**УДК 004**

**Д.Мартинов, магістр гр. КН-22М-1**
*Центральноукраїнський національний технічний університет*

# ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ СТИСКУ ЗОБРАЖЕНЬ ЗА ДОПОМОГОЮ ФРАКТАЛІВ

У статті розроблено програмне забезпечення, яке призначено для системи стиску зображень за допомогою фракталів. Метою розробки є дослідження та програмна реалізація системи стиску зображень за допомогою фракталів. Об’єктом дослідження є процес стиску зображень за допомогою фракталів. Предметом дослідження є методи стиску зображень за допомогою фракталів. Методи дослідження базуються на методах обробки зображень, методах математичної статистики, методах розробки програмного забезпечення. Результат роботи – програмна реалізація системи стиску зображень за допомогою фракталів. В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

**Постановка проблеми.** Існує багато різних методів стиснення фрактальних зображень, і вони є лише жменькою з усіх існуючих методів стиснення. Техніка стиснення зображень вважається фрактальною, якщо її основною філософією є використання самоподібності, яка природно виникає в багатьох зображеннях. Часто можна знайти частину зображення, яка, якщо її змінити певним чином, підійде до іншої частини того самого зображення.

Звідси походить «фрактальна» частина, оскільки фрактали часто описують як самоподібні об’єкти. Пізніше ми повернемося до зв'язку між фракталами та методами стиснення зображень. Ми також торкнемося переваг і недоліків загального методу та продемонструємо просту схему стиснення фрактального зображення. Але перш ніж наважуватися на тему фрактального стиснення зображень, нам потрібні деякі знання про математичні об’єкти, які називаються фракталами.

Слово «фрактал» походить від латинського слова fract або fractus, що означає «зламаний» або «нерівний», і було створено Бенуа Б. Мандельбротом. Замість того, щоб намагатися отримати точне визначення фракталу, можна підійти до предмета іншим шляхом.

Як стверджує Фальконер у «Математичні основи та застосування фрактальної геометрії» [2], ми могли б скласти список властивостей, які характеризують фрактали. Фрактальна множина F може мати не всі, але принаймні деякі з наступних властивостей:

– F має певну форму самоподібності;
– F деталізується на кожній шкалі;
– Як правило, фрактальна розмірність F (визначена якимось чином) не обов’язково повинна бути цілим числом;
– У більшості випадків F має простий алгоритмічний опис.

Найпоширенішим прикладом фрактала є самоповторювана (самоподібна) геометрична фігура. Під цим ми маємо на увазі набір, який складається з менших копій самого себе.

**Аналіз останніх досліджень і публікацій.** При аналізі останніх досліджень і публікацій [1-20] було виявлено певні прогалини у забезпеченні системи стиску зображень за допомогою фракталів.

**Мета й завдання дослідження.** Метою роботи є дослідження та програмна реалізація системи стиску зображень за допомогою фракталів.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем стиску зображень за допомогою фракталів.
– Дослідження системи стиску зображень за допомогою фракталів.
– Програмна реалізація системи стиску зображень за допомогою фракталів.

*Об'єктом дослідження* є процес стиску зображень за допомогою фракталів.

*Предметом дослідження* є методи стиску зображень за допомогою фракталів.

*Методи дослідження* базуються на методах обробки зображень, методах математичної статистики, методах розробки програмного забезпечення.

**Виклад основного матеріалу.** Фотографія у фізичному світі – це просто аркуш паперу з написом, але фотографія в цифровому світі – це набір маленьких логічних одиниць, які називаються пікселями. Роздільна здатність цифрового зображення означає кількість пікселів, з яких воно складається. Якщо зображення має ширину n пікселів і висоту m пікселів, ми говоримо, що роздільна здатність зображення становить n×m. Можна розглядати цифрове зображення з роздільною здатністю n×m як матрицю n на m, де кожен запис матриці представляє піксель. Значення запису, значення пікселя, є тим, що визначає колір пікселя. Кількість різних кольорів, які можуть бути представлені пікселем, залежить від кількості біт на піксель, який використовується. Наприклад, 8-бітний колір дозволяє відображати $2^8$ кольорів. Біт (скорочення від «двійкова цифра») містить одне двійкове значення «0» або «1», тому він може відповідати лише на просте запитання «так» або «ні». Кожен піксель сірого цифрового зображення часто складається з 8 біт, що означає, що для сірого цифрового зображення з роздільною здатністю 1024x1024 потрібно $1024 \cdot 1024 \cdot 8 = 8{,}4 \cdot 10^6$ біт для зберігання. Ми називатимемо розмір пам'яті, необхідний для збереження зображення, як розмір пам'яті зображення. Розмір пам'яті зазвичай вимірюється в байтах, де один байт = 8 біт.

Незважаючи на те, що високошвидкісний доступ до Інтернету розширюється по всьому світу, а швидкість з'єднання зростає, вона все ще обмежена. Час, необхідний для надсилання файлу даних, залежить від швидкості з'єднання, а також від обсягу пам'яті файлу. Таким чином, надсилання зображення високої роздільної здатності або колекції зображень все одно може зайняти значну кількість часу. Стиснення зображень зменшує кількість інформації, яку необхідно передати, а також час, який це займе. Але як стиснути зображення? Важливою якістю людського ока є його нечутливість до різноманітних втрат інформації. Іншими словами, зображення можна змінити таким чином, що людське око не помітить. Якщо є велика кількість надлишкових даних, які не впливають на «ширшу картину», то дані можуть бути сильно стиснуті. Методи, які втрачають частину інформації під час стиснення, називаються методами стиснення з втратами, тоді як їхні аналоги, де вихідні дані не втрачаються, називаються методами стиснення без втрат [3].

Основна ідея фрактального стиснення зображень полягає в збереженні (також називається кодуванням) зображень у вигляді набору перетворень. Щоб це було корисним, має бути спосіб розпакувати зображення, тобто спосіб реконструювати зображення зі збереженої інформації. Декомпресія (або декодування) передбачає багаторазове застосування перетворення до довільного початкового зображення, щоб отримати зображення, яке є або оригінальним, або, у більшості випадків, дуже схожим на нього. Кожне зображення на малюнку 7 можна зберігати як набір афінних перетворень замість значень пікселів. Якщо числа в перетвореннях належать до загальновживаного типу даних float, тоді розмір пам'яті кожного числа становить 32 біти. Для збереження дерева як набору перетворень, наприклад, потрібно лише 4 перетворення × 6 чисел на перетворення × 32 біти на число = 768 біт. Однак для його збереження як колекції пікселів потрібно $512 \cdot 512 \cdot 1 = 262{,}144$ біт для роздільної здатності 512×512 (оскільки це лише чорно-білий колір, нам потрібен лише 1 біт для збереження кольору). Маючи це на увазі, можна поставити запитання, чи можемо ми знайти невелику кількість афінних перетворень, що представляють будь-яке зображення? Відповідь просто ні, оскільки природне зображення не є абсолютно самоподібним, але вони також не

повністю позбавлені самоподібності. Як було зазначено раніше, дивлячись на зображення, можна знайти його частину, яка, якщо її масштабувати та повернути, вписується в іншу частину того самого зображення. Ці типи самоподібності можна знайти на більшості зображень облич, автомобілів, гір тощо. Щоб використати ці подібності, нам потрібно певним чином розділити зображення та порівняти ці фрагменти між собою.

**Перетворення кольорів RGB у кольори YUV**

У стиску JPEG застосовується система кольорів *YUV*. Поділ даних *RGB* на дані *YUV* дозволяє програмі стиску приділяти більше увагу даним про яскравість *(Y)*, чим даним про колір *(UV)*. Цей процес називається підвибіркою, так як три компоненти вибираються з різною частотою. Так метод підвибірки, що називається *YUV411*, на кожну вибірку кольору робить чотири вибірки даних про яскравість. Наприклад, якщо рисунок не піддавався підвибірці, те величини *YUV* зустрічаються в ньому з однаковою частотою. При використанні підвибірки *YUV411* на шість значень вибірки обробленого файлу доводиться дванадцять значень вибірки вихідного файлу. Таким чином, підвибірка даних відразу зменшує розмір файлу зображення.

**Метод стиску JPEG**

Процес стиску за схемою JPEG складається із трьох кроків (не вважаючи підвибірку). Перший крок – це запис зміни значень пікселів у вигляді зміни частот: як швидко міняються яскравість і колір пікселів. Другий крок – угруповання цих окремих змін частот за середнім значенням (перший етап стиску). І третій етап – стиск цих усереднених даних за допомогою модифікованого алгоритму кодування Хаффмана.

**Зміна частоти**

JPEG визначає зміну частоти даних за допомогою дискретного перетворення Фур'є (ДПФ), що застосовується для кожної піксельной компоненти в обраній області пікселів. Наприклад, вибирається група з $8\times 8$ пікселів, і до неї застосовується перетворення ДПФ: спочатку до величин червоних компонентів у всій групі, потім зелених, і, нарешті, синіх.

Замість дійсних значень пікселів величини ДПФ зберігають швидкість зміни інтенсивності від пікселя до пікселю. Насправді даних про частоту виходить більше, ніж вихідних піксельних даних, але наступні два кроки це усувають.

**Усереднення**

Після обчислення за допомогою ДПФ значень зміни частоти ці величини усередняються відповідно до плаваючої шкали відносної важливості. Це значить, що зміни частоти, які менше впливають на загальний вид зображення (наприклад, швидкі зміни частоти), усередняються більше інших значень. Саме на цій стадії стиску зображення відбуваються втрати, так як величина застосовуваного усереднення може регулюватися.

**Стиск методом Хаффмана**

Остаточно усереднені дані про частоту стискуються за допомогою модифікованого алгоритму кодування Хаффмана, що особливо ефективний для цього типу даних, так як будує таблиці кодів, що базуються на частоті повторення величин.

**Фрактальний стиск**

– Виберемо максимальне число N фрагментів $R_i$.

– Додамо фрагмент $R_1 = \Omega$ у список перетворень і позначимо його як неопрацьований.

– Поки є неопрацьовані фрагменти в списку виконувати:

1. Для кожного неопрацьованого фрагмента знайти відповідні $D_{j(i)}$, $W_i$ і $F_i$.
2. Знайти в списку фрагмент $R_i$ найбільшого розміру й найбільшою метрикою $\sigma^2_{R_i}(W_i, D_{j(i)}, F_i)$.

3. Якщо число фрагментів у списку менше N, тоді розбити фрагмент $R_i$ на більше дрібні й занести їх у список як неопрацьовані. Викреслити з писку фрагмент $R_i$.

**Обчислювальний експеримент**

Описаний метод фрактального стиску статичних зображень був запрограмований із застосуванням об'єктно-орієнтованого підходу. Програма працює як із чорно-білими, так і з повнокольоровими зображеннями формату *Windows Bitmap*.

Для оцінки якості закодованого 8-бітного зображення або одного каналу кольорового використовувалося співвідношення сигнал-шум, вимірюване в децибелах (d) і визначене як:

$$PSNR = 10 \cdot \lg\left(\frac{255^2}{\sum (f(\xi,\eta) - \tilde{f}(\xi,\eta))^2}\right),$$

де підсумовування ведеться по всім пікселям вихідного $f$ й наближеного $\tilde{f}$ зображення.

Алгоритм кодування реалізований у двох варіантах.

1. Алгоритм кодування з фіксованим розміром блоків.

Алгоритм оперує із прямокутними областями однакового розміру. Розмір областей фіксований від початку роботи алгоритму до кінця.

Перевагою даного алгоритму полягає в тому, що при відповідному виборі розмірів оброблюваних областей забезпечується рівномірна якість кодування всього зображення. Недоліком алгоритму є малий коефіцієнт стиску.

2. Алгоритм кодування з розбивкою зображення на дерево підблоків.

За основу був узятий *алгоритм 1* з попереднього параграфа. При недостатній точності кодування оброблюваний фрагмент розбивається на чотири частини, кожна з яких обробляється так, як і всі інші.

Дана розбивка ефективно з погляду зберігання його у файлі. Так будь-яку таку розбивку можна представити у вигляді двійкової послідовності символів, де 1 – означає те, що блок розбитий, 0 – інакше. При тестуванні алгоритм показав кращі результати, ніж його попередник, по ступеню стиску, але не завжди забезпечує достатню точність кодування деяких однотонних областей.

Ступінь стиску зображення при фрактальному кодуванні залежить від розмірів оброблюваних R-блоків, параметрів $s_i$ і $o_i$ перетворення $F_i$. Для чорно-білих зображень вона може бути обчислена за формулою:

$$\frac{2^8 * \textit{розмір блоку у пікселях}}{\textit{число біт для } s + \textit{число біт для } o + \lceil \log_2(8 \times \textit{розмір кодової книги}) \rceil},$$

де $\lceil\ \rceil$ – символ округлення нагору. Таким чином, ми можемо одержати додатковий ступінь стиску, якщо квантувати параметри $s_i$ й $o_i$. Очевидно, що після квантування цих параметрів середньоквадратичне відхилення між вихідним і наближеним зображенням зросте на величину не більшу, ніж

$$\left(\frac{\max\limits_i(s_i) - \min\limits_i(s_i)}{2^{\textit{число біт для } s}} * 2^8 + \frac{\max\limits_i(o_i) - \min\limits_i(o_i)}{2^{\textit{число біт для } o}}\right)^2.$$

Пошук фрактальної моделі (для кожного R-блоку відповідний D-блок і перетворення *W*) зображення здійснювався за допомогою модифікованого генетичного алгоритму, описаного раніше. При проведенні обчислювального експерименту ГА припиняв свою роботу з витікання заданого числа поколінь або при досягненні певної якості кодування, тобто якщо ступінь пристосованості найкращої особи буде вище деякого заздалегідь заданого числа. Кращі результати були отримані при запуску ГА з наступними параметрами $P_{Mut} = 0.1$, $P_{Cross} = 0.75$.

Слід зазначити, що використання генетичних алгоритмів для завдання фрактального стиску є одним з варіантів відходу від повного перебору. Так для зображення розміром $256 \times 256$ пікселів, для кожного R-блоку потрібно перебрати $2^8 * 2^8 * 2^3$ комбінацій D-блоків з аффіним перетворенням $W$. Наступна діаграма показує середнє число перелічених варіантів при роботі ГА залежно від числа поколінь і розміру популяції.

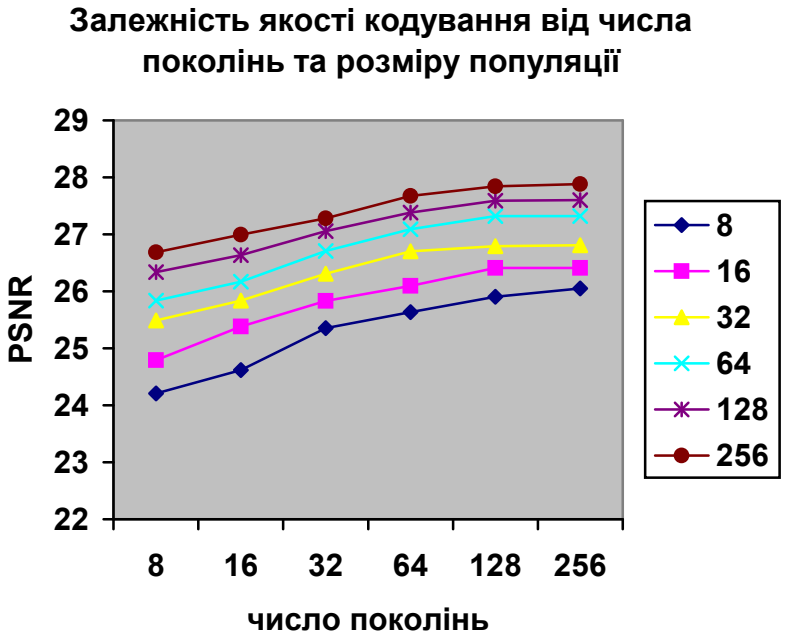


Рисунок 1 – Результати застосування генетичних алгоритмів для фрактального стиску

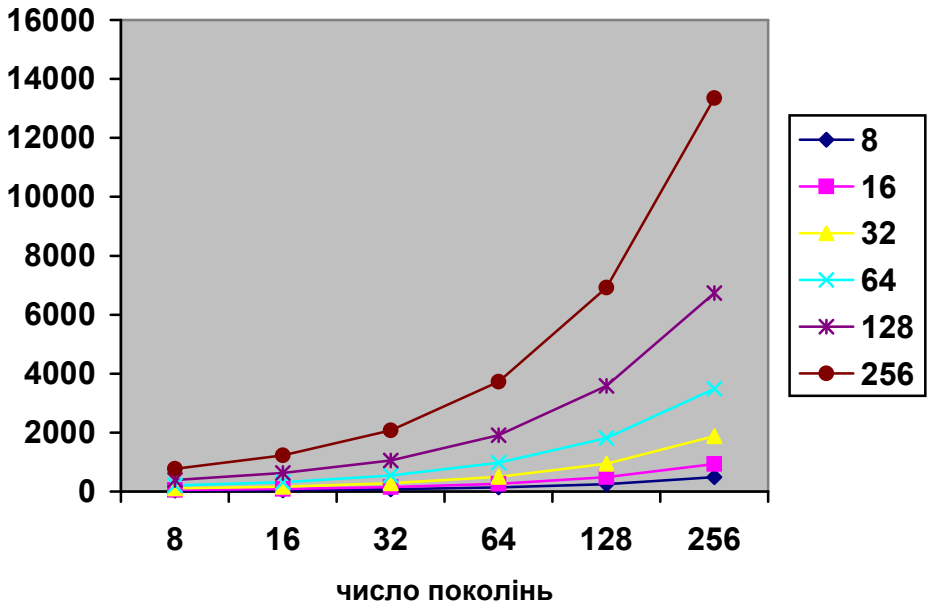


Рисунок 2 – Середнє число перелічених варіантів при роботі ГА

Висновок:

– Із цих графіків видно, що запропонований ГА має значні перевагу в порівнянні з алгоритмом повного перебору за часом роботи.

– Представляється доцільним використання даного алгоритму в сполученні з іншими евристиками такими, як алгоритм класифікації Фішера, пошук з обліком фрактальної розмірності й т.д.

– Наведений обчислювальний експеримент показує, що ГА може бути використаний для стиску графічних зображень. Надалі було б цікаво зрівняти ефективність ГА з іншими алгоритмами пошуку перетворення, що кодує.

**Розробка структурної схеми**

Структурна схема системи наведена на рисунку 3. З нього видно, що система складається з наступних структурних блоків:

– Вхідне зображення.
– Блок параметрів стиснення.
– Блок стиснення за допомогою фракталів.
– Кодер.
– Блок запису у файл стисненого зображення.
– Блок зберігання стисненого зображення.
– Блок читання з файлу.
– Декодер.
– Блок декомпресії за допомогою фракталів.
– Вихідне зображення.

Перед тим, як перейти до докладного опису алгоритмів фрактального кодування, коротко перелічимо особливості фрактального кодування:

– У технології фрактального кодування закладений великий потенціал, але вона не стандартизована.

– Відноситься до методу стиску із втратами (дані не відновлюються у вихідному виді).

– При стиску використовуються системи ітеруємих функцій.

– Компресія даних повільна, декодування – швидке.

– Ми можемо вибирати великий діапазон значень дозволу зображень, досягаючи теоретично високих показників стиску.

– Технологія запатентована.

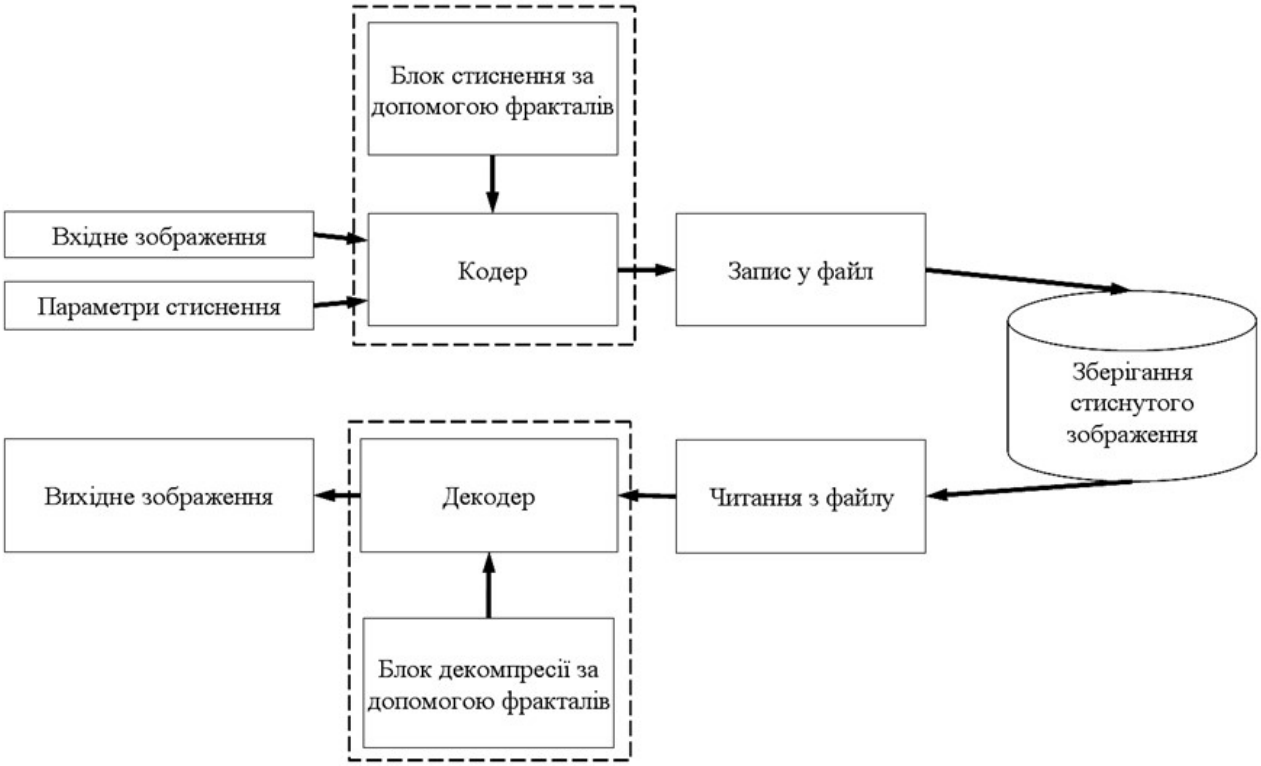


Рисунок 3 – Структурна схема системи

Після робіт Бенуа Мандельброта математики продовжували пошук підстав для застосування фрактальної геометрії. Так, Джон Хатчинсон в 1981 році розробив апарат теорії ітерованих функцій. Пізніше Майкл Барнсли написав книгу «Фракталы всюди» (*Fr*A*c*t*A*ls Ever*yw*here, 1988), що є однієї із ключових робіт із фрактальному стиску. У книзі викладені математичні основи систем ітеруємих функцій, головним результатом є теорема колажу (яка пояснює, які умови необхідні для відтворення зображення).

Фрактальна математика (пряма завдання) добре підходить для моделювання ландшафтів. Зворотне завдання полягає в тім, щоб застосувати системи ітеруємих функцій для стиску зображень.

Еволюція розвитку алгоритмів фрактального кодування складається із двох частин: класична (Майкл Барнсли й Алан Слоун) і сучасна (Майкл Барнсли й Арно Жакан (A. Jaquin)).

На сьогодні відомо кілька основних алгоритмів фрактального кодування – базовий алгоритм, алгоритм кодування FE (з виділенням характеристичних особливостей і ряд інших алгоритмів).

Зображення (у градаціях сірого) інтерпретуються як речовинні функції $f(x,y)$, певні на одиничному квадраті $I^2 = IxI$. Тобто, можна записати, що

$$f : I^2 \to \{1,2,...,N\},$$

тут N – число відтінків сірого. Взагалі, робота із зображеннями представляється особливо трудомісткою, коли говоримо про кольорові зображення, оскільки в них точка представляється у вигляді сукупності трьох основних квітів (червоних, зелених і синього) або їхніх похідних. У зображеннях у градаціях сірого кожна точка характеризується відтінком сірого, яскравістю (діапазон від 0 до 255). Тому спочатку розглядають роботу з напівтоновими зображеннями, а потім переходять до кольорових зображень.

Можна ввести метрику на цих функціях (тому що ми працюємо з метричними просторами) у такий спосіб:

$$d_2(f,g)=\left(\int\limits_{I^2}\left|f(x,y)-g(x,y)\right|^2 dxdy\right)^{1/2}.$$

Якщо функції безперервні, але оскільки ми будемо працювати з повним метричним простором $F$, певним на одиничному квадраті, а зображення є цифровими, то інтеграл убирається, співвідношення (9) записується в наступному вигляді:

$$d_{rms}=\left[\sum_{i=1}^{n}\sum_{j=1}^{m}\left|f(x_i,y_j)-g(x_i,y_j)\right|^2\right]^{1/2}.$$

Цифрові зображення являють собою матрицю фіксованих значень функції $f(x,y)$, узятих у фіксованих точках $f(x_i,y_j)$. Наведене співвідношення (10) називається середньоквадратичним відхиленням (root meAn square). Цей показник (крім стиску зображень) може використовуватися як міра мінливості значень ознак, ступеня відхилення бажаних показників від спостережуваних. Він буде використовуватися в тому числі й для оцінки ефективності кодування зображень.

У фрактальному кодуванні використовується система ітеруємих функцій, більше загального виду. Вона називається кусочно-визначена система ітеруємих функцій (PIFS). Цей тип системи ітеруємих функцій складається з повного метричного простору $X$, набору підобластей і набору стискаючих відображень.

Ми також можемо визначити афінні перетворення, які переводять у себе одиничний квадрат $I^2 \rightarrow I^2$ у такий спосіб:

$$\overline{w_i}(x,y)=A_i\begin{pmatrix}x\\y\end{pmatrix}+b_i.$$

Тут $A_i$ – матриці перетворень розміром $2x2$, а $b_i$ – вектора зрушення (словом, тут звичайне афінне перетворення). А тепер можна записати відображення $w_i$: $F \rightarrow F$ у загальному виді.

$$\overline{w_i}(f)(x,y)=s_i f(\overline{w_i^{-1}}(x,y))+o_i,$$

за умови, що перетворення $\overline{w_i}$ оборотне й $(x,y)\in R_i$. Константа $s_i$ розширює (або звужує) діапазон значень функції $f$, тобто управляє контрастністю (для зображень у градаціях сірого). Величина $o_i$ відповідає за зміну яскравості зображення. Перетворення $\overline{w_i}$ називається просторовою складовою перетворення $w_i$. Задане співвідношення (12) є базовим для зображень у градаціях сірого, ми його будемо використовувати при стиску.

У випадку використання кусочно-определенной системи ітеруємих функцій фіксована точка (або аттрактор) є зображенням $f$, для якої виконується $W(f)=f$. Теорема про стискаючі відображення говорить, що $W$ у результаті буде зображенням, а ми зможемо порахувати послідовності $W(f_0)$, $W(W(f_0))$, $W(W(W(f_0)))$, де $f_0$ – яке-небудь зображення. Оскільки ми знаємо, що відображення $W$ – стискаюче на просторі зображень, то ми в результаті одержимо єдину фіксовану точку, що є якимсь зображенням.

Припустимо, що нам необхідно закодувати зображення $f$. Це означає, що нам необхідно визначити набір перетворень $w_1, w_2, \ldots, w_N$, і при цьому:

$$W=\bigcup_{i=1}^{N} w_i, f=x_w.$$

Нам потрібно, щоб *f* була нерухливою точкою перетворення *W*. Співвідношення для фіксованих точок показує, як нам можна цього досягти:

$$f = W(f) = w_1(f) \cup w_2(f) \cup ... \cup w_N(f).$$

Ми намагаємося знайти розбивку *f* на області, у результаті якого кожна область є зменшеною копією цілого зображення.

Мінімізація цього рівняння означає наступне: ми, по-перше повинні добре підібрати область $D_i$ таким чином, щоб між областями був достатній збіг. По-друге, необхідно знайти гарні значення контрасту і яскравості (співвідношення 12), тобто значення коефіцієнтів $s_i$ і $o_i$, для перетворення $w_i$. Для кожної області $D_i$ ми зможемо порахувати, використовуючи аналітичні методи, значення коефіцієнтів, у такий спосіб досягається як можна менше значення величини $d_{rms}$.

Приведемо базовий алгоритм фрактального кодування.

1.Розбиваємо зображення *f* на непересічні рангові блоки $\{R_i\}$. Рангові блоки являють собою прямокутники (у найпростішому випадку квадрати), але можуть використовуватися й інші способи розбивки, наприклад, як рангові області можуть використовуватися трикутники. Блоки можуть бути однаковими, а може використовуватися адаптивна розбивка – методом квадродерева.

2. Покриваємо зображення послідовністю доменних блоків, що можливо перекриваються. Домени можуть бути різних розмірів, звичайно їхня кількість обчислюється сотнями й тисячами.

Домени можуть перекриватися, а можуть розміщатися на деякій відстані друг від друга. Доменне зображення темніше, ніж вихідне, оскільки яскравість при перетвореннях змінюється. Афінне перетворення записується в такий спосіб (для зображень у градаціях сірого):

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = w_i \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_i & b_i & 0 \\ c_i & d_i & 0 \\ 0 & 0 & s_i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \\ o_i \end{bmatrix}$$

Величина $s_i$ означає яскравість пікселя й діє подібно кнопці настроювання контрасту. Коли цей параметр дорівнює 0, те пікселі домену перетворяться в чорний колір, якщо параметр дорівнює 1, то домен залишається таким же, якщо параметр установлений у межах від 0 до 1 (найчастіше беруть 0.75), чим значення більше, тим більше контраст області. Параметр $o_i$ відповідає за світність пікселя і його зміна аналогічно зміні настроюванню яскравості. Позитивні значення роблять блок пікселів більше світлим, негативні значення – більше темним. Коли їсти можливість контролювати показники контрасту і яскравості, то можна здійснити розширене афінне перетворення, що відображає доменні блоки в рангові блоки.

Матрична частина перетворення відповідає за поворот і зміну яскравості зображення (коефіцієнти $A_i$, $b_i$, $c_i$, $d_i$ і $s_i$), а вектор $[e_i, f_i, o_i]$ за зрушення й зміну контрасту.

Зміна яскравості доменного зображення відбувається в такий спосіб: після його одержання яскравість всіх пікселів множиться на деяку величину, розповсюджене значення – 0.75.

3.Для кожного рангового блоку знаходимо домен і відповідне перетворення, що щонайкраще перекриває ранговий блок. Настроюються параметри перетворення – яскравість і контраст, що забезпечує найкращу відповідність.

При використанні даного підходу до кодування розмір доменних блоків завжди вдвічі більше, ніж розмір регіонів. Якщо ранговий блок має розмір 8*x*8, то домен завжди 16*x*16.

Одним із ключових параметрів є зсув доменів (домени можуть перекриватися, а можуть розташовуватися друг від друга на деякій відстані). Процес фрактального стиску полягає в пошуку самоподібних областей зображення. У цьому випадку послідовно

перебираються всі регіони, і для кожного регіону підбирається найбільш схожий на нього доменний блок. Таким чином, застосовується сукупність перетворень: порівняння блоків по пікселям і афінні перетворення. Найпоширеніші наступні:

1. Поворот на 0 градусів.
2. Поворот на 90 градусів.
3. Поворот на 180 градусів.
4. Поворот на 270 градусів.
5. Симетрія щодо осі *X*.
6. Симетрія щодо осі *Y*.
7. Симетрія щодо головної діагоналі.
8. Симетрія щодо побічної діагоналі.

Від характеру вихідного зображення залежить, які перетворення можна виконувати. Можна вибрати всі перетворення, а можна – виконати тільки перетворення повороту або тільки симетрію. Але рекомендується виконати всі 8 перетворень.

У результаті кодування у файл записується код зображення:

1. Кількість регіонів по горизонталі й вертикалі.
2. Розмір регіону.
3. Коефіцієнти афінних перетворень:

– Координати домену.
– Номер афінного перетворення.
– Різницю усередненої яскравості між регіоном і доменом.

У файл, таким чином, зберігаються тільки числові коефіцієнти, а не саме зображення. Чисел досить для розпакування (виконання зворотних перетворень, при декодуванні 2 і 4 перетворення міняються місцями).

**Висновки**. У статті наведені теоретичне узагальнення й рішення наукового завдання дослідження методів стиску зображень за допомогою фракталів. Рішення даного завдання полягало у вирішенні наступних задач: Був проведений огляд існуючих систем стиску зображень за допомогою фракталів; Досліджена система стиску зображень за допомогою фракталів; На основі отриманих результатів досліджень створена програмна реалізація системи стиску зображень за допомогою фракталів. Розроблені під час виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання стиску зображень за допомогою фракталів. Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

## Список літератури


1. Smirnov, O., Odarchenko, R., Smirnova, T., Bondar, S., Volosheniuk, D. «Optimal Structure Construction of Private 5G Network for the Needs of Enterprises». Lecture Notes on Data Engineering and Communications Technologies, 2023, 178, pp. 208–223.
2. Smirnov, O., Neskorodieva, T., Fedorov, E., Rudakov, K., Neskorodieva, A. «Method Detection Audit Data Anomalies on Basis Restricted Cauchy Machine» CEUR Workshop Proceedings, Volume 3187, 2022,
3. Smirnov O., Smirnova T., Anas M. Al-Oraiqat, Drieiev O., Polishchuk L., Sheroz Khan, Yassin M. Y. Hasan, Aladdein M. Amro, Hazim S. AlRawashdeh «Method for Determining Treated Metal Surface Quality Using Computer Vision Technology». Sensors (Basel, Switzerland) Volume 22, Issue 16, 6223, 2022.
4. Smirnov O., Kuznetsov A., Kryvinska N., Kiian A., Kuznetsova K. «Full Non-Binary Constant-Weight Codes». SN Computer Science, Vol 2, 337, 2021. https://doi.org/10.1007/s42979-021-00739-w
5. Smirnov O., Kuznetsov A., Zhora V., Onikiychuk A., Pieshkova O. «Hiding Messages in Audio Files Using Direct Spread Spectrum». 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2021, Cracow, Poland, 22-25 September 2021. P. 414-418.
6. Smirnov O., Kuznetsov A., Lokotkova I., Kuznetsova T., Florov S., Lebid O. «Using Orthogonal Signals to Hide Information in Images». 4 IEEE International Conference on Advanced Information and Communication Technologies (AICT) – 2021, Lviv, Ukraine, September 21-25, 2021. P. 255-260.
7. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering

functions in the framework of nonlinear-feedback shift register». International Journal of Computing; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256.
8. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». CEUR Workshop Proceedings. Volume 2740, 2020, Pages 102-114.
9. Smirnov O., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». Journal of theoretical and applied information technology Vol.98. No 21, 2020, P. 3334-3346.
10. Smirnov O., Kuznetsov A., Kovalchuk D., Kuznetsova T. «New technique for data hiding in cover images using adaptively generated pseudorandom sequences». CEUR Workshop Proceedings Volume 2654, 2020, Pages 1-14.
11. Смірнов О.А., Дрєєва Г.М., Дрєєв О.М., Смірнова Т.В. «Фрактальний аналіз генератора самоподібного трафіку на основі ланцюга Маркова». Центральноукраїнський науковий вісник. Технічні науки. № 2(33). с. 161-172, 2019.
12. О. Смірнов, Є. Деменко, О. Онікійчук, А. Арищенко, Л. Горбачова, «Формування псевдовипадкових послідовностей для приховування даних в зображеннях» Комп'ютерні науки та кібербезпека. № 4. С. 30-37. 2019.
13. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В. Поліщук Л.І. Проектування комп'ютерних систем та мереж. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2019. – 264 с.
14. Smirnov, O., Kuznetsov, A., Kuznetsova., K. Synthesis of Discrete Signals with Improved Correlation Properties. Монографія: In.: ISCI'2019: Information Security in Critical Infrastructures. Collective monograph. Edited by Ivan D. Gorbenko and Alexandr A. Kuznetsov, ASC Academic Publishing, USA, 2019, pp. 281-299. – ISBN: 978-0-9989826-8-7 (Hardback), ISBN: 978-0-9989826-9-4 (Ebook).
15. Смірнов О.А., Дрєєва Г.М. Метод генерування фрактального трафіку за допомогою моделі генератора на графі. Монографія: Інформаційна безпека та інформаційні технології монографія / за заг. ред. В. С. Пономаренка. – Х: Вид. Рожко С.Г. 2019. С. 123-139
16. Дрєєва Г.М., Смірнов О.А., Дрєєв О.М. Метод генерування фрактальноподібної числової послідовності на основі скінченного автомату для моделювання трафіку у мережі. Центральноукраїнський науковий вісник. Технічні науки. № 1(32). с. 173-183, 2019.
17. Смірнов О.А., Дрєєв О.М. Порівняння бітових щільностей при використанні різних методів кодування інформації. Збірник наукових праць "Системи обробки інформації". - Випуск 2 (118). т.2. - Х.: ХУПС - 2014. - С. 64-67
18. Смірнов О.А., Дрєєв О.М. Порівняння бітових щільностей при використанні різних методів кодування інформації. Збірник тез VI міжнародної науково-практичної конференції "Проблеми та перспективи розвитку ІТ-індустрії". м. Харків. 17-18 квітня 2014р. – Харків: ХНЄУ. - 2014. - С. 240.
19. Смірнов О.А., Дрєєв О.М., Доренський О.П. «Дослідження впливу ступеня стиснення зображень на оперативність їх доставки у телекомунікаційній системі. Збірник наукових праць "Системи обробки інформації". – Випуск 8(115). – Х.: ХУПС – 2013. – С. 234-239.
20. Смірнов О.А., Доренський О.П., Дрєєв О.М. Аналіз процесів стиснення та відновлення зображень на основі цифрових методів. Наука і техніка Повітряних сил Збройних Сил України. – Випуск 3(12). – Х.: ХУПС. – 2013. – С.122-127.
21. Смірнов О.А., Мелешко Є.В., Семенов С.Г. Методи та засоби обробки сигналів і даних в інформаційних системах. Навчальний посібник. – Кіровоград: КНТУ 2012. – 250 с.