МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
«КИЇВСЬКИЙ АВІАЦІЙНИЙ ІНСТИТУТ»

ШЕВЧЕНКО Андрій Костянтинович



УДК 004.932:004.272 (043.3)

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ОБРОБКИ ГРАФІЧНИХ ДАНИХ ДЛЯ SIMD-АРХІТЕКТУР ПРОЦЕСОРІВ

Спеціальність 05.13.06 – «Інформаційні технології»

Автореферат
дисертації на здобуття наукового ступеня
кандидата технічних наук

Київ – 2026

Дисертацією є рукопис.

Робота виконана на кафедрі прикладної математики, факультету комп'ютерних наук та технологій Національного університету «Київський авіаційний інститут» Міністерства освіти і науки України.

Науковий керівник: доктор технічних наук, професор **Приставка Пилип Олександрович**, Національний університет «Київський авіаційний інститут», головний науковий співробітник наукової групи кафедри прикладної математики.

Офіційні опоненти: доктор технічних наук, професор **Кучанський Олександр Юрійович**, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», професор кафедри біомедичної кібернетики;

кандидат технічних наук, доцент **Мирутенко Лариса Вікторівна**, Київський національний університет імені Тараса Шевченка, доцент кафедри кібербезпеки та захисту інформації.

Захист відбудеться « 17 » вересня 2026 р. о 15:00 на засіданні спеціалізованої вченої ради Д 26.062.01 при Національному університеті «Київський авіаційний інститут» за адресою: 03058, м. Київ, пр. Любомира Гузара, 1, корпус 1, ауд. 1.334.

З дисертацією можна ознайомитись у Науково-технічній бібліотеці Національного університету «Київський авіаційний інститут» за адресою: 03058, м. Київ, пр. Любомира Гузара, 1.

Автореферат розісланий « 17 » серпня 2026 р.

Вчений секретар
спеціалізованої вченої ради,
к.т.н., доц. Анна ІЛЬЄНКО

ЗАГАЛЬНА ХАРАКТЕРИСТИКА РОБОТИ

Актуальність. Обробка цифрових зображень (ЦЗ) та відеоконтенту є невід'ємною частиною сучасного інформаційного світу і застосовується в найрізноманітніших сферах: від побутових додатків у смартфонах до критично важливих систем у промисловості, медицині, безпеці та обороні. Операція згортки (ОЗ) є основою багатьох алгоритмів обробки ЦЗ, впливаючи на їхню ефективність та швидкість. З огляду на експоненційне зростання обсягів візуальних даних і вимог до їх аналізу в реальному часі, дослідження, спрямовані на підвищення швидкодії обробки ЦЗ, набувають особливої актуальності. Світові тенденції демонструють активний перехід до використання пристроїв IoT для локальної обробки даних (edge computing). Це зумовлено потребою у зменшенні затримок, підвищенні надійності та забезпеченні конфіденційності даних. Однак такі платформи, зокрема на базі архітектури ARM64, що домінує в мобільних та вбудованих пристроях, зазвичай мають обмежені обчислювальні ресурси. Стандартні підходи до обробки ЦЗ не завжди оптимально використовують специфіку архітектури ARM64 та її SIMD-розширення NEON64, що призводить до низької швидкодії.

Запропонована в дисертації інформаційна технологія (ІТ) підвищення швидкодії обробки цифрових RGB-зображень шляхом оптимізації ОЗ з використанням 16-бітних цілочисельних операцій та SIMD-інструкцій NEON64 для архітектури ARM64 є унікальною, оскільки поєднує ці підходи, що досі не було широко досліджено в контексті ОЗ ЦЗ.

Актуальність цього дослідження для України сьогодні є надзвичайно високою. В умовах сучасних викликів ефективна обробка візуальної інформації на пристроях з процесорами архітектури ARM64 має критичне значення для сфери безпеки та оборони. Це стосується систем відеоспостереження, аналізу супутникових знімків і, особливо, обробки даних з безпілотних літальних апаратів (БПЛА). Запропонована ІТ дозволяє підвищити продуктивність обробки ЦЗ на бортових комп'ютерах БПЛА, які часто базуються на енергоефективних ARM-процесорах. Це сприятиме зменшенню затримок під час аналізу даних у реальному часі, підвищенню автономності БПЛА в завданнях виявлення та розпізнавання об'єктів (у БПЛА/дронах із машинним зором). Така оптимізація є важливою для створення інтелектуальніших та енергоефективніших безпілотних систем для потреб Збройних Сил України.

Зв'язок роботи з науковими програмами, темами та планами. Дисертаційне дослідження проводилось в рамках НДР на базі Національного університету «Київський авіаційний інститут», зокрема: НДР №1062-ДБ16 «Автоматизація розпізнавання та класифікації цільових об'єктів за відеоданими з камери безпілотного повітряного судна» (№ д\р 0116U004638); НДР № 247-ДБ19 «Розроблення та виготовлення програмно-апаратних засобів цільового навантаження для повітряного спостереження та альтернативної навігації літального апарату» (№ д\р 0119U100553); НДР №479-ДБ24 «Розроблення багатокритеріальної завадозахищеної навігаційної системи для безпілотних апаратів» (№д\р 0124U000291).

Мета і задачі дослідження. Метою роботи є розв'язання науково-технічної задачі, що полягає в розробці інформаційної технології обробки цифрових зображень та відео з використанням швидкодіючої операції згортки (ШОЗ) для ЦП архітектури ARM64, яка б дозволила в режимі реального часу проводити фільтрацію та масштабування потокового відео та ЦЗ на одноплатних комп'ютерах з обмеженими обчислювальними ресурсами.

Для досягнення поставленої мети необхідно розв'язати такі **основні задачі:**

1. Провести аналіз існуючих програмно-апаратних та архітектурно-інформаційних засобів для забезпечення, підвищення та реалізації швидкодіючих обчислень, для ОЗ ЦЗ і програмного забезпечення в цілому. Визначити потенційні напрями вдосконалення ПЗ для цілочисельних обчислень.
2. Розробити математичну модель ОЗ та формалізовану нотацію для опису векторизованих обчислень з використанням SIMD NEON64.
3. Розробити метод ШОЗ для обробки 8-бітних RGB ЦЗ з використанням 16-бітних цілочисельних операцій, для забезпечення підвищення швидкодії на центральних процесорах архітектури ARM64.
4. Розробити ІТ підвищення швидкодії обробки RGB ЦЗ, для фільтрації та масштабування потокового відео та ЦЗ на одноплатних комп'ютерах з ЦП ARM64 в режимі реального часу.
5. Провести експериментальне дослідження швидкодії розробленого алгоритму на різних платформах SBC (Raspberry Pi 4, Orange Pi 5 Pro) для різних розмірів ЦЗ (256×144, 426×240, 640×360, 854×480, 1280×720, 1920×1080, 2560×1440, 3840×2160).
6. Проаналізувати отримані результати та визначити показники ефективності запропонованого підходу для різних розмірів ЦЗ та апаратних платформ.
7. Розробити методику експериментальної оцінки та порівняння швидкодії різних програмних реалізацій ОЗ на платформах з архітектурою центрального процесора ARM64 та рекомендації щодо застосування оптимізованого алгоритму ОЗ в системах реального часу та вбудованих пристроях.

Об'єктом дослідження є процеси обробки цифрових RGB-зображень операцією згортки на платформах з архітектурою ARM64.

Предметом дослідження є методи, алгоритми та інформаційна технологія векторизованої ОЗ (ВОЗ) з використанням SIMD-інструкцій ARM64 NEON для підвищення швидкодії обробки ЦЗ.

Методи дослідження.

Для вирішення поставлених завдань використано такі методи:

1. Теоретичні методи: аналіз та узагальнення наукової літератури з питань оптимізації ОЗ та обробки ЦЗ; математичне моделювання процесів обробки ЦЗ із використанням операторів згортки.
2. Емпіричні методи: експериментальне дослідження швидкодії розробленого алгоритму на різних платформах ARM64; порівняльний аналіз швидкодії розробленого алгоритму з існуючими рішеннями.
3. Методи програмної інженерії: розробка та оптимізація програмного коду з використанням мов C/C++ та асемблерних вставок (Inline Assembly); використання інструментів профілювання для аналізу швидкодії коду.
4. Методи комп'ютерного моделювання: симуляція роботи алгоритму на різних типах ЦЗ та з різними параметрами згортки.
5. Статистичні методи: обробка та аналіз експериментальних даних; візуалізація результатів досліджень з використанням графіків та діаграм.
6. Методи оптимізації: застосування технологій SIMD, зокрема інструкцій NEON64; оптимізація алгоритмів для роботи з цілочисельними даними.
7. Методи системного аналізу: аналіз ефективності розробленого алгоритму в контексті різних сценаріїв застосування та апаратних платформ.

Наукова новизна одержаних результатів полягає у такому:

Вперше:

1. Розроблено та теоретично обґрунтовано метод ШОЗ для обробки 8-бітних RGB ЦЗ з використанням 16-бітних цілочисельних операцій, що забезпечує підвищення швидкодії на центральних процесорах архітектури ARM64

порівняно з аналогічним підходом на основі функції cv::filter2D бібліотеки OpenCV: для ядер згортки (ЯЗ) середнього розміру (8×8–11×11) досягнуто прискорення в діапазоні 3,1–11,1 раза для Orange Pi 5 Pro та 2,7–10,2 раза для Raspberry Pi 4; для великих ЯЗ (12×12–15×15), де бібліотека OpenCV може використовувати FFT-алгоритми, запропонований метод демонструє прискорення 1,8–5,8 раза для Orange Pi 5 Pro та 1,7–5,2 раза для Raspberry Pi 4; для операцій масштабування ЦЗ, що включають чотириразове застосування ОЗ, розроблений підхід (zoomUP_2__NEON64) показує прискорення до 6,3 раза (Orange Pi 5 Pro) та до 6,2 раза (Raspberry Pi 4) для ЯЗ середнього розміру порівняно з аналогічною операцією на базі cv::filter2D;

2. Запропоновано формалізовану нотацію для опису векторизованих обчислень з використанням SIMD NEON64, яка дозволяє представляти опис векторизованої частини коду у вигляді структурованих виразів, що сприяє кращій читабельності та знижує поріг входу при засвоєнні технік експертної векторизації.

Удосконалено технологію обробки ЦЗ на комп'ютерах, що підтримують паралелізм обчислень, шляхом запропонованої в роботі ІТ обробки зображень та відео з використанням ШОЗ для архітектури ARM64, яка дозволяє в режимі реального часу проводити фільтрацію та масштабування потокового відео та ЦЗ на одноплатних комп'ютерах з ЦП ARM64, як-от: Raspberry Pi 4 та Orange Pi 5 Pro.

Набули подальшого розвитку лінійні оператори на основі лінійних комбінацій B-сплайнів, що є близькими до інтерполяційних у середньому, а саме: отримано лінійні оператори 16-бітних цілочисельних операцій, які гарантують відсутність переповнення при 16-бітному обчисленні векторизованого скалярного добутку (ВСД), що дозволяє їх застосування для обробки ЦЗ на основі розробленого методу, а саме для субсмугової фільтрації, контрастування, цифрової стабілізації ЦЗ та для рекурентного двократного збільшення лінійних розмірів ЦЗ.

Практичне значення отриманих результатів:

1. Розроблено алгоритм методу ШОЗ для обробки RGB ЦЗ з використанням 16-бітних цілочисельних обчислень (Розділ 3), що включає формалізацію векторизованих обчислень, адаптацію ЯЗ та його (алгоритму) оптимізацію для SIMD-архітектур ЦП.

2. На основі розробленої алгоритмізації створено програмну реалізацію ОЗ шляхом використання інструкцій SIMD NEON ARM64, що забезпечує прискорення обробки ЦЗ до 11,1 раза (Orange Pi 5 Pro) та до 10,2 раза (Raspberry Pi 4) порівняно з аналогічним підходом на основі функції cv::filter2D бібліотеки OpenCV на SBC та подібних системах.

3. Запропоновано та апробовано методику експериментальної оцінки та порівняння швидкодії різних програмних реалізацій ОЗ на платформах з архітектурою центрального процесора ARM64, яка дозволяє об'єктивно визначати переваги запропонованих та оптимізованих у роботі алгоритмів.

4. Розроблена технологія теоретично має кращу енергоефективність порівняно з реалізаціями на основі операцій з плаваючою комою завдяки використанню цілочисельних обчислень та скороченню часу виконання операцій завдяки використанню SIMD ARM64 NEON, що є перспективним напрямом для подальших досліджень енергоспоживання на мобільних та вбудованих пристроях.

5. Вперше проведено комплексне дослідження ефективності запропонованої ОЗ для різних розмірів ЦЗ (від 256×144 до 3840×2160) та розмірів ЯЗ (від 2×2 до 15×15) на SBC з архітектурою ARM64 (Orange Pi 5 Pro, Raspberry Pi 4), що дозволило виявити найбільш швидкодіючі сценарії застосування розробленого методу, зокрема при використанні ЯЗ розміром 8×8 було отримано

найбільші показники підвищення швидкодії, а саме, від 5,2 до 11,1 раза (Orange Pi 5 Pro) та від 4,5 до 10,2 раза (Raspberry Pi 4) порівняно з cv::filter2D OpenCV.

6. Розроблений у роботі алгоритм ОЗ та рекомендації щодо його застосування можуть бути використані в системах комп'ютерного зору на базі архітектури ЦП ARM64, що дозволяє підвищити швидкодію обробки ЦЗ без необхідності оновлення апаратного забезпечення, зокрема в автономних UAV-системах, що відповідає актуальним пріоритетним напрямам досліджень та розвитку обороноздатності України.

7. Результати дослідження можуть бути застосовані для покращення швидкодії систем на основі архітектури ЦП ARM64, зокрема систем відеоспостереження, аналізу супутникових знімків та обробки даних з камер БПЛА, що є актуальним при вирішенні завдань національної безпеки та оборони.

8. Розроблений метод ШОЗ може бути інтегрований в існуючі бібліотеки обробки ЦЗ (наприклад, OpenCV) для підвищення їх ефективності на платформах ARM64, що є перспективним напрямом розвитку високопродуктивних бібліотек обробки зображень.

9. Результати дослідження можуть бути покладені в основу розробки спеціалізованих SIMD-методів обробки ЦЗ, які будуть використовувати описану в (Додаток А) методику підвищення ШОЗ.

10. Запропоновані алгоритми ШОЗ можуть бути застосовані для прискорення ОЗ в згорткових нейронних мережах на мобільних та вбудованих пристроях, що сприятиме розвитку мобільних додатків з елементами штучного інтелекту.

11. Розроблена ІТ та програмне забезпечення (бібліотека libconv_lib.so на C++ з SIMD ARM64 NEON-оптимізаціями та Python-інтерфейс) надають готовий інструментарій для розробників систем, що потребують високої швидкодії обробки ЦЗ на платформах ARM64, зокрема в системах бортової обробки даних БПЛА, мобільних робототехнічних комплексах та інших критичних застосунках, де поєднання швидкодії та енергоефективності є ключовим.

12. Результати досліджень впроваджено в освітній процес при розробці курсів: «Архітектура обчислювальних систем та операційні системи», «Системне програмування» та «Програмування мовою асемблер».

13. Результати роботи впроваджено у виробничу діяльність ТОВ «ЕЛІСАТ» (вбудоване програмне забезпечення пристроїв збору й обробки даних з датчиків) та ТОВ «ІНСПІРІТО ТІМ» (програмно-апаратні засоби обробки відеоданих з бортових камер БПЛА), що підтверджено відповідними актами впровадження (див. Додаток Г та Додаток Д).

Особистий внесок здобувача. Дисертаційна робота є результатом самостійного дослідження. У роботах, виконаних разом зі співавторами, автору належить: у роботах [1, 2, 4–10, 13, 14] – розробка концепції та архітектури інформаційної технології підвищення швидкодії обробки цифрових зображень на основі векторизованої операції згортки з використанням 16-бітних цілочисельних обчислень та SIMD ARM64 NEON; у роботах [2, 5] – теоретичне обґрунтування та розробка математичних моделей операції згортки для 16-бітних цілочисельних обчислень при обробці 8-бітних RGB-зображень, у тому числі побудова умови коректності, методики адаптації знакових ядер згортки до беззнакових цілочисельних операцій та оптимізації для розріджених ядер; у роботах [2, 4] – формулювання та обґрунтування формалізованої нотації для опису векторизованих обчислень з використанням SIMD NEON64; у роботах [1, 2, 4] – безпосередня програмна реалізація ключових компонентів інформаційної технології, у тому числі розробка бібліотеки на C/C++ з NEON64-оптимізаціями та Python-інтерфейсу для прикладного використання; у роботах [1–4] – планування, організація та проведення експериментальних досліджень оцінки швидкодії розробленої технології, у тому числі порівняльний аналіз з функцією cv::filter2D бібліотеки

OpenCV на платформах Raspberry Pi 4 та Orange Pi 5 Pro; у роботах [2, 4, 5, 7] – аналіз отриманих експериментальних даних та формулювання висновків і рекомендацій за результатами дослідження.

Апробація результатів дисертації. Результати дисертаційної роботи доповідались та обговорювались на: XVI, XIX, XXI, XXIII Міжнародній науково-практичній конференції здобувачів вищої освіти і молодих учених «ПОЛІТ. Сучасні проблеми науки» (м. Київ, 2016, 2019, 2021, 2023), VIII, XXI Всесвітньому конгресі «Авіація у XXI столітті» – «Безпека в авіації та космічні технології» (м. Київ, 2018, 2022), XIII, XVI Міжнародній науково-технічній конференції «AVIA» (м. Київ, 2017, 2023).

Публікації. Основні положення дисертаційної роботи опубліковано в 7 статтях, з яких 3 – у виданнях, що входять до переліку фахових ВАК (нині – категорія Б МОН України), 4 – у виданнях, що індексуються в наукометричних базах Scopus та Web of Science, а також у 7 тезах доповідей на конференціях.

Структура роботи та її обсяг. Дисертація складається з анотації, вступу, чотирьох розділів, висновків, додатків, списку використаних джерел і має 155 сторінок основного тексту, 11 рисунків, 18 таблиць, 91 сторінку додатків. Список використаних джерел містить 100 найменувань і займає 11 сторінок. Загальний обсяг роботи 246 сторінок.

ОСНОВНА ЧАСТИНА

У вступі обґрунтовано актуальність роботи, поставлено мету і задачі досліджень, визначено наукову новизну і практичну цінність отриманих результатів, наведено дані про їх апробацію та впровадження.

У першому розділі проведено аналіз сучасних підходів до обробки ЦЗ, зокрема ОЗ як ключового етапу в задачах фільтрації, масштабування, детекції ознак та обчислень у CNN. Розглянуто два напрямки оптимізації обчислення ОЗ. Апаратне прискорення досягається використанням SIMD-розширень ЦП (NEON64 для ARM64, SSE/AVX для x86/64), графічних прискорювачів (CUDA, OpenCL) та спеціалізованих співпроцесорів. Програмне прискорення досягається за рахунок використання автовекторизації сучасних компіляторів (GCC, Clang) та низькорівневих SIMD-оптимізацій, наявних у бібліотеках OpenCV та ARM Compute Library. Проаналізовано два підходи до ручної векторизації: Intrinsic-функції та Inline Assembly. Досліджено техніки оптимізації коду – розгортання циклів та кеш-ефективну організацію доступу до даних для ARM64. Виявлено, що механізм Universal Intrinsics в OpenCV забезпечує кросплатформеність, але вносить додаткові накладні витрати порівняно з експертною векторизацією для конкретної архітектури.

На основі проведеного аналізу у розділі зроблено такі висновки:

1. ОЗ є ресурсоємною, але найвживанішою операцією обробки ЦЗ.
2. ОЗ має високу ефективність при реалізації на SIMD-архітектурах завдяки можливості паралельної обробки та мінімізації пам’яттєвих звернень.
3. Існуючі програмні рішення, як cv::filter2D(...), використовуючи Universal Intrinsics, вносять додаткові накладні витрати у порівнянні з експертною векторизацією коду, що виконані для конкретної архітектури.
4. Архітектура ARM64 з розширенням NEON64 є перспективною і найвикористованішою платформою та є перспективною для дослідження і реалізації на ній в подальшому ШОЗ.
5. Є актуальна потреба в розробці нової ІТ, котра надає спеціалізовану ШОЗ ЦЗ із урахуванням особливостей ЦП ARM64 і 16-бітних обчислень.

Другий розділ присвячено теоретичному обґрунтуванню та формалізації лінійних операторів, призначених для ефективної обробки ЦЗ у форматі RGB із застосуванням 16-бітних цілочисельних обчислень. Мета розділу – надати

математичну основу для побудови операторів, що відповідають вимогам обмежених ресурсів одноплатних комп'ютерів (SBC) на архітектурі ARM64, а також забезпечують точність і стабільність обробки.

Операцію згортки над кольоровою складовою зображення подамо так:

$$F\left(p^{i,j}\right)=\sum_{ii=i-r_i}^{i+r_i}\sum_{jj=j-r_j}^{j+r_j}\gamma_{ii-i,jj-j}p_{ii,jj}\,, \tag{1}$$

де: $p_{i,j}$ – кольорова складова растру; (i,j) – індекс пікселя растру; $\gamma_{ii-i,jj-j}$ – елемент маски фільтру; $(2r_i+1)\times(2r_j+1)$ – розмір маски фільтра.

Основна вимога, яка ставиться до реалізації (1) при використанні 16-бітних обчислень – забезпечення допустимого діапазону значень, що не виходять за межі типу *int16_t*. З огляду на особливості апаратної реалізації SIMD-інструкцій NEON64 та підтримку 16-бітних векторних регістрів, використання (1) вимагає попередньої перевірки ЯЗ на узгодженість з розрядністю. Ці обмеження подано у вигляді нерівності (2), яка гарантує, що максимальне значення згортки не перевищить межі цілочисельного 16-бітного представлення.

$$\sum_{ii=i-r_i}^{i+r_i}\sum_{jj=j-r_j}^{j+r_j}\left|\gamma_{ii-i,jj-j}\right|\le 127 \tag{2}$$

Далі подано приклади масок, що задовольняють умові (2) при вирішенні задач обробки ЦЗ при 16-бітних обчисленнях.

Для реалізації субполосної фільтрації послідовності $\left\{p_{i,j}\right\}_{i,j\in\mathbb{Z}}$ припускаємо, що значення інтенсивності у кожному пікселі можна подати у вигляді суми:

$$p_{i,j}=pL_{i,j}+pH_{i,j}\,,$$

де: $pL_{i,j}$, $pH_{i,j}$ – відповідно, низько- та високочастотні складові, які можна визначати на основі таких лінійних операторів:

$$pL_{i,j}=L\left(p^{i,j}\right)=\sum_{ii=i-2}^{i+2}\sum_{jj=j-2}^{j+2}\gamma L_{ii-i,jj-j}p_{ii,jj}\,,\quad i,j\in\mathbb{Z}\,,$$

$$pH_{i,j}=H\left(p^{i,j}\right)=\sum_{ii=i-2}^{i+2}\sum_{jj=j-2}^{j+2}\gamma H_{ii-i,jj-j}p_{ii,jj}\,,\quad i,j\in\mathbb{Z}$$

де:

$$\gamma L=\frac{1}{121}\begin{pmatrix}0&0&1&0&0\\0&6&15&6&0\\1&15&33&15&1\\0&6&15&6&0\\0&0&1&0&0\end{pmatrix};\qquad \gamma H=\frac{1}{121}\begin{pmatrix}0&0&-1&0&0\\0&-6&-15&-6&0\\-1&-15&88&-15&-1\\0&-6&-15&-6&0\\0&0&-1&0&0\end{pmatrix}. \tag{3}$$

Для контрастування ЦЗ можна використовувати оператор:

$$pK_{i,j} = K\left(p^{i,j}\right) = \sum_{ii=i-2}^{i+2}\sum_{jj=j-2}^{j+2} \gamma K_{ii-i,jj-j} p_{ii,jj}\ , \quad i,j \in \mathbb{Z}\ ,$$

де $\left\{pK_{i,j}\right\}_{i,j\in\mathbb{Z}}$ – кольорова складова відконтрастованого растру;

$$\gamma K = \frac{1}{16}\begin{pmatrix} 0 & 0 & 2 & 0 & 0 \\ 0 & 3 & -13 & 3 & 0 \\ 2 & -13 & 48 & -13 & 2 \\ 0 & 3 & -13 & 3 & 0 \\ 0 & 0 & 2 & 0 & 0 \end{pmatrix}, \tag{4}$$

який, до того ж, є псевдо зворотнім оператором оператора $L\left(p^{i,j}\right)$ низькочастотної фільтрації з маскою (3), тобто виконується приблизне співвідношення:

$$p_{i,j} \approx K\left(L\left(p^{i,j}\right)\right), \quad i,j \in \mathbb{Z}\ , \tag{5}$$

Стабілізація цифрового зображення, спотвореного мікрорухом камери фіксації (підвищення різкості) можлива за використанням такого оператора:

$$pR_{i,j} = R\left(p^{i,j}\right) = \sum_{ii=i-2}^{i+2}\sum_{jj=j-2}^{j+2} \gamma R_{ii-i,jj-j} p_{ii,jj}\ , \quad i,j \in \mathbb{Z}\ ,$$

де $\left\{pR_{i,j}\right\}_{i,j\in\mathbb{Z}}$ – кольорова складова растру з покращеною різкістю;

$$\gamma R = \frac{1}{42}\begin{pmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & -8 & 1 & 0 \\ -1 & -8 & 74 & -8 & -1 \\ 0 & 1 & -8 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{pmatrix} \quad \text{чи} \quad \gamma R = \frac{1}{15}\begin{pmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & 1 & -4 & 1 & 0 \\ -1 & -4 & 31 & -4 & -1 \\ 0 & 1 & -4 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{pmatrix} \tag{6}$$

Зауважимо, що використання масок (6) забезпечує, відповідно, менше та більше підвищення різкості.

Для чотирикратного рекурентного поповнення кількості членів послідовності $\left\{p_{i,j,0}\right\}_{i,j\in\mathbb{Z}}$ (двократного рекурентного збільшення лінійних розмірів ЦЗ) достатньо на кожному κ-му ($\kappa = 1,2,\ldots$) кроці рекурсії визначати члени нової послідовності кольорових складових растру $\left\{p_{2i,2j,\kappa}\right\}_{i,j\in\mathbb{Z}}$, на основі лінійних операторів, що основані на даних попереднього кроку рекурсії: $p_{2i,2j,\kappa} = A\left(p^{\kappa-1,i,j}\right)$, $p_{2i+1,2j,\kappa} = B\left(p^{\kappa-1,i,j}\right)$, $p_{2i,2j+1,\kappa} = C\left(p^{\kappa-1,i,j}\right)$, $p_{2i+1,2j+1,\kappa} = D\left(p^{\kappa-1,i,j}\right)$, $i,j \in \mathbb{Z}$, де $A\left(p^{\kappa-1,i,j}\right)$; $B\left(p^{\kappa-1,i,j}\right)$; $C\left(p^{\kappa-1,i,j}\right)$; $D\left(p^{\kappa-1,i,j}\right)$ можна задавати з умов масштабування. Так, якщо є потреба збільшувати ЦЗ, то мають місце такі оператори:

$$p_{a,b,\kappa} = \sum_{ii=i-2}^{ii+2}\sum_{jj=j-2}^{j+2} \gamma\Lambda_{ii-i,jj-j} p_{ii,jj,\kappa-1}\ ,$$

де:

$$\Lambda = \begin{cases} A: a = 2i, b = 2j, \\ B: a = 2i+1, b = 2j, \\ C: a = 2i, b = 2j+1, \\ D: a = 2i+1, b = 2j+1; \end{cases} ; \tag{7}$$

$$\gamma A = \frac{1}{50}\begin{pmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ -1 & 2 & 46 & 2 & -1 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{pmatrix}; \qquad \gamma B = \frac{1}{72}\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -6 & 0 & 0 \\ -1 & 2 & 40 & 2 & -1 \\ -1 & 2 & 40 & 2 & -1 \\ 0 & 0 & -6 & 0 & 0 \end{pmatrix};$$

$$\gamma C = \frac{1}{72}\begin{pmatrix} 0 & 0 & -1 & -1 & 0 \\ 0 & 0 & 2 & 2 & 0 \\ 0 & -6 & 40 & 40 & -6 \\ 0 & 0 & 2 & 2 & 0 \\ 0 & 0 & -1 & -1 & 0 \end{pmatrix}; \qquad \gamma D = \frac{1}{20}\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 0 \\ 0 & -1 & 7 & 7 & -1 \\ 0 & -1 & 7 & 7 & -1 \\ 0 & 0 & -1 & -1 & 0 \end{pmatrix}. \tag{8}$$

Для рекурентного двократного збільшення масштабу (двократного зменшення горизонтального та вертикального розмірів) ЦЗ необхідно на кожному κ-му ($\kappa = 1,2,\ldots$) кроці рекурсії чотирикратно зменшувати кількість пікселів, звільнюючи у новому κ-му растрі місце з-під трьох пікселів: праворуч, зверху та зверху-навскіс від кожного (i,j)-го пікселя $(\kappa-1)$-го растру. В такому разі величини $p_{i,j,\kappa}$ визначаються на підставі деякого лінійного функціоналу:

$$p_{i,j,\kappa} = \sum_{ii=2i-r}^{2i+r}\sum_{jj=2j-r}^{2j+r} \gamma_{ii-i,jj-j}\, p_{ii,jj,\kappa-1},\ i,j \in \mathbb{Z},\ r = 1,2,\ldots, \tag{9}$$

що побудований на даних попереднього кроку рекурсії, а в якості γ можна використовувати маски (3, 4) та (6).

Крім того, у другому розділі дисертаційної роботи досліджено питання точності результатів згортки у 16-бітному представленні у порівнянні з 32-бітним еталонним результатом, отриманим за допомогою бібліотеки OpenCV. Проведено оцінку похибки для різних типів фільтрів і підтверджено придатність запропонованих схем для використання в системах реального часу. Отримані векторизовані оператори формують обчислювальні схеми з мінімальним використанням арифметичних операцій, що задовольняють вимогам високої швидкодії та можливості впровадження в енергообмежене середовище, зокрема на БПЛА та мобільних пристроях.

Таким чином, у другому розділі сформовано математичну та алгоритмічну базу для побудови швидкодіючого та енергоефективного векторизованого ЯЗ на SIMD ARM64 NEON із застосуванням 16-бітних цілочисельних обчислень, що лягло в основу ІТ, розробленої в наступних розділах роботи.

У третьому розділі дисертації запропоновано ІТ обробки ЦЗ та відео з використанням ШОЗ для архітектури ARM64, здійснено розробку алгоритмічного та програмного забезпечення для реалізації ШОЗ RGB-ЦЗ з використанням NEON64 на архітектурі ЦП ARM64. Розділ містить опис ШОЗ, виконаний за допомогою формалізованої нотації, яка була використана для опису ШОЗ та інструкцій SIMD ARM NEON64. Самі інструкції NEON64, описані формалізованою нотацією, наведено у Додатку А.

На основі математичних операторів, типу (3–9), запропонованих у розділі 2, реалізовано набір обчислювальних процедур, що забезпечують на основі SIMD одночасну обробку фрагментів ЦЗ, використовуючи 128-бітні NEON-регістри. Приділено увагу розгортанню циклів, попередньому завантаженню рядків (***prefetch***), вирівнюванню даних в пам'яті та опису за допомогою формалізованої нотації векторних обчислень для типів даних (int16x8_t, uint8x8_t, uint8x16_t), що зберігає лінійність обчислень і мінімізує операції завантаження.

Розроблені процедури ШОЗ включають такі етапи опису як: завантаження даних (LD3), яке виконується 1 раз на 16 операцій; векторне множення з коефіцієнтами ЯЗ та накопичення результату SMLAL/SMLAL2; операції конвертації даних (SXTL) з s16 ⇒f32 та з s32 ⇒ f32 – (SCVTF).

Значущим є використання операції циклічного зсуву, яка ефективно зсуває дані одного векторного регістру за рахунок даних сусіднього/вказаного (реалізована за допомогою інструкції ***ext*** векторного регістра). Циклічний зсув прибирає як мінімум 15 операцій завантаження, що і призводить до ефективного розмотування циклів ОЗ ЦЗ (детально описано в Додатку А). Нехай flt_{flt_i,flt_j} є елементами відповідного двовимірного вектора (матриці) *flt*, де: $flt_i = \overline{0, flt_rows - 1}$, де *flt_rows* відповідно позначають кількість рядків у матриці *flt*; $flt_j = \overline{0, flt_cols - 1}$, де *flt_cols* відповідно позначають кількість стовпців у матриці *flt*. Отже можна переписати ОЗ (1) у вигляді виразу:

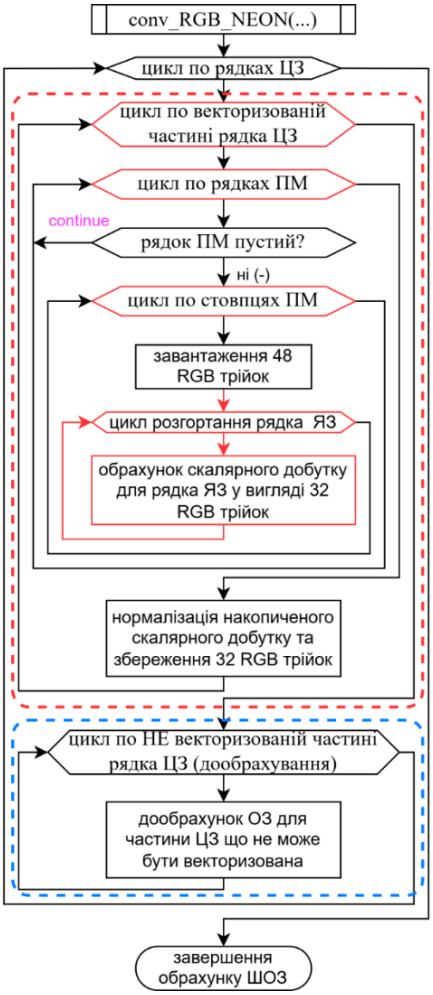


Рисунок 1 – Блок-схема алгоритму обчислення ШОЗ для цілочисельних ЯЗ

$$dst_{src_i+\left\lfloor\frac{flt_rows}{2}\right\rfloor,src_j+\left\lfloor\frac{flt_cols}{2}\right\rfloor} = \sum_{flt_i=0}^{flt_rows-1}\sum_{flt_j=0}^{flt_cols-1} src_{src_i+flt_i,src_j+flt_j}\cdot flt_{flt_i,flt_j}, \quad (10)$$

де, матриця вхідного ЦЗ SRC має наступні діапазони значень індексів $src_i = \overline{0,src_rows - flt_rows}$ та $src_j = \overline{0,src_cols - flt_cols}$. Також у (10) використовується взаємо-пов'язаність індексів матриць SRC та DST. Т.ч. рівняння (10) є аналогом/продовженням основного рівняння (1).

Для спрощення рівняння (10) до рівня цілочисельних операцій, можна перетворити (10) у наступне рівняння:

$$dst_{src_i+\left\lfloor\frac{flt_rows}{2}\right\rfloor,src_j+\left\lfloor\frac{flt_cols}{2}\right\rfloor} = flt_div\cdot\left(\sum_{flt_i=0}^{flt_rows-1}\sum_{flt_j=0}^{flt_cols-1} src_{src_i+flt_i,src_j+flt_j}\times flt_{flt_i,flt_j}\right), \quad (11)$$

де, flt_div має тип *float*. Дужки введені для позначення першочерговості виконання операцій.

Процес векторизації коду у даному дослідженні передбачає використання шести векторних регістрів для завантаження/збереження базової частини рядка ЦЗ розміром в 32 8-бітних трійок RGB. Це призводить, наприкінці, до стабільно зайнятих 24 векторних регістрів. З цього випливає, що кожен канал ЦЗ займає на початку обчислень 2 векторних регістра, які згодом «розкриваються» у 8 векторних регістрів для підтримки 32-бітних операцій. Тож, можна обрахувати кількість ітерацій обчислень ШОЗ ЦЗ, знаючи: розміри ЦЗ; розміри ЯЗ; базову кількість векторів для збереження одного каналу ЦЗ – 2, що вміщує в себе 32 елементи одного каналу ЦЗ. Кількість векторизованих ОЗ для одного рядка ЦЗ визначається наступним чином:

$$NC_vec = \left\lfloor\frac{src_cols - flt_cols + 1}{32}\right\rfloor, \quad (12)$$

де $\lfloor\bullet\rfloor$ – округлення до цілого вниз, NC_vec – від англ. «number of columns that were vectorized», а діапазон обрахунків для рядка ЦЗ можна записати так:

$$st32N = \overline{0,NC_vec-1}, \quad (13)$$

Знаючи NC_vec (12) можна обрахувати загальну кількість саме векторизованих ОЗ для всього ЦЗ:

$$N_vec = NC_vec\cdot NR_vec, \quad (14)$$

де – $NR_vec = src_rows - flt_rows + 1$ (від англ. «number of rows»).

Використовуючи (12, 14) можна записати (11) у вигляді, що описує обчислення тільки векторизованої частини ОЗ ЦЗ:

$$dst_{src_i+\left\lfloor\frac{flt_rows}{2}\right\rfloor,idx_{base}(st32N)+j+\left\lfloor\frac{flt_cols}{2}\right\rfloor} = flt_div\cdot \left(\sum_{flt_i=0}^{flt_rows-1}\sum_{flt_j=0}^{flt_cols-1} src_{src_i+flt_i,idx_{base}(st32N)+j+flt_j}\cdot flt_{flt_i,flt_j}\right) \quad (15)$$

де: $idx_{base}(st32N) = 32\cdot st32N$, $j = \overline{0,31}$.

Блок-схема алгоритму обчислення ШОЗ наведена на (рис. 1); її можна поділити на дві частини: 1) ШОЗ ЦЗ (векторизована частина ОЗ ЦЗ (1-ше обведення)) та дообрахунок ОЗ ЦЗ (не векторизована частина ОЗ ЦЗ (2-ге обведення)). 2) обрахунок ВСД по рядках та у стовпцях ЯЗ/ПМ і накопичення результату ВСД у 12 векторних регістрах; 3) використавши накопичений ВСД, провести завершення обрахунку ШОЗ ЦЗ та збереження результату у 32 RGB пікселі вихідного ЦЗ. Даний процес повторюється NC_vec разів для кожного рядка ЦЗ (12).

Запропонована реалізація ШОЗ є ядром розробленої інформаційної технології (ІТ) обробки ЦЗ і демонструє здатність до масштабування на різних пристроях з ЦП ARM64.

Усі реалізовані та описані далі програмні компоненти оптимізовано на рівні асемблерного коду із використанням Inline Assembly, що дозволяє безпосередньо керувати обчисленнями на рівні SIMD-регістрів.

Тож, для даного дослідження сформульовано завдання для проведення векторізації (15), котре можна розбити на дві частини: 1 – ВСД по рядку ЯЗ; 2 – ВСД по стовпцю ЯЗ – найнавантаженіша частина обрахуків для ШОЗ. Щодо завершальних обчислень ОЗ ЦЗ, вони виконуються за допомогою мови подібних алгоритмів, що було використано для векторизованої частини C/C++, за виключенням того, що не було використано SIMD (без мануальної/штучної векторизації). Основні кроки алгоритму обчислення ШОЗ ЦЗ наведені для 1ї частини (ШОЗ ЦЗ) і включають у себе завантаження 48 RGB елементів (32 RGB пікселя для обрахунку результату і 16 для розмотування циклу і обрахунку скалярного добутку) у 6 векторні регістри (за умови що предікт-матриці (ПМ) не є пустою).

Формула кількості кроків дообрахування ОЗ ЦЗ для одного рядка, записується наступним чином (базуючись на (12)):

$$NC_rest = (src_cols - flt_cols + 1) - NC_vec \cdot 32 . \quad (16)$$

Таким чином, можна попередньо описати алгоритм обчислення ОЗ ЦЗ у вигляді блок-схеми на основі (11) використовуючи кількість ШОЗ для одного рядка ЦЗ (12) і знаючи кількість дообрахувань (16) ОЗ, використавши показник (14).

Розроблена ІТ підвищення швидкодії обробки ЦЗ базується на методі ШОЗ для платформ ARM64 та складається з таких компонентів:

1. Набір спеціалізованих лінійних операторів для обробки ЦЗ під час 16-бітних обчислень, що складаються з 8-бітних цілочисельних ЯЗ розміром 3×3 – 5×5 та коефіцієнтів нормалізації типу (3–9) і забезпечують: низькочастотну, високочастотну та субсмугову фільтрацію; контрастування та підвищення різкості зображень; рекурентне двократне масштабування з покращенням якості.

2. Метод ШОЗ – обґрунтований підхід до виконання ОЗ для ЦЗ, що ґрунтується на: використанні 16-бітних цілочисельних обчислень замість 32-бітних з плаваючою комою, що забезпечує оптимальне співвідношення точності та швидкодії; паралельній обробці даних за допомогою SIMD NEON64.

3. Алгоритмічне забезпечення ядра ІТ ШОЗ включає: алгоритм розгортання циклів ШОЗ для мінімізації накладних витрат на керування циклами; кеш-

ефективну організацію доступу до даних зображення; алгоритм кластеризації ЯЗ та формування ПМ для оптимізації обробки розріджених матриць під час виконання ШОЗ.

4. Формалізована нотація для опису векторизованих обчислень SIMD NEON64, що дозволяє абстрагуватися від конкретних асемблерних інструкцій та представляти операції у вигляді математичних виразів.

5. Програмна реалізація ІТ у вигляді: обчислювального ядра conv_RGB_NEON, реалізованого як С-бібліотека (libconv_lib.so); програмного інтерфейсу для інтеграції з Python-застосунками.

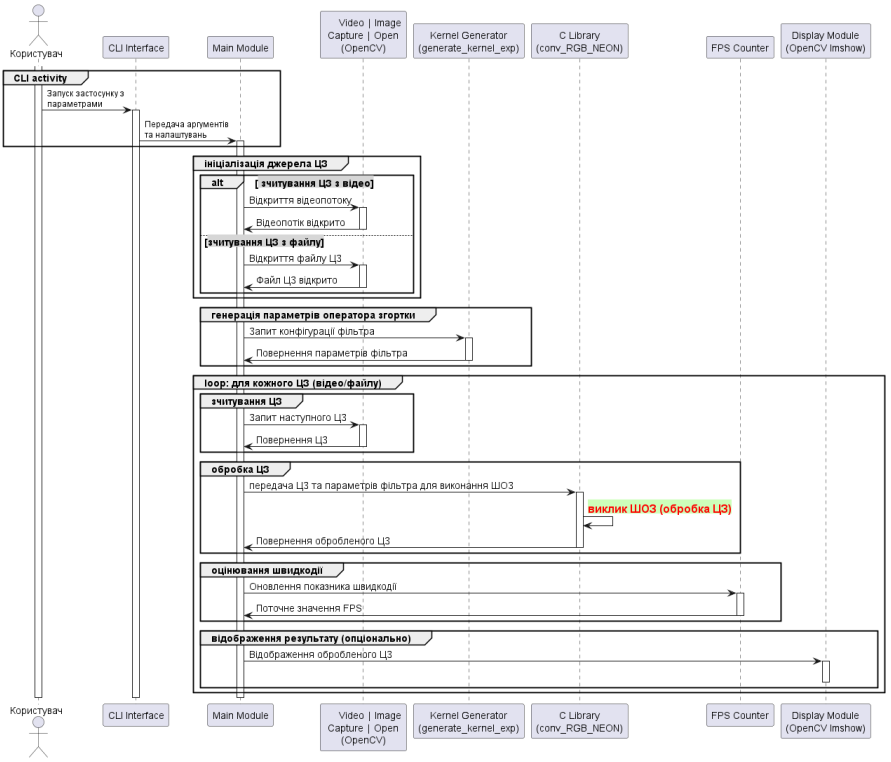


Рисунок 2 – Інтеграція ШОЗ як елемента підвищення швидкодії ІТ

Програмна реалізація бібліотеки libconv_lib.so базується, як на ядрі ІТ ШОЗ – conv_RGB_NEON(...), так і на функціях подібних conv_RGB_NEON(...), що не використовують ПМ; використання в Python застосунку здійснюється через інтерфейс модуля ctypes. Все це забезпечує контроль над обчислювальними ресурсами і зменшує кількість множень на 0.

Запропонована ІТ може бути інтегрована в edge-пристрої, з акцентом на високу пропускну здатність обробки зображень у реальному часі. Аргументовано доцільність використання саме 16-бітних цілочисельних обчислень, які, на

відміну від операцій з плаваючою комою, забезпечують кращу швидкодію при збереженні достатньої точності результату.

Складові бібліотеки libconv_lib.so реалізують всі необхідні функції для обробки зображень, включаючи ініціалізацію, виклик ШОЗ, масштабування та вивід результатів.

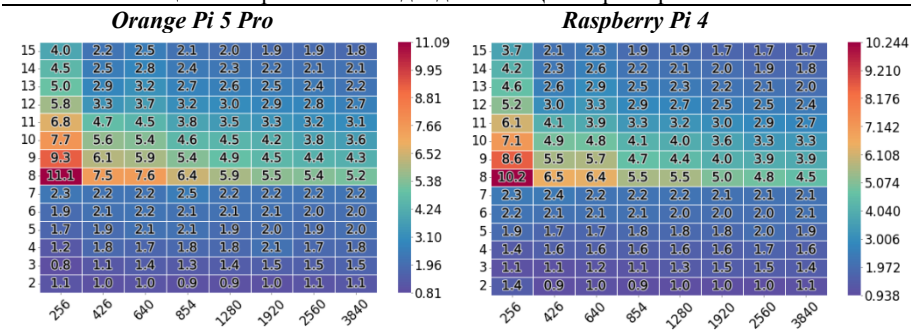
Для наочного представлення функціональної структури ІТ у роботі наведено схему взаємодії компонентів (рис. 2). На ній зображено повний цикл обробки одного кадру зображення – від отримання вхідного зображення, формування ЯЗ, виклику функції conv_RGB_NEON, виконання векторизованої згортки, збереження результату до аналізу швидкодії.

У третьому розділі також наведено технічні аспекти програмної реалізації ядра ІТ, включно з форматом функцій, сигнатурою виклику, специфікою компіляції під ARM64-платформу, з урахуванням параметрів компіляторів та особливостей підтримки SIMD-розширень у GCC і Clang. Окрему увагу приділено методикам практичного застосування ІТ для обробки як статичних зображень, так і потокового відео. Висвітлено реалізацію інтерфейсних компонентів, включаючи Python-обгортку. Також надано приклади сценаріїв використання ІТ в задачах комп'ютерного зору. Наведено приклади поєднання кількох ЯЗ, адаптації параметрів залежно від типу задачі, а також вказано обмеження й рекомендації щодо їх ефективного налаштування.

У четвертому розділі роботи наведено результати експериментального дослідження швидкодії запропонованої ІТ обробки ЦЗ, реалізованої на базі NEON64 для архітектури ARM64. Дослідження охоплює повний цикл оцінки ефективності програмної реалізації та порівняння з існуючими підходами, зокрема з функцією cv::filter2D(...).

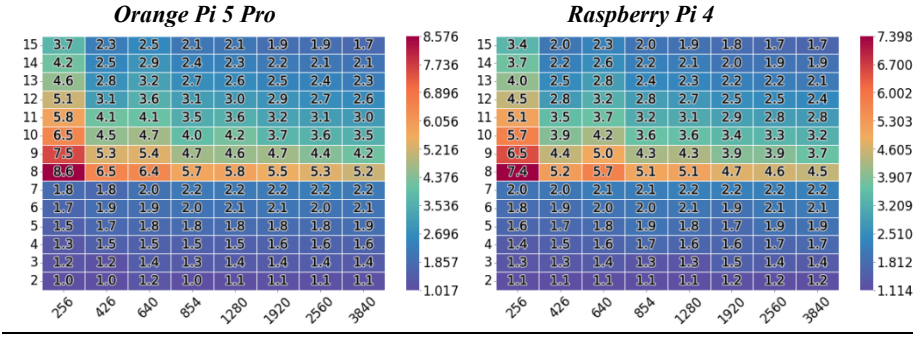
Проведення вимірювань здійснювалось на SBC-платформі Orange Pi 5 Pro (Ubuntu 24.04) та Raspberry Pi 4 (Raspberry Pi OS на базі Debian). Для забезпечення відтворюваності експериментів застосовувалися Shell-скрипти, що забезпечують послідовну зміну параметрів (розмірів ЯЗ, роздільної здатності кадру, типу ядра) та накопичення статистично значущої вибірки з не менше ніж 150 вимірів для кожної конфігурації.

Таблиця 1 – Порівняння швидкодії ШОЗ ЦЗ. Безрозмірний показник



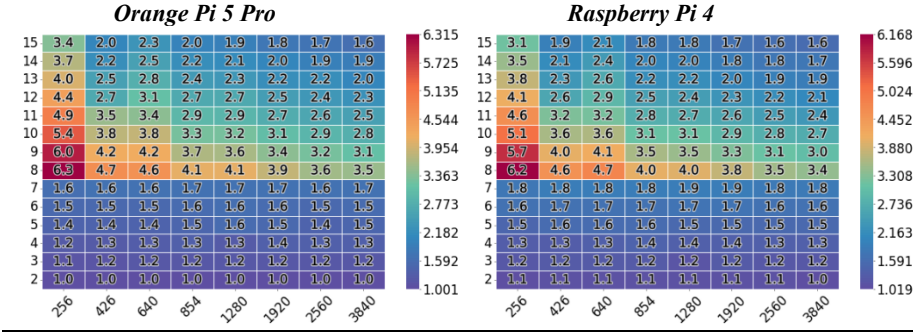
Для проведення досліджень було сформовано тестове середовище, що включало: дві SBC-платформи – Orange Pi 5 Pro та Raspberry Pi 4; еталонне ПЗ (бібліотека OpenCV (cv::filter2D) з використанням зібраної в релізній конфігурації + NEON64-акселерація); модуль тестування на Shell для фіксації часу виконання та завершення виконання поточної конфігурації і т.п. Було використано набір із 8 тестових відеофайлів, котрі мають різну роздільну здатність ЦЗ (від 256×144 до 3840×2160) та вмісту (статичні, динамічні сцени, зображення з шумом, із дронів). Для кожного зображення виконувалась фільтрація з використанням типових ЯЗ розмірами від 2×2 до 15×15.

Таблиця 2 – Показники порівняння швидкодії ШОЗ ЦЗ в Python-додатку на різних платформах. Безрозмірний показник.



Оцінювання проводилося за такими критеріями: забір не менше 150 FPS показників для кожної конфігурації – [ширина ЦЗ × ширину ЯЗ]; максимальна похибка, порівняно з 32-бітним результатом OpenCV, має не перевищувати 1; обсяг зайнятої оперативної пам'яті не має перевищувати 70 % від загального обсягу доступної RAM.

Таблиця 3 – Показники порівняння швидкодії ШОЗ ЦЗ операцій масштабування в Python-додатку. Безрозмірний показник



У таблиці (табл. 1) подано усереднені показники швидкодії для функцій cv::filter2D(...) та conv_RGB_NEON(...) при обробці ЦЗ розмірами від 256×144

до 3840×2160 пікселів, із застосуванням ЯЗ розмірами від 2×2 до 15×15. Як видно з (табл. 1), запропоноване SIMD-рішення демонструє перевагу до 11,1 раза на Orange Pi 5 Pro та до 10,2 раза на Raspberry Pi 4 за швидкодією порівняно з реалізацією OpenCV за рівних умов, що свідчить про платформну незалежність. Також це підтверджує доцільність використання низькорівневої оптимізації на рівні векторних регістрів.

Особливий інтерес становить залежність швидкодії від розміру ЯЗ. Як показано в (табл. 2 та табл. 3), які узагальнюють результати для всіх протестованих ЯЗ на зображенні розміром 1920×1080. Падіння FPS для OpenCV носить експоненційний характер, тоді як для запропонованого алгоритму спостерігається більш пологий спад, що свідчить про високу стійкість до збільшення обчислювального навантаження. В (табл. 1) та (табл. 2) фіксується середня швидкодія при обробці відеокадрів різної роздільної здатності згорткою розміром 9×9.

Таблиця 4 – Прискорення ШОЗ відносно cv::filter2D.

Функції\Платформа	Діапазон розмірів ЯЗ	Прискорення (Безрозмірний показник)	
		Orange Pi 5 Pro	Raspberry Pi 4
ШОЗ VS cv::filter2D (табл. 1)	2×2–7×7	0,8–2,5 раза	0,9–2,4 раза
	8×8–11×11	3,1–11,1 раза	2,7–10,2 раза
	12×12–15×15	1,8–5,8 раза	1,7–5,2 раза
Python-додаток: ШОЗ VS cv::filter2D (табл. 2)	2×2–7×7	1,0–2,2 раза	1,1–2,2 раза
	8×8–11×11	3,0–8,6 раза	2,8–7,4 раза
	12×12–15×15	1,7–5,1 раза	1,7–4,5 раза
Операція масштабування ЦЗ (Python-додаток) (табл. 3)	2×2–7×7	1,0–1,7 раза	1,0–1,9 раза
	8×8–11×11	2,5–6,3 раза	2,4–6,2 раза
	12×12–15×15	1,6–4,4 раза	1,6–4,1 раза

Отримані дані демонструють перевагу запропонованої ІТ, що проявляється при зростанні роздільної здатності зображення, коли використання SIMD-інструкцій дає змогу істотно зменшити час обробки одного кадру/ЦЗ в умовах використання Python-інтерфейсу. Спостерігається стабільне збереження переваги ЯІТ на всіх конфігураціях та платформах.

У результаті дослідження встановлено, що запропонована ІТ демонструє значне прискорення виконання ОЗ порівняно з функцією cv::filter2D, досягаючи показників, продемонстрованих у (табл. 4). Загалом, експериментальні дослідження підтверджують, що запропонована ІТ забезпечує високу швидкодію, точність та портованість – що робить її придатною для практичного використання у мобільних, розподілених та автономних системах.

ВИСНОВКИ

У роботі вирішено науково-технічну задачу розробки ІТ обробки ЦЗ та відео з використанням ОЗ для ЦП архітектури ARM64 для підвищення швидкодії на сучасних одноплатних вбудованих системах (SBC). За результатами дисертаційних досліджень отримано такі загальні висновки:

1. Проведений аналіз існуючих програмно–апаратних та архітектурно-інформаційних засобів для забезпечення, підвищення та реалізації швидкодіючих обчислень, для ОЗ ЦЗ та ПЗ в цілому дозволив визначити потенційні напрями та підходи вдосконалення швидкодії для цілочисельних обчислень;

2. Отримано лінійні оператори (3–8) для ШОЗ, котрі ефективно використовують 16-бітні обчислення, при обробці ЦЗ, а саме: субсмугова фільтрація;

контрастування; цифрова стабілізація зображень; рекурентне двократне збільшення масштабу;

3. Запропоновано математичну модель ОЗ з використанням 16-бітних цілочисельних операцій для 8-бітних RGB ЦЗ та 8-бітних ЯЗ;

4. Запропоновано формалізовану нотацію для опису векторизованих обчислень з використанням SIMD NEON64, яка дозволяє абстрагуватися від асемблерних інструкцій та представляти векторні SIMD операції у вигляді математичних/структурованих виразів, що сприяє кращій читабельності та знижує поріг входу, при засвоєнні технік експертної векторизації;

5. Розроблено та теоретично обґрунтовано метод ШОЗ для обробки 8-бітних RGB ЦЗ з використанням 16-бітних цілочисельних операцій, який забезпечує підвищення швидкодії на ЦП архітектури ARM64 порівняно з аналогічним підходом на основі функції cv::filter2D(…) бібліотеки OpenCV:

• для ядер згортки середнього розміру (8×8–11×11) досягнуто прискорення в діапазоні 3,1–11,1 раза для Orange Pi 5 Pro та 2,7–10,2 раза для Raspberry Pi 4;

• для великих ядер згортки (12×12–15×15), де бібліотека OpenCV може використовувати FFT-алгоритми, запропонований метод демонструє прискорення 1,8–5,8 раза для Orange Pi 5 Pro та 1,7–5,2 раза для Raspberry Pi 4;

• для операцій масштабування зображень, що включають чотирикратне застосування операції згортки, розроблений підхід (zoomUP_2__NEON64) показує прискорення до 6,3 раза (Orange Pi 5 Pro) та до 6,2 раза (Raspberry Pi 4) для ядер середнього розміру порівняно з аналогічною операцією на базі cv::filter2D.

6. Розроблено та реалізовано алгоритм ШОЗ, проведено оптимізацію цього алгоритму використовуючи технологію NEON64 та експериментальні дослідження швидкодії розробленого алгоритму на різних платформах SBC (Raspberry Pi 4 та Orange Pi 5 Pro) для різних розмірів ЦЗ (від 256×144 до 3840×2160) та розмірів ЯЗ (від 2×2 до 15×15);

7. Розроблена ІТ (основою якої є ШОЗ) дозволяє в режимі реального часу проводити фільтрацію та масштабування потокового відео та ЦЗ на одноплатних комп'ютерах з ЦП ARM64, як–от: Raspberry Pi 4 та Orange Pi 5 Pro;

8. Запропонована методика експериментальної оцінки та порівняння швидкодії різних програмних реалізацій операції згортки на платформах з архітектурою ЦП ARM64, дозволила визначати переваги запропонованих алгоритмів на основі ШОЗ;

9. Проведено порівняльний аналіз швидкодії розробленого алгоритму з існуючими рішеннями, зокрема з функцією cv::filter2D бібліотеки OpenCV. У результаті дослідження встановлено, що запропонована інформаційна технологія демонструє значне прискорення виконання операції згортки порівняно з функцією cv::filter2D (бібліотеки OpenCV), досягаючи:

• максимального прискорення для ядер розміром 8–11 у 3,1–11,1 раза (Orange Pi 5 Pro) та 2,7–10,2 раза (Raspberry Pi 4);

• помірного для ядер розміром 2–7 у 0,8–2,5 раза (Orange Pi 5 Pro) та 0,9–2,4 раза (Raspberry Pi 4) з урахуванням витрат вводу-виводу вимірювального конвеєра; за прямим вимірюванням самої операції згортки – у 1,3–2,8 раза;

• стабільно високого для ядер розміром 12–15 у 1,8–5,8 раза (Orange Pi 5 Pro) та 1,7–5,2 раза (Raspberry Pi 4).

Розроблена технологія потенційно може мати кращу енергоефективність порівняно з реалізаціями на основі операцій з плаваючою комою завдяки використанню цілочисельних обчислень та скороченню часу виконання операцій. Дослідження енергоспоживання розробленої ІТ на мобільних та вбудованих

пристроях є перспективним напрямом подальших досліджень. Особливо значущі переваги ІТ проявилися у сценаріях реального часу: наприклад, при обробці відеопотоку з частотою 25 FPS на платформі Orange Pi 5 Pro було досягнуто стабільної швидкодії 26–29 FPS для операції згортки для RGB ЦЗ.

Запропоновані алгоритми швидкодіючої операції згортки можуть бути застосовані для прискорення ОЗ в згорткових нейронних мережах на мобільних та вбудованих пристроях, що є перспективним напрямом для розвитку мобільних додатків з елементами штучного інтелекту та систем постпроцесингу на платформах ARM64.

Матеріали дисертаційного дослідження використані для розробки навчальних курсів та лабораторних робіт з дисциплін: «Архітектура обчислювальних систем та операційні системи» для студентів спеціальності 113 «Прикладна математика»; «Системне програмування» для студентів спеціальності 113 «Прикладна математика»; «Програмування мовою асемблер» для студентів спеціальності 113 «Прикладна математика».

Результати дисертаційного дослідження також впроваджено у виробничу діяльність підприємств ТОВ «ЕЛІСАТ» (вбудоване програмне забезпечення пристроїв збору й обробки даних з датчиків) та ТОВ «ІНСПІРІТО ТІМ» (програмно-апаратні засоби обробки відеоданих з бортових камер БПЛА).

ПУБЛІКАЦІЇ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

1. Приставка П. О., Шевченко А. К. Дослідження реалізації лінійного оператора згортки цифрового зображення при 16-бітних обчисленнях. Актуальні проблеми автоматизації та інформаційних технологій. 2016. Т. 20 : 1. С. 78–90. URL: https://doi.org/10.15421/431608.

2. Шевченко А. К. Вибір програмно-апаратних засобів для реалізації операції згортки з підвищеною швидкістю. Наукоємні технології. 2018. Т. 37 : 1. С. 49–54. URL: https://doi.org/10.18372/2310-5461.37.12369.

3. Shevchenko A., Tymchyshyn V. A SIMD-based approach to the enhancement of convolution operation performance. CEUR Workshop Proceedings. 2019. Vol. 2588. P. 447–458. URL: https://ceur-ws.org/Vol-2588/paper37.pdf

4. Shevchenko A., Prystavka P., Tymchyshyn V. Research on possible convolution operation speed enhancement via AArch64 SIMD. Advances in Computer Science for Engineering and Education. Lecture Notes on Data Engineering and Communications Technologies. Cham : Springer, 2022. Vol. 134. P. 61–75. URL: https://doi.org/10.1007/978-3-031-04812-8_6.

5. Prystavka P., Shevchenko A., Rokitianska I. Comparative analysis of detector-tracker architecture for object tracking based on SBC for UAV. 2024 IEEE 7th International Conference on Actual Problems of Unmanned Aerial Vehicles Development (APUAVD), Kyiv, Ukraine, 22–24 October 2024. 2024. P. 175–178. URL: https://doi.org/10.1109/APUAVD64488.2024.10765890 (date of access: 29.07.2025).

6. Shevchenko A., Prystavka P. Enhancement of convolution operation performance using SIMD of AArch64 (short paper). CEUR Workshop Proceedings. 2024. Vol. 3826. P. 385–391. URL: https://ceur-ws.org/Vol-3826/short32.pdf (date of access: 26.05.2026).

7. Шевченко А. К. Підвищення швидкодії операції згортки RGB зображень з використанням алгоритму кластеризації ядер згортки для архітектури процесорів ARM64. Problems of Informatization and Management. 2025. Т. 1 : 81. С. 140–

149. URL: https://doi.org/10.18372/2073-4751.81.20144 (дата звернення: 25.05.2026).

8. Приставка П. О., Шевченко А. К. Особливості застосування технології ARM NEON при обробці зображень. Політ. Сучасні проблеми науки : XVI Міжнар. наук.-практ. конф. здобувачів вищ. освіти і молодих учених, м. Київ, 25 берез. 2016 р. НАУ, 2016. С. 135.

9. Шевченко А. К., Приставка П. О. Оптимізація обробки цифрового зображення з використанням технології ARM NEON. АВІА-2017 : XIII Міжнар. наук.-техн. конф., м. Київ, 19–21 квіт. 2017 р. НАУ, 2017. С. 2.27–2.29.

10. Prystavka P. O., Shevchenko A. K. A brief overview of modern software optimization approaches. Авіація у XXI столітті – Безпека в авіації та космічні технології : VIII Всесвіт. конгр., м. Київ, 10–12 жовт. 2018 р. НАУ, 2018. С. 2.3.22–2.3.24.

11. Шевченко А. К. Огляд програмних засобів для підвищення швидкодії операції згортки. Політ. Сучасні проблеми науки : XIX Міжнар. наук.-практ. конф. здобувачів вищ. освіти і молодих учених, м. Київ, 1–5 квіт. 2019 р. НАУ, 2019. С. 35–37.

12. Shevchenko A. K. Cache-friendly code for convolution operation. Політ. Сучасні проблеми науки : XXI Міжнар. наук.-практ. конф. здобувачів вищ. освіти і молодих учених, м. Київ, 5–9 квіт. 2021 р. НАУ, 2021. С. 18–19.

13. Приставка П. О., Шевченко А. К. Дослідження швидкодії операції згортки що було оптимізовано за допомогою SIMD NEON ARM64 в умовах обробки відео з файлу. АВІА-2023 : XVI Міжнар. наук.-техн. конф., м. Київ, 18–20 квіт. 2023 р. НАУ, 2023. С. 22–24.

14. Шевченко А. К., Приставка П. О. Дослідження можливостей підвищення швидкодії операції згортки за допомогою кластеризації розріджених ядер матриць згортки на архітектурі AArch64. Політ. Сучасні проблеми науки : XXIII Міжнар. наук.-практ. конф. здобувачів вищ. освіти і молодих учених, м. Київ, 4–7 квіт. 2023 р. НАУ, 2023. С. 18–21.

АНОТАЦІЯ

Шевченко А. К. Інформаційна технологія обробки графічних даних для SIMD-архітектур процесорів. – Рукопис.

Дисертація на здобуття наукового ступеня кандидата технічних наук за спеціальністю 05.13.06 «Інформаційні технології». – Національний університет «Київський авіаційний інститут», Київ, 2026.

Дисертаційну роботу присвячено розробці інформаційної технології (ІТ) підвищення швидкодії обробки 8-бітних цифрових RGB-зображень на системах з архітектурою ARM64. Запропонована ІТ базується на використанні оптимізованої векторизованої операції згортки із застосуванням 16-бітних цілочисельних обчислень та SIMD-інструкцій NEON для ЦП архітектури ARM64. Актуальність роботи зумовлена поширенням ARM64-платформ у вбудованих системах та мобільних пристроях, зокрема в безпілотних літальних апаратах, що потребують високої швидкодії обробки зображень в умовах обмежених ресурсів та вимог реального часу.

У роботі вперше розроблено та теоретично обґрунтовано метод векторизованої операції згортки, що базується на 16-бітних цілочисельних операціях. Запропоновано формалізований підхід до побудови векторизованих згорткових операторів, розроблено обчислювальне ядро conv_RGB_NEON та реалізовано

модульну ІТ у вигляді С-бібліотеки з Python-інтерфейсом. Проведено оптимізацію обчислень із врахуванням кеш-ефективних структур коду, обробки розріджених ядер згортки та розроблено нотацію для опису SIMD NEON64 операцій, яка є основою опису векторизованої ОЗ.

Експериментальні дослідження на одноплатних ARM64-комп'ютерах (Raspberry Pi 4, Orange Pi 5 Pro) показали значне прискорення виконання операції згортки (до 11,1 раза на Orange Pi 5 Pro та до 10,2 раза на Raspberry Pi 4, зокрема до 2,5 раза для малих ядер 2×2–7×7) порівняно з функцією cv::filter2D бібліотеки OpenCV. При цьому точність обробки залишається високою, з максимальною абсолютною похибкою не більше 1 (одиниці) на кожен колірний канал 8-бітного RGB цифрового зображення, що візуально не впливає на якість зображення. Розроблена ІТ адаптована для обробки відео в реальному часі (досягнуто стабільної швидкодії 26–29 кадрів/секунду при обробці відеопотоку з дрона на Orange Pi 5 Pro).

Практична значущість роботи полягає у створенні готового програмного інструментарію для розробників систем, що потребують високопродуктивної обробки зображень на платформах ARM64, зокрема в системах бортової обробки даних БПЛА, мобільних додатках, системах відеоспостереження та edge-computing. Результати дослідження впроваджено: при виконанні науково-дослідних робіт подвійного призначення НДР №1062-ДБ16, №247-ДБ19 та №479-ДБ24 на базі Національного університету «Київський авіаційний інститут»; у навчальному процесі Національного університету «Київський авіаційний інститут» при викладанні дисциплін «Архітектура обчислювальних систем та операційні системи», «Системне програмування» та «Програмування мовою асемблер»; у виробничій діяльності ТОВ «ЕЛІСАТ» (вбудоване програмне забезпечення пристроїв збору й обробки даних з датчиків) та ТОВ «ІНСПІРІТО ТІМ» (програмно-апаратні засоби обробки відеоданих з бортових камер БПЛА).

Ключові слова: SIMD, ARM64, NEON64, операція згортки, 16-бітні обчислення, обробка цифрових зображень, векторизація, швидкодія, інформаційна технологія, реальний час, одноплатні комп'ютери, БПЛА.

ABSTRACT

Shevchenko A. K. Information technology of graphic data processing for processor SIMD architectures. – Manuscript.

Thesis for a Candidate of Technical Science degree in specialty 05.13.06 «Information technologies». – National University «Kyiv Aviation Institute», Kyiv, 2026.

The dissertation is dedicated to the development of an information technology (IT) for increasing the speed of processing 8-bit digital RGB images on systems with the ARM64 architecture. The proposed IT is based on the use of an optimized vectorized convolution operation employing 16-bit integer calculations and SIMD NEON instructions for ARM64 architecture CPUs. The relevance of the work is driven by the widespread use of ARM64 platforms in embedded systems and mobile devices, particularly in unmanned aerial vehicles, which require high-performance image processing under conditions of limited resources and real-time demands.

The purpose of the work is to solve the scientific and technical problem of developing an information technology for processing digital images and video by means of a high-speed convolution operation for ARM64 architecture CPUs, which enables real-time filtering and scaling of streaming video and digital images on single-board computers with limited computing resources.

The scientific novelty of the obtained results is as follows. For the first time, a method of the high-speed convolution operation for processing 8-bit RGB digital images using 16-bit integer operations has been developed and theoretically substantiated, which increases performance on ARM64 architecture CPUs compared with the analogous approach based on the cv::filter2D function of the OpenCV library. For the first time, a formalized notation for describing vectorized computations using SIMD NEON64 has been proposed, which allows the vectorized part of the code to be represented as structured expressions, thereby improving readability and lowering the entry threshold for mastering expert vectorization techniques. The technology of digital image processing on computers supporting computational parallelism has been improved by means of the proposed information technology of image and video processing based on the high-speed convolution operation for the ARM64 architecture, which enables real-time filtering and scaling of streaming video and digital images on single-board computers with ARM64 CPUs, such as Raspberry Pi 4 and Orange Pi 5 Pro. Linear operators based on linear combinations of B-splines that are close to interpolation in the mean have been further developed, namely: linear operators of 16-bit integer operations have been obtained that guarantee the absence of overflow in the 16-bit computation of the vectorized dot product, which allows their application to subband filtering, contrast enhancement, digital image stabilization and recurrent twofold enlargement of the linear dimensions of digital images.

A formalized approach to constructing vectorized convolution operators has been proposed, the conv_RGB_NEON computational core has been developed, and a modular IT has been implemented as a C-library with a Python interface. Optimization of computations has been carried out considering cache-efficient code structures and the processing of sparse convolution kernels.

Experimental studies on single-board ARM64 computers (Raspberry Pi 4, Orange Pi 5 Pro) showed a significant acceleration of the convolution operation (up to 11.1 times on Orange Pi 5 Pro and up to 10.2 times on Raspberry Pi 4, particularly up to 2.5 times for small 2×2–7×7 kernels) compared to the cv::filter2D function of the OpenCV library. The processing accuracy remains high, with a maximum absolute error not exceeding 1 (unit) per color channel of an 8-bit RGB digital image, which does not visually affect image quality. The developed IT is adapted for real-time video processing (stable performance of 26–29 frames/second achieved when processing video stream from a drone on Orange Pi 5 Pro).

The practical significance of the work lies in the creation of a ready-to-use software toolkit for developers of systems requiring high-performance image processing on ARM64 platforms, particularly in UAV onboard data processing systems, mobile applications, video surveillance systems, and edge-computing. The research results have been implemented: within dual-purpose R&D projects No. 1062-DB16, No. 247-DB19, and No. 479-DB24 at the National University “Kyiv Aviation Institute”; in the educational process of the National University “Kyiv Aviation Institute” in teaching the courses “Computer Systems Architecture and Operating Systems”, “System Programming”, and “Assembly Language Programming”; and in the operational activities of ELISATLLC (embedded software for sensor data acquisition and processing devices) and INSPIRITO TIM LLC (hardware and software systems for processing video data from UAV onboard cameras).

Keywords**:** SIMD, ARM64, NEON64, convolution operation, 16-bit computing, digital image processing, vectorization, high performance, information technology, real-time, single-board computers, UAV.