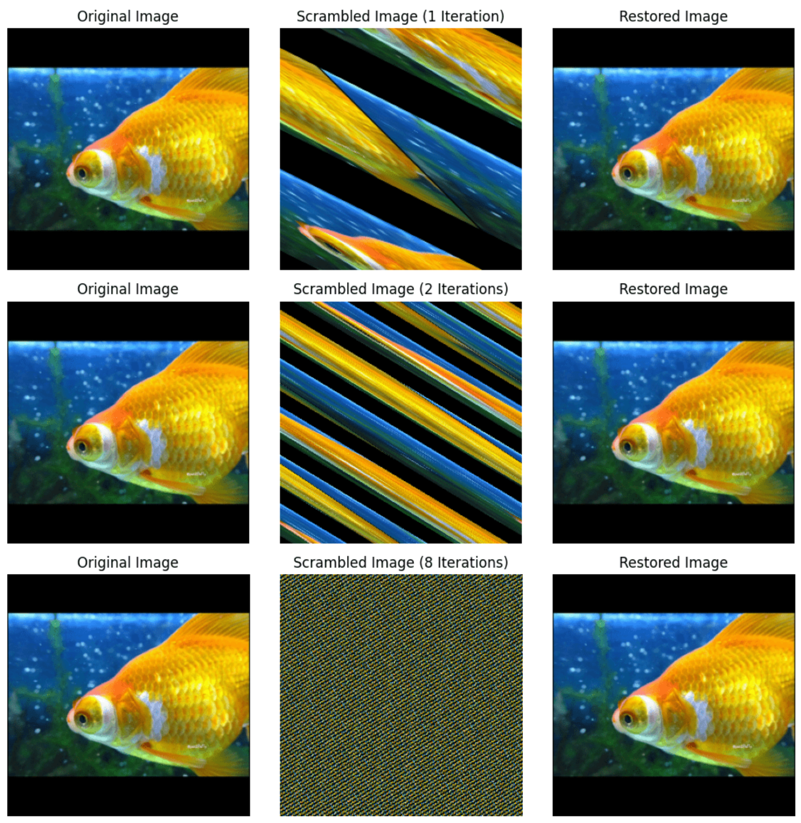

Original Image
Scrambled Image (1 Iteration)
Restored Image
Original Image
Scrambled Image (2 Iterations)
Restored Image
Original Image
Scrambled Image (8 Iterations)
Restored Image

Figure 28: Arnold Transformation
Row 1 - Scramble for 1 iteration, Row 2 - Scramble for 2 iteration, Row 3 - Scramble for 8 iteration

5.3 Combination Image Fusion techniques with Scrambling algorithms

Since both the image fusion phase and the image scrambling phase independently demonstrate a nearly complete strategy for fusing multiple images into a single image and concealing visual features, integrating these two phases into a unified framework can significantly enhance the concealment of secret images. The fusion phase ensures that multiple images are blended at the pixel level, distributing information, while the scrambling phase introduces an additional layer of transformation, making it even more challenging to reveal the original images. By combining these two techniques, the resulting stego image achieves a higher level of visual feature concealment for the hidden images. This integrated approach ensures that secret images remain imperceptible to unauthorized viewers while still allowing precise reconstruction when processed through the appropriate decoding mechanisms.

5.3.1 Image Fusion using Pixel Wise Fusion and Four Nearest Neighbor combination with Scrambling

Since pixel-wise fusion in image fusion enhances granularity of blending, perceptual quality, information distribution, and data recovery ability, while Arnold scrambling effectively conceals visual features, these two techniques were integrated into a single encoder to generate a stego image devoid of any recognizable traces of the original images. On the decoding side, Reverse Arnold Scrambling successfully reconstructs the fused image, while four-nearest-neighbor refinement minimizes visual feature loss during reconstruction. By combining these two techniques into a single decoder, the process ensures accurate recovery of the original images with minimal loss of visual fidelity.

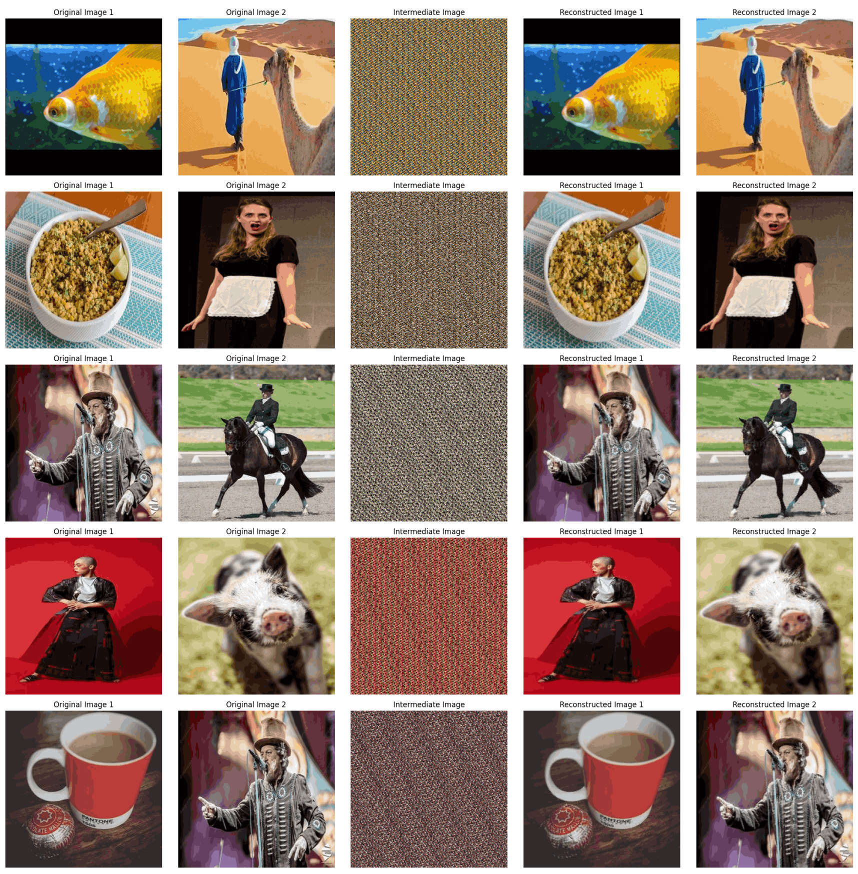


Figure 29: Pixel Wise Fusion with Arnold Scrambling

5.3.2 Image Fusion using LSB Manipulation combination with Scrambling

LSB manipulation was utilized in the encoding process to embed secret image information while ensuring minimal visual distortion in the stego image. Arnold scrambling was still applied to enhance feature concealment, making the hidden content more secure against unauthorized access. On the decoding side, Reverse Arnold Scrambling was used to recover the scrambled stego image, followed by LSB extraction to retrieve the original images. This approach aimed to balance imperceptibility and recoverability, ensuring that the reconstructed images maintained structural integrity while remaining securely hidden within the stego image.

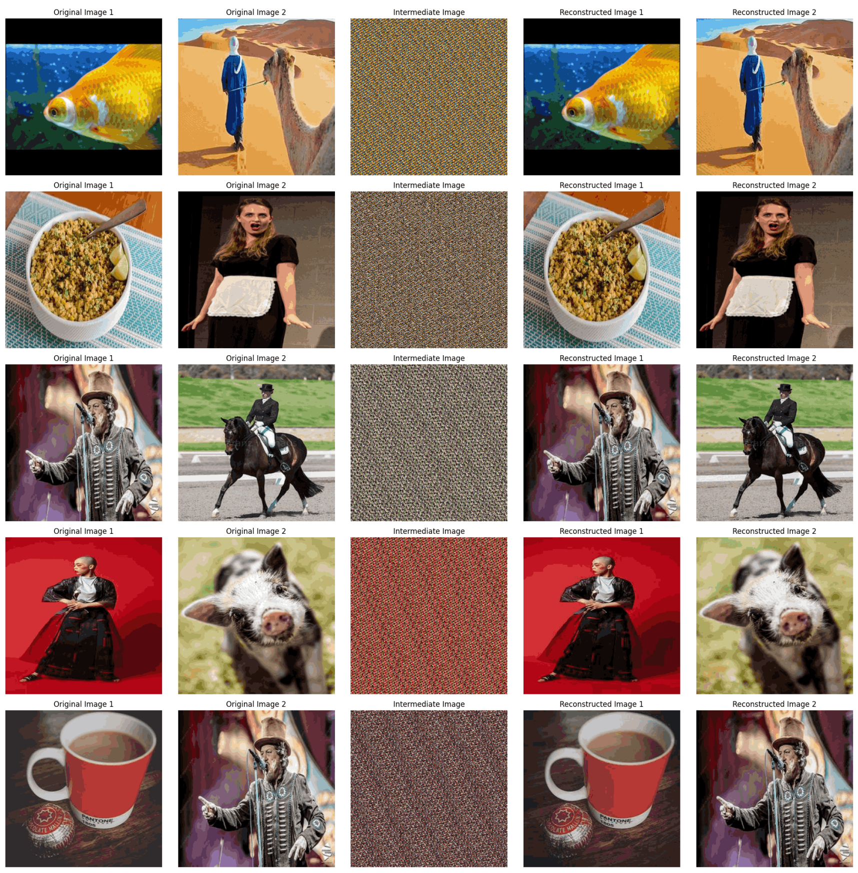


Figure 30: LSB Manipulation Fusion with Arnold Scrambling

5.3.3 Image Fusion using Pixel Wise Fusion, LSB manipulation and Four Nearest Neighbor combination with Scrambling

To enhance the effectiveness of coverless multi-image steganography, both pixel-wise manipulation and Least Significant Bit (LSB) manipulation techniques are utilized due to their strong fusion capabilities and high accuracy in image recovery. Since each method independently demonstrates effective results in compressing and reconstructing two secret images into a single intermediate representation, combining these techniques offers an opportunity to further increase the hiding capacity. This integrated approach enables the embedding of multiple images without the need for a predefined cover image, while still ensuring the quality and recoverability of the original content.

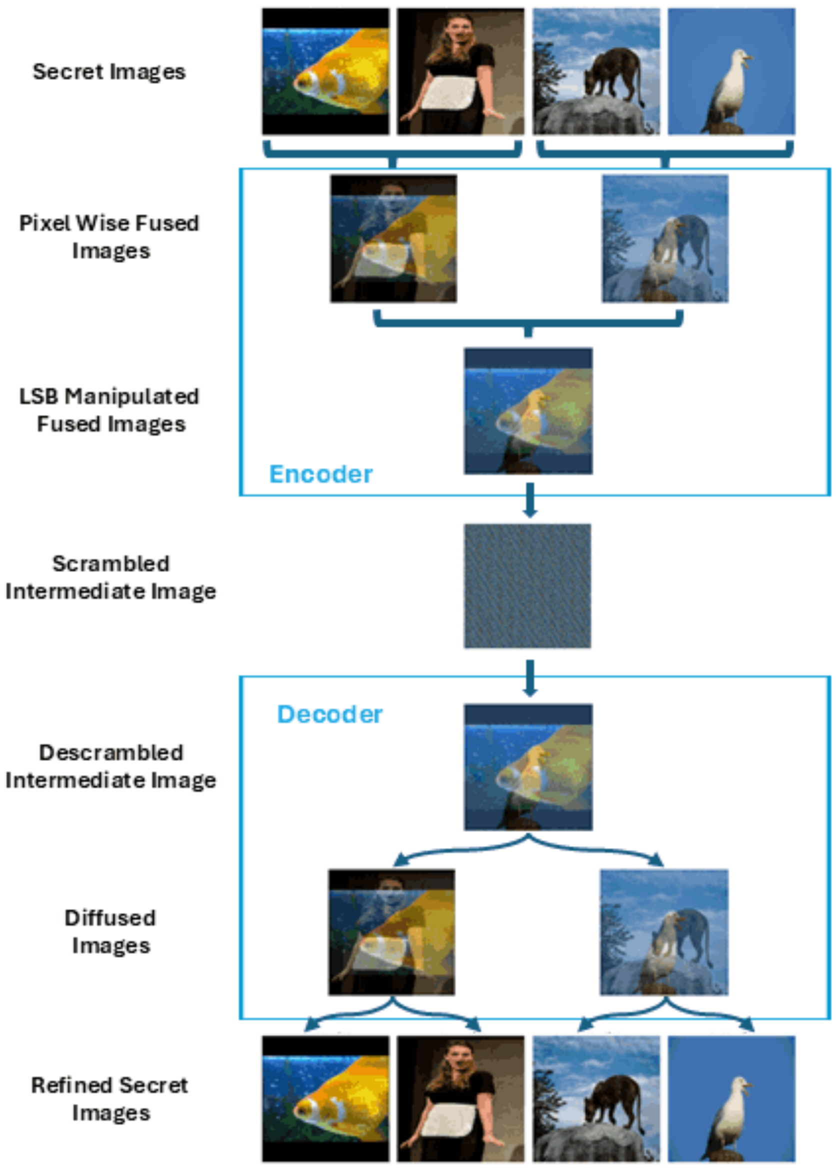

Secret Images
Pixel Wise Fused Images
LSB Manipulated Fused Images
Encoder
Scrambled Intermediate Image
Decoder
Descrambled Intermediate Image
Diffused Images
Refined Secret Images

Figure 31: LSB Manipulation and Pixel wise Manipulation with Arnold Scrambling

As illustrated in the figure 31, the system consists of two major stages: the Encoding Phase and the Decoding Phase.

- **Encoding Phase**
  Initially, pairs of secret images are merged through pixel-wise fusion, where corresponding pixel values from two images are interleaved to generate a visually meaningful fused image. This fused output is then further processed using the LSB technique, embedding another fused image within it at the bit level, increasing the capacity to store information. The result is a visually complex intermediate image. To conceal patterns and prevent detection, a scrambling algorithm is applied, producing a scrambled intermediate image that hides visual features of the original secret inputs.

- **Decoding Phase**

The decoding process begins by descrambling the intermediate image, reversing the applied scrambling pattern. The reverse LSB manipulation is then used to separate the two fused images originally embedded. Following this, pixel-wise diffusion techniques are employed to recover the original secret images from the fused pairs. Lastly, 4-Nearest Neighbor (4NN) refinement is used to address minor inconsistencies and improve the clarity of the reconstructed images, restoring them to a visually high-quality form without noticeable loss.

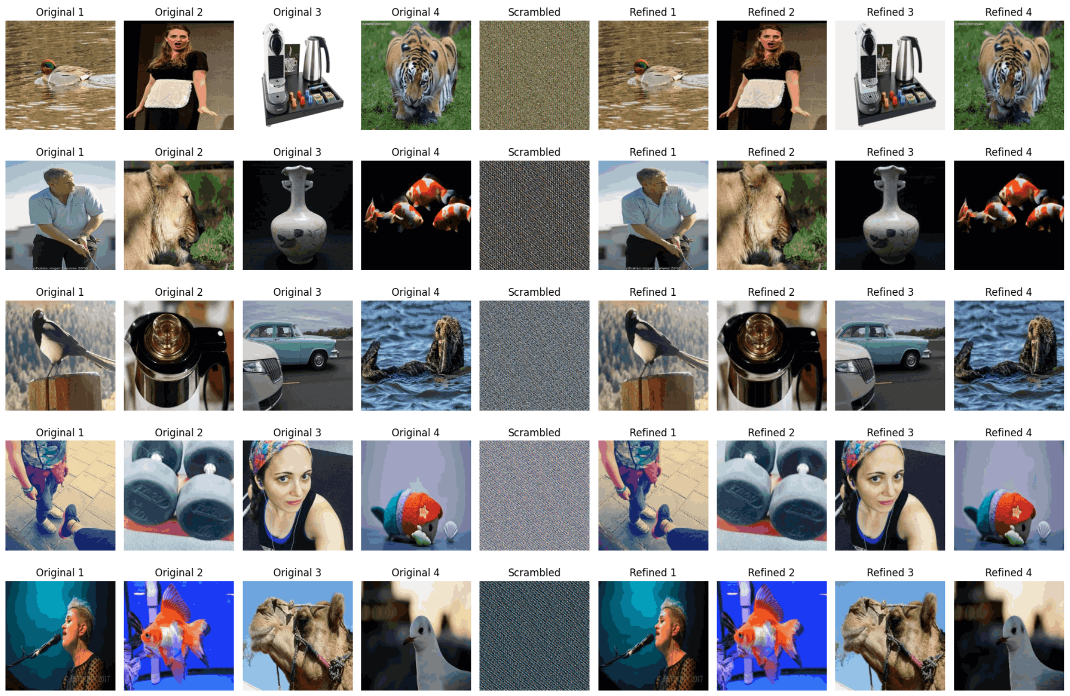


Figure 32: Combination of Pixel wise manipulation, LSB manipulation and Four Nearest Neighbor Refinement Experimental Results

Set	Image 1	Image 2	Image 3	Image 4
Set 1	0.038089	0.028266	0.054370	0.037963
Set 2	0.032272	0.035413	0.032377	0.014077
Set 3	0.031593	0.030789	0.033720	0.035847
Set 4	0.037244	0.030585	0.033351	0.029530
Set 5	0.026088	0.033396	0.038054	0.030229

Table 3: Mean Squared Differences for 5 Image Sets

The integration of pixel-wise fusion, LSB embedding, and scrambling achieves a balance between security, high embedding capacity, and reconstruction accuracy. By eliminating the need for a cover image and relying solely on fusion and manipulation techniques, this approach enhances the practicality and stealth of multi-image steganography. The use of 4NN refinement ensures that despite partial data recovery, the reconstructed images remain perceptually similar to the originals.

This pipeline effectively enables the secure embedding and accurate recovery of multiple secret images through a lightweight and coverless design.

5.3.4 Comparison between Fusion with Pixel Wise, Fusion with LSB Manipulation and Fusion with both Pixel wise and LSB

While both pixel-wise fusion and LSB manipulation offer unique advantages in image fusion, pixel-wise fusion proves to be more robust in scenarios involving additional trans formations, such as CycleGAN-based image generation and reconstruction. LSB manipulation primarily focuses on preserving the MSBs of both images while embedding them into the LSBs of the fused image. However, this approach introduces a critical vulnerability that, any slight alteration in the reconstructed LSBs due to transformations can significantly impact the recovery process, potentially leading to an entirely incorrect original image. In contrast, pixel-wise fusion interleaves pixels from both images at a fine-grained level, ensuring a more balanced and evenly distributed representation of information across the entire image. This method minimizes the risk of erroneous reconstructions caused by small perturbations, making it more resilient to transformations and processing variations. Additionally, pixel-wise fusion ensures that features from both images are consistently retained, improving perceptual quality and making it less susceptible to distortions introduced during subsequent processing steps. Thus, for applications where post-processing transformations are involved, pixel-wise fusion offers a more reliable and visually coherent approach compared to LSB manipulation.

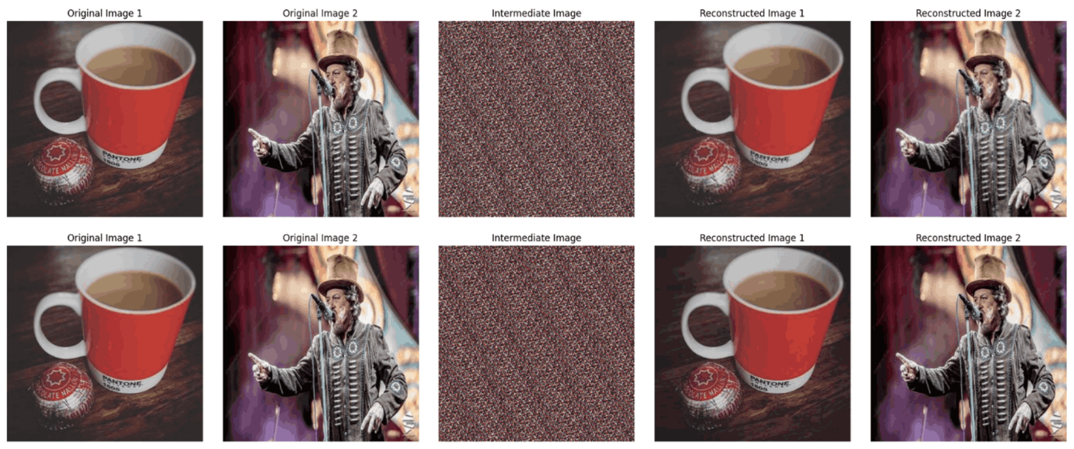


Figure 33: Comparison between Pixel Wise Fusion and LSB Manipulation
Row 1: Pixel Wise manipulation and Four Nearest Neighbor Refinement with Arnold Scrambling, Row 2: LSB Manipulation and Four Nearest Neighbor Refinement with Arnold Scrambling

	Pixel Wise	**LSB Manipulation**	**Pixel + LSB**
Image 1	0.015325	0.028741	0.020960
Image 2	0.006600	0.031398	0.026985
Image 3	-	-	0.034361
Image 4	-	-	0.038224
Total Error	0.021925	0.060139	0.120530
Avg Error	0.010963	0.030070	0.030133

Table 4: Transposed comparison of pixel wise, LSB manipulation, and combined approaches.

Furthermore, the combination of both techniques achieves higher embedding capacity while allowing scrambling algorithms to function effectively. This hybrid approach is particularly suitable for algorithmic implementations that demand both security and high capacity. However, when combining these two techniques, each individual pixel value becomes more significant. As a result, applying further generative methods, such as GAN-based transformations, can lead to inaccurate reconstructions. Even minor changes to the intermediate image values can compromise the entire recovery process, making the model highly sensitive to transformation-based alterations.

5.3.5 Vulnerability of Arnold Transformation and the solution

The Arnold scrambling transformation is widely used for image encryption and feature concealment due to its ability to reorder pixel positions in a deterministic yet seemingly chaotic manner. When combined with an image fusion phase, it further enhances security by obscuring the original image content, making it nearly impossible to visually identify the embedded secret images. However, a significant vulnerability of the Arnold transformation lies in its reversibility. Since it follows a periodic pattern, repeated applications of the scrambling algorithm or its inverse, the reverse Arnold transformation can eventually regenerate the original fused image. This unintended periodicity weakens security, as an adversary with knowledge of the transformation parameters could iteratively reverse the process, revealing the visual features of both secret images.
To mitigate this vulnerability, it is crucial to integrate unique and adaptive transformation mechanisms that do not exhibit predictable periodicity. One effective solution is the introduction of dynamic key-based scrambling algorithms, where the transformation parameters vary for each instance. This variability enhances security by preventing attackers from identifying consistent patterns. Additionally, the incorporation of generative models, such as CycleGAN, introduces a layer of synthetic data generation during the transformation process. These models are capable of learning complex mappings between image domains, ensuring that the scrambled image retains essential information while deviating significantly from any predictable periodic structure. By integrating such unique generators capable

of synthesizing altered yet recoverable scrambled images, the system ensures that visual features remain concealed while still allowing accurate reconstruction when needed. This approach significantly strengthens the robustness of image security systems, preventing unauthorized access to hidden information while maintaining high fidelity in reconstruction for legitimate use cases.

5.4 Steganographic key based coverless multi image steganography

The steganographic method begins by taking four secret images and a steganographic key phrase as input. This key phrase plays a crucial role in determining the security and uniqueness of the transformation process. The system uses the key to generate a unique three-digit number by converting the text into a non-reversible numeric form using a hashing mechanism. This ensures that the transformation is one-way consistent meaning, the same key will always produce the same number, but the process cannot be reversed to retrieve the key, thereby enhancing security.

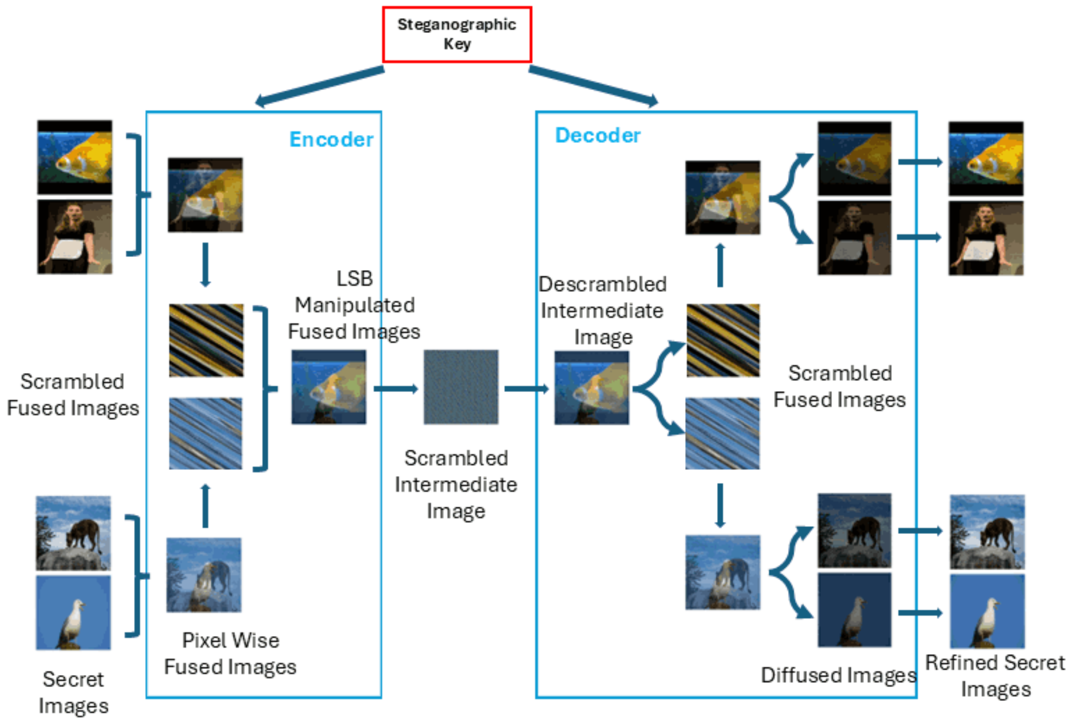


Figure 34: Steganographic key based coverless multi image steganography architecture

5.4.1 Encoding Process

The encoding process is designed to securely transform four secret images into a single scrambled intermediate image through a sequence of deterministic and key-driven operations. It leverages pixel-wise fusion, LSB manipulation, and scrambling all governed by a steganographic key.

- **Phase 1** – Pixel-Wise Fusion: The first step involves fusing pairs of secret images using a pixel-wise fusion strategy. This method fuses two images by alternating their pixel values in a predefined interleaving pattern, resulting in two new fused images. This fusion technique retains a balanced amount of visual and data features from both source images.

- **Phase 2** – Independent Scrambling: After fusion, each of the two fused images is scrambled independently. The number of scrambling rounds applied to each image is determined by the unique three-digit number generated from the steganographic key. This means that even if two images are processed together, their scrambling paths differ, reinforcing security and making unauthorized reconstruction highly challenging.

- **Phase 3** – LSB-Based Fusion: The scrambled fused images from Phase 2 are then fused using the Least Significant Bit (LSB) manipulation technique. This phase embeds one image into another at the bit level, resulting in a single intermediate image that subtly holds the information of both secret images while appearing visually inconspicuous.

- **Phase 4** – Intermediate Image Scrambling: The intermediate image is then scrambled again, this time as a whole. The number of scrambling rounds is again derived from the same three-digit number obtained using the steganographic key. This added layer of scrambling enhances the obfuscation of the hidden data, making the final scrambled intermediate image highly secure.

5.4.2 Decoding Process

The decoding process aims to faithfully reconstruct the four secret images from the scrambled intermediate image using the same steganographic key that was used during encoding. Since the encoding process is deterministic but non-reversible in terms of key derivation, the same key ensures consistent transformation parameters for decoding.

- **Phase 5** – Reverse Scrambling of Intermediate Image: The scrambled intermediate image is subjected to a reverse scrambling algorithm to retrieve the original intermediate image. The number of reverse scrambling rounds matches those used in Phase 4 of the encoding process and is determined using the three-digit number derived from the steganographic key. This step is crucial in unraveling the final layer of pixel scrambling applied during encoding.

- **Phase 6** – Reverse LSB Manipulation: Once the intermediate image is recovered, reverse LSB manipulation is performed to extract the two previously fused and scrambled images. This operation separates the bit-level data that was embedded during the LSB-based fusion phase, revealing the two fused components.

- **Phase 7** – Independent Reverse Scrambling of Fused Images: Each of the extracted fused images is then reverse scrambled independently. Similar to the encoding phase, the number of rounds is determined using the unique three-digit number generated from the key. Because each fused image was scrambled differently during encoding, reversing the scrambling process independently for each one ensures accurate reconstruction.

- **Phase 8** – Pixel-wise Defusion into Original Images: After descrambling, the fused images still contain information from two secret images each, fused using pixel-wise interleaving. This step diffuses (or separates) each fused image back into the two original secret images by reversing the pixel-wise fusion pattern. The output is four distinct secret images, partially reconstructed.

- **Phase 9** – 4-Nearest Neighbor (4NN) Refinement: Finally, a 4-Nearest Neighbor (4NN) refinement technique is applied to the recovered images. This method enhances image quality by filling in gaps or smoothing inconsistencies based on surrounding pixel values. It helps restore finer details and improve the perceptual quality of the images, resulting in outputs that are visually closer to the originals.

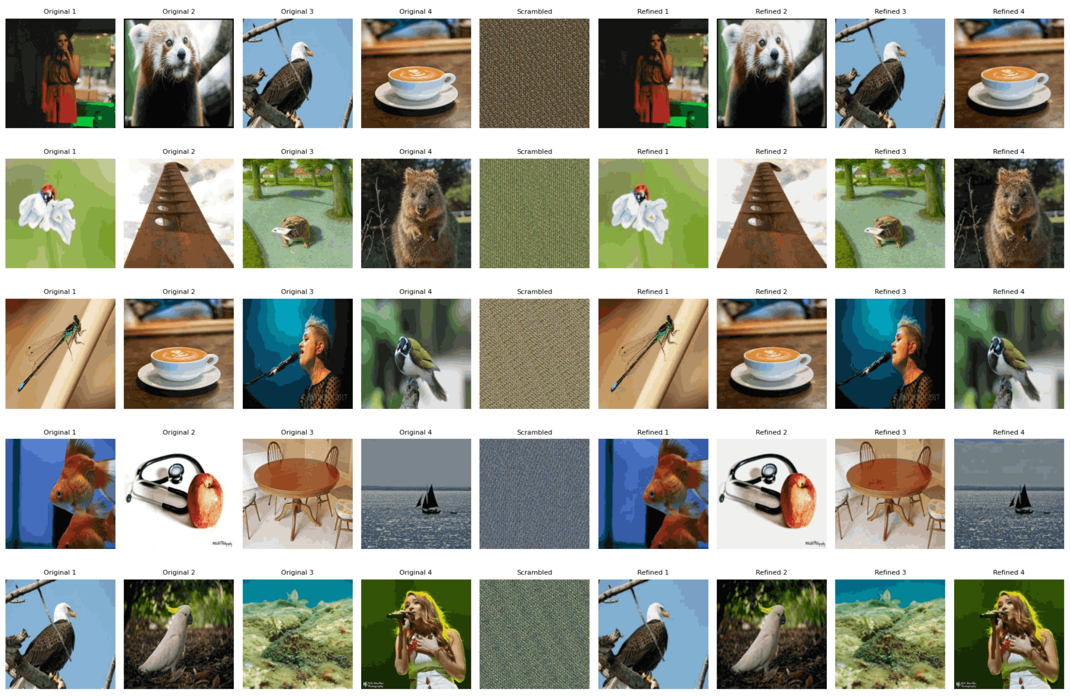


Figure 35: Steganographic key based Image Fusion Results

The proposed steganographic framework introduces a highly secure and structured approach to hiding multiple secret images within a single intermediate image. By employing a non-reversible key-based transformation mechanism, along with independent multi-round scrambling and pixel-wise fusion techniques, the process ensures both high-level concealment and fidelity in reconstruction. The integration of LSB manipulation and 4-Nearest Neighbor refinement further enhances the robustness and visual quality of recovered images. Most importantly, the use of a unique steganographic key for guiding each transformation stage ensures that only authorized parties with the correct key can accurately decode and retrieve the hidden content. This comprehensive method not only protects sensitive visual data during transmission or storage but also makes it a reliable solution for secure image-based information hiding.

5.5 Explore the behavior of CycleGAN

Figure 36 illustrates a CycleGAN framework, where two generators are used to perform bidirectional image transformation. "Generator 1" transforms source images into target images, and "Generator 2" reverses this process by transforming target images back into the original source images. This cycle ensures that the generated target images retain the key characteristics of the source images, while also enabling the reconstruction of the source images from the generated ones.

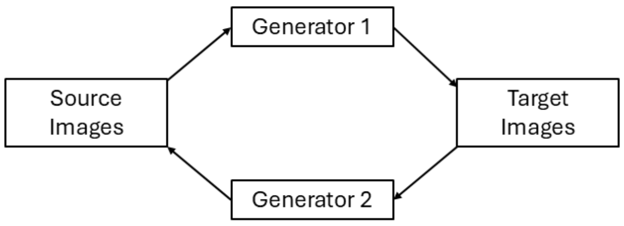


Figure 36: CycleGAN for Image Transformation

5.5.1 Behavior of CycleGAN according to Different Sizes of Datasets

This section explores the impact of dataset size on the performance of CycleGAN in transforming secret images into artistic representations while keeping the same batch size of 4 and same number of training epochs of 100 epochs. By training the model on datasets of varying sizes, from as few as 6 images to a large set of 6300 images, the study examines how data availability influences generalization, overfitting, and reconstruction quality. The findings highlight the advantages of larger datasets in improving transformation stability and visual fidelity while demonstrating the challenges posed by limited training samples.

- **Single Secret Images to Paint Arts Transformation (Dataset size -6)**
  This small dataset serves as a proof-of-concept to validate the feasibility of transforming single secret images into artistic representations. Due to the

limited number of images, the model may overfit, leading to less generalization.

- **Single Secret Images to Paint Arts Transformation (Dataset size -250)**
  This dataset provides a more diverse set of images, allowing the model to learn better feature mappings between secret images and their artistic transformations. While overfitting is reduced compared to the smaller dataset, the model may still require further optimization to generalize effectively across various image styles.
- **Single Secret Images to Paint Arts Transformation (Dataset size -500)**
  With 500 images, the model benefits from a broader range of training samples, leading to improved robustness in generating artistic representations. The increased dataset size enhances the model's ability to generalize and capture intricate details, ensuring more stable and visually appealing transformations while reducing dependency on specific patterns in the data.
- **Single Secret Images to Paint Arts Transformation (Dataset size -6300)**
  A significantly large dataset of 6300 images allows the model to achieve high-quality artistic transformations while maintaining the structural integrity of the secret images. The diverse data distribution strengthens the model's generalization ability, enabling it to produce visually coherent and stylistically diverse outputs. Advanced training strategies such as adversarial loss and perceptual loss can be effectively applied to further refine the results.

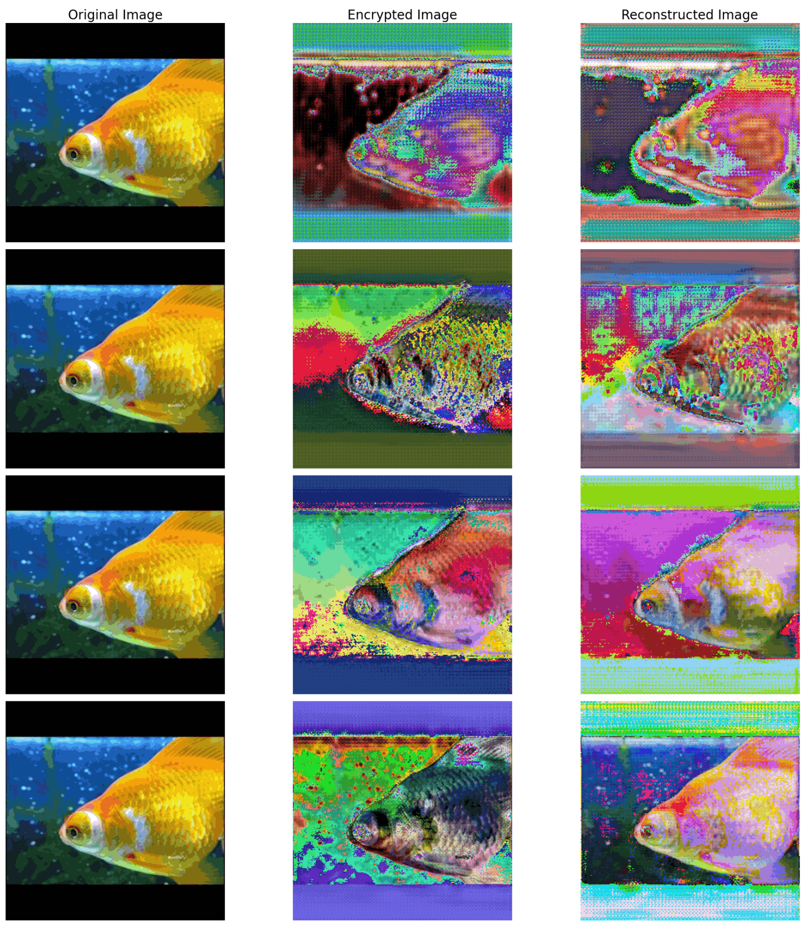


Figure 37: Image Transformation over different sizes of dataset (Row 1: Dataset size - 6, Row 2: Dataset size - 250, Row 3: Dataset size - 500, Row 4: Dataset size - 6000 & Column 1: Original Image, Column 2: Generated Image, Column 3: Reconstructed Image)

Figure 37 illustrates the steganographic performance of CycleGAN across different dataset sizes, where models trained on varying amounts of data are evaluated on unseen secret images. Each row represents a different dataset size, with the first row showing results for a dataset of 6 images, the second row for 250 images, the third row for 500 images, and the final row showcasing the performance on a much larger dataset of 6300 images. This allows for a comparison of how dataset size influences the model's ability to transform secret images into artistic representations.

Dataset Size	**Mean Squared Error**	**Absolute Mean Pixel Error**
6 Images	0.176572	0.337011
250 Images	0.336253	0.512487
500 Images	0.061227	0.191331
6300 Images	0.012110	0.081347

Table 5: Error values on Image Transformation over different sizes of datasets

Table 5 presents the error values when comparing the original images to the recovered images, providing an assessment of the model's performance in terms of

reconstruction accuracy. The table displays the Mean Squared Error and Absolute Mean Pixel Error for different dataset sizes, allowing a detailed comparison of the reconstruction quality across varying amounts of training data. The error values highlight how the model's ability to recover the original images improves as the dataset size increases, with lower error values observed for larger datasets, indicating better generalization and more accurate reconstructions. This table offers insights into the relationship between dataset size and model performance in terms of error minimization.
Through extensive experimentation, it was observed that dataset size plays a crucial role in the performance of CycleGAN. Specifically, when using small datasets with fewer than 3000 images per domain, the quality and stability of transformations were noticeably impacted.

- **Challenges with Small Datasets**
  With limited data, CycleGAN struggled to generalize the transformation patterns effectively. This often resulted in poor quality outputs with inconsistent mappings between the source and target domains.

- **Benefits of Larger Datasets**
  In contrast, with datasets containing over 3000 images per domain, CycleGAN achieved more stable and realistic transformations. With sufficient data, CycleGAN could retain structural details in transformations and capture domain-specific features, leading to more realistic and coherent results.

- **Effect on Reversibility**
  In experiments with larger datasets, CycleGAN also demonstrated improved reversibility, where transformed images could be more effectively mapped back to their original domain compared to smaller datasets.

In summary, dataset size directly affects CycleGAN's ability to perform effective transformations between domains. To achieve high-quality results with CycleGAN, it is recommended to have larger databases with accurate images from each domains. This ensures that the model has sufficient data to learn the complex mappings required for stable and consistent transformations, particularly in tasks involving intricate or structurally diverse domains.

5.5.2 Fused Images into Paint Arts

In this experiment, CycleGAN was employed for the image transformation phase, focusing on evaluating its effectiveness in converting fused images into paint art and subsequently reconstructing them. Instead of incorporating a scrambling phase, the study aimed to directly analyze CycleGAN's ability to learn the mappings between the fused image domain and the paint art domain. The source domain is consisted of fused images generated through pixel-wise fusion, while the target

domain is comprised paint art images. By training CycleGAN in this setting, the goal was to assess its capability in preserving critical visual features while transforming fused images into artistic representations and accurately recovering them back to their original form.
The results revealed that while CycleGAN successfully mapped the fused images into paint art, the reconstructed images suffered from significant quality degradation. Despite the loss in fine details and original texture, the recovered images retained the essential structural features of the original inputs, making them recognizable. This indicates that while CycleGAN can effectively learn transformations between these domains, additional refinements—such as improved loss functions, perceptual loss integration, or adversarial regularization—may be required to enhance reconstruction quality and better preserve the integrity of the original fused images.

Parameter	**Value**
Dataset Size	6300 Fused Images
Batch Size	4
Learning Rate	0.0002
Lambda Value for Identity Loss	0
Lambda Value for Cycle Loss	4
Number of Epochs	100
Total Time (seconds)	35920
Total Time (approx.)	10 hours

Table 6: Training Configuration and Time

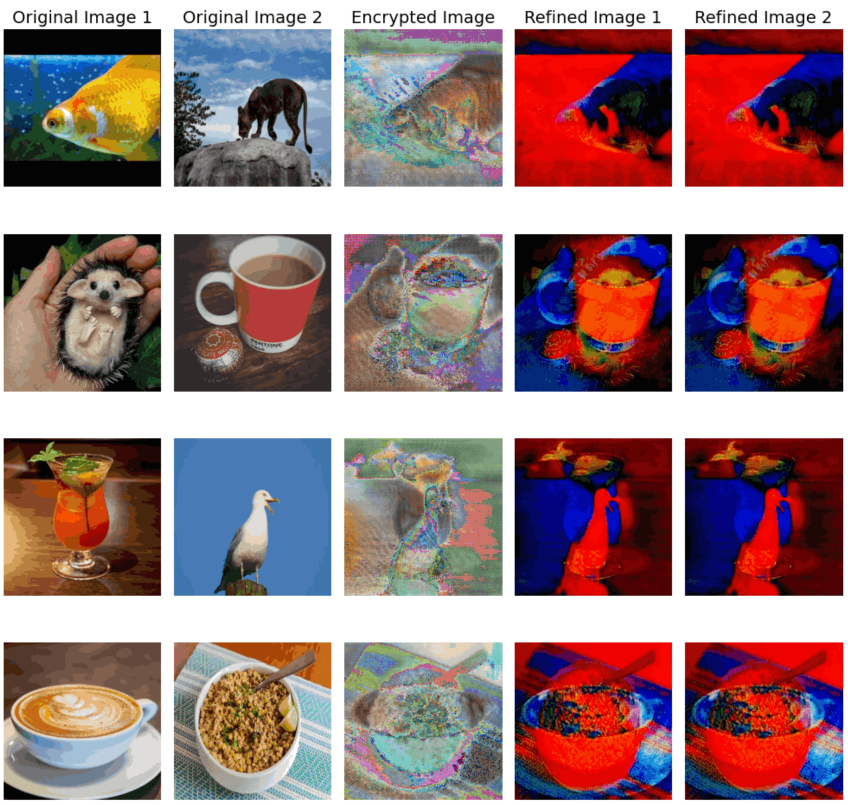


Figure 38: CycleGan on Fused Images into Paint Arts

The results shown in Figure 38 highlight the limitations of CycleGAN in accurately reconstructing the original images from their fused representations. While the transformation into the paint art domain was successful, the recovery process did not preserve the original colors, leading to significant color distortions in the reconstructed images. Additionally, the separation of the two secret images was not achieved as expected. This is primarily due to the model's inability to recover pixel values in a structured manner that aligns with the designated pixel positions for each secret image. As a result, instead of accurately defusing the two images, the reconstructed outputs exhibit overlapping structures where both secret images share a similar visual pattern with altered color distributions. This suggests that while CycleGAN learns meaningful transformations, it lacks the precision required for controlled pixel-wise recovery, necessitating further improvements in the reconstruction process, potentially through architectural modifications or loss function refinements tailored for pixel-accurate retrieval.

Adversarial loss alone cannot handle the reconstruction of fused image pixel values in a structured manner because it primarily focuses on making the generated image appear visually realistic to a discriminator, rather than enforcing precise spatial or pixel-level accuracy. In the case of fused images, where pixel positions correspond to specific regions or patterns from multiple secret images, adversarial loss does not provide guidance to preserve or recover these designated positions. As a result, it fails to ensure that the reconstruction aligns with the original pixel-wise layout required for accurately separating and restoring each embedded image.

5.5.3 Fused and Scrambled Images into Paint Arts

In this experiment, CycleGAN was utilized for the image transformation phase, focusing on evaluating its effectiveness in converting scrambled fused images into paint art and subsequently reconstructing them. The source domain consisted of fused images that underwent four iterations of Arnold scrambling, while the target domain comprised paint art images. The study aimed to analyze CycleGAN's capability to learn mappings between scrambled images and artistic representations without any prior knowledge of the original fused images. By training the model in this setting, the objective was to determine whether CycleGAN could effectively translate scrambled inputs into meaningful outputs while still retaining the potential for reconstruction.

Parameter	**Value**
Dataset Size	6300 Fused and Scrambled Images
Batch Size	4
Learning Rate	0.0002
Lambda Value for Identity Loss	0
Lambda Value for Cycle Loss	4
Number of Epochs	100
Total Time (seconds)	34286
Total Time (approx.)	10 hours

Table 7: Training Configuration and Time

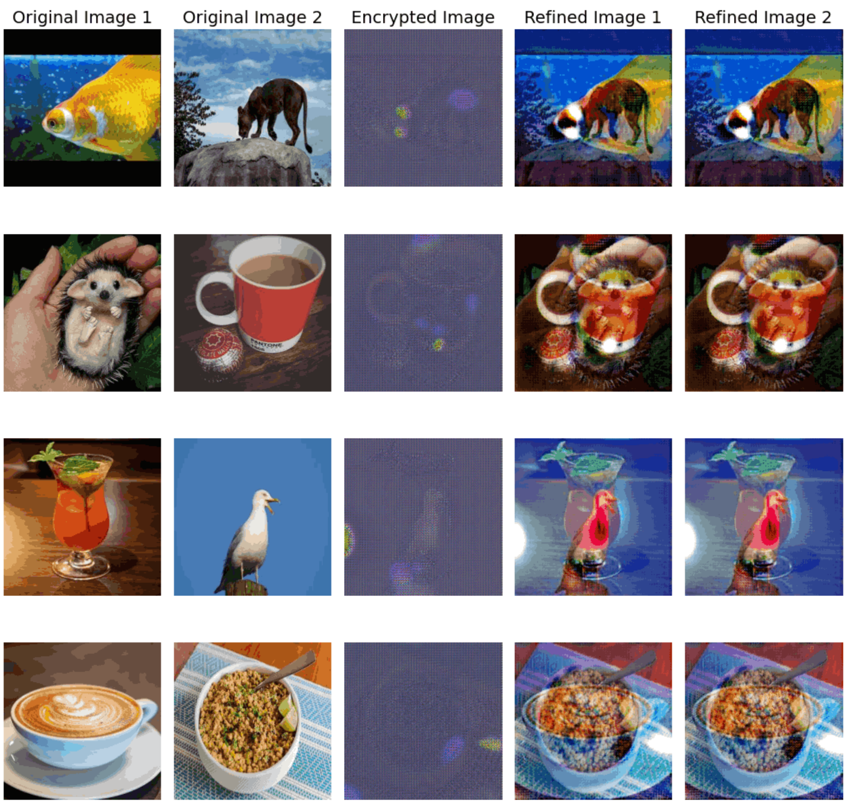


Figure 39: CycleGan on Fused Scrambled Images into Paint Arts with no iterations of scrambling

The results in Figure 39 demonstrated that while CycleGAN was capable of concealing the original features of secret images when no scrambling iterations were applied, the reconstructed outputs remained recognizable, albeit with significant distortions. Despite the overall degradation in image quality, key structural elements of the original content were still preserved, making the recovered images partially identifiable. This suggests that CycleGAN can extract meaningful visual features even from transformed images. However, without prior knowledge of the original secret images, the reconstruction process does not occur as intended.
A major limitation observed was the failure to achieve the expected separation of the two secret images. This issue arises primarily due to the model's inability to recover pixel values in a structured manner that aligns with the designated pixel positions for each secret image. Consequently, instead of accurately defusing the fused images into their respective original images, the reconstructed outputs exhibit overlapping structures, where both secret images share similar visual patterns of a fused image. This indicates that while CycleGAN is effective in learning complex transformations, it lacks the precision needed for controlled pixel-wise recovery. Addressing this challenge would require improvements in the reconstruction process, such as refining the model's architecture, integrating the image fusion phase to the image transformation phase during training process, and defining new loss functions based on original secret images and reconstructed secret images.
Originally, the model was trained using fused images followed by 4 iterations of

scrambling rounds, and paint-art images. However, during evaluation, while the fused images without scrambling iterations were reconstructed effectively, the performance significantly deteriorated for fused images that had undergone 4 or 8 iterations of scrambling. Figure 40 and Figure 41 displays the model evaluation over fused images which applied 4 iterations of scrambling and 8 iterations of scrambling respectively.

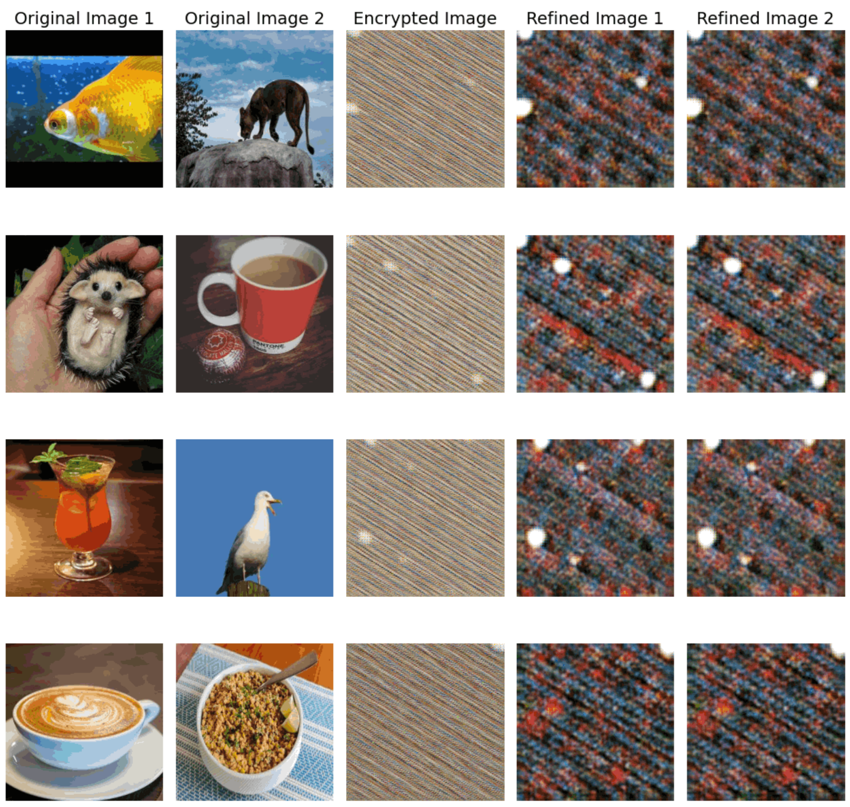


Figure 40: CycleGan on Fused Scrambled Images into Paint Arts with 4 iterations of scrambling

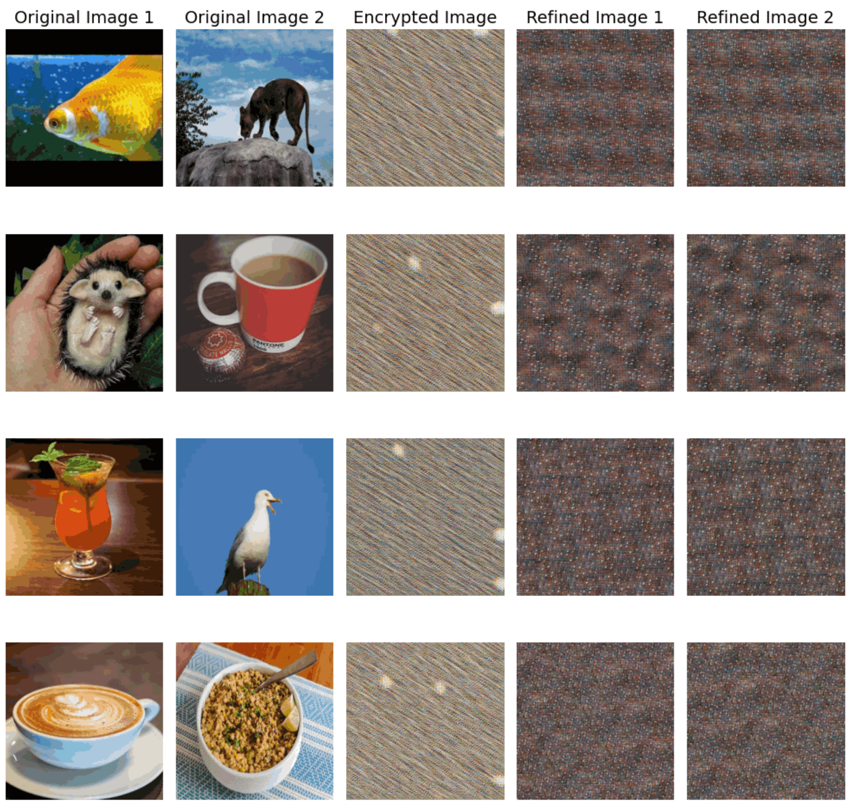


Figure 41: CycleGan on Fused Scrambled Images into Paint Arts with 8 iterations of scrambling

Since the generators are trained to transform between scrambled images and paint art representations, the generator responsible for producing the scrambled image primarily focuses on generating outputs that appear visually realistic to the discriminator. This objective, driven by adversarial loss, emphasizes overall appearance rather than enforcing strict spatial or pixel-level accuracy. Consequently, the generator fails to reconstruct the original scrambled structure with the necessary precision, particularly when multiple secret images are fused. This highlights the limitation of relying solely on adversarial loss and emphasizes the need for a more advanced training architecture—one that incorporates structure-preserving constraints or additional loss functions tailored for pixel-wise alignment and accurate spatial reconstruction.

5.6 Image Fusion (Pixel wise manipulation with Four Nearest Neighbor Refinement) and Image Transformation (CycleGAN)

The experiment was conducted using a small dataset of six secret images due to computational performance constraints. The model was trained with a batch size of 2, a learning rate of 0.0002, and a total of 100 epochs. Identity loss and cycle loss were assigned weights of 4 and 2, respectively, to balance the objectives of the model. The entire training process took approximately 14.4 hours (51870 seconds) to complete.

Parameter	**Value**
Dataset Size	6 Secret Images
Batch Size	2
Learning Rate	0.0002
Lambda Value for Identity Loss	4
Lambda Value for Cycle Loss	2
Number of Epochs	100
Total Time (seconds)	51870
Total Time (approx.)	14.4 hours

Table 8: Training Configuration and Time

Each training iteration begins with two source images, which are fused using pixel-wise fusion to create a single input image during training. This fused image is then processed by a GAN-based generator that attempts to map it into the domain of paint arts. To ensure that the generated image resembles actual paintings, a discriminator is trained in parallel to distinguish between real paint art images and the synthetic stego images. The output of this process is the stego image, which serves as the encoded representation of the secret images.
Following the generation of the stego image, another generator is tasked with reconstructing the fused image from it. This stage involves an additional discriminator, which differentiates between real and reconstructed fused images. To enhance training stability, an intermediate loss is computed between the regenerated fused image and the initially fused image to encourage accurate reconstruction.
Once the fused image is reconstructed, the model proceeds to recover the original secret images. This is achieved through pixel-wise diffusion followed by four-nearest-neighbor refinement, which corrects any distortions introduced during the transformations. The final output is compared to the original secret images using refinement loss to measure reconstruction accuracy.

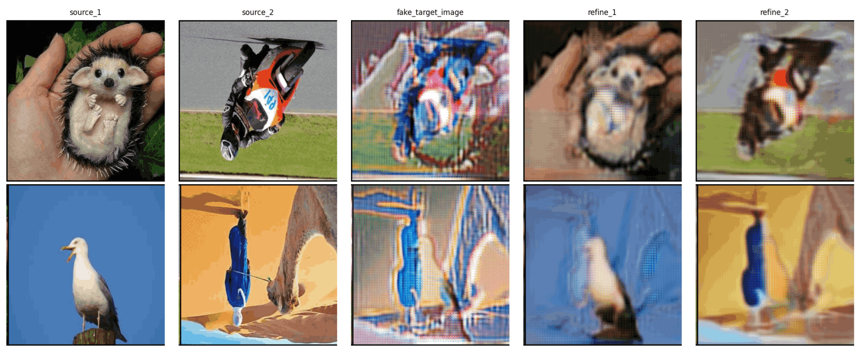


Figure 42: Image Fusion and CycleGan (Saved Images during training)

Figure 42 demonstrates the results of image fusion and CycleGAN-based transformations during training. The stego image, intended to conceal the visual features of the secret images, reveals notable traces of the original content. While the generated fake target image exhibits a transformed style, the structural details of the source images are still distinguishable. This indicates that the model has not fully learned to obscure the visual identity of the secret images, potentially due to limitations in dataset size and the complexity of the generator. As a result, the reconstructed images retain significant portions of the original features, highlighting the need for further improvements in the model's ability to fully disguise the secret images within the stego representation.

5.7 Image Fusion (Pixel wise manipulation with Four Nearest Neighbor Refinement), Image Scrambling (Arnold Scrambling) and Image Transformation (CycleGAN)

The experiment was designed to evaluate the effectiveness of the proposed multi-image steganographic model based on CycleGAN, incorporating image fusion and scrambling techniques. Due to computational constraints, only six images per domain were used throughout the training process. Despite this limitation, the model demonstrated a significant improvement in visual feature concealment within the generated stego images.

The dataset consisted of six secret images carefully selected for training. To enhance security, each image underwent four Arnold transformations, ensuring that the secret images were sufficiently distorted before being embedded. A small batch size of 2 was used to maintain stability during training, while the learning rate was set to 0.0002, a standard choice in CycleGAN training, ensuring smooth optimization. The loss function weights were configured with identity loss at 4 and cycle consistency loss at 2 to balance the learning process.

Parameter	Value
Dataset Size	6 Secret Images
Scrambling Iterations	4 Arnold Iterations
Batch Size	2
Learning Rate	0.0002
Lambda Value for Identity Loss	4
Lambda Value for Cycle Loss	2
Number of Epochs	100
Total Time (seconds)	73644
Total Time (approx.)	20.5 hours

Table 9: Training Configuration and Time

Within this training architecture, each iteration begins with the selection of two source images, which are then fused using pixel-wise fusion. The fused image undergoes four iterations of Arnold scrambling to enhance security and obfuscate its visual features. The scrambled image is then passed through a GAN generator,

which aims to map it into the domain of paint arts, generating a synthetic image. Simultaneously, a discriminator is trained to distinguish whether the generated image is a synthetic stego image or an actual paint art. This generated image serves as the stego image, which carries the concealed information.
Using the stego image, a second generator attempts to reconstruct the scrambled image. Another discriminator is employed here to differentiate between real scrambled images and those generated by the model. At this stage, an intermediate loss is calculated by comparing the regenerated scrambled image with the initially fused scrambled image to ensure proper reconstruction. Once the scrambled image is reconstructed, it undergoes four reverse Arnold scrambling iterations to regenerate the fused image.
To retrieve the original secret images, pixel-wise diffusion is applied, followed by four-nearest-neighbor refinement to enhance the reconstructed images' quality. The final refinement step ensures that the recovered secret images closely resemble the original ones. A refinement loss is calculated by comparing the reconstructed refined images with the original secret images, ensuring that the model learns to preserve and restore the hidden information accurately.

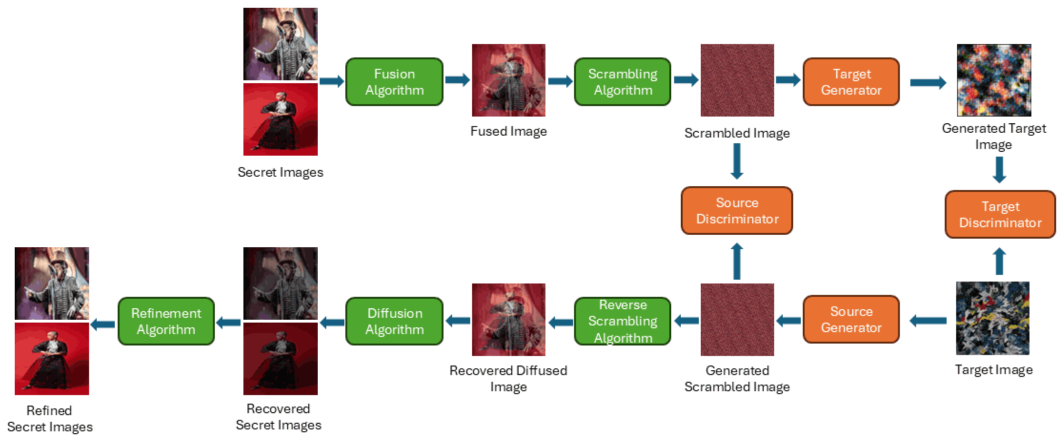


Figure 43: Training Architecture

Upon analyzing the generated stego images, it was observed that they exhibited strong visual concealment, effectively hiding secret image features. No identifiable visual details from the original secret images were present in the stego images, making them highly secure. This level of visual obfuscation significantly improved the ability to protect sensitive content from unauthorized detection.

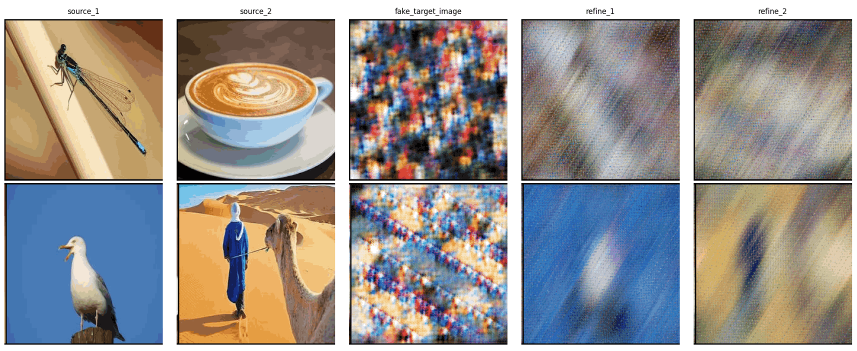


Figure 44: Pixel Wise Image Fusion, Scrambling and CycleGAN combination (Saved Images during training)

In terms of reconstruction performance, the CycleGAN model showed improved convergence when attempting to reconstruct secret images. However, the reconstructed outputs were highly blurred, reducing their clarity and making precise image retrieval challenging.
During the evaluation, the model failed to generate stego images as expected. Instead, it produced black-and-white distorted versions of the fused secret images. This distortion affected the reconstruction process, making it difficult to retrieve original images with high fidelity.

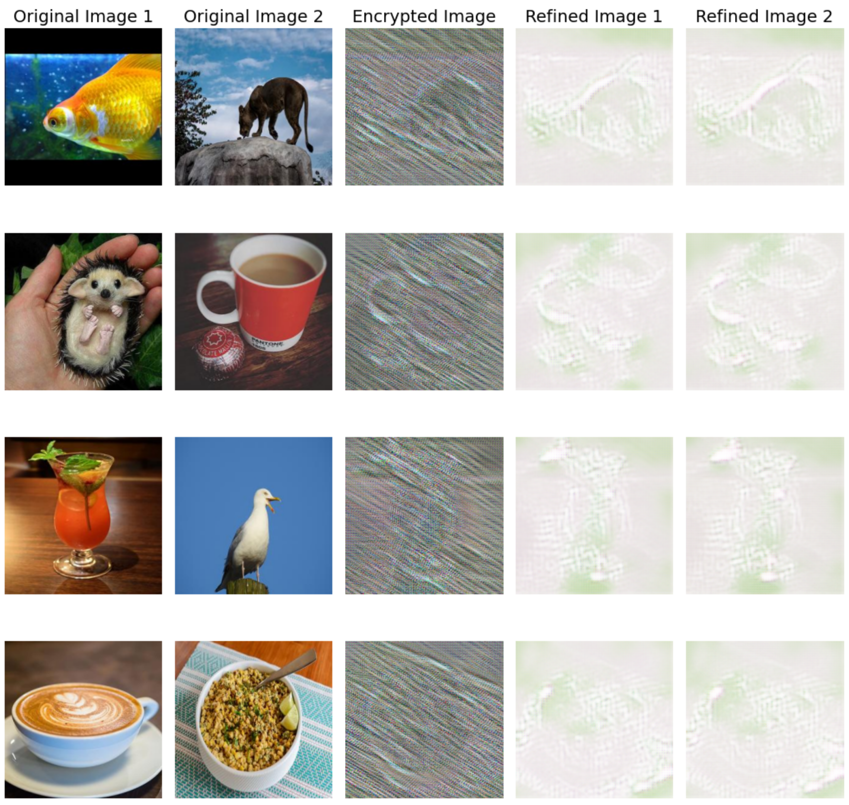


Figure 45: Evaluate Performance

Although the proposed model successfully achieved strong visual feature concealment, it struggled to balance steganographic security with image reconstruction quality. The black-and-white distortions in the generated stego images impacted the model's ability to reconstruct the original secret images effectively. This suggests that further refinements, such as improving the loss functions, adjusting the CycleGAN architecture, or increasing the dataset size, are necessary for better performance.

5.8 Image Fusion (Pixel wise manipulation), Image Scrambling (Arnold Scrambling) and Image Transformation (CycleGAN) with Improved Loss Functions and Complex Generator Architecture

The experiment involved key parameters that influenced the model's performance. The dataset consisted of only 6 secret images per domain due to computational constraints, limiting generalization. The model applied 4 Arnold scrambling iterations to conceal the secret images, with a batch size of 2 for efficient training. A learning rate of 0.0002 was chosen to allow gradual weight updates without overshooting, while the lambda values for identity loss and cycle loss were set to 4 and 2, respectively. The identity loss lambda value encouraged preservation of the original images during transformations, while the cycle loss lambda value ensured accurate reconstruction of the secret images. These parameters, while designed to balance image transformation and feature concealment, were constrained by the small dataset, limiting the model's ability to generalize.

Parameter	**Value**
Dataset Size	6 Secret Images
Scrambling Iterations	4 Arnold Iterations
Batch Size	2
Learning Rate	0.0002
Lambda Value for Identity Loss	4
Lambda Value for Cycle Loss	2
Number of Epochs	100
Total Time (seconds)	152853
Total Time (approx.)	42.46 hours

Table 10: Training Configuration and Time

Within this training architecture, each iteration begins by selecting two source images, which are then fused using pixel-wise fusion. This fused image undergoes four iterations of Arnold scrambling to enhance security by disrupting its pixel arrangement. The scrambled image is then fed into a GAN generator, which attempts to map it into the domain of paint arts, generating a synthetic stego image. To ensure the quality and realism of this transformation, a discriminator is trained simultaneously to differentiate between actual paint art images and the generated stego images. The discriminator's feedback helps the generator improve its ability

to create convincing stego images.
Once the stego image is generated, a second generator attempts to reconstruct the scrambled image from it. During this process, an intermediate loss is calculated between the regenerated scrambled image and the original fused scrambled image to ensure that the network learns to retain essential features. The reconstructed scrambled image then undergoes four reverse Arnold scrambling iterations to retrieve the fused image, reversing the pixel displacement applied earlier.
To reconstruct the original secret images, the fused image undergoes pixel-wise diffusion, followed by four-nearest-neighbor refinement. This refinement process enhances the quality of the reconstructed images by correcting distortions introduced during scrambling and GAN processing. A refinement loss is calculated between the reconstructed refined images and the original secret images to measure the accuracy of recovery. Additionally, another discriminator is introduced at this stage to distinguish between real secret images and reconstructed ones, further improving the generator's ability to restore the original images with high fidelity.

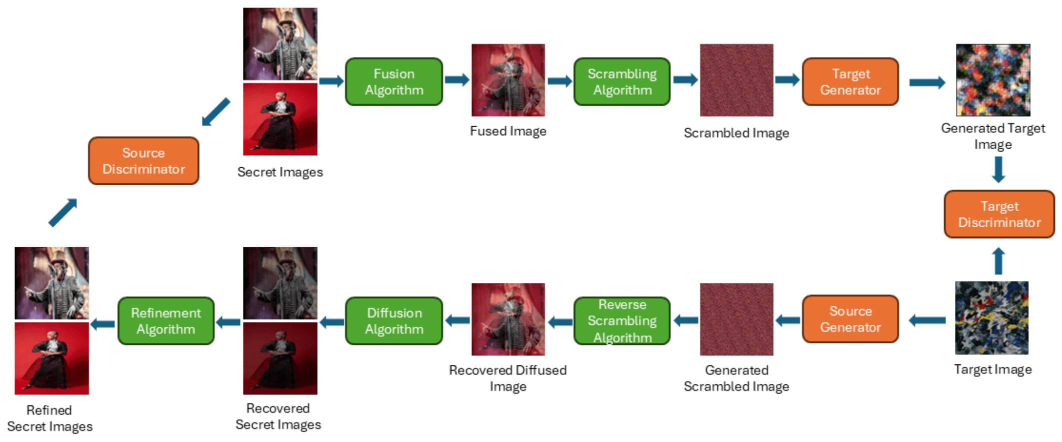


Figure 46: Training Architecture with Improved Loss Functions

With the use of only 6 images per domain (due to computational limitations), the model was able to generate a stego image with enhanced visual feature concealment. Notably, no visual features from the original secret images were shown in the stego image. This ensured that the secret images were effectively concealed. However, the smaller dataset led to better convergence for reconstructing the original images, even though the reconstructed images were completely blurred.

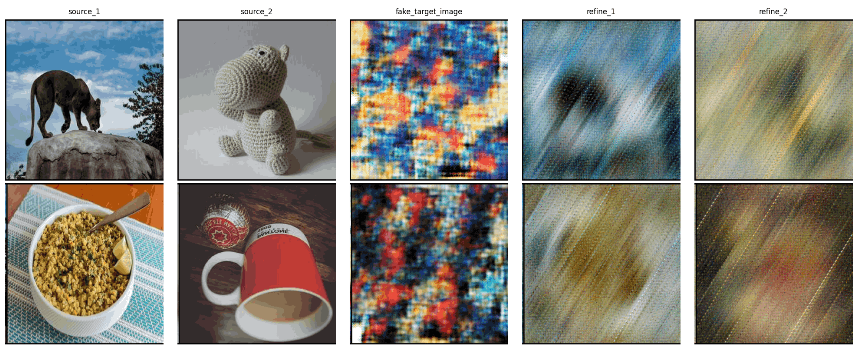


Figure 47: Pixel Wise Image Fusion, Scrambling and CycleGAN combination with Improved Loss Functions (Saved Images during training)

During evaluation, the generation of the stego image was not as expected but showed improvements compared to previous versions. The generated images in previous experiment, were closer to a black-and-white version of the fused and distorted secret images. This improvement in generation helped in the subsequent reconstruction, producing better results with a mixture of colors. However, since the dataset size was so small, the model did not generalize well, and the evaluations showed limitations in preserving the original image data into the target pixel values.

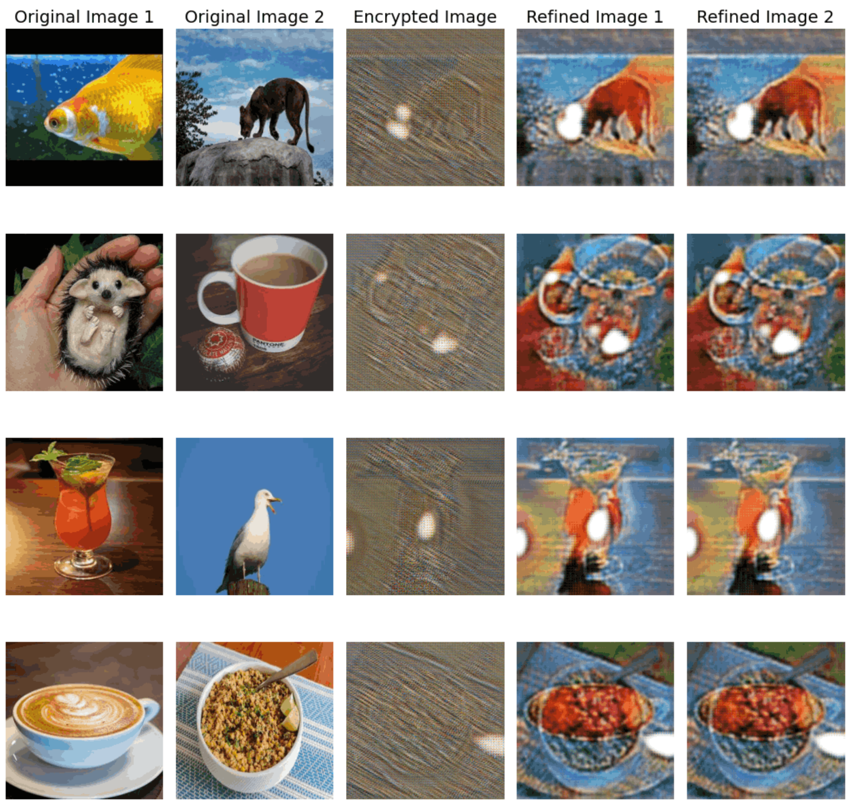


Figure 48: Evaluate Performance with Improved Loss Functions

An essential realization in this architecture is that the discriminators should focus on distinguishing between the original secret images and the reconstructed refined images, rather than between the fused scrambled images and the regener-

ated scrambled images. Since the ultimate goal is to recover the original images without revealing visual features in the stego image, the discriminator's role should be to ensure that the refined images retain as much of the original image quality as possible. This distinction helps the model prioritize meaningful reconstruction rather than just learning the transformation and scrambling process. By training the discriminator to compare real secret images with reconstructed ones, the network is better guided toward generating high-fidelity recoveries, making the overall steganographic framework more effective.
As a future direction, training the model on a more powerful machine, with a larger dataset containing more than 10,000 images per domain, could lead to better generalization. This larger training set would allow the model to learn more generalized features and improve the evaluation process.

5.9 Chapter Summary

This section presents the experimental setup and results of the proposed steganographic methodology, which demonstrates enhanced hiding capacity, improved reconstruction quality, and higher reconstruction accuracy. It also explores various techniques employed for image fusion, diffusion, and refinement, validating their effectiveness in achieving secure and efficient coverless multi-image steganography.

6 Critical Evaluation of Results

Since the proposed GAN-based coverless multi-image steganography methodology has not yet been successfully implemented and requires further development, the evaluation in this study is conducted on the Steganographic Key-based coverless multi-image steganography methodology.

6.1 Image Quality Evaluation

The proposed model was evaluated using a test dataset comprising 50 images. From this dataset, 100 input sets were created by randomly selecting 4 images per set. Each set was used to generate an intermediate image based on a unique steganographic key. The corresponding recovered images were then analyzed to assess the quality of image recovery. To provide a comprehensive evaluation, the MSE was calculated separately for each of the four recovered images in every input set.

Table 11: Mean Squared Error for Recovered Images per Set

	Image 1	**Image 2**	**Image 3**	**Image 4**
Set 1	9.181	8.758	9.006	5.312
Set 2	10.720	11.209	10.359	9.008
Set 3	9.181	5.312	9.820	6.880
Set 4	5.345	10.709	10.359	8.510
Set 5	8.746	8.931	9.006	6.880
Total Error	43.173	44.917	48.579	36.590
Avg Error	8.635	8.983	9.716	7.318

Table 12: Evaluation Metrics for Each Recovered Image on 100 sets

	Image 1	**Image 2**	**Image 3**	**Image 4**
Average MSE	137.055	136.095	130.568	128.920
Average RMSE	11.507	11.506	11.278	11.278
Average PSNR	27.039895	27.023080	27.191004	27.246834

The values obtained from this evaluation can be subjected to a comparison against existing image steganography methods.

Table 13: Comparison against existing image steganography methods

	Baluja	**Dharmawimala**	**lakshan**	**Proposed Method**
Average MSE	-	77.87	546.25	133.3013
Average RMSE	-	8.74	22.5	11.373
Average PSNR	27.51	29.39	27.13	27.246834

6.2 Steganalysis Evaluation

A critical aspect of evaluating the effectiveness of any steganographic system lies in its resilience against steganalysis techniques. The proposed coverless steganographic models provide a notable advantage in this regard by eliminating the dependency on predefined cover images. Traditional steganalysis algorithms often rely on detecting statistical anomalies or pixel-level alterations in the cover-stego image pairs. However, since the proposed methodology generates intermediate (stego) images from scratch without modifying any known cover, these conventional detection techniques become significantly less effective. This inherently enhances the stealth of the communication, making it more secure and less susceptible to automated steganalysis tools. Furthermore, by concealing visual features rather than embedding data into existing pixel values, the system minimizes detectable distortions, providing an advanced level of undetectability and robustness against analysis-based attacks. This design choice marks a substantial contribution toward developing highly secure, coverless multi-image steganographic systems.

6.3 Hiding Capacity

The embedding capacity of coverless image steganography models refers to the number of bits a carrier image can represent, typically measured in bits per cover. In the proposed model, the embedding capacity is calculated as $\mathbf{256 \times 256 \times 8 \times 3 \times 4}$ bits per image.

When evaluating the relative capacity, which denotes the number of bits per pixel that can be hidden, the model achieves a remarkable rate of $\mathbf{8 \times 3 \times 4 = 96}$ bits per pixel.

This represents a significant improvement compared to existing methods in the domain of coverless multi-image steganography. Notably, the proposed model maintains this high embedding capacity without any reduction in image size and ensures no visual feature leakage through the intermediate image. Moreover, the model achieves high recovery accuracy, preserving visual fidelity nearly identical to the original images, both in terms of human visual perception and quantitative reconstruction metrics.

Method	**Absolute Capacity (bits/image)**	**Image Size**	**Relative Capacity (bits/pixel)**
Dharmawimala (2023)	$256 \times 256 \times 8 \times 1 \times 2$	256×256	16
Lakshan (2024)	$256 \times 256 \times 8 \times 3 \times 2$	256×256	48
Proposed Method	$256 \times 256 \times 8 \times 3 \times 4$	256×256	96

Table 14: Comparison of Absolute and Relative Embedding Capacities

6.4 Robustness

To evaluate the robustness performance of the proposed model, a series of attacks were applied to the carrier (intermediate) image. After each perturbation, the model was used to reconstruct the secret images, and the accuracy of recovery was assessed. The Bit Error Rate (BER) was employed as the primary evaluation metric, computed by comparing the recovered secret images with their original counterparts. BER quantitatively measures the ratio of erroneous bits in the reconstructed images, thus serving as an indicator of the model's ability to resist distortions and adversarial manipulations. A lower BER indicates a more robust system capable of maintaining high fidelity under various attack scenarios. This analysis provides valuable insight into the model's resilience and effectiveness under challenging conditions.

$$\text{BER} = \frac{ER_a}{ER_o}$$

Where:

- ER_o: Error rate without any attack (baseline error).
- ER_a: Error rate after applying a specific attack.

A lower BER value indicates higher robustness and better preservation of the hidden information despite distortions.

- **Geometric Attacks**
  The proposed methodology demonstrated limited robustness against geometric transformations such as scaling, rotation, cropping, padding, and flipping. These attacks disrupted the spatial arrangement of the fused and scrambled images, which in turn affected the recovery process. Due to the multiple rounds of scrambling applied at different phases, the reconstructed images under geometric attacks were mostly distorted and appeared as scrambled versions of the original secret images. This indicates that the methodology is not resilient to spatial transformations that alter the geometric consistency of the image.

  - **Scaling Attack:** Resizing the image to a different resolution and restoring it to the original size, potentially distorting pixel-level information.

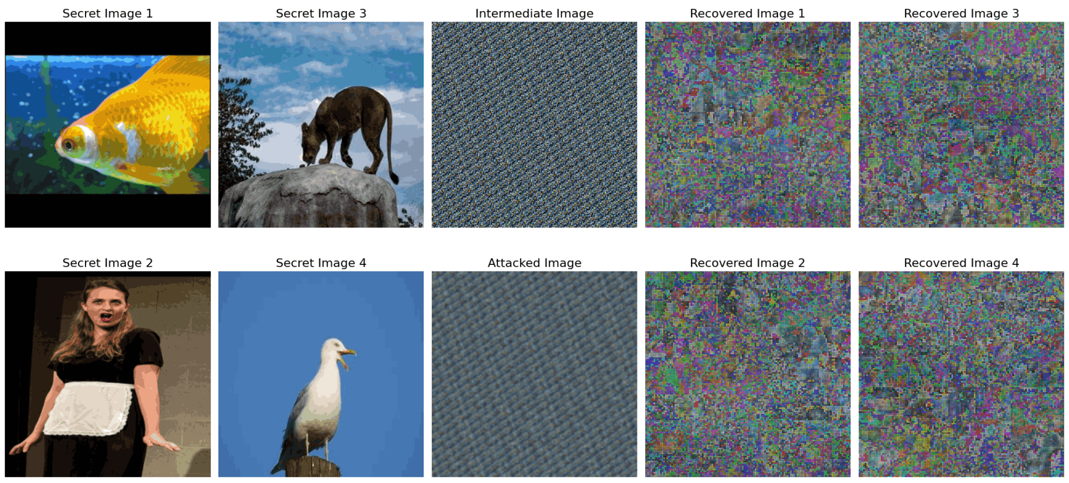


Figure 49: Scaling Attack (Down Scale) - BER RMSE: 7.155403

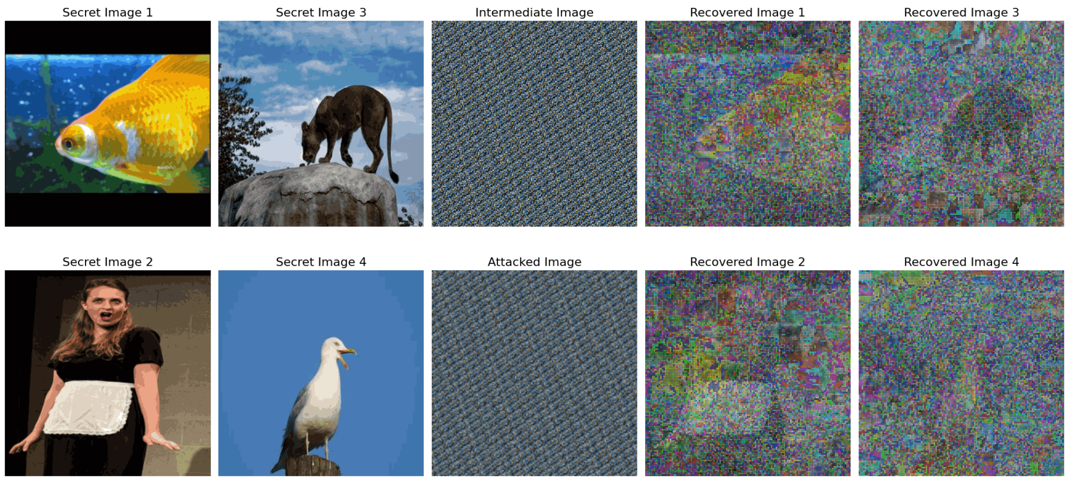


Figure 50: Scaling Attack (Up Scale) - BER RMSE: 6.368462

– **Rotational Attack:** Rotating the image at various angles to test recovery consistency after geometric transformations.

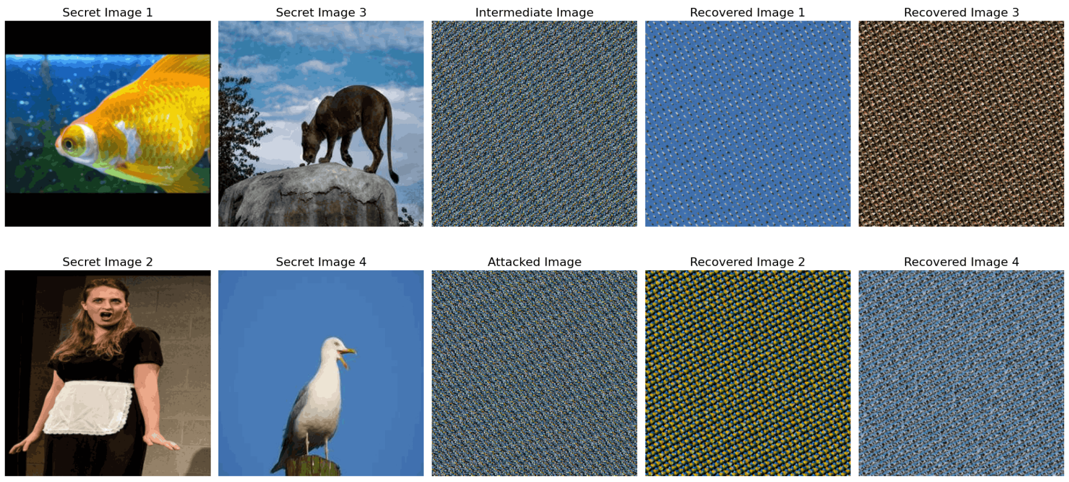


Figure 51: Rotational Attack (90 Degree) - BER RMSE: 8.826214

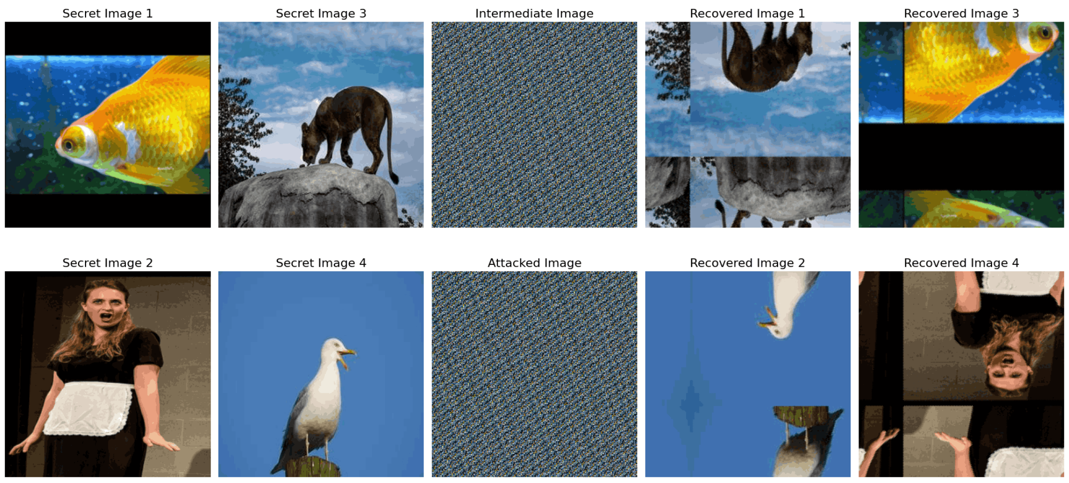


Figure 52: Rotational Attack (180 Degree) - BER RMSE: 9.358952

– **Cropping and Padding Attack:** Cropping portions of the image and padding them back to original dimensions, leading to spatial content loss or misalignment.

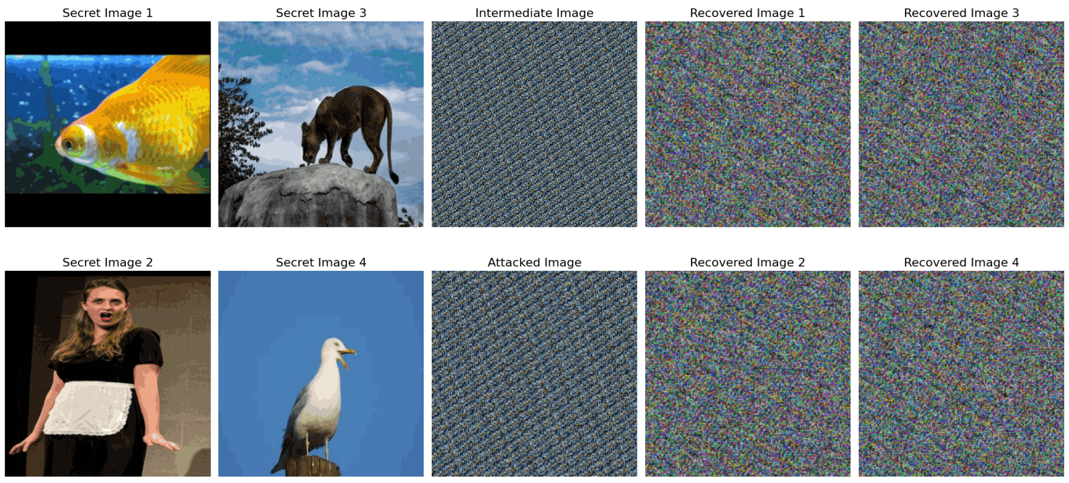


Figure 53: Cropping and Padding Attack - BER RMSE: 8.322171

– **Flipping Attack:** Horizontally or vertically flipping the image to simulate basic transformations.

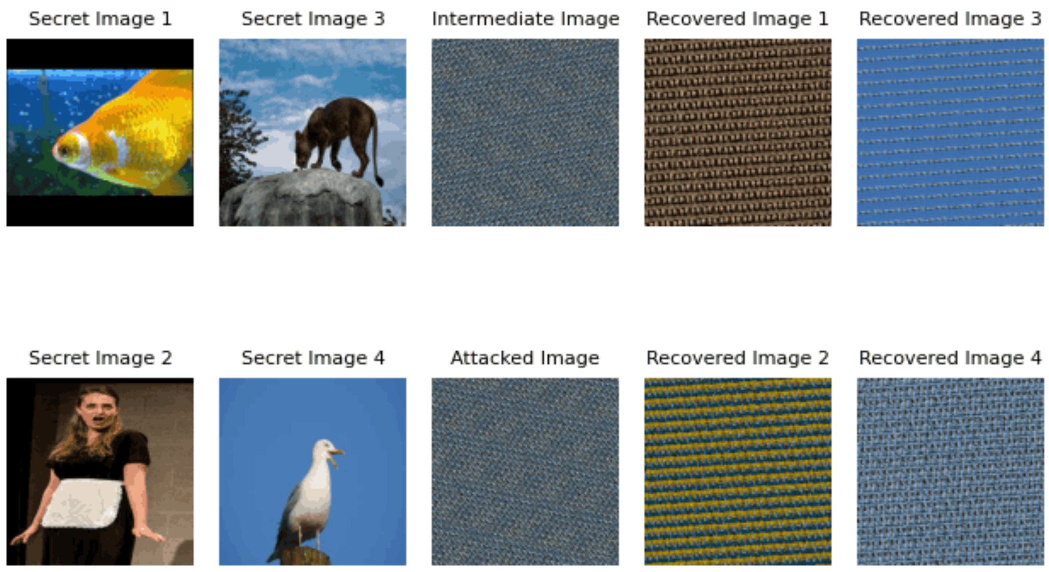


Figure 54: Flipping Attack (Horizontal) - BER RMSE: 8.757904

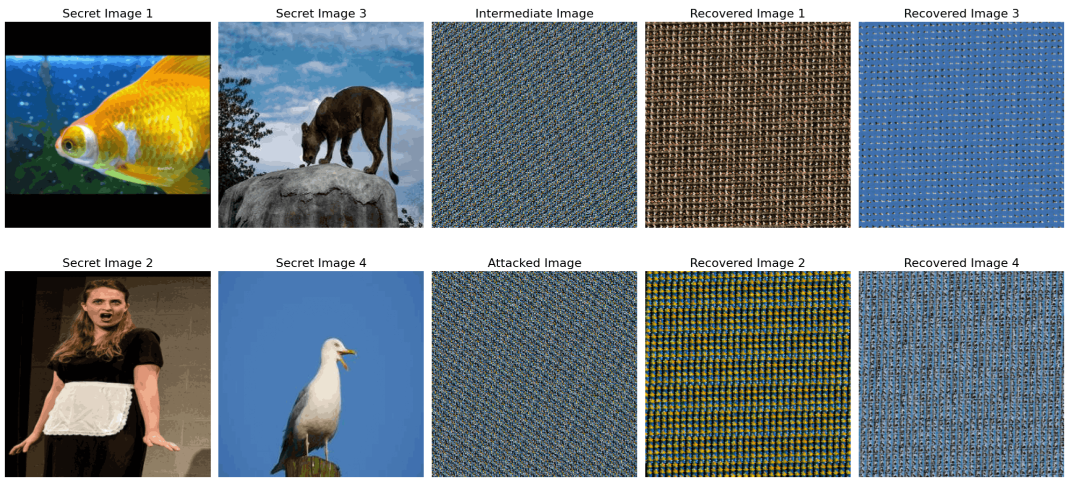


Figure 55: Flipping Attack (Vertical) - BER RMSE Error: 8.322171

- **Noise-Based Attacks**
  In terms of noise-based attacks, the proposed methodology showed moderate resilience. When Gaussian noise and speckle noise were introduced, the reconstructed outputs became noisy; however, the core visual features of the secret images remained recognizable to a certain extent. This suggests that although the model cannot completely eliminate the impact of such noise, it maintains a basic level of visual integrity in the recovered outputs. Most notably, under Salt and Pepper noise, the methodology performed exceptionally well, producing reconstructed images that were nearly identical to the originals with minimal degradation.
  - **Gaussian Noise Attack:** Introducing normally distributed noise to simulate natural disturbances during transmission.

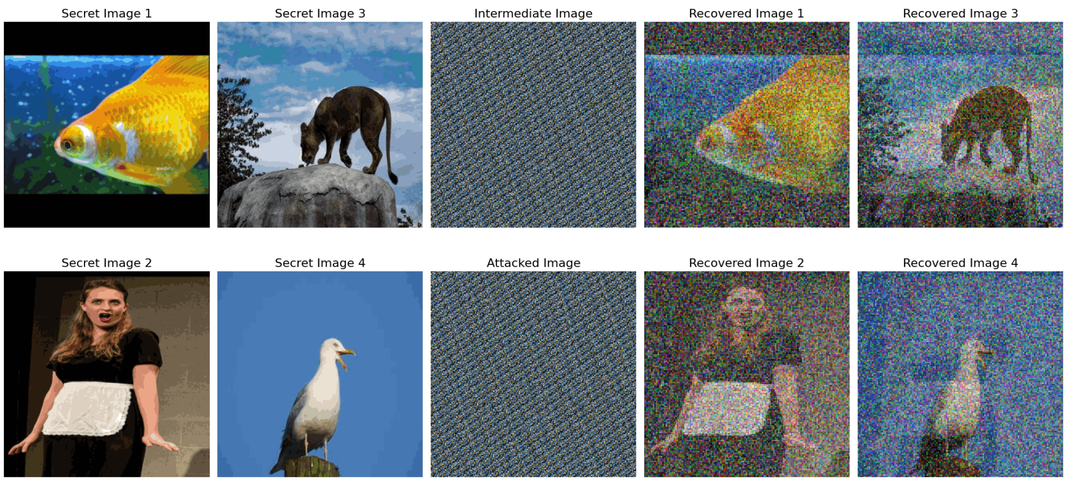


Figure 56: Gaussian Attack - BER RMSE: 5.846397

– **Salt and Pepper Noise Attack:** Randomly replacing pixel values with black or white to mimic data dropouts or extreme channel interference.

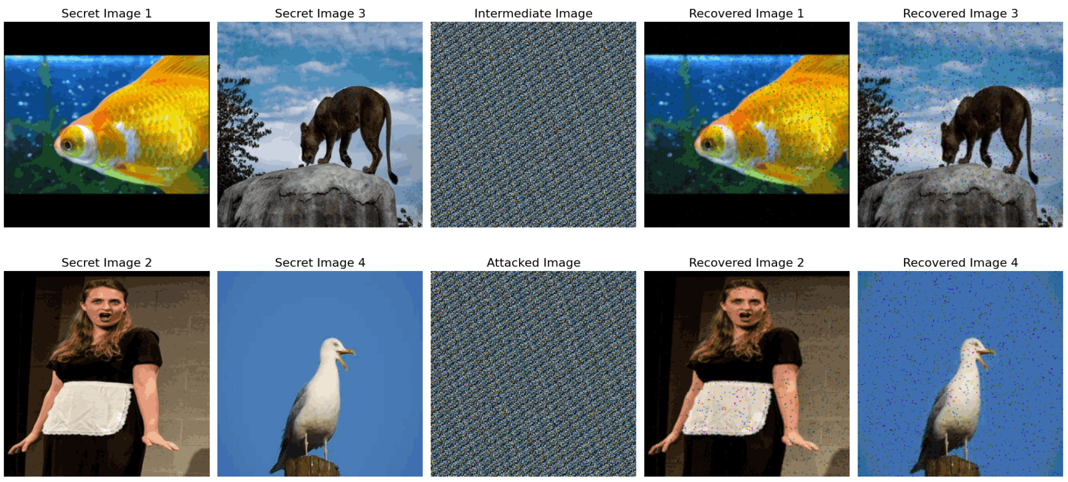


Figure 57: Salt and Pepper Noise Attack - BER RMSE: 1.695954

– **Speckle Noise Attack:** Applying multiplicative noise that simulates granular distortions commonly encountered in radar or ultrasound imagery.

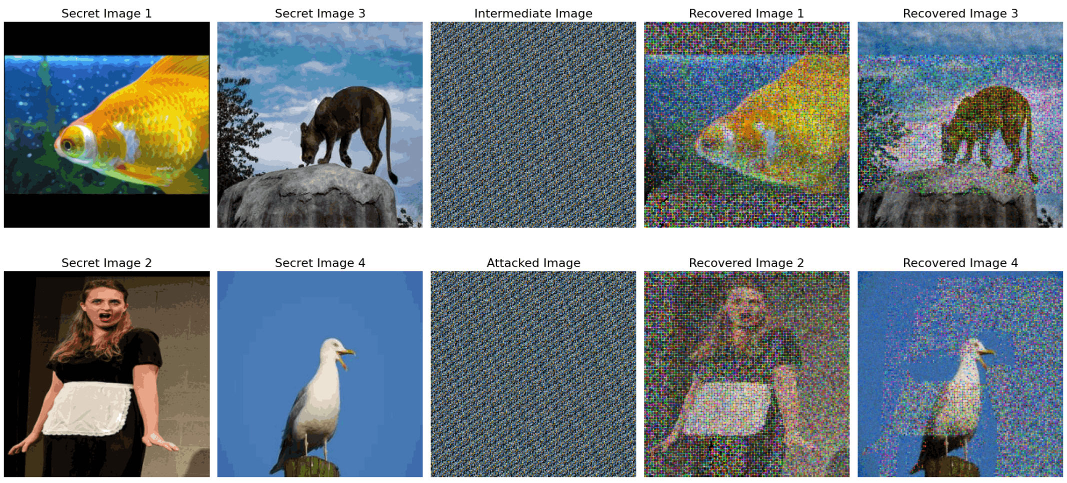


Figure 58: Speckle Noise Attack - BER RMSE: 5.257035

- **Filtering and Blurring Attacks**
  The methodology proved ineffective against filtering and blurring attacks. When subjected to mean filtering, median filtering, and motion blur, the reconstructed images suffered from significant quality loss. The visual clarity and structural fidelity of the original secret images were heavily compromised, indicating that the method does not possess sufficient robustness to defend against such smoothing operations. These attacks likely disrupted critical pixel-level features required for accurate reconstruction.

  – **Median Filtering Attack:** Employing non-linear filtering to remove noise, which may disrupt crucial pixel patterns.

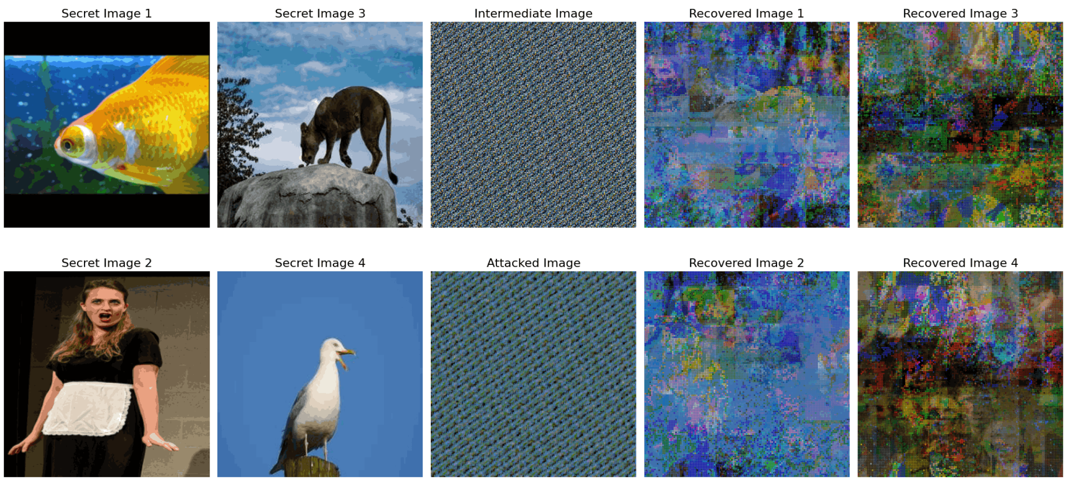


Figure 59: Median Filtering Attack - BER RMSE: 7.596181

  – **Mean Filtering Attack:** Using averaging filters to smooth the image can blur important details, reducing the accuracy of image recovery by removing critical visual features.

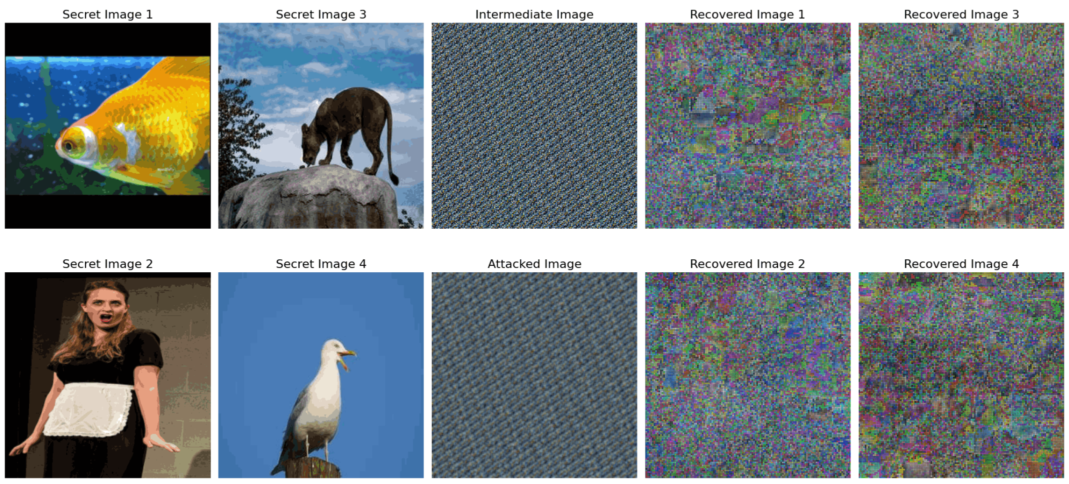


Figure 60: Mean Filtering Attack - BER RMSE: 7.400240

- **Motion Blur Attack:** Simulating camera or object movement to create streaking artifacts that obscure fine image details.

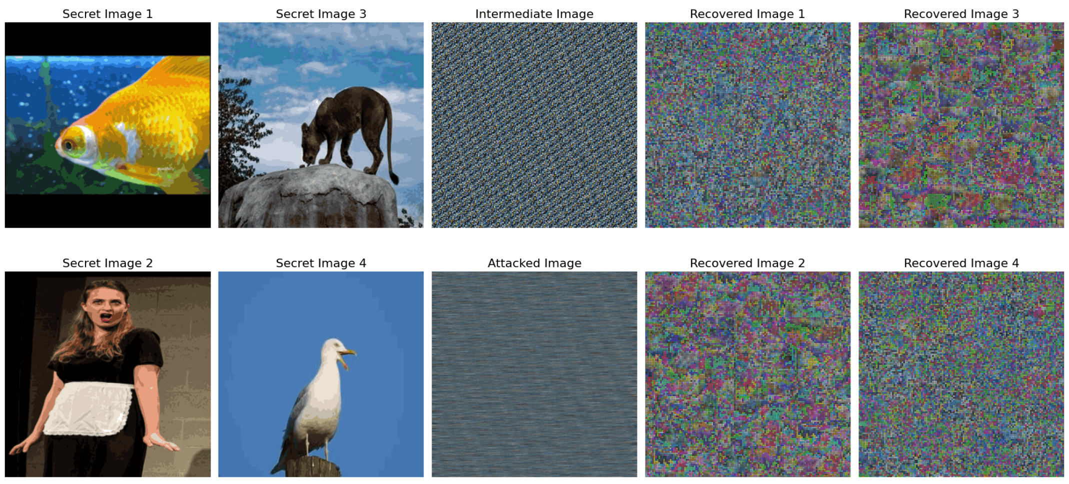


Figure 61: Motion Blur Attack - BER RMSE: 7.190984

- **Compression and Encoding Attacks**
  Under JPEG compression, the proposed methodology was partially successful. Although compression artifacts introduced visible distortions, the recovered images retained a basic representation of the original secret images. In many cases, the outputs resembled black-and-white versions of the original images, with high levels of noise. However, the fundamental visual structure was preserved, allowing the viewer to infer the original content. This shows some degree of compression resilience but highlights a need for further enhancement.

  - **JPEG Compression Attack:** Reducing image quality through lossy compression to evaluate performance under data degradation.

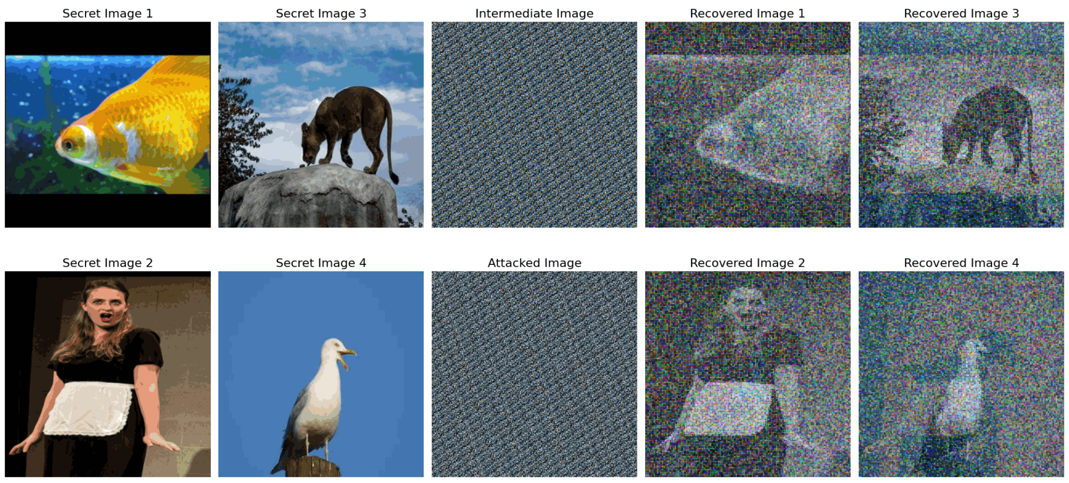


Figure 62: JPEG Compression Attack - BER RMSE: 6.173029

- **Color and Channel Distortion Attacks**
  The proposed methodology was not resilient against color distortion attacks, particularly color jitter. The reconstructed images were often distorted, containing a mixture of visual elements from the original inputs but lacking coherent structure. On the other hand, channel shuffle attacks were better tolerated. Although the resulting images experienced notable changes in color representation, their structural content and clarity were preserved. This suggests that while the model can handle some level of channel manipulation, it is vulnerable to complex color distortions.

  - **Color Jitter Attack:** Randomly altering brightness, contrast, saturation, and hue to simulate variations in lighting conditions.

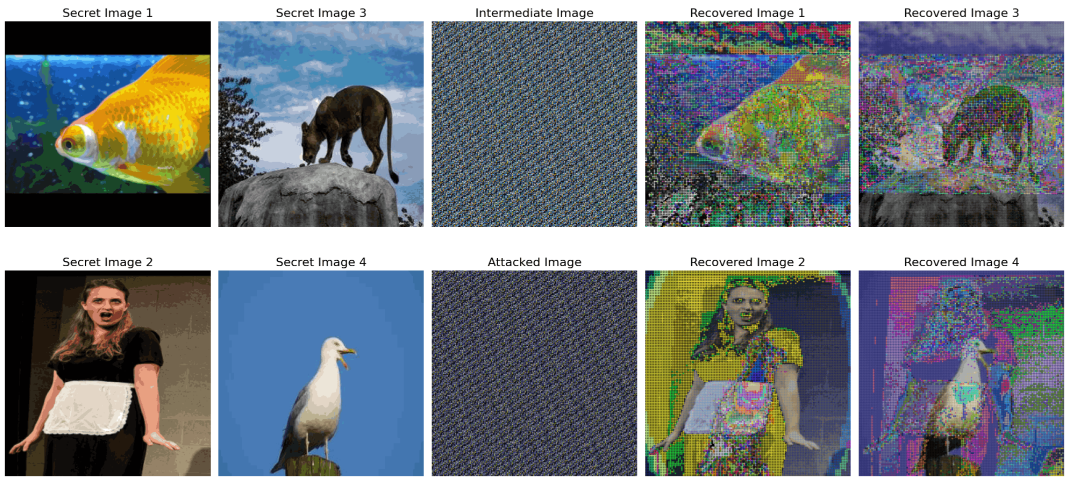


Figure 63: Color Jitter Attack - BER RMSE: 6.121747

  - **Channel Shuffle Attack:** Randomly permuting the RGB channels to challenge the model's ability to recover based on disrupted color semantics.

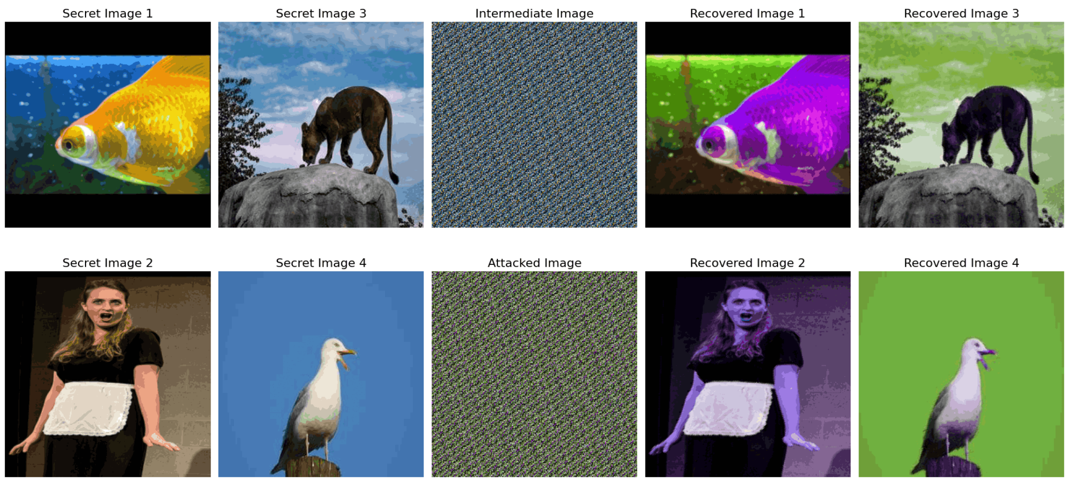


Figure 64: Channel Shuffle Attack - BER RMSE Error: 2.564809

- **Content Alteration Attacks**
  The proposed methodology demonstrated strong resistance to content alteration attacks. When random pixel deletion and random pixel modification were applied, the reconstructed images remained largely intact, with only minimal noise or visual artifacts. The ability to recover secret images under these conditions indicates the model's effectiveness in handling random, non-structured disruptions. This resilience further supports the robustness of the approach for real-world communication scenarios where content integrity may be partially compromised.

  – **Random Deletion Attack:** Randomly deleting half of the pixel values in the intermediate image.

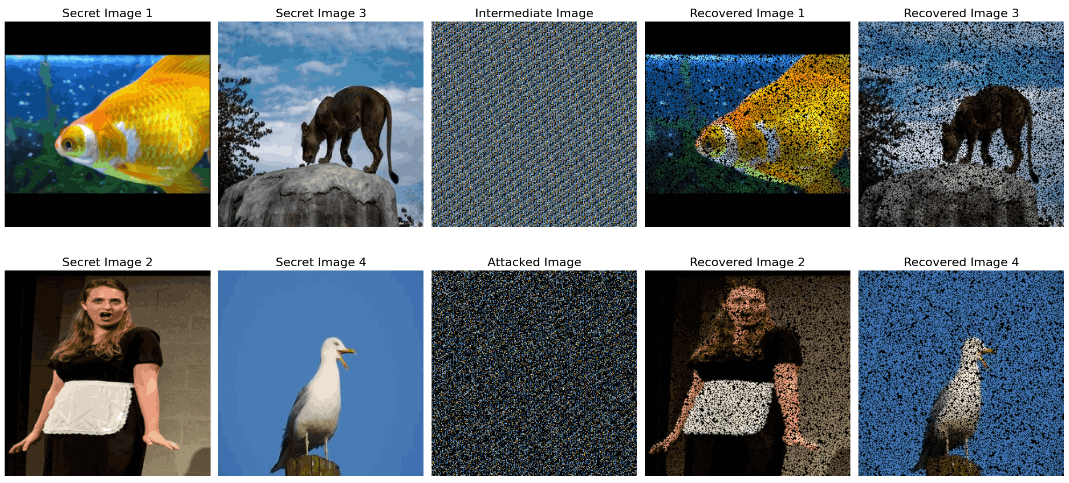


Figure 65: Random Deletion Attack - BER RMSE: 6.214473

  – **Random Alteration Attack:** Randomly applying arbitrary changes to half of the pixel values in the intermediate image.

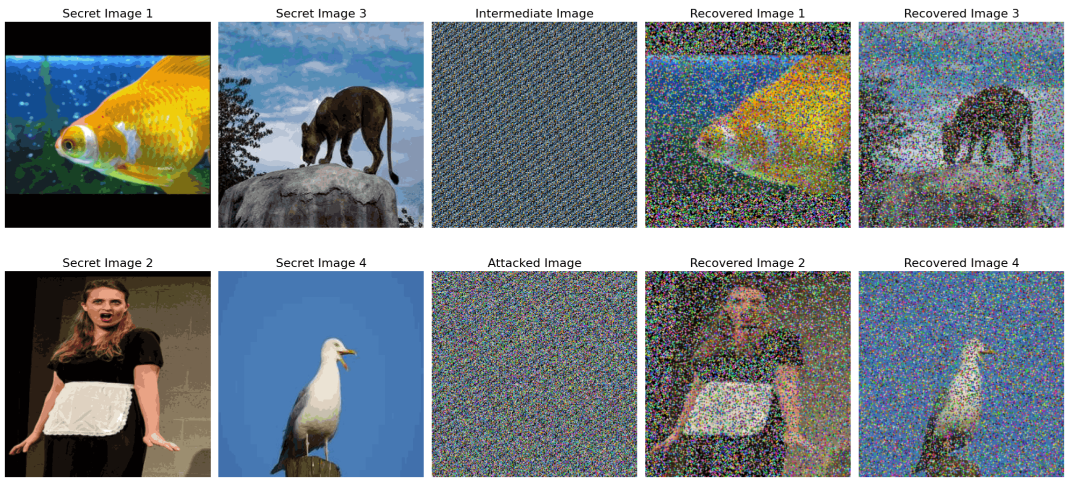


Figure 66: Random Alteration Attack - BER RMSE: 5.689601

6.5 Security

The steganographic model is evaluated for security based on the role of the steganographic key used during encoding and decoding. The results confirm that only the exact secret key used during the encoding process can successfully decode and reconstruct the original images. Any attempt to decode the intermediate image using alternative or incorrect keys results in the recovery of meaningless or visually incoherent outputs. This key-dependent decoding mechanism ensures that the embedded content remains inaccessible to unauthorized parties.
Unlike GAN-based steganographic models, which may be vulnerable if the generator architectures or weights are exposed, this steganographic key-based method offers an additional layer of security. Even if the underlying algorithm is known, the absence of the correct key prevents the reconstruction of the secret images. This property significantly enhances the security of the proposed approach, ensuring that only authorized users with the exact steganographic key can access the hidden content, thereby mitigating the risks associated with model reuse or leakage.

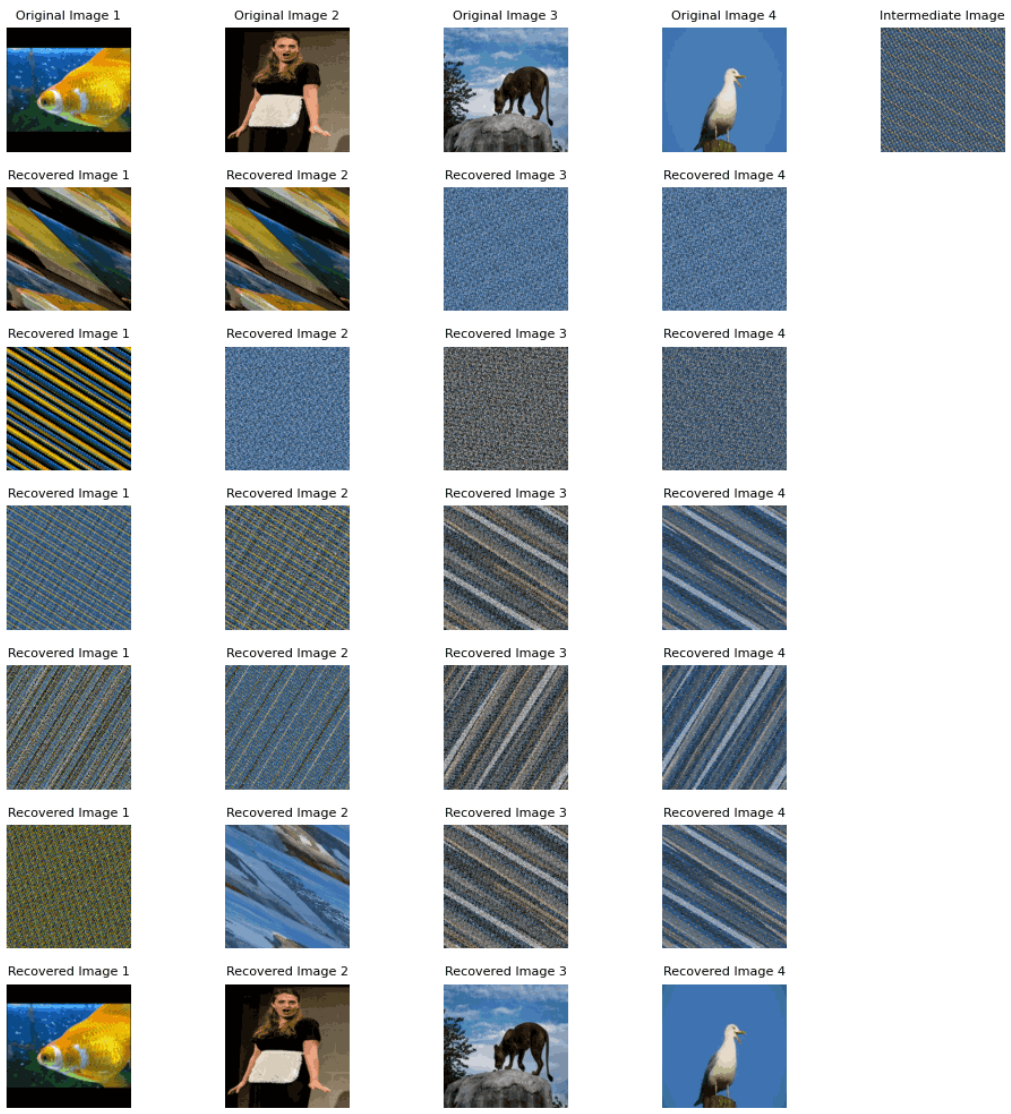


Figure 67: Security Evaluation with Steganographic key usage
Row 1 : Orignal images and Intermediate Image (Stego Image)
Row 2-6 : Decoding using wrong keys
Row 7 : Decoding using correct key

6.6 Overall Evaluation Summary

The proposed methodology demonstrates exceptional performance when evaluated against existing multi-image steganographic techniques, particularly in terms of reconstruction quality. The reconstructed images closely resemble the original secret images to the human eye, with only minimal statistical differences. This highlights a significant improvement in the fidelity of reconstruction. The model's ability to preserve visual quality is largely attributed to the strategic integration of multiple scrambling phases and effective image fusion mechanisms, which also contribute to enhanced visual feature concealment.
In terms of hiding capacity, the methodology achieves a remarkable improvement by embedding four secret images within a single fused image, reaching a capac-

ity of 96 bits per pixel. This greatly surpasses the capabilities of conventional multi-image steganographic methods, positioning the proposed approach as a high-capacity solution in the domain. The fused image, which contains synthesized information from all four secret images, supports secure and efficient data hiding without compromising image integrity.
The coverless nature of the methodology further enhances its stealth and resistance to traditional steganalysis tools, which are typically designed to detect modifications in predefined cover images. By avoiding the use of an explicit cover and instead relying on synthesized intermediate images that conceal visual features, the system significantly reduces detectable distortions. This design provides a high level of undetectability and robustness against analysis-based attacks.
However, the methodology exhibits certain limitations in resilience under specific attack scenarios. It is particularly vulnerable to geometric attacks such as scaling, rotation, cropping, and flipping, as well as filtering and blurring attacks. These transformations disrupt the spatial and structural consistency needed for accurate reconstruction. On the other hand, it shows moderate resilience to noise-based attacks (e.g., Gaussian, speckle, and salt-and-pepper), compression artifacts, and color or channel distortion attacks, with some visual fidelity being preserved in the outputs.
Most notably, the methodology demonstrates strong resistance against content alteration attacks such as random pixel deletion and modification. Even under such disruptive conditions, the reconstructed images remain visually clear and structurally accurate, showcasing the robustness of the proposed approach in maintaining information integrity in real-world communication scenarios.
In addition to its technical robustness, the proposed methodology offers enhanced security through the use of a steganographic key, which is required for both encoding and decoding processes. This key-dependency ensures that even if the encoding algorithm is publicly known, unauthorized parties cannot recover the original secret images without the exact key. This mechanism adds a critical layer of access control and significantly reduces the risk of unintended data recovery, reinforcing the method's suitability for secure communication. The security-centric design not only protects against visual and statistical detection but also mitigates the threat of reverse-engineering or model misuse.

6.7 Chapter Summary

This section provides a critical evaluation of the proposed model through comprehensive image quality, steganalysis, hiding capacity, security and robustness assessments. Both qualitative and quantitative results are presented to demonstrate the model's effectiveness, security, and resilience under various conditions and evaluation metrics.

7 Conclusion

7.1 Conclusions about Research Questions

The primary objective of this research was to develop a novel coverless multi-image steganographic method capable of transforming multiple secret images into a single carrier image while ensuring minimal reconstruction loss and effective concealment of visual features. To address this, the study focused on the question of 'How can coverless multi image steganography methods be improved to reconstruct the original secret images with minimal loss of visual features?'. Through a systematic exploration of fusion, diffusion, and refinement techniques, the research introduced a model that significantly improves visual feature concealment and reconstruction accuracy.
The study further addressed the challenge of increasing hiding capacity without compromising security or image quality. By integrating key findings from the literature, particularly building on Baluja's image-to-image hiding approach, a new coverless multi-image steganography method was proposed that demonstrates a higher hiding capacity of 96 bits per pixel while maintaining high reconstruction fidelity and undetectability, in order to address the research question 'How can coverless multi-image steganography methods be improved to achieve higher hiding capacity while preserving reconstruction quality and security?'.
Additionally, the integration of advanced GAN-based techniques was explored to enhance the concealment of visual features and the overall security of the system, to address the research question 'How GAN based coverless multi image steganography methods can be improved to enhance the visual feature concealment and security performance in GAN based multi-image steganography?'. Despite hardware limitations that constrained training complexity and dataset size, the results indicate that GAN integration has the potential to further advance coverless multi-image steganography.
In conclusion, the research successfully answers the posed questions by presenting a steganographic key-based methodology that significantly advances the state of coverless multi-image steganography. The proposed method achieves high reconstruction quality, improved hiding capacity, and strong resistance to detection, marking a valuable contribution to the field.

7.2 Conclusions about Research Problem

The research problem addressed in this study centered on improving the concealment of visual features in stego images within the domain of coverless multi-image steganography. To effectively tackle this, the study explored a range of techniques involving image fusion, diffusion, and transformations. By integrating a novel fusion approach with advanced transformation methods, a new coverless steganographic framework was developed. This approach demonstrated significant improvements in terms of visual feature concealment, hiding capacity, reconstruction ability, and overall security. The experimental results confirmed that the proposed methodology effectively addressed the research problem, surpassing the performance of existing techniques. Consequently, this research contributes meaningfully to the field by offering an innovative and more secure method for multi-image steganography.

7.3 Limitations

The proposed steganographic key-based image fusion methodology is specifically designed to operate with exactly four secret images. While it is possible to adapt the method for fewer inputs by duplicating images, the original design expects four distinct images, which may limit its flexibility in certain use cases. Additionally, the methodology is initially tailored for 256×256 RGB images. Although alternative modifications can enable its application to images of other resolutions, such adjustments may affect performance or require further tuning.

Furthermore, due to hardware limitations, the GAN-based model was constrained to training on a minimal dataset of six 256×256 images. This limitation impacted the ability to scale the model's complexity and fully explore its potential. A small dataset size, coupled with limited training epochs, reduced the efficiency of the model and led to less sharp reconstructions. Although the model successfully demonstrated pixel-wise recovery and maintained structural integrity, the resulting images were comparatively more blurred. The model's overall performance is closely tied to the complexity of the network architecture and the size and quality of the dataset used.

7.4 Future Directions

As future directions, the proposed GAN-based steganographic architecture can be further explored with enhanced computational resources, allowing experimentation on larger and more diverse datasets. This would also enable the deployment of more sophisticated generator architectures, potentially improving the quality and fidelity of the generated and reconstructed images. Once stable performance is achieved, additional enhancements through hyperparameter tuning could be investigated to refine the model further.

Moreover, the proposed steganographic key-based image fusion methodology holds potential for integration with GAN techniques to improve the overall hiding capacity. While this integration is likely to increase the amount of secret data that can be embedded, it may also introduce challenges in maintaining high reconstruction accuracy.

Future work can also focus on developing a more robust and adaptable coverless multi-image steganography model that performs reliably under various attack scenarios and communication distortions. Investigating different fusion, scrambling, and reconstruction techniques, as well as extending the system to support variable input sizes and formats, can significantly contribute to making the methodology more practical and secure in real-world applications.

References

Baluja, S. (2017), 'Hiding images in plain sight: Deep steganography', **30**.
URL: *https://proceedings.neurips.cc/paper$_f$iles/paper/2017/file/838e8afb1ca34354ac209f53d9*
Paper.pdf

Choudary, A. (2023), 'Steganography tutorial – a complete guide for beginners'.
URL: *https://www.edureka.co/blog/steganography-tutorial*

Chu, C., Zhmoginov, A. & Sandler, M. (2017), 'Cyclegan, a master of steganography', *ArXiv* **abs/1712.02950**.
URL: *https://api.semanticscholar.org/CorpusID:26516599*

Dharmawimala, Y. (2023), 'Coverless multi-image steganography using generative adversarial networks'.

Dickson, B. (2020), 'What is steganography? a complete guide to the ancient art of concealing messages'.
URL: *https://portswigger.net/daily-swig/what-is-steganography-a-complete-guide-to-the-ancient-art-of-concealing-messages*

Duan, X., Li, B., Guo, D., Zhang, Z. & Ma, Y. (2020), 'A coverless steganography method based on generative adversarial network', *EURASIP Journal on Image and Video Processing* **2020**(1), 18.
URL: *https://doi.org/10.1186/s13640-020-00506-6*

Felix, K., Yossi, A., Bhiksha, R., Rita, S. & Joseph, K. (2020), 'Hide and speak: Towards deep neural networks for speech steganography'.
URL: *https://arxiv.org/abs/1902.03083*

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. (2014), 'Generative adversarial nets', p. 2672–2680.

Lakshan, M. (2024), 'Gan-based coverless multi-image steganography with rgb secret images'.

Liu, J., Ke, Y., Zhang, Z., Lei, Y., Li, J., Zhang, M. & Yang, X. (2020), 'Recent advances of image steganography with generative adversarial networks', *IEEE Access* **8**, 60575–60597.

Liu, X., Ma, Z., Guo, X., Hou, J., Schaefer, G., Wang, L., Wang, V. & Fang, H. (2020), 'Camouflage generative adversarial network: Coverless full-image-to-image hiding', pp. 166–172.

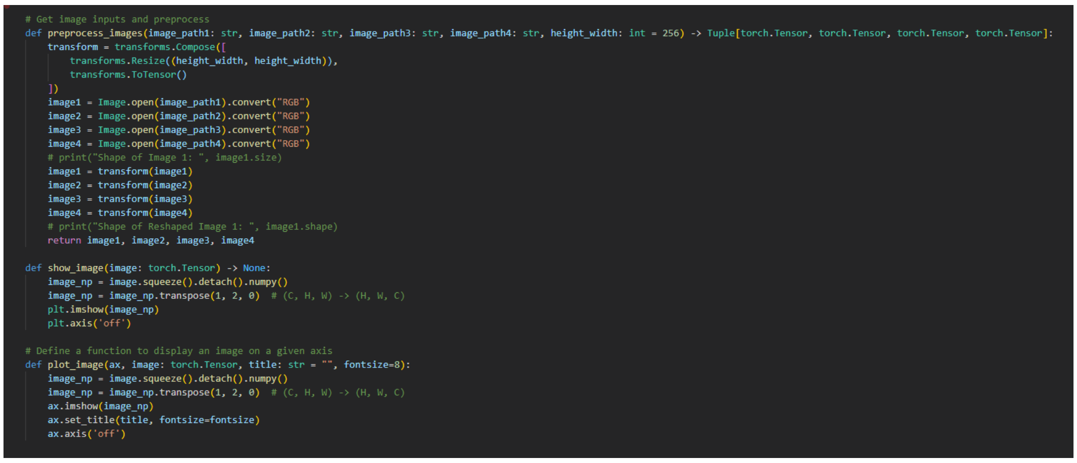
Min, L., Ting, L. & Jie, H. (2013), 'Arnold transform based image scrambling method'.

Qian, Y., Dong, J., Wang, W. & Tan, T. (2015), ‘Deep learning for steganalysis via convolutional neural networks’, *Proceedings of SPIE - The International Society for Optical Engineering* **9409**.

Semilof, M. & Clark, C. (2023), ‘Steganography’.
URL: *https://www.techtarget.com/searchsecurity/definition/steganography*

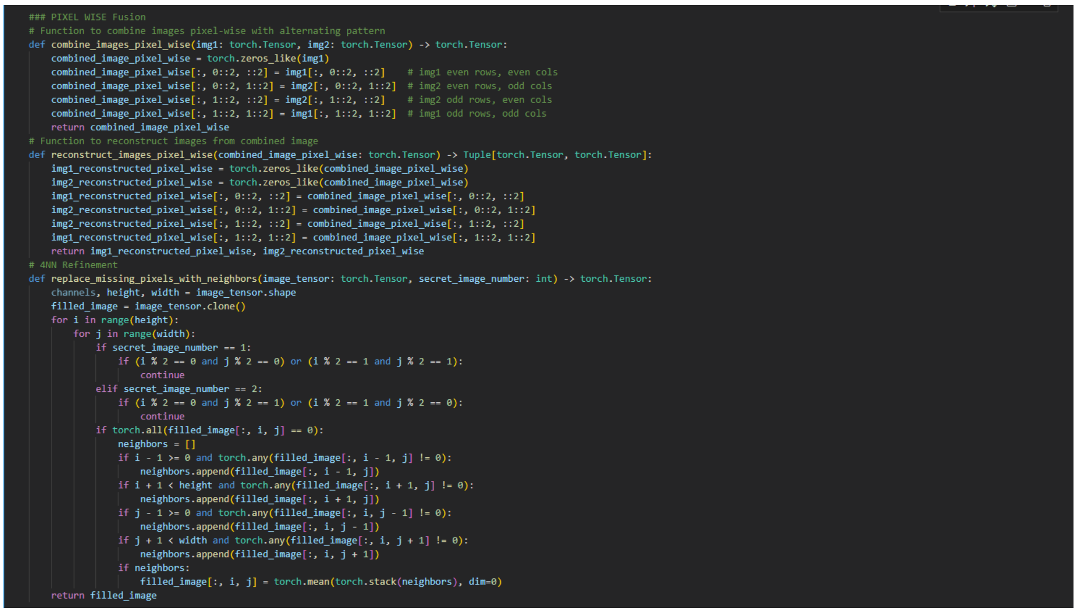
Tidmarsh, D. (2023), ‘A guide to steganography: Meaning, types, tools, techniques’.
URL: *https://www.eccouncil.org/cybersecurity-exchange/ethical-hacking/what-is-steganography-guide-meaning-types-tools/*

8 Appendix



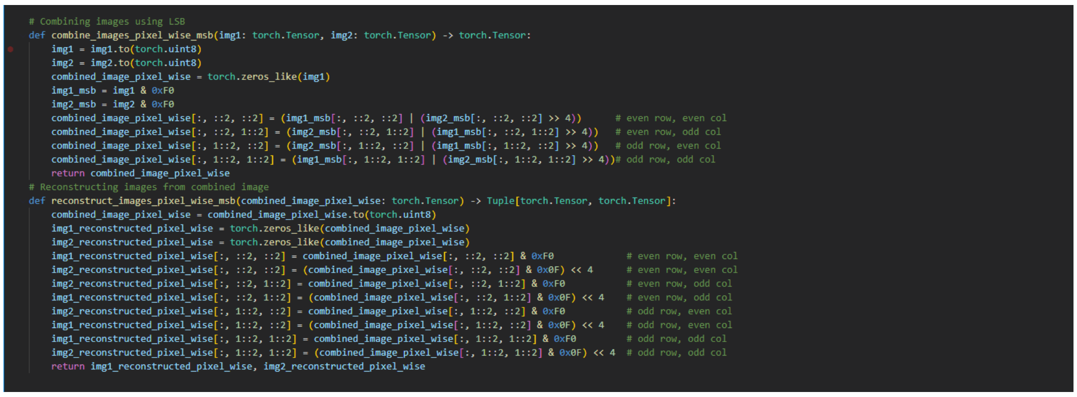
```python
# Get image inputs and preprocess
def preprocess_images(image_path1: str, image_path2: str, image_path3: str, image_path4: str, height_width: int = 256) -> Tuple[torch.Tensor, torch.Tensor, torch.Tensor, torch.Tensor]:
    transform = transforms.Compose([
        transforms.Resize((height_width, height_width)),
        transforms.ToTensor()
    ])
    image1 = Image.open(image_path1).convert("RGB")
    image2 = Image.open(image_path2).convert("RGB")
    image3 = Image.open(image_path3).convert("RGB")
    image4 = Image.open(image_path4).convert("RGB")
    # print("Shape of Image 1: ", image1.size)
    image1 = transform(image1)
    image2 = transform(image2)
    image3 = transform(image3)
    image4 = transform(image4)
    # print("Shape of Reshaped Image 1: ", image1.shape)
    return image1, image2, image3, image4

def show_image(image: torch.Tensor) -> None:
    image_np = image.squeeze().detach().numpy()
    image_np = image_np.transpose(1, 2, 0)  # (C, H, W) -> (H, W, C)
    plt.imshow(image_np)
    plt.axis('off')

# Define a function to display an image on a given axis
def plot_image(ax, image: torch.Tensor, title: str = "", fontsize=8):
    image_np = image.squeeze().detach().numpy()
    image_np = image_np.transpose(1, 2, 0)  # (C, H, W) -> (H, W, C)
    ax.imshow(image_np)
    ax.set_title(title, fontsize=fontsize)
    ax.axis('off')
```

Figure 68: Code Snippet for Predefined functions



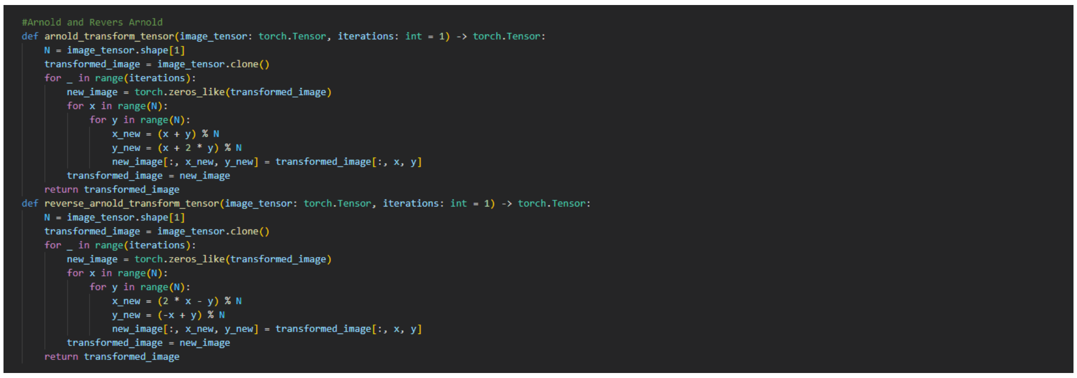
```python
### PIXEL WISE Fusion
# Function to combine images pixel-wise with alternating pattern
def combine_images_pixel_wise(img1: torch.Tensor, img2: torch.Tensor) -> torch.Tensor:
    combined_image_pixel_wise = torch.zeros_like(img1)
    combined_image_pixel_wise[:, 0::2, ::2] = img1[:, 0::2, ::2]    # img1 even rows, even cols
    combined_image_pixel_wise[:, 0::2, 1::2] = img2[:, 0::2, 1::2]  # img2 even rows, odd cols
    combined_image_pixel_wise[:, 1::2, ::2] = img2[:, 1::2, ::2]    # img2 odd rows, even cols
    combined_image_pixel_wise[:, 1::2, 1::2] = img1[:, 1::2, 1::2]  # img1 odd rows, odd cols
    return combined_image_pixel_wise
# Function to reconstruct images from combined image
def reconstruct_images_pixel_wise(combined_image_pixel_wise: torch.Tensor) -> Tuple[torch.Tensor, torch.Tensor]:
    img1_reconstructed_pixel_wise = torch.zeros_like(combined_image_pixel_wise)
    img2_reconstructed_pixel_wise = torch.zeros_like(combined_image_pixel_wise)
    img1_reconstructed_pixel_wise[:, 0::2, ::2] = combined_image_pixel_wise[:, 0::2, ::2]
    img2_reconstructed_pixel_wise[:, 0::2, 1::2] = combined_image_pixel_wise[:, 0::2, 1::2]
    img2_reconstructed_pixel_wise[:, 1::2, ::2] = combined_image_pixel_wise[:, 1::2, ::2]
    img1_reconstructed_pixel_wise[:, 1::2, 1::2] = combined_image_pixel_wise[:, 1::2, 1::2]
    return img1_reconstructed_pixel_wise, img2_reconstructed_pixel_wise
# 4NN Refinement
def replace_missing_pixels_with_neighbors(image_tensor: torch.Tensor, secret_image_number: int) -> torch.Tensor:
    channels, height, width = image_tensor.shape
    filled_image = image_tensor.clone()
    for i in range(height):
        for j in range(width):
            if secret_image_number == 1:
                if (i % 2 == 0 and j % 2 == 0) or (i % 2 == 1 and j % 2 == 1):
                    continue
            elif secret_image_number == 2:
                if (i % 2 == 0 and j % 2 == 1) or (i % 2 == 1 and j % 2 == 0):
                    continue
            if torch.all(filled_image[:, i, j] == 0):
                neighbors = []
                if i - 1 >= 0 and torch.any(filled_image[:, i - 1, j] != 0):
                    neighbors.append(filled_image[:, i - 1, j])
                if i + 1 < height and torch.any(filled_image[:, i + 1, j] != 0):
                    neighbors.append(filled_image[:, i + 1, j])
                if j - 1 >= 0 and torch.any(filled_image[:, i, j - 1] != 0):
                    neighbors.append(filled_image[:, i, j - 1])
                if j + 1 < width and torch.any(filled_image[:, i, j + 1] != 0):
                    neighbors.append(filled_image[:, i, j + 1])
                if neighbors:
                    filled_image[:, i, j] = torch.mean(torch.stack(neighbors), dim=0)
    return filled_image
```

Figure 69: Code Snippet for Pixel Wise Fusion



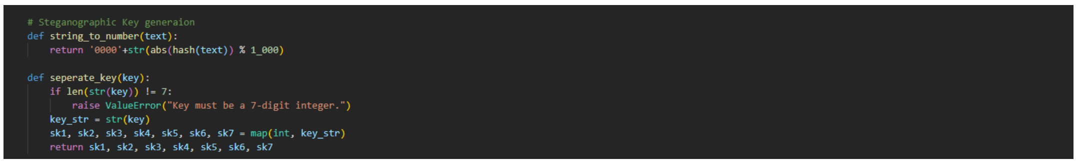
```python
# Combining images using LSB
def combine_images_pixel_wise_msb(img1: torch.Tensor, img2: torch.Tensor) -> torch.Tensor:
    img1 = img1.to(torch.uint8)
    img2 = img2.to(torch.uint8)
    combined_image_pixel_wise = torch.zeros_like(img1)
    img1_msb = img1 & 0xF0
    img2_msb = img2 & 0xF0
    combined_image_pixel_wise[:, ::2, ::2] = (img1_msb[:, ::2, ::2] | (img2_msb[:, ::2, ::2] >> 4))      # even row, even col
    combined_image_pixel_wise[:, ::2, 1::2] = (img2_msb[:, ::2, 1::2] | (img1_msb[:, ::2, 1::2] >> 4))   # even row, odd col
    combined_image_pixel_wise[:, 1::2, ::2] = (img2_msb[:, 1::2, ::2] | (img1_msb[:, 1::2, ::2] >> 4))   # odd row, even col
    combined_image_pixel_wise[:, 1::2, 1::2] = (img1_msb[:, 1::2, 1::2] | (img2_msb[:, 1::2, 1::2] >> 4))# odd row, odd col
    return combined_image_pixel_wise
# Reconstructing images from combined image
def reconstruct_images_pixel_wise_msb(combined_image_pixel_wise: torch.Tensor) -> Tuple[torch.Tensor, torch.Tensor]:
    combined_image_pixel_wise = combined_image_pixel_wise.to(torch.uint8)
    img1_reconstructed_pixel_wise = torch.zeros_like(combined_image_pixel_wise)
    img2_reconstructed_pixel_wise = torch.zeros_like(combined_image_pixel_wise)
    img1_reconstructed_pixel_wise[:, ::2, ::2] = combined_image_pixel_wise[:, ::2, ::2] & 0xF0            # even row, even col
    img2_reconstructed_pixel_wise[:, ::2, ::2] = (combined_image_pixel_wise[:, ::2, ::2] & 0x0F) << 4     # even row, even col
    img2_reconstructed_pixel_wise[:, ::2, 1::2] = combined_image_pixel_wise[:, ::2, 1::2] & 0xF0          # even row, odd col
    img1_reconstructed_pixel_wise[:, ::2, 1::2] = (combined_image_pixel_wise[:, ::2, 1::2] & 0x0F) << 4   # even row, odd col
    img2_reconstructed_pixel_wise[:, 1::2, ::2] = combined_image_pixel_wise[:, 1::2, ::2] & 0xF0          # odd row, even col
    img1_reconstructed_pixel_wise[:, 1::2, ::2] = (combined_image_pixel_wise[:, 1::2, ::2] & 0x0F) << 4   # odd row, even col
    img1_reconstructed_pixel_wise[:, 1::2, 1::2] = combined_image_pixel_wise[:, 1::2, 1::2] & 0xF0        # odd row, odd col
    img2_reconstructed_pixel_wise[:, 1::2, 1::2] = (combined_image_pixel_wise[:, 1::2, 1::2] & 0x0F) << 4  # odd row, odd col
    return img1_reconstructed_pixel_wise, img2_reconstructed_pixel_wise
```

Figure 70: Code Snippet for LSB Manipulation Fusion



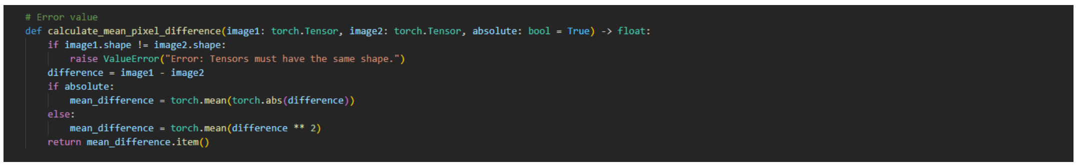
```python
#Arnold and Revers Arnold
def arnold_transform_tensor(image_tensor: torch.Tensor, iterations: int = 1) -> torch.Tensor:
    N = image_tensor.shape[1]
    transformed_image = image_tensor.clone()
    for _ in range(iterations):
        new_image = torch.zeros_like(transformed_image)
        for x in range(N):
            for y in range(N):
                x_new = (x + y) % N
                y_new = (x + 2 * y) % N
                new_image[:, x_new, y_new] = transformed_image[:, x, y]
        transformed_image = new_image
    return transformed_image
def reverse_arnold_transform_tensor(image_tensor: torch.Tensor, iterations: int = 1) -> torch.Tensor:
    N = image_tensor.shape[1]
    transformed_image = image_tensor.clone()
    for _ in range(iterations):
        new_image = torch.zeros_like(transformed_image)
        for x in range(N):
            for y in range(N):
                x_new = (2 * x - y) % N
                y_new = (-x + y) % N
                new_image[:, x_new, y_new] = transformed_image[:, x, y]
        transformed_image = new_image
    return transformed_image
```

Figure 71: Code Snippet for Arnold Transformation



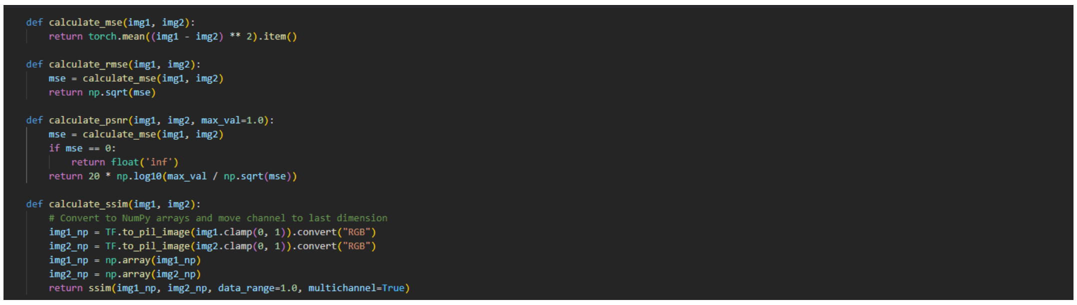
```python
# Steganographic Key generaion
def string_to_number(text):
    return '0000'+str(abs(hash(text)) % 1_000)

def seperate_key(key):
    if len(str(key)) != 7:
        raise ValueError("Key must be a 7-digit integer.")
    key_str = str(key)
    sk1, sk2, sk3, sk4, sk5, sk6, sk7 = map(int, key_str)
    return sk1, sk2, sk3, sk4, sk5, sk6, sk7
```

Figure 72: Code Snippet for Steganographic key generation



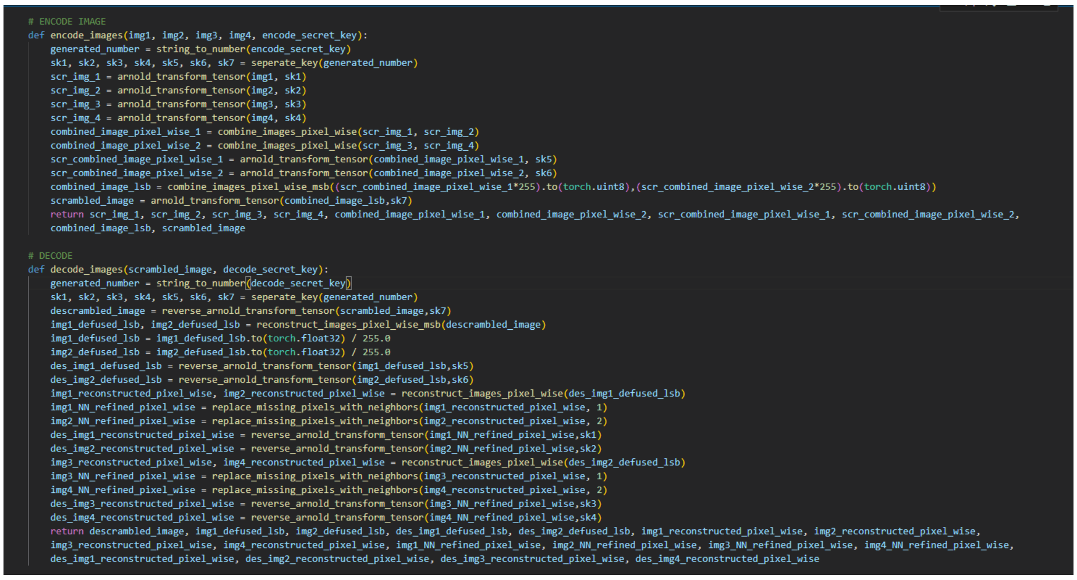
```python
# Error value
def calculate_mean_pixel_difference(image1: torch.Tensor, image2: torch.Tensor, absolute: bool = True) -> float:
    if image1.shape != image2.shape:
        raise ValueError("Error: Tensors must have the same shape.")
    difference = image1 - image2
    if absolute:
        mean_difference = torch.mean(torch.abs(difference))
    else:
        mean_difference = torch.mean(difference ** 2)
    return mean_difference.item()
```

Figure 73: Code Snippet for Calculation Difference Error 1



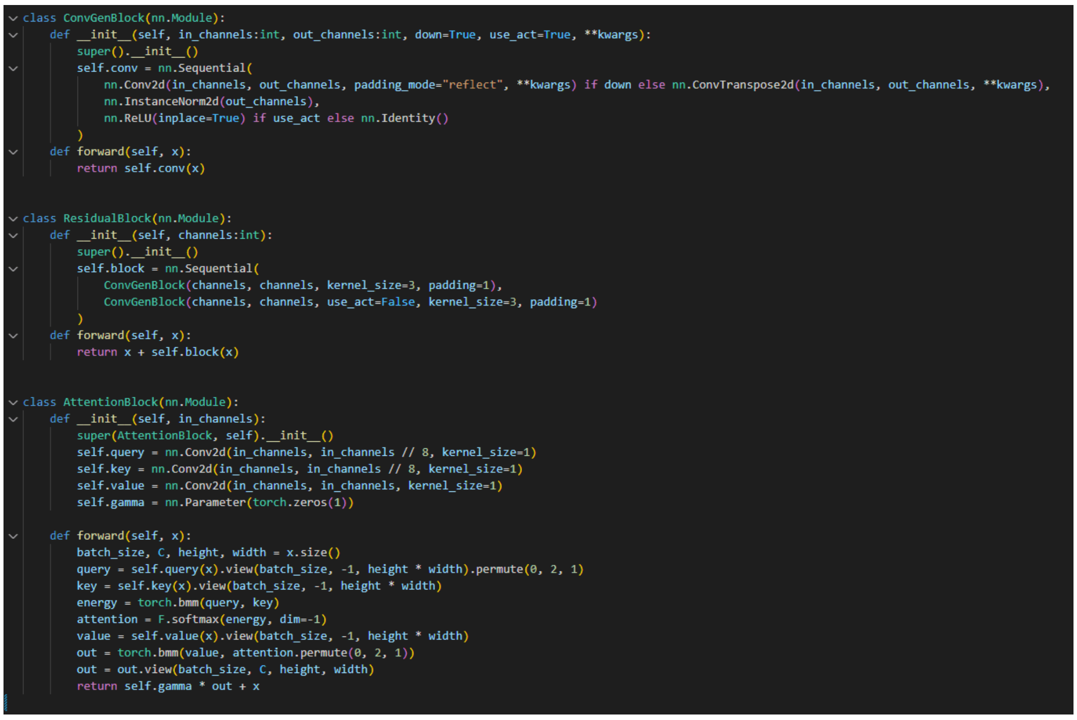
```python
def calculate_mse(img1, img2):
    return torch.mean((img1 - img2) ** 2).item()

def calculate_rmse(img1, img2):
    mse = calculate_mse(img1, img2)
    return np.sqrt(mse)

def calculate_psnr(img1, img2, max_val=1.0):
    mse = calculate_mse(img1, img2)
    if mse == 0:
        return float('inf')
    return 20 * np.log10(max_val / np.sqrt(mse))

def calculate_ssim(img1, img2):
    # Convert to NumPy arrays and move channel to last dimension
    img1_np = TF.to_pil_image(img1.clamp(0, 1)).convert("RGB")
    img2_np = TF.to_pil_image(img2.clamp(0, 1)).convert("RGB")
    img1_np = np.array(img1_np)
    img2_np = np.array(img2_np)
    return ssim(img1_np, img2_np, data_range=1.0, multichannel=True)
```

Figure 74: Code Snippet for Calculation Difference Error 2



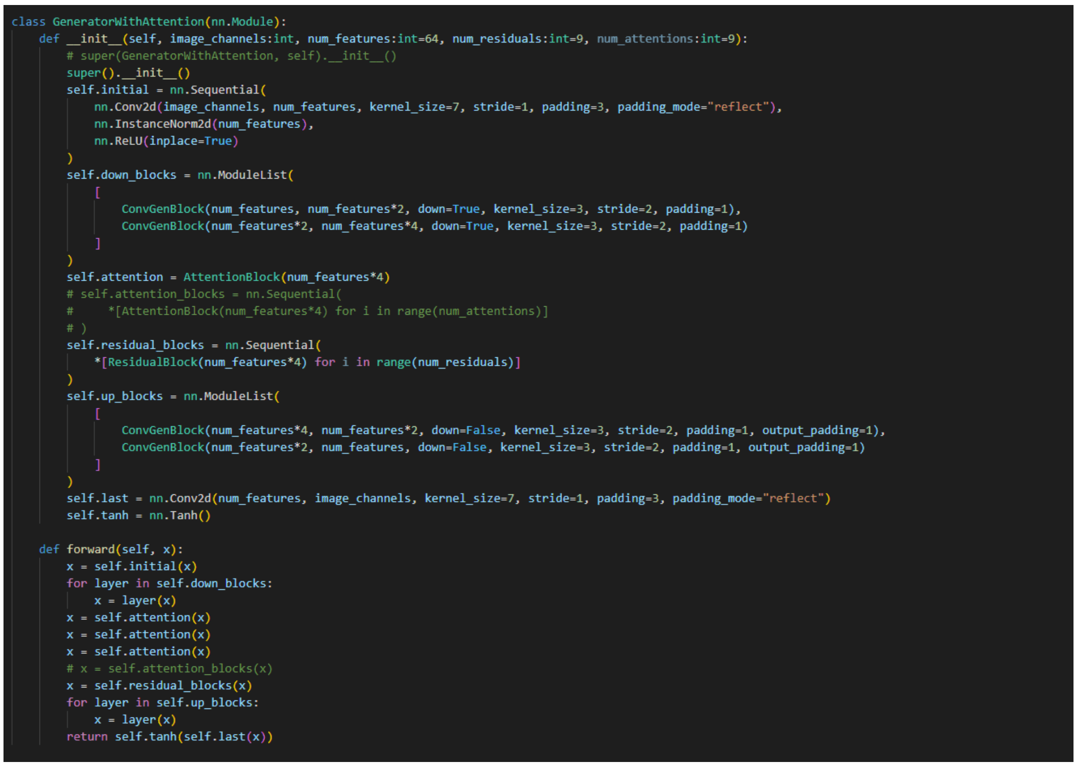
```
# ENCODE IMAGE
def encode_images(img1, img2, img3, img4, encode_secret_key):
    generated_number = string_to_number(encode_secret_key)
    sk1, sk2, sk3, sk4, sk5, sk6, sk7 = seperate_key(generated_number)
    scr_img_1 = arnold_transform_tensor(img1, sk1)
    scr_img_2 = arnold_transform_tensor(img2, sk2)
    scr_img_3 = arnold_transform_tensor(img3, sk3)
    scr_img_4 = arnold_transform_tensor(img4, sk4)
    combined_image_pixel_wise_1 = combine_images_pixel_wise(scr_img_1, scr_img_2)
    combined_image_pixel_wise_2 = combine_images_pixel_wise(scr_img_3, scr_img_4)
    scr_combined_image_pixel_wise_1 = arnold_transform_tensor(combined_image_pixel_wise_1, sk5)
    scr_combined_image_pixel_wise_2 = arnold_transform_tensor(combined_image_pixel_wise_2, sk6)
    combined_image_lsb = combine_images_pixel_wise_msb((scr_combined_image_pixel_wise_1*255).to(torch.uint8),(scr_combined_image_pixel_wise_2*255).to(torch.uint8))
    scrambled_image = arnold_transform_tensor(combined_image_lsb,sk7)
    return scr_img_1, scr_img_2, scr_img_3, scr_img_4, combined_image_pixel_wise_1, combined_image_pixel_wise_2, scr_combined_image_pixel_wise_1, scr_combined_image_pixel_wise_2,
    combined_image_lsb, scrambled_image

# DECODE
def decode_images(scrambled_image, decode_secret_key):
    generated_number = string_to_number(decode_secret_key)
    sk1, sk2, sk3, sk4, sk5, sk6, sk7 = seperate_key(generated_number)
    descrambled_image = reverse_arnold_transform_tensor(scrambled_image,sk7)
    img1_defused_lsb, img2_defused_lsb = reconstruct_images_pixel_wise_msb(descrambled_image)
    img1_defused_lsb = img1_defused_lsb.to(torch.float32) / 255.0
    img2_defused_lsb = img2_defused_lsb.to(torch.float32) / 255.0
    des_img1_defused_lsb = reverse_arnold_transform_tensor(img1_defused_lsb,sk5)
    des_img2_defused_lsb = reverse_arnold_transform_tensor(img2_defused_lsb,sk6)
    img1_reconstructed_pixel_wise, img2_reconstructed_pixel_wise = reconstruct_images_pixel_wise(des_img1_defused_lsb)
    img1_NN_refined_pixel_wise = replace_missing_pixels_with_neighbors(img1_reconstructed_pixel_wise, 1)
    img2_NN_refined_pixel_wise = replace_missing_pixels_with_neighbors(img2_reconstructed_pixel_wise, 2)
    des_img1_reconstructed_pixel_wise = reverse_arnold_transform_tensor(img1_NN_refined_pixel_wise,sk1)
    des_img2_reconstructed_pixel_wise = reverse_arnold_transform_tensor(img2_NN_refined_pixel_wise,sk2)
    img3_reconstructed_pixel_wise, img4_reconstructed_pixel_wise = reconstruct_images_pixel_wise(des_img2_defused_lsb)
    img3_NN_refined_pixel_wise = replace_missing_pixels_with_neighbors(img3_reconstructed_pixel_wise, 1)
    img4_NN_refined_pixel_wise = replace_missing_pixels_with_neighbors(img4_reconstructed_pixel_wise, 2)
    des_img3_reconstructed_pixel_wise = reverse_arnold_transform_tensor(img3_NN_refined_pixel_wise,sk3)
    des_img4_reconstructed_pixel_wise = reverse_arnold_transform_tensor(img4_NN_refined_pixel_wise,sk4)
    return descrambled_image, img1_defused_lsb, img2_defused_lsb, des_img1_defused_lsb, des_img2_defused_lsb, img1_reconstructed_pixel_wise, img2_reconstructed_pixel_wise,
    img3_reconstructed_pixel_wise, img4_reconstructed_pixel_wise, img1_NN_refined_pixel_wise, img2_NN_refined_pixel_wise, img3_NN_refined_pixel_wise, img4_NN_refined_pixel_wise,
    des_img1_reconstructed_pixel_wise, des_img2_reconstructed_pixel_wise, des_img3_reconstructed_pixel_wise, des_img4_reconstructed_pixel_wise
```

Figure 75: Code Snippet for Encoder and Decoder of Steagnographic key based coverless multi image steganography



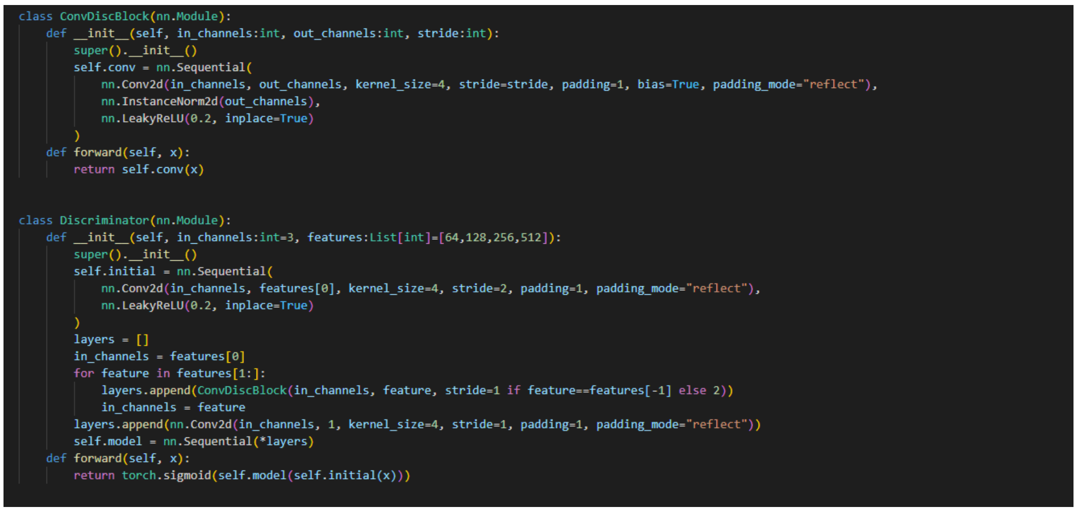
```
class ConvGenBlock(nn.Module):
    def __init__(self, in_channels:int, out_channels:int, down=True, use_act=True, **kwargs):
        super().__init__()
        self.conv = nn.Sequential(
            nn.Conv2d(in_channels, out_channels, padding_mode="reflect", **kwargs) if down else nn.ConvTranspose2d(in_channels, out_channels, **kwargs),
            nn.InstanceNorm2d(out_channels),
            nn.ReLU(inplace=True) if use_act else nn.Identity()
        )
    def forward(self, x):
        return self.conv(x)


class ResidualBlock(nn.Module):
    def __init__(self, channels:int):
        super().__init__()
        self.block = nn.Sequential(
            ConvGenBlock(channels, channels, kernel_size=3, padding=1),
            ConvGenBlock(channels, channels, use_act=False, kernel_size=3, padding=1)
        )
    def forward(self, x):
        return x + self.block(x)


class AttentionBlock(nn.Module):
    def __init__(self, in_channels):
        super(AttentionBlock, self).__init__()
        self.query = nn.Conv2d(in_channels, in_channels // 8, kernel_size=1)
        self.key = nn.Conv2d(in_channels, in_channels // 8, kernel_size=1)
        self.value = nn.Conv2d(in_channels, in_channels, kernel_size=1)
        self.gamma = nn.Parameter(torch.zeros(1))

    def forward(self, x):
        batch_size, C, height, width = x.size()
        query = self.query(x).view(batch_size, -1, height * width).permute(0, 2, 1)
        key = self.key(x).view(batch_size, -1, height * width)
        energy = torch.bmm(query, key)
        attention = F.softmax(energy, dim=-1)
        value = self.value(x).view(batch_size, -1, height * width)
        out = torch.bmm(value, attention.permute(0, 2, 1))
        out = out.view(batch_size, C, height, width)
        return self.gamma * out + x
```

Figure 76: Code Snippet for Generator Layers



```python
class GeneratorWithAttention(nn.Module):
    def __init__(self, image_channels:int, num_features:int=64, num_residuals:int=9, num_attentions:int=9):
        # super(GeneratorWithAttention, self).__init__()
        super().__init__()
        self.initial = nn.Sequential(
            nn.Conv2d(image_channels, num_features, kernel_size=7, stride=1, padding=3, padding_mode="reflect"),
            nn.InstanceNorm2d(num_features),
            nn.ReLU(inplace=True)
        )
        self.down_blocks = nn.ModuleList(
            [
                ConvGenBlock(num_features, num_features*2, down=True, kernel_size=3, stride=2, padding=1),
                ConvGenBlock(num_features*2, num_features*4, down=True, kernel_size=3, stride=2, padding=1)
            ]
        )
        self.attention = AttentionBlock(num_features*4)
        # self.attention_blocks = nn.Sequential(
        #     *[AttentionBlock(num_features*4) for i in range(num_attentions)]
        # )
        self.residual_blocks = nn.Sequential(
            *[ResidualBlock(num_features*4) for i in range(num_residuals)]
        )
        self.up_blocks = nn.ModuleList(
            [
                ConvGenBlock(num_features*4, num_features*2, down=False, kernel_size=3, stride=2, padding=1, output_padding=1),
                ConvGenBlock(num_features*2, num_features, down=False, kernel_size=3, stride=2, padding=1, output_padding=1)
            ]
        )
        self.last = nn.Conv2d(num_features, image_channels, kernel_size=7, stride=1, padding=3, padding_mode="reflect")
        self.tanh = nn.Tanh()

    def forward(self, x):
        x = self.initial(x)
        for layer in self.down_blocks:
            x = layer(x)
        x = self.attention(x)
        x = self.attention(x)
        x = self.attention(x)
        # x = self.attention_blocks(x)
        x = self.residual_blocks(x)
        for layer in self.up_blocks:
            x = layer(x)
        return self.tanh(self.last(x))
```

Figure 77: Code Snippet for Generator



```python
class ConvDiscBlock(nn.Module):
    def __init__(self, in_channels:int, out_channels:int, stride:int):
        super().__init__()
        self.conv = nn.Sequential(
            nn.Conv2d(in_channels, out_channels, kernel_size=4, stride=stride, padding=1, bias=True, padding_mode="reflect"),
            nn.InstanceNorm2d(out_channels),
            nn.LeakyReLU(0.2, inplace=True)
        )
    def forward(self, x):
        return self.conv(x)


class Discriminator(nn.Module):
    def __init__(self, in_channels:int=3, features:List[int]=[64,128,256,512]):
        super().__init__()
        self.initial = nn.Sequential(
            nn.Conv2d(in_channels, features[0], kernel_size=4, stride=2, padding=1, padding_mode="reflect"),
            nn.LeakyReLU(0.2, inplace=True)
        )
        layers = []
        in_channels = features[0]
        for feature in features[1:]:
            layers.append(ConvDiscBlock(in_channels, feature, stride=1 if feature==features[-1] else 2))
            in_channels = feature
        layers.append(nn.Conv2d(in_channels, 1, kernel_size=4, stride=1, padding=1, padding_mode="reflect"))
        self.model = nn.Sequential(*layers)
    def forward(self, x):
        return torch.sigmoid(self.model(self.initial(x)))
```

Figure 78: Code Snippet for Discriminator