УДК 004.932:51-7
DOI: 10.31866/2617-796x.2.1.2019.175653

Коцюбівська Катерина,
кандидат технічних наук, доцент,
Київський національний університет культури і мистецтв,
Київ, Україна
katysivak@gmail.com
http://orcid.org/0000-0001-6911-2770

Тимошенко Вікторія,
магістрант кафедри комп'ютерних наук,
Київський національний університет культури і мистецтв,
Київ, Україна
zbyrko.victoria@gmail.com
https://orcid.org/0000-0003-4622-3114

МАТЕМАТИЧНІ МЕТОДИ ОБРОБКИ ЗОБРАЖЕНЬ

Мета дослідження. Вичення специфіки кодування зображень методом сплайнової інтерполяції, та порвняння вказаного методу з іншими математичними методами кодування та обробки зображень.

Методи дослідження математичні та алгоритмічні моделі та методи розв'язку задачі згладжування на основі сплайн-апроксимації, а також можливість застосування відповідного математичного апарату до кодування та обробки зображень.

Новизною дослідження є виокремлення алгоритму стиснення зображень на основі методів сплайнової апроксимації. Такий підхід до обробки зображень дозволяє не тільки зменшити розміри файлів зображень, але й обирати необхідну якість відновлення, залежно від подальшого використання зображення.

Висновки. В роботі проведено порівняння існуючих методів кодування зображень, та вказано на переваги використання сплайнової інтерполяції при кодуванні і декодуванні зображень.

Ключові слова: сплайн; сплайн-апроксимація; сплайн-інтерполяція; згладжування; кодування; декодування; обробка зображень.

© Коцюбівська К. І.
© Тимошенко В. В.

Вступ. Технологія обробки цифрових зображень досягла останнім часом великих успіхів завдяки тому, що цифрові методи обробки зображень мають ряд переваг перед аналоговими. Однак, при цьому виникає нова проблема – великі розміри файлів, які підлягають збереженню, передачі і обробці.

Розв'язком проблеми є зменшення обсягу інформації шляхом стиснення даних, або іншими словами кодування зображень.

В зображеннях розрізняють два види надлишковості:

– статична надлишковість;

– фізіологічна надлишковість.

Перша пов'язана з тим, що інформація в зображенні не носить випадкового характеру. Сусідні відліки часто мають подібні значення яскравості, в чому і виявляється властива їм просторова кореляція. Якщо відповідним чином використовувати цю властивість, то можна різко зменшити число біт для представлення зображення в цифровій формі.

Фізіологічна надлишковість обумовлена недосконалістю зорового аналізатора людини, не здатного сприймати (розрізняти) всю інформацію, яка виводиться. Зменшення фізіологічної надлишковості в більшій степені скорочує статистичну надлишковість і навпаки.

Задачею роботи буде дослідження специфіки одного з напрямків кодування зображень – кодування методом сплайнової інтерполяції.

Результати дослідження. Найбільш відомий і простий алгоритм стиснення інформації оборотним шляхом – це кодування серій послідовностей (Run Length Encoding – RLE). Суть методів даного підходу складається в заміні ланцюжків чи серій байтів, які повторюються, чи їхніх послідовностей на один байт, що кодує, і лічильник числа їхніх повторень. Проблема всіх аналогічних методів полягає лише у визначенні способу, за допомогою якого алгоритм, що розпаковує, міг би відрізнити в результуючому потоці байтів кодовану серію від інших – некодованих послідовностей байтів. Рішення проблеми досягається звичайно проставленням міток спочатку кодованих ланцюжків. Такими мітками можуть бути, наприклад, характерні значення бітів у першому байті кодованої серії, значення першого байта кодованої серії і т. п. Дані методи, як правило, досить ефективні для стиснення растрових графічних зображень (BMP, PCX, TIF, GIF), тому що останні містять досить багато довгих серій повторюючихся послідовностей байтів. Недоліком методу RLE є досить низький степінь стиснення чи вартість кодування файлів з малим числом серій і, що ще гірше – з малим числом повторюваних байтів у серіях.

Кращий, середній і гірший коефіцієнти стиснення – 1/32, 1/2, 2/1. Ситуація, коли файл збільшується в два рази, для цього простого алгоритму не така вже й рідкісна. Її можна легко одержати, застосовуючи групове кодування до оброблених кольорових фотографій.

До позитивних сторін алгоритму, мабуть, можна віднести тільки те, що він не вимагає додаткової пам'яті при роботі, і швидко виконується. Орієнтовано алгоритм на зображення з невеликою кількістю кольорів: ділову і наукову графіку. Застосовується у форматах PCX, TIFF, BMP. Цікава особливість групового ко-

дування в PCX полягає в тому, що степінь архівації для деяких зображень може бути істотно підвищена усього лише за рахунок зміни порядку кольорів у палітрі зображення.

Стиснення без застосування методу RLE. Процес стиснення даних без застосування методу RLE можна розбити на два етапи – моделювання (modelling) і власне кодування (encoding). Ці процеси і їхні реалізуючі алгоритми досить незалежні і різнопланові.

Під кодуванням звичайно розуміють обробку потоку символів (у нашому випадку байтів чи напівбайтів) у деякому алфавіті, причому частоти появи символів у потоці різні. Метою кодування є перетворення цього потоку в потік біт мінімальної довжини, що досягається зменшенням ентропії вхідного потоку шляхом обліку частот символів. Довжина коду, що представляє символи з алфавіту потоку повинна бути пропорційна обсягу інформації вхідного потоку, а довжина символів потоку в бітах може бути не кратною 8 і навіть перемінною. Якщо розподіл ймовірностей частот появи символів з алфавіту вхідного потоку відомо, то можна побудувати модель оптимального кодування. Однак, через існування величезного числа різних форматів файлів задача значно ускладнюється тому що розподіл частот символів даних заздалегідь невідомо. У такому випадку, у загальному вигляді, використовуються два підходи.

Перший полягає в перегляді вхідного потоку і побудові кодування на підставі зібраної статистики (при цьому потрібно два проходи по файлі – один для перегляду і збору статистичної інформації, другий – для кодування, що трохи обмежує сферу застосування таких алгоритмів, тому що, таким чином, виключається можливість однопроходного кодування «на льоту», застосовуваного в телекомунікаційних системах, де й обсяг даних, деколи не відомий, а їхня повторна передача або розбір може зайняти невиправдано багато часу). У вихідний потік записується статистична схема використаного кодування. Даний метод відомий як статичне кодування Хаффмена (англ. Huffman).

Другий метод – метод адаптивного кодування (adaptive coder method). Його загальний принцип полягає в тому, щоб змінювати схему кодування в залежності від характеру змін вхідного потоку. Такий підхід має однопрохідний алгоритм і не вимагає збереження інформації про використане кодування в явному вигляді. Адаптивне кодування може дати великий степінь стиснення, у порівнянні зі статичним, оскільки більш повно враховуються зміни частот вхідного потоку. Даний метод відомий як динамічне кодування Хаффмена (Білинський, Огородник та Юкиш, 2018).

У статичному кодуванні Хаффмена вхідним символам (ланцюжкам бітів різної довжини) ставляться у відповідність ланцюжка бітів, також, перемінної довжини – їхні коди. Довжина коду кожного символу береться пропорційної двійковому логарифму його частоти, узятому зі зворотним знаком. А загальний набір усіх різних символів, що зустрілися, складає алфавіт потоку. Це кодування є префіксним, що дозволяє легко його декодувати результативний потік, тому що при префіксному кодуванні код будь-якого символу не є префіксом коду ніякого іншого символу – алфавіт унікальний.

При використанні адаптивного кодування Хаффмена ускладнення алгоритму полягає в необхідності постійного коректування дерева і кодів символів основного алфавіту відповідно до статистики вхідного потоку, що змінюється.

Методи Хаффмена дають досить високу швидкість і помірно гарну якість стиснення. Ці алгоритми давно відомі і широко застосовуються як у програмних (усілякі компресори, архіватори і програми резервного копіювання файлів і дисків), так і в апаратних (системи стиснення «прошиті» у модеми і факси, сканери) реалізаціях.

Однак, кодування Хаффмена має мінімальну надмірність за умови, що кожен символ кодується в алфавіті коду символу окремим ланцюжком із двох біт – {0, 1}. Основним же недоліком даного методу є залежність ступеня стиснення від наближеності ймовірностей символів до 2 у деякому негативному ступені, що зв'язано з тим, що кожен символ кодується цілим числом біт. Так при кодуванні потоку з двосимвольним алфавітом стиснення завжди відсутнє, тому що незважаючи на різні імовірності появи символів у вхідному потоці алгоритм фактично зводить їх до 1/2.

Дана проблема, як правило, розв'язується шляхом введення в алфавіт вхідного потоку нових символів виду 'ab', 'abc',... і т. п., де a, b, c – символи первинного вихідного алфавіту. Такий процес називається сегментацією чи блокуванням вхідного потоку. Однак, сегментація не дозволяє цілком позбутися від втрат у стисненні (вони лише зменшуються пропорційно розміру блоку), але приводить до різкого росту розмірів дерева кодування, і, відповідно, довжині коду символів вторинних алфавітів. Так, якщо, наприклад, символами вхідного алфавіту є байти зі значеннями від 0 до 255, то при блокуванні по два символи ми одержуємо 65 536 символів (різних комбінацій) і стільки ж листків дерева кодування, а при блокуванні по три – 16 777 216! Звичайно, при такому ускладненні, відповідно зростуть вимоги і до пам'яті і часу побудови дерева, а при адаптивному кодуванні – і до часу відновлення дерева, що приведе до різкого збільшення часу стиснення. Напроти, у середньому, утрати складуть 1/2 біта на символ при відсутності сегментації, і 1/4 чи 1/6 біта відповідно при її наявності, для блоків довжиною 2 і 3 біти.

Арифметичне кодування. Зовсім інше рішення пропонує так зване арифметичне кодування (англ. Witten). Арифметичне кодування є методом, що дозволяє упаковувати символи вхідного алфавіту без утрат за умови, що відомо розподіл частот цих символів і він є найбільш оптимальним, тому що досягається теоретична границя ступеня стиснення.

Передбачувана необхідна послідовність символів, при стиснення методом арифметичного кодування розглядається як деяка двійкова дріб з інтервалу [0, 1]. Результат стиснення представляється як послідовність двійкових цифр із запису цього дробу. Ідея методу полягає в наступному: вихідний текст розглядається як запис цього дробу, де кожен вхідний символ є «цифрою» з вагою, пропорційною імовірності його появи. Цим пояснюється інтервал, що відповідає мінімальній і максимальній ймовірностям появи символу в потоці.

При розробці цього методу виникають дві проблеми: по-перше, необхідна арифметика з плаваючою точкою, теоретично, необмеженої точності, і, по-друге, результат кодування стає відомий лише при закінченні вхідного потоку. Однак, подальші дослідження показують, що можна практично без втрат обійтися цілочисленою арифметикою невеликої точності (16–32 розряду), а також домогтися інкрементальної роботи алгоритму: цифри коду можуть видаватися послідовно в міру читання вхідного потоку при обмеженні числа символів вхідного ланцюжка яким-небудь розумним числом (Кодування графічних даних, б.р.).

Кодування сортуванням. Тут не можна не згадати простий і досить ефективний метод кодування джерела з невідомим розподілом частот, відомий як стиснення за допомогою «стопки книг» чи як стиснення чи сортуванням хешуванням. Ідея методу полягає в наступному: нехай алфавіт джерела складається з N символів з номерами 1, 2, ..., N. Алгоритм, що кодує, зберігає послідовність символів, що представляє собою деяку перестановку символів у послідовності первинного вхідного алфавіту. При надходженні на вхід деякого символу c, що має в цій переставленій послідовності номер i, що кодує алгоритм записує код цього символу (наприклад, монотонний код). Надійшовший потім символ переставляється в початок послідовності і номера всіх символів, що стоять перед c, збільшуються на 1. Таким чином, символи що найбільш часто зустрічаються будуть переходити в початок списку і мати більш короткі коди, що у свою чергу знизить обсяг вихідного потоку при їхньому записі як символи вихідного потоку.

Двоступінчасте кодування. Алгоритм Лемпеля-Зіва. Усі розглянуті вище методи і моделі кодування припускали в якості вхідних даних ланцюжки символів (тексти) у деякому кінцевому алфавіті. При цьому залишалося відкритим питання про зв’язок цього вхідного алфавіту алгоритму, що кодує, з даними, які підлягають упаковці (звичайно також представленими у вигляді ланцюжків в алфавіті (при байтові організації звичайно складається з 256 різних символів – значень байт).

У найпростішому випадку для кодування як вхідний алфавіт, можна використовувати саме ці символи (байти) вхідного потоку. Ступінь стиснення при цьому відносно невелика – для текстових файлів порядку 50%. Набагато більшої ступені стиснення можна домогтися при виділенні з вхідного потоку повторюваних ланцюжків – блоків, і кодування посилань на ці ланцюжки з побудовою хеш таблиць від першого до n-го рівня.

Метод, про який і піде мова, належить Лемпелю і Зиву, і звичайно називається LZ-compression. Суть його полягає в наступному: пакувальник постійно зберігає деяку кількість останніх оброблених символів у буфері. По мірі обробки вхідного потоку знову надійшовші символи попадають у кінець буфера, зсовуючи попередні символи і витісняючи самі старі. Розміри цього буфера варіюються в різних реалізаціях систем, що кодують. Потім, після побудови хеш таблиць алгоритм виділяє (шляхом пошуку в словнику) саму довгу початкову підстроку вхідного потоку, що збігається з однією з підстрок у словнику, і видає на вихід пари (length, distance), де length – довжина знайденої в словнику підстроки, а distance – відстань від неї до вхідної підстроки (тобто фактично індекс підстроки в буфері, від-

нятий з його розміру). У випадку, якщо така підстрока не знайдена, у вихідний потік просто копіюється черговий символ вхідного потоку (Лутчин та Лутчин, 2011).

У первісній версії алгоритму пропонувалося використовувати найпростіший пошук по всьому словнику. Час стиснення при такій реалізації був пропорційний добутку довжини вхідного потоку на розмір буфера, що зовсім непридатне для практичного використання. Однак, надалі, було запропоновано використовувати двійкове дерево і хешування для швидкого пошуку в словнику, що дозволило на порядок підвищити швидкість роботи алгоритму.

Таким чином, алгоритм Лемпеля-Зіва перетворює один потік вихідних символів у два паралельних потоки довжин і індексів у таблиці (length + distance). Очевидно, що ці потоки є потоками символів із двома новими алфавітами, і до них можна застосувати один із згадуваних вище методів (RLE, кодування Хаффмена чи арифметичне кодування). Так ми приходимо до схеми двоступінчастого кодування – найбільш ефективної з практично використовуваних у даний час. При реалізації цього методу необхідно домогтися узгодженого виведення обох потоків в один файл. Ця проблема звичайно зважується шляхом почергового запису кодів символів з обох потоків.

Алгоритм Лемпеля-Зіва-Велча (Lempel-Ziv-Welch – LZW). Даний алгоритм відрізняють висока швидкість роботи як при упакуванні, так і при розпакуванні, досить скромні вимоги до пам’яті і проста апаратна реалізація. Недолік – низький ступінь стиснення в порівнянні зі схемою двоступінчастого кодування. Припустимо, що в нас є словник, що зберігає рядки тексту і містить порядку від 2-х до 8-ми тисяч пронумерованих гнізд. Запишемо в перші 256 гнізд рядки, що складаються з одного символу, номер якого дорівнює номеру гнізда. Алгоритм переглядає вхідний потік, розбиваючи його на підстроки і додаючи нові гнізда в кінець словника. Прочитаємо кілька символів у рядок s і знайдемо в словнику рядок t – самий довгий префікс s. Нехай він знайдений у гнізді з номером n. Виведемо число n у вихідний потік, перемістимо покажчик вхідного потоку на length(t) символів вперед і додамо в словник нове гніздо, що містить рядок t+c, де c – черговий символ на вході (відразу після t). Алгоритм перетворить потік символів на вході в потік індексів осередків словника на виході. При розмірі словника в 4096 гнізд можна передавати 12 біт на кожен індекс. Кожен розпізнаний ланцюжок додає в словник одне гніздо. При переповненні словника пакувальник може або припинити його заповнення, або очистити (цілком чи частково).

При практичній реалізації цього алгоритму варто врахувати, що будь-яке гніздо словника, крім найперших, утримуючих односимвольні ланцюжки, зберігає копію деякого іншого гнізда, до якої в кінець приписаний один символ. Унаслідок цього можна обійтися простою обліковою структурою з одним зв’язком.

JBIG. Алгоритм розроблений групою експертів ISO (Joint Bi-level Experts Group) спеціально для стиснення однобітних чорно-білих зображень. Наприклад, факсів чи відсканованих документів. У принципі може застосовуватися і до 2-х, і до 4-х бітових картинок. При цьому алгоритм розбиває їх на окремі бітові площини. JBIG дозволяє керувати такими параметрами, як порядок розбивки зображення на бітові площини, ширина смуг у зображенні, рівні маштабування. Остання

можливість дозволяє легко орієнтуватися в базі великих по розмірам зображень, переглядаючи спочатку їхні зменшені копії. Набудовуючи ці параметри, можна використовувати цікавий ефект при одержанні зображення по мережі або по будь-якому іншому каналу, пропускна здатність якого мала в порівнянні з можливостями процесора. Розпаковуватися зображення на екрані буде поступово, як би повільно «виявляючись». При цьому людина починає аналізувати картинку задовго до кінця процесу розархівації.

Алгоритм побудований на базі Q-кодувальника, патентом на який володіє IBM. Q-кодер також, як і алгоритм Хаффмана, використовує для тих символів, що частіше з'являються короткі ланцюжки, а для тих, що рідше з'являються – довгі. Однак, на відміну від нього, в алгоритмі використовуються і послідовності символів. Характерною рисою JBIG є різке зниження ступеня стиснення при підвищенні рівня шумів вхідної картинки.

Lossless JPEG. Цей алгоритм, розроблений групою експертів в області фотографії (Joint Photographic Expert Group). На відміну від JBIG, Lossless JPEG орієнтований на повнокольорові 24-бітні безпалітрові зображення. Він являє собою спеціальну реалізацію JPEG без втрат. Коефіцієнти стиснення: 1/20, 1/2, 1. Lossless JPEG рекомендується застосовувати в тих додатках, де необхідно побітова відповідність вихідного і розархівованого зображень.

Спробуємо на цьому етапі зробити деякі узагальнення. З одного боку, приведені алгоритми досить універсальні і покривають усі типи зображень, з іншого боку – у них, по сьогоднішніх мірках, занадто маленький коефіцієнт архівації. Використовуючи один з алгоритмів стиснення без втрат, можна забезпечити коефіцієнт архівації зображення приблизно в два рази. У той же час алгоритми стиснення з втратами оперують з коефіцієнтами 10–200 разів. Крім можливості модифікації зображення, одна з основних причин подібної різниці полягає в тому, що традиційні алгоритми орієнтовані на роботу з ланцюжком. Вони не враховують так називану «когерентність областей» у зображеннях. Ідея когерентності областей полягає в малій зміні кольору і структури зображення на невеликій ділянці. Всі алгоритми, про які мова йтиме нижче, були створені пізніше спеціально для стиснення графіки і використовують цю ідею (Кодування графічних даних, б.р.).

Слід зазначити, що й у класичних алгоритмах можна використовувати ідею когерентності. Існують алгоритми обходу зображення по «фрактальні» кривої, при роботі яких воно також витягається в ланцюжок; але за рахунок того, що крива оббігає області зображення по складній траєкторії, ділянки близьких кольорів у ланцюжку, що виходить, подовжуються.

Рекурсивне стиснення. Цей вид архівації відомий досить давно і прямо виходить з ідеї використання когерентності областей. Орієнтовано алгоритм на кольорові і чорно-білі зображення з плавними переходами. Ідеальний для картинок типу рентгенівських знімків. Коефіцієнт стиснення задається і варіюється в межах 5–20 разів. При спробі задати більший коефіцієнт, на різких границях, що особливо проходять по діагоналі, виявляється «сходовий ефект» – сходинки різної яскравості, розміром у кілька пікселів. У якомусь розумінні рекурсивний стиснення є частковим випадком JPEG.

JPEG – один із самих нових і досить могутніх алгоритмів. Практично він стає стандартом де-факто для повнокольорових зображень. Оперує алгоритм областями 8x8, на яких яскравість і колір міняються порівняно плавно. Унаслідок цього, при розкладанні матриці такої області в подвійний ряд по косинусах значимими виявляються тільки перші коефіцієнти. Таким чином, стиснення у JPEG здійснюється за рахунок малої величини значень амплітуд високих частот у реальних зображеннях.

Коефіцієнт архівації в JPEG може змінюватися в межах від 2 до 200 разів. Як і в будь-якого іншого алгоритму стиснення з втратами, у JPEG свої особливості. Найбільш відомі – «ефект Гіббса» і дроблення зображення на квадрати 8x8. Перший виявляється біля різких границь предметів, утворюючи своєрідний «ореол». Він добре помітний, якщо, припустимо, поверх фотографії зробити напис кольором, що сильно відрізняється від тіла. Розбивка на квадрати відбувається, коли задається занадто великий коефіцієнт архівації для даної конкретної картинки.

Не дуже приємною властивістю JPEG є також те, що нерідко горизонтальні і вертикальні смуги на дисплеї абсолютно не видимі, і можуть проявитися тільки при друці у вигляді муарового візерунка. Він виникає при накладенні похилого растра друку на смуги зображення. Через ці сюрпризи JPEG не рекомендується активно використовувати в поліграфії, задаючи високі коефіцієнти. Однак при архівації зображень, призначених для перегляду людиною, він на даний момент незамінний (Кодування графічних даних, б.р.).

Широке застосування JPEG стримується, мабуть, лише тим, що він оперує 24-бітними зображеннями. Тому для того, щоб із прийнятною якістю подивитися картинку на звичайному моніторі в 256-кольоровій палітрі, потрібно застосування відповідних алгоритмів і, отже, визначений час. У додатках, орієнтованих на причепливого користувача, таких, наприклад, як ігри, подібні затримки неприйнятні. Крім того, якщо наявні у вас зображення, допустимо, у 8-бітному форматі GIF перевести в 24-бітний JPEG, а потому назад у GIF для перегляду, то втрата якості відбудеться двічі при обох перетвореннях. Проте виграш у розмірах архівів найчастіше великий (у 3–20 разів!), а втрати якості настільки малі, що збереження зображень у JPEG виявляється дуже ефективним. JPEG реалізований у форматах JPG і TIFF

Процес кодування по схемі JPEG можна розділити на такі етапи (рис. 1):

Перетворення зображення в оптимальний колірний простір (тільки для кольорових зображень).

Субдискретизація компонент кольорорізницевих сигналів шляхом усереднення груп пікселів (тільки у випадках кодування кольорові зображення).

Виконання ДКП для зменшення надмірності даних зображення.

Квантування кожного блоку коефіцієнтів ДКП із використанням вагових функцій оптимізації з урахуванням сприйняття відеосигналу зоровим аналізатором людини.

Кодування результуючих коефіцієнтів з використанням статистичного кодування Хаффмана.

Критичним, з погляду обчислювальних витрат, є виконання ДКП. Однак, завдяки використанню швидких алгоритмів (таких як і для ДПФ) число арифметичних операцій може бути зменшене в десятки разів.

Рис. 1. Схема кодування зображень за стандартом JPEG

Зображення розбивається на блоки розміром 8x8 (при кодуванні кольорових зображень кожен компонент обробляється окремо). У межах кожного блоку виконується двовимірне ДКП відповідно до виразу:

$$F(u,v)=\frac{1}{4}\cdot C(u)\cdot C(v)\cdot\sum_{i=0}^{7}\sum_{j=0}^{7}f(i,j)\cdot\cos\frac{(2i+1)\cdot u\cdot\pi}{16}\cdot\cos\frac{(2j+1)\cdot v\cdot\pi}{16}$$

де

$$C(x)=\begin{Bmatrix}1/2 & x=0\\ 1 & x\neq 0\end{Bmatrix},$$

u, v = 0,1,2...7....7

При декодуванні обчислюється зворотнє ДКП:

$$f(i,j)=\frac{1}{4}\cdot\sum\sum C(u)\cdot C(v)\cdot F(u,v)\cdot\cos\frac{(2i+1)\cdot u\cdot\pi}{16}\cdot\cos\frac{(2j+1)\cdot v\cdot\pi}{16},$$

i, j = 0,1,2 … 7.

Квантування виконується за рахунок розподілу кожного коефіцієнта ДКП на свій «коефіцієнт квантування» з округленням результату до цілого. Терми більшого порядку квантуються з великим «коефіцієнтом квантування». Крім того, для сигналів яскравості кольору використовуються різні таблиці квантування, тому

що око людини має різну чутливість до яскравості і кольору зображення (Білинський, Огородник та Юкиш, 2018).

На етапі статистичного кодування специфікація JPEG допускає застосування крім алгоритму Хаффмана і інших методів з метою зменшення обсягу інформації.

Серед недоліків JPEG і MPEG кодування слід зазначити такі: недостатньо високі коефіцієнти стиснення складних зображень, неможливість зміни роздільної здатності, втрата достовірності відновлених зображень.

Однак, незважаючи на це методи мають найкращі реально досягнуті характеристики по сукупності таких параметрів як коефіцієнт стиснення, якість, швидкодія і підтримуються основними виробниками комп'ютерної техніки і техніки зв'язку.

Кілька слів необхідно сказати про модифікації цього алгоритму. Хоча JPEG і є стандартом ISO, формат його файлів не був зафіксований. Користуючись цим, виробники використовують свої, несумісні між собою формати, і, отже, можуть змінити алгоритм. Так, внутрішні таблиці алгоритму, рекомендовані ISO, заміняються ними на свої власні. Крім того, легка плутанина присутня при завданні ступеня втрат. Наприклад, при тестуванні з'ясовується, що «відмінна» якість, «100%» і «0 балів» дають істотно розрізняються картинки. При цьому, до речі, «100%» якості не означає стиснення без втрат. Зустрічаються також варіанти JPEG для специфічних додатків.

Ця група алгоритмів, які відносяться до фрактального стиснення, очевидно, є самою перспективною і розвивається зараз найбільш бурхливо. Перші практичні результати були отримані у 1992 році і справили приголомшливе враження. Коефіцієнт стиснення у фрактальних алгоритмів варіюється в межах 2–2000. Причому великі коефіцієнти досягаються на реальних зображеннях, що, узагалі говорячи, нетипово для попередніх алгоритмів. Крім того, при розархівації зображення можна маштабувати. Унікальна особливість цього алгоритму полягає в тому, що збільшене зображення не дробиться на квадрати. В фрактальному стисненні використовується принципово нова ідея – не близькість кольорів у локальній області, а схожість різних по розміру областей зображення. Це, безумовно, найбільш прогресивний підхід на сьогоднішній день. Алгоритм орієнтований на повнокольорові зображення і зображення в градаціях сірого кольору.

Його особливістю є потреба в колосальних обчислювальних потужностях при архівації. При цьому розпакування вимагає менше обчислень, чим у JPEG. Причому, якщо в попередніх алгоритмів коефіцієнт симетричності (відношення часу архівації до часу розархівації) не перевищував 3, то у фрактальному алгоритмі він коливається в межах 1 000–10 000. Як наслідок – основні роботи зараз ведуться по розпаралелюваню і прискоренню його роботи. Фрактальне стиснення реалізоване у форматі FIF.

Інтерполяційні кодуючі системи основані на чисельних методах апроксимації, через які послідовність або двохмірний масив відліків яскравості наближено представляються через неперервні функції. Процедура інтерполяції може бути застосована на етапі перетворення зображення в кодований сигнал (інтерполя-

ція на передаючі стороні) або вона може бути частиною процесу відновлення зображення по кодованому сигналу (інтерполяція на приймальні стороні).

В кодуючих системах з інтерполяцією на передаючі стороні значення яскравості апроксимуються неперервними функціями з раніше встановленою точністю. Інтерполяція може проводитись вздовж стрічки розгортки або охвачувати деяку частину площини зображення (Кодування графічних даних, б.р.).

Інтерполятор нульового порядку працює наступним чином: для всіх елементів зображення встановлений однаковий інтервал допустимих спотворень, в межах якого будується набір відрізків горизонтальних прямих максимальної довжини без додаткових обмежень на розміщення їх початкових та кінцевих точок. Всілякий елемент зображення перекривається яким-небудь із цих відрізків. По каналу зв'язку передаються вертикальна координата кожного відрізка та його довжина, виражена числом елементів. При відновлені зображення на приймальні стороні всі елементи в межах відрізку набувають рівень, відповідний його вертикальні координаті. Цей варіант інтерполяції надає більшу свободу в виборі можливих комбінацій відрізків горизонтальних прямих та в зв'язку з цим дозволяє отримати найбільш ефективне представлення вихідних даних за допомогою мінімальної кількості відрізків. Однак обсяг обчислювальних операцій, необхідних для побудови такого оптимального наближення, часто виявляється дуже великим. Спрощений варіант інтерполяції нульового порядку представляє собою кодування довжин серій з вказуванням яскравості першого елементу серії.

Дія різних інтерполяторів першого порядку полягає в наступному: всякий елемент зображення перекривається будь-яким із прямолінійних відрізків, розміщення яких в межах допустимого інтервалу похибок не пов'язано з додатковими обмеженнями на розміщення початкових та кінцевих точок. Обчислювальна процедура апроксимації може бути в деякій мірі спрощена з'єднанням початку чергового відрізка з кінцем попереднього відрізка. Подальше спрощення полягає в тому, щоб в якості початкових та кінцевих точок використовувати значення яскравості елементів, наближення такого типу часто називають веєрною інтерполяцією (Коцюбівська, та ін., 2018).

Поліномінальні функції більш високого порядку, наприклад кубічні сплайни, також можуть бути застосовані для кодування з інтерполяцією, однак підвищення порядку поліномів супроводжується швидким зростанням обсягу обчислень. Можна також ставити задача двомірної інтерполяції нульового та першого порядків, однак відповідні інтерполятори важко реалізувати на практиці.

Висновки. Перелік приведених алгоритмів далеко не повний, але, дає уявлення про основні тенденції розвитку алгоритмів архівації статичних растрових зображень.

Закінчуючи огляд алгоритмів архівації, мені хотілося б нагадати, що ця область розвивається дуже швидко. Щорічно з'являються нові алгоритми і десятки модифікацій відомих.

Будуть проводитися пошуки і дослідження нових методів кодування, що можуть забезпечити високі коефіцієнти стиснення і високу якість відновленого зображення, незважаючи на їхню складність. З іншого боку будуть проводитися

пошуки більш простих методів, будуть досліджуватися комбінації різних методів з урахуванням технічних можливостей.

СПИСОК ПОСИЛАНЬ

Білинський, Й.Й., Огородник, К.В. та Юкиш, М.Й., 2011. *Електронні системи*. [online]. Вінниця: ВНТУ. Доступно: <https://posibnyky.vntu.edu.ua/pdf/000807.pdf> [Дата звернення 16 квітня 2019].

Кодування графічних даних. [online] Доступно: <https://stud.com.ua/54383/informatika/koduvannya_grafichnih_danih> [Дата звернення 16 квітня 2019].

Коцюбівська, К.І., Чайковська, О.А., Толмач М.С. та Хрущ, С.С., 2018. Стиснення зображень методами кубічних сплайн-функцій.*Технологічний аудит та резерви виробництва*, 3, с. 136-154.

Лутчин, М.М. та Лутчин, Т.М., 2011. *Графічне кодування зображень.* [online] Доступно: http://ena.lp.edu.ua:8080/Bitstream/Ntb/12233/1/13_ГрафічнеКодуванняЗображень%20.pdf> [Дата звернення 16 квітня 2019].

REFERENCES

Bilynskyi, Y.Y., Ohorodnyk, K.V. and Yukysh, M.Y., 2011. *Elektronni systemy* [Electronic systems]. [online]. Vinnytsia: VNTU. Available at: <https://posibnyky.vntu.edu.ua/pdf/000807.pdf> [Accessed 16 April 2019].

Koduvannia hrafichnykh danykh [Graphic data encoding]. [online] Available at: <https://stud.com.ua/54383/informatika/koduvannya_grafichnih_danih> [Accessed 16 April 2019].

Kotsiubivska, K.I., Chaikovska, O.A., Tolmach M.S. ta Khrushch, S.S., 2018. Stysnennia zobrazhen metodamy kubichnykh splain-funktsii [Image compression by cubic spline function methods]. *Tekhnolohichnyi audyt ta rezervy vyrobnytstva*, 3, pp. 136-154.

Lutchyn, M.M. and Lutchyn, T.M., 2011. *Hrafichne koduvannia zobrazhen* [Graphic encoding of images]. [online] Available at: <http://ena.lp.edu.ua:8080/Bitstream/Ntb/12233/1/13_HrafichneKoduvanniaZobrazhen%20.pdf> [Accessed 16 April 2019].

UDC 004.932:51-7

Kotsiubivska Kateryna,
Ph.D. in Technical Sciences, Associate Professor,
Kyiv National University of Culture and Arts,
Kyiv, Ukraine
katysivak@gmail.com
http://orcid.org/0000-0001-6911-2770

Tymoshenko Viktoria,
Master Student of the Computer Science Department,
Kyiv National University of Culture and Arts,
Kyiv, Ukraine
zbyrko.victoria@gmail.com
https://orcid.org/0000-0003-4622-3114

MATHEMATICAL METHODS OF IMAGE PROCESSING

The purpose of the study is to study the specificity of image encoding by spline interpolation, and the equation of the indicated method with other mathematical methods of encoding and image processing.

Research methods. The mathematical and algorithmic models and methods of solving the problem of smoothing on the basis of spline approximation, as well as the possibility of using an appropriate mathematical apparatus for encoding and image processing.

The novelty of the research is the isolation of the compression algorithm of images based on the methods of spline approximation. This approach to image processing can not only reduce the size of image files, but also choose the desired quality of recovery, depending on the further use of the image.

Conclusions The work compares existing image coding methods and points out the benefits of using spline interpolation when encoding and decoding images.

Key words: spline; spline approximation; spline-interpolation; smoothing; coding; decoding; image processing.

УДК 004.932:51-7

Коцюбивская Екатерина,
кандидат технических наук, доцент,
Киевский национальный университет культуры и искусств,
Киев, Украина
katysivak@gmail.com
http://orcid.org/0000-0001-6911-2770

Тимошенко Виктория,
магистрант кафедры компьютерных наук,
Киевский национальный университет культуры и искусств,
Киев, Украина
zbyrko.victoria@gmail.com
https://orcid.org/0000-0003-4622-3114

МАТЕМАТИЧЕСКИЕ МЕТОДЫ ОБРАБОТКИ ИЗОБРАЖЕНИЙ

Цель исследования изучение специфики кодирования изображений методом сплайновой интерполяции, и порвняння указанного метода с другими математическими методами кодирования и обработки изображений.

Методы исследования. Математические и алгоритмические модели и методы решения задачи сглаживания на основе сплайн-аппроксимации, а также возможность применения соответствующего математического аппарата к кодированию и обработки изображений.

Новизной исследования является выделение алгоритма сжатия изображений на основе методов сплайновой аппроксимации. Такой подход к обработке изображений позволяет не только уменьшить размеры файлов изображений, но и выбирать необходимое качество восстановления, в зависимости от дальнейшего использования изображения.

Выводы. В работе проведено сравнение существующих методов кодирования изображений, и указано на преимущества использования сплайновой интерполяции при кодировании и декодировании изображений.

Ключевые слова: сплайн; сплайн-аппроксимация; сплайн-интерполяция; сглаживание; кодирование; декодирование; обработка изображений.

11.05.2019