DOI: https://doi.org/10.23939/csn2024.02.014
УДК 004.382

ОГЛЯД МОЖЛИВОСТЕЙ АЛГОРИТМУ JPEG-LS ДЛЯ ЙОГО ВИКОРИСТАННЯ ІЗ СКАНЕРАМИ ЗЕМНОЇ ПОВЕРХНІ

Т.Л. Грицько, Д. Ленський, В.С. Глухов

Національний університет «Львівська політехніка»
кафедра ЕОМ
E-mail: taras.l.hrytsko@lpnu.ua, daniel.lenskyi.mkisp.2023@lpnu.ua, valerii.s.hlukhov.lpnu.ua

© Грицько Т.Л., Ленський Д., Глухов В.С. 2024

В статті досліджуються можливості реалізації алгоритму стиснення зображень JPEG-LS на програмованих логічних інтегральних схемах (ПЛІС) для обробки монохромних відеопотоків від сканерів земної поверхні. Проведено порівняння програмних реалізацій алгоритмів, їх рівень стиснення та час виконання. Розглянуто способи покращення роботи ПЛІС, використовуючи паралельну обробку даних та оптимізовані структури даних для прискорення процесів стиснення та відновлення. Результати тестування програмної реалізації алгоритму демонструють середню швидкість обробки 179,2Мбіт/с при стисненні та 169,6Мбіт/с при відновленні зображень. Можливо досягнути рівня стиснення від 1,2 до 7,4 залежно від складності зображення.

Ключові слова: JPEG-LS, обробка відеопотоків, обробка зображень, ПЛІС, програмовані логічні інтегральні схеми, стиснення відео, стиснення зображень.

1. Вступ

В сучасному динамічному світі запит на отримання інформації, та зокрема зображень і відео, постійно зростає. Збільшення якості та розміру зображень, а також їх трансляція з різних систем збору даних, потребує збільшення швидкості передачі даної інформації. Досягти цього можна за рахунок збільшення ширини каналу зв'язку, але оскільки вона є обмеженою фізично, іншим способом оптимізації швидкості передачі даних є стискання. Однак, стискання також має другу сторону медалі – а саме затрату процесорного часу та пам'яті на стискання і відновлення даних. Дана стаття розглядає можливі алгоритми стиснення зображень, можливість їх реалізації на ПЛІС та результати оптимізації роботи алгоритму за рахунок використання засобів ПЛІС для обробки відеопотоків.

У статті представлено огляд алгоритму JPEG-LS [1] для стиснення монохромних відеопотоків з акцентом на реалізацію алгоритму на програмованих логічних схемах (ПЛІС). Реалізація алгоритму на ПЛІС [2] дозволяє прискорити обробку зображень за рахунок апаратної паралельності, конвеєрних технік та оптимізації використання ресурсів.

Реалізація на ПЛІС від компанії CAST забезпечує швидкість стиснення до 4Гбіт/с при частоті 500MHz на 28нм технології. Висока частота роботи даного пристрою вимагає більшої затрати ресурсів, тому також проаналізовано структуру, яка забезпечує швидкість обробки до 100 мегапікселів на секунду при частоті роботи ПЛІС 100 МГц та можливості збільшення її продуктивності за рахунок використання модульного підходу в реалізації алгоритму (більше деталей цієї реалізації у Розділі 4).

Швидкість відеопотоків від сканерів земної поверхні можуть досягати 4Гпікселів/с, відеопотоки монохромні, мають до 16 біт/піксель. Для досягнення заданої швидкості необхідно виконувати стискання фрагментів вхідних даних паралельно.

2. Огляд літературних джерел

Алгоритми стиснення зображень та відеопотоків класифікуються за трьома основними критеріями: наявністю втрат, принципом роботи та типом вхідних даних (Рис. 1). За наявністю втрат їх можна поділити на дві великі категорії: алгоритми стиснення без втрат та алгоритми стиснення з втратами [3].

За принципом роботи алгоритми стиснення поділяються на декілька типів. Алгоритми перетворення (transform coding) використовують математичні перетворення для представлення даних у новій формі, що зменшує обсяг інформації. До прикладу, це алгоритми на основі дискретного косинусного перетворення (DCT). Кодування з використанням словника (dictionary coding), наприклад LZW, замінює повторювані фрагменти даних коротшими кодами, таким чином зменшуючи обсяг файлу. Предиктивне кодування (predictive coding) передбачає використання попередніх значень для передбачення наступних, що дозволяє зменшити кількість інформації для зберігання. Ентропійне кодування передбачає присвоєння коротших кодів частіше вживаним символам, що також ефективно зменшує обсяг даних. Одним із прикладів такого підходу є кодування Хаффмана.

Тип вхідних даних визначає поділ методів стиснення для відео на дві категорії: стиснення кадрів (Intraframe compression або spatial compression) і стиснення послідовностей кадрів (Interframe compression або temporal compression). У випадку стиснення кадрів кожен кадр відео стискається як окреме статичне зображення. Такі алгоритми, як JPEG, PNG або HEVC/Intra, застосовуються саме для цього підходу. Натомість стиснення послідовностей кадрів враховує зв'язок між послідовними кадрами відео, що дозволяє значно зменшити обсяг даних. Прикладами таких алгоритмів є MPEG-1, H.264, HEVC та AV1.

Коли йдеться про стиснення без втрат, такі як RLE, LZW, та JPEG-LS, мається на увазі можливість повного відновлення вихідного зображення або відео після стиснення, що є критично важливим у ситуаціях, де потрібно зберегти точність та якість, наприклад, для архівування або професійної роботи з графікою. Для алгоритмів стиснення з втратами , як-от JPEG, JPEG2000, такого точного відновлення не існує, що робить їх більш ефективними для зберігання та передачі великих обсягів даних, але після декомпресії відновлюється зображення або відео, яке не обов'язково буде повністю ідентичним оригіналу.

Стиснення даних можна розділити на декілька категорій, серед яких знаходяться методи стиснення для зображень і методи стиснення для відео. Стиснення зображень зазвичай застосовується для статичних даних, таких як фотографії. Стиснення відео враховує динамічний характер відеопотоків, де кадри змінюються в часі, що дозволяє використовувати додаткові алгоритми для зменшення розміру файлів. Серед методів стиснення існують дві групи: методи без втрат інформації та методи з втратами інформації. Методи стиснення без втрат інформації дозволяють повністю зберегти оригінальну якість відео, але забезпечують відносно невелике скорочення розміру файлу. Стиснення з втратами значно зменшує розмір відеофайлів, що особливо важливо для потокового мовлення та зберігання великої кількості контенту, однак супроводжується деякою втратою якості, яка в багатьох випадках є прийнятною для кінцевих користувачів.

До алгоритмів стиснення без втрат відносяться:
алгоритм RLE [4],
алгоритм LZW [5],
алгоритм Хаффмана [6],
алгоритми на основі ДКП (DCT, дискретного косинусного перетворення) [3];
JBIG [7];
JPEG-LS [1],
JPEG-XL [8],
NBLI [9]
WEBP [10],
PNG [10].
Алгоритми стиснення з втратами включають:
алгоритми JPEG, JPEG2000 [10],

фрактальний алгоритм [11],

вейвлетний (рекурсивний або хвильовий) алгоритм (DWT) [3].

До основних алгоритмів, у яких використано стиснення кадрів (Intraframe Compression), належать:

JPEG, JPEG 2000, JPEG-LS;

PNG;

HEVC/Intra (H.265 Intra) [12].

До основних алгоритмів, де використано стиснення послідовностей кадрів (Interframe Compression), належать:

MPEG-1, MPEG-2, MPEG-4 [13];

H.264/AVC [12];

HEVC (H.265) [12];

AV1 [14].

У джерелах [15] і [23] наведено класичну і сучасну бази зображень для тестування алгоритмів стиснення. Бази містять зображення різного розміру та складності.

У [16] міститься програма для тестування алгоритмів Image Compression Benchmark, яку авторами було використано для порівняння різних алгоритмів стиснення зображень.

Джерело [17] містить посилання на сайт розробників стандарту JPEG - об'єднаної групи експертів із фотографії (Joint Photographic Experts Group).

У [18] і [19] наводяться приклади ядер енкодера та декодера від компанії CAST.

У статті [20] наводиться приклад JPEG-LS енкодера, реалізованого на ПЛІС, для стиснення даних з супутників.

У [21] і [22] наведено приклади реалізації алгоритму JPEG-LS на ПЛІС із застосуванням конвеєрних підходів для покращення роботи алгоритму.

У джерелі [24] реалізація енкодера для алгоритму JPEG-LS на ПЛІС, написана мовою Verilog.

3. Постановка задачі

У проаналізованих джерелах не розглядалася задача використання алгоритму JPEG-LS у перспективних сканерах земної поверхні, для яких характерне формування відеопотоків, які не поділяються на кадри, тобто, представляють із себе один неперервний кадр.

4. Мета статті

Метою статті є порівняння характеристик алгоритму JPEG-LS з характеристиками інших алгоритмів стиснення зображень, аналіз алгоритму JPEG-LS для визначення можливостей його використання у сканерах земної поверхні, варіантів його реалізації на ПЛІС та способів оптимізації. Основні завдання включають аналіз особливостей алгоритму JPEG-LS та його реалізацій на ПЛІС, а також аналіз способів підвищення швидкодії алгоритму.

5. Порівняння методів стиснення

Алгоритми стиснення зображень та відеопотоків класифікуються за трьома основними критеріями: наявністю втрат, принципом роботи та типом вхідних даних (Рис. 1).

У таблиці 1 наведено результати порівняння методів стиснення даних за різними ознаками. Вона охоплює основні алгоритми стиснення, тип стиснення (з втратами або без втрат), сфери використання, підтримку кольорів, переваги та недоліки кожного методу. В таблиці також зазначено приклади форматів файлів, які використовуються для зберігання даних, стиснених за допомогою цих методів. Окремо виділено метод JPEG-LS, оскільки він має унікальні характеристики і рідко використовується у загальному випадку.

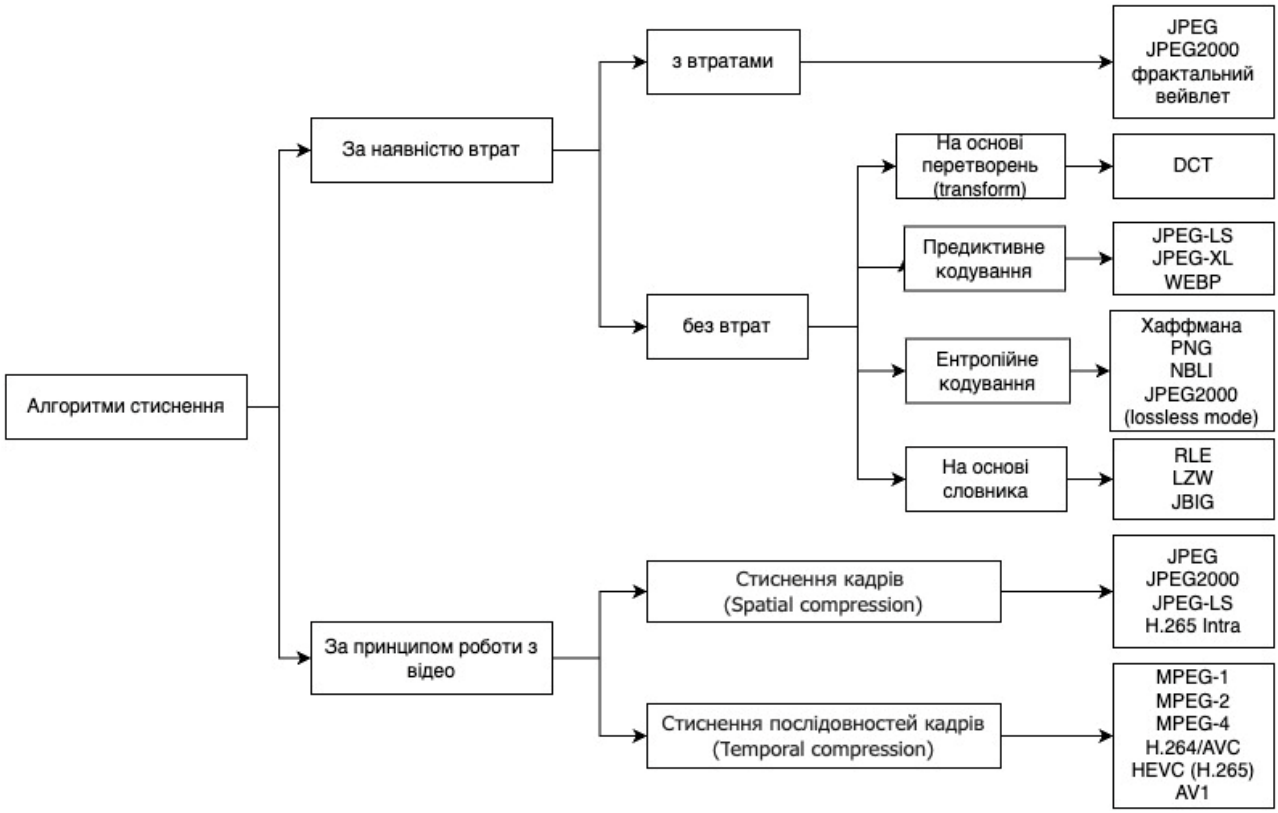

Рис. 1 Класифікація алгоритмів стиснення зображень

Таблиця 1

Результати порівняння методів стиснення

Метод	Тип (з втратами / без втрат)	Застосування	Переваги	Недоліки
JPEG	З втратами	Обробка зображень (фото, графіка)	Високий ступінь стиснення, популярність	Втрата якості, особливо при високому стисненні
JPEG-LS	Без втрат	Медичні зображення, точні наукові знімки	Стиснення без втрат, точність	Менший коефіцієнт стиснення порівняно з JPEG
H.264 (AVC)	З втратами	Відео кодеки для трансляції та зберігання	Високий коефіцієнт стиснення, популярність	Складний декодер, втрата якості при надмірному стисненні
HEVC (H.265)	З втратами	Сучасні відео кодеки для 4K і 8K відео	Висока ефективність стиснення, новий стандарт	Складність обробки, потребує потужного обладнання
Huffman Coding	Без втрат	Текстові файли, архіви, низькорівневе стиснення	Простота реалізації, ефективність для текстів	Менш ефективний для складних даних, чутливий до структури даних

RLE (Run-Length Encoding)	Без втрат	Стиснення зображень з великими однорідними областями	Простота, швидкість	Неефективний для складних зображень
LZW (Lempel-Ziv-Welch)	Без втрат	GIF-анімації, файли TIFF, архіви	Підтримка багатьох форматів, ефективність	Не підходить для мультимедійних файлів

6. Вибір алгоритму

Облітаючи земну орбіту, супутник постійно сканує поверхню Землі та передає ці дані безперервним потоком. Тому, при виборі алгоритму нас цікавили алгоритми стиснення без втрат, з роботою з кадром (безперервним) та з можливістю реалізації на ПЛІС (тобто з низькою складністю алгоритму та обмеженим використанням таких ресурсів, як пам’ять чи час роботи).

Для вибору алгоритму стиснення зображень було проведено порівняльний аналіз програмних реалізацій таких алгоритмів, як JBIG, NBLI, PNG, WEBP, JPEG-XL, JPEG2000 (lossless mode) та JPEG-LS. Тестування проведено на ПК Intel Core I7 12700H, 16GB DDR4 in 3200MHz, ОС Windows 10.

Ключовими показниками алгоритмів стиснення є складність алгоритму, рівень стиснення та час виконання. На Рис. 2 представлено порівняння часу виконання програмної реалізації алгоритму і розміру після стиснення для цих алгоритмів. Зображення для тестування алгоритмів у кількості 64 одиниці та сумарним розміром 750 мегабайт взято з бази стандартних зображень для тестування алгоритмів стиснення [15]. Результати виконання алгоритмів було взято з програми Lossless Image Compression Benchmark [16] та доповнено авторами, а саме використанням тестів з програми для алгоритму JBIG, який використовувався у попередніх порівняннях, але не входив у стандартний набір програми. По осі X відображено час стиснення та відновлення в секундах, по осі У – розмір стиснутого зображення в мегабайтах. Відповідно, чим ближче обидва показники до нуля, тим менший розмір стиснутого зображення і відповідно кращий рівень стиснення, а також менший час виконання стиснення та відновлення).

На основі цього графіку можна виділити такі алгоритми:

NBLI – рівень стиснення в межах 2,73:1 – 2,77:1, однак залежно від даних час виконання від 38,8с до 493,7с;

JPEG-XL – рівень стиснення в межах 2,38:1 – 2,78:1, час виконання від 13,7с до 5496,3с;

та JPEG-LS – рівень стиснення в межах 2,64:1 – 2,71:1, час виконання від 13,7с до 18,5с.

При передачі відеопотоку з супутника, для уникнення затримок передачі зображення, особливу увагу потрібно звернути на передбачуваність часу виконання алгоритму. Саме тому вибір зупинився на алгоритмі JPEG-LS.

7. Характеристики алгоритму JPEG-LS

JPEG-LS - це стандарт стиснення зображень без втрат, який забезпечує ефективне і швидке кодування та декодування. Він використовує прогнозуючий алгоритм LOCO-I, який заснований на локальній кореляції пікселів, та кодування залишків для стискання даних. Це дозволяє JPEG-LS досягати високої швидкості роботи, зберігаючи при цьому точність і якість зображення. Основними сферами застосування JPEG-LS є медична візуалізація, факсимільний зв'язок і архівування зображень, де важливо зберігати всі деталі без спотворень.

JPEG-LS не набув такої популярності, як інші формати, наприклад, JPEG або PNG, через обмежене застосування у загальному користуванні та невелику кількість підтримуваних пристроїв. Проте його переваги в точності і швидкості стиснення роблять його важливим інструментом у сферах, де необхідне збереження вихідної якості без втрат.

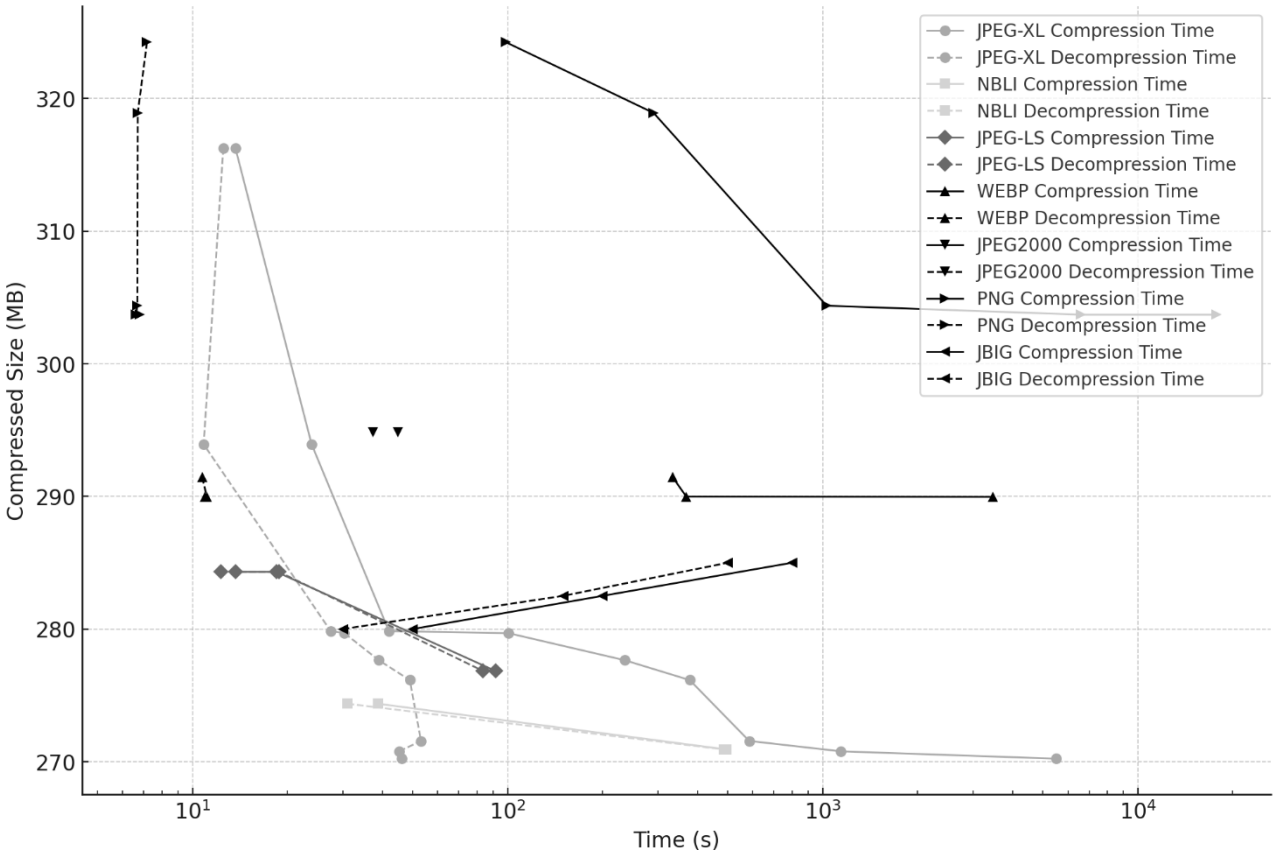

Рис. 2 Порівняння часу роботи алгоритмів і розміру зображень після стиснення.

Алгоритм JPEG-LS використовує метод прогнозування для обробки пікселів, ґрунтуючись на значеннях сусідніх пікселів. Прогнозування базується на припущенні, що піксель має схожі значення з навколишніми пікселями, і потім кодується різниця між передбаченим і фактичним значенням пікселя. Розроблений для забезпечення низької обчислювальної складності при збереженні рівня стиснення. Алгоритм розроблено об'єднаною групою експертів із фотографії (Joint Photographic Experts Group, JPEG) [17]. Основні етапи алгоритму JPEG-LS представлені на Рис. 3.

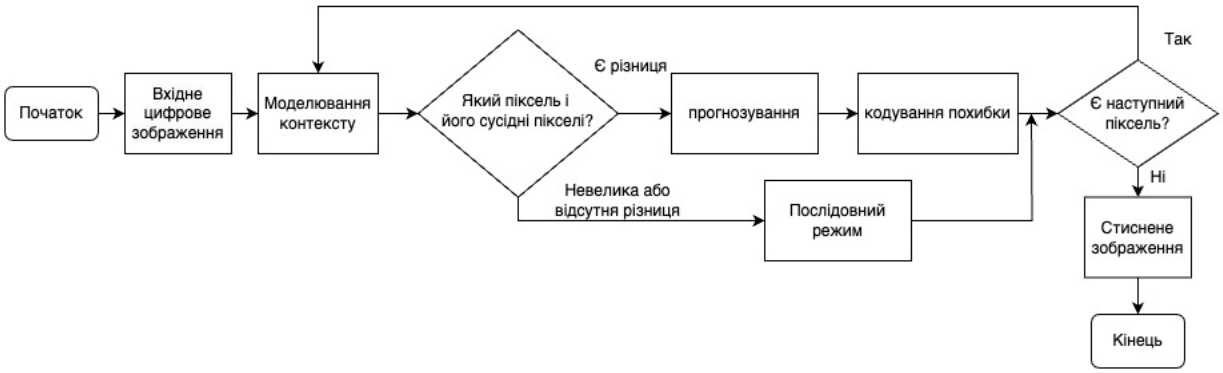

Рис. 3 Схема алгоритму стиснення зображень JPEG-LS

Розглянемо основні етапи алгоритму JPEG-LS:

Моделювання контексту (даних про сусідні пікселі): Контекстне моделювання полягає в аналізі локальних характеристик зображення навколо поточного пікселя. Для зменшення використання пам'яті, цей процес може бути оптимізованим шляхом пропуску непотрібних обчислень при виявленні однакових послідовностей пікселів.

Прогнозування: JPEG-LS використовує предиктивний (прогнозований) підхід для оцінки значення поточного пікселя на основі сусідніх пікселів. Кожен піксель порівнюється з його

найближчими сусідами для прогнозування його значення. На Рис. 4 зображено пікселі a, b, c та d, які використовуються для вибору режиму роботи алгоритму. Залежно від контексту кодер вибирає режим: послідовний (run mode) або регулярний (regular mode). Послідовний режим вибирають, якщо наступні пікселі, швидше за все, збігатимуться, регулярний – якщо ні. У разі використання послідовного режиму роботи кодер починає перегляд поточного рядка з пікселя x і знаходить найбільшу довжину серії пікселів, що збігаються з контекстним пікселем a. Отже, в межах поточного рядка отримують послідовність однакових пікселів, які збігаються за значенням з відомим пікселем a. Після цього кодується довжина послідовності. У разі використання регулярного режиму для обчислення прогнозованого значення пікселя x (Px) використовують значення пікселів a, b і c. Також обчислюється так звана помилка прогнозу (Errval). Її значення дорівнює різниці значення x і Px. Errval коригується, а потім кодується за допомогою кодів Голомба [3]. Код Голомба залежить від a, b, c, d і Errval цих самих пікселів. Основні прогнози, що використовуються для вибору режиму роботи кодера:

Лінійні комбінації сусідніх пікселів (ліворуч, зверху, зліва-зверху).

Виявлення країв для коригування прогнозованих значень на основі геометрії зображення.

Кодування похибки: Після отримання прогнозованого значення обчислюється похибка (залишок) між реальним та прогнозованим значенням пікселя. Ці залишки кодуються за допомогою кодів Голомба-Райса [3], які є дієвими для даних з невеликою варіативністю, характерною для зображень. Це особливо важливо для зниження енергоспоживання та підвищення швидкості роботи системи.

Послідовний режим кодування (Run-Length): JPEG-LS має спеціальний режим для роботи з серіями пікселів однакових значень, що часто зустрічаються в реальних зображеннях (наприклад, фони або однорідні ділянки). Цей режим дозволяє кодувати такі серії, зменшуючи об'єм даних, що передаються. У ПЛІС він може бути реалізований за допомогою буферів для збереження серій пікселів і спеціальних схем для обробки серій у реальному часі.

c	b	d
a	x	

Рис. 4 Контекстні пікселі для пікселя x.

8. Огляд поточних реалізацій алгоритму JPEG-LS на ПЛІС

Одним з аналогів для проведення проектування є ядра JPEG-LS (Encoder і Decoder) від компанії CAST [18] [19]. Ці пристрої пропонують реалізацію стандарту JPEG-LS для стиснення зображень без втрат і з мінімальними втратами.

Ядра підтримують розмір вхідних зображень до 64Kx64K пікселів, до 4 кольорів та до 16 бітів на колір. Мають дві реалізації – оптимізовану для обробки цілого зображення з обробкою одного символу за такт і швидкістю до 4Гбіт/с при частоті 500MHz, а також для потоку даних – з заданою кількістю символів за такт за рахунок паралельного використання кількох ядер.

Для реалізації JPEG-LS на ПЛІС також проаналізовано структуру, яка забезпечує швидкість обробки монохромних потоків даних (bitstream) до 100 мегапікселів на секунду при частоті роботи ПЛІС 100 МГц (Рис. 5) [20]. Основні компоненти цієї структури включають:

Модуль предикції: Реалізує лінійні прогнози з використанням лише операцій додавання та віднімання, що дозволяє досягти затримки не більше 2 тактів на піксель.

Модуль обчислення залишків: Виконує віднімання передбаченого значення від реального. З чотирьох базових арифметичних операцій (+, -, *, /) використовується лише віднімання, що мінімізує апаратні ресурси.

Контекстний модуль: Визначає контексти для моделювання залишків, використовуючи оптимізовану схему індексації, що зменшує кількість звернень до пам'яті на 44,1% [21].

Модуль кодування: Реалізує алгоритм Голомба-Райса [3] для кодування залишків.

9. Методи оптимізації

При реалізації JPEG-LS на ПЛІС варто звернути увагу на такі методи оптимізації:

Використання модульного дизайну: модульний дизайн дозволяє розбити алгоритм на менші, взаємопов'язані блоки (модулі), кожен із яких виконує специфічні завдання. Такий підхід збільшує масштабованість всього процесу, гнучкість (можливість масштабувати модулі, які дають затримку для процесу), спрощує відлагодження (кожен модуль тестується окремо).

Розпаралелювання: опрацювання до 4 пікселів одночасно підвищує продуктивність на 280-320% [21].

Використання конвеєрних підходів: розподіл роботи алгоритму на незалежні етапи, особливо з використанням модульного дизайну. Середній коефіцієнт стиснення може досягати 2,61:1 для різних вхідних потоків зображень. Максимальна робоча частота може досягати 103 МГц [2]. Приклад такої реалізації наведено у Рис. 5.

Завдяки застосуванню техніки переадресації даних (forwarding) для роботи конвеєра без затримок (non-stalling pipeline), частота роботи була збільшена з 100 до 155,2 МГц, а використання пам’яті зменшено на 24%. Архітектура була описана мовою Verilog HDL та реалізована на ПЛІС ALTERA Stratix II [22].

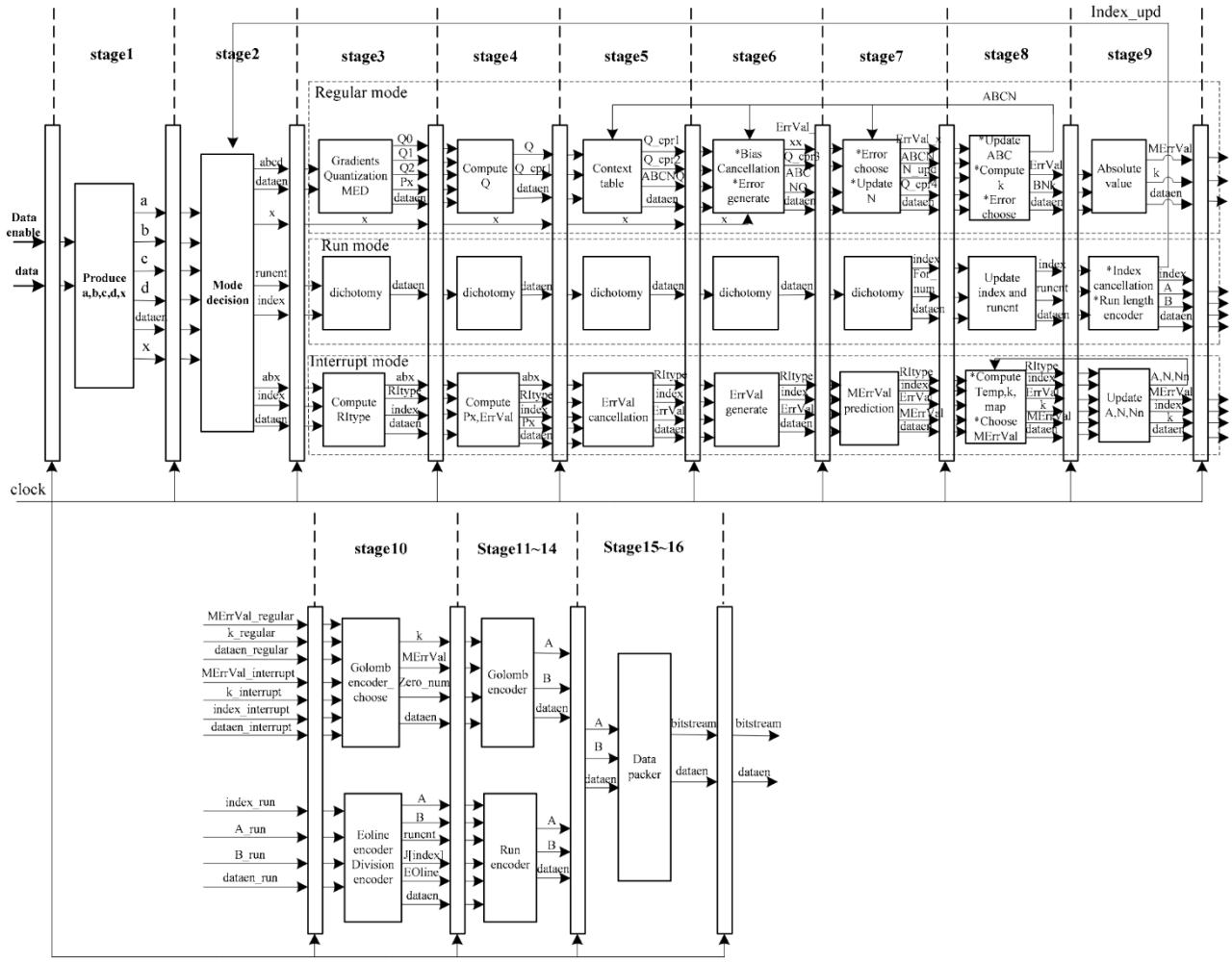

Рис. 5 Схема конвеєрної реалізації алгоритму JPEG-LS

10. Результати дослідження

Для оцінки реалізації алгоритму JPEG-LS на ПЛІС авторами було проведено серію експериментів з використанням набору тестових зображень різного розміру та складності. Зображення було взято як з традиційного набору для тестування алгоритмів стиснення SIPI [15], так і з більш сучасного набору [23], що включає зображення роздільної здатності до 7215x5412 пікселів. Для проведення тестів зображення були перетворені у монохромні з використанням 8 бітів на піксель, та перетворено в формат pgm. Для запуску тестів було використано ПЛІС Artix-7 xc7a35tcsg324-2 при частоті 75 MHz в симуляторі Xilinx ISE, використовуючи реалізацію алгоритму FPGA-JPEG-LS-Encoder [24]. Порівняння швидкості стиснення зображень на основі проведених тестів наведено в Рис. 6.

Аналіз результатів показує, що середня швидкість стиснення варіюється від 112,72Мбіт/с до 362,8Мбіт/с, залежно від характеристик зображення. Зображення з однорідним вмістом, такі як "artificial", стискаються при швидкості 585,12Мбіт/с. Обробка природніх зображень, таких як "4.1.pgm" розміром 256x256 пікселів, показала швидкість 102,4Мбіт/с, що може бути пов'язано з накладними витратами на ініціалізацію алгоритму для малих зображень.

Коефіцієнт стиснення коливається від 1,214 в найгіршому варіанті до 7,429 в найкращому, із середнім значенням у 3,4 та медіанним значенням 3,7. Середня швидкість обробки досягає 1,434Гбіт/с при стисненні та 1,357Гбіт/с при відновленні зображень.

Тести виконано на ПК Intel Pentium 2020M @ 2.4GHz, 8GB DDR3 SODIMM з частотою 1600MHz, ОС Windows 10.

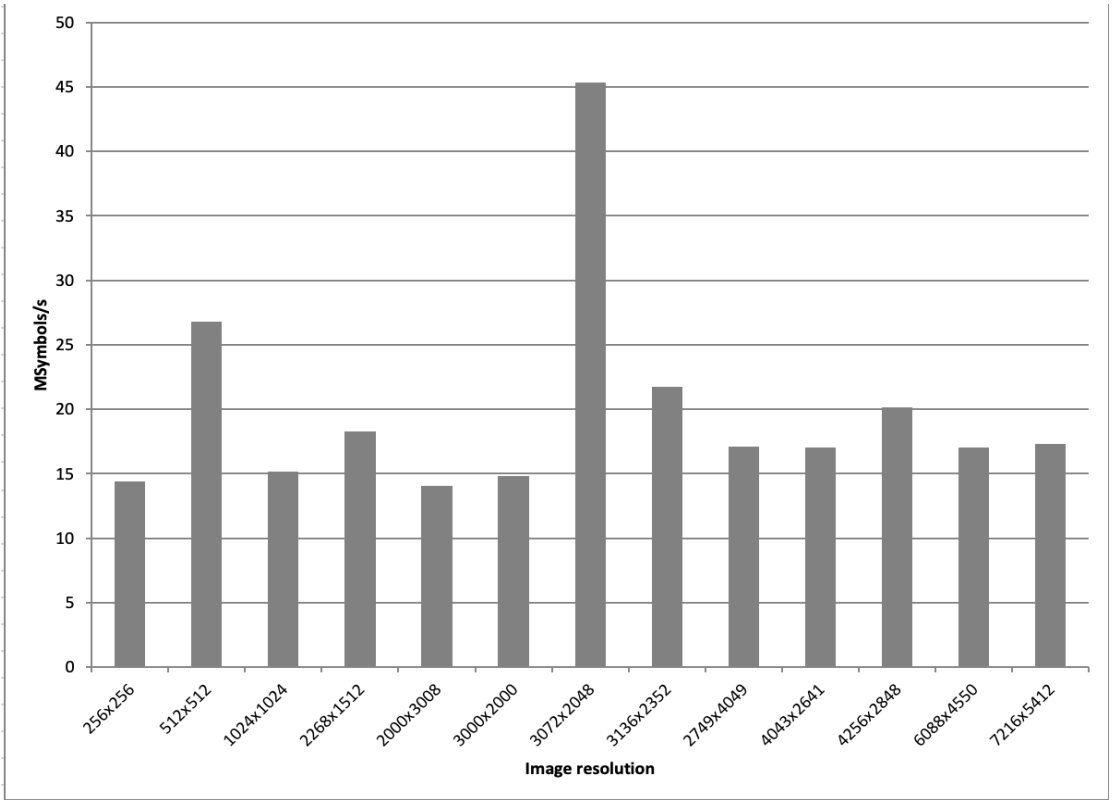

Рис. 6 Середні значення швидкості стиснення зображень різного розміру алгоритмом JPEG-LS апаратної реалізації FPGA-JPEG-LS-Encoder

11. Висновки

Авторами статті проведено аналіз наявних алгоритмів стиснення зображень та особливості їх застосування при стисненні монохромних відеопотоків з супутника.

Було проведено тестування програмної реалізації енкодера та декодера. Відомі апаратні реалізації Алгоритму JPEG-LS показали такі характеристики роботи: рівень стиснення в межах 2,64:1

– 2,71:1, час виконання від 13,7с до 18,5с для 64 зображення різного розміру із загальним розміром файлів 750МБ для монохромних 8-бітових зображень у форматі pgm.

Проведено інформаційний пошук, який показав, яких результатів можна досягнути при використанні алгоритму JPEG-LS:

Розглянуто способи покращення роботи ПЛІС, а саме використання паралельної обробки, що дозволяє обробляти до 4 пікселів одночасно, підвищуючи загальну продуктивність на 280-320%. Також проаналізовано застосування конвеєрних підходів: при розподілі роботи алгоритму на незалежні етапи, середній коефіцієнт стиснення може досягати 2,61:1. Максимальна робоча частота може досягати 103 МГц.

Швидкість обробки: апаратно можливо досягнути середньої швидкості стиснення до 1,434Гбіт/с, для досягнення необхідної швидкості 4Гбіт/с необхідно використати розпаралелювання на рівні ПЛІС та виконувати обробку відеопотоку через 2-3 паралельні процеси, або використати ПЛІС з підтримкою вищої тактової частоти.

12. Подяки

Ця стаття використовує матеріали та результати, отримані авторами під час науково-дослідної роботи “Інтелектуальні методи та інструменти проєктування модульних автономних кіберфізичних систем”, НДР 0124U002340 від 09.03.2024, яка проводиться на кафедрі ЕОМ Інституту комп'ютерних технологій, автоматики та метрології Національного університету «Львівська політехніка» у 2024-2028 роках.

Список літератури

1. M. J. Weinberger, G. Seroussi, and G. Sapiro, “The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS,” *IEEE Transactions on Image Processing*, vol. 9, no. 8, pp. 1309–1324, Aug. 2000, doi: 10.1109/83.855427.

2. X. Dong and P. Li, “Implementation of A Real-Time Lossless JPEG-LS Compression Algorithm Based on FPGA,” in *2022 14th International Conference on Signal Processing Systems (ICSPS)*, Nov. 2022, pp. 523–528. doi: 10.1109/ICSPS58776.2022.00096.

3. S. M. S. H. M. A. Hossain, “Classification on Image Compression Methods: Review Paper,” *International Journal of Data Science Research*, vol. 1, no. 1, Art. no. 1, Apr. 2018, Accessed: Sep. 25, 2024. [Online]. Available: http://ojs.mediu.edu.my/index.php/IJDSR/article/view/1395

4. C. Dunn, “Smile! You’re on RLE!,” *The Transactor*, vol. 7 (6), pp. 16–18.

5. “LZW Compression Encoding.” Accessed: Oct. 16, 2024. [Online]. Available: https://www.loc.gov/preservation/digital/formats/fdd/fdd000135.shtml

6. T. H. Cormen, Ed., *Introduction to algorithms*, 2nd. ed., 10th pr. Cambridge, Mass.: MIT Press [u.a.], 2007.

7. V. Kyrki, “JBIG image compression standard,” Apr. 1999.

8. J. Alakuijala *et al.*, “Benchmarking JPEG XL image compression,” in *Optics, Photonics and Digital Technologies for Imaging Applications VI*, P. Schelkens and T. Kozacki, Eds., Online Only, France: SPIE, Apr. 2020, p. 32. doi: 10.1117/12.2556264.

9. Dr.W.X, *WangXuan95/NBLI*. (Oct. 07, 2024). C++. Accessed: Oct. 08, 2024. [Online]. Available: https://github.com/WangXuan95/NBLI

10. E. Öztürk and A. Mesut, “Performance Evaluation of JPEG Standards, WebP and PNG in Terms of Compression Ratio and Time for Lossless Encoding,” in *2021 6th International Conference on Computer Science and Engineering (UBMK)*, Sep. 2021, pp. 15–20. doi: 10.1109/UBMK52708.2021.9558922.

11. “Fractal image compression - ProQuest.” Accessed: Oct. 16, 2024. [Online]. Available: https://www.proquest.com/docview/215266230?sourcetype=Scholarly%20Journals

12. “Real-Time H.265/HEVC Intra Encoding with a Configurable Architecture on FPGA Platform.” Accessed: Oct. 16, 2024. [Online]. Available: https://cje.ejournal.org.cn/en/article/doi/10.1049/cje.2019.06.020

13. “Standards – MPEG.” Accessed: Oct. 16, 2024. [Online]. Available: https://www.mpeg.org/standards/

14. “RTP Payload Format For AV1.” Accessed: Oct. 16, 2024. [Online]. Available: https://aomediacodec.github.io/av1-rtp-spec/#71-media-type-definition

15. “SIPI Image Database.” Accessed: Oct. 09, 2024. [Online]. Available: https://sipi.usc.edu/database/

16. Dr.W.X, *WangXuan95/Image-Compression-Benchmark*. (Oct. 16, 2024). Python. Accessed: Oct. 17, 2024. [Online]. Available: https://github.com/WangXuan95/Image-Compression-Benchmark

17. “JPEG - About JPEG.” Accessed: Oct. 16, 2024. [Online]. Available: https://jpeg.org/about.html

18. “JPEG-LS-E | Lossless & Near-Lossless JPEG-LS Encoder IP Core,” CAST. Accessed: Oct. 17, 2024. [Online]. Available: https://www.cast-inc.com/compression/lossless-image-compression/jpeg-ls-e

19. “JPEG-LS-D | Lossless & Near-Lossless JPEG-LS Decoder IP Core,” CAST. Accessed: Oct. 17, 2024. [Online]. Available: https://www.cast-inc.com/compression/lossless-image-compression/jpeg-ls-d

20. W. Wei, J. Lei, and Y. Li, “Onboard optimized hardware implementation of JPEG-LS encoder based on FPGA,” in *Satellite Data Compression, Communications, and Processing VIII*, SPIE, Oct. 2012, pp. 49–58. doi: 10.1117/12.930869.

21. S.-L. Chen, T.-Y. Liu, C.-W. Shen, and M.-C. Tuan, “VLSI Implementation of a Cost-Efficient Near-Lossless CFA Image Compressor for Wireless Capsule Endoscopy,” *IEEE Access*, vol. 4, pp. 10235–10245, 2016, doi: 10.1109/ACCESS.2016.2638475.

22. H. Daryanavard, O. Abbasi, and R. Talebi, “FPGA implementation of JPEG-LS compression algorithm for real time applications,” in *2011 19th Iranian Conference on Electrical Engineering*, May 2011, pp. 1–4. Accessed: Oct. 17, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/5955727

23. N. Asuni and A. Giachetti, “TESTIMAGES: a Large-scale Archive for Testing Visual Devices and Basic Image Processing Algorithms,” 2014, *The Eurographics Association*. doi: 10.2312/STAG.20141242.

24. Dr.W.X, *WangXuan95/FPGA-JPEG-LS-encoder*. (Oct. 17, 2024). Verilog. Accessed: Oct. 17, 2024. [Online]. Available: https://github.com/WangXuan95/FPGA-JPEG-LS-encoder

REVIEW OF THE CAPABILITIES OF THE JPEG-LS ALGORITHM FOR ITS USE WITH EARTH SURFACE SCANNERS

T.L. Hrytsko, D. Lenskiy, V. Hlukhov

Lviv Polytechnic National University, 12, Bandera Str, Lviv, 79013, Ukraine
Department of Electronic Computing Machines
E-mail: taras.l.hrytsko@lpnu.ua, daniel.lenskyi.ki.2019@lpnu.ua, valerii.s.hlukhov.lpnu.ua

© Hrytsko T.L., Lenskiy D., Hlukhov V.S. 2024

The article explores the possibilities of implementing the JPEG-LS image compression algorithm on Field Programmable Gate Arrays (FPGA) for processing monochrome video streams from Earth surface scanners. A comparison of software implementations of the algorithms, their compression ratio, and execution time is conducted. Methods for improving FPGA performance are considered, using parallel data processing and optimized data structures to accelerate compression and decompression processes. Test results of the software implementation of the algorithm show an average processing speed of 179.2 Mbit/s during compression and 169.6 Mbit/s during decompression. A compression ratio from 1.2 to 7.4 can be achieved depending on the complexity of the image.

Key words: FPGA, JPEG-LS, Field-programmable gate arrays, Image compression, Image processing, Video compression, Video stream processing.