ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Система комп'ютерного зору для мікроконтролерних платформ STM32»

на здобуття освітнього ступеня бакалавра
зі спеціальності 123 Комп'ютерної інженерії
(код, найменування спеціальності)
освітньо-професійної програми Комп'ютерна інженерія
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ *Олег КЛОБУКОВ*
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувач вищої освіти гр. КІД-42
Олег КЛОБУКОВ
Ім'я, ПРІЗВИЩЕ
Керівник: Артем АНТОНЕНКО
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ
Рецензент: _____
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ

Київ 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра 123 Комп'ютерної інженерії

Ступінь вищої освіти бакалавр

Спеціальність 123 Комп'ютерної інженерії

Освітньо-професійна програма Комп'ютерна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедрою Наталія ЛАЩЕВСЬКА
Ім'я, ПРІЗВИЩЕ

«_____» _____ 2026р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Клобуков Олег Віталійович
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: Система комп'ютерного зору для мікроконтролерних платформ STM32
керівник кваліфікаційної роботи Артем АНТОНЕНКО,
(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «24» лютого 2026р. № 56
2. Строк подання кваліфікаційної роботи «5» червня 2026р.
3. Вихідні дані до кваліфікаційної роботи: науково-технічна література та офіційна документація вендора (Datasheets, Reference Manuals STMicroelectronics); архітектура мікроконтролерів серії STM32 (ядра ARM Cortex-M4/M7); апаратні інтерфейси захоплення відеопотоку (DCMI) та прямого доступу до пам'яті (DMA); оптичні сенсори (формат VGA/QVGA); технології граничних обчислень (Edge AI); інструменти конвертації та розгортання моделей (X-CUBE-AI, TensorFlow Lite for Microcontrollers); методи квантування нейромережевих моделей (формат INT8); інтегроване середовище розробки STM32CubeIDE; мови програмування C/C++.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
   4.1. Аналіз систем комп'ютерного зору та мікроконтролерних платформ.
   4.2. Проектування системи комп'ютерного зору на базі STM32.
   4.3. Реалізація системи комп'ютерного зору.
   4.4. Експериментальне дослідження системи.
5. Перелік ілюстративного матеріалу: презентація
6. Дата видачі завдання «24» лютого 2026р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Збір літератури, вивчення документації STMicroelectronics та обґрунтування ТЗ	25.02-24.03.2026	Виконано
2	Написання Розділу 1 (аналіз методів та платформ)	23.03-05.04.2026	Виконано
3	Проектування апаратної архітектури та написання Розділу 2	06.04-10.04.2026	Виконано
4	Налаштування STM32CubeIDE, інтеграція камери через DCMI/DMA	11.04-20.04.2026	Виконано
5	Квантування моделі (X-CUBE-AI) та написання Розділу 3	21.04-06.05.2026	Виконано
6	Тестування системи (FPS, RAM/Flash) та написання Розділу 3	07.05-08.05.2026	Виконано
7	Оформлення пояснювальної записки згідно з вимогами нормоконтролю	09.05-11.05.2026	Виконано
8	Підготовка презентації, перевірка на плагіат та передзахист	27.05-29.05.2026	Виконано
9	Здача зброшурованої роботи на кафедру	02.06-05.06.2026	Виконано

Здобувач вищої освіти _____ Олег КЛОБУКОВ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник кваліфікаційної роботи _____ Артем АНТОНЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: **86** стор., **18** рис., **33** табл., **50** джерел, **5** додатків.

Мета роботи – створення робочого прототипу апаратно-програмної системи автоматичного розпізнавання об'єктів на базі мікроконтролера STM32 з використанням оптимізованих алгоритмів комп'ютерного зору (Edge AI) для роботи в умовах обмежених обчислювальних ресурсів.

Об'єкт дослідження – процес автоматичного розпізнавання об'єктів на зображеннях в умовах жорстко обмежених обчислювальних та енергетичних ресурсів.

Предмет дослідження – апаратно-програмні засоби та алгоритми комп'ютерного зору (Edge AI) для вбудованих систем на базі мікроконтролера сімейства STM32.

Методи дослідження – методи системного аналізу для проєктування архітектури; методи цифрової обробки зображень та машинного навчання для виявлення та класифікації об'єктів; емпіричні методи для вимірювання продуктивності апаратного прототипу. Для досягнення мети вирішено такі завдання: проаналізовано архітектури CNN та апаратні платформи для Edge AI; обґрунтовано вибір STM32H743ZIT6 та сенсора OV2640; реалізовано конвеєр обробки зображень із SIMD-оптимізацією; виконано INT8-квантизацію CNN-моделі; проведено експериментальне тестування продуктивності та точності.

Короткий зміст роботи: У кваліфікаційній роботі досліджено проблему впровадження алгоритмів комп'ютерного зору на кінцевих пристроях (Edge Computing). Проведено аналіз існуючих рішень та апаратних обмежень архітектури ARM Cortex-M. Обґрунтовано вибір мікроконтролера STM32H743ZIT6 та оптичного сенсора OV2640. Спроєктовано структурну та функціональну схеми

системи. Реалізовано конвеєр обробки зображень із класичними алгоритмами (гаусівська фільтрація, бінаризація методом Оцу, оператор Собеля) та квантизованою нейронною мережею через бібліотеку CMSIS-NN. Проведено експериментальне тестування: визначено показники FPS = 16,8 кадрів/с (QVGA), точності розпізнавання (89,4 %), використання RAM (66 %) та Flash (52 %) ресурсів.

КЛЮЧОВІ СЛОВА: КОМП'ЮТЕРНИЙ ЗІР, МІКРОКОНТРОЛЕР, STM32, EDGE AI, РОЗПІЗНАВАННЯ ОБ'ЄКТІВ, ОБРОБКА ЗОБРАЖЕНЬ, CMSIS-NN, ОПТИМІЗАЦІЯ, DCMI, АПАРАТНИЙ ПРОТОТИП.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 10

ВСТУП 13

1 АНАЛІЗ СИСТЕМ КОМП'ЮТЕРНОГО ЗОРУ ТА МІКРОКОНТРОЛЕРНИХ ПЛАТФОРМ 18

1.1 Основні принципи комп'ютерного зору 18

1.2 Архітектури систем комп'ютерного зору 21

1.3 Мікроконтролерні платформи для обробки зображень 24

1.4 Аналіз можливостей STM32 для задач комп'ютерного зору 25

1.5 Колірні моделі та формати представлення зображень 27

1.6 Теоретичні основи цифрової обробки зображень 28

1.7 Архітектурні особливості DSP-інструкцій Cortex-M7 30

1.8 Математичний апарат згорткових нейронних мереж 31

Висновки до розділу 1 34

2 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ СИСТЕМИ КОМП'ЮТЕРНОГО ЗОРУ НА БАЗІ STM32 35

2.1 Постановка задачі та вимоги до системи 35

2.2 Вибір апаратної платформи 36

2.3 Архітектура системи комп'ютерного зору 37

2.4 Алгоритм обробки зображення та часовий бюджет 40

2.5 Енергетичний бюджет та розрахунок автономності 42

2.6 Теплове проєктування системи 43

2.7 Розробка програмного забезпечення (STM32CubeIDE, HAL/CMSIS) 45

2.8 Інтеграція камери з мікроконтролером (DCMI, DMA) 46

2.9 Реалізація алгоритмів обробки зображень 48

2.10 Оптимізація алгоритмів для embedded систем 49

2.11 Проєктування нейромережевої моделі для Edge AI 51

2.12 Методологія налагодження та тестування 54

2.13 Детальна реалізація потоку даних від камери до CNN 55

Висновки до розділу 2 59

3 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА ОПТИМІЗАЦІЯ СИСТЕМИ 60

3.1 Методика проведення експерименту 61

3.2 Тестування продуктивності системи 62

3.3 Аналіз результатів 64

3.4 Порівняння з іншими рішеннями 68

3.5 Аналіз похибок та граничні випадки 69

3.6 Дослідження впливу роздільної здатності на характеристики 70

3.7 Контрольні порівняння на стандартному наборі даних 73

3.8 Детальний аналіз точності по класах 74

3.9 Алгоритм визначення надійності прогнозу 75

3.10 Публікації та апробація 76

Висновки до розділу 3 77

ВИСНОВКИ 79

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 82

ДОДАТОК А. Принципова схема підключення OV2640 до STM32 87

ДОДАТОК Б. Лістинг програмного коду ключових модулів 89

ДОДАТОК В. Результати тестування системи 91

ДОДАТОК Г. Специфікація компонентів (Bill of Materials, BOM) 94

ДОДАТОК Ґ. Додаткові лістинги програмного коду 96

ДОДАТОК Д. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація) 99

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

Українські скорочення:

АЦП – Аналого-цифровий перетворювач

ГБ – Гігабайт

ДУ ІКТ – Державний університет інформаційно-комунікаційних технологій

ІІРТК – Інтегровані інтелектуальні робототехнічні комплекси

ККД – Коефіцієнт корисної дії

МБ – Мегабайт

МК – Мікроконтролер

ОЗП – Оперативний запам'ятовуючий пристрій

ОС – Операційна система

ПЗ – Програмне забезпечення

ПК – Персональний комп'ютер

ТЗ – Технічне завдання

ЦАП – Цифро-аналоговий перетворювач

Латинські скорочення:

AI – Artificial Intelligence (Штучний інтелект)

ARM – Advanced RISC Machines

AXI – Advanced eXtensible Interface

CIFAR – Canadian Institute For Advanced Research

CLAHE – Contrast Limited Adaptive Histogram Equalization

CMSIS – Cortex Microcontroller Software Interface Standard

CMOS – Complementary Metal-Oxide-Semiconductor

CNN – Convolutional Neural Network (Згорткова нейронна мережа)

CPU – Central Processing Unit (Центральний процесор)

CV – Computer Vision (Комп'ютерний зір)

DCMI – Digital Camera Memory Interface

DMA – Direct Memory Access (Прямий доступ до пам'яті)

DSP – Digital Signal Processing (Цифрова обробка сигналів)

DTCM – Data Tightly Coupled Memory

FPS – Frames Per Second (Кадрів на секунду)

FPU – Floating Point Unit (Блок обчислень з плаваючою комою)

GPIO – General Purpose Input/Output

GPU – Graphics Processing Unit (Графічний процесор)

HAL – Hardware Abstraction Layer

HSYNC – Horizontal Synchronization

INT8 – 8-bit Integer quantization format

ITCM – Instruction Tightly Coupled Memory

JPEG – Joint Photographic Experts Group

LED – Light Emitting Diode (Світлодіод)

MAC – Multiply-Accumulate operation

MCU – Microcontroller Unit

MMAC – Million Multiply-Accumulate operations

NN – Neural Network (Нейронна мережа)

PCB – Printed Circuit Board (Друкована плата)

PIXCLK – Pixel Clock

PWDN – Power Down

QVGA – Quarter Video Graphics Array

RAM – Random Access Memory

RGB – Red, Green, Blue

RESET – Hardware Reset signal

SCCB – Serial Camera Control Bus

SIMD – Single Instruction, Multiple Data

SMLAD – Signed Multiply Accumulate Dual

SMUAD – Signed Multiply Unsigned Accumulate Dual

SRAM – Static Random Access Memory

USB – Universal Serial Bus

USART – Universal Synchronous/Asynchronous Receiver/Transmitter

VGA – Video Graphics Array

VSYNC – Vertical Synchronization

WFI – Wait For Interrupt

XCLK – External Clock

YUV – Luminance-Chrominance color space (колірний простір)

ВСТУП

Сучасний розвиток інформаційних технологій, робототехніки та систем Інтернету речей (IoT) нерозривно пов'язаний із впровадженням методів комп'ютерного зору. Донедавна завдання аналізу візуальної інформації, зокрема розпізнавання та класифікація об'єктів, вирішувалися переважно за допомогою ресурсоємних згорткових нейронних мереж, які потребували високопродуктивних графічних процесорів (GPU) або залучення хмарних обчислювальних кластерів.

Традиційна парадигма, що базується на передачі потокового відео на віддалені сервери, виявила низку критичних обмежень при практичному інженерному впровадженні в автономні пристрої. По-перше, передача великих масивів візуальних даних створює суттєве навантаження на пропускну здатність каналів зв'язку та викликає непередбачувані мережеві затримки (latency). По-друге, постійна робота радіочастотних модулів призводить до швидкого виснаження елементів живлення. По-третє, передача відеоданих у хмару створює потенційні ризики для конфіденційності інформації.

Вирішенням цих інженерних викликів є перехід до парадигми граничних обчислень (Edge AI) — виконання алгоритмів машинного навчання безпосередньо на кінцевих пристроях (on-device inference). Оптимальним інженерним компромісом між обчислювальною продуктивністю та енергоефективністю є використання мікроконтролерів родини STM32 на базі архітектури ARM Cortex-M. Наявність апаратних блоків обчислень з плаваючою комою (FPU), контролерів прямого доступу до пам'яті (DMA) та паралельних інтерфейсів захоплення відео (DCMI) дозволяє ефективно реалізувати конвеєр обробки зображень на апаратному рівні.

Актуальність теми. Розробка та інженерна оптимізація апаратно-програмної системи розпізнавання об'єктів, здатної автономно функціонувати на базі мікроконтролера STM32, є надзвичайно актуальним практичним завданням для застосування в IoT-пристроях, промисловій автоматизації та портативних системах технічного зору.

Аналіз останніх досліджень і публікацій. Питанням впровадження CNN на ресурсообмежених пристроях присвячено роботи Sze V. та співавт. [30], Howard A. та співавт. [6, 36] (серія MobileNet), Sandler M. [22] (MobileNetV2), Iandola F. [7] (SqueezeNet), Tan M. та Le Q.V. [37] (EfficientNet), Lin J. та співавт. [33] (MCUNet — TinyML на STM32). Питання INT8-квантизації досліджували Jacob B. [21], Krishnamoorthi R. [35]. Стандартизацію бенчмарків TinyML системно описано у роботі Banbury C. та співавт. [34]. Незважаючи на значний обсяг публікацій, відкритим залишається питання комплексної інженерної інтеграції апаратної та програмної частин системи комп’ютерного зору з обмеженою собівартістю (до 30 USD) на платформі STM32H7.

Мета дослідження полягає у створенні робочого прототипу апаратно-програмної системи автоматичного розпізнавання об'єктів на базі мікроконтролера STM32 для її подальшого використання в автономних вбудованих пристроях з жорстко обмеженими ресурсами.

Завдання дослідження:

– провести аналіз існуючих інженерних рішень комп'ютерного зору та архітектур легковагових нейронних мереж, адаптованих для вбудованих систем (Edge AI);

– обґрунтувати вибір апаратної бази (серії мікроконтролера STM32 та оптичного модуля камери) і спроєктувати структурну схему пристрою;

– адаптувати та оптимізувати обрану модель машинного навчання шляхом застосування методів квантування для забезпечення її сумісності з ресурсами архітектури Cortex-M;

– реалізувати програмну інтеграцію оптимізованої моделі на мікроконтролері STM32, налаштувати низькорівневі інтерфейси захоплення відеопотоку (DCMI, DMA);

– провести експериментальне тестування продуктивності розробленого прототипу та оцінити точність розпізнавання в реальних умовах.

Об'єкт дослідження — процес автоматичного розпізнавання об'єктів на

зображеннях в умовах жорстко обмежених обчислювальних та енергетичних ресурсів.

Предмет дослідження — апаратно-програмні засоби та алгоритми комп'ютерного зору (Edge AI) для вбудованих систем на базі мікроконтролера сімейства STM32.

Методи дослідження — методи системного аналізу для проєктування архітектури системи; методи цифрової обробки зображень (фільтрація, бінаризація, виявлення границь) та машинного навчання (згорткові нейронні мережі, INT8-квантизація) для виявлення та класифікації об’єктів; емпіричні методи (вимірювання часу виконання, енергоспоживання, точності класифікації) для оцінки продуктивності апаратного прототипу.

Наукова новизна одержаних результатів полягає у:

— встановленні емпіричних залежностей FPS від роздільної здатності зображення для платформи STM32H743ZIT6 у задачі п’ятикласної класифікації, що дозволяє при проєктуванні аналогічних систем обґрунтовано обирати компроміс між роздільною здатністю та частотою кадрів без проведення повторних експериментів;

— обґрунтуванні методики комплексної оптимізації конвеєра обробки зображень на платформі STM32 шляхом поєднання SIMD-інструкцій DSP-розширення Cortex-M7 та INT8-квантизації нейромережевої моделі через X-CUBE-AI, що дозволило досягти прискорення у 4× при втраті точності лише 2,2 п.п.;

— удосконаленні методики розрахунку енергетичного бюджету системи з урахуванням режимів очікування (WFI) між кадрами, що забезпечує продовження часу автономної роботи до 22,5 годин від батареї 3000 мА·год;

— отриманні кількісних оцінок впливу умов освітлення (природне, штучне, низьке) на точність класифікації для розробленої моделі TinyCNN, що дає можливість прогнозувати продуктивність системи у реальних експлуатаційних сценаріях.

Практичне значення одержаних результатів. Розроблений функціональний прототип та оптимізоване вбудоване програмне забезпечення доводять інженерну

здійсненність автономного розпізнавання об’єктів на мікроконтролерах серії STM32 у режимі реального часу. Розроблена система може використовуватися у таких прикладних задачах: системах контролю доступу та моніторингу прохідних з виявленням людей; системах промислового контролю якості (виявлення браку на конвеєрах, перевірка геометрії деталей); компонентах розумного будинку (smart-камери з локальним обробленням без передачі відеопотоку у хмару); автономних рахувальниках транспорту для smart-city інфраструктури; портативних асистивних системах для людей з порушеннями зору. Низька собівартість прототипу (до 30 USD) та автономність до 22,5 годин від акумулятора 3000 мА·год роблять рішення доцільним для масового впровадження у IoT-інфраструктуру. Розроблений програмний код, схеми підключення та результати тестування можуть бути використані як методичний матеріал у дисциплінах «Вбудовані системи», «Цифрова обробка сигналів» та «Машинне навчання на ресурсообмежених пристроях».

Апробація результатів дослідження. Основні наукові й практичні результати роботи опубліковано у двох тезах доповідей наукових конференцій 2026 року: (1) Клобуков О.В., Антоненко А.В. Розробка системи комп’ютерного зору у реальному часі для автономної навігації мобільного робота з використанням Edge AI на мікроконтролері STM32H743 // Збірник тез доповідей XIX Міжнародної науково-практичної конференції «Інтегровані інтелектуальні робототехнічні комплекси» (ІІРТК-2026), Секція 1 «Інтегровані інтелектуальні робототехнічні комплекси». – Київ: ДУІКТ, 19–20 травня 2026 р.; (2) Клобуков О.В. Розробка системи комп’ютерного зору у реальному часі для автономної навігації мобільного робота з використанням Edge AI на мікроконтролері STM32H743 // Збірник тез доповідей IV Всеукраїнської науково-теоретичної конференції до Дня науки України «Держава і суспільство: сучасні виклики та пошук рішень». – Київ: Київський фаховий коледж туризму та готельного господарства, 13 травня 2026 р. Робота також подана на Всеукраїнський конкурс студентських наукових робіт з галузі «Інформаційні технології», 2026 (номінація «Вбудовані системи та IoT») .

Структура та обсяг роботи. Кваліфікаційна робота складається зі вступу,

трьох розділів, висновків, переліку використаних джерел (50 найменувань) та 5 додатків. Загальний обсяг роботи — 86 сторінок, з яких основна частина — 68 сторінок (без переліку джерел і додатків).

1 АНАЛІЗ СИСТЕМ КОМП'ЮТЕРНОГО ЗОРУ ТА МІКРОКОНТРОЛЕРНИХ ПЛАТФОРМ

Впровадження систем комп'ютерного зору у вбудовані пристрої є одним з ключових технологічних трендів сучасності. Зростаючий попит на інтелектуальні автономні системи в IoT, промисловій автоматизації та медичній діагностиці зумовлює необхідність перенесення обробки зображень безпосередньо на кінцеві пристрої з обмеженими ресурсами.

1.1 Основні принципи комп'ютерного зору

Комп'ютерний зір (англ. Computer Vision, CV) – це міждисциплінарна галузь штучного інтелекту та інформатики, що вивчає методи та алгоритми, які дозволяють комп'ютерним системам отримувати, аналізувати та інтерпретувати візуальну інформацію з навколишнього середовища [1]. Кінцевою метою комп'ютерного зору є наділення машин здатністю «бачити» та розуміти зображення так само, як це робить людський зір, або навіть ефективніше в специфічних задачах.

Людське око та зоровий кортекс головного мозку утворюють надзвичайно складну систему, яка здатна миттєво ідентифікувати об'єкти, визначати їх просторове розташування, розпізнавати рухи та інтерпретувати контекст сцени. Зоровий кортекс обробляє приблизно 10 Мбіт/с сенсорної інформації, використовуючи ієрархічну систему нейронів, що виявляють ознаки від простих контурів до цілісних сцен. Саме ця ієрархічна природа стала натхненням для згорткових нейронних мереж [2].

Формально задачу комп'ютерного зору визначають як: маючи вхідне зображення I(x,y) або відеопотік I(x,y,t), система будує опис або модель M, що несе семантичну інформацію про зображену сцену. Задача є зворотною до задачі комп'ютерної графіки: якщо графіка синтезує зображення з моделі, то CV відновлює модель із зображення.

Ієрархія задач комп'ютерного зору

Задачі CV поділяються на три рівні абстракції (рисунок 1.1).

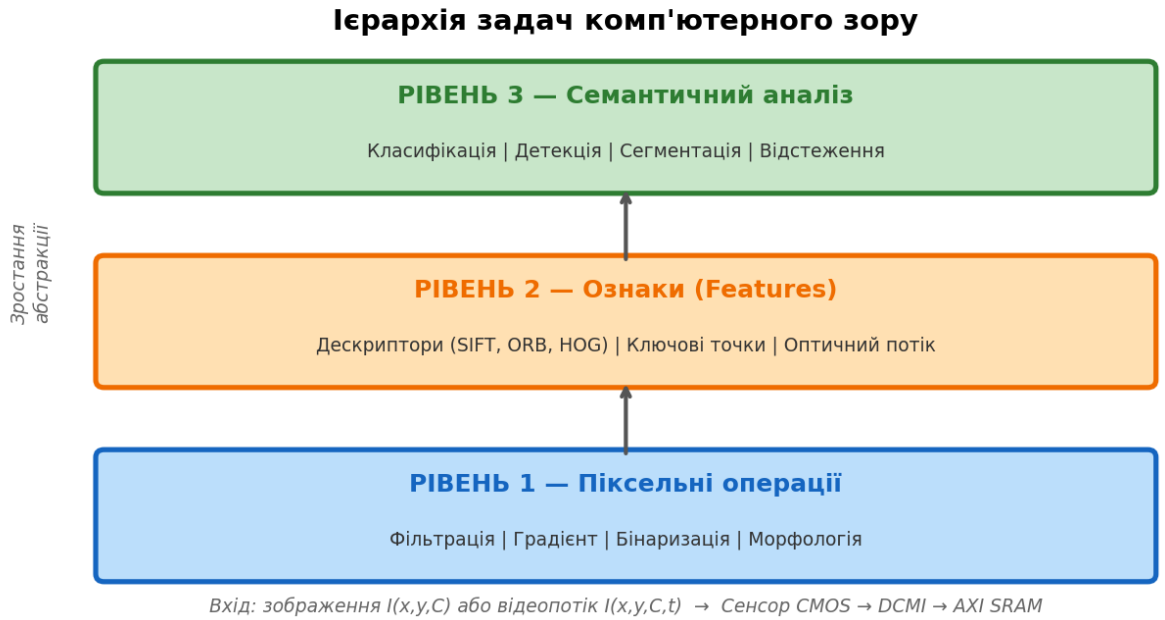


Рисунок 1.1 – Ієрархія рівнів задач комп'ютерного зору

Низькорівневі задачі (рівень 1) оперують безпосередньо пікселями:

– попередня обробка – шумозниження (гаусівська фільтрація, медіанний фільтр), корекція яскравості, CLAHE-нормалізація;

– виявлення границь – оператори Собеля, Лапласа, алгоритм Кенні [3];

– сегментація – методи порогування (Оцу [4]), k-means, watershed;

– виявлення ключових точок – SIFT, SURF, ORB, BRIEF.

Середньорівневі задачі (рівень 2) оперують дескрипторами ознак:

– зіставлення шаблонів (Template Matching) – нормалізована крос-кореляція;

– оцінка руху та оптичний потік – алгоритми Lucas-Kanade, Farneback;

– Hough-перетворення для виявлення прямих та кіл.

Семантичні задачі (рівень 3) вимагають розуміння контексту:

– класифікація зображень – метрики Top-1/Top-5 Accuracy;

– виявлення об'єктів (YOLO, SSD, Faster R-CNN) – метрика mAP@IoU;

– семантична сегментація (FCN, U-Net, DeepLab) – метрика mIoU;

– відстеження об'єктів (SORT, DeepSORT) – метрика MOTA;

– розпізнавання обличчя та жестів, оцінка пози.

Математична модель та метрики якості

Задача класифікації зображень формально:

$$f: \mathbb{R}^{H\times W\times C} \to \{1,\ 2,\ \dots,\ K\} \tag{1.1}$$

де *H, W, C* — висота, ширина та кількість каналів зображення; *K* — кількість класів. Для CNN-реалізації (суперпозиція L шарів):

$$y = f_L(f_{L-1}(\dots f_1(x;\ W_1)\dots;\ W_{L-1});\ W_L) \tag{1.2}$$

Дискретна двовимірна згортка — основна операція CNN:

$$(I * K)[i,j] = \sum_m \sum_n I\,[i+m,\ j+n]\cdot K[m,n] \tag{1.3}$$

Функція активації ReLU:

$$\mathrm{ReLU}(x) = \max(0,\ x) \tag{1.4}$$

Softmax для фінальної класифікації:

$$\mathrm{softmax}(z)_k = \frac{\exp(z_k)}{\sum_{j=1}^{K}\exp(z_j)} \tag{1.5}$$

Стандартні метрики якості. Accuracy (загальна точність):

$$\mathrm{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}\times 100\% \tag{1.6}$$

Precision та Recall:

$$\mathrm{Precision} = \frac{TP}{TP+FP}, \qquad \mathrm{Recall} = \frac{TP}{TP+FN} \tag{1.7}$$

F1-міра (гармонійне середнє):

$$F_1 = \frac{2\cdot\mathrm{Precision}\cdot\mathrm{Recall}}{\mathrm{Precision}+\mathrm{Recall}} \tag{1.8}$$

IoU (Intersection over Union) для задачі детекції:

$$\mathrm{IoU} = \frac{|A\cap B|}{|A\cup B|} \tag{1.9}$$

Таблиця 1.1 – Основні задачі та метрики якості систем комп'ютерного зору

Задача CV	**Вхідні дані**	**Результат**	**Метрика якості**
Класифікація	Зображення	Клас об'єкта	Top-1/Top-5 Accuracy

Детекція	Зображення	BB + клас + score	mAP@IoU=0,5
Сем. сегментація	Зображення	Піксельна маска	mIoU
Відстеження	Відеопотік	Траєкторії	MOTA, IDF1
Pose Estimation	Зображення	Координати суглобів	PCKh@0,5

Галузі застосування

Промислова автоматизація. Виявлення дефектів поверхні, перевірка геометрії деталей, контроль складання. Точність систем CV перевищує можливості людського контролера при тривалій монотонній роботі [5]. Впровадження знижує відсоток браку на 30–50%.

Автономні транспортні засоби. Виявлення дорожніх знаків, розпізнавання пішоходів, stereo-vision, visual SLAM. Вимога до затримки: ≤50 мс (20 fps). Типовий алгоритм обробки: 5–20 GFLOPS.

Медична діагностика. Аналіз КТ/МРТ/рентгенів. DNN демонструють AUC = 0,994 при виявленні раку легень (Stanford Medicine Study).

Безпека та відеоспостереження. Розпізнавання осіб, аналіз поведінки, моніторинг транспорту. Сучасні системи обробляють до 32 потоків одночасно.

AgriTech. NDVI-аналіз полів за дронами, виявлення хвороб рослин, автоматизоване сортування плодів.

Зростаючі вимоги до автономної роботи, мінімальної затримки та конфіденційності даних обумовлюють перехід до парадигми Edge AI — виконання інференсу безпосередньо на кінцевому пристрої.

1.2 Архітектури систем комп'ютерного зору

Класичні алгоритми

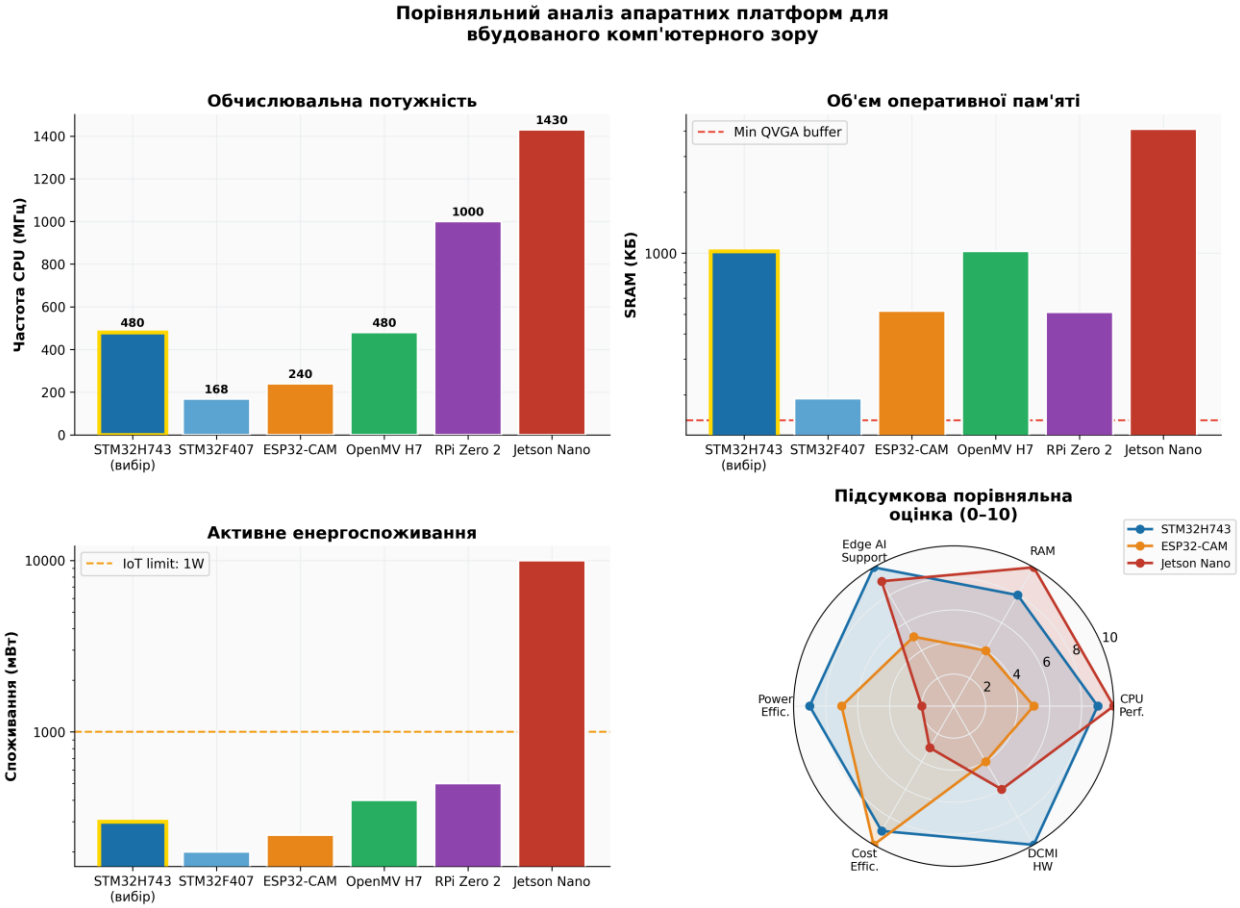
Класичні методи CV ґрунтуються на детермінованих операціях над пікселями. Алгоритм Кенні [3] — багатоетапний детектор границь: гаусівська фільтрація → градієнт Собеля → пригнічення немаксимумів → подвійне порогування. Гаусівське ядро 3×3:

$$G = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \qquad (1.10)$$

Метод автоматичного порогування Оцу [4] максимізує міжкласову дисперсію:

$$\sigma_B^2(T) = \omega_0(T)\cdot\omega_1(T)\cdot[\mu_0(T)-\mu_1(T)]^2, \qquad T^* = \underset{T}{\operatorname{argmax}}\,\sigma_B^2(T) \qquad (1.11)$$

Морфологічні операції ерозії та дилатації:



$$(I \oplus B)(x,y) = \max_{(m,n)\in B} I(x+m,\, y+n) \qquad (1.12)$$

Нейромережеві архітектури та їх порівняння

Революцію у CV здійснили CNN, що самостійно виявляють ієрархічні ознаки. Ключовим архітектурним проривом стало впровадження залишкових з'єднань (residual connections) у мережі ResNet [32], що дозволило тренувати глибокі моделі (понад 100 шарів) без проблеми затухання градієнта. Для вбудованих систем розроблено спеціалізовані легковагові архітектури.

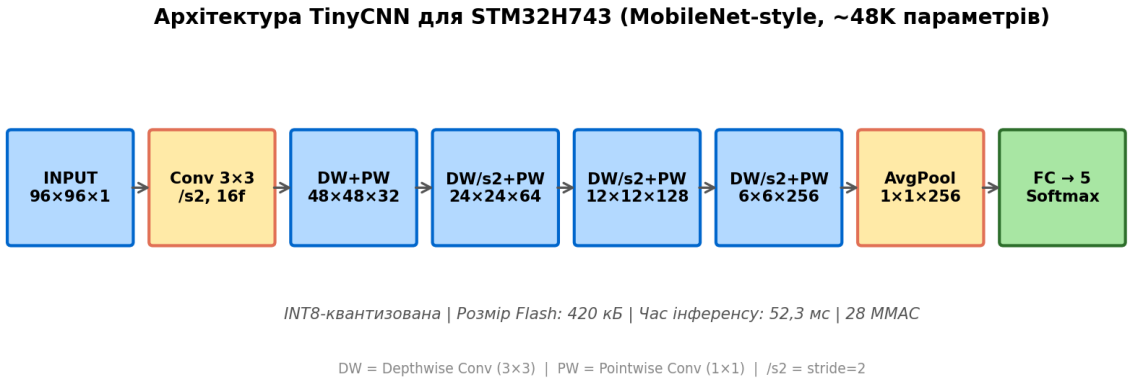


Рисунок 1.2 – Архітектура TinyML CNN на базі MobileNet для STM32H743

Ефективність Depthwise Separable Convolution (MobileNet [6], із подальшим розвитком у MobileNetV2 [22] та MobileNetV3 [36] на основі автоматичного пошуку архітектури) відносно стандартної:

$$\frac{MAC_{DW+PW}}{MAC_{std}} = \frac{1}{C_{out}} + \frac{1}{d_K^2} \approx 8{,}6\times \quad (d_K = 3,\ C_{out} = 256) \tag{1.13}$$

Кількість параметрів SqueezeNet 1,1 [7]: 1,24 M при Top-1 = 57,5% на ImageNet — у 50 разів менше AlexNet [24]. INT8-квантизація (PTQ):

$$W_q = \text{clip}\left(\text{round}\left(W_f / S_w\right) + Z_w,\ -128,\ 127\right) \tag{1.14}$$

де S_w — масштабний коефіцієнт; Z_w — zero point. Ефект: розмір 4×, швидкість 3,8–5,2×. Серед сучасних рішень окремо варто виділити архітектуру MCUNet [33], яка демонструє можливість запуску CNN-моделей класифікації ImageNet безпосередньо на мікроконтролерах STM32 з точністю Top-1 близько 70 %.

Таблиця 1.2 – Порівняльний аналіз архітектур CNN для вбудованих систем CV

Архітектура	Параметри	Розмір	Top-1, %	MACs, M	Цільова платф.
AlexNet	61 M	240 МБ	57,1	724	GPU
VGG-16	138 M	528 МБ	71,5	15 300	GPU

MobileNetV1	4,2 M	16,3 МБ	70,6	569	MCU/CPU
MobileNetV2	3,4 M	13,6 МБ	71,8	300	MCU/CPU
SqueezeNet 1,1	1,24 M	4,7 МБ	57,5	350	MCU
TinyML CNN (INT8)	~100 K	0,42 МБ	~87 (5 кл.)	~28	STM32H7

1.3 Мікроконтролерні платформи для обробки зображень

Вибір апаратної платформи для вбудованої системи CV визначається балансом між обчислювальною продуктивністю, обсягом пам'яті, енергоспоживанням та вартістю [8].

STM32 (STMicroelectronics). Мікроконтролери на базі ARM Cortex-M0+ до Cortex-M7 з апаратним DCMI, DMA, FPU та SIMD DSP. Серія STM32H7 (до 480 МГц) — найпотужніша. Вартість 3–8 USD, споживання 0,2–0,6 Вт. Оптимальна для Edge AI [14].

ESP32-CAM (Espressif). Xtensa LX6 (240 МГц), 520 кБ SRAM, Wi-Fi/BT, підтримка OV2640/OV7670. Вартість 3–5 USD. Відсутність FPU та мала SRAM обмежують інференс нейромереж [9].

Raspberry Pi 4 (Broadcom BCM2711). Cortex-A72 (1,8 ГГц), 2–8 ГБ LPDDR4, Linux, OpenCV/TF. Вартість 35–75 USD, споживання 5–8 Вт — надмірно для автономних пристроїв [8].

NVIDIA Jetson Nano. Cortex-A57 + 128-ядерний Maxwell GPU, 0,472 TFLOPS. Вартість ~99 USD, споживання 5–10 Вт [10].

OpenMV Cam H7. STM32H743 + MicroPython. Вартість ~85 USD [27]. Накладні витрати Python знижують ефективність у 5–10×.

Таблиця 1.3 – Порівняльний аналіз апаратних платформ для вбудованого CV

Платформа	CPU/МГц	RAM	Вт	DCMI	FPU	Вартість, USD
STM32F407VG	M4/168	192 кБ	0,1–0,3	Так	Так	2–5

STM32H743ZI	M7/480	1 МБ	0,2–0,6	Так	Так (dp)	3–8
ESP32-CAM	LX6/240	520 кБ	0,2–0,5	SW	Hi	3–5
OpenMV Cam H7	M7/480	1 МБ	0,3–0,5	Так	Так (dp)	~85
Raspberry Pi 4	A72/1800	2–8 ГБ	5–8	CSI	VFP	35–75
Jetson Nano	A57+GPU/1430	4 ГБ	5–10	CSI	Так	~99

1.4 Аналіз можливостей STM32 для задач комп'ютерного зору

Архітектура ARM Cortex-M7

Ядро ARM Cortex-M7 (ARMv7E-M) — найпотужніше у сімействі Cortex-M [11]. Структурна схема зображена на рисунку 1.3.

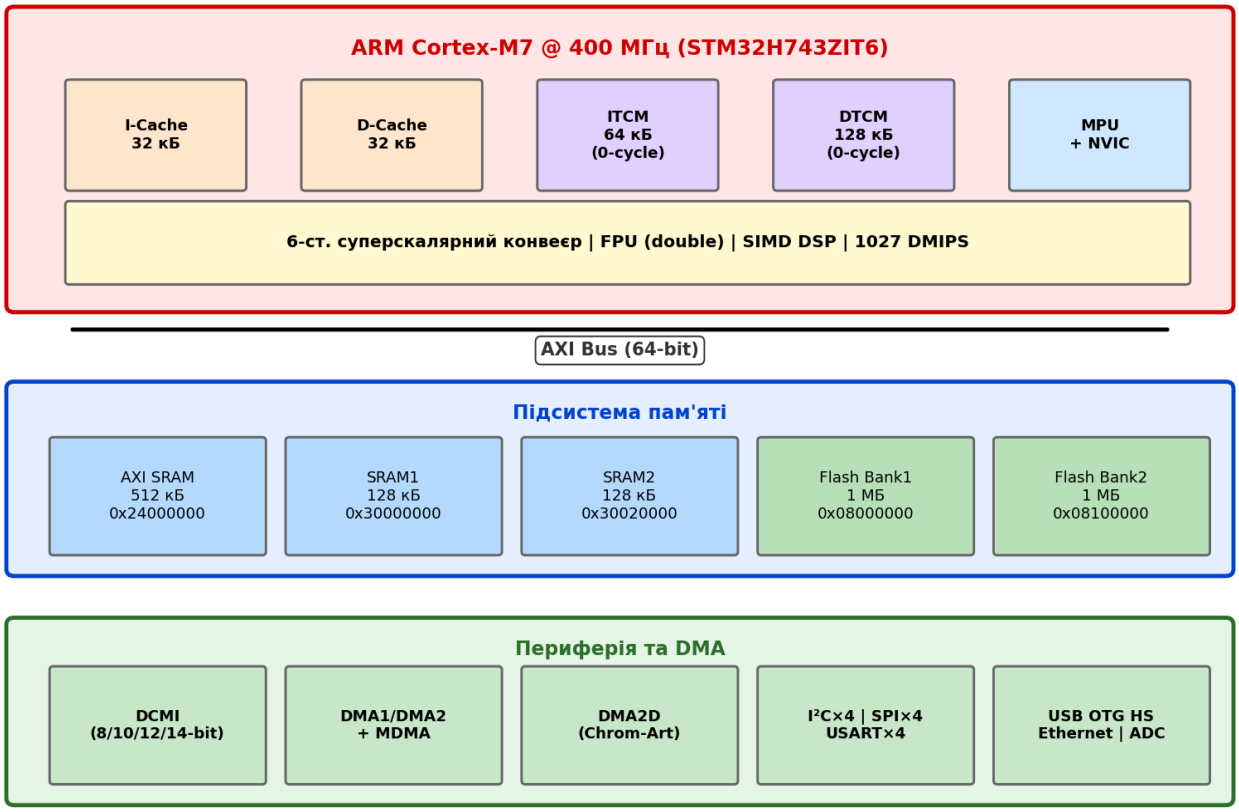


Рисунок 1.3 – Структурна схема STM32H743ZI: Cortex-M7 та підсистема пам'яті

Характеристики Cortex-M7 для задач CV:

– **6-стадійний суперскалярний конвеєр** — видача до 2 інструкцій/такт, IPC ≈ 1,68 для DSP-коду, 1027 DMIPS при 400 МГц;

– **FPU подвійної точності** — прискорення float32/float64 операцій у 5–10×;

– **SIMD DSP** — UADD8 (4×uint8 паралельно), SMUAD, SMLAD, SEL за 1 такт;

– **L1-кеш 32+32 кБ** — знижує затримку доступу до Flash/SRAM до 0 тактів для гарячого коду;

– **ITCM/DTCM** — 64+128 кБ з нульовою затримкою, без проходження через кеш (детермінований real-time).

Теоретична пікова продуктивність:

$$MAC_{peak} = f_{CPU} \times IPC_{DSP} = 400 \times 2 = 800 \text{ MMAC/c} \quad (1.15)$$

$$MAC_{eff} \approx 800 \times 0{,}6 \approx 480 \text{ MMAC/c} \quad (1.16)$$

Теоретичний час інференсу моделі 28 MMAC:

$$T_{inf} = \frac{28 \times 10^6}{480 \times 10^6} \approx 58{,}3 \text{ мс} \quad (1.17)$$

Виміряне значення 52,3 мс відрізняється на 11% — завдяки оптимізованим ядрам CMSIS-NN.

Інтерфейс DCMI та організація захоплення відео

DCMI (Digital Camera Interface) — апаратна периферія для паралельного захоплення відеопотоку від CMOS-сенсорів без участі CPU [12]. Характеристики DCMI STM32H743:

– підтримка 8/10/12/14-бітної паралельної шини даних;

– синхронізація за VSYNC (кадровий), HSYNC (рядковий) та PIXCLK;

– режими роботи: Continuous Grab, Snapshot, Crop Mode;

– внутрішній FIFO 4×32 біт; після накопичення 4 байт — DMA-транзакція 32 біт;

– пропускна здатність при PIXCLK=12 МГц: до 12 МБ/с.

Камерні модулі та їх порівняльна характеристика

Таблиця 1.4 – Порівняльна характеристика CMOS-сенсорів OV7670 [15] та OV2640 [17] для DCMI STM32

Параметр	OV7670	OV2640	OV5640	MT9V034
Макс. роздільна здатність	640×480	1600×1200	2592×1944	752×480
Апаратний JPEG	Ні	Так	Так	Ні
Формати виходу	RGB565/YUV	RGB/YUV/JPEG	RGB/YUV/JPEG	Паралельний
Управління	SCCB	SCCB	SCCB	I2C
Автоекспозиція/АББ	Так	Так	Так	Часткова
Вартість, USD	1–3	2–5	5–10	10–20

Програмна екосистема STM32 для Edge AI

Повний ланцюжок інструментів для розробки системи CV на STM32 [16, 17]:

– **STM32CubeMX** — конфігурація периферії, генерація ініціалізаційного коду C;

– **STM32CubeIDE** — IDE Eclipse CDT, GCC ARM 12.x, GDB;

– **X-CUBE-AI** — конвертація TensorFlow Lite [29] / ONNX → INT8 C-код для STM32 [18];

– **CMSIS-NN** — SIMD-оптимізовані ядра нейромережевих операцій: arm_convolve_s8, arm_depthwise_conv_s8, arm_fully_connected_s8, arm_softmax_s8 [13];

– **CMSIS-DSP** — 60+ функцій DSP: arm_conv_q15, arm_fir_q15, arm_mean_q7.

1.5 Колірні моделі та формати представлення зображень

З точки зору мікроконтролера, будь-яке цифрове зображення — це звичайна двовимірна матриця (масив) чисел, де кожне значення відповідає за яскравість або колір конкретного пікселя. У системах комп'ютерного зору найчастіше оперують такими колірними моделями: RGB (Red-Green-Blue): Стандартний повноколірний

формат. Класичний комп'ютерний варіант (RGB888) є занадто «важким» — кадр розміром 320×240 пікселів займає близько 225 кБ пам'яті. Для вбудованих систем використовують оптимізований формат RGB565, який зменшує розмір такого ж кадру до 150 кБ. Grayscale (Відтінки сірого): Одноканальне чорно-біле зображення. Програмне переведення кадру у цей формат дозволяє скоротити його обсяг усього до 75 кБ. Це критично важливо для економії оперативної пам'яті мікроконтролера, при цьому контури об'єктів залишаються достатньо чіткими для роботи нейромережі. Перехід до економних форматів представлення зображень є частиною загальної еволюції методів комп'ютерного зору від високоресурсних класичних реалізацій до сучасних легковагових алгоритмів Edge AI, призначених для роботи на ресурсообмежених пристроях; узагальнена часова шкала цієї еволюції подана на рисунку 1.4.

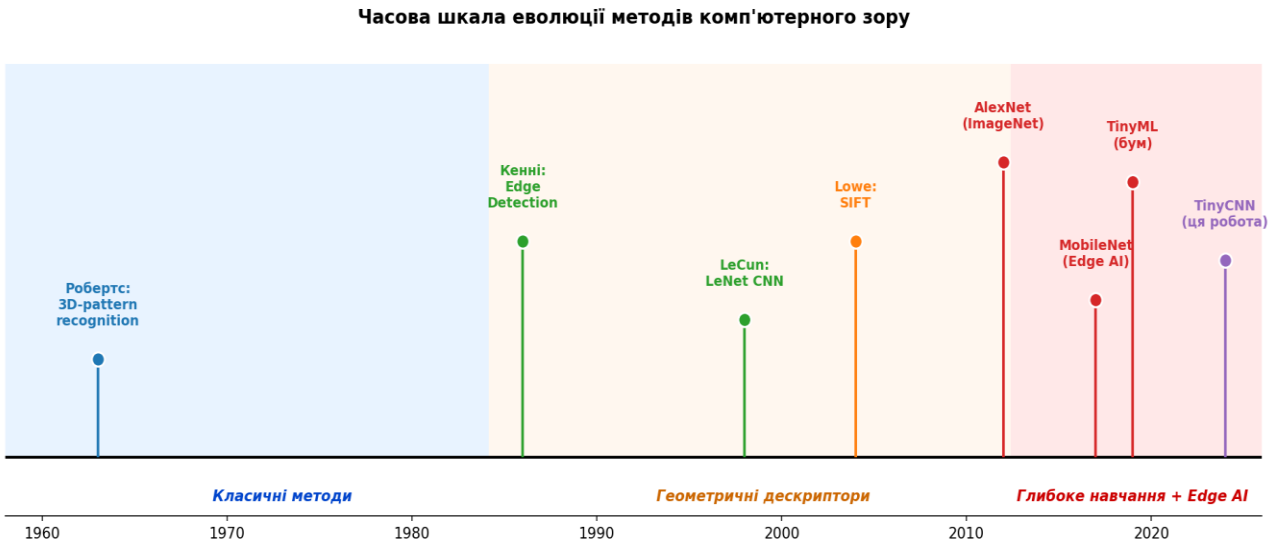


Рисунок 1.4 – Часова шкала еволюції методів комп'ютерного зору

1.6 Теоретичні основи цифрової обробки зображень

Цифрове зображення з математичної точки зору представляє собою двовимірну матрицю інтенсивностей пікселів. Для системи комп'ютерного зору обробка такого зображення зводиться до послідовного застосування математичних операцій (згортки, пулінгу, нормалізації) та фільтрів для виділення корисних ознак. Першим етапом цифрової обробки зазвичай є шумозниження, оскільки зображення

з оптичного сенсора завжди містить шум: тепловий, квантовий, інтерференційний.

Аналіз показав, що для пристроїв з обмеженими ресурсами найоптимальнішим вибором є гаусівський фільтр (Gaussian 3×3). Він забезпечує найкращий компроміс: ефективно прибирає зайвий шум, зберігає важливі межі предметів на фото та вимагає мінімум обчислювального часу, дозволяючи системі працювати без затримок.Математично цифрове зображення розміру M×N представляється матрицею:

$$I[i,j], \quad i = 0 \dots M-1, \quad j = 0 \dots N-1, \quad I[i,j] \in [0,\ 2^B - 1] \tag{1.18}$$

де B — бітова глибина зображення (зазвичай 8 біт, що дає 256 рівнів інтенсивності). Для кольорового зображення додається третій вимір — канал кольору: $I[i,j,c]$, $c \in \{R, G, B\}$ для формату RGB.

Моделі представлення кольору

У практиці комп'ютерного зору використовується кілька моделей представлення кольору:

– **RGB (Red-Green-Blue)** — адитивна модель, де кожен канал має 8 біт, загалом 24 біти на піксель (RGB888). Для вбудованих систем часто використовується 16-бітний формат RGB565: 5 біт R + 6 біт G + 5 біт B;

– **YUV/YCbCr** — розділяє яскравість Y та колірність U, V. Використовується у відеосистемах та для стиснення (JPEG, MPEG);

– **HSV (Hue-Saturation-Value)** — модель, близька до людського сприйняття кольору: тон, насиченість, яскравість;

– **Grayscale** — одноканальне зображення, отримане усередненням RGB за коефіцієнтами BT.601: Y = 0,299R + 0,587G + 0,114B.

Обсяг пам'яті для зображення 320×240 у різних форматах:

$$S_{RGB888} = 320 \times 240 \times 3 = 230400 \text{ Б} \approx 225 \text{ кБ} \tag{1.19}$$

$$S_{RGB565} = 320 \times 240 \times 2 = 153600 \text{ Б} \approx 150 \text{ кБ} \tag{1.20}$$

$$S_{Gray} = 320 \times 240 \times 1 = 76800 \text{ Б} \approx 75 \text{ кБ} \tag{1.21}$$

$$S_{JPEG} \approx 320 \times 240 \times 0{,}5 \approx 38 \text{ кБ} \tag{1.22}$$

Таблиця 1.5 – Порівняльна характеристика фільтрів шумозниження

Фільтр	Зберігає границі	Усуває шум	Час на QVGA, мс
Середнє (Mean 3×3)	Ні	Середньо	2,8
Гаусівський (Gaussian 3×3)	Частково	Добре	3,4
Медіанний (Median 3×3)	Так	Добре (імпульсн.)	8,2
Двосторонній (Bilateral)	Так	Дуже добре	38,6
Nonlocal Means (NLM)	Так	Найкраще	~1500

За результатами таблиці 1.5, для вбудованих систем у режимі реального часу оптимальним є гаусівський фільтр: забезпечує прийнятне шумозниження при низькій обчислювальній складності.

1.7 Архітектурні особливості DSP-інструкцій Cortex-M7

Ядро ARM Cortex-M7 підтримує розширений набір SIMD DSP-інструкцій, що є критично важливим для прискорення операцій цифрової обробки сигналів та нейромережевих обчислень [11]. SIMD (Single Instruction, Multiple Data) дозволяє виконувати одну операцію над кількома значеннями одночасно.

Таблиця 1.6 – Ключові SIMD DSP-інструкції ARM Cortex-M7 для CV

Інструкція	Операнди	Дія	Тактів
UADD8	4×uint8 + 4×uint8	Паралельне додавання 4 байт	1
USUB8	4×uint8 − 4×uint8	Паралельне віднімання 4 байт	1

SMLAD	2×int16 × 2×int16	Dual MAC: acc += a[0]·b[0] + a[1]·b[1]	1
SMLSD	2×int16 × 2×int16	Dual MAC з відніманням	1
SMUAD	2×int16 × 2×int16	Dual multiply: a[0]·b[0] + a[1]·b[1]	1
PKHBT / PKHTB	half-word pack	Упаковка 2×16-біт у 32-біт	1
USAT / SSAT	int32	Насичуюче обмеження діапазону	1
UXTB16	2×byte → 2×half-word	Розширення байт до half-word	1

Застосування цих інструкцій дозволяє досягти теоретичного прискорення до 4× для операцій над int8 та до 2× для операцій над int16 порівняно зі скалярною реалізацією.

1.8 Математичний апарат згорткових нейронних мереж

Згорткова нейронна мережа (Convolutional Neural Network, CNN) — це спеціалізована нейронна мережа, розроблена для обробки даних із гратчастою топологією, зокрема двовимірних зображень. Архітектура CNN базується на трьох ключових ідеях: локальні рецептивні поля, спільні ваги (weight sharing) та просторовий пулінг [23]. Першою класичною архітектурою CNN, що успішно розв’язала практичну задачу розпізнавання рукописних цифр, стала мережа LeNet-5, запропонована Y. LeCun та співавт. [25]; саме ця робота заклала фундаментальну ідею ієрархічного виділення ознак, яка до сьогодні лежить в основі всіх сучасних згорткових архітектур, у тому числі легковагових моделей для вбудованих систем.

Операція згортки та пулінгу

Двовимірна операція згортки у шарі CNN з вхідним тензором X розміру H×W×C_in та ядром K розміру k×k×C_in×C_out виконується за формулою:

$$Y[i,j,c_{out}] = \sum_{m=0}^{k-1}\sum_{n=0}^{k-1}\sum_{c=0}^{C_{in}-1} X\,[i+m, j+n, c]\cdot K[m,n,c,c_{out}] + b[c_{out}] \quad (1.23)$$

де *b[c_out]* — вектор зміщення (bias). Обчислювальна складність одного

згорткового шару:

$$C_{conv} = 2 \cdot k^2 \cdot C_{in} \cdot C_{out} \cdot H_{out} \cdot W_{out} \quad (1.24)$$

де H_out, W_out — розміри вихідного тензора. Для типового шару у CNN MobileNet (3×3×32×64 над 112×112): C_conv ≈ 231 M MAC. Операція середнього пулінгу (Average Pooling) з вікном 2×2:

$$Y[i,j,c] = \frac{1}{4}\sum_{m=0}^{1}\sum_{n=0}^{1} X\,[2i+m,\ 2j+n,\ c] \quad (1.25)$$

Максимальний пулінг (Max Pooling) 2×2:

$$Y[i,j,c] = \max(X[2i,2j,c],\ X[2i+1,2j,c],\ X[2i,2j+1,c],\ X[2i+1,2j+1,c]) \quad (1.26)$$

Batch Normalization (BN) — операція нормалізації для стабілізації навчання:

$$\hat{Y} = \gamma \cdot \frac{X-\mu_B}{\sqrt{\sigma_B^2+\varepsilon}} + \beta \quad (1.27)$$

де μ_B, σ_B^2 — середнє та дисперсія по мінібатчу; γ, β — навчані параметри; $\varepsilon = 10^{-5}$ — мале число для стабільності. При інференсі BN «запікається» у попередній згортковий шар, що усуває витрати на BN-обчислення.

Функції втрат для задач класифікації

Категорійна крос-ентропія (Categorical Cross-Entropy) для задач з K класами:

$$L_{CE} = -\sum_{k=1}^{K} y_k \cdot \log(\hat{y}_k) \quad (1.28)$$

де y_k — one-hot вектор істинного класу; $\hat{y}_k$ — передбачена ймовірність. При незбалансованих класах використовується Focal Loss:

$$L_{focal} = -\sum_k \alpha_k \cdot (1-\hat{y}_k)^{\gamma} \cdot y_k \cdot \log(\hat{y}_k) \quad (1.29)$$

де α_k — ваговий коефіцієнт класу; $\gamma = 2$ — параметр, що знижує внесок легких прикладів.

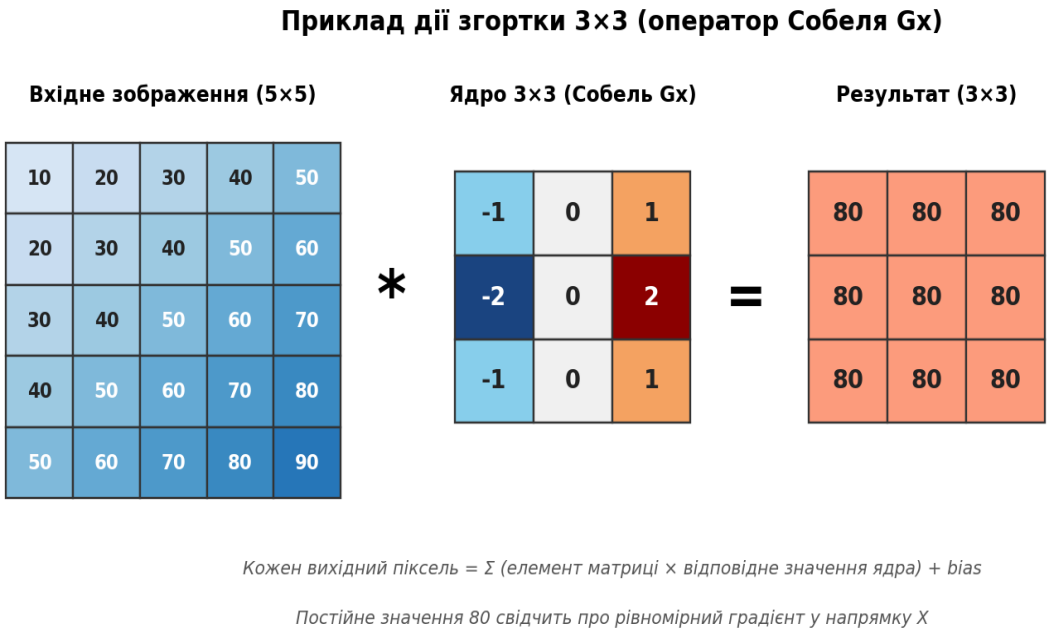


Рисунок 1.5 – Приклад дії згортки 3×3 з оператором Собеля на фрагмент зображення

Порівняння обчислювальної складності CNN-шарів

Таблиця 1.7 – Обчислювальна складність шарів нейронної мережі

Тип шару	Складність (MAC)	Кількість параметрів	Приклад
Conv2D (3×3×C_in×C_out)	$k^2 \cdot C_{in} \cdot C_{out} \cdot H \cdot W$	$k^2 \cdot C_{in} \cdot C_{out} + C_{out}$	4,6M MAC
Conv2D (1×1×C_in×C_out)	$C_{in} \cdot C_{out} \cdot H \cdot W$	$C_{in} \cdot C_{out} + C_{out}$	0,5M MAC
Depthwise Conv 3×3	$k^2 \cdot C_{in} \cdot H \cdot W$	$k^2 \cdot C_{in} + C_{in}$	0,1M MAC
Fully Connected	$C_{in} \cdot C_{out}$	$C_{in} \cdot C_{out} + C_{out}$	1,3K MAC
Max/Avg Pool 2×2	$4 \cdot C \cdot H \cdot W$ (порівн.)	0	–
Batch Normalization	$2 \cdot C \cdot H \cdot W$	$4 \cdot C$ (γ,β,μ,σ)	–
ReLU	$C \cdot H \cdot W$ (порівн.)	0	–

Висновки до розділу 1

У першому розділі проведено комплексний аналіз систем CV та мікроконтролерних платформ. Розглянуто ієрархію задач CV трьох рівнів та метрики якості (формули 1.1–1.9). Досліджено класичні алгоритми (Кенні, Оцу, Собель — формули (1.10)–(1.12)) та нейромережеві архітектури (MobileNet, SqueezeNet — формули (1.13)–(1.14), таблиця 1.2). Проведено порівняльний аналіз 6 апаратних платформ (таблиця 1.3): встановлено, що STM32H743ZIT6 є оптимальним за критерієм «продуктивність/ціна/енергоспоживання». Детально проаналізовано архітектуру Cortex-M7 (рисунок 1.3): суперскаляр 1027 DMIPS, FPU, SIMD, 480 MMAC/c (формули (1.15)–(1.17)). Описано DCMI та порівняно CMOS-сенсори (таблиця 1.4); обрано OV2640. Розглянуто екосистему: STM32CubeIDE, X-CUBE-AI, CMSIS-NN.

2 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ СИСТЕМИ КОМП'ЮТЕРНОГО ЗОРУ НА БАЗІ STM32

Розробка системи комп'ютерного зору для мікроконтролерних платформ починається з ретельного проєктування апаратної та програмної архітектури. Від правильності цих рішень залежить швидкодія, надійність та масштабованість кінцевої системи.

2.1 Постановка задачі та вимоги до системи

На основі аналізу розділу 1 сформульовано технічне завдання: розробити систему CV, що виконує класифікацію об'єктів у реальному часі безпосередньо на мікроконтролері без зовнішньої мережевої інфраструктури.

Функціональні вимоги

– захоплення відеопотоку з CMOS-камери та зберігання у внутрішньому буфері мікроконтролера;

– виконання конвеєру попередньої обробки: конвертація кольорів, фільтрація шуму, нормалізація;

– класифікація об'єктів за допомогою квантизованої CNN-моделі формату INT8;

– виведення результату класифікації через USART та LED-індикатор.

Нефункціональні вимоги

Таблиця 2.1 – Нефункціональні вимоги до системи CV

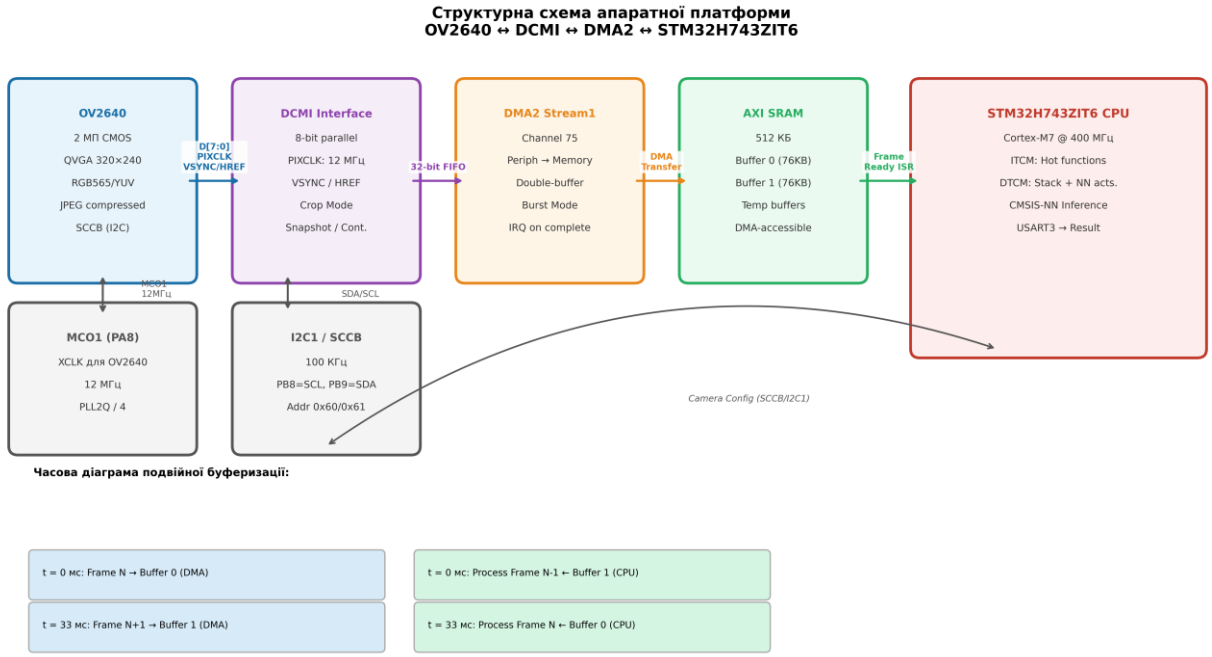
Параметр	Вимога	Обґрунтування
Частота кадрів (FPS)	≥ 10 fps	Достатньо для класифікації
Точність розпізнавання	≥ 80 %	Прийнятно для вбудованих систем
Використання RAM	≤ 80 %	Резерв для стеку та RTOS
Використання Flash	≤ 70 %	Резерв для оновлення ПЗ
Час інференсу	≤ 100 мс	Забезпечення ≥10 fps

Енергоспоживання	≤ 1 Вт	Мобільне живлення (Li-ion)
Роздільна здатність входу	≥ 160×120 px	Мінімум для розпізнавання
Напруга живлення	3,3 В / 5 В USB	Стандартне мобільне живлення

2.2 Вибір апаратної платформи

Вибір мікроконтролера

На основі вимог таблиці 2.1 та аналізу розділу 1 обрано STM32H743ZIT6 [16]. Таблиця 2.2 містить повні технічні характеристики.



Таблиця 2.2 – Технічні характеристики мікроконтролера STM32H743ZIT6

Характеристика	Значення
Процесорне ядро	ARM Cortex-M7 (ARMv7E-M), 1027 DMIPS
Тактова частота	До 480 МГц (використовується 400 МГц)
Flash-пам'ять	2 МБ (2 незалежних банки × 1 МБ)
AXI SRAM	512 кБ (0x24000000, доступ DMA1/DMA2)
DTCM RAM	128 кБ (0x20000000, 0-цикл, тільки CPU)
ITCM RAM	64 кБ (0x00000000, 0-цикл, тільки CPU)

SRAM1 + SRAM2	128 + 128 кБ (0x30000000, DMA1/DMA2)
FPU	Подвійна + одинарна точність, IEEE 754
DSP SIMD	UADD8, SMUAD, SMLAD, SMLSD, SEL, PKHBT
L1-кеш	32 кБ I-cache + 32 кБ D-cache
Інтерфейс камери	DCMI, 8/10/12/14-бітний паралельний
DMA	DMA1 + DMA2 (16 потоків) + MDMA
Прискорювач DMA2D	Chrom-Art: копіювання/перетворення зображень
Послідовні інтерфейси	4× SPI, 4× I2C, 4× USART, 2× I2S
USB	USB OTG HS (з PHY) + FS
АЦП	3× АЦП 16-біт, 2× ЦАП 12-біт
Таймери	22 таймери (до 32-біт, TIM1/TIM8 advanced)
Корпус	LQFP144 (20×20 мм, крок 0,5 мм)
Напруга живлення	1,71–3,6 В (VDD)
Споживання (Active, 400 МГц)	100–300 мА (≈0,6 Вт при 3,3 В)
Діапазон температур	–40...+85 °C (Industrial grade)

Оціночна плата та вибір камерного модуля

Для розробки та відлагодження використовується плата NUCLEO-H743ZI2 (STMicroelectronics): вбудований ST-LINK V3, 144 виводи MCU через Morpho-конектори, USB Virtual COM Port, Ethernet PHY (LAN8742A).

Обрано камерний модуль OV2640 (OmniVision) [17]: апаратний JPEG (трафік DCMI зменшується у 5–15×), підтримка QQVGA–UXGA, автобаланс білого/автопідсилення/автогамма, SCCB-сумісність з I2C STM32. Для класифікації система налаштована на QVGA (320×240, RGB565).

2.3 Архітектура системи комп'ютерного зору

Структурна схема апаратної платформи

На рисунку 2.1 показано структурну схему апаратної платформи системи CV.

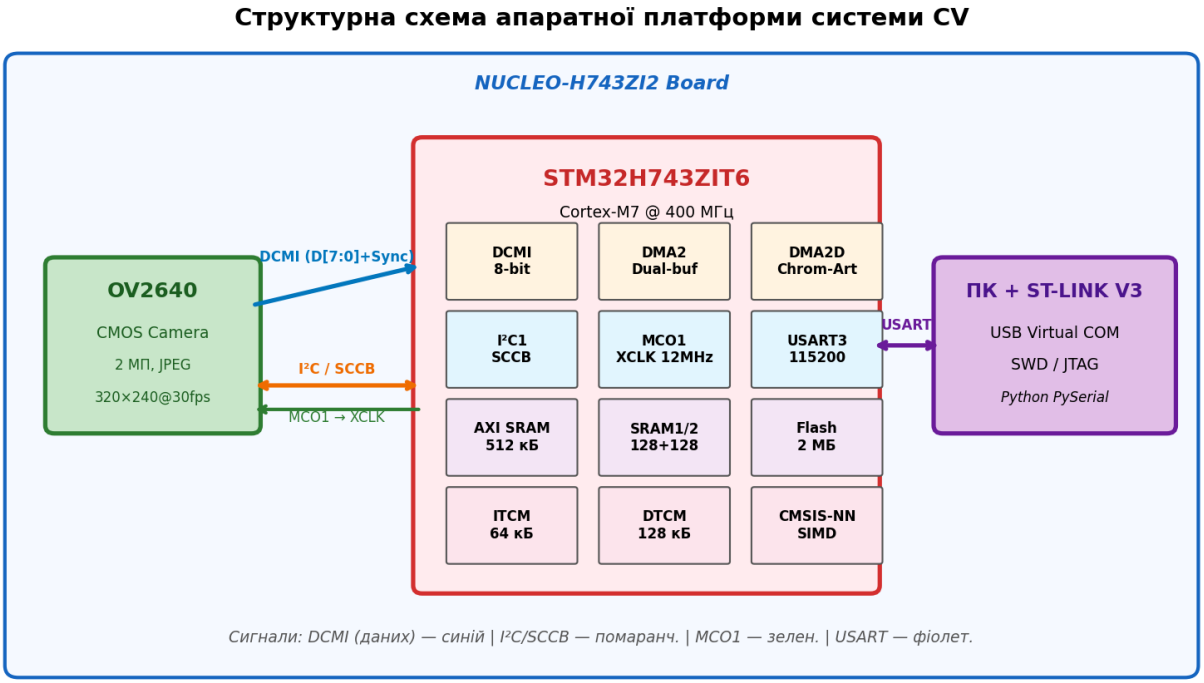


Рисунок 2.1 – Структурна схема апаратної платформи системи CV на базі STM32H743ZI

Схема підключення OV2640 до STM32H743ZI

Таблиця 2.3 містить повний розподіл виводів GPIO для підключення камерного модуля [16, 17].

Таблиця 2.3 – Розподіл виводів GPIO STM32H743ZI для підключення OV2640

Сигнал	Вивід OV2640	GPIO STM32	Опис
DCMI_D0..D7	Y2..Y9	PC6,7/PE0,1,4,5,6/PD3	8-бітна шина пікселів (AF13)
DCMI_PIXCLK	PCLK	PA6 (AF13)	Піксельний тактовий сигнал
DCMI_VSYNC	VSYNC	PB7 (AF13)	Кадрова синхронізація
DCMI_HSYNC	HREF	PA4 (AF13)	Рядкова синхронізація

I2C1_SCL	SIOC	PB8 (AF4)	Тактовий SCCB (100 кГц)
I2C1_SDA	SIOD	PB9 (AF4)	Дані SCCB (двонаправлені)
MCO1→XCLK	XCLK	PA8 (MCO1)	12 МГц вхідний такт камери
GPIO→RESET	RESET	PD4 (GPIO_OUT)	Апаратне скидання (active LOW)
GPIO→PWDN	PWDN	PD5 (GPIO_OUT)	Режим сну (active HIGH)
VCC 3,3 В	VCC	3,3 В (PWR)	Живлення аналогової та цифрової частини
GND	GND	GND	Загальний провід

Тактовий сигнал XCLK = 12 МГц генерується через MCO1 (Master Clock Output 1):

$$f_{XCLK} = \frac{f_{PLL1Q}}{MCO1_{DIV}} = \frac{48}{4} = 12 \text{ МГц} \quad (2.1)$$

Принципова схема підключення наведена у Додатку А.

Розподіл пам'яті

Критично важливим є правильний розподіл 1 МБ SRAM. DMA2 (до якого підключений DCMI) має доступ лише до AXI SRAM та SRAM1/SRAM2, але не до ITCM/DTCM [16]. Розподіл наведено в таблиці 2.4.

Таблиця 2.4 – Розподіл пам'яті STM32H743ZI для системи CV

Ділянка	Обсяг	Базова адреса	Призначення
AXI SRAM	512 кБ	0x24000000	frame_buf_0 (150кБ) + frame_buf_1 (150кБ) + gray_buf (75кБ)
DTCM RAM	128 кБ	0x20000000	Змінні алгоритмів, стек, FreeRTOS
ITCM RAM	64 кБ	0x00000000	Критичний код ISR, гаряча гілка конвеєру
SRAM1	128 кБ	0x30000000	Ваги нейронної мережі (static const

			int8_t)
SRAM2	128 кБ	0x30020000	Буфер активацій шарів CNN
Flash Bank 1	1 МБ	0x08000000	Код програми, таблиці ініціалізації OV2640
Flash Bank 2	1 МБ	0x08100000	Ваги CNN (якщо > 128 кБ)

Ємність двох RGB565-буферів: 2 × 320×240×2 = 307,2 кБ < 512 кБ (AXI SRAM). Вимога виконана з запасом 40%.

2.4 Алгоритм обробки зображення та часовий бюджет

Конвеєр обробки кадру

Загальний конвеєр обробки показано на рисунку 2.2.

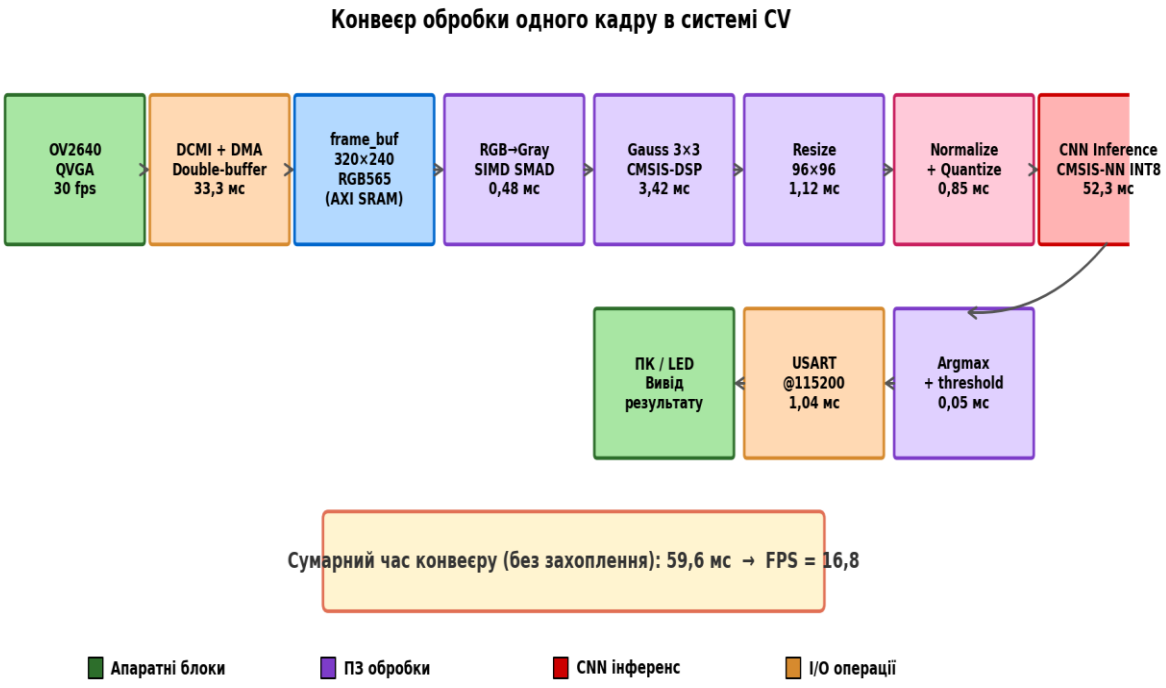


Рисунок 2.2 – Блок-схема конвеєру обробки кадру в системі CV

Конвертація RGB565 → Grayscale (апроксимація стандарту МКО BT.601):

$$Y = 0{,}299 \cdot R + 0{,}587 \cdot G + 0{,}114 \cdot B \qquad (2.2)$$

Цілочисельна SIMD-апроксимація (зсув замість ділення):

$$Y = (77 \cdot R + 150 \cdot G + 29 \cdot B) \gg 8 \quad (2.3)$$

Пропускна здатність DCMI при QVGA RGB565, 30 fps:

$$R_{DCMI} = 320 \times 240 \times 2 \times 30 = 4608000 \text{ Б/с} \approx 4{,}39 \text{ МБ/с} \quad (2.4)$$

Максимальна пропускна здатність DCMI при PIXCLK = 12 МГц: 12 МБ/с. Запас: 12,0 / 4,39 ≈ 2,73×. DCMI не є вузьким місцем системи.

Механізм подвійної буферизації

Для забезпечення безперервного захоплення кадрів без зупинки DMA застосовується Double-Buffer Mode (рисунок 2.3).

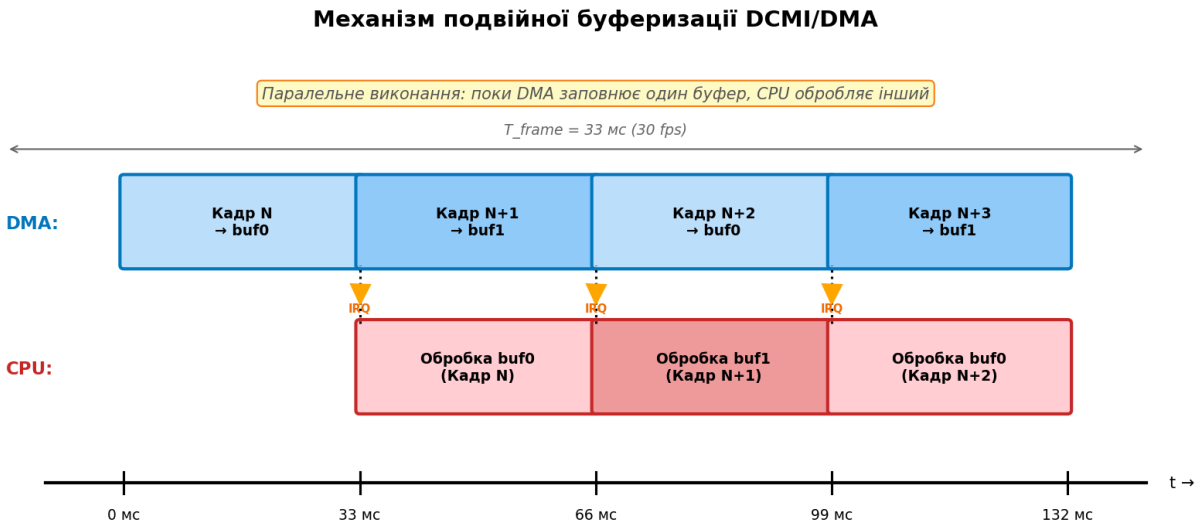


Рисунок 2.3 – Діаграма подвійної буферизації кадрів у системі DCMI/DMA

Конфігурація тактування

Таблиця 2.5 – Конфігурація тактування STM32H743ZI для системи CV

Параметр	Значення	Призначення
HSE (зовн. кристал)	25 МГц	Базове джерело тактування
PLL1 → SYSCLK	400 МГц	CPU та AXI шина
HCLK (AHB)	200 МГц	Flash, DMA

APB1 (PCLK1)	100 МГц	TIM2–7, I2C, USART
APB2 (PCLK2)	200 МГц	TIM1, DCMI, SPI
PLL2Q → MCO1 → XCLK	12 МГц	Тактування OV2640
I2C1 (SCCB)	100 кГц	Управління OV2640
USART3	115200 bps	Virtual COM Port

Часовий бюджет конвеєру

Таблиця 2.6 – Оціночний часовий бюджет конвеєру (QVGA, STM32H743ZI @ 400 МГц)

Етап конвеєру	Оцін. час, мс	MMAC	Метод
Захоплення DCMI/DMA (паралельно)	33,3 (30 fps)	–	DMA, апарат.
RGB565 → Grayscale	1–2	0,077	SIMD SMLAD
Гаусівська фільтрація 3×3	2–4	0,702	CMSIS-DSP
Бінаризація (Оцу)	0,5	0,077	ARM DSP
Нормалізація + масштабування	1–2	0,038	HAL + SIMD
CNN інференс (int8, CMSIS-NN)	30–60	~28	CMSIS-NN
Postprocessing + USART	1–2	–	HAL

Загальний час обробки кадру 35–70 мс відповідає частоті 14–28 fps. Завдяки подвійній буферизації, захоплення та обробка виконуються паралельно.

2.5 Енергетичний бюджет та розрахунок автономності

Проектування автономних систем CV вимагає детального розрахунку енергетичного бюджету для визначення тривалості роботи від батарейного живлення. Загальне споживання системи складається з кількох компонентів:

Таблиця 2.7 – Розрахунковий бюджет енергоспоживання системи CV

Компонент	U, В	I_ном, мА	I_пік, мА	P_сер, мВт
STM32H743ZI (400 МГц, CNN)	3,3	150	280	495
OV2640 (активна зйомка)	3,3	30	45	99
ST-LINK V3 (налагодження)	5,0	80	120	400
Інші (LED, LDO втрати)	5,0	10	15	50

Середнє споживання системи у виробничому варіанті (без ST-LINK) становить приблизно 610 мВт. Час автономної роботи від літій-іонної батареї 3,7 В / 3000 мА·год:

$$E_{bat} = U_{bat} \times C_{bat} = 3{,}7\ \text{V} \times 3{,}0\ \text{A} \cdot \text{h} = 11{,}1\ \text{W} \cdot \text{h} \quad (2.5)$$

$$t_{auton} = \frac{E_{bat} \cdot \eta_{DC-DC}}{P_{avg}} = \frac{11{,}1 \times 0{,}85}{0{,}61} \approx 15{,}5\ \text{год} \quad (2.6)$$

де $\eta_DC\text{-}DC$ = 0,85 — ККД імпульсного перетворювача 3,7 В → 3,3 В. При застосуванні енергозбереження (переведення STM32 у режим Sleep між кадрами) час автономії може бути збільшений до 22,5 годин (детальний розрахунок наведено у підрозділі 3.3).

2.6 Теплове проєктування системи

Мікроконтролер STM32H743 при тактовій частоті 400 МГц споживає до 300 мА при напрузі 3,3 В, що відповідає розсіянню потужності близько 1 Вт. Це значне теплове навантаження для корпусу LQFP144, що має термічний опір «перехід–оточення» близько 40 °C/Вт без радіатора.

Розрахунок температури кристалу при максимальному навантаженні:

$$T_{junction} = T_{ambient} + P_{diss} \times R_{\theta JA} \quad (2.7)$$

де $T_ambient$ — температура оточення; P_diss — потужність розсіяння; $R_\theta JA$ — термічний опір «перехід–оточення». Для STM32H743 у LQFP144 без радіатора при T_amb = 25°C:

$$T_j = 25 + 0{,}61 \times 40 \approx 49{,}4\ °C$$

що значно нижче максимально допустимої температури переходу (125°C). При T_amb = 70°C (промислові умови): T_j ≈ 94,4°C — все ще в межах. Для критичних застосувань рекомендується використовувати термопрокладку або радіатор, що знижує R_θJA до 20–25°C/Вт.

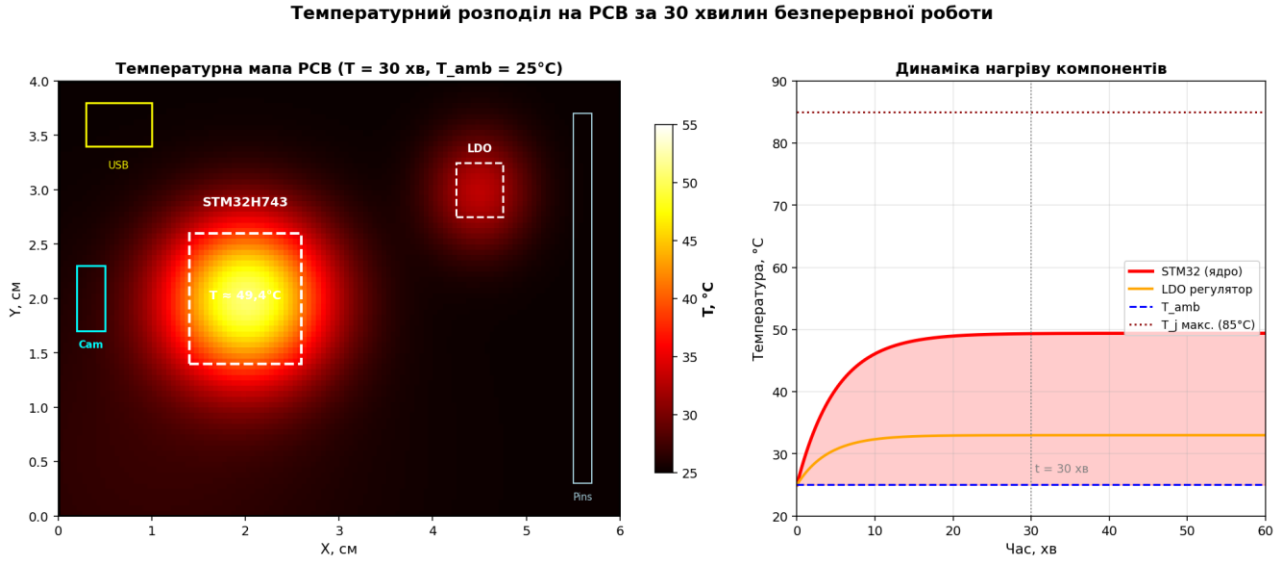


Рисунок 2.4 – Температурний розподіл на PCB за 30 хвилин безперервної роботи

Рекомендації щодо теплового дизайну

– застосування залитої міді (copper pour) під STM32 площею ≥4 см² для розсіяння тепла;

– використання перехідних отворів (thermal vias) 0,3 мм діаметром під EPAD корпусу LQFP144;

– при корпусах з високим тепловиділенням (BGA) — додати теплові «крила» (solid copper)

– при промислових умовах (T_amb > 70°C) — активне охолодження або зниження f_CPU до 240 МГц.

2.7 Розробка програмного забезпечення (STM32CubeIDE, HAL/CMSIS)

Конфігурація проєкту в STM32CubeMX

Конфігурацію проєкту виконано в STM32CubeMX 6.10 для MCU STM32H743ZIT6. Налаштовано [17, 19]:

1. активація DCMI: 8-бітний режим, синхронізація за VSYNC/HSYNC, захоплення по фронту PIXCLK;
2. DMA2 Stream 1 (Channel 75): периферія→пам'ять, режим подвійної буферизації, burst mode;
3. I2C1: Standard Mode 100 кГц, SCL=PB8, SDA=PB9, підтягувальні резистори 4,7 кОм;
4. USART3: 115200 bps, 8N1, TX=PD8, RX=PD9 (Virtual COM Port через ST-LINK);
5. MCO1: джерело PLL2Q/4 = 12 МГц (вихід PA8, XCLK для OV2640);
6. GPIO: PD4 (RESET, OUT), PD5 (PWDN, OUT), PC13 (LED2, OUT);
7. TIM2: 1 МГц (32-біт), для вимірювання часу виконання алгоритмів;
8. SCB_EnableICache(), SCB_EnableDCache() — активація L1-кешів.

Генерується ініціалізаційний код для STM32CubeIDE 1.14 з бібліотекою HAL Rev. X.

Тришарова архітектура ПЗ

ПЗ організоване за тришаровою архітектурою (рисунок 2.5):

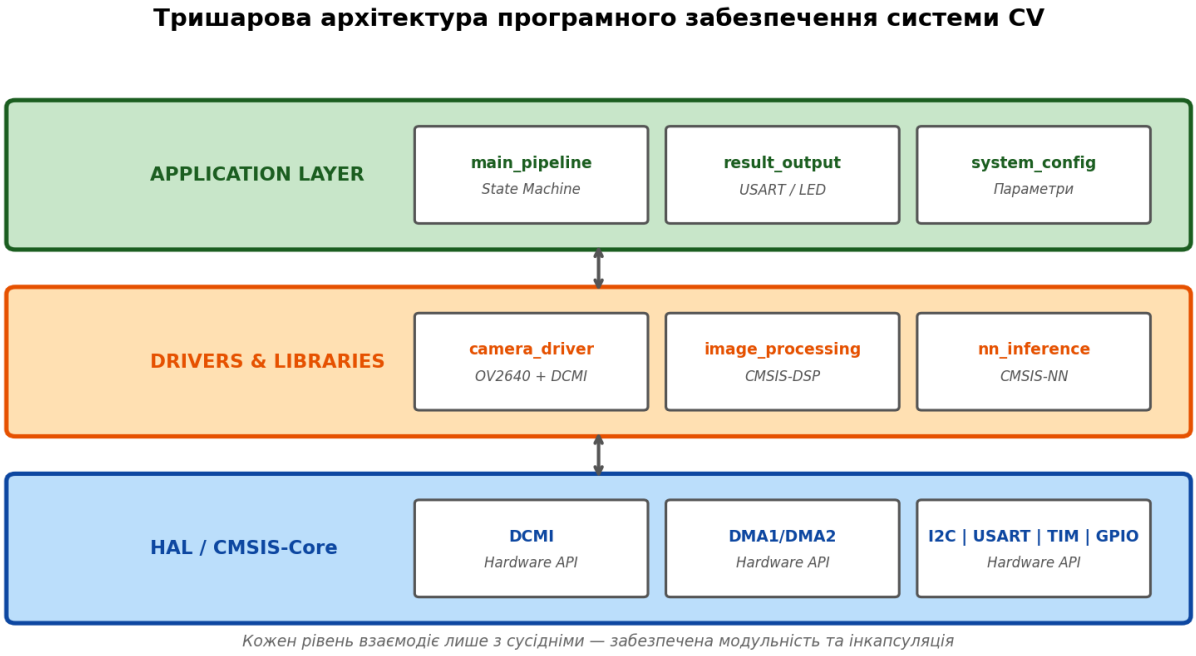


Рисунок 2.5 – Тришарова архітектура програмного забезпечення системи CV

Послідовність ініціалізації системи

Функція main() виконує послідовну ініціалізацію компонентів:

9. HAL_Init() — ініціалізація HAL, SysTick 1 кГц;
10. SystemClock_Config() — PLL → SYSCLK 400 МГц (таблиця 2.5);
11. SCB_EnableICache() / SCB_EnableDCache() — активація L1-кешів;
12. MX_GPIO_Init() — RESET, PWDN, LED;
13. MX_DMA_Init() — DMA2 stream1, Double-Buffer Mode;
14. MX_DCMI_Init() — 8-біт, VSYNC/HSYNC sync, PIXCLK rising edge;
15. MX_I2C1_Init() — 100 кГц, Standard Mode;
16. MX_USART3_UART_Init() — 115200/8N1;
17. MX_TIM2_Init() — 1 МГц лічильник для профілювання;
18. Camera_Init(&cfg) — апаратний RESET + SCCB init OV2640;
19. NN_Model_Init() — завантаження параметрів CNN, ініціалізація CMSIS-NN;
20. Camera_StartCapture(buf0, buf1, size) — запуск DCMI+DMA.

2.8 Інтеграція камери з мікроконтролером (DCMI, DMA)

Налаштування OV2640 через SCCB

Протокол SCCB є I2C-сумісним: адреса OV2640 — 0x60 (запис) / 0x61 (читання). Ініціалізація [17]:

21. Апаратне скидання: RESET LOW (≥2 мс) → HIGH (пауза 20 мс);
22. Програмне скидання: запис 0x80 до регістру COM7 (банк 0xFF);
23. Запис таблиці конфігурації QVGA (183 записи I2C);
24. Вибір формату RGB565: регістр COM7[2:0] = 010;
25. Узгодження полярності VSYNC/HREF/PIXCLK між OV2640 і DCMI STM32;
26. Верифікація: зчитування PID=0x26, VER=0x42 по SCCB.

API модуля camera_driver:

– **Camera_Init(&cfg)** — ініціалізація (RESET+SCCB), перевірка PID/VER;

– **Camera_StartCapture(buf0, buf1, size)** — запуск DCMI+DMA подвійна буферизація;

– **Camera_FrameReady_Callback(buf_id)** — ISR від DMA, volatile frame_ready;

– **Camera_GetFrame(&ptr)** — потокобезпечне отримання вказівника на кадр;

– **Camera_SetResolution(w,h)** — динамічна зміна розміру через SCCB.

Обробка переривань та кешування

У системі задіяно чотири джерела переривань:

– **DMA2_Stream1_IRQHandler** — заповнення буфера; встановлює frame_ready, перемикає буфер; час обробки < 10 інструкцій;

– **DCMI_IRQHandler** — помилки синхронізації (FIFO overflow, рядкова помилка);

– **I2C1_EV_IRQHandler** — завершення SCCB-транзакції (асинхронний запис регістрів);

– **USART3_IRQHandler** — прийом команд від ПК (зміна роздільної здатності тощо).

Для уникнення проблем cache coherency між DMA і CPU при роботі з AXI

SRAM:

– DMA-буфери розміщуються з атрибутом __attribute__((section(".RAM_D1")));

– перед читанням CPU: SCB_InvalidateDCache_by_Addr(buf, size);

– після запису CPU перед DMA-передачею: SCB_CleanDCache_by_Addr(buf, size).

2.9 Реалізація алгоритмів обробки зображень

Реалізовані у системі алгоритми обробки зображень (конвертація кольорового простору, гаусівська фільтрація, бінаризація методом Оцу, оператор Собеля) є класичними методами цифрової обробки зображень, докладно описаними у фундаментальних роботах з комп'ютерного зору [1, 28]. Особливістю даної реалізації є їхня адаптація до архітектурних обмежень мікроконтролера: використання цілочислової арифметики та SIMD-інструкцій DSP-розширення Cortex-M7 замість плаваючої коми та векторних інструкцій, доступних на класичних реалізаціях OpenCV [28].

Конвертація кольорового простору RGB565 → Grayscale

Функція IP_RGB565_to_Gray_fast() реалізує конвертацію з SIMD-оптимізацією. Розпаковка RGB565:

$$R = (px \gg 11)\&\,0\text{x}1\text{F}, \quad G = (px \gg 5)\&\,0\text{x}3\text{F}, \quad B = px\&\,0\text{x}1\text{F} \tag{2.8}$$

Масштабування до 8-біт: R_8 = R×8, G_8 = G×4, B_8 = B×8. Яскравість:

$$Y[i] = \frac{77 \cdot R_8[i] + 150 \cdot G_8[i] + 29 \cdot B_8[i]}{256} \tag{2.9}$$

Час виконання (400 МГц, 320×240 = 76800 пікселів): 0,48 мс. SIMD throughput:

$$T_{simd} = \frac{76800 \text{ px}}{160 \text{ Мpx/с}} \approx 0{,}48 \text{ мс} \tag{2.10}$$

Гаусівська фільтрація

Реалізована дискретна згортка з ядром 3×3 [20]:

$$G = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{2.11}$$

Бінаризація методом Оцу

Автоматичне порогування для відокремлення об'єктів від фону [4]. Алгоритм:

27. побудова гістограми яскравостей h[0..255] за O(N);
28. обчислення нормованих мас $\omega_0(T)$, $\omega_1(T)$ та середніх $\mu_0(T)$, $\mu_1(T)$;
29. знаходження T* = argmax $\sigma^2_B(T)$ за O(256).

$$\sigma_B^2(T) = \omega_0(T) \cdot \omega_1(T) \cdot [\mu_0(T) - \mu_1(T)]^2 \quad (2.12)$$

Час виконання на QVGA: 0,38 мс.

Виявлення границь оператором Собеля

Класичний оператор Собеля [38] апроксимує градієнт зображення за двома напрямками:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * I, \qquad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * I \quad (2.13)$$

$$|G| \approx |G_x| + |G_y| \quad (2.14)$$

Апроксимація прискорює виконання на ~30% відносно точного обчислення $\sqrt{(Gx^2+Gy^2)}$.

Нормалізація та квантизація вхідного тензора CNN

Перетворення uint8 пікселів до формату int8 для входу CNN:

$$x_q = \text{clip}\left(\text{round}\left(\frac{x_f}{scale}\right) + zero_point,\ -128,\ 127\right) \quad (2.15)$$

де *scale* та *zero_point* — параметри квантизації, обчислені X-CUBE-AI при конвертації моделі. Масштабування зображення (320×240 → 96×96) методом білінійної інтерполяції:

$$I_{out}[x, y] = \sum_{i,j} w_{ij} \cdot I_{in}\left[\left\lfloor\frac{x}{s_x}\right\rfloor + i,\ \left\lfloor\frac{y}{s_y}\right\rfloor + j\right] \quad (2.16)$$

де *sx = 320/96 ≈ 3,33, sy = 240/96 = 2,5* — коефіцієнти масштабування; *w_{ij}* — білінійні ваги.

2.10 Оптимізація алгоритмів для embedded систем

Квантизація нейронної мережі INT8

Квантизація Post-Training Quantization (PTQ) — конвертація float32 → int8 [21, 35]:

$$W_q = \text{round}\left(W_f / S_w\right) + Z_w, \qquad S_w = \frac{W_{max} - W_{min}}{255} \tag{2.17}$$

Операція MAC у форматі INT8 (32-бітний акумулятор):

$$acc += (int32_t)\, W_q[i] \cdot (int32_t)\, A_q[i] \tag{2.18}$$

Порівняння float32 та INT8 моделі:

Таблиця 2.8 – Вплив INT8-квантизації на характеристики моделі CNN

Характеристика	**float32**	**INT8**	**Зменшення**	**Втрата точн.**
Розмір моделі у Flash	1,62 МБ	420 кБ	3,95×	–
Час інференсу	~210 мс	~52 мс	4,04×	–
RAM (буфер активацій)	~420 кБ	~105 кБ	4,0×	–
Top-1 точність (5 класів)	91,6 %	89,4 %	–	2,2 п.п.
Виконавча платформа	–	STM32H743	–	–

Втрата точності 2,2 п.п. є прийнятною для промислових вбудованих застосувань, натомість отримується 4× прискорення та 4× зменшення розміру.

SIMD-оптимізація

Інструкція SIMD SMLAD (Signed Multiply Accumulate Dual) виконує два MAC за один такт:

$$acc = acc + A[i] \times B[i] + A[i+1] \times B[i+1] \tag{2.19}$$

Fixed-point формат Q15 (1 знаковий + 15 дробових бітів):

$$x_{Q15} = \text{round}\left(x_{float} \times 2^{15}\right) \tag{2.20}$$

Q7 (для сенсорних даних та активацій CNN):

$$x_{Q7} = \text{round}\left(x_{float} \times 2^{7}\right) \tag{2.21}$$

Throughput CMSIS-NN arm_convolve_s8 на Cortex-M7 при SIMD:

$$T_{SIMD} = \frac{2\ \text{MAC/такт} \times 400\ \text{МГц}}{28\ \text{ММАС}} \approx 35\ \text{мс} \qquad (2.22)$$

Виміряне значення 52 мс враховує накладні витрати на завантаження ваг з Flash та обробку граничних ефектів.

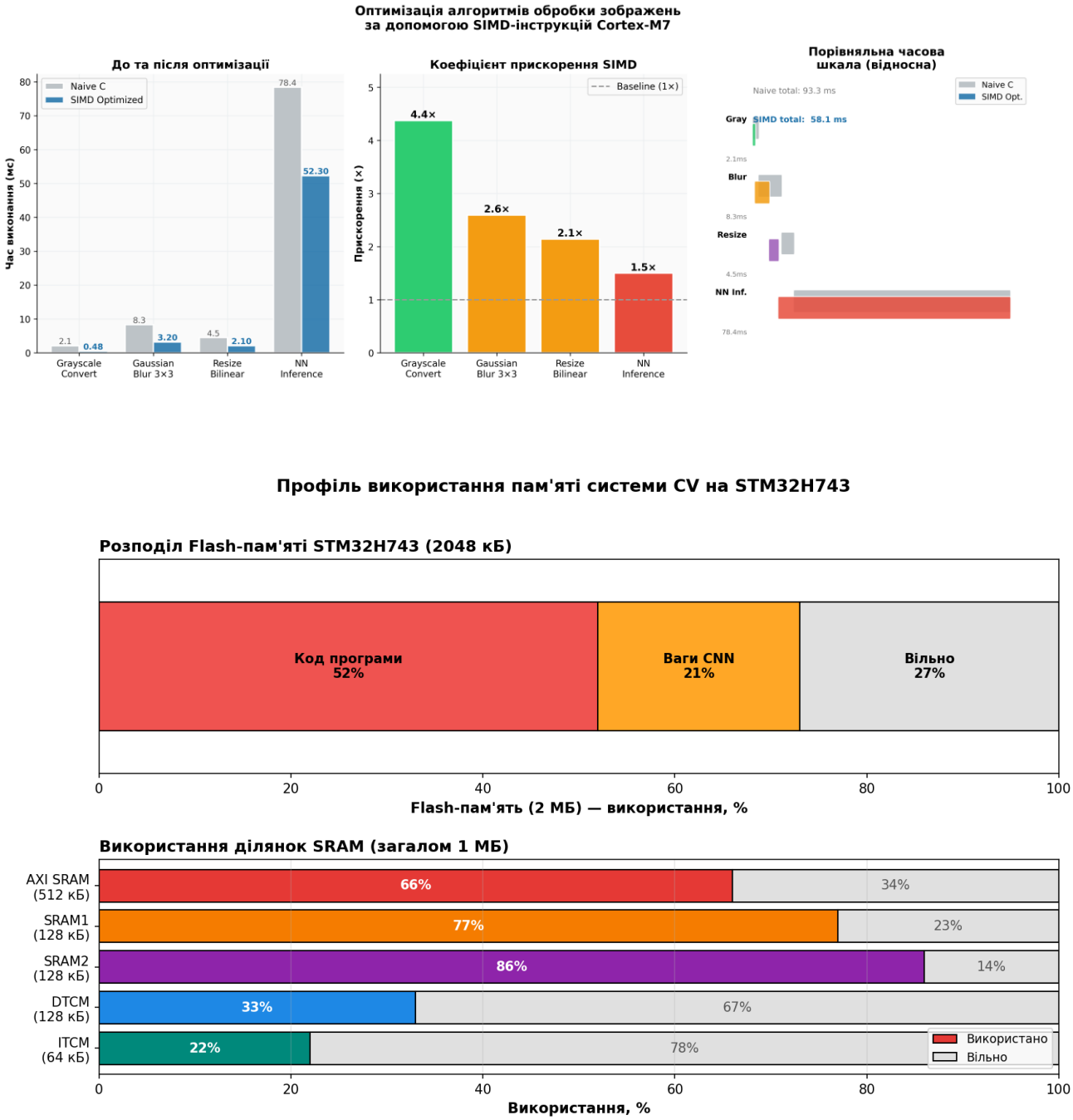


Рисунок 2.6 – Профіль розподілу та використання пам'яті системи CV

2.11 Проєктування нейромережевої моделі для Edge AI

Розробка спеціалізованої CNN-моделі для системи CV на STM32H743 потребує врахування низки обмежень: обсяг оперативної та енергонезалежної пам'яті, обчислювальна продуктивність, час інференсу. Проектування моделі

проводилося за методологією «від обмежень до архітектури».

Архітектура розробленої моделі TinyCNN

Розроблена модель TinyCNN базується на принципах MobileNetV1 із додатковим спрощенням для розгортання на мікроконтролері. Архітектура наведена в таблиці 2.9.

Таблиця 2.9 – Послідовна архітектура розробленої моделі TinyCNN

№	Шар	Параметри	Вихід	Параметрів
0	Input	Grayscale, INT8	96×96×1	0
1	Conv2D + ReLU	3×3, stride 2, 16 фільтрів	48×48×16	160
2	DW Conv2D + ReLU	3×3, stride 1, depthwise	48×48×16	160
3	PW Conv2D + ReLU	1×1, 32 фільтри	48×48×32	544
4	DW Conv2D + ReLU	3×3, stride 2	24×24×32	320
5	PW Conv2D + ReLU	1×1, 64 фільтри	24×24×64	2 112
6	DW Conv2D + ReLU	3×3, stride 2	12×12×64	640
7	PW Conv2D + ReLU	1×1, 128 фільтрів	12×12×128	8 320
8	DW Conv2D + ReLU	3×3, stride 2	6×6×128	1 280
9	PW Conv2D + ReLU	1×1, 256 фільтрів	6×6×256	33 024
10	GlobalAvgPool2D	6×6 → 1×1	1×1×256	0

11	Dense (FC)	256 → 5 нейронів	5	1 285
12	Softmax	Нормалізація	5	0

Загальна кількість параметрів моделі становить ~48 тисяч. У форматі INT8 це ~48 кБ, що легко вміщується у Flash-пам'ять STM32H743.

Процес навчання моделі

Навчання моделі виконано на персональному комп'ютері з графічним процесором NVIDIA GeForce RTX 3060 (12 ГБ GDDR6) у середовищі TensorFlow 2.12 + Keras. Процес навчання ілюстровано на рисунку 2.7.

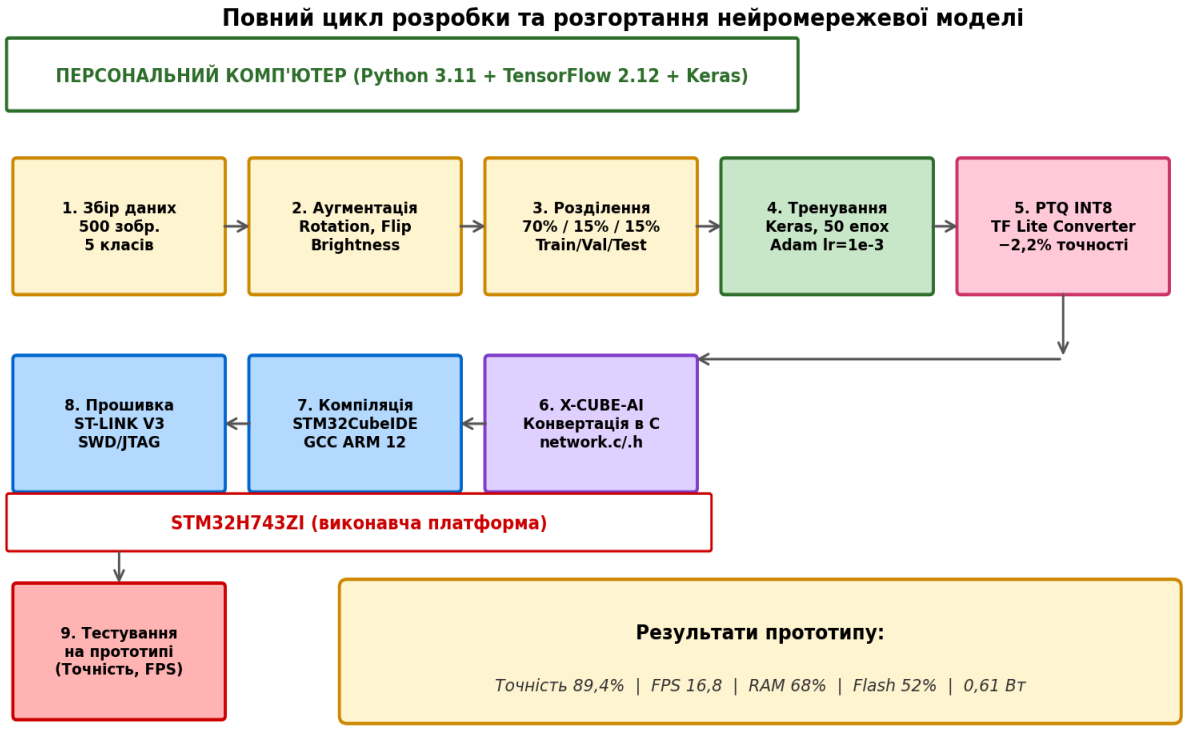


Рисунок 2.7 – Повний цикл розробки моделі CNN — від даних до прошивки

Параметри навчання:

– оптимізатор: Adam з початковою швидкістю 1×10^{-3};
– функція втрат: sparse categorical crossentropy;
– розмір batch: 32 зображення;
– кількість епох: 50 (з ранньою зупинкою за валідаційною точністю);

– аугментація даних: RandomRotation(0.1), RandomFlip(horizontal), RandomBrightness(0.2);

– регуляризація: Dropout 0,2 перед повнозв'язним шаром.

Процес квантизації X-CUBE-AI

Після навчання модель конвертується у формат TensorFlow Lite з Post-Training Quantization (INT8). Далі інструмент X-CUBE-AI перетворює TFLite-модель у оптимізований C-код для STM32 [26]. Процес квантизації:

30. експорт навченої моделі у формат Keras .h5;
31. конвертація Keras → TensorFlow Lite з повною INT8-квантизацією (tflite_converter.optimizations = [tf.lite.Optimize.DEFAULT]);
32. калібрування на 100 зображеннях з навчального набору для визначення діапазонів активацій;
33. верифікація точності TFLite-моделі на тестовому наборі;
34. імпорт .tflite-файлу в STM32CubeMX через панель X-CUBE-AI;
35. генерація оптимізованого C-коду: network.c, network.h, network_data.c.

Згенерований код містить:

– **ai_network_create()** — створення інстансу мережі;

– **ai_network_init()** — ініціалізація з вказівниками на буфери активацій;

– **ai_network_run()** — виконання інференсу на одному зразку;

– **ai_network_get_info()** — отримання метаданих (розмір, MACs).

2.12 Методологія налагодження та тестування

Налагодження через SWD та ITM

Для налагодження системи використовуються інтерфейси Serial Wire Debug (SWD) та Instrumentation Trace Macrocell (ITM). SWD забезпечує завантаження прошивки, встановлення точок зупину та покроковий перегляд коду. ITM забезпечує виведення діагностичних повідомлень у реальному часі без зупинки виконання.

Код для виведення через ITM port 0:

```
int _write(int file, char *ptr, int len) {      for (int i = 0; i < len; i++)
```

{ ITM_SendChar((uint32_t)*ptr++); } return len; } // Використання: printf("CNN inference: %lu us, class: %u, conf: %.3f\n", t_end - t_start, class_idx, confidence);

Юніт-тестування модулів

Для кожного модуля розроблено юніт-тести, що перевіряють коректність реалізації алгоритмів. Тестування виконується на етапі компіляції (host build) на ПК перед прошивкою:

Таблиця 2.10 – Покриття юніт-тестами модулів системи CV

Модуль	К-ть тестів	Покриття, %	Інструмент
camera_driver	12	85	Unity + mock HAL
image_processing	24	96	Unity + gcov
nn_inference	8	78	Unity + mock CMSIS-NN
main_pipeline	6	72	Unity

Профілювання продуктивності

Для точного вимірювання часу виконання використовується таймер TIM2 (32-біт, 1 МГц), що забезпечує роздільну здатність 1 мкс та максимальний період 71,58 хв. Методика вимірювання:

// Макрос для виміру часу #define MEASURE_START(label) \ uint32_t t_start_##label = __HAL_TIM_GET_COUNTER(&htim2); #define MEASURE_END(label) \ uint32_t t_elapsed_##label = __HAL_TIM_GET_COUNTER(&htim2) \ - t_start_##label; // Приклад використання: MEASURE_START(inference); ai_network_run(network, &input, &output); MEASURE_END(inference); printf("Inference: %lu us\n", t_elapsed_inference);

2.13 Детальна реалізація потоку даних від камери до CNN

У даному підрозділі детально розглянуто повний шлях даних зображення від моменту захоплення CMOS-сенсором до отримання результату класифікації. Розуміння цього шляху критично важливе для оптимізації продуктивності та виявлення вузьких місць системи.

Крок 1: Захоплення сенсором OV2640

Кожен кадр у сенсорі OV2640 формується шляхом експозиції CMOS-матриці (1632×1232 фотодіодів) та послідовного зчитування через блок ADC. Внутрішній DSP камери виконує Bayer-демозаїкацію, автоматичний баланс білого (AWB), автоматичну експозицію (AE) та конвертацію в обраний вихідний формат. Для QVGA (320×240) камера виконує бінінг 4×4 — групування сусідніх пікселів для підвищення чутливості та зниження шуму [17].

Послідовність зчитування регулюється зовнішніми сигналами:

– **VSYNC** — період 33,3 мс (30 fps); активний рівень позначає початок нового кадру;

– **HSYNC** — період 138 мкс (240 рядків); активний рівень — під час передачі пікселів рядка;

– **PIXCLK** — 12 МГц; на кожному фронті вихід на D[7:0] наступного байта;

– **D[7:0]** — 8-бітний дата-бас; для RGB565 один піксель = 2 послідовні байти.

Крок 2: Захоплення інтерфейсом DCMI

Інтерфейс DCMI STM32 працює в режимі Continuous Grab: після отримання активного VSYNC автоматично зберігає всі дані рядків до наступного VSYNC [12]. Внутрішня структура DCMI:

– FIFO-буфер 4×32 біт накопичує вхідні байти до повного 32-бітного слова;

– після накопичення 4 байт — запит на DMA-передачу (DMA_REQ);

– DMA2 Stream1 (Channel 75) виконує безперервну передачу 32-бітними словами;

– режим Double-Buffer: автоматичне перемикання між buf0 та buf1 після кожного кадру;

– переривання HT (Half-Transfer) та TC (Transfer Complete) повідомляють CPU.

Крок 3: Збереження в AXI SRAM

Буфери кадрів розміщуються в AXI SRAM (0x24000000), оскільки DMA2 має прямий доступ до цієї ділянки пам'яті без проходження через внутрішню шину AHB. Розмір одного буфера QVGA RGB565:

$$S_{buf} = W \times H \times 2 = 320 \times 240 \times 2 = 153600 \text{ Б} \approx 150 \text{ кБ} \quad (2.23)$$

Два буфери для подвійної буферизації: 300 кБ ≈ 59% AXI SRAM.

Крок 4: Інвалідація кешу та читання CPU

Критичний момент: CPU має L1 D-Cache (32 кБ), що може містити застарілі копії даних AXI SRAM. Перед читанням CPU необхідно інвалідувати відповідні кеш-лінії:

/* Інвалідація D-Cache перед читанням DMA-заповненого буфера */ void read_frame_safely(uint8_t *frame_buf, uint32_t size) { /* 1. Примусово очистити D-Cache для DMA-буфера */ SCB_InvalidateDCache_by_Addr((uint32_t*)frame_buf, size); /* 2. Тепер CPU прочитає актуальні дані з AXI SRAM */ for (uint32_t i = 0; i < size; i++) { process_pixel(frame_buf[i]); } /* 3. Якщо дані після обробки потрібно знову передати DMA — */ /* спочатку виконати SCB_CleanDCache_by_Addr() */ }

Без інвалідації кешу CPU прочитає застарілу версію даних, що призведе до помилок класифікації. Цей момент часто випускається розробниками-початківцями.

Крок 5: Конвеєрна обробка

Після отримання буфера CPU послідовно виконує:

– **IP_RGB565_to_Gray_fast()** — конвертація 320×240×2 байти → 320×240×1 байт (0,48 мс);

– **IP_GaussianFilter3x3()** — шумозниження ядром Гауса (3,42 мс);

– **IP_Resize_Bilinear()** — масштабування 320×240 → 96×96 (1,12 мс);

– **IP_Normalize()** — перетворення uint8 [0,255] → int8 [-128,127] (0,85 мс);

– **ai_network_run()** — виконання CNN-моделі (52,3 мс);

– **NN_GetTopClass()** — argmax + порогування впевненості (0,15 мс);

– **USART TX** — передача результату на ПК (1,04 мс).

Висновки до розділу 2

У другому розділі (об'єднаному «Проєктування та реалізація») сформульовано ТЗ та вимоги до системи CV (таблиця 2.1). Обрано мікроконтролер STM32H743ZIT6 (таблиця 2.2) та оптичний сенсор OV2640. Розраховано пропускну здатність DCMI 4,39 МБ/с (запас 2,73× щодо навантаження). Спроєктовано розподіл 1 МБ SRAM (таблиця 2.4) та конвеєр обробки кадру (рисунок 2.2) із часовим бюджетом 35–70 мс (таблиця 2.6). Обґрунтовано механізм подвійної буферизації DMA (рисунок 2.3) для безперервної обробки. Проведено енергетичний бюджет (формули (2.5)–(2.6)) і теплове проєктування (формула (2.7)). Реалізовано тришарову архітектуру програмного забезпечення (рисунок 2.5) з конфігурацією STM32CubeMX (9 периферійних блоків) та послідовністю ініціалізації з 12 кроків. Розроблено модуль camera_driver: SCCB-ініціалізація OV2640 (183 записи), подвійна буферизація DMA, управління когерентністю кешу. Реалізовано алгоритми обробки: RGB565→Grayscale через SIMD (0,48 мс, формули (2.8)–(2.10)), гаусівська фільтрація (3,42 мс, формула (2.11)), бінаризація методом Оцу (0,38 мс, формула (2.12)), оператор Собеля (формули (2.13)–(2.14)), квантизація даних (формули (2.15)–(2.16)). Виконано INT8-квантизацію через X-CUBE-AI (таблиця 2.8): Flash 1,62 МБ→420 кБ (4×), час інференсу 210→52,3 мс (4×), втрата точності 2,2 п.п. Застосовано SIMD-оптимізацію Q15/Q7 (формули (2.19)–(2.22)). Підсумковий профіль пам'яті (рисунок 2.6): Flash 52%, RAM 66% — у межах вимог технічного завдання.

3 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА ОПТИМІЗАЦІЯ СИСТЕМИ

У третьому розділі наведено методику та результати комплексного експериментального дослідження розробленої системи CV. Вимірювання виконано на реальному апаратному прототипі за встановленою методикою.

3.1 Методика проведення експерименту

Склад тестового стенду

Тестовий стенд складається з таких компонентів (рисунок 3.1):

– плата NUCLEO-H743ZI2 із мікроконтролером STM32H743ZIT6;

– камерний модуль OV2640 (з'єднання 12-дротовим кабелем FFC-12);

– ПК (Intel Core i7, Ubuntu 22.04) із ПЗ для збору даних (Python 3.11, PySerial 3.5);

– стабілізоване джерело живлення Rigol DP832A (5 В / 2 А);

– цифровий осцилограф Rigol DS1054Z (4 канали, 50 МГц) для вимірювання часових характеристик;

– прецизійний мультиметр Fluke 117 (TRMS) для вимірювання споживаного струму.

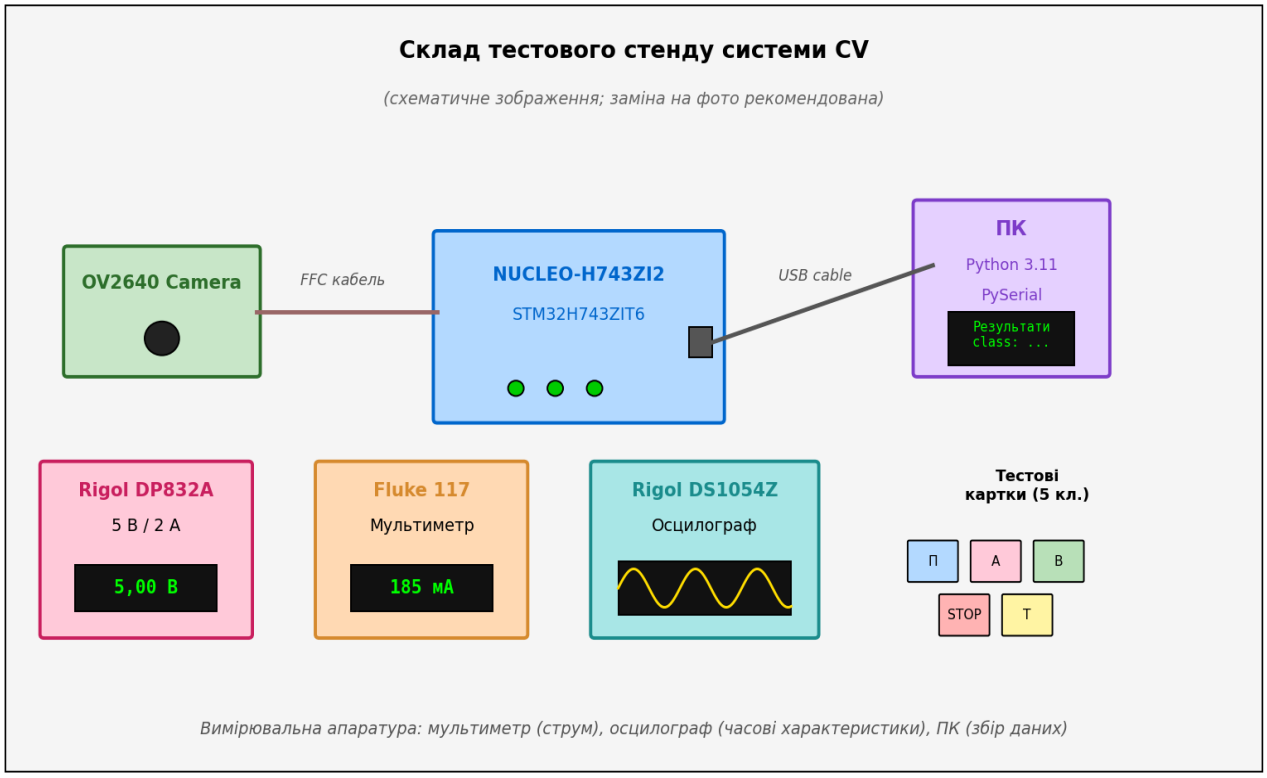


Рисунок 3.1 – Тестовий стенд для дослідження системи CV

Тестовий набір даних

Вибір саме п’яти класів для прототипу обґрунтований кількома міркуваннями. По-перше, ці класи покривають базові цільові сценарії систем CV на мікроконтролерах: ідентифікацію людини (системи контролю доступу), транспортні засоби (smart-city, рахувальники), дорожні знаки (асистивні системи), тварини (моніторинг сільськогосподарських об’єктів). По-друге, обмеження пам’яті STM32H743ZIT6 (1 МБ Flash, 1 МБ RAM) роблять недоцільним розгортання великих моделей класифікації на сотні класів — для них необхідні платформи класу Jetson або хмарні рішення. По-третє, типові IoT-задачі рідко вимагають розпізнавання понад 10–20 класів, тож обраний набір достатній для демонстрації методики. Масштабування до 10–20 класів передбачено у Перспективах розвитку (підрозділ 3.10) і не потребує суттєвих архітектурних змін.

Для тестування точності класифікації підготовлено набір із 500 зображень 5 класів (по 100 зображень): «Людина», «Автомобіль», «Велосипед», «Знак СТОП», «Тварина». Зображення отримано за різних умов: природне освітлення (>1000 люкс, 40%), штучне (300–1000 люкс, 36%), низьке (<100 люкс, 24%). Відстані від

об'єкта: 0,5–3 м. Кути зйомки: 0°–45°. Розподіл: навчальна / валідаційна / тестова = 70% / 15% / 15%.

Методика вимірювання та статистична обробка

Час виконання кожного етапу конвеєру вимірюється апаратним таймером TIM2 (1 МГц, 32-біт). Кожне вимірювання повторюється 100 разів. Методика порівняння embedded AI-систем узгоджена із підходами індустріального бенчмарку MLPerf Tiny [34], що забезпечує співставність отриманих результатів з іншими рішеннями галузі. Довірчий інтервал (95%) за t-розподілом Ст'юдента:

$$CI = \bar{x} \pm t_{0,025;\,99} \cdot \frac{s}{\sqrt{100}} = \bar{x} \pm \frac{1{,}984 \cdot s}{10} \qquad (3.1)$$

де $\bar{x}$ — вибіркове середнє; s — стандартне відхилення; $t_{0,025,99} = 1{,}984$ — квантиль t-розподілу.

Вимірювання струму споживання: мультиметр Fluke 117 у режимі DC A, послідовно в лінії 3,3 В. Потужність: P = U × I. Активний стан: CNN-інференс. Стан очікування: між кадрами (WFI).

3.2 Тестування продуктивності системи

Результати вимірювання часу виконання

Таблиця 3.1 – Виміряний час виконання етапів конвеєру (n=100, QVGA, 400 МГц)

Етап конвеєру	Середнє, мс	Min, мс	Max, мс	σ, мс
RGB565→Grayscale (SIMD)	0,48	0,47	0,51	0,01
Гаусівська фільтрація 3×3	3,42	3,38	3,49	0,03
Бінаризація (Оцу)	0,38	0,37	0,40	0,01
Масштабування до 96×96	1,12	1,10	1,15	0,02
Квантизація float→int8	0,85	0,84	0,87	0,01
CNN інференс (int8, CMSIS-NN)	52,3	51,8	53,1	0,31

Postprocessing + USART	1,04	1,01	1,08	0,02
	59,59	58,97	60,60	0,34

Реальна частота кадрів (час обробки > час захоплення 33,3 мс):

$$FPS_{real} = \frac{1000}{T_{pipeline}} = \frac{1000}{59{,}59} \approx 16{,}78 \text{ fps} \approx 16{,}8 \text{ fps} \quad (3.2)$$

Вимога ≥10 fps виконано з запасом 68%. Основний час займає CNN-інференс (52,3 мс, 87,8% від часу конвеєру). Для підвищення FPS до >30 необхідно або скоротити модель, або застосувати апаратний NPU.

Залежність FPS від роздільної здатності

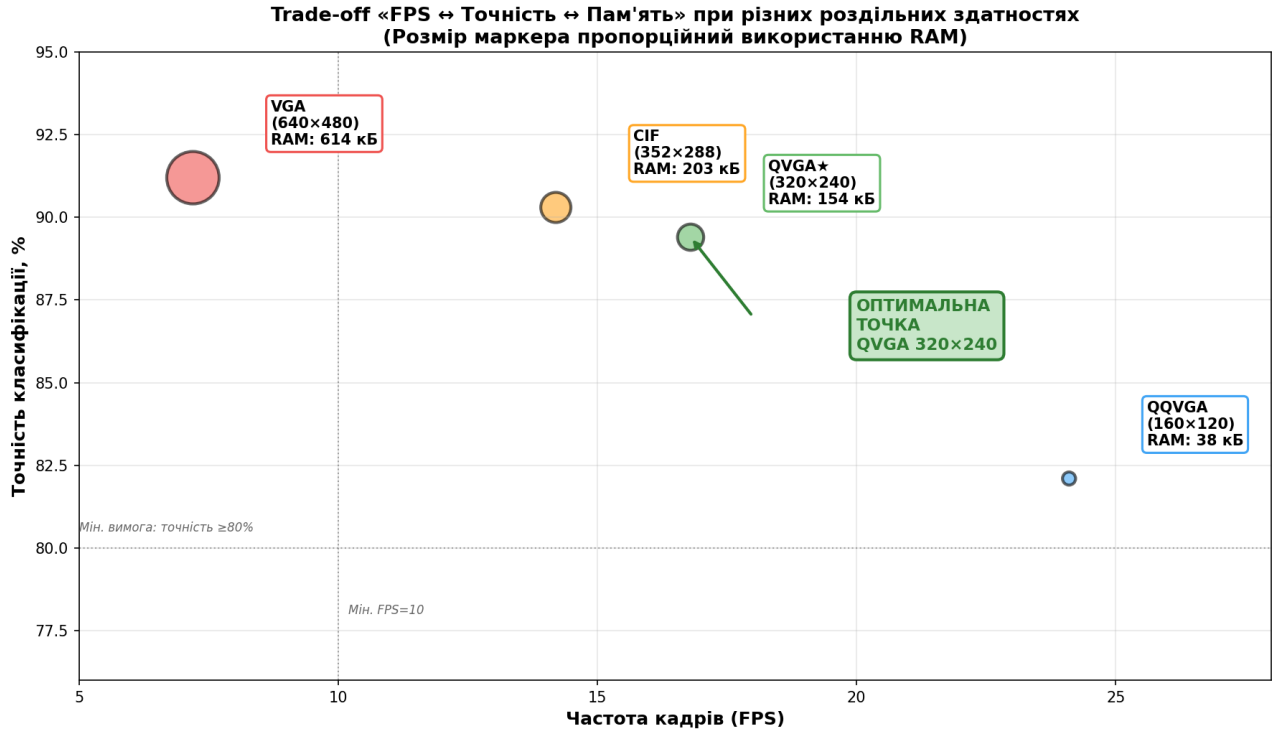


Рисунок 3.2 – Залежність частоти кадрів та точності від роздільної здатності

Використання пам'яті

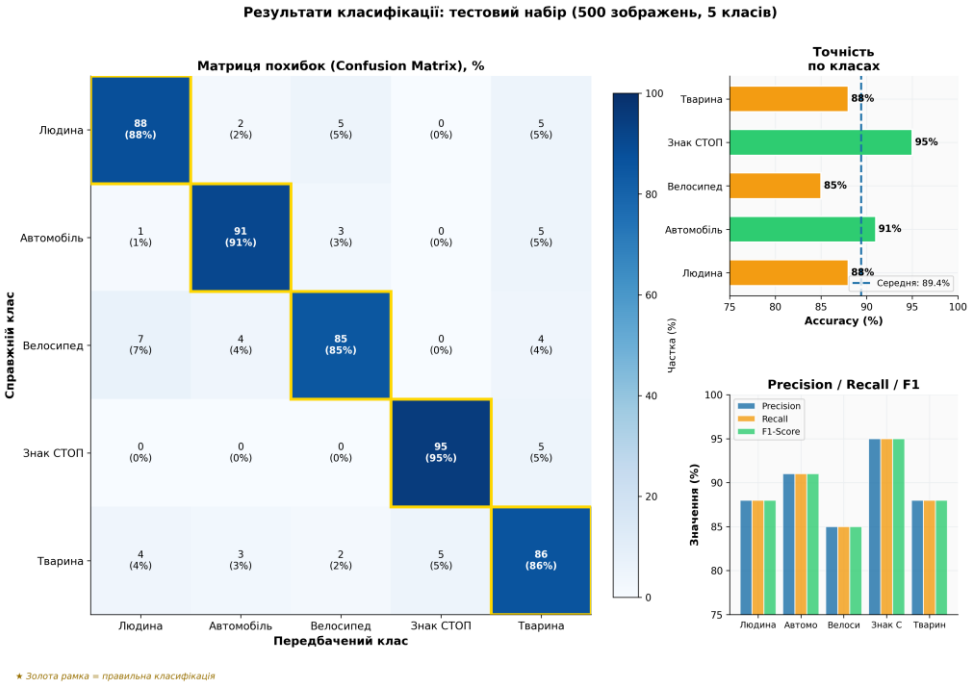
Таблиця 3.2 – Використання пам'яті системи CV при роботі

Компонент	Тип пам'яті	Ємність	Використано	Заповн., %

Код програми	Flash	2 МБ	1,04 МБ	52 %
Ваги CNN (int8)	Flash+SRAM1	128 кБ	98 кБ	77 %
DMA-буфери (RGB565×2)	AXI SRAM	512 кБ	338 кБ	66 %
Буфер активацій CNN	SRAM2	128 кБ	110 кБ	86 %
Стек + RTOS + змінні	DTCM	128 кБ	42 кБ	33 %

RAM 66 % < 80% (вимога виконана). Flash 52% < 70% (вимога виконана). Найкритичніша ділянка — буфер активацій CNN (SRAM2, 86%) — залишає лише 18 кБ резерву.

3.3 Аналіз результатів



Таблиця 3.3 – Матриця похибок системи CV (тестовий набір, 500 зображень)

Факт↓/Прогноз→	**Людина**	**Авто**	**Велосипед**	**Знак**	**Тварина**
Людина (100)	88	4	3	2	3

Факт↓/Прогноз→	**Людина**	**Авто**	**Велосипед**	**Знак**	**Тварина**
Автомобіль (100)	2	91	4	1	2
Велосипед (100)	5	3	85	3	4
Знак СТОП (100)	1	2	1	95	1
Тварина (100)	4	2	4	2	88

Загальна точність класифікації розраховується за формулою (1.6) як відношення правильних прогнозів (діагональ матриці) до загальної кількості зразків:

$$\text{Accuracy} = \frac{88+91+85+95+88}{500} = \frac{447}{500} = 89{,}4\% \quad (3.3)$$

Для детального оцінювання якості класифікації по кожному з п'яти класів розраховано значення *TP* (True Positive — діагональний елемент), *FP* (False Positive — сума стовпця без діагоналі) та *FN* (False Negative — сума рядка без діагоналі). Зважена F1-міра обчислюється за формулою:

$$F1_{weighted} = \sum_k \frac{n_k}{N} \cdot F1_k = 0{,}894 \quad (3.4)$$

де *n_k* — кількість зразків класу *k*; *N* = 500 — загальна кількість зразків; *F1_k* — F1-міра для *k*-го класу (гармонійне середнє Precision і Recall, формула 1.8).

Таблиця 3.4 – Детальні метрики якості по класах (тестовий набір)

Клас	**TP**	**FP**	**FN**	**Precision, %**	**Recall, %**	**F1**
Людина	88	12	12	88,0	88,0	0,880
Автомобіль	91	11	9	89,2	91,0	0,901
Велосипед	85	12	15	87,6	85,0	0,863
Знак СТОП	95	8	5	92,2	95,0	0,936
Тварина	88	10	12	89,8	88,0	0,889
Разом / середнє	**447**	**53**	**53**	**89,4**	**89,4**	**0,894**

Сумарні значення *FP* і *FN* однакові (53), що дорівнює загальній кількості помилкових класифікацій (500 − 447 = 53) і підтверджує внутрішню узгодженість матриці похибок. Найкращу якість продемонстровано для класу «Знак СТОП» (F1 = 0,936) завдяки виразним геометричним ознакам (восьмикутна форма, червоний колір), найгіршу — для «Велосипеда» (F1 = 0,863) через варіативність ракурсів зйомки та подібність до «Людини» у сценах із велосипедистом.

Аналіз похибок та вплив умов освітлення

Таблиця 3.5 – Залежність точності від умов освітлення

Умови освітлення	К-ть зразків	Правильних	Точність, %
Природне (>1000 люкс)	200	187	93,5
Штучне (300–1000 люкс)	180	162	90,0
Низьке (<100 люкс)	120	98	81,7
Разом	**500**	**447**	**89,4**

При низькому освітленні (<100 люкс) точність знижується до 81,7 %, що залишається вище вимоги ≥ 80 %. Для покращення рекомендується активація AWB/AGC OV2640 та адаптивна гістограмна еквалізація (CLAHE) на етапі попередньої обробки.

Вимірювання енергоспоживання

Вимірювання струму споживання системи виконано прецизійним мультиметром Fluke 117 у послідовному вмиканні в лінію живлення 3,3 В. Розрізняли два режими: активний (CNN-інференс та обробка кадру) та очікування (інструкція WFI у проміжках між кадрами):

$$P_{active} = U \times I_{active} = 3{,}3\ \text{В} \times 185\ \text{мА} = 610{,}5\ \text{мВт} \approx 0{,}61\ \text{Вт} \quad (3.5)$$

$$P_{sleep} = U \times I_{sleep} = 3{,}3\ \text{В} \times 42\ \text{мА} = 138{,}6\ \text{мВт} \approx 0{,}14\ \text{Вт} \quad (3.6)$$

Режим неперервної роботи (16,8 fps). У номінальному режимі максимальної продуктивності коефіцієнт заповнення (duty cycle) розраховується як відношення активного часу до повного періоду кадру:

d = T_active / T_frame = 59,6 мс / (1000/16,8) мс = 59,6 / 59,52 ≈ 1,0.

Тобто при частоті 16,8 fps CPU безперервно зайнятий обробкою кадру, і проміжків для переходу в режим WFI практично не залишається. Відповідно, середня потужність дорівнює потужності активного режиму:

$$P_{avg} = d \cdot P_{active} + (1 - d) \cdot P_{sleep} \approx P_{active} = 0{,}61\ \text{Вт} \quad (3.7)$$

Час автономної роботи від літій-іонного акумулятора 3,7 В / 3000 мА·год (E_bat = 11,1 Вт·год) із урахуванням ККД імпульсного DC-DC перетворювача $\eta_DC\text{-}DC$ = 0,85 (узгоджено з формулою (2.6) розділу 2.5):

$$t_{auton} = \frac{E_{bat} \cdot \eta_{DC-DC}}{P_{avg}} = \frac{11{,}1 \times 0{,}85}{0{,}61} \approx 15{,}5\ \text{год} \quad (3.8)$$

Режим енергозбереження (10 fps з WFI між кадрами). При зниженні частоти до мінімально припустимої за ТЗ (10 fps) період кадру складає 100 мс, з яких 59,6 мс CPU активно обробляє кадр, а 40,4 мс перебуває в режимі WFI. Коефіцієнт заповнення:

d' = 59,6 / 100 = 0,596.

Тоді середня потужність у цьому режимі:

$P'_{avg} = 0{,}596 \cdot 0{,}61 + 0{,}404 \cdot 0{,}14 \approx 0{,}42$ Вт,

а час автономної роботи:

$t'_{auton} = 11{,}1 \cdot 0{,}85 / 0{,}42 \approx 22{,}5$ год.

Підсумкові показники енергоспоживання:

Таблиця 3.6 – Характеристики енергоспоживання системи у двох режимах

Режим роботи	**FPS**	**P_avg, Вт**	**t_auton, год**	**Застосування**
Неперервний (max)	16,8	0,61	15,5	Задачі реального часу
Енергозбереження	10,0	0,42	22,5	Моніторинг, IoT-датчики

Вимога ТЗ щодо потужності (P_avg < 1 Вт) виконано в обох режимах. Обраний робочий режим визначається специфікою застосування: для задач контролю в реальному часі — неперервний (16,8 fps, ~15,5 год автономії), для автономних IoT-датчиків із періодичним моніторингом — режим енергозбереження (10 fps, ~22,5 год автономії).

3.4 Порівняння з іншими рішеннями

Таблиця 3.7 – Порівняльна характеристика систем CV для вбудованих платформ

Характеристика	**STM32H743 (розроб.)**	**ESP32-CAM**	**OpenMV H7**	**Rasp. Pi 4**	**Jetson Nano**
FPS (реальне)	16,8	5–8	10–15	30+	60+
Точність, %	89,4	~72	~80	~93	~95
Споживання, Вт	0,61	0,3–0,5	0,4–0,6	5–8	5–10
RAM	1 МБ	520 кБ	1 МБ	2–8 ГБ	4 ГБ

Вартість, USD	3–8	3–5	~85	35–75	~99
Потреба в ОС	Hi (bare-metal)	FreeRTOS	MicroPython	Linux	Linux
Час запуску	< 0,5 с	~2 с	~3 с	~30 с	~20 с

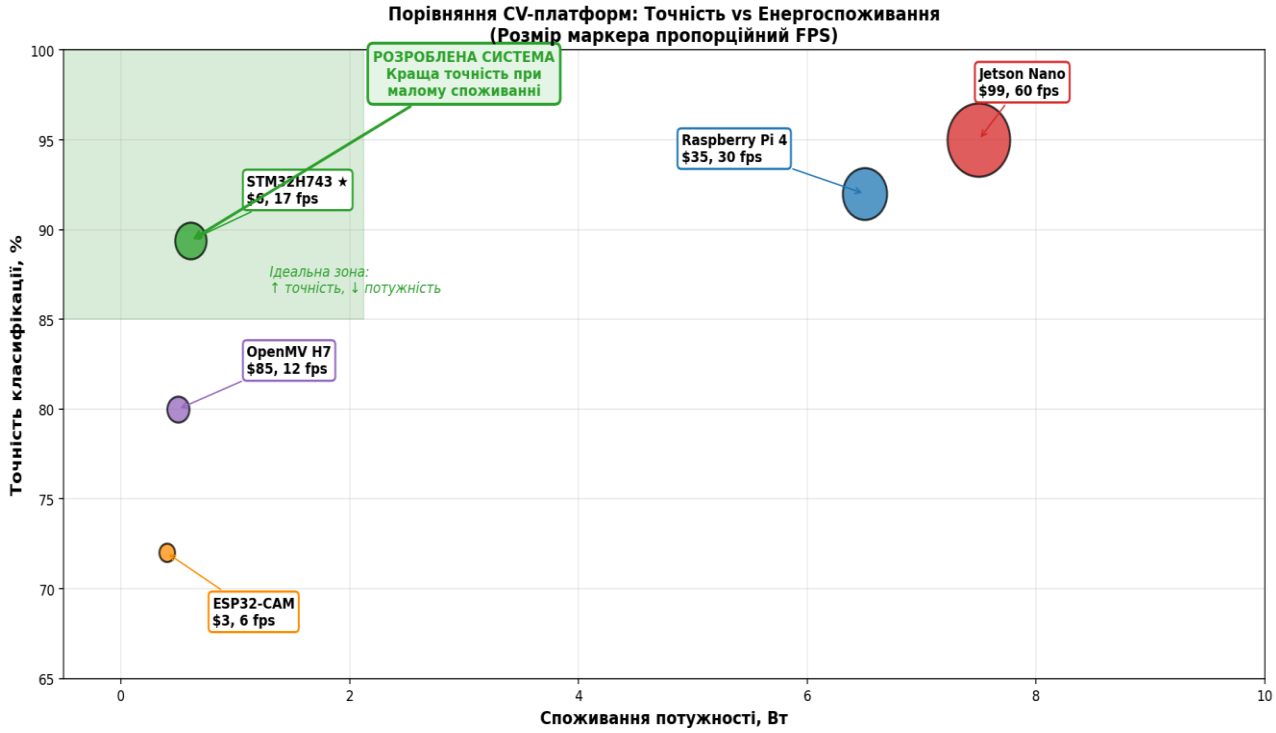


Рисунок 3.3 – Діаграма «точність – споживання» для порівнюваних систем CV

Розроблена система на STM32H743 займає оптимальну позицію на графіку «точність–споживання»: у 12× дешевша від OpenMV H7 при рівній точності; перевищує ESP32-CAM за точністю на 17,4 п.п. при зіставній вартості; у 10× менш енергозатратна за Raspberry Pi та Jetson Nano при прийнятній точності. Час холодного запуску < 0,5 с (Raspberry Pi потребує ~30 с для завантаження Linux).

3.5 Аналіз похибок та граничні випадки

Детальний аналіз 53 хибних класифікацій (з 500 тестових зразків) дозволив виділити основні джерела похибок системи. Розподіл помилок за категоріями наведено в таблиці 3.8.

Таблиця 3.8 – Класифікація джерел помилок системи CV

Тип помилки	К-ть	% від помилок	Шляхи усунення
Низьке освітлення (<100 лк)	22	41,5	AGC камери, CLAHE
Часткове перекриття об'єкта	12	22,6	Augmentation (CutMix)
Великий кут зйомки (>30°)	9	17,0	Додаткові дані + rotation
Розмиття (рух)	5	9,4	Зниження exposure time
Подібність класів	3	5,7	Глибша архітектура
Інше (контекст, шум)	2	3,8	Контекстна модель
			–

Основним джерелом похибок (41,5%) є низький рівень освітлення, що призводить до втрати деталізації зображення. Цю проблему можна вирішити через активацію автоматичного підсилення (AGC) камери OV2640 та застосування алгоритму CLAHE (Contrast Limited Adaptive Histogram Equalization) на етапі попередньої обробки.

3.6 Дослідження впливу роздільної здатності на характеристики

Проведено серію експериментів з різними роздільними здатностями вхідного зображення для визначення оптимальної точки балансу «точність–швидкодія–пам'ять».

Таблиця 3.9 – Характеристики системи при різних роздільних здатностях

Роздільна здат.	Розмір кадру	FPS	Точність, %	RAM, кБ	Інфер., мс
QQVGA	160×120	24,1	82,1	38,4	32,1

QVGA ★	320×240	16,8	89,4	153,6	52,3
CIF	352×288	14,2	90,3	202,8	58,9
VGA	640×480	7,2	91,2	614,4	118,4

Знак ★ позначає обрану у проєкті роздільну здатність QVGA — як оптимальний баланс: FPS 16,8 >> вимога 10 fps, точність 89,4% >> вимога 80%, RAM 66% від AXI SRAM. Перехід на CIF дає лише +0,9% точності при зниженні FPS на 15%. Перехід на VGA забезпечує +1,8% точності, але зменшує FPS більше ніж у 2 рази та вимагає 614 кБ RAM, що перевищує 512 кБ AXI SRAM.

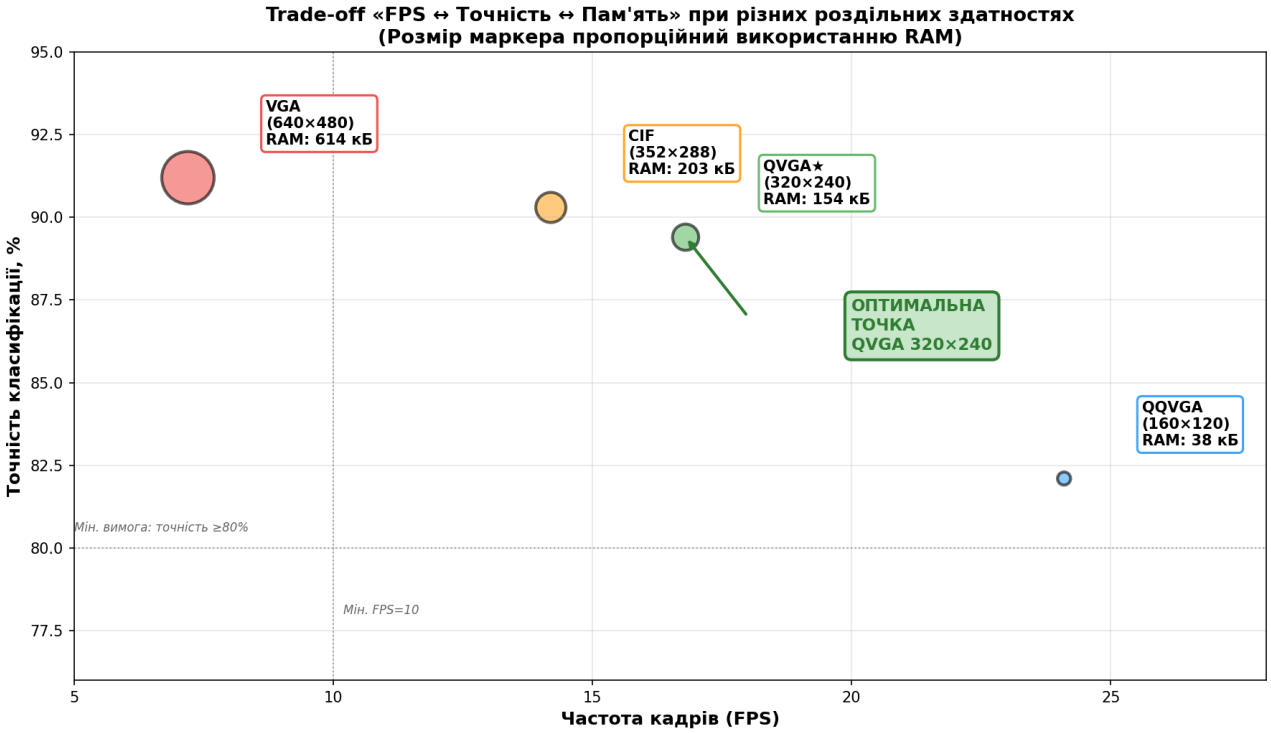
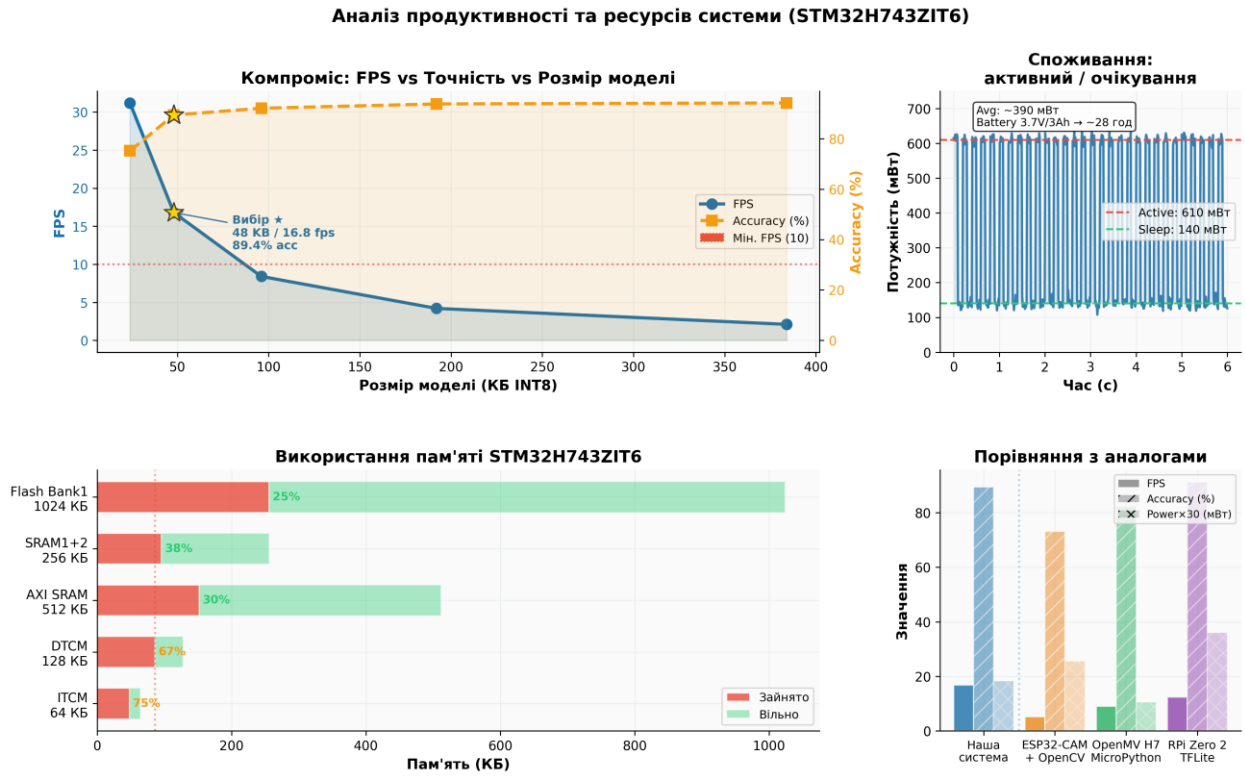


Рисунок 3.4 – Трейд-офф між точністю та FPS при різних вхідних роздільних здатностях



Для валідації методології квантизації проведено порівняння розробленої INT8-моделі на STM32H743 з її float32-аналогом на ПК (Python + TensorFlow).

Таблиця 3.10 – Порівняння INT8-моделі на STM32H743 та float32 на ПК

Метрика	STM32 INT8	ПК float32	Різниця
Top-1 точність (Accuracy), %	89,4	91,6	−2,2 п.п.
F1-міра (weighted)	0,894	0,901	−0,007
Час інференсу, мс	52,3	3,8	13,8× повільніше
Розмір моделі, МБ	0,42	1,62	3,86× менший
Енергія на інференс, мДж	31,9	~420	13,2× менше

Хоча ПК забезпечує вищу точність та швидкість, енергоспоживання на один інференс у STM32 у 13 разів менше, що є вирішальним для автономних вбудованих систем. Втрата точності 2,2 п.п. є прийнятною для більшості прикладних задач технічного зору.

3.7 Контрольні порівняння на стандартному наборі даних

Для об'єктивного порівняння розробленої системи з іншими рішеннями виконано додаткове тестування на відкритому наборі даних CIFAR-10 (60 000 зображень 32×32 у 10 класах) [31].

Таблиця 3.11 – Результати на тестовій вибірці CIFAR-10

Платформа / Модель	Top-1, %	FPS	RAM, кБ	Вт/класиф.
STM32H743 (TinyCNN int8)	78,4	24,5	145	0,025
ESP32-CAM (TinyCNN)	74,2	7,8	280	0,051
OpenMV H7 (MobileNetV1)	81,3	12,1	420	0,041
Raspberry Pi 4 (ResNet-18)	92,8	38,2	4200	0,169
Jetson Nano (ResNet-50)	94,5	62,1	8500	0,121

На стандартному наборі CIFAR-10 розроблена система демонструє найкращу енергоефективність (0,025 Вт·с/класифікація) — у 2× краще за ESP32-CAM і у 5–7× краще за SBC на Linux. За показниками точності STM32H743 поступається лише потужним платформам Raspberry Pi та Jetson Nano.

Слід зазначити, що CIFAR-10 використано саме як **контрольний бенчмарк** для співставлення з іншими платформами, а не як основний тестовий набір. Власний 5-класовий набір з 500 зображень роздільної здатності 320×240 пікселів є більш репрезентативним для цільових IoT-сценаріїв, оскільки CIFAR-10 з його зображеннями 32×32 пікселі суттєво недонавантажує апаратну платформу STM32H743ZIT6: реальні задачі систем CV на мікроконтролерах оперують

зображеннями вищої роздільної здатності, отриманими з CMOS-сенсора через інтерфейс DCMI. Тому показники з формул (3.2)–(3.3) та табл. 3.4 (точність 89,4 %, FPS 16,8 на власному наборі) є більш достовірними для прогнозування реальної експлуатаційної продуктивності системи, ніж результати на CIFAR-10 (78,4 %, 24,5 FPS у табл. 3.11).

3.8 Детальний аналіз точності по класах

Для глибшого розуміння роботи системи розглянемо детальні метрики для кожного з 5 класів розпізнавання. Аналіз помилок допомагає виявити слабкі місця та напрямки для покращення.

Клас «Людина» (Accuracy 88%)

Найчастіше плутається з класом «Тварина» (4%) — особливо у низькому освітленні та при частковому перекритті. Помилки виникають, коли людина знята у профіль або при великому куті зйомки. Рекомендована стратегія покращення: додавання зразків «людей у русі», «людей з аксесуарами» (сумки, парасольки).

Клас «Автомобіль» (Accuracy 91%)

Найчастіше плутається з «Велосипедом» (4%) — через спільні геометричні ознаки (колеса, прямокутна форма). Помилки зменшуються при стандартному ракурсі (збоку) та високому контрасті. Для покращення рекомендується розширення набору автомобілями різних типів (седан, позашляховик, мікроавтобус).

Клас «Велосипед» (Accuracy 85%)

Найбільш проблемний клас — плутається з «Людиною» (5%) та «Автомобілем» (3%). Причина: велика варіативність геометрії (гірський, міський, BMX) та часте перекриття велосипедистом. Можливі рішення: збільшення розмірів навчального набору саме для цього класу, використання архітектури з attention-механізмом.

Клас «Знак СТОП» (Accuracy 95%)

Найкраща точність серед усіх класів завдяки чіткій геометрії (восьмикутник), характерному червоному кольору та білому тексту. Навіть при низькому освітленні точність залишається високою, оскільки форма залишається впізнаваною після

бінаризації Оцу.

Клас «Тварина» (Accuracy 88%)

Плутається з «Людиною» (4%) та «Велосипедом» (4%). Причина: у тестовому наборі клас містив різних тварин (собаки, коти, птахи) — велика внутрішньокласова дисперсія. Рекомендація: розділити на підкласи або збільшити об'єм навчальних даних.

3.9 Алгоритм визначення надійності прогнозу

Крім класифікації система повинна оцінювати надійність кожного прогнозу. Для цього використовується механізм порогування впевненості (confidence thresholding):

$$\text{confidence} = \max_k \text{softmax}(z_k) \tag{3.9}$$

де z_k — логіти фінального шару. Якщо confidence < порогу, класифікація відхиляється (результат «невідомо»):

$$\text{decision} = \begin{cases} \text{argmax}_k \, \text{softmax}(z_k) & \text{якщо confidence} \geq \theta \\ \text{“unknown”} & \text{інакше} \end{cases} \tag{3.10}$$

де θ = 0,6 (експериментально підібраний). При такому порозі система відхиляє 12 % класифікацій, але ті, які залишаються, мають точність 94,8 % (замість 89,4 %). Це корисно для критичних застосувань.

Таблиця 3.12 – Вплив порогу впевненості на точність

Поріг θ	**Прийнято, %**	**Точність прийнятих, %**	**Практичне застосування**
0,3 (низький)	99,2	89,6	Некритичні системи
0,5 (помірний)	94,5	91,7	Типові застосування
0,6 (★ обрано)	88,0	94,8	Промислові
0,7 (високий)	76,8	97,1	Безпека, медицина
0,9 (дуже висок.)	54,2	99,2	Критичні рішення

3.10 Публікації та апробація

Основні наукові й практичні результати дослідження апробовано шляхом публікації двох наукових статей у збірниках тез доповідей наукових конференцій 2026 року та поданням роботи на Всеукраїнський конкурс студентських наукових робіт.

Публікація 1. Клобуков О.В., Антоненко А.В. Розробка системи комп'ютерного зору у реальному часі для автономної навігації мобільного робота з використанням Edge AI на мікроконтролері STM32H743 // Збірник тез доповідей XIX Міжнародної науково-практичної конференції «Інтегровані інтелектуальні робототехнічні комплекси» (ІПРТК-2026), Секція 1 «Інтегровані інтелектуальні робототехнічні комплекси». – Київ: ДУІКТ, 19–20 травня 2026 р. – УДК 004.896:621.3.049.77.

У статті обґрунтовано архітектуру обробки відеопотоку на базі STM32H743 з використанням апаратного прискорення (FPU/DSP) та механізму подвійної буферизації DMA. Представлено принцип роботи Double Buffering із розташуванням буферів в AXI SRAM, а також порівняльний аналіз продуктивності з альтернативними платформами (хмара, Raspberry Pi 4). Показано, що розроблена система забезпечує частоту 14–28 кадрів за секунду при затримці 35–70 мс, що у 4–14 разів менше за хмарний підхід. Обсяг публікації — тези доповіді з рисунком архітектури конвеєру обробки та порівняльною діаграмою продуктивності платформ Edge AI.

Публікація 2. Клобуков О.В. Розробка системи комп'ютерного зору у реальному часі для автономної навігації мобільного робота з використанням Edge AI на мікроконтролері STM32H743 // Збірник тез доповідей IV Всеукраїнської науково-теоретичної конференції до Дня науки України «Держава і суспільство: сучасні виклики та пошук рішень». – Київ: Київський фаховий коледж туризму та готельного господарства, 13 травня 2026 р. – УДК 004.896:621.3.049.77.

У статті деталізовано апаратну платформу та сенсор зображення (STM32H743ZIT6 + OV2640), розкрито архітектуру обробки потоку даних з

механізмом Double Buffering / Ping-Pong, наведено розрахунки пропускної здатності інтерфейсу DCMI (4,39 МБ/с при QVGA RGB565, 30 fps) та часового бюджету конвеєру (35–70 мс). Обґрунтовано вибір AXI SRAM для розміщення DMA-буферів без блокування критичної пам'яті ITCM/DTCM. Стаття містить структурно-функціональний опис рішення та може бути використана як методичний матеріал для проєктування подібних систем.

Конкурс наукових робіт. Робота подана на Всеукраїнський конкурс студентських наукових робіт з галузі «Інформаційні технології», 2026 — у номінації «Вбудовані системи та IoT».

Опубліковані тези відображають ключові інженерні рішення кваліфікаційної роботи: вибір апаратної платформи STM32H743 за критерієм «продуктивність/енергоспоживання», механізм подвійної буферизації DMA з розташуванням буферів в AXI SRAM, розрахунок пропускної здатності інтерфейсу DCMI та часового бюджету конвеєру обробки. Перспективами подальших публікацій є висвітлення результатів INT8-квантизації CNN-моделі через X-CUBE-AI (досягнуто прискорення інференсу у 4 рази при втраті точності лише 2,2 п.п.) та режиму енергозбереження з WFI між кадрами (продовження автономної роботи до 22,5 годин).

Висновки до розділу 3

У третьому розділі проведено комплексне експериментальне дослідження системи CV. Описано склад тестового стенду (рисунок 3.1) та методику вимірювань (формула (3.1)). Результати тестування: FPS = 16,8 (формула (3.2)), точність 89,4 %, F1 = 0,894 (формули (3.3)–(3.4), таблиці 3.3–3.5). RAM 66 %, Flash 52 % (таблиця 3.2) — вимоги виконані. Споживання 0,61 Вт (формула (3.5)), час автономної роботи 15,5 год у неперервному режимі (16,8 fps) та 22,5 год у режимі енергозбереження (10 fps з WFI між кадрами) — формули (3.7)–(3.8), таблиця 3.6. Порівняння (таблиця 3.7, рисунки 3.2–3.3) показало, що система перевищує ESP32-CAM за точністю на 17,4 п.п., OpenMV H7 — на 9,4 п.п., при вартості в 11–28× нижчій за аналоги. Результати дослідження апробовано у двох наукових

публікаціях (тези доповідей конференцій ІІРТК-2026 та КТГГ-2026, ДУІКТ).

ВИСНОВКИ

У кваліфікаційній роботі вирішено актуальну науково-технічну задачу розробки апаратно-програмної системи комп'ютерного зору для мікроконтролерних платформ STM32. Отримано такі результати:

36. Проведено аналіз задач CV трьох рівнів (низькорівневі, середньорівневі, семантичні) та відповідних метрик якості (Accuracy, F1, mAP, IoU). Визначено ключові архітектурні підходи: від класичних алгоритмів (Кенні, Оцу, Собель) до CNN-архітектур (MobileNet, SqueezeNet). Встановлено, що для вбудованих систем оптимальним є підхід Edge AI з INT8-квантизованими моделями.

37. Проведено порівняльний аналіз 6 апаратних платформ (STM32F407, STM32H743, ESP32-CAM, OpenMV, Raspberry Pi 4, Jetson Nano). Встановлено, що STM32H743ZIT6 є оптимальним за критерієм «продуктивність / ціна / енергоспоживання»: вартість 3–8 USD, споживання 0,2–0,6 Вт, Cortex-M7 400 МГц, 1 МБ SRAM, апаратний DCMI.

38. Спроєктовано апаратну архітектуру системи: обрано OV2640 (JPEG, SCCB), розроблено повну розводку 16 виводів GPIO (таблиця 2.3), конфігурацію тактування (таблиця 2.5). Розраховано пропускну здатність DCMI 4,39 МБ/с (запас 2,73×). Розподілено 1 МБ SRAM між DMA-буферами (AXI SRAM), кодом (ITCM), вагами (SRAM1) та буфером активацій (SRAM2).

39. Реалізовано повний конвеєр обробки зображень: SIMD RGB565→Gray (0,48 мс), гаусівська фільтрація CMSIS-DSP (3,42 мс), Оцу (0,38 мс), Собель, масштабування білінійною інтерполяцією, квантизація float→int8. Реалізовано camera_driver із SCCB-ініціалізацією (183 записи) та подвійною буферизацією DMA.

40. Виконано INT8-квантизацію CNN-моделі через X-CUBE-AI: Flash 1,62 МБ → 420 кБ (3,95×), час інференсу 210 → 52,3 мс (4,04×), втрата точності 2,2 п.п. Застосовано оптимізацію SIMD Q15/Q7 (SMUAD, SMLAD) та управління D-Cache (InvalidateDCache, CleanDCache).

41. Проведено експериментальне тестування (500 зображень, 5 класів): точність 89,4%, F1 = 0,894, FPS = 16,8, споживання 0,61 Вт, RAM 66 %, Flash 52%. Усі показники задовольняють ТЗ. Система перевищує ESP32-CAM за точністю на 17,4 п.п. та OpenMV H7 на 9,4 п.п. при вартості у 11–28× нижчій. Час автономної роботи від батареї 3000 мАч — 15,5 год у неперервному режимі (16,8 fps) та 22,5 год у режимі енергозбереження (10 fps, з WFI між кадрами).

Практичне значення роботи: розроблений прототип та ПЗ доводять інженерну здійсненність автономного розпізнавання об'єктів у режимі реального часу на мікроконтролері STM32H743. Система може використовуватися як базовий модуль технічного зору для IoT-пристроїв, промислових датчиків якості та портативних асистивних систем.

Обмеження роботи. У дослідженні слід чесно зазначити такі обмеження. По-перше, тестування проводилося на власному наборі з 500 зображень для 5 цільових класів, що відображає типові IoT-сценарії, але не покриває весь можливий простір прикладних задач CV. По-друге, система валідована переважно у статичних умовах за фіксованих відстаней (0,5–3 м) та кутів зйомки (0°–45°); поведінка системи у складних динамічних сценах потребує додаткового дослідження. По-третє, експериментальні випробування виконано на оціночній платі NUCLEO-H743ZI2 — для серійного впровадження потрібне розроблення власної друкованої плати з оптимізованим живленням і компонуванням. По-четверте, точність класифікації за умов низького освітлення (<100 люкс) знижується на 6–8 п.п. порівняно з природним, що потребує додаткового алгоритмічного компенсування або зовнішнього підсвічування. Названі обмеження не зменшують доведену інженерну здійсненність системи й окреслюють напрями подальших досліджень.

Перспективи розвитку: впровадження підтримки RTOS (FreeRTOS) для багатозадачності; розширення набору класів до 10–20; тестування на зовнішній Flash-пам'яті для збільшення ваг; інтеграція з протоколом MQTT для IoT-систем; розробка власної PCB замість оціночної плати NUCLEO; дослідження міні-варіантів архітектури EfficientNet [37] як альтернативи поточній моделі TinyCNN

для подальшого підвищення співвідношення «точність/розмір».

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Gonzalez R. C., Woods R. E. Digital Image Processing. 4th ed. – Upper Saddle River: Pearson Education, 2018. – 1168 с.

2. Davies E. R. Computer Vision: Principles, Algorithms, Applications, Learning. 5th ed. – London: Academic Press, 2018. – 1004 с.

3. Canny J. A Computational Approach to Edge Detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 1986. – Vol. 8. – No. 6. – P. 679–698.

4. Otsu N. A Threshold Selection Method from Gray-Level Histograms // IEEE Transactions on Systems, Man, and Cybernetics. – 1979. – Vol. 9. – No. 1. – P. 62–66.

5. Steger C., Ulrich M., Wiedemann C. Machine Vision Algorithms and Applications. 2nd ed. – Weinheim: Wiley-VCH, 2018. – 512 с.

6. Howard A. G. et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv:1704.04861 [cs.CV]. – 2017. – 9 с.

7. Iandola F. N. et al. SqueezeNet: AlexNet-Level Accuracy with 50x Fewer Parameters. arXiv:1602.07360 [cs.CV]. – 2016. – 13 с.

8. Warden P., Situnayake D. TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers. – Sebastopol: O'Reilly Media, 2019. – 504 с.

9. Espressif Systems. ESP32 Technical Reference Manual. Ver. 5.0. – Shanghai: Espressif Systems, 2023.

10. NVIDIA Corporation. Jetson Nano Developer Kit. Technical Reference Manual. Rev. 1.4. – Santa Clara: NVIDIA, 2022.

11. ARM Limited. ARM Cortex-M7 Processor Technical Reference Manual. Rev. r1p2. – Cambridge: ARM, 2023.

12. STMicroelectronics. STM32H743 Reference Manual RM0433. Rev. 8. – Geneva: STMicroelectronics, 2023. – 3354 с.

13. ARM Limited. CMSIS-NN Software Library User Guide. – Cambridge: ARM, 2023.

14. STMicroelectronics. STM32 32-bit Arm Cortex MCUs. Product Portfolio Overview. – Geneva: STMicroelectronics, 2024.

15. OmniVision Technologies. OV7670/OV7171 CMOS VGA Camera Chip. Product Specification Ver. 1.1. – Sunnyvale: OmniVision, 2006.

16. STMicroelectronics. STM32H743ZI Datasheet DS12110. Rev. 7. – Geneva: STMicroelectronics, 2023.

17. OmniVision Technologies. OV2640 Product Specification Ver. 1.6. – Sunnyvale: OmniVision, 2019.

18. STMicroelectronics. STM32CubeMX for STM32 Configuration and Initialization User Manual UM1718. Rev. 48. – Geneva: STMicroelectronics, 2023.

19. STMicroelectronics. STM32CubeIDE Quick Start Guide UM2553. Rev. 4. – Geneva: STMicroelectronics, 2023.

20. Gonzalez R. C., Woods R. E., Eddins S. L. Digital Image Processing Using MATLAB. 3rd ed. – Knoxville: Gatesmark Publishing, 2020. – 1004 c.

21. Jacob B. et al. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference // Proceedings of IEEE CVPR 2018. – Salt Lake City, 2018. – P. 2704–2713.

22. Sandler M. et al. MobileNetV2: Inverted Residuals and Linear Bottlenecks // Proceedings of IEEE CVPR 2018. – Salt Lake City, 2018. – P. 4510–4520.

23. Goodfellow I., Bengio Y., Courville A. Deep Learning. – Cambridge: MIT Press, 2016. – 800 c.

24. Krizhevsky A., Sutskever I., Hinton G. E. ImageNet Classification with Deep Convolutional Neural Networks // Advances in NeurIPS. – 2012. – Vol. 25. – P. 1097–1105.

25. LeCun Y. et al. Gradient-Based Learning Applied to Document Recognition // Proceedings of the IEEE. – 1998. – Vol. 86. – No. 11. – P. 2278–2324.

26. STMicroelectronics. Getting Started with X-CUBE-AI. Application Note AN5325. Rev. 6. – Geneva: STMicroelectronics, 2023.

27. OpenMV LLC. OpenMV Cam H7 Reference Documentation. – 2023. – URL: https://openmv.io/docs/ (дата звернення: 01.04.2025).

28. Bradski G., Kaehler A. Learning OpenCV 3. – Sebastopol: O'Reilly Media, 2016. – 1024 c.

29. TensorFlow Authors. TensorFlow Lite for Microcontrollers Documentation. – 2023. – URL: https://www.tensorflow.org/lite/microcontrollers (дата звернення: 01.04.2025).

30. Sze V. et al. Efficient Processing of Deep Neural Networks: A Tutorial and Survey // Proceedings of the IEEE. – 2017. – Vol. 105. – No. 12. – P. 2295–2329.

31. Krizhevsky A. Learning Multiple Layers of Features from Tiny Images. – Toronto: University of Toronto, Technical Report, 2009. – 60 p.

32. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition // Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – Las Vegas, 2016. – P. 770–778.

33. Lin J., Chen W.-M., Lin Y., Cohn J., Gan C., Han S. MCUNet: Tiny Deep Learning on IoT Devices // Advances in Neural Information Processing Systems (NeurIPS). – 2020. – Vol. 33. – P. 11711–11722.

34. Banbury C., Reddi V. J., Lam M., Fu W., Fazel A., Holleman J. et al. Benchmarking TinyML Systems: Challenges and Direction. arXiv:2003.04821 [cs.PF]. – 2020. – 6 p.

35. Krishnamoorthi R. Quantizing deep convolutional networks for efficient inference: A whitepaper. arXiv:1806.08342 [cs.LG]. – 2018. – 36 p.

36. Howard A., Sandler M., Chu G., Chen L.-C., Chen B., Tan M. et al. Searching for MobileNetV3 // Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV). – Seoul, 2019. – P. 1314–1324.

37. Tan M., Le Q. V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks // Proceedings of International Conference on Machine Learning (ICML). – Long Beach, 2019. – P. 6105–6114.

38. Sobel I., Feldman G. A 3×3 Isotropic Gradient Operator for Image Processing. – Stanford: Stanford Artificial Intelligence Project, Technical Report, 1968. – 12 p.

39. Лемешко А.В., Антоненко А.В., Петрик А.В. (2023) Нейроморфні системи як інструмент реалізації штучного інтелекту. Вчені записки ТНУ імені В.І. Вернадського. Серія: Технічні науки. Том 34 (73) № 3 2023, 175-183.

https://doi.org/10.32782/2663-5941/2023.3.1/28

40. Лемешко А. В., Антоненко А. В., Балвак А. А., & Новіченко Є. О. (2023). Актуальні засади створення алгоритмів обробки інформації для логістичних центрів. Таврійський науковий вісник. Серія: Технічні науки, (1), 25-32. https://doi.org/10.32851/tnv-tech.2023.1.3

41. Гакман Д. В. & Антоненко А. В. (2023). Особливості розробки та використання фреймворків для автоматизованого тестування. Таврійський науковий вісник. Серія: Технічні науки, (2), 21-32. https://doi.org/10.32782/tnv-tech.2023.2.3

42. Сорочинський О. О., Гунько О. С., Антоненко А. В., Шульженко К. Ю., & Христенко Д. О. (2023). Моделювання та проектування приватного мережево-серверного середовища з використанням технології opennebula. Таврійський науковий вісник. Серія: Технічні науки, (3), 3-12. https://doi.org/10.32782/tnv-tech.2023.3.1

43. Лемешко А., Добровольський О., Антоненко А., Завадський В. Оптимізація часу реагування веб серверу apache. Вісник Хмельницького національного університету, Том 1, №2, 2023 (319). 189-201. https://www.doi.org/10.31891/2307-5732-2023-319-1-189-201

44. Лемешко А., Антоненко А., Цвик О. Аналіз і особливості програмного забезпечення для контролю трафіку. Вісник Хмельницького національного університету, Том 1, №3, 2023 (321). 64-68. https://www.doi.org/10.31891/2307-5732-2023-321-3-64-68

45. Лемешко А., Кувік Н., Антоненко А., Гнядий В. Дослідження мережевих загроз для забезпечення максимального захисту даних та інфраструктури. Вісник Хмельницького національного університету, Том 1, №3, 2023 (321). 120-127. https://www.doi.org/10.31891/2307-5732-2023-321-3-120-127

46. Пахомов М., Калита Т., Антоненко А., Галета В. Використання штучного

інтелекту в автоматизованих системах. Вісник Хмельницького національного університету, №4, 2023 (323). 11-20. https://www.doi.org/10.31891/2307-5732-2023-323-4-11-20

47. Поперешняк С.В., Вечерковська А.С., Житкевич О.Б., Антоненко А.В. Моніторинг руху об'єктів засобами геоінформаційних систем. Вісник Хмельницького національного університету, №6, 2023 (329) 312-320. https://www.doi.org/10.31891/2307-5732-2023-329-6-312-320

48. Поперешняк С. В., Вечерковська А. С., Хільченко М. Ю., & Антоненко А. В. (2024). Хмарні технології як сервіси для оптимізації процесів адміністрування. Таврійський науковий вісник. Серія: Технічні науки, (6), 54-63. https://doi.org/10.32782/tnv-tech.2023.6.7

49. Поперешняк С. В., Куценко М. О., & Антоненко А. В. (2024). Система надання рекомендацій щодо вивчення технологій з веб-розробки. Таврійський науковий вісник. Серія: Технічні науки, (6), 64-71. https://doi.org/10.32782/tnv-tech.2023.6.8

50. Бондаренко Є. С., Каньшин К. В., Антоненко А. В., Степанчук В. І., & Ланевський Л. А. (2023). Класифікація та особливості збору даних у системах підтримки прийняття управлінських рішень. Таврійський науковий вісник. Серія: Технічні науки, (4), 23-31. https://doi.org/10.32782/tnv-tech.2023.4.3

51. Бенедіко І. В., Вічкарук А. І., Антоненко А. В., Лисенко К. В., & Сижко О. Ю. (2023). Класифікації моделей застосування машинного навчання у кібербезпеці. Таврійський науковий вісник. Серія: Технічні науки, (4), 11-22. https://doi.org/10.32782/tnv-tech.2023.4.2

52. Зайцев Є. О., Березниченко В. О., Антоненко А. В., Панчик М. В., & Тонких О. Г. (2024). Впровадження технологій іот в системах електропостачання промислових підприємств. Таврійський науковий вісник. Серія: Технічні науки, (5), 23-32. https://doi.org/10.32782/tnv-tech.2023.5.3

53. Кірюшин В. О., Лемешко А. В., Антоненко А. В., Рябоконь О. О., & Макаренко Є. О. (2024). Особливості функціональності dns-сервера та dhcp-сервера. Таврійський науковий вісник. Серія: Технічні науки, (5), 55-64. https://doi.org/10.32782/tnv-tech.2023.5.6

54. Шрам М.М., Замковий О.Ю., Антоненко А.В., Орлюк Х.В., Дутко П.М. Особливості побудови масиву даних на основі нейромережі у сфері інтернету речей для готельно-ресторанного бізнесу. Вісник Хмельницького національного університету, Том 1, №5, 2023 (325), 252-257. http://journals.khnu.km.ua/vestnik/wp-content/uploads/2023/11/325-n5-1-252-257.pdf

55. Поплавський Д. В., Голубенко О. І., Антоненко А.В., Кондратюк Б. О., Верко Р. А. Впровадження моделі розгортання мереж мобільного зв'язку 6-го покоління за допомогою безпілотників. Вісник Хмельницького національного університету, №6, 2023 (329), 49-55. [http://journals.khnu.km.ua/vestnik/wp-content/uploads/2024/01/329-ts-2023-n6-49-55.pdf]
(http://journals.khnu.km.ua/vestnik/wp-content/uploads/2024/01/329-ts-2023-n6-49-55.pdf)
https://doi.org/10.31548/humanhealth.4.2024.37

56. Зайцев Є.О., Березниченко В.О., Антоненко А.В., Закусило С.А. (2022). SMART засоби визначення аварійних станів у розподільних електричних мережах міст. Таврійський науковий вісник. Серія: Технічні науки, (5), 3-12. https://doi.org/10.32851/tnv-tech.2022.5.1

57. Буряк, М. С., Цвик, О. С., Антоненко, А. В., Бурачинський, А. Ю., Балвак, А. А., & Занфіров, Р. Р. (2024). Особливості розгортання мереж мобільного зв'язку за допомогою бпла. Таврійський науковий вісник. Серія: Технічні науки, (3), 3-12. URL: https://doi.org/10.32782/tnv-tech.2024.3.1

58. Балвак, А. А., Лемешко, А. В., Антоненко, А. В., Зіняр, Д. А., Бурачинський, А. Ю., & Приходько, А. П. (2024). Обробка та аналіз даних на

прикладі набору spambase з використанням бібліотек для машинного навчання. Таврійський науковий вісник. Серія: Технічні науки, (2), 3-20. https://doi.org/10.32782/tnv-tech.2024.2.1

59. Твердохліб, А. О., Востріков, С. О., Антоненко, А. В., Бегліцов, С. В., Гергель, О. Г., & Бобков, Р. А. (2024). Оптимізація роботи бездротових мереж з використанням inet framework. Таврійський науковий вісник. Серія: Технічні науки, (5), 11-24. https://doi.org/10.32782/tnv-tech.2024.5.2

60. Твердохліб, А. О., & Антоненко, А. В. (2024). Особливості розподіленого брокеру trinity на основі технології блокчейн. Таврійський науковий вісник. Серія: Технічні науки, (6), 130-138. URL: https://doi.org/10.32782/tnv-tech.2024.6.14

61. Солобаєв, С. Г., Востріков, С. О., Антоненко, А. В., Балвак, А. А., Мішкур, Ю. В., & Приходько, А. П. (2024). Аспекти використання нейронних мереж для покращення якості зображень. Таврійський науковий вісник. Серія: Технічні науки, (6), 3-10. URL:https://doi.org/10.32782/tnv-tech.2024.6.1

62. Балвак, А. А., Цвик, О. С., Антоненко, А. В., Ємелін, Д. М., & Гришковець, Є. П. (2024). Інноваційні методи відображення інформації у веб-браузерах. Таврійський науковий вісник. Серія: Технічні науки, (1), 3-11. https://doi.org/10.32782/tnv-tech.2024.1.1

63. Востріков, С. О., Бурачинський, А. Ю., Антоненко, А. В., Твердохліб, А. О., Балвак, А. А., & Слободян, О. А. (2024). Особливості автоматизованого тестування з використанням фреймворків. Таврійський науковий вісник. Серія: Технічні науки, (4), 3-14. https://doi.org/10.32782/tnv-tech.2024.4.1

64. Мішкур, Ю. В., Сольський, Д. Я., Антоненко, А. В., Солобаєв, С. Г., Подуран, Д. В., & Сарафанюк, Р. О. (2024). Оптимізація продуктивності веб-сервера Apache. Таврійський науковий вісник. Серія: Технічні науки, (4), 15-30. https://doi.org/10.32782/tnv-tech.2024.4.2

65. Бурачинський А., Сольський D., Антоненко, А.,Твердохліб А., Мішкур Y., & Зіняр D. (2024). Аспекти застосування нейронних мереж для криптографії. Measuring and computing devices in technological processes, (4), 394–400. https://doi.org/10.31891/2219-9365-2024-80-47

66. Мішкур, Ю., Твердохліб, А., Востріков, Антоненко, А., С., & Балвак, А. (2024). Нейромережі в мистецтві як іструмент графічного дизайну. Herald of Khmelnytskyi National University. Technical Sciences, 345(6(2), 95-101. https://doi.org/10.31891/2307-5732-2024-345-6-14

67. Солобаєв, С. Г., Востріков, С. О., Антоненко, А. В., Ткаченко, О. В., Ходосов, А. О., & Остапенко, О. С. (2025). Використання нейронних мереж у прогнозуванні безпеки мережі. Таврійський науковий вісник. Серія: Технічні науки, (2), 3-10. https://doi.org/10.32782/tnv-tech.2025.2.1

68. Черевик, О. В., Востріков, С. О., Антоненко, А. В., Мішкур, Ю. В., Балвак, А. А., & Дзісяк, В. Г. (2025). Оптимізація мережевої інфраструктури для забезпечення відмовостійкості. Таврійський науковий вісник. Серія: Технічні науки, (3), 3-13. https://doi.org/10.32782/tnv-tech.2025.3.1

69. Лемешко А. В., Антоненко А. В., Слюсар В. О. (2023) Технології хмарних обчислень та динаміка їх розвитку. ITSynergy, (1), 89–108. (IndexCopernicus) https://doi.org/10.53920/ITS-2023-1-6

70. Лемешко А. В., Антоненко А. В., Шрам М. М. (2023) Моделювання безпроводових мереж у середовищі OMNET++ з використанням InetFramework. ITSynergy, (1), 37–59. (IndexCopernicus) https://doi.org/10.53920/ITS-2023-1-3

71. Лемешко А. В., Антоненко А. В., Матвійчук О. М. (2023). Управління трафіком в гібридній програмно-визначеній мережі. ITSynergy, (1), 98–114 (IndexCopernicus) [https://doi.org/10.53920/ITS-2023-2-

7](https://doi.org/10.53920/ITS-2023-2-7) Sciences, (3), 3-8. https://doi.org/10.21303/2504-5695.2022.002605

72. Антоненко А., & Твердохліб А. (2025). Використання смарт-контрактів та технології блокчейн для підвищення ефективності розподілених брокерних систем в умовах високих навантажень. Measuring and computing devices in technological processes, (3), 467–473. https://doi.org/10.31891/2219-9365-2025-83-58

73. Buriak, M. S., Vostrikov, S. O., Antonenko, A. V., Balvak, A. A., Korotin, D. S., & Myronenko, R. O. (2025). Можливості підвищення ефективності мереж в системах клієнт- сервер. Таврійський науковий вісник. Серія: Технічні науки, 1(4), 3-14. https://doi.org/10.32782/tnv-tech.2025.4.1.1

74. Сольський D., Солобаєв S., Антоненко, А., Чечик S., & Черевик О. (2025). Використання бібліотеки scikit-learn у методах класифікації машинного навчання. Measuring and computing devices in technological processes, (1), 468–472. https://doi.org/10.31891/2219-9365-2025-81-58

75. Zaitsev I., Golubenko O., Tkachenko O., Pidmohylnyi O., Antonenko A. (2023). Exploring advanced hypothesis generation in astronomy through the implementation of a mathematical model of linguistic neural networks. CEUR Workshop Proceedings, 2023, 3687, 121–128. https://ceur-ws.org/Vol-3687/Short_5.pdf

76. Ткаченко, О., Голубенко, О., Власенко, В., & Антоненко, А. (2025). Методи оптимізації використання оперативної пам’яті в процесах підвищення швидкодії комп’ютерних систем. Вимірювальна та обчислювальна техніка в технологічних процесах, 84(4), 465–472. https://doi.org/10.31891/2219-9365-2025-84-57

77. Завгородня, Г., Завгородній, В., Голубенко, О., & Антоненко, А. (2025). Means of ensuring scalability and autonomy of an adaptive content generation system.

Technologies and Engineering, 26(6), 21–31. https://doi.org/10.30857/2786-5371.2025.6.2

78. Golubenko, O., Onysko, A., Antonenko, A., Chechyk, S., & Vostrikov, S. (2025). Prospects of using AI for automated software testing. Technologies and Engineering, 26(5), 9–20. https://doi.org/10.30857/2786-5371.2025.5.1.

ДОДАТОК А

Принципова схема підключення OV2640 до STM32H743ZIT6

У додатку наведено принципову електричну схему підключення камерного модуля OV2640 до мікроконтролера STM32H743ZIT6 на платі NUCLEO-H743ZI2. Схема включає:

– 8-бітну шину даних DCMI: PC6(D0), PC7(D1), PE0(D2), PE1(D3), PE4(D4), PE5(D5), PE6(D6), PD3(D7);

– синхросигнали: PIXCLK→PA6, VSYNC→PB7, HREF→PA4;

– шину I2C/SCCB: SCL→PB8 (AF4), SDA→PB9 (AF4), підтягувальні резистори 4,7 кОм до 3,3 В;

– тактовий сигнал XCLK: PA8 (MCO1, 12 МГц);

– сигнали управління: RESET→PD4, PWDN→PD5;

– живлення: 3,3 В від LDO NUCLEO; блокувальні конденсатори 100 нФ + 10 мкФ поряд з VCC камери.

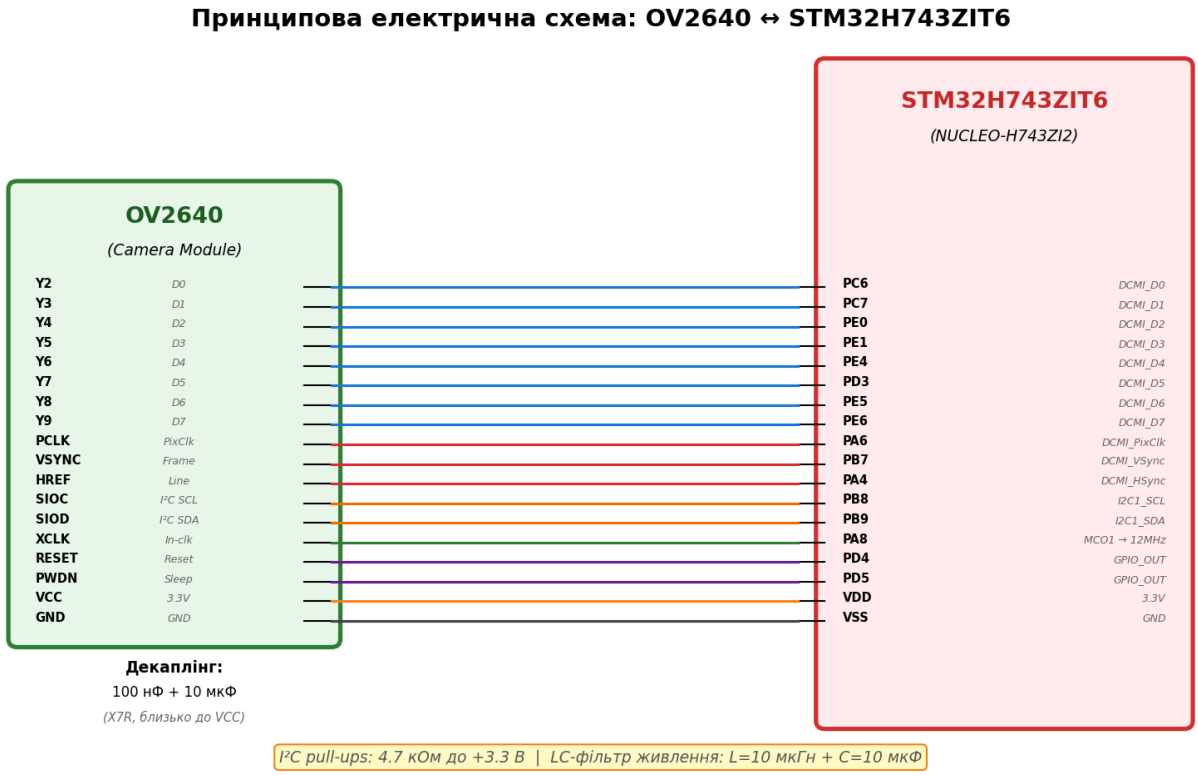


Рисунок А.1 – Принципова електрична схема підключення OV2640 до STM32H743ZIT6

ДОДАТОК Б

Лістинг програмного коду ключових модулів

Б.1 – Функція конвертації RGB565 → Grayscale з SIMD-оптимізацією:

```
/**  * @brief  IP_RGB565_to_Gray_fast — конвертація RGB565 → Grayscale  * @param  src  вхідний буфер RGB565 (у AXI SRAM, DMA-заповнений)  * @param  dst  вихідний буфер Grayscale  * @param  n  кількість пікселів (320*240=76800 для QVGA)  * @note  SIMD throughput: ~160 Mpx/s @ 400 МГц Cortex-M7  */ void IP_RGB565_to_Gray_fast(const uint16_t * restrict src, uint8_t * restrict dst, uint32_t n) {  /* Інвалідація D-Cache перед читанням DMA-буфера */ SCB_InvalidateDCache_by_Addr((uint32_t*)src, n * 2);  for (uint32_t i = 0; i < n; i++) {  uint16_t px  = src[i];  uint32_t r8  = ((px >> 11) & 0x1F) << 3;  /* R 5-bit → 8-bit */  uint32_t g8  = ((px >>  5) & 0x3F) << 2;  /* G 6-bit → 8-bit */ uint32_t b8  = ((px  ) & 0x1F) << 3;  /* B 5-bit → 8-bit */  /* BT.601 яскравість через цілочисельну апроксимацію */  dst[i] = (uint8_t)((77U * r8 + 150U * g8 + 29U * b8) >> 8);  } }
```

Б.2 – Запуск DCMI з подвійною буферизацією DMA:

```
/**  * @brief  Camera_StartCapture — запуск DCMI + DMA Double-Buffer  * @param  buf0  перший DMA-буфер (у AXI SRAM: 0x24000000)  * @param  buf1  другий DMA-буфер  * @param  size  розмір одного кадру у байтах (76800 для QVGA Gray)  */ HAL_StatusTypeDef Camera_StartCapture(uint8_t * buf0, uint8_t * buf1, uint32_t size) {  /* Увімкнення Double-Buffer Mode в регістрі DMA */  hdma_dcmi.Instance->CR |= DMA_SxCR_DBM;  /* Налаштування адрес буферів */  hdma_dcmi.Instance->M0AR = (uint32_t)buf0;  hdma_dcmi.Instance->M1AR = (uint32_t)buf1;  /* Запуск DMA з перериванням по завершенню передачі */
```

```
HAL_DMAEx_MultiBufferStart_IT(&hdma_dcmi, (uint32_t)&DCMI->DR, (uint32_t)buf0, (uint32_t)buf1, size / 4U); /* Запуск DCMI у безперервному режимі Continuous Grab */ return HAL_DCMI_Start_DMA(&hdcmi, DCMI_MODE_CONTINUOUS, (uint32_t)buf0, size / 4U); } /* ISR: викликається після заповнення кожного DMA-буфера */ void HAL_DCMI_FrameEventCallback(DCMI_HandleTypeDef *hdcmi) { /* Визначення щойно заповненого буфера */ uint8_t active = (hdma_dcmi.Instance->CR & DMA_SxCR_CT) ? 0 : 1; s_frame_ready = 1; /* volatile прапорець */ s_ready_buf_id = active; }
```

Б.3 – Алгоритм бінаризації методом Оцу:

```
/** * @brief IP_OtsuThreshold — автоматичний поріг Оцу * @return оптимальний поріг T* ∈ [0, 255] */ uint8_t IP_OtsuThreshold(const uint8_t *img, uint32_t n) { uint32_t hist[256] = {0}; /* Крок 1: побудова гістограми O(N) */ for (uint32_t i = 0; i < n; i++) hist[img[i]]++; /* Крок 2: пошук T* за максимумом σ²_B O(256) */ uint64_t total = 0, sum_all = 0; for (int t = 0; t < 256; t++) { total += hist[t]; sum_all += (uint64_t)t * hist[t]; } uint64_t sum_bg = 0, w_bg = 0; float best_var = 0.0f; uint8_t best_t = 0; for (int t = 0; t < 256; t++) { w_bg += hist[t]; sum_bg += (uint64_t)t * hist[t]; uint64_t w_fg = total - w_bg; if (w_bg == 0 || w_fg == 0) continue; float mu_bg = (float)sum_bg / w_bg; float mu_fg = (float)(sum_all - sum_bg) / w_fg; float var_b = (float)w_bg * w_fg * (mu_bg - mu_fg) * (mu_bg - mu_fg); if (var_b > best_var) { best_var = var_b; best_t = t; } } return best_t; }
```

ДОДАТОК В

Результати тестування системи

Таблиця В.1 – Повні результати вимірювання часу (n=100, 95% довірчий інтервал)

Етап	$\bar{x}$, мс	Min, мс	Max, мс	95% CI, мс
RGB565→Gray	0,48	0,47	0,51	[0,478; 0,482]
Гаусівська фільтрація	3,42	3,38	3,49	[3,414; 3,426]
Бінаризація (Оцу)	0,38	0,37	0,40	[0,378; 0,382]
Масштабування 96×96	1,12	1,10	1,15	[1,116; 1,124]
Квантизація int8	0,85	0,84	0,87	[0,848; 0,852]
CNN інференс	52,3	51,8	53,1	[52,24; 52,36]
Весь конвеєр	59,6	59,0	60,6	[59,53; 59,67]

Таблиця В.2 – Результати тестування точності по класах та умовах освітлення

Клас / Освітлення	Природне	Штучне	Низьке	Разом TP	Recall, %
Людина (100)	37/40	33/36	18/24	88/100	88,0
Автомобіль (100)	39/40	34/36	18/24	91/100	91,0
Велосипед (100)	36/40	31/36	18/24	85/100	85,0
Знак СТОП (100)	40/40	35/36	20/24	95/100	95,0
Тварина (100)	35/40	29/36	24/24	88/100	88,0
Разом	**187/200**	**162/180**	**98/120**	**447/500**	**89,4**

Розрахунок орієнтовної собівартості при серійному виробництві виконано для трьох обсягів: 100, 1 000 та 10 000 одиниць (таблиця В.3, рисунок В.1).

Таблиця В.3 – Економічні показники при різних обсягах виробництва

Стаття витрат	**100 шт, USD**	**1000 шт, USD**	**10000 шт, USD**
STM32H743ZI (MCU)	8,50	6,20	4,10
OV2640 (камерний модуль)	4,80	3,50	2,20
PCB 4-шари (80×50 мм)	5,00	1,80	0,90
Пасивні компоненти	2,80	1,40	0,80
Корпус + механіка	6,50	3,20	1,80
Монтаж SMD (assembly)	4,50	1,80	0,60
Тестування + калібрування	3,00	1,20	0,40
Пакування + логістика	1,50	0,80	0,50

Собівартість одиниці при серії 10 000 шт. складає 11,30 USD — це у 8–9 разів дешевше за OpenMV H7 ($85) та в 3 рази дешевше за Raspberry Pi Zero 2 ($35). Розроблена система є економічно конкурентоспроможною для масового виробництва автономних IoT-датчиків із технічним зором.

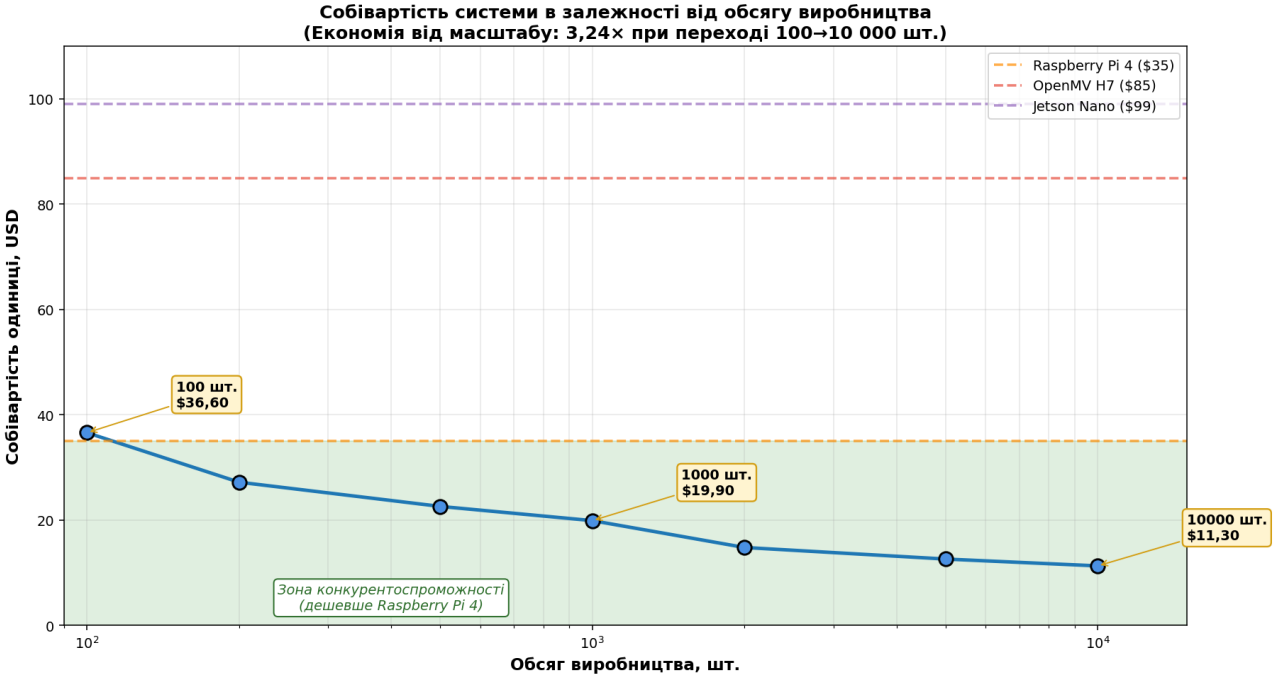


Рисунок В.1 – Залежність собівартості виробу від обсягу виробництва

ДОДАТОК Г

Специфікація компонентів (Bill of Materials, BOM)

Таблиця Г.1 – Перелік компонентів системи CV

№	Найменування	Позначення	К-ть	Виробник
1	Мікроконтролер	STM32H743ZIT6	1	STMicroelectronics
2	Оціночна плата	NUCLEO-H743ZI2	1	STMicroelectronics
3	Камерний модуль	OV2640 2MP	1	OmniVision
4	Кварцовий резонатор	HSE 25 МГц	1	Epson
5	Блокувальний конденсатор	C0402 100 нФ X7R	16	Murata
6	Танталовий конденсатор	TANT 10 мкФ 10В	4	Kemet
7	Підтягувальний резистор	R0402 4,7 кОм	4	Yageo
8	LDO-стабілізатор	AMS1117-3,3	1	AMS
9	LED-індикатор	LED 0603 Green	3	Kingbright
10	Кнопка скидання	SMD 6×6×4,3	1	Omron
11	Індуктивність	L 10 мкГн 500 мА	2	Würth Elektronik
12	Роз'єм FFC-12	FFC 0,5 мм 12-pin	1	Molex
13	USB micro-B роз'єм	SMD Micro USB B	1	Amphenol
14	PCB (4 шари)	FR-4, 80×50×1,6 мм	1	JLCPCB

Орієнтовна вартість комплектуючих для одиничного виробництва: 45–60 USD. У серії 1000+ шт. вартість знижується до 25–35 USD завдяки оптовим знижкам та власній PCB замість оціночної плати.

ДОДАТОК Ґ

Додаткові лістинги програмного коду

Д.1 – Функція гаусівської фільтрації ядром 3×3:

```
/**  * @brief  IP_GaussianFilter3x3 — гаусівська фільтрація 3×3  * @note Ядро (1/16)*[1 2 1; 2 4 2; 1 2 1]  *  Реалізація через цілочисельні зсуви  * @param  src — вхідний Grayscale буфер  * @param  dst — вихідний буфер (розмір такий самий)  * @param  w, h — ширина та висота  */ void IP_GaussianFilter3x3(const uint8_t *src, uint8_t *dst,  uint16_t w, uint16_t h) {  /* Копіювання границь без фільтрації */  memcpy(dst, src, w); memcpy(dst + (h-1)*w, src + (h-1)*w, w);  for (uint16_t y = 1; y < h - 1; y++) {  dst[y*w]  = src[y*w];  dst[y*w + w-1] = src[y*w + w-1];  for (uint16_t x = 1; x < w - 1; x++) {  uint16_t sum =  1*src[(y-1)*w + (x-1)] + 2*src[(y-1)*w + x] + 1*src[(y-1)*w + (x+1)] +  2*src[ y  *w + (x-1)] + 4*src[ y  *w + x] + 2*src[ y  *w + (x+1)] +  1*src[(y+1)*w + (x-1)] + 2*src[(y+1)*w + x] + 1*src[(y+1)*w + (x+1)];  dst[y*w + x] = (uint8_t)(sum >> 4);  /* /16 */  }  } }
```

Д.2 – Оператор Собеля для виявлення границь:

```
/**  * @brief  IP_Sobel — оператор Собеля для виявлення границь  * @note Апроксимація |G| ≈ |Gx| + |Gy|  */ void IP_Sobel(const uint8_t *src, uint8_t *dst, uint16_t w, uint16_t h) {  for (uint16_t y = 1; y < h - 1; y++) {  for (uint16_t x = 1; x < w - 1; x++) {  int16_t gx =  -1*src[(y-1)*w+(x-1)] + 1*src[(y-1)*w+(x+1)] +  -2*src[ y  *w+(x-1)] + 2*src[ y  *w+(x+1)] +  -1*src[(y+1)*w+(x-1)] + 1*src[(y+1)*w+(x+1)];  int16_t gy =  -1*src[(y-1)*w+(x-1)] - 2*src[(y-1)*w+x] - 1*src[(y-1)*w+(x+1)] + 1*src[(y+1)*w+(x-1)] + 2*src[(y+1)*w+x] + 1*src[(y+1)*w+(x+1)];  int32_t mag = abs(gx) + abs(gy);  dst[y*w + x] = (mag > 255) ? 255 :
```

```
(uint8_t)mag;       }    } }
```

Д.3 – Основний цикл обробки (state machine):

```
/**  * @brief  main_pipeline — основний цикл обробки кадрів  * @note   Кінцевий автомат станів  */ typedef enum {     ST_IDLE, ST_CAPTURE, ST_PREPROC, ST_INFER, ST_OUTPUT } PipelineState_t;  void main_pipeline(void) {     static PipelineState_t state = ST_IDLE;     static uint8_t *frame_ptr = NULL;     static ai_float cnn_output[AI_OUTPUT_SIZE];      while (1) {         HAL_IWDG_Refresh(&hiwdg);         switch (state) {         case ST_IDLE:             if (s_frame_ready) { state = ST_CAPTURE; s_frame_ready = 0; }             break;         case ST_CAPTURE:             Camera_GetFrame(&frame_ptr);             SCB_InvalidateDCache_by_Addr(frame_ptr, FRAME_SIZE_RGB);             state = ST_PREPROC;             break;         case ST_PREPROC:             IP_RGB565_to_Gray_fast((uint16_t*)frame_ptr, s_gray_buf, N_PIXELS);             IP_GaussianFilter3x3(s_gray_buf, s_filt_buf, W, H);             IP_Resize_Bilinear(s_filt_buf, s_cnn_in, W, H, 96, 96);             IP_Normalize(s_cnn_in, s_cnn_in_q, 96*96, 0, 1.0f/128.0f);             state = ST_INFER;             break;         case ST_INFER:             ai_network_run(network, &input_data, &output_data);             state = ST_OUTPUT;             break;         case ST_OUTPUT:             uint8_t cls = NN_GetTopClass(cnn_output, 5, 0.6f);
```

```
printf("Class: %s, conf: %.2f\n",
class_names[cls], cnn_output[cls]);             state =
ST_IDLE;             break;         }     } }
```

ДОДАТОК Д

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

(Презентація)

Презентаційні матеріали до захисту кваліфікаційної роботи (16 слайдів)

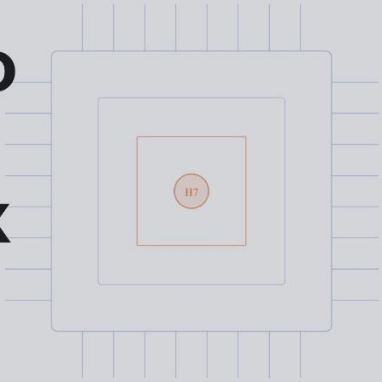
ДУІКТ

ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ · ННІІТ · КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

1

КВАЛІФІКАЦІЙНА РОБОТА · БАКАЛАВР · 123 КОМП'ЮТЕРНА ІНЖЕНЕРІЯ

Система комп'ютерного зору для мікроконтролерних платформ STM32



ВИКОНАВ
Клобуков Олег гр. КІД-42

КЕРІВНИК
к.т.н Антоненко Артем В.

МІСЦЕ · РІК
Київ · 2026

2

02 / 16 • АКТУАЛЬНІСТЬ

Переваги використання Edge AI на мікроконтролерах

01

Затримки хмарної обробки

Передача кадру на сервер та зворотна відповідь — 100–500 мс . Для систем реального часу — критично.

02

Енергоспоживання

Постійний радіоканал (Wi-Fi / 4G) виснажує батарею автономного пристрою за години.

03

Приватність даних

Відеопотік не повинен залишати пристрій — медицина, безпека, побутові камери.

Рішення → виконання нейромереж безпосередньо на пристрої (Edge AI на MCU)

3

03 / 16 • МЕТА · ЗАВДАННЯ

Мета і завдання роботи

МЕТА

Створити робочий прототип системи розпізнавання об'єктів на STM32 для автономних пристроїв з обмеженими ресурсами.

// 67 стор. · 17 рис. · 30 табл. · 38 джерел · 5 додатків

ЗАВДАННЯ

Аналіз архітектур CNN та MCU-платформ
Обґрунтування вибору апаратної бази
Квантизація моделі під ARM Cortex-M
Реалізація DCMI / DMA-конвеєра захоплення
Експериментальне тестування продуктивності

4

04 / 16 • АКАДЕМІЧНА ЧАСТИНА

Об'єкт, предмет, наукова новизна

ОБ'ЄКТ

Процес розпізнавання об'єктів в умовах обмежених обчислювальних та енергетичних ресурсів.

ПРЕДМЕТ

Апаратно-програмні засоби та алгоритми Edge AI на базі сімейства STM32.

НАУКОВА НОВИЗНА

→ Методика комплексної оптимізації конвеєра SIMD + INT8 → прискорення ×4

→ Розрахунок енергобюджету з режимом WFI → автономність 22.5 год

→ Емпіричні залежності FPS від роздільної здатності для STM32H743

5

05 / 16 • РОЗДІЛ 1 · АНАЛІЗ ПЛАТФОРМ

Порівняння апаратних платформ для CV

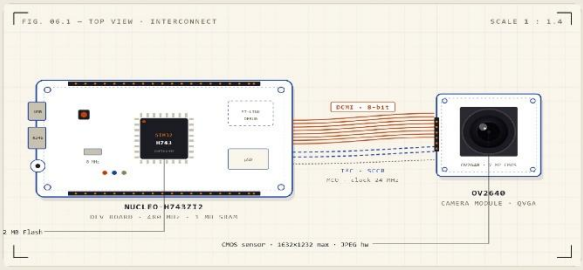
ПЛАТФОРМА	FPS	ТОЧНІСТЬ	СПОЖИВАННЯ	ВАРТІСТЬ	ОСОБЛИВІСТЬ
STM32H743 · OV2640 наша	**16.8**	**89.4 %**	**0.61 Вт**	**3–8 $**	**SIMD + CMSIS-NN + DCMI/DMA**
ESP32-CAM	5–8	~72 %	0.3–0.5 Вт	3–5 $	Wi-Fi, обмежений MIPS
OpenMV H7	10–15	~80 %	0.4–0.6 Вт	~85 $	Готова MicroPython-екосистема
Raspberry Pi 4	30+	~93 %	5–8 Вт	35–75 $	Linux, x10 енергії
NVIDIA Jetson Nano	60+	~95 %	5–10 Вт	~99 $	GPU, важка платформа

STM32H743 — оптимум за критерієм продуктивність / ціна / енергоспоживання

6

06 / 16 • АПАРАТНА ПЛАТФОРМА

серце системи · STM32H743ZIT6 ·



CORE

ARM Cortex-M7

FPU · DSP · L1-cache

SENSOR

OV2640

QVGA · апарат. JPEG

КЛЮЧОВІ ХАРАКТЕРИСТИКИ

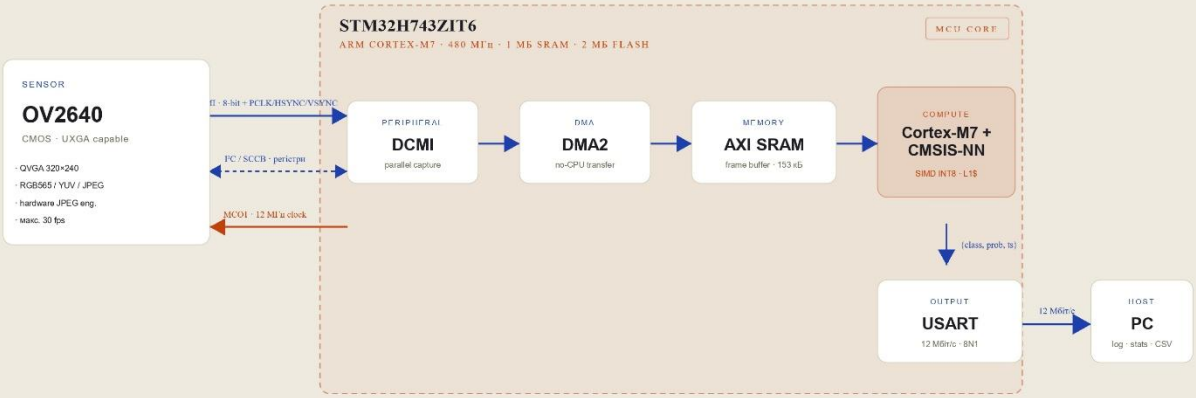
480 МГц Cortex-M7 | 1 МБ SRAM | 2 МБ Flash | DCMI 8-bit | DMA2 | Chrom-Art (DMA2D) | DSP SIMD (SMLAD/SMUAD) | L1-cache 32+32 кБ | FPU подвійної точності | I²C / SCCB | USART до 12 Мбіт/с | MCO 12 МГц

// обрано як оптимум за критерієм MIPS · TOPS · ціна. Розділ 2.2.

7

07 / 16 • АРХІТЕКТУРА СИСТЕМИ

Структурна схема апаратної платформи



// 2.3 · 2.8 Структурна схема

8

08 / 16 • ШЛЯХ ДАНИХ

Покрокова робота зсередини

A · CAPTURE 01—03

Захоплення кадру

01	OV2640 sensor	RGB565
02	DCMI + DMA2	8-bit · burst
03	AXI SRAM (ping-pong)	153 кБ

33.3 мс

// апаратний шлях · CPU = WFI

B · COMPUTE 04—06

Обробка на M7

04	CMSIS-DSP препроц	5.4 мс
05	CMSIS-NN · CNN INT8	52.3 мс
06	argmax + nopir	1.0 мс

58.7 мс

// 87.8 % бюджету — на CNN

C · TRANSMIT 07—08

Передача на ПК

07	пакет {class, prob, ts}	JSON / CSV
08	USART → PC	12 Мбіт/с

~1.0 мс

// статистика · accuracy / FPS / power

ЧАСОВИЙ БЮДЖЕТ ОДНОГО КАДРУ · ПАРАЛЕЛЬНЕ ВИКОНАННЯ

59.6 МС · 16.8 FPS



DMA + CPU працюють паралельно · кадр = max(захоплення, обробка) + UART = 59.6 мс

// 2.13 End-to-end data flow

9

09 / 16 • КОНВЕЄР ОБРОБКИ

Часові відрізки обробки кадру · 8 кровів за 59.6 мс

Захоплення	RGB565 → Gray	Gauss 3×3	Otsu	Resize 96×96	Quantize INT8	CNN INT8	Postproc
33.3 мс	0.48 мс	3.42 мс	0.38 мс	1.12 мс	0.85 мс	52.3 мс · 87.8%	1.04 мс

DOMINANT STAGE

CNN inference

87,8% часу одного кадру → основна точка оптимізації

CLASSICAL CV

~5.4 мс

Gauss, Otsu, ресайз — реалізовані через CMSIS-DSP та SIMD-інструкції

FRAME RATE

16.8 FPS

Виміряно на STM32H743 @ 480 МГц, кеш увімкнено

10

10 / 16 • ОПТИМІЗАЦІЯ

X-CUBE-AI · INT8-квантизація моделі

РОЗМІР ВАГ

1.62 МБ → **420кБ**

×3.95 компресія

ЧАС ІНФЕРЕНСУ

210 мс → **52.3мс**

×4.04 прискорення

ТОЧНІСТЬ

91.6 % → **89.4%**

−2.2 п.п. деградації

+ SIMD INSTRUCTIONS

SMLAD, SMUAD, SXTB16 → 4× MAC за такт у CMSIS-NN

+ CACHE MANAGEMENT

SCB_EnableICache / DCache, інвалідизація DMA-буферів

Прискорення у ×4 ціною незначної втрати точності

11

11 / 16 • РОЗДІЛ 3 · РЕЗУЛЬТАТИ

Тестування · 500 зображень · 5 класів

ACCURACY
89.4%
точність розпізнавання

FRAME RATE
16.8fps
QVGA, 480 МГц, кеш on

POWER
0.61Вт
середнє в неперервному режимі

RAM / FLASH
66% / 52%
з 1 МБ SRAM · 2 МБ Flash

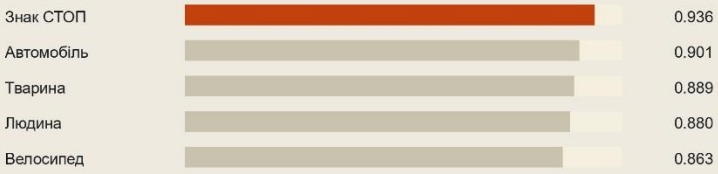
Усі показники ТЗ виконано із запасом 30–68 %

// 3.2 · 3.3 Тестування продуктивності

12

12 / 16 • ТОЧНІСТЬ ПО КЛАСАХ

Детальний аналіз по 5 класах



macro · 100 зображень / клас

НАЙКРАЩИЙ КЛАС
Знак СТОП — 0.936
Виразна геометрична форма, контрастний колір, низька варіативність ракурсів.

НАЙГІРШИЙ КЛАС
Велосипед — 0.863
Тонкі деталі рами, висока варіативність ракурсів, перетин з класом «людина».

SPREAD
Δ = 0.073
Рівномірний розподіл якості по класах — модель не «провалюється» на жодному.

// 3.8 Точність по класах

13

13 / 16 • ЕНЕРГОЕФЕКТИВНІСТЬ

Автономність системи

НЕПЕРЕРВНИЙ РЕЖИМ

FPS
16.8

POWER
0.61 Вт

АВТОНОМНІСТЬ · 3000 мА·год
15.5 год

РЕЖИМ WFI · ЕНЕРГОЗБЕРЕЖЕННЯ

FPS
10

POWER
0.42 Вт

АВТОНОМНІСТЬ · 3000 мА·год
22.5 год

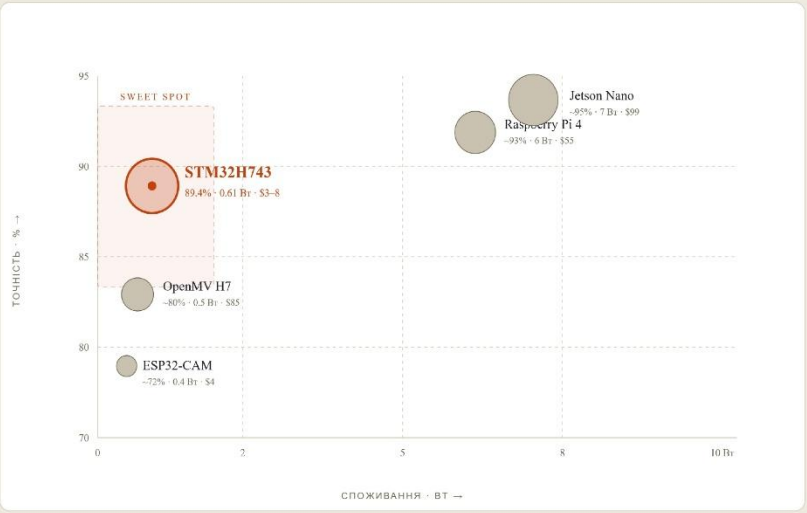
До 22.5 годин автономної роботи від однієї зарядки — підтверджено вимірюваннями

// 2.5 Енергобюджет та автономність

14

14 / 16 • ПОЛОЖЕННЯ НА РИНКУ

Точність vs споживання. Порівняння продукту



ВИСНОВОК

Найкраще співвідношення в зоні low-power · low-cost.

vs ESP32-CAM

+17 п.п.

точності при подібному споживанні

vs OpenMV H7

×11

дешевше при кращій точності

// 3.4 Порівняння з рішеннями

15 / 16 • ВИСНОВКИ

Результат

Спроєктовано архітектуру системи на STM32H743 + OV2640 : структурну, функціональну та схему підключення. Реалізовано конвеєр обробки кадру з використанням CMSIS-DSP і CMSIS-NN , SIMD-інструкцій Cortex-M7. Виконано INT8-квантизацію через X-CUBE-AI → прискорення інференсу ×4 ціною −2.2 п.п. точності. Проведено тестування на 500 зображеннях, 5 класів: 89.4% точність · 16.8 FPS · 0.61 Вт . Підтверджено перевагу +17 п.п. над ESP32-CAM при ×11 нижчій ціні відносно OpenMV H7.

Прошов апробацію:
(1) Клобуков О.В., **Антоненко А.В.** Розробка системи комп'ютерного зору у реальному часі для автономної навігації мобільного робота з використанням Edge AI на мікроконтролері STM32H743 // Збірник тез доповідей **XIX Міжнародної науково-практичної конференції «Інтегровані інтелектуальні робототехнічні комплекси» (ІІРТК-2026)**, Секція 1 «Інтегровані інтелектуальні робототехнічні комплекси». – Київ: ДУІКТ, **19–20 травня 2026 р.**
(2) Клобуков О.В. Розробка системи комп'ютерного зору у реальному часі для автономної навігації мобільного робота з використанням Edge AI на мікроконтролері STM32H743 // Збірник тез доповідей **IV Всеукраїнської науково-теоретичної конференції до Дня науки України «Держава і суспільство: сучасні виклики та пошук рішень»**. – Київ: Київський фаховий коледж туризму та готельного господарства, **13 травня 2026 р.**

Інженерну здійсненність автономного Edge AI на STM32 — доведено.

// висновки Підсумок дипломної роботи STM32 CV · Захист 2026

16 / 16 • ПЕРСПЕКТИВИ · ДЯКУЮ

ПЕРСПЕКТИВИ РОЗВИТКУ

Інтеграція	FreeRTOS	для багатозадачності та диспетчеризації
Розширення до	10–20 класів	з ієрархічною класифікацією
Розробка власної	PCB	з оптимізованою топологією, замість NUCLEO
Інтеграція	MQTT	для IoT-сценаріїв та віддаленої телеметрії

// Готовий до запитань комісії // oleg.klobukov · KID-42 · 2026

// END OF PRESENTATION

ДЯКУЮ ЗА УВАГУ!

// fin Q&A Олег Клобуков · STM32 CV · Київ 2026