Міністерство освіти і науки України
Харківський національний університет ім. В. Н. Каразіна
Факультет комп'ютерних наук
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

"Допущено до захисту"
В.о. завідувача кафедри БІСТ
Мелкозьорова О.М._____
"_____" червня 2024р.

Пояснювальна записка

до кваліфікаційної роботи бакалавра

на тему: «Дослідження існуючих методів стеганографії в зображеннях: розробка та порівняльний аналіз»

оцінка « »

Голова ЕК

Лемешко О. В._____

Керівник ст. викладач Шеханін К. Ю.

Рецензент _____Лисицький К. Є.

Виконавець: студент групи КБ-41

_____Анохін Д. А.

Харків – 2024

РЕФЕРАТ

Пояснювальна записка до дипломного проекту містить 63 сторінок, 21 рисунок, 7 таблиць, 4 додатки, 4 лістинги та 43 посилання на джерела.

Метою даної дипломної роботи є дослідження існуючих методів стеганографії в зображеннях, їх розвиток та порівняльний аналіз. Основні завдання роботи включають огляд теоретичних аспектів стеганографії, аналіз існуючих методів приховування інформації у зображеннях, розробку досліджених методів на основі аналізу та порівняння їх ефективності.

Предмет дослідження включає вивчення різних підходів та інструментів для вбудовування інформації в зображення, аналіз ефективності та стійкості цих методів до виявлення та спотворень.

Об'єктом роботи є цифрові зображення, в які вбудовується прихована інформація за допомогою різних стеганографічних методів.

У результаті дослідження проведено огляд основних понять та визначень стеганографії, розглянуто області застосування та практичні аспекти реалізації стеганографічних систем. Здійснено порівняльний аналіз існуючих методів приховування даних у зображеннях, включаючи методи приховування у просторовій та частотній областях. Також описано процес розробки досліджених стеганографічних методів, таких як метод LSB та метод Коха-Жао, і проведено їх порівняння за кількісними та якісними критеріями оцінювання стеганосистем.

Результати дослідження можуть бути використані для подальшого розвитку технологій стеганографії та підвищення рівня захисту інформації в умовах сучасних кіберзагроз.

Ключові слова: СТЕГАНОГРАФІЯ, ЗАХИСТ ІНФОРМАЦІЇ, ПОРІВНЯЛЬНИЙ АНАЛІЗ, ЦИФРОВІ ЗОБРАЖЕННЯ, МЕТОД LSB, МЕТОД КОХА-ЖАО, ПРИХОВУВАНН ІНФОРМАЦІЇ, АНАЛІЗ ЕФЕКТИВНОСТІ.

ABSTRACT

The explanatory note to the diploma project comprises 63 pages, 21 figures, 7 tables, 4 appendices, 4 listings, and 43 references.

The aim of this diploma work is to investigate existing methods of steganography in images, their development, and comparative analysis. The primary tasks of the work include reviewing the theoretical aspects of steganography, analyzing existing methods of hiding information in images, and developing researched methods based on the analysis and comparison of their effectiveness.

The subject of the research includes the study of various approaches and tools for embedding information in images, as well as analyzing the effectiveness and robustness of these methods against detection and distortion.

The object of the work is digital images into which hidden information is embedded using various steganographic methods.

The research results provide an overview of the main concepts and definitions of steganography, examine the areas of application, and discuss the practical aspects of implementing steganographic systems. A comparative analysis of existing methods of hiding data in images, including methods of hiding in spatial and frequency domains, is conducted. The process of developing researched steganographic methods, such as the LSB method and the Koch-Zhao method, is described, and their comparison is made based on quantitative and qualitative criteria for evaluating steganographic systems.

The research results can be used for further development of steganography technologies and enhancing the level of information protection under modern cyber threats.

Keywords: STEGANOGRAPHY, INFORMATION PROTECTION, COMPARATIVE ANALYSIS, DIGITAL IMAGES, LSB METHOD, KOCH-ZHAO METHOD, INFORMATION HIDING, EFFECTIVENESS ANALYSIS.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

AD - Average Difference

API - Application Programming Interface

BMP - Bitmap Image File

DCT - Discrete Cosine Transform

DVD - Digital Versatile Disc

DWT - Discrete Wavelet Transform

FFT - Fast Fourier Transform

GIF - Graphics Interchange Format

GLM - Gray Level Modification

IDE - Integrated Development Environment

IF - Image Fidelity

IT - Information Technology

JCA - Java Cryptography Architecture

JPEG - Joint Photographic Experts Group

JVM - Java Virtual Machine

LSB - Least Significant Bit

MD - Maximum Difference

NC - Normalized Cross-Correlation

PNG - Portable Network Graphics

PSNR - Peak Signal-to-Noise Ratio

PVD - Pixel Value Difference

RGB - Red Green Blue

SNR - Signal-to-Noise Ratio

TIFF - Tagged Image File Format

WebP - Web Picture Format

ВЧ – Високочастотний

ДКП - Дискретне Косинусне Перетворення

ЗДКП - Зворотне Дискретне Косинусне Перетворення

ЗС - Зорова Система

ЗСЛ - Зорова Система Людини

КС - Комп'ютерна Стеганографія

НЧ – Низькочастотний

ООП - Об'єктно-Орієнтоване Програмування

СЧ – Середньочастотний

ЦВЗ - Цифровий Водний Знак

ЦОС - Цифрова Обробка Сигналів

ВСТУП

У сучасному цифровому світі проблема захисту інформації та забезпечення її конфіденційності набуває все більшої актуальності. Розвиток інформаційних технологій відкриває нові можливості для передачі та обміну даними, проте водночас породжує нові загрози та виклики в галузі інформаційної безпеки. Одним із ефективних інструментів для приховування факту передачі даних є стеганографія – наука про методи вбудовування інформації у цифрові об'єкти, такі як зображення, аудіо чи відео файли.

Стеганографія знаходить широке застосування в різноманітних сферах, зокрема для захисту авторських прав, секретної передачі повідомлень, цифрового маркування контенту тощо. Особливої актуальності ця технологія набуває в умовах зростаючих кіберзагроз та необхідності забезпечення захищених комунікацій. Проте, досягнення високої ефективності та стійкості стеганографічних методів залишається складним завданням через постійну появу нових викликів, таких як активні атаки на стеганосистеми, вимоги до підвищення пропускної здатності каналів прихованої передачі даних, необхідність забезпечення стійкості до спотворень тощо.

Актуальність теми дослідження зумовлена зростанням кількості кіберзагроз та необхідністю вдосконалення методів захисту інформації в цифровому середовищі. У світі, де обсяг цифрових даних постійно зростає, важливо вміти розробляти ефективні методи приховування інформації, які дозволять забезпечити конфіденційність та цілісність переданих даних. Стеганографія, як один із перспективних методів інформаційної безпеки, потребує постійного розвитку та вдосконалення, щоб відповідати сучасним вимогам та викликам.

Метою даної дипломної роботи є дослідження існуючих методів стеганографії в зображеннях, їх розвиток та порівняльний аналіз. Основні завдання роботи включають огляд теоретичних аспектів стеганографії, аналіз існуючих методів приховування інформації у зображеннях, розробку досліджених методів на основі аналізу та порівняння їх ефективності.

У першій частині роботи розглядаються основні поняття та визначення стеганографії, а також області застосування та практичні аспекти реалізації стеганографічних систем. Далі здійснюється огляд існуючих методів приховування даних у зображеннях, включаючи методи приховування у просторовій та частотній областях. У третій частині роботи описується процес розробки досліджуваних стеганографічних методів, таких як метод LSB та метод Коха-Жао, а також інструменти, які були використані для їх реалізації. Завершує роботу порівняльний аналіз досліджених та розроблених методів з урахуванням кількісних критеріїв оцінювання стеганосистем.

Таким чином, дана дипломна робота має на меті внесення нового вкладу у розвиток технологій стеганографії, що сприятиме підвищенню рівня захисту інформації в умовах сучасних кіберзагроз.

1. ТЕОРЕТИЧНІ АСПЕКТИ СТЕГАНОГРАФІЇ

1.1 Основні поняття та визначення стеганографії

Історія стеганографії сягає ще давніх часів, коли люди намагалися використовувати приховану інформацію для передачі таємних повідомлень, зберігаючи їх вміст від розкриття.

Стеганографія має походження від грецьких слів "steganos", що означає прихований або таємний, та "graphein" - писати. Стеганографію можна описати як науку про методи приховування факту передачі інформації шляхом вбудовування її у інші, нешкідливі на перший погляд, дані. На відміну від криптографії, яка займається захистом змісту повідомлення, стеганографія приховує сам факт передачі інформації [1].

Приховування інформації шляхом стеганографії має багатовікову історію. Шифрування секретної інформації з'явилося ще за часів греків [2]. Греки використовували його для відправки важливих повідомлень під час війн. Прихована інформація спочатку була безладною. Повідомлення надсилалися пішки, якщо ви хотіли сховати лист, у вас було два варіанти: запам'ятати його гінцем, або сховати його на гінці.

Один з відомих стародавніх методів приховування інформації полягав у написанні повідомлень на дерев'яних дощечках, вкритих воском. Поверх цього воскового шару зазвичай писали безневинне, на перший погляд, повідомлення. Однак під цим верхнім шаром прихованим чином розміщувалася таємна інформація. Такий спосіб дозволяв передавати секретні відомості, не викликаючи підозр у тих, хто перехоплював дощечки. Цей метод отримав широке поширення у Стародавній Греції та Римі. Іншим поширеним прийомом стародавньої стеганографії було голення голови раба та нанесення тексту повідомлення безпосередньо на його шкіру. Після цього раба відправляли до адресата, і доки волосся знову не відросло, текст залишався прихованим.

Ще одним прикладом стародавньої стеганографії було застосування так званих "симпатичних чорнил" - особливих речовин, які ставали видимими лише

при нагріванні або обробці хімічними реагентами. Таким чином, зовні безневинні записи містили приховане повідомлення, яке проявлялося лише за певних умов. Цей метод дозволяв передавати секретну інформацію, не привертаючи уваги перехоплювачів.

Проаналізувавши історію стеганографії, можна провести аналогію та зрозуміти, що основною метою стеганографії на сьогодення є забезпечення таємної передачі даних шляхом їх маскування в "невинних" на вигляд цифрових об'єктах, таких як зображення, аудіо, відео тощо. Таким чином, стеганографія дозволяє приховати не лише зміст, але й сам факт існування повідомлення [3].

У сучасному світі дані надсилаються через інтернет, щоб запобігти будь-якому несанкціонованого доступу шляхом шифрування. Різні методи, що використовуються для приховування інформації можна класифікувати так як показано на рис. 1.1.

Рисунок 1.1 – Техніки приховування інформації [4]

Як вже було зазначено раніше, стеганографія - це наука і мистецтво спілкування у спосіб, який приховує секретні дані всередині інших даних, які називаються даними прикриття. Перевага стеганографії полягає в тому, щоб приховати секретне повідомлення всередині даних прикриття так, що ніхто не може навіть виявити друге повідомлення в даних прикриття. Різні типи носіїв-

контейнерів показані на рис. 1.2. Криптографія походить з грецької мови, що означає "прикрите письмо". Криптографія - це метод передачі повідомлення в чітко визначеній формі таким чином, щоб його міг прочитати та інтерпретувати передбачуваний одержувач [5].

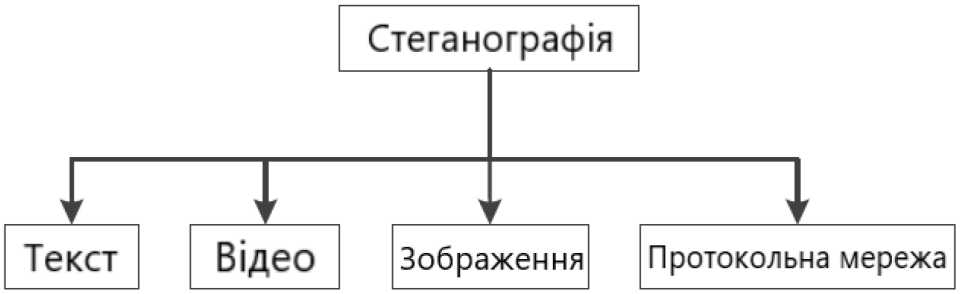

Рисунок 1.2 – Різноманітні типи носіїв-контейнерів

Історично, стеганографія була одним з перших способів приховання інформації, але з часом її значення зменшилося через розвиток криптографії. Проте, в останні десятиліття інтерес до стеганографії відродився, завдяки широкому поширенню мультимедійних технологій та появі нових каналів передачі інформації. Це призвело до розвитку нових методів стеганографії, які базуються на особливостях комп'ютерних файлів та мереж. Таким чином, можна говорити про зародження такого напряму у сфері захисту інформації, як комп'ютерна стеганографія.

Комп'ютерна стеганографія (КС) має два важливі шляхи застосування: перший стосується цифрової обробки сигналів (ЦОС), а другий - ні. У першому випадку прихована інформація вбудована безпосередньо у цифрові дані, які зазвичай мають аналогову структуру, таку як мова, аудіозапис, зображення або відеозапис. Стосовно другого напрямку, приватна інформація розміщується в заголовку файлу або пакета даних, але цей підхід, як правило, не використовується через його відносну простоту отримання та/або знищення конфіденційних даних [6].

Незважаючи на існування різних методів комп'ютерної стеганографії, все ж таки більшість сучасних досліджень у цій галузі так чи інакше пов'язані саме з цифровою обробкою сигналів, що дозволяє виокремити такий напрям, як цифрова стеганографія. [7]

Цифрова стеганографія є напрямом класичної стеганографії, який використовується для вбудовування додаткової інформації в цифрові об'єкти, такі як зображення, аудіо або відео файли. Слід зауважити, що цей процес зазвичай призводить до певних спотворень вихідних об'єктів. Ця технологія була створена для встановлення таємного зв’язку, що є класичним призначенням стеганографії, але в останній час вона також застосовується з метою захисту інтелектуальної власності [8].

Стеганографічні методи можна умовно поділити на кілька основних категорій. Серед них можна виділити такі категорії:

1) Класична стеганографія – спосіб приховування даних, який реалізується завдяки використанню технічних засобів захисту інформації [6]. На сьогодення класична стеганографія включає в себе такі методи (рис. 1.3):

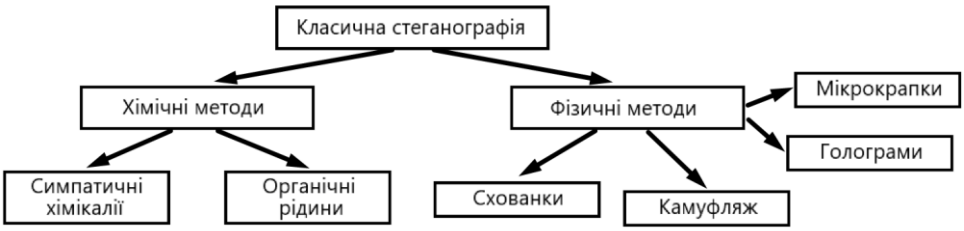

Рисунок 1.3 – Методи класичної стеганографії [9]

2) Цифрова стеганографія - вбудовування повідомлень у цифрові об'єкти, такі як зображення, аудіо, відео файли [6, 10]. До цифрової стеганографії, можна віднести такі методи (рис. 1.4):

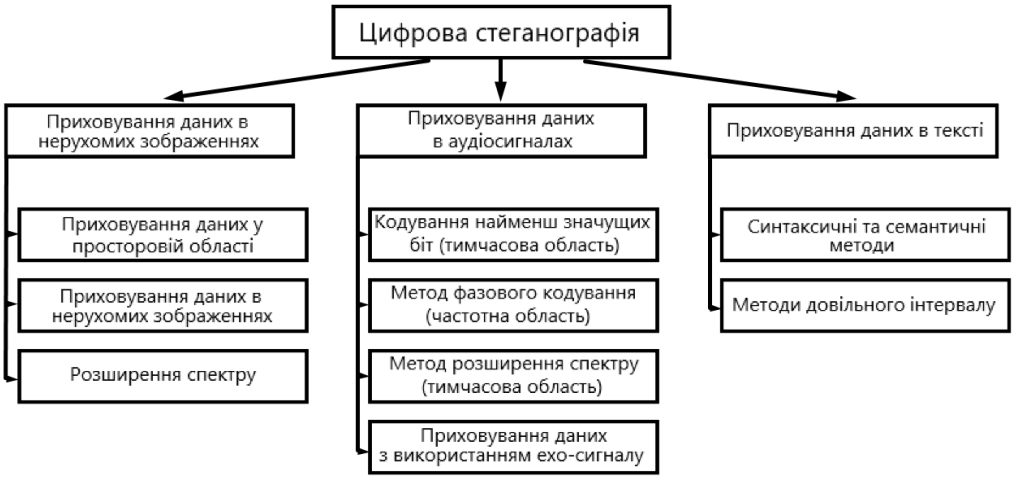

Рисунок 1.4 – Методи цифрова стеганографії [9]

3) Лінгвістична стеганографія - приховування інформації в природній мові, наприклад, шляхом використання прихованих значень слів, граматичних конструкцій тощо [11, 12]. Лінгвістична стеганографія розділяється на такі основні методи (рис. 1.5):

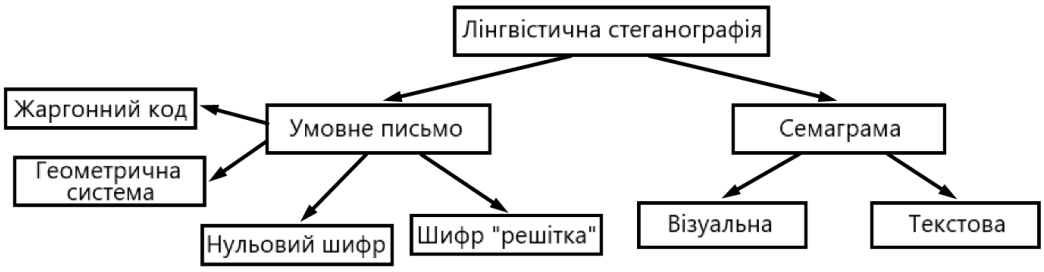

Рисунок 1.5 – Методи лінгвістичної стеганографії [9]

Крім того, слід зазначити ключові поняття, які зазвичай використовуються у стеганографії.

Контейнер (cover object) - цифровий об'єкт (зображення, аудіо, відео), в який вбудовується приховане повідомлення. Контейнер повинен виглядати природно та не викликати підозр.

Повідомлення (message) - інформація, що підлягає прихованій передачі. Повідомлення може бути текстом, зображенням, аудіо/відео файлом або будь-якими іншими даними.

Стеганоключ (stegokey) - секретний ключ, який визначає алгоритм вбудовування/вилучення повідомлення з контейнера. Стеганоключ забезпечує конфіденційність та цілісність прихованої інформації.

Стеганоконтейнер (stego-object) - контейнер, модифікований для вбудовування прихованого повідомлення. Стеганоконтейнер повинен зовні нічим не відрізнятися від звичайного контейнера.

Стеганоаналіз (steganalysis) - процес виявлення та/або вилучення прихованої інформації із стеганоконтейнера [3].

1.2 Області застосування та практичні аспекти реалізації стеганографічних систем

Стеганографія знаходить широке застосування в різних галузях, де потрібно забезпечити конфіденційність та цілісність інформації. Основними напрямами застосування стеганографії є:

1) Вбудовування інформації з метою її прихованої передачі.
2) Вбудовування цифрових водяних знаків для реалізації захисту від копіювання та несанкціонованого використання.
3) Вбудовування ідентифікаційних номерів для відстеження несанкціонованого тиражування.
4) Вбудовування заголовків для збереження інформації в єдиному цифровому об'єкті.

Можна побачити, що напрями застосування стеганографії є розгалуженими, тому можна назвати різноманітні сфери застосування, де ця технологія може бути використана. Виходячи з цього, виділимо такі області застосування стеганографії, які зображені у таблиці 1.1 [13].

Таблиця 1.1 – Області застосування стеганографії

Область застосування	Приклади застосування
1. Аутентифікація	Системи відео-спостереження, голосової пошти, електронної комерції, електронне конфіденційне діловодство.
2. Захист від копіювання	Контроль за копіюванням (DVD), електронна комерція, розповсюдження мультимедійної інформації (відео за запитом).
3. Прихована анотація документів	Картографія, медичні знімки, мультимедійні бази даних.
4. Прихований зв'язок	Розвідувальні та військові додатки, а також у випадках, коли криптографію використовувати не є можливим

Набором методів та інструментів, що використовуються для створення прихованого каналу передачі інформації називають стеганографічною системою, яку представлено на рис. 1.6. Контейнер або дані прикриття - це назва об'єкта або дані, які використовуються для приховування секретної інформації

(повідомлення). Стеганоданими називаються дані, які отримані шляхом вбудовування секретних даних у дані прикриття. Дані прикриття надсилаються одержувачу через мережу [14]. Методи стеганографії можна розділити на методи просторової області та перетворення [15].

Рисунок 1.6 – Узагальнена модель стеганосистеми [16]

Як можна побачити з рис. 1.6, для реалізації стеганографічних систем використовуються багато елементів, детальний опис яких буде наведено нижче.

Повідомлення - це дані або інформація, яку потрібно приховати. Це може бути текст, зображення, аудіо, відео або будь-який інший цифровий контент.

Контейнер - це файл або носій, такий як зображення, аудіо, відео або текстовий документ, в який буде вбудовуватися приховане повідомлення. Контейнер повинен бути достатнього розміру, щоб вмістити повідомлення без помітних змін зовнішнього вигляду або властивостей.

Стеганокодер - це алгоритм або програмний компонент, який здійснює процес вбудовування прихованого повідомлення в контейнер. Він використовує стеганоключ та певний метод для приховування даних.

Стегоконтейнер - це результат роботи стеганокодера, тобто контейнер з вбудованим повідомленням. Зовнішній вигляд і властивості стегоконтейнера повинні бути максимально близькими до оригінального контейнера.

Канал прихованого обміну (стеганоканал) - це середовище, через яке стегоконтейнер передається від відправника до одержувача. Це може бути мережа Інтернет, фізичний носій даних або будь-який інший канал передачі інформації.

Стеганодекодер - це алгоритм або програмний компонент, який відповідає за витягування прихованого повідомлення зі стегоконтейнера. Він використовує

той самий стеганоключ, що і стеганокодер, для визначення місця розташування та методу вбудовування повідомлення.

Стеганоключ - це секретний ключ, який використовується як стеганокодером, так і стеганодекодером для забезпечення коректного вбудовування та витягування повідомлення. Ключ повинен бути відомий лише відправнику та одержувачу для забезпечення конфіденційності стеганосистеми.

Практична реалізація стеганографічних систем, як правило, включає кілька ключових етапів. Першим кроком є вибір відповідного контейнера для вбудовування повідомлення. Контейнер має бути достатньо ємним, щоб вмістити повідомлення, але при цьому не викликати підозр у потенційного порушника. Зазвичай для цього обирають різні цифрові медіа-файли, такі як зображення, аудіо чи відео, які можуть легко переміщуватись каналами зв'язку [18].

Наступним кроком є розробка ефективного алгоритму для вбудовування повідомлення в обраний контейнер. Цей алгоритм повинен забезпечувати максимальну невидимість прихованого повідомлення, щоб воно не привертало увагу порушника. Залежно від типу контейнера та виду повідомлення, можуть використовуватись різноманітні стеганографічні техніки, такі як модуляція з розширенням спектра, квантування коефіцієнтів дискретного косинусного перетворення, чи приховування у найменш значущих бітах.

Після вбудовування повідомлення, стегоконтейнер може бути безпечно переданий одержувачу. Цей етап також вимагає уважності, щоб передача контейнера не викликала підозр у спостерігача. Можуть використовуватись різноманітні способи маскування, наприклад, передача контейнера в рамках звичайної ділової кореспонденції.

Нарешті, одержувач, що володіє необхідним стеганоключем, може вилучити приховане повідомлення зі стегоконтейнера. Ключ може бути обраний заздалегідь між відправником і одержувачем, або ж бути частиною самого алгоритму вбудовування. Завдяки стеганоключу одержувач може однозначно

ідентифікувати та витягти приховане повідомлення, в той час як сторонній спостерігач навіть не помітить його наявності.

Стеганографічні методи можна класифікувати за різними критеріями, які визначають їхні характеристики та сфери застосування. Одним із критеріїв є призначення методу: для забезпечення конфіденційності даних, захисту авторських прав або автентифікації даних.

Іншим важливим критерієм є принцип приховування повідомлень. Існують методи, які приховують дані в просторовій області (наприклад, заміна найменш значущих бітів у графічних файлах), та методи, які працюють у частотній області (наприклад, вбудовування даних у коефіцієнти перетворення).

Класифікація також здійснюється за форматом контейнера, в який вбудовуються приховані дані. Це можуть бути текстові файли, аудіофайли, зображення або відео. Окрім того, контейнери можуть бути потоковими (наприклад, потокове відео) або фіксованими (наприклад, статичне зображення).

Кожна з цих категорій має свої переваги та недоліки щодо стійкості до зовнішніх впливів та розміру вбудованого повідомлення. Розробники стеганографічних систем прагнуть досягти оптимального балансу між цими факторами для задоволення потреб певного застосування.

Загалом, існує компроміс між надійністю стеганографічної системи та обсягом даних, які можна приховати. Цю залежність можна представити графічно, як зазначено на рис. 1.7.

Рисунок 1.7 – Графічна залежність між надійністю стеганографічної системи та обсягом даних [17]

За цією залежністю можна помітити, що зі зростанням кількості вбудованих даних надійність системи погіршується, при цьому розмір контейнера залишається постійним.

1.3 Цифрові формати представлення зображень

Цифрові зображення можуть зберігатися в різних форматах, кожен з яких має свої особливості, переваги та недоліки. Вибір формату зображення є важливим аспектом при розробці стеганографічних методів, оскільки він визначає способи вбудовування та вилучення прихованої інформації [13].

Розглянемо основні формати цифрових зображень, що застосовуються в стеганографії (рис. 1.8).

BMP (Bitmap) - це один з найпростіших растрових форматів зображень, який не використовує жодного алгоритму стиснення. Кожен піксель зображення представлений окремим набором бітів, що визначають його колір. Хоча BMP-файли забезпечують найвищу якість зображення, їх основним недоліком є великий розмір, що робить їх менш зручними для передачі та зберігання. Втім, відсутність стиснення дозволяє легко вбудовувати дані в окремі біти кожного пікселя [13].

JPEG (Joint Photographic Experts Group) – є одним з найпопулярніших форматів зображень, особливо для зберігання та передачі фотографій. Він використовує алгоритм стиснення з втратами, заснований на дискретному косинусному перетворенні (ДКП) та квантуванні коефіцієнтів. Це дозволяє значно зменшити розмір файлу, але призводить до втрати деякої інформації та погіршення якості зображення. Для стеганографії в JPEG-зображеннях часто використовуються методи, що базуються на модифікації коефіцієнтів ДКП або заміні молодших бітів у квантованих коефіцієнтах [13].

PNG (Portable Network Graphics) – є растровим форматом із стисненням без втрат, що робить його ідеальним для зображень з чіткими лініями, текстом або іншими векторними елементами. PNG підтримує прозорість і може зберігати метадані. Хоча PNG-файли забезпечують високу якість зображення, їх розмір

зазвичай більший, ніж у форматів із стисненням з втратами. Для вбудовування даних у PNG-зображеннях часто використовується заміна молодших бітів пікселів або приховування інформації в метаданих [13].

GIF (Graphics Interchange Format) - це формат для зберігання зображень з обмеженою кольоровою палітрою (до 256 кольорів). Він використовує стиснення без втрат, що робить його придатним для зображень з простими кольорами та чіткими лініями, такими як логотипи або іконки. Однак, через обмежену колірну палітру, GIF не підходить для зберігання складних фотографій. У GIF-зображеннях дані можна вбудовувати, змінюючи молодші біти індексів кольорової палітри [13].

TIFF (Tagged Image File Format) - є гнучким растровим форматом, який підтримує різні типи стиснення, включаючи без втрат і з втратами. Він широко використовується в професійній обробці зображень та публікаціях через свою високу якість і можливість зберігання метаданих. TIFF-файли можуть бути досить великими, якщо використовується стиснення без втрат. Для вбудовування даних у TIFF-зображеннях можна використовувати методи, подібні до тих, що застосовуються для BMP- або JPEG-файлів, залежно від типу стиснення [13].

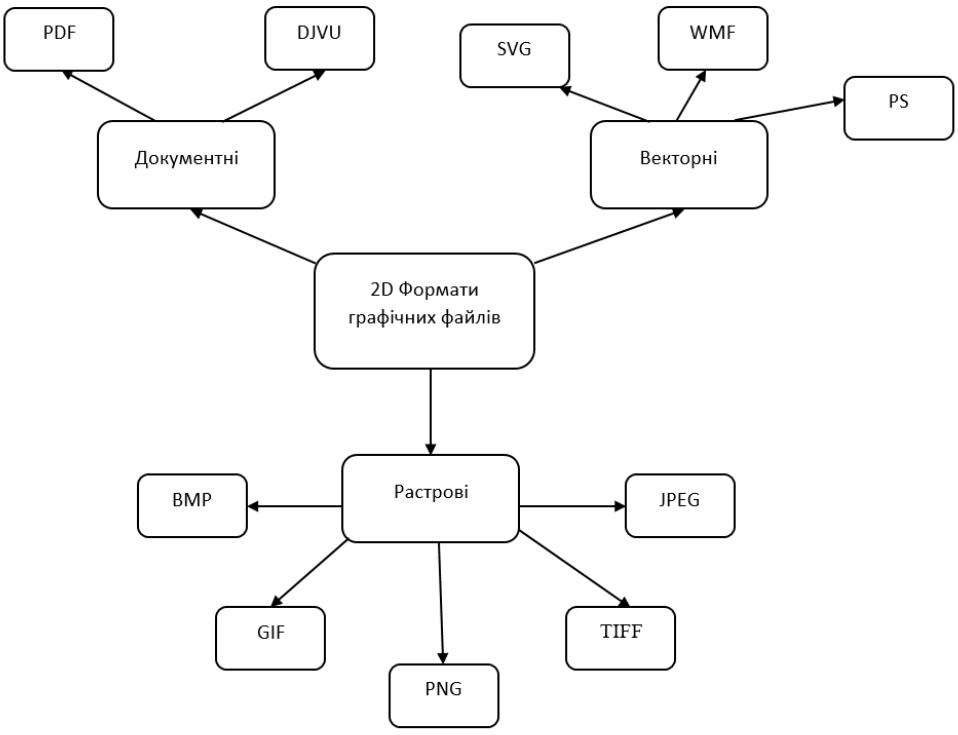

Рисунок 1.8 – Схематичний поділ 2D форматів графічних файлів

Кожен з цих форматів має свої особливості структури та алгоритмів кодування, що впливає на можливості та обмеження стеганографічних методів. Наприклад, у випадку JPEG-зображень стеганографічні дані можуть приховуватись у коефіцієнтах дискретного косинусного перетворення, тоді як для BMP-файлів доцільно використовувати приховування у молодших бітах пікселів. Для PNG-зображень можна застосовувати методи заміни молодших бітів пікселів або вбудовування даних у метадані файлу. У GIF-форматі дані можна вбудовувати, змінюючи індекси кольорової палітри. Для TIFF-зображень можна використовувати аналогічні підходи, як і для BMP або JPEG, залежно від типу стиснення. Для новітніх форматів, таких як WebP, стеганографічні методи можуть базуватися на модифікації коефіцієнтів перетворень, подібно до JPEG [19].

Вибір конкретного методу вбудовування даних залежить від особливостей формату зображення, таких як тип стиснення (з втратами чи без втрат), кольорова модель, наявність метаданих тощо. Наприклад, формати без втрат, такі як BMP, PNG та GIF, дозволяють зберігати приховану інформацію шляхом модифікації молодших бітів пікселів без значного спотворення зображення. З іншого боку, формати з втратами, як-от JPEG, вимагають більш витончених методів, щоб уникнути помітних спотворень через алгоритми стиснення.

При використанні стеганографічних методів, окрім особливостей структури та алгоритмів кодування, важливим аспектом є стійкість вбудованих даних до різних видів атак та спотворень. Стійкість визначається тим, наскільки добре приховані дані можуть зберегтися після проведення операцій обробки зображення, таких як обрізання, масштабування, фільтрація та стиснення. Різні формати зображень мають різну стійкість до таких атак.

Формати без втрат, такі як BMP, PNG та GIF, забезпечують високу стійкість до операцій без втрати інформації, таких як обрізання або масштабування. Однак вони можуть бути вразливими до операцій з втратами, таких як застосування фільтрів або додаткове стиснення з втратами.

З іншого боку, формати з втратами, такі як JPEG, вже зазнали втрати інформації під час первинного стиснення. Тому вони можуть краще протистояти додатковим операціям зі стисненням без значного погіршення якості зображення та втрати вбудованих даних. Проте, такі формати можуть бути більш вразливими до операцій обрізання чи масштабування.

Щодо формату TIFF, він може мати різну стійкість залежно від використаного типу стиснення (з втратами чи без втрат). Тому його стійкість треба окремо розглядати для кожного конкретного випадку.

2 ОГЛЯД ІСНУЮЧИХ МЕТОДІВ СТЕГАНОГРАФІЇ В ЗОБРАЖЕННЯХ

2.1 Особливості сприйняття зображень людською зоровою системою

Розуміння принципів функціонування людської зорової системи (ЗС) є важливим для практичної реалізації ефективних методів стеганографії, особливо в контексті приховування даних у цифрових зображеннях. Оскільки стеганографічні методи спрямовані на непомітне вбудовування повідомлень, знання особливостей людського зорового сприйняття дозволяє розробляти такі алгоритми, які б не викликали візуальних артефактів у контейнері [20].

Зорова система людини (ЗСЛ) має ряд властивостей, які впливають на помітність прихованих даних у зображеннях та можуть бути використані при реалізації стеганографічних методів. Ці властивості можна розділити на дві категорії: низькорівневі (фізіологічні) та високорівневі (психофізіологічні) [21].

До низькорівневих властивостей ЗСЛ належать:

1) Чутливість до зміни яскравості (контрасту) зображення. Встановлено, що для середніх діапазонів значень яскравості контраст залишається приблизно постійним, в той час як для малого та великого значення яскравості, значення порогу непомітності (ΔI) збільшується. Завдяки дослідженням стало відомо, що при маленькому значенні яскравості поріг непомітності зменшується, тобто ЗСЛ є більш чутливою до шуму в даному діапазоні. З’ясовано, що $\Delta I \approx (0.01 \div 0.03) \cdot I$ для середніх значень яскравості.

2) Частотна чутливість. ЗСЛ є більш сприйнятливою до низькочастотного (НЧ) шуму порівняно з високочастотним (ВЧ) шумом. Це пов'язано з нерівномірністю амплітудно-частотної характеристики ЗСЛ. Адитивний шум є більш помітним на НЧ ділянках зображення порівняно з ВЧ ділянками (ефект маскування).

3) Ефект маскування. Коли на око одночасно діють два сигнали зі схожими характеристиками, вони збуджують одні й ті самі підканали в ЗСЛ, що призводить до збільшення порога виявлення зорового сигналу. Найсильніше

ефект маскування проявляється, коли обидва сигнали мають однакову просторову орієнтацію та розташовані в одному місці.

Для того, щоб краще розуміти залежність мінімального контрасту $\frac{\Delta I}{I}$ від яскравості, можна подивитися на рис. 2.1.

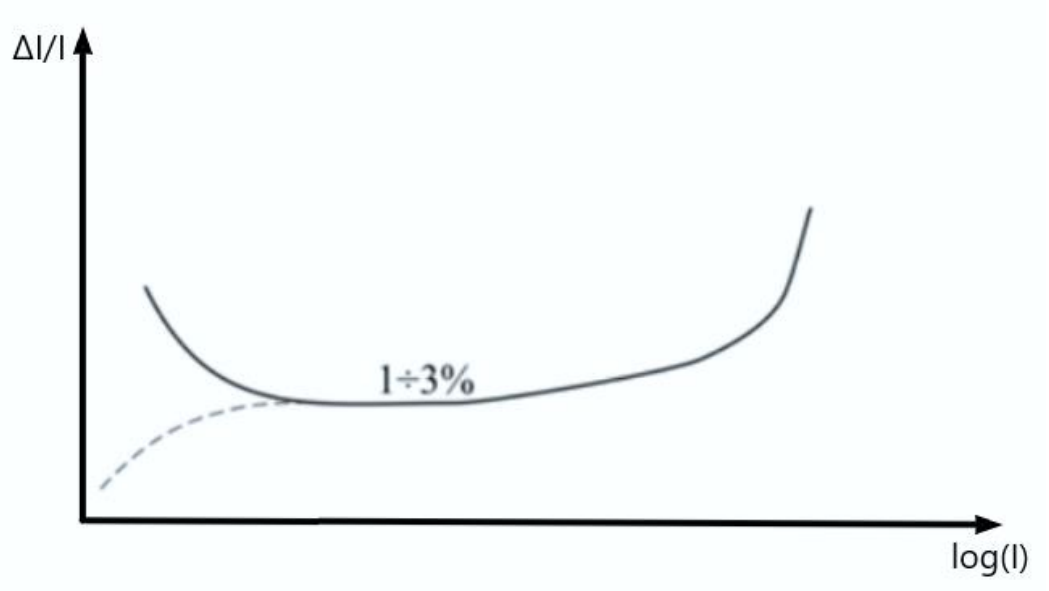

Рисунок 2.1 - Вплив чутливості до зміни контрасту на поріг непомітності ΔI [21]

Перед тим, як перейти до високорівневих властивостей зорової системи, слід зазначити, що вони мають декілька відмінностей від низькорівневих, адже вони проявляються тільки після обробки первинної інформації мозком. Мозок видає команди для адаптації зорової системи під конкретне зображення.

До високорівневих властивостей ЗСЛ відносяться:

1) Чутливість до контрасту. Висококонтрастні області зображення та різниця в яскравості привертають більше уваги.

2) Чутливість до розміру. Великі області зображення є більш помітними, ніж менші за розміром, але існує поріг насиченості, після досягнення якого подальші збільшення розміру не є важливим.

3) Чутливість до форми. Довгі та тонкі об'єкти привертають більше уваги, ніж закруглені та однорідні.

4) Чутливість до кольорів. Людська зорова система має неоднакову чутливість до різних кольорів. Деякі кольори є більш помітними для сприйняття,

ніж інші, особливо якщо вони контрастують з кольорами фону або елементів переднього плану. Зокрема, червоний колір вважається найбільш привертаючим увагу, тоді як синій колір є відносно менш помітним для людського ока.

5) Чутливість до місця розміщення. Люди схильні більше вдивлятися у центр зображення та уважніше розглядати предмети переднього плану.

Розуміння цих особливостей сприйняття зображень людською зоровою системою дозволяє розробляти більш ефективні стеганографічні методи, які можуть приховувати дані в зображеннях таким чином, щоб вони були практично непомітними для людського ока.

2.2 Система стеганографії зображень

Структурна схема загальної системи стеганографії зображень наведено на рис. 2.2.

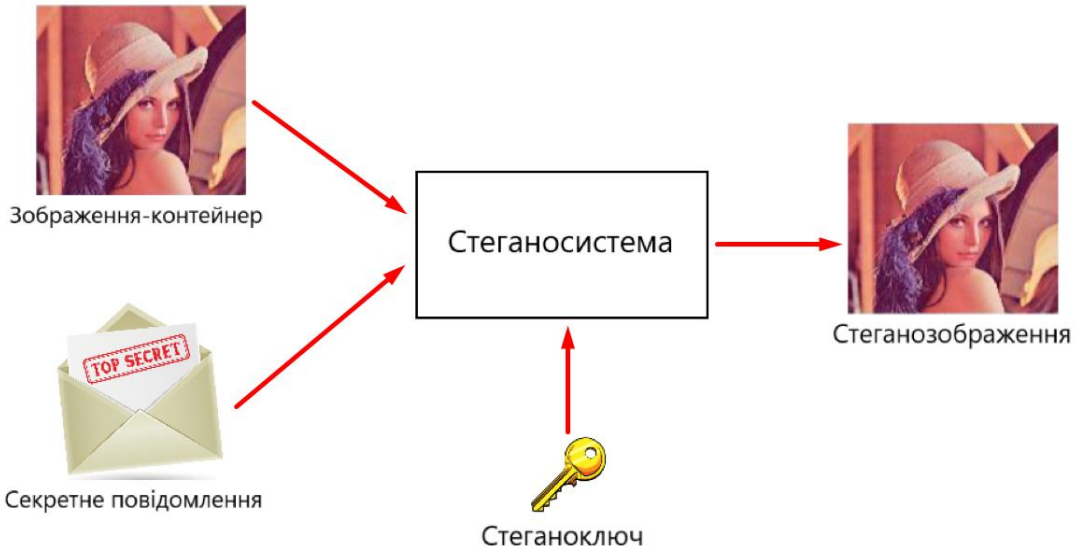

Рисунок 2.2 - Загальна форма стеганографії зображень

Повідомлення вбудовується в цифрове зображення (зображення-контейнер) за допомогою алгоритму вбудовування з використанням секретного ключа. Отримане стеганозображення передається каналом зв'язку до приймача, де воно обробляється алгоритмом вилучення з використанням того ж самого ключа. Під час передачі стеганозображення, його можуть спостерігати неавторизовані користувачі, які лише помітять передачу зображення, не дізнавшись про існування прихованого повідомлення [22].

A. Методи стеганографії зображень

Метод заміни в просторовій області: В цій техніці замінюються лише молодші значущі біти базового зображення без зміни всього базового зображення. Це найпростіший метод приховування даних, але він дуже слабкий проти простих атак, таких як стиснення, перетворення тощо.

Метод перетворення області: Різноманітними техніками перетворення області є дискретне косинусне перетворення (DCT), дискретне вейвлет-перетворення (DWT) та швидке перетворення Фур'є (FFT), які використовуються для приховування інформації в коефіцієнти перетворення базового зображення, що робить їх набагато стійкішими до таких атак, як стиснення, фільтрація тощо.

Метод розширеного спектру: Повідомлення розподіляється по широкій смузі частот, ширшій за мінімально необхідну для передачі інформації. Співвідношення сигнал/шум у кожній смузі частот є малим. Отже, не руйнуючи вміщуюче зображення, дуже важко повністю видалити повідомлення.

Статистичний метод: Базове зображення розбивається на блоки, а біти повідомлення приховуються в кожному блоці. Інформація кодується шляхом зміни різних числових властивостей вміщуючого зображення. Блоки базового зображення залишаються незмінними, якщо блок повідомлення дорівнює нулю.

Метод спотворення: Інформація зберігається шляхом спотворення сигналу. Кодер додає послідовність змін у вміщуюче зображення, а декодер перевіряє різні відмінності між оригінальним і спотвореним зображенням, щоб відновити секретне повідомлення.

B. Стеганоаналіз

Стеганоаналіз - це наука про виявлення прихованої інформації. Основна мета стеганоаналізу - зламати стеганографію та виявити стеганозображення. Майже всі алгоритми стеганоаналізу покладаються на стеганографічні алгоритми, що вводять статистичні відмінності між зображенням-контейнером та стеганозображенням. Стеганоаналіз має справу з трьома важливими категоріями, а саме:

1) Візуальні атаки: У цих типах атак за допомогою комп'ютера або неозброєним оком виявляють наявність прихованої інформації, яка допомагає розділити зображення на бітові площини для подальшого аналізу.
2) Статистичні атаки: Ці типи атак є більш потужними та успішними, оскільки вони виявляють найменші зміни у статистичній поведінці зображення. Статистичні атаки можна поділити на пасивні атаки та активні атаки. Пасивні атаки пов'язані з виявленням наявності або відсутності прихованого повідомлення або використаного алгоритму вбудовування тощо. В той час як активні атаки використовуються для дослідження довжини вбудованого повідомлення, розташування прихованого повідомлення або секретного ключа, що був використаний для вбудовування.
3) Структурні атаки: Формат даних файлів змінюється, коли в них вбудовуються дані, які потрібно приховати; виявлення цих характерних структурних змін може допомогти нам виявити присутність зображення.

2.3 Огляд стеганографічних методів приховування даних у зображеннях

2.3.1 Методи приховування даних у просторовій області зображень

Просторові стеганографічні методи базуються на заміні найменш значущих бітів зображення бітами секретного повідомлення [21, 23-31]. Це досягається шляхом використання колірного представлення пікселів зображення за допомогою моделі RGB, де кожен піксель описується комбінацією значень червоного, зеленого та синього кольорів. Перевага просторових стеганографічних методів полягає у їх простоті реалізації, оскільки вони не потребують складних обчислень або перетворень над даними зображення. Серед найбільш використовуваних методів вбудовування стеганоданих у просторову область зображення, можна зазначити такі:

1) Приховування даних за допомогою LSB (найменш значущого біта): В літературі запропоновано різні методи приховування даних. Один з найпоширеніших методів базується на маніпулюванні найменш значущим бітом (LSB). Даний метод використовує першу низькорівневу властивість зорової системи людини (ЗСЛ), тобто слабку чутливість до незначної зміни яскравості [21, 29, 32-35].

Метод LSB вбудовує секретний біт даних у найменший значущий біт пікселя вихідного зображення. Зміна буде відбуватися у найменшому значущому біті, а інші біти залишаються незмінними. LSB методи зазвичай досягають високої пропускної здатності, але вони вразливі до незначних маніпуляцій із зображенням, таких як обрізання та стиснення. Перевагою цього методу є зрозумілість, швидка реалізація та висока пропускна здатність. Недоліком LSB є відсутність секретних даних та вразливість до стеганоаналізу [18].

Цей метод найкраще працює, коли файл довший за файл повідомлення та якщо зображення має відтінки сірого. При застосуванні методу LSB до кожного байту 24-бітового зображення, три біти можуть бути закодовані в кожен піксель (рис. 2.3).

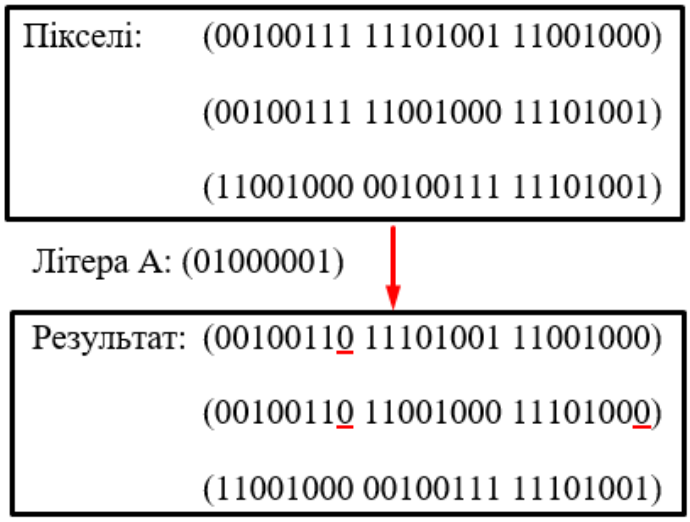

Рисунок 2.3 – Приховування літери А

Як бачимо три підкреслені біти - це єдині три біти, які було змінено. Вставка LSB в середньому вимагає зміни лише половини бітів у зображенні. Оскільки для приховування 8-бітної літери А потрібно лише вісім байт, дев'ятий

байт з трьох пікселів можна використати для початку приховування наступного символу прихованого повідомлення [36-39].

2) Приховування даних за допомогою PVD (різниця значень пікселів): Метод різниці значень пікселів, запропонований Wu та Tsai [40], може успішно забезпечити як високу здатність вбудовування, так і надзвичайну непомітність для стеганозображення. Метод різниці значень пікселів сегментує вихідне зображення на блоки, що не перекриваються, з двома з'єднувальними пікселями, і змінює різницю пікселів у кожному блоці (парі) для вбудовування даних. Чим більша різниця у вихідних значеннях пікселів, тим більша модифікація. Якщо різниця пікселів має мале значення, то це означає, що блок є гладкою областю, тоді як більше значення вказує на те, що блок є крайовою/шумовою областю. Основна ідея методу PVD полягає в тому, щоб знайти більше крайових областей, щоб приховати більше секретних даних, оскільки людський зір краще розрізняє крайові області, ніж гладкі. На етапі вилучення необхідна оригінальна таблиця діапазону. Вона використовується для розділення стеганозображення тим самим методом, що і для вихідного зображення. Це дозволяє відновити вбудовані дані зі стеганографічного зображення. Таким чином, метод PVD дозволяє приховувати дані безпосередньо в цифрових зображеннях із високою пропускною здатністю та непомітністю для людського ока.

На основі методу PVD, також були запропоновані різні підходи. Серед них Chang E. та Potdar V. пропонують новий метод, що використовує тристороннє розрізнення значень пікселів, який є кращим за оригінальний метод PVD з точки зору можливості вбудовування та PSNR [41].

3) Приховування даних за допомогою GLM (модифікація рівнів сірого): У 2004 році Potdar V. та Chang E. запропонували метод GLM, який використовується для вбудовування даних шляхом модифікації рівнів сірого кольору зображення пікселів зображення [21]. Стеганографія модифікації рівнів сірого - це метод відображення даних (не вбудовування або приховування їх) шляхом модифікації рівнів сірого пікселів зображення. Техніка GLM використовує концепцію непарних і парних чисел для відображення даних на

зображенні. Вона являє собою один-до-одного відображення між двійковими даними та вибраними пікселями на зображенні. Із заданого зображення виділяється набір пікселів на основі математичної функції. Рівень сірого цих пікселів досліджуються і порівнюються з бітовим потоком, який має бути відображений на зображенні.

4) Приховування даних за допомогою псевдовипадкової перестановки: Метод псевдовипадкової перестановки є подальшим розвитком методу LSB і використовує ті ж низькорівневі особливості, що і LSB. Метод заснований на використанні простих шифрів перестановки (старих спартанських шифрів). В основі даного методу лежить генератор псевдовипадкових чисел, який генерує послідовність індексів та зберігає секретне повідомлення в пікселі з відповідним індексом. Перевагами даного методу є висока пропускна здатність та, при підвищенні розміру блоку розбитих інформаційних даних, висока стійкість до детектування. Однак підвищення значення n веде й до різкого підвищення складності реалізації. Крім того, ще одним недоліком є наявність випадкової відстані між вбудованими бітами, що призводить до низької стійкості до стеганоаналізу [21, 27, 29].

5) Приховування даних за допомогою псевдовипадкового інтервалу: Метод псевдовипадкового інтервалу також є подальшим розвитком методу LSB, який використовує ту ж саму низькорівневу властивість зорової системи людини. В основі методу лежить використання набору псевдовипадкових чисел (що задаються секретним ключем), які визначають псевдовипадковий інтервал між окремими пікселями зображення, в які методом LSB вбудовуються інформаційні біти. Перевагами даного методу є висока швидкість перетворення та стійкість до детектування та вилучення повідомлення. Недоліком є те, що всі методи на основі модифікації LSB уразливі до руйнування повідомлення за допомогою обнулення або псевдовипадкового заповнення всіх LSB контейнерів. Такі системи є аналогами крихкої стеганосистеми [21, 27, 29].

2.3.2 Методи приховування даних у частотній області зображень

Як вже зазначалося раніше, стеганографічні методи заміни є нестійкими до будь-яких спотворень, а щодо застосування операцій стиснення з втратами, то даний процес взагалі призводить до повного знищення вього секретного повідомлення, схованого методом LSB у зображенні. Більш стійкими до всіляких спотворень, в тому ж числі й до атак стиснення, є методи, які для приховування інформації використовують не тимчасову область, а частотну, що робить їх стійкими до різних операцій обробки зображень, таких як стиснення, покращення тощо [21, 23, 25, 26, 27, 29, 31]. Є декілька способів представлення зображення в частотній області: дискретне косинусне перетворення (DCT), швидке перетворення Фур'є (FFT) і вейвлет-перетворення перетворення (DWT). Основний підхід до приховування інформації за допомогою DCT, FFT або вейвлет-перетворення полягає в перетворенні вихідного зображення, налаштуванні коефіцієнтів, а потім зворотному перетворенні. Якщо вибір коефіцієнтів вдалий, а розмір змін допустимий, тоді результат буде досить близьким до оригіналу.

1) Приховування даних за допомогою алгоритму Коха-Жао: Одним із ключових методів приховування інформації в частотній області цифрового зображення є алгоритм Коха-Жао. Цей підхід базується на використанні дискретного косинусного перетворення (ДКП) для перетворення зображення у частотну область [21, 23, 25, 34]. Спочатку вхідне зображення розбивається на блоки розміром 8 на 8 пікселів. Для кожного блоку виконується ДКП, в результаті чого отримується матриця коефіцієнтів, які представляють амплітуди відповідних частотних складових зображення в цьому блоці.

Далі відбувається процес вбудовування прихованих даних шляхом модифікації деяких обраних коефіцієнтів ДКП. Принцип полягає в тому, що для вбудовування біта "0" обраний коефіцієнт збільшується таким чином, щоб його абсолютна різниця з початковим значенням була більшою за заданий поріг. Для вбудовування біта "1" коефіцієнт зменшується, щоб абсолютна різниця була меншою за поріг. Цей поріг є параметром алгоритму.

Після модифікації коефіцієнтів ДКП для кожного блоку виконується зворотне дискретне косинусне перетворення (ЗДКП), щоб отримати модифіковані значення пікселів. Результуюче зображення із вбудованими прихованими даними називається стеганозображенням.

Фактично, алгоритм Коха-Жао тісно пов'язаний з процесом стиснення зображень у форматі JPEG, оскільки воно також використовує ДКП. Це дозволяє вбудовувати приховані дані безпосередньо в процес JPEG-компресії (рис. 2.4).

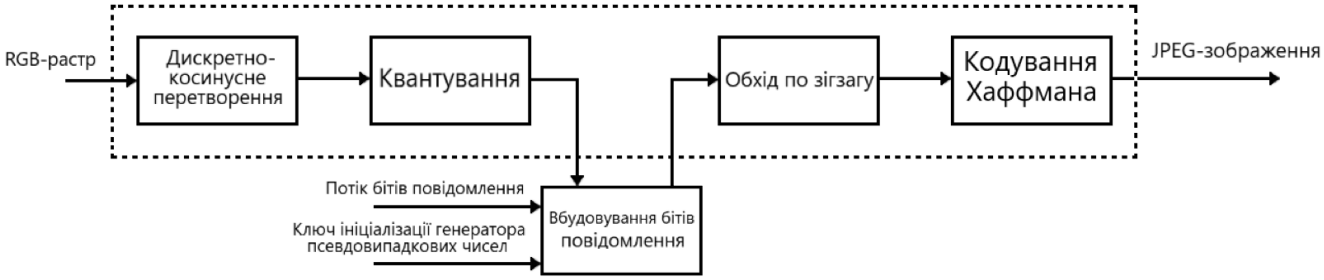

Рисунок 2.4 - Приховування даних у процесі JPEG-стиснення зображення

Одним з основних недоліків методу Коха-Жао є його обчислювальна складність, пов'язана з необхідністю виконувати ДКП та ЗДКП для кожного блоку зображення. Це може бути ресурсномістким процесом, особливо для великих зображень.

Перевагами даного методу є відносно висока пропускна здатність та стійкість до деяких типів атак та спотворень зображення, оскільки приховані дані вбудовуються в частотну область, яка є менш чутливою до певних видів шумів та артефактів.

2) Приховування даних за допомогою методу Бенгама-Мемона-Ео-Юнга: Метод Бенгама-Мемона-Ео-Юнга намагається оптимізувати оригінальний підхід Коха-Жао, зменшуючи спотворення, які вноситься у вихідне зображення [21, 23, 34]. Це досягається шляхом ретельного вибору блоків зображення для вбудовування інформації. Обираються ті блоки, модифікація яких призведе до найменших візуальних змін у зображенні. Крім того, замість одного, для вбудовування використовується два коефіцієнти ДКП, що дозволяє зменшити внесені спотворення. При виборі блоків керуються двома основними критеріями: відсутність різких переходів яскравості та достатня монотонність блоку.

Перевагами даного методу є покращена непомітність стеганозображення в порівнянні з базовим алгоритмом Коха-Жао, оскільки блоки для вбудовування обираються ретельно за критеріями відсутності різких переходів яскравості та достатньої монотонності, що допомагає зробити вбудовування менш помітним для людського ока. Крім того, використання двох коефіцієнтів ДКП для вбудовування замість одного дозволяє додатково зменшити візуальні спотворення порівняно з оригінальним підходом.

Недоліками даного методу є підвищена обчислювальна складність, оскільки потрібно ретельно аналізувати блоки зображення для вибору оптимальних, придатних для вбудовування згідно встановлених критеріїв. Також потенційно нижча пропускна здатність в порівнянні з простим методом Коха-Жао через обмеження на вибір блоків, які відповідають критеріям без різких переходів та достатньої монотонності.

3) Приховування даних за допомогою методу Ху-Ву: Метод Ху-Ву, був розроблений спеціально для вбудовування цифрових водяних знаків (ЦВЗ) у зображення. У цьому підході використовується модифікація коефіцієнтів ДКП блоків вихідного зображення (контейнера) для приховування бітів ЦВЗ [13, 21, 23, 34]. Перед вбудовуванням зображення ЦВЗ, яке зазвичай є чорно-білим, нормалізується діленням на 255. Після вилучення ЦВЗ з контейнера воно відновлюється множенням на те саме значення 255.

Перевагами даного методу є його спеціалізація на вбудовуванні цифрових водяних знаків (ЦВЗ) у зображення, нормалізація ЦВЗ перед вбудовуванням, що може покращити його непомітність, а також можливість відновити оригінальний ЦВЗ після його вилучення зі стеганозображення.

Недоліками цього підходу є його обмеженість лише задачею вбудовування ЦВЗ, неможливість використання для приховування загальних даних. Також метод все ще може вносити певні спотворення у вихідне зображення, хоча й менші, ніж базовий алгоритм Коха-Жао. Крім того, використання ДКП передбачає підвищену обчислювальну складність реалізації.

3. РОЗРОБКА СТЕГАНОГРАФІЧНИХ МЕТОДІВ ПРИХОВУВАННЯ ІНФОРМАЦІЇ З ЗОБРАЖЕННЯХ

3.1 Вибір інструментів розробки для реалізації стеганографічних методів

Вибір відповідних інструментів розробки, зокрема мови програмування та середовища розробки, є критичним етапом при створенні будь-якого програмного забезпечення, включаючи реалізацію стеганографічних методів приховування інформації у зображеннях. Правильний вибір може значно вплинути на ефективність, продуктивність, зручність використання та загальну якість кінцевого продукту. У цьому розділі я розглянув деякі аспекти, які впливають на вибір мови програмування, та обґрунтував використання мови програмування Java для реалізації стеганографічних алгоритмів.

Перш за все, варто звернути увагу на поточні тенденції популярності мов програмування згідно з даними статистичного опитування. На рис. 3.1 представлено рейтинг найпопулярніших мов станом за 2024 рік, який був сформований на основі анкет від IT-спеціалістів з України.

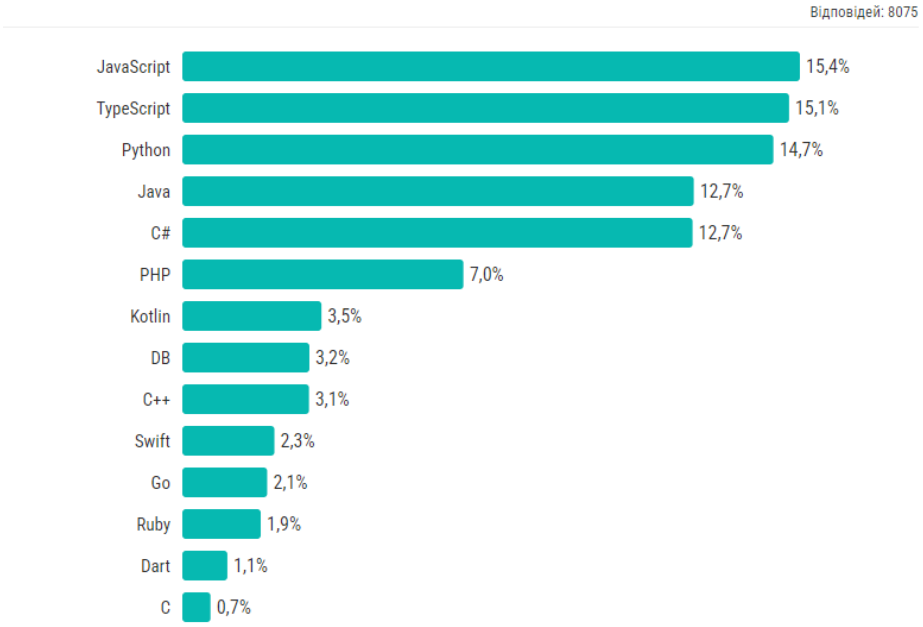

Рисунок 3.1 – Найпопулярніші мови програмування за 2024 рік [43]

Згідно з цими даними, найбільш поширеними мовами є JavaScript (15,4%), TypeScript (15,1%), Python (14,7%) та Java (12,7%) [43]. Незважаючи на те, що JavaScript, TypeScript та Python посідають вищі позиції, мова програмування

Java, навіть із плином часу, залишається одним із лідерів і має низку переваг, які роблять її привабливим вибором для реалізації стеганографічних методів.

Основною перевагою Java є її об'єктно-орієнтована природа, яка забезпечує зручність та гнучкість у розробці програмного забезпечення. Об'єктно-орієнтоване програмування (ООП) дозволяє структурувати код у вигляді класів та об'єктів, що полегшує організацію, повторне використання та підтримку коду. Ця парадигма особливо корисна при розробці складних систем, таких як стеганографічні методи, де необхідно працювати з різними типами даних (зображеннями, текстовими повідомленнями тощо) та реалізовувати різноманітні алгоритми.

Крім того, Java має потужну бібліотеку для роботи з зображеннями, відому як Java Image I/O (javax.imageio). Ця бібліотека забезпечує зручний інтерфейс для завантаження, обробки та збереження різноманітних форматів зображень, включаючи JPEG, PNG, BMP та інші. Вона також надає функціональність для маніпуляцій з пікселями зображень, що є надзвичайно важливим для реалізації стеганографічних методів, де необхідно модифікувати значення пікселів для вбудовування прихованих даних.

Java також підтримує багато алгоритмів криптографії та стиснення через Java Cryptography Architecture (JCA) та Java Compression API. Ці можливості можуть бути корисними при реалізації стеганографічних методів, оскільки часто потрібно шифрувати приховані дані перед їх вбудовуванням у зображення, а також застосовувати алгоритми стиснення для ефективного використання обмеженого місця для прихованих даних.

Ще слід зазначити таку особливість Java як її кросплатформеність. Тобто програми, які написані на Java, можуть працювати на різних операційних системах, таких як Windows, macOS, Linux та інших, без необхідності додаткової компіляції. Це забезпечується завдяки використанню Java Virtual Machine (JVM), яка виконує скомпільований байт-код Java на різних платформах. Ця особливість може бути корисною для стеганографічних систем, які потребують взаємодії між різними платформами та операційними системами.

Java має широкий вибір інструментів для налагодження, профілювання та тестування, що полегшує процес розробки та забезпечення якості програмного забезпечення. Ці інструменти включають інтегровані середовища розробки (IDE), такі як IntelliJ IDEA, Eclipse та NetBeans, а також інструменти для юніт-тестування, такі як JUnit. Для написання коду для реалізації стеганографічних методів, я використовував IntelliJ IDEA (рис. 3.2).

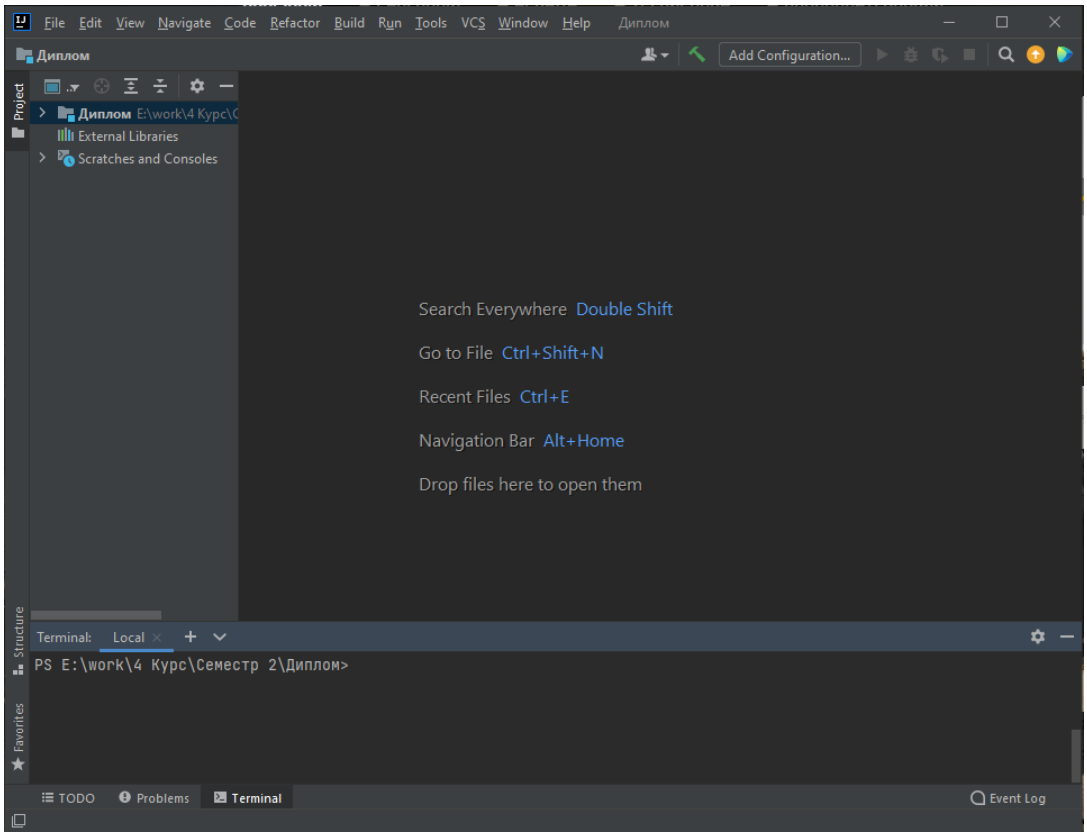

Рисунок 3.2 – Вікно інтегрованого середовища розробки IntelliJ IDEA

IntelliJ IDEA є потужним та зручним IDE, що надає багатий функціонал для написання, налагодження та рефакторингу коду на Java. Вона пропонує інтелектуальне автодоповнення коду, інтеграцію з популярними системами контролю версій, такими як Git, та підтримку різноманітних фреймворків і бібліотек. Крім того, IntelliJ IDEA має вбудовані інструменти для профілювання та оптимізації продуктивності додатків, а також можливості для юніт-тестування за допомогою таких фреймворків, як JUnit. Крім того, Java має активну спільноту розробників, що забезпечує постійну підтримку, оновлення та доступність численних бібліотек і фреймворків для різноманітних завдань.

З огляду на всі вищезазначені переваги, мовою програмування для реалізації стеганографічних методів приховування інформації у зображеннях було обрано Java. Ця мова забезпечує гнучкість, зручність використання, підтримку бібліотек для роботи з зображеннями, криптографією та стисненням, а також кросплатформеність та наявність інструментів для розробки і тестування.

3.2 Розробка методу LSB для приховування даних у зображеннях.

Метод заміни найменш значущих бітів (Least Significant Bit, LSB) є одним з найпоширеніших підходів до приховування інформації у зображеннях. Як вже зазначалося в попередньому розділі, його принцип полягає в заміні найменш значущих бітів пікселів зображення-контейнера на біти прихованого повідомлення. Ця заміна здійснюється таким чином, щоб різниця між оригінальним та модифікованим зображеннями була непомітна для людського ока.

На рис. 3.3 проілюстровано процес перетворення вихідного тексту на бітову послідовність для подальшого вбудовування у зображення. Спочатку текст перетворюється на рядок символів (string), де кожен символ представлений своїм числовим кодом (наприклад, у кодуванні ASCII, “H” = 72, “e” = 101, і так далі). Після цього, ці числові коди перетворюються на послідовність байтів (1 byte/char), де кожен байт складається з 8 бітів (8 bits/byte).

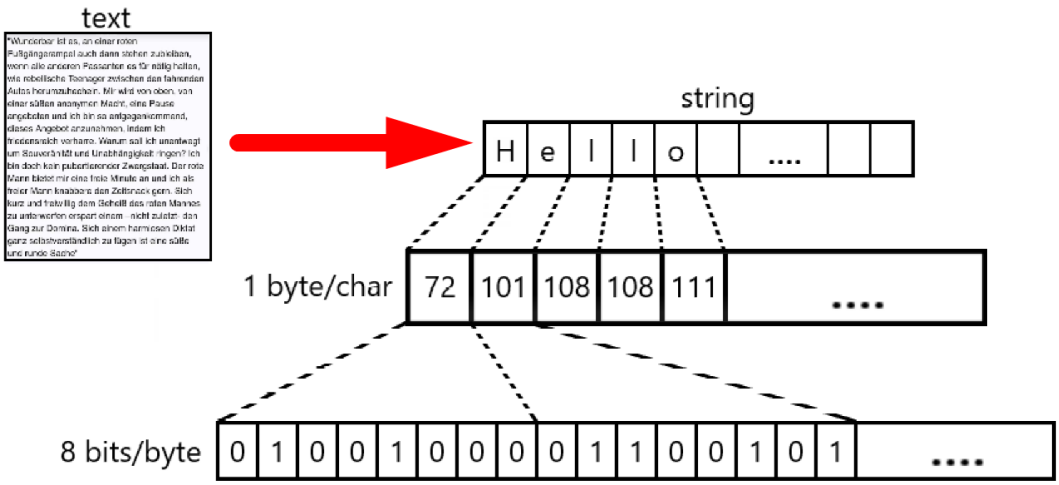

Рисунок 3.3 – Принцип приховування повідомлення

Як відомо, у форматі BMP зображення зберігається у вигляді матриці значень відтінків кольору для кожного пікселя. Оскільки кожен компонент кольорового простору RGB (червоний, зелений і синій канали) зберігається в одному байті, він може набувати значень від 0 до 255. Це відповідає глибині кольору в 24 біти.

Ключовою особливістю людського зору є його обмежена здатність розрізняти незначні відхилення у відтінках кольору. Тому у випадку 24-бітних кольорових зображень, зміна лише одного найменш значущого біта в кожному з трьох кольорових каналів (червоний, зелений, синій) призводить до зміни інтенсивності кольору на менше ніж 1%, що є непомітним для людського ока. Саме цю особливість використовує метод LSB для вбудовування прихованих даних.

Розглянемо детальніше процес вбудовування прихованого повідомлення за допомогою LSB (рис. 3.4). Нехай ми маємо 24-бітне зображення у градаціях сірого, де кожен піксель кодується трьома байтами, що представляють значення каналів RGB. Змінюючи найменш значущий біт у кожному з цих байтів, ми можемо вбудувати приховане повідомлення, не спричиняючи помітних змін для людського ока. Більше того, такі незначні зміни можуть навіть не відобразитися на низькоякісних пристроях виведення зображень.

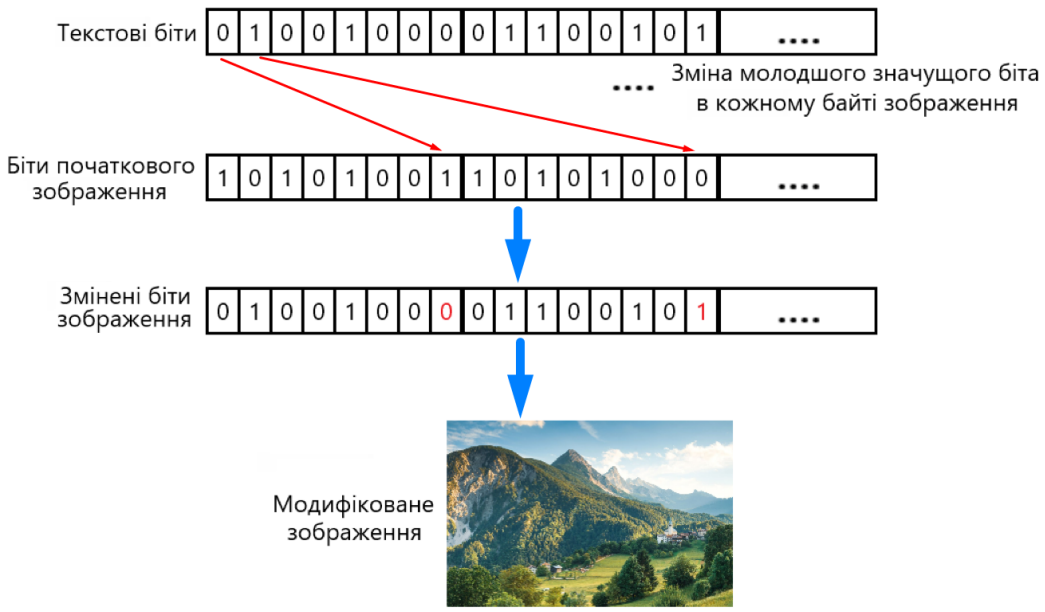

Рисунок 3.4 – Приховування інформації у зображенні

Щодо пропускної здатності методу LSB, якщо не враховувати службову інформацію на початку файлу зображення (яка зазвичай є незначною порівняно з розміром самого зображення), то максимальний розмір прихованого повідомлення може становити 1/8 від розміру контейнера у випадку використання одного найменш значущого біта кожного байта кольорової матриці пікселів. Якщо ж використовувати два найменш значущі біти, то пропускна здатність зросте до 1/4 від розміру контейнера.

Наприклад, якщо зображення має розмір 1280 на 720 пікселів, і кожен піксель кодується 24 бітами (по 8 біт на канал RGB), то загальний розмір зображення становитиме приблизно 2,76 МБ (1280 * 720 * 3 байти). У випадку використання одного найменш значущого біта в кожному байті матриці кольорів, максимальний розмір прихованого повідомлення може становити до 345 КБ (2,76 МБ / 8 = 0,345 МБ). Якщо ж використовувати два найменш значущі біти в кожному байті, пропускна здатність зросте до 690 КБ (2,76 МБ / 4 = 0,69 МБ), що дозволить приховати більший обсяг даних у зображенні.

Знаючи всі ці особливості та принципи вбудовування інформації у зображення методом LSB, можна приступити до написання коду на мові програмування Java для алгоритму заміни найменш значущих бітів.

Для реалізації алгоритму LSB було створено два класи:

1) Клас LSB_encode (Додаток А)

У даному класі виконується процес вбудовування таємного повідомлення у зображення-контейнер за допомогою методу LSB. Цей клас приймає два файли: повідомлення, яке потрібно зашифрувати (MESSAGEFILE), та зображення-контейнер (COVERIMAGEFILE), в яке буде сховане повідомлення.

Спочатку відбувається зчитування повідомлення з файлу та перетворення його у бінарний вигляд, де кожен символ представлений як послідовність з 8 бітів. Далі зчитується зображення-контейнер у форматі BMP. Перед вбудовуванням повідомлення у зображення, визначається його довжина у бітах, яка також записується у перші 32 біти стеганозображення для подальшого відновлення. Потім, у циклі, пікселі зображення проходяться один за одним, і у

найменш значущий біт синього каналу кожного пікселя записуються біти повідомлення та його довжини, заміщуючи початкове значення цього біта. Таким чином, повідомлення "розпорошується" по всьому зображенню, змінюючи його колір незначним чином, що непомітно для людського ока.

В результаті ми отримуємо стеганозображення у вигляді BMP-файлу "stego_image.bmp" (змінна STEGIMAGEFILE).

На лістингу 3.1 зображена функція hideTheMessage, яка реалізує алгоритм вбудовування секретного повідомлення в зображення методом LSB.

Лістинг 3.1 - Функція алгоритму вбудовування методом LSB

```
public static void hideTheMessage(int[] bits, BufferedImage theImage) throws
Exception {
        File f = new File(STEGIMAGEFILE);
        int bit_l = bits.length;
        int[] bl_msg = new int[32];
        // Вивід байтової довшини повідомлення
        // System.out.println("byte length " + bits.length / 8);
        String bl_s = String.format("%32s", Integer.toBinaryString(bits.length /
8)).replace(' ', '0');
        for (int i1 = 0; i1 < 32; i1++) {
            bl_msg[i1] = bl_s.charAt(i1) - '0';
        }
        int j = 0;
        int currentBitEntry = 32;

        for (int x = 0; x < theImage.getWidth(); x++) {
            for (int y = 0; y < theImage.getHeight(); y++) {
                int currentPixel = theImage.getRGB(x, y);
                int red = (currentPixel >> 16) & 255;
                int green = (currentPixel >> 8) & 255;
                int blue = currentPixel & 255;
                if (x == 0 && y < 32) {
                    blue = (blue & ~1) | bl_msg[y];
                } else if (currentBitEntry < bits.length + 32) {
                    blue = (blue & ~1) | bits[j++];
                    currentBitEntry++;
                }
                int rgb = (255 << 24) | (red << 16) | (green << 8) | blue;
                theImage.setRGB(x, y, rgb);
            }
        }
        ImageIO.write(theImage, "bmp", f);
    }
```

2) Клас LSB_decode (Додаток Б)

Клас LSB_decode реалізує процес витягування таємного повідомлення із стеганозображення за допомогою методу LSB. Цей клас приймає один файл – стагенозображення (STEGIMAGEFILE), в якому міститься наше зашифроване повідомлення.

Спочатку відбувається зчитування стеганозображення у форматі BMP у буфер зображення. Потім, у циклі, пікселі зображення проходяться один за одним, і з найменш значущого біта синього каналу витягуються біти зашифрованого повідомлення та його довжини. На початку зчитуються перші 32 біти, які містять інформацію про довжину повідомлення у бітах. Далі, знаючи цю довжину, з наступних пікселів витягуються біти самого повідомлення. Вся бінарна послідовність зберігається у рядку “b_msg”. Після того, як всі біти були витягнуті, бінарна послідовність перетворюється на текст за допомогою стандартної кодової таблиці UTF-8.

Отриманий текст виводиться у консоль та записується у новий текстовий файл "message_dec.txt" (змінна DECODEDMESSAGEFILE).

На лістингу 3.2 зображена функція DecodeTheMessage, яка реалізує алгоритм витягу таємного повідомлення із стеганозображення методом LSB.

Лістинг 3.2 - Функція алгоритму вилучення методом LSB

```
public static void DecodeTheMessage(BufferedImage yImage) throws Exception {
        int currentBitEntry = 0;
        StringBuilder bx_msg = new StringBuilder();
        for (int x = 0; x < yImage.getWidth(); x++) {
            for (int y = 0; y < yImage.getHeight(); y++) {
                int currentPixel = yImage.getRGB(x, y);
                int blue = currentPixel & 255;
                if (x == 0 && y < 32) {
                    bx_msg.append(blue & 1);
                    if (y == 31) {
                        len = Integer.parseInt(bx_msg.toString(), 2);
                    }
                } else if (currentBitEntry < len * 8) {
                    b_msg += (blue & 1);
                    currentBitEntry++;
                }
            }
        }
        // Вивід секретного повідомлення в бітовому представленні
        //System.out.println("bin value of msg hided in img is " + b_msg);
    }
```

Для демонстрації результатів виконання розробленого алгоритму заміни найменш значущих бітів було обрано вихідне зображення формату BMP, розмірами 1200 на 783 пікселів, яке слугувало незаповненим стеганоконтейнером (рис. 3.5) та секретне повідомлення, розміром 88000 символів (88000 байтів), яке буде вбудовуватися в наше зображення-контейнер.

Для нашого зображення максимальний розмір прихованого повідомлення може становити до 352 КБ.

Рисунок 3.5 – Вихідне зображення-контейнер

На рис. 3.6 представлено те ж саме зображення, але вже з вбудованим повідомленням за допомогою методу LSB.

Рисунок 3.6 – Заповнений зображення-контейнер

При порівнянні рисунків 3.5 і 3.6 та аналізі максимальної пропускної здатності передачі повідомлення для цього типу зображення, стає очевидним, що при довжині вбудованого повідомлення в 88000 байтів (приблизно 25% від

максимального розміру зображення-контейнера) візуально неможливо відрізнити порожній контейнер від контейнера, заповненого за допомогою алгоритму заміни LSB.

3.3 Розробка методу Коха-Жао для приховування даних у зображеннях.

Один із найпоширеніших на сьогодні методів приховування таємної інформації у частотній області зображення грунтується на відносній заміні величин коефіцієнтів ДКП, який свого часу описали Кох і Жао.

Стеганографічний метод Коха-Жао базується на двовимірному дискретному косинусному перетворенні (ДКП). Алгоритм вбудовування повідомлення складається з наступних кроків:

1) Оригінальне зображення розбивається на блоки розміром 8 на 8 пікселів.

2) До кожного блоку застосовується ДКП, результатом є матриця коефіцієнтів Di (i = 1, ... , N; N – це кількість блоків) розміром 8 на 8.

3) Вибирається послідовність блоків, у які буде виконуватись вбудовування. У кожен блок записується 1 біт інформації.

4) У кожному блоці вибираються два коефіцієнти ДКП, що знаходяться в середньо-частотному діапазоні коефіцієнтів, які симетричні відносно головної діагоналі (Di[3,4] та Di[4,3], Di[3,5] та Di[5,3], Di[4,5] та Di[5,4]).

5) Для передачі біта 0 необхідно, щоб різниця модулів пари коефіцієнтів перевищувала деяке додатне значення $M0$, для передачі біта 1 різниця повинна бути меншою за -$M0$. Таким чином, при передачі 0 ми збільшуємо модуль першого коефіцієнта і зменшуємо модуль другого. При передачі 1 зменшуємо модуль першого коефіцієнта і збільшуємо модуль другого.

6) Переходимо по кожному блоку і виконуємо кроки 4 і 5.

7) Для кожного блоку виконуємо зворотне ДКП.

Вибір середньочастотних коефіцієнтів ДКП обумовлений необхідністю мінімізації впливу вбудовування на візуальні властивості модифікованого зображення. Вибір високочастотних або низькочастотних коефіцієнтів призводить до появи ефектів, які будуть помітні візуально.

При екстракції повідомлення вважається, що пари змінних коефіцієнтів ДКП відомі. Алгоритм вилучення збігається з алгоритмом вбудовування на перших чотирьох кроках:

5) Обчислюється різниця між значеннями модулів для пар коефіцієнтів, в які було вбудовано дані.

6) Якщо різниця більша за $M0$, то був вбудований біт 0. Якщо різниця значень менша ніж -$M0$, то був вбудований біт 1.

7) Послідовно вилучаються біти, вбудовані у всі блоки.

Аналіз алгоритмів вбудовування та вилучення показує, що для успішної атаки на стеганографічний метод Коха-Жао необхідно визначити блоки, в які було вбудовано повідомлення було вбудовано повідомлення, індекси змінних коефіцієнтів ДКП та порогове значення $M0$. Для коректного вилучення повідомлення сторони, що відправляють і отримують, повинні володіти загальною секретною інформацію про параметри вбудовування.

На рис. 3.7 Зображено приклад масиву коефіцієнтів дискретного косинусного перетворення (ДКП) для 8 на 8 блоку пікселів цифрового зображення. Різні заштриховані області позначають компоненти різного діапазону величин - високі (ВЧ), середні (СЧ) та низькі (НЧ) частотні компоненти ДКП.

Цей рисунок добре ілюструє той факт, що в методі Коха-Жао вбудовування прихованих даних відбувається саме в середньочастотні компоненти ДКП. Як було зазначено, вибір середньочастотних коефіцієнтів зумовлений необхідністю мінімізувати візуально помітні спотворення зображення після вбудовування інформації. Використання високочастотних або низькочастотних компонентів могло б призвести до появи артефактів, помітних оком.

Рисунок 3.7 – Масив коефіцієнтів ДКП [21]

Припустимо, ми маємо зображення розміром 1280 на 720 пікселів, яке було розділене на блоки 8 на 8 пікселів для застосування двовимірного дискретного косинусного перетворення (ДКП). У цьому випадку загальна кількість блоків 8 на 8 становитиме 14400 (1280 * 720 / (8 * 8)). У методі Коха-Жао в кожен блок вбудовується один біт даних шляхом модифікації пари середньочастотних коефіцієнтів ДКП. Отже, максимальна пропускна здатність методу для цього зображення становитиме 14400 бітів, або приблизно 1,8 КБ (14400 / 8).

Проте варто зазначити, що фактична пропускна здатність може бути дещо меншою, оскільки метод потребує додаткових службових даних для вказання позицій вбудованих бітів, парних індексів коефіцієнтів ДКП та порогового значення M0. Хоча пропускна здатність методу Коха-Жао є нижчою, ніж у методу заміни найменш значущих бітів, його перевагою є більша стійкість до атак і менший вплив на візуальні властивості зображення завдяки використанню середньочастотних коефіцієнтів ДКП.

Маючи ґрунтовне розуміння принципу роботи та особливостей вбудовування секретної інформації в зображення методом Коха-Жао, можна розпочати написання коду на мові Java для реалізації алгоритму Коха-Жао.

Для реалізації алгоритму Коха-Жао було створено два класи:

1) Клас KochZhao _encode (Додаток В)

Клас KochZhao_encode відповідає за вбудовування секретного повідомлення у зображення методом Коха-Жао. Даний клас приймає два файли: повідомлення, яке потрібно зашифрувати (MESSAGEFILE), та зображення-контейнер (COVERIMAGEFILE), в яке буде сховане повідомлення.

Слід зазначити, що в моєму коді присутні три важливих глобальних параметри, які відповідають за налаштування шифрування для даного алгоритму, а саме: BLOCK_SIZE, THRESHOLD та CONSTANT.

Параметр BLOCK_SIZE є розміром блоку, на який розбивається зображення для виконання дискретного косинусного перетворення (ДКП). Тобто в програмній реалізації він використовується для розбиття зображення на блоки відповідної довжини і обробки кожного блоку окремо за допомогою ДКП. У стандартному JPEG-кодеку і в багатьох алгоритмах стеганографії розмір блоку ДКП становить 8 на 8 пікселів, тому параметр BLOCK_SIZE має значення 8. Це пов'язано з ефективністю обчислень і компромісом між якістю зображення та ступенем стиснення. Змінювати даний параметр не рекомендується, оскільки це стандартний розмір блоку для ДКП. Інші розміри блоків можуть призвести до складнощів з обробкою зображення і суттєво змінити якість та обчислювальну ефективність.

Параметр THRESHOLD є пороговим значенням, яке використовується для визначення, чи необхідно змінювати коефіцієнти ДКП для вбудовування біта інформації. Якщо різниця між абсолютними значеннями двох вибраних коефіцієнтів менша або рівна цьому порогу, проводяться зміни для вбудовування біта. В моїй реалізації параметр THRESHOLD має значення 25, тому якщо різниця менша або рівна 25, вбудовується біт '1', інакше вбудовується біт '0'. Слід зазначити, що даний параметр можна підлаштовувати для досягнення кращого

балансу між непомітністю стеганографії та надійністю вбудовування/декодування. Вищі значення можуть зменшити вплив на зображення, але зроблять стеганографію менш надійною.

Параметр CONSTANT є константою, яка додається до одного з коефіцієнтів ДКП для забезпечення вбудовування біта. Ця константа вибирається таким чином, щоб зміна була достатньо великою для коректного декодування, але не настільки великою, щоб суттєво погіршити якість зображення або викликати артефакти. В моїй реалізації параметр CONSTANT має значення 50. Даний параметр можна підлаштовувати для кращого компромісу між надійністю вбудовування і якістю зображення. Збільшення значення може покращити коректність декодування, але погіршити якість зображення.

Крім того, у реалізації цього алгоритму використовуються власноруч написані функції для дискретного косинусного перетворення (ДКТ) та зворотного дискретного косинусного перетворення (ЗДКТ). Ці функції обчислюють ДКТ і ЗДКТ для блоків зображення розміром 8x8, що дозволяє маніпулювати частотними компонентами зображення для вбудовування та вилучення повідомлень.

Також, для оптимізації обчислень, використовуються заздалегідь обчислені масиви cos_t та e. Масив cos_t зберігає значення косинусів, які необхідні для обчислення ДКТ, тоді як масив e містить нормалізаційні коефіцієнти, що використовуються для забезпечення правильності результатів ДКТ і ЗДКТ.

Розглянувши основні параметри, перейдемо до роботи коду даного алгоритму. Перш ніж розпочати процес вбудовування, відбувається зчитування з текстового файлу секретного повідомлення, яке потрібно зашифрувати. Після цього відбувається завантаження зображення-контейнера, в яке буде вбудовано наше повідомлення. Процес вбудовування відбувається шляхом виконання дискретного косинусного перетворення (ДКТ) на кожному блоку розміром 8 на 8 пікселів у зображенні. Далі, виконується порівняння зміни в значенні ДКТ-

коефіцієнтів між двома пікселями у верхньому правому куті блоку (3,4) та (4,3) за допомогою порогового значення THRESHOLD. Якщо зміна перевищує поріг, то біт залучається у вбудовування. При цьому, якщо біт повідомлення дорівнює 1, то виконується модифікація коефіцієнта (3,4) шляхом додавання до нього встановленого параметру CONSTANT. В іншому випадку, якщо біт дорівнює 0, модифікується коефіцієнт (4,3) шляхом додавання CONSTANT. Процес виконується для кожного блоку в зображенні, доки вся інформація не буде вбудована.

В результаті формується стеганозображення у вигляді BMP-файлу "stego_image1.bmp" (змінна STEGIMAGEFILE).

На лістингу 3.3 зображена функція insertMessage, яка реалізує алгоритм вбудовування секретного повідомлення в зображення методом Коха-Жао.

Лістинг 3.3 - Функція алгоритму вбудовування методом Коха-Жао

```
public static void insertMessage(BufferedImage theImage, String text) throws
Exception{
        File f = new File(STEGIMAGEFILE);
        int[][][] temp = new int[BLOCK_SIZE][BLOCK_SIZE][3];
        int[][] dct = new int[BLOCK_SIZE][BLOCK_SIZE];
        int l = 0;
        int column = theImage.getHeight(), row = theImage.getWidth(), type;
        if(row > column){
            type = row;
            row = column;
            column = type;
        }
        outer: for (int i = 0; i < row; i += BLOCK_SIZE) {
            for (int j = 0; j < column; j += BLOCK_SIZE) {
                if (l >= text.length() * 8) break;
                for (int x = 0; x < BLOCK_SIZE; x++) {
                    for (int y = 0; y < BLOCK_SIZE; y++) {
                        Color color = new Color(theImage.getRGB(j + x, i + y));
                        temp[x][y][0] = color.getRed();
                        temp[x][y][1] = color.getGreen();
                        temp[x][y][2] = color.getBlue();
                    }
                }

                dct = dct(dct, temp);
                int diff = Math.abs(dct[3][4]) - Math.abs(dct[4][3]);

                if (getBit(text, l)) {
                    if (diff <= THRESHOLD) {
                        dct[3][4] = (dct[3][4] >= 0) ? Math.abs(dct[4][3]) +
CONSTANT : -Math.abs(dct[4][3]) - CONSTANT;
                    }
                } else {
                    if (diff >= -THRESHOLD) {
                        dct[4][3] = (dct[4][3] >= 0) ? Math.abs(dct[3][4]) +
CONSTANT : -Math.abs(dct[3][4]) - CONSTANT;
                    }
```

```
                }

                temp = idct(dct, temp);

                for (int x = 0; x < BLOCK_SIZE; x++) {
                    for (int y = 0; y < BLOCK_SIZE; y++) {
                        Color newColor = new Color(temp[x][y][0], temp[x][y][1],
temp[x][y][2]);
                        theImage.setRGB(j + x, i + y, newColor.getRGB());
                    }
                }
                l++;
            }
            if (l >= text.length() * 8) break;
        }
        ImageIO.write(theImage, "bmp", f);
    }
```

2) Клас KochZhao _decode (Додаток Г)

Клас KochZhao_decode відповідає за вилучення секретного повідомлення із зображення-контейнера, в якому воно було вбудоване методом Коха-Жао. Даний клас приймає тільки один файл – стеганозображення (STEGIMAGEFILE), в якому міститься наше зашифроване повідомлення.

Клас завантажує стеганографічне зображення, і на його основі виконує процес вилучення повідомлення. Процес вилучення розпочинається з аналізу кожного блоку розміром 8x8 пікселів у зображенні. Для кожного блоку, виконується ДКТ, що дозволяє отримати частотні коефіцієнти блоку. Потім порівнюються значення двох специфічних ДКТ-коефіцієнтів між пікселями (3,4) та (4,3). Різниця між цими коефіцієнтами визначає значення вбудованого біта. Якщо різниця у значеннях перевищує встановлений поріг THRESHOLD, це означає, що вбудований біт дорівнює 1. Якщо ж різниця менша за THRESHOLD, то вбудований біт дорівнює 0. Далі відновлений біт додається до відновлюваного повідомлення. Після вилучення біта, виконується перевірка, чи не вичерпана вся інформація повідомлення. Коли всі біти вилучені, їх об'єднують у текстовий рядок, який може бути збережений у файлі.

Отриманий текст виводиться у консоль та записується у новий текстовий файл "message_dec1.txt" (змінна DECODEDMESSAGEFILE).

На лістингу 3.4 зображена функція readMessage, яка реалізує алгоритм витягу таємного повідомлення із стеганозображення методом Коха-Жао.

Лістинг 3.4 - Функція алгоритму вилучення методом Коха-Жао

```
public static String readMessage(BufferedImage image) {
        int[][][] temp = new int[BLOCK_SIZE][BLOCK_SIZE][3];
        int[][] dct = new int[BLOCK_SIZE][BLOCK_SIZE];
        int l = 0, p = 0, b = 0;
        int column = image.getHeight(), row = image.getWidth(), type;
        if(row > column){
            type = row;
            row = column;
            column = type;
        }
        StringBuilder out = new StringBuilder();
        outer: for (int i = 0; i < row; i += BLOCK_SIZE) {
            for (int j = 0; j < column; j += BLOCK_SIZE) {
                for (int x = 0; x < BLOCK_SIZE; x++) {
                    for (int y = 0; y < BLOCK_SIZE; y++) {
                        Color = new Color(image.getRGB(j + x, i + y));
                        temp[x][y][0] = color.getRed();
                        temp[x][y][1] = color.getGreen();
                        temp[x][y][2] = color.getBlue();
                    }
                }

                dct = dct(dct, temp);
                int diff = Math.abs(dct[3][4]) - Math.abs(dct[4][3]);
                int a;

                if (diff >= THRESHOLD) {
                    a = 1;
                } else if (diff <= -THRESHOLD) {
                    a = 0;
                } else {
                    break outer;
                }

                b |= a << p;
                if (p == 7) {
                    out.append((char) b);
                    b = 0;
                }
                p = (p < 7) ? p + 1 : 0;
                l++;
            }
        }
        return out.toString();
    }
```

Для демонстрації результатів виконання розробленого алгоритму Коха-Жао було обрано вихідне зображення формату BMP, розмірами 1200 на 783 пікселів, яке слугувало незаповненим стеганоконтейнером (рис. 3.8) та секретне повідомлення, розміром 454 символів (454 байтів), яке буде вбудовуватися в наше зображення-контейнер. Максимальний розмір прихованого повідомлення цим методом для цього зображення може становити до 1835 байтів.

Рисунок 3.8 – Вихідне зображення-контейнер

На рисунку 3.9 представлено те ж саме зображення, але вже з вбудованим повідомленням за допомогою методу Коха-Жао.

Рисунок 3.9 – Заповнений зображення-контейнер

Порівнюючи рисунки 3.8 та 3.9 і оцінюючи максимальну пропускну здатність передачі повідомлення для цього типу зображення, стає зрозуміло, що при довжині вбудованого повідомлення 454 байти (приблизно 25% від максимального розміру контейнера) візуально важко відрізнити порожній контейнер від контейнера, заповненого з використанням алгоритму Коха-Жао. Але все ж таки при детальному огляді, можна відрізнити заповнену частину нашого зображення-контейнера від незаповненої.

4. ПОРІВНЯЛЬНИЙ АНАЛІЗ ДОСЛІДЖЕНИХ ТА РОЗРОБЛЕНИХ МЕТОДІВ

4.1 Критерії оцінювання стеганосистем

Вибір правильних критеріїв для оцінювання стеганосистем є критично важливим, щоб можна було провести об'єктивне порівняння та зробити обґрунтовані висновки. У цьому розділі ми розглянемо основні критерії, які можна використовувати для оцінки методів стеганографії.

Під час порівняльної оцінки якості різних стеганографічних інструментів можна використовувати широко визнані метрики, які дозволяють отримати кількісні та якісні показники.

4.1.1 Кількісні критерії оцінювання

Існують певні кількісні критерії, застосовані для порівняння ефективності стеганоінструментів. Ці метрики спочатку були розроблені для роботи з растровими зображеннями на рівні окремих пікселів. Однак, після відповідної адаптації, їх можна застосовувати й до інших способів представлення зображень, а також до аудіоданих. Зазначені показники дозволяють об'єктивно оцінити та порівняти різні стеганосистеми між собою [21].

Перед тим як перейти до перерахування деяких основних кількісних критеріїв, слід зазначити параметри, які використовуються для підрахунку кожного із них, а саме:

- $C_{x,y}$ – значення пікселя оригінального контейнера;
- $S_{x,y}$ – значення відповідного пікселя заповненого контейнера;
- x, y – виступають у ролі координат пікселя в растровій сітці зображення;
- rows (C) – кількість рядків масиву C;
- cols (C) – кількість стовбців масиву C.

Середня абсолютна різниця (AD) - це показник, який відображає усереднене значення модуля відмінностей між пікселями контейнера до та після

вбудовування прихованих даних. Високе значення AD свідчить про низьку якість стеганозображення [21].

$$AD = \frac{1}{X \times Y} \sum_{x=1}^{row(C)} \sum_{y=1}^{cols(C)} \left| C_{x,y} - S_{x,y} \right| \quad (4.1)$$

Співвідношення сигналу до шуму (SNR) є одним з найпоширеніших критеріїв для аналізу спотворень, які виникають у контейнері внаслідок вбудовування прихованих даних. Це безрозмірна величина, що відображає співвідношення корисного сигналу до шумових завад. Чим більше значення SNR, тим менші спотворення вносяться в зображення.

$$SNR = \frac{\sum_{x=1}^{row(C)} \sum_{y=1}^{cols(C)} (C_{x,y})^2}{\sum_{x=1}^{row(C)} \sum_{y=1}^{cols(C)} (C_{x,y} - S_{x,y})^2} \quad (4.2)$$

Якість зображення (IF) є ключовою характеристикою для оцінки ефективності стеганографічних алгоритмів, що працюють із зображеннями, та який визначає ступінь подібності порожнього контейнера до заповненого. Ця особливість пов'язана з тим, що візуальний стеганоаналіз базується на здатності людського зору виявляти розбіжності між оригінальним та змінений зображеннями.

$$IF = 1 - \frac{\sum_{x=1}^{row(C)} \sum_{y=1}^{cols(C)} (C_{x,y} - S_{x,y})^2}{\sum_{x=1}^{row(C)} \sum_{y=1}^{cols(C)} (C_{x,y})^2} \quad (4.3)$$

Нормована взаємна кореляція (NC) - це метрика, що характеризує ступінь подібності між оригінальним зображенням-контейнером та зображенням після вбудовування прихованих даних. Значення NC близьке до 1 вказує на високий рівень схожості двох зображень.

$$NC = \frac{\sum_{x=1}^{row(C)} \sum_{y=1}^{cols(C)} (C_{x,y} \times S_{x,y})}{\sum_{x=1}^{row(C)} \sum_{y=1}^{cols(C)} (C_{x,y})^2} \quad (4.4)$$

Максимальна різниця (MD) - показник, який відображає максимальну абсолютну відмінність між пікселями оригінального та змінного зображень. Низьке значення MD свідчить про незначні зміни в стеганозображенні порівняно з вихідним контейнером, що відповідає високій якості стеганозображення.

$$MD = \max_{x,y} \left| C_{x,y} - S_{x,y} \right| \quad (4.5)$$

4.1.2 Якісні критерії оцінювання

Як зазначалося, крім кількісних, є й якісні критерії, які дозволяють більш комплексно оцінити стеганосистеми. Серед найважливіших якісних критеріїв стеганосистеми можна віднести такі:

Пропускна спроможність - це кількість бітів прихованого повідомлення, яку можна передати у зображенні фіксованого розміру за допомогою певного стеганометоду. Стеганосистема повинна забезпечувати необхідну ємність для вбудовування даних.

Стійкість характеризує здатність прихованої інформації зберігатися після застосування до стеганозображення типових операцій обробки: лінійних та нелінійних фільтрів, стиснення з втратами, змін контрастності, колірні операції, масштабування, обертання, додавання шуму, обрізання тощо. Стійкість не стосується атак, заснованих на знанні алгоритму приховування/вилучення, а лише сліпих, нецілеспрямованих модифікацій зображень.

Невидимість - це неможливість для людського зору виявити наявність прихованого повідомлення без використання спеціальних засобів стеганоаналізу. Ця характеристика ґрунтується виключно на властивостях зорової системи людини. Приховані дані вважаються невидимими, якщо пересічна людина не може відрізнити стеганозображення від оригінального контейнера.

Захищеність передбачає, що вбудована інформація не може бути видалена за допомогою цілеспрямованих атак, навіть за відомого алгоритму приховування/вилучення (окрім секретного ключа), та принаймні одного стеганозображення. Сюди також належать процедурні атаки, наприклад атаки на часткові модифікації носія через наявність вкладення.

Обчислювальна складність вбудовування та вилучення - це кількість операцій, необхідних для приховування повідомлення у контейнері та його подальшого вилучення. Стеганосистема має демонструвати прийнятний рівень складності реалізації, при цьому можлива асиметрія між кодуванням та декодуванням.

Ці вимоги конкурують між собою, і неможливо досягти їх оптимальних значень одночасно. Завжди потрібен компроміс між пропускною здатністю, невидимістю, стійкістю та складністю реалізації залежно від потреб застосування.

4.2 Порівняльний аналіз стеганографічних алгоритмів приховування інформації у зображеннях.

У цьому розділі буде представлено порівняльний аналіз різних стеганографічних алгоритмів приховування інформації у зображеннях, використовуючи кількісні критерії.

Для оцінки ефективності та надійності кожного алгоритму, були проведені тести на десяти заготовлених зображеннях з різною довжиною та типом вміщеної інформації. Кожен тестований алгоритм використовував контейнер з наповненістю 25%, що дозволило забезпечити однакові умови для всіх методів та об'єктивно порівняти їхні результати. Для тестування були обрані розроблені алгоритми та пару досліджувальні алгоритми. Серед розроблених алгоритмів: метод LSB (просторова область) та Коха-Жао (частотна область). Серед досліджувальних алгоритмів було обрано такі: метод псевдовипадкової перестановки й псевдовипадкового інтервалу (просторова область) та метод Ху і Ву (частотна область).

За результатами раніше встановлених кількісних критеріїв оцінювання стеганосистем, було сформонано декілька порівняльних таблиць з усередненими значеннями по кожному із критеріїв (таблиці 4.1-4.5).

Таблиця 4.1 - Середня абсолютна різниця (AD)

Область вбудовування	Стеганографічний алгоритм	Середня абсолютна різниця (AD)
Просторова	LSB	0.532519
	Псевдовипадкової перестановки	0.005973
	Псевдовипадкового інтервалу	0.007311
Частотна	Коха-Жао	117.748
	Ху-Ву	0.793275

Проаналізувавши результати із таблиці 4.1 можна зробити такі висновки, щодо ефективності стеганографічних алгоритмів відносно критерію AD.

Аналіз середньої абсолютної різниці (AD) показує, що серед методів просторової області найкращі результати щодо мінімізації змін у зображенні мають методи псевдовипадкової перестановки та псевдовипадкового інтервалу, що робить їх більш ефективним для приховування інформації без суттєвих видимих змін. Метод LSB також показав гарний результат, зміни в зображенні після приховування інформації даним методом будуть непомітними для людського ока, але при застосуванні спеціалізованих методів є вирогідність виявлення. У частотної області, метод Ху-Ву показав кращі результати, ніж метод Коха-Жао, значно зменшуючи видимі зміни у зображенні. Коха-Жао, відносно інших методів, має велике значення AD, що свідчить про наявність значних змін у зображенні після приховування інформації.

Таблиця 4.2 - Співвідношення сигналу до шуму (SNR)

Область вбудовування	Стеганографічний алгоритм	Співвідношення сигналу до шуму (SNR)
Просторова	LSB	56217 (47.49 дБ)
	Псевдовипадкової перестановки	$4.873*10^6$ (66.88 дБ)
	Псевдовипадкового інтервалу	$3.187*10^6$ (65.03 дБ)
Частотна	Коха-Жао	201.544 (23.04 дБ)
	Ху-Ву	30.731 (14.87 дБ)

Проаналізувавши результати із таблиці 4.2 можна зробити такі висновки, щодо ефективності стеганографічних алгоритмів відносно критерію SNR.

Аналіз співвідношення сигналу до шуму (SNR) показує, що серед методів просторової області найкращі результати мають методи псевдовипадкової перестановки та псевдовипадкового інтервалу, які значно перевищують показники методу LSB. Це свідчить про високу ефективність цих методів у збереженні якості зображення при приховуванні інформації, тому зміни в зображенні будуть абсолютно непомітними. Щодо методів у частотній області, результати є значно меншими. Метод Коха-Жао демонструє дещо вищі показники SNR у порівнянні з методом Ху-Ву, забезпечуючи помірну якість

зображення, але при детальному розгляді можуть бути помітні зміни. Стосовно методу Ху-Ву, то він має найгірший результат серед даних методів, тому тут можна спостерігати деякі втрати якості зображення, що робить його менш ефективним у порівнянні з іншими методами.

Таблиця 4.3 - Якість зображення (IF)

Область вбудовування	Стеганографічний алгоритм	Якість зображення (IF)
Просторова	LSB	0.999893
	Псевдовипадкової перестановки	0.999998
	Псевдовипадкового інтервалу	0.999997
Частотна	Коха-Жао	0.993215
	Ху-Ву	0.968645

Проаналізувавши результати із таблиці 4.3 можна зробити такі висновки, щодо ефективності стеганографічних алгоритмів відносно критерію IF.

Аналіз якості зображення (IF) показує, що серед методів просторової області найкращі результати мають методи псевдовипадкової перестановки та псевдовипадкового інтервалу, які демонструють майже ідеальне збереження якості зображення. Метод LSB також показав відмінний результат, але трохи нижчий за результати псевдовипадкових методів. У частотній області методи також продемонстрували гарні результати. Коха-Жао показав вищі показники IF у порівнянні з методом Ху-Ву, що свідчить про більш кращу здатність зберігати якість зображення. Проте, обидва методи частотної області дещо поступаються методам просторової області за цим критерієм.

Таблиця 4.4 - Нормована взаємна кореляція (NC)

Область вбудовування	Стеганографічний алгоритм	Нормована взаємна кореляція (NC)
Просторова	LSB	0.999342
	Псевдовипадкової перестановки	0.999995
	Псевдовипадкового інтервалу	0.999993
Частотна	Коха-Жао	0.984986
	Ху-Ву	0.861273

Проаналізувавши результати із таблиці 4.4 можна зробити такі висновки, щодо ефективності стеганографічних алгоритмів відносно критерію NC.

Аналіз нормованої взаємної кореляції (NC) показує, що серед методів просторової області всі мають відмінні результати, але найкращі результати знову ж таки мають методи псевдовипадкової перестановки та псевдовипадкового інтервалу, які демонструють майже ідеальну кореляцію між оригінальним і модифікованим зображенням. У частотній області метод Коха-Жао показав вищі показники NC у порівнянні з методом Ху-Ву, що вказує на високу кореляцію між оригінальним і модифікованим зображенням, але все ж таки результати є дещо нижчими, ніж показники методів просторової області. Стосовно методу Ху-Ву, він має найнижче значення NC серед усіх розглянутих методів, що вказує на помітні зміни в структурі зображення після вбудовування інформації й робить його менш ефективним у порівнянні з іншими методами.

Таблиця 4.5 - Максимальна різниця (MD)

Область вбудовування	Стеганографічний алгоритм	Максимальна різниця (MD)
Просторова	LSB	1
	Псевдовипадкової перестановки	1
	Псевдовипадкового інтервалу	1
Частотна	Коха-Жао	39
	Ху-Ву	78

Проаналізувавши результати із таблиці 4.5 можна зробити такі висновки, щодо ефективності різних стеганографічних алгоритмів відносно критерію MD.

Аналіз максимальної різниці (MD) показує, що методи просторової області (LSB, псевдовипадкова перестановка, псевдовипадковий інтервал) мають однаково низьке значення максимальної різниці, що свідчить про незначні зміни в стеганозображенні порівняно з вихідним контейнером й відповідає високій якості стеганозображення. Щодо методів частотної області, то результати є значно гіршими. Метод Коха-Жао показав значну максимальну різницю, що вказує на помітні зміни, які можуть бути помітні при детальному розгляді зображення. Метод Ху-Ву має найбільше значення MD серед усіх розглянутих методів – 78. Це свідчить про те, що використання цього методу значно змінює зображення, тому візуальна якість зображення є найгіршою в порівнянні з іншими методами,.

Під час реалізації та тестування зазначених вище алгоритмів були виявлені певні недоліки, які виникають як на етапі вбудовування, так і на етапі вилучення прихованого повідомлення. Ці недоліки можуть впливати на ефективність алгоритмів, їхню стійкість до модифікацій контейнера, продуктивність, а також на якість кінцевого зображення.

У таблиці 4.6 наведені детальні описи цих недоліків для кожного алгоритма вбудовування інформації в зображення.

Таблиця 4.6 - Недоліки алгоритмів вбудовування інформації в зображення

Область вбудовування	Стеганографічний алгоритм	Виявлені недоліки
Просторова	LSB	Є дуже чутливим до будь-яких змін у зображенні-контейнері, таких як стиснення або редагування, що може призвести до втрати прихованої інформації або помилок при її витягненні. З цього й випливає наявність помилок при декодуванні.
	Псевдовипадкової перестановки	Хоча забезпечує кращу прихованість, потребує значної кількості обчислювальних операцій для вбудовування і витягнення, що може вплинути на продуктивність. Також є вразливим до змін в контейнері.
	Псевдовипадкового інтервалу	Алгоритм може легко піддаватись впливу змін в контейнері, що ускладнює точне декодування прихованої інформації, зокрема при найменших змінах в зображенні.
Частотна	Коха-Жао	Метод вимагає значної кількості обчислювальних операцій для вбудовування та витягнення даних, що може вплинути на продуктивність. Також має низьку стійкість до атак методом сліпого перебору.
	Ху-Ву	Має обмежену ємність контейнера для приховування інформації, що знижує обсяг даних, які можуть бути приховані. Крім того, є вразливим до статистичних тестів, що може призвести до його виявлення.

ВИСНОВКИ

У сучасних умовах активного розвитку цифрових технологій та зростання обсягів інформаційного трафіку в Інтернеті забезпечення конфіденційності та безпеки переданих даних набуває критичної важливості. Стеганографія, як галузь науки про приховування інформації у цифрових носіях, відіграє ключову роль у вирішенні цього завдання, надаючи можливість приховувати секретні дані всередині звичайних файлів, таких як зображення.

У даній дипломній роботі проведено ґрунтовне дослідження існуючих методів стеганографії в зображеннях, їх розробка та порівняльний аналіз з метою визначення найефективніших підходів до приховування інформації в зображеннях для забезпечення конфіденційності й безпеки передачі даних. Детально розглянуто теоретичні основи стеганографії, принципи роботи стеганографічних систем, особливості цифрового представлення зображень та специфіку сприйняття зображень людським зором. Виявлено, що стеганографія є ефективним засобом для захисту інформації шляхом її приховування у цифрових об'єктах, таких як зображення, аудіо та відео файли.

Проведено детальний аналіз сучасних методів приховування даних у зображеннях, включаючи методи, що працюють у просторовій та частотній областях. Оцінено їх переваги та недоліки, що дозволило визначити найбільш підходящі області застосування для різних методів стеганографії залежно від конкретних вимог до безпеки, пропускної здатності та якості зображень.

На основі аналізу існуючих підходів було розроблено та протестовано на практичних прикладах найпопулярніші методи стеганографії, такі як метод заміни найменш значущого біта (LSB) в просторовій області та метод Коха-Жао в частотній області. Проведено порівняння досліджуваних та розроблених методів за кількісними критеріями, такими як середня абсолютна різниця, співвідношення сигналу до шуму, якість зображення, нормована взаємна кореляція та максимальна різниця. Це дозволило визначити найефективніші методи для різних умов застосування.

Отримані результати можуть бути використані для подальшого розвитку технологій стеганографії та підвищення рівня захисту інформації в системах інформаційної безпеки. Розроблені методи можуть бути модифіковані та адаптовані для застосування в сучасних системах інформаційної безпеки з метою забезпечення конфіденційності переданих даних та протидії кіберзагрозам.

Отже, результати проведеного дослідження існуючих методів стеганографії в зображеннях, їх розробки та порівняльного аналізу вказують на досягнення поставлених цілей і підтверджують актуальність розроблених методів приховування інформації в зображеннях для забезпечення конфіденційності та безпеки передачі даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Johnson N.F., Jajodia S. Exploring Steganography: Seeing the Unseen. IEEE Computer. 1998. Vol. 31, No. 2. P. 26-34 [Електронний ресурс]. – Режим доступу: http://www.fim.uni-linz.ac.at/lva/Rechtliche_Aspekte/2001SS/Stegano/leseecke/steganography%20seeing%20the%20unseen%20by%20neil%20f.%20johnson.pdf
2. M. Alotaibi, D. Al-hendi, B. Alroithy, and A. Gutub, “Secure Mobile Computing Authentication Utilizing Hash, Cryptography and Steganography Combination,” vol. 2, no. 1, 2019 [Електронний ресурс] / ResearchGate. – Режим доступу: https://www.researchgate.net/publication/334461824_Secure_Mobile_Computing_Authentication_Utilizing_Hash_Cryptography_and_Steganography_Combination
3. Криптографія та стеганографія: теорія і практика / В.С. Мокін, О.М. Романюк, Б.П. Мокін. Вінниця: ВНТУ, 2015. 356 с.
4. S. M. Thampi, “Information Hiding Techniques : A Tutorial Review,” 2004 [Електронний ресурс] / ResearchGate. – Режим доступу: https://www.researchgate.net/publication/1912614_Information_Hiding_Techniques_A_Tutorial_Review
5. M. Hussain and M. Hussain, “A survey of image steganography,” International Journal of Advanced Science and Technology, vol. 54, 2013 [Електронний ресурс] / ResearchGate. – Режим доступу: https://www.researchgate.net/publication/271863505_A_Survey_of_Image_Steganography_Techniques
6. Грибунин В. Г. Цифровая стеганография / В. Г. Грибунин, И. Н. Оков, И. В. Туринцев. – М. : Солон-Пресс, 2002. – 272 с [Електронний ресурс] / RoyalLib. – Режим доступу: https://royallib.com/read/gribunin_vadim/tsifrovaya_steganografiya.html#0
7. Husrev T. Sencar. Data Hiding Fundamentals And Applications / Husrev T. Sencar, Mahalingam Ramkumar, Ali N. Akansu // Digital Multimedia. ELSEVIER science and technology books. – 2004. – 364 p [Електронний ресурс] / GoogleBooks. –

Режим доступу: https://books.google.com.ua/books/about/Data_Hiding_Fundamentals_and_Application.html?id=YxJDK-pllMEC&redir_esc=y

8. Поліновський В.В. Інформаційна технологія для досліджень методів стеганографії і стегоаналізу [Електронний ресурс]. / В.В. Поліновський // Міжвузівський збірник "Комп'ютерно-інтегровані технології: освіта, наука, виробництво ". – Луцьк, 2011. - №5 – с.236-242. – Режим доступу: https://icyb180.org.ua/wp-content/uploads/2012/04/informatsiyna-tehnologiya-dlya-doslidzhennya-metodiv-steganografiyi-i-stegoanalizu.pdf
9. Сучасні стеганографічні методи захисту інформації / [Стасюк О.І., Гнатюк С.О., Довгич Н.І., Літош М.С.] [Електронний ресурс] / ResearchGate. – Режим доступу: https://www.researchgate.net/publication/311663916_SUCASNI_STEGANOGRAFICNI_METODI_ZAHISTU_INFORMACII
10. Барсуков В.С. Компьютерная стеганография вчера, сегодня, завтра. Технологии информационной безопасности 21 века / В.С. Барсуков, А.П. Романцов ; – М : "Специальная Техника", 2007. – 225 с.
11. Conway M. Steganography, Signals Intelligence, and Terrorism // Knowledge, Technology and Policy. – 2003. –V.16, No2. – P. 45-47.
12. Конахович Г.Ф. Компьютерная стеганография. Теория и практика / Г.Ф. Конахович, А.Ю Пузыренко –К.: МК-Пресс, 2006. – 249 с.
13. Стеганографія: навчальний посібник [Електронний ресурс] / О. О. Кузнецов, С. П. Євсеєв, О. Г. Король. – Х. : Вид. ХНЕУ, 2011. – 232 с. – Режим доступу: http://repository.hneu.edu.ua/bitstream/123456789/2289/1/Стеганография.pdf
14. M. G.R and A. Danti, "A Novel Hash Based Least Significant Bit (2-3-3) Image Steganography In Spatial Domain," International Journal of Security, Privacy and Trust Management (IJSPTM), vol. 4, no. 1, 2015 [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1503.03674
15. S. Sharma1 and U. Kumar, "Review of Transform Domain Techniques for Image Steganography," International Journal of Science and Research (IJSR), vol. 4, no. 5, 2015 [Електронний ресурс] / ResearchGate. – Режим доступу:

https://www.researchgate.net/publication/287210148_Review_of_Transform_Domain_Techniques_for_Image_Steganography

16. Dutta1, Poulami, Bhattacharyya1, Debnath & Kim2, Tai-hoon " Data Hiding in Audio Signal: A Review " International Journal of Database Theory and Application, Vol. 2, No. 2, June 2009 [Електронний ресурс]. – Режим доступу: https://www.academia.edu/6341503/Data_Hiding_in_Audio_Signal_A_Review
17. Генне О.В. Основные положения стеганографии [Електронний ресурс] // Защита информации. Конфидент – 2000. No3 – 56 с. – Режим доступу: http://www.infocity.kiev.ua/hack/content/hack037.phtml
18. Provos N., Honeyman P. Hide and Seek: An Introduction to Steganography. IEEE Security & Privacy. 2003. Vol. 1, No. 3. P. 32-44 [Електронний ресурс]. – Режим доступу: http://niels.xtdnet.nl/papers/practical.pdf
19. Johnson N.F., Jajodia S. Exploring Steganography: Seeing the Unseen. IEEE Computer. 1998. Vol. 31, No. 2. P. 26-34 [Електронний ресурс]. – Режим доступу: http://www.fim.uni-linz.ac.at/lva/Rechtliche_Aspekte/2001SS/Stegano/leseecke/steganography%20seeing%20the%20unseen%20by%20neil%20f.%20johnson.pdf
20. Gonzalez R.C., Woods R.E. Digital Image Processing. Pearson, 2018. 1168 p [Електронний ресурс]. – Режим доступу: https://dl.icdst.org/pdfs/files4/01c56e081202b62bd7d3b4f8545775fb.pdf
21. Конахович Г. Ф. Компьютерная стеганография : теория и практика [Електронний ресурс] / Г. Ф.Конахович, А. Ю. Пузыренко. — К. : МК-Пресс, 2006.— 288 с. – Режим доступу: https://studfile.net/preview/7379018/
22. Souvik Bhattacharyya and Gautam Sanyal, "Data Hiding in Images in Discrete Wavelet Domain Using PMM," International Journal of Electrical and Computer Engineering, 2010 [Електронний ресурс] / ResearchGate. – Режим доступу: https://www.researchgate.net/publication/236953151_Data_Hiding_in_Images_in_Discrete_Wavelet_Domain_Using_PMM
23. Сорока Н. И. Теория передачи информации : конспект лекций [для студентов специальности 1-53 01 07 «Информационные технологии и управление в

технических системах»] [Електронний ресурс] /Н. И. Сорока, Г. А. Кривинченко. — Минск: БГУИР,2005. — 301 с. – Режим доступу: https://www.studmed.ru/soroka-ni-krivinchenko-ga-teoriya-peredachi-informacii_7545cad69de.html

24. Методы цифровой стеганографии для защиты изобразительной информации / В.Н.Горбачев, Е.М. Кайнарова, А.И. Кулик, И.К. Метелев.— М. : МГУП, 2011. — 224 с.
25. Бабич І. В. Огляд стеганографічних методів перетворення інформації в зображеннях [Електронний ресурс] /І.В. Бабич, С.А. Паламарчук, Н.А. Паламарчук, В.В.Овсянніков // Захист інформації. — 2012. — No 1. —С. 18-24. – Режим доступу: https://jrnl.nau.edu.ua/index.php/ZI/article/view/2058
26. Nagham Hamid, Abid Yahya, R. Badlishah Ahmad and Osamah Ma. Al-Qershi Image Steganography Techniques: An Overview, International Journal of Computer Science and Security (IJCSS),Volume 6, Issue 3, 2012 [Електронний ресурс]. – Режим доступу: https://www.cscjournals.org/manuscript/Journals/IJCSS/Volume6/Issue3/IJCSS-670.pdf
27. Johnson N. F. A Survey of steganographic techniques / N.F. Johnson, S. Katzenbeisser //Information Hiding Techniques for Steganography and Digital Watermarking. — Ed. London : Artech House,2000. — PP. 43-78 [Електронний ресурс] / ResearchGate. – Режим доступу: https://www.researchgate.net/publication/245096254_A_survey_of_steganographic_techniques
28. Cheddad A. Digital Image Steganography:Survey and Analysis of Current Methods / A. Cheddad, J. Condell, K. Curran, P. Kevitt [Електронний ресурс] // 15th Annual IEEEInternational Conference and Workshop on theEngineering of Computer Based Systems, 2008. – Режим доступу: https://abbascheddad.net/Survey.pdf
29. Хорошко В. А. Методы и средства защиты информации / В. А. Хорошко, А. А. Чекатков. — К. :Юниор, 2003. — 501 с.

30. Ажбаев Т. Г. Анализ стойкости современных стеганографических алгоритмов / Т. Г. Ажбаев,И.М. Ажмухамедов // Вестник Астраханского государственного технического университета. —2008. — С. 56-61
31. Стасюк О.І. Сучасні стеганографічні методи захисту інформації / О.І. Стасюк, С.О.Гнатюк, Н.І. Довгич, М.С. Літош // Захистінформації. — 2011. — No1 (50). — С. 56–63
32. Рябко Б.Я. Основы современной криптографии и стеганографии [Електронний ресурс] / Б.Я. Рябко, А.Н. Фионов.— М. : Горячая линия – Телеком, 2010. — 232 с. – Режим доступу: https://dokumen.pub/978-5-9912-0150-6.html
33. Методы цифровой стеганографии для защиты изобразительной информации / В.Н.Горбачев, Е.М. Кайнарова, А.И. Кулик, И.К. Метелев.— М. : МГУП, 2011. — 224 с.
34. Тарасов Д. О. Класифікація та аналіз безкоштовних програмних засобів стеганографії [Електронний ресурс] / Д.О. Тарасов, А. С. Мельник, М. М. Голобородько //Інформаційні системи та мережі, 2010. — С. 365-373. – Режим доступу: https://science.lpnu.ua/sites/default/files/journal-paper/2019/apr/16081/vis673ism-365-373.pdf
35. Алефиренко В. М. Основы защиты информации [Електронний ресурс] / В. М. Алефиренко, Ю. В.Шамгин. — Мн. : БГУИР, 2004. — 43 с. – Режим доступу: https://www.bsuir.by/m/48559.pdf
36. J.Y. Hsiao. C.C. Chang. and C.-S. Chan. Finding optimal least-significant-bit substitution in image hiding by dynamic programming strategy. Pattern Recognition, 36:1583–1595, 2003 [Електронний ресурс] / ResearchGate. – Режим доступу: https://www.researchgate.net/publication/223679082_Finding_optimal_least-significant-bit_substitution_in_image_hiding_by_dynamic_programming_strategy
37. C.K. Chan. and L. M.Cheng. Hiding data in images by simple lsb substitution. Pattern Recognition, 37:469–474, 2004 [Електронний ресурс] / ResearchGate. –

Режим доступу: https://www.researchgate.net/publication/222564467_Hiding_data_in_images_by_simple_LSB_substitution

38. Y. K. Lee. and L. H. Chen. High capacity image steganographic model.IEE Proc.-Vision, Image and Signal Processing, 147:288–294, 2000 [Електронний ресурс] / ResearchGate. – Режим доступу: https://www.researchgate.net/publication/3359120_High_capacity_image_steganography_model
39. C.F. Lin. R.Z. Wang. and J.C. Lin. Image hiding by optimal lsb substitution and genetic algorithm. Pattern Recognition, 34:671–683,2001 [Електронний ресурс] / ResearchGate. – Режим доступу: https://www.researchgate.net/publication/220599098_Image_hiding_by_optimal_LSB_substitution_and_genetic_algorithm
40. D.C. Wu. and W.H. Tsai. A steganographic method for images by pixel-value differencing. Pattern Recognition Letters, 24:1613–1626, 2003 [Електронний ресурс] / ResearchGate. – Режим доступу: https://www.researchgate.net/publication/223256802_A_steganographic_method_for_images_by_pixel_value_differncing
41. P Huang. K.C. Chang., C.P Chang. and T.M Tu. A novel image steganography method using tri-way pixel value differencing. Journal of Multimedia, 3, 2008 [Електронний ресурс] / ResearchGate. – Режим доступу: https://www.researchgate.net/publication/42803920_A_Novel_Image_Steganographic_Method_Using_Tri-way_Pixel-Value_Differencing
42. Potdar V.and Chang E. Gray level modification steganography for secret communication. In IEEE International Conference on Industrial Informatics., pages 355–368, Berlin, Germany, 2004 [Електронний ресурс] / ResearchGate. – Режим доступу: https://www.researchgate.net/publication/4137627_Grey_level_modification_steganography_for_secret_communication
43. Рейтинг мов програмування 2024 [Електронний ресурс]. – Режим доступу: https://dou.ua/lenta/articles/language-rating-2024/

ДОДАТОК А

```
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.List;
import javax.imageio.ImageIO;

public class LSB_encode {
    static final String MESSAGEFILE = "message.txt";
    static final String COVERIMAGEFILE = "image.bmp";
    static final String STEGIMAGEFILE = "stego_image.bmp";

    public static void main(String[] args) throws Exception {
        String contentOfMessageFile = readMessageFile();
        int[] bits = bit_Msg(contentOfMessageFile);
        System.out.println("Вміст текстового повідомлення " +
contentOfMessageFile);
        // Вивід повідомлення в бітовому представленні
        /*
        for (int bit : bits) {
            System.out.print(bit);
        }
        */
        System.out.println();
        BufferedImage theImage = readImageFile(COVERIMAGEFILE);
        hideTheMessage(bits, theImage);
    }

    public static String readMessageFile() throws IOException {
        List<String> lines = Files.readAllLines(Paths.get(MESSAGEFILE),
StandardCharsets.UTF_8);
        return String.join("\n", lines);
    }

    public static int[] bit_Msg(String msg) {
        byte[] bytes = msg.getBytes(StandardCharsets.UTF_8);
        int[] b_msg = new int[bytes.length * 8];
        int j = 0;
        for (byte b : bytes) {
            String x_s = String.format("%8s", Integer.toBinaryString(b &
0xFF)).replace(' ', '0');
            for (int i1 = 0; i1 < 8; i1++) {
                b_msg[j++] = x_s.charAt(i1) - '0';
            }
        }
        return b_msg;
    }

    public static BufferedImage readImageFile(String COVERIMAGEFILE) {
        BufferedImage theImage = null;
        File p = new File(COVERIMAGEFILE);
        try {
            theImage = ImageIO.read(p);
        } catch (IOException e) {
            e.printStackTrace();
            System.exit(1);
        }
        return theImage;
```

```
    }

    public static void hideTheMessage(int[] bits, BufferedImage theImage) throws
Exception {
        File f = new File(STEGIMAGEFILE);
        int[] bl_msg = new int[32];
        // Вивід байтової довшини повідомлення
        // System.out.println("byte length " + bits.length / 8);
        String bl_s = String.format("%32s", Integer.toBinaryString(bits.length /
8)).replace(' ', '0');
        for (int i1 = 0; i1 < 32; i1++) {
            bl_msg[i1] = bl_s.charAt(i1) - '0';
        }
        int j = 0;
        int currentBitEntry = 32;

        for (int x = 0; x < theImage.getWidth(); x++) {
            for (int y = 0; y < theImage.getHeight(); y++) {
                int currentPixel = theImage.getRGB(x, y);
                int red = (currentPixel >> 16) & 255;
                int green = (currentPixel >> 8) & 255;
                int blue = currentPixel & 255;
                if (x == 0 && y < 32) {
                    blue = (blue & ~1) | bl_msg[y];
                } else if (currentBitEntry < bits.length + 32) {
                    blue = (blue & ~1) | bits[j++];
                    currentBitEntry++;
                }
                int rgb = (255 << 24) | (red << 16) | (green << 8) | blue;
                theImage.setRGB(x, y, rgb);
            }
        }
        ImageIO.write(theImage, "bmp", f);
    }
}
```

ДОДАТОК Б

```
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.nio.charset.StandardCharsets;
import javax.imageio.ImageIO;

public class LSB_decode {
    static final String STEGIMAGEFILE = "stego_image.bmp";
    static final String DECODEDMESSAGEFILE = "message_dec.txt";

    public static String b_msg = "";
    public static int len = 0;

    public static void main(String[] args) throws Exception {
        BufferedImage yImage = readImageFile(STEGIMAGEFILE);
        DecodeTheMessage(yImage);
        byte[] bytes = new byte[len];
        for (int i = 0; i < len; i++) {
            String sub = b_msg.substring(i * 8, (i + 1) * 8);
            bytes[i] = (byte) Integer.parseInt(sub, 2);
        }
        String msg = new String(bytes, StandardCharsets.UTF_8);
        // Вивід секретного повідомлення
        System.out.println("Декодоване повідомлення: " + msg);

        try (PrintWriter out = new PrintWriter(new
FileWriter(DECODEDMESSAGEFILE, false))) {
            out.write(msg);
        }
    }

    public static BufferedImage readImageFile(String COVERIMAGEFILE) {
        BufferedImage theImage = null;
        File p = new File(COVERIMAGEFILE);
        try {
            theImage = ImageIO.read(p);
        } catch (IOException e) {
            e.printStackTrace();
            System.exit(1);
        }
        return theImage;
    }

    public static void DecodeTheMessage(BufferedImage yImage) throws Exception {
        int currentBitEntry = 0;
        StringBuilder bx_msg = new StringBuilder();
        for (int x = 0; x < yImage.getWidth(); x++) {
            for (int y = 0; y < yImage.getHeight(); y++) {
                int currentPixel = yImage.getRGB(x, y);
                int blue = currentPixel & 255;
                if (x == 0 && y < 32) {
                    bx_msg.append(blue & 1);
                    if (y == 31) {
                        len = Integer.parseInt(bx_msg.toString(), 2);
                    }
                } else if (currentBitEntry < len * 8) {
                    b_msg += (blue & 1);
                    currentBitEntry++;
```

```
                }
            }
        }
        // Вивід секретного повідомлення в бітовому представленні
        //System.out.println("Вивід повідомлення в бітовому представленні: " +
b_msg);
    }
}
```

ДОДАТОК В

```
import java.awt.Color;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.List;
import javax.imageio.ImageIO;


public class KochZhao_encode {

    private static final int BLOCK_SIZE = 8;
    private static final double THRESHOLD = 25.0;
    private static final int CONSTANT = 50;

    static final String MESSAGEFILE = "message1.txt";
    static final String COVERIMAGEFILE = "image.bmp";
    static final String STEGIMAGEFILE = "stego_image1.bmp";

    static final double[][] cos_t = {
            {1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0},
            {0.9807853, 0.8314696, 0.5555702, 0.1950903, -0.1950903, -0.5555702,
-0.8314696, -0.9807853},
            {0.9238795, 0.3826834, -0.3826834, -0.9238795, -0.9238795, -
0.3826834, 0.3826834, 0.9238795},
            {0.8314696, -0.1950903, -0.9807853, -0.5555702, 0.5555702,
0.9807853, 0.1950903, -0.8314696},
            {0.7071068, -0.7071068, -0.7071068, 0.7071068, 0.7071068, -
0.7071068, -0.7071068, 0.7071068},
            {0.5555702, -0.9807853, 0.1950903, 0.8314696, -0.8314696, -
0.1950903, 0.9807853, -0.5555702},
            {0.3826834, -0.9238795, 0.9238795, -0.3826834, -0.3826834,
0.9238795, -0.9238795, 0.3826834},
            {0.1950903, -0.5555702, 0.8314696, -0.9807853, 0.9807853, -
0.8314696, 0.5555702, -0.1950903}
    };

    static final double[][] e = {
            {0.125, 0.176777777, 0.176777777, 0.176777777, 0.176777777,
0.176777777, 0.176777777, 0.176777777},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25}
    };

    public static void main(String[] args) throws Exception {
        String contentOfMessageFile = readMessageFile();
        BufferedImage theImage = ImageIO.read(new File(COVERIMAGEFILE));
        insertMessage(theImage, contentOfMessageFile);
    }

    public static String readMessageFile() throws IOException {
        List<String> lines = Files.readAllLines(Paths.get(MESSAGEFILE));
        return String.join("\n", lines);
    }
```

```
    public static void insertMessage(BufferedImage theImage, String text) throws
Exception{
        File f = new File(STEGIMAGEFILE);
        int[][][] temp = new int[BLOCK_SIZE][BLOCK_SIZE][3];
        int[][] dct = new int[BLOCK_SIZE][BLOCK_SIZE];
        int l = 0;
        int column = theImage.getHeight(), row = theImage.getWidth(), type;
        if(row > column){
            type = row;
            row = column;
            column = type;
        }
        outer: for (int i = 0; i < row; i += BLOCK_SIZE) {
            for (int j = 0; j < column; j += BLOCK_SIZE) {
                if (l >= text.length() * 8) break;
                for (int x = 0; x < BLOCK_SIZE; x++) {
                    for (int y = 0; y < BLOCK_SIZE; y++) {
                        Color color = new Color(theImage.getRGB(j + x, i + y));
                        temp[x][y][0] = color.getRed();
                        temp[x][y][1] = color.getGreen();
                        temp[x][y][2] = color.getBlue();
                    }
                }

                dct = dct(dct, temp);
                int diff = Math.abs(dct[3][4]) - Math.abs(dct[4][3]);

                if (getBit(text, l)) {
                    if (diff <= THRESHOLD) {
                        dct[3][4] = (dct[3][4] >= 0) ? Math.abs(dct[4][3]) +
CONSTANT : -Math.abs(dct[4][3]) - CONSTANT;
                    }
                } else {
                    if (diff >= -THRESHOLD) {
                        dct[4][3] = (dct[4][3] >= 0) ? Math.abs(dct[3][4]) +
CONSTANT : -Math.abs(dct[3][4]) - CONSTANT;
                    }
                }

                temp = idct(dct, temp);

                for (int x = 0; x < BLOCK_SIZE; x++) {
                    for (int y = 0; y < BLOCK_SIZE; y++) {
                        Color newColor = new Color(temp[x][y][0], temp[x][y][1],
temp[x][y][2]);
                        theImage.setRGB(j + x, i + y, newColor.getRGB());
                    }
                }
                l++;
            }
            if (l >= text.length() * 8) break;
        }
        ImageIO.write(theImage, "bmp", f);
    }

    public static int[][] dct(int[][] dct, int[][][] arr) {
        double temp;
        for (int i = 0; i < BLOCK_SIZE; i++) {
            for (int j = 0; j < BLOCK_SIZE; j++) {
                temp = 0.0;
                for (int x = 0; x < BLOCK_SIZE; x++) {
                    for (int y = 0; y < BLOCK_SIZE; y++) {
                        temp += cos_t[i][x] * cos_t[j][y] * arr[x][y][2];
```

```
                }
            }
            dct[i][j] = (int) (e[i][j] * temp);
        }
    }
    return dct;
}

public static int[][][] idct(int[][] dct, int[][][] arr) {
    double temp;
    for (int i = 0; i < BLOCK_SIZE; i++) {
        for (int j = 0; j < BLOCK_SIZE; j++) {
            temp = 0;
            for (int x = 0; x < BLOCK_SIZE; x++) {
                for (int y = 0; y < BLOCK_SIZE; y++) {
                    temp += dct[x][y] * cos_t[x][i] * cos_t[y][j] * e[x][y];
                }
            }
            int value = (int) Math.round(temp);
            arr[i][j][2] = Math.min(255, Math.max(0, value));
        }
    }
    return arr;
}

public static boolean getBit(String str, int pos) {
    return (str.charAt(pos / 8) & (1 << (pos % 8))) > 0;
}
}
```

ДОДАТОК Г

```
import java.awt.Color;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.FileWriter;
import java.io.PrintWriter;
import javax.imageio.ImageIO;


public class KochZhao_decode {

    private static final int BLOCK_SIZE = 8;
    private static final double THRESHOLD = 25.0;

    static final String STEGIMAGEFILE = "stego_image1.bmp";
    static final String DECODEDMESSAGEFILE = "message_dec1.txt";


    static final double[][] cos_t = {
            {1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0},
            {0.9807853, 0.8314696, 0.5555702, 0.1950903, -0.1950903, -0.5555702,
-0.8314696, -0.9807853},
            {0.9238795, 0.3826834, -0.3826834, -0.9238795, -0.9238795, -
0.3826834, 0.3826834, 0.9238795},
            {0.8314696, -0.1950903, -0.9807853, -0.5555702, 0.5555702,
0.9807853, 0.1950903, -0.8314696},
            {0.7071068, -0.7071068, -0.7071068, 0.7071068, 0.7071068, -
0.7071068, -0.7071068, 0.7071068},
            {0.5555702, -0.9807853, 0.1950903, 0.8314696, -0.8314696, -
0.1950903, 0.9807853, -0.5555702},
            {0.3826834, -0.9238795, 0.9238795, -0.3826834, -0.3826834,
0.9238795, -0.9238795, 0.3826834},
            {0.1950903, -0.5555702, 0.8314696, -0.9807853, 0.9807853, -
0.8314696, 0.5555702, -0.1950903}
    };

    static final double[][] e = {
            {0.125, 0.176777777, 0.176777777, 0.176777777, 0.176777777,
0.176777777, 0.176777777, 0.176777777},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25},
            {0.176777777, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25}
    };

    public static void main(String[] args) throws Exception {
        BufferedImage imageWithMessage = ImageIO.read(new File(STEGIMAGEFILE));
        String msg = readMessage(imageWithMessage);
        // Вивід секретного повідомлення
        System.out.println("Декодоване повідомлення: " + msg);
        try (PrintWriter out = new PrintWriter(new
FileWriter(DECODEDMESSAGEFILE, false))) {
            out.write(msg);
        }
    }

    public static String readMessage(BufferedImage image) {
        int[][][] temp = new int[BLOCK_SIZE][BLOCK_SIZE][3];
```

```
        int[][] dct = new int[BLOCK_SIZE][BLOCK_SIZE];
        int l = 0, p = 0, b = 0;
        int column = image.getHeight(), row = image.getWidth(), type;
        if(row > column){
            type = row;
            row = column;
            column = type;
        }
        StringBuilder out = new StringBuilder();
        outer: for (int i = 0; i < row; i += BLOCK_SIZE) {
            for (int j = 0; j < column; j += BLOCK_SIZE) {
                for (int x = 0; x < BLOCK_SIZE; x++) {
                    for (int y = 0; y < BLOCK_SIZE; y++) {
                        Color color = new Color(image.getRGB(j + x, i + y));
                        temp[x][y][0] = color.getRed();
                        temp[x][y][1] = color.getGreen();
                        temp[x][y][2] = color.getBlue();
                    }
                }

                dct = dct(dct, temp);
                int diff = Math.abs(dct[3][4]) - Math.abs(dct[4][3]);
                int a;

                if (diff >= THRESHOLD) {
                    a = 1;
                } else if (diff <= -THRESHOLD) {
                    a = 0;
                } else {
                    break outer;
                }

                b |= a << p;
                if (p == 7) {
                    out.append((char) b);
                    b = 0;
                }
                p = (p < 7) ? p + 1 : 0;
                l++;
            }
        }
        return out.toString();
    }

    public static int[][] dct(int[][] dct, int[][][] arr) {
        double temp;
        for (int i = 0; i < BLOCK_SIZE; i++) {
            for (int j = 0; j < BLOCK_SIZE; j++) {
                temp = 0.0;
                for (int x = 0; x < BLOCK_SIZE; x++) {
                    for (int y = 0; y < BLOCK_SIZE; y++) {
                        temp += cos_t[i][x] * cos_t[j][y] * arr[x][y][2];
                    }
                }
                dct[i][j] = (int) (e[i][j] * temp);
            }
        }
        return dct;
    }

}
```