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
1701 “Інформаційна безпека”

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 2015

Методичні рекомендації до лабораторних робіт з дисципліни “Стеганографія” для студентів напрямку 6.170101 “Безпека інформаційних і комунікаційних систем” спеціальності 1701 “Інформаційна безпека” / Упоряд. Кузнецов О.О. – Харків: ХНУ ім. В.Н. Каразіна, 2015. – ХХХ с.

Упорядники: О.О. Кузнецов

Рецензенти: О.В.Лемешко, д.т.н., доц. професор каф. 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 бланк звіту 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: $\nu = 1/L_п$. 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

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

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…0,004 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:

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

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).

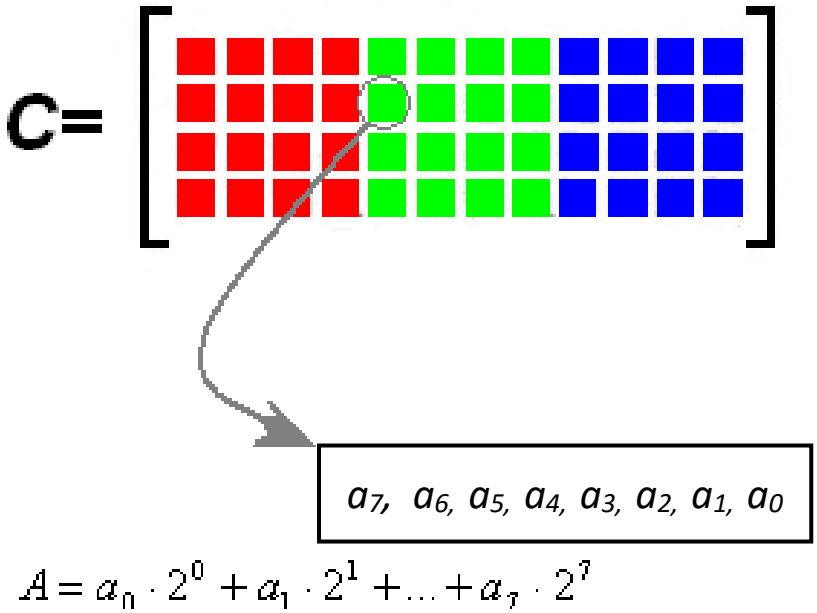


Рис. 4.1. Уявлення компонент кольоровості *R*, *G* і *B* у вигляді підмасивів масиву C

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).

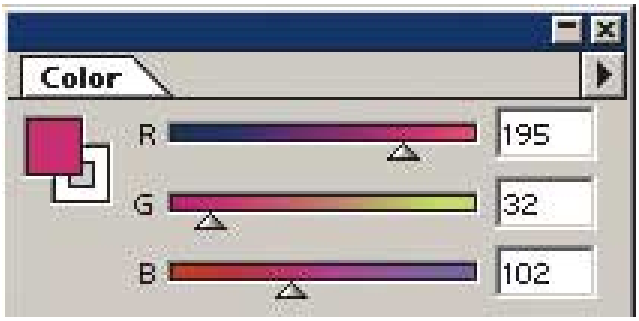


Рис. 1.2. Приклад кодування растрових даних зображення

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).

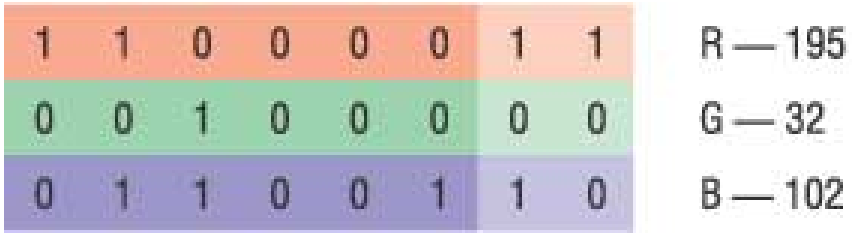


Рис. 1.3. Приклад опису трьох байтів бітовому вигляді

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 перешкод.

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. 1.7).

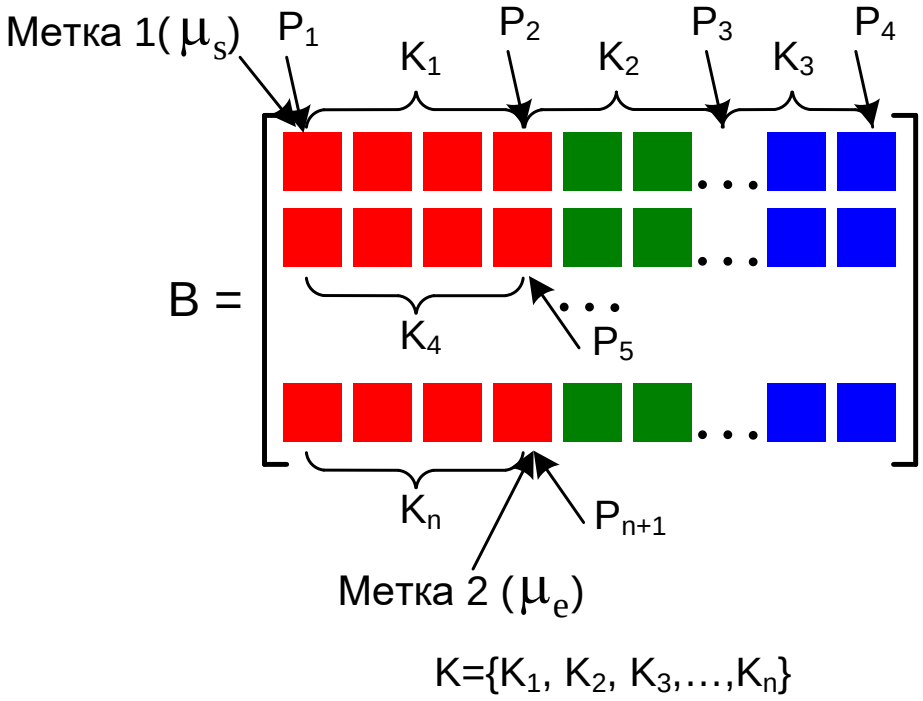


Рис. 1.7. Приклад використання підмасиву B синього колірного компоненту зображення у якості контейнеру

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.

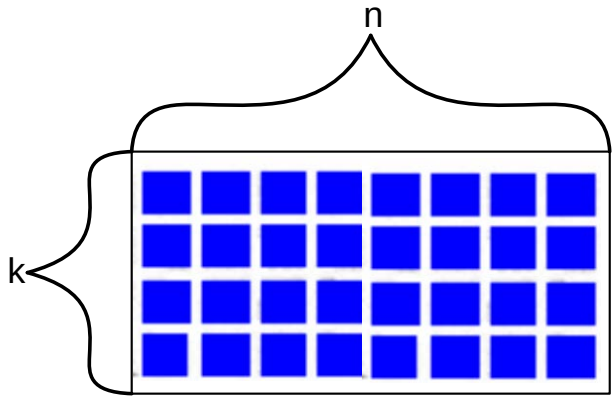
Результуюче кольорове зображення визначатиметься масивом об'єднання колірних масивів R, G, B^*.

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. 4.8.

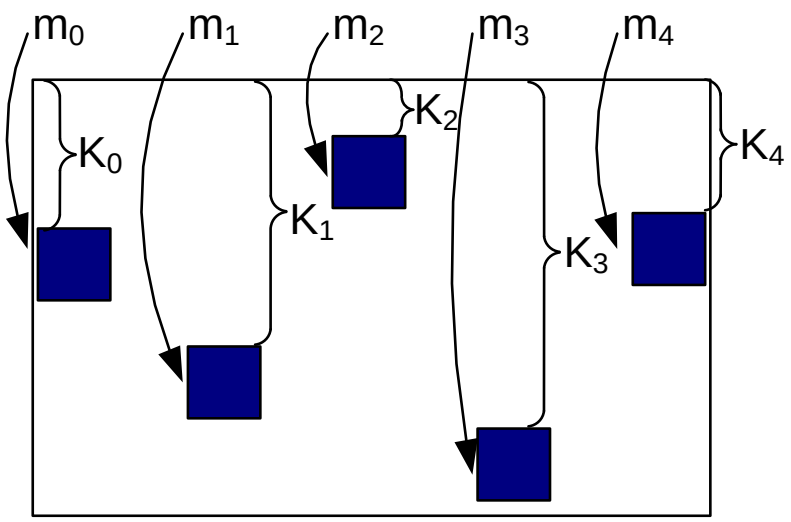


Рис. 1.8. Схема вбудовування зображення

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. 4.1.)

Таблиця 4.1.

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 повідомленя 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:

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

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:

$$m = \{1\,0\,1\,0\,1\,1\,0\,1\,0\,1\,1\,0\,1\,1\,0\}$$

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

Маємо таке:

$$M_0 = \{1\,0\,1\,0\,1\} \qquad M^*_0 = M_0 \cdot P = \{0\,1\,0\,1\,1\}$$
$$M_1 = \{1\,0\,1\,0\,1\} \qquad M^*_1 = M_1 \cdot P = \{0\,1\,0\,1\,1\}$$
$$M_2 = \{1\,0\,1\,0\,1\} \qquad M^*_2 = M_2 \cdot P = \{0\,1\,1\,0\,1\}$$

Отже $m^* = \{0\,1\,0\,1\,1\,0\,1\,0\,1\,1\,0\,1\,1\,0\,1\}$

Для зворотного перетворення виконаємо наступне:

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

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

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

Отже маємо $m = \{1\,0\,1\,0\,1\,1\,0\,1\,0\,1\,1\,0\,1\,1\,0\}.$

Розглянемо тепер використання переставного шифру в методі псевдовипадкового переставлення. Загальна схема вбудовування бітів даних зображена на рис. 4.8.

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).

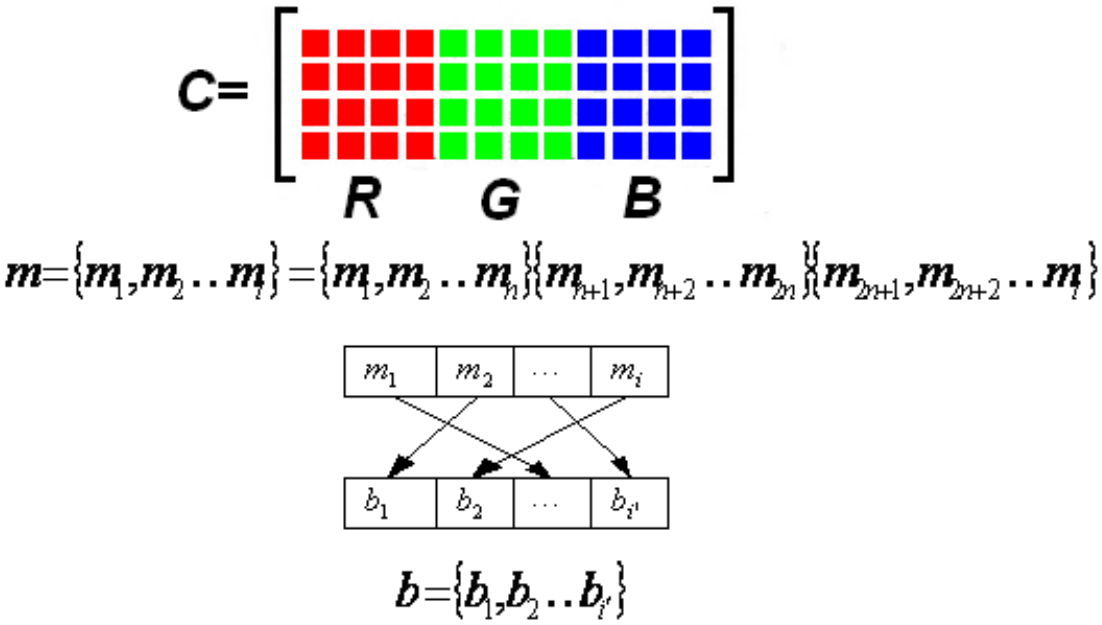


Рис. 4.8. Приклад здійснення псевдовипадкового переставлення

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. Вопросы для поточного контроля подготовленности студентов к выполнению лабораторной работы №1

1. Математическая модель и структурная схема криптографической (секретной) и стеганографической системы защиты информации.
2. Основные области практического использования стеганографических методов защиты информации. Исторические примеры использования стеганографических систем защиты информации.
3. Классификация стеганографических систем. Закрытые, полузакрытые, открытые стеганографические системы. Хрупкие, полухрупкие, робастные стеганографические системы.
4. Классификация и виды контейнеров по различным признакам. Примеры использования различных контейнеров для организации скрытной передачи информации.
5. Низкоуровневые свойства зрительной системы человека. Практические примеры использования низкоуровневых свойств зрительной системы человека в стеганографии.
6. Высокоуровневые свойства зрительной системы человека. Практические примеры использования высокоуровневых свойств зрительной системы человека в стеганографии.
7. Современные форматы неподвижных изображений. Структура файла неподвижного изображения в формате bmp24. Растровые данные изображения.
8. Метод встраивания информации в неподвижные изображения на основе модификации наименее значимого бита (метод LSB). Достоинства и недостатки.
9. Простейшие симметричные шифры. Простой перестановочный шифр.
10. Метод встраивания информации с использованием псевдослучайной перестановки (метод ПСП). Достоинства и недостатки.
11. Метод встраивания информации с использованием псевдослучайного интервала (метод ПСИ). Достоинства и недостатки.

5. Руководство к выполнению лабораторной работы №1

Задание 1. Реализация алгоритмов встраивания и извлечения сообщений в пространственной области неподвижных изображений методом LSB

1.1. Загружаем исходные данные: контейнер - неподвижное изображение (в формате *.bmp24); информационное сообщение – текстовый документ (в формате *.txt). Для этого в среде MathCAD выполняем следующие действия.

1.1.1. Выполняем команду чтения растровых данных неподвижного изображения из заданного файла в виде двумерного массива целых чисел:

«C:=READRGB(“[имя файла].bmp”)».

Элементы массива $C_{i,j}$ лежат в интервале [0...255] и задают значения яркости одного из трех цветов (красного, зеленого или синего) изображения. Координаты элементов массива задают месторасположение отдельных пикселей изображения. Массив C состоит из трех подмассивов равного размера (для хранения значений яркостей соответствующих цветов).

Например, пусть в качестве контейнера выступает неподвижное изображение, которое хранится в файле с именем «1.bmp». Тогда, после выполнения команды чтения растровых данных

«C:=READRGB(“1.bmp”)»

имеем:

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 графического отображения загруженных данных выполняем действия: «Вставить», «Изображение». В поле ввода источника изображений вносим имя файла или имя переменной, в которой храниться массив данных.

После выполнения команд графического отображения контейнера для рассматриваемого примера имеем следующие изображения:

"1.bmp"

C

Графическое отображение массива растровых данных C состоит из трех фрагментов изображения, каждый фрагмент соответствует отображению одного из цветовых каналов (красного, зеленого или синего) в градациях серого цвета.

Выполняем команду чтения данных из канала красного цвета растровых данных неподвижного изображения из заданного файла в виде массива целых чисел:

«R:=READ_RED(“[имя файла].bmp”)».

Элементы массива $R_{i,j}$ также лежат в интервале [0...255] и задают значения яркости красного цвета. Для графического отображения загруженных данных выполняем действия: «Вставить», «Изображение». В поле ввода источника изображений вносим имя переменной R, в которой храниться массив данных канала красного цвета.

Для рассматриваемого примера выполняем команду

«R:=READ_RED(“1.bmp”)»,

после чего получим:

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

Выполняем аналогичные команды чтения данных из каналов зеленого и синего цвета растровых данных неподвижного изображения в виде массивов целых чисел:

«G:=READ_ GREEN (“[имя файла].bmp”)»,
«G:=READ_BLUE(“[имя файла].bmp”)».

Для рассматриваемого примера выполняем команды

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

после чего получим:

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	…

Загруженные массивы красного, зеленого и синего цвета (массивы R, G, и B) являются подмассивами массива растровых данных C. Аналогично, графическая интерпретация массивов R, G, и B (в виде изображений в градациях серого цвета) объединена в графическом представлении массива C. Полноцветное представление изображения получим следующим образом:

R, G, B

При нарушении порядка следования массивов растровых данных, характеризующих информацию о каналах цветности, изображение будет искажено, например:

G, B, R

B, R, G

1.1.2. Выполняем команду чтения информационных данных текстового документа из заданного файла в виде одномерного массива целых чисел:

«M:=READBIN(“[имя файла].txt”, byte)».

Элементы массива M_i лежат в интервале [0...255] и задают значения символов информационного сообщения в кодировке ASCII (кодировка ASCII приведена в приложении). Координаты элементов массива M задают месторасположение отдельных символов информационного сообщения.

Например, пусть в качестве информационных данных выступает текстовый документ, который хранится в файле с именем «1.txt». Тогда, после выполнения команды чтения символов информационного сообщения в кодировке ASCII

«M:=READBIN(“1.txt”, byte)»

имеем:

M =

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

Значение нулевого элемента массива M равно M_0=200, что соответствует символу «И» в кодировке ASCII. Следующий элемент массива M_1=237 соответствует символу «н» в кодировке ASCII. Набор первых семи элементов массива информационных данных

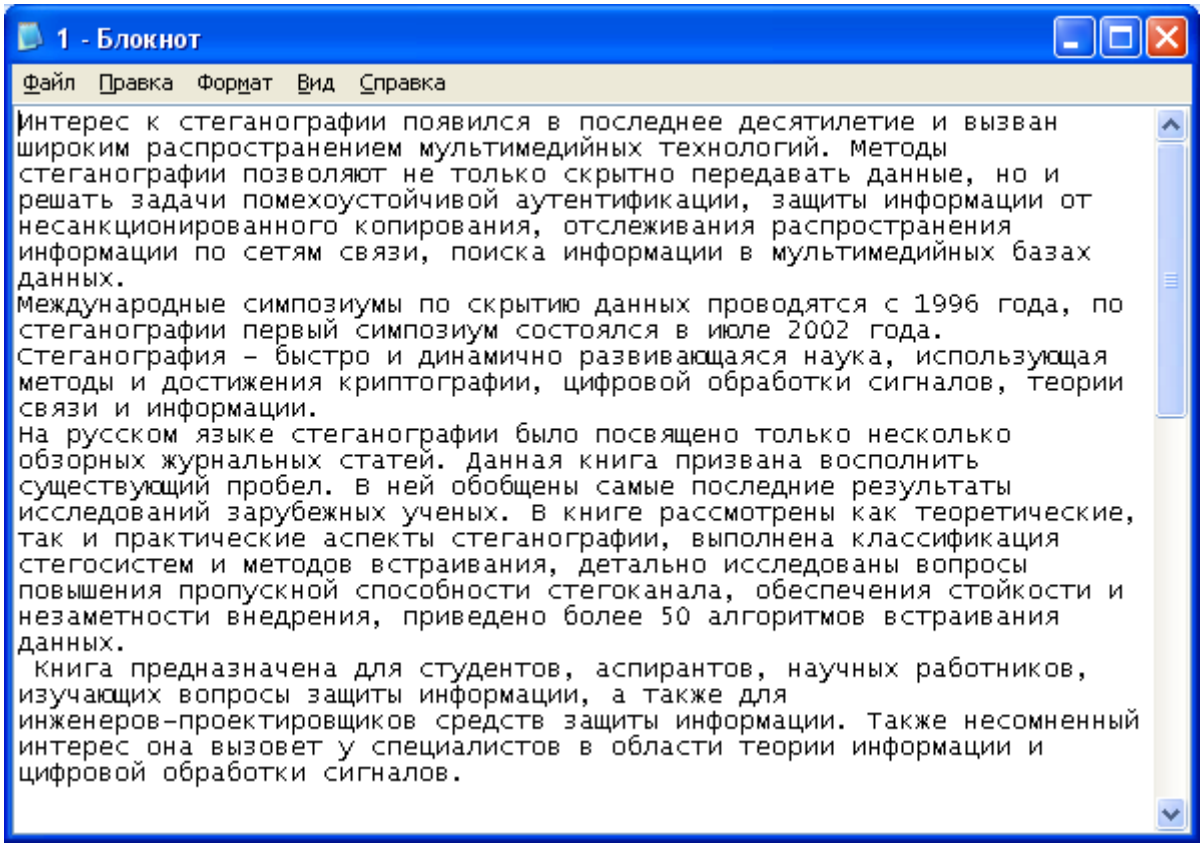
{200, 237, 242, 229, 240, 229, 241}

соответствует набору символов сообщения

{И, н, т, е, р, е, с}

в кодировке ASCII.

Для рассматриваемого примера в качестве информационного сообщения выбран первый абзац из аннотации книги «Цифровая стеганография» авторов В. Г. Грибунина, И. Н. Окова, И. В. Туринцева.



1.2. Преобразуем массив информационных данных

Информационные данные в массиве M представлены в виде набора целых чисел из интервала [0...255] и задают значения символов информационного сообщения в кодировке ASCII. Для рассматриваемых стеганографических методов встраивание информации в неподвижные

изображения осуществляется побитно. Т.е. информационные данные из массива M следует предварительно подготовить, преобразовав их в битовый массив. Для этого в среде MathCAD используем следующие функции.

1.2.1. Функция преобразования вектора-столбца из восьми бит в десятичный код:

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

Аргументом x функции B_D(x) является двоичный вектор-столбец из восьми бит:

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

Значением функции B_D(x) является целое число в десятичном коде:

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

Например, пусть аргумент x функции B_D(x) задан следующим образом:

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

Тогда значение функции B_D(x) равно

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

1.2.2. Функция преобразования целого числа в десятичном коде в двоичный вектор-столбец из восьми бит:

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

Аргументом x функции D_B(x) является целое число в десятичном коде, значением функции является двоичный вектор-столбец из восьми бит.

Алгоритм вычисления значения функции D_B(x) следующий. На каждой итерации (для каждого значения цикловой переменной i) вычисляются следующие значения:

$V_i \leftarrow \text{mod}(x, 2)$ - приведение десятичного числа x по модулю 2;

$x \leftarrow \text{floor}\left(\frac{x}{2} \right)$ - взятие целой части от деления целого числа x на два.

Т.е. после каждой итерации число x уменьшается в два раза, а значение i-го бита возвращаемого функцией двоичного вектора-столбца приравнивается результату приведения текущего значения x по модулю два.

Например, пусть аргумент функции D_B(x) равен 27. Тогда значение функции D_B(x) вычисляется следующим образом:

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 образом, вычисленное значение функции D_B(x) равно

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

1.2.3. Используя функцию D_B(x) преобразуем массив M информационных целых чисел в битовый массив.

Для этого воспользуемся следующей процедурой:

```
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 формирования битового массива информационных данных следующий. Для каждого значения цикловой переменной i (значение i пробегает по всем индексам массива M) выполняется преобразование $D\_B(M_i)$ по рассмотренному выше правилу. Т.е. для всех элементов массива M формируются соответствующие битовые векторы-столбцы. Все элементы каждого вектора столбца перезаписываются в массив M_b под соответствующим индексом. Таким образом, каждый сформированный бит записывается в элемент M_b_{i8+j} массива M_b, где j – цикловая переменная, которая пробегает все индексы текущего (i-ого) вектора столбца.

Для примера рассмотрим исходный вектор-столбец M, содержащий целые числа (коды) информационного сообщения. Пусть M – массив целых чисел из предыдущего примера.

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	…

Выполнение процедуры преобразования массива M в битовый массив M_b происходит следующим образом:

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. Реализуем алгоритм встраивания данных в пространственную область изображений методом LSB. Для этого воспользуемся следующей процедурой:

```
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
```

Приведенная процедура реализует поэлементное встраивание битового массива информационных данных M_b в наименее значащие биты подряд следующих байт массива растровых данных канала красного цвета, т.е. встраивание осуществляется только в LSB массива R. Первые три строки процедуры выполняют перезапись данных из массива растровых данных канала красного цвета в новый массив S. Далее, для всех элементов битового

массива информационных данных M_b вычисляются координаты (номера столбцов и строк) элементов массива R, в которые они будут встроены. Так, разделив значение индекса l на количество строк в массиве R, получим номер (индекс i) того столбца, в элементы которого будет встроен l-ый бит сообщения. Выполнив вычисления l – j*rows(R) получим номер (индекс j) той строки, в элементы которой будет встроен l-ый бит сообщения. Битовое представление десятичного числа $R_{j,i}$ записывается в переменную V (это преобразование реализуется с помощью рассмотренной выше функции). Текущий бит сообщения M_b_l заносится в нулевой (наименее значимый) бит массива V_0, после чего выполняем обратное преобразование массива V с измененным LSB. Полученное десятичное число записываем в массив S с теми же индексами i и j. Таким образом, все биты информационного сообщения из массива M_b записываются в наименее значимые биты байт канала красного цвета изображения. Если размер сообщения не велик, то оставшиеся не использованными элементы массив S записываются исходные данные из массива R.

Для визуального просмотра результата встраивания информационных данных выведем исходный массив растровых данных красного цвета R и полученный массив с измененными наименее значимыми битами. Для рассматриваемого примера имеем:

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	...

Из представленных данных видно, что, например, значения $R_{0,0}$, $R_{1,0}$, $R_{2,2}$ и др. полностью идентичны соответствующим значениям $S_{0,0}$, $S_{1,0}$, $S_{2,2}$. Практически это означает, что наименее значимых бит в этих элементах массива R совпали со встраиваемыми информационными битами сообщения. Напротив, значения $R_{0,1}$, $R_{1,1}$, $R_{2,1}$ отличаются на единицу от соответствующих значений массива S. Это означает изменение наименее

значимого бита элемента контейнера в процессе встраивания сообщения. Отметим, что внесенные искажения лежат ниже порога зрительной чувствительности человека.

Графическая интерпретация пустого и заполненного контейнера (канала красного цвета в градациях серого) приведена на следующем рисунке, из которого следует, что визуально внесенные искажения не заметны, что подтверждает вывод о чувствительности органов зрения человека.

R

S

Полученный заполненный массив S записываем в канал красного цвета контейнера. Для выполнения это операции с использованием команды

«WRITERGB([имя файла].bmp):=augment(S,G,B)»

под соответствующим именем записываем на физический носитель сформированный контейнер с измененными LSB в канале красного цвета. Для рассматриваемого примера выполняем команду

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

выполнение которой формирует на физическом носителе новый файл с именем «Stego.bmp».

Для графического отображения исходного (пустого) и заполненного контейнера выполним вставку соответствующих изображений:

"1.bmp"

"Stego.bmp"

Убеждаемся в отсутствии видимых искажений.

1.4. Реализуем алгоритм извлечения данных из пространственной области изображений методом LSB. Для этого в новом окне среды MathCAD выполняем команды чтения растровых данных неподвижного изображения из заданного файла (файла заполненного контейнера) в виде двумерного массива целых чисел. Для рассматриваемого примера выполняем команды:

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

Получим следующий 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

Далее, воспользовавшись рассмотренной функцией D_B(x) извлекаем наименее значимые биты из массива данных канала красного цвета. Для этого используем следующую 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 выполняет для всех элементов массива R формирование двоичного кода десятичного числа и записывает его в переменную V. Нулевой (наименее значимый) бит массива V заносится в соответствующий элемент массива M_b1. Индекс элементов массива M_b1 изменяется в зависимости от номеров строк и столбцов обрабатываемого элемента массива R. В результате имеем линейный битовый массив M_b1, заполненный наименее значимыми битами массива растровых данных канала красного цвета заполненного контейнера.

1.5. Преобразуем массив информационных данных.

Для формирования текстового сообщения по извлеченным битам сформируем массив M1. Для этого воспользуемся следующей процедурой:

```
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 формирования массива M1 следующий. Для всех элементов массива M_b1 вычисляем значение индекса l – текущего номера десятичного числа (кода информационного символа). Для этого берем целую часть от деления i (индекса бита в массиве M_b1) на восемь (число бит в одном информационном символе в кодировке ASCII). Далее, все биты текущего символа записываем в служебную переменную V, после чего с использованием функции B_D(x) поочередно вычисляем коды информационных символов. Полученные целые числа (коды символов в кодировке ASCII) записываем в массив 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	...

Выполнение процедуры преобразования битового массива M_b1 в массив десятичных чисел M1 происходит следующим образом:

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. Полученный массив целых чисел записываем на физический носитель в виде текстового файла. Для этого воспользуемся командой:

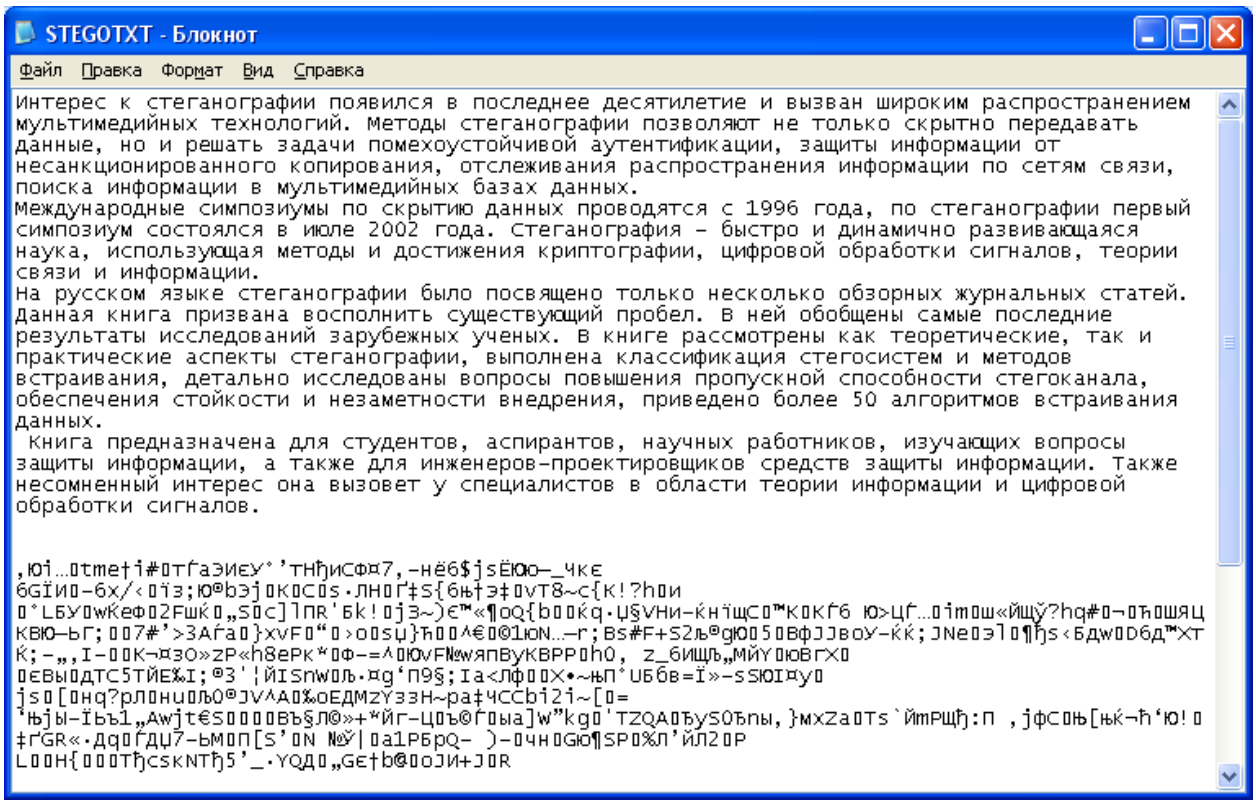
«WRITEBIN(“[имя файла].txt”, byte,1):=M1».

Для рассматриваемого примера выполним команду:

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

в результате которой на физическом носителе будет записан текстовый файл под именем “STEGOTXT.txt”.

Следует отметить, что приведенная выше процедура извлекает все наименее значимые биты контейнера (из канала красного цвета), т.е. извлекаются все LSB даже из тех байт, в которые не выполнялось встраивание информации. После формирования сообщения в конце текстового файла “STEGOTXT.txt” могут присутствовать символы, полученные в результате извлечения LSB не модифицированных в процессе встраивания информации, т.е. т.н. «случайные» символы.



Последний рисунок наглядно демонстрирует правильность работы рассмотренных выше процедур и функций. Информационное сообщение, встроенное в пространственную область неподвижного изображения методом LSB, извлечено правильно.

Задание 2. Экспериментальные исследования зрительного порога чувствительности человека к изменению яркости изображений

2.1. Внесем изменения в реализацию алгоритма встраивания данных в неподвижные изображения методом LSB. Для этого изменим порядковый номер бита, используемый для встраивания информации, в двоичном представлении элементов контейнера (отдельных байт яркости конкретных пикселей изображения).

```
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]))
          V0 ← 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]))
          V1 ← 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]))
          V2 ← M_b[l]
          S[j,i] ← B_D(V)
      S
```

Первая приведенная процедура реализует встраивание информационных данных в наименее значимые (нулевые) биты контейнера (в биты V_0). Вторая и третья процедуры реализуют встраивание информационных данных в следующие по значимости биты контейнера (в биты V_1 и V_2). Приведем пример графического изображения контейнера для каждого из рассматриваемых случаев.

S

S

S

Очевидно, что видимых искажений при встраивании информационных сообщений в нулевой, первый или во второй по значимости бит обнаружить не удается. Это объясняется тем, что максимальные искажения, вносимые в отдельные пиксели изображения посредством изменения уровня их яркости, для каждого из рассматриваемых случаев не превышают величин $2^0=1$, $2^1=2$, $2^2=4$ соответственно, что лежит ниже порога чувствительности зрительной системы человека к незначительному изменению яркости изображения.

Продолжим изменять порядковый номер бита, используемый для встраивания информации, в двоичном представлении элементов контейнера, до тех пор, пока не визуально не станут видны вносимые искажения яркости отдельных пикселей изображения.

```
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
```

Приведенные процедуры реализуют встраивание информационных данных в следующие по значимости биты контейнера (в биты V_3, V_4 и V_5).

2.2. Экспериментально установим, модификация каких бит изображения не приводит к заметным искажениям. Для этого приведем пример графического изображения контейнера для каждого из рассматриваемых случаев (встраивание осуществлялось в биты V_3, V_4 и V_5).

S

S

S

Из приведенных изображений следует, что встраивание данных в третьи по значимости биты контейнеры приводит к едва заметным искажениям (яркость соответствующих пикселей изменилась на 2^3=8 уровней). Модификация четвертых и пятых по значимости бит приводит к значительным искажениям изображения (яркость соответствующих пикселей изменилась на 2^4=16 и 2^5=32 уровней, соответственно). Следовательно, из результатов экспериментальных исследований следует, что зрительная система человека для рассматриваемого примера изображения чувствительна к изменению третьего, четвертого и т.д. битов (по их значимости) контейнера (отдельных байт яркости конкретных пикселей изображения).

2.3. Рассчитаем порог зрительной чувствительности системы человека к незначительному изменению яркости изображения.

Используем экспериментальные данные для расчета порога зрительной чувствительности системы человека к незначительному изменению яркости изображения. Обозначим символом Δ величину вносимых искажений яркости (как число уровней квантования) отдельных пикселей изображения при использовании стеганографического алгоритма встраивания информации в неподвижные изображения на основе модификации отдельных бит контейнера. По спецификации формата изображений *bmp24 общее число уровней квантования яркости отдельных пикселей равно 2^8=256. Тогда зрительный порог чувствительности (ПЧ) системы человека к незначительному изменению яркости изображения определим как

$$ПЧ=(\Delta/256)*100\%.$$

Для рассматриваемого примера видимые искажения были обнаружены после модификации третьих по значимости битов контейнера, соответствующая величина Δ=8. Следовательно, порог зрительной чувствительности системы человека, вычисленный по эмпирическим данным, составляет:

$$ПЧ=(\Delta/256)*100\%=(8/256)*100\%=3{,}125\%,$$

что согласуется с известными теоретическими данными.

Задание 3. Реализация алгоритмов встраивания и извлечения сообщений методом псевдослучайной перестановки

3.1. Загружаем исходные данные (см. п. 1.1.).

3.2. Преобразуем массив информационных данных (см. п. 1.2.).

3.3. Вводим секретное правило псевдослучайной перестановки

Встраиваемая информация предварительно обрабатывается простым перестановочным шифром. Зададим секретный ключ - правило псевдослучайной перестановки в виде перестановочной матрицы размером nxn, где n-размер обрабатываемого блока информационных бит. Пусть, например, n=10. Тогда перестановочная матрица состоит из двумерного массива 10x10 бит, причем в каждой строке и в каждом столбце массива содержится только один единичный элемент, все остальные элементы – нули.

Для рассматриваемого примера зададим перестановочную матрицу следующим образом.

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

3.4. Разбиваем информационное сообщение на блоки равной длины и обрабатываем простым перестановочным шифром (шифруем). Для этого воспользуемся следующей процедурой:

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

Алгоритм преобразования работает следующим образом. Информационное сообщение разбивается на блоки равной длины (длины n), после чего каждый блок поэлементно записывается в служебную переменную V. Таким образом, на каждом цикле (для каждого блока данных) в переменной V хранятся текущие n бит сообщения. Вектор V умножается на перестановочную матрицу P, чем обеспечивается шифрование простым перестановочным шифром. Обработанные таким образом данные объединяются в единый битовый массив M_b_P.

Для рассматриваемого примера информационных данных (см. п. 1.1.) алгоритм разбиения информационного сообщения на блоки равной длины и обработки простым перестановочным шифром функционирует следующим образом.

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. Реализуем алгоритм встраивания данных в пространственную область изображений методом LSB (см. п. 1.3.).

3.6. Реализуем алгоритм извлечения данных из пространственной области изображений методом LSB (см. п. 1.4.).

3.7. Разбиваем извлеченное сообщение (записанное в битовый массив M_b1) на блоки равной длины и обрабатываем простым перестановочным шифром (расшифровываем). Для этого воспользуемся следующей процедурой:

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

Алгоритм преобразования работает следующим образом. Извлеченное сообщение разбивается на блоки равной длины (длины n), после чего каждый блок поэлементно записывается в служебную переменную V. Таким образом, на каждом цикле (для каждого блока извлеченных данных) в переменной V хранятся текущие n бит. Вектор V умножается на матрицу P^{-1} (матрицу, обратную введенной выше перестановочной матрицы P), чем обеспечивается расшифрование извлеченных данных простым перестановочным шифром. Обработанные таким образом данные объединяются в единый битовый массив M_b1_P.

Для рассматриваемого примера информационных данных (см. п. 3.4.) алгоритм разбиения извлеченного сообщения на блоки равной длины и обработки простым перестановочным шифром (расшифрования) функционирует следующим образом.

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. Преобразуем массив информационных данных (см. п. 1.5.).

3.9. Полученный массив целых чисел записываем на физический носитель в виде текстового файла (см. п. 1.6.).

Задание 4. Реализация алгоритмов встраивания и извлечения сообщений методом псевдослучайного интервала

4.1. Загружаем исходные данные (см. п. 1.1.).

4.2. Преобразуем массив информационных данных (см. п. 1.2.).

4.3. Вводим секретное правило псевдослучайной перестановки.

Секретным ключом выступает правило, по которому отдельные биты информационного сообщения встраиваются в LSB байт контейнера, т.е. секретный ключ – это набор псевдослучайных чисел, которые задают величины интервалов между пикселями изображения, модифицируемыми в процессе встраивания информации.

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	…

Предположим, что информационное сообщение встраивается побитно в блок данных изображений, причем один бит сообщения встраивается в один столбец массива данных контейнера. Номер встраиваемого бита задает номер столбца контейнера, в который будет встраиваться бит сообщения. Текущее значение секретного ключа задает номер строки контейнера, в который будет встроен текущий бит сообщения. Таким образом, в качестве секретного ключа будем использовать массив псевдослучайных чисел в интервале допустимых номеров строк контейнера, размер массива равен числу столбцов контейнера. Для реализации такого подхода используем следующую процедуру.

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

Из приведенного примера следует, что первый бит сообщения будет встроен в 153-ю строку первого столбца массива данных контейнера, второй бит сообщения будет встроен в 155-ю строку второго столбца сообщения и т.д.

4.4. Реализуем алгоритм встраивания данных в пространственную область изображений методом ПСИ. Для этого воспользуемся следующей процедурой.

```
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 работает следующим образом. Исходный контейнер (массив R данных канала красного цвета) перезаписывается в новый массив S. Далее, в каждом столбце контейнера считывается значение яркости (десятичное число) из строки с номером, задаваемым секретным ключом. В считанном числе заменяется наименее значимый бит данных на текущий встраиваемый бит.Далее записываем заполненный контейнер на физический носитель (см. п. 1.3.).

4.5. Реализуем алгоритм встраивания данных в пространственную область изображений методом ПСИ. Для этого после загрузки данных контейнера (см. п. 1.4) воспользуемся следующей процедурой.

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

Алгоритм работает следующим образом. Во всех столбцах контейнера поочередно считываются значения из строк, номера которых заданы значением секретного ключа. Из считанных данных извлекаются наименее значимые биты, которые несут информационное содержание встроенного сообщения.

4.6. Преобразуем массив информационных данных (см. п. 1.5.).

4.7. Полученный массив целых чисел записываем на физический носитель в виде текстового файла (см. п. 1.6.).

6. Приклад оформлення звіту з лабораторної роботи

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

Вихідні дані:

"1.bmp"

R

```
C := READRGB("1.bmp")
R := READ_RED("1.bmp")
G := READ_GREEN("1.bmp")
B := READ_BLUE("1.bmp")
M := READBIN("3.txt" , "byte")
```

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

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

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

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} ← V_0
        | M_b1
```

M_b1 =

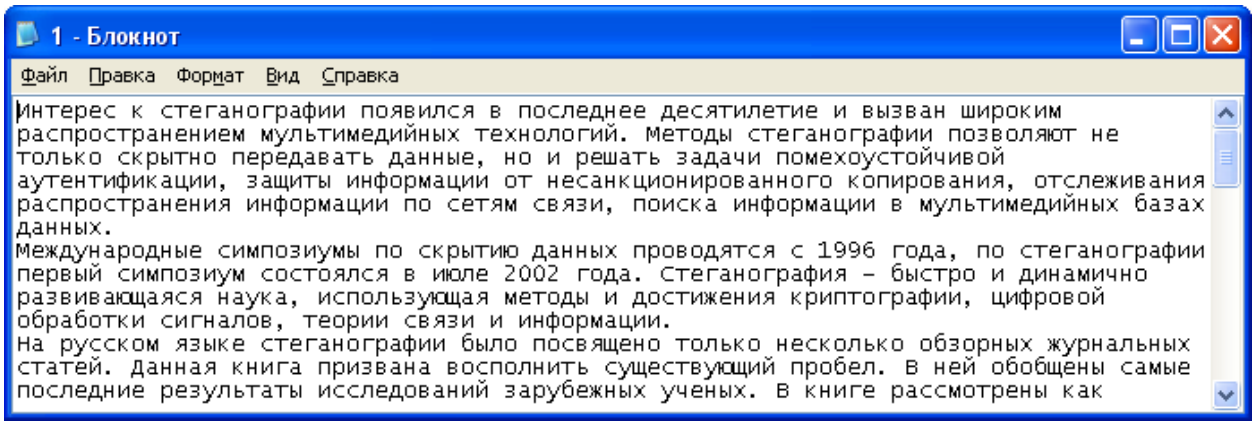
	0
0	0
1	0
2	0
3	0
4	1
5	1
6	1
7	1
8	1
9	0
10	1
11	1
12	0
13	1
14	1
15	...

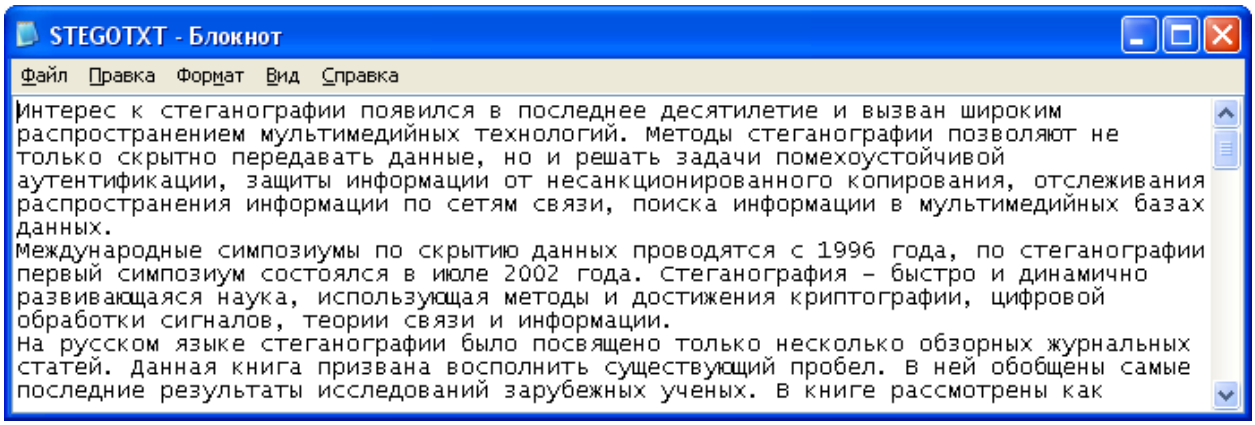
```
M1 := | for i ∈ 0.. rows(M_b1) − 8
      |   l ← floor(i/8)
      |   for j ∈ 0.. 7
      |     V_j ← M_b1_{l·8+j}
      |   M1_l ← B_D(V)
      | M1
```

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 ПСП:

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

$n := 10$

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

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 ПСІ:

```
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 просторової 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 лабораторної роботи

1. Реалізувати у середовищі символьної математики MathCAD алгоритми приховування та вилучення даних у просторової 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 вказівки 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);
   c.
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

…

4. Вопросы для поточного контроля подготовленности студентов к выполнению лабораторной работы №2

1. Простейшие помехоустойчивые линейные блоковые коды. Коды с проверкой четности, коды Хемминга. Матричное описание линейных блоковых кодов. Полиномиальное описание циклических кодов.
2. Метод блокового встраивания и его связь с линейными блоковыми кодами с контролем четности. Вероятностные характеристики метода блокового встраивания: вероятность правильного извлечения сообщений и вероятность возникновения ошибок.

3. Простейшие генераторы псевдослучайных чисел. Встроенные датчики псевдослучайных чисел в среде символьной математики MathCAD.
4. Понятие контрастности изображения. Чувствительность зрительной системы человека к незначительному изменению контрастности. Встраивания данных в неподвижные изображения методом квантования.
5. Криптографические и не криптографические генераторы псевдослучайных последовательностей. Доказуемо стойкие генераторы псевдослучайных последовательностей. Генераторы RSA и BBS.
6. Генераторы псевдослучайных последовательностей на регистрах сдвига. Генератор последовательностей максимального периода. Конгруэнтные генераторы псевдослучайных последовательностей. Линейный конгруэнтный генератор и инверсивный конгруэнтный генератор.
7. Методы экстраполяции (предсказания) случайных сигналов. Линейное предсказание и дельта-модуляция. Дифференциальная импульсно-кодовая модуляция. Сплайн интерполяция и интерполяционные многочлены Ньютона и Лагранжа.
8. Метод встраивания данных в неподвижные изображения Куттера-Джордана-Боссена (метод «креста»). Линейное предсказание сигналов при извлечении данных методом «креста».
9. Структура линейных блоковых кодов, стандартное расположение. Декодирование линейных блоковых кодов. Вероятностные характеристики помехоустойчивого кодирования. Вероятность обнаружения и не обнаружения ошибок линейными блоковыми кодами. Вероятность исправления ошибок декодером линейного блокового кода. Вероятность появления ошибок на выходе декодера.

5. Руководство к выполнению лабораторной работы №2

Задание 1. Реализация в среде MathCAD алгоритмов встраивания и извлечения сообщений в пространственной области неподвижных изображений методом блочного встраивания

1.1. Загружаем исходные данные: контейнер - неподвижное изображение (в формате *.bmp24); информационное сообщение – текстовый документ (в формате *.txt). Для этого в среде MathCAD выполняем следующие действия, аналогичные п. 1.1. руководства к лабораторной работе №1.

1.2. Преобразуем массив информационных данных. Для этого в среде MathCAD выполняем следующие действия, аналогичные п. 1.2. руководства к лабораторной работе №1.

1.3. Реализуем алгоритм встраивания данных в пространственную область изображений методом блочного встраивания. Для этого воспользуемся следующей процедурой:

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

Приведенная процедура реализует поэлементное встраивание битового массива информационных данных M_b в биты четности отдельных блоков изображения. При этом изображение разбито на блоки по столбцам, т.е. каждый столбец массива растровых данных канала красного цвета представляет собой отдельный блок изображения, в который встраивается соответствующий бит информационного сообщения. Бит четности b для каждого блока вычисляется во второй строке процедуры, посредством суммирования по модулю два всех элементов блока. Если бит четности текущего блока не совпадает со значением встраиваемого в данный блок информационного бита производится модификация (инвертирование) наименее значимого бита в первой строке блока (следующие три строки

процедуры). При этом изменяется значение бита четности, которое после произведенной модификации совпадает со значением встраиваемого информационного бита данных. Если значение бита четности изначально совпадало со значением встраиваемого информационного бита данных, тогда производится перезапись первого элемента текущего блока контейнера. Аналогичным образом перезаписываются и все остальные элементы блока контейнера, которые не модифицируются (последние строки процедуры).

Таким образом, встраивание данных осуществляется в биты четности отдельных блоков изображения, при этом модифицируются первые элементы блока. Результат встраивания (заполненный контейнер) храниться в массиве S1. Следует отметить, что значение наименее значимого бита первого элемента блока не всегда будет совпадать со значением встроенного информационного бита. Совпадают лишь бит четности и информационный бит данных.

Для визуального просмотра результата встраивания информационных данных выведем исходный массив растровых данных красного цвета R и полученный массив S1 с измененными битами четности блоков. Для рассматриваемого примера имеем:

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	...

Из представленных данных видно, что, например, значения $R_{0,1}$, $R_{0,2}$, $R_{0,3}$ полностью идентичны соответствующим значениям $S_{0,1}$, $S_{0,2}$, $S_{0,3}$. Практически это означает, что значение битов четности первого, второго и третьего столбцов массива R совпали со встраиваемыми информационными битами сообщения. Напротив, значение $R_{0,0}$ отличается на единицу от соответствующего значения массива S1. Это означает изменение бита четности нулевого блока контейнера в процессе встраивания сообщения.

Отметим, что внесенные искажения лежат ниже порога зрительной чувствительности человека.

Графическая интерпретация пустого и заполненного контейнера (канала красного цвета в градациях серого) приведена на следующем рисунке, из которого следует, что визуально внесенные искажения не заметны, что подтверждает вывод о чувствительности органов зрения человека.

R

S1

Полученный заполненный массив S1 записываем в канал красного цвета контейнера. Выполняем команду

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

В результате выполнения команды система MathCAD формирует на физическом носителе новый файл с именем «Stego_Blok.bmp».

Для графического отображения исходного (пустого) и заполненного контейнера выполним вставку соответствующих изображений:

"1.bmp"

"Stego_Blok.bmp"

Убеждаемся в отсутствии видимых искажений.

1.4. Реализуем алгоритм извлечения данных из пространственной области изображений методом блочного встраивания. Для этого в том же окне среды MathCAD выполняем команды чтения растровых данных неподвижного изображения из заданного файла (файла заполненного контейнера) в виде двумерного массива целых чисел. Для рассматриваемого примера выполняем команды:

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

Получим следующий 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

Далее вычисляем биты четности для каждого блока данных контейнера и формируем массив извлеченных информационных бит. Для этого используем следующую процедуру:

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

Данная процедура выполняет для всех столбцов (блоков) массива R1 вычисление бита четности, который и является встроенным информационным битом. В результате имеем линейный битовый массив M_b1, заполненный битами четности отдельных блоков массива растровых данных канала красного цвета заполненного 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	...

Сравнение данных массивов встроенных и извлеченных бит данных позволяет подтвердить правильность выполнения алгоритмов встраивания-извлечения.

1.5., 1.6. Формирование массива целых чисел, соответствующих ASCII кодировке встроенных символов сообщения и запись в текстовый файл извлеченных информационных данных осуществляется аналогично п. 1.5., 1.6. руководства к лабораторной работе №1.

Задание 2. Реализация в среде MathCAD алгоритмов встраивания и извлечения сообщений в пространственной области неподвижных изображений методом квантования

2.1., 2.2. Загружаем исходные данные и преобразуем массив информационных данных (согласно п. 1.1, 1.2).

2.3. Реализуем алгоритм встраивания данных в пространственную область изображений методом квантования. Для этого вначале сформируем случайный секретный ключ - таблицу квантования d, воспользовавшись следующей процедурой:

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

Данная процедура псевдослучайным образом (с использованием встроенного датчика “rnd()”) заполняет таблицу квантования для всех возможных значений перепадов яркости изображения. Пример заполненной таблицы имеет вид:

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	...

Нулевая строка массива d заполнена всеми возможными (от -255 до +255) значениями перепадов яркости, первая строка заполнена датчиком “rnd()”. Функция “ceil()” округляет аргумент до ближайшего целого, функция “rnd()” формирует равномерно распределенные на заданном участке псевдослучайное значение. Подробнее об используемых функциях изложено в приложении.

Для встраивания данных с использованием секретного ключа d воспользуемся следующей процедурой, которая реализует поэлементное встраивание битового массива информационных данных M_b в значения разностей элементов нулевого и первого столбца массива красного цвета контейнера. Другим словами каждый отдельный бит встраивается в одну разность, номер бита задает номер строки контейнера.

Текущее значение разности b находится в таблице квантования. Значение встраиваемого бита M_b_i сравнивается с битовым значением $d_{1,b+255}$ из второй строки матрицы квантования. При совпадении этих значений уровень контрастности в данной позиции не изменяется.

```
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
```

В случае несовпадения по заранее заданному правилу (в данном случае по правилу «поиск вправо») находится ближайшая позиция, для которой значения M_b_i и $d_{1,b+255}$ совпадают. Встраивание информации в таком случае состоит в модификации разности (в соответствии с найденным значением из таблицы квантования). Остальная часть изображения, не участвующая в модификации разности, перезаписывается из пустого контейнера без изменения.

Таким образом, встраивание данных осуществляется в значения разности между отдельными элементами массива R. Результат встраивания (заполненный контейнер) храниться в массиве S2. Для визуального просмотра результата встраивания информационных данных выведем исходный массив растровых данных красного цвета R и полученный массив S2 с измененными значениями разностей. Для рассматриваемого примера имеем:

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	…

Из представленных данных видно, что, например, значения разностей

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

полностью идентичны соответствующим значениям разностей

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

Практически это означает, что значение соответствующих битов из второй строки таблицы квантования совпало в этих позициях со значением встраиваемых информационных бит данных. Напротив, значения разностей

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

отличается от соответствующих значений разностей

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

Это означает изменение текущей разности в соответствии с найденным значением в таблице квантования. Так, например, значение разности

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

было изменено на значение разности

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

Как видно из приведенной выше таблицы квантования d, для значения разности

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

соответствующее значение из второй строки равно

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

Для встраивания информационного бита со значением «0» по правилу «поиск вправо» значение разности модифицируется на ближайшее найденное справа значение, для которого значение из второй строки таблицы квантования и значения встраиваемого бита совпадут. Очевидно, что это

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

и имеем соответствующее значение разности

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

что полностью подтверждает правильность работы алгоритма встраивания.

Следует отметить, что в предлагаемой реализации модификация разности достигается лишь изменением элемента контейнера в нулевом столбце, т.е. за счет модификации значений $S2_{i,0}$. Практически это означает, что все искажения будут сосредоточены в одном столбце. Абсолютное значение вносимых искажений определяется статистическими свойствами используемой в качестве секретного ключа псевдослучайной последовательности, т.е. второй строки таблицы квантования. Для эффективных криптографических генераторов с равновероятным распределением формируемых значений вносимые искажения лежат ниже порога зрительной чувствительности человека.

Графическая интерпретация пустого и заполненного контейнера (канала красного цвета в градациях серого) приведена на следующем рисунке, из которого следует, что визуально внесенные искажения не заметны, что подтверждает вывод о чувствительности органов зрения человека к незначительному изменению контрастности.

R

S2

Полученный заполненный массив S2 записываем в канал красного цвета контейнера. Выполняем команду

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

В результате выполнения команды система MathCAD формирует на физическом носителе новый файл с именем «Stego_ Kvant.bmp».

Для графического отображения исходного (пустого) и заполненного контейнера выполним вставку соответствующих изображений:

"1.bmp"

"Stego_Kvant.bmp"

Убеждаемся в отсутствии видимых искажений.

2.4. Реализуем алгоритм извлечения данных из пространственной области изображений методом квантования. Для этого в том же окне среды MathCAD выполняем команды чтения растровых данных неподвижного изображения из заданного файла (файла заполненного контейнера) в виде двумерного массива целых чисел. Для рассматриваемого примера выполняем команды:

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

Получим следующий 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

Далее вычисляем биты значения разности b между элементами первых двух столбцов и находим соответствующее битовое значение из второй строки таблицы квантования. Для этого используем следующую процедуру:

```
M_b2 := | for i ∈ 0.. rows(R2) − 1
        |     b ← R2[i,0] − R2[i,1]
        |     M_b2[i] ← d[1, b+255]
        | M_b2
```

Данная процедура выполняет для всех строк массива R2 вычисление значения разности и соответствующего ему бита данных, который и является встроенным информационным битом. В результате имеем линейный битовый массив M_b2, заполненный соответствующими битами из второй строки таблицы квантования.

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	…

Сравнение данных массивов встроенных и извлеченных бит данных позволяет подтвердить правильность выполнения алгоритмов встраивания-извлечения.

2.5., 2.6. Формирование массива целых чисел, соответствующих ASCII кодировке встроенных символов сообщения и запись в текстовый файл извлеченных информационных данных осуществляется аналогично п. 1.5., 1.6.

Задание 3. Реализация в среде MathCAD алгоритмов встраивания и извлечения сообщений в пространственной области неподвижных изображений методом Куттера-Джордана-Боссена (методом «креста»)

3.1., 3.2. Загружаем исходные данные и преобразуем массив информационных данных (согласно п. 1.1, 1.2).

3.3. Реализуем алгоритм встраивания данных в пространственную область изображений методом квантования. Для этого вначале реализуем функцию вычисления яркости отдельного пикселя с заданными координатами:

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

и установим параметры метода квантования:

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

а также определим функцию модификации отдельного пикселя следующим образом

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

Очевидно, что значение $\lambda(x,y)$ определяется как полноцветная яркость пикселя по значениям яркости трех цветовых компонент с соответствующим весовым коэффициентом. Выбранные параметры $\gamma = 0{,}05$ (энергия встраиваемого бита) и $\sigma = 3$ (размер области предсказания) являются простейшими показателями надежной работы стаганоалгоритма.

Функция встраивания SV(x,y,b) состоит в модификации яркости синего цвета заданного пикселя на долю его полноцветной яркости, задаваемой параметром γ.

Для примера, покажем правильность работы функции встраивания.

Значение яркости пикселя с координатами (7,7) соответствует

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

Соответствующее значение яркости синего цвета пикселя

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

Возможная модификация яркости синего цвета пикселя согласно функции встраивания принимает значения $208 \pm [0{,}05 \cdot 176{,}42] = 208 \pm 9$. Очевидно, что алгоритм вычисления функции SV(7,7,0) = 199 работает правильно.

Для встраивания массива информационных данных M_b воспользуемся следующей процедурой. Она реализует поэлементное встраивание битового массива в значения яркостей синего цвета посредством модификации функцией встраивания SV(x,y,b).

Областью для встраивания выбрана диагональ массива яркостей синего цвета контейнера. Остальные элементы контейнера (не из области модификации) подлежат перезаписыванию из 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
```

Следует отметить, что встраивание начинается не с пиксела с координатами (0, 0), а с пикселя, имеющего координаты (σ,σ). Это выполнено для возможности в дальнейшем осуществить предсказание методом «креста».

Для визуального просмотра результата встраивания информационных данных выведем исходный массив растровых данных синего цвета B и полученный массив S2 с измененными значениями разностей. Для рассматриваемого примера имеем:

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	…

Из представленных данных видно, что, например, значения

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

что соответствует встраиванию «1» в эти позиции.

Значение $S3_{5,5} < B_{5,5}$ соответствует встраиванию «0».

Следует отметить, что величина вносимых искажений определяется введенным значением $\gamma = 0{,}05$ (энергия встраиваемого бита), как доли полноцветной яркости пикселя, приходящейся на модификацию яркости синего цвета.

Графическая интерпретация пустого и заполненного контейнера (канала синего цвета в градациях серого) приведена на следующем рисунке, из которого следует, что визуально внесенные искажения не заметны, что подтверждает вывод о чувствительности органов зрения человека к незначительному изменению синего цвета.

S3

B

Полученный заполненный массив S3 записываем в канал синего цвета контейнера. Выполняем команду

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

В результате выполнения команды система MathCAD формирует на физическом носителе новый файл с именем «Stego_Krest.bmp».

Для графического отображения исходного (пустого) и заполненного контейнера выполним вставку соответствующих изображений:

"1.bmp"

"Stego_Krest.bmp"

Убеждаемся в отсутствии видимых искажений.

3.4. Реализуем алгоритм извлечения данных из пространственной области изображений методом «креста». Для этого в том же окне среды MathCAD выполняем команды чтения растровых данных неподвижного изображения из заданного файла (файла заполненного контейнера) в виде двумерного массива целых чисел. Для рассматриваемого примера выполняем команды:

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

Получим следующий 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

Далее, для каждого встроенного бита информации вычисляем предсказанное значение b синего цвета и сравниваем с наблюдаемым значением $B3_{i,i}$. Используем следующую процедуру:

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

В результате сравнения b с наблюдаемым значением $B3_{i,i}$ принимаем решение о значении встроенного бита информации. Выполнение предсказания о значении яркости синего цвета производим для каждого пикселя, подлежавшего модификации. Позиция (координаты) модифицированного пикселя являются секретной ключевой информацией.

Приведем результат работы данной процедуры извлечения данных для рассматриваемого примера.

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	...

Очевидно, что результат извлечения первых трех бит неправильный. Наступление такого события не исключено логикой алгоритма извлечения, вероятность его возникновения определяется статистическими свойствами контейнера. Повысить вероятность правильного извлечения информационных бит данных можно за счет повышения энергии встраиваемых бит данных, т.е. посредством увеличения коэффициента γ. Однако подобная процедура неизбежно приведет к увеличению вносимых искажений в контейнер-изображение.

3.5., 3.6. Формирование массива целых чисел, соответствующих ASCII кодировке встроенных символов сообщения и запись в текстовый файл извлеченных информационных данных осуществляется аналогично п. 1.5., 1.6.

Задание 4. Исследование вероятностных характеристик стенографического метода встраивания данных Куттера-Джордана-Боссена (метода «креста»)

4.1. Проведем оценку вероятности правильного извлечения сообщения и величины вносимых искажений от коэффициента γ. Для этого будем последовательно увеличивать величину γ и для каждого значения рассчитывать частоту v правильно извлеченных информационных бит. Одновременно будем рассчитывать усредненную величину w внесенных искажений, выраженной в процентном соотношении к максимальному значению яркости. Используем для этого следующие процедуры:

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

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

Так, для рассматриваемого примера при γ = 0,45 имеем следующие значения:

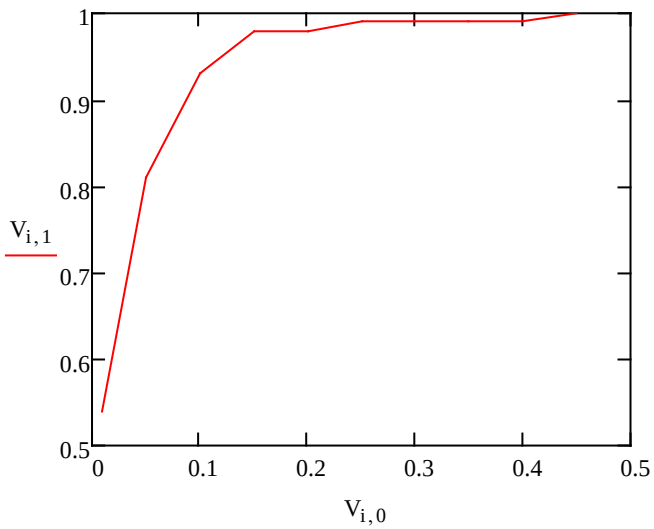
v = 1 w = 20.809

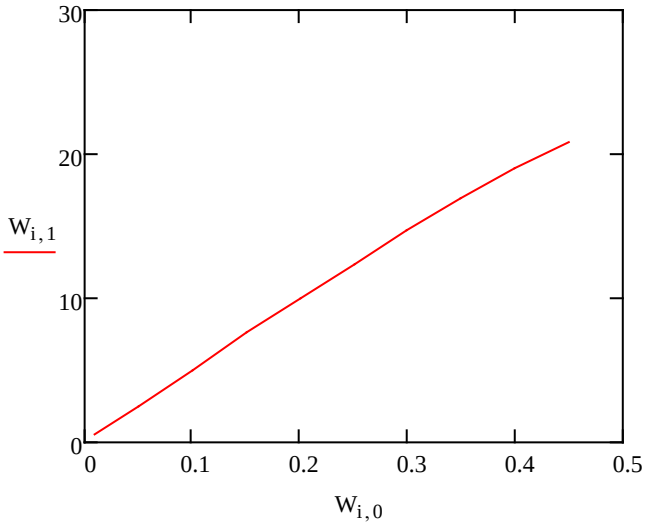
Полученные эмпирические данные занесем в соответствующие таблицы:

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

4.2. Построим графики полученных эмпирических зависимостей:

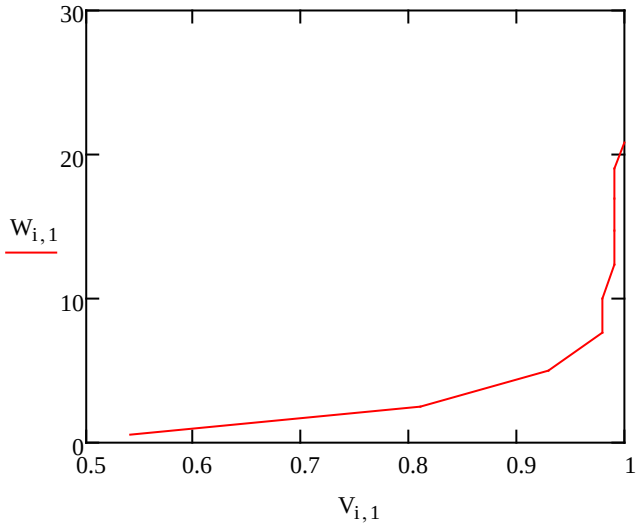
$$i := 0..9$$





Очевидно, что величина вносимых искажений растет линейно от коэффициента γ. Однако эмпирическая зависимость вероятности правильного извлечения информационных данных ведет себя иначе. При малых значениях коэффициента γ величина W растет быстро, однако при γ > 0,2 дальнейшее увеличение энергии встраивания не приводит к существенному повышению вероятности правильного извлечения, повышать величину V в данном случае не целесообразно.

4.3. Построим интегральный график зависимости величины W вносимых искажений в контейнер-изображение при обеспечении соответствующей вероятности V правильного извлечения информационных данных:



Очевидно, что эффективное сокрытие встраиваемых информационных данных без внесения значительных искажений (W < 5%) в контейнер-изображение будет наблюдаться только при вероятности правильного извлечения данных V < 0,8 … 0,9, что соответствует энергии встраивания γ = 0,05 … 0,15. Повышение достоверности извлекаемых данных за счет дальнейшего увеличения энергии встраивания нецелесообразно, поскольку это приводит к внесению неоправданно высоких искажений в контейнер-изображение. В данном случае наиболее перспективным является реализация помехоустойчивого кодирования информационных данных и контроль возникающих при стеганографических преобразованиях ошибок.

Задание 5 (дополнительное). Реализация помехоустойчивого кодирования информационных данных для повышения вероятностных характеристик стенографического метода встраивания данных Куттера-Джордана-Боссена (метода «креста»)

5.1. Реализуем помехоустойчивое кодирование простейшим линейным блоковым кодом Хемминга. Для этого введем следующие порождающую и проверочную матрицы

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

5.2. Реализуем алгоритмы кодирования и декодирования отдельных кодовых слов. Для этого воспользуемся следующими функциями

```
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
```

В качестве примера рассмотрим информационный вектор:

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

После выполнения функции кодирования имеем следующее кодовое слово:

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

Внесем ошибку в произвольном кодовом символе, например, $c_3 := 1$
Имеем следующее слово с ошибкой, которое, после декодирования, восстанавливается в безошибочную последовательность:

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

5.3. Реализуем алгоритм помехоустойчивого кодирования массива информационных данных:

```
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 рассматриваемого примера имеем:

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	…

Реализованная процедура считывает по четыре бита из массива M_b и кодирует их помехоустойчивым кодом Хемминга. Результат кодирования блоками по семь бит записывается в массив M_b_cod.

5.4. Реализовываем встраивание сформированных данных в контейнер-изображение методом «креста». Для этого воспользуемся рассмотренными в п. 3.3. процедурами:

```
γ := 0.05

SV(x, y, b) := round[B_(x,y) + (2·b − 1)·γ·λ(x, y)]
```

```
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 результате формируем массив S4 синего цвета со встроенными данными. Сформируем заполненный контейнер и просмотрим результат:

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

S4

B

"1.bmp"

"Stego_Krest_cod.bmp"

5.5. Реализовываем извлечение сформированных данных в контейнер-изображение методом «креста». Для этого воспользуемся рассмотренными 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

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

В результате выполнения рассмотренных процедур формируем массив извлеченных данных:

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. Реализуем помехоустойчивое декодирование извлеченных данных:

```
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
```

Приведенная процедура считывает извлеченные данные блоками по семь бит и декодирует их рассмотренной в п. 5.2. функцией. В результате формируем массив извлеченных данных с исправленными ошибками.

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. Исследуем вероятностные характеристики метода «креста» с использованием помехоустойчивого кодирования. Для этого воспользуемся рассмотренными 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

Полученные эмпирические данные занесем в соответствующие таблицы и сравним с уже имеющимися зависимостями:

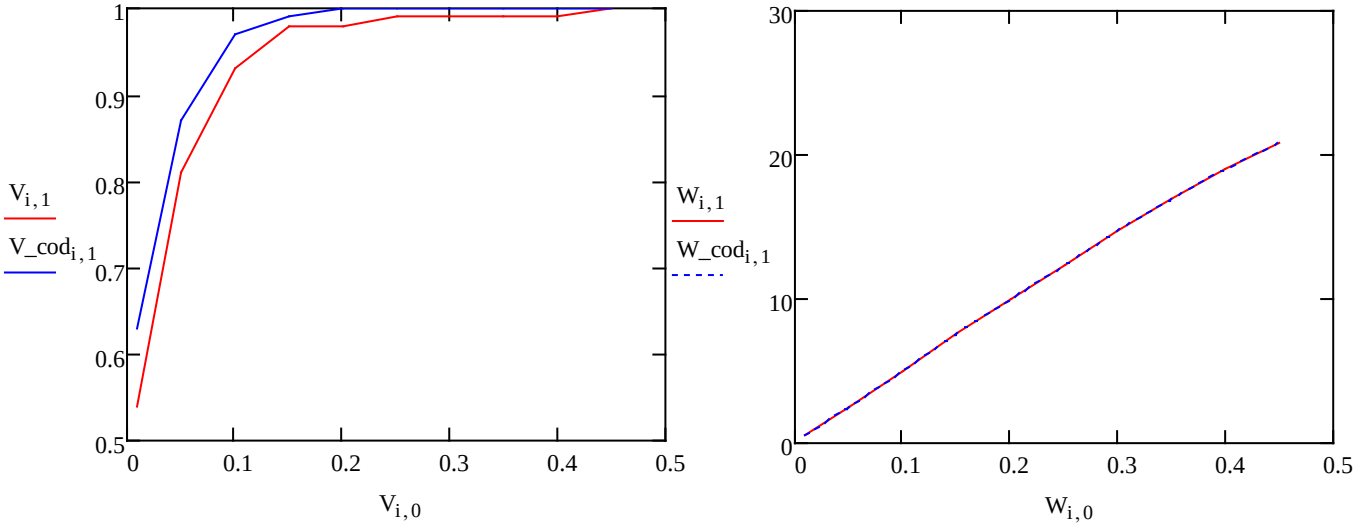
$$V := \begin{pmatrix} 0.01 & 0.54 \\ 0.05 & 0.81 \\ 0.1 & 0.93 \\ 0.15 & 0.98 \\ 0.2 & 0.98 \\ 0.25 & 0.99 \\ 0.3 & 0.99 \\ 0.35 & 0.99 \\ 0.4 & 0.99 \\ 0.45 & 1 \end{pmatrix} \quad V\_cod := \begin{pmatrix} 0.01 & 0.63 \\ 0.05 & 0.87 \\ 0.1 & 0.97 \\ 0.15 & 0.99 \\ 0.2 & 1 \\ 0.25 & 1 \\ 0.3 & 1 \\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

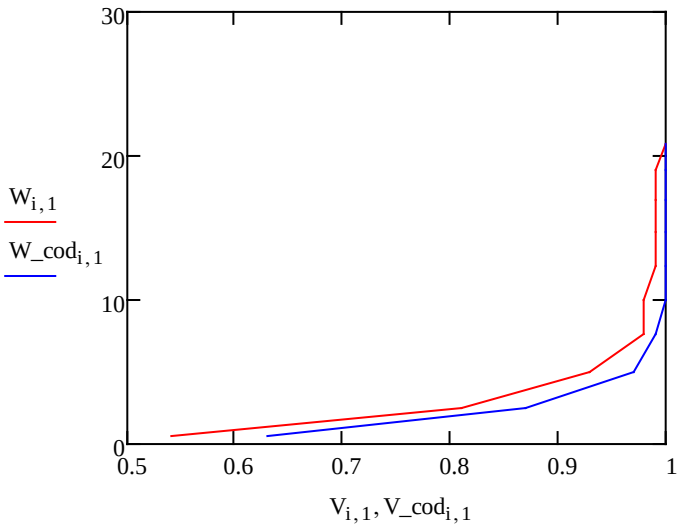
$$W := \begin{pmatrix} 0.01 & 0.55 \\ 0.05 & 2.55 \\ 0.1 & 5.0 \\ 0.15 & 7.6 \\ 0.2 & 10 \\ 0.25 & 12.4 \\ 0.3 & 14.7 \\ 0.35 & 16.9 \\ 0.4 & 19 \\ 0.45 & 20.8 \end{pmatrix} \quad W\_cod := \begin{pmatrix} 0.01 & 0.55 \\ 0.05 & 2.55 \\ 0.1 & 5.0 \\ 0.15 & 7.6 \\ 0.2 & 10 \\ 0.25 & 12.4 \\ 0.3 & 14.7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

Очевидно, что использование помехоустойчивого кодирования позволило незначительно повысить вероятность правильного извлечения встроенных данных.

Построим соответствующие графики:

$i := 0..9$





Полученные зависимости показывают, что использование даже простейшего помехоустойчивого кода Хемминга (синяя кривая) позволяет улучшить вероятностные характеристики метода «креста». Уровень вносимых искажений при этом не изменяется. В тоже время использование кода Хемминга привело к снижению практически в два раза объема встроенных информационных данных. Для практического использования целесообразно применение мощных помехоустойчивых кодов, которые позволят добиться безошибочного извлечения при незначительном снижении объема встроенных информационных данных.

6. Приклад оформлення звіту з лабораторної роботи №2

Лабораторная работа №2
Скрытие данных в пространственной области изображений методом блочного скрытия, методом квантования, методом "креста"

"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

	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	...

$R =$

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

$D\_B(x) :=$ 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

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

$M =$

	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	...

$M\_b =$

$S1 :=$ 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 $j \in 1..\mathrm{rows}(R)-1$: $S1_{j,i} \leftarrow R_{j,i}$

$S1$

$WRITERGB("Stego\_Blok.bmp") := augment(S1, G, B)$

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 квантования

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

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	…

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

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 "креста"

$\lambda(x,y) := 0.29890 \cdot R_{x,y} + 0.58662 \cdot G_{x,y} + 0.11448 \cdot B_{x,y}$ $\gamma := 0.05$ $\sigma := 3$

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

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

```
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
```

WRITERGB("Stego_Krest.bmp") := augment(R, G, 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	…

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	…

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

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	…

Исследование вероятностных характеристик

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

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

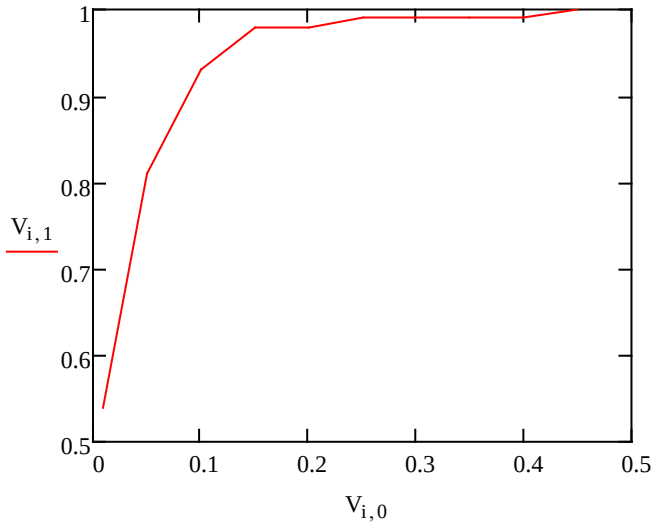
v = 0.81

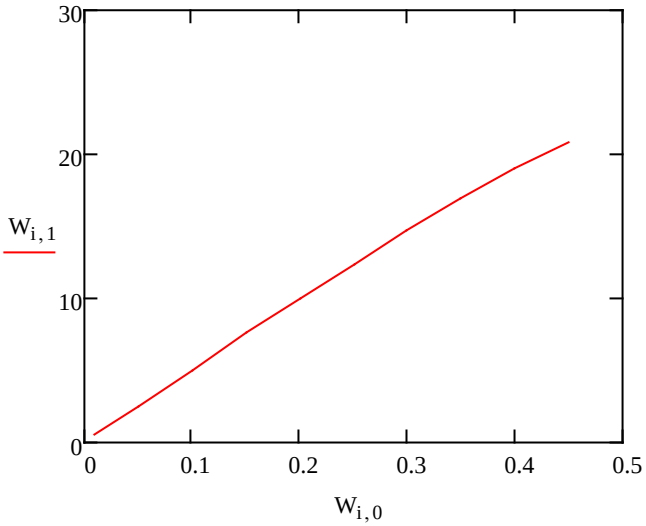
w = 2.55

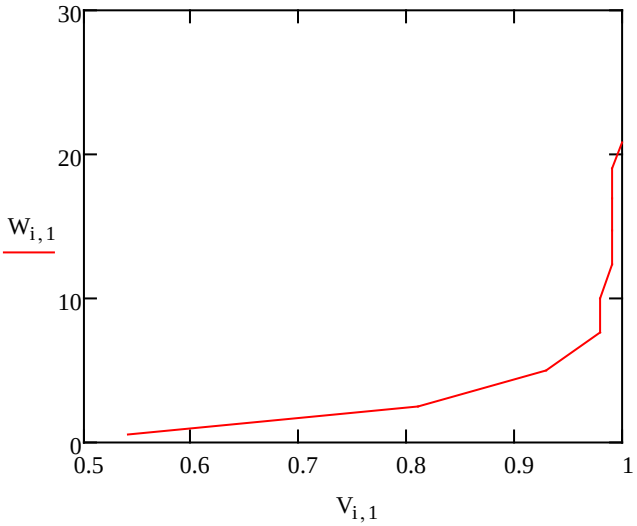
$$V := \begin{pmatrix} 0.01 & 0.54 \\ 0.05 & 0.81 \\ 0.1 & 0.93 \\ 0.15 & 0.98 \\ 0.2 & 0.98 \\ 0.25 & 0.99 \\ 0.3 & 0.99 \\ 0.35 & 0.99 \\ 0.4 & 0.99 \\ 0.45 & 1 \end{pmatrix}$$

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

$i := 0..9$







Помехоустойчивое кодирование информационных данных

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

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

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

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

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

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

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

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

$M\_b =$

	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	…

$$\gamma := 0.05$$

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

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

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

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

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

```
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	…

Исследование вероятностных 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

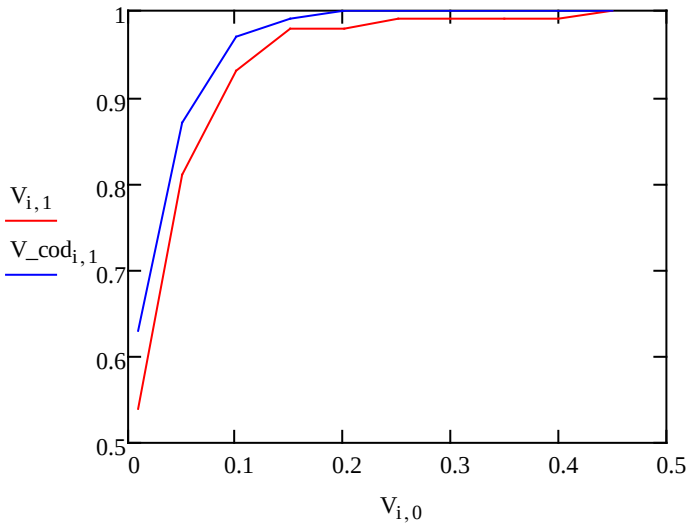
W :=

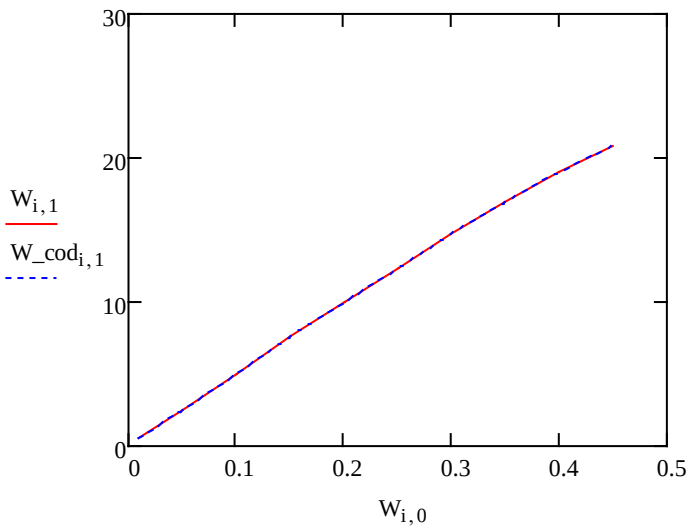
0.01	0.55
0.05	2.55
0.1	5.0
0.15	7.6
0.2	10
0.25	12.4
0.3	14.7
0.35	16.9
0.4	19
0.45	20.8

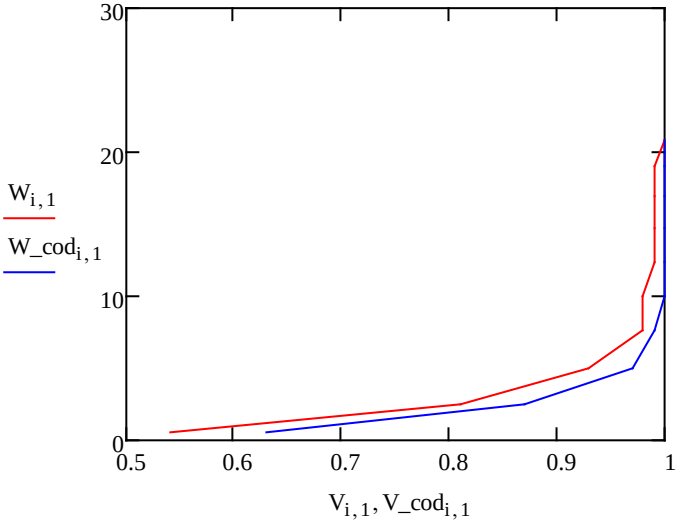
W_cod :=

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

$i := 0..9$


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


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


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

Лабораторна робота №3 «Приховування даних в просторової 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

Перешкодозахищені системи зв'язку та управління

Прагнення забезпечити високу перешкодозахищеність, множинний безконфліктний доступ до каналу, а також енергетичний і інформаційний захист повідомлень, які передаються, привело до створення широкополосних перешкодозахищених 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 перешкодозахищеного 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 перешкоди $P_п$.

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 перешкодозахищених 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 перешкоди 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 перешкодозахисних 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:

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

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

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.

Рис. 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).

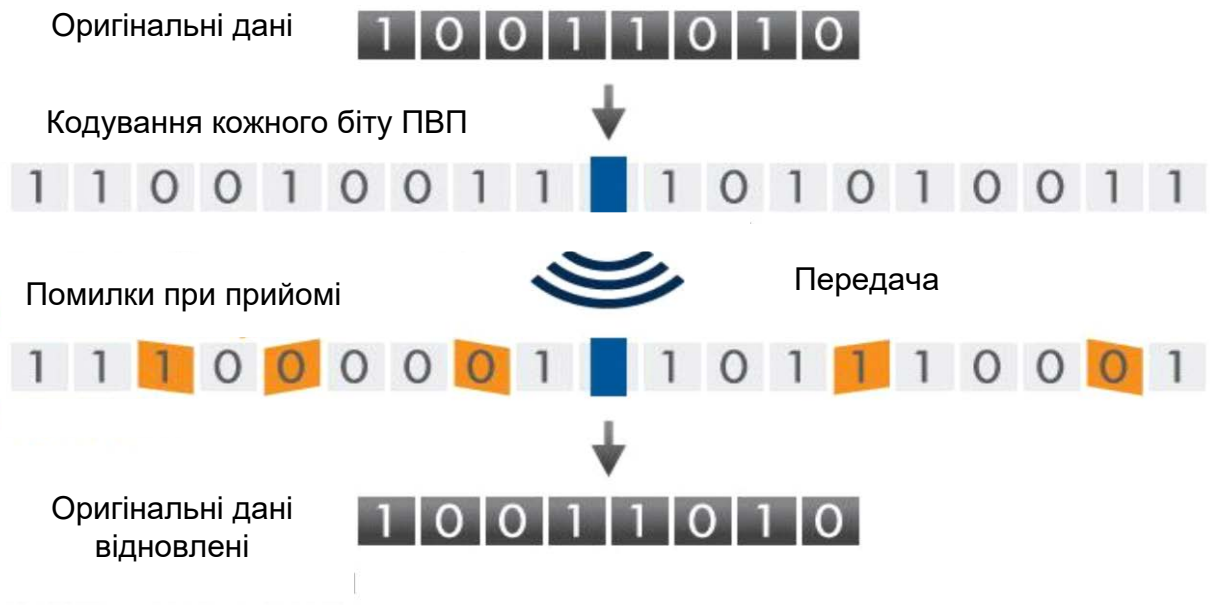


Рис. 3.2. Технологія кодового розділення каналів CDMA

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

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

тобто значення інформаційного сигналу на приймальній стороні визначається згідно виразу

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

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

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

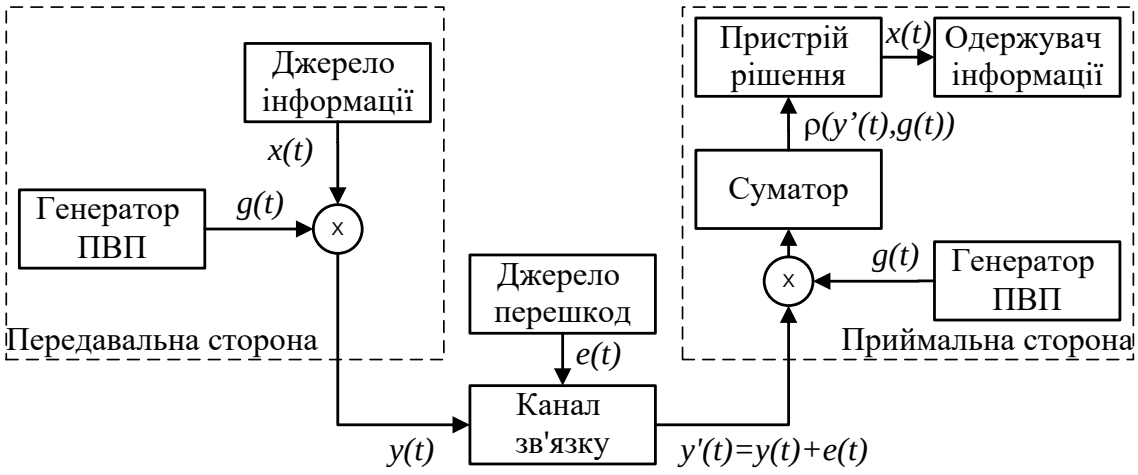


Рис. 3.3. Структурна схема тракту прийому-передачі інформації з використанням прямого розширення спектру

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 перешкодозахисних 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:

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

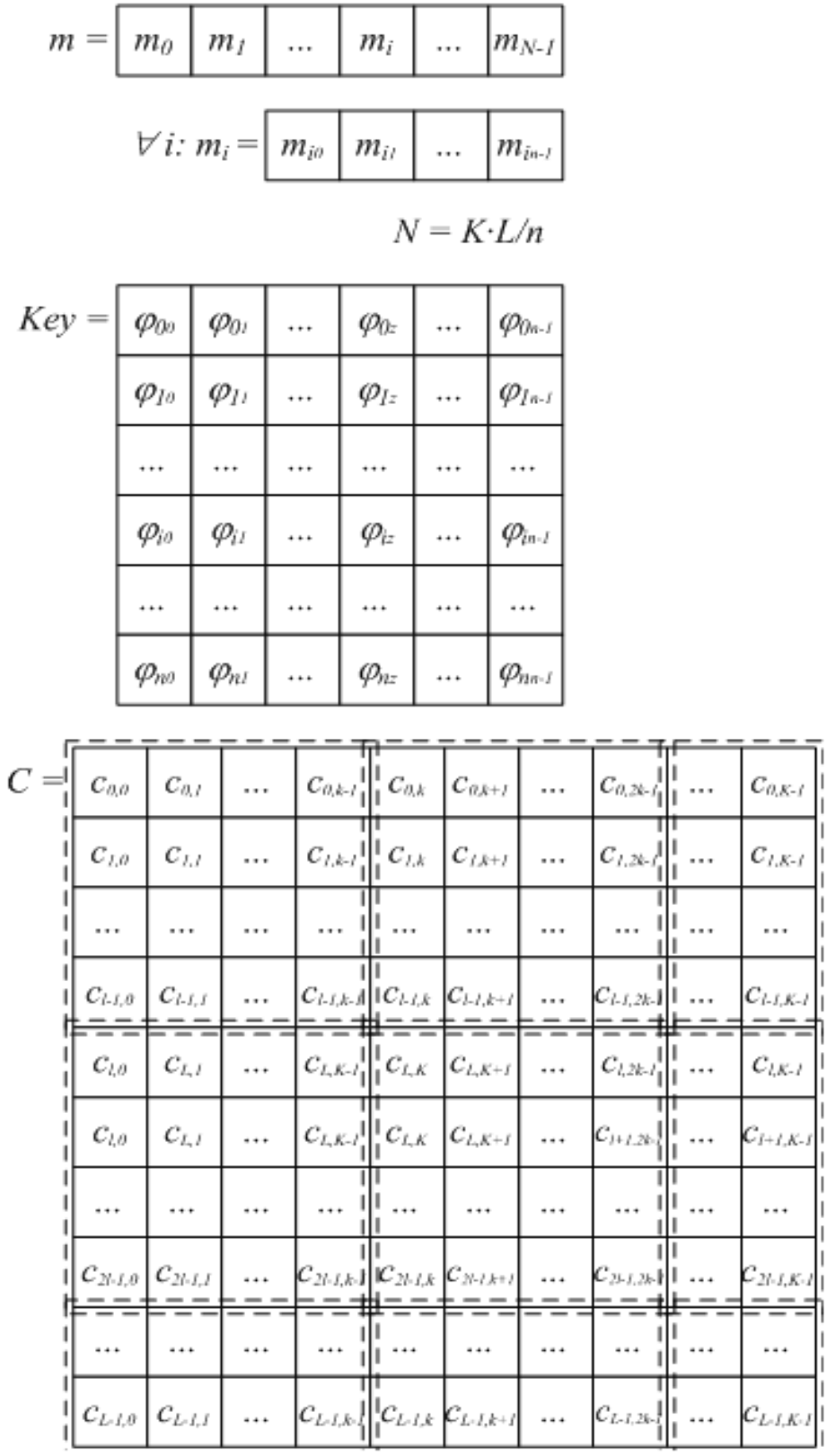


Рис. 3.4. Формальне представлення інформаційного повідомлення, контейнера-зображення і ключових даних

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:

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

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

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

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, $i = 0,..., N-1$ 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 $\Phi_j \in \Phi$, $i = 0,..., N-1$. Для цього кожен блок S_i представляється у формі вектора $S_i = (S_{i_0}, S_{i_1}, ..., S_{i_{n-1}})$, $i = 0,..., N-1$

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):

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

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:

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

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

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

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 $\rho\left(S_i,\Phi_j\right)$ в (3.15) залежить тільки від $m_{i_j}(t)$, звідки маємо:

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

Якщо $\rho\left(S_i,\Phi_j\right)=0$ 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.

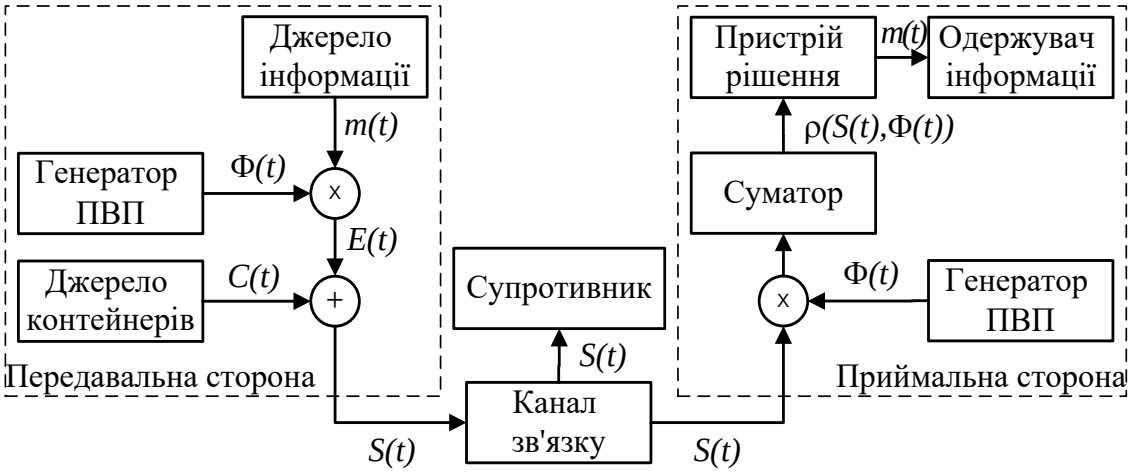


Рис. 3.5. Структурна схема вбудовування інформації в контейнер-зображення для скритної передачі повідомлень

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 перешкоди $y'(t) = y(t) + e(t)$ 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 перешкодозахисту), 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 захования 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 перешкодозахисного 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:

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

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:

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

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.

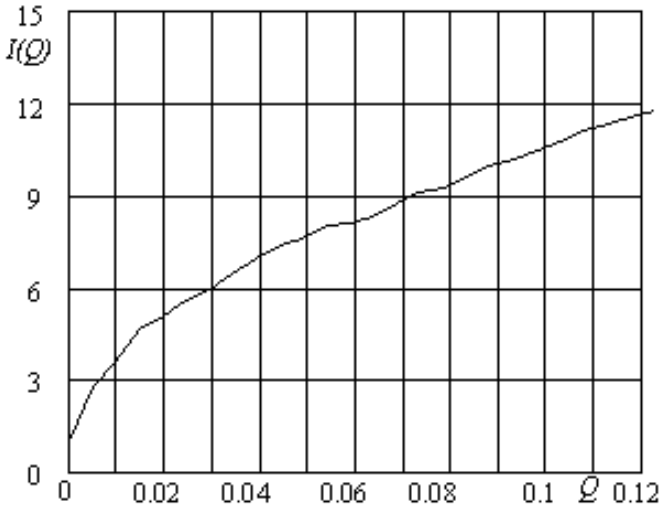


Рис. 3.6. Залежність I(Q)

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).

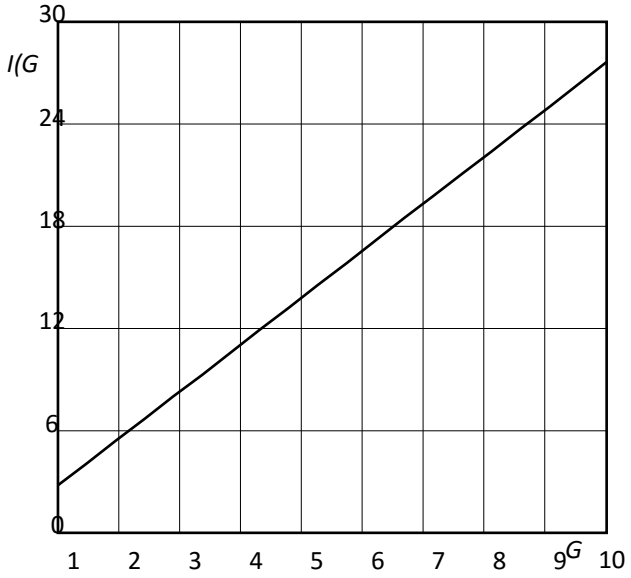


Рис. 3.7. Залежність $I(G)$ при $Q = 0.005$

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.

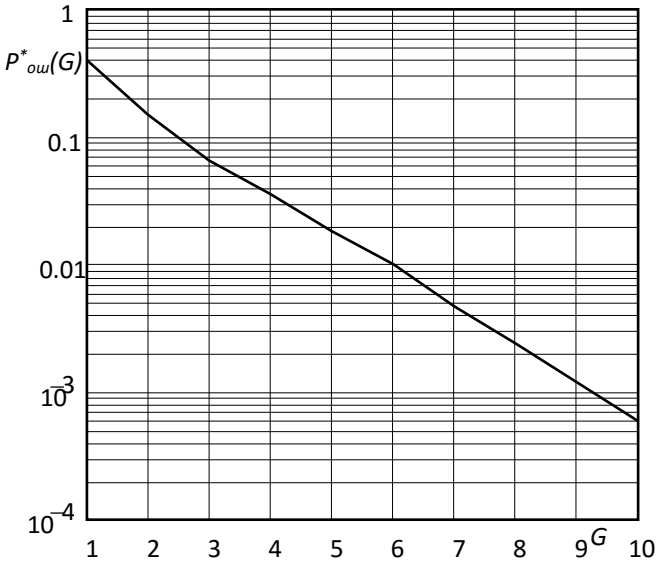


Рис. 3.8. Залежність $P^*_{ош}(G)$ при $Q = 0.005$

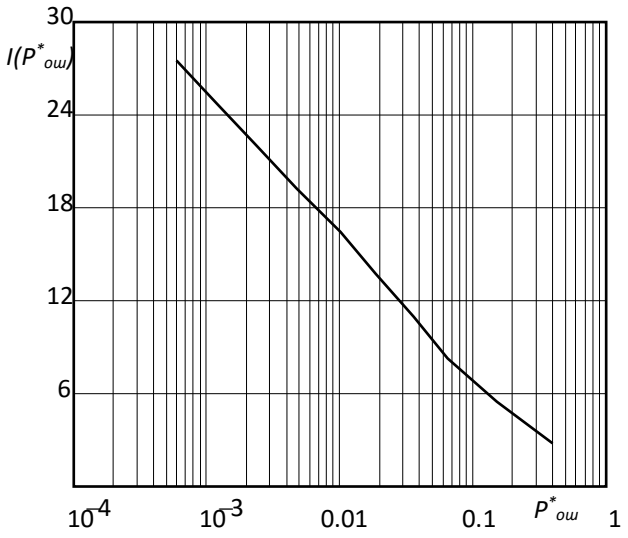


Рис. 3.9. Залежність $I(P^*_{ош})$ при $Q = 0.005$

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 перешкодостійким 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 перешкоди 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. Вопросы для поточного контроля подготовленности студентов к выполнению лабораторной работы №3

1. Методи розширення спектру, які застосовуються для перешкодо захищеної 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. Руководство к выполнению лабораторной работы №3

Задание 1. Реализация в среде MathCAD алгоритмов формирования ансамблей ортогональных дискретных сигналов Уолша-Адамара и алгоритмов кодирования информационных бит данных сложными дискретными сигналами

1.1. Загружаем исходные данные: контейнер - неподвижное изображение (в формате *.bmp24); информационное сообщение – текстовый документ (в формате *.txt). Для этого в среде MathCAD выполняем следующие действия, аналогичные п. 1.1. руководства к лабораторной работе №1.

1.2. Преобразуем массив информационных данных. Для этого в среде MathCAD выполняем следующие действия, аналогичные п. 1.2. руководства к лабораторной работе №1.

1.3. Реализуем алгоритм формирования матриц Адамара. Для этого воспользуемся следующей процедурой:

```
H0 := (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
       Hi ← F1
H
```

Процедура итеративно формирует матрицы Адамара H_1, H_2, …, H_8. Матрица H_0, состоящая из одного элемента, задается в качестве начального значения итеративной процедуры. Остальные матрицы формируются по рекуррентному правилу:

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

Результатом выполнения процедуры является массив матриц H, каждый элемент которого является матрицей 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. Реализуем алгоритм формирования ансамблей ортогональных дискретных сигналов Уолша-Адамара. Для этого сформируем массив строк матрицы Адамара, каждый элемент сформированного таким образом массива является дискретным сигналом Уолша-Адамара:

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

Просмотрим, в качестве примера, несколько дискретных сигналов:

$ArrayFunction_5 =$

	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. Реализуем алгоритм кодирования информационных бит данных сложными дискретными сигналами. Для этого сформируем модулированное информационное сообщение, для чего преобразуем массив информационных бит в массив, состоящий из «1» и «-1» следующей процедурой:

```
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 результате получим массив m, сравним его с исходным массивом данных:

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	...

Встраивать сообщение в контейнер будем построчно (несколько бит в одну строку контейнера). Кодирование сложными дискретными сигналами произведем следующим образом:

```
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
```

Значение «k» задает число информационных бит, встраиваемых в один фрагмент (в одну строку) контейнера. Значение «g» задает «энергию» встраиваемых бит сообщения, т.е. фактически является коэффициентом

усиления встраиваемого сообщения. В данном случае кодирование сложными сигналами производится без усиления (g = 1). Само кодирование состоит в умножении модулированного сообщения на сформированные выше дискретные сигналы Уолша-Адамара. Результатом выполнения процедуры кодирования является массив «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	…

Каждый элемент массива представляет собой сумму произведений k модулированных сообщений и дискретных сигналов Уолша-Адамара. Всего массив «Sum» содержит «Rows(R)» элементов по числу строк контейнера. Каждый элемент массива «Sum» предназначен для встраивания в отдельную строку контейнера-изображения. Максимальное абсолютное значение элементов массива «Sum» задает максимальную величину вносимых при встраивании сообщения искажений. Эта величина не будет превосходить g·k, т.е. величина вносимых искажений непосредственно определяется коэффициентом усиления информационного сообщения и числом встраиваемых бит данных в один фрагмент изображения.

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. Реализуем алгоритм встраивания информационных данных в пространственную область изображения на основе прямого расширения спектра с использованием ортогональных дискретных сигналов 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 встраивания состоит в суммировании данных контейнера (цифрового изображения) с модулированным сложными дискретными сигналами информационным сообщением.

Используемые ограничения предназначены для учета возможностей выхода значений отдельных элементов заполненного контейнера за диапазон допустимых значений яркостей отдельных пикселей изображения. В результате выполнения процедуры встраивания данных получаем массив заполненного контейнера (стеганограмму). Сравним массив данных пустого и заполненного контейнера:

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	…

Результат сравнения показывает, что максимально вносимые изменения в контейнер-изображения не превосходят величины $g \cdot k = 4$. Так, например, значение яркости красного цвета отдельного пикселя $R_{1,3} = 86$ в ходе встраивания информации изменилось на значение $S_{1,3} = 90$, т.е. абсолютное

изменение яркости красного цвета в данном пикселе составило максимальное значение. В большинстве своем изменения яркостей отдельных пикселей изображения лежат ниже этого, порогового значения.
Просмотрим результат встраивания:

S

R

Очевидно, что в результате встраивания данных с выбранными параметрами (коэффициент усиления g и число бит k, встраиваемых в один элемент контейнера) внесенные искажения в контейнер-изображения лежат ниже порога чувствительности зрительной системы человека и не могут быть визуально обнаружены.

Полученный заполненный массив S записываем в канал красного цвета заполненного контейнера-стеганограммы. Выполняем команду

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

В результате выполнения команды система MathCAD формирует на физическом носителе новый файл с именем «Stego_Adamar_1_4.bmp».

Для графического отображения исходного (пустого) и заполненного контейнера выполним вставку соответствующих изображений:

"Stego_Adamar_1_4.bmp"

"1.bmp"

Убеждаемся в отсутствии видимых искажений.

2.2. Реализуем алгоритм корреляционного приема дискретных сигналов. Для этого воспользуемся следующей функцией, предназначенной для расчета коэффициента корреляции:

```
MultString(A, B) := | X ← 0
                    | for i ∈ 0.. 255
                    |   X ← X + A_i·B_i
                    | X
```

Приведенная функция «MultString(A,B)» вычисляет скалярное произведение векторов «A» и «B», результатом является коэффициент корреляции аргументов функции.

Для проверки правильности вычислений выполним расчет коэффициента корреляции двух ортогональных векторов. Для этого запишем:

$$\mathrm{MultString}\left(\mathrm{ArrayFunction}_2, \mathrm{ArrayFunction}_3\right) = 0$$

Очевидно, что использование в качестве аргументов функции скалярного произведения двух ортогональных сигналов Уолша-Адамара приводит к нулевому результату, что подтверждает правильность работы реализованной функции.

Рассмотрим теперь массив модулированных информационных данных «m», массив данных – результат кодирования информационных данных сложными сигналами «Sum», а также сами сложные сигналы, используемые при встраивании информации «$\mathrm{ArrayFunction}_1$», «$\mathrm{ArrayFunction}_2$», «$\mathrm{ArrayFunction}_3$», «$\mathrm{ArrayFunction}_4$»:

m =

	0
0	-1
1	-1
2	-1
3	1
4	-1
5	-1
6	1
7	...

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	...

Вычислим коэффициент корреляции массива «Sum_0» со всеми четырьмя ортогональными сигналами, используемыми при встраивании информационных данных. Получим:

$$MultString\left(Sum_0, ArrayFunction_1\right) = -256$$

$$MultString\left(Sum_0, ArrayFunction_2\right) = -256$$

$$MultString\left(Sum_0, ArrayFunction_3\right) = -256$$

$$MultString\left(Sum_0, ArrayFunction_4\right) = 256$$

Очевидно, что знак коэффициента корреляции совпадает с первыми четырьмя элементами модулированного информационного сообщение «m». Следующий элемент информационных данных после модуляции соответствует массиву «Sum_1» и ортогональному сигналу «$ArrayFunction_1$». Проверим правильность работы алгоритма:

$$MultString\left(Sum_1, ArrayFunction_1\right) = -256$$

Знак коэффициента корреляции в данном случае также совпадает со встраиваемым элементом данных «m_4».

2.3. Реализуем алгоритм извлечения информационных данных из пространственной области изображения на основе прямого расширения спектра с использованием ортогональных дискретных сигналов Уолша-Адамара. Для этого в том же окне среды MathCAD выполняем команды чтения растровых данных неподвижного изображения из заданного файла (файла заполненного контейнера) в виде двумерного массива целых чисел. Для рассматриваемого примера выполняем команды:

«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”)».

Получим следующий 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

Далее формируем массив строк заполненного контейнера следующей процедурой:

```
ArrayString := | for i ∈ 0.. rows(R1) − 1
               |   for j ∈ 0.. cols(R1) − 1
               |     a_j ← R1_{i,j}
               |   ArrayString_i ← a
               | ArrayString
```

В результате получаем массив строк, в каждую из которых с помощью k ортогональных дискретных сигнала Уолша-Адамара встроено k информационных бит сообщения:

$ArrayString_0 =$

	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 извлечения встроенного сообщения воспользуемся процедурой, основанной на рассмотренной выше функции вычисления коэффициента корреляции «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 извлечения отдельных элементов сообщения состоит в сопоставлении результата вычисления коэффициента корреляции с пороговым значением «0». В каждой строке контейнера встроено k элементов сообщения, т.е. для каждой строки k раз производим вычисление коэффициента корреляции.

В результате имеем массив данных «m1», в котором содержатся извлеченные данные. Сравним встроенные данные с извлеченными:

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 преобразования извлеченных данных в битовую форму используем процедуру:

$$
\begin{array}{l}
M\_b1 := \left| \begin{array}{l}
\text{for } i \in 0..\,\text{rows}(m1) - 1 \\
\quad \left| \begin{array}{l}
M\_b1_i \leftarrow 1 \text{ if } m1_i = 1 \\
M\_b1_i \leftarrow 0 \text{ if } m1_i = -1
\end{array} \right. \\
M\_b1
\end{array} \right.
\end{array}
$$

В результате получим битовый массив данных, сравним его со встроенным битовым массивом:

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	...

Сравнение массивов встроенных и извлеченных данных подтверждает правильность работы алгоритмов встраивания-извлечения.

Завдання 3. Проведення експериментальних досліджень ймовірнісних властивостей реалізованого методу, отримання емпіричних залежностей ймовірності правильного вилучення даних та частки внесених при цьому похибок у контейнер-зображення

3.1. Проведем оценку вероятности правильного извлечения сообщения и величины вносимых искажений как от пропускной способности стегано канала (задаваемой величиной k), так и от коэффициента усиления g.
Первую эмпирическую зависимость построим следующим образом. Зафиксируем g = 1, и при этом значении будем последовательно увеличивать величину k. Для каждого значения k рассчитаем частоту Posh ошибочно извлеченных информационных бит. Одновременно будем рассчитывать усредненную величину w внесенных искажений, выраженной в процентном соотношении к максимальному значению яркости. Используем для этого следующие процедуры:

$$\text{Posh} := \left| \begin{array}{l} a \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(M\_b1) - 1 \\ \quad a \leftarrow a + 1 \text{ if } M\_b1_i \neq M\_b_i \\ \text{Posh} \leftarrow \dfrac{a}{\text{rows}(M\_b1)} \\ \text{Posh} \end{array} \right.$$

$$w := \left| \begin{array}{l} w \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(R1) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R1) - 1 \\ \quad\quad w \leftarrow w + \left| R1_{i,j} - R_{i,j} \right| \\ w \leftarrow \dfrac{w \cdot 100}{\text{rows}(R1) \cdot \text{cols}(R1) \cdot 256} \\ w \end{array} \right.$$

Для рассматриваемого примера при g = 1 и k = 4 имеем следующие значения:

$\text{Posh} = 0.093$ $\quad$ $w = 0.586$

Полученные эмпирические данные занесем в соответствующие таблицы:

$$\text{Posh\_k} := \begin{pmatrix} 0 & 0 \\ 1 & 0.006 \\ 2 & 0.053 \\ 4 & 0.093 \\ 8 & 0.121 \\ 16 & 0.126 \\ 32 & 0.145 \\ 64 & 0.148 \\ 128 & 0.148 \\ 255 & 0.15 \end{pmatrix} \qquad \text{W\_k} := \begin{pmatrix} 0 & 0 \\ 1 & 0.39 \\ 2 & 0.39 \\ 4 & 0.586 \\ 8 & 0.871 \\ 16 & 1.244 \\ 32 & 1.723 \\ 64 & 2.385 \\ 128 & 3.286 \\ 256 & 4.5 \end{pmatrix}$$

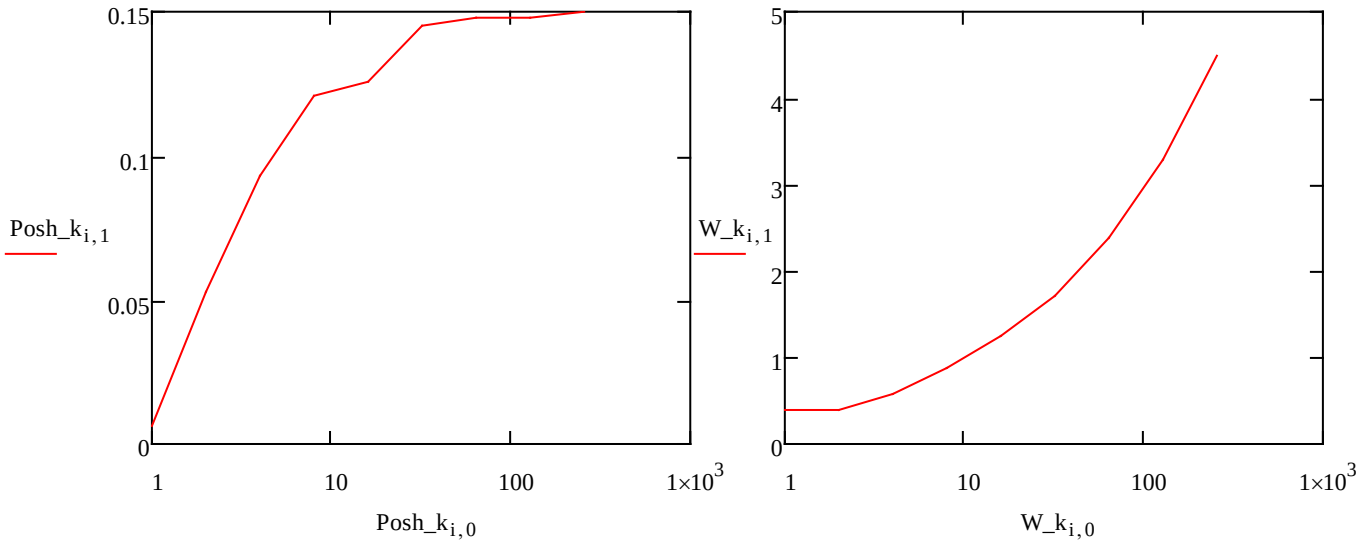
Для построения второй эмпирической зависимости зафиксируем величину k = 4 и последовательно увеличивая коэффициент g будем рассчитывать частоту Posh ошибочно извлеченных информационных бит и усредненную величину w внесенных искажений.

Полученные эмпирические данные занесем в соответствующие таблицы:

$$Posh\_g := \begin{pmatrix} 1 & 0.093 \\ 2 & 0.018 \\ 3 & 0.003 \\ 4 & 0 \end{pmatrix} \qquad W\_g := \begin{pmatrix} 1 & 0.586 \\ 2 & 1.17 \\ 3 & 1.754 \\ 4 & 2.338 \end{pmatrix}$$

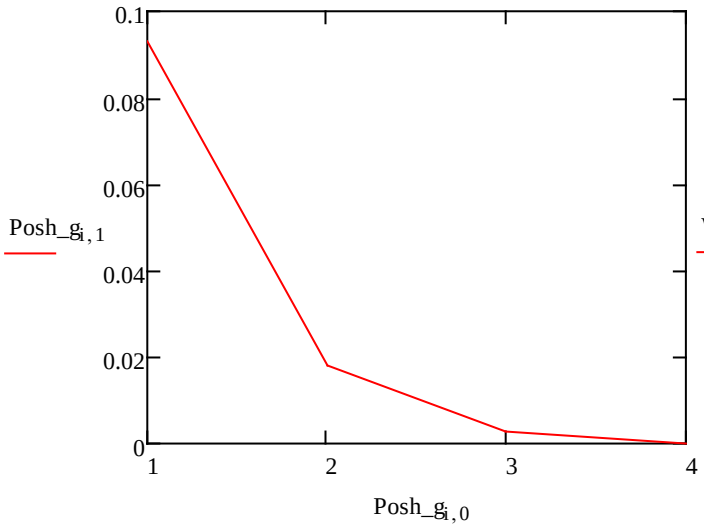
3.2. Построим графики полученных эмпирических зависимостей (для фиксированного g = 1 с изменяемым k):

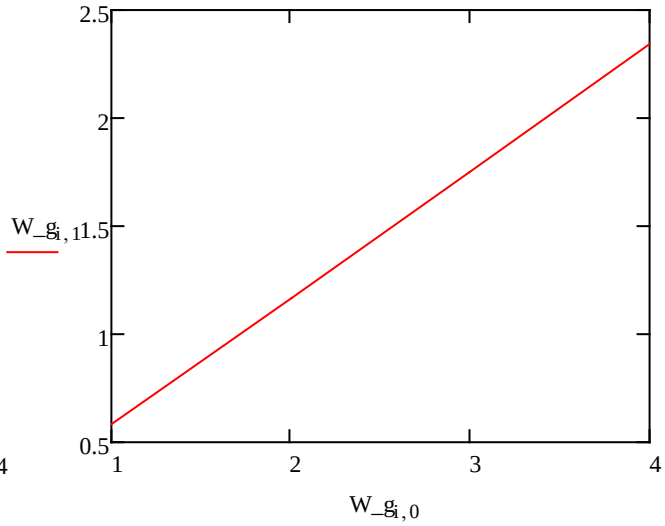
$i := 0..9$



Очевидно, что повышение числа встраиваемых бит данных приводит увеличению как вероятности ошибочного извлечения данных, так и к повышению доли вносимых искажений в контейнер-изображение. Следует отметить, что увеличение числа встраиваемых бит на один порядок (с 10 до 100 и выше) приводит к незначительному (менее 0,05) увеличению вероятности ошибочного извлечения, в то время как доля вносимых искажений увеличивается при этом в 4-5 раз.

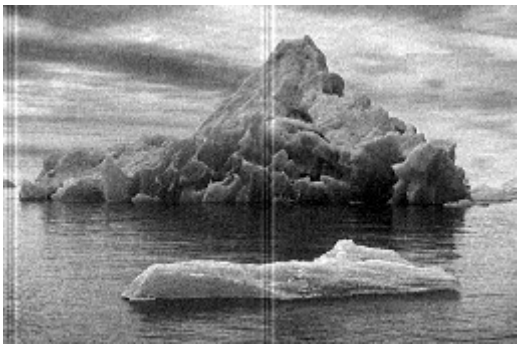
3.3. Построим графики полученных эмпирических зависимостей (для фиксированного k = 4 с изменяемым g):





Приведенные зависимости свидетельствуют, что увеличение коэффициента усиления g приводит к резкому снижению вероятности ошибочного извлечения информационных бит данных и одновременному повышению величины вносимых искажений. Из приведенных графиков следует, что при g = 4 обеспечивается безошибочное извлечение информационных данных, доля вносимых искажений в среднем лежит ниже порога зрительной чувствительности человека.

В тоже время следует отметить, что рассчитанное значение w является усредненной величиной, характеризующей долю внесенных искажений в среднем, по всем пикселям контейнера-изображения. Отдельные пиксели или группа пикселей могут быть искажены очень сильно, доля вносимых искажений для этих фрагментов изображения может существенно превышать рассчитанное усредненное значение w. Для примера приведем контейнер-изображение, заполненный с показателями: k = 128, g = 1



S

R

Как следует из приведенных выше графиков встраивание с такими параметрами дает усредненное значение доли вносимых искажений в пределах порога зрительной чувствительности человека. Однако, как видно из приведенных изображений для некоторых фрагментов искажения очень существенны. Избавиться от подобных негативных факторов возможно посредством адаптивного формирования дискретных сигналов, учуивающего

особенности используемого контейнера-изображения. Кроме того, использование при встраивании данных ортогональных дискретных сигналов Уолша-Адамара не всегда оправдано в стеганографии. Подобные сигналы на отдельных участках имеют вид детерминированных последовательностей. Например, сигнал «$ArrayFunction_0$» вовсе не использовался нами при встраивании информации, поскольку он состоит из последовательности одних единичных символов. Альтернативой использования ортогональных сигналов Уолша-Адамара являются квазиортогональные дискретные последовательности, обладающие псевдослучайной структурой и не содержащие (в идеальном случае) детерминированные участки.

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. Реализуем алгоритм формирования квазиортогональных дискретных сигналов следующим образом:

```
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 формирования отдельных элементов последовательностей используем встроенную функцию генерации псевдослучайных чисел «rnd()», которая формирует рациональное число, лежащее в заданном диапазоне.

Функция «ceil()» округляет полученный результат до ближайшего целого числа.

После преобразования «0» в «-1» получим массив «ArrayFunction1», элементами которого являются псевдослучайные последовательности – сформированные дискретные сигналы. Значение коэффициента взаимной корреляции сформированных сигналов (в силу псевдослучайности их формирования) не будет значительно отличатся от нуля, т.е. будем считать сформированное множество последовательностей ансамблем квазиортогональных дискретных сигналов.

Так, например, для первого и седьмого дискретного сигнала

$ArrayFunction1_1 =$

	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	…

коэффициент взаимной корреляции равен

$$MultString\left(ArrayFunction1_2, ArrayFunction1_7\right) = 26$$

4.2. Для встраивания информационных сообщений с использованием сформированного ансамбля квазиортогональных дискретных сигналов разобьем битовый массив «M_b» на подблоки и сформируем массив десятичных 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	…

Элементами сформированного массива «M_d» являются десятичные числа, каждое из которых в двоичном представлении соответствует подблоку из десяти бит массива «M_b».

4.3. Реализуем алгоритм кодирования квазиортогональными дискретными сигналами:

$g := 40$

$Sum1 :=$ for $i \in 0..rows(R) - 1$
$Sum1_i \leftarrow g \cdot ArrayFunction1(M\_d_i)$
$Sum1$

$Sum1_0 =$

	0
0	-40
1	-40
2	40
3	40
4	-40
5	-40
6	-40
7	-40
8	…

В результате выполнения приведенной процедуры формируем массив «Sum1», элементами которого являются усиленные коэффициентам «g» дискретные сигналы из массива «ArrayFunction1». Номера используемых сигналов соответствуют десятичному представлению информационных блоков встраиваемого сообщения.

4.4. Реализуем алгоритм встраивания информационного сообщения в контейнер-изображение посредством наложения модулированного сообщения на массив яркостей канала красного цвета:

$S1 :=$ for $i \in 0..rows(R) - 1$
for $j \in 0..cols(R) - 1$
$S1_{i,j} \leftarrow R_{i,j} + (Sum1_i)_j$
$S1_{i,j} \leftarrow 255$ if $S1_{i,j} > 255$
$S1_{i,j} \leftarrow 0$ if $S1_{i,j} < 0$
$S1$

В результате выполнения приведенной процедуры формируем массив «S1». Сравним каналы красного цвета пустого и заполненного 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

После выполнения команды записи

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

получим соответствующий контейнер-изображение

"Stego_Kvasi.bmp"

"1.bmp"

Очевидно, что встраивание с таким высоким значением коэффициента усиления (g=40) приводит к появлению существенных искажений, наглядно представленных на приведенных изображениях.

4.5. Реализуем алгоритм извлечения сообщений с использованием квазиортогональных дискретных сигналов. Для этого произведем считывание растровых данных из контейнера-изображения:

```
C2 := READRGB("Stego_Kvasi.bmp")
R2 := READ_RED("Stego_Kvasi.bmp")
G2 := READ_GREEN("Stego_Kvasi.bmp")
B2 := READ_BLUE("Stego_Kvasi.bmp")
```

В результате получим:

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

Сформируем из считанного массива красного цвета «R2» массив строк 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	…

после чего сформируем массив десятичных чисел «M_d1», элементами которого будут номера квазиотрогональных сигналов из массива «ArrayFunction1», которые дают наибольшее значение коэффициента корреляции со строками контейнера-изображения:

```
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
```

Фактически приведенная процедура реализует корреляционный прием (в терминах статистической теории связи).

Сравним извлеченный массив «M_d1» с тем массивом, который был встроен в контейнер-изображение:

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	…

Очевидно, что извлеченный массив десятичных чисел отличается от встроенного массива, что объясняется, очевидно, сильной корреляцией используемых квазиортогональных дискретных сигналов с отдельными элементами контейнера-изображения.

4.6. Преобразуем извлеченный массив «M_d1» в двоичный вид:

```
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
```

Получим массив «M_d1», сравним его со встроенным двоичным массивом «M_d»:

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. Проведем оценку вероятности правильного извлечения сообщения и величины вносимых искажений от коэффициента усиления g. Для этого рассчитаем частоту ошибочно извлеченных информационных бит и оценку вносимых искажений в контейнер-изображение:

$$Posh := \left| \begin{array}{l} a \leftarrow 0 \\ \text{for } i \in 0..rows(M\_b2) - 1 \\ \quad a \leftarrow a + 1 \text{ if } M\_b2_i \neq M\_b_i \\ Posh \leftarrow \dfrac{a}{rows(M\_b2)} \\ Posh \end{array} \right.$$

$$w := \left| \begin{array}{l} w \leftarrow 0 \\ \text{for } i \in 0..rows(R2) - 1 \\ \quad \text{for } j \in 0..cols(R2) - 1 \\ \quad\quad w \leftarrow w + \left| R2_{i,j} - R_{i,j} \right| \\ w \leftarrow \dfrac{w \cdot 100}{rows(R2) \cdot cols(R2) \cdot 256} \\ w \end{array} \right.$$

$Posh = 0.104$

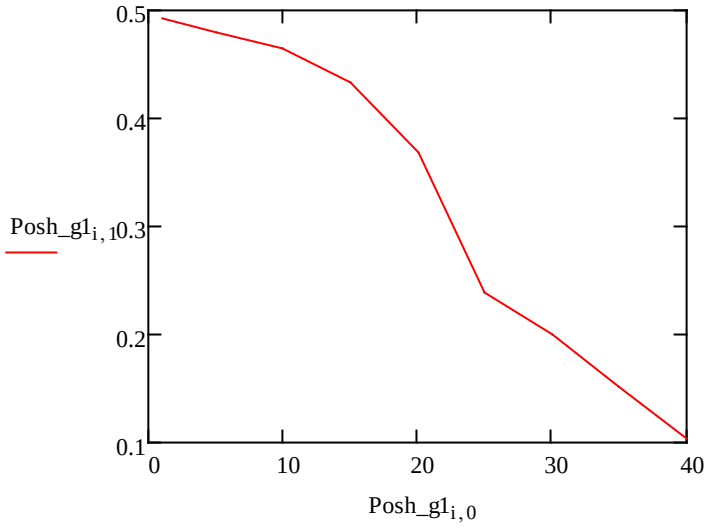
$w = 15.311$

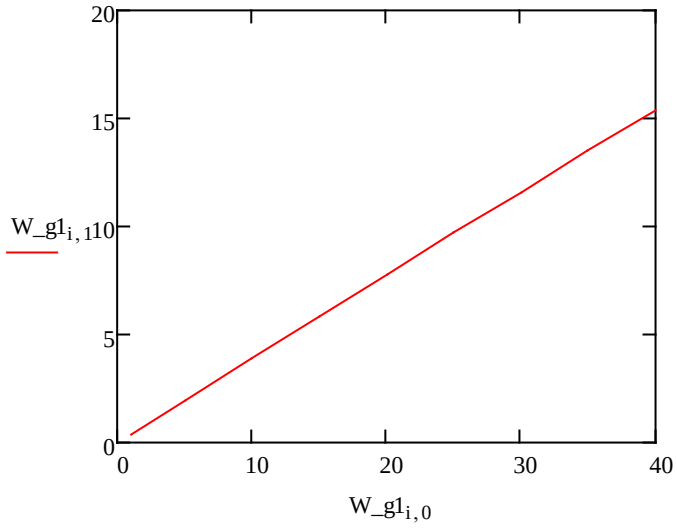
Последовательно изменяя коэффициент усиления g и выполняя процедуры встраивания и извлечения сообщения получим соответствующие эмпирические оценки, которые занесем в таблицы:

$$Posh\_g1 := \begin{pmatrix} 1 & 0.493 \\ 5 & 0.479 \\ 10 & 0.464 \\ 15 & 0.434 \\ 20 & 0.368 \\ 25 & 0.238 \\ 30 & 0.2 \\ 35 & 0.151 \\ 40 & 0.104 \end{pmatrix} \qquad W\_g1 := \begin{pmatrix} 1 & 0.39 \\ 5 & 1.951 \\ 10 & 3.897 \\ 15 & 5.838 \\ 20 & 7.771 \\ 25 & 9.692 \\ 30 & 11.596 \\ 35 & 13.473 \\ 40 & 15.331 \end{pmatrix}$$

4.8. Построим графики плаченных эмпирических оценок:

$i := 0..9$





Полученные эмпирические зависимости показывают, что повышение коэффициента усиления приводит к резкому снижению вероятности ошибочного извлечения информационных бит сообщения. Однако это также ведет к увеличению вносимых искажений в контейнер-изображение. Однако по сравнению с использование ортогональных дискретных сигналов (см. рисунки п. 3.3.) применение квазиортогональных сигналов приводит к меньшему искажению контейнера. Так, например, при встраивании k = 4 бит сообщения в одну строку контейнера с использованием ортогональных дискретных сигналов при коэффициенте усиления g = 4 величина вносимых искажений составляет более 2,33%. При большем числе вносимых бит данных (10 бит в одну строку контейнера), а следовательно и при большей пропускной способности стеганографического канала передачи данных применение квазиортогональных дискретных сигналов даже с большим значением коэффициента усиления (g = 5) приводит к меньшим искажениям контейнера, величина вносимых искажений не превосходит 2%.

Таким образом, применение квазиортогональных дискретных сигналов позволяет существенно повысить пропускную способность стеганоканалов при меньшей величине вносимых искажений. В тоже время, использование квазиортогональных дискретных сигналов существенно повышает вероятность ошибочного извлечения бит сообщения (за счет сильной коррелированности с отдельными фрагментами контейнера-изображения). Избавиться от этого негативного фактора возможно за счет адаптивного формирования квазиортогональных дискретных сигналов с учетом особенностей используемого контейнера-изображения.

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. Реализуем адаптивный алгоритм формирования квазиортогональных дискретных сигналов с учетом особенностей используемого контейнера-изображения. Для этого разобьем массив яркостей красного цвета на строки следующим образом:

```
R_Arr := | for i ∈ 0.. rows(R) − 1
         |   for j ∈ 0.. 255
         |     a_j ← R_(i,j)
         |   R_Arr_i ← a
         | R_Arr
```

Сформированный массив «R_Arr» в качестве элементов содержит строки массива яркостей красного цвета контейнера-изображения.

Алгоритм адаптивного формирования квазиортогональных дискретных сигналов представим следующей процедурой:

```
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 алгоритма состоит в формировании псевдослучайных последовательностей и вычислении коэффициента корреляции со всеми элементами массива «R_Arr», т.е. со всеми строками контейнера. Если коэффициент корреляции для всех строк контейнера не превосходит заранее заданной величины (в данном случае значения 1024) сформированная последовательность используется в качестве квазиортогонального дискретного сигнала. Если коэффициент корреляции для какой-либо строки контейнера превзойдет заданное значение сформированная последовательность бракуется и формируется другая последовательность. Очевидно, что время формирования ансамбля дискретных сигналов зависит от пороговой величины, с которой сравнивается значение коэффициента корреляции. При ее уменьшении резко возрастают временные затраты на формирование ансамбля сигнала, однако малое граничное значение обеспечит слабую коррелированность формируемых квазиортогональных дискретных сигналов с отдельными фрагментами контейнера-изображения. Для примера приведем один из дискретных сигналов и значение коэффициента корреляции с одой из строк 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 встраивания сообщения воспользуемся следующей процедурой:

$g := 9$

```
Sum2 := | for i ∈ 0.. rows(R) − 1
        |   Sum2_i ← g·ArrayFunction2(M_d_i)
        | Sum2
```

Сформированный массив «Sum2» в качестве элементов содержит модулированное квазиортогональными сигналами сообщение. Для его встраивания выполним наложение массива «Sum2» на контейнер-изображение:

```
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
```

Получим заполненный контейнер, сравним его с исходным:

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

Запишем сформированный контейнер в файл и просмотрим результат:

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

"Stego_Kvasi_ad.bmp"

"1.bmp"

Как видно из приведенных рисунков сформированный контейнер практически неотличим от исходного. Тем не менее в него встроено более 1600 бит информационного сообщения.

5.3. Для извлечения информационного сообщения считаем данные 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

Сформируем массив строк заполненного контейнера:

```
ArrayString2 := | for i ∈ 0.. rows(R3) − 1
                |     for j ∈ 0.. cols(R3) − 1
                |       a_j ← R3_(i,j)
                |     ArrayString2_i ← a
                | ArrayString2
```

и извлечем встроенное сообщение в виде десятичного массива данных:

```
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
```

Полученный результат сравним с массивом встроенных данных:

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	...

Очевидно, что использование адаптивно формируемых дискретных сигналов позволило существенно повысить вероятность правильного извлечения сообщений.

5.4. Преобразуем извлеченный массив данных в двоичный вид и сравним полученный результат с теми двоичными данными, которые были встроены 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. Проведем оценку вероятности правильного извлечения сообщения и величины вносимых искажений от коэффициента усиления g. Для этого рассчитаем частоту ошибочно извлеченных информационных бит и оценку вносимых искажений в контейнер-изображение:

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

$$w := \left| \begin{array}{l} w \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(R3) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R3) - 1 \\ \quad\quad w \leftarrow w + \left| R3_{i,j} - R_{i,j} \right| \\ w \leftarrow \dfrac{w \cdot 100}{\text{rows}(R3) \cdot \text{cols}(R3) \cdot 256} \\ w \end{array} \right.$$

Posh = 0

w = 3.508

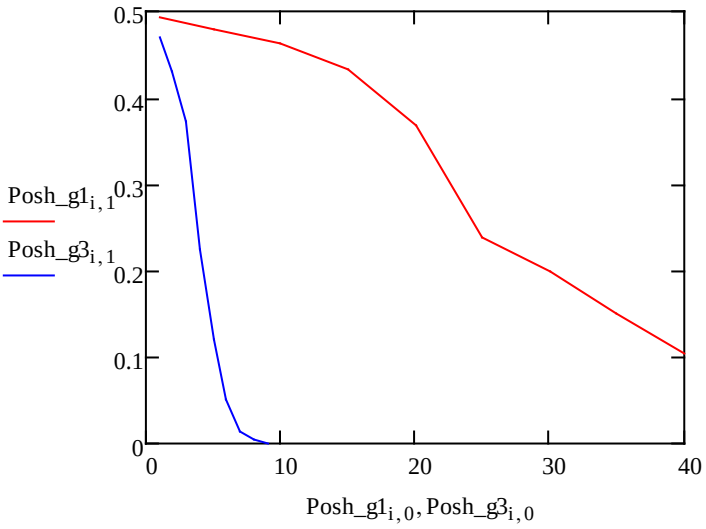
Последовательно изменяя коэффициент усиления g и выполняя процедуры встраивания и извлечения сообщения получим соответствующие эмпирические оценки, которые занесем в таблицы:

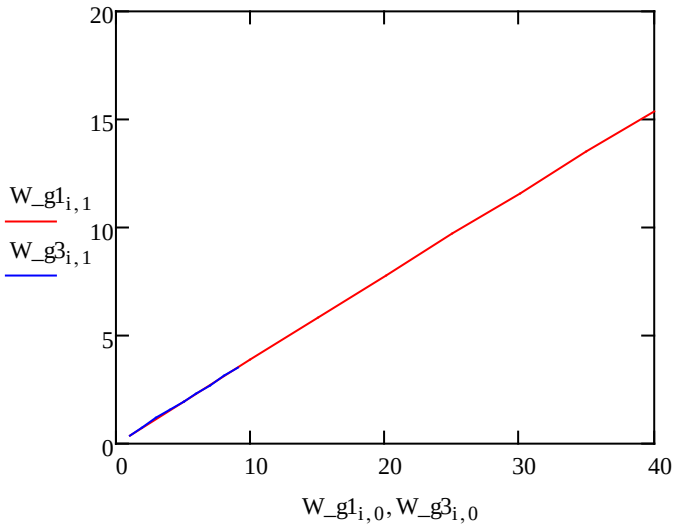
$$Posh\_g3 := \begin{pmatrix} 1 & 0.47 \\ 2 & 0.43 \\ 3 & 0.373 \\ 4 & 0.224 \\ 5 & 0.121 \\ 6 & 0.051 \\ 7 & 0.015 \\ 8 & 0.005 \\ 9 & 0 \end{pmatrix}$$

$$W\_g3 := \begin{pmatrix} 1 & 0.39 \\ 2 & 0.781 \\ 3 & 1.171 \\ 4 & 1.561 \\ 5 & 1.951 \\ 6 & 2.34 \\ 7 & 2.73 \\ 8 & 3.119 \\ 9 & 3.508 \end{pmatrix}$$

5.6. Построим графики плаченных эмпирических оценок и сравним их с полученными ранее зависимостями для случая использования квазиортогональных дискретных сигналов без адаптации к используемому контейнеру:

$$i := 0..9$$





Синим цветом на графиках отображены результаты моделирования стеганосистем с адаптивным формированием сигналов, красным – без адаптации. Очевидно, что без увеличения величины вносимых искажений удалось существенно снизить вероятность ошибочного извлечения информационных бит сообщений. По сравнению с использованием ортогональных дискретных сигналов удалось существенно увеличить пропускную способность стеганоканала при сравнимых вносимых искажениях. В качестве примера рисунках приведем следующие контейнеры:

S

S2

R

Первый контейнер заполнен с использованием ортогональных дискретных сигналов с параметрами «k = 4» и «g = 4», т.е. в каждую строку контейнера встроено четыре бита, всего в контейнер встроено 676 бит информации. Эти параметры обеспечивают практически безошибочное извлечение сообщения, однако максимальная величина вносимых искажений в отельные пиксели изображения составляет k*g=16 уровней яркости.

Второй контейнер заполнен с использованием адаптивно формируемых с учетом свойств контейнера-изображения квазиортогональных дискретных сигналов. При этом использован коэффициент усиления «g = 9», который также обеспечивает практически безошибочное извлечение информационного сообщения, однако максимальная величина вносимых искажений в отельные пиксели изображения составляет g=9 уровней яркости. И хотя усредненное значение вносимых искажений для ортогональных сигналов несколько ниже, их абсолютное значение существенно (почти в два раза) может превосходить аналогичный показатель для адаптивно формируемых квазиортогональных сигналов. Влияние снижение максимального уровня вносимых искажений на отдельные пиксели изображения визуально заметно на приведенных рисунках. Третий контейнер представляет собой не модифицированный (пустой) контейнер.

Таким образом, как следует из полученных результатов применение адаптивно формируемых квазиортогональных дискретных сигналов позволяет существенно повысить пропускную способность стеганоканала при сравнимых искажениях, вносимых в контейнер-изображение. Вносимые искажения можно еще более снизить, уменьшив численный порог, ограничивающий коэффициент взаимной корреляции в алгоритме адаптивного формирования дискретных сигналов.

6. Приклад оформлення звіту з лабораторної роботи

Лабораторная работа №3
Встраивание данных в пространственную область неподвижных изображений на основе прямого расширения спектра.

1.

"Picture.bmp"

$C := READRGB("Picture.bmp")$

$R := READ\_RED("Picture.bmp")$

$G := READ\_GREEN("Picture.bmp")$

$B := READ\_BLUE("Picture.bmp")$

$M := READBIN("Text.txt","byte")$

Функуия преобразования сообщени
з двоичного вида в десятичный

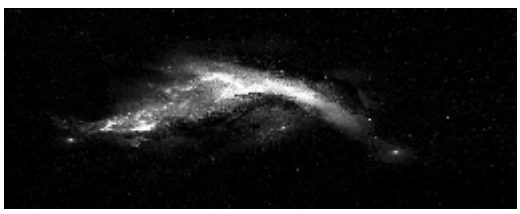
$$B2D(x) := \sum_{i=0}^{7} \left(x_i \cdot 2^i \right)$$

Функция преобразования сообщения их десятичного вида в двоинный:

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

Функция преобразования с десятичного массива Мв д воичный:

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



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	...

Формирование матриц 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	...

Массив ортогональных функций

```
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	...

Преобразуем битовое сообщение:

$m :=$ for $i \in 0..\text{rows}(M\_b) - 1$: $m_i \leftarrow 1$ if $M\_b_i = 1$; $m_i \leftarrow -1$ if $M\_b_i = 0$; m

$m2b(m) :=$ for $i \in 0..\text{rows}(m) - 1$: $m1_i \leftarrow 1$ if $m_i = 1$; $m1_i \leftarrow 0$ if $m_i = -1$; $m1$

Модулируем каждый информационный бит с помощью ПСП длинны 256 бит:

$$Sum := \text{for } i \in 0..\text{rows}(R) - 1 \quad a \leftarrow \sum_{j=0}^{k-1} \left[g \cdot \left(m_{k \cdot i + j} \cdot \text{ArrayFunction}_{j+1} \right) \right] \quad Sum_i \leftarrow a \quad Sum$$

Наложение модулированного сообщение на контейнер:

$S :=$ for $i \in 0..\text{rows}(R) - 1$; for $j \in 0..\text{cols}(R) - 1$: $S_{i,j} \leftarrow R_{i,j} + (Sum_i)_j$; $S_{i,j} \leftarrow 255$ if $S_{i,j} > 255$; $S_{i,j} \leftarrow 0$ if $S_{i,j} < 0$; S

Пустой и заполненный контейнеры:

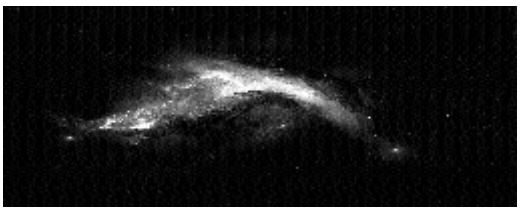
$R =$

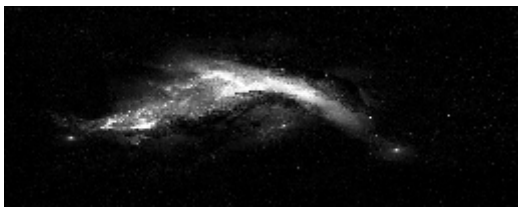
	0	1	2	3	4	5
0	10	5	6	4	1	8
1	8	0	7	23	1	19
2	3	7	4	2	0	7
3	0	11	3	6	3	4
4	6	0	5	2	8	0
5	4	0	2	2	1	6
6	9	0	4	2	5	1
7	8	4	0	1	1	4
8	3	3	0	1	1	6
9	0	3	1	3	5	2
10	3	12	2	0	6	...

$S =$

	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"

Функция расчета коэффициента корреляции:

$$MultString(A,B) := \left| \begin{array}{l} X \leftarrow 0 \\ \text{for } i \in 0..255 \\ \quad X \leftarrow X + A_i \cdot B_i \\ X \end{array} \right.$$

По результатам исчеслений видно что сообщение корелировано с ПСП (1,2,3,4), а сами ПСП между собой не корелированы.

$MultString(ArrayFunction_2, ArrayFunction_3) = 0$

$MultString(Sum_0, ArrayFunction_1) = -1.024 \times 10^3$

$MultString(Sum_0, ArrayFunction_2) = -1.024 \times 10^3$

$MultString(Sum_0, ArrayFunction_3) = 1.024 \times 10^3$

$MultString(Sum_0, ArrayFunction_4) = -1.024 \times 10^3$

$MultString(Sum_0, ArrayFunction_5) = 0$

Извлечение

$C1 := READRGB("Stego1.bmp")$

$G1 := READ\_GREEN("Stego1.bmp")$

$R1 := READ\_RED("Stego1.bmp")$

$B1 := READ\_BLUE("Stego1.bmp")$

Массив строк контейнера

$$ArrayString := \left| \begin{array}{l} \text{for } i \in 0..rows(R1)-1 \\ \quad \left| \begin{array}{l} \text{for } j \in 0..cols(R1)-1 \\ \quad a_j \leftarrow R1_{i,j} \\ ArrayString_i \leftarrow a \end{array} \right. \\ ArrayString \end{array} \right.$$

Сравнение извлеченного и встроенного сообщения

$$m1 := \left| \begin{array}{l} \text{for } i \in 0..rows(R1)-1 \\ \quad \text{for } j \in 0..k-1 \\ \quad \left| \begin{array}{l} m1_{k \cdot i+j} \leftarrow 1 \text{ if } MultString(ArrayString_i, ArrayFunction_{j+1}) > 0 \\ m1_{k \cdot i+j} \leftarrow -1 \text{ if } MultString(ArrayString_i, ArrayFunction_{j+1}) \le 0 \end{array} \right. \\ m1 \end{array} \right.$$

Преобразуем сообщение в бинарный 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

Расчет вероятности ошибочного извлечения информационных бит:

$Posh :=$
$a \leftarrow 0$
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

$Posh = 0$

$W\_k := \begin{pmatrix} 0 & 0 \\ 1 & 0.39 \\ 2 & 0.39 \\ 4 & 0.586 \\ 8 & 0.871 \\ 16 & 1.244 \\ 32 & 1.723 \\ 64 & 2.385 \\ 128 & 3.286 \\ 256 & 4.5 \end{pmatrix}$

$Posh\_k := \begin{pmatrix} 0 & 0 \\ 1 & 0.006 \\ 2 & 0.053 \\ 4 & 0.093 \\ 8 & 0.121 \\ 16 & 0.126 \\ 32 & 0.145 \\ 64 & 0.148 \\ 128 & 0.148 \\ 255 & 0.15 \end{pmatrix}$

$Posh\_g := \begin{pmatrix} 1 & 0.093 \\ 2 & 0.018 \\ 3 & 0.003 \\ 4 & 0 \end{pmatrix}$

$W\_g := \begin{pmatrix} 1 & 0.586 \\ 2 & 1.17 \\ 3 & 1.754 \\ 4 & 2.338 \end{pmatrix}$

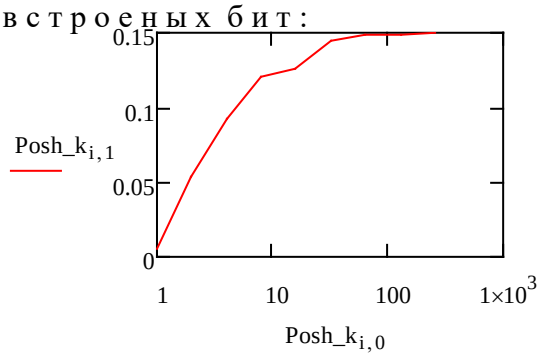
Расчет доли внесенных искажений в контейнер-изображение:

$w :=$
$w \leftarrow 0$
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

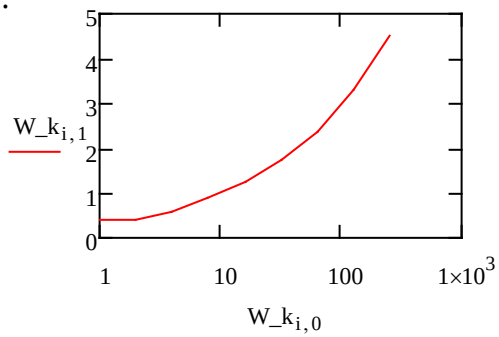
$w = 1.699$

$i := 0..9$

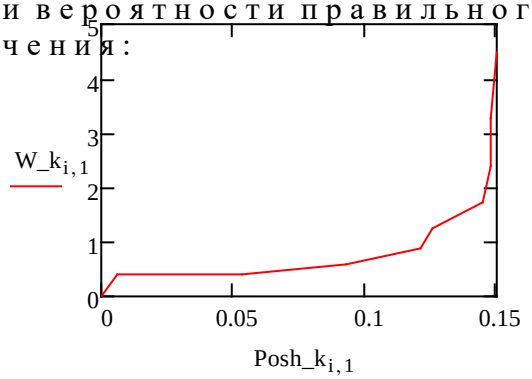
Вероятность ошибки от количесива встроеных бит:



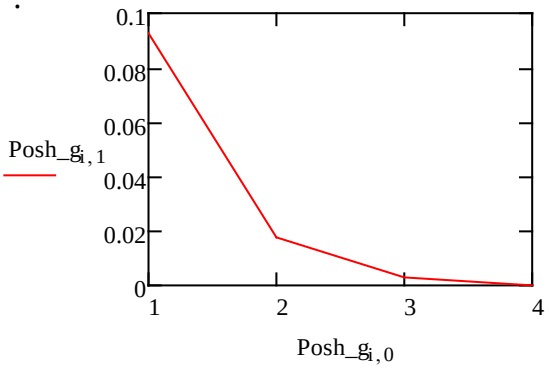
Коефициент вносимых искажений от еоличества встроеных бит:



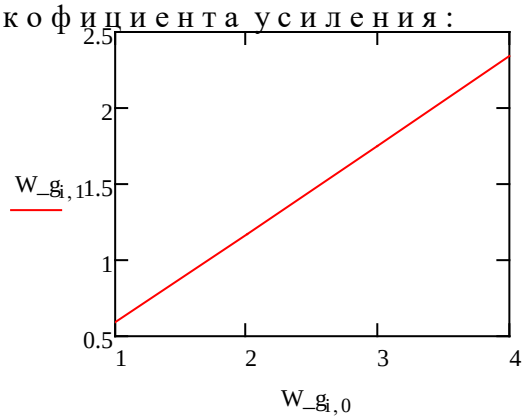
Зависимость вносимых искажений и вероятности правильного извлечения:



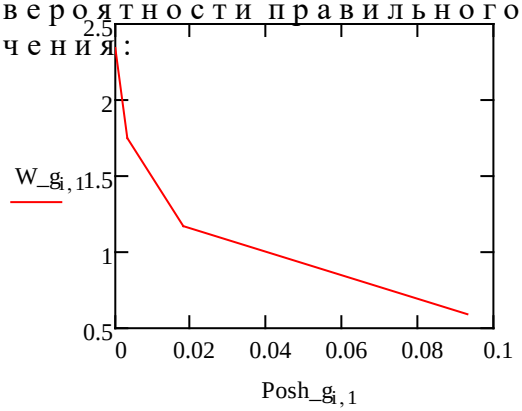
Зависимость правильного извлечения от коефициента усиления:



Зависимость вносимых искажений от коефициента усиления:



Зависимость вносимых искажений и вероятности правильного извлечения:



Многоосновное стеганографическое кодирование

Формируем квазиортогональные дискретные сигналы

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

$MultString(ArrayFunction1_1, ArrayFunction1_2) = 8$

$MultString(ArrayFunction1_2, ArrayFunction1_3) = 10$

$MultString(ArrayFunction1_1, ArrayFunction1_3) = 18$

Разбиваем сообщение на блоки по 10 бит и формируем десятичный массив данных

M_d :=
for i ∈ 0.. rows(R) − 1
  a ← 0
  for j ∈ 0.. 9
    $a \leftarrow a + M\_b_{10\cdot i+j}\cdot 2^j$
  M_d_i ← a
M_d

Заменяем блок из 10 бит на соответсвующую ему ПСП

g := 4

Sum1 :=
for i ∈ 0.. rows(R) − 1
  $Sum1_i \leftarrow g\cdot ArrayFunction1_{(M\_d_i)}$
Sum1

Встраиваем полученое промодулированое сообщение в контейнер:

S1 :=
for i ∈ 0.. rows(R) − 1
  for j ∈ 0.. cols(R) − 1
    $S1_{i,j} \leftarrow R_{i,j} + (Sum1_i)_j$
    $S1_{i,j} \leftarrow 255$ if $S1_{i,j} > 255$
    $S1_{i,j} \leftarrow 0$ if $S1_{i,j} < 0$
S1

Многоосновное стеганографическое кодирование

Формируем квазиортогональные дискретные сигналы

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

$MultString(ArrayFunction1_1, ArrayFunction1_2) = 8$

$MultString(ArrayFunction1_2, ArrayFunction1_3) = 10$

$MultString(ArrayFunction1_1, ArrayFunction1_3) = 18$

Разбиваем сообщение на блоки по 10 бит и формируем десятичный массив данных

M_d :=
for i ∈ 0.. rows(R) − 1
  a ← 0
  for j ∈ 0.. 9
    $a \leftarrow a + M\_b_{10\cdot i+j}\cdot 2^j$
  M_d_i ← a
M_d

Заменяем блок из 10 бит на соответсвующую ему ПСП

g := 4

Sum1 :=
for i ∈ 0.. rows(R) − 1
  $Sum1_i \leftarrow g\cdot ArrayFunction1_{(M\_d_i)}$
Sum1

Встраиваем полученое промодулированое сообщение в контейнер:

S1 :=
for i ∈ 0.. rows(R) − 1
  for j ∈ 0.. cols(R) − 1
    $S1_{i,j} \leftarrow R_{i,j} + (Sum1_i)_j$
    $S1_{i,j} \leftarrow 255$ if $S1_{i,j} > 255$
    $S1_{i,j} \leftarrow 0$ if $S1_{i,j} < 0$
S1

Преобразуем извлеченное сообщение в битовый вид

$$M\_b2 := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(M\_d1) - 1 \\ \quad \left| \begin{array}{l} x \leftarrow M\_d1_i \\ \text{for } j \in 0..9 \\ \quad \left| \begin{array}{l} M\_b2_{i \cdot 10 + j} \leftarrow \text{mod}(x, 2) \\ x \leftarrow \text{floor}\left(\frac{x}{2}\right) \end{array} \right. \end{array} \right. \\ M\_b2 \end{array} \right.$$

$$M2 := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(M\_b2) - 8 \\ \quad \left| \begin{array}{l} l \leftarrow \text{floor}\left(\frac{i}{8}\right) \\ \text{for } j \in 0..7 \\ \quad \left| \begin{array}{l} V_j \leftarrow M\_b2_{l \cdot 8 + j} \end{array} \right. \\ M_l \leftarrow B2D(V) \end{array} \right. \\ M \end{array} \right.$$

WRITEBIN("STEGOTXT2.txt" , "byte" , 1) := M2

Расчет вероятности ошибочного извлечения информационных бит:

$$Posh := \left| \begin{array}{l} a \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(M\_b2) - 1 \\ \quad a \leftarrow a + 1 \text{ if } M\_b2_i \neq M\_b_i \\ Posh \leftarrow \frac{a}{\text{rows}(M\_b2)} \\ Posh \end{array} \right.$$

$$Posh\_g1 := \begin{pmatrix} 1 & 0.493 \\ 5 & 0.479 \\ 10 & 0.464 \\ 15 & 0.434 \\ 20 & 0.368 \\ 25 & 0.238 \\ 30 & 0.2 \\ 35 & 0.151 \\ 40 & 0.104 \end{pmatrix}$$

Posh = 0.199

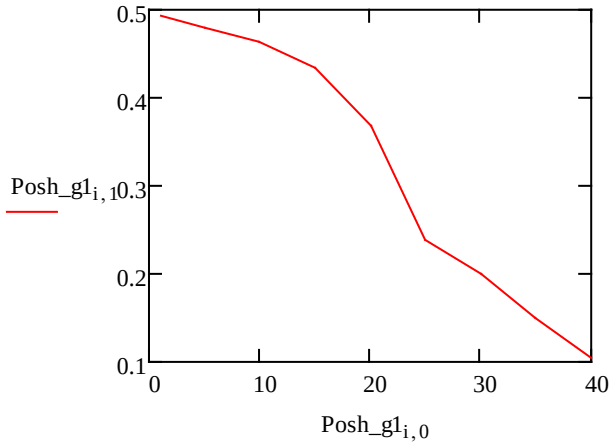
Расчет доли внесенных искажений в контейнер-изображение:

$$w := \left| \begin{array}{l} w \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(R2) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R2) - 1 \\ \quad\quad w \leftarrow w + \left| R2_{i,j} - R_{i,j} \right| \\ w \leftarrow \frac{w \cdot 100}{\text{rows}(R2) \cdot \text{cols}(R2) \cdot 256} \\ w \end{array} \right.$$

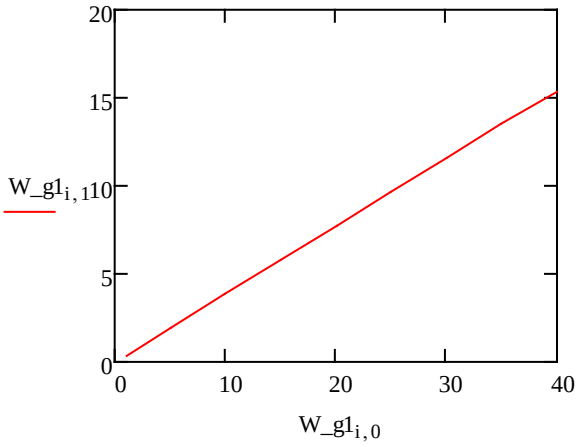
$$W\_g1 := \begin{pmatrix} 1 & 0.39 \\ 5 & 1.951 \\ 10 & 3.897 \\ 15 & 5.838 \\ 20 & 7.771 \\ 25 & 9.692 \\ 30 & 11.596 \\ 35 & 13.473 \\ 40 & 15.331 \end{pmatrix}$$

w = 1.286

Зависимость вероятности ошибки от коэффициента усиления:



Зависимость вносимых искажен
от коефициента усиления:



Адаптивое формирование дискретных сигналов:

```
R_Arr := | for i ∈ 0.. rows(R) − 1
         |     for j ∈ 0.. 255
         |         a_j ← R_(i,j)
         |     R_Arr_i ← a
         | R_Arr
```

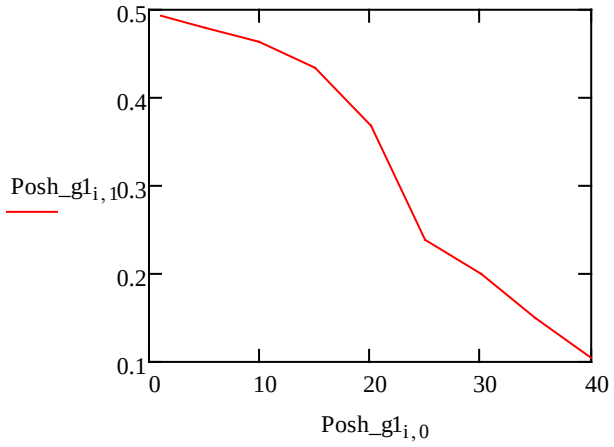
Адаптивно формируем квазиортогональные дискретные сигналы

```
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
```

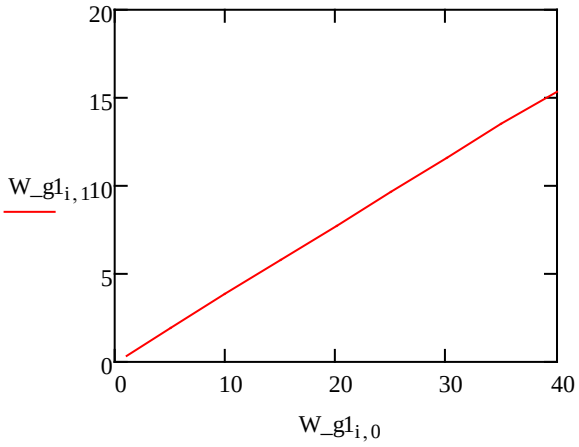
$MultString(R\_Arr_2, ArrayFunction2_2) = 29$

$MultString(ArrayFunction_1, ArrayFunction2_2) = -2$

Зависимость вероятности ошибки от коэффициента усиления:



Зависимость вносимых искажен
от коефициента усиления:



Адаптивое формирование дискретных сигналов:

```
R_Arr := | for i ∈ 0.. rows(R) − 1
         |     for j ∈ 0.. 255
         |         a_j ← R_(i,j)
         |     R_Arr_i ← a
         | R_Arr
```

Адаптивно формируем квазиортогональные дискретные сигналы

```
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
```

$MultString(R\_Arr_2, ArrayFunction2_2) = 29$

$MultString(ArrayFunction_1, ArrayFunction2_2) = -2$

Извлекаем блоки по 10 бит из строк 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
```

Преобразуем извлеченное сообщение в битовый вид

```
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

Расчет вероятности ошибочного извлечения информационных бит:

```
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

$$Posh\_g3 := \begin{pmatrix} 1 & 0.47 \\ 2 & 0.43 \\ 3 & 0.373 \\ 4 & 0.224 \\ 5 & 0.121 \\ 6 & 0.051 \\ 7 & 0.015 \\ 8 & 0.005 \\ 9 & 0 \end{pmatrix}$$

$$Posh\_g := \begin{pmatrix} 1 & 0.093 \\ 2 & 0.018 \\ 3 & 0.003 \\ 4 & 0 \end{pmatrix}$$

Расчет доли внесенных искажений в контейнер-изображение:

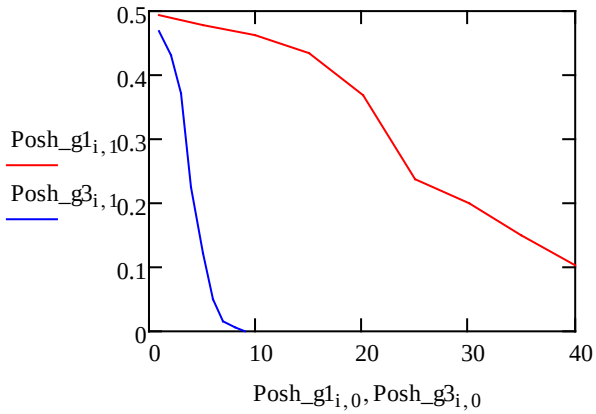
$$w := \left| \begin{array}{l} w \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(R3) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R3) - 1 \\ \quad\quad w \leftarrow w + \left| R3_{i,j} - R_{i,j} \right| \\ w \leftarrow \dfrac{w \cdot 100}{\text{rows}(R3) \cdot \text{cols}(R3) \cdot 256} \\ w \end{array} \right.$$

$w = 1.285$

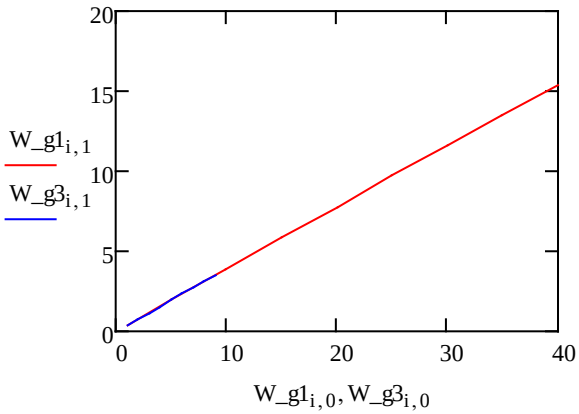
$$W\_g3 := \begin{pmatrix} 1 & 0.39 \\ 2 & 0.781 \\ 3 & 1.171 \\ 4 & 1.561 \\ 5 & 1.951 \\ 6 & 2.34 \\ 7 & 2.73 \\ 8 & 3.119 \\ 9 & 3.508 \end{pmatrix}$$

$$W\_g := \begin{pmatrix} 1 & 0.586 \\ 2 & 1.17 \\ 3 & 1.754 \\ 4 & 2.338 \end{pmatrix}$$

Зависимость ошибки извлечения от коефициента усиления:



Зависимость вносимых искажений от коефициента усиления:



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

…

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. Руководство к выполнению лабораторной работы №4

Задание 1. Реализация в среде MathCAD алгоритмов прямого и обратного дискретно-косинусного преобразования. Исследование эффекта частотной чувственности зрительной системы человека

1.1. Загружаем исходные данные: контейнер - неподвижное изображение (в формате *.bmp24); информационное сообщение – текстовый документ (в формате *.txt). Для этого в среде MathCAD выполняем действия, аналогичные описанным в п. 1.1. руководства к лабораторной работе №1.

1.2. Преобразуем массив информационных данных. Для этого в среде MathCAD выполняем действия, аналогичные описанным в п. 1.2. руководства к лабораторной работе №1.

1.3. Реализуем алгоритмы прямого и обратного дискретно-косинусного преобразования. Для этого в среде MathCAD выполняем последовательность преобразований, представленных на рис. 4.1.

$N := 8 \quad P := 1$

$C :=$ for $i \in 0..N-1$
  $C_i \leftarrow \frac{1}{\sqrt{2}}$ if $i = 0$
  $C_i \leftarrow 1$ if $i > 0$
  C

$$C = \begin{pmatrix} 0.707 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$T(V) :=$ for $i \in 0..N-1$
  for $j \in 0..N-1$
    $T_{i,j} \leftarrow \text{round}\left[\frac{2}{N} \cdot C_i C_j \cdot \frac{1}{P} \cdot \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \left[V_{x,y} \cdot \cos\left[\frac{(2x+1) \cdot i \cdot \pi}{2N}\right] \cdot \cos\left[\frac{(2y+1) \cdot j \cdot \pi}{2N}\right]\right]\right]$
  T

$V(T) :=$ for $i \in 0..N-1$
  for $j \in 0..N-1$
    $V_{i,j} \leftarrow \text{round}\left[\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \left[T_{x,y} \cdot P \cdot \frac{2}{N} \cdot C_x C_y \cdot \cos\left[\frac{(2i+1) \cdot x \cdot \pi}{2N}\right] \cdot \cos\left[\frac{(2j+1) \cdot y \cdot \pi}{2N}\right]\right]\right]$
  V

Рис. 4.1 – Реализация дискретно-косинусного преобразования в среде символьной математики MathCAD

На рис. 4.1 приведены следующие элементы.

Переменная N задает размер матрицы, над которой выполняется преобразование, переменная P задает величину порога загрубления. Если P = 1, тогда загрубление не производится. Используем значение N = 8, т.к. такой параметр использует алгоритм сжатия JPEG.

Переменная C содержит служебный массив данных из восьми элементов, необходимых для корректного вычисления дискретно-косинусного преобразования.

Функция T(V) реализует прямое дискретно-косинусное преобразование массива V из NxN чисел. В качестве аргумента функции T(V) выступают отдельные блоки растровых данных в пространственной области.

Функция V(T) реализует обратное дискретно-косинусное преобразование массива T из NxN чисел. В качестве аргумента функции V(T) выступают отдельные блоки растровых данных в частотной области. Переменная P задает величину порога загрубления коэффициентов дискретно-косинусного преобразования.

1.4. Разобьем исходное изображение на блоки, размером NxN пикселей каждый, выполним прямое дискретно-косинусное преобразование для каждого блока изображения. Для этого в среде MathCAD выполняем последовательность преобразований, представленных 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
```

Рис. 4.2 – Разбиение контейнера-изображения на блоки и выполнение над ними дискретно-косинусного преобразования

Величины mn и mm задают число блоков, на которые разбито изображение. В массиве r будут содержаться блоки изображения размером NxN пикселей в пространственной области, а в массиве tr будут хранится те же блоки, но уже в частотной области, т.е. это массивы коэффициентов дискретно-косинусного преобразования. Например, для блока с номерами 1,2 имеем значения, приведенные на рис. 4.3.

$$r_{1,2} = \begin{pmatrix} 24 & 24 & 23 & 22 & 21 & 20 & 20 & 20 \\ 23 & 22 & 21 & 21 & 19 & 19 & 18 & 17 \\ 24 & 22 & 20 & 20 & 20 & 19 & 18 & 16 \\ 25 & 23 & 22 & 21 & 21 & 20 & 19 & 18 \\ 24 & 24 & 23 & 22 & 22 & 21 & 20 & 19 \\ 23 & 23 & 23 & 22 & 21 & 21 & 20 & 20 \\ 22 & 22 & 21 & 21 & 20 & 20 & 20 & 19 \\ 24 & 22 & 21 & 21 & 21 & 21 & 20 & 19 \end{pmatrix} \qquad tr_{1,2} = \begin{pmatrix} 168 & 13 & 0 & 2 & 0 & 1 & 0 & 0 \\ -1 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ 4 & -1 & 0 & -3 & -1 & -1 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{pmatrix}$$

Рис. 4.3 – Пример блоков контейнера-изображения в пространственной и частотной области

Значения массива $r_{1,2}$ (см. рис. 4.3) характеризуют величину яркости (красного цвета) отдельных пикселей изображения, а значения массива $tr_{1,2}$ характеризуют величину отдельных коэффициентов дискретно-косинусного преобразования, вычисленных для блока $r_{1,2}$. Левая верхняя часть массива $tr_{1,2}$ соответствует низкочастотной области изображения, именно здесь сосредоточена основная «энергия» реалистичных изображений, что наглядно видно по значениям массива $tr_{1,2}$. Правая нижняя часть соответствует высокочастотной области, значения коэффициентов дискретно-косинусного преобразования в которой характеризуют высокочастотную, контрастную часть изображения. Для реалистичных изображений высокочастотная область содержит низкие по абсолютной величине значения, что наглядно видно на рис. 4.3.

1.5. Для исследования эффекта частотной чувственности зрительной системы человека изменим величины коэффициентов дискретно-косинусного преобразования в низкочастотной и высокочастотной области изображения. Внесенные изменения будем оценивать визуально, для чего выполним обратное дискретно-косинусное преобразование над измененным массивом коэффициентов. Для этого, например, выполним преобразования, приведенные на рис. 4.4.

Суть преобразований, приведенных на рис. 4.4, следующая. Для блока с номерами 7,7 на 30% увеличен коэффициент дискретно-косинусного преобразования с индексом (0,0). С использованием описанной в п. 1.3 функции V(T) для исходного и измененного массива коэффициентов выполнено обратное дискретно-косинусное преобразование. Результат изменения яркости пикселей (в увеличенном масштабе показаны два изображения 8x8 пикселей) показывает, что внесенные искажения визуально обнаруживаются (общий фон рисунка слева значительно темнее рисунка справа). Таким образом, даже незначительное (в пределах 30%) искажение низкочастотных коэффициентов дискретно-косинусного преобразования приводит к внесению видимых искажений, общий фон изменяется, что хорошо обнаруживается визуальным осмотром.

$A := tr_{7,7}$ $B := A$

$B_{0,0} := A_{0,0} + \text{floor}\left(0.3 \cdot A_{0,0}\right)$

$$A = \begin{pmatrix} 444 & 11 & 0 & 0 & 0 & 1 & -1 & 1 \\ -32 & 2 & -2 & 2 & 0 & 0 & 1 & 1 \\ -3 & 0 & -2 & 0 & 1 & 1 & 1 & 0 \\ -7 & -2 & 1 & -1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & -1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 577 & 11 & 0 & 0 & 0 & 1 & -1 & 1 \\ -32 & 2 & -2 & 2 & 0 & 0 & 1 & 1 \\ -3 & 0 & -2 & 0 & 1 & 1 & 1 & 0 \\ -7 & -2 & 1 & -1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & -1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$a := V(A)$ $b := V(B)$

$$a = \begin{pmatrix} 50 & 48 & 50 & 48 & 49 & 48 & 47 & 45 \\ 54 & 53 & 53 & 53 & 52 & 50 & 49 & 48 \\ 56 & 56 & 54 & 54 & 54 & 53 & 52 & 50 \\ 58 & 58 & 56 & 55 & 55 & 53 & 54 & 53 \\ 58 & 58 & 58 & 56 & 55 & 55 & 55 & 56 \\ 59 & 60 & 59 & 58 & 57 & 56 & 58 & 57 \\ 61 & 60 & 60 & 60 & 59 & 57 & 59 & 59 \\ 63 & 63 & 63 & 63 & 61 & 60 & 60 & 60 \end{pmatrix} \qquad b = \begin{pmatrix} 67 & 64 & 67 & 65 & 66 & 64 & 63 & 62 \\ 71 & 70 & 70 & 70 & 68 & 66 & 65 & 65 \\ 73 & 73 & 71 & 71 & 71 & 70 & 69 & 66 \\ 74 & 75 & 72 & 72 & 72 & 70 & 71 & 69 \\ 75 & 75 & 75 & 73 & 72 & 72 & 72 & 73 \\ 76 & 77 & 75 & 74 & 74 & 73 & 75 & 74 \\ 77 & 77 & 77 & 77 & 76 & 74 & 76 & 75 \\ 80 & 80 & 79 & 80 & 77 & 77 & 77 & 76 \end{pmatrix}$$

Рис. 4.4 – Демонстрация эффекта высокой частотной чувственности зрительной системы человека к незначительному изменению низкочастотных коэффициентов изображения

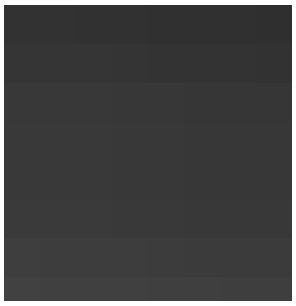
Для рассмотренного примера внесем также изменения в коэффициент дискретно-косинусного преобразования с индексом (7,6). Это высокочастотный коэффициент и, согласно теоретическим сведениям, чувствительность зрительной системы человека к таким изменениям очень низкая. Увеличим выбранный коэффициент дискретно-косинусного на 100% и проведем аналогичные преобразования (см. рис. 4.5). Как видно из приведенных данных даже после внесения значительных изменений высокочастотного коэффициента (увеличение на 100%) искажения визуально не обнаруживаются.

$$B_{7,6} := A_{7,6} + \mathrm{floor}\left(A_{7,6}\right)$$

$$A = \begin{pmatrix} 444 & 11 & 0 & 0 & 0 & 1 & -1 & 1 \\ -32 & 2 & -2 & 2 & 0 & 0 & 1 & 1 \\ -3 & 0 & -2 & 0 & 1 & 1 & 1 & 0 \\ -7 & -2 & 1 & -1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & -1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 444 & 11 & 0 & 0 & 0 & 1 & -1 & 1 \\ -32 & 2 & -2 & 2 & 0 & 0 & 1 & 1 \\ -3 & 0 & -2 & 0 & 1 & 1 & 1 & 0 \\ -7 & -2 & 1 & -1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & -1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 2 & 0 \end{pmatrix}$$

$a := V(A)$ $\qquad b := V(B)$

$$a = \begin{pmatrix} 50 & 48 & 50 & 48 & 49 & 48 & 47 & 45 \\ 54 & 53 & 53 & 53 & 52 & 50 & 49 & 48 \\ 56 & 56 & 54 & 54 & 54 & 53 & 52 & 50 \\ 58 & 58 & 56 & 55 & 55 & 53 & 54 & 53 \\ 58 & 58 & 58 & 56 & 55 & 55 & 55 & 56 \\ 59 & 60 & 59 & 58 & 57 & 56 & 58 & 57 \\ 61 & 60 & 60 & 60 & 59 & 57 & 59 & 59 \\ 63 & 63 & 63 & 63 & 61 & 60 & 60 & 60 \end{pmatrix} \qquad b = \begin{pmatrix} 50 & 48 & 50 & 48 & 49 & 48 & 47 & 45 \\ 54 & 53 & 53 & 53 & 52 & 50 & 49 & 48 \\ 56 & 56 & 55 & 54 & 54 & 53 & 52 & 50 \\ 58 & 58 & 56 & 55 & 55 & 53 & 55 & 52 \\ 58 & 58 & 58 & 56 & 55 & 55 & 55 & 56 \\ 59 & 60 & 59 & 58 & 57 & 56 & 58 & 57 \\ 61 & 60 & 60 & 60 & 59 & 57 & 59 & 59 \\ 63 & 63 & 63 & 63 & 61 & 60 & 60 & 60 \end{pmatrix}$$



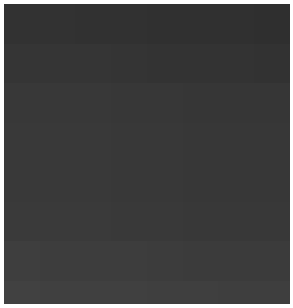


Рис. 4.5 – Демонстрация эффекта низкой частотной чувственности зрительной системы человека к незначительному изменению высокочастотных коэффициентов изображения

Проведем дополнительные исследования, усилим вносимые искажения высокочастотного коэффициенты. Для этого изменим выбранный коэффициент на 1000% и проведем соответствующие преобразования (см. рис. 4.6).

Как видно из приведенных на рис. 4.6 данных общий фон изображения не изменился, однако появились незначительные высокочастотные искажения, которые при естественном масштабе также визуально не детектируются.

$$B_{7,6} := A_{7,6} + 10 \cdot \mathrm{floor}\left(A_{7,6}\right)$$

$$A = \begin{pmatrix} 444 & 11 & 0 & 0 & 0 & 1 & -1 & 1 \\ -32 & 2 & -2 & 2 & 0 & 0 & 1 & 1 \\ -3 & 0 & -2 & 0 & 1 & 1 & 1 & 0 \\ -7 & -2 & 1 & -1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & -1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 444 & 11 & 0 & 0 & 0 & 1 & -1 & 1 \\ -32 & 2 & -2 & 2 & 0 & 0 & 1 & 1 \\ -3 & 0 & -2 & 0 & 1 & 1 & 1 & 0 \\ -7 & -2 & 1 & -1 & 0 & 0 & 1 & 1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ -2 & -1 & 0 & 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 11 & 0 \end{pmatrix}$$

$$a := V(A) \qquad b := V(B)$$

$$a = \begin{pmatrix} 50 & 48 & 50 & 48 & 49 & 48 & 47 & 45 \\ 54 & 53 & 53 & 53 & 52 & 50 & 49 & 48 \\ 56 & 56 & 54 & 54 & 54 & 53 & 52 & 50 \\ 58 & 58 & 56 & 55 & 55 & 53 & 54 & 53 \\ 58 & 58 & 58 & 56 & 55 & 55 & 55 & 56 \\ 59 & 60 & 59 & 58 & 57 & 56 & 58 & 57 \\ 61 & 60 & 60 & 60 & 59 & 57 & 59 & 59 \\ 63 & 63 & 63 & 63 & 61 & 60 & 60 & 60 \end{pmatrix} \qquad b = \begin{pmatrix} 51 & 47 & 50 & 48 & 49 & 48 & 46 & 45 \\ 53 & 54 & 52 & 53 & 52 & 48 & 50 & 48 \\ 57 & 54 & 56 & 53 & 53 & 55 & 50 & 51 \\ 57 & 60 & 54 & 56 & 56 & 51 & 57 & 52 \\ 59 & 56 & 60 & 55 & 54 & 57 & 53 & 57 \\ 59 & 62 & 57 & 58 & 58 & 54 & 60 & 56 \\ 61 & 59 & 61 & 60 & 59 & 59 & 58 & 59 \\ 63 & 64 & 62 & 64 & 61 & 60 & 60 & 59 \end{pmatrix}$$

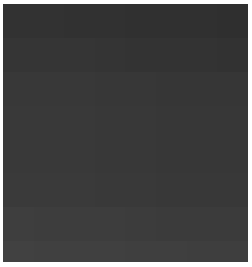


Рис. 4.6 – Демонстрация эффекта низкой частотной чувственности зрительной системы человека к незначительному изменению высокочастотных коэффициентов изображения

Следовательно, внесение изменений в различные частотные компоненты по равному влияет на восприятие этих изменений зрительной системой человека: низкочастотные искажения визуально детектируются, высокочастотные искажения, как правило, незаметны. В этом и проявляется эффект частотной чувствительности, который будем использовать в дальнейшем при реализации методов стеганографического встраивания.

Задание 2. Реализация в среде MathCAD алгоритмов встраивания и извлечения сообщений в частотную область изображений (метод Коха-Жао)

2.1. Реализуем алгоритм встраивания информационных данных в частотную область изображения на основе кодирования разниц абсолютных значений коэффициентов дискретно-косинусного преобразования. Для этого в среде MathCAD выполняем последовательность преобразований, представленных на рис. 4.7 и 4.8.

$$Pr := 5$$

$$H(H1, H2) := \begin{cases} -1 \\ 1 \text{ if } |H1| - |H2| > Pr \\ 0 \text{ if } |H1| - |H2| < -Pr \end{cases}$$

$$\begin{aligned}
&Input(TRR, m) := \left| \begin{array}{l}
TRR \leftarrow TRR \\
\text{if } m = 1 \wedge m \neq H\left(TRR_{3,1}, TRR_{1,3}\right) \vee H\left(TRR_{3,1}, TRR_{1,3}\right) = -1 \\
\quad \left| \begin{array}{l} TRR_{3,1} \leftarrow \left|TRR_{1,3}\right| + Pr \text{ if } TRR_{3,1} > 0 \\ TRR_{3,1} \leftarrow -\left|TRR_{1,3}\right| - Pr \text{ if } TRR_{3,1} \leq 0 \end{array} \right. \\
\text{if } m = 0 \wedge m \neq H\left(TRR_{3,1}, TRR_{1,3}\right) \vee H\left(TRR_{3,1}, TRR_{1,3}\right) = -1 \\
\quad \left| \begin{array}{l} TRR_{1,3} \leftarrow \left|TRR_{3,1}\right| + Pr \text{ if } TRR_{1,3} > 0 \\ TRR_{1,3} \leftarrow -\left|TRR_{3,1}\right| - Pr \text{ if } TRR_{1,3} \leq 0 \end{array} \right. \\
Input \leftarrow TRR
\end{array} \right.
\end{aligned}$$

Рис. 4.7 – Встраивание одного информационного бита в частотную область одного 8x8 блока изображения

Величина Pr задает порог изменения частотных коэффициентов при встраивании информационных бит.

Процедура H(H1, H2) реализует логическое правило изменения абсолютного значения разниц коэффициентов дискретно-кожсинусного преобразования, обозначенных переменными H1 и H2:

- если первый коэффициент по абсолютному значению больше второго на величину Pr, тогда это соответствует встраиванию бита «1»;
- если первый коэффициент по абсолютному значению меньше второго на величину Pr, тогда это соответствует встраиванию бита «0»;
- если разница абсолютных значений коэффициентов лежит в диапазоне от –Pr до Pr, тогда это соответствует неопределенной ситуации, когда нельзя детектировать ни бит «1», ни бит «0».

С использованием функции H(H1, H2) в процедуре Input(TRR, m) осуществляется кодирование разниц абсолютных значений коэффициентов дискретно-косинусного преобразования в одном блоке 8x8 коэффициентов. Входными данными является массив TRR размерностью 8×8 целых чисел, а

также битовая переменная m, которая передает значение встраиваемого бита. В качестве изменяемых выбраны коэффициенты в среднечастотной области с номерами (3,1) и (1,3).

В следующей процедуре tr (см. рис. 4.8) реализуется побитное встраивание информации в отдельные блоки контейнера с помощью циклического вызова процедуры Input(TRR, m), где в качестве TRR выступает текущий блок контейнера, а в качестве m – текущее значение встраиваемого бита.

```
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
```

Рис. 4.8 – Побитное встраивание последовательности информационных бит в частотную область изображения

Процедура vr реализует последовательное обратное дискретно-косинусное преобразование над измененными блоками контейнера. После чего с помощью процедуры R2 блоки объединяются в один массив данных, т.е. формируется новое изображение в пространственной области с уже встроенными информационными данными.

2.2. Выполним зрительное сравнение двух изображений – до и после встраивания информационных сообщений. Для этого воспользуемся массивами растровых данных (яркостей пикселей красного цвета) R и R2. Массив R содержит растровые данные до встраивания, R2 – полученный при выполнении предыдущего пункта массив.

Следует отметить, что полученное изображение R2 может содержать некорректные значения, т.к. внесение изменений в частотную область непосредственно влияет и на значения в пространственной области. Новые значения в пространственной области могут быть выше 255 или ниже 0,

однако в обрабатываемом формате изображения допустимыми значениями являются только целые числа от 0 до 255. Значение 256 средой символьной математики будет интерпретировано как число 0, 257 – как число 2, значение -3 – как число 253 и т.д. Для избежание такой ложной интерпретации данных выполним следующую процедуру (см. рис. 4.9), которая округлит все числа, большие 255 к 255, а все числа меньшие 0 к 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

Рис. 4.9 – Обработка массива растровых данных и вывод полученного изображения

На рис. 4.9 приведены для визуального сравнения два изображения: до встраивания информационных данных (слева) и после внесенных изменений (справа). Очевидно, что визуально приведенные изображения не отличаются.

Для количественной оценки различий изображений вычислим среднее арифметическое поэлементной разности массивов R и R2 (см. рис. 4.10).

$$RAZ := \left| \begin{array}{l} RAZ \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(R2) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R2) - 1 \\ \quad\quad RAZ \leftarrow RAZ + \left| R_{i,j} - R2_{i,j} \right| \\ RAZ \leftarrow \dfrac{RAZ}{\text{rows}(R)\cdot\text{cols}(R)} \\ RAZ \end{array} \right.$$

$$RAZ = 0.974$$

Рис. 4.10 – Количественная оценка различий между изображениями до и после встраивания информационного сообщения

Очевидно, что изображения отличаются очень незначительно, полученная величина усредненных искажений лежит ниже порога чувствительности зрительной системы человека, т.е. при визуальном осмотре искажения не обнаруживаются.

2.3. Реализуем алгоритм извлечения информационных данных из частотной области изображения. Для этого в среде MathCAD выполним последовательность преобразований, представленных 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	…

Рис. 4.11 – Побитное извлечение последовательности информационных бит из частотной области изображения и сравнение данных

В первой процедуре tr1 (см. рис. 4.11) реализуется прямое дискретно-косинусное преобразование всех блоков контейнера. В следующей процедуре m1 производится вычисление информационных бит посредством извлечения из среднечастотной области массивов коэффициентов дискретно-косинусного преобразования. Для этого в каждом блоке сравниваются абсолютные значения коэффициентов номерами (3,1) и (1,3). Если абсолютное значение коэффициента с номером (3,1) больше абсолютного значения коэффициента с номером (1,3) – тогда детектируется единичный информационный бит. В противном случае – детектируется нулевой информационный бит.

На рис. 4.11 для сравнения приведены также значения массивов информационных бит до встраивания (слева)и после извлечения (справа). Как видно из приведенного примера первые 15 информационных бит совпадают.

Для количественной вероятности ошибочного извлечения информационных данных выполним следующие операции (см. рис. 4.12).

$$
\begin{aligned}
Po :=\ & Po \leftarrow 0 \\
& \text{for } i \in 0..\,rows(m1) - 1 \\
& \quad Po \leftarrow Po + 1 \ \text{ if } \ m_i \neq m1_i \\
& Po \leftarrow \frac{Po}{rows(m1)} \\
& Po
\end{aligned}
$$

$$Po = 0$$

Рис. 4.12 – Оценка вероятности ошибочного извлечения информационных данных

Как следует из полученных результатов информационные биты, извлеченные из частотной области контейнера-изображения, полностью совпали с исходными данными. Это прогнозируемо, т.к. на заполненный контейнер не производилось никаких воздействий.

Задание 3. Реализация в среде MathCAD стеганоатаки на основе использования алгоритма сжатия JPEG и исследование ее возможностей

3.1. Для реализации стеганоатаки сперва сохраним заполненный контейнер-изображение (стеганограмму) в виде отдельного файла. Для этого сформируем массивы яркостей зеленого G2 и синего B2 цвета и выполним соответствующую команду «WRITERGB» для записи растровых данных в файл (см. рис. 4.13). В результате выполнения этой команды в папке с реализацией алгоритмов будет сформирован файл «Stego.bmp».

Для визуального сравнения пустого и заполненного контейнера выведем на экран изображения до (слева) и после (справа) встраивания (см. рис. 4.13).

3.2. Сымитируем стеганоатаку на основе использования алгоритма сжатия JPEG. Для этого откроем файл «Stego.bmp» внешним графическим редактором, например, Adobe Photoshop, Corel PHOTO-PAINT или Microsoft Paint. На рис. 4.14 приведен пример для случая использования графического редактора Corel PHOTO-PAINT. В открытом редакторе сохраним (экспортируем) изображение в формате JPEG. При этом будем использовать высокое качество[1], принятое по умолчанию (см. рис. 4.14).

[1] В графическом редакторе Microsoft Paint возможность выбора качества изображения не предусмотрено. При выполнении этого задания лабораторной работы следует использовать те настройки графического редактора, которые приняты по умолчанию.

$$G2 := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(R2) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R2) - 1 \\ \quad\quad G2_{i,j} \leftarrow G_{i,j} \\ G2 \end{array} \right. \qquad B2 := \left| \begin{array}{l} \text{for } i \in 0..\text{rows}(R2) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R2) - 1 \\ \quad\quad B2_{i,j} \leftarrow B_{i,j} \\ B2 \end{array} \right.$$

$$\text{WRITERGB}(\text{"Stego.bmp"}) := \text{augment}(R2, G2, B2)$$

"1"

"Stego"

Рис. 4.13 – Запись заполненного контейнера-изображения в файл «Stego.bmp» и вывод изображения

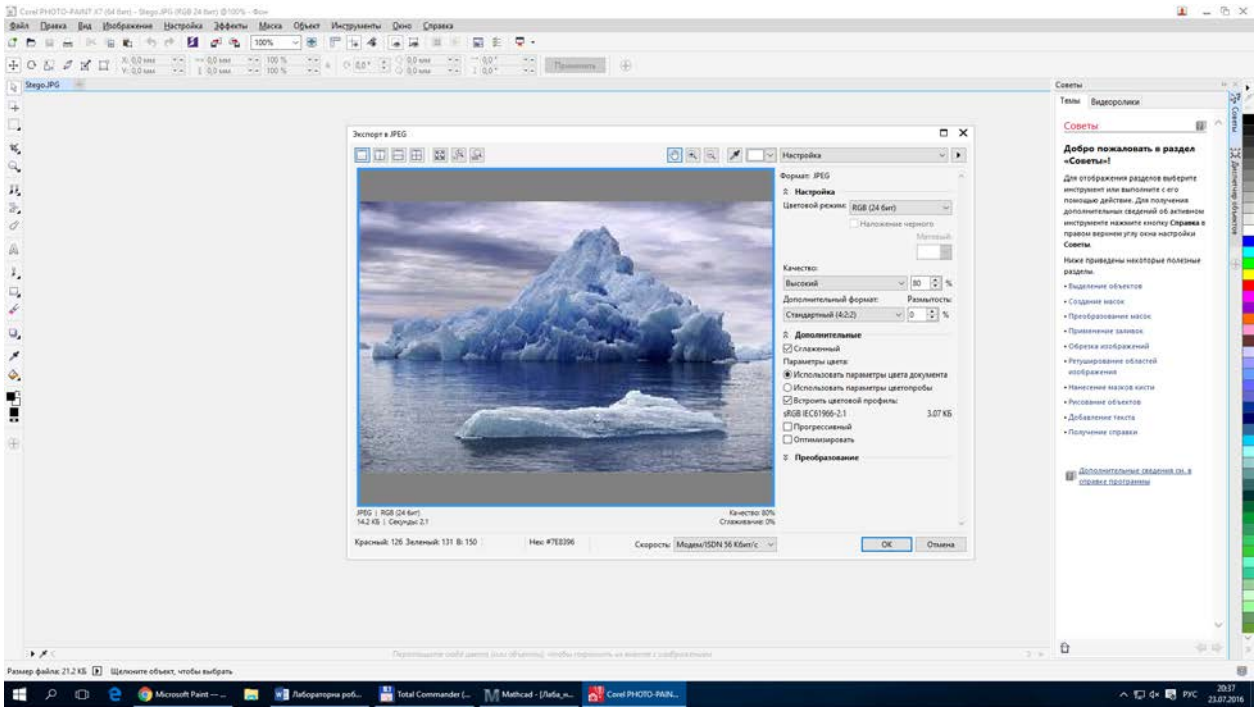


Рис. 4.14 – Имитация стеганоатаки на основе алгоритма сжатия JPEG в графическом редакторе Corel PHOTO-PAINT

Полученное в результате указанных преобразований изображение будет сохранено в формате JPEG, при этом содержащиеся в нем информационные данные могут исказиться в результате выполнения алгоритма сжатия JPEG.

3.3. Извлечем встроенные данные из сжатого (атакованого) контейнера-изображения. Для этого выполним преобразования, приведенные на рис. 4.15 (по аналогии с преобразованиями, изложенными в п. 2.3).

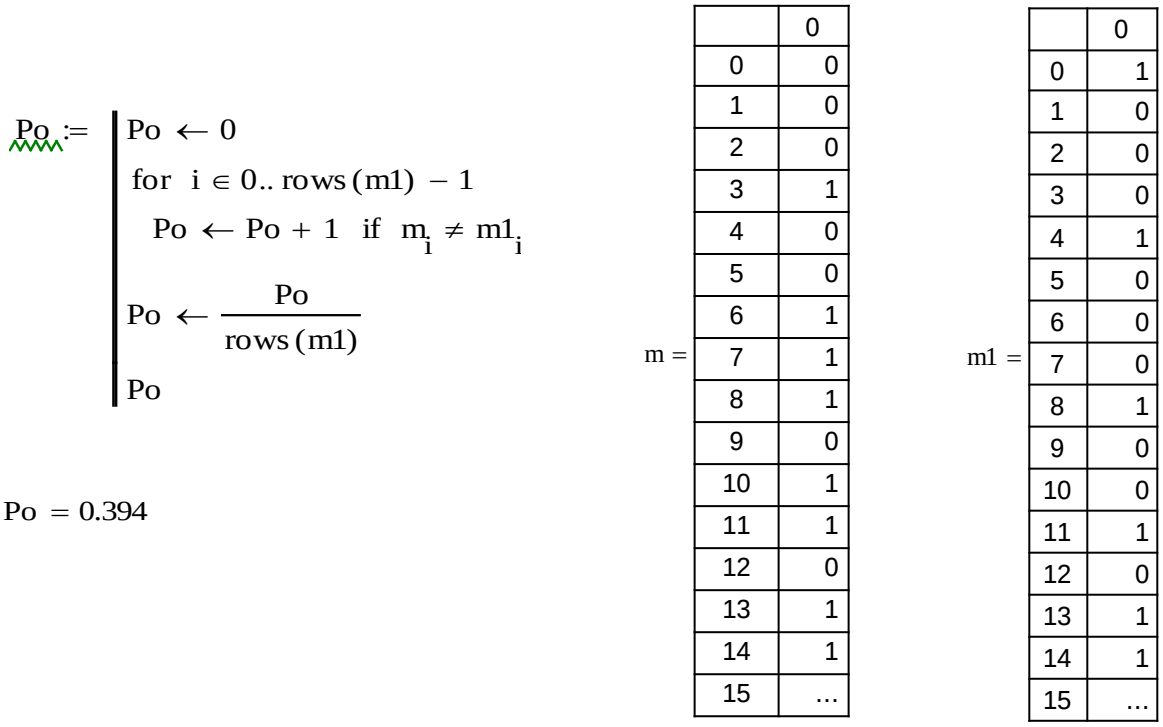
```
R_Stego := READ_RED("Stego.jpg")

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
```

Рис. 4.15 – Побитное извлечение последовательности информационных бит из частотной области сжатого изображения и сравнение данных

Первой командой считывается массив яркостей красного цвета в переменную «R_Stego». Далее этот массив разбивается на блоки размером NxN элементов и над каждым блоком с помощью функции T(V) выполняется прямое дискретно-косинусное преобразование. Затем, как и в п. 2.3, выполняется последовательное извлечение информационных бит, результат извлечения записывается в массив «m1».

3.4. Для количественной вероятности ошибочного извлечения информационных данных выполним следующие операции (см. рис. 4.16). На рисунке приведены также в качестве примера первые пятнадцать бит встроенных и извлеченных из сжатого контейнера информационных данных.

Как следует из приведенных на рис. 4.16 данных сжатие изображение привело к существенному (около 40%) искажению информационных бит. Это наглядно подтверждает и приведенный на рисунке пример.



	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	…

Рис. 4.16 – Эмпирическая оценка вероятности ошибочного извлечения информационных данных и сравнение с исходными данными

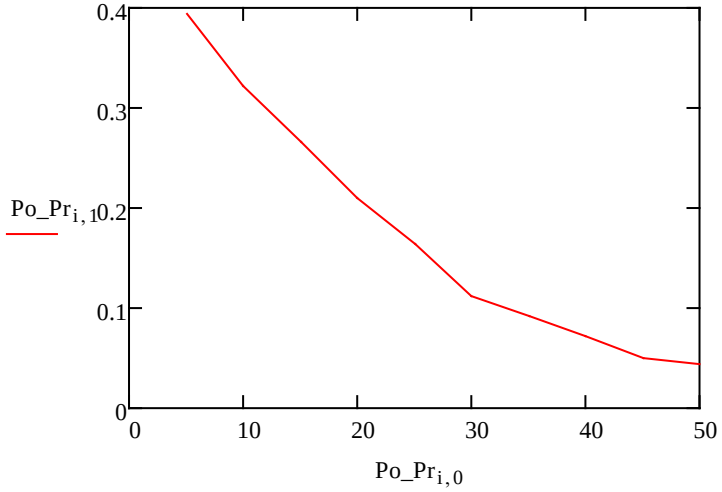
3.5. Уменьшим число возникающих ошибочных данных при извлечении информационных сообщений. Для этого изменим параметр «Pr» - величину порога изменения частотных коэффициентов при встраивании информационных бит (см. п. 2.1). Выберем значение порога равным 10 и повторим все выполненные ранее процедуры: встраивание, сохранение изображение в виде файла-изображения, сжатие изображение алгоритмом JPEG (имитация стеганоатаки) и извлечение сообщения из сжатого изображения. Эмпирическая оценка вероятности ошибочного извлечения информационных данных (см. п. 3.4) дает значение 0,323, т.е. число ошибок уменьшилось. Однако увеличение порога «Pr» неизбежно приведет к увеличению вносимых искажений в контейнер-изображение. Эмпирическая оценка (см. п. 2.2) подтверждает это, полученное значение 1,273 (по сравнению с 0,974 при пороге равном 5). Повторим соответствующие экспериментальные исследования для различных значений порога «Pr»: 15, 20, 25, 30, 35, 40, 45, 50. Полученные эмпирические оценки вероятности ошибочного извлечения информационных данных и средней величины вносимых искажений в контейнер-изображение сведем в соответствующие таблицы (см. рис. 4.17).

В таблице «Po_Pr» приведены полученные опытные данные, полученные в результате эмпирической оценки вероятности ошибочного извлечения информационных данных (второй столбец) в зависимости от величины порога «Pr» (первый столбец). В таблице «RAZ_Pr» приведены полученные опытные данные, полученные в результате эмпирической оценки средней величины вносимых искажений в контейнер-изображение (второй столбец) в зависимости от величины порога «Pr» (первый столбец).

На рис. 4.17 приведены также эмпирические зависимости в виде графиков, построенных по соответствующим табличным значениям.

$$Po\_Pr := \begin{pmatrix} 5 & 0.394 \\ 10 & 0.323 \\ 15 & 0.266 \\ 20 & 0.21 \\ 25 & 0.165 \\ 30 & 0.113 \\ 35 & 0.092 \\ 40 & 0.073 \\ 45 & 0.051 \\ 50 & 0.045 \end{pmatrix} \qquad RAZ\_Pr := \begin{pmatrix} 5 & 0.974 \\ 10 & 1.273 \\ 15 & 1.692 \\ 20 & 2.134 \\ 25 & 2.601 \\ 30 & 3.121 \\ 35 & 3.568 \\ 40 & 4.012 \\ 45 & 4.560 \\ 50 & 5.069 \end{pmatrix}$$

$i := 0..9$



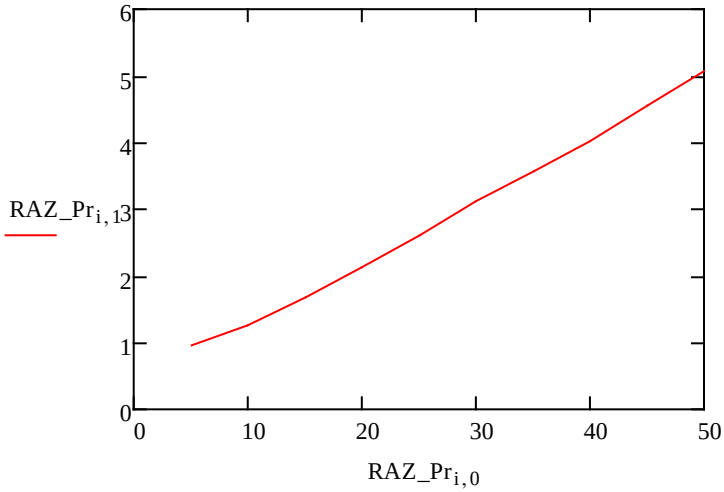


Рис. 4.17 – Построение эмпирических зависимостей вероятности ошибочного извлечения информационных данных и средней величины вносимых искажений в контейнер-изображение от величины порога «Pr»

Как видно из приведенных на рис. 4.17 зависимостей с увеличением величины порога «Pr» вероятность ошибочного извлечения информационных бит сообщения резко снижается. Однако это ведет к аналогичному повышению вносимых искажений в контейнер-изображение. Если использовать величину порога чувствительности зрительной системы человека в 2-3% от максимальной яркости изображения, тогда внесение искажений меньших 256*0,02 = 5,12 уровней яркости достигается только при величине порога «Pr» меньшим 50. Следовательно, реализованный метод стенографического встраивания информации позволяет передавать скрытые (от визуального обнаружения) сообщения с вероятностью ошибки извлечения не меньшей 0,05. Снижение ошибок в информационных данных может быть достигнуто за счет использования помехоустойчивого кодирования (см. лабораторную работу №2) и/или более надежных методов.

Задание 4. Реализация в среде MathCAD усовершенствованных алгоритмов встраивания и извлечения сообщений в частотную область изображений (метод Бенгама-Мемона-Ео-Юнга)

4.1. Реализуем алгоритм встраивания информационных данных в частотную область изображения на основе усовершенствованного правила кодирования разниц абсолютных значений коэффициентов дискретно-косинусного преобразования (с помощью метода Бенгама-Мемона-Ео-Юнга). Для этого в среде MathCAD выполняем последовательность преобразований, представленных на рис. 4.18 и 4.19.

$$
\begin{aligned}
&\text{Input}(TRR, m) := \\
&\quad \text{if } m = 1 \\
&\quad\quad TRR_{3,1} \leftarrow \left|TRR_{1,3}\right| + Pr \text{ if } TRR_{3,1} > 0 \\
&\quad\quad TRR_{3,1} \leftarrow -\left|TRR_{1,3}\right| - Pr \text{ if } TRR_{3,1} \le 0 \\
&\quad\quad TRR_{3,2} \leftarrow \left|TRR_{1,3}\right| + Pr \text{ if } TRR_{3,2} > 0 \\
&\quad\quad TRR_{3,2} \leftarrow -\left|TRR_{1,3}\right| - Pr \text{ if } TRR_{3,2} \le 0 \\
&\quad \text{if } m = 0 \\
&\quad\quad TRR_{1,3} \leftarrow \left|TRR_{3,1}\right| + Pr \text{ if } TRR_{1,3} > 0 \\
&\quad\quad TRR_{1,3} \leftarrow -\left|TRR_{3,1}\right| - Pr \text{ if } TRR_{1,3} \le 0 \\
&\quad\quad TRR_{2,3} \leftarrow \left|TRR_{3,1}\right| + Pr \text{ if } TRR_{2,3} > 0 \\
&\quad\quad TRR_{2,3} \leftarrow -\left|TRR_{3,1}\right| - Pr \text{ if } TRR_{2,3} \le 0 \\
&\quad TRR
\end{aligned}
$$

Рис. 4.18 – Усовершенствованное правило кодирования разниц абсолютных значений коэффициентов дискретно-косинусного преобразования

На рис. 4.18 приведено правило кодирования разниц абсолютных значений коэффициентов, что соответствует первому усовершенствованию в соответствии с методом Бенгама-Мемона-Ео-Юнга. Вместо двух

коэффициентов дискретно-косинусного преобразования (в методе Коха-Жао) используется три коэффициента и, по утверждению авторов метода это существенно улучшает эксплуатационные характеристики стеганографической защиты []. Второе усовершенствование, основанное на отбраковке блоков, предлагается реализовать самостоятельно.

На рис. 4.19 приведено описание процедур встраивания данных в контейнер-изображение с помощью усовершенствованной процедуры кодирования разниц абсолютных значений коэффициентов.

```
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
```

Рис. 4.19 – Встраивание данных в контейнер-изображение с помощью усовершенствованной процедуры кодирования разниц абсолютных значений коэффициентов дискретно-косинусного преобразования

Преобразования, описание которых приведено на рис. 4.19, аналогичны тем, которые рассмотрены на рис. 4.8. По аналогии с рис. 4.9 на рис. 4.20 приведена окончательная обработка массива растровых данных и вывод полученного изображения.

4.2. Для количественной оценки вносимых искажений в контейнер-изображение вычислим среднее арифметическое поэлементной разности массивов R (до встраивания) и R2 (после встраивания). Полученные результаты приведены 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

Рис. 4.20 – Обработка массива растровых данных и вывод полученного изображения

$$RAZ := \left| \begin{array}{l} RAZ \leftarrow 0 \\ \text{for } i \in 0..\text{rows}(R2) - 1 \\ \quad \text{for } j \in 0..\text{cols}(R2) - 1 \\ \quad\quad RAZ \leftarrow RAZ + \left| R_{i,j} - R2_{i,j} \right| \\ RAZ \leftarrow \dfrac{RAZ}{\text{rows}(R)\cdot\text{cols}(R)} \\ RAZ \end{array} \right.$$

$$RAZ = 1.815$$

Рис. 4.21 – Количественная оценка различий между изображениями до и после встраивания информационного сообщения

Очевидно, что величина вносимых искажений в усовершенствованном методе по сравнению с методом Коха-Жао существенно (практически в два раза) возросла. Это объясняется отсутствием процедуры отбраковки и встраиванием данных в 3 (вместо двух) коэффициента дискретно-косинусного преобразования. Однако этот частный случай не информативен. Необходимо оценить также вероятность ошибочного извлечения информационных данных, а также исследовать соответствующие зависимости для различных значений порога «Pr».

4.3. Реализуем алгоритм извлечения информационных данных из частотной области изображения. Для этого в среде MathCAD выполним последовательность преобразований, представленных на рис. 4.22 (по аналогии с представленными на рис. 4.11преобразованиями).

```
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
```

Рис. 4.22 – Побитное извлечение последовательности информационных бит из частотной области изображения и сравнение данных

После выполнения дискретно-косинусного преобразования всех блоков контейнера (процедура «tr1») производится вычисление информационных бит (массив «m1») посредством извлечения из среднечастотной области массивов коэффициентов дискретно-косинусного преобразования. Для этого в каждом блоке сравниваются абсолютные значения коэффициентов номерами (3,1), (1,3) и (3,2). Если абсолютное значение коэффициента с номером (3,1) больше абсолютного значения коэффициента с номером (1,3) и, одновременно, коэффициента с номером (3,2) – тогда детектируется единичный информационный бит. Если абсолютное значение коэффициента с номером (3,1) меньше или равно абсолютному значению коэффициента с номером (1,3) и, одновременно, коэффициенту с номером (3,2) – тогда детектируется нулевой информационный бит.

Для количественной вероятности ошибочного извлечения информационных данных выполним операции, приведенные на рис. 4.23 (по аналогии с рис. 4.12).

Извлеченные из частотной области контейнера-изображения информационные биты полностью совпали (см. рис. 4.23) с исходными данными (как и в методе прототипе), что объясняется отсутствием вносимых искажений в контейнер-изображение.

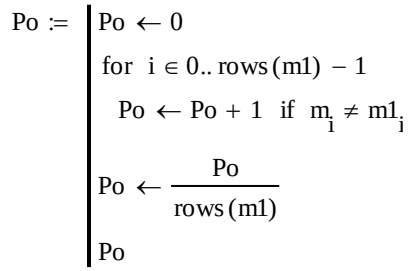


Рис. 4.23 – Оценка вероятности ошибочного извлечения информационных данных

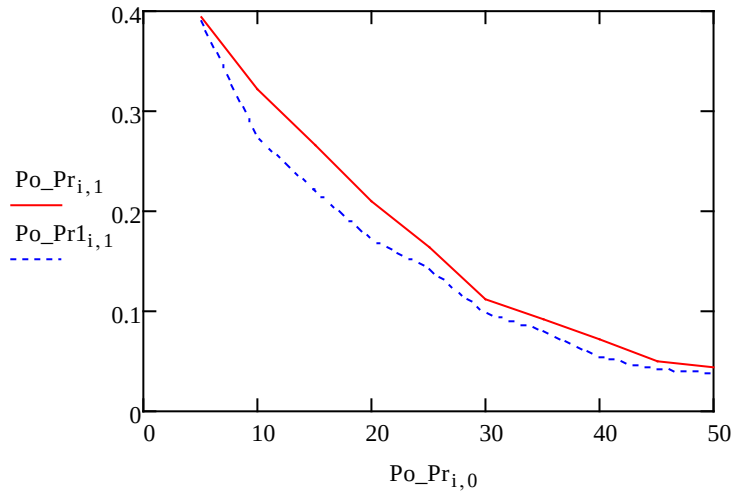
4.4. Проведем эмпирические исследования эффективности усовершенствованного метода стегнаографического преобразования (по аналогии с рассмотренным выше заданием 3). Экспериментальные исследования доли вносимых искажений в контейнер-изображение и числа возникающих ошибок при извлечении информационных данных проведем для различных значений порога «Pr»: 15, 20, 25, 30, 35, 40, 45, 50 (как и при выполнении задания 3). Полученные эмпирические оценки вероятности ошибочного извлечения информационных данных и средней величины вносимых искажений в контейнер-изображение сведем в соответствующие таблицы, которые приведены на рис. 4.24 (по аналогии с рис. 4.17).

Таблицы «Po_Pr1» и «RAZ_Pr1» характеризуют величину ошибок в извлеченных данных и уровень вносимых искажений в контейнер-изображение. Они заполнены такимже способом, как и соответствующие таблицы «Po_Pr» и «RAZ_Pr» на рис. 4.17.

На рис. 4.24 приведены также эмпирические зависимости в виде графиков, построенных по соответствующим табличным значениям. На графиках сплошной линией приведены эмпирические зависимости, характеризующие эффективность метода Коха-Жао, прерывистой линией – данные для усовершенствованного метода. Как следует из приведенных зависимостей усовершенствованный метод действительно позволяет снизить число возникающих ошибок при извлечении информационных данных (первый график). Однако его использование сопряжено также и с увеличением вносимых искажений в контейнер-изображение (второй график). Если зафиксировать уровень внесенных искажений в 2-3% от максимальной яркости изображения (256*0,02 = 5,12), тогда величина порога «Pr» не должна превышать 40 (см. второй график). Это примерно соответствует вероятности ошибки извлечения около 0,05. Т.е. с точки зрения величины вносимых искажений и ошибок, возникающих при извлечении информационных данных, метод Коха-Жао и усовершенствованный метод (метод Бенгама-Мемона-Эо-Юнга) сопоставимы по эффективности. Это объясняется отсутствием процедуры отбраковки блоков в усовершенствованном методе (данную процедуру предлагается реализовать самостоятельно).

$$Po\_Pr1 := \begin{pmatrix} 5 & 0.391 \\ 10 & 0.275 \\ 15 & 0.221 \\ 20 & 0.173 \\ 25 & 0.142 \\ 30 & 0.099 \\ 35 & 0.081 \\ 40 & 0.055 \\ 45 & 0.042 \\ 50 & 0.038 \end{pmatrix} \qquad RAZ\_Pr1 := \begin{pmatrix} 5 & 1.815 \\ 10 & 2.273 \\ 15 & 2.692 \\ 20 & 3.142 \\ 25 & 3.351 \\ 30 & 3.721 \\ 35 & 4.298 \\ 40 & 5.431 \\ 45 & 6.169 \\ 50 & 6.801 \end{pmatrix}$$

$$i := 0..9$$



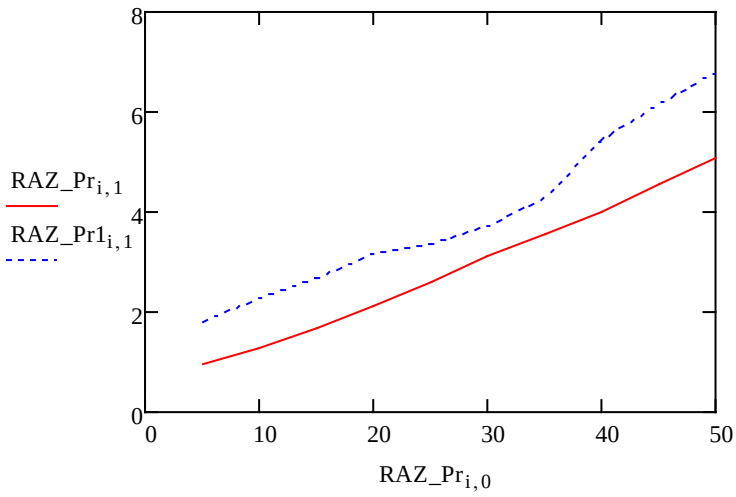


Рис. 4.24 – Построение эмпирических зависимостей вероятности ошибочного извлечения информационных данных и средней величины вносимых искажений в контейнер-изображение от величины порога «Pr»

Дополнительное задание. Реализация в среде MathCAD алгоритмов встраивания и извлечения сообщений в частотную область изображений методом Д.Фридрих. (предлагается к самостоятельному выполнению)

Навчальне видання

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ
до лабораторних робіт з дисципліни
“Стеганографія”

для студентів напрямку
6.170101 “Безпека інформаційних і комунікаційних систем”
спеціальності
1701 “Інформаційна безпека”

Упорядник: КУЗНЕЦОВ Олександр Олександрович

Відповідальний випусковий С.Г. Рассомахін

Редактор

План 2015, 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