Міністерство освіти і науки України
Харківський національний університет імені В.Н. Каразіна

До друку і в світ дозволяю.
Перший проректор
_____
“____”_____ 201__ р.

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ
до лабораторних робіт з дисципліни
«Стеганографія»

для студентів напрямку
6.170101 «Безпека інформаційних і комунікаційних систем»
(спеціальності **125 «Кібербезпека»**)

Всі цитати, цифровий, фактичний матеріал та бібліографічні відомості перевірені, написання одиниць відповідає стандартам

РЕКОМЕНДОВАНО
науково-методичною радою університету.
Протокол №
від “___” “_____”

Упорядник: _____О.О. Кузнецов

Відповідальний випусковий: _____С.Г. Рассомахін

Начальник методичного відділу _____
Начальник КВВ ННВПЦ _____

Харків 2016

Методичні рекомендації до лабораторних робіт з дисципліни «Стеганографія» для студентів напрямку 6.170101 «Безпека інформаційних і комунікаційних систем» (спеціальності 125 «Кібербезпека» / Упоряд. Кузнецов О.О. – Харків: ХНУ ім. В.Н. Каразіна, 2016. – ХХХ с.

Упорядники: О.О. Кузнецов

Рецензенти:
О.В.Лемешко, д.т.н., доц. професор каф. Телекомунікаційних систем ХНУРЕ

ЗМІСТ

Лабораторна робота №1 «Приховування даних в просторовій області нерухомих зображень шляхом модифікації найменш значущого біта»

1. Мета та завдання лабораторної роботи

Мета роботи: закріпити теоретичні знання за темою «Приховування даних у просторової області нерухомих зображень шляхом модифікації найменш значущого біту даних (НЗБ, LSB – Least Significant Bit)», набути практичних вмінь та навичок щодо розробки стеганографічних систем, дослідити властивості стеганографічних методів, що засновані на низькорівневих властивостях зорової системи людини (ЗСЛ).

Лабораторна робота №1 виконується у середовищі символьної математики MathCAD версії 12 або вище. Допускається виконання лабораторної роботи із використанням інших середовищ або мов програмування, які вивчалися студентами під час навчання.

Завдання лабораторної роботи

1. **Завдання 1. Реалізація алгоритмів приховування і вилучення повідомлень методом заміни найменш значущого біту даних:**
   - реалізувати у середовищі символьної математики MathCAD (або іншого середовища / мови програмування) алгоритми приховування та вилучення даних у просторової області зображень шляхом модифікації найменш значущого біту даних (методом LSB);
   - застосовуючи розроблену програмну реалізацію виконати стеганографічне кодування інформаційного повідомлення, тобто сформувати заповнений контейнер (стеганограму);
   - виконати зорове порівняння пустого та заповненого контейнера та переконатися у відсутності помітних похибок;
   - виконати вилучення вбудованого повідомлення, переконатися в його автентичності;
   - отримати заповнені контейнери від інших груп розробників та переконатися у відсутності помітних похибок;
   - вилучити повідомлення із отриманих стеганоконтейнерів інших груп розробників та переконатися у їхній автентичності.
2. **Завдання 2. Експериментальні дослідження зорового порогу чуттєвості людини до змін яскравості зображень:**
   - виконати приховування даних у різні за значущістю біти зображення, починаючи з найменш значущого;
   - експериментально встановити, модифікація яких бітів зображення не приводить до помітних похибок;
   - розрахувати за отриманими емпіричними даними зоровий поріг чуттєвості до змін яскравості нерухомих зображень.

3. **Завдання 3. Реалізація алгоритмів приховування і вилучення повідомлень методом псевдовипадкової перестановки:**

– реалізувати у середовищі символьної математики MathCAD (або іншого середовища / мови програмування) алгоритм приховування та вилучення даних у просторової області зображень методом псевдовипадкової перестановки (методом ПВП);
– застосовуючи розроблену програмну реалізацію виконати стеганографічне кодування інформаційного повідомлення (сформувати стеганограму). Виконати вилучення вбудованого повідомлення, переконатися в його автентичності;
– ввести інший секретний ключ (таблицю псевдовипадкової перестановки) та спробувати вилучити інформаційне повідомлення з контейнеру, переконатися в спотворенні інформації. Розрахувати кількість можливих секретних ключів та ймовірність вгадування секретного ключа зловмисником.

4. **Завдання 3. Реалізація алгоритмів приховування і вилучення повідомлень методом псевдовипадкового інтервалу:**

– реалізувати у середовищі символьної математики MathCAD (або іншого середовища / мови програмування) алгоритм приховування та вилучення даних у просторової області зображень методом та псевдовипадкового інтервалу (методом ПВІ);
– застосовуючи розроблену програмну реалізацію сформувати стеганограму, виконати вилучення вбудованого повідомлення, переконатися в його автентичності;
– ввести інший секретний ключ (який задає правило формування псевдовипадкового інтервалу) та спробувати вилучити інформаційне повідомлення з контейнеру, переконатися в спотворенні інформації. Розрахувати кількість можливих секретних ключів та ймовірність вгадування секретного ключа зловмисником.

2. Методичні вказівки з організації самостійної роботи

1. Вивчити теоретичний матеріал лекції «Приховування даних у просторової області зображень шляхом модифікації найменш значущого біту даних».
2. Вивчити матеріал основного джерела літератури (Конахович Г.Ф., Пузыренко А.Ю. Компьютерная стеганография):
   - особливості зорової системи людини (ст. 73-75);
   - приховування даних у просторовій області зображень (ст. 76);
   - метод заміни найменш значущого біту (ст. 76-89);
   - метод псевдовипадкового інтервалу (ст. 89-92);
   - метод псевдовипадкової перестановки (ст. 92-96).
3. Вивчити основні команди у середовищі символьної математики MathCAD для роботи із зображеннями.

4. Підготувати відповіді на контрольні запитання.
5. Підготувати звіт з лабораторної роботи.

Допуск до виконання лабораторної роботи здійснюється за результатами письмового опитування (контрольної роботи).

3. Загальнотеоретичні положення за темою лабораторної роботи

3.1 Властивості зорової системи людини, які використовуються в стеганографії

Властивості зорової системи людини (ЗСЛ), які використовуються в стеганографії, можна розділити на дві основні групи: ***низькорівневі*** («фізіологічні») і ***високорівневі*** («психофізіологічні»).

Виділяють три найважливіші ***низькорівневі властивості,*** що впливають на помітність стороннього шуму в зображенні:

- мала чутливість до незначних змін яскравості зображення;
- мала чутливість до незначних змін контрастності зображення;
- частотна чутливість;
- ефект маскування;
- мала чутливість до незначних змін яскравості каналу синього кольору зображення.

Розглянемо ці властивості більш докладно.

Чутливість до зміни яскравості. Здатність ока людини реагувати на світлове роздратування характеризується *чутливістю*. Чутливість ока до дії випромінювання визначається величиною, яка є зворотною до яскравості $L_П$, що викликає граничне роздратування: $v = 1/L_П$. Чутливість може виражатися і в одиницях, зворотних до граничної освітленості спостережуваного зображення. Дослідження показали, що поріг чутливості ЗСЛ до зміни яскравості дорівнює 2-3%. Тобто, наприклад, якщо яскравість зображення у цифровому вигляді кодується цілими числами в діапазоні 0 ... 255 (всього $L_m = 256$ рівнів квантування), тоді зміна яскравості на $\Delta L = 8$ рівнів змінить освітленість на

$$\Delta = \frac{\Delta \mathrm{L}}{\mathrm{L_m}} 100\% = \frac{8}{256} 100\% \approx 3\%$$

відносно граничної освітленості і ця зміна не призведе до видимих викривлень зображення.

Чутливість до зміни контрастності. Якщо на око людини впливає зображення з яскравістю ділянок L, тоді спостерігач реагує не тільки на абсолютну зміну яскравості ΔL, а і на її відносне значення $\Delta L/L$. Мінімальна відносна зміна яскравості $\Delta L/L$, яка сприймається спостерігачем, називається *відносним різницевим порогом роздратування*.

На практиці вважають, що поріг чутливості ЗСЛ до незначних змін контрастності складає 2 .. 4%, тобто спотворення, що вносяться, з відношенням $\Delta L/L < 0,02\ldots0,04$ ЗСЛ не сприймає. І навпаки, якщо спотворення зображень приводять до $\Delta L/L > 0,04$ такі спотворення людина відмітить. Оцінка $0,002\ldots0,004$ отримана емпіричним шляхом, тобто вона може бути відмінною для різних довжин хвиль (різних колірних складових зображення) і є індивідуальною характеристикою органів зору конкретної людини. Наприклад, якщо як зображення використовувати *.bmp файл з кодуванням кожного піксела 24 байтами (по 8 біт на кожен канал червоного, зеленого і синього кольору), а відповідний поріг чутливості ЗСЛ до зміни контрастності (відносної зміни яскравості до граничної освітленості) складе:

$$\Delta L/L = \Delta L/256 = 0,02\ldots0,04\ \Delta\ \text{L/L= }\Delta\ \text{L/256=0,02…0,04,}$$

тоді при зміні $\Delta L = 5\ldots10$ в кодуванні файлу *.bmp ЗСЛ спотворень не сприйме.

Частотна чутливість ЗСЛ проявляється в тім, що людина набагато більше сприйнятлива до низькочастотного (НЧ), ніж до високочастотного (ВЧ) шуму. Це пов’язане з нерівномірністю амплітудно-частотної характеристики системи зору людини.

Ефект маскування. Елементи ЗСЛ розділяють вступний відеосигнал на окремі компоненти. Кожна складова збуджує нервові закінчення ока через ряд підканалів. Виділювані оком компоненти мають різні просторові й частотні характеристики, а також різну орієнтацію (горизонтальну, вертикальну, діагональну). У випадку одночасного впливу на око двох компонентів з подібними характеристиками збуджуються ті самі підканали. Це приводить до ефекту маскування, що полягає в збільшенні порога виявлення відеосигналу в присутності іншого сигналу, що володіє аналогічними характеристиками. Тому, адитивний шум набагато помітніше на гладких ділянках зображення, ніж на високочастотних, тобто в останньому випадку спостерігається маскування. Найбільше сильно ефект маскування проявляється, коли обидва сигнали мають однакову орієнтацію й місце розташування.

Чутливість до зміни каналу синього кольору. Ще одним відомим феноменом ЗСЛ є її чутливість до змін колірних каналів. Якщо проаналізувати криві залежностей спектральної чутливості людського ока до потоку світлового випромінювання, можна помітити, що людина дуже добре здатна сприймати зелені і зелено-жовті кольори, тоді як його чутливість до синім кольорам помітно нижче. Крім того, очному кришталику важче фокусуватися на предмети, якщо вони забарвлені в синьо-фіолетові тони. Це пояснюється падінням спектральної чутливості ока в цих областях спектру. Тому окуляри іноді роблять не нейтрально-прозорими, а із забарвлених в жовтий або коричневий колір стекол, які фільтрують синьо-фіолетову складову спектру.

Існує припущення, що знижена чутливість ЗСЛ до змін в каналах синього кольору пов'язана з переважанням в природі предметів (об'єктів) що

мають зелений колір, і практично повною відсутністю останніх строго синього кольору. На практиці, різна чутливість ЗСЛ до колірних складових растрових даних виражається в оцінці повнокольорової яскравості пікселя

$$Y = 0{,}58662 \cdot G + 0{,}2989 \cdot R + 0{,}11448 \cdot B$$

тобто внесок каналу зеленого кольору в сприйману яскравість пікселя складає близько 60%, відповідний внесок червоного кольору - близько 30%, і лише близько 10 % - це внесок каналу синього кольору. Таким чином, будемо вважати, що чутливість ЗСЛ до зміни синього кольору приблизно в 6 разів нижче ніж до зеленого і в 3 рази ніж до червоного.

Високорівневі властивості ЗСЛ поки що рідко враховуються при побудові стеганоалгоритмов. Вони відрізняються від низькорівневих тим, що виявляються "повторно" – обробивши первинну інформацію від ЗСЛ, мозок видає команди на "підстроювання" зорової системи під зображення.

Перерахуємо основні з цих властивостей:

- чутливість до контрасту – висококонтрастні ділянки зображення і перепади яскравості обертають на себе більше уваги;
- чутливість до розміру – великі ділянки зображення "помітніші" в порівнянні з меншими за розміром, причому існує поріг насиченості, коли подальше збільшення розміру не грає ролі;
- чутливість до форми – довгі і тонкі об'єкти викликають більше уваги, чим закруглені і однорідні;
- чутливість до кольору – деякі кольори (наприклад, червоний) "помітніші", ніж інші; цей ефект посилюється, якщо фон заднього плану відрізняється від кольору фігур на ньому;
- чутливість до місця розміщення – людина схильна в першу чергу розглядати центр зображення; також уважніше розглядаються фігури переднього плану, чим заднього;
- чутливість до зовнішніх подразників - рух очей спостерігачів залежить від конкретної обстановки, від отриманих ними перед переглядом або під час його інструкцій, додаткової інформації.

Високорівневі властивості ЗСЛ часто використовують у рекламі, в політиці, в різних шоу, тощо, бо це гарний спосіб впливати на свідомість людини за допомогою різних особливостей розумової діяльності, які зумовлені загальним рівнем культури людини та рівнем її освіти, національності, професійної приналежності, наявними традиціями, менталітетом та інше.

Цифровий формат кодування зображень Bitmap Picture. Формат Windows BMP є одним з вбудованих форматів зображень в операційних системах Microsoft Windows. Він підтримує зображення з 1, 4, 8, 16, 24 і 32 бітами на піксел, хоча файли BMP з 16 і 32 бітами на піксел використовуються рідко. Для зображень з 4 і 8 бітами на піксел формат BMP підтримує також просте RLE-стиснення (кодування довжин серій). Проте, стиснення в BMP-форматі має ефект тільки за наявності в зображенні

великих областей однакового кольору, що обмежує цінність вбудованого алгоритму стиснення.

Розглянемо структуру BMP-файлу (тут і далі будемо розглядати тільки файли BMP-24 із кодуванням 24 бітів на піксель). Він містить точкове (растрове) зображення і складається із трьох основних розділів: заголовка файлу, заголовка растра та растрових даних.

Заголовок файлу містить інформацію про файл (його тип, обсяг і т.п.). У заголовок растра винесена інформація про ширину й висоту зображення, кількість бітів на піксель, розмір растра, глибина кольору, коефіцієнт компресії і т.д. В першу чергу нас цікавлять растрові дані – інформація про колір кожного пікселя зображення.

Колір пікселя визначається об'єднанням трьох основних колірних складових: червоної, зеленої та синьої (скорочено, *RGB*). Кожної з них відповідає своє значення інтенсивності, що може змінюватися від 0 до 255. Отже, за кожний з колірних каналів відповідає 8 бітів (1 байт), а глибина кольору зображення в цілому – 24 біта (3 байти).

Для обробки зображення необхідно перевести колірні характеристики кожного його пікселя в числову матрицю, що представляє з себе масив, який складається з трьох підмасивів розкладу кольорового зображення на компоненти R, G і B. При цьому три колірні компоненти розміщуються один за одним в загальному масиві C (див. рис. 1.1).

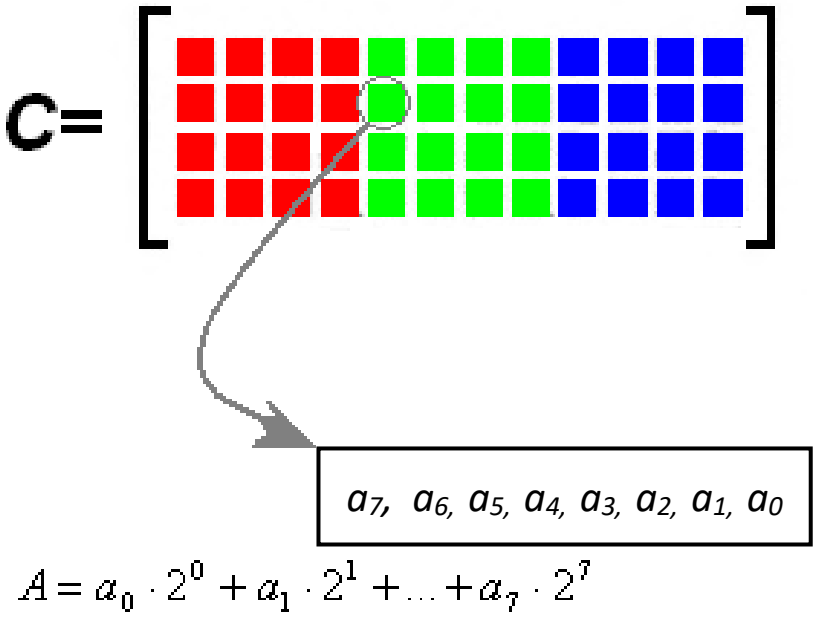


Рис. 1.1. Уявлення компонент кольоровості *R*, *G* і *B* у вигляді підмасивів масиву C

Таким чином, одна точка зображення у форматі BMP-24 кодується трьома байтами, кожний з яких відповідає за інтенсивність одного з трьох складових кольорів (див. рис. 1.2).

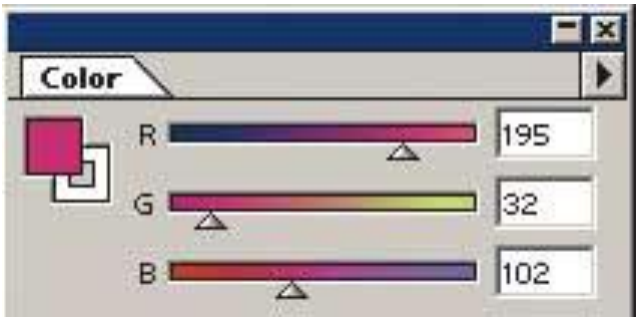


Рис. 1.2. Приклад кодування растрових даних зображення

В результаті змішення кольорів з червоного (R), зеленого (G) і синього (B) каналів піксел одержує потрібний відтінок.

Щоб наочніше побачити принцип дії методу LSB, розпишемо кожний з трьох байтів в бітовий вигляд (рис. 1.3).

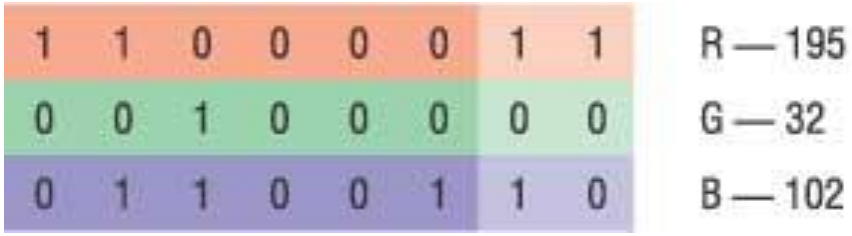


Рис. 1.3. Приклад опису трьох байтів бітовому вигляді

Молодші розряди (на рис. 1.3 вони розташовані справа) у меншій мірі впливають на підсумкове зображення, ніж старші. З цього можна зробити висновок, що заміна одного або декілька молодших бітів, на інші біти тільки трохи спотворить відтінок пікселя і спостерігач не помітить зміни.

Метод заміни найменш значущого біта (НЗБ, LSB – Least Significant Bit) найпоширеніший серед методів заміни в просторовій області зображення. Цей метод використовує першу низькорівневу властивість ЗСЛ – слабку чутливість до незначних змін яскравості зображення.

Молодший значущий біт при кодуванні яскравості пікселя несе в собі найменше інформації. Тобто людина в більшості випадків не здатна помітити змін у цьому біті. Фактично, НЗБ – це шум, тому його можна використовувати для вбудовування інформації шляхом заміни менш значущих бітів пикселей зображення бітами секретного повідомлення. При цьому, для зображення в градаціях сірого (кожний пиксель зображення кодується одним байтом) обсяг вбудованих даних може становити 1/8 від загального обсягу контейнера. Наприклад, у зображення розміром 512×512 пікселів у форматі bmp24 можна вмонтувати ~32 кбайт інформації. Якщо ж модифікувати два молодших біти (що також практично непомітно), то пропускну здатність такого стегаканалу можна збільшити ще вдвічі.

Популярність даного методу обумовлена його простотою й тим, що він дозволяє приховувати у відносно невеликих файлах досить великі обсяги

інформації (пропускна спроможністю створюваного скритного каналу зв'язку становить при цьому від 12,5 до 30%). Метод найчастіше працює з растровими зображеннями, представленими у форматі без компресії (наприклад, GIF і BMP)

Метод НЗБ має дуже низьку стеганографическую стійкість до атак пасивного і активного порушників. Основний його недолік – висока чутливість до найменших спотворень контейнера. Для ослаблення цієї чутливості часто додатково застосовують завадостійке кодування.

Так, наприклад, якщо інформаційне повідомлення вбудовується в зображення в форматі bmp24 із використанням тільки LSB, маємо (у відсотках до максимального рівня яскравості):

$$\Delta = \frac{2^0}{2^8}100\,\% = \frac{1}{256}100\,\% < 0{,}4\%\,.$$

При вбудовуванні в перші три найменш значущі біти маємо:

$$\Delta = \frac{2^0 + 2^1 + 2^2}{2^8}100\,\% = \frac{7}{256}100\,\% < 3\%\,.$$

Таким чином, використання перших трьох біт приводить до внесення похибок, що лежать нижче порогу ЗСЛ до змін яскравості (3%).

Слід вказати на переваги та недоліки методу LSB.

Переваги:

1) Висока пропускна спроможність створюваного стегаканалу. Фактично мова йде щонайменше про 1/8 об'єму контейнеру. Збільшення кількості бітів, що використовуються при вбудовуванні, веде до збільшення пропускної спроможності стегаканалу (до 2/8=1/4, або навіть до 3/8).

2) Простота практичної реалізації та обумовлена цим велика швидкість перетворення як при вбудовуванні, так і при видобуванні бітів інформації.

Недоліки:

1) Відсутність секретного ключа обумовлює дуже малу стійкість до атак зловмисників. Фактично, супротивник може зчитувати інформаційні повідомлення з LSB без якихось завад.

2) Дуже низька стійкість до детектування повідомлень супротивником. Наприклад, найпростіший статистичний тест LSB заповненого контейнеру дає змогу супротивнику встановити факт вбудовування інформації.

3) Дуже низька стійкість до геометричних (повороти, масштабування, зміна пропорцій) атак та атак стиснення.

4) Псевдовипадкові зміни LSB контейнеру або їх обнуління гарантовано руйнують вбудоване повідомлення.

Подальшим розвитком методу LSB, є методи псевдовипадкового інтервалу та псевдовипадкового переставлення.

Метод псевдовипадкового інтервалу. У розглянутому вище простому прикладі виконується заміна найменш значущого біта всіх послідовно

розміщених пікселів зображення, що спрощує атаки зловмисникам та знижує ефективність стегаканалу. Інший підхід - метод випадкового інтервалу, полягає у випадковому розподілі бітів секретного повідомлення по контейнеру, внаслідок чого відстань між двома вбудованими бітами визначається псевдовипадково. Ця методика особливо ефективна у разі, коли бітова довжина секретного повідомлення істотно менша за кількість пікселів зображення.

Розглянемо простий випадок цього методу, коли інтервал між двома послідовними вбудовуваннями бітів повідомлення задається значенням секретного ключа $K = \{K_1, K_2, \ldots, K_n\}$.

Нехай M – повідомлення, яке необхідне приховати. У якості контейнеру C використаємо підмасив B синього колірного компоненту зображення (див. рис. 1.4).

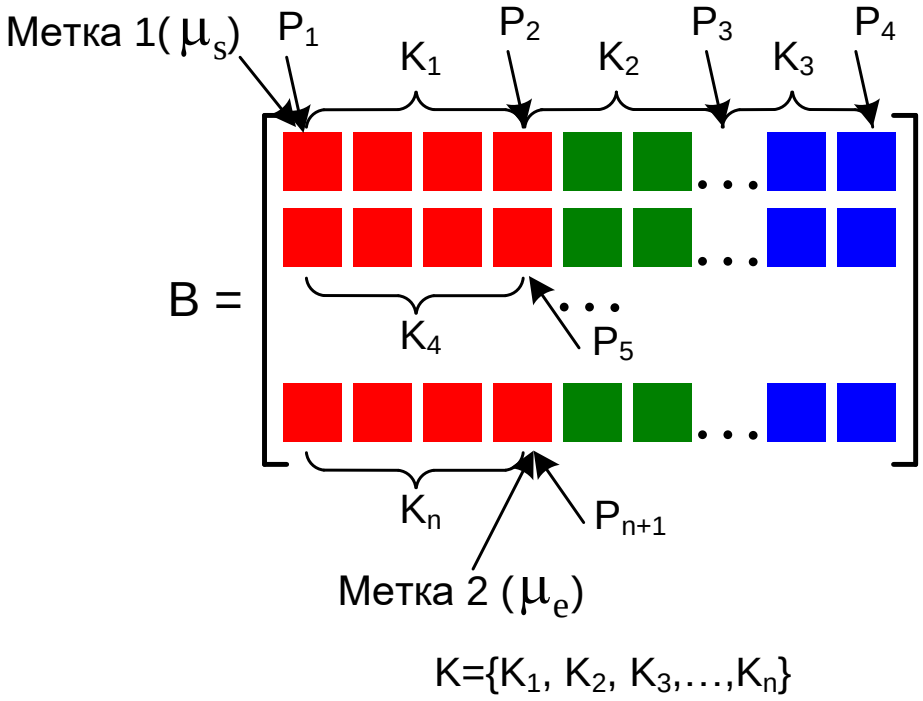


Рис. 1.4. Приклад використання підмасиву B синього колірного компоненту зображення у якості контейнеру

Визначимо мітки, які встановлюватимуть межі корисного повідомлення в контейнері. На відміну від попереднього методу, стартова мітка визначатиме порядковий номер елементу контейнера, починаючи з якого в останній заноситимуться дані. Нехай μ_S буде початковою міткою, а μ_e – мітка яка сигналізує про завершення корисної частини серед добутих символів.

Приймемо, що при внесенні бітів повідомлення в контейнер із змінним кроком, величина останнього обумовлена значенням секретного ключа K_i, i=1…n, i – номер вбудованого біту.

Обмежуючу мітку μ_e додамо до тексту повідомлення, яке підлягає прихованню.

Кожен символ повідомлення переводимо в двійковий формат, кожен розряд якого записується замість наймолодшого біта числа P_i, відповідного значенню інтенсивності синього кольору певного пікселя. При цьому елементи масиву контейнеру перебираються не послідовно, а із змінним кроком, який задається значенням секретного ключа.

Стартовий елемент задається міткою μ_s. Після проведеної зміни, модифіковане двійкове число P_i переводиться у формат десяткового і записується у відповідну позицію масиву B^*, який на початку був прийнятий рівним масиву B.

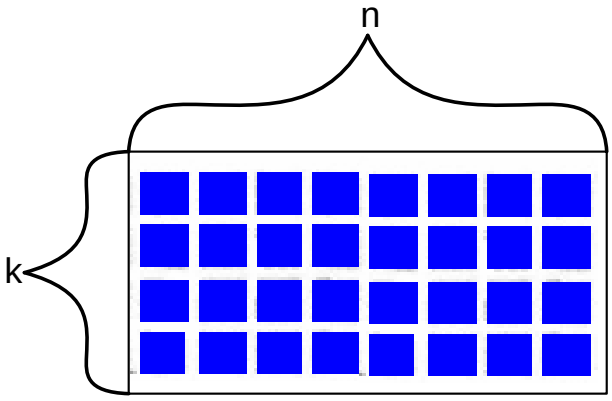
Результуюче кольорове зображення визначатиметься масивом об'єднання колірних масивів R, G, B^*.

При отриманні прихованого повідомлення повинні бути відомі параметри μ^*_s, μ^*_e, K^* і, зрозуміло, масив B^*, який, як передбачається, містить приховані дані.

Отримання повідомлення з масиву B^* виконується в зворотному, по відношенню до операції вбудовування порядку, після чого одержуємо вектор інформаційних біт даних. З одержаного вектора шляхом порівняння з мітками μ_s та μ_e виділеного фрагмента вилучається корисне повідомлення.

Найпростіше реалізувати розглянутий метод можна у такий спосіб. Нехай, наприклад, $M=\{m_0,m_1,m_2,\ldots m_n\}$ – інформаційне повідомлення із n біт, тобто $m_i=\begin{cases}0\\1\end{cases}$.

Як контейнер будемо використовувати масив даних розміром n стовпців та k строк. (одного з кольорів):



У якості секретного ключа будемо використовувати вектор $K=\{k_0,k_1,\ldots k_{n-1}\}$, де K_i – деяке число, яке лежить в діапазоні $[0\ldots k]$.

Таким чином вбудовування біту m_0 будемо виконувати у стовпець під номером N=0, біту m_1 – у стовпець під номером 1 і так далі. Біт m_0

записується у LSB байту контейнеру, у рядку k_0, m_1 – у рядку k_1 і т.і. Схема вбудовування зображення представлена на рис. 1.5.

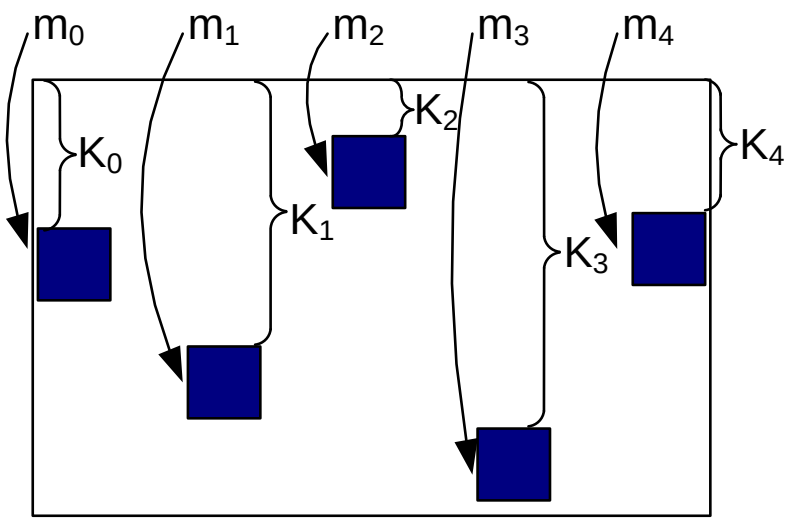


Рис. 1.5. Схема вбудовування зображення

Таким чином, один біт інформації записується у один стовпець контейнеру у строку, яка задається секретним ключем.

Розглянутий метод має наступні *переваги:*

1) Введення секретного ключа, який задає правило вбудовування інформації, значно підвищує стійкість методу до детектування та вставки інформаційних даних. Звісно, статистика інформаційних біт даних дещо «просочується» у статистику LSB контейнеру, але рознесення їх по контейнеру через псевдовипадковий інтервал значно ускладнює детектування повідомлення.

2) Простота реалізації методу та обумовлена цим велика швидкість перетворення як при вбудовуванні, так і при видобуванні бітів інформації.

Недоліки:

1) Значно зменшується пропускна спроможність стегаканалу. В середньому зменшення пропускної спроможності у N разів, де

$$N = \frac{1}{n}\sum_{i=1}^{n} K_i$$ – середнє значення псевдо випадкового інтервалу.

2) Дуже низька стійкість до геометричних (повороти, масштабування, зміна пропорцій) атак та атак стиснення.

3) Псевдовипадкові зміни контейнеру руйнують вбудоване повідомлення.

Для подальшого підвищення стійкості до негативних дій зловмисників вбудування інформаційного повідомлення доцільно попередньо шифрувати. Тому наступний метод, метод псевдо випадкового переставлення, саме і

побудований на основі використання найпростішого перестановочного шифру.

Метод псевдовипадкового переставлення. Недоліком методу псевдовипадкового інтервалу є те, що біти повідомлення в контейнері розміщені в тій же послідовності, що і в самому повідомленні, і лише інтервал між ними змінюється псевдовипадково. Тому для контейнерів фіксованого розміру доцільнішим є використання методу псевдовипадкового переставлення. Сутність його полягає у використанні переставного шифру для попередньої обробки інформаційних біт даних.

У **переставному шифрі** міняється не відкритий текст, який полягає шифруванню, а порядок символів. Наприклад у **простому стовбцьовому переставному шифрі** відкритий текст пишеться горизонтально на розграфленому листі паперу фіксованої ширини, а шифротекст прочитується по вертикалі. Розшифруванням є запис шифротексту вертикально на листі розграфленого паперу фіксованої ширини і потім зчитування відкритого тексту горизонтальне (див. табл. 1.1.)

Таблиця 1.1.

Простій стовбцьовий переставний шифр

Відкритий текст:
ЦЕПРИКЛАДПРОСТОГОСТОВПЬОВОГОПЕРЕСТАВНОГОШИФРУАААА

Ц	Е	П	Р	И	К	Л
А	Д	П	Р	О	С	Т
О	Г	О	С	Т	О	В
Б	Ц	Ь	О	В	О	Г
О	П	Е	Р	Е	С	Т
А	В	Н	О	Г	О	Ш
И	Ф	Р	У	А	А	А

Шифрограма:
ЦАОБОАИЕДГЦПВФППОЬЕНРРРСОРОУИОТВЕГАКСООСОАЛТВГТША

Оскільки символи шифротексту ті ж, що і у відкритому тексті, частотний аналіз шифротексту покаже, що кожна буква зустрічається приблизно з тією ж частотою, що і звичайне. Це дасть криптоаналітику можливість застосувати різні методи, визначаючи правильний порядок символів для отримання відкритого тексту. Застосування до шифротексту другого переставного фільтру значно підвищить безпеку. Існують і ще складніші переставні фільтри, але комп'ютери можуть розкрити майже все з них.

Німецький шифр ADFCVX, використаний в ході Першої світової війни, був переставним фільтром у поєднанні з простій підстановкою. Цей для свого часу дуже складний алгоритм був розкритий Жоржем Пенвеном (Georges Painvin), французьким криптоаналітиком.

Хоча багато сучасних алгоритмів використовують перестановку, з цим пов'язана проблема використання великого об'єму пам'яті, а також іноді потрібна робота з повідомленнями певного розміру.

Формалізуємо переставний шифр, стосовно використання його для стеганографічного перетворення цифрових повідомлень.

Нехай $m = \{m_0, m_1, \ldots m_{N-1}\}$ - інформаційне повідомлення із N бітів. Розіб'ємо його на блоки по n бітів, отримаємо:

$$\mathrm{m} = \left\{ \mathrm{M}_0, \mathrm{M}_1, \ldots \mathrm{M}_{\frac{\mathrm{N}}{\mathrm{n}}-1} \right\}, \text{ де } \mathrm{M_i} = \{\mathrm{m_{n \cdot i+0}}, \mathrm{m_{n \cdot i+1}}, \mathrm{m_{n \cdot i+2}}, \ldots \mathrm{m_{n \cdot i+n-1}}\}.$$

$$\text{Тобто } \mathrm{m} = \left\{ \underbrace{\mathrm{m_0}, \mathrm{m_1}, \mathrm{m_2}, \ldots \mathrm{m_{n-1}}}_{\mathrm{M_0}}, \underbrace{\mathrm{m_n}, \mathrm{m_{n+1}}}_{\mathrm{M_1}}, \ldots \underbrace{\mathrm{m_{N-1}}}_{\mathrm{M_{N/n-1}}} \right\}.$$

У якості ключа будемо використовувати переставну матрицю P розміром $\mathrm{n} \times \mathrm{n}$ двійкових елементів, причому у кожному стовпці та у кожному рядку матриці є тільки одна «1», решта заповнюється «0». Переставна матриця P задає правило псевдовипадкового переставлення:

$$\begin{aligned} \mathrm{M_i^*} &= \mathrm{M_i} \cdot \mathrm{P} = \{\mathrm{m_{n \cdot i+0}}, \mathrm{m_{n \cdot i+1}}, \mathrm{m_{n \cdot i+2}}, \ldots \mathrm{m_{n \cdot i+n-1}}\} \times [\mathrm{P}] = \\ &= \{\mathrm{m^*_{n \cdot i+0}}, \mathrm{m^*_{n \cdot i+1}}, \mathrm{m^*_{n \cdot i+2}}, \ldots \mathrm{m^*_{n \cdot i+n-1}}\} \end{aligned}.$$

Для фіксованого значення n існують n! різних переставних матриць, кожна з яких задає своє правило переставлення. Після переставлення сформуємо масив:

$$\mathrm{m^*} = \{\mathrm{m^*_0}, \mathrm{m^*_1}, \ldots \mathrm{m^*_{N-1}}\}.$$

Для виконання зворотного перетворення треба обчислити наступне:

$$\mathrm{M_i} = \mathrm{M_i^*} \cdot \mathrm{P^{-1}},$$

де $\mathrm{P^{-1}}$ – матриця, зворотна матриці P, тобто $\mathrm{P^{-1}} \cdot \mathrm{P} = \mathrm{I}$.

Але для переставної матриці можна записати $\mathrm{P^{-1}} = \mathrm{P^T}$, отже:

$$\mathrm{M_i} = \mathrm{M_i^*} \cdot \mathrm{P^T}.$$

Наведемо приклад:

$$m = \{101011010110110\}$$

$$n=5,\ P = \begin{vmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{vmatrix}.$$

Маємо таке:

$$M_0 = \{10101\} \qquad M^*_0 = M_0 \cdot P = \{01011\}$$
$$M_1 = \{10101\} \qquad M^*_1 = M_1 \cdot P = \{01011\}$$
$$M_2 = \{10101\} \qquad M^*_2 = M_2 \cdot P = \{01101\}$$

Отже $m^* = \{010110101101101\}$

Для зворотного перетворення виконаємо наступне:

$$M_0 = M^*_0 \cdot P^T = \{01011\} \cdot \begin{vmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{vmatrix} = \{10101\},$$

$$M_1 = M^*_1 \cdot P^T = \{01011\} \cdot \begin{vmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{vmatrix} = \{10101\},$$

$$M_2 = M^*_2 \cdot P^T = \{01101\} \cdot \begin{vmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{vmatrix} = \{10110\}.$$

Отже маємо $m = \{101011010110110\}$.

Розглянемо тепер використання переставного шифру в методі псевдовипадкового переставлення. Загальна схема вбудовування бітів даних зображена на рис. 1.6.

Сутність методу псевдовипадкового переставлення полягає у попередній обробці інформаційних даних переставним шифруванням (за розглянутою вище схемою) та вбудовуванні отриманих бітів даних у біти контейнера за допомогою методу LSB або псевдо випадкового інтервалу.

Вочевидь, що додаткове шифрування інформаційних бітів контейнеру поліпшує властивості стегасистеми, отже маємо наступні *переваги:*

1) Підвищення стійкості до детектування повідомлення зловмисниками. Використання шифру знижує статистику вбудованих бітів даних, отже «зашумляє» відповідні LSB. Виявляти таки повідомлення дуже складно.

2) Пропускна спроможність не змінюється, тобто при вбудовуванні во всі LSB контейнера пропускна спроможність дуже висока (1/8, 2/8-1/4, або навіть 3/8).

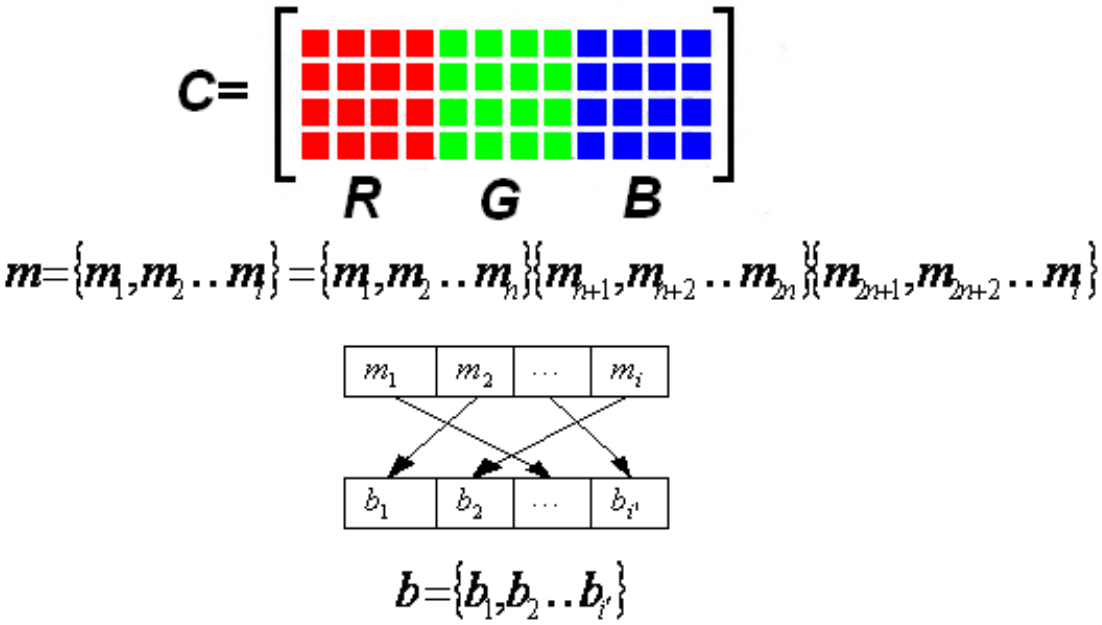


Рис. 1.6. Приклад здійснення псевдовипадкового переставлення

Але нажаль, метод псевдовипадкового переставлення має такі *недоліки:*

1) Дуже низька стійкість до геометричних (повороти, масштабування, зміна пропорцій) атак та атак стиснення.

2) Псевдовипадкові зміни LSB контейнеру або їх обнуління руйнують вбудоване повідомлення.

4. Запитання для поточного контролю підготованості студентів до виконання лабораторної роботи № 1

1. Математична модель та структурна схема криптографічної (секретної) та стеганографічної системи захисту інформації.
2. Основні галузі практичного використання стеганографічних методів захисту інформації. Історичні приклади використання стеганографічних систем захисту інформації.
3. Класифікація стеганографічних систем. Закриті, напівзакриті, відкриті стеганографічні системи. Хиткі, напівхиткі, робастні стеганографічні системи.
4. Класифікація та види контейнерів за різними ознаками. Приклади використання різних контейнерів для організації прихованої передачі інформації.
5. Низькорівневі властивості зорової системи людини. Практичні приклади використання низькорівневих властивостей зорової системи людини в стеганографії.
6. Високорівневі властивості зорової системи людини. Практичні приклади використання високорівневих властивостей зорової системи людини в стеганографії.
7. Сучасні формати нерухомих зображень. Структура файлу нерухомого зображення у форматі bmp24. Растрові дані зображення.
8. Метод вбудовування інформації в нерухомі зображення на основі модифікації найменш значущого біта (метод LSB). Переваги та недоліки.
9. Найпростіші симетричні шифри. Простий перестановочний шифр.
10. Метод вбудовування інформації з використанням псевдовипадкової перестановки (метод ПВП). Переваги та недоліки.
11. Метод вбудовування інформації з використанням псевдовипадкового інтервалу (метод ПВІ). Переваги та недоліки.

5. Інструкція до виконання лабораторної роботи №1

Завдання 1. Реалізація алгоритмів вбудовування та вилучення повідомлень в просторовій області нерухомих зображень методом LSB

1.1. Завантажуємо вихідні дані: контейнер - нерухоме зображення (в форматі *.bmp24); інформаційне повідомлення - текстовий документ (у форматі *.txt). Для цього в середовищі MathCAD виконуємо наступні дії.

1.1.1. Виконуємо команду читання растрових даних нерухомого зображення з заданого файлу в вигляді двовимірного масиву цілих чисел:

«C:=READRGB(“[ім’я файлу].bmp”)».

Елементи масиву $C_{i,j}$ знаходяться в інтервалі [0...255] і задають значення яскравості одного з трьох кольорів (червоного, зеленого або синього) зображення. Координати елементів масиву задають розташування окремих пікселів зображення. Масив C складається з трьох підмасивів рівного розміру (для зберігання значень яскравості відповідних кольорів).

Наприклад, нехай в якості контейнера виступає нерухоме зображення, яке зберігається в файлі з ім'ям «1.bmp». Тоді, після виконання команди читання растрових даних

«C:=READRGB(“1.bmp”)»

маємо:

C =

	0	1	2	3	4	5	6	7	8	9
0	86	79	72	72	72	69	71	74	77	85
1	110	97	90	86	78	71	70	70	77	79
2	132	120	112	105	96	88	81	83	80	80
3	122	116	105	104	103	102	101	99	93	90
4	131	122	117	118	118	116	105	107	105	110
5	147	147	148	148	150	153	145	135	127	120
6	169	164	167	170	173	175	164	155	152	144
7	189	195	193	189	183	172	173	173	177	168
8	191	192	194	199	194	187	182	182	183	182
9	186	188	194	198	192	187	187	180	175	177
10	195	196	199	200	201	190	192	186	174	167
11	185	189	202	203	203	199	203	199	200	199
12	192	196	198	199	204	202	206	199	194	197
13	177	185	187	186	180	178	179	177	175	180
14	173	176	174	166	165	165	163	161	158	162
15	160	162	158	153	156	158	157	156	153	…

.

Для графічного відображення завантажених даних виконуємо дії: «Вставити», «Зображення». В полі введення джерела зображень вносимо ім'я файлу або ім'я змінної, в якій зберігається масив даних.

Після виконання команд графічного відображення контейнера для розглянутого прикладу маємо такі зображення:

"1.bmp"

C

Графічне відображення масиву растрових даних C складається з трьох фрагментів зображення, кожен фрагмент відповідає відображенню одного з колірних каналів (червоного, зеленого або синього) в градаціях сірого кольору.

Виконуємо команду читання даних з каналу червоного кольору растрових даних нерухомого зображення з заданого файлу в вигляді масиву цілих чисел:

«R:=READ_RED(“[ім’я файлу].bmp”)».

Елементи масиву $R_{i,j}$ також знаходяться в інтервалі [0...255] і задають значення яскравості червоного кольору. Для графічного відображення завантажених даних виконуємо дії: «Вставити», «Зображення». В полі введення джерела зображень вносимо ім'я змінної R, в якій зберігається масив даних каналу червоного кольору.

Для розглянутого прикладу виконуємо команду

«R:=READ_RED(“1.bmp”)»,

після чого отримуємо:

R =

	0	1	2	3	4	5	6	7	8	9
0	86	79	72	72	72	69	71	74	77	85
1	110	97	90	86	78	71	70	70	77	79
2	132	120	112	105	96	88	81	83	80	80
3	122	116	105	104	103	102	101	99	93	90
4	131	122	117	118	118	116	105	107	105	110
5	147	147	148	148	150	153	145	135	127	120
6	169	164	167	170	173	175	164	155	152	144
7	189	195	193	189	183	172	173	173	177	168
8	191	192	194	199	194	187	182	182	183	182
9	186	188	194	198	192	187	187	180	175	177
10	195	196	199	200	201	190	192	186	174	167
11	185	189	202	203	203	199	203	199	200	199
12	192	196	198	199	204	202	206	199	194	197
13	177	185	187	186	180	178	179	177	175	180
14	173	176	174	166	165	165	163	161	158	162
15	160	162	158	153	156	158	157	156	153	…

R

Виконуємо аналогічні команди читання даних з каналів зеленого і синього кольору растрових даних нерухомого зображення у вигляді масивів цілих чисел:

«G:=READ_ GREEN (“[ім’я файлу].bmp”)»,
«G:=READ_BLUE(“[ім’я файлу].bmp”)».

Для розглянутого прикладу виконуємо команди

«B:=READ_ GREEN (“1.bmp”)»,
«B:=READ_BLUE(“1.bmp”)»,

після чого отримуємо:

G

B

G =

	0	1	2	3	4	5	6	7	8	9
0	91	83	79	77	75	73	73	75	83	90
1	112	101	93	89	82	76	76	76	81	82
2	134	122	118	107	103	90	88	90	86	85
3	124	118	109	107	109	104	103	101	98	94
4	133	125	118	124	120	119	110	110	107	115
5	147	146	151	145	151	153	145	135	132	120
6	168	163	167	170	172	174	163	153	152	149
7	187	195	189	185	182	171	172	172	175	169
8	190	192	191	193	192	184	182	180	179	177
9	185	187	196	195	193	188	185	180	175	177
10	194	196	194	199	195	189	189	182	173	165
11	184	190	199	200	197	201	201	197	196	197
12	192	195	198	199	201	199	202	196	191	194
13	175	185	186	185	181	178	180	177	178	181
14	171	175	173	169	165	165	164	159	160	163
15	160	164	160	159	159	161	160	160	158	…

B =

	0	1	2	3	4	5	6	7	8	9
0	135	128	123	122	119	124	120	124	124	130
1	155	142	141	134	133	122	122	124	120	134
2	174	159	151	152	141	136	133	135	126	130
3	162	160	151	145	147	147	146	144	139	142
4	172	161	159	164	158	160	150	150	153	149
5	184	181	190	184	183	191	176	166	166	159
6	198	197	200	203	206	207	195	189	188	180
7	225	222	226	222	218	206	204	208	213	201
8	223	226	222	224	219	215	210	219	213	212
9	220	221	230	229	224	221	224	214	209	210
10	224	228	231	225	229	221	224	215	209	207
11	221	217	227	231	232	233	222	229	229	222
12	222	224	228	230	231	231	229	225	223	226
13	215	211	218	217	211	208	208	209	203	208
14	209	207	204	198	197	197	192	199	190	194
15	200	196	192	190	189	191	193	189	190	…

Завантажені масиви червоного, зеленого і синього кольору (масиви R, G, і B) є подмасивами масиву растрових даних C. Аналогічно, графічна інтерпретація масивів R, G і B (у вигляді зображень в градаціях сірого кольору) об'єднана в графічному представленні масиву C. Повноколірне представлення зображення отримаємо в такий спосіб:

R, G, B

При порушенні порядку проходження масивів растрових даних, що характеризують інформацію про канали кольоровості, зображення буде спотворене, наприклад:

G, B, R

B, R, G

1.1.2. Виконуємо команду читання інформаційних даних текстового документа з заданого файлу у вигляді одновимірного масиву цілих чисел

«M:=READBIN(“[ім’я файлу].txt”, byte)».

Елементи масиву M_i знаходяться в інтервалі [0...255] і задають значення символів інформаційного повідомлення в кодуванні ASCII (кодування ASCII наведене в додатку). Координати елементів масиву M задають розташування окремих символів інформаційного повідомлення.

Наприклад, нехай в якості інформаційних даних виступає текстовий документ, який зберігається в файлі з ім'ям «1.txt». Тоді, після виконання команди читання символів інформаційного повідомлення в кодуванні ASCII

«M:=READBIN(“1.txt”, byte)»

отримуємо:

M =

	0
0	200
1	237
2	242
3	229
4	240
5	229
6	241
7	32
8	...

Значення нульового елементу масиву M дорівнює $M_0=200$, що відповідає символу «И» в кодуванні ASCII. Наступний елемент масиву $M_1=237$ відповідає символу «н» в кодуванні ASCII. Набір перших семи елементів масиву інформаційних даних

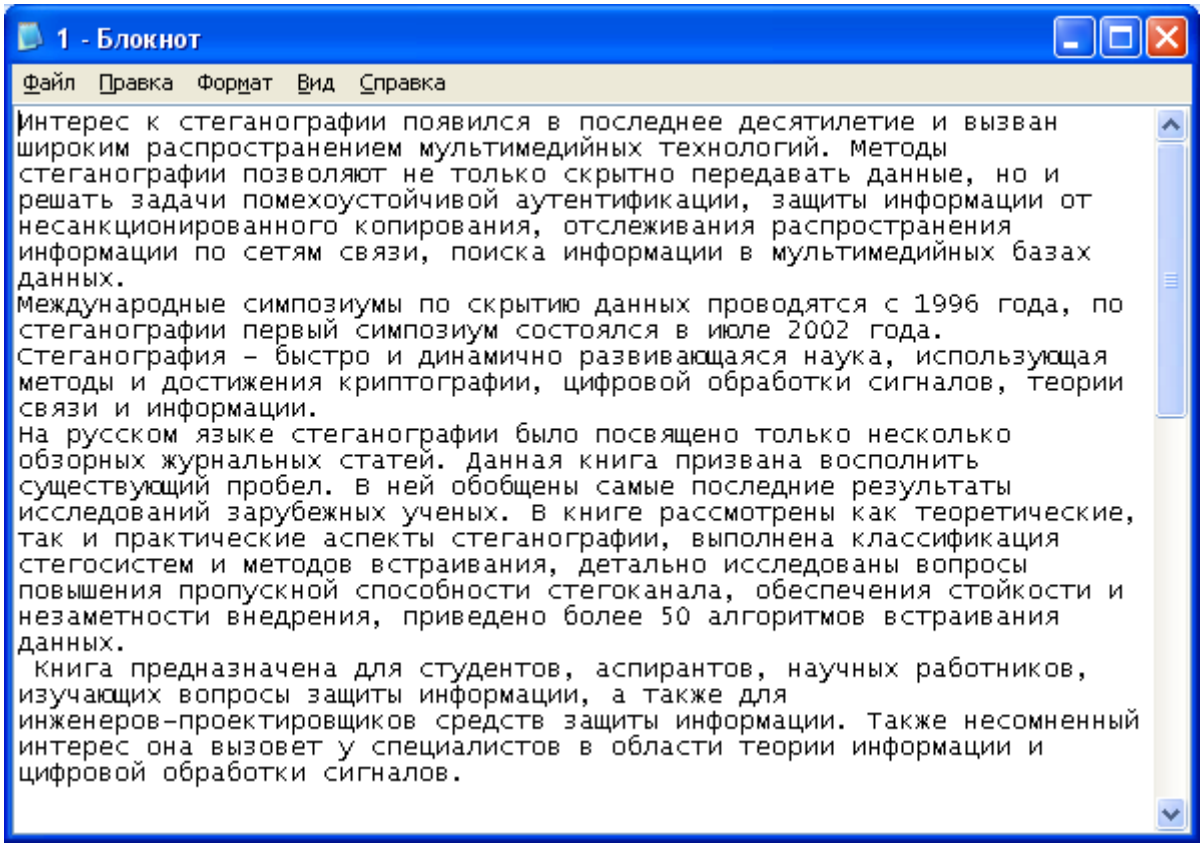
{200, 237, 242, 229, 240, 229, 241}

відповідає набору символів повідомлення

{И, н, т, е, р, е, с}

в кодуванні ASCII.

Для розглянутого прикладу в якості інформаційного повідомлення обрано перший абзац з анотації книги «Цифровая стеганография» авторів В.Г. Грибунина, И.Н. Окова, И.В. Туринцева.



1.2. Перетворимо масив інформаційних даних

Інформаційні дані в масиві M представлені у вигляді набору цілих чисел з інтервалу [0...255] і задають значення символів інформаційного повідомлення в кодуванні ASCII. Для розглянутих стеганографічних методів вбудовування інформації в нерухомі зображення здійснюється побітово.

Тобто інформаційні дані з масиву M слід попередньо підготувати, перетворивши їх в бітовий масив. Для цього в середовищі MathCAD використовуємо такі функції.

1.2.1. Функція перетворення вектора-стовпця з восьми біт в десятковий код:

$$B\_D(x) := \sum_{i=0}^{7} \left(x_i \cdot 2^i \right)$$

Аргументом x функції B_D (x) є двійковий вектор-стовпець з восьми біт:

$$x := \left(x_0 \;\; x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_5 \;\; x_6 \;\; x_7 \right)^T$$

Значенням функції B_D (x) є ціле число в десятковому коді:

$$B\_D(x) := x_0 \cdot 2^0 + x_1 \cdot 2^1 + x_2 \cdot 2^2 + x_3 \cdot 2^3 + x_4 \cdot 2^4 + x_5 \cdot 2^5 + x_6 \cdot 2^6 + x_7 \cdot 2^7$$

Наприклад, нехай аргумент x функції B_D (x) заданий наступним чином:

$$x := (1 \;\; 1 \;\; 0 \;\; 0 \;\; 1 \;\; 0 \;\; 0 \;\; 1)^T$$

Тоді значення функції B_D(x) дорівнює

$$B\_D(x) = 147$$

1.2.2. Функція перетворення цілого числа в десятковому коді в двійковий вектор-стовпець з восьми біт:

$$D\_B(x) := \left| \begin{array}{l} \text{for } i \in 0..7 \\ \quad \left| \begin{array}{l} V_i \leftarrow \text{mod}(x, 2) \\ x \leftarrow \text{floor}\left(\frac{x}{2} \right) \end{array} \right. \\ V \end{array} \right.$$

Аргументом x функції D_B (x) є ціле число в десятковому коді, значенням функції є двійковий вектор-стовпець з восьми біт.

Алгоритм обчислення значення функції D_B (x) є наступним. На кожній ітерації (для кожного значення циклової змінної i) обчислюються наступні значення:

$V_i \leftarrow \text{mod}(x, 2)$ - приведення десяткового числа x за модулем 2;

$x \leftarrow \text{floor}\left(\frac{x}{2} \right)$ - визначення цілої частини від ділення цілого числа x на два.

Тобто після кожної ітерації число x зменшується в два рази, а значення i-го біта, яке повертається функцією дійкового вектора-стовпця, прирівнюється до результату приведення поточного значення x за модулем 2.

Наприклад, нехай аргумент функції D_B (x) дорівнює 27. Тоді значення функції D_B (x) обчислюється таким чином:

i=0: V_0=mod(27,2)=1, x=floor(27/2)=13;
i=1: V_1=mod(13,2)=1, x=floor(13/2)=6;
i=2: V_2=mod(6,2)=0, x=floor(6/2)=3;
i=3: V_3=mod(3,2)=1, x=floor(3/2)=1;
i=4: V_4=mod(1,2)=1, x=floor(1/2)=0;
i=5: V_5=mod(0,2)=0, x=floor(0/2)=0;
i=6: V_6=mod(0,2)=0, x=floor(0/2)=0;
i=7: V_7=mod(0,2)=0, x=floor(0/2)=0.

Таким чином, обчислене значення функції D_B(x) дорівнює

$$x = V = (1\ \ 1\ \ 0\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0)^T$$

1.2.3. Використовуючи функцію D_B (x) перетворимо масив M інформаційних цілих чисел в бітовий масив.

Для цього скористаємося наступною процедурою:

```
M_b := | for i ∈ 0..rows(M) − 1
       |     V ← D_B(M_i)
       |     for j ∈ 0..7
       |       M_b_{i·8+j} ← V_j
       | M_b
```

Алгоритм формування бітового масиву інформаційних даних наступний. Для кожного значення циклової змінної i (значення i пробігає по всіх індексах масиву M) виконується перетворення $D\_B(M_i)$ за розглянутим вище правилом. Тобто для всіх елементів масиву M формуються відповідні бітові вектори-стовпці. Всі елементи кожного вектора-стовпця перезаписуються до масиву M_b під відповідним індексом. Таким чином, кожен сформований біт записується в елемент M_b_{i8+j} масиву M_b, де j - циклова змінна, яка пробігає всі індекси поточного (i-ого) вектора-стовпця.

Для прикладу розглянемо вихідний вектор-стовпець M, що містить цілі числа (коди) інформаційного повідомлення. Нехай M - масив цілих чисел з попереднього прикладу.

M =

	0
0	200
1	237
2	242
3	229
4	240
5	229
6	241
7	32
8	234
9	32
10	241
11	242
12	229
13	227
14	224
15	237
16	238
17	227
18	240
19	224
20	…

Виконання процедури перетворення масиву M в бітовий масив M_b відбувається наступним чином:

i=0: V= D_B(200) =$(0\ 0\ 0\ 1\ 0\ 0\ 1\ 1)^T$,

j=0: M_b_0=0,

j=1: M_b_1=0,

j=2: M_b_2=0,

j=3: M_b_3=1,

j=4: M_b_4=0,

j=5: M_b_5=0,

j=6: M_b_6=1,

j=7: M_b_7=1;

i=1: V= D_B(237) =$(1\ 0\ 1\ 1\ 0\ 1\ 1\ 1)^T$,

j=0: M_b_8=0,

j=1: M_b_9=0,

j=2: M_b_{10}=0,

j=3: M_b_{11}=1,

j=4: M_b_{12}=0,

j=5: M_b_{13}=0,

j=6: M_b_{14}=1,

j=7: M_b_{15}=1;

…

1.3. Реалізуємо алгоритм вбудовування даних в просторову область зображень методом LSB. Для цього скористаємося наступною процедурою:

```
S := for j ∈ 0..rows(R) − 1
       for i ∈ 0..cols(R) − 1
         S[j,i] ← R[j,i]
     for l ∈ 0..rows(M_b) − 1
       i ← floor(l / rows(R))
       j ← l − i·rows(R)
       V ← (D_B(R[j,i]))
       V_0 ← M_b_l
       S[j,i] ← B_D(V)
     S
```

Наведена процедура реалізує поелементне вбудовування бітового масиву інформаційних даних M_b в найменш значущі біти послідовних байтів масиву растрових даних каналу червоного кольору, тобто вбудовування здійснюється тільки в LSB масиву R. Перші три рядки процедури виконують перезаписування даних з масиву растрових даних каналу червоного кольору

в новий масив S. Далі, для всіх елементів бітового масиву інформаційних даних M_b обчислюються координати (номери стовпців і рядків) елементів масиву R, в які вони будуть вбудовані. Так, розділивши значення індексу l на кількість рядків у масиві R, отримаємо номер (індекс i) того стовпця, в елементи якого буде вбудований l-ий біт повідомлення. Виконавши обчислення $1 - j * rows(R)$ отримаємо номер (індекс j) того рядки, в елементи якого буде вбудований l-ий біт повідомлення. Бітове подання десяткового числа $R_{j,i}$ записується у змінну V (це перетворення реалізується за допомогою розглянутої вище функції). Поточний біт повідомлення M_b_l заноситься в нульовий (найменш значущий) біт масиву V_0, після чого виконуємо зворотне перетворення масиву V зі зміненим LSB. Отримане десяткове число записуємо в масив S з тими самими індексами i та j. Таким чином, всі біти інформаційного повідомлення з масиву M_b записуються в найменш значущі біти байт каналу червоного кольору зображення. Якщо розмір повідомлення є невеликим, тоді невикористані елементи масиву S заповнюються вихідними даними з масиву R.

Для візуального перегляду результату вбудовування інформаційних даних виведемо вихідний масив растрових даних червоного кольору R та отриманий масив зі зміненими найменш значущими бітами. Для розглянутого прикладу маємо:

R =

	0	1	2	3	4
0	86	79	72	72	72
1	110	97	90	86	78
2	132	120	112	105	96
3	122	116	105	104	103
4	131	122	117	118	118
5	147	147	148	148	150
6	169	164	167	170	173
7	189	195	193	189	183
8	191	192	194	199	194
9	186	188	194	198	192
10	195	196	199	200	201
11	185	189	202	203	203
12	192	196	198	199	204
13	177	185	187	186	180
14	173	176	174	166	165
15	160	162	158	153	…

S =

	0	1	2	3	4
0	86	78	73	72	72
1	110	96	90	86	79
2	132	121	112	105	97
3	123	116	105	105	103
4	130	123	117	119	119
5	146	147	149	149	150
6	169	165	167	170	173
7	189	194	192	189	182
8	191	192	195	199	194
9	186	188	194	198	193
10	195	197	198	201	201
11	185	188	203	203	203
12	192	197	199	199	205
13	177	185	187	186	180
14	173	177	174	166	165
15	161	162	158	152	…

З представлених даних видно, що, наприклад, значення $R_{0,0}$, $R_{1,0}$, $R_{2,2}$ та інші повністю ідентичні відповідним значенням $S_{0,0}$, $S_{1,0}$, $S_{2,2}$. Практично це означає, що найменш значущі біти в цих елементах масиву R співпали з вбудованими інформаційними бітами повідомлення. Навпаки, значення $R_{0,1}$, $R_{1,1}$, $R_{2,1}$ відрізняються на одиницю від відповідних значень масиву S. Це означає зміну найменш значущого біта елемента контейнера в процесі

вбудовування повідомлення. Відзначимо, що внесені спотворення знаходяться нижче порога зорової чутливості людини.

Графічна інтерпретація порожнього і заповненого контейнера (каналу червоного кольору в градаціях сірого) наведена на наступному рисунку, з якого видно, що візуально внесені спотворення є непомітними, це підтверджує висновок про чутливість органів зору людини.

R

S

Отриманий заповнений масив S записуємо в канал червоного кольору контейнера. Для виконання цієї операції з використанням команди

«WRITERGB([ім’я файлу].bmp):=augment(S,G,B)»

під відповідним ім'ям записуємо на фізичний носій сформований контейнер зі зміненими LSB в каналі червоного кольору. Для розглянутого прикладу виконуємо команду

«WRITERGB(Stego.bmp):=augment(S,G,B)»,

виконання якої формує на фізичному носії новий файл з ім'ям «Stego.bmp».

Для графічного відображення вихідного (порожнього) і заповненого контейнера виконаємо вставку відповідних зображень:

"1.bmp"

"Stego.bmp"

Переконуємося в відсутності видимих спотворень.

1.4. Реалізуємо алгоритм вилучення даних з просторової області зображень методом LSB. Для цього в новому вікні середовища MathCAD виконуємо

команди читання растрових даних нерухомого зображення із заданого файлу (файлу заповненого контейнера) у вигляді двовимірного масиву цілих чисел. Для розглянутого прикладу виконуємо команди:

«C:=READRGB(“Stego.bmp”)», «R:=READ_RED(“Stego.bmp”)»,
«G:=READ_ GREEN (“Stego.bmp”)»,«B:=READ_BLUE(“Stego.bmp”)».

Отримуємо такий результат:

R =

	0	1	2	3	4	5
0	86	78	73	72	72	69
1	110	96	90	86	79	70
2	132	121	112	105	97	88
3	123	116	105	105	103	102
4	130	123	117	119	119	117
5	146	147	149	149	150	152
6	169	165	167	170	173	174
7	189	194	192	189	182	...

R

Далі, скориставшись розглянутою функцією D_B(x), витягуємо (отримуємо) найменш значущі біти з масиву даних каналу червоного кольору. Для цього використовуємо наступну процедуру:

```
M_b1 := | for i ∈ 0..cols(R) − 1
        |   for j ∈ 0..rows(R) − 1
        |     V ← D_B(R_{j,i})
        |     M_b1_{i·rows(R)+j} ← V_0
        | M_b1
```

M_b1 =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	...

Процедура виконує для всіх елементів масиву R формування двійкового коду десяткового числа і записує його в змінну V. Нульовий (найменш значущий) біт масиву V заноситься до відповідного елемент масиву M_b1. Індекс елементів масиву M_b1 змінюється в залежності від номерів рядків і стовпців оброблюваного елемента масиву R. У результаті отримуємо лінійний бітовий масив M_b1, заповнений найменш значущими бітами масиву растрових даних каналу червоного кольору заповненого контейнера.

1.5. Перетворимо масив інформаційних даних.

Для формування текстового повідомлення за витягнутими (отриманими) битами сформуємо масив M1. Для цього скористаємося наступною процедурою:

```
M1 := | for i ∈ 0.. rows(M_b1)/8 − 1
      |     for j ∈ 0..7
      |       V_j ← M_b1_(i·8+j)
      |     M1_i ← B_D(V)
      | M1
```

Алгоритм формування масиву M1 наступний. Для всіх елементів масиву M_b1 обчислюємо значення індексу l - поточного номеру десяткового числа (коду інформаційного символу). Для цього беремо цілу частину від ділення і (індексу біта в масиві M_b1) на вісім (число біт в одному інформаційному символі в кодуванні ASCII). Далі, всі біти поточного символу записуємо в службову змінну V, після чого за допомогою функції B_D(x) по черзі обчислюємо коди інформаційних символів. Отримані цілі числа (коди символів в кодуванні ASCII) записуємо в масив M1.

M1 =

	0
0	200
1	237
2	242
3	229
4	240
5	229
6	241
7	32
8	234
9	32
10	241
11	242
12	229
13	227
14	224
15	237
16	238
17	227
18	240
19	224
20	…

Виконання процедури перетворення бітового масиву M_b1 в масив десяткових чисел M1 відбувається наступним чином:

i=0:

j=0: $V_0=M\_b1_0=0$,
j=1: $V_1=M\_b1_1=0$,
j=2: $V_2=M\_b1_2=0$,
j=3: $V_3=M\_b1_3=1$,
j=4: $V_4=M\_b1_4=0$,
j=5: $V_5=M\_b1_5=0$,
j=6: $V_6=M\_b1_6=1$,
j=7: $V_7=M\_b1_7=1$;

$M1_0=B\_D(V)=B\_D((0\ 0\ 0\ 1\ 0\ 0\ 1\ 1)^T)=200$;

i=1:

j=0: $V_0=M\_b1_8=1$,
j=1: $V_1=M\_b1_9=0$,
j=2: $V_2=M\_b1_{10}=1$,
j=3: $V_3=M\_b1_{11}=1$,
j=4: $V_4=M\_b1_{12}=0$,
j=5: $V_5=M\_b1_{13}=1$,
j=6: $V_6=M\_b1_{14}=1$,
j=7: $V_7=M\_b1_{15}=1$,

$M1_1=B\_D(V)=B\_D((1\ 0\ 1\ 1\ 0\ 1\ 1\ 1)^T)=237$;

…

1.6. Отриманий масив цілих чисел записуємо на фізичний носій у вигляді текстового файлу. Для цього скористаємося командою:

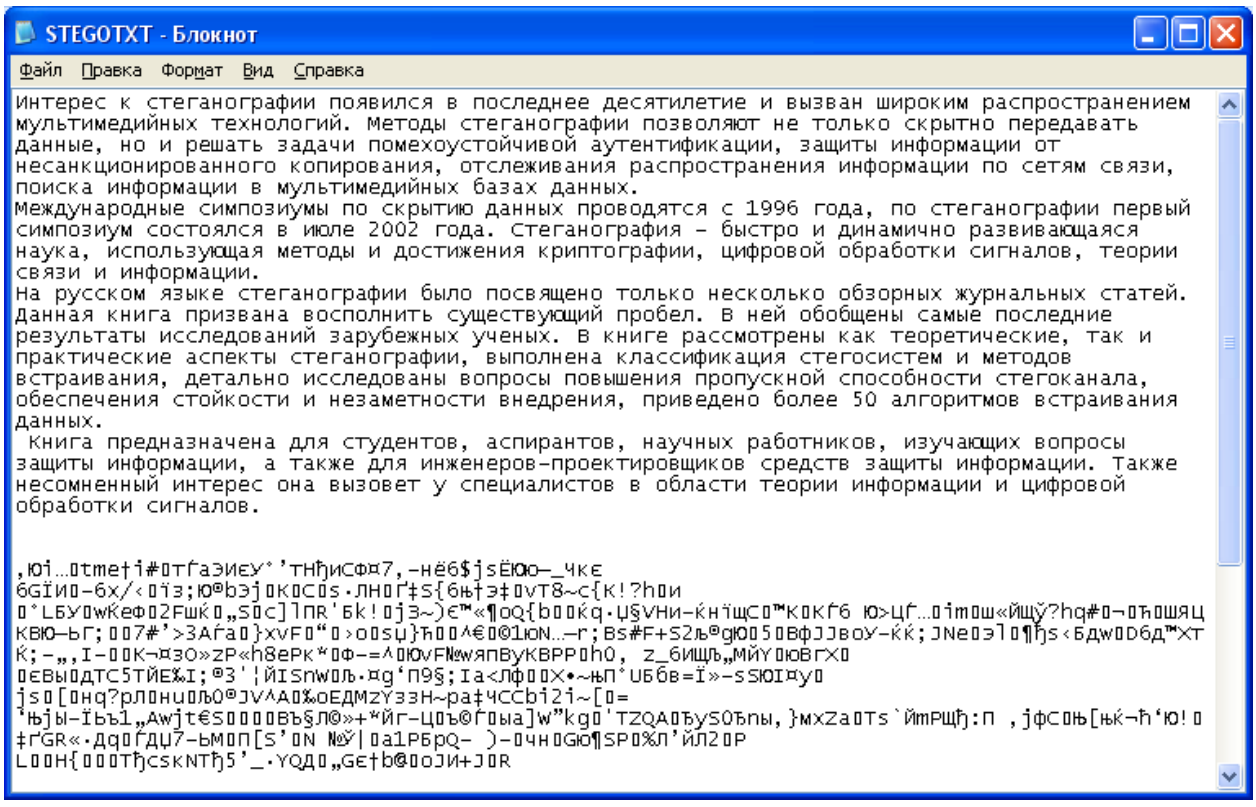
«WRITEBIN(“[ім’я файлу].txt”, byte,1):=M1».

Для розглянутого прикладу виконуємо команду:

«WRITEBIN(“STEGOTXT.txt”, byte,1):=M1»,

в результаті якої на фізичному носії буде записаний текстовий файл з ім'ям "STEGOTXT.txt".

НЕобхідно зазначити, що наведена вище процедура витягує (дозволяє отримати) всі найменш значущі біти контейнера (з каналу червоного кольору), тобто витягуються всі LSB навіть з тих байтів, в які не виконувалося вбудовування інформації. Після формування повідомлення в кінці текстового файлу "STEGOTXT.txt" можуть бути присутні символи, отримані в результаті вилучення LSB, немодифіковані в процесі вбудовування інформації, тобто так звані «випадкові» символи.



Останній рисунок наочно демонструє правильність роботи розглянутих вище процедур і функцій. Інформаційне повідомлення, вбудоване в просторову область нерухомого зображення методом LSB, вилучено правильно.

Завдання 2. Експериментальні дослідження зорового порогу чутливості людини до змінення яскравості зображень

2.1. Внесемо зміни в реалізацію алгоритму вбудовування даних в нерухомі зображення методом LSB. Для цього змінимо порядковий номер біта, який використовується для вбудовування інформації в двійковому поданні елементів контейнера (окремих байт яскравості конкретних пікселів зображення).

```
S := for j ∈ 0..rows(R) − 1
       for i ∈ 0..cols(R) − 1
         S_{j,i} ← R_{j,i}
     for l ∈ 0..rows(M_b) − 1
       i ← floor(l / rows(R))
       j ← l − i·rows(R)
       V ← (D_B(R_{j,i}))
       V_0 ← M_b_l
       S_{j,i} ← B_D(V)
     S
```

```
S := for j ∈ 0..rows(R) − 1
       for i ∈ 0..cols(R) − 1
         S_{j,i} ← R_{j,i}
     for l ∈ 0..rows(M_b) − 1
       i ← floor(l / rows(R))
       j ← l − i·rows(R)
       V ← (D_B(R_{j,i}))
       V_1 ← M_b_l
       S_{j,i} ← B_D(V)
     S
```

```
S := for j ∈ 0..rows(R) − 1
       for i ∈ 0..cols(R) − 1
         S_{j,i} ← R_{j,i}
     for l ∈ 0..rows(M_b) − 1
       i ← floor(l / rows(R))
       j ← l − i·rows(R)
       V ← (D_B(R_{j,i}))
       V_2 ← M_b_l
       S_{j,i} ← B_D(V)
     S
```

Перша наведена процедура реалізує вбудовування інформаційних даних в найменш значущі (нульові) біти контейнера (в біти V_0). Друга і третя процедури реалізують вбудовування інформаційних даних в наступні за значимістю біти контейнера (в біти V_1 та V_2). Наведемо приклад графічного зображення контейнера для кожного з розглянутих випадків.

S

S

S

Вочевидь, що видимих спотворень при вбудовуванні інформаційних повідомлень в нульовий, перший або в другий за значимістю біти виявити не вдається. Це пояснюється тим, що максимальні спотворення, які вносяться до окремих пікселів зображення за допомогою зміни рівня їх яскравості, для кожного з розглянутих випадків не перевищують величин $2^0=1$, $2^1=2$, $2^2=4$, відповідно, що знаходиться нижче порога чутливості зорової системи людини до незначного зміни яскравості зображення.

Продовжимо змінювати порядковий номер біта, який використовується для вбудовування інформації, в двійковому поданні елементів контейнера, до тих пір, поки візуально не стануть видні спотворення яскравості окремих пікселів зображення.

```
S := | for j ∈ 0..rows(R) − 1
     |   for i ∈ 0..cols(R) − 1
     |     S_{j,i} ← R_{j,i}
     | for l ∈ 0..rows(M_b) − 1
     |   i ← floor(l / rows(R))
     |   j ← l − i·rows(R)
     |   V ← (D_B(R_{j,i}))
     |   V_3 ← M_b_l
     |   S_{j,i} ← B_D(V)
     | S
```

```
S := | for j ∈ 0..rows(R) − 1
     |   for i ∈ 0..cols(R) − 1
     |     S_{j,i} ← R_{j,i}
     | for l ∈ 0..rows(M_b) − 1
     |   i ← floor(l / rows(R))
     |   j ← l − i·rows(R)
     |   V ← (D_B(R_{j,i}))
     |   V_4 ← M_b_l
     |   S_{j,i} ← B_D(V)
     | S
```

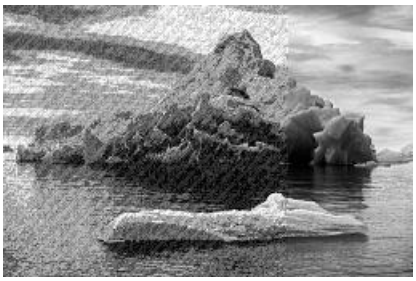
```
S := | for j ∈ 0..rows(R) − 1
     |   for i ∈ 0..cols(R) − 1
     |     S_{j,i} ← R_{j,i}
     | for l ∈ 0..rows(M_b) − 1
     |   i ← floor(l / rows(R))
     |   j ← l − i·rows(R)
     |   V ← (D_B(R_{j,i}))
     |   V_5 ← M_b_l
     |   S_{j,i} ← B_D(V)
     | S
```

Наведені процедури реалізують вбудовування інформаційних даних в наступні за значимістю біти контейнера (в біти V_3, V_4 и V_5).

2.2. Експериментально встановимо, модифікація яких бітів зображення не призводить до помітних спотворень. Для цього наведемо приклад графічного зображення контейнера для кожного з розглянутих випадків (вбудовування здійснювалося в біти V_3, V_4 и V_5).

S

S



S

З наведених зображень зрозуміло, що вбудовування даних в треті за значимістю біти контейнеру призводить до ледь помітних спотворень (яскравість відповідних пікселів змінилася на 2^3=8 рівнів). Модифікація четвертих і п'ятих за значимістю бітів призводить до значних спотворень зображення (яскравість відповідних пікселів змінилася на 2^4=16 та 2^5=32 рівні, відповідно). Отже, з результатів експериментальних досліджень отримуємо, що зорова система людини для розглянутого прикладу зображення чутлива до зміни третього, четвертого і т.д. бітів (за їх значимістю) контейнера (окремих байт яскравості конкретних пікселів зображення).

2.3. Розрахуємо поріг зорової чутливості системи людини до незначної зміни яскравості зображення.

Використовуємо експериментальні дані для розрахунку порогу зорової чутливості системи людини до незначної зміни яскравості зображення. Позначимо символом Δ величину внесених спотворень яскравості (як число рівнів квантування) окремих пікселів зображення при використанні стеганографічного алгоритму вбудовування інформації в нерухомі зображення на основі модифікації окремих бітів контейнера. За специфікацією формату зображень *bmp24 загальне число рівнів квантування яскравості окремих пікселів дорівнює 2^8=256. Тоді зоровий поріг чутливості (ПЧ) системи людини до незначної зміни яскравості зображення визначимо як

$$ПЧ=(Δ/256)*100\%.$$

Для розглянутого прикладу видимі спотворення були виявлені після модифікації третіх за значимістю бітів контейнера, відповідна величина Δ=8. Отже, поріг зорової чутливості системи людини, обчислений за емпіричними даними, становить:

$$ПЧ=(Δ/256)*100\%=(8/256)*100\%=3,125\%,$$

що узгоджується з відомими теоретичними даними.

Завдання 3. Реалізація алгоритмів вбудовування та вилучення повідомлень методом псевдовипадкової перестановки

3.1. Завантажуємо вихідні дані (див. п. 1.1.).

3.2. Перетворюємо масив інформаційних даних (див. п. 1.2.).

3.3. Вводимо секретне правило псевдовипадкової перестановки

Вбудовувана інформація попередньо обробляється простим перестановочним шифром. Задамо секретний ключ - правило псевдовипадкової перестановки у вигляді перестановочної матриці розміром nxn, де n-розмір оброблюваного блоку інформаційних бітів. Нехай, наприклад, n=10. Тоді перестановочна матриця складається з двомірного масиву 10x10 бітів, причому в кожному рядку і в кожному стовпці масиву міститься тільки один одиничний елемент, всі інші елементи - нулі.

Для розглянутого прикладу задамо перестановочну матрицю наступним чином:

$$P := \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

3.4. Розбиваємо інформаційне повідомлення на блоки однакової довжини і обробляємо простим перестановочним шифром (шифруємо). Для цього скористаємося наступною процедурою:

$$\begin{array}{l} \text{M\_b\_P} := \left| \begin{array}{l} \text{for } i \in 0..\left(\dfrac{\text{rows(M\_b)}}{n} - 1 \right) \\ \quad \left| \begin{array}{l} \text{for } j \in 0..n-1 \\ \quad V_j \leftarrow \text{M\_b}_{i \cdot n + j} \end{array} \right. \\ \quad V1 \leftarrow (V)^T \cdot P \\ \quad \text{for } j \in 0..n-1 \\ \quad \text{M\_b\_P}_{i \cdot n + j} \leftarrow \left(V1^T \right)_j \\ \text{M\_b\_P} \end{array} \right. \end{array}$$

Алгоритм перетворення працює наступним чином. Інформаційне повідомлення розбивається на блоки однакової довжини (довжини n), після чого кожен блок поелементно записується в службову змінну V. Таким чином, на кожному циклі (для кожного блоку даних) в змінній V зберігаються поточні n бітів повідомлення. Вектор V множиться на перестановочну матрицю P, чим забезпечується шифрування простим перестановочним шифром. Оброблені таким чином дані об'єднуються в єдиний бітовий масив M_b_P.

Для розглянутого прикладу інформаційних даних (див. п. 1.1.) алгоритм розбиття інформаційного повідомлення на блоки однакової довжини і обробки простим перестановочним шифром функціонує наступним чином.

M_b =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	1
16	0
17	1
18	0
19	0
20	1
21	1
22	1
23	1
24	…

i=0: V=(0 0 0 1 0 0 1 1 1 0)^T,
V1=V^T*P=(0 1 0 0 1 1 0 0 1 0),
j=0: M_b_P$_0$=0,
j=1: M_b_P$_1$=1,
j=2: M_b_P$_2$=0,
j=3: M_b_P$_3$=0,
j=4: M_b_P$_4$=1,
j=5: M_b_P$_5$=1,
j=6: M_b_P$_6$=0,
j=7: M_b_P$_7$=0,
j=8: M_b_P$_8$=1,
j=9: M_b_P$_9$=0;
i=0: V=(1 1 0 1 1 1 0 1 0 0)^T,
V1=V^T*P=(1 1 1 0 0 0 1 0 1 1),
j=0: M_b_P$_{10}$=1,
j=1: M_b_P$_{11}$=1,
j=2: M_b_P$_{12}$=1,
j=3: M_b_P$_{13}$=0,
j=4: M_b_P$_{14}$=0,
j=5: M_b_P$_{15}$=0,
j=6: M_b_P$_{16}$=1,
j=7: M_b_P$_{17}$=0,
j=8: M_b_P$_{18}$=1,
j=9: M_b_P$_{19}$=1;
…

M_b_P =

	0
0	0
1	1
2	0
3	0
4	1
5	1
6	0
7	0
8	1
9	0
10	1
11	1
12	1
13	0
14	0
15	0
16	1
17	0
18	1
19	1
20	1
21	1
22	1
23	1
24	…

3.5. Реалізуємо алгоритм вбудовування даних в просторову область зображень методом LSB (див. п. 1.3.).

3.6. Реалізуємо алгоритм вилучення даних з просторової області зображень методом LSB (див. п. 1.4.).

3.7. Розбиваємо отримане повідомлення (записане в бітовий масив M_b1) на блоки однакової довжини і обробляємо простим перестановочним шифром (розшифровуємо). Для цього скористаємося наступною процедурою

```
M_b1_P := | for i ∈ 0..(rows(M_b1)/n − 1)
          |   for j ∈ 0..n − 1
          |     V_j ← M_b1_(i·n+j)
          |   V1 ← (V)^T · P^(−1)
          |   for j ∈ 0..n − 1
          |     M_b1_P_(i·n+j) ← (V1^T)_j
          | M_b1_P
```

Алгоритм перетворення працює наступним чином. Отримане повідомлення розбивається на блоки однакової довжини (довжини n), після чого кожен блок поелементно записується в службову змінну V. Таким чином, на кожному циклі (для кожного блоку витягнутих даних) в змінній V зберігаються поточні n бітів. Вектор V множиться на матрицю P^{-1} (матрицю, зворотну до введеної вище перестановочної матриці P), завдяки чому забезпечується розшифрування отриманих даних простим перестановочним шифром. Оброблені таким чином дані об'єднуються в єдиний бітовий масив M_b1_P.

Для розглянутого прикладу інформаційних даних (див. п. 3.4.) алгоритм розбиття отриманого повідомлення на блоки однакової довжини і обробки простим перестановочним шифром (розшифрування) функціонує наступним чином.

M_b1 =

	0
0	0
1	1
2	0
3	0
4	1
5	1
6	0
7	0
8	1
9	0
10	1
11	1
12	1
13	0
14	0
15	0
16	1
17	0
18	1
19	1
20	1
21	1
22	1
23	1
24	…

i=0: $V=(0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 0)^T$,
$V1=V^T*P=(0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 1\ 0)$,
j=0: $M\_b\_P_0=0$,
j=1: $M\_b\_P_1=0$,
j=2: $M\_b\_P_2=0$,
j=3: $M\_b\_P_3=1$,
j=4: $M\_b\_P_4=0$,
j=5: $M\_b\_P_5=0$,
j=6: $M\_b\_P_6=1$,
j=7: $M\_b\_P_7=1$,
j=8: $M\_b\_P_8=1$,
j=9: $M\_b\_P_9=0$;
i=0: $V=(1\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 1\ 1)^T$,
$V1=V^T*P=(1\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0)$,
j=0: $M\_b\_P_{10}=1$,
j=1: $M\_b\_P_{11}=1$,
j=2: $M\_b\_P_{12}=0$,
j=3: $M\_b\_P_{13}=1$,
j=4: $M\_b\_P_{14}=1$,
j=5: $M\_b\_P_{15}=1$,
j=6: $M\_b\_P_{16}=0$,
j=7: $M\_b\_P_{17}=1$,
j=8: $M\_b\_P_{18}=0$,
j=9: $M\_b\_P_{19}=0$;
…

M_b1_P =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	1
16	0
17	1
18	0
19	0
20	1
21	1
22	1
23	1
24	…

3.8. Перетворимо масив інформаційних даних (див. п. 1.5.).

3.9. Отриманий масив цілих чисел записуємо на фізичний носій у вигляді текстового файлу (див. п. 1.6.).

Завдання 4. Реалізація алгоритмів вбудовування та вилучення повідомлень методом псевдовипадкового інтервалу

4.1. Завантажуємо вихідні дані (див. п. 1.1.).

4.2. Перетворюємо масив інформаційних даних (див. п. 1.2.).

4.3. Вводимо секретне правило псевдовипадкової перестановки.

Секретним ключем виступає правило, за яким окремі біти інформаційного повідомлення вбудовуються в LSB байт контейнера, тобто

секретний ключ - це набір псевдовипадкових чисел, які задають величини інтервалів між пікселями зображення, які модифікуються в процесі вбудовування інформації.

key =

	0
0	153
1	155
2	25
3	96
4	159
5	97
6	43
7	59
8	134
9	11
10	99
11	33
12	108
13	102
14	74
15	…

Припустимо, що інформаційне повідомлення вбудовується побітово в блок даних зображень, причому один біт повідомлення вбудовується в один стовпець масиву даних контейнера. Номер вбудованого біта задає номер стовпчика контейнера, в який буде вбудовуватися біт повідомлення. Поточне значення секретного ключа задає номер рядка контейнера, в який буде вбудований поточний біт повідомлення. Таким чином, в якості секретного ключа будемо використовувати масив псевдовипадкових чисел в інтервалі допустимих номерів рядків контейнера, розмір масиву дорівнює числу стовпців контейнера. Для реалізації такого підходу використовуємо таку процедуру.

```
key := | for i ∈ 0..cols(R) − 1
       |   key_i ← floor(rnd(rows(R)))
       | key
```

З наведеного прикладу видно, що перший біт повідомлення буде вбудований в 153-й рядок першого стовпця масиву даних контейнера, другий біт повідомлення буде вбудований в 155-й рядок другого стовпця повідомлення і т.д.

4.4. Реалізуємо алгоритм вбудовування даних в просторову область зображень методом ПСІ. Для цього скористаємося наступною процедурою.

```
S := | for j ∈ 0..rows(R) − 1
     |   for i ∈ 0..cols(R) − 1
     |     S_{j,i} ← R_{j,i}
     | for i ∈ 0..cols(R) − 1
     |   | V ← D_B(R_{key_i, i})
     |   | V_0 ← M_b_i
     |   | S_{key_i, i} ← B_D(V)
     | S
```

Алгоритм працює наступним чином. Вихідний контейнер (масив R даних каналу червоного кольору) перезаписується в новий масив S. Далі, в кожному стовпці контейнера зчитується значення яскравості (десяткове число) з рядка з номером, що задається ключем. У отриманому числі

замінюється найменш значимий біт даних на поточний вбудовуваний біт. Далі записуємо заповнений контейнер на фізичний носій (див. п. 1.3.).

4.5. Реалізуємо алгоритм вбудовування даних в просторову область зображень методом ПСІ. Для цього після завантаження даних контейнера (див. п. 1.4) скористаємося такою процедурою.

```
M_b1 := | for i ∈ 0..cols(R) − 1
        |     V ← D_B(R_(key_i, i))
        |     M_b1_i ← V_0
        | M_b1
```

Алгоритм працює наступним чином. У всіх стовпцях контейнера по черзі зчитуються значення з рядків, номери яких задані значенням секретного ключа. З отриманих даних витягуються найменш значущі біти, які несуть інформаційний зміст вбудованого повідомлення.

4.6. Перетворюємо масив інформаційних даних (див. п. 1.5.).

4.7. Отриманий масив цілих чисел записуємо на фізичний носій у вигляді текстового файлу (див. п. 1.6.).

6. Приклад оформлення звіту з лабораторної роботи

Лабораторна робота №1
Приховування даних в просторовій області зображень шляхом модифікації найменш значущого біта

Вихідні дані:

"1.bmp"

C := READRGB("1.bmp")

R := READ_RED("1.bmp")

G := READ_GREEN("1.bmp")

B := READ_BLUE("1.bmp")

M := READBIN("3.txt","byte")

R

C =

	0	1	2	3	4	5	6	7
0	86	79	72	72	72	69	71	74
1	110	97	90	86	78	71	70	70
2	132	120	112	105	96	88	81	83
3	122	116	105	104	103	102	101	99
4	131	122	117	118	118	116	105	107
5	147	147	148	148	150	153	145	135
6	169	164	167	170	173	175	164	155
7	189	195	193	189	183	172	173	173
8	191	192	194	199	194	187	182	182
9	186	188	194	198	192	187	187	180
10	195	196	199	200	201	190	192	186
11	185	189	202	203	203	199	203	199
12	192	196	198	199	204	202	206	199
13	177	185	187	186	180	178	179	177
14	173	176	174	166	165	165	163	161
15	160	162	158	153	156	158	157	156

R =

	0	1	2	3	4	5	6	7
0	86	79	72	72	72	69	71	74
1	110	97	90	86	78	71	70	70
2	132	120	112	105	96	88	81	83
3	122	116	105	104	103	102	101	99
4	131	122	117	118	118	116	105	107
5	147	147	148	148	150	153	145	135
6	169	164	167	170	173	175	164	155
7	189	195	193	189	183	172	173	173
8	191	192	194	199	194	187	182	182
9	186	188	194	198	192	187	187	180
10	195	196	199	200	201	190	192	186
11	185	189	202	203	203	199	203	199
12	192	196	198	199	204	202	206	199
13	177	185	187	186	180	178	179	177
14	173	176	174	166	165	165	163	161
15	160	162	158	153	156	158	157	...

M =

	0
0	200
1	237
2	242
3	229
4	240
5	229
6	241
7	32
8	234
9	32
10	241
11	242
12	229
13	...

Програмна реалізація алгоритмів приховування повідомлень методом LSB:

$$B\_D(x) := \sum_{i=0}^{7} \left(x_i \cdot 2^i \right)$$

$$D\_B(x) := \left| \begin{array}{l} \text{for } i \in 0..7 \\ \quad \left| \begin{array}{l} V_i \leftarrow \text{mod}(x, 2) \\ x \leftarrow \text{floor}\left(\frac{x}{2} \right) \end{array} \right. \\ V \end{array} \right.$$

$$M\_b := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(M) - 1 \\ \quad \left| \begin{array}{l} V \leftarrow D\_B\left(M_i \right) \\ \text{for } j \in 0..7 \\ \quad M\_b_{i \cdot 8 + j} \leftarrow V_j \end{array} \right. \\ M\_b \end{array} \right.$$

$M\_b =$

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	1

$$S := \left| \begin{array}{l} \text{for } j \in 0..\text{rows}(R) - 1 \\ \quad \text{for } i \in 0..\text{cols}(R) - 1 \\ \quad\quad S_{j,i} \leftarrow R_{j,i} \\ \text{for } l \in 0..\text{rows}(M\_b) - 1 \\ \quad \left| \begin{array}{l} i \leftarrow \text{floor}\left(\frac{l}{\text{rows}(R)} \right) \\ j \leftarrow l - i \cdot \text{rows}(R) \\ V \leftarrow D\_B\left(R_{j,i} \right) \\ V_0 \leftarrow M\_b_l \\ S_{j,i} \leftarrow B\_D(V) \end{array} \right. \\ S \end{array} \right.$$

$$WRITERGB("Stego.bmp") := \text{augment}(S, G, B)$$

Візуальне порівняння пустого та заповненого контейнерів:

"1.bmp"

"Stego.bmp"

Програмна реалізація алгоритмів вилучення повідомлень методом LSB:

```
M_b1 := | for i ∈ 0..cols(R) − 1
        |   for j ∈ 0..rows(R) − 1
        |     V ← D_B(R[j,i])
        |     M_b1[i·rows(R)+j] ← V0
        | M_b1
```

M_b1 =

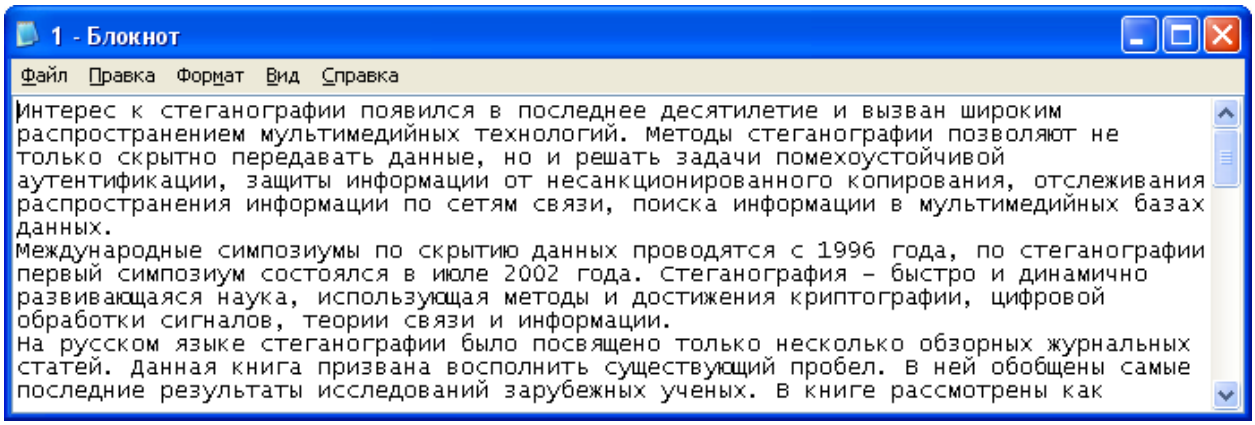
	0
0	0
1	0
2	0
3	0
4	1
5	1
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	...

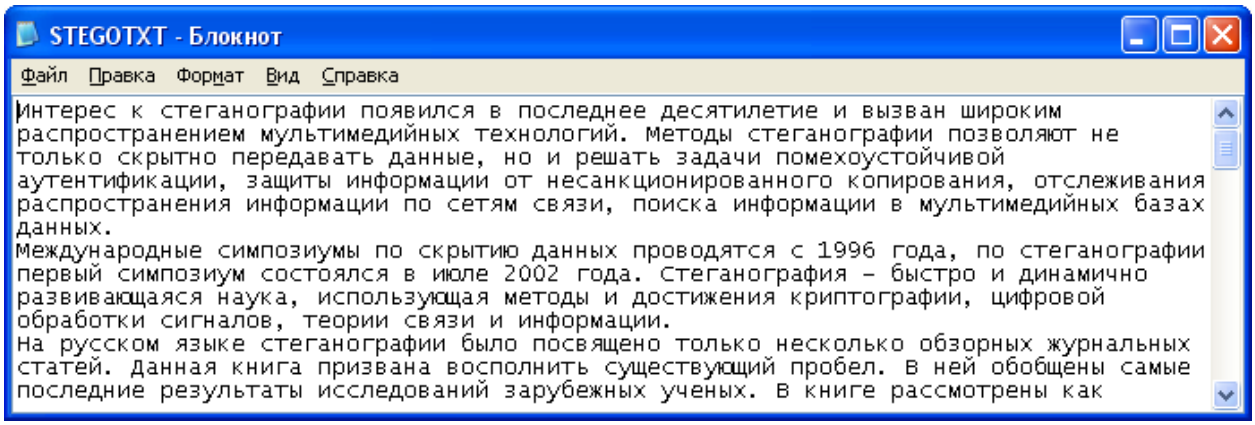
```
M1 := | for i ∈ 0..rows(M_b1) − 8
      |   l ← floor(i/8)
      |   for j ∈ 0..7
      |     V_j ← M_b1[l·8+j]
      |   M1_l ← B_D(V)
      | M1
```

M1 =

	0
0	240
1	109
2	233
3	239
4	168
5	196
6	117
7	226
8	59
9	6
10	231
11	194
12	108
13	253
14	28
15	...

```
WRITEBIN("STEGOTXT.txt","byte",1) := M1
```





Програмна реалізація алгоритмів приховування та вилучення повідомлень методом ПВП:

$$P := \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$n := 10$

$$M\_b\_P := \left| \begin{array}{l} \text{for } i \in 0..\left(\frac{\text{rows}(M\_b)}{n} - 1\right) \\ \quad \left| \begin{array}{l} \text{for } j \in 0..n-1 \\ \quad V_j \leftarrow M\_b_{i \cdot n + j} \\ V1 \leftarrow (V)^T \cdot P \\ \text{for } j \in 0..n-1 \\ \quad M\_b\_P_{i \cdot n + j} \leftarrow \left(V1^T\right)_j \end{array} \right. \\ M\_b\_P \end{array} \right.$$

$M\_b =$

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	…

$M\_b\_P =$

	0
0	0
1	1
2	0
3	0
4	1
5	1
6	0
7	0
8	1
9	0
10	1
11	1
12	1
13	0
14	0
15	…

$$M\_b1\_P := \left| \begin{array}{l} \text{for } i \in 0..\left(\frac{\text{rows}(M\_b1)}{n} - 1\right) \\ \quad \left| \begin{array}{l} \text{for } j \in 0..n-1 \\ \quad V_j \leftarrow M\_b1_{i \cdot n + j} \\ V1 \leftarrow (V)^T \cdot P^{-1} \\ \text{for } j \in 0..n-1 \\ \quad M\_b\_P1_{i \cdot n + j} \leftarrow \left(V1^T\right)_j \end{array} \right. \\ M\_b\_P1 \end{array} \right.$$

$M\_b1 =$

	0
0	0
1	1
2	0
3	0
4	1
5	1
6	0
7	0
8	1
9	0
10	1
11	1
12	1
13	0
14	0
15	…

$M\_b1\_P =$

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	…

Програмна реалізація алгоритмів приховування та вилучення повідомлень методом ПВІ:

```
key := | for i ∈ 0..cols(R) − 1
       |   key_i ← floor(rnd(rows(R)))
       | key
```

```
S := | for j ∈ 0..rows(R) − 1
     |   for i ∈ 0..cols(R) − 1
     |     S_{j,i} ← R_{j,i}
     | for i ∈ 0..cols(R) − 1
     |   | V ← D_B(R_{key_i,i})
     |   | V_0 ← M_b_i
     |   | S_{key_i,i} ← B_D(V)
     | S
```

key =

	0
0	153
1	155
2	25
3	96
4	159
5	97
6	43
7	59
8	134
9	11
10	99
11	…

R =

	0	1	2	3	4
148	162	151	160	164	161
149	147	154	153	159	135
150	144	134	84	118	146
151	97	123	177	153	140
152	98	116	83	92	106
153	125	129	144	143	134
154	113	105	92	101	107
155	132	121	105	84	110
156	104	89	97	111	108
157	122	126	110	103	…

S =

	0	1	2	3	4
148	162	151	160	164	161
149	147	154	153	159	135
150	144	134	84	118	146
151	97	123	177	153	140
152	98	116	83	92	106
153	124	129	144	143	134
154	113	105	92	101	107
155	132	120	105	84	110
156	104	89	97	111	108
157	122	126	110	103	…

```
M_b1 := | for i ∈ 0..cols(R) − 1
        |   | V ← D_B(S_{key_i,i})
        |   | M_b1_i ← V_0
        | M_b1
```

M_b =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	…

M_b1 =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	…

Лабораторна робота №2 «Приховування даних в просторовій області нерухомих зображень методом блокового вбудовування, методом квантування та методом "хреста"»

1. Мета та завдання лабораторної роботи

Мета роботи: закріпити теоретичні знання за темою «Приховування даних у просторової області нерухомих зображень методом блокового вбудовування, методом квантування та методом "хреста"», набути практичних вмінь та навичок щодо розробки стеганографічних систем, дослідити властивості стеганографічних методів, що засновані на низькорівневих властивостях зорової системи людини (ЗСЛ).

Лабораторна робота №2 виконується у середовищі символьної математики MathCAD версії 12 або вище.

Завдання до лабораторної роботи

1. Реалізувати у середовищі символьної математики MathCAD алгоритми приховування та вилучення даних у просторовій області зображень методом блокового вбудовування. Виконати зорове порівняння пустого та заповненого контейнера та переконатися у відсутності помітних похибок. Переконатися в автентичності вилученого повідомлення. Отримати заповнені контейнери від інших груп та переконатися у відсутності помітних похибок. Вилучити повідомлення інших груп із отриманих заповнених контейнерів та переконатися у їхній автентичності.
2. Реалізувати у середовищі символьної математики MathCAD алгоритми приховування та вилучення даних у просторової області зображень методом квантування.
3. Реалізувати у середовищі символьної математики MathCAD алгоритми приховування та вилучення даних у просторової області зображень методом Куттера-Джордана-Боссена (методом "хреста").
4. Провести експериментальні дослідження ймовірнісних властивостей методу «хреста», отримати емпіричні залежності ймовірності правильного вилучення даних та частки внесених при цьому похибок у контейнер-зображення.
5. (Додаткове завдання). Реалізувати у середовищі символьної математики MathCAD алгоритми завадостійкого кодування інформаційних даних для покращення ймовірнісних властивостей стеганографічного методу вбудовування даних Куттера-Джордана-Боссена (методу "хреста").

2. Методичні рекомендації з організації самостійної роботи

1. Вивчити теоретичний матеріал лекції «Приховування даних у просторової області зображень методом блокового вбудовування, методом квантування та методом "хреста"».
2. Вивчити матеріал основного джерела літератури (Конахович Г.Ф., Пузыренко А.Ю. Компьютерная стеганография):
   a. метод блокового вбудовування (ст. 97-98);
   b. метод квантування (ст. 103 - 106);
   c. метод Куттера-Джордана-Боссена (ст. 106-110).
3. Вивчити матеріал додаткових джерел:
   a. структура лінійних блокових кодів, стандартне розташування, коди Хемінга (Р. Блейхут. Теория и практика кодов, контролирующих ошибки, ст. 61 - 73);
   b. принципи побудови та властивості генераторів псевдовипадкових послідовностей (Иванов М.А., Чугунков И.В. Теория, применение и оценка качества генераторов псевдослучайных последовательностей, ст. 5 - 64);
4. Вивчити основні команди у середовищі символьної математики MathCAD щодо роботи із зображеннями.
5. Підготувати відповіді на контрольні запитання.
6. Підготувати бланк звіту з лабораторної роботи.

Допуск до виконання лабораторної роботи здійснюється за результатами письмового опитування.

3. Загальнотеоретичні положення за темою лабораторної роботи

3.1 Метод квантування зображення

До методів приховування в просторовый області можна також віднести метод квантування зображення, заснований на міжпіксельный залежності, яку можна описати деякою функцією Θ. У найпростішому випадку можна обчислити різницю між суміжними пікселями c_i; і c_{i+1} (або c_{i-1} і c_i) і задати її як параметр функції Θ: $\Delta_i = \Theta(c_i - c_{i+1})$, де Δ_i - дискретна апроксимація різниці сигналів $c_i - c_{i+1}$.

Оскільки Δ_i - ціле число, а реальна різниця $c_i - c_{i+1}$ - дійсне число, то виникають помилки квантування $\delta_i = \Delta_i - \varepsilon_i$. Для сильно корелюється сигналів ця помилка є близькою до нуля: $\delta_i \approx 0$.

При цьому методі приховування інформації проводиться шляхом коригування різносного сигналу Δ_i. Стеганоключ є таблицею, яка кожному можливому значенню Δ_i ставить у відповідність певний біт, наприклад:

Δ_i	-4	-3	-2	-1	0	1	2	3	4
bi	1	0	1	1	0	0	1	0	1

Щоб приховати і-тий біт повідомлення обчислюється різниця Δ_i. Якщо при цьому bi, не відповідає секретному біту, який необхідно приховати, то значення Δ_i замінюється найближчим Δ_j, для якого така умова виконується. При цьому відповідним чином коригуються значення інтенсивностей пікселів, між якими обчислювалася різниця Δ_i. Вилучення секретного повідомлення здійснюється відповідно до значення b_i^*, яку відповідає різниці Δ_i^*.

3.2 Метод Куттера-Джордана-Боссена

Куттер (M. Kutter), Джордан (F. Jordan) і Боссен (F. Bossen) запропонували алгоритм вбудовування в канал синього кольору зображення, що має RGB-кодування, оскільки до синього кольору зорова система людини є найменш чутливою. Розглянемо алгоритм передачі одного біта секретної інформації в запропонованому методі.

Нехай M_i - біт, який підлягає вбудовуванню, $C = \{R, G, B\}$ - зображення-контейнер, p=(x,y) псевдовипадковий піксель контейнера, в який буде виконуватися вбудовування.

Секретний біт M_i вбудовується в канал синього кольору шляхом модифікації яскравості

$$\lambda_{x,y} = 0.29890 \cdot R_{x,y} + 0.58662 \cdot G_{x,y} + 0.11448 \cdot B_{x,y}$$

$$B'_{x,y} = \begin{cases} B_{x,y} - \upsilon \cdot \lambda_{x,y}, & \text{при } m_i = 0; \\ B_{x,y} - \upsilon \cdot \lambda_{x,y}, & \text{при } m_i = 1; \end{cases} = B_{x,y} + (2 \cdot m_i - 1) \cdot \upsilon \cdot \lambda_{x,y}$$

де υ - константа, яка визначає енергію вбудованого сигналу. Її величина залежить від призначення стеганосистеми. Чим більше υ, тим вище стійкість вбудованої інформації до спотворень, проте і тим сильніше її помітність.

Одержувач отримує біт, не маючи первинного зображення, тобто, "наосліп". Для цього виконується прогнозування значення первинного, немодифікованого пікселя на основі значень сусідніх пікселів. Для отримання оцінки пікселя запропоновано використовувати значення декількох пікселів, розміщених в тому ж стовпці і в тому ж рядку масиву графічного контейнера. Використовують "хрест" пікселів розміром 7x7. Оцінка $\hat{B}^*_{x,y}$ виходить у вигляді

$$\hat{B}^*_{x,y} = \frac{1}{4 \cdot \sigma} \cdot \left[\sum_{i=-\sigma}^{+\sigma} B^*_{x+i,y} + \sum_{j=-\sigma}^{+\sigma} B^*_{x+i,j} - 2 \cdot B^*_{x,y} \right]$$

де σ - кількість пікселів зверху (знизу, зліва, справа) від оцінюваного пікселя (в разі хреста 7x7 $\sigma = 3$).

Під час вилучення вбудованого біта обчислюється різниця δ між поточним ($B^*_{x,y}$) і прогнозованим ($\hat{B}^*_{x,y}$) значеннями інтенсивності пікселя p = (x, y):

$$\delta = B^*_{x,y} - \hat{B}^*_{x,y}$$

Знак δ означатиме вбудований біт: якщо $\delta < 0$, то Mi = 0; якщо $\delta > 0$, то Mi = 1.

Функції вбудовування та вилучення в даному методі є несиметричними, тобто, функція вилучення не є зворотною функцією до функції вбудовування. Хоча, як зазначають автори методу, правильне розпізнавання біта повідомлення в разі застосування описаних вище процедур є високоймовірним, проте не стовідсотковим. Для зменшення ймовірності помилок вилучення було запропоновано в процесі вбудовування кожен біт повторювати кілька разів (багаторазове вбудовування). Оскільки при цьому кожен біт був повторений τ разів, то виходить τ оцінок одного біта повідомлення. Секретний біт вилучається за результатами усереднення різниці між реальним і оціненим значеннями інтенсивності пікселя в отриманому контейнері:

$$\delta = \tau^{-1} \cdot \sum_{i=1}^{\tau} \left[B^*_{x,y} - \hat{B}^*_{x,y} \right]$$

Як і в попередньому випадку, знак усередненої різниці δ визначатиме значення вбудованого біта. Цей алгоритм є стійким до багатьох відомих видів атак: НЧ фільтрації зображення, його компресії відповідно до алгоритму JPEG, обрізанню країв.

4. Питання для поточного контролю підготовленості студентів до виконання лабораторної роботи №2

1. Найпростіші завадостійкі лінійні блокові коди. Коди з перевіркою парності, коди Хемінга. Матричний опис лінійних блокових кодів. Поліноміальний опис циклічних кодів.
2. Метод блокового вбудовування та його зв'язок з лінійними блоковими кодами з контролем парності. Ймовірнісні характеристики методу блокового вбудовування: ймовірність правильного вилучення повідомлень та ймовірність виникнення помилок.
3. Найпростіші генератори псевдовипадкових чисел. Вбудовані датчики псевдовипадкових чисел в середовищі символьної математики MathCAD
4. Поняття контрастності зображення. Чутливість зорової системи людини до незначної зміни контрастності. Вбудовування даних в нерухомі зображення методом квантування.

5. Криптографічні та некриптографічні генератори псевдовипадкових послідовностей. Доказово стійкі генератори псевдовипадкових послідовностей. Генератори RSA та BBS.
6. Генератори псевдовипадкових послідовностей на регістрах зсуву. Генератор послідовностей максимального періоду. Конгруентні генератори псевдовипадкових послідовностей. Лінійний конгруентний генератор та інверсивний конгруентний генератор.
7. Методи екстраполяції (передбачення) випадкових сигналів. Лінійне передбачення і дельта-модуляція. Диференціальна імпульсно-кодова модуляція. Сплайн інтерполяція і інтерполяційні багаточлени (поліноми) Ньютона та Лагранжа.
8. Метод вбудовування даних в нерухомі зображення Куттера-Джордана-Боссена (метод «хреста»). Лінійне передбачення сигналів при отриманні (вилученні) даних методом «хреста».
9. Структура лінійних блокових кодів, стандартне розташування. Декодування лінійних блокових кодів. Ймовірнісні характеристики завадостійкого кодування. Ймовірність виявлення та невиявлення помилок лінійними блоковими кодами. Ймовірність виправлення помилок декодером лінійного блокового коду. Ймовірність появи помилок на виході декодера.

5. Інструкція до виконання лабораторної роботи №2

Завдання 1. Реалізація в середовищі MathCAD алгоритмів вбудовування та вилучення (отримання) повідомлень в просторовій області нерухомих зображень методом блокового вбудовування

1.1. Завантажуємо вихідні дані: контейнер - нерухоме зображення (в форматі *.bmp24); інформаційне повідомлення - текстовий документ (у форматі *.txt). Для цього в середовищі MathCAD виконуємо наступні дії, аналогічні п. 1.1. інструкції до лабораторної роботи №1.

1.2. Перетворимо масив інформаційних даних. Для цього в середовищі MathCAD виконуємо наступні дії, аналогічні п. 1.2. інструкції до лабораторної роботи №1.

1.3. Реалізуємо алгоритм вбудовування даних в просторову область зображень методом блокового вбудовування. Для цього скористаємося наступною процедурою:

```
S1 := | for i ∈ 0..cols(R) − 1
      |   b ← mod( Σ_{j=0}^{rows(R)−1} R_{j,i}, 2 )
      |   if M_b_i ≠ b
      |     P ← D_B(R_{0,i})
      |     P_0 ← P_0 ⊕ 1
      |     S1_{0,i} ← B_D(P)
      |   S1_{0,i} ← R_{0,i} if M_b_i = b
      |   for j ∈ 1..rows(R) − 1
      |     S1_{j,i} ← R_{j,i}
      | S1
```

Наведена процедура реалізує поелементне вбудовування бітового масиву інформаційних даних M_b в біти парності окремих блоків зображення. При цьому зображення розбите на блоки по стовпчиках, тобто кожен стовпець масиву растрових даних каналу червоного кольору являє собою окремий блок зображення, в який вбудовується відповідний біт інформаційного повідомлення. Біт парності b для кожного блоку обчислюється у другому рядку процедури, за допомогою підсумовування за модулем два всіх елементів блоку. Якщо біт парності поточного блоку не збігається зі значенням вбудованого в даний блок інформаційного біта, проводиться модифікація (інвертування) найменш значущого біта в першому рядку блоку (наступні три рядки процедури). При цьому змінюється значення біта

парності, яке після виробленої модифікації збігається зі значенням вбудованого інформаційного біта даних. Якщо значення біта парності спочатку збігалося зі значенням вбудованого інформаційного біта даних, тоді проводиться перезапис першого елемента поточного блоку контейнера. Аналогічним чином перезаписуються і всі інші елементи блоку контейнера, що не модифікуються (останні рядки процедури).

Таким чином, вбудовування даних здійснюється в біти парності окремих блоків зображення, при цьому модифікуються перші елементи блоку. Результат вбудовування (заповнений контейнер) зберігається в масиві S1. Слід зазначити, що значення найменш значущого біта першого елемента блоку не завжди збігатиметься зі значенням вбудованого інформаційного біта. Збігаються лише біт парності та інформаційний біт даних.

Для візуального перегляду результату вбудовування інформаційних даних виведемо вихідний масив растрових даних червоного кольору R та отриманий масив S1 зі зміненими бітами парності блоків. Для розглянутого прикладу маємо:

R =

	0	1	2	3	4
0	86	79	72	72	72
1	110	97	90	86	78
2	132	120	112	105	96
3	122	116	105	104	103
4	131	122	117	118	118
5	147	147	148	148	150
6	169	164	167	170	173
7	189	195	193	189	183
8	191	192	194	199	194
9	186	188	194	198	192
10	195	196	199	200	201
11	185	189	202	203	203
12	192	196	198	199	204
13	177	185	187	186	180
14	173	176	174	166	165
15	160	162	158	153	…

S1 =

	0	1	2	3	4
0	87	79	72	72	72
1	110	97	90	86	78
2	132	120	112	105	96
3	122	116	105	104	103
4	131	122	117	118	118
5	147	147	148	148	150
6	169	164	167	170	173
7	189	195	193	189	183
8	191	192	194	199	194
9	186	188	194	198	192
10	195	196	199	200	201
11	185	189	202	203	203
12	192	196	198	199	204
13	177	185	187	186	180
14	173	176	174	166	165
15	160	162	158	153	…

З представлених даних видно, що, наприклад, значення $R_{0,1}$, $R_{0,2}$, $R_{0,3}$ повністю ідентичні відповідним значенням $S_{0,1}$, $S_{0,2}$, $S_{0,3}$. Практично це означає, що значення бітів парності першого, другого і третього стовпців масиву R збіглися з вбудованими інформаційними бітами повідомлення. Навпаки, значення $R_{0,0}$ відрізняється на одиницю від відповідного значення масиву S1. Це означає зміну біта парності нульового блоку контейнера в процесі вбудовування повідомлення. Відзначимо, що внесені спотворення знаходяться нижче порога зорової чутливості людини.

Графічна інтерпретація порожнього і заповненого контейнера (каналу червоного кольору в градаціях сірого) наведена на наступному рисунку, з

якого слідує, що візуально внесені спотворення не помітні, що підтверджує висновок про чутливість органів зору людини.

R

S1

Отриманий заповнений масив S1 записуємо в канал червоного кольору контейнера. Виконуємо команду

«WRITERGB("Stego_Blok.bmp"):=augment(S1,G,B)».

В результаті виконання команди система MathCAD формує на фізичному носії новий файл з ім'ям «Stego_Blok.bmp».

Для графічного відображення вихідного (порожнього) і заповненого контейнера виконаємо вставку відповідних зображень:

"1.bmp"

"Stego_Blok.bmp"

Переконуємося в відсутності видимих спотворень.

1.4. Реалізуємо алгоритм вилучення даних з просторової області зображень методом блочного вбудовування. Для цього в тому ж вікні середовища MathCAD виконуємо команди читання растрових даних нерухомого зображення з заданого файлу (файлу заповненого контейнера) у вигляді двовимірного масиву цілих чисел. Для розглянутого прикладу виконуємо команди:

«C1:=READRGB(“Stego_Blok.bmp”)», «R1:=READ_RED(“Stego_Blok.bmp”)»,
«G1:=READ_GREEN(“Stego_Blok.bmp”)»,
«B1:=READ_BLUE(“Stego_Blok.bmp”)».

Отримуємо наступний результат:

R1 =

	0	1	2	3	4
0	87	79	72	72	72
1	110	97	90	86	78
2	132	120	112	105	96
3	122	116	105	104	103
4	131	122	117	118	118
5	147	147	148	148	150
6	169	164	167	170	173
7	189	195	193	189	…

R1

Далі обчислюємо біти парності для кожного блоку даних контейнера і формуємо масив отриманих (витягнутих) інформаційних бітів. Для цього використовуємо наступну процедуру:

$$M\_b1 := \left| \begin{array}{l} \text{for } i \in 0..\text{cols}(R1) - 1 \\ \quad M\_b1_i \leftarrow \text{mod}\left(\sum_{j=0}^{\text{rows}(R)-1} R1_{j,i}, 2 \right) \\ M\_b1 \end{array} \right.$$

Дана процедура виконує для всіх стовпців (блоків) масиву R1 обчислення біта парності, який і є вбудованим інформаційним бітом. В результаті маємо лінійний бітовий масив M_b1, заповнений бітами парності окремих блоків масиву растрових даних каналу червоного кольору заповненого контейнера.

M_b1 =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	…

M_b =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	…

Порівняння даних масивів вбудованих та витягнутих (отриманих) бітів даних дозволяє підтвердити правильність виконання алгоритмів вбудовування-вилучення.

1.5., 1.6. Формування масиву цілих чисел, що відповідають ASCII кодуванню вбудованих символів повідомлення, та запис в текстовий файл витягнутих (отриманих) інформаційних даних здійснюється аналогічно п.1.5., 1.6. інструкції до лабораторної роботи №1.

Завдання 2. Реалізація в середовищі MathCAD алгоритмів вбудовування та вилучення повідомлень в просторовій області нерухомих зображень методом квантування

2.1., 2.2. Завантажуємо вихідні дані та перетворюємо масив інформаційних даних (згідно п. 1.1, 1.2).

2.3. Реалізуємо алгоритм вбудовування даних в просторову область зображень методом квантування. Для цього спочатку сформуємо випадковий секретний ключ - таблицю квантування d, скориставшись наступною процедурою:

```
d := | for i ∈ 0..510
     |     d[0,i] ← i − 255
     |     d[1,i] ← ceil(rnd(2)) − 1
     | d
```

Дана процедура псевдовипадковим чином (з використанням вбудованого датчика "rnd ()") заповнює таблицю квантування для всіх можливих значень перепадів яскравості зображення. Приклад заповненої таблиці має вигляд:

d =

	250	251	252	253	254	255	256	257	258	259	260
0	-5	-4	-3	-2	-1	0	1	2	3	4	5
1	0	0	0	1	0	1	1	1	0	1	...

Нульовий рядок масиву d заповнений усіма можливими (від -255 до +255) значеннями перепадів яскравості, перший рядок заповнений датчиком "rnd()". Функція "ceil()" округлює аргумент до найближчого цілого, функція "rnd()" формує рівномірно розподілені на заданій ділянці псевдовипадкові значення. Детальніше про використані функціях викладено в додатку.

Для вбудовування даних з використанням секретного ключа d скористаємося такою процедурою, яка реалізує поелементне вбудовування бітового масиву інформаційних даних M_b в значення різниць елементів нульового і першого стовпця масиву червоного кольору контейнера. Іншим словами кожен окремий біт вбудовується в одну різницю, номер біта задає номер рядка контейнера.

Поточне значення різниці b знаходиться в таблиці квантування. Значення вбудованого біта M_b_i порівнюється з бітовим значенням $d_{1,b+255}$ з другого рядка матриці квантування. При співпадінні цих значень рівень контрастності в даній позиції не змінюється.

```
S2 := | for i ∈ 0..rows(R) − 1
      |   b ← R_{i,0} − R_{i,1}
      |   S2_{i,0} ← R_{i,0}  if M_b_i = d_{1,b+255}
      |   if M_b_i ≠ d_{1,b+255}
      |     j ← 1
      |     while M_b_i ≠ d_{1,b+255+j} ∧ j < 509
      |       j ← j + 1
      |     S2_{i,0} ← R_{i,0} + d_{0,b+255+j} − b
      |   for j ∈ 1..cols(R) − 1
      |     S2_{i,j} ← R_{i,j}
      | S2
```

При неспівпадінні за заздалегідь заданим правилом (в даному випадку за правилом «пошук вправо») знаходиться найближча позиція, для якої значення M_b_i та $d_{1,b+255}$ збігаються (співпадають). Вбудовування інформації в такому випадку полягає в модифікації різниці (відповідно до знайдених значень з таблиці квантування). Інша частина зображення, яка не бере участі в модифікації різниці, перезаписується з пустого контейнера без зміни.

Таким чином, вбудовування даних здійснюється в значення різниці між окремими елементами масиву R. Результат вбудовування (заповнений контейнер) зберігається в масиві S2. Для візуального перегляду результату вбудовування інформаційних даних виведемо вихідний масив растрових даних червоного кольору R і отриманий масив S2 зі зміненими значеннями різниць. Для розглянутого прикладу маємо:

R =

	0	1	2	3	4
0	86	79	72	72	72
1	110	97	90	86	78
2	132	120	112	105	96
3	122	116	105	104	103
4	131	122	117	118	118
5	147	147	148	148	150
6	169	164	167	170	173
7	189	195	193	189	183
8	191	192	194	199	194
9	186	188	194	198	192
10	195	196	199	200	201
11	185	189	202	203	203
12	192	196	198	199	204
13	177	185	187	186	180
14	173	176	174	166	165
15	160	162	158	153	…

S2 =

	0	1	2	3	4
0	88	79	72	72	72
1	110	97	90	86	78
2	132	120	112	105	96
3	122	116	105	104	103
4	132	122	117	118	118
5	150	147	148	148	150
6	169	164	167	170	173
7	193	195	193	189	183
8	191	192	194	199	194
9	187	188	194	198	192
10	195	196	199	200	201
11	185	189	202	203	203
12	192	196	198	199	204
13	183	185	187	186	180
14	174	176	174	166	165
15	160	162	158	153	…

З представлених даних видно, що, наприклад, значення різниць

$$b = R_{1,0} - R_{1,1},\ b = R_{2,0} - R_{2,1},\ b = R_{3,0} - R_{3,1}$$

повністю ідентичні відповідним значенням різниць

$$b = S2_{1,0} - S2_{1,1},\ b = S2_{2,0} - S2_{2,1},\ b = S2_{3,0} - S2_{3,1}.$$

Практично це означає, що значення відповідних бітів з другого рядка таблиці квантування збіглося в цих позиціях зі значенням вбудованих інформаційних бітів даних. Навпаки, значення різниць

$$b = R_{0,0} - R_{0,1},\ b = R_{4,0} - R_{4,1},\ b = R_{5,0} - R_{5,1}$$

відрізняються від відповідних значень різниць

$$b = S2_{0,0} - S2_{0,1},\ b = S2_{4,0} - S2_{4,1},\ b = S2_{5,0} - S2_{5,1}.$$

Це означає зміну поточної різниці відповідно до знайдених значень в таблиці квантування. Так, наприклад, значення різниці

$$b = R_{5,0} - R_{5,1} = 147 - 147 = 0$$

було змінено на значення різниці

$$b = S2_{5,0} - S2_{5,1} = 150 - 147 = 3.$$

Як видно з наведеної вище таблиці квантування d, для значення різниці

$$b = d_{0,255} = 0$$

відповідне значення з другого рядка дорівнює

$$d_{1,255} = 1.$$

Для вбудовування інформаційного біта зі значенням «0» за правилом «пошук вправо» значення різниці модифікується на найближче знайдене справа значення, для якого значення з другого рядка таблиці квантування і значення вбудованого біта співпадуть. Очевидно, що це

$$d_{1,258} = 0$$

і маємо відповідне значення різниці

$$b = d_{0,258} = 3,$$

що повністю підтверджує правильність роботи алгоритму вбудовування.

Слід зазначити, що в запропонованій реалізації модифікація різниці досягається лише зміною елемента контейнера в нульовому стовпці, тобто за рахунок модифікації значень $S2_{i,0}$. Практично це означає, що всі спотворення будуть зосереджені в одному стовпці. Абсолютне значення внесених спотворень визначається статистичними властивостями використовуваної в якості секретного ключа псевдовипадкової послідовності, тобто другого рядка таблиці квантування. Для ефективних криптографічних генераторів з рівноймовірним розподілом формованих значень внесені спотворення знаходяться нижче порога зорової чутливості людини.

Графічна інтерпретація порожнього і заповненого контейнера (каналу червоного кольору в градаціях сірого) приведена на наступному рисунку, з якого слідує, що візуально внесені спотворення не помітні, що підтверджує висновок про чутливість органів зору людини до незначної зміни контрастності.

R

S2

Отриманий заповнений масив S2 записуємо в канал червоного кольору контейнера. Виконуємо команду

«WRITERGB("Stego_Kvant.bmp"):=augment(S2,G,B)».

В результаті виконання команди система MathCAD формує на фізичному носії новий файл з ім'ям «Stego_ Kvant.bmp».

Для графічного відображення вихідного (порожнього) і заповненого контейнера виконаємо вставку відповідних зображень:

"1.bmp"

"Stego_Kvant.bmp"

Переконуємося в відсутності видимих спотворень.

2.4. Реалізуємо алгоритм вилучення даних з просторової області зображень методом квантування. Для цього в тому ж вікні середовища MathCAD виконуємо команди читання растрових даних нерухомого зображення з заданого файлу (файлу заповненого контейнера) у вигляді двовимірного масиву цілих чисел. Для розглянутого прикладу виконуємо команди:

«C2:=READRGB(“Stego_Kvant.bmp”)»,
«R2:=READ_RED(“Stego_Kvant.bmp”)»,
«G2:=READ_GREEN(“Stego_Kvant.bmp”)»,
«B2:=READ_BLUE(“Stego_Kvant.bmp”)».

Отримаємо наступний результат:

R2 =

	0	1	2	3	4
0	88	79	72	72	72
1	110	97	90	86	78
2	132	120	112	105	96
3	122	116	105	104	103
4	132	122	117	118	118
5	150	147	148	148	150
6	169	164	167	170	173
7	193	195	193	189	…

R2

Далі обчислюємо біти значення різниці b між елементами перших двох стовпців і знаходимо відповідне бітове значення з другого рядка таблиці квантування. Для цього використовуємо наступну процедуру:

$$M\_b2 := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(R2) - 1 \\ \quad \left| \begin{array}{l} b \leftarrow R2_{i,0} - R2_{i,1} \\ M\_b2_i \leftarrow d_{1,b+255} \end{array} \right. \\ M\_b2 \end{array} \right.$$

Дана процедура виконує для всіх рядків масиву R2 обчислення значення різниці і відповідного йому біта даних, який і є вбудованим інформаційним бітом. В результаті маємо лінійний бітовий масив M_b2, заповнений відповідними бітами з другого рядка таблиці квантування.

M_b2 =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	…

M_b =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	…

Порівняння даних масивів вбудованих і витягнутих (отриманих) бітів даних дозволяє підтвердити правильність виконання алгоритмів вбудовування-вилучення.

2.5., 2.6. Формування масиву цілих чисел, що відповідають ASCII кодуванню вбудованих символів повідомлення і запис в текстовий файл витягнутих (отриманих) інформаційних даних здійснюється аналогічно п.1.5., 1.6.

Завдання 3. Реалізація в середовищі MathCAD алгоритмів вбудовування та вилучення повідомлень в просторовій області нерухомих зображень методом Куттера-Джордана-Боссена (методом «хреста»)

3.1., 3.2. Завантажуємо вихідні дані і перетворюємо масив інформаційних даних (згідно п. 1.1, 1.2).

3.3. Реалізуємо алгоритм вбудовування даних в просторову область зображень методом квантування. Для цього спочатку реалізуємо функцію обчислення яскравості окремого пікселя із заданими координатами:

$$\lambda(x,y) := 0.29890 R_{x,y} + 0.58662 G_{x,y} + 0.11448 B_{x,y}$$

і встановлюємо параметри методу квантування:

$$\gamma := 0.05 \qquad \sigma := 3$$

а також визначаємо функцію модифікації окремого пікселя в такий спосіб

$$SV(x,y,b) := \text{round}\left[B_{x,y} + (2\cdot b - 1)\cdot\gamma\cdot\lambda(x,y)\right]$$

Зрозуміло, що значення λ(x,y) визначається як повнокольорова яскравість пікселя по значеннях яскравості трьох колірних компонент з відповідним ваговим коефіцієнтом. Обрані параметри γ = 0,05 (енергія вбудованого біта) і σ = 3 (розмір області передбачення) є найпростішими показниками надійної роботи стеганоалгоритма.

Функція вбудовування SV(x,y,b) полягає в модифікації яскравості синього кольору заданого пікселя на частку його повноколірної яскравості, що задається параметром γ.

Для прикладу, покажемо правильність роботи функції вбудовування.

Значення яскравості пікселя з координатами (7,7) відповідає

$$\lambda(7,7) = 176{,}42.$$

Відповідне значення яскравості синього кольору пікселя

$$B(7,7) = 208.$$

Можлива модифікація яскравості синього кольору пікселя згідно з функцією вбудовування приймає значення $208 \pm [0{,}05\cdot 176{,}42] = 208 \pm 9$. Зрозуміло, що алгоритм обчислення функції SV(7,7,0) = 199 працює правильно.

Для вбудовування масиву інформаційних даних M_b скористаємося такою процедурою. Вона реалізує поелементне вбудовування бітового масиву в значення яскравостей синього кольору за допомогою модифікації функцією вбудовування SV(x,y,b).

Областю для вбудовування обрано діагональ масиву яскравостей синього кольору контейнера. Інші елементи контейнера (не з області модифікації) підлягають перезапису з пустого контейнера.

```
S3 := | for i ∈ 0..cols(B) − 1
      |   for j ∈ 0..rows(B) − 1
      |     S3_{j,i} ← B_{j,i}
      | for i ∈ σ..rows(B) − σ − 1
      |   b ← SV(i, i, M_b_{i−σ})
      |   S3_{i,i} ← b   if 0 ≤ b ≤ 255
      |   S3_{i,i} ← 255 if b > 255
      |   S3_{i,i} ← 0   if b < 0
      | S3
```

Слід зазначити, що вбудовування починається не з пікселя з координатами (0, 0), а з пікселя, що має координати (σ,σ). Це виконано для можливості в подальшому здійснити передбачення методом «хреста».

Для візуального перегляду результату вбудовування інформаційних даних виведемо вихідний масив растрових даних синього кольору B і отриманий масив S2 зі зміненими значеннями різниць. Для розглянутого прикладу маємо:

S3 =

	0	1	2	3	4	5
0	135	128	123	122	119	124
1	155	142	141	134	133	122
2	174	159	151	152	141	136
3	162	160	151	151	147	147
4	172	161	159	164	164	160
5	184	181	190	184	183	183
6	198	197	200	203	206	207
7	225	222	226	222	218	206
8	223	226	222	224	219	215
9	220	221	230	229	224	221
10	224	228	231	225	229	221
11	221	217	227	231	232	233
12	222	224	228	230	231	231
13	215	211	218	217	211	208
14	209	207	204	198	197	197
15	200	196	192	190	189	…

B =

	0	1	2	3	4	5
0	135	128	123	122	119	124
1	155	142	141	134	133	122
2	174	159	151	152	141	136
3	162	160	151	145	147	147
4	172	161	159	164	158	160
5	184	181	190	184	183	191
6	198	197	200	203	206	207
7	225	222	226	222	218	206
8	223	226	222	224	219	215
9	220	221	230	229	224	221
10	224	228	231	225	229	221
11	221	217	227	231	232	233
12	222	224	228	230	231	231
13	215	211	218	217	211	208
14	209	207	204	198	197	197
15	200	196	192	190	189	…

З представлених даних видно, що, наприклад, значення

$$S3_{3,3} > B_{3,3},\ S3_{4,4} > B_{4,4},$$

що відповідає вбудовуванню «1» в ці позиції.

Значення $S3_{5,5} < B_{5,5}$ відповідає вбудовуванню «0».

Слід зазначити, що величина внесених спотворень визначається введеним значенням $\gamma = 0{,}05$ (енергія вбудованого біта), як частки повноколірної яскравості пікселя, що припадає на модифікацію яскравості синього кольору.

Графічна інтерпретація порожнього і заповненого контейнера (каналу синього кольору в градаціях сірого) приведена на наступному рисунку, з якого видно, що візуально внесені спотворення не помітні, що підтверджує висновок про чутливість органів зору людини до незначної зміни синього кольору.

S3

B

Отриманий заповнений масив S3 записуємо в канал синього кольору контейнера. Виконуємо команду

«WRITERGB("Stego_Krest.bmp"):=augment(R,G,S3)».

В результаті виконання команди система MathCAD формує на фізичному носії новий файл з ім'ям «Stego_Krest.bmp».

Для графічного відображення вихідного (порожнього) і заповненого контейнера виконаємо вставку відповідних зображень:

"1.bmp"

"Stego_Krest.bmp"

Переконуємося в відсутності видимих спотворень.

3.4. Реалізуємо алгоритм вилучення даних з просторової області зображень методом «хреста». Для цього в тому ж вікні середовища MathCAD виконуємо команди читання растрових даних нерухомого зображення з заданого файлу (файлу заповненого контейнера) у вигляді двовимірного масиву цілих чисел. Для розглянутого прикладу виконуємо команди:

«C3:=READRGB(“Stego_Krest.bmp ”)»,
«R3:=READ_RED(“Stego_Krest.bmp”)»,
«G3:=READ_GREEN(“Stego_Krest.bmp”)»,
«B3:=READ_BLUE(“Stego_Krest.bmp”)».

Отримуємо наступний результат:

B3 =

	0	1	2	3	4
0	135	128	123	122	119
1	155	142	141	134	133
2	174	159	151	152	141
3	162	160	151	151	147
4	172	161	159	164	164
5	184	181	190	184	183
6	198	197	200	203	206
7	225	222	226	222	…

B3

Далі, для кожного вбудованого біта інформації обчислюємо передбачене значення b синього кольору і порівнюємо зі стостережувальним значенням $B3_{i,i}$. Використовуємо таку процедуру:

$$M\_b3 := \left| \begin{array}{l} \text{for } i \in \sigma..\text{rows}(B3) - \sigma - 1 \\ \quad \left| \begin{array}{l} b \leftarrow \dfrac{\left(\displaystyle\sum_{j=i-\sigma}^{i-1} B3_{i,j} + \sum_{j=i-\sigma}^{i-1} B3_{j,i} + \sum_{j=i+1}^{i+\sigma} B3_{i,j} + \sum_{j=i+1}^{i+\sigma} B3_{j,i} \right)}{4\sigma} \\ M\_b3_{i-\sigma} \leftarrow 1 \text{ if } b < B3_{i,i} \\ M\_b3_{i-\sigma} \leftarrow 0 \text{ if } b > B3_{i,i} \end{array} \right. \\ M\_b3 \end{array} \right.$$

В результаті порівняння b зі спостережуваним значенням $B3_{i,i}$ приймаємо рішення про значення вбудованого біта інформації. Виконання передбачення про значення яскравості синього кольору виронуємо для кожного пікселя, що підлягав модифікації. Позиція (координати) модифікованого пікселя є секретною ключовою інформацією.

Наведемо результат роботи даної процедури вилучення даних для розглянутого прикладу.

M_b3 =

	0
0	0
1	0
2	1
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	0
13	1
14	1
15	…

M_b =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	0
13	1
14	1
15	…

Зрозуміло, що результат вилучення перших трьох бітів є неправильним. Наставання такої події не виключено логікою алгоритму вилучення, ймовірність його виникнення визначається статистичними властивостями контейнера. Підвищити ймовірність правильного вилучення інформаційних бітів даних можна за рахунок підвищення енергії вбудованих бітів даних, тобто за допомогою збільшення коефіцієнта γ. Однак подібна процедура неминуче призведе до збільшення внесених спотворень в контейнер-зображення.

3.5., 3.6. Формування масиву цілих чисел, що відповідають ASCII кодуванню вбудованих символів повідомлення і запис в текстовий файл отриманих інформаційних даних здійснюється аналогічно п. 1.5., 1.6.

Завдання 4. Дослідження ймовірносних характеристик стеганографічного методу вбудовування даних Куттера-Джордана-Боссена (методу «хреста»)

4.1. Проведемо оцінку ймовірності правильного вилучення повідомлення і величини внесених спотворень від коефіцієнта γ. Для цього будемо послідовно збільшувати величину γ і для кожного значення розраховувати частоту v правильно отриманих інформаційних бітів. Одночасно будемо розраховувати усереднену величину w внесених спотворень, виражену у відсотковому співвідношенні до максимального значення яскравості. Використаємо для цього наступні процедури:

$$v := \left| \begin{array}{l} v \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(M\_b3) - 1 \\ \quad v \leftarrow v + 1 \text{ if } M\_b3_i = M\_b_i \\ v \leftarrow \dfrac{v}{\text{rows}(M\_b3)} \\ v \end{array} \right.$$

$$w := \left| \begin{array}{l} w \leftarrow 0 \\ \text{for } i \in \sigma..\text{rows}(B3) - \sigma - 1 \\ \quad w \leftarrow w + \left| B3_{i,i} - B_{i,i} \right| \\ w \leftarrow \dfrac{w \cdot 100}{\text{rows}(M\_b3) \cdot 256} \\ w \end{array} \right.$$

Так, для розглянутого прикладу при γ = 0,45 маємо наступні значення:

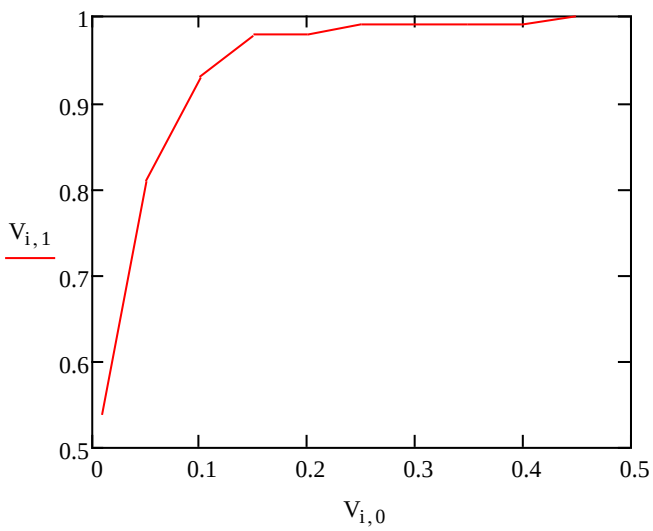
v = 1 w = 20.809

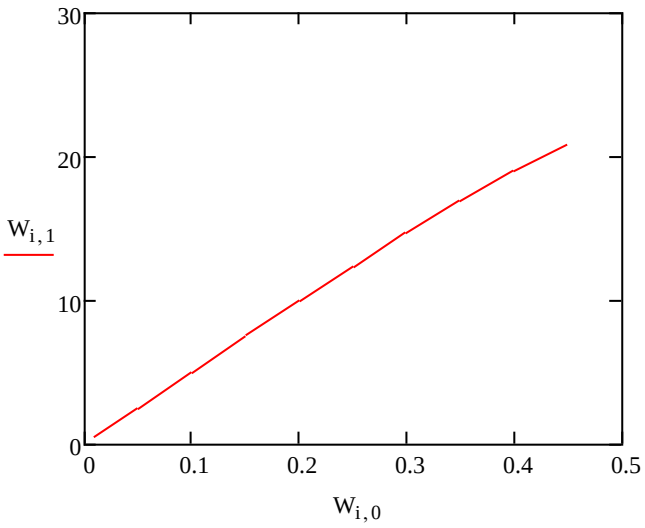
Отримані емпіричні дані занесемо до відповідних таблиць:

$$V := \begin{pmatrix} 0.01 & 0.54 \\ 0.05 & 0.81 \\ 0.1 & 0.93 \\ 0.15 & 0.98 \\ 0.2 & 0.98 \\ 0.25 & 0.99 \\ 0.3 & 0.99 \\ 0.35 & 0.99 \\ 0.4 & 0.99 \\ 0.45 & 1 \end{pmatrix} \qquad W := \begin{pmatrix} 0.01 & 0.55 \\ 0.05 & 2.55 \\ 0.1 & 5.0 \\ 0.15 & 7.6 \\ 0.2 & 10 \\ 0.25 & 12.4 \\ 0.3 & 14.7 \\ 0.35 & 16.9 \\ 0.4 & 19 \\ 0.45 & 20.8 \end{pmatrix}$$

4.2. Побудуємо графіки отриманих емпіричних залежностей:

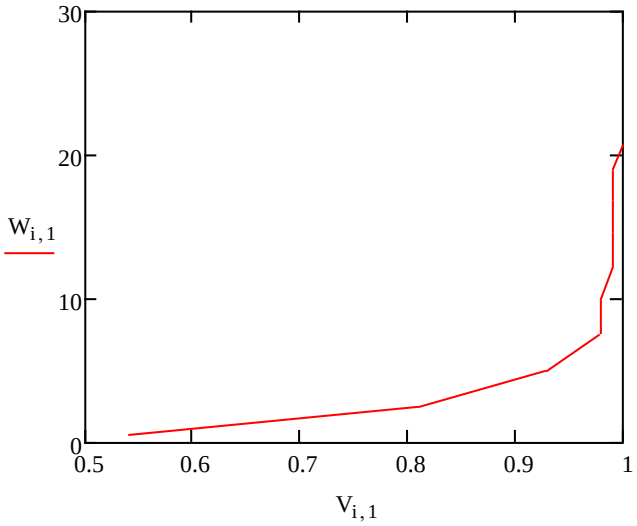
$$i := 0..9$$





Зрозуміло, що величина внесених спотворень зростає лінійно від коефіцієнта γ. Однак емпірична залежність ймовірності правильного вилучення інформаційних даних поводиться інакше. При малих значеннях коефіцієнта γ величина W зростає швидко, однак при γ> 0,2 подальше збільшення енергії вбудовування не призводить до суттєвого підвищення ймовірності правильного вилучення, підвищувати величину V в даному випадку недоцільно.

4.3. Побудуємо інтегральний графік залежності величини W внесених спотворень в контейнер-зображення при забезпеченні відповідної ймовірності V правильного вилучення інформаційних даних:



Зрозуміло, що ефективне приховування вбудованих інформаційних даних без внесення значних спотворень (W <5%) в контейнер-зображення буде спостерігатися тільки при ймовірності правильного вилучення даних V<0,8 ... 0,9, що відповідає енергії вбудовування γ = 0,05 ... 0,15. Підвищення достовірності отриманих даних за рахунок подальшого збільшення енергії вбудовування є недоцільним, оскільки це призводить до внесення невиправдано високих спотворень в контейнер-зображення. В даному випадку найбільш перспективним є реалізація завадостійкого кодування інформаційних даних і контроль помилок, що виникають при стеганографічних перетвореннях.

Завдання 5 (додаткове). Реалізація завадостійкого кодування інформаційних даних для підвищення ймовірносних характеристик стеганографічного методу вбудовування даних Куттера-Джордана-Боссена (методу «хреста»)

5.1. Реалізуємо завадостійке кодування найпростішим лінійним блоковим кодом Хеммінга. Для цього введемо такі породжуючи та перевірочну матриці

$$Gen := \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \qquad H := \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

5.2. Реалізуємо алгоритми кодування і декодування окремих кодових слів. Для цього скористаємося наступними функціями

```
cod(inf) := | for i ∈ 0..6
           |   c_i ← 0
           |   for j ∈ 0..3
           |     c_i ← (c_i) ⊕ (inf_j · Gen_j,i)
           | c
```

```
decod(c) := | for i ∈ 0..2
            |   s_i ← 0
            |   for j ∈ 0..6
            |     s_i ← (s_i) ⊕ (c_j · H_i,j)
            | ss ← s_0 + s_1·2 + s_2·4
            | cc ← c
            | cc_4 ← (c_4) ⊕ 1 if ss = 1
            | cc_5 ← (c_5) ⊕ 1 if ss = 2
            | cc_2 ← (c_2) ⊕ 1 if ss = 3
            | cc_6 ← (c_6) ⊕ 1 if ss = 4
            | cc_0 ← (c_0) ⊕ 1 if ss = 5
            | cc_3 ← (c_3) ⊕ 1 if ss = 6
            | cc_1 ← (c_1) ⊕ 1 if ss = 7
            | cc
```

Як приклад розглянемо інформаційний вектор:

$$inf := \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

Після виконання функції кодування маємо наступне кодове слово:

$$\underset{\sim}{c} := cod(inf) \qquad c = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

Внесемо помилку в довільному кодовому символі, наприклад, $c_3 := 1$

Маємо наступне слово з помилкою, яке після декодування відновлюється в безпомилкову послідовність:

$$c = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} \qquad decod(c) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

5.3. Реалізуємо алгоритм завадостійкого кодування масиву інформаційних даних:

```
M_b_cod := | for i ∈ 0..ceil(rows(M_b)/4) - 1
           |   for j ∈ 0..3
           |     inf_j ← M_b_(4·i+j)
           |   c ← cod(inf)
           |   for l ∈ 0..6
           |     M_b_cod_(7·i+l) ← c_l
           | M_b_cod
```

Для розглянутого прикладу маємо:

M_b =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	0
13	1
14	1
15	…

M_b_cod =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	0
8	0
9	1
10	1
11	1
12	0
13	1
14	0
15	…

Реалізована процедура зчитує по чотири біти з масиву M_b і кодує їх завадостійким кодом Хеммінга. Результат кодування блоками по сім бітів записується в масив M_b_cod.

5.4. Реалізуємо вбудовування сформованих даних в контейнер-зображення методом «хреста». Для цього скористаємося розглянутими в п. 3.3. процедурами:

γ := 0.05

$$SV(x,y,b) := \mathrm{round}\left[B_{x,y} + (2\cdot b - 1)\cdot\gamma\cdot\lambda(x,y)\right]$$

```
S4 := | for i ∈ 0..cols(B) − 1
      |   for j ∈ 0..rows(B) − 1
      |     S4[j,i] ← B[j,i]
      | for i ∈ σ..rows(B) − σ − 1
      |   b ← SV(i, i, M_b_cod[i−σ])
      |   S4[i,i] ← b   if 0 ≤ b ≤ 255
      |   S4[i,i] ← 255 if b > 255
      |   S4[i,i] ← 0   if b < 0
      | S4
```

В результаті формуємо масив S4 синього кольору з вбудованими даними. Сформуємо заповнений контейнер і переглянемо результат:

WRITERGB("Stego_Krest_cod.bmp") := augment(R, G, S4)

S4

B

"1.bmp"

"Stego_Krest_cod.bmp"

5.5. Реалізуємо витяг сформованих даних в контейнер-зображення методом «хреста». Для цього скористаємося розглянутими в п. 3.4. процедурами:

B4 := READ_BLUE("Stego_Krest_cod.bmp")

B4 =

	0	1	2	3	4
0	135	128	123	122	119
1	155	142	141	134	133
2	174	159	151	152	141
3	162	160	151	151	147
4	172	161	159	164	164
5	184	181	190	184	183
6	198	197	200	203	206
7	225	222	226	222	…

B4

M_b4 :=
for $i \in \sigma..rows(B4) - \sigma - 1$

$$b \leftarrow \frac{\left(\sum_{j=i-\sigma}^{i-1} B4_{i,j} + \sum_{j=i-\sigma}^{i-1} B4_{j,i} + \sum_{j=i+1}^{i+\sigma} B4_{i,j} + \sum_{j=i+1}^{i+\sigma} B4_{j,i}\right)}{4\sigma}$$

$M\_b4_{i-\sigma} \leftarrow 1 \text{ if } b < B4_{i,i}$

$M\_b4_{i-\sigma} \leftarrow 0 \text{ if } b > B4_{i,i}$

M_b4

В результаті виконання розглянутих процедур формуємо масив витягнутих даних:

M_b4 =

	0
0	0
1	0
2	1
3	1
4	0
5	0
6	1
7	0
8	0
9	1
10	1
11	1
12	0
13	1
14	0
15	…

M_b_cod =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	0
8	0
9	1
10	1
11	1
12	0
13	1
14	0
15	…

5.6. Реалізуємо завадостійке декодування витягнутих даних:

```
M_b_decod := | for i ∈ 0..floor(rows(M_b4)/7) − 1
             |   for j ∈ 0..6
             |     c_j ← M_b4_(7·i+j)
             |   c ← decod(c)
             |   for l ∈ 0..3
             |     M_b_decod_(4·i+l) ← c_l
             | M_b_decod
```

Наведена процедура зчитує витягнуті дані блоками по сім бітів і декодує їх розглянутою в п. 5.2. функцією. В результаті формуємо масив витягнутих даних з виправленими помилками.

M_b =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	0
13	1
14	1
15	…

M_b_decod =

	0
0	0
1	0
2	1
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	1
13	1
14	0
15	…

5.7. Дослідимо ймовірносні характеристики методу «хреста» з використанням завадостійкого кодування. Для цього скористаємося розглянутими в п. 4.1. - 4.3. процедурами:

```
v := | v ← 0
     | for i ∈ 0..rows(M_b_decod) − 1
     |   v ← v + 1 if M_b_decod_i = M_b_i
     | v ← v / rows(M_b_decod)
     | v
```

v = 0.804

```
w := | w ← 0
     | for i ∈ σ..rows(B4) − σ − 1
     |   w ← w + |B4_(i,i) − B_(i,i)|
     | w ← w·100 / (rows(M_b3)·256)
     | w
```

w = 2.55

Отримані емпіричні дані занесемо до відповідних таблиць і порівняємо з вже наявними залежностями:

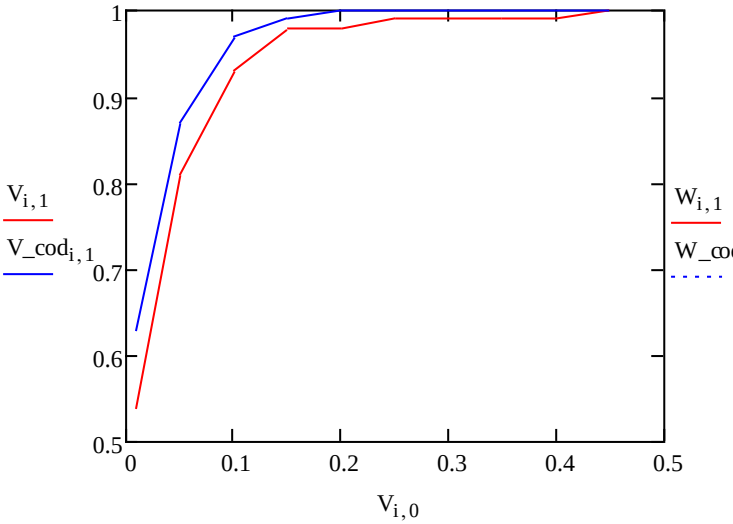
$$V := \begin{pmatrix} 0.01 & 0.54 \\ 0.05 & 0.81 \\ 0.1 & 0.93 \\ 0.15 & 0.98 \\ 0.2 & 0.98 \\ 0.25 & 0.99 \\ 0.3 & 0.99 \\ 0.35 & 0.99 \\ 0.4 & 0.99 \\ 0.45 & 1 \end{pmatrix} \quad V\_cod := \begin{pmatrix} 0.01 & 0.63 \\ 0.05 & 0.87 \\ 0.1 & 0.97 \\ 0.15 & 0.99 \\ 0.2 & 1 \\ 0.25 & 1 \\ 0.3 & 1 \\ 0.35 & 1 \\ 0.4 & 1 \\ 0.45 & 1 \end{pmatrix}$$

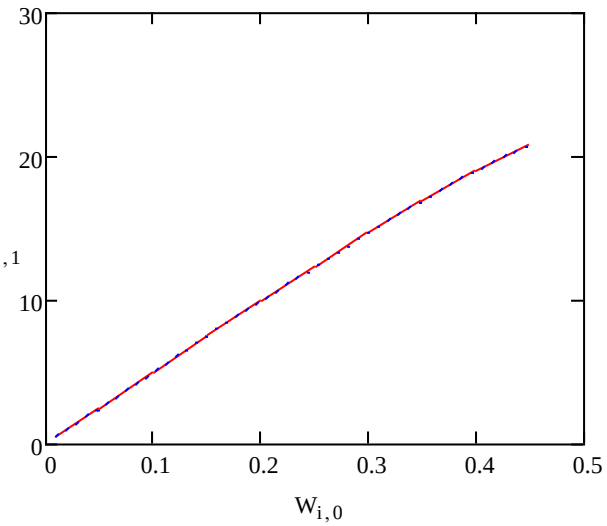
$$W := \begin{pmatrix} 0.01 & 0.55 \\ 0.05 & 2.55 \\ 0.1 & 5.0 \\ 0.15 & 7.6 \\ 0.2 & 10 \\ 0.25 & 12.4 \\ 0.3 & 14.7 \\ 0.35 & 16.9 \\ 0.4 & 19 \\ 0.45 & 20.8 \end{pmatrix} \quad W\_cod := \begin{pmatrix} 0.01 & 0.55 \\ 0.05 & 2.55 \\ 0.1 & 5.0 \\ 0.15 & 7.6 \\ 0.2 & 10 \\ 0.25 & 12.4 \\ 0.3 & 14.7 \\ 0.35 & 16.9 \\ 0.4 & 19 \\ 0.45 & 20.8 \end{pmatrix}$$

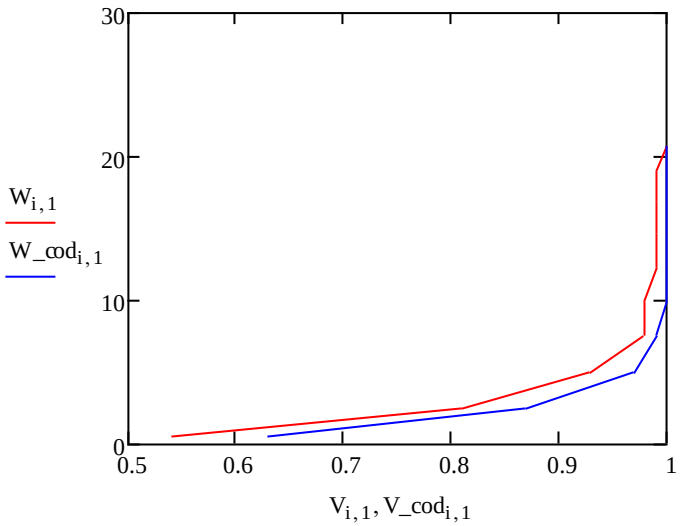
Зрозуміло, що використання завадостійкого кодування дозволило незначно підвищити ймовірність правильного вилучення вбудованих даних.

Побудуємо відповідні графіки:

$i := 0..9$







Отримані залежності показують, що використання навіть найпростішого завадостійкого коду Хеммінга (синя крива) дозволяє поліпшити ймовірнісні характеристики методу «хреста». Рівень внесених спотворень при цьому не змінюється. У той же час використання коду Хеммінга призвело до зниження практично в два рази обсягу вбудованих інформаційних даних. Для практичного використання доцільно застосування потужних завадостійких кодів, які дозволять домогтися безпомилкового вилучення при незначному зниженні обсягу вбудованих інформаційних даних

6. Приклад оформлення звіту з лабораторної роботи №2

Лабораторна робота №2
Приховування даних в просторовій області зображень методом блокового приховування, методом квантування, методом «хреста»

"1.bmp"

C := READRGB("1.bmp")

R := READ_RED("1.bmp")

G := READ_GREEN("1.bmp")

B := READ_BLUE("1.bmp")

M := READBIN("2.txt","byte")

R

R =

	0	1	2	3	4	5	6	7
0	86	79	72	72	72	69	71	74
1	110	97	90	86	78	71	70	70
2	132	120	112	105	96	88	81	83
3	122	116	105	104	103	102	101	99
4	131	122	117	118	118	116	105	107
5	147	147	148	148	150	153	145	135
6	169	164	167	170	173	175	164	155
7	189	195	193	189	183	172	173	173
8	191	192	194	199	194	187	182	182
9	186	188	194	198	192	187	187	180
10	195	196	199	200	201	190	192	186
11	185	189	202	203	203	199	203	199
12	192	196	198	199	204	202	206	199
13	177	185	187	186	180	178	179	177
14	173	176	174	166	165	165	163	161
15	160	162	158	153	156	158	157	...

$$B\_D(x) := \sum_{i=0}^{7} \left(x_i \cdot 2^i \right)$$

D_B(x) :=
for $i \in 0..7$
  $V_i \leftarrow \mathrm{mod}(x, 2)$
  $x \leftarrow \mathrm{floor}\left(\frac{x}{2}\right)$
V

M_b :=
for $i \in 0..\mathrm{rows}(M) - 1$
  $V \leftarrow D\_B(M_i)$
  for $j \in 0..7$
    $M\_b_{i \cdot 8 + j} \leftarrow V_j$
M_b

M_b =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	0
13	1
14	1
15	...

S1 :=
for $i \in 0..\mathrm{cols}(R) - 1$
  $b \leftarrow \mathrm{mod}\left(\sum_{j=0}^{\mathrm{rows}(R)-1} R_{j,i}, 2 \right)$
  if $M\_b_i \neq b$
    $P \leftarrow D\_B(R_{0,i})$
    $P_0 \leftarrow P_0 \oplus 1$
    $S1_{0,i} \leftarrow B\_D(P)$
  $S1_{0,i} \leftarrow R_{0,i}$ if $M\_b_i = b$
  for $j \in 1..\mathrm{rows}(R) - 1$
    $S1_{j,i} \leftarrow R_{j,i}$
S1

M =

	0
0	203
1	224
2	225
3	238
4	240
5	224
6	242
7	238
8	...

WRITERGB("Stego_Blok.bmp") := augment(S1, G, B)

S1 =

	0	1	2	3	4
0	87	79	72	72	72
1	110	97	90	86	78
2	132	120	112	105	96
3	122	116	105	104	103
4	131	122	117	118	118
5	147	147	148	148	150
6	169	164	167	170	173
7	189	195	193	189	183
8	191	192	194	199	194
9	186	188	194	198	192
10	195	196	199	200	201
11	185	189	202	203	203
12	192	196	198	199	204
13	177	185	187	186	180
14	173	176	174	166	165
15	160	162	158	153	…

R =

	0	1	2	3	4
0	86	79	72	72	72
1	110	97	90	86	78
2	132	120	112	105	96
3	122	116	105	104	103
4	131	122	117	118	118
5	147	147	148	148	150
6	169	164	167	170	173
7	189	195	193	189	183
8	191	192	194	199	194
9	186	188	194	198	192
10	195	196	199	200	201
11	185	189	202	203	203
12	192	196	198	199	204
13	177	185	187	186	180
14	173	176	174	166	165
15	160	162	158	153	…

S1

R

"Stego_Blok.bmp"

"1.bmp"

C1 := READRGB("Stego_Blok.bmp")

R1 := READ_RED("Stego_Blok.bmp")

G1 := READ_GREEN("Stego_Blok.bmp")

B1 := READ_BLUE("Stego_Blok.bmp")

R1 =

	0	1	2	3	4
0	87	79	72	72	72
1	110	97	90	86	78
2	132	120	112	105	96
3	122	116	105	104	103
4	131	122	117	118	118
5	147	147	148	148	150
6	169	164	167	170	173
7	189	195	193	189	…

R1

$$M\_b1 := \left| \begin{array}{l} \text{for } i \in 0..\text{cols}(R1) - 1 \\ \quad M\_b1_i \leftarrow \text{mod}\left(\sum_{j=0}^{\text{rows}(R)-1} R1_{j,i}, 2 \right) \\ M\_b1 \end{array} \right.$$

M_b1 =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	…

M_b =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	…

Метод квантування

$$d := \left| \begin{array}{l} \text{for } i \in 0..510 \\ \quad d_{0,i} \leftarrow i - 255 \\ \quad d_{1,i} \leftarrow \text{ceil}(\text{rnd}(2)) - 1 \\ d \end{array} \right.$$

d =

	250	251	252	253	254	255	256	257	258
0	-5	-4	-3	-2	-1	0	1	2	…

$$S2 := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(R) - 1 \\ \quad b \leftarrow R_{i,0} - R_{i,1} \\ \quad S2_{i,0} \leftarrow R_{i,0} \text{ if } M\_b_i = d_{1,b+255} \\ \quad \text{if } M\_b_i \neq d_{1,b+255} \\ \quad\quad j \leftarrow 1 \\ \quad\quad \text{while } M\_b_i \neq d_{1,b+255+j} \wedge j < 509 \\ \quad\quad\quad j \leftarrow j + 1 \\ \quad\quad S2_{i,0} \leftarrow R_{i,0} + d_{0,b+255+j} - b \\ \quad \text{for } j \in 1..\text{cols}(R) - 1 \\ \quad\quad S2_{i,j} \leftarrow R_{i,j} \\ S2 \end{array} \right.$$

S2 =

	0	1	2	3	4
0	88	79	72	72	72
1	110	97	90	86	78
2	132	120	112	105	96
3	122	116	105	104	103
4	132	122	117	118	118
5	150	147	148	148	150
6	169	164	167	170	173
7	193	195	193	189	183
8	191	192	194	199	194
9	187	188	194	198	192
10	195	196	199	200	201
11	185	189	202	203	203
12	192	196	198	199	204
13	183	185	187	186	180
14	174	176	174	166	165
15	160	162	158	153	…

WRITERGB("Stego_Kvant.bmp") := augment(S2, G, B)

S2

R

"1.bmp"

"Stego_Kvant.bmp"

C2 := READRGB("Stego_Kvant.bmp")

R2 := READ_RED("Stego_Kvant.bmp")

G2 := READ_GREEN("Stego_Kvant.bmp")

B2 := READ_BLUE("Stego_Kvant.bmp")

R2 =

	0	1	2	3	4
0	88	79	72	72	72
1	110	97	90	86	78
2	132	120	112	105	96
3	122	116	105	104	103
4	132	122	117	118	118
5	150	147	148	148	150
6	169	164	167	170	173
7	193	195	193	189	…

M_b2 :=
for $i \in 0..rows(R2) - 1$
  $b \leftarrow R2_{i,0} - R2_{i,1}$
  $M\_b2_i \leftarrow d_{1,b+255}$
M_b2

R2

M_b2 =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	0
13	1
14	1
15	…

M_b =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	0
13	1
14	1
15	…

Метод «хреста»

$\lambda(x,y) := 0.29890 R_{x,y} + 0.58662 G_{x,y} + 0.11448 B_{x,y}$ $\gamma := 0.0$ $\sigma := 3$

$SV(x,y,b) := \text{round}\left[B_{x,y} + (2\cdot b - 1)\cdot\gamma\cdot\lambda(x,y)\right]$

Приклад: $\lambda(7,7) = 176.42$ $B_{7,7} = 208$ $SV(7,7,0) = 199$

$S3 :=$ for $i \in 0..\text{cols}(B) - 1$
  for $j \in 0..\text{rows}(B) - 1$
    $S3_{j,i} \leftarrow B_{j,i}$
  for $i \in \sigma..\text{rows}(B) - \sigma - 1$
    $b \leftarrow SV\left(i, i, M\_b_{i-\sigma}\right)$
    $S3_{i,i} \leftarrow b$ if $0 \le b \le 255$
    $S3_{i,i} \leftarrow 255$ if $b > 255$
    $S3_{i,i} \leftarrow 0$ if $b < 0$
  S3

$S3 =$

	0	1	2	3	4	5
0	135	128	123	122	119	124
1	155	142	141	134	133	122
2	174	159	151	152	141	136
3	162	160	151	151	147	147
4	172	161	159	164	164	160
5	184	181	190	184	183	183
6	198	197	200	203	206	207
7	225	222	226	222	218	206
8	223	226	222	224	219	215
9	220	221	230	229	224	221
10	224	228	231	225	229	221
11	221	217	227	231	232	233
12	222	224	228	230	231	231
13	215	211	218	217	211	208
14	209	207	204	198	197	197
15	200	196	192	190	189	…

$WRITERGB("Stego\_Krest.bmp") := augment(R, G, S3)$

S3

B

"1.bmp"

"Stego_Krest.bmp"

C3 := READRGB("Stego_Krest.bmp")

R3 := READ_RED("Stego_Krest.bmp")

G3 := READ_GREEN("Stego_Krest.bmp")

B3 := READ_BLUE("Stego_Krest.bmp")

B3

B3 =

	0	1	2	3	4
0	135	128	123	122	119
1	155	142	141	134	133
2	174	159	151	152	141
3	162	160	151	151	147
4	172	161	159	164	164
5	184	181	190	184	183
6	198	197	200	203	206
7	225	222	226	222	…

M_b3 :=
for $i \in \sigma..rows(B3) - \sigma - 1$

$$b \leftarrow \frac{\left(\sum_{j=i-\sigma}^{i-1} B3_{i,j} + \sum_{j=i-\sigma}^{i-1} B3_{j,i} + \sum_{j=i+1}^{i+\sigma} B3_{i,j} + \sum_{j=i+1}^{i+\sigma} B3_{j,i}\right)}{4\sigma}$$

$M\_b3_{i-\sigma} \leftarrow 1$ if $b < B3_{i,i}$

$M\_b3_{i-\sigma} \leftarrow 0$ if $b > B3_{i,i}$

M_b3

M_b3 =

	0
0	0
1	0
2	1
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	0
13	1
14	1
15	…

M_b =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	0
13	1
14	1
15	…

Дослідження ймовірносних характеристик

v :=
v ← 0

for $i \in 0..rows(M\_b3) - 1$

$v \leftarrow v + 1$ if $M\_b3_i = M\_b_i$

$$v \leftarrow \frac{v}{rows(M\_b3)}$$

v

w :=
w ← 0

for $i \in \sigma..rows(B3) - \sigma - 1$

$w \leftarrow w + \left|B3_{i,i} - B_{i,i}\right|$

$$w \leftarrow \frac{w \cdot 100}{rows(M\_b3) \cdot 256}$$

w

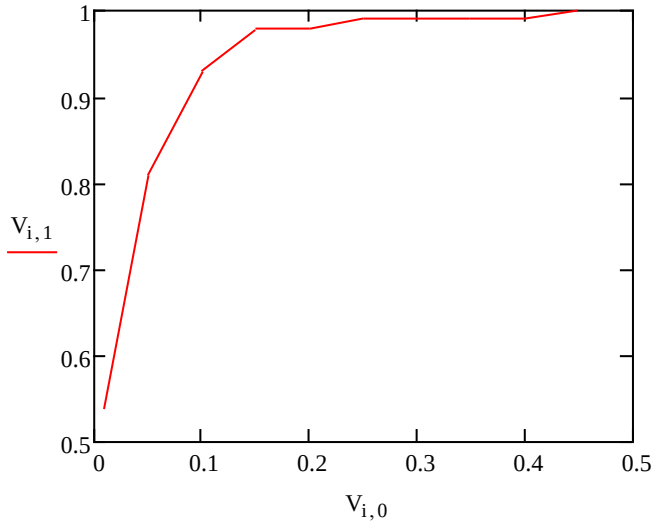
v = 0.81

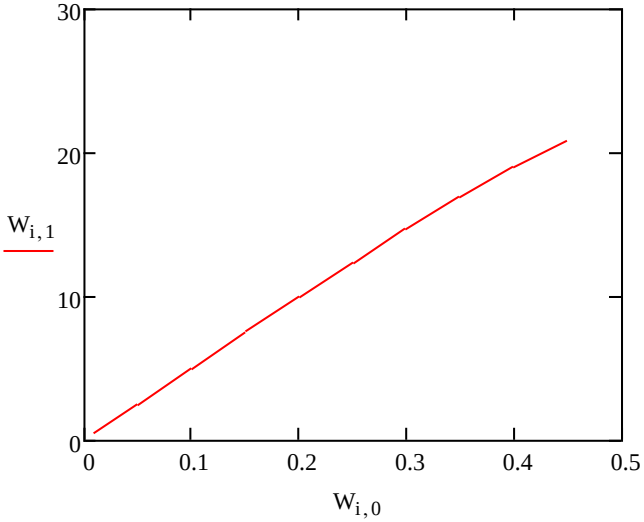
w = 2.55

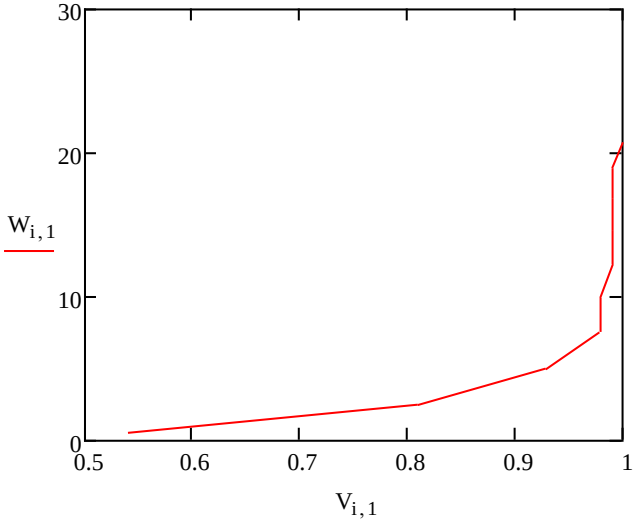
V :=
$$\begin{pmatrix} 0.01 & 0.54 \\ 0.05 & 0.81 \\ 0.1 & 0.93 \\ 0.15 & 0.98 \\ 0.2 & 0.98 \\ 0.25 & 0.99 \\ 0.3 & 0.99 \\ 0.35 & 0.99 \\ 0.4 & 0.99 \\ 0.45 & 1 \end{pmatrix}$$

W :=
$$\begin{pmatrix} 0.01 & 0.55 \\ 0.05 & 2.55 \\ 0.1 & 5.0 \\ 0.15 & 7.6 \\ 0.2 & 10 \\ 0.25 & 12.4 \\ 0.3 & 14.7 \\ 0.35 & 16.9 \\ 0.4 & 19 \\ 0.45 & 20.8 \end{pmatrix}$$

$i := 0..9$







Завадостійке кодування інформаційних даних

$$Gen := \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

$$H := \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

$$\begin{aligned} &cod(inf) := \left| \begin{array}{l} \text{for } i \in 0..6 \\ \quad \left| \begin{array}{l} c_i \leftarrow 0 \\ \text{for } j \in 0..3 \\ \quad c_i \leftarrow \left(c_i\right) \oplus \left(inf_j \cdot Gen_{j,i}\right) \end{array} \right. \\ c \end{array} \right. \end{aligned}$$

$$inf := \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \qquad c := cod(inf)$$

$$c = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} \qquad c_3 := 1$$

$$c = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} \qquad decod(c) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

$$decod(c) := \left| \begin{array}{l} \text{for } i \in 0..2 \\ \quad s_i \leftarrow 0 \\ \quad \text{for } j \in 0..6 \\ \quad\quad s_i \leftarrow (s_i) \oplus (c_j \cdot H_{i,j}) \\ ss \leftarrow s_0 + s_1 \cdot 2 + s_2 \cdot 4 \\ cc \leftarrow c \\ cc_4 \leftarrow (c_4) \oplus 1 \text{ if } ss = 1 \\ cc_5 \leftarrow (c_5) \oplus 1 \text{ if } ss = 2 \\ cc_2 \leftarrow (c_2) \oplus 1 \text{ if } ss = 3 \\ cc_6 \leftarrow (c_6) \oplus 1 \text{ if } ss = 4 \\ cc_0 \leftarrow (c_0) \oplus 1 \text{ if } ss = 5 \\ cc_3 \leftarrow (c_3) \oplus 1 \text{ if } ss = 6 \\ cc_1 \leftarrow (c_1) \oplus 1 \text{ if } ss = 7 \\ cc \end{array} \right.$$

$$M\_b\_cod := \left| \begin{array}{l} \text{for } i \in 0..ceil\left(\frac{rows(M\_b)}{4}\right) - 1 \\ \quad \text{for } j \in 0..3 \\ \quad\quad inf_j \leftarrow M\_b_{4 \cdot i + j} \\ \quad c \leftarrow cod(inf) \\ \quad \text{for } l \in 0..6 \\ \quad\quad M\_b\_cod_{7 \cdot i + l} \leftarrow c_l \\ M\_b\_cod \end{array} \right.$$

$M\_b =$

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	0
13	1
14	1
15	…

$M\_b\_cod =$

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	0
8	0
9	1
10	1
11	1
12	0
13	1
14	0
15	…

$\gamma := 0.05$

$$SV(x, y, b) := round\left[B_{x,y} + (2 \cdot b - 1) \cdot \gamma \cdot \lambda(x, y)\right]$$

$$S4 := \left| \begin{array}{l} \text{for } i \in 0..cols(B) - 1 \\ \quad \text{for } j \in 0..rows(B) - 1 \\ \quad\quad S4_{j,i} \leftarrow B_{j,i} \\ \text{for } i \in \sigma..rows(B) - \sigma - 1 \\ \quad b \leftarrow SV(i, i, M\_b\_cod_{i-\sigma}) \\ \quad S4_{i,i} \leftarrow b \text{ if } 0 \le b \le 255 \\ \quad S4_{i,i} \leftarrow 255 \text{ if } b > 255 \\ \quad S4_{i,i} \leftarrow 0 \text{ if } b < 0 \\ S4 \end{array} \right.$$

$WRITERGB("Stego\_Krest\_cod.bmp") := augment(R, G, S4)$

S4

B

"1.bmp"

"Stego_Krest_cod.bmp"

B4 := READ_BLUE("Stego_Krest_cod.bmp")

B4 =

	0	1	2	3	4
0	135	128	123	122	119
1	155	142	141	134	133
2	174	159	151	152	141
3	162	160	151	151	147
4	172	161	159	164	164
5	184	181	190	184	183
6	198	197	200	203	206
7	225	222	226	222	...

B4

$$M\_b4 := \left| \begin{array}{l} \text{for } i \in \sigma..\text{rows}(B4) - \sigma - 1 \\ \quad \left| \begin{array}{l} b \leftarrow \dfrac{\left(\displaystyle\sum_{j=i-\sigma}^{i-1} B4_{i,j} + \sum_{j=i-\sigma}^{i-1} B4_{j,i} + \sum_{j=i+1}^{i+\sigma} B4_{i,j} + \sum_{j=i+1}^{i+\sigma} B4_{j,i} \right)}{4\sigma} \\ M\_b4_{i-\sigma} \leftarrow 1 \text{ if } b < B4_{i,i} \\ M\_b4_{i-\sigma} \leftarrow 0 \text{ if } b > B4_{i,i} \end{array} \right. \\ M\_b4 \end{array} \right.$$

```
M_b_decod := | for i ∈ 0..floor(rows(M_b4)/7) − 1
             |   for j ∈ 0..6
             |     c_j ← M_b4_(7·i+j)
             |   c ← decod(c)
             |   for l ∈ 0..3
             |     M_b_decod_(4·i+l) ← c_l
             | M_b_decod
```

M_b4 =

	0
0	0
1	0
2	1
3	1
4	0
5	0
6	1
7	0
8	0
9	1
10	1
11	1
12	0
13	1
14	0
15	…

M_b_cod =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	0
8	0
9	1
10	1
11	1
12	0
13	1
14	0
15	…

M_b =

	0
0	1
1	1
2	0
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	0
13	1
14	1
15	…

M_b_decod =

	0
0	0
1	0
2	1
3	1
4	0
5	0
6	1
7	1
8	0
9	0
10	0
11	0
12	1
13	1
14	0
15	…

Дослідження ймовірностих характеристик

```
v := | v ← 0
     | for i ∈ 0..rows(M_b_decod) − 1
     |   v ← v + 1 if M_b_decod_i = M_b_i
     | v ← v / rows(M_b_decod)
     | v
```

```
w := | w ← 0
     | for i ∈ σ..rows(B4) − σ − 1
     |   w ← w + |B4_(i,i) − B_(i,i)|
     | w ← w·100 / (rows(M_b3)·256)
     | w
```

v = 0.804

w = 2.55

V :=

0.01	0.54
0.05	0.81
0.1	0.93
0.15	0.98
0.2	0.98
0.25	0.99
0.3	0.99
0.35	0.99
0.4	0.99
0.45	1

V_cod :=

0.01	0.63
0.05	0.87
0.1	0.97
0.15	0.99
0.2	1
0.25	1
0.3	1
0.35	1
0.4	1
0.45	1

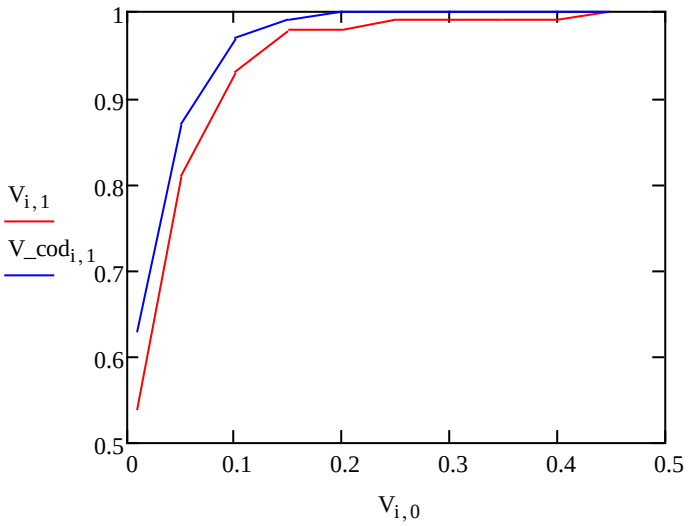
W :=

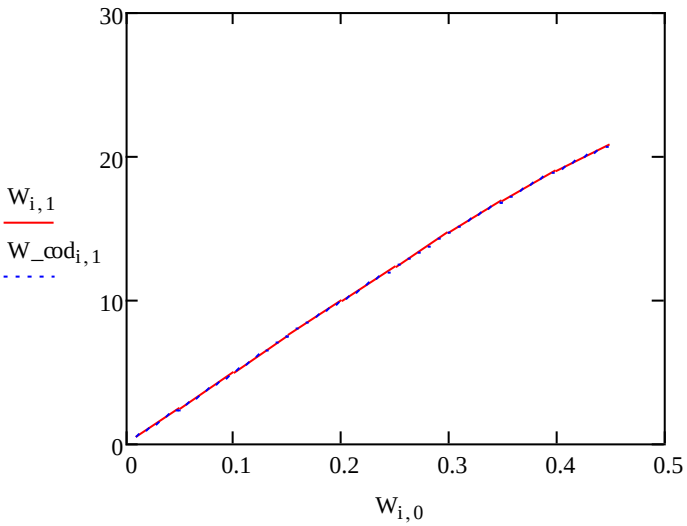
0.01	0.55
0.05	2.55
0.1	5.0
0.15	7.6
0.2	10
0.25	12.4
0.3	14.7
0.35	16.9
0.4	19
0.45	20.8

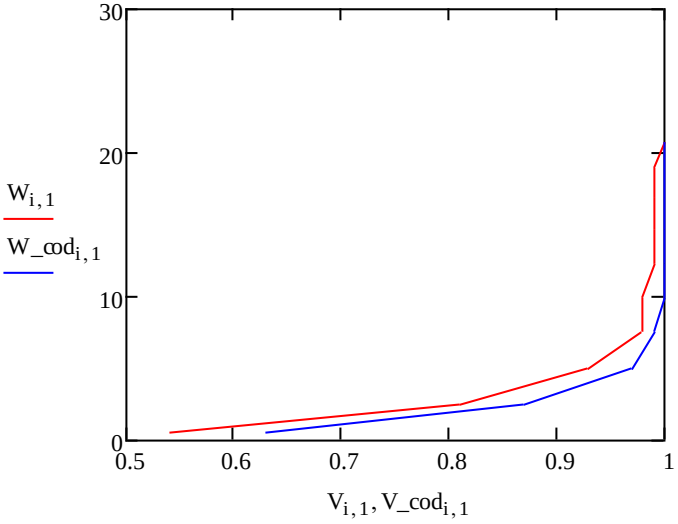
W_cod :=

0.01	0.55
0.05	2.55
0.1	5.0
0.15	7.6
0.2	10
0.25	12.4
0.3	14.7
0.35	16.9
0.4	19
0.45	20.8

$i := 0..9$


1
0.9
0.8
0.7
0.6
0.5
0
0.1
0.2
0.3
0.4
0.5
$V_{i,1}$
$V\_cod_{i,1}$
$V_{i,0}$


30
20
10
0
0
0.1
0.2
0.3
0.4
0.5
$W_{i,1}$
$W\_cod_{i,1}$
$W_{i,0}$


30
20
10
0
0.5
0.6
0.7
0.8
0.9
1
$W_{i,1}$
$W\_cod_{i,1}$
$V_{i,1}, V\_cod_{i,1}$

Лабораторна робота №3 «Приховування даних в просторовій області нерухомих зображень на основі прямого розширення спектру»

1. Мета та завдання лабораторної роботи

Мета роботи: закріпити теоретичні знання за темою «Приховування даних у просторової області нерухомих зображень на основі прямого розширення спектру», набути практичних вмінь та навичок щодо розробки стеганографічних систем, дослідити властивості стеганографічних методів, що засновані на низькорівневих властивостях зорової системи людини (ЗСЛ).

Лабораторна робота №3 виконується у середовищі символьної математики MathCAD версії 12 або вище.

Завдання лабораторної роботи

1. Реалізувати у середовищі символьної математики MathCAD алгоритми формування ансамблів ортогональних дискретних сигналів Уолша-Адамара. Реалізувати алгоритм кодування інформаційних бітів даних складними дискретними сигналами.
2. Реалізувати у середовищі символьної математики MathCAD алгоритми приховування даних у просторову області зображень шляхом прямого розширення спектрів із використанням ортогональних дискретних сигналів. Виконати зорове порівняння пустого та заповненого контейнера та зробити відповідні висновки. Реалізувати алгоритми кореляційного прийому дискретних сигналів. Реалізувати алгоритми вилучення даних з просторової області зображень на основі прямого розширення спектру.
3. Провести експериментальні дослідження ймовірнісних властивостей реалізованого методу, отримати емпіричні залежності ймовірності правильного вилучення даних та частки внесених при цьому похибок у контейнер-зображення.
4. Реалізувати у середовищі символьної математики MathCAD алгоритми формування ансамблів квазіортогональних дискретних сигналів. Реалізувати алгоритми приховування та вилучення даних в просторової області зображень із використанням квазіортогональних дискретних сигналів.
5. (Додаткове завдання). Реалізувати у середовищі символьної математики MathCAD адаптивний алгоритм формування квазіортогональних дискретних сигналів. Реалізувати алгоритми приховування та вилучення даних із адаптовано формованими квазіортогональними дискретними сигналами, отримати емпіричні залежності ймовірності правильного вилучення даних та частки внесених при цьому похибок у контейнер-зображення.

2. Методичні вказівки з організації самостійної роботи

1. Вивчити теоретичний матеріал лекції «Приховування даних у просторової області нерухомих зображень на основі прямого розширення спектру».
2. Вивчити матеріал основного джерела літератури (Конахович Г.Ф., Пузыренко А.Ю. Компьютерная стеганография): метод розширення спектру (ст. 180-189).
3. Вивчити матеріал додаткових джерел:
   a. основи використання складних сигналів у системах зв’язку (Стасєв Ю.В. Основи теорії побудови сигналів, ст. 5 - 13);
   b. дискретні сигнали (Стасєв Ю.В. Основи теорії побудови сигналів, ст. 14 - 22).
4. Вивчити основні команди у середовищі символьної математики MathCAD щодо роботи із зображеннями.
5. Підготувати відповіді на контрольні запитання.
6. Підготувати бланк звіту з лабораторної роботи.

Допуск до виконання лабораторної роботи здійснюється за результатами письмового опитування.

3. Загальнотеоретичні положення за темою лабораторної роботи

Завадозахищені системи зв’язку та управління

Прагнення забезпечити високу завадозахищеність, множинний безконфліктний доступ до каналу, а також енергетичний і інформаційний захист повідомлень, які передаються, привело до створення широкополосних завадозахищених адресних систем зв'язку, що працюють в режимі вільного доступу з кодовим ущільненням каналів. Вільний доступ забезпечується тим, що загальний широкополосний радіотракт може використовуватися абонентами в разі необхідності, коли з'являються повідомлення, які підлягають передачі.

Теоретичною основою завадозахищеного зв'язку є відома теорема Шеннона щодо пропускної здатності каналу зв'язку, яка стверджує, що при швидкості передачі інформації R, меншій пропускної здатності каналу зв'язку C, існують такі засоби кодування інформації, які дозволяють передавати цю інформацію з заданою якістю при будь-якому, як завгодно малому, відношенні потужності сигналу P_c до потужності завади $P_п$.

Ця теорема не вказує на конкретні методи кодування, однак чітко формулює шлях досягнення заданої якості передачі.

У відповідності з теоремою Шеннона пропускна здатність каналу зв'язку дорівнює:

$$C = \Delta F_k \log_2\left(1+\frac{P_c}{P_п}\right), \tag{3.1}$$

де ΔF_k – ширина полоси пропускання каналу.

Поділивши обидві частини рівності (3.1) на ΔF_k і помінявши основу логарифму, отримаємо

$$\frac{C}{\Delta F_k} = 1{,}44 \cdot \ln\left(1+\frac{P_c}{P_п}\right). \tag{3.2}$$

При $\frac{P_c}{P_п} < 1$, що представляє інтерес для завадозахищених радіоканалів, вираз (3.2) буде мати такий вигляд:

$$\frac{C}{\Delta F_k} = 1{,}44 \cdot \left[\frac{P_c}{P_п} - \frac{1}{2}\left(\frac{P_c}{P_п}\right)^2 + \frac{1}{3}\left(\frac{P_c}{P_п}\right)^3 k\right]. \tag{3.3}$$

Враховуючи, що $\frac{P_c}{P_п} < 1$, і нехтуючи членами ряду вищих порядків, можна записати

$$\frac{C}{\Delta F_k} = 1{,}44 \cdot \frac{P_c}{P_п}. \tag{3.4}$$

Вираз (3.4) вказує шлях досягнення заданої якості передачі при як завгодно малому відношенні $\frac{P_c}{P_п}$. Вважаючи, що ширина полоси пропускання каналу дорівнює ширині спектру використовуваних сигналів $\Delta F_k = \Delta F_c$, з (3.4) витікає, що при зменшенні відношення потужності сигналу

до потужності завади необхідно застосовувати такі методи кодування, які призводять до розширення спектра сигналів.

Для C=R, неважко помітити, що зберігання рівняння (3.4) при зменшенні відношення $\frac{P_c}{P_п}$ досягається пропорційним збільшенням відношення $\frac{\Delta F_c}{R}$.

Метод передачі інформації, при якому сигнал займає полосу частот, що набагато переважає полосу частот повідомлення, називається широкополосним.

Ортогональні, субортогональні та квазіортогональні дискретні сигнали, їх кореляційні та ансамблеві властивості

Як уже було відмічено для побудови сучасних завадозахисних систем цифрового зв'язку використовуються методи теорії дискретних сигналів, кореляційного і спектрального аналізу. При цьому з погляду ефективного використання частотно-часових і енергетичних ресурсів каналів зв'язку найбільш перспективними вважаються широкосмугові системи з шумоподібними дискретними сигналами і прямим розширенням спектру.

Залежно від способу формування і статистичних властивостей кодові послідовності, що використаються в системах зв'язку з кодовим розділенням каналів, розділяються на ортогональні, субортогональні (інша назва трансортогональні) і квазіортогональні.

Нехай $S_i = \left(\varphi_{i_0}, \varphi, ..., \varphi_{i_{n-1}}\right)$ – двійкова послідовність псевдовипадкових чисел ППВЧ (кодовий сигнал) з множини $\Phi = \left\{\Phi_0, \Phi_1, ..., \Phi_{M-1}\right\}$ потужності $|S| = M$.

Елементи двійкової ППВЧ приймають одне із значень:

$$\Phi_{i_z} = \begin{cases} +1 \\ -1 \end{cases}, \; z = 0, ..., n-1.$$

Нормована періодична функція взаємної кореляції (ПФВК) характеризує відгук обладнання на періодичну послідовність сигналів, відмінних від очікуваного сигналу і визначається за виразом:

$$R_{i,j}^{ПФВК}(\ell) = \frac{1}{n}\left(\Phi_{i_0}\Phi_{j_{(\ell)mod(n)}} + \Phi_{i_1}\Phi_{j_{(\ell+1)mod(n)}} + \cdots + \Phi_{i_{n-1}}\Phi_{j_{(\ell+n-1)mod(n)}}\right) =$$

$$= \frac{1}{n}\sum_{z=0}^{n-1}\Phi_{i_z}\Phi_{j_{(\ell+z)mod(n)}}.$$

Нормована періодична функція автокореляції (ПФАК) характеризує відгук обладнання на періодичну послідовність очікуваних сигналів і визначається за виразом:

$$R_{i,i}^{\text{ПФАК}}(\ell) = \frac{1}{n}\left(\Phi_{i_0}\Phi_{i_{(\ell)\bmod(n)}} + \Phi_{i_1}\Phi_{i_{(\ell+1)\bmod(n)}} + \cdots + \Phi_{i_{n-1}}\Phi_{i_{(\ell+n-1)\bmod(n)}}\right) =$$

$$= \frac{1}{n}\sum_{z=0}^{n-1}\Phi_{i_z}\Phi_{i_{(\ell+z)\bmod(n)}}.$$

Значення функцій кореляції при фіксованому $\ell = 0$ називають коефіцієнтом кореляції $\rho_{ij} = R_{i,j}^{\text{ПФВК}}(0)$, який в загальному випадку змінюється від -1 до +1.

Коефіцієнт взаємної кореляції ортогональних послідовностей, за визначенням, рівний нулю, тобто

$$\rho_{ij} = 0.$$

Невелике значення коефіцієнта взаємно кореляційній функції (ВКФ) забезпечують субортогональні (трансортогональні) коди, для яких

$$\rho_{ij} = \begin{cases} -1/N, \text{ де N непарне,} \\ -1/(N-1), \text{ де N парне.} \end{cases} \qquad (3.5)$$

При великих значеннях N відмінністю між коефіцієнтами кореляції ортогональних і трансортогональних кодів можна практично нехтувати.

Існує декілька способів генерації ортогональних кодів. Найбільш поширений – за допомогою послідовностей Уолша довжини 2^i. Вони утворюються на основі рядків матриці Адамара H_i, які у свою чергу будуються за рекурентним правилом:

$$H_i = \begin{bmatrix} H_{i-1} & H_{i-1} \\ H_{i-1} & -H_{i-1} \end{bmatrix}, \; H_0 = [1].$$

Використовуючи приведене правило отримаємо:

$$H_1 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \; H_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix},$$

$$H_3 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}.$$

Багатократне повторення процедури дозволяє сформувати матрицю розміру 2^i, для якої характерна взаємна ортогональность всіх рядків і стовпців.

Такий спосіб формування сигналів реалізований в стандарті IS-95, де довжина послідовностей Уолша вибрана рівною 64. Відмітимо, що відмінність між рядками матриці Адамара і послідовностями Уолша полягає лише в тому, що в останніх використовуються уніполярні сигнали вигляду {1,0}.

На прикладі матриці Адамара легко проілюструвати і принцип побудови трансортогональних кодів. Так, можна переконатися, що якщо з матриці викреслити перший стовпець, що складається з одних одиниць, то ортогональні коди Уолша трансформуються в трансортогональні, у яких для будь-яких двох послідовностей число незбігів символів перевищує число збігів рівно на одиницю. Отже, виконується рівняння (3.5).

Інший важливий різновид кодів - біортогональний код, який формується з ортогонального коду і його інверсії. Головна позитивна якість біортогональних кодів в порівнянні з ортогональними - можливість передачі сигналу в удвічі меншій смузі частот. Скажімо, біортогональний блоковий код, що використається в WCDMA, дозволяє передавати сигнал транспортного формату TFI.

Відзначимо, що ортогональним кодам властиві два принципові недоліки.

1. Максимальне число можливих кодів обмежене їх довжиною (у стандарті IS-95 число кодів дорівнює 64), а відповідно, вони мають обмежений адресний простір.

2. Ще один недолік ортогональних кодів полягає в тому, що функція взаємної кореляції рівна нулю лише «в точці», тобто за відсутності тимчасового зсуву між кодами. Тому такі сигнали використовуються лише в синхронних системах і переважно в прямих каналах (від базової станції до абонента)

Прагнення підвищити абонентську місткість систем зв'язку з кодовим розділенням каналів неминуче приводить до використання великих ансамблів т.з. квазіортогональних сигналів, тобто великої множині таких псевдовипадкових послідовностей, коефіцієнт кореляції між якими дуже

близький до нуля (майже ортогональні сигнали). Так, в проекті стандарту cdma2000 запропонований метод генерації квазіортогональних кодів шляхом множення послідовностей Уолша на спеціальну маскуючу функцію. Цей метод дозволяє за допомогою однієї такій функції отримати набір квазіортогональних послідовностей Quasi-Orthogonal Function Set (QOFS). За допомогою m маскуючих функцій і ансамблю кодів Уолша завдовжки 2n можна створити (m+1) 2n QOF- послідовностей.

Псевдовипадкова послідовність (ПВП) - послідовність чисел, яка була обчислена за деяким певним арифметичним правилом, але має всі властивості випадкової послідовності чисел в рамках вирішуваного завдання.

Хоча псевдовипадкова послідовність в цьому сенсі частіш, як може показатися, позбавлена закономірностей, проте, будь-який псевдовипадковий генератор з кінцевим числом внутрішніх станів повториться після дуже довгої послідовності чисел.

Псевдовипадкова двійкова послідовність — окремий випадок ПВП, у якій елементи приймають два можливі значення 0 та 1 (або -1 та +1).

Одне з перших формулювань деяких основоположних правил для статистичних властивостей періодичних псевдовипадкових послідовностей була представлена Соломоном Голомбом.

Три основних правила здобули популярність як постулати Голомба.

1. Кількість "1" в кожному періоді повинна відрізнятися від кількості "0" не більш, ніж на одиницю.

2. У кожному періоді половина серій (з однакових символів) повинна мати довжину один, одна чверть повинна мати довжину два, одна восьма повинна мати довжину три і так далі. Більш того, для кожної з цих довжин повинна бути однакова кількість серій з "1" та "0".

3. Припустимо, у нас є дві копії однієї і тієї ж послідовності періоду p, зсуванні щодо один одного на деяке значення d. Тоді для кожного d:

$$0 <= d <= p-l,$$

ми можемо підрахувати кількість узгодженостей між цими двома послідовностями Ad, і кількість неузгодженостей Dd. Коефіцієнт автокореляції для кожного d визначається співвідношенням (Ad - Dd)/p і ця функція автокореляції приймає різні значення у міру того, як d проходить всі допустимі значення.

Тоді для будь-якої послідовності, що задовольняє правилу 3, автокореляційна функція (АКФ) повинна приймати лише два значення.

Правило 3 – це технічний вираз того, що Голомб описав як поняття незалежних випробувань: знання деякого попереднього значення послідовності в принципі не допомагає припущенням про поточне значення. Ще одна точка зору на АКФ полягає в тому, що це певна міра здатності, що дозволяє розрізняти послідовність та її копію, але ту, що починається в деякій іншій точці циклу.

Послідовність, що задовольняє правилам 1-3 часто іменується "псевдо-шумовою-послідовністю". До аналізованої послідовності застосовується широкий спектр різних статистичних тестів для дослідження того, наскільки добре вона узгоджується з допущенням, що для генерації використовувалося абсолютно випадкове джерело.

Методи розширення спектру для підвищення ефективності передачі дискретних повідомлень

В існуючих на сьогоднішній день системах передачі дискретних повідомлень використовуються два методи розширення спектру:

- *псевдовипадкова перебудова робочої частоти* (ППРЧ) (англ. FHSS — Frequency Hopping Spread Spectrum). Суть методу полягає в періодичній стрибкоподібній зміні частоти, що несе по деякому алгоритму, відомому приймачу і передавачу. Перевага методу - простота реалізації. Метод використовується в Bluetooth;

- *розширення спектру методом прямої послідовності* (ПРС) (англ. DSSS — Direct Sequence Spread Spectrum). Метод по ефективності перевершує ППРЧ, але складніше в реалізації. Суть методу полягає в підвищенні тактової частоти модуляції, при цьому кожному символу переданого повідомлення ставиться у відповідність деяка достатньо довга псевдовипадкова послідовність (ПВП). Метод використовується в таких системах як CDMA і системах стандарту IEEE 802.11.

Розширення спектру псевдовипадковою перебудовою робочої частоти. Для того, щоб радіообмін не можна було перехопити або заглушити вузькосмуговим шумом, було запропоновано вести передачу з постійною зміною несучої в межах широкого діапазону частот. В результаті потужність сигналу розподілялася по всьому діапазону, і прослуховування якоїсь певної частоти давало тільки невеликий шум. Послідовність несучих частот, була псевдовипадковою, відомою тільки передавачу і приймачу. Спроба заглушення сигналу в якомусь вузькому діапазоні також не дуже погіршувала сигнал, оскільки заглушувалася тільки невелика частина інформації. Ідею цього методу ілюструє рис. 3.1.

Протягом фіксованого інтервалу часу передача ведеться на незмінній несучої частоті. На кожній несучої частоті, для передачі дискретній інформації застосовуються стандартні методи модуляції, такі як FSK або PSK. Для того, щоб приймач синхронізувався з передавачем, для позначення початку кожного періоду передачі протягом деякого часу передаються синхробіти. Отже корисна швидкість цього методу кодування виявляється менше через постійні накладні витрати на синхронізацію.

Рисунок 3.1 - Розширення спектру стрибкоподібною перебудовою частоти

Несуча частота, змінюється відповідно до номерів частотних підканалів, що виробляються алгоритмом псевдовипадкових чисел. Псевдовипадкова послідовність залежить від деякого параметра, який називають початковим числом. Якщо приймачу і передавачу відомі алгоритм і значення початкового числа, то вони міняють частоти в однаковій послідовності, що зветься послідовністю псевдовипадкової перебудови частоти.

Методи FHSS використовуються в бездротових технологіях IEEE 802.11 та Bluetooth. В FHSS підхід до використання частотного діапазону не такий, як в інших методах кодування – замість економного витрачання вузької смуги робиться спроба зайняти весь доступний діапазон. На перший погляд це здається не дуже ефективним – адже в кожен момент часу в діапазоні працює тільки один канал. Проте останнє твердження не завжди справедливо – коди розширеного спектру можна використовувати і для мультиплексування декількох каналів в широкому діапазоні. Зокрема, методи FHSS дозволяють організувати одночасну роботу декількох каналів шляхом вибору для кожного каналу таких псевдовипадкових послідовностей, щоб в кожен момент часу кожен канал працював на своїй частоті (звичайно, це можна зробити, тільки якщо число каналів не перевищує числа частотних підканалів).

Розширення спектру методом прямої послідовності.

В методі прямого послідовного розширення спектру також використовується весь частотний діапазон, виділений для однієї лінії зв'язку. На відміну від методу FHSS, весь частотний діапазон займається не за рахунок постійних перемикань з частоти на частоту, а за рахунок того, що кожен біт інформації замінюється N-бітами, так що тактова швидкість передачі сигналів збільшується в N разів. А це, у свою чергу, означає, що

спектр сигналу також розширюється в N разів. Достатньо відповідним чином вибрати швидкість передачі даних і значення N, щоб спектр сигналу заповнив весь діапазон (рис. 3.2).

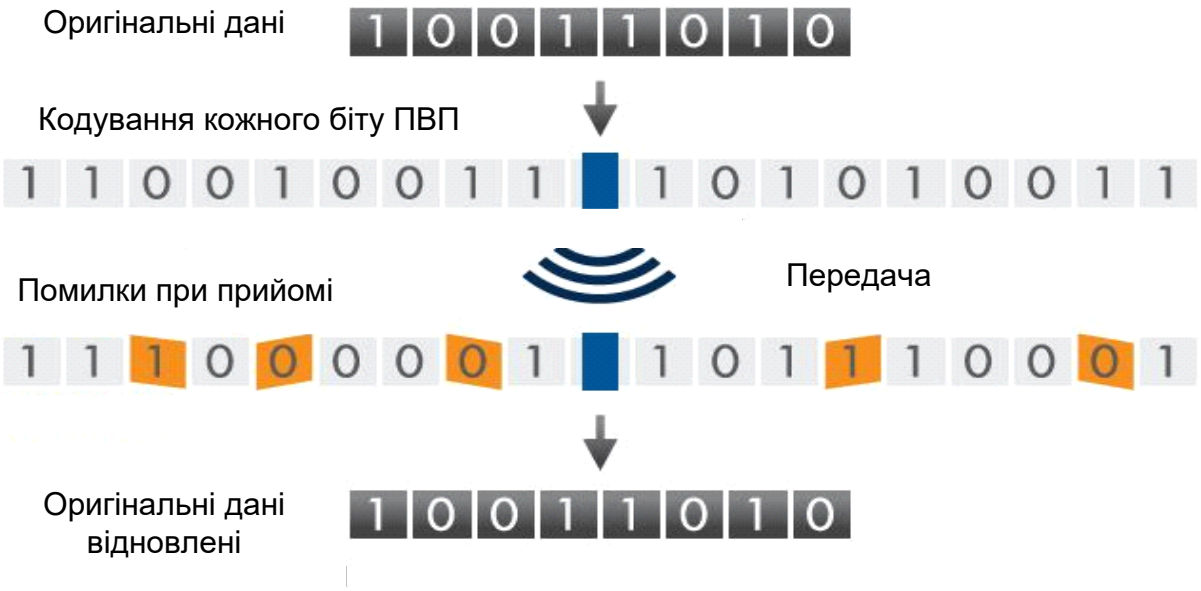


Рисунок 3.2 - Технологія кодового розділення каналів CDMA

Для передачі даних в широкосмуговій системі зв'язку інформаційний сигнал $x(t)=\begin{cases}+1\\-1\end{cases}$ модулюється за допомогою його множення на розширюючий кодовий сигнал $g(t)=\Phi_i \in \Phi$ - псевдовипадкову послідовність з розглянутих вище ансамблів дискретних сигналів. Оскільки кодовий сигнал по своїх статистичних властивостях подібний шуму, то одержаний розширений сигнал

$$y'(t)=y(t)+e(t) \tag{3.6}$$

слабо відрізняться від шумів в каналі зв'язку, що і дозволяє здійснити приховану передачу.

При прийомі в демодуляторі одержаний сигнал $y'(t)=y(t)+e(t)$ як суміш переданої послідовності $y(t)$ і подій в каналі зв'язку помилок $e(t)$ умножається на синхронізовану копію розширяльного сигналу $g(t)$. Іншими словами, на приймальній стороні здійснюється обчислення коефіцієнта кореляції, значення якого визначає правило ухвалення рішення:

$$\rho(y'(t),g(t))=\frac{1}{n}\sum_{z=0}^{n-1}x(t)\Phi_{i_z}\Phi_{i_z}+\frac{1}{n}\sum_{z=0}^{n-1}e(t)\Phi_{i_z}. \tag{3.7}$$

Враховуючи псевдовипадковість Φ_i, використовуваних в якості $g(t)$, другим доданкам в правій частині рівності можна нехтувати (кількість «+1» приблизно рівна кількості «-1»), тобто

$$\rho\left(y'(t), g(t)\right) \approx \rho\left(y(t), g(t)\right) = x(t)\frac{1}{n}\sum_{z=0}^{n-1}\left(\Phi_{i_z}\right)^2 = x(t), \qquad (3.8)$$

тобто значення інформаційного сигналу на приймальній стороні визначається згідно виразу

$$x(t) = \begin{cases} +1, \text{при}\ \rho\left(y'(t), g(t)\right) \approx +1; \\ -1, \text{при}\ \rho\left(y'(t), g(t)\right) \approx -1; \end{cases} \qquad (3.9)$$

де знак «≈» припускає наявність помилок, викликаних природними або навмисними завадами в каналі зв'язку.

Структурна схема тракту прийому-передачі інформації з використанням прямого розширення спектру приведена на рис. 3.3.

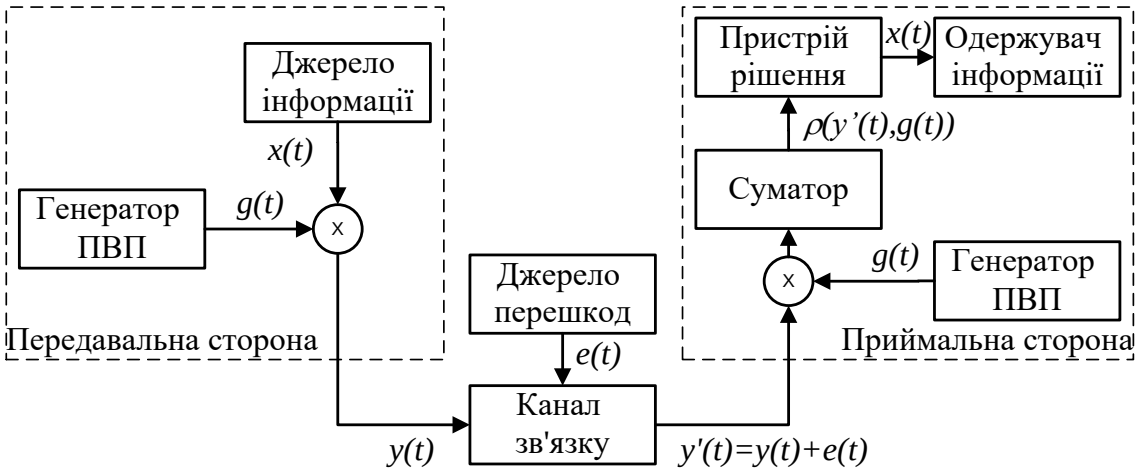


Рисунок 3.3 - Структурна схема тракту прийому-передачі інформації з використанням прямого розширення спектру

Припустимо, що часова тривалість немодульованого сигналу $x(t)$ рівна T, а його частота відповідно рівна $F\left(x(t)\right) = \frac{1}{T}$. Передача модульованого сигналу $y(t)$ при тій же часовій тривалості T приведе до розширення частотного спектру переданого сигналу, пропорційно числу елементів псевдовипадкової послідовності, тобто пропорційно довжині n: $F\left(y(t)\right) = n\frac{1}{T} = nF\left(x(t)\right)$. Проте, використання прямого розширення спектру переданого сигналу забезпечує одночасну передачу багатьох інших

інформаційних сигналів в тій же смузі частот. Це витікає з взаємної ортогональності (квазіортогональності) вживаних ансамблів дискретних сигналів. Дійсно, якщо на приймальній стороні прийнята адитивна суміш $\sum_{\ell} y_l(t)$ декількох модульованих сигналів, тоді обчислення коефіцієнта кореляції дасть наступне:

$$\rho\left(\sum_{\ell} y_{\ell}(t), g(t)\right) = \frac{1}{n}\sum_{\ell}\sum_{z=0}^{n-1} x_{\ell}(t)\Phi_{\ell_z}\Phi_{i_z} \quad . \tag{3.10}$$

Але всі послідовності з множини мають низьке значення взаємної кореляції, тобто при $\ell \neq i$ маємо $\rho(\Phi_{\ell}, \Phi_i) = 0$ (для ортогональних сигналів маємо рівність $\rho(\Phi_{\ell}, \Phi_i) = 0$. Отже, всіма доданками при $\ell \neq i$ в правій частині рівності (3.10) можна нехтувати. Звідси, за наявності в адитивній суміші $\sum_{\ell} y_{\ell}(t)$ дискретного сигналу $\Phi_{\ell=i}$ маємо вираз (3.8) і відповідне правило ухвалення рішення (3.9).

Мета кодування методом DSSS та ж, що і методом FHSS, – підвищення стійкості до завад. Вузькосмугова завада спотворюватиме тільки певні частоти спектру сигналу, так що приймач з великим ступенем імовірності зможе правильно розпізнати передану інформацію.

Код, яким замінюється двійкова одиниця початкової інформації, називається розширюючою послідовністю, а кожен біт такої послідовності - чіпом (елементарним сигналом). Відповідно, швидкість передачі результуючого коду називають чіповою швидкістю. Двійковий нуль кодується інверсним значенням розширюючеї послідовності. Приймачі повинні знати розширюючу послідовність, яку використовує передавач, щоб зрозуміти передану інформацію.

Кількість бітів в розширюючої послідовності визначає коефіцієнт розширення початкового коду. Як і у разі FHSS, для кодування бітів результуючого коду може використовуватися будь-який вид модуляції, наприклад BFSK.

Чим більше коефіцієнт розширення, тим ширше спектр результуючого сигналу і вище ступінь заглушення завад. Але при цьому росте займаний каналом діапазон спектру. Зазвичай коефіцієнт розширення має значення від 10 до 100.

Приклад. Дуже часто як значення розширюючої послідовності беруть послідовність Баркера (Barker), яка складається з 11 бітів: 10110111000. Якщо передавач використовує цю послідовність, то передача трьох бітів 110 веде до передачі наступних бітів:

10110111000 10110111000 01001000111.

Послідовність Баркера дозволяє приймачу швидко синхронізуватися з передавачем, тобто надійно виявляти початок послідовності. Приймач визначає таку подію, по черзі порівнюючи отримувані біти із зразком

послідовності. Дійсно, якщо порівняти послідовність Баркера з такою ж послідовністю, але зсуненою на один біт вліво або вправо, ми отримаємо менше половини збігів значень бітів. Таким чином, навіть при спотворенні декількох бітів з великою часткою імовірності приймач правильно визначить початок послідовності, а значить, зможе правильно інтерпретувати отримувану інформацію.

Перерахуємо деякі властивості сигналів з прямим розширенням спектру, найбільш важливі з погляду організації множинного доступу в системах зв'язку з пересувними об'єктами.

1. *Множинний доступ*. Якщо одночасно декілька абонентів використовують канал передачі, то в каналі одночасно присутні декілька сигналів з прямим розширенням спектру. У приймачі сигналу конкретного абонента здійснюється зворотна операція – згортання сигналу цього абонента шляхом використання того ж псевдовипадкового сигналу, який був використаний в передавачі цього абонента. Ця операція концентрує потужність широкосмугового сигналу, що приймається, знову у вузькій смузі частот, рівній ширині спектру інформаційних символів. Якщо взаємна кореляційна функція між псевдовипадковими сигналами даного абонента і інших абонентів достатньо мала, то при когерентному прийомі в інформаційну смугу приймача абонента потрапить лише незначна частка потужності сигналів решти абонентів. Сигнал конкретного абонента буде прийнятий вірно

2. *Багатопроменева інтерференція*. Якщо псевдовипадковий сигнал, використовуваний для розширення спектру має ідеальну автокореляційну функцію, значення якої поза інтервалом $[-t_0,+ t_0]$ дорівнює нулю, і якщо сигнал, що приймається, і копія цього сигналу в іншому промені зсуванні в часі на величину, велику $2t_0$, то при згортанні сигналу його копія може розглядатися як заважаюча інтерференція, що вносить лише малу частку потужності в інформаційну смугу.

3. *Вузькосмугова завада*. При когерентному прийомі в приймачі здійснюється множення прийнятого сигналу на копію псевдовипадкового сигналу, використованого для розширення спектру в передавачі. Отже, в приймачі здійснюватиметься операція розширення спектру вузькосмугової завади, аналогічна тій, яка виконувалася з інформаційним сигналом в передавачі. Отже, спектр вузькосмугової завади в приймачі буде розширений у B раз, де B – коефіцієнт розширення, так що в інформаційну смугу частот потрапить лише мала частка потужності завади, у B раз менше початкової потужності завади.

4. *Імовірність перехоплення*. Оскільки сигнал з прямим розширенням спектру займає всю смугу частот системи протягом усього часу передачі, то його випромінювана потужність, що доводиться на 1 Гц смуги, матиме дуже малі значення. Отже, виявлення такого сигналу є дуже важким завданням.

Таким чином, перспективним напрямом в розвитку сучасних систем широкосмугового зв'язку з прямим розширенням спектру є розробка і

дослідження методів синтезу великих ансамблів квазіортогональних дискретних сигналів з покращуваними ансамблевими, структурними і кореляційними властивостями.

Розглянутий підхід до організації цифрових завадозахисних каналів зв'язку знайшов застосування при побудові стеганографіческіх методів захисту інформації. Так, наприклад, розширення спектру прямою послідовністю використане для створення стеганографіческого методу вбудовування даних в нерухомі зображення. Розглянемо один з варіантів реалізації цього методу, авторами якого є Сміт (J.R. Smith) і Коміські (B.O. Comiskey), проведемо дослідження його ефективності з погляду забезпечуваної пропускної спроможності стеганографіческого каналу зв'язку і стійкості, що досягається, до несанкціонованого витягання інформаційних повідомлень.

Пряме розширення спектру в стеганографії

В методі Сміта-Коміські, як і в розглянутих вище системах зв'язку з прямим розширенням спектру, інформаційне повідомлення побітно модулюється шляхом множення на ансамбль ортогональних сигналів. Потім промодульоване повідомлення вбудовується в контейнер - нерухоме зображення.

Введемо деякі умовні позначення і математичні співвідношення, які, по аналогії з розглянутими вище системами широкосмугового цифрового зв'язку дозволять досліджувати особливості побудови і інформаційного обміну даних в стеганостістеме.

Представимо інформаційне повідомлення m, що підлягає вбудовуванню в цифровий контейнер-зображення, у вигляді блоків m_i рівної довжини, тобто $m = (m_0, m_1, ..., m_{N-1})$, де кожен блок m_i – послідовність (вектор) з n біт:

$$m_i = (m_{i_0}, m_{i_1}, ..., m_{i_{n-1}}).$$

Контейнер-зображення розглядатимемо як масив даних C розмірністю $K \cdot L$, розбитий на підблоки розміром $k \cdot l = n$. Як елементи масиву C можуть виступати, наприклад, растрові дані використовуваного зображення.

Секретними ключовими даними є набір базисних функцій $Key = \Phi = \{\Phi_0, \Phi_1, ..., \Phi_{M-1}\}$, де всі базисні функції $\Phi_i = (\varphi_{i_0}, \varphi_{i_1}, ..., \varphi_{i_{n-1}})$ – взаємно ортогональні дискретні сигнали з довжиною, рівною розміру блоку n повідомлення m_i, тобто для будь-яких $i, j \in [0, ..., M-1]$ виконується рівність

$$\rho(\Phi_i, \Phi_j) = \frac{1}{n}\sum_{z=0}^{n-1}\Phi_{i_z}\Phi_{j_z} = \begin{cases} +1, \text{при } i = j; \\ -1, \text{при } i \neq j. \end{cases}$$

Формальне графічне представлення інформаційного повідомлення, контейнера-зображення і ключових даних приведене на рис. 6.6.

Метою стеганографічного перетворення інформації є вбудовування кожного окремого блоку повідомлення m_i у відповідний блок контейнера-зображення. В блок даних цифрового зображення розмірністю $K \cdot L$ елементів може бути вбудовано $K \cdot \frac{L}{n}$ блоків інформаційного повідомлення, тобто до $K \cdot L$ бітів.

Розбиття контейнера на блоки може бути довільним, проте, як показує практика, найбільш доцільним (менший, на відміну від одновимірного уявлення, чисельний розкид значень в блоці) є двовимірне розбиття, приведене на рис. 3.4. Як ключові дані (масиву базисних функцій $Key = \Phi$) використаємо розглянуті вище ансамблі ортогональних дискретних сигналів Уолша-Адамара.

Вбудовування інформаційного повідомлення здійснюється таким чином. Кожен блок повідомлення $m_{i_j}, j = 0,..., n-1$ зіставляється з окремим блоком контейнера-зображення. Кожен інформаційний біт блоку $m_{i_j}, j = 0,..., n-1$ представляється у вигляді інформаційного сигналу $m_{i_j}(t) = \begin{cases} +1, m_{i_j} = 1; \\ -1, m_{i_j} = 0; \end{cases}$ і по аналогії з (3.6) модулюється розширюючим кодовим сигналом (базисними функціями), тобто ПВП $\Phi_j \in \Phi$.

В результаті, для кожного інформаційного блоку формується модульований інформаційний сигнал:

$$E_i(t) = \sum_{j=0}^{n-1}\sum_{z=0}^{n-1} m_{i_j}(t)\Phi_{j_z}. \qquad (3.11)$$

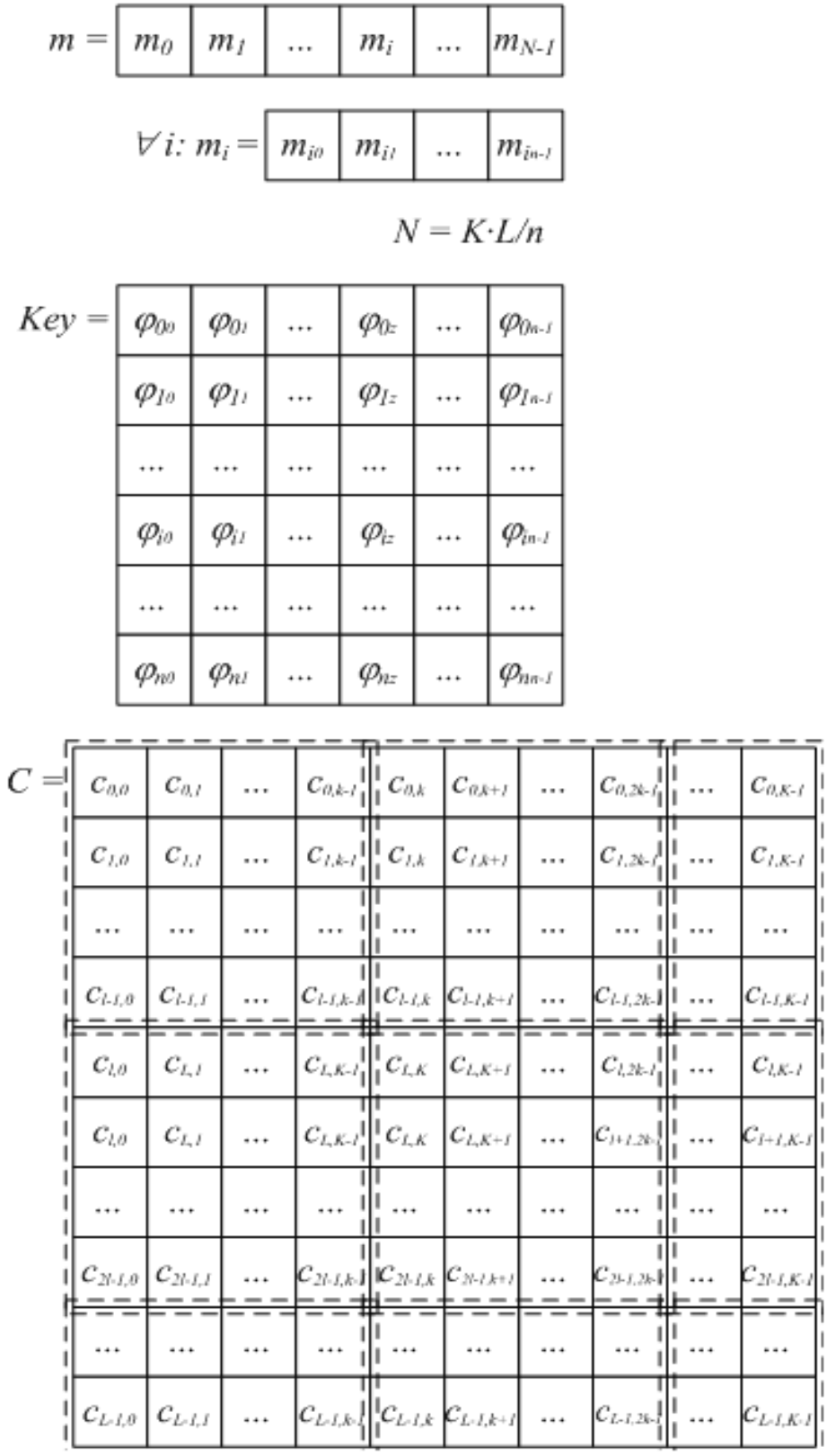


Рисунок 3.4 - Формальне представлення інформаційного повідомлення, контейнера-зображення і ключових даних

Отриманий блок повідомлення E_i попіксельно підсумовується з підблоком контейнеру.

Позначимо блоки контейнера таким чином (див. рис. 3.4):

$$C_0 = \begin{pmatrix} c_{0,0} & c_{0,1} & \dots & c_{0,k-1} \\ c_{1,0} & c_{1,1} & \dots & c_{1,k-1} \\ \dots & \dots & \dots & \dots \\ c_{\ell-1,0} & c_{\ell-1,1} & \dots & c_{\ell-1,k-1} \end{pmatrix}, \; C_1 = \begin{pmatrix} c_{0,k} & c_{0,k+1} & \dots & c_{0,2k-1} \\ c_{1,k} & c_{1,k+1} & \dots & c_{1,2k-1} \\ \dots & \dots & \dots & \dots \\ c_{\ell-1,k} & c_{\ell-1,k+1} & \dots & c_{\ell-1,2k-1} \end{pmatrix}, \dots,$$

$$C_{N-1} = \begin{pmatrix} c_{L-l-1,K-k-1} & c_{L-l-1,K-k} & \dots & c_{L-l-1,K-1} \\ c_{L-l,K-k-1} & c_{L-l,K-k} & \dots & c_{L-l,K-1} \\ \dots & \dots & \dots & \dots \\ c_{L-1,K-k-1} & c_{l-1,k+1} & \dots & c_{L-1,K-1} \end{pmatrix}.$$

Відповідні модульовані інформаційні сигнали $E_i(t)$ представимо у вигляді двовимірного масиву даних:

$$E_i = \begin{pmatrix} E_{i_0} & E_{i_1} & \dots & E_{i_{k-1}} \\ E_{i_k} & E_{i_{k+1}} & \dots & E_{i_{2k-1}} \\ \dots & \dots & \dots & \dots \\ E_{i_{(\ell-1)(k-1)-k+1=n-k+1}} & E_{i_{(\ell-1)(k-1)-k+2=n-k+2}} & \dots & E_{i_{(\ell-1)(k-1)=n-1}} \end{pmatrix}, i = 0,\dots, N-1.$$

Тоді стеганограма (заповнений контейнер) формується за допомогою об'єднання масивів даних S_i, $i = 0,\dots, N-1$:

$$S_i = C_i + E_i \cdot G, \tag{3.12}$$

де $G > 0$ - коефіцієнт посилення розширюючого сигналу, що задає «енергію» вбудованих біт інформаційної послідовності.

Таким чином, заповнений контейнер S утворюється з сформованих блоків S_i, $i = 0,\dots, N-1$ за допомогою їх об'єднання як це показано на рис. 3.4 для початкового (порожнього) контейнеру C.

На етапі вибудовування даних немає необхідності володіти інформацією про первинний контейнер C. Операція декодування полягає у відновленні прихованого повідомлення шляхом проектування кожного блоку S_i, одержаного стеганозображення S на всі базисні функції $\Phi_j \in \Phi$, $i = 0,\dots, N-1$. Для цього кожен блок S_i представляється у формі вектора $S_i = (S_{i_0}, S_{i_1}, \dots, S_{i_{n-1}})$, $i = 0,\dots, N-1$

Щоб витягнути j-й біт повідомлення з i-го блоку стеганозображення необхідно обчислити коефіцієнт кореляції між Φ_j і прийнятим блоком S_i (представленого у вигляді вектору):

$$\rho(S_i,\Phi_j)=\frac{1}{n}\sum_{z=0}^{n-1}S_{i_z}\Phi_{j_z}=G\cdot\frac{1}{n}\sum_{z=0}^{n-1}E_{i_z}\Phi_{j_z}+\frac{1}{n}\sum_{z=0}^{n-1}C_{i_z}\Phi_{j_z}\ , \qquad (313)$$

де під C_i розуміється одновимірний масив, тобто відповідний блок контейнеру, представлений у формі вектора.

Припустимо, що масив C_i має випадкову статистичну структуру, тобто другий додаток в правій частині виразу (3.13) близько нуля і їм можна нехтувати. Тоді маємо:

$$\rho(S_i,\Phi_j)\approx G\cdot E_i\cdot\Phi_j=G\cdot\sum_{l=0}^{n-1}\sum_{z=0}^{n-1}m_{i_x}(t)\cdot\Phi_{l_z}\Phi_{j_z}\ . \qquad (3.14)$$

По аналогії з (3.10) відзначимо, що всі послідовності з множині Φ взаємно ортогональні, тобто при $l\neq j$ маємо $\rho(\Phi_i,\Phi_j)=0$. Отже, всіма додатками в правій частині рівності (6.14) при $l\neq j$ можна нехтувати. Звідси маємо:

$$\rho(S_i,\Phi_j)\approx G\cdot m_{i_j}(t)\cdot\frac{1}{n}\sum_{z=0}^{n-1}(\Phi_{j_z})^2=G\cdot m_{i_j}(t)\ . \qquad (3.15)$$

Згідно з правилом виділення корисного сигналу:

$$x(t)=\begin{cases}\text{"1"},\text{при polarity}>0;\\ \text{"0"},\text{при polarity}<0;\\ \text{сторонній сигнал},\text{при polarity}=0,\end{cases} \qquad (3.16)$$

значення $m_{i_j}(t)$ можуть бути легко відновлені за допомогою знакової функції (polarity - полярність пику кореляційної функції).

Оскільки $G>0$ і $n>0$ знак $\rho(S_i,\Phi_j)$ в (3.15) залежить тільки від $m_{i_j}(t)$, звідки маємо:

$$m_{i_j}(t)=\text{sign}\left(\rho(S_i,\Phi_j)\right)=\begin{cases}-1,\text{при}\,\rho(S_i,\Phi_j)<0;\\ +1,\text{при}\,\rho(S_i,\Phi_j)>0;\\ ?,\text{при}\,\rho(S_i,\Phi_j)=0;\end{cases} \qquad (3.17)$$

Якщо $\rho(S_i,\Phi_j)=0$ в (3.17) вважатимемо, що вбудована інформація була втрачена.

Структурна схема вбудовування інформації в контейнер-зображення з використанням прямого розширення спектру для скритної передачі повідомлень представлена на рис.3.5.

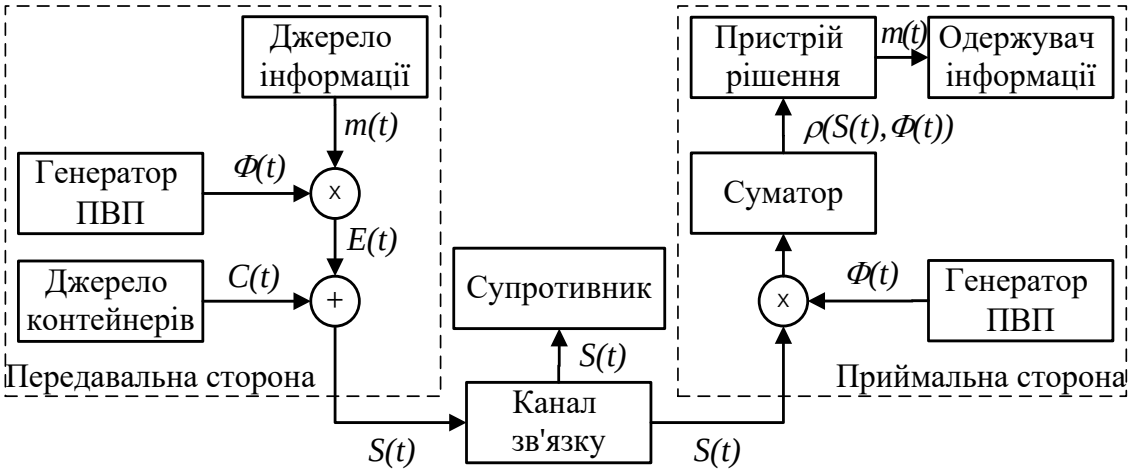


Рисунок 3.5 - Структурна схема вбудовування інформації в контейнер-зображення для скритної передачі повідомлень

Рис. 3.5. показує, то що процес вбудовування інформаційних повідомлень для скритної передачі дуже схожий на процес розширення спектру дискретних сигналів в системах зв'язку (див. рис. 3.3). Поелементне складання модульованого повідомлення $E(t)$ з контейнером-зображенням $C(t)$ (див. вираз (3.12)) слід інтерпретувати як накладення помилок $e(t)$ на корисний сигнал в каналі зв'язку $y(t)$. Завдання вибудовування повідомлення $m(t)$ з $S(t)$ на приймальній стороні стеганосистеми еквівалентно завданню детектування $x(t)$ з суміші корисного сигналу і завади $y'(t) = y(t) + e(t)$ в широкосмуговій системі зв'язку. Іншими словами, розглянута стеганосистема успадковує всі переваги широкосмугових систем зв'язку: стійкість до несанкціонованого добування вбудованих повідомлень (аналог скритності в системі зв'язку), стійкість до руйнування або модифікації вбудованих повідомлень (аналог завадозахисту), стійкість до нав'язування помилкових повідомлень (аналог імітостійкості в системі зв'язку).

Таким чином, використання прямого розширення спектру дискретних сигналів дозволяє здійснити вбудовування інформаційних даних в нерухомі зображення для скритної передачі і реалізувати, таким чином, стеганографічний захист інформації.

Оцінка ефективності стеганосистеми

Під ефективністю технічної системи в широкому сенсі розуміють відповідність результату виконання деякої операції потрібному параметру.

При цьому технічна система виступає в ролі засобу реалізації досліджуваної операції.

Стосовно даного процесу стеганографічна система виступає в ролі технічного засобу реалізації операції, метою якої є захована від супротивника факту здійснення скритної передачі інформації. Таким чином, з урахуванням функціонального призначення стеганосистеми, введемо наступні показники ефективності.

1. Пропускна спроможність – відношення об'єму V вбудованої в контейнер інформації до загального об'єму D контейнеру

$$Q = \frac{V}{D}. \tag{3.18}$$

2. Об'єм ключових даних (у бітах)

$$\ell_{Key} = \log_2(|Key|), \tag{3.19}$$

де $|Key|$ - потужність множини ключових даних.

3. Стійкість стеганографічного методу оцінюватимемо як величину, зворотну потужності множини секретних ключових даних. Її можна трактувати як імовірнісний показник підбору секретного ключа:

$$W = \frac{1}{|Key|} = 2^{-l_{Key}}. \tag{3.20}$$

4. Величина спотворень, що вносяться, як процентне відношення середньоарифметичного всіх абсолютних значень Δ - змін даних контейнера до максимально можливого значення Δ_{max}:

$$I = \frac{\Delta_{ср}}{\Delta_{max}} \cdot 100 = \frac{100}{\Delta_{max} \cdot D} \cdot \sum_{i=1}^{D} |\Delta_i|, \tag{3.21}$$

де Δ_i – Δ-зміни i-го елементу контейнеру.

5. Імовірність помилкового вибудовування інформаційних даних повідомлення

$$P_{ош} = \lim_{D\to\infty} \frac{V_{ош}}{D} = 1 - \lim_{D\to\infty} \frac{V - V_{ош}}{D}, \tag{3.22}$$

де $V_{ош}$ – об’єм помилково вибудованих даних.

Використовуючи (3.18) – (3.22) оцінимо ефективність розглянутого стеганографічного методу захисту інформації.

1. *Пропускна спроможність.* На кожен n-елементний блок S_i заповненого контейнера (стеганограми) доводиться n-бітовий вектор вбудованого повідомлення m_i (див. вирази (3.11) (3.12)). Отже, $Q = \frac{1}{B}$, де B-об'єм даних, що доводиться на один елемент контейнера. Для випадку вбудовування в растрові дані зображення (колірна модель R,G,B) з 8 бітовим кодуванням кожного кольору маємо $B = 8$ і $Q = \frac{1}{8}$.

2. *Об'єм ключових даних.* Ключовими даними є ансамбль дискретних сигналів, утворений рядками матриці Адамара порядку n. Отже, під множиною ключових даних слід розуміти множину різних (неізоморфних) матриць Адамара, кожна з матриць задає ансамбль дискретних сигналів. В таблиці 3.1 приведені деякі оцінки потужності M_A цієї множини.

Таблиця 3.1

Число ансамблів дискретних сигналів Уолша-Адамара

n	M_A
64	19
100	1
256	54
512	102
1024	162
2000	9
4000	16
10000	10

Приведені оцінки потужності M_A дають оцінку числа ансамблів дискретних сигналів Уолша-Адамара, тобто оцінку потужності нееквівалентних ключів стеганосистеми. Отже, об'єм ключових даних оцінюється як $l_{key} = \log_2(M_A)$.

3. *Імовірність підбору секретного ключа* $W = (M_A)^{-1}$.

4. Для оцінки *величини спотворень, що вносяться,* скористаємось виразом (3.12). Другий доданок в правій частині (3.12) визначає величину Δ – змін елементів даних контейнеру. Співмножник E_i формується в результаті підсумовування n дискретних сигналів (що приймають значення ± 1) з відповідними полярностями (що задаються $m_{i_j}(t)$). Отже, всі елементи E_i прийматимуть значення з діапазоні $[-n,...,+n]$, а відповідні Δ –зміни

елементів контейнеру не перевищуватимуть $|\Delta_i| \le n \cdot G$. Звідки маємо верхню оцінку величини спотворень, що вносяться:

$$I = \frac{\Delta_{ср}}{\Delta_{max}} \cdot 100 \le \frac{n \cdot G}{\Delta_{max}} \cdot 100. \qquad (3.23)$$

Для випадку вбудовування в растрові дані зображення (колірна модель R,G,B) з 8 бітовим кодуванням кожного кольору і використання дискретних сигналів з $n = 256$ навіть при $G = 1$ спотворення, що вносяться, можуть досягати 100%. Знизити спотворення, що вносяться, можна за рахунок скорочення числа вбудованих біт даних m_{i_j} (зменшивши число доданків в (6.11)), що неминуче приведе до зниження пропускної спроможності стеганографічного каналу зв'язку.

5. *Імовірність помилкового добування.* Добування інформаційного повідомлення, також як і при організації завадозахисного зв'язку (див. (3.6) – (3.9)), здійснюється кореляційним способом (див. (3.12) – (3.15)). Отже, помилка добування відбудеться при зміні знаку коефіцієнту кореляції $\rho(S_i, \Phi_j)$ у виразі (3.17).

Представимо коефіцієнт $\rho(S_i, \Phi_j)$ у вигляді:

$$\rho(S_i, \Phi_j) = \rho(C_i + E_i \cdot G, \Phi_j) = \rho(C_i, \Phi_j) + \rho(E_i \cdot G, \Phi_j).$$

Останній доданок не змінює знак $\rho(S_i, \Phi_j)$, подія $\rho(S_i, \Phi_j) = \rho(E_i \cdot G, \Phi_j)$ відповідає безпомилковому добуванню повідомлення (див. (3.16) (3.17)).

Отже, помилка добування інформаційного біту m_{i_j} повідомлення відбудеться при настанні події:

$$\left|\rho(C_i, \Phi_j)\right| > \rho\left|E_i \cdot G, \Phi_j\right| = \left|G \cdot m_{i_j}\right| = G, \qquad (3.24)$$

тобто у тому випадку, коли абсолютне значення коефіцієнта кореляції, що був використаний для вбудовування біту m_{i_j} дискретного сигналу Φ_j з блоком контейнеру C_i, в який цей біт вбудовується, перевершить коефіцієнт посилення G.

Таким чином, запишемо:

$$P_{ош} = P\left(\left|\rho(C_i, \Phi_j)\right| > G\right)$$

де $P(x)$- імовірність настання випадкової події x.

Іншими словами, правильне добування вбудованого повідомлення є випадковою подією, імовірність $P_{б.ош}$ якої безпосередньо пов'язана із статистичними властивостями використовуваного контейнера-зображення. Для безпомилкового добування повідомлення

$$P_{ош} = 0, P_{б.ош} = 1 - P_{ош} = 1, \qquad (3.25)$$

слід прагнути до взаємної ортогональності окремих фрагментів зображення C_i і використовуваних як секретні ключі дискретних сигналів Φ_j. В цьому випадку подія

$$\left|\rho(C_i, \Phi_j)\right| = 0 < G$$

для всіх $i = 0,..., N-1$ є достовірним і виконується (3.25).

В той же час, як показали експериментальні дослідження, коефіцієнт кореляції, як правило, значно більше нуля $\left|\rho(C_i, \Phi_j)\right| >> 0$ і дуже часто виникає подія (3.24). Річ у тому, що елементи дискретних сигналів $\Phi_j \in \Phi$ приймають значення $\begin{cases} +1 \\ -1 \end{cases}$, а відповідний нормований коефіцієнт кореляції $\rho(\Phi_i, \Phi_j)$ по абсолютному значенню не перевершує довжини n послідовності і лежить в діапазоні $[0,...,1]$, звідки власне і слідує умова (3.24).

Проте елементи контейнеру C_i приймають значення з числового поля $[0,..., Y]$, розмірність якого задається способом кодування даних зображення. Наприклад, при вбудовуванні інформації в растрові дані зображення (колірна модель R,G,B) з 8 бітовим кодуванням кожного кольору відповідні C_i приймають значення з діапазону цілих чисел $[0,...,255]$. Іншими словами, абсолютне значення нормованого щодо n коефіцієнту кореляції $\left|\rho(C_i, \Phi_j)\right|$ лежатиме в діапазоні $[0,..., Y]$ і для безпомилкового добування всіх біт повідомлення (3.24) необхідно виконати умову $G > Y$.

Як показали дослідження підвищення G веде до неминучого зростання величини спотворень (3.23), що вносяться. При $I > 2...3\%$ (поріг зорової чутливості людини) вони стають помітні сторонньому спостерігачу, що компрометує стеганоканал і робить неможливим використання розглянутої стеганосистеми.

Таким чином, в ході досліджень виявлені наступні суперечності, які лежать в основі розробки і використанні стеганографічних систем з розширенням спектру дискретних сигналів:

- імовірність правильного добування вбудованих даних $P_{б.ош}$ лежить в прямій залежності від величини спотворень I, що вносяться ;

- величина спотворень I, що вносяться, лежить в прямій залежності від об'єму вбудованих біт даних, тобто від пропускної спроможності стеганоканалу Q;

- імовірність правильного добування вбудованих даних $P_{б.ош}$ безпосередньо залежить від статистичних властивостей використовуваного контейнеру-зображення.

В результаті проведених експериментів, одержані наступні емпіричні оцінки:

- залежності величини спотворень I, що вносяться, від пропускної спроможності Q стеганоканалу;

- залежності величини спотворень I, що вносяться, і частоти помилок добування $P^*_{ош} \approx P_{ош}$ від коефіцієнта посилення G;

- залежності величини спотворень I, що вносяться, від частоти помилок добування $P^*_{ош} \approx P_{ош}$.

Дослідження проводилися при вбудовуванні інформаційних даних в растрові дані зображення (колірна модель R,G,B) з 8 бітовим кодуванням кожного кольору. Одержані емпіричні залежності приведені на рис. 3.6 – 3.9.

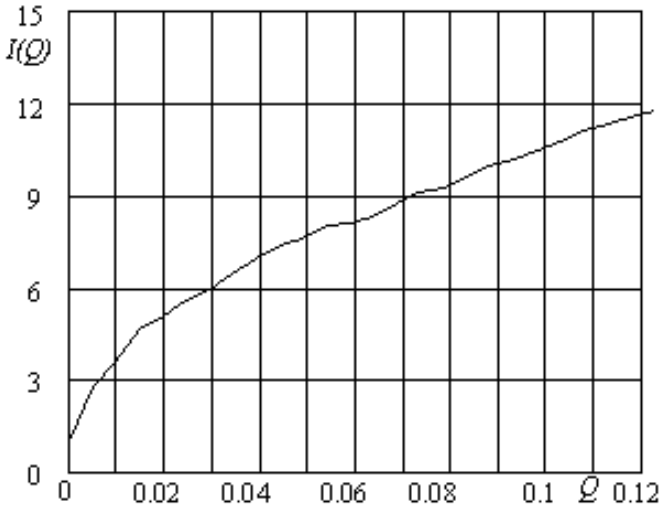


Рисунок 3.6 - Залежність I(Q)

Аналіз експериментально одержаних залежностей підтверджує зроблені раніше висновки, збіжність результатів експерименту з теоретичними міркуваннями свідчить про достовірність отриманих результатів.

З приведеної на рис. 3.6. залежності виходить, що підвищення пропускної спроможності стеганоканалу веде до різкого збільшення спотворень, що вносяться, в контейнер-зображення. Непомітні для стороннього спостерігача спотворення (лежачі нижче за поріг чутливості зорової системи людини) вносяться лише при $Q \le 0.005$. Це відповідає

вбудовуванню не більше 10 бітів в один блок зображення, тобто модуляції до десяти інформаційних сигналів $m_{i_j}(t)$, $j = 0,...,9$ у виразі (6.11).

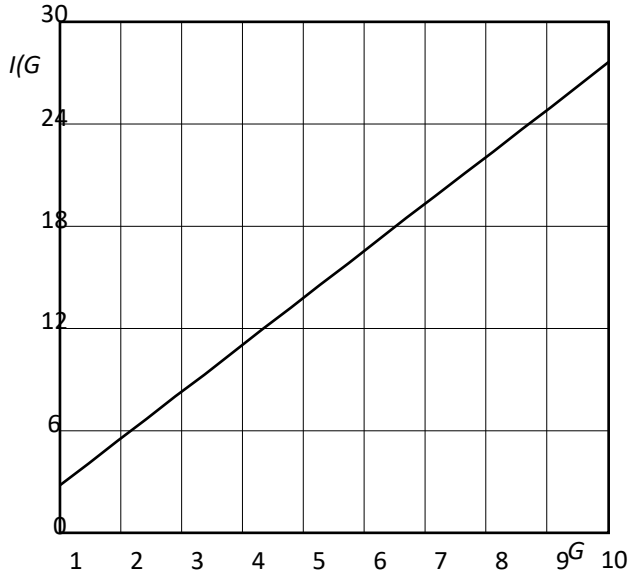


Рисунок 3.7 - Залежність $I(G)$ при $Q = 0.005$

Залежності, приведені на рис. 3.7, 6.8 свідчать, що коефіцієнт посилення, що був використаний у виразах (3.12) - (3.14) дозволяє істотно понизити імовірність помилкового добування інформаційних даних. На жаль, це досягається за рахунок різкого підвищення спотворень, що вносяться, у контейнер-зображення. Залежності одержані при $Q = 0.005$. Очевидно, що для такої величини пропускної спроможності коефіцієнт посилення не може перевершувати 1 .. 1,5 (див. рис. 3.7). Проте навіть для таких значень імовірність помилкового добування велика і лежить в діапазоні 0,1 .. 0,5.

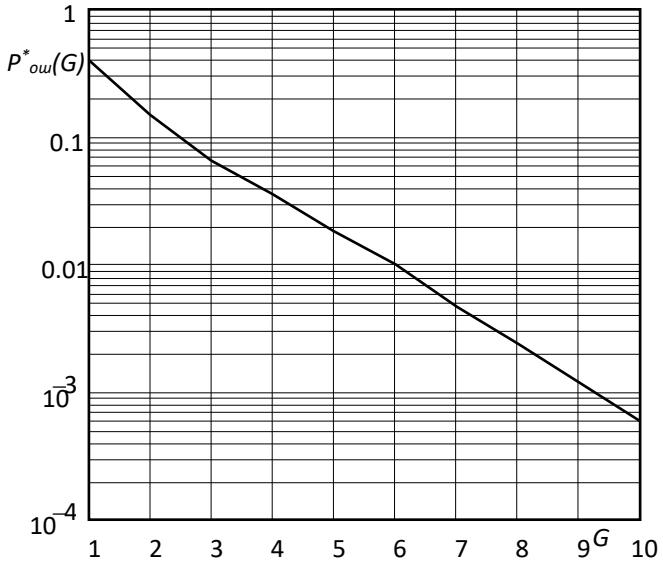


Рисунок 3.8 - Залежність $P^*_{ош}(G)$ при $Q = 0.005$

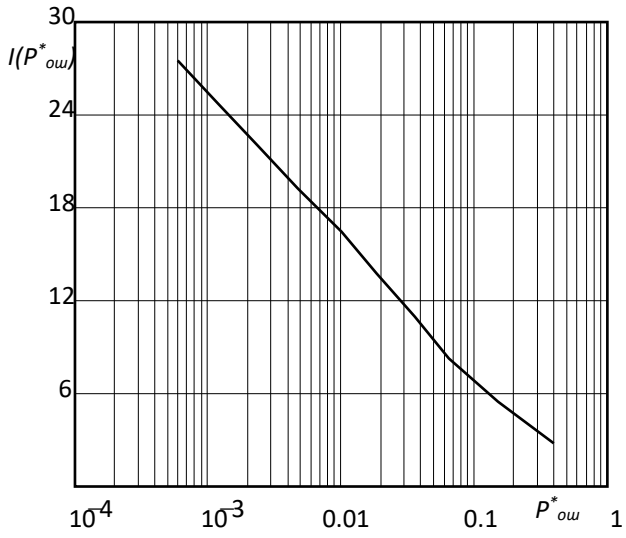


Рисунок 3.9 - Залежність $I(P^*_{ош})$ при $Q = 0.005$

Інтегральна залежність $I(P^*_{ош})$, яка приведена на рис. 3.9, узагальнює приведені на рис. 3.7, 3.8 дані. Для фіксованої пропускної спроможності $Q = 0.005$ одержана емпірична крива, яка характеризує залежність величини спотворень, що вносяться в контейнер-зображення і імовірність помилкового добування інформаційних даних. Для $Q = 0.005$ добитися низьких спотворень, які лежать нижче за поріг зорової чутливості людини ($I \le 2...3\%$), можна тільки при дуже високій імовірності помилкового добування інформаційних даних ($P_{ош} \ge 0.1$). Очевидно, що практичне застосування подібних стеганосистем необхідно поєднувати з завадостійким кодуванням інформаційних даних, що дозволить істотно понизити $P_{ош}$.

В результаті проведених досліджень показано, що використання в стеганографічних цілях прямого розширення спектру дискретних сигналів дозволяє здійснити скритне вбудовування інформаційних повідомлень в нерухомі зображення. Завдання добування повідомлення на приймальній стороні стеганосистеми еквівалентне завданню виявлення інформації з суміші корисного сигналу і завади в широкосмуговій системі зв'язку.

В ході досліджень виявлені наступні недоліки стеганографічних систем з розширенням спектру дискретних сигналів: імовірність правильного добування вбудованих даних залежить від величини спотворень, що вносяться, яка в свою чергу залежить від забезпечуваної пропускної спроможності стеганоканалу. Інакше кажучи, практична побудова стеганосистеми зв'язана з пошуком компромісу між величиною спотворень, що вносяться, імовірністю правильного добування повідомлення на приймальній стороні і забезпечуваною пропускною спроможністю. Крім того, в ході досліджень встановлено, що імовірність правильного добування вбудованих даних безпосередньо залежить від статистичних властивостей контейнера-зображення, що використовується.

4. Питання для поточного контролю підготовленості студентів до виконання лабораторної роботи №3

1. Методи розширення спектру, які застосовуються для завадозахищеної передачі повідомлень. Переваги систем зв'язку з розширеним спектром.
2. Пряме розширення спектру в системах зв'язку. Кодовий розподіл каналів. Використання ортогональних дискретних сигналів в системах CDMA.
3. Матриці Адамара. Формування ортогональних дискретних сигналів Уолша-Адамара. Ансамблеві та кореляційні властивості сигналів Уолша-Адамара.
4. Кореляційний прийом дискретних сигналів. Структурна схема та математична модель системи зв'язку з кореляційним прийомом дискретних сигналів.
5. Метод приховування даних у просторової області нерухомих зображень на основі прямого розширення спектру. Використання ансамблів ортогональних дискретних сигналів Уолша-Адамара в якості таємного ключа для приховування даних.
6. Структурна схема та математична модель стеганографічної системи з приховуванням даних у просторової області нерухомих зображень на основі прямого розширення спектру. Вилучення вбудованих даних за допомогою кореляційного приймача дискретних сигналів.
7. Квазіортогональні дискретні сигнали. Похідні ортогональні сигнали. Застосування квазіортогональних дискретних сигналів для побудови ефективних стеганографічних систем.
8. Ймовірнісні властивості методу приховування даних у просторової області нерухомих зображень на основі прямого розширення спектру, залежності ймовірності правильного вилучення даних та частки внесених при цьому похибок у контейнер-зображення. Додаткові вимоги до ансамблів дискретних сигналів, що застосовуються в стеганографічному перетворенні.
9. Адаптоване до властивостей контейнера формування квазіортогональних дискретних сигналів. Приховування та вилучення даних із адаптовано формованими квазіортогональними дискретними сигналами, їх вплив на ймовірність правильного вилучення даних та частку внесених при цьому похибок у контейнер-зображення.

5. Інструкція до виконання лабораторної роботи №3

Завдання 1. Реалізація в середовищі MathCAD алгоритмів формування ансамблів ортогональних дискретних сигналів Уолша-Адамара та алгоритмів кодування інформаційних бітів даних складними дискретними сигналами

1.1. Завантажуємо вихідні дані: контейнер - нерухоме зображення (в форматі *.bmp24); інформаційне повідомлення - текстовий документ (у форматі *.txt). Для цього в середовищі MathCAD виконуємо наступні дії, аналогічні п. 1.1. інструкції до лабораторної роботи №1.

1.2. Перетворюємо масив інформаційних даних. Для цього в середовищі MathCAD виконуємо наступні дії, аналогічні п. 1.2. інструкції до лабораторної роботи №1.

1.3. Реалізуємо алгоритм формування матриць Адамара. Для цього скористаємося наступною процедурою:

$H_0 := (1)$

H :=
for i ∈ 1..8
  F ← H_{i-1}
  for j ∈ 0..rows(F) − 1
    for jj ∈ 0..cols(F) − 1
      a ← $F_{jj,j}$
      $F1_{jj,j}$ ← a
  for j ∈ 0..rows(F) − 1
    for jj ∈ 0..cols(F) − 1
      a ← $F_{jj,j}$
      $F1_{jj+cols(F),j}$ ← a
  for j ∈ 0..rows(F) − 1
    for jj ∈ 0..cols(F) − 1
      a ← $F_{jj,j}$
      $F1_{jj,j+rows(F)}$ ← a
  for j ∈ 0..rows(F) − 1
    for jj ∈ 0..cols(F) − 1
      a ← $F_{jj,j}$
      $F1_{jj+cols(F),j+rows(F)}$ ← −a
  H_i ← F1
H

Процедура итеративно формує матриці Адамара H_1, H_2, …, H_8. Матриця H_0, що складається з одного елемента, задається в якості початкового значення итеративної процедури. Решта матриць формуються згідно з рекурентним правилом:

$$H_i = \begin{pmatrix} H_{i-1} & H_{i-1} \\ H_{i-1} & -H_{i-1} \end{pmatrix}.$$

Результатом виконання процедури є масив матриць H, кожний елемент якого є матрицею Адамара:

$$H_1 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \qquad H_2 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

$$H_3 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix}$$

$H_8 =$

	0	1	2	3	4	5	6	7	8	9
0	1	1	1	1	1	1	1	1	1	1
1	1	-1	1	-1	1	-1	1	-1	1	-1
2	1	1	-1	-1	1	1	-1	-1	1	1
3	1	-1	-1	1	1	-1	-1	1	1	-1
4	1	1	1	1	-1	-1	-1	-1	1	1
5	1	-1	1	-1	-1	1	-1	1	1	-1
6	1	1	-1	-1	-1	-1	1	1	1	1
7	1	-1	-1	1	-1	1	1	-1	1	-1
8	1	1	1	1	1	1	1	1	-1	-1
9	1	-1	1	-1	1	-1	1	-1	-1	1
10	1	1	-1	-1	1	1	-1	-1	-1	-1
11	1	-1	-1	1	1	-1	-1	1	-1	1
12	1	1	1	1	-1	-1	-1	-1	-1	-1
13	1	-1	1	-1	-1	1	-1	1	-1	1
14	1	1	-1	-1	-1	-1	1	1	-1	-1
15	1	-1	-1	1	-1	1	1	-1	-1	…

1.4. Реалізуємо алгоритм формування ансамблів ортогональних дискретних сигналів Уолша-Адамара. Для цього сформуємо масив рядків матриці Адамара, кожен елемент сформованого таким чином масиву є дискретним сигналом Уолша-Адамара:

```
ArrayFunction := | for i ∈ 0..255
                 |     for j ∈ 0..255
                 |         a_j ← (H_8)_{i,j}
                 |     ArrayFunction_i ← a
                 | ArrayFunction
```

Розглянемо, як приклад, кілька дискретних сигналів:

$ArrayFunction_5 =$

	0
0	1
1	-1
2	1
3	-1
4	-1
5	1
6	-1
7	1
8	1
9	…

$ArrayFunction_{45} =$

	0
0	1
1	-1
2	1
3	-1
4	-1
5	1
6	-1
7	1
8	-1
9	…

1.5. Реалізуємо алгоритм кодування інформаційних бітів даних складними дискретними сигналами. Для цього сформуємо модульоване інформаційне повідомлення, для чого перетворимо масив інформаційних бітів в масив, що складається з «1» і «-1» такою процедурою:

```
m := | for i ∈ 0..rows(M_b) − 1
     |     m_i ← 1  if M_b_i = 1
     |     m_i ← −1  if M_b_i = 0
     | m
```

В результаті отримаємо масив m, порівняємо його з вихідним масивом даних:

m =

	0
0	-1
1	-1
2	-1
3	1
4	-1
5	-1
6	1
7	1
8	1
9	-1
10	1
11	1
12	-1
13	1
14	1
15	…

M_b =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	…

Вбудовувати повідомлення в контейнер будемо по рядках (кілька бітів в один рядок контейнера). Кодування складними дискретними сигналами зробимо наступним чином:

```
k := 4        g := 1
Sum := | for i ∈ 0..rows(R) − 1
       |     a ← Σ_{j=0}^{k−1} [g·(m_{k·i+j}·ArrayFunction_{j+1})]
       |     Sum_i ← a
       | Sum
```

Значення «k» задає число інформаційних бітів, вбудованих в один фрагмент (в один рядок) контейнера. Значення «g» задає «енергію» вбудованих бітів

повідомлення, тобто фактично є коефіцієнтом посилення вбудованого повідомлення. В даному випадку кодування складними сигналами проводиться без посилення (g = 1). Саме кодування полягає в множенні модульованого повідомлення на сформовані вище дискретні сигнали Уолша-Адамара. Результатом виконання процедури кодування є масив «Sum»:

$Sum_0 =$

	0
0	-2
1	2
2	2
3	2
4	-4
5	0
6	0
7	0
8	-2
9	2
10	2
11	2
12	-4
13	0
14	0
15	…

$Sum_{57} =$

	0
0	2
1	2
2	-2
3	2
4	0
5	0
6	-4
7	0
8	2
9	2
10	-2
11	2
12	0
13	0
14	-4
15	…

Кожен елемент масиву представляє собою суму добутків k модульованих повідомлень і дискретних сигналів Уолша-Адамара. Всього масив «Sum» містить «Rows (R)» елементів за числом рядків контейнера. Кожен елемент масиву «Sum» призначений для вбудовування в окремий рядок контейнера-зображення. Максимальне абсолютне значення елементів масиву «Sum» задає максимальну величину внесених спотворень при вбудовуванні повідомлення. Ця величина не буде перевершувати g·k, тобто величина внесених спотворень безпосередньо визначається коефіцієнтом посилення інформаційного повідомлення і числом вбудованих бітів даних в один фрагмент зображення.

Завдання 2. Реалізація у середовищі символьної математики MathCAD алгоритмів приховування та вилучення даних у просторовій області зображень шляхом прямого розширення спектрів із використанням ортогональних дискретних сигналів

2.1. Реалізуємо алгоритм вбудовування інформаційних даних в просторову область зображення на основі прямого розширення спектра з використанням ортогональних дискретних сигналів Уолша-Адамара:

```
S := | for i ∈ 0..rows(R) − 1
     |   for j ∈ 0..cols(R) − 1
     |     S_{i,j} ← R_{i,j} + (Sum_i)_j
     |     S_{i,j} ← 255 if S_{i,j} > 255
     |     S_{i,j} ← 0 if S_{i,j} < 0
     | S
```

Процедура вбудовування полягає в підсумовуванні даних контейнера (цифрового зображення) з модульованими складними дискретними сигналами інформаційного повідомлення.

Використовувані обмеження призначені для обліку можливостей виходу значень окремих елементів заповненого контейнера за діапазон допустимих значень яскравості окремих пікселів зображення. В результаті виконання процедури вбудовування даних отримуємо масив заповненого контейнера (стеганограмму). Порівняємо масив даних порожнього і заповненого контейнера:

Результат порівняння показує, що максимальні вносимі зміни в контейнер-зображення не перевершують величини g·k = 4. Так, наприклад, значення яскравості червоного кольору окремого пікселя $R_{1,3}$ = 86 в ході вбудовування інформації змінилося на значення $S_{1,3}$ = 90, тобто абсолютна зміна яскравості червоного кольору в даному пікселі дорівнює максимальному значенню. Здебільшого зміни яскравості окремих пікселів зображення знаходяться нижче цього, порогового значення.

S =

	0	1	2	3	4	5
0	84	81	74	74	68	69
1	110	97	90	90	76	69
2	134	118	114	107	96	84
3	124	118	103	106	103	102
4	129	124	115	116	118	120
5	151	147	148	148	152	151
6	169	160	167	170	175	173
7	191	197	191	191	183	172
8	187	192	194	199	192	189
9	190	188	194	198	194	185
10	195	192	199	200	203	188
11	187	191	200	205	203	199
12	190	194	200	197	204	202
13	181	185	187	186	182	176
14	169	176	174	166	163	167
15	158	164	156	151	156	…

R =

	0	1	2	3	4	5
0	86	79	72	72	72	69
1	110	97	90	86	78	71
2	132	120	112	105	96	88
3	122	116	105	104	103	102
4	131	122	117	118	118	116
5	147	147	148	148	150	153
6	169	164	167	170	173	175
7	189	195	193	189	183	172
8	191	192	194	199	194	187
9	186	188	194	198	192	187
10	195	196	199	200	201	190
11	185	189	202	203	203	199
12	192	196	198	199	204	202
13	177	185	187	186	180	178
14	173	176	174	166	165	165
15	160	162	158	153	156	…

Переглянемо результат вбудовування:

S

R

Зрозуміло, що в результаті вбудовування даних з обраними параметрами (коефіцієнт посилення g і число бітів k, вбудованих в один елемент контейнера) внесені спотворення в контейнер-зображення знаходяться нижче порога чутливості зорової системи людини і не можуть бути візуально виявлені.

Отриманий заповнений масив S записуємо в канал червоного кольору заповненого контейнера-стеганограми. Виконуємо команду

«WRITERGB("Stego_Adamar_1_4.bmp"):=augment(S,G,B)».

В результаті виконання команди система MathCAD формує на фізичному носії новий файл з ім'ям «Stego_Adamar_1_4.bmp».

Для графічного відображення вихідного (порожнього) і заповненого контейнера виконуємо вставку відповідних зображень:

"Stego_Adamar_1_4.bmp"

"1.bmp"

Переконуємося в відсутності видимих спотворень.

2.2. Реалізуємо алгоритм кореляційного прийому дискретних сигналів. Для цього скористаємося наступною функцією, яка призначена для розрахунку коефіцієнта кореляції:

$$\begin{aligned} &MultString(A,B) := \\ &\quad X \leftarrow 0 \\ &\quad \text{for } i \in 0..255 \\ &\quad\quad X \leftarrow X + A_i \cdot B_i \\ &\quad X \end{aligned}$$

Наведена функція «MultString(A,B)» обчислює скалярний добуток векторів «A» і «B», результатом є коефіцієнт кореляції аргументів функції.

Для перевірки правильності обчислень виконуємо розрахунок коефіцієнта кореляції двох ортогональних векторів. Для цього запишемо:

$$\mathrm{MultString}\left(\mathrm{ArrayFunction}_2, \mathrm{ArrayFunction}_3\right) = 0$$

Зрозуміло, що використання в якості аргументів функції скалярного добутку двох ортогональних сигналів Уолша-Адамара призводить до нульового результату, що підтверджує правильність роботи реалізованої функції.

Розглянемо тепер масив модульованих інформаційних даних «m», масив даних - результат кодування інформаційних даних складними сигналами «Sum», а також самі складні сигнали, що використовуються при встановленні інформації «ArrayFunction1», «ArrayFunction2», «ArrayFunction3», «ArrayFunction4»:

$m =$

	0
0	-1
1	-1
2	-1
3	1
4	-1
5	-1
6	1
7	…

$\mathrm{Sum}_0 =$

	0
0	-2
1	2
2	2
3	2
4	-4
5	0
6	0
7	…

$\mathrm{ArrayFunction}_1 =$

	0
0	1
1	-1
2	1
3	-1
4	1
5	-1
6	1
7	…

$\mathrm{ArrayFunction}_2 =$

	0
0	1
1	1
2	-1
3	-1
4	1
5	1
6	-1
7	…

$\mathrm{ArrayFunction}_3 =$

	0
0	1
1	-1
2	-1
3	1
4	1
5	-1
6	-1
7	…

$\mathrm{ArrayFunction}_4 =$

	0
0	1
1	1
2	1
3	1
4	-1
5	-1
6	-1
7	…

Обчислимо коефіцієнт кореляції масиву «Sum_0» з усіма чотирма ортогональними сигналами, що використовуються при встановленні інформаційних даних. Отримаємо:

$$\mathrm{MultString}\left(\mathrm{Sum}_0, \mathrm{ArrayFunction}_1\right) = -256$$

$$\mathrm{MultString}\left(\mathrm{Sum}_0, \mathrm{ArrayFunction}_2\right) = -256$$

$$\mathrm{MultString}\left(\mathrm{Sum}_0, \mathrm{ArrayFunction}_3\right) = -256$$

$$\mathrm{MultString}\left(\mathrm{Sum}_0, \mathrm{ArrayFunction}_4\right) = 256$$

Зрозуміло, що знак коефіцієнта кореляції збігається з першими чотирма елементами модульованого інформаційного повідомлення «m».

Наступний елемент інформаційних даних після модуляції відповідає масиву «Sum_1» та ортогональному сигналу «ArrayFunction1». Перевіримо правильність роботи алгоритму:

$$MultString\left(Sum_1, ArrayFunction_1\right) = -256$$

Знак коефіцієнта кореляції в даному випадку також співпадає з вбудовуваним елементом даних «m_4».

2.3. Реалізуємо алгоритм вилучення інформаційних даних з просторової області зображення на основі прямого розширення спектра з використанням ортогональних дискретних сигналів Уолша-Адамара. Для цього в тому ж вікні середовища MathCAD виконуємо команди читання растрових даних нерухомого зображення з заданого файлу (файлу заповненого контейнера) у вигляді двовимірного масиву цілих чисел. Для розглянутого прикладу виконуємо команди:

«C1:=READRGB(“Stego_Adamar_1_4.bmp”)»,
«R1:=READ_RED(“Stego_Adamar_1_4.bmp”)»,
«G1:=READ_GREEN(“Stego_Adamar_1_4.bmp”)»,
«B1:=READ_BLUE(“Stego_Adamar_1_4.bmp”)».

Отримуємо наступний результат:

R1 =

	0	1	2	3	4
0	84	81	74	74	68
1	110	97	90	90	76
2	134	118	114	107	96
3	124	118	103	106	103
4	129	124	115	116	118
5	151	147	148	148	152
6	169	160	167	170	175
7	191	197	191	191	…

R1

Далі формуємо масив рядків заповненого контейнера такою процедурою:

```
ArrayString := | for i ∈ 0..rows(R1) − 1
               |   for j ∈ 0..cols(R1) − 1
               |     a_j ← R1_{i,j}
               |   ArrayString_i ← a
               | ArrayString
```

В результаті отримуємо масив рядків, в кожну з яких за допомогою k ортогональних дискретних сигналів Уолша-Адамара вбудовано k інформаційних бітів повідомлення:

$ArrayString_0 =$

	0
0	84
1	81
2	74
3	74
4	68
5	69
6	71
7	…

$ArrayString_1 =$

	0
0	110
1	97
2	90
3	90
4	76
5	69
6	68
7	…

$ArrayString_2 =$

	0
0	134
1	118
2	114
3	107
4	96
5	84
6	81
7	…

Для вилучення вбудованого повідомлення скористаємося процедурою, заснованої на розглянутій вище функції обчислення коефіцієнта кореляції «MultString»:

```
m1 := | for i ∈ 0..rows(R1) − 1
      |   for j ∈ 0..k − 1
      |     m1_{k·i+j} ← 1  if MultString(ArrayString_i, ArrayFunction_{j+1}) > 0
      |     m1_{k·i+j} ← −1 if MultString(ArrayString_i, ArrayFunction_{j+1}) ≤ 0
      | m1
```

Правило вилучення окремих елементів повідомлення полягає в зіставленні результату обчислення коефіцієнта кореляції з граничним значенням «0». У кожному рядку контейнера вбудовано k елементів повідомлення, тобто для кожного рядка k раз виконуємо обчислення коефіцієнта кореляції.

В результаті маємо масив даних «m1», в якому містяться витягнуті дані. Порівняємо вбудовані дані з витягнутими:

m1 =

	0
0	-1
1	-1
2	-1
3	1
4	-1
5	-1
6	1
7	1
8	1
9	-1
10	1
11	1
12	-1
13	1
14	1
15	…

m =

	0
0	-1
1	-1
2	-1
3	1
4	-1
5	-1
6	1
7	1
8	1
9	-1
10	1
11	1
12	-1
13	1
14	1
15	…

2.4. Для перетворення витягнутих даних в бітову форму використовуємо процедуру:

```
M_b1 := | for i ∈ 0..rows(m1) − 1
        |     M_b1_i ← 1 if m1_i = 1
        |     M_b1_i ← 0 if m1_i = −1
        | M_b1
```

В результаті отримаємо бітовий масив даних. Порівняємо його з вбудованим бітовим масивом:

M_b1 =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	…

M_b =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	…

Порівняння масивів вбудованих і витягнутих даних підтверджує правильність роботи алгоритмів вбудовування-вилучення.

Завдання 3. Проведення експериментальних досліджень ймовірносних властивостей реалізованого методу, отримання емпіричних залежностей ймовірності правильного вилучення даних та частки внесених при цьому похибок у контейнер-зображення

3.1. Проведемо оцінку ймовірності правильного вилучення повідомлення і величини внесених спотворень як від пропускної здатності стеганоканалу (задається величиною k), так і від коефіцієнта посилення g.

Першу емпіричну залежність побудуємо таким чином. Зафіксуємо g=1, і при цьому значенні будемо послідовно збільшувати величину k. Для кожного значення k розрахуємо частоту Posh помилково витягнутих інформаційних бітів. Одночасно будемо розраховувати усереднену величину w внесених спотворень, виражену у відсотковому співвідношенні до максимального значення яскравості. Використаємо для цього наступні процедури:

$$\text{Posh} := \left| \begin{array}{l} a \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(M\_b1) - 1 \\ \quad a \leftarrow a + 1 \text{ if } M\_b1_i \neq M\_b_j \\ \text{Posh} \leftarrow \dfrac{a}{\text{rows}(M\_b1)} \\ \text{Posh} \end{array} \right.$$

$$w := \left| \begin{array}{l} w \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(R1) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R1) - 1 \\ \quad\quad w \leftarrow w + \left| R1_{i,j} - R_{i,j} \right| \\ w \leftarrow \dfrac{w \cdot 100}{\text{rows}(R1) \cdot \text{cols}(R1) \cdot 256} \\ w \end{array} \right.$$

Для розглянутого прикладу при g = 1 и k = 4 отримуємо наступні значення:

$$\text{Posh} = 0.093 \qquad w = 0.586$$

Отримані емпіричні дані заносимо у відповідні таблиці:

$$\text{Posh\_k} := \begin{pmatrix} 0 & 0 \\ 1 & 0.006 \\ 2 & 0.053 \\ 4 & 0.093 \\ 8 & 0.121 \\ 16 & 0.126 \\ 32 & 0.145 \\ 64 & 0.148 \\ 128 & 0.148 \\ 255 & 0.15 \end{pmatrix} \qquad \text{W\_k} := \begin{pmatrix} 0 & 0 \\ 1 & 0.39 \\ 2 & 0.39 \\ 4 & 0.586 \\ 8 & 0.871 \\ 16 & 1.244 \\ 32 & 1.723 \\ 64 & 2.385 \\ 128 & 3.286 \\ 256 & 4.5 \end{pmatrix}$$

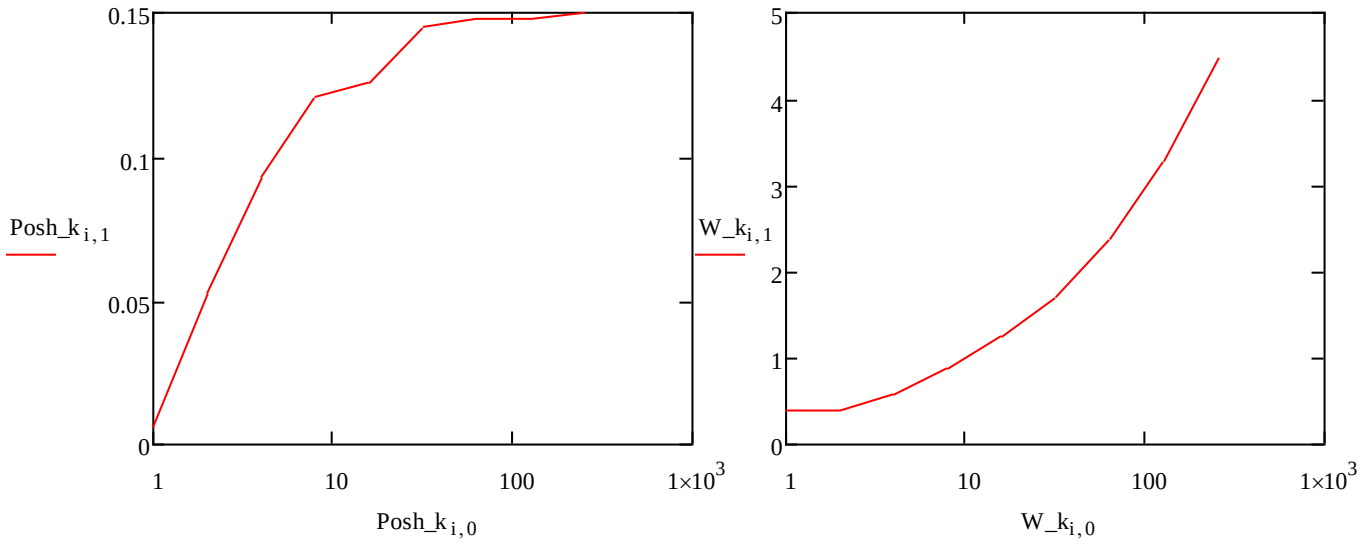
Для побудови другої емпіричної залежності зафіксуємо величину k=4 і, послідовно збільшуючи коефіцієнт g, будемо розраховувати частоту Posh помилково витягнутих інформаційних бітів та усереднену величину w внесених спотворень.

Отримані емпіричні дані заносимо у відповідні таблиці:

$$Posh\_g := \begin{pmatrix} 1 & 0.093 \\ 2 & 0.018 \\ 3 & 0.003 \\ 4 & 0 \end{pmatrix} \qquad W\_g := \begin{pmatrix} 1 & 0.586 \\ 2 & 1.17 \\ 3 & 1.754 \\ 4 & 2.338 \end{pmatrix}$$

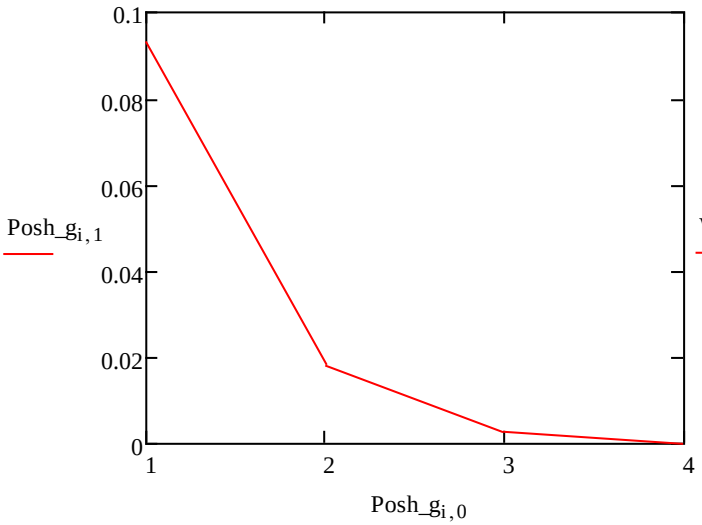
3.2. Побудуємо графіки отриманих емпіричних залежностей (для фіксованого g=1 зі змінним k):

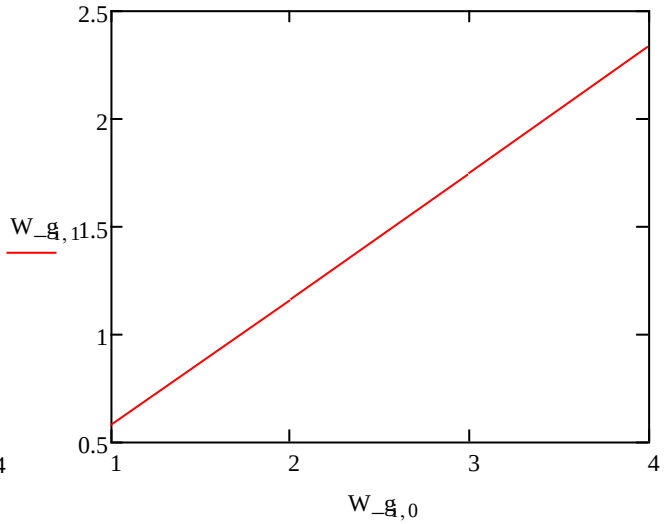
$i := 0..9$



зрозуміло, що підвищення числа вбудованих бітів даних призводить до збільшення як ймовірності помилкового вилучення даних, так і до підвищення частки внесених спотворень в контейнер-зображення. Слід зазначити, що збільшення числа вбудованих бітів на один порядок (з 10 до 100 і вище) призводить до незначного (менше 0,05) збільшення ймовірності помилкового вилучення, в той час як частка внесених спотворень збільшується при цьому в 4-5 разів.

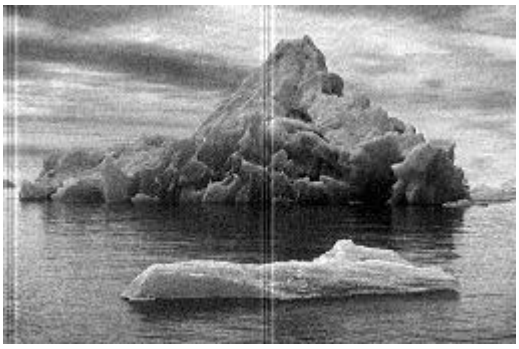
3.3. Побудуємо графіки отриманих емпіричних залежностей (для фіксованого k=4 зі змінним g):





Наведені залежності свідчать, що збільшення коефіцієнта посилення g призводить до різкого зниження ймовірності помилкового вилучення інформаційних бітів даних і одночасного до підвищення величини внесених спотворень. З наведених графіків видно, що при g=4 забезпечується безпомилковий витяг інформаційних даних, частка внесених спотворень в середньому знаходиться нижче порога зорової чутливості людини.

У той же час слід зазначити, що розраховане значення w є усередненою величиною, що характеризує частку внесених спотворень в середньому за всіма пікселям контейнера-зображення. Окремі пікселі або група пікселів можуть бути перекручені дуже сильно, частка внесених спотворень для цих фрагментів зображення може істотно перевищувати розраховане середнє значення w. Для прикладу наведемо контейнер-зображення, заповнений з показниками: k = 128, g = 1



S

R

Як випливає з наведених вище графіків вбудовування з такими параметрами дає усереднене значення частки внесених спотворень в межах порогу зорової чутливості людини. Однак, як видно з наведених зображень для деяких фрагментів спотворення дуже істотні. Позбутися від подібних негативних факторів можливо за допомогою адаптивного формування дискретних сигналів, що враховує особливості використовуваного контейнера-зображення. Крім того, використання при встановленні даних

ортогональних дискретних сигналів Уолша-Адамара не завжди виправдано в стеганографії. Подібні сигнали на окремих ділянках мають вигляд детермінованих послідовностей. Наприклад, сигнал «$ArrayFunction_0$» зовсім не використовувався нами при встановленні інформації, оскільки він складається з послідовності одних одиничних символів. Альтернативою використання ортогональних сигналів Уолша-Адамара є квазіортогональні дискретні послідовності, що мають псевдовипадкову структуру і не містять (в ідеальному випадку) детерміновані ділянки.

Завдання 4. Реалізація у середовищі символьної математики MathCAD алгоритмів формування ансамблів квазіортогональних дискретних сигналів та алгоритмів приховування та вилучення даних в просторової області зображень із використанням квазіортогональних дискретних сигналів

4.1. Реалізуємо алгоритм формування квазіортогональних дискретних сигналів наступним чином:

```
ArrayFunction1 := | for i ∈ 0..1023
                  |   for j ∈ 0..255
                  |     b ← ceil(rnd(2)) − 1
                  |     a_j ← 1  if b = 1
                  |     a_j ← −1  if b = 0
                  |   ArrayFunction1_i ← a
                  | ArrayFunction1
```

Для формування окремих елементів послідовностей використаємо вбудовану функцію генерації псевдовипадкових чисел «rnd()», яка формує раціональне число, яке знаходиться в заданому діапазоні.

Функція «ceil()» округлює отриманий результат до найближчого цілого числа.

Після перетворення «0» в «-1» отримаємо масив «ArrayFunction1», елементами якого є псевдовипадкові послідовності - сформовані дискретні сигнали. Значення коефіцієнта взаємної кореляції сформованих сигналів (в силу псевдовипадковості їх формування) значно не відрізняються від нуля, тобто вважатимемо сформовану множину послідовностей ансамблем квазіортогональних дискретних сигналів.

Так, наприклад, для першого і сьомого дискретного сигналу

$ArrayFunction1_1 =$

	0
0	1
1	1
2	-1
3	1
4	1
5	1
6	-1
7	-1
8	1
9	-1
10	1
11	-1
12	1
13	1
14	-1
15	…

$ArrayFunction1_7 =$

	0
0	-1
1	1
2	1
3	-1
4	-1
5	1
6	1
7	1
8	1
9	-1
10	-1
11	-1
12	-1
13	-1
14	-1
15	…

коефіцієнт взаємної кореляції дорівнює

$MultString\left(ArrayFunction1_2, ArrayFunction1_7\right) = 26$

4.2. Для вбудовування інформаційних повідомлень з використанням сформованого ансамблю квазіортогональних дискретних сигналів розіб'ємо бітовий масив «M_b» на підблоки і сформуємо масив десяткових чисел:

```
M_d := | for i ∈ 0..rows(R) − 1
       |   a ← 0
       |   for j ∈ 0..9
       |     a ← a + M_b[10·i+j]·2^j
       |   M_d[i] ← a
       | M_d
```

$M_d =$

	0
0	456
1	187
2	607
3	963
4	485
5	60
6	674
7	131
8	…

Елементами сформованого масиву «M_d» є десяткові числа, кожне з яких в двійковому поданні відповідає підблоку з десяти бітів масиву «M_b».

4.3. Реалізуємо алгоритм кодування квазіортогональними дискретними сигналами:

$g_x := 4C$

```
Sum1 := | for i ∈ 0..rows(R) − 1
        |   Sum1_i ← g·ArrayFunction1(M_d_i)
        | Sum1
```

$Sum1_0 =$

	0
0	-40
1	-40
2	40
3	40
4	-40
5	-40
6	-40
7	-40
8	…

В результаті виконання наведеної процедури формуємо масив «Sum1», елементами якого є дискретні сигнали з масиву «ArrayFunction1», посилені коефіцієнтами «g». Номери використовуваних сигналів відповідають десятковим поданням інформаційних блоків вбудованого повідомлення.

4.4. Реалізуємо алгоритм вбудовування інформаційного повідомлення в контейнер-зображення за допомогою накладення модульованого повідомлення на масив яскравостей каналу червоного кольору:

```
S1 := | for i ∈ 0..rows(R) − 1
      |   for j ∈ 0..cols(R) − 1
      |     S1_i,j ← R_i,j + (Sum1_i)_j
      |     S1_i,j ← 255 if S1_i,j > 255
      |     S1_i,j ← 0 if S1_i,j < 0
      | S1
```

В результаті виконання наведеної процедури формуємо масив «S1». Порівняємо канали червоного кольору порожнього і заповненого контейнера:

$S1 =$

	0	1	2	3
0	46	39	112	112
1	70	137	50	126
2	172	80	152	145
3	162	76	145	144
4	171	82	157	158
5	107	187	188	108
6	129	124	127	210
7	229	235	153	229
8	151	232	234	239
9	226	148	154	…

$R =$

	0	1	2	3
0	86	79	72	72
1	110	97	90	86
2	132	120	112	105
3	122	116	105	104
4	131	122	117	118
5	147	147	148	148
6	169	164	167	170
7	189	195	193	189
8	191	192	194	199
9	186	188	194	…

S1

R

Після виконання командного запису

```
WRITERGB("Stego_Kvasi.bmp" ) := augment(S1, G, B)
```

отримуємо відповідне контейнер-повідомлення

"Stego_Kvasi.bmp"

"1.bmp"

Зрозуміло, що вбудовування з таким високим значенням коефіцієнта посилення (g = 40) призводить до появи суттєвих перекручень, наочно представлених на наведених зображеннях.

4.5. Реалізуємо алгоритм вилучення повідомлень з використанням квазіортогональних дискретних сигналів. Для цього зробимо зчитування растрових даних з контейнера-зображення:

```
C2 := READRGB("Stego_Kvasi.bmp" )
R2 := READ_RED("Stego_Kvasi.bmp" )
G2 := READ_GREEN("Stego_Kvasi.bmp" )
B2 := READ_BLUE("Stego_Kvasi.bmp" )
```

В результаті отримуємо:

R2 =

	0	1	2	3	4
0	46	39	112	112	32
1	70	137	50	126	38
2	172	80	152	145	136
3	162	76	145	144	63
4	171	82	157	158	78
5	107	187	188	108	190
6	129	124	127	210	133
7	229	235	153	229	…

R2

Сформуємо зі зчитаного масиву червоного кольору «R2» масив рядків контейнера

```
ArrayString1 := | for i ∈ 0..rows(R2) − 1
                |   for j ∈ 0..cols(R2) − 1
                |     a_j ← R2_{i,j}
                |   ArrayString1_i ← a
                | ArrayString1
```

$ArrayString1_0$ =

	0
0	46
1	39
2	112
3	112
4	32
5	29
6	31
7	34
8	…

після чого сформуємо масив десяткових чисел «M_d1», елементами якого будуть номери квазіотрогональних сигналів з масиву «ArrayFunction1», які дають найбільше значення коефіцієнта кореляції з рядками контейнера-зображення:

```
M_d1 := | for i ∈ 0..rows(R1) − 1
        |   a ← 0
        |   for j ∈ 0..1023
        |     if MultString(ArrayString1_i, ArrayFunction1_j) > a
        |       a ← MultString(ArrayString1_i, ArrayFunction1_j)
        |       M_d1_i ← j
        | M_d1
```

Фактично наведена процедура реалізує кореляційний прийом (в термінах статистичної теорії зв'язку).

Порівняємо витягнутий масив «M_d1» з тим масивом, який був вбудований в контейнер-зображення:

M_d =

	0
0	456
1	187
2	607
3	963
4	485
5	60
6	674
7	131
8	753
9	380
10	574
11	899
12	749
13	251
14	782
15	…

M_d1 =

	0
0	456
1	561
2	607
3	561
4	561
5	60
6	674
7	561
8	561
9	561
10	574
11	561
12	561
13	561
14	561
15	…

Зрозуміло, що отриманий масив десяткових чисел відрізняється від вбудованого масиву, що пояснюється, вочевидь, сильною кореляцією використовуваних квазіортогональних дискретних сигналів з окремими елементами контейнера-зображення.

4.6. Перетворимо отриманий масив «M_d1» в двійковий вигляд:

```
M_b2 := for i ∈ 0..rows(M_d1) − 1
          x ← M_d1_i
          for j ∈ 0..9
            M_b2_(i·10+j) ← mod(x, 2)
            x ← floor(x / 2)
        M_b2
```

Отримаємо масив «M_d1». Порівняємо його з вбудованим двійковим масивом «Md»:

M_b2 =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	…

M_b =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	…

4.7. Проведемо оцінку ймовірності правильного вилучення повідомлення і величини внесених спотворень від коефіцієнта посилення g. Для цього розрахуємо частоту помилково отриманих інформаційних бітів і оцінку внесених спотворень в контейнер-зображення:

$$\text{Posh} := \left| \begin{array}{l} a \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(M\_b2) - 1 \\ \quad a \leftarrow a + 1 \text{ if } M\_b2_i \neq M\_b_i \\ \text{Posh} \leftarrow \dfrac{a}{\text{rows}(M\_b2)} \\ \text{Posh} \end{array} \right.$$

$$\text{Posh} = 0.104$$

$$w := \left| \begin{array}{l} w \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(R2) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R2) - 1 \\ \quad\quad w \leftarrow w + \left| R2_{i,j} - R_{i,j} \right| \\ w \leftarrow \dfrac{w \cdot 100}{\text{rows}(R2) \cdot \text{cols}(R2) \cdot 256} \\ w \end{array} \right.$$

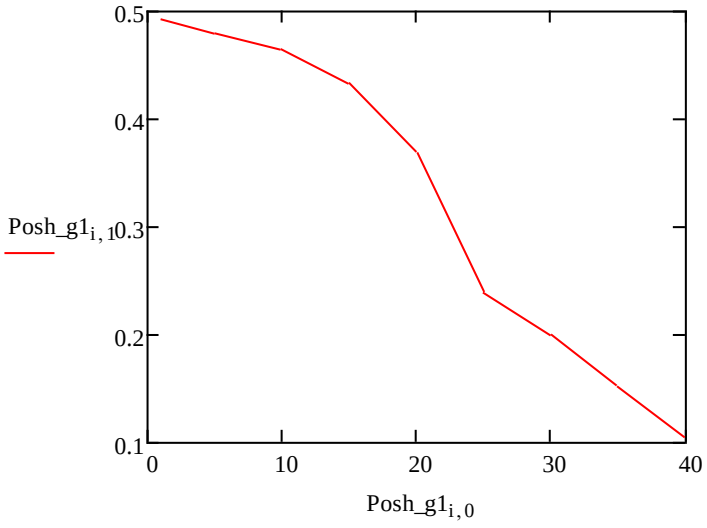
$$w = 15.311$$

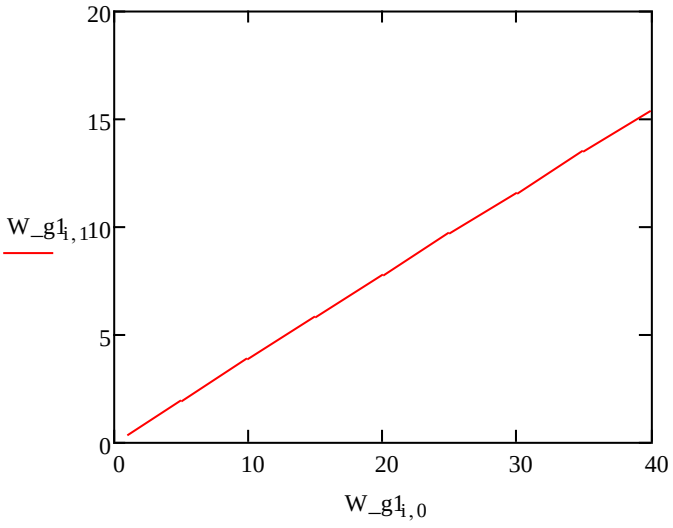
Послідовно змінюючи коефіцієнт посилення g і виконуючи процедури вбудовування та вилучення повідомлення, отримаємо відповідні емпіричні оцінки, які занесемо в таблиці:

$$\text{Posh\_g1} := \begin{pmatrix} 1 & 0.493 \\ 5 & 0.479 \\ 10 & 0.464 \\ 15 & 0.434 \\ 20 & 0.368 \\ 25 & 0.238 \\ 30 & 0.2 \\ 35 & 0.151 \\ 40 & 0.104 \end{pmatrix} \qquad \text{W\_g1} := \begin{pmatrix} 1 & 0.39 \\ 5 & 1.951 \\ 10 & 3.897 \\ 15 & 5.838 \\ 20 & 7.771 \\ 25 & 9.692 \\ 30 & 11.596 \\ 35 & 13.473 \\ 40 & 15.331 \end{pmatrix}$$

4.8. Побудуємо графіки отриманих емпіричних оцінок:

$i := 0..9$





Отримані емпіричні залежності показують, що підвищення коефіцієнта посилення призводить до різкого зниження ймовірності помилкового вилучення інформаційних бітів повідомлення. Однак це також веде до збільшення внесених спотворень в контейнер-зображення. Однак у порівнянні з використанням ортогональних дискретних сигналів (див. рисунки п. 3.3.) застосування квазіортогональних сигналів призводить до меншого спотворення контейнера. Так, наприклад, при встановленні k=4 біт повідомлення в один рядок контейнера з використанням ортогональних дискретних сигналів при коефіцієнті посилення g = 4 величина внесених спотворень становить понад 2,33%. При більшій кількості внесених бітів даних (10 бітів в один рядок контейнера), а отже і при більшій пропускній здатності стеганографічного каналу передачі даних застосування квазіортогональних дискретних сигналів навіть з великим значенням коефіцієнта посилення (g = 5) призводить до менших спотворень контейнера, величина внесених спотворень не перевищує 2%.

Таким чином, застосування квазіортогональних дискретних сигналів дозволяє істотно підвищити пропускну здатність стеганоканалів при меншій величині внесених спотворень. У той же час, використання квазіортогональних дискретних сигналів істотно підвищує ймовірність помилкового вилучення бітів повідомлення (за рахунок сильної кореляції з окремими фрагментами контейнера-зображення). Позбутися цього негативного фактора можливо за рахунок адаптивного формування квазіортогональних дискретних сигналів з урахуванням особливостей використовуваного контейнера-зображення.

Завдання 5. (Додаткове завдання). Реалізація у середовищі символьної математики MathCAD адаптивного алгоритму формування квазіортогональних дискретних сигналів. Реалізація алгоритмів приховування та вилучення даних із адаптовано формованими квазіортогональними дискретними сигналами, отримання емпіричних залежностей ймовірності правильного вилучення даних та частки внесених при цьому похибок у контейнер-зображення

5.1. Реалізуємо адаптивний алгоритм формування квазіортогональних дискретних сигналів з урахуванням особливостей використовуваного контейнера-зображення. Для цього розіб'ємо масив яскравостей червоного кольору на рядки наступним чином:

```
R_Arr := | for i ∈ 0..rows(R) − 1
         |     for j ∈ 0..255
         |       a_j ← R_(i,j)
         |     R_Arr_i ← a
         | R_Arr
```

Сформований масив «R_Arr» в якості елементів містить рядки масиву яскравостей червоного кольору контейнера-зображення.

Алгоритм адаптивного формування квазіортогональних дискретних сигналів представимо наступною процедурою:

```
ArrayFunction2 := | i ← 0
                  | while i < 1024
                  |     for j ∈ 0..255
                  |         b ← ceil(rnd(2)) − 1
                  |         a_j ← 1 if b = 1
                  |         a_j ← −1 if b = 0
                  |     ArrayFunction2_i ← a
                  |     b ← 0
                  |     jj ← 0
                  |     while jj < rows(R_Arr) ∧ b = 0
                  |         a ← MultString(R_Arr_jj, ArrayFunction2_i)
                  |         b ← b + 1 if |a| > 1000
                  |         jj ← jj + 1
                  |     i ← i + 1 if b = 0
                  | ArrayFunction2
```

Суть алгоритму полягає в формуванні псевдовипадкових послідовностей і обчисленні коефіцієнта кореляції з усіма елементами масиву «R_Arr», тобто з усіма рядками контейнера. Якщо коефіцієнт кореляції для всіх рядків контейнера не перевищує заздалегідь заданої величини (в даному випадку значення 1024) сформована послідовність використовується в якості квазіортогонального дискретного сигналу. Якщо коефіцієнт кореляції для будь-якого рядка контейнера перевищить задане значення, сформована послідовність бракується і формується інша послідовність. Зрозуміло, що час формування ансамблю дискретних сигналів залежить від граничної величини, з якою порівнюється значення коефіцієнта кореляції. При її зменшенні різко зростають тимчасові витрати на формування ансамблю сигналу, проте мале граничне значення забезпечить слабку кореляцію формованих квазіортогональних дискретних сигналів з окремими фрагментами контейнера-зображення. Для прикладу наведемо один з дискретних сигналів і значення коефіцієнта кореляції з одним з рядків контейнера:

$ArrayFunction2_{777} =$

	0
0	1
1	1
2	1
3	1
4	1
5	1
6	-1
7	-1
8	-1
9	1
10	1
11	-1
12	-1
13	-1
14	1
15	…

$$MultString(R\_Arr_2, ArrayFunction2_{777}) = -562$$

5.2. Для вбудовування повідомлення скористаємося такою процедурою:

$g := 9$

```
Sum2 := | for i ∈ 0..rows(R) − 1
        |   Sum2_i ← g·ArrayFunction2(M_d_i)
        | Sum2
```

Сформований масив «Sum2» в якості елементів містить модульоване квазіортогональними сигналами повідомлення. Для його вбудовування виконаємо накладення масиву «Sum2» на контейнер-зображення:

```
S2 := | for i ∈ 0..rows(R) − 1
      |   for j ∈ 0..cols(R) − 1
      |     S2_{i,j} ← R_{i,j} + (Sum2_i)_j
      |     S2_{i,j} ← 255 if S2_{i,j} > 255
      |     S2_{i,j} ← 0 if S2_{i,j} < 0
      | S2
```

Отримаємо заповнений контейнер, порівняємо його з початковим:

S2 =

	0	1	2	3
0	77	70	81	81
1	119	106	99	77
2	141	129	121	114
3	113	125	96	95
4	122	131	126	109
5	156	138	157	139
6	178	173	158	179
7	198	204	184	…

R =

	0	1	2	3
0	86	79	72	72
1	110	97	90	86
2	132	120	112	105
3	122	116	105	104
4	131	122	117	118
5	147	147	148	148
6	169	164	167	170
7	189	195	193	…

S2

R

Запишемо сформований контейнер в файл і подивимося результат:

```
WRITERGB("Stego_Kvasi_ad.bmp") := augment(S2, G, B)
```

"Stego_Kvasi_ad.bmp"

"1.bmp"

Як видно з наведених рисунків сформований контейнер практично не відрізняється від початкового. Проте в нього вбудовано більше 1600 бітів інформаційного повідомлення.

5.3. Для вилучення інформаційного повідомлення розрахуємо дані контейнера:

C3 := READRGB("Stego_Kvasi_ad.bmp")

R3 := READ_RED("Stego_Kvasi_ad.bmp")

G3 := READ_GREEN("Stego_Kvasi_ad.bmp")

B3 := READ_BLUE("Stego_Kvasi_ad.bmp")

R3 =

	0	1	2	3	4
0	77	70	81	81	63
1	119	106	99	77	69
2	141	129	121	114	87
3	113	125	96	95	112
4	122	131	126	109	109
5	156	138	157	139	159
6	178	173	158	179	164
7	198	204	184	180	…

R3

Сформуємо масив рядків заповненого контейнера:

```
ArrayString2 := | for i ∈ 0..rows(R3) − 1
                |     for j ∈ 0..cols(R3) − 1
                |       a_j ← R3_(i,j)
                |     ArrayString2_i ← a
                | ArrayString2
```

і винесемо вбудоване повідомлення у вигляді десяткового масиву даних:

```
M_d2 := | for i ∈ 0..rows(R3) − 1
        |   a ← 0
        |   for j ∈ 0..1023
        |     if MultString(ArrayString2_i, ArrayFunction2_j) > a
        |       a ← MultString(ArrayString2_i, ArrayFunction2_j)
        |       M_d2_i ← j
        | M_d2
```

Отриманий результат можна порівняти з масивом вбудованих даних:

M_d =

	0
0	456
1	187
2	607
3	963
4	485
5	60
6	674
7	131
8	753
9	380
10	574
11	899
12	749
13	251
14	782
15	…

M_d2 =

	0
0	456
1	187
2	607
3	963
4	485
5	60
6	674
7	131
8	753
9	380
10	574
11	899
12	749
13	251
14	782
15	…

Зрозуміло, що використання адаптивно формованих дискретних сигналів дозволило істотно підвищити ймовірність правильного вилучення повідомлень.

5.4. Перетворимо отриманий масив даних в двійковий вигляд і порівняємо отриманий результат з тими двійковими даними, які були вбудовані в контейнер:

```
M_b3 := | for i ∈ 0..rows(M_d2) − 1
        |   x ← M_d2_i
        |   for j ∈ 0..9
        |     M_b3_(i·10+j) ← mod(x, 2)
        |     x ← floor(x / 2)
        | M_b3
```

M_b3 =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	…

M_b =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	…

5.5. Проведемо оцінку ймовірності правильного вилучення повідомлення і величини внесених спотворень від коефіцієнта посилення g. Для цього розрахуємо частоту помилково витягнутих інформаційних бітів і оцінку внесених спотворень в контейнер-зображення:

Posh :=
a ← 0
for i ∈ 0..rows(M_b3) − 1
  a ← a + 1 if M_b3_i ≠ M_b_j
Posh ← a / rows(M_b3)
Posh

w :=
w ← 0
for i ∈ 0..rows(R3) − 1
  for j ∈ 0..cols(R3) − 1
    w ← w + |R3_{i,j} − R_{i,j}|
w ← w·100 / (rows(R3)·cols(R3)·256)
w

Posh = 0

w = 3.508

Послідовно змінюючи коефіцієнт посилення g і виконуючи процедури вбудовування та вилучення повідомлення, отримаємо відповідні емпіричні оцінки, які занесемо в таблиці:

Posh_g3 :=

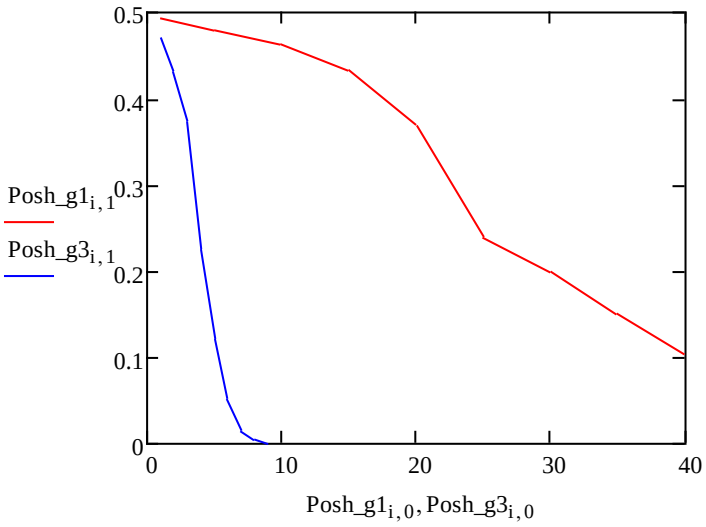
1	0.47
2	0.43
3	0.373
4	0.224
5	0.121
6	0.051
7	0.015
8	0.005
9	0

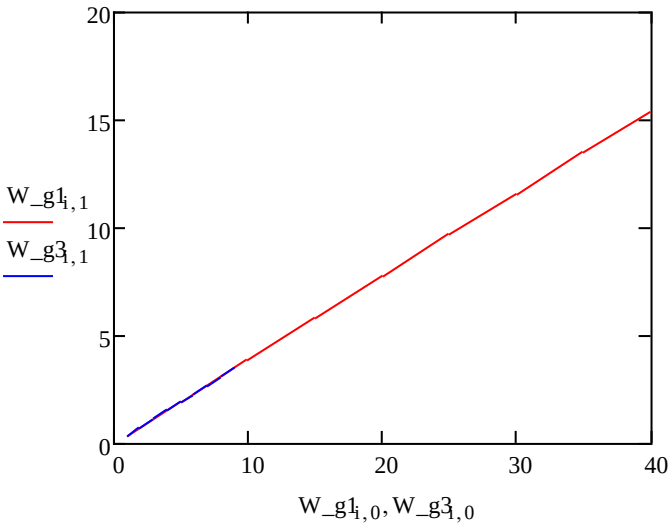
W_g3 :=

1	0.39
2	0.781
3	1.171
4	1.561
5	1.951
6	2.34
7	2.73
8	3.119
9	3.508

5.6. Побудуємо графіки отриманих емпіричних оцінок і порівняємо їх з отриманими раніше залежностями для випадку використання квазіортогональних дискретних сигналів без адаптації до використовуваного контейнеру:

$$i := 0..9$$





Синім кольором на графіках відображені результати моделювання стеганосистем з адаптивним формуванням сигналів, червоним - без адаптації. Зрозуміло, що без збільшення величини внесених спотворень вдалося істотно знизити ймовірність помилкового вилучення інформаційних бітів повідомлень. У порівнянні з використанням ортогональних дискретних сигналів вдалося істотно збільшити пропускну здатність стеганоканала при порівняних внесених викривленнях. Як приклад на рисунках наведемо наступні контейнери:

S

S2

R

Перший контейнер заповнений з використанням ортогональних дискретних сигналів з параметрами «k = 4» і «g = 4», тобто в кожен рядок контейнера вбудовано чотири біти, всього в контейнер вбудовано 676 бітів інформації. Ці параметри забезпечують практично безпомилкове отримання повідомлення, проте максимальна величина внесених спотворень в окремі пікселі зображення становить k * g = 16 рівнів яскравості.

Другий контейнер заповнений з використанням адаптивно сформованих з урахуванням властивостей контейнера-зображення квазіортогональних дискретних сигналів. При цьому використаний коефіцієнт посилення «g = 9», який також забезпечує практично безпомилкове отримання інформаційного повідомлення, проте максимальна величина внесених спотворень в окремі пікселі зображення становить g = 9 рівнів яскравості. І хоча усереднене значення внесених спотворень для ортогональних сигналів трохи нижче, їх абсолютне значення істотно (майже в два рази) може перевершувати аналогічний показник для адаптивно сформованих квазіортогональних сигналів. Вплив зниження максимального рівня внесених викривлень на окремі пікселі зображення візуально помітне на наведених рисунках. Третій контейнер являє собою немодифікований (порожній) контейнер.

Таким чином, як видно з отриманих результатів застосування адаптивно сформованих квазіортогональних дискретних сигналів дозволяє істотно підвищити пропускну здатність стеганоканала при порівняних викривленнях, що вносяться до контейнера-зображення. Внесені спотворення можна ще більше знизити, зменшивши чисельний поріг, що обмежує коефіцієнт взаємної кореляції в алгоритмі адаптивного формування дискретних сигналів.

6. Приклад оформлення звіту з лабораторної роботи

Лабораторна робота № 3
Вбудовування даних в просторову область нерухомих зображень на основі прямого розширення спектру.

1.

"Picture.bmp"

C := READRGB("Picture.bmp")

R := READ_RED("Picture.bmp")

G := READ_GREEN("Picture.bmp")

B := READ_BLUE("Picture.bmp")

M := READBIN("Text.txt","byte")

Функція перетворення повідомлення з двійкового виду в десятковий

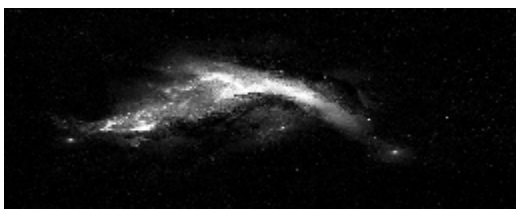
$$B2D(x) := \sum_{i=0}^{7} \left(x_i \cdot 2^i \right)$$

Функція перетворення повідомлення з десяткового виду в двійковий

$$D\_B(x) := \left| \begin{array}{l} \text{for } i \in 0..7 \\ \quad \left| \begin{array}{l} V_i \leftarrow \text{mod}(x, 2) \\ x \leftarrow \text{floor}\left(\frac{x}{2}\right) \end{array} \right. \\ V \end{array} \right.$$

Функція перетворення з десяткового масива M в двійковий:

$$M\_b := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(M) - 1 \\ \quad \left| \begin{array}{l} V \leftarrow D\_B\left(M_i\right) \\ \text{for } j \in 0..7 \\ \quad M\_b_{i \cdot 8 + j} \leftarrow V_j \end{array} \right. \\ M\_b \end{array} \right.$$



R

M =

	0
0	68
1	69
2	70
3	32
4	76
5	69
6	80
7	80
8	65
9	82
10	68
11	32
12	76
13	89
14	82
15	…

M_b =

	0
0	0
1	0
2	1
3	0
4	0
5	0
6	1
7	0
8	1
9	0
10	1
11	0
12	0
13	0
14	1
15	…

Формування матриць Адамара

$H_0 := (1)$

```
H := for i ∈ 1..8
       F ← H_(i−1)
       for j ∈ 0..rows(F) − 1
         for jj ∈ 0..cols(F) − 1
           a ← F_(jj,j)
           F1_(jj,j) ← a
       for j ∈ 0..rows(F) − 1
         for jj ∈ 0..cols(F) − 1
           a ← F_(jj,j)
           F1_(jj+cols(F),j) ← a
       for j ∈ 0..rows(F) − 1
         for jj ∈ 0..cols(F) − 1
           a ← F_(jj,j)
           F1_(jj,j+rows(F)) ← a
       for j ∈ 0..rows(F) − 1
         for jj ∈ 0..cols(F) − 1
           a ← F_(jj,j)
           F1_(jj+cols(F),j+rows(F)) ← −a
       H_i ← F1
     H
```

$H_8 =$

	0	1	2	3	4	5	6	7	8	9
0	1	1	1	1	1	1	1	1	1	1
1	1	-1	1	-1	1	-1	1	-1	1	-1
2	1	1	-1	-1	1	1	-1	-1	1	1
3	1	-1	-1	1	1	-1	-1	1	1	-1
4	1	1	1	1	-1	-1	-1	-1	1	1
5	1	-1	1	-1	-1	1	-1	1	1	-1
6	1	1	-1	-1	-1	-1	1	1	1	1
7	1	-1	-1	1	-1	1	1	-1	1	-1
8	1	1	1	1	1	1	1	1	-1	-1
9	1	-1	1	-1	1	-1	1	-1	-1	1
10	1	1	-1	-1	1	1	-1	-1	-1	-1
11	1	-1	-1	1	1	-1	-1	1	-1	1
12	1	1	1	1	-1	-1	-1	-1	-1	-1
13	1	-1	1	-1	-1	1	-1	1	-1	1
14	1	1	-1	-1	-1	-1	1	1	-1	-1
15	1	-1	-1	1	-1	1	1	-1	-1	…

Масив ортогональних функцій

```
ArrayFunction := for i ∈ 0..255
                   for j ∈ 0..255
                     a_j ← (H_8)_(i,j)
                   ArrayFunction_i ← a
                 ArrayFunction
```

$ArrayFunction_5 =$

	0
0	1
1	-1
2	1
3	-1
4	-1
5	1
6	-1
7	1
8	1
9	-1
10	1
11	…

Перетворимо бітове повідомлення:

```
m := | for i ∈ 0..rows(M_b) − 1
     |   m_i ← 1  if M_b_i = 1
     |   m_i ← −1 if M_b_i = 0
     | m
```

```
m2b(m) := | for i ∈ 0..rows(m) − 1
          |   m1_i ← 1 if m_i = 1
          |   m1_i ← 0 if m_i = −1
          | m1
```

Модулюємо кожен інформаційний біт за допомогою ПВП довжиною 256 бітів:

$k := 4 \qquad g := 4$

$$Sum := \left|\begin{array}{l} \text{for } i \in 0..rows(R) - 1 \\ \quad a \leftarrow \sum_{j=0}^{k-1} \left[g \cdot \left(m_{k \cdot i + j} \cdot ArrayFunction_{j+1} \right) \right] \\ \quad Sum_i \leftarrow a \\ Sum \end{array}\right.$$

Накладення модульованого повідомлення на контейнер:

$$S := \left|\begin{array}{l} \text{for } i \in 0..rows(R) - 1 \\ \quad \text{for } j \in 0..cols(R) - 1 \\ \quad\quad S_{i,j} \leftarrow R_{i,j} + \left(Sum_i\right)_j \\ \quad\quad S_{i,j} \leftarrow 255 \text{ if } S_{i,j} > 255 \\ \quad\quad S_{i,j} \leftarrow 0 \text{ if } S_{i,j} < 0 \\ S \end{array}\right.$$

Порожній та заповнений контейнери:

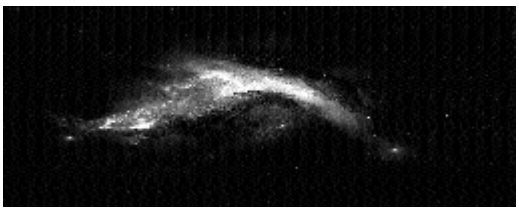
R =

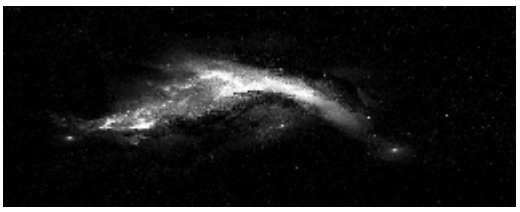
	0	1	2	3	4	5
0	10	5	6	4	1	8
1	8	0	7	23	1	19
2	3	7	4	2	0	7
3	0	11	3	6	3	4
4	6	0	5	2	8	0
5	4	0	2	2	1	6
6	9	0	4	2	5	1
7	8	4	0	1	1	4
8	3	3	0	1	1	6
9	0	3	1	3	5	2
10	3	12	2	0	6	…

S =

	0	1	2	3	4	5
0	2	0	0	12	1	8
1	0	0	0	31	1	19
2	3	0	4	2	8	0
3	0	3	0	14	3	4
4	6	0	0	2	16	8
5	0	0	0	10	1	6
6	0	0	4	2	0	9
7	0	12	0	0	1	20
8	3	3	0	17	0	0
9	0	0	0	11	5	2
10	3	0	2	0	14	…

WRITERGB("Stego1.bmp") := augment(S, G, B)





S

R

"Stego1.bmp"

"Picture.bmp"

Функція розрахунку коефіцієнта кореляції:

$$\text{MultString}(A,B) := \left| \begin{array}{l} X \leftarrow 0 \\ \text{for } i \in 0..255 \\ \quad X \leftarrow X + A_i \cdot B_i \\ X \end{array} \right.$$

За результатами обчислень видно, що повідомлення корельовано з ПВП (1,2,3,4), а самі ПВП між собою некорельовані.

$$\text{MultString}\left(\text{ArrayFunction}_2, \text{ArrayFunction}_3\right) = 0$$

$$\text{MultString}\left(\text{Sum}_0, \text{ArrayFunction}_1\right) = -1.024 \times 10^3$$

$$\text{MultString}\left(\text{Sum}_0, \text{ArrayFunction}_2\right) = -1.024 \times 10^3$$

$$\text{MultString}\left(\text{Sum}_0, \text{ArrayFunction}_3\right) = 1.024 \times 10^3$$

$$\text{MultString}\left(\text{Sum}_0, \text{ArrayFunction}_4\right) = -1.024 \times 10^3$$

$$\text{MultString}\left(\text{Sum}_0, \text{ArrayFunction}_5\right) = 0$$

Вилучення

$$C1 := \text{READRGB}(\text{"Stego1.bmp"})$$

$$G1 := \text{READ\_GREEN}(\text{"Stego1.bmp"})$$

$$R1 := \text{READ\_RED}(\text{"Stego1.bmp"})$$

$$B1 := \text{READ\_BLUE}(\text{"Stego1.bmp"})$$

Масив рядків контейнера

$$\text{ArrayString} := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(R1) - 1 \\ \quad \left| \begin{array}{l} \text{for } j \in 0..\text{cols}(R1) - 1 \\ \quad a_j \leftarrow R1_{i,j} \\ \text{ArrayString}_i \leftarrow a \end{array} \right. \\ \text{ArrayString} \end{array} \right.$$

Порівняння отриманого та вбудованого повідомлень

$$m1 := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(R1) - 1 \\ \quad \text{for } j \in 0..k - 1 \\ \quad \left| \begin{array}{l} m1_{k \cdot i + j} \leftarrow 1 \text{ if } \text{MultString}\left(\text{ArrayString}_i, \text{ArrayFunction}_{j+1}\right) > 0 \\ m1_{k \cdot i + j} \leftarrow -1 \text{ if } \text{MultString}\left(\text{ArrayString}_i, \text{ArrayFunction}_{j+1}\right) \le 0 \end{array} \right. \\ m1 \end{array} \right.$$

Перетворюємо повідомлення в бінарний вид:

$m2 := m2b(m1)$ $M1 :=$ for $i \in 0..rows(m2) - 8$; $l \leftarrow floor\left(\frac{i}{8}\right)$; for $j \in 0..7$; $V_j \leftarrow m2_{l \cdot 8 + j}$; $M_l \leftarrow B2D(V)$; M

$WRITEBIN("STEGOTXT1.txt","byte",1) := M1$

Розрахунок ймовірності помилкового вилучення інформаційних бітів:

$Posh :=$ $a \leftarrow 0$; for $i \in 0..rows(m2) - 1$; $a \leftarrow a + 1$ if $m2_i \neq M\_b_i$; $Posh \leftarrow \frac{a}{rows(m2)}$; $Posh$

$Posh = 0$

$$W\_k := \begin{pmatrix} 0 & 0 \\ 1 & 0.39 \\ 2 & 0.39 \\ 4 & 0.586 \\ 8 & 0.871 \\ 16 & 1.244 \\ 32 & 1.723 \\ 64 & 2.385 \\ 128 & 3.286 \\ 256 & 4.5 \end{pmatrix} \qquad Posh\_k := \begin{pmatrix} 0 & 0 \\ 1 & 0.006 \\ 2 & 0.053 \\ 4 & 0.093 \\ 8 & 0.121 \\ 16 & 0.126 \\ 32 & 0.145 \\ 64 & 0.148 \\ 128 & 0.148 \\ 255 & 0.15 \end{pmatrix}$$

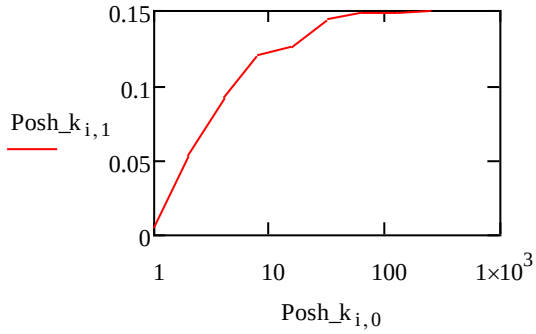
$$Posh\_g := \begin{pmatrix} 1 & 0.093 \\ 2 & 0.018 \\ 3 & 0.003 \\ 4 & 0 \end{pmatrix} \qquad W\_g := \begin{pmatrix} 1 & 0.586 \\ 2 & 1.17 \\ 3 & 1.754 \\ 4 & 2.338 \end{pmatrix}$$

Розрахунок частки внесених викривлень в контейнер-повідомлення:

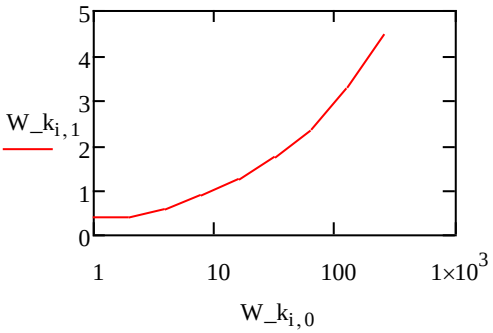
$w :=$ $w \leftarrow 0$; for $i \in 0..rows(R1) - 1$; for $j \in 0..cols(R1) - 1$; $w \leftarrow w + \left|R1_{i,j} - R_{i,j}\right|$; $w \leftarrow \frac{w \cdot 100}{rows(R1) \cdot cols(R1) \cdot 256}$; w

$w = 1.699$ $i := 0..9$

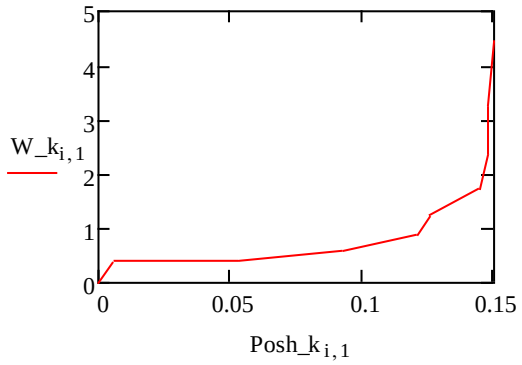
Ймовірність помилки від кількості вбудованих бітів:



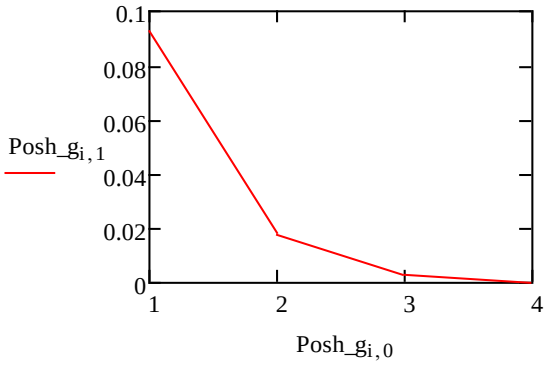
Коефіцієнт внесених спотворень від кількості вбудованих бітів:



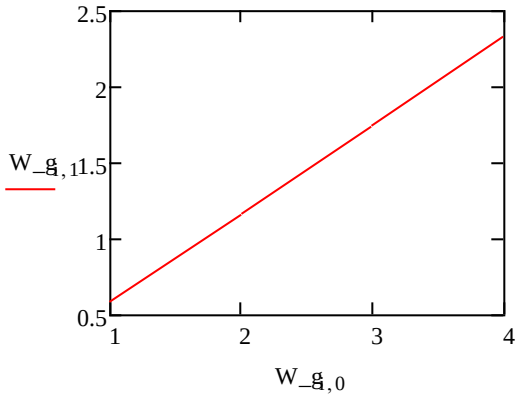
Залежність внесених спотворень та ймовірність правильного вилучення:



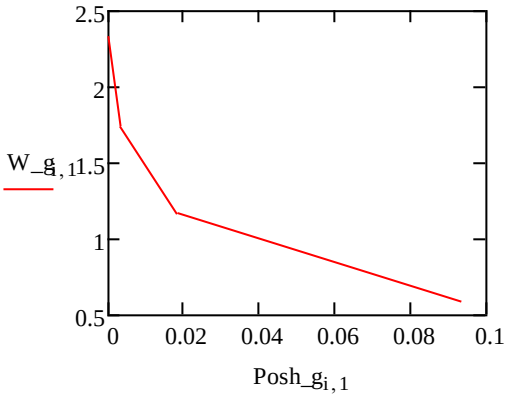
Залежність правильного вилучення від коефіцієнту посилення:



Залежність внесених спотворень від коефіцієнту посилення:



Залежність внесених спотворень та ймовірності правильно вилучення:



Багатоосновне стеганографічне кодування

Формуємо квазіортогональні дискретні сигнали

```
ArrayFunction1 := | for i ∈ 0..1023
                  |   for j ∈ 0..255
                  |     b ← ceil(rnd(2)) − 1
                  |     a_j ← 1  if b = 1
                  |     a_j ← −1  if b = 0
                  |   ArrayFunction1_i ← a
                  | ArrayFunction1
```

$$MultString\left(ArrayFunction1_1, ArrayFunction1_2\right) = 8$$

$$MultString\left(ArrayFunction1_2, ArrayFunction1_3\right) = 10$$

$$MultString\left(ArrayFunction1_1, ArrayFunction1_3\right) = 18$$

Розбиваємо повідомлення на блоки по 10 бітів та формуємо десятковий масив даних

```
M_d := | for i ∈ 0..rows(R) − 1
       |   a ← 0
       |   for j ∈ 0..9
       |     a ← a + M_b_{10·i+j}·2^j
       |   M_d_i ← a
       | M_d
```

Замінюємо блок з 10 бітів на відповідну йому ПВП

```
g := 4
Sum1 := | for i ∈ 0..rows(R) − 1
        |   Sum1_i ← g·ArrayFunction1_(M_d_i)
        | Sum1
```

Вбудовуємо отримане модульоване повідомлення в контейнер:

```
S1 := | for i ∈ 0..rows(R) − 1
      |   for j ∈ 0..cols(R) − 1
      |     S1_{i,j} ← R_{i,j} + (Sum1_i)_j
      |     S1_{i,j} ← 255  if S1_{i,j} > 255
      |     S1_{i,j} ← 0  if S1_{i,j} < 0
      | S1
```

Перетворюємо вилучене повідомлення в бітовий вигляд

```
M_b2 := | for i ∈ 0..rows(M_d1) − 1
        |   x ← M_d1_i
        |   for j ∈ 0..9
        |     M_b2_(i·10+j) ← mod(x, 2)
        |     x ← floor(x/2)
        | M_b2
```

```
M2 := | for i ∈ 0..rows(M_b2) − 8
      |   l ← floor(i/8)
      |   for j ∈ 0..7
      |     V_j ← M_b2_(l·8+j)
      |   M_l ← B2D(V)
      | M
```

WRITEBIN("STEGOTXT2.txt","byte",1) := M2

Розрахунок ймовірності помилкового вилучення інформаційних бітів:

```
Posh := | a ← 0
        | for i ∈ 0..rows(M_b2) − 1
        |   a ← a + 1 if M_b2_i ≠ M_b_j
        | Posh ← a / rows(M_b2)
        | Posh
```

Posh = 0.199

$$Posh\_g1 := \begin{pmatrix} 1 & 0.493 \\ 5 & 0.479 \\ 10 & 0.464 \\ 15 & 0.434 \\ 20 & 0.368 \\ 25 & 0.238 \\ 30 & 0.2 \\ 35 & 0.151 \\ 40 & 0.104 \end{pmatrix}$$

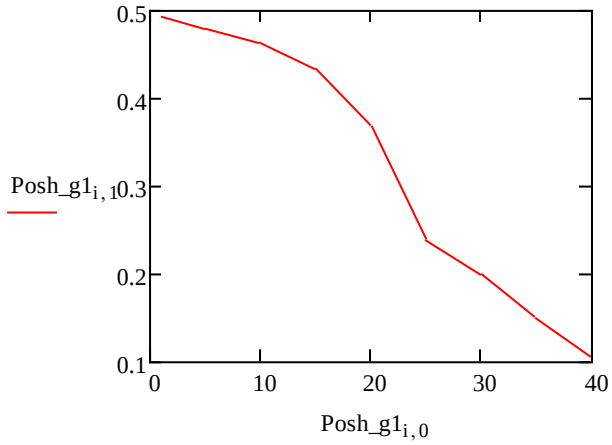
Розрахунок частки внесених спотворень в контейнер-зображення:

```
w := | w ← 0
     | for i ∈ 0..rows(R2) − 1
     |   for j ∈ 0..cols(R2) − 1
     |     w ← w + |R2_(i,j) − R_(i,j)|
     | w ← w·100 / (rows(R2)·cols(R2)·256)
     | w
```

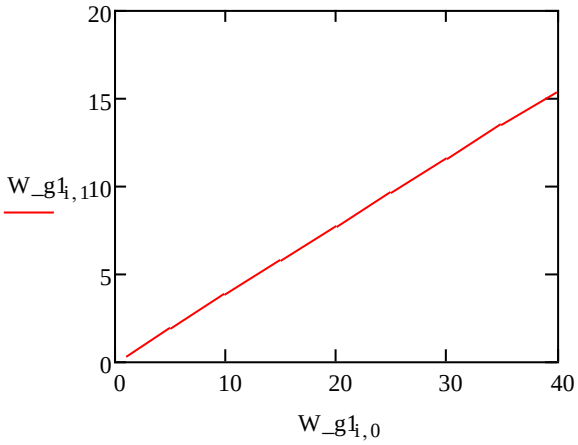
w = 1.286

$$W\_g1 := \begin{pmatrix} 1 & 0.39 \\ 5 & 1.951 \\ 10 & 3.897 \\ 15 & 5.838 \\ 20 & 7.771 \\ 25 & 9.692 \\ 30 & 11.596 \\ 35 & 13.473 \\ 40 & 15.331 \end{pmatrix}$$

Залежність ймовірності помилки від коефіцієнта посилення



Залежність внесених спотворень від коефіцієнта посилення:



Адаптивне формування дискретних сигналів:

```
R_Arr := | for i ∈ 0..rows(R) − 1
         |     for j ∈ 0..255
         |         a_j ← R_(i,j)
         |     R_Arr_i ← a
         | R_Arr
```

Адаптивно формуємо квазіортогональні дискретні сигнали

```
ArrayFunction2 := | i ← 0
                  | while i < 1024
                  |     for j ∈ 0..255
                  |         b ← ceil(rnd(2)) − 1
                  |         a_j ← 1 if b = 1
                  |         a_j ← −1 if b = 0
                  |     ArrayFunction2_i ← a
                  |     b ← 0
                  |     jj ← 0
                  |     while jj < rows(R_Arr) ∧ b = 0
                  |         a ← MultString(R_Arr_jj, ArrayFunction2_i)
                  |         b ← b + 1 if |a| > 1000
                  |         jj ← jj + 1
                  |     i ← i + 1 if b = 0
                  | ArrayFunction2
```

$MultString(R\_Arr_2, ArrayFunction2_2) = 29$

$MultString(ArrayFunction_1, ArrayFunction2_2) = -2$

Вилучаємо блоки по 10 бітів з рядків контейнера:

```
M_d2 := | for i ∈ 0..rows(R3) − 1
        |   a ← 0
        |   for j ∈ 0..1023
        |     if MultString(ArrayString2_i, ArrayFunction2_j) > a
        |       a ← MultString(ArrayString2_i, ArrayFunction2_j)
        |       M_d2_i ← j
        | M_d2
```

Перетворюємо вилучене повідомлення в бітовий вигляд

```
M_b3 := | for i ∈ 0..rows(M_d2) − 1
        |   x ← M_d2_i
        |   for j ∈ 0..9
        |     M_b3_(i·10+j) ← mod(x, 2)
        |     x ← floor(x / 2)
        | M_b3
```

```
M3 := | for i ∈ 0..rows(M_b3) − 8
      |   l ← floor(i / 8)
      |   for j ∈ 0..7
      |     V_j ← M_b3_(l·8+j)
      |   M_l ← B2D(V)
      | M
```

WRITEBIN("STEGOTXT3.txt","byte",1) := M3

Розрахунок ймовірності помилкового вилучення інформаційних бітів:

```
Posh := | a ← 0
        | for i ∈ 0..rows(M_b3) − 1
        |   a ← a + 1 if M_b3_i ≠ M_b_i
        | Posh ← a / rows(M_b3)
        | Posh
```

Posh = 0.062

$$Posh\_g3 := \begin{pmatrix} 1 & 0.47 \\ 2 & 0.43 \\ 3 & 0.373 \\ 4 & 0.224 \\ 5 & 0.121 \\ 6 & 0.051 \\ 7 & 0.015 \\ 8 & 0.005 \\ 9 & 0 \end{pmatrix} \qquad Posh\_g := \begin{pmatrix} 1 & 0.093 \\ 2 & 0.018 \\ 3 & 0.003 \\ 4 & 0 \end{pmatrix}$$

Розрахунок частки внесених спотворень в контейнер-зображення:

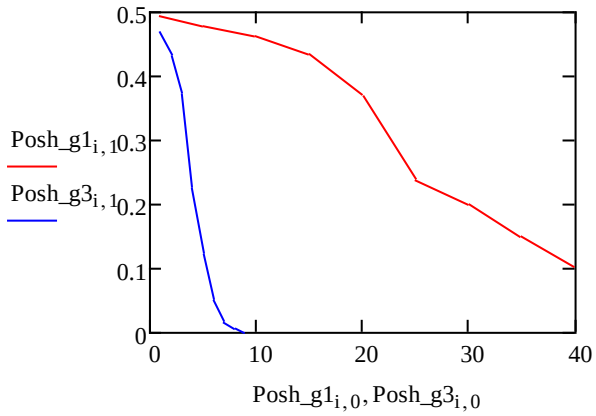
$$w := \left| \begin{array}{l} w \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(R3) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R3) - 1 \\ \quad\quad w \leftarrow w + \left| R3_{i,j} - R_{i,j} \right| \\ w \leftarrow \dfrac{w \cdot 100}{\text{rows}(R3) \cdot \text{cols}(R3) \cdot 256} \\ w \end{array} \right.$$

$w = 1.285$

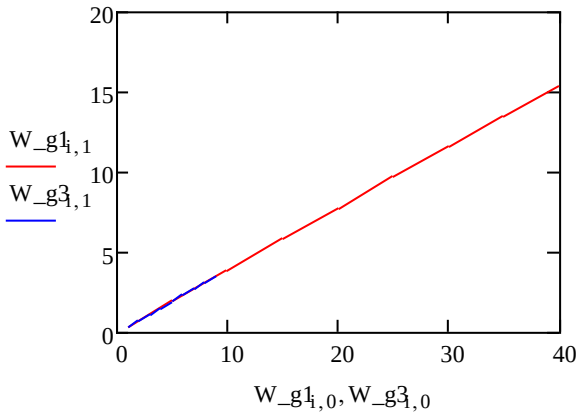
$$W\_g3 := \begin{pmatrix} 1 & 0.39 \\ 2 & 0.781 \\ 3 & 1.171 \\ 4 & 1.561 \\ 5 & 1.951 \\ 6 & 2.34 \\ 7 & 2.73 \\ 8 & 3.119 \\ 9 & 3.508 \end{pmatrix}$$

$$W\_g := \begin{pmatrix} 1 & 0.586 \\ 2 & 1.17 \\ 3 & 1.754 \\ 4 & 2.338 \end{pmatrix}$$

Залежність помилки вилучення від коефіцієнта посилення:



Залежність внесених спотворень від коефіцієнта посилення:



Лабораторна робота №4. «Приховування даних в частотній області нерухомих зображень на основі кодування різниці абсолютних значень коефіцієнтів дискретно-косинусного перетворення»

1. Мета та завдання лабораторної роботи

Мета роботи: закріпити теоретичні знання з теми «Приховування даних в частотній області нерухомих зображень», придбати практичні вміння та навички з розробки стеганографічних систем, дослідити властивості стеганографічних методів, заснованих на низькорівеневих властивостях зорової системи людини (ЗСЛ), зокрема, частотної чутливості.

Лабораторна робота №4 виконується в середовищі символьної математики MathCAD версії 12 або вище.

Завдання лабораторної роботи

1. Реалізувати у середовищі символьної математики MathCAD алгоритми прямого та зворотного дискретно-косинусного перетворення. Дослідити ефект частотної чуттєвості зорової системи людини, а саме як зміна коефіцієнтів дискретно-косинусного перетворення у різних частотних областях впливає на наявність видимих викривлень зображень.
2. Реалізувати у середовищі символьної математики MathCAD алгоритми приховування даних у частотну область нерухомих зображень шляхом кодування різниці абсолютних значень коефіцієнтів дискретно-косинусного перетворення (метод Коха-Жао). Виконати зорове порівняння пустого та заповненого контейнера та зробити відповідні висновки. Реалізувати алгоритми вилучення даних з частотної області зображень методом Коха-Жао.
3. Реалізувати імітацію стеганоатаки на основі стиску зображення алгоритмом JPEG. Дослідити ймовірнісні властивості реалізованих алгоритмів до та після реалізації атаки, а саме, отримати емпіричні залежності ймовірності правильного вилучення даних та частки внесених при цьому похибок у контейнер-зображення. Збільшуючи величину внесених викривлень коефіцієнтів дискретно-косинусного перетворення досягти зменшення помилки вилучення інформаційних даних навіть при імітації атаки стиском.
4. Реалізувати у середовищі символьної математики MathCAD алгоритми приховування даних у частотну область нерухомих зображень шляхом кодування декількох різниць абсолютних значень коефіцієнтів дискретно-косинусного перетворення (удосконалений метод Коха-Жао – метод Бенгама-Мемона-Ео-Юнга). Виконати зорове порівняння пустого та заповненого контейнера та зробити

відповідні висновки. Реалізувати алгоритми вилучення даних з частотної області зображень методом Бенгама-Мемона-Ео-Юнга. Дослідити ймовірнісні властивості реалізованих алгоритмів.

5. (Додаткове завдання). Реалізувати у середовищі символьної математики MathCAD алгоритми приховування та вилучення інформаційних даних у частотну область нерухомих зображень методом Фрідріх.

2. Методичні вказівки з організації самостійної роботи

1. Вивчити теоретичний матеріал лекції «Приховування даних в частотній області нерухомих зображень на основі кодування різниці абсолютних значень коефіцієнтів дискретно-косинусного перетворення».
2. Вивчити матеріал основного джерела літератури (Конахович Г.Ф., Пузыренко А.Ю. Компьютерная стеганография): приховування даних в частотній області зображень (ст. 126-179).
3. Вивчити матеріал додаткових джерел:
   a. About JPEG (http://www.pcs-ip.eu/index.php/main/edu/5);
   b. The Discrete Cosine Transform (http://www.cs.cf.ac.uk/Dave/ Multimedia/node231.html).
4. Вивчити основні команди у середовищі символьної математики MathCAD щодо роботи з зображеннями.
5. Підготувати відповіді на контрольні запитання.
6. Підготувати бланк звіту з лабораторної роботи.

Допуск до виконання лабораторної роботи здійснюється за результатами письмового опитування.

3. Загальнотеоретичні положення за темою лабораторної роботи

3.1 Приховування даних в частотній області зображення

Стеганографічні методи приховування даних в просторовій області зображення є нестійкими до більшості з відомих видів спотворень. Так, наприклад, використання операції компресії з втратами (щодо зображення, це може бути JPEG-компресія) призводить до часткового або, що більш ймовірно, повного знищення вбудованої в контейнер інформації. Більш стійкими до різноманітних спотворень, в тому числі і компресії, є методи, які використовують для приховування даних не просторову область контейнера, а частотну.

Існує кілька способів представлення зображення в частотній області. При цьому використовується та чи інша декомпозиція зображення,

використовуваного в якості контейнера. Наприклад, існують методи на основі використання дискретного косинусного перетворення (ДКП), дискретного перетворення Фур'є (ДПФ), вейвлет-перетворення, перетворення Карунена-Лоєва і деякі інші. Подібні перетворення можуть застосовуватися або до окремих частин зображення, або до зображення вцілому.

Найбільшого поширення серед всіх ортогональних перетворень в стеганографії отримали вейвлет-перетворення і ДКП, що певною мірою пояснюється значним поширенням їх використання при компресії зображень. Крім того, для приховування даних доцільно застосовувати саме те перетворення зображення, якому воно буде піддаватися згодом при можливій компресії. Наприклад, відомо, що алгоритм ДКП є базовим в стандарті JPEG, а вейвлет-перетворення - в стандарті JPEG2000.

Стеганоалгоритм може бути досить стійким до подальшої компресії зображення, тільки якщо він буде враховувати особливості алгоритму перспективного стиснення. При цьому, звичайно, стеганоалгоритм, в основу якого закладено вейвлет-перетворення, зовсім не обов'язково буде стійким до діскретно-косинусного алгоритму стиснення, і навпаки. Великі труднощі виникають при виборі методу стеганоперетворення під час приховування даних в потоковому відео. Причина цього - однією зі складових алгоритмів компресії відеоінформації (на додаток до компресії нерухомого кадру), є кодування векторів компенсації руху. При компресії нерухомих зображень ця компенсація відсутня за непотрібністю. Щоб бути в достатній мірі стійким, стеганоалгоритм повинен враховувати цей фактор.

Залишається також відкритим питання про існування стійкого стеганоперетворення, яке було б незалежним від застосовуваного в подальшому алгоритмі компресії.

На сьогоднішній день відомо досить велика кількість моделей, що дозволяють оцінити пропускну здатність каналу передачі прихованих даних. Розглянемо одну з них.

Нехай C - первинне зображення (контейнер-оригінал), M - повідомлення, яке підлягає приховуванню. Тоді модифіковане зображення (стеганоконтейнер) S=C+M. Також передбачається, що модифіковане зображення S візуально не відрізняється від первинного і може бути піддано в стеганоканалі компресії з втратами: $S^{\nabla} = \Theta(S)$, де $\Theta(\bullet)$ - оператор компресії.

Завдання адресата - вилучити з отриманого контейнера S^{∇} вбудовані на попередньому етапі біти даних M_i.

Постає питання – яку кількість бітів можна ефективно вбудовувати в зображення і з часом вилучити з нього за умови задовільно низької ймовірності помилок на останньому етапі. Іншими словами, яка пропускна здатність каналу передачі прихованих даних за умови наявності в каналі зв'язку певного алгоритму компресії? Блок-схема такого стеганоканала представлена на рис. 4.1.

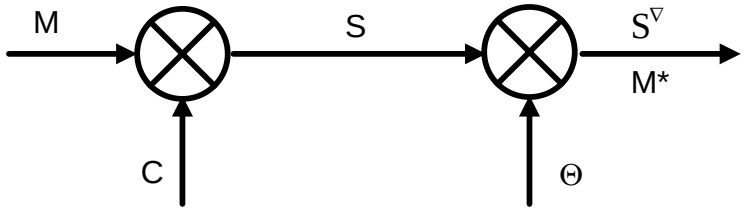


Рисунок 4.1 - Блок-схема стеганоканалу з атакою у вигляді компресії

Повідомлення M передається по каналу, який має два джерела "шуму": C- зображення-контейнер і "шум" Θ, що виникає в результаті операцій компресії / декомпресії. При цьому S^{∇} і M* - можливо спотворені стеганоконтейнер і, як результат, - оцінка корисного повідомлення.

Структурна схема стеганосистеми представлена на рис. 4.2.

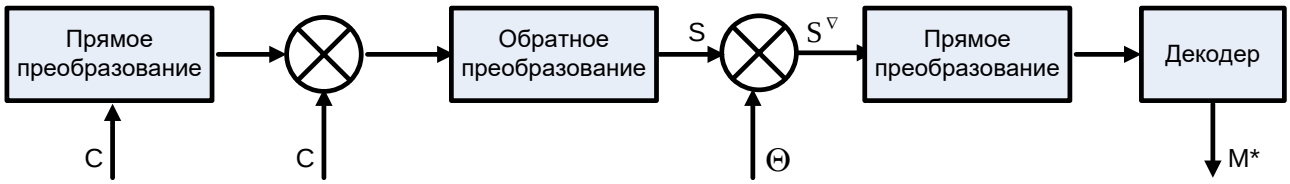


Рисунок 4.2 - Структурна схема стеганосистеми при наявності в стеганоканалі атаки компресії

Зображення C розкладається на D субсмуг (пряме перетворення), в кожну з яких вбудовується приховувана інформація M. Після зворотного перетворення виходить модифіковане зображення S. Після компресії / декомпресії Θ в каналі зв'язку знаходиться зображення S^{∇}, яке на приймаючій стороні знову піддається прямому перетворенню і з кожної субсмуги D незалежно витягується приховане повідомлення - оцінка M*.

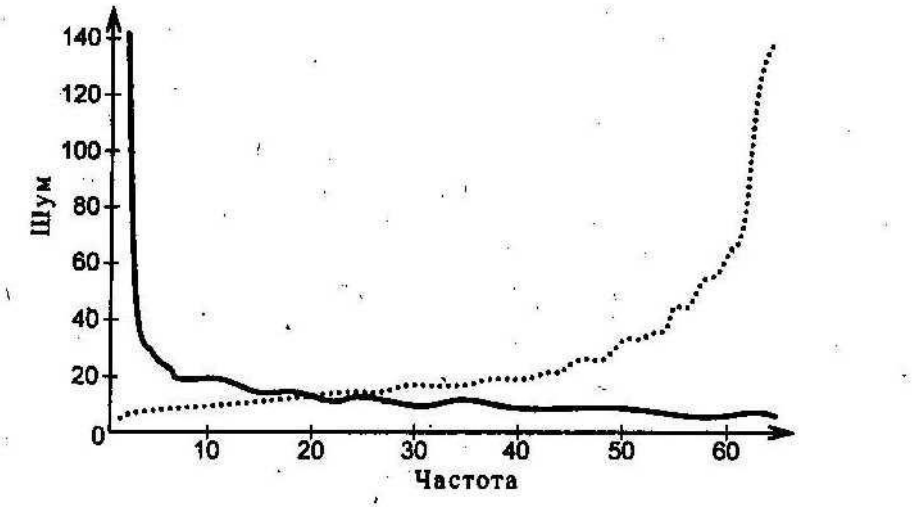


Рисунок 4.3 - Залежність шуму зображення (суцільна лінія) і шум обробки (пунктирна лінія) від частоти

Реальні зображення не є випадковими процесами з рівномірно розподіленими значеннями величин. Відомо, і даний факт використовується в алгоритмах компресії, що велика частина енергії зображень зосереджена в низькочастотнній (НЧ) області спектру. Звідси і виникає необхідність у здійсненні декомпозиції зображення на субсмуги, до яких додається стеганоповідомлення. НЧ субсмуги містять основну частину енергії зображення і, таким чином, носять шумовий характер. Високочастотні (ВЧ) субсмуги спектру зображення найбільшим чином піддаються впливу з боку різноманітних алгоритмів обробки, таких як, наприклад, компресія або НЧ-фільтрація. Таким чином, можна зробити висновок, що для вбудовування повідомлення найоптимальнішими є середньочастотні (СЧ) субсмуги спектру зображення. Типовий розподіл шуму зображення та шуму-обробки за спектром частоти зображено на рис. 4.3.

Стеганографічний канал можна розкласти на ряд незалежних підканалів. Таке розкладання відбувається за рахунок виконання прямого і зворотного перетворень. У кожному з D підканалів існує по два джерела шуму. Нехай, $\sigma^2_{\tilde{N},\Theta_j}$ при j = 1, ..., D - дисперсія коефіцієнтів перетворення (шум зображення) в кожному з підканалів. Тоді вираз для пропускної здатності каналу стеганосистеми набуде вигляду:

$$B = \frac{X \cdot Y}{2 \cdot D} \cdot \sum_{j=1}^{D} \log_2 \left(1 + \frac{\nu_j^2}{\sigma_{C_j}^2 + \sigma_{\Theta_j}^2} \right) \textit{бітів}$$

де ν_j - візуальний поріг для j-ої субсмуги,

ν_j^2 - максимально допустима енергія стеганоповідомлення, виходячи з вимог збереження візуальної якості зображення);

X і Y - піксельний розмір зображення-контейнера.

Вибір значення візуального порогу базується на урахуванні властивостей зорової системи людини. Відомо, що шум у ВЧ областях зображення більш прийнятний, ніж в НЧ областях.

Можна ввести деякі вагові коефіцієнти: $\nu_j^2 = \kappa \cdot \sigma^{2\cdot\alpha}_{\tilde{N},\Theta_j}$, де $0 \le \alpha \le 1$, а $\kappa << \sigma_{\tilde{N},\Theta_j} \forall$ j - константа.

Випадок, коли $\alpha = 0$, відповідає рівномірному розподілу стеганограми за всіма субсмугами. Випадок $\alpha = 1$ відповідає розподілу стеганограми відповідно до дисперсії субсмуг.

Після деяких спрощень можна отримати вираз для пропускної здатності каналу передачі прихованих даних:

$$B = \frac{X \cdot Y}{2 \cdot D} \cdot \sum_{j=1}^{D} \log_2 \left(1 + \frac{\kappa \cdot \sigma^{2\cdot\alpha}_{\tilde{N},\Theta_j}}{\sigma^2_{C_j}} \right) \approx \frac{X \cdot Y}{2 \cdot D} \log_2 \left(1 + \sum_{j=1}^{D} \frac{\kappa_1}{\sigma^{2\cdot(1-\alpha)}_{C_j}} \right). \qquad (3.1)$$

Знак наближення в виразі (3.1) є справедливим, $\kappa_1 \cdot \sigma^{2\cdot\alpha}_{\tilde{N}_j} / \sigma^2_{C_j} << 1$ для будь-якого значення. Зрозуміло, що при $\alpha = 1$ декомпозиція жодним чином не впливатиме на пропускну здатність стеганоканалу. При $\alpha < 1$ пропускна

здатність буде зростати за рахунок того, що в області з низькою дисперсією (високочастотній області) до стеганосигналу додається відносно більше енергії.

Відомо, що перетворення можна впорядкувати по досяжним виграшами від алгоритму кодування (рис. 4.4).

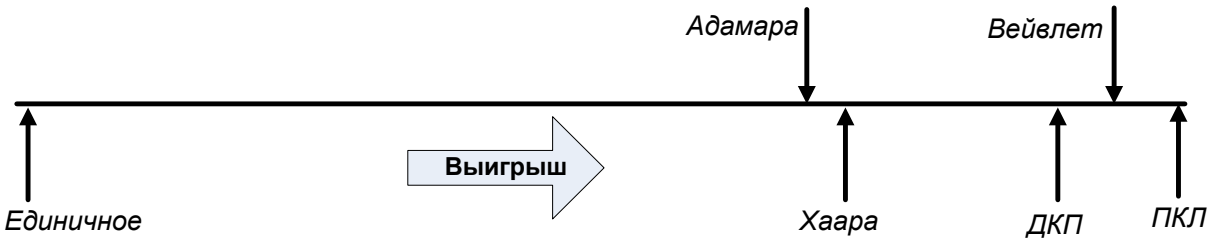


Рисунок 4.4 - Види перетворень, впорядковані за досяжними виграшами від використання

Під виграшем від кодування мається на увазі ступінь перерозподілу дисперсій коефіцієнтів перетворення. Найбільший виграш дає перетворення Карунена-Лоєва (ПКЛ), найменший - розкладання за базисом одиничного імпульсу (тобто відсутність перетворення).

Перетворення, які характеризуються високими значеннями виграшу від кодування, такі як ДКП, вейвлет-перетворення, характеризуються різко нерівномірним розподілом дисперсій коефіцієнтів субсмуг. Високочастотні субсмуги не підходять для вбудовування через значний шум обробки, а низькочастотні - через значний шум зображення (див. Рис. 4.3). Тому доводиться обмежуватися середньочастотними смугами, у яких шум зображення приблизно дорівнює шуму обробки. Оскільки таких смуг мала кількість, то пропускна здатність стеганоканалу є порівняно малою.

У разі застосування перетворення з більш низьким виграшем від кодування, наприклад, перетворення Адамара або Фур'є, існує більше блоків, у яких шум зображення приблизно дорівнює шуму обробки, а, отже, і пропускна здатність вище. Тобто для підвищення пропускної здатності стеганографічного каналу доцільно застосовувати перетворення з меншими виграшами від кодування, які погано підходять для компресії сигналів.

Ефективність застосування вейвлет-перетворення і ДКП для компресії зображень пояснюється тим, що вони добре моделюють процес обробки зображення в ЗСЛ, відділяючи суттєві деталі від другорядних. Таким чином, дані перетворення більш доцільно використовувати в випадків присутності активного порушника, оскільки модифікація значущих коефіцієнтів може призвести до неприйнятного спотворення зображення.

При застосуванні перетворень з низькими значеннями виграшу від кодування існує значна небезпека руйнування вбудованих даних, у зв'язку з тим, що коефіцієнти перетворення менш стійкі до модифікацій. Однак при

цьому існує велика гнучкість у виборі перетворення, і якщо останнє невідомо порушнику, то модифікувати стеганограму буде істотно складніше.

Під час цифрової обробки зображення часто застосовується двовимірна версія дискретного косинусного перетворення:

$$\Omega(\upsilon,\nu)=\frac{\xi(\upsilon)\cdot\xi(\nu)}{\sqrt{2N}}\cdot\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}C(x,y)\cdot\cos\left[\frac{\pi\cdot\upsilon\cdot(2x+1)}{2N}\right]\cdot\cos\left[\frac{\pi\cdot\upsilon\cdot(2y+1)}{2N}\right]; \quad (3.2)$$

$$S(x,y)=\frac{1}{\sqrt{2N}}\cdot\sum_{\upsilon=0}^{N-1}\sum_{\upsilon=0}^{N-1}\xi(\upsilon)\cdot\xi(\nu)\cdot\Omega(\upsilon,\nu)\cdot\cos\left[\frac{\pi\cdot\upsilon\cdot(2x+1)}{2N}\right]\cdot\cos\left[\frac{\pi\cdot\upsilon\cdot(2y+1)}{2N}\right].$$

де C(c,y) і S(x,y) - відповідно, елементи оригінального і відновленого за коефіцієнтами ДКП зображення розмірністю $N\times N$; x, y - просторові координати пікселів зображення; $\Omega(\upsilon,\nu)$ - масив коефіцієнтів ДКП; υ,ν - координати в частотній області; $\xi(\upsilon)=1/\sqrt{2}$, якщо $\upsilon=0$, і $\xi(\upsilon)=1$, якщо $\upsilon>0$.

Розглянемо існуючі методи, які базуються на алгоритмі ДКП.

3.2 Метод відносної заміни величин коефіцієнтів ДКП (метод Коха і Жао)

Один з найбільш поширених на сьогодні методів приховування конфіденційної інформації в частотній області зображення полягає у відносній заміні величин коефіцієнтів ДКП, який свого часу описали Кох (E. Koch) і Жао (J. Zhao).

На початковому етапі первинне зображення розбивається на блоки розмірністю 8x8 пікселів. ДКП застосовується до кожного блоку - формула (3.2), в результаті чого отримують матриці 8x8 коефіцієнтів ДКП, які часто позначають $\Omega_b(\upsilon,\nu)$, де b - номер блоку контейнера C, а (υ,ν) - позиція коефіцієнта в цьому блоці. Кожен блок при цьому призначений для приховування одного біта даних.

Було запропоновано дві реалізації алгоритму: псевдовипадково можуть обиратися два або три коефіцієнти ДКП. Розглянемо перший варіант.

Під час організації секретного каналу абоненти повинні попередньо домовитися про два конкретні коефіцієнти ДКП з кожного блоку, які будуть використовуватися для приховування даних. Задамо дані коефіцієнти їх координатами в масивах коефіцієнтів ДКП: (υ_1,ν_1) і (υ_2,ν_2). Крім цього, зазначені коефіцієнти повинні відповідати косинус-функції з середніми частотами, що забезпечить прихованість інформації в істотних для ЗСЛ областях сигналу, до того ж інформація не буде спотворюватися при JPEG-компресії з малим коефіцієнтом стиснення.

Безпосередньо процес приховування починається з випадкового вибору блоку C_b зображення, призначеного для кодування b-го біту повідомлення. Вбудовування інформації здійснюється таким чином: для передачі біта "0" прагнуть, щоб різниця абсолютних значень коефіцієнтів ДКП перевищувала

деяку позитивну величину, а для передачі біта "1" ця різниця робиться меншою в порівнянні з деякою від'ємною величиною:

$$\begin{cases} \left|\Omega_b(\upsilon_1,\nu_1)\right| - \left|\Omega_b(\upsilon_2,\nu_2)\right| > P, \text{ при } m_b = 0; \\ \left|\Omega_b(\upsilon_1,\nu_1)\right| - \left|\Omega_b(\upsilon_2,\nu_2)\right| < -P, \text{ при } m_b = 1. \end{cases} \quad (3.3)$$

Таким чином, первинне зображення спотворюється за рахунок внесення змін до коефіцієнтів ДКП, якщо їх відносна величина не відповідає приховуваному біту. Чим більше значення P, тим стеганосистема, створена на основі даного методу, є більш стійкою до компресії, однак якість зображення при цьому значно погіршується.

Після відповідного внесення корекції в значення коефіцієнтів, які повинні задовольняти нерівності (3.3), проводиться зворотне ДКП.

Для вилучення даних в декодері виконується аналогічна процедура вибору коефіцієнтів, а рішення про переданий біт приймається за наступним правилом:

$$\begin{cases} m_b^* = 0, \quad \text{при} \left|\Omega_b^*(\upsilon_1,\nu_1)\right| > \left|\Omega_b(\upsilon_2,\nu_2)\right| \\ m_b^* = 1, \quad \text{при} \left|\Omega_b^*(\upsilon_1,\nu_1)\right| < \left|\Omega_b(\upsilon_2,\nu_2)\right| \end{cases} \quad (3.4)$$

3.3 Метод Бенгама-Мемона-Эо-Юнг

Бенгам (D. Benham), Мемон (N. метопах), Ео (B.-L. Yeo) і Юнг (Minerva Yeung) запропонували оптимізовану версію вищерозглянутого методу. Причому оптимізація була проведена ними за двома напрямками: по-перше, було запропоновано для вбудовування використовувати не всі блоки, а тільки найбільш підходящі для цього, по-друге, в частотній області блоку для вбудовування вибираються не два, а три коефіцієнти ДКП, що істотно зменшує візуальні спотворення контейнера. Розглянемо відмічені удосконалення більш докладно.

Придатними для вбудовування інформації вважаються такі блоки зображення, які одночасно задовольняють наступним двом вимогам:

- блоки не повинні мати різких переходів яскравості;
- блоки не повинні бути занадто монотонними.

Блоки, які не відповідають першій вимозі, характеризуються наявністю надто великих значень низькочастотних коефіцієнтів ДКП, порівнянних за своїми розмірами з DC-коефіцієнтом. Для блоків, які не задовольняють другій вимозі, характерно рівність нулю більшості високочастотних коефіцієнтів. Зазначені особливості є критерієм відбракування непридатних блоків.

Зазначені вимоги відбракування враховуються використанням двох порогових коефіцієнтів: $\mathbf{P_L}$ (для першої вимоги) і $\mathbf{P_H}$ (для другої вимоги), перевищення ($\mathbf{P_L}$) або недосягнення ($\mathbf{P_H}$) яких буде вказувати на те, що даний блок є непридатним для модифікації в частотній області.

Вбудовування в блок біта повідомлення відбувається наступним чином. Вибираються (для більшої стійкості стеганосистеми - псевдовипадково) три коефіцієнти ДКП блоку з середньочастотної області з координатами (υ_1, ν_1),

(υ_2,ν_2) і (υ_3,ν_3). Якщо необхідно провести вбудовування "0", ці коефіцієнти змінюються таким чином (якщо, звичайно, це необхідно), щоб третій коефіцієнт став меншим за будь-який з перших двох; якщо необхідно приховати "1", він стає великим у порівнянні з першим і другим коефіцієнтами:

$$\begin{cases} \left.\begin{array}{l} |\Omega_b(\upsilon_3,\nu_3)| < |\Omega_b(\upsilon_1,\nu_1)|; \\ |\Omega_b(\upsilon_3,\nu_3)| < |\Omega_b(\upsilon_2,\nu_2)| \end{array}\right\}, \textit{при } m_b = 0; \\ \left.\begin{array}{l} |\Omega_b(\upsilon_3,\nu_3)| > |\Omega_b(\upsilon_1,\nu_1)| \\ |\Omega_b(\upsilon_3,\nu_3)| > |\Omega_b(\upsilon_2,\nu_2)| \end{array}\right\}, \textit{при } m_b = 1. \end{cases} \tag{3.5}$$

Як і в попередньому методі, для прийняття рішення про достатність розрізнення зазначених коефіцієнтів ДКП, в вираз (3.5) вводиться значення порогу розрізнення P:

$$\begin{cases} |\Omega_b(\upsilon_3,\nu_3)| < \min\left(|\Omega_b(\upsilon_1,\nu_1)|, |\Omega_b(\upsilon_2,\nu_2)|\right) - P, \textit{при } m_b = 0; \\ |\Omega_b(\upsilon_3,\nu_3)| > \max\left(|\Omega_b(\upsilon_1,\nu_1)|, |\Omega_b(\upsilon_2,\nu_2)|\right) + P, \textit{при } m_b = 1. \end{cases} \tag{3.6}$$

У тому, коли така модифікація призводить до занадто великої деградації зображення, коефіцієнти не змінюють, і блок в якості контейнера не використовується.

Використання трьох коефіцієнтів замість двох і, що найголовніше, відмова від модифікації блоків зображення в разі неприйнятних їх спотворень, зменшує похибки, що вносяться повідомленням. Одержувач завжди може визначити блоки, в які не проводилося вбудовування, просто повторивши аналіз, аналогічний виконаному на передавальній стороні.

3.4 Метод Фридрих

Алгоритм, запропонований Джесікою Фридрих (J. Fridrich), по суті є комбінацією двох алгоритмів: відповідно до одного з них приховувані дані вбудовуються в низькочастотні, а за іншим - в середньочастотні коефіцієнти ДКП. Каскадне використання двох різних алгоритмів дозволяє отримати хороші результати щодо стійкості стеганографічної системи до атак.

Зображення, яке планується використовувати в якості контейнера, конвертується в сигнал з нульовим математичним очікуванням (сподіванням) і певним стандартним відхиленням таким чином, щоб НЧ-коефіцієнти ДКП, які будуть обчислені в подальшому, потрапляли в попередньо заданий незмінний діапазон. Запропоноване перетворення

$$G = \frac{1024}{\sqrt{X \cdot Y}} \cdot \frac{C - \overline{C}}{\sigma(C)} \tag{3.7}$$

де X, Y - розмірність зображення C в пікселях; $\overline{C}$ і $\sigma(C)$ - відповідно, математичне очікування (сподівання) і стандартне відхилення значень

яскравості пікселів зображення, - трансформує напівтонове зображення C в двовимірний сигнал G з нульовим математичним очікуванням (сподіванням). В цьому випадку абсолютне значення максимального НЧ-коефіцієнта ДКП сигналу G не буде перевищувати поріг (200 ... 250). Крім того, стверджується, що дане перетворення можна застосувати для широкого кола різноманітних зображень: як з великими однорідними областями, так і сильно текстурованих.

Для сигналу-зображення G проводиться обчислення коефіцієнтів ДКП, з усієї множини яких модифікуються тільки низькочастотні. Причому модифікування проводиться таким чином, щоб в коефіцієнтах було закодовано приховуване повідомлення W, що представляє собою сигнал у вигляді послідовності чисел {-1,1}. Для цього попередньо необхідно визначити геометричну прогресію дійсних чисел

$$\tau_{i+1} = \frac{1+\alpha}{1-\alpha} \cdot \tau_i; \quad \tau_1 = 1 \tag{3.8}$$

параметризовані (що настроюються) за допомогою параметра $\alpha \in (0,1)$.

Для значень $t > 1, \tau_i \le t < \tau_{i+1}$ визначається індексна функція

$$\text{ind}(t) = (-1)^i, \text{ если } t \in [\tau_i, \tau_{i+1}) \tag{3.9}$$

що дозволяє для кожного дійсного числа $t > 1$ визначити його індекс (+ / -1). Зрозуміло, що зазначений індекс може бути змінений шляхом додавання або ж віднімання числа, що не перевищує значення $\alpha \cdot t$. На рис. 4.5 наведені індексні функції для $\alpha = 0.1, 0.2 \; ma \; 0.3$.

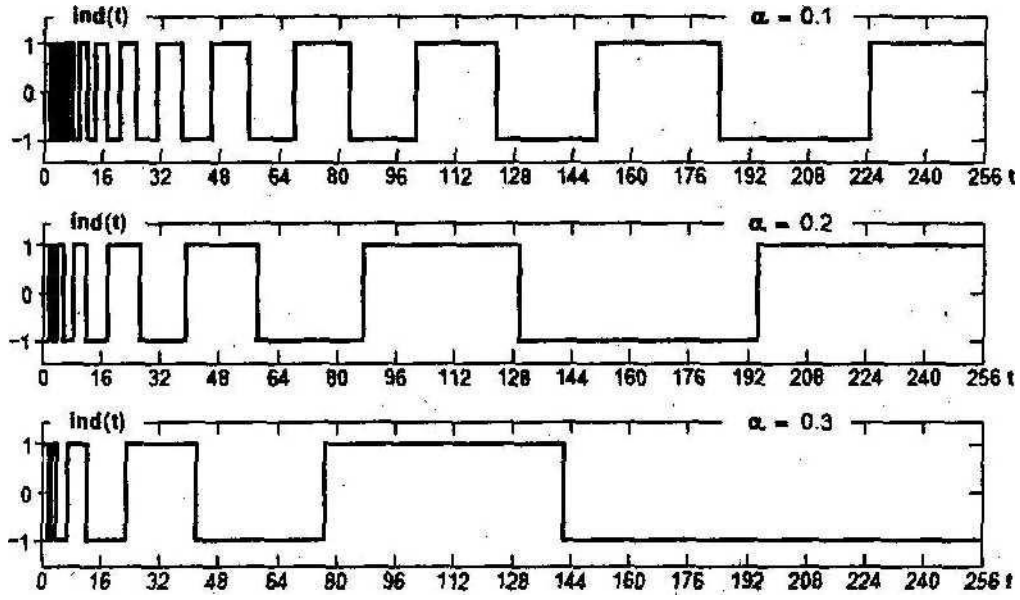


Рисунок 4.5 - Индексна функція **Ind(t)** при значеннях $\alpha = 0.1, 0.2 \; ma \; 0.3$

Для вбудовування масиву повідомлення W, кожен окремий біт якого може приймати значення $W_j \in \{-1, 1\}$, при $j \in \{1, 2, \ldots, \aleph_W\}$, вибираються $\aleph_{\Omega low} = \aleph_W$ НЧ-коефіцієнтів ДКП - Ω_j, значення яких змінюються таким

чином, щоб виконувалася умова $\mathrm{ind}\left(\left|\Omega^{\wedge}_{j}\right|\right)=W_j$, де $\Omega^{\wedge}_{j}$ - модифіковане значення коефіцієнта ДКП. У тому випадку $|\Omega_j|<1$, якщо, коефіцієнт для вбудовування не використовується. Завдяки властивостям індексного функції, кожен коефіцієнт буде змінений не більше ніж на $100\cdot\alpha$ відсотків. Також зазначається, що зміни будуть носити випадкових характер, оскільки не існує ніяких підстав вважати, що коефіцієнти ДКП на початковому етапі кодування є наслідком певного повідомлення.

Найбільша стійкість стеганосистеми до спотворень контейнера досягається при встановленні в якості нових значень коефіцієнтів ДКП середини інтервалів $[\tau_i,\tau_{i+1})$. Однак це може послужити появі скупчень однакових коефіцієнтів ДКП, що робить систему ненадійною з точки зору можливого стеганографічного аналізу. Значення параметра а вибирається таким чином, щоб вбудовування повідомлення не призводило до помітних для ока спотворень контейнера.

Операція вилучення проводиться шляхом виконання аналогічних з операцією вбудовування перетворень контейнера, який підозрюється на наявність прихованого повідомлення:

- конвертація в сигнал з нульовим математичним очікуванням (сподіванням) за формулою (3.7);
- обчислення коефіцієнтів ДКП конвертованого зображення;
- обчислення для заздалегідь обумовлених коефіцієнтів ДКП індексного функції (3.9) при заданому параметрі α;
- формування з отриманих індексів масиву витягнутого повідомлення.

Крім того, Фридрих запропонувала метод детектування наявності / відсутності вбудованого повідомлення в контейнері, що може бути корисним при захисті цифрового контенту (інформаційного вмісту) за допомогою ЦВЗ. Дана операція передбачає обізнаність одержувача щодо змісту прихованого повідомлення.

У зв'язку з тим, що більшість з $\aleph_{\Omega\mathrm{low}}$ НЧ-коефіцієнтів було піддано модифікації під час кодування, просте обчислення кореляції між W_j та $\mathrm{ind}\left(\left|\Omega^{\wedge}_{j}\right|\right)$, зумовлювало б собою нестійкість методу, оскільки малі, візуально незначні коефіцієнти ДКП роблять такий же ваговий внесок в загальну енергію сигналу, як і великі, візуально більш значущі коефіцієнти.

Оскільки попередньо було висунуто умову, що контейнер з вбудованим повідомленням не повинен привертати увагу, ми не можемо вбудовувати дані тільки в коефіцієнти, що мають велике значення. Крім того, позиції найбільших коефіцієнтів ДКП первинного і модифікованого зображень можуть не збігатися, що унеможливить безпомилкову ідентифікацію тих з них, в які було зроблено вбудовування. У запропонованій автором системі вбудовування здійснюється в усі НЧ-коефіцієнти, незалежно від їх значення (звичайно, крім тих, які не перевищують одиниці), проте тільки найбільші з них враховуються згодом

при обчисленні коефіцієнта кореляції, що зважується з енергією абсолютних значень коефіцієнтів ДКП:

$$K = \frac{\sum_{j=1}^{\aleph_{\Omega low}} \left|\Omega_j^*\right|^\beta \cdot \mathrm{ind}\left(\left|\Omega^*_j\right|\right) W_j}{\sum_{j=1}^{\aleph_{\Omega low}} \left|\Omega_j^*\right|^\beta} \tag{3.10}$$

Таке зважування автоматично робить більш виразними найбільші значення коефіцієнтів, одночасно пригнічуючи незначні, які могли зазнати змін в результаті будь-яких операцій з обробки зображення. Параметр встановлює важливість зважування. Якщо $\beta = 0$, то обчислюється звичайний, незважений коефіцієнт кореляції. Значення β, яке є близьким до одиниці, призводить до сингулярності (виродження) системи детектування: функція виявлення буде залежати тільки від значення всього лише одного біта, що відповідає найбільшому коефіцієнту ДКП. Автор методу рекомендує використовувати значення $\beta \in (0.5,1)$.

Більш стійкою до атак дану систему можна зробити шляхом пошуку максимального значення коефіцієнта кореляції щодо стандартного відхилення значень яскравості пікселів зображення, підозрюваного на присутність вбудованого повідомлення.

Масштабування (3.7) залежить від стандартного відхилення значень яскравості пікселів, яке може бути суттєво спотворено, якщо зображення з вбудованим повідомленням було піддано згладжуванню, або, наприклад, додатково зашумлено. Як наслідок, коефіцієнти ДКП такого зображення будуть промасштабовані за допомогою фіксованого коефіцієнта (відношення стандартних відхилень оригінального і досліджуваного на наявність прихованого повідомлення зображень $d = \sigma(C)/\sigma(S)$). Однак повідомлення, закодованого в коефіцієнтах ДКП, лінійні зміни не торкнуться. Це робить доцільним використання простого одновимірного пошуку правильного масштабу d, який би максимізував значення коефіцієнта кореляції (оскільки первинне зображення, що використовується в якості контейнера, в детекторі відсутнє). Таким чином, доповнена функція детектування має наступний вигляд:

$$K' = \max_{d \in (1-\delta, 1+\delta)} K(d) \frac{\sum_{j=1}^{\aleph_{\Omega low}} \left|\Omega_j^*\right|^\beta \cdot \mathrm{ind}\left(\left|\Omega^*_j\right|\right) W_j}{\sum_{j=1}^{\aleph_{\Omega low}} \left|\Omega_j^*\right|^\beta} \tag{3.11}$$

Встановлено, що навіть при значних спотвореннях зображення в результаті атак, достатнім буде крок відхилення масштабу $\delta = 0{,}25$.

Труднощі детектування, що виникають при цьому, вимагають зменшення інформаційного змісту повідомлення і додавання коригувальних бітів. Отже, оскільки внесок в виявлення повідомлення вносять тільки найбільші коефіцієнти ДКП, інформаційний зміст повідомлення довжиною

$\aleph_W$ є лише певною часткою від $\aleph_W$. Крім того, цілком очевидно, що одновимірний пошук масштабного коефіцієнта, який максимізує коефіцієнт кореляції, збільшує відсоток помилкових виявлень.

Для досягнення властивостей високої стійкості до атак на стеганосистему при одночасному мінімальному (наскільки, звичайно, це можливо) спотворенні контейнера Фридрих було запропоновано вмонтувати в контейнер додаткове повідомлення, використовуючи методику розширення спектру. При цьому вбудовування повідомлення здійснюється шляхом додавання шумоподібного сигналу до СЧ-коефіцієнтів ДКП зображення. Кількість таких коефіцієнтів ($\aleph_{\Omega mid}$) складає приблизно 30% від загальної кількості коефіцієнтів ДКП.

Вважається, що інформація, яку несе додаткове повідомлення, складається з $\aleph_{W^+}$ символів W_j^+, кожен з яких може бути представлений десятковим цілим числом, $1 \le W_j^+ \le \max(W^+)$.

Для кожного j-го символу генерується послідовність $\xi^{(j)}$ ПВЧ, рівномірно розподілених в інтервалі [0,1]. Початковий стан генератора ПВЧ може виступати в ролі секретного ключа. Потужність j-ої множини ПВЧ: $\left|\xi^{(j)}\right| \ge \aleph_{\Omega mid} + \max(W^+)$.

Для подання окремого символу повідомлення W^+, з множини $\xi^{(j)}$ виділяється сегмент $\eta^{|j|} = \xi^{(j)}_{W_j^+}, \ldots . \xi^{(j)}_{W_j^+ + \aleph_{\Omega mid}-1}$, який містить $\aleph_{\Omega mid}$ елементів.

В результаті, повідомлення із $\aleph_{W^+}$ символів може бути представлено у вигляді такої суми:

$$Spr = \frac{\left[\sum_{j=1}^{\aleph_{W^+}} \eta^{(j)}\right] - \frac{\aleph_{W^+}}{2}}{\sqrt{\aleph_{W^+}/12}} \qquad (3.12)$$

Сигнал з розширеним спектром Spr характеризується приблизно нормальним (гаусовим) розподілом з нульовим математичним очікуванням (сподіванням) і одиничним стандартним відхиленням (точність апроксимації зростає зі збільшенням значення $\aleph_{W^+}$. Надалі сигнал Spr множиться на параметр γ, який регулює відношення "стійкість / помітність вбудовування" і поелементно підсумовується з $\aleph_{\Omega mid}$ вибраними СЧ-коефіцієнтами. Вилучення повідомлення виконується шляхом попереднього обчислення коефіцієнтів ДКП зображення і виділення серед них саме середньочастотних (дана операція повинна бути узгодженою з відповідною дією на етапі вбудовування). Використовуючи секретний ключ / алгоритм, здійснюється генерація послідовностей ПВЧ (загальною кількістю $\aleph_{W^+}$, якщо цей параметр відомий; в іншому випадку - за обставинами, виходячи з аналізу вже отриманої частини повідомлення) довжиною $\aleph_{\Omega mid} + \max(W^+)$.

З кожної послідовності $\xi^{(j)}$ виділяється $\max(W^+)$ сегментів довжиною $\aleph_{\Omega mid}$ елементів, для яких розраховується взаємна кореляція з вектором виділених СЧ-коефіцієнтів. Позиція найбільшого значення функції кореляції в отриманому при цьому векторі і буде визначати значення, яке ймовірно мав вбудований символ W^+_j.

Збільшення параметрів α і, особливо, γ робить дану стеганосистему ще більш стійкою до атаки компресією, однак при цьому сильно страждає якість зображення.

4. Питання для поточного контролю підготовленості студентів до виконання лабораторної роботи №4

1. Основні етапи алгоритму стиску зображень JPEG. Які етапи алгоритму JPEG призводять до стиску зображення?
2. Дискретно-косинусне перетворення. Основні співвідношення та властивості.
3. Метод приховування даних у частотну області нерухомих зображень шляхом кодування різниці абсолютних значень коефіцієнтів дискретно-косинусного перетворення (метод Коха-Жао).
4. Метод приховування даних у частотну область нерухомих зображень шляхом кодування декількох різниць абсолютних значень коефіцієнтів дискретно-косинусного перетворення (удосконалений метод Коха-Жао – метод Бенгама-Мемона-Ео-Юнга).
5. Метод приховування цифрових водяних знаків (ЦВЗ) у частотну область нерухомих зображень Хсу-Ву. В чому перевага цього методу у порівнянні із методом Коха-Жао?
6. Приховування та вилучення інформаційних даних у частотну область нерухомих зображень методом Джесіки Фрідріх. Переваги та недоліки методу.

5. Інструкція до виконання лабораторної роботи №4

Завдання 1. Реалізація в середовищі MathCAD алгоритмів прямого та зворотного дискретно-косинусного перетворення. Дослідження ефекту частотної чутливості зорової системи людини

1.1. Завантажуємо вихідні дані: контейнер - нерухоме зображення (в форматі *.bmp24); інформаційне повідомлення - текстовий документ (у форматі *.txt). Для цього в середовищі MathCAD виконуємо дії, аналогічні описаним в п. 1.1. інструкції до лабораторної роботи №1.

1.2. Перетворюємо масив інформаційних даних. Для цього в середовищі MathCAD виконуємо дії, аналогічні описаним в п. 1.2. інструкції до лабораторної роботи №1.

1.3. Реалізуємо алгоритми прямого та зворотного дискретно-косинусного перетворення. Для цього в середовищі MathCAD виконуємо послідовність перетворень, представлених на рис. 4.1.

$N := 8 \quad P := 1$

$C := \left| \begin{array}{l} \text{for } i \in 0..N-1 \\ \quad \left| \begin{array}{l} C_i \leftarrow \frac{1}{\sqrt{2}} \text{ if } i = 0 \\ C_i \leftarrow 1 \text{ if } i > 0 \end{array} \right. \\ C \end{array} \right.$

$$C = \begin{pmatrix} 0.707 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$$T(V) := \left| \begin{array}{l} \text{for } i \in 0..N-1 \\ \quad \text{for } j \in 0..N-1 \\ \quad\quad T_{i,j} \leftarrow \text{round}\left[\frac{2}{N} \cdot C_i C_j \cdot \frac{1}{P} \cdot \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \left[V_{x,y} \cdot \cos\left[\frac{(2x+1) \cdot i \cdot \pi}{2N} \right] \cdot \cos\left[\frac{(2y+1) \cdot j \cdot \pi}{2N} \right] \right] \right] \\ T \end{array} \right.$$

$$V(T) := \left| \begin{array}{l} \text{for } i \in 0..N-1 \\ \quad \text{for } j \in 0..N-1 \\ \quad\quad V_{i,j} \leftarrow \text{round}\left[\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \left[T_{x,y} \cdot P \cdot \frac{2}{N} \cdot C_x C_y \cdot \cos\left[\frac{(2i+1) \cdot x \cdot \pi}{2N} \right] \cdot \cos\left[\frac{(2j+1) \cdot y \cdot \pi}{2N} \right] \right] \right] \\ V \end{array} \right.$$

Рисунок 4.1 – Реалізація дискретно-косинусного перетворення в середовищі символьної математики MathCAD

На рис. 4.1 наведені наступні елементи.

Змінна N задає розмір матриці, над якою виконується перетворення, змінна P задає величину порога загрубления. Якщо P = 1, тоді загрублення не проводиться. Використовуємо значення N = 8, тому що такий параметр використовує алгоритм стиснення JPEG.

Змінна C містить службовий масив даних з восьми елементів, необхідних для коректного обчислення дискретно-косинусного перетворення.

Функція T(V) реалізує пряме дискретно-косинусное перетворення масиву V з NxN чисел. В якості аргумента функції T(V) використовуються окремі блоки растрових даних в просторовій області.

Функція V(T) реалізує зворотне дискретно-косинусное перетворення масиву T з NxN чисел. В якості аргумента функції V (T) використовуються окремі блоки растрових даних в частотній області. Змінна P задає величину порога загрублення коефіцієнтів дискретно-косинусного перетворення.

1.4. Розіб'ємо вихідне зображення на блоки, розміром NxN пікселів кожен, виконаємо пряме дискретно-косинусное перетворення для кожного блоку зображення. Для цього в середовищі MathCAD виконуємо послідовність перетворень, представлених на рис. 4.2.

$$nn := \frac{cols(R)}{N} \qquad mm := \frac{rows(R)}{N}$$

```
r := | for x ∈ 0..mm − 1
     |   for y ∈ 0..nn − 1
     |     | for i ∈ 0..N − 1
     |     |   for j ∈ 0..N − 1
     |     |     RR[i,j] ← R[i+x·N, j+y·8]
     |     | r[x,y] ← RR
     | r
```

```
tr := | for i ∈ 0..mm − 1
      |   for j ∈ 0..nn − 1
      |     tr[i,j] ← T(r[i,j])
      | tr
```

Рисунок 4.2 – Розбиття контейнера-зображення на блоки та виконання над ними дискретно-косинусного перетворення

Величини mn і mm задають число блоків, на які розбито зображення. У масиві r будуть міститися блоки зображення розміром NxN пікселів в просторовій області, а в масиві tr будуть зберігатися ті ж блоки, але вже в частотній області, тобто це масиви коефіцієнтів дискретно-косинусного перетворення. Наприклад, для блоку з номерами 1,2 маємо значення, наведені на рис. 4.3.

$$r_{1,2} = \begin{pmatrix} 24 & 24 & 23 & 22 & 21 & 20 & 20 & 20 \\ 23 & 22 & 21 & 21 & 19 & 19 & 18 & 17 \\ 24 & 22 & 20 & 20 & 20 & 19 & 18 & 16 \\ 25 & 23 & 22 & 21 & 21 & 20 & 19 & 18 \\ 24 & 24 & 23 & 22 & 22 & 21 & 20 & 19 \\ 23 & 23 & 23 & 22 & 21 & 21 & 20 & 20 \\ 22 & 22 & 21 & 21 & 20 & 20 & 20 & 19 \\ 24 & 22 & 21 & 21 & 21 & 21 & 20 & 19 \end{pmatrix} \qquad tr_{1,2} = \begin{pmatrix} 168 & 13 & 0 & 2 & 0 & 1 & 0 & 0 \\ -1 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 4 & -1 & 0 & -3 & -1 & -1 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{pmatrix}$$

Рисунок 4.3 – Приклад блоків контейнера-зображення в просторовій та частотній області

Значення масиву $r_{1,2}$ (див. Рис. 4.3) характеризують величину яскравості (червоного кольору) окремих пікселів зображення, а значення масиву $tr_{1,2}$ характеризують величину окремих коефіцієнтів дискретно-косинусного перетворення, обчислених для блоку $r_{1,2}$. Ліва верхня частина масиву $tr_{1,2}$ відповідає низькочастотній області зображення, саме тут зосереджена основна «енергія» реалістичних зображень, що наочно видно за значеннями масиву $tr_{1,2}$. Права нижня частина відповідає високочастотній області, значення коефіцієнтів дискретно-косинусного перетворення в якій характеризують високочастотну, контрастну частину зображення. Для реалістичних зображень високочастотна область містить низькі по абсолютній величині значення, що наочно видно на рис. 4.3.

1.5. Для дослідження ефекту частотної чуттєвості зорової системи людини змінимо величини коефіцієнтів дискретно-косинусного перетворення в низькочастотній і високочастотній області зображення. Внесені зміни будемо оцінювати візуально, для чого виконаємо зворотне дискретно-косинусное перетворення над зміненим масивом коефіцієнтів. Для цього, наприклад, виконаємо перетворення, наведені на рис. 4.4.

Суть перетворень, наведених на рис. 4.4, наступна. Для блоку з номерами 7,7 на 30% збільшений коефіцієнт дискретно-косинусного перетворення з індексом (0,0). З використанням описаної в п. 1.3 функції V(T) для вихідного і зміненого масиву коефіцієнтів виконано зворотне дискретно-косинусное перетворення. Результат зміни яскравості пікселів (в збільшеному масштабі показані два зображення 8x8 пікселів) показує, що внесені спотворення візуально виявляються (загальний фон рисунка зліва значно темніше рисунка справа). Таким чином, навіть незначне (в межах 30%) спотворення низькочастотних коефіцієнтів дискретно-косинусного перетворення призводить до внесення видимих спотворень, загальний фон змінюється, що добре виявляється візуальним оглядом.

$A := tr_{7,7} \qquad B := A$

$B_{0,0} := A_{0,0} + \mathrm{floor}\left(0.3 A_{0,0}\right)$

$$A = \begin{pmatrix} 444 & 11 & 0 & 0 & 0 & 1 & -1 & 1 \\ -32 & 2 & -2 & 2 & 0 & 0 & 1 & 1 \\ -3 & 0 & -2 & 0 & 1 & 1 & 1 & 0 \\ -7 & -2 & 1 & -1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & -1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 577 & 11 & 0 & 0 & 0 & 1 & -1 & 1 \\ -32 & 2 & -2 & 2 & 0 & 0 & 1 & 1 \\ -3 & 0 & -2 & 0 & 1 & 1 & 1 & 0 \\ -7 & -2 & 1 & -1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & -1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$a := V(A)$ $\qquad b := V(B)$

$$a = \begin{pmatrix} 50 & 48 & 50 & 48 & 49 & 48 & 47 & 45 \\ 54 & 53 & 53 & 53 & 52 & 50 & 49 & 48 \\ 56 & 56 & 54 & 54 & 54 & 53 & 52 & 50 \\ 58 & 58 & 56 & 55 & 55 & 53 & 54 & 53 \\ 58 & 58 & 58 & 56 & 55 & 55 & 55 & 56 \\ 59 & 60 & 59 & 58 & 57 & 56 & 58 & 57 \\ 61 & 60 & 60 & 60 & 59 & 57 & 59 & 59 \\ 63 & 63 & 63 & 63 & 61 & 60 & 60 & 60 \end{pmatrix} \qquad b = \begin{pmatrix} 67 & 64 & 67 & 65 & 66 & 64 & 63 & 62 \\ 71 & 70 & 70 & 70 & 68 & 66 & 65 & 65 \\ 73 & 73 & 71 & 71 & 71 & 70 & 69 & 66 \\ 74 & 75 & 72 & 72 & 72 & 70 & 71 & 69 \\ 75 & 75 & 75 & 73 & 72 & 72 & 72 & 73 \\ 76 & 77 & 75 & 74 & 74 & 73 & 75 & 74 \\ 77 & 77 & 77 & 77 & 76 & 74 & 76 & 75 \\ 80 & 80 & 79 & 80 & 77 & 77 & 77 & 76 \end{pmatrix}$$

Рисунок 4.4 – Демонстрація ефекту високої частотної чутливості зорової системи людини до незначного змінення низькочастотних коефіцієнтів зображення

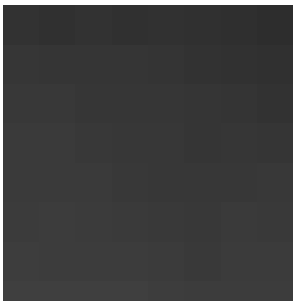
Для розглянутого прикладу внесемо також зміни в коефіцієнт дискретно-косинусного перетворення з індексом (7,6). Це високочастотний коефіцієнт і, згідно з теоретичними відомостями, чутливість зорової системи людини до таких змін дуже низька. Збільшимо обраний коефіцієнт дискретно-косинусного на 100% і проведемо аналогічні перетворення (див. Рис. 4.5). Як видно з наведених даних навіть після внесення значних змін високочастотного коефіцієнта (збільшення на 100%) спотворення візуально не виявляються.

$$B_{7,6} := A_{7,6} + \mathrm{floor}\left(A_{7,6}\right)$$

$$A = \begin{pmatrix} 444 & 11 & 0 & 0 & 0 & 1 & -1 & 1 \\ -32 & 2 & -2 & 2 & 0 & 0 & 1 & 1 \\ -3 & 0 & -2 & 0 & 1 & 1 & 1 & 0 \\ -7 & -2 & 1 & -1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & -1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 444 & 11 & 0 & 0 & 0 & 1 & -1 & 1 \\ -32 & 2 & -2 & 2 & 0 & 0 & 1 & 1 \\ -3 & 0 & -2 & 0 & 1 & 1 & 1 & 0 \\ -7 & -2 & 1 & -1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & -1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 2 & 0 \end{pmatrix}$$

$$a := V(A) \qquad b := V(B)$$

$$a = \begin{pmatrix} 50 & 48 & 50 & 48 & 49 & 48 & 47 & 45 \\ 54 & 53 & 53 & 53 & 52 & 50 & 49 & 48 \\ 56 & 56 & 54 & 54 & 54 & 53 & 52 & 50 \\ 58 & 58 & 56 & 55 & 55 & 53 & 54 & 53 \\ 58 & 58 & 58 & 56 & 55 & 55 & 55 & 56 \\ 59 & 60 & 59 & 58 & 57 & 56 & 58 & 57 \\ 61 & 60 & 60 & 60 & 59 & 57 & 59 & 59 \\ 63 & 63 & 63 & 63 & 61 & 60 & 60 & 60 \end{pmatrix} \qquad b = \begin{pmatrix} 50 & 48 & 50 & 48 & 49 & 48 & 47 & 45 \\ 54 & 53 & 53 & 53 & 52 & 50 & 49 & 48 \\ 56 & 56 & 55 & 54 & 54 & 53 & 52 & 50 \\ 58 & 58 & 56 & 55 & 55 & 53 & 55 & 52 \\ 58 & 58 & 58 & 56 & 55 & 55 & 55 & 56 \\ 59 & 60 & 59 & 58 & 57 & 56 & 58 & 57 \\ 61 & 60 & 60 & 60 & 59 & 57 & 59 & 59 \\ 63 & 63 & 63 & 63 & 61 & 60 & 60 & 60 \end{pmatrix}$$



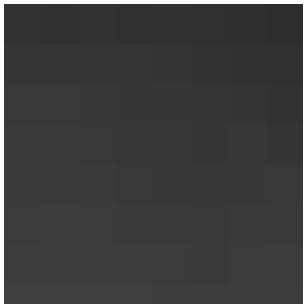


Рисунок 4.5 – Демонстрація ефекту низької частотної чутливості зорової системи людини до незначного змінення високочастотних коефіцієнтів зображення

Проведемо додаткові дослідження, посилимо внесені спотворення високочастотного коефіцієнту. Для цього змінимо обраний коефіцієнт на 1000% і проведемо відповідні перетворення (див. Рис. 4.6).

Як видно з наведених на рис. 4.6 даних загальний фон зображення не змінився, проте з'явилися незначні високочастотні спотворення, які при природному масштабі також візуально не фіксуються.

$$B_{7,6} := A_{7,6} + 10 \cdot \mathrm{floor}\left(A_{7,6}\right)$$

$$A = \begin{pmatrix} 444 & 11 & 0 & 0 & 0 & 1 & -1 & 1 \\ -32 & 2 & -2 & 2 & 0 & 0 & 1 & 1 \\ -3 & 0 & -2 & 0 & 1 & 1 & 1 & 0 \\ -7 & -2 & 1 & -1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & -1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 444 & 11 & 0 & 0 & 0 & 1 & -1 & 1 \\ -32 & 2 & -2 & 2 & 0 & 0 & 1 & 1 \\ -3 & 0 & -2 & 0 & 1 & 1 & 1 & 0 \\ -7 & -2 & 1 & -1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & -1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 11 & 0 \end{pmatrix}$$

$a := V(A)$ $\qquad b := V(B)$

$$a = \begin{pmatrix} 50 & 48 & 50 & 48 & 49 & 48 & 47 & 45 \\ 54 & 53 & 53 & 53 & 52 & 50 & 49 & 48 \\ 56 & 56 & 54 & 54 & 54 & 53 & 52 & 50 \\ 58 & 58 & 56 & 55 & 55 & 53 & 54 & 53 \\ 58 & 58 & 58 & 56 & 55 & 55 & 55 & 56 \\ 59 & 60 & 59 & 58 & 57 & 56 & 58 & 57 \\ 61 & 60 & 60 & 60 & 59 & 57 & 59 & 59 \\ 63 & 63 & 63 & 63 & 61 & 60 & 60 & 60 \end{pmatrix} \qquad b = \begin{pmatrix} 51 & 47 & 50 & 48 & 49 & 48 & 46 & 45 \\ 53 & 54 & 52 & 53 & 52 & 48 & 50 & 48 \\ 57 & 54 & 56 & 53 & 53 & 55 & 50 & 51 \\ 57 & 60 & 54 & 56 & 56 & 51 & 57 & 52 \\ 59 & 56 & 60 & 55 & 54 & 57 & 53 & 57 \\ 59 & 62 & 57 & 58 & 58 & 54 & 60 & 56 \\ 61 & 59 & 61 & 60 & 59 & 59 & 58 & 59 \\ 63 & 64 & 62 & 64 & 61 & 60 & 60 & 59 \end{pmatrix}$$

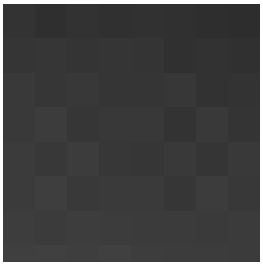


Рисунок 4.6 – Демонстрація ефекту низької частотної чутливості зорової системи людини до незначної зміни високочастотних коефіцієнтів зображення

Отже, внесення змін в різні частотні компоненти по-різному впливає на сприйняття цих змін зоровою системою людини: низькочастотні спотворення візуально фіксуються, високочастотні спотворення, як правило, непомітні. В цьому і проявляється ефект частотної чутливості, який будемо використовувати в подальшому при реалізації методів стеганографічного вбудовування.

Завдання 2. Реалізація в середовищі MathCAD алгоритмів вбудовування та вилучення повідомлень в частотну область зображень (метод Коха-Жао)

2.1. Реалізуємо алгоритм вбудовування інформаційних даних в частотну область зображення на основі кодування різниць абсолютних значень коефіцієнтів дискретно-косинусного перетворення. Для цього в середовищі MathCAD виконуємо послідовність перетворень, представлених на рис. 4.7 і 4.8.

$$Pr := 5$$

$$H(H1, H2) := \begin{cases} -1 \\ 1 \text{ if } |H1| - |H2| > Pr \\ 0 \text{ if } |H1| - |H2| < -Pr \end{cases}$$

$$\text{Input}(TRR, m) := \begin{cases} TRR \leftarrow TRR \\ \text{if } m = 1 \wedge m \neq H\left(TRR_{3,1}, TRR_{1,3}\right) \vee H\left(TRR_{3,1}, TRR_{1,3}\right) = -1 \\ \quad \begin{cases} TRR_{3,1} \leftarrow |TRR_{1,3}| + Pr \text{ if } TRR_{3,1} > 0 \\ TRR_{3,1} \leftarrow -|TRR_{1,3}| - Pr \text{ if } TRR_{3,1} \leq 0 \end{cases} \\ \text{if } m = 0 \wedge m \neq H\left(TRR_{3,1}, TRR_{1,3}\right) \vee H\left(TRR_{3,1}, TRR_{1,3}\right) = -1 \\ \quad \begin{cases} TRR_{1,3} \leftarrow |TRR_{3,1}| + Pr \text{ if } TRR_{1,3} > 0 \\ TRR_{1,3} \leftarrow -|TRR_{3,1}| - Pr \text{ if } TRR_{1,3} \leq 0 \end{cases} \\ \text{Input} \leftarrow TRR \end{cases}$$

Рисунок 4.7 – Вбудовування одного інформаційного біта в частотну область одного 8x8 блоку зображення

Величина Pr задає поріг зміни частотних коефіцієнтів при вбудовуванні інформаційних бітів.

Процедура H (H1, H2) реалізує логічне правило зміни абсолютного значення різниць коефіцієнтів дискретно-косинусного перетворення, позначених змінними H1 і H2:

- якщо перший коефіцієнт за абсолютним значенням більше другого на величину Pr, тоді це відповідає вбудовуванню біта «1»;
- якщо перший коефіцієнт за абсолютним значенням менше другого на величину Pr, тоді це відповідає вбудовуванню біта «0»;
- якщо різниця абсолютних значень коефіцієнтів знаходиться в діапазоні від -Pr до Pr, тоді це відповідає невизначеній ситуації, коли не можна детектувати ні біт «1», ні біт «0».

З використанням функції H(H1,H2) в процедурі Input(TRR,m) здійснюється кодування різниць абсолютних значень коефіцієнтів дискретно-косинусного перетворення в одному блоці 8x8 коефіцієнтів. Вхідними даними є масив TRR розмірністю 8×8 цілих чисел, а також бітова змінна m,

яка передає значення вбудованого біта. В якості змінних обрані коефіцієнти в середньочастотній області з номерами (3,1) і (1,3).

У наступній процедурі tr (див. Рис. 4.8) реалізується побітове вбудовування інформації в окремі блоки контейнера за допомогою циклічного виклику процедури Input(TRR,m), де в якості TRR виступає поточний блок контейнера, а в якості m - поточне значення вбудованого біта.

```
tr := | num ← 0
      | for x ∈ 0..mm − 1
      |   for y ∈ 0..nn − 1
      |     | break if x·nn + y ≥ rows(m) − 1
      |     | tr_{x,y} ← Input(tr_{x,y}, m_{num})
      |     | num ← num + 1
      | tr
```

```
vr := | for i ∈ 0..mm − 1
      |   for j ∈ 0..nn − 1
      |     vr_{i,j} ← V(tr_{i,j})
      | vr
```

```
R2 := | for x ∈ 0..rows(vr) − 1
      |   for y ∈ 0..cols(vr) − 1
      |     | temp1 ← vr_{x,y}
      |     | for i ∈ 0..N − 1
      |     |   for j ∈ 0..N − 1
      |     |     temp2_{x·N+i, y·N+j} ← temp1_{i,j}
      | temp2
```

Рисунок 4.8 – Побітове вбудовування послідовності інформаційних бітів в частотну область зображення

Процедура vr реалізує послідовне зворотне дискретно-косинусное перетворення над зміненими блоками контейнера. Після чого за допомогою процедури R2 блоки об'єднуються в один масив даних, тобто формується нове зображення в просторовій області з вже вбудованими інформаційними даними.

2.2. Виконаємо зорове порівняння двох зображень - до і після вбудовування інформаційних повідомлень. Для цього скористаємося масивами растрових даних (яскравостей пікселів червоного кольору) R і R2. Масив R містить растрові дані до вбудовування, R2 – масив, отриманий при виконанні попереднього пункту.

Слід зазначити, що отримане зображення R2 може містити некоректні значення, тому що внесення змін до частотної області безпосередньо впливає і на значення в просторовій області. Нові значення в просторовій області можуть бути вище 255 або нижче 0, проте в оброблюваному форматі зображення допустимими значеннями є тільки цілі числа від 0 до 255. Значення 256 середовищем символьної математики буде інтерпретовано як

число 0, 257 - як число 2, значення -3 - як число 253 і т.д. Для уникнення такої помилкової інтерпретації даних виконаємо наступну процедуру (див. Рис. 4.9), яка округлить всі числа, більші за 255 до 255, а всі числа менші 0 до 0.

```
R2 := | for i ∈ 0..rows(R2) − 1
      |   for j ∈ 0..cols(R2) − 1
      |     R2[i,j] ← 0   if R2[i,j] < 0
      |     R2[i,j] ← 255 if R2[i,j] > 255
      | R2
```

R

R2

Рисунок 4.9 – Обробка масиву растрових даних та виведення отриманого зображення

На рис. 4.9 наведені для візуального порівняння два зображення: до вбудовування інформаційних даних (зліва) і після внесених змін (праворуч). Зрозуміло, що візуально наведені зображення не відрізняються.

Для кількісної оцінки відмінностей зображень обчислимо середнє арифметичне поелементної різниці масивів R і R2 (див. Рис. 4.10).

Для количественной оценки различий изображений вычислим среднее арифметическое поэлементной разности массивов R и R2 (см. рис. 4.10).

$$RAZ := \left| \begin{array}{l} RAZ \leftarrow 0 \\ \text{for } i \in 0..rows(R2) - 1 \\ \quad \text{for } j \in 0..cols(R2) - 1 \\ \quad\quad RAZ \leftarrow RAZ + \left| R_{i,j} - R2_{i,j} \right| \\ RAZ \leftarrow \dfrac{RAZ}{rows(R) \cdot cols(R)} \\ RAZ \end{array} \right.$$

$$RAZ = 0.974$$

Рисунок 4.10 – Кількісна оцінка відмінностей між зображеннями до і після вбудовування інформаційного повідомлення

Зрозуміло, що зображення відрізняються дуже незначимо, отримана величина усереднених спотворень знаходиться нижче порога чутливості зорової системи людини, тобто при візуальному огляді спотворення не виявляються.

2.3. Реалізуємо алгоритм вилучення інформаційних даних з частотної області зображення. Для цього в середовищі MathCAD виконаємо послідовність перетворень, представлених на рис. 4.11.

```
tr1 := | for i ∈ 0..mm − 1
       |   for j ∈ 0..nn − 1
       |     tr[i,j] ← T(vr[i,j])
       | tr

m1 := | num ← 0
      | for x ∈ 0..mm − 1
      |   for y ∈ 0..nn − 1
      |     | TRR1 ← tr1[x,y]
      |     | m1[num] ← 1 if |TRR1[3,1]| > |TRR1[1,3]|
      |     | m1[num] ← 0 if |TRR1[3,1]| ≤ |TRR1[1,3]|
      |     | num ← num + 1
      | m1
```

m =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	…

m1 =

	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	…

Рисунок 4.11 – Побітовий витяг послідовності інформаційних бітів з частотної області зображення і порівняння даних

У першій процедурі tr1 (див. Рис. 4.11) реалізується пряме дискретно-косинусное перетворення всіх блоків контейнера. У наступній процедурі m1 виконується обчислення інформаційних бітів за допомогою вилучення з середньочастотної області масивів коефіцієнтів дискретно-косинусного перетворення. Для цього в кожному блоці порівнюються абсолютні значення коефіцієнтів з номерами (3,1) і (1,3). Якщо абсолютне значення коефіцієнта з номером (3,1) більше абсолютного значення коефіцієнта з номером (1,3) - тоді детектується одиничний інформаційний біт. В іншому випадку - детектується нульовий інформаційний біт.

На рис. 4.11 для порівняння наведені також значення масивів інформаційних бітів до вбудовування (зліва) і після вилучення (праворуч). Як видно з наведеного прикладу перші 15 інформаційних бітів збігаються.

Для кількісної ймовірності помилкового вилучення інформаційних даних виконаємо такі операції (див. Рис. 4.12).

$$Po := \left| \begin{array}{l} Po \leftarrow 0 \\ \text{for } i \in 0..rows(m1) - 1 \\ \quad Po \leftarrow Po + 1 \text{ if } m_i \neq m1_i \\ Po \leftarrow \dfrac{Po}{rows(m1)} \\ Po \end{array} \right.$$

$$Po = 0$$

Рисунок 4.12 – Оцінка ймовірності помилкового вилучення інформаційних даних

Як видно з отриманих результатів інформаційні біти, витягнуті з частотної області контейнера-зображення, повністю збіглися з вихідними даними. Це прогнозовано, тому що на заповнений контейнер не проводилося ніяких впливів.

Завдання 3. Реалізація в середовищі MathCAD стеганоатаки на основі використання алгоритму стискання JPEG та дослідження її можливостей

3.1. Для реалізації стеганоатаки спершу збережемо заповнений контейнер-зображення (стеганограму) у вигляді окремого файлу. Для цього сформуємо масиви яскравостей зеленого G2 і синього B2 кольору і виконаємо відповідну команду «WRITERGB» для запису растрових даних в файл (див. Рис. 4.13). В результаті виконання цієї команди в теці з реалізацією алгоритмів буде сформований файл «Stego.bmp».

Для візуального порівняння порожнього і заповненого контейнера виведемо на екран зображення до (зліва) і після (праворуч) вбудовування (см. Рис. 4.13).

3.2. Зимітуємо стеганоатаку на основі використання алгоритму стиснення JPEG. Для цього відкриємо файл «Stego.bmp» зовнішнім графічним редактором, наприклад, Adobe Photoshop, Corel PHOTO-PAINT або Microsoft Paint. На рис. 4.14 наведено приклад для випадку використання графічного редактора Corel PHOTO-PAINT. У відкритому редакторі збережемо (експортуємо) зображення у форматі JPEG. При цьому будемо використовувати високу якість[1], прийняту за замовчуванням (див. Рис. 4.14).

[1] В графічному редакторі Microsoft Paint можливість вибору якості зображення не передбачена. При виконання цього завдання лабораторної роботи слід використовувати ті налаштування графічного редактора, які прийнято за замовченням.

$$G2 := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(R2) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R2) - 1 \\ \quad\quad G2_{i,j} \leftarrow G_{i,j} \\ G2 \end{array} \right. \qquad B2 := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(R2) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R2) - 1 \\ \quad\quad B2_{i,j} \leftarrow B_{i,j} \\ B2 \end{array} \right.$$

$$\text{WRITERGB}(\text{"Stego.bmp"}) := \text{augment}(R2, G2, B2)$$

"1"

"Stego"

Рисунок 4.13 – Запис заповненого контейнера-зображення в файл «Stego.bmp» і виведення зображення

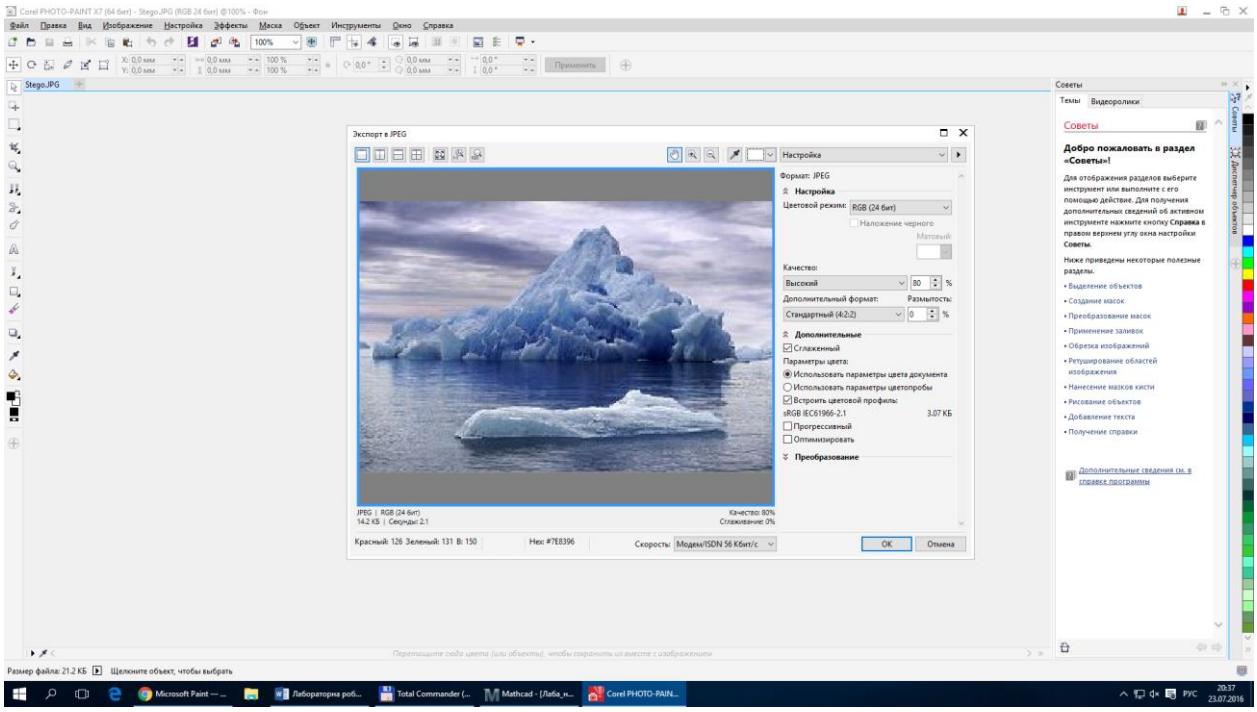


Рисунок 4.14 – Імітація стеганоатаки на основі алгоритму стиснення JPEG в графічному редакторі Corel PHOTO-PAINT

Отримане в результаті зазначених перетворень зображення буде збережено у форматі JPEG, при цьому інформаційні дані, що містяться в ньому, можуть спотворитися в результаті виконання алгоритму стиснення JPEG.

3.3. Вилучимо вбудовані дані зі стисненого (атакованого) контейнера-зображення. Для цього виконаємо перетворення, наведені на рис. 4.15 (за аналогією з перетвореннями, викладеними в п. 2.3)

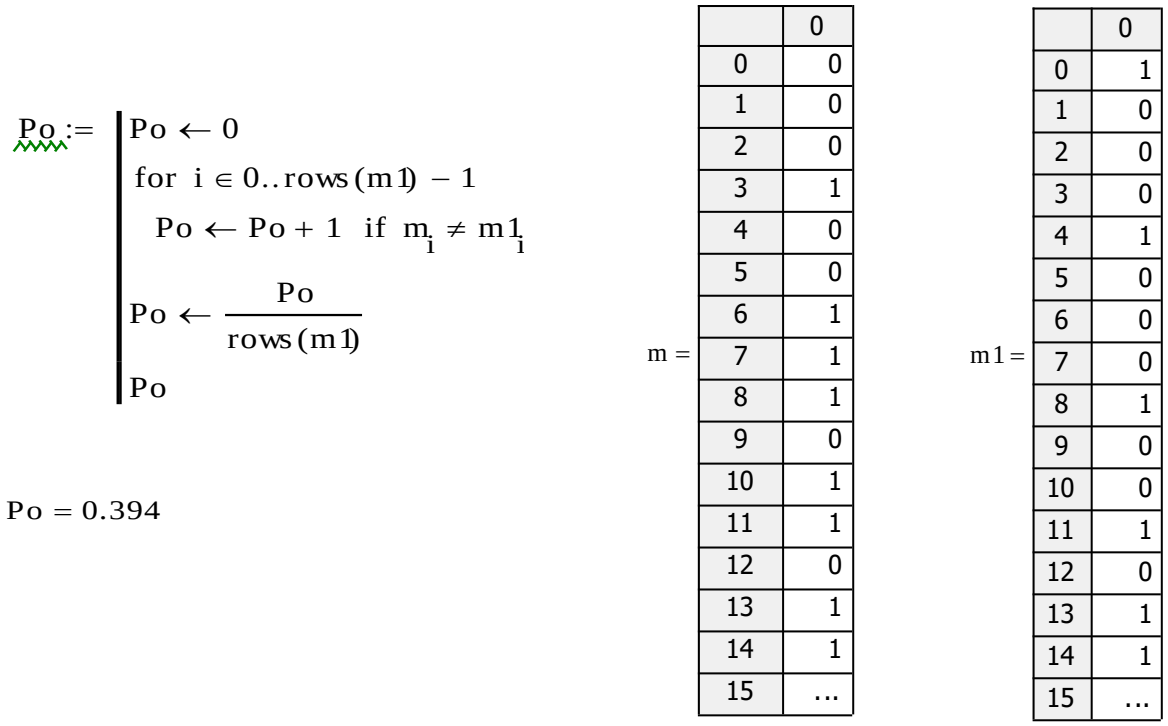
```
R_Stego := READ_RED("Stego.jpg")

r2 := | for x ∈ 0..mm − 1
      |   for y ∈ 0..nn − 1
      |     for i ∈ 0..N − 1
      |       for j ∈ 0..N − 1
      |         RR[i,j] ← R_Stego[i+x·N, j+y·N]
      |     r[x,y] ← RR
      | r

tr2 := | for i ∈ 0..mm − 1
       |   for j ∈ 0..nn − 1
       |     tr[i,j] ← T(r2[i,j])
       | tr

m1 := | num ← 0
      | for x ∈ 0..mm − 1
      |   for y ∈ 0..nn − 1
      |     TRR1 ← tr2[x,y]
      |     m1[num] ← 1 if |TRR1[3,1]| > |TRR1[1,3]|
      |     m1[num] ← 0 if |TRR1[3,1]| ≤ |TRR1[1,3]|
      |     num ← num + 1
      | m1
```

Рисунок 4.15 – Побітове вилучення послідовності інформаційних бітів з частотної області стисненого зображення і порівняння даних

Першою командою зчитується масив яскравостей червоного кольору в змінну «R_Stego». Далі цей масив розбивається на блоки розміром NxN елементів, і над кожним блоком за допомогою функції T(V) виконується пряме дискретно-косинусное перетворення. Потім, як і в п. 2.3, виконується послідовне вилучення інформаційних бітів, результат вилучення записується в масив «m1».

3.4. Для кількісної ймовірності помилкового вилучення інформаційних даних виконаємо такі операції (див. Рис. 4.16). На рисунку наведено також в якості прикладу перші п'ятнадцять бітів вбудованих і вилучених зі стисненого контейнера інформаційних даних.

Як видно з наведених на рис. 4.16 даних стиснення зображення призвело до істотного (близько 40%) спотворення інформаційних бітів. Це наочно підтверджує і наведений на рисунку приклад.



	0
0	0
1	0
2	0
3	1
4	0
5	0
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	…

	0
0	1
1	0
2	0
3	0
4	1
5	0
6	0
7	0
8	1
9	0
10	0
11	1
12	0
13	1
14	1
15	…

Рисунок 4.16 – Емпірична оцінка ймовірності помилкового вилучення інформаційних даних та порівняння з вихідними даними

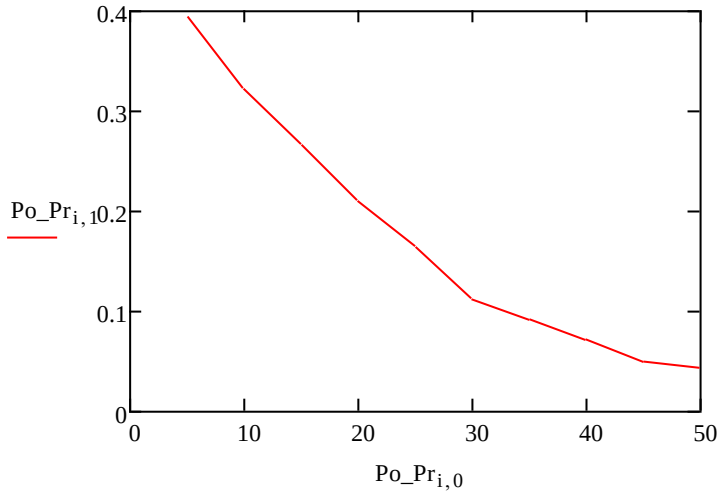
3.5. Зменшимо число виникаючих помилкових даних при вилученні інформаційних повідомлень. Для цього змінимо параметр «Pr» - величину порога зміни частотних коефіцієнтів при вбудовуванні інформаційних бітів (див. п. 2.1). Виберемо значення порога рівним 10 і повторимо всі виконані раніше процедури: вбудовування, збереження зображення у вигляді файлу-зображення, стиснення зображення алгоритмом JPEG (імітація стеганоатаки) і витлучення повідомлення зі стисненого зображення. Емпірична оцінка ймовірності помилкового вилучення інформаційних даних (див. п. 3.4) дає значення 0,323, тобто число помилок зменшилося. Однак збільшення порогу «Pr» неминуче призведе до збільшення внесених спотворень в контейнер-зображення. Емпірична оцінка (див. п. 2.2) підтверджує це, отримане значення 1,273 (в порівнянні з 0,974 при порозі рівному 5). Повторимо відповідні експериментальні дослідження для різних значень порогу «Pr»: 15, 20, 25, 30, 35, 40, 45, 50. Отримані емпіричні оцінки ймовірності помилкового вилучення інформаційних даних та середньої величини внесених спотворень в контейнер-зображення зведемо до відповідних таблиць (см. рис. 4.17).

У таблиці «Po_Pr» наведені отримані експериментальні дані, отримані в результаті емпіричної оцінки ймовірності помилкового вилучення інформаційних даних (другий стовпець) в залежності від величини порога «Pr» (перший стовпець). У таблиці «RAZ_Pr» наведені отримані експериментальні дані, отримані в результаті емпіричної оцінки середньої величини внесених спотворень в контейнер-зображення (другий стовпець) в залежності від величини порога «Pr» (перший стовпець).

На рис. 4.17 приведені також емпіричні залежності у вигляді графіків, побудованих за відповідними табличними значеннями.

$$Po\_Pr := \begin{pmatrix} 5 & 0.394 \\ 10 & 0.323 \\ 15 & 0.266 \\ 20 & 0.21 \\ 25 & 0.165 \\ 30 & 0.113 \\ 35 & 0.092 \\ 40 & 0.073 \\ 45 & 0.051 \\ 50 & 0.045 \end{pmatrix} \qquad RAZ\_Pr := \begin{pmatrix} 5 & 0.974 \\ 10 & 1.273 \\ 15 & 1.692 \\ 20 & 2.134 \\ 25 & 2.601 \\ 30 & 3.121 \\ 35 & 3.568 \\ 40 & 4.012 \\ 45 & 4.560 \\ 50 & 5.069 \end{pmatrix}$$

$i := 0..9$



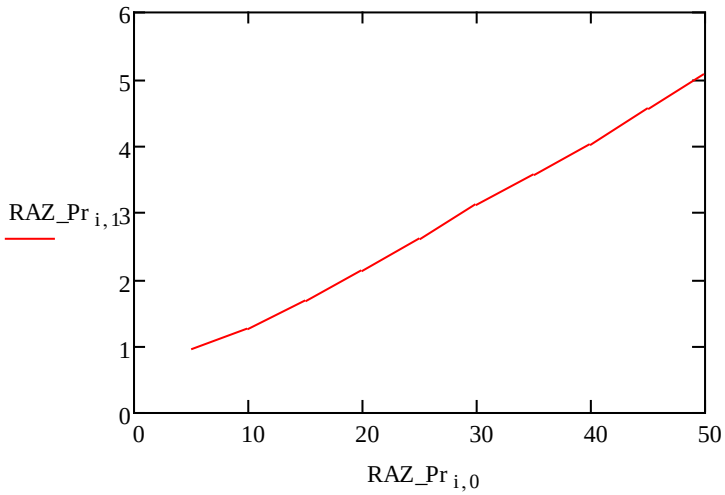


Рисунок 4.17 – Побудова емпіричних залежностей ймовірності помилкового вилучення інформаційних даних та середньої величини внесених спотворень в контейнер-зображення від величини порога «Pr»

Як видно з наведених на рис. 4.17 залежностей зі збільшенням величини порога «Pr» ймовірність помилкового вилучення інформаційних бітів повідомлення різко знижується. Однак це веде до аналогічного підвищення внесених спотворень в контейнер-зображення. Якщо використовувати величину порога чутливості зорової системи людини у 2-3% від максимальної яскравості зображення, тоді внесення спотворень менших за 256*0,02 = 5,12 рівнів яскравості досягається тільки при величині порога «Pr» меншому за 50. Отже, реалізований метод стенографічного вбудовування інформації дозволяє передавати приховані (від візуального виявлення) повідомлення з ймовірністю помилки витягу не меншою 0,05. Зниження помилок в інформаційних даних може бути досягнуто за рахунок використання завадостійкого кодування (див. Лабораторну роботу №2) та / або більш надійних методів.

Завдання 4. Реалізація в середовищі MathCAD вдосконалених алгоритмів вбудовування та вилучення повідомлень в частотну область зображень (метод Бенгама-Мемона-Ео-Юнга)

4.1. Реалізуємо алгоритм вбудовування інформаційних даних в частотну область зображення на основі вдосконаленого правила кодування різниць абсолютних значень коефіцієнтів дискретно-косинусного перетворення (за допомогою методу Бенгама-Мемон-Ео-Юнга). Для цього в середовищі MathCAD виконуємо послідовність перетворень, представлених на рис. 4.18 і 4.19.

$$
\begin{aligned}
&\text{Input}(TRR, m) := \\
&\quad \text{if } m = 1 \\
&\quad\quad TRR_{3,1} \leftarrow \left|TRR_{1,3}\right| + Pr \text{ if } TRR_{3,1} > 0 \\
&\quad\quad TRR_{3,1} \leftarrow -\left|TRR_{1,3}\right| - Pr \text{ if } TRR_{3,1} \le 0 \\
&\quad\quad TRR_{3,2} \leftarrow \left|TRR_{1,3}\right| + Pr \text{ if } TRR_{3,2} > 0 \\
&\quad\quad TRR_{3,2} \leftarrow -\left|TRR_{1,3}\right| - Pr \text{ if } TRR_{3,2} \le 0 \\
&\quad \text{if } m = 0 \\
&\quad\quad TRR_{1,3} \leftarrow \left|TRR_{3,1}\right| + Pr \text{ if } TRR_{1,3} > 0 \\
&\quad\quad TRR_{1,3} \leftarrow -\left|TRR_{3,1}\right| - Pr \text{ if } TRR_{1,3} \le 0 \\
&\quad\quad TRR_{2,3} \leftarrow \left|TRR_{3,1}\right| + Pr \text{ if } TRR_{2,3} > 0 \\
&\quad\quad TRR_{2,3} \leftarrow -\left|TRR_{3,1}\right| - Pr \text{ if } TRR_{2,3} \le 0 \\
&\quad TRR
\end{aligned}
$$

Рисунок 4.18 – Вдосконалене правило кодування різниць абсолютних значень коефіцієнтів дискретно-косинусного перетворення

На рис. 4.18 приведено правило кодування різниць абсолютних значень коефіцієнтів, що відповідає першому удосконаленню відповідно до методу Бенгама-Мемон-Ео-Юнга. Замість двох коефіцієнтів дискретно-косинусного

перетворення (в методі Коха-Жао) використовується три коефіцієнти і, за твердженням авторів методу, це істотно покращує експлуатаційні характеристики стеганографічної захисту []. Друге вдосконалення, засноване на відбракуванні блоків, пропонується реалізувати самостійно.

На рис. 4.19 приведено опис процедур вбудовування даних в контейнер-зображення за допомогою вдосконаленої процедури кодування різниць абсолютних значень коефіцієнтів.

```
tr := | for i ∈ 0..mm − 1
      |   for j ∈ 0..nn − 1
      |     tr_{i,j} ← T(r_{i,j})
      | tr
```

```
tr := | num ← 0
      | for x ∈ 0..mm − 1
      |   for y ∈ 0..nn − 1
      |     | break if x·nn + y ≥ rows(m) − 1
      |     | tr_{x,y} ← Input(tr_{x,y}, m_{num})
      |     | num ← num + 1
      | tr
```

```
vr := | for i ∈ 0..mm − 1
      |   for j ∈ 0..nn − 1
      |     vr_{i,j} ← V(tr_{i,j})
      | vr
```

```
R2 := | for x ∈ 0..rows(vr) − 1
      |   for y ∈ 0..cols(vr) − 1
      |     | temp1 ← vr_{x,y}
      |     | for i ∈ 0..N − 1
      |     |   for j ∈ 0..N − 1
      |     |     temp2_{x·N+i, y·N+j} ← temp1_{i,j}
      | temp2
```

Рисунок 4.19 – Вбудовування даних в контейнер-зображення за допомогою вдосконаленої процедури кодування різниць абсолютних значень коефіцієнтів дискретно-косинусного перетворення

Перетворення, опис яких наведено на рис. 4.19, аналогічні тим, які розглянуті на рис. 4.8. За аналогією з рис. 4.9 на рис. 4.20 приведена остаточна обробка масиву растрових даних і виведення отриманого зображення.

4.2. Для кількісної оцінки внесених спотворень в контейнер-зображення обчислимо середнє арифметичне поелементної різниці масивів R (до вбудовування) і R2 (після вбудовування). Отримані результати наведені на рис. 4.21.

```
R2 := | for x ∈ 0..rows(vr) − 1
      |   for y ∈ 0..cols(vr) − 1
      |     | temp1 ← vr_{x,y}
      |     | for i ∈ 0..N − 1
      |     |   for j ∈ 0..N − 1
      |     |     temp2_{x·N+i, y·N+j} ← temp1_{i,j}
      | temp2
```

R

R2

Рисунок 4.20 – Обробка масиву растрових даних та виведення отриманого зображення

```
RAZ := | RAZ ← 0
       | for i ∈ 0..rows(R2) − 1
       |   for j ∈ 0..cols(R2) − 1
       |     RAZ ← RAZ + |R_{i,j} − R2_{i,j}|
       | RAZ ← RAZ / (rows(R)·cols(R))
       | RAZ
```

RAZ = 1.815

Рисунок 4.21 – Кількісна оцінка різниць між зображеннями до та після вбудовування інформаційного повідомлення

Зрозуміло, що величина внесених спотворень в удосконаленому методі в порівнянні з методом Коха-Жао істотно (практично в два рази) зросла. Це пояснюється відсутністю процедури відбраковування та вбудовуванням даних в 3 (замість двох) коефіцієнти дискретно-косинусного перетворення. Однак цей окремий випадок є неінформативним. Необхідно оцінити також ймовірність помилкового вилучення інформаційних даних, а також дослідити відповідні залежності для різних значень порога «Pr».

4.3. Реалізуємо алгоритм вилучення інформаційних даних з частотної області зображення. Для цього в середовищі MathCAD виконаємо послідовність перетворень, представлених на рис. 4.22 (за аналогією з перетвореннями, представленими на рис. 4.11).

```
tr1 := | for i ∈ 0..mm − 1
       |   for j ∈ 0..nn − 1
       |     tr_{i,j} ← T(vr_{i,j})
       | tr

m1 := | num ← 0
      | for x ∈ 0..mm − 1
      |   for y ∈ 0..nn − 1
      |     | TRR1 ← tr1_{x,y}
      |     | m1_{num} ← 1  if |TRR1_{3,1}| > |TRR1_{1,3}| ∧ |TRR1_{3,2}| > |TRR1_{1,3}|
      |     | m1_{num} ← 0  if |TRR1_{3,1}| ≤ |TRR1_{1,3}| ∧ |TRR1_{3,2}| ≤ |TRR1_{1,3}|
      |     | num ← num + 1
      | m1
```

Рисунок 4.22 – Побітове вилучення послідовності інформаційних бітів з частотної області зображення і порівняння даних

Після виконання дискретно-косинусного перетворення всіх блоків контейнера (процедура «tr1») проводиться обчислення інформаційних бітів (масив «m1») за допомогою вилучення з середньочастотної області масивів коефіцієнтів дискретно-косинусного перетворення. Для цього в кожному блоці порівнюються абсолютні значення коефіцієнтів з номерами (3,1), (1,3) і (3,2). Якщо абсолютне значення коефіцієнта з номером (3,1) більше абсолютного значення коефіцієнта з номером (1,3) і, одночасно, коефіцієнта з номером (3,2) - тоді детектується одиничний інформаційний біт. Якщо абсолютне значення коефіцієнта з номером (3,1) менше або дорівнює абсолютному значенню коефіцієнта з номером (1,3) і, одночасно, коефіцієнту з номером (3,2) - тоді детектується нульовий інформаційний біт.

Для кількісної ймовірності помилкового вилучення інформаційних даних виконаємо операції, наведені на рис. 4.23 (за аналогією з рис. 4.12).

Витягнуті з частотної області контейнера-зображення інформаційні біти повністю співпали (див. Рис. 4.23) з вихідними даними (як і в методі прототипі), що пояснюється відсутністю внесених спотворень в контейнер-зображення.

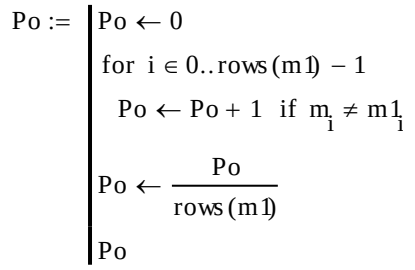


Рисунок 4.23 – Оцінка ймовірності помилкового вилучення інформаційних даних

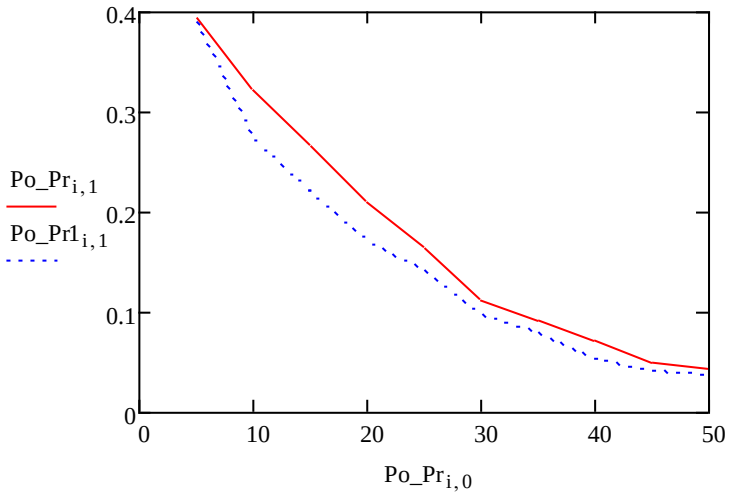
4.4. Проведемо емпіричні дослідження ефективності вдосконаленого методу стеганографічного перетворення (за аналогією з розглянутим вище завданням 3). Експериментальні дослідження часток внесених спотворень в контейнер-зображення і числа виникаючих помилок при вбудовуванні інформаційних даних проведемо для різних значень порога «Pr»: 15, 20, 25, 30, 35, 40, 45, 50 (як і при виконанні завдання 3). Отримані емпіричні оцінки ймовірності помилкового вилучення інформаційних даних та середньої величини внесених спотворень в контейнер-зображення зведемо до відповідних таблиць, які наведені на рис. 4.24 (за аналогією з рис. 4.17).

Таблиці «Po_Pr1» і «RAZ_Pr1» характеризують величину помилок у витягнутих даних і рівень внесених спотворень в контейнер-зображення. Вони заповнені таким самим способом, як і відповідні таблиці «Po_Pr» і «RAZ_Pr» на рис. 4.17.

На рис. 4.24 приведені також емпіричні залежності у вигляді графіків, побудованих за відповідними табличними значеннями. На графіках суцільною лінією наведені емпіричні залежності, що характеризують ефективність методу Коха-Жао, переривчастою лінією - дані для вдосконаленого методу. Як видно з наведених залежностей, вдосконалений метод дійсно дозволяє знизити число виникаючих помилок при вбудовуванні інформаційних даних (перший графік). Однак його використання пов'язане також і зі збільшенням внесених спотворень в контейнер-зображення (другий графік). Якщо зафіксувати рівень внесених спотворень у 2-3% від максимальної яскравості зображення (256*0,02 = 5,12), тоді величина порога «Pr» не повинна перевищувати 40 (див. другий графік). Це приблизно відповідає ймовірності помилки вилучення близько 0,05. Тобто з точки зору величини внесених спотворень і помилок, що виникають при вбудовуванні інформаційних даних, метод Коха-Жао і вдосконалений метод (метод Бенгама-Мемон-Ео-Юнга) можна порівняти за ефективністю. Це пояснюється відсутністю процедури відбракування блоків в удосконаленому методі (дану процедуру пропонується реалізувати самостійно).

$$Po\_Pr1 := \begin{pmatrix} 5 & 0.391 \\ 10 & 0.275 \\ 15 & 0.221 \\ 20 & 0.173 \\ 25 & 0.142 \\ 30 & 0.099 \\ 35 & 0.081 \\ 40 & 0.055 \\ 45 & 0.042 \\ 50 & 0.038 \end{pmatrix} \qquad RAZ\_Pr1 := \begin{pmatrix} 5 & 1.815 \\ 10 & 2.273 \\ 15 & 2.692 \\ 20 & 3.142 \\ 25 & 3.351 \\ 30 & 3.721 \\ 35 & 4.298 \\ 40 & 5.431 \\ 45 & 6.169 \\ 50 & 6.801 \end{pmatrix}$$

$$i := 0..9$$



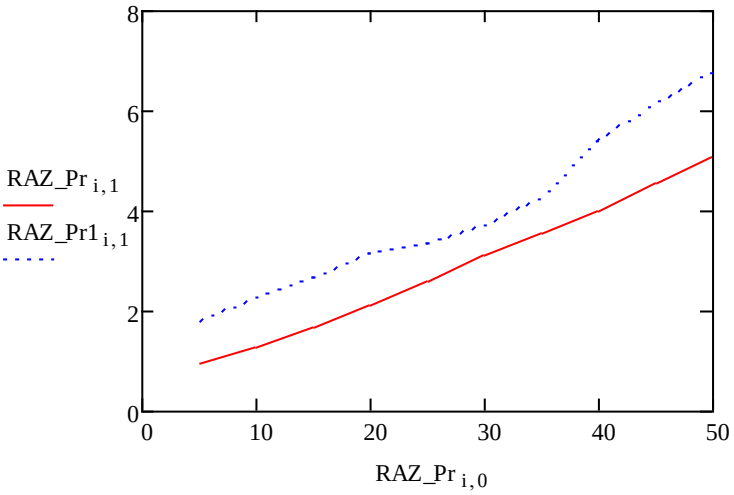


Рисунок 4.24 – Побудова емпіричних залежностей ймовірності помилкового вилучення інформаційних даних та середньої величини внесених спотворень в контейнер-зображення від величини порога «Pr»

Додаткове завдання. Реалізація в середовищі MathCAD алгоритмів вбудовування та вилучення повідомлень в частотну область зображень методом Д.Фридрих. (пропонується до самостійного виконання)

Навчальне видання

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ
до лабораторних робіт з дисципліни
“Стеганографія”

для студентів напрямку
6.170101 “Безпека інформаційних і комунікаційних систем”,
спеціальності
125 “Кібербезпека”

Упорядник: КУЗНЕЦОВ Олександр Олександрович

Відповідальний випусковий С.Г. Рассомахін

Редактор

План 2016, поз.

Підп. до друку	Формат 60×84 1/16.	Спосіб друку – ризографія.
Умов.друк.арк.	Облік. вид.арк.	Тираж прим.
Зам. №	Ціна договірна.	

ХНУ ім. Каразіна. Україна. 61022, Харків, площа Свободи 4

Віддруковано в навчально-науковому
видавничо-поліграфічному центрі ХНУ ім. Каразіна
61022, Харків, площа Свободи 4