Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ДОСЛІДЖЕННЯ ТА РЕАЛІЗАЦІЯ МЕТОДУ СУПЕРПІКСЕЛЬНОЇ ОБРОБКИ ЗОБРАЖЕНЬ З ВИКОРИСТАННЯМ FPGA
(тема)

Виконала:
студентка 2 курсу, групи ІНФм-20-1
Лисенко А.О.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Кобилін О.А.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2021 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Лисенко А.О.
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження та реалізація методу суперпіксельної обробки зображень з використанням FPGA

затверджена наказом по університету від «22» жовтня 2021 року № 1574Ст.

2. Термін подання студентом роботи до екзаменаційної комісії ___ _____ 2021 р.

3. Вихідні дані до роботи Теоретичні відомості про методи обробки зображень; Результати дослідження методу суперпіксельної обробки зображень; Теоретичні відомості про методи розпізнавання відеозображень з використанням програмованих вентильних логічних матриць (FPGA)

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд основних методів обробки зображень.
2. Аналіз методу сегментації зображень.
3. Дослідження методу суперпіксельної обробки зображень.
4. Реалізація методу суперпіксельної обробки з використанням FPGA.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність проблеми обчислювальної складності під час обробки зображень, постановка задачі реалізації методу суперпіксельної обробки з використанням програмованих логічних інтегральних схем (FPGA), графічні результати використання методу суперпіксельної обробки зображень.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Консультант з дотримання діючих стандартів та норм	Доцент Бєлова Н.В.		

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	22.10.2021	
2	Аналіз завдання, підбір літератури	22.10.21-24.10.21	
3	Аналіз літератури з досліджуваної проблеми	25.10.21-27.10.21	
4	Аналіз технічних засобів суперпіксельної обробки зображень з використанням FPGA	28.10.21-30.10.21	
5	Аналіз методу суперпіксельної обробки зображень з використанням FPGA	31.10.21-03.11.21	
6	Вибір алгоритму і середовища програмної реалізації	04.11.21-06.11.21	
7	Оформлення пояснювальної записки	07.11.21-19.11.21	
8	Перевірка на плагіат	23.11.2021	
9	Рецензування	01.12.2021	
10	Підготовка презентації та доповіді	02.12.2021	
11	Занесення роботи в електронний архів	06.12.2021	
12	Попередній захист кваліфікаційної роботи	06.12.2021	

Дата видачі завдання _____ _____ 2021 р.

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 69 с., 9 рис., 45 джерел.

ОБРОБКА ЗОБРАЖЕНЬ, СЕГМЕНТАЦІЯ, СУПЕРПІКСЕЛІ, СУПЕРПІКСЕЛЬНІ АЛГОРИТМИ, ПЛІС, FPGA, HDL.

Об’єктом дослідження є послідовність різноракурсних зображень об’єктів.

Метою дослідження є огляд та аналіз існуючих методів обробки зображень для подальшого дослідження алгоритму суперпіксельної обробки зображень.

Використано методи суперпіксельної обробки, об’єднання областей, навчання математичної моделі для виконання суперпіксельної сегментації. Проведено дослідження основних методів обробки зображень, алгоритму первинної та суперпіксельної сегментації зображень. Досліджено алгоритм формування суперпіксельного представлення зображення, розроблено алгоритм суперпіксельної обробки з використанням програмованих логічних інтегральних схем.

У результаті проведеного дослідження запропоновано суперпіксельний алгоритм обробки відеопотоку на основі FPGA.

IMAGE PROCESSING, SEGMENTATION, SUPERPIXELS, SUPERPIXEL ALGORITHMS, FPGA, HDL.

The object of research is a sequence of different-view images of objects.

The purpose of the study is to review and analyze existing methods of image processing for further study of the algorithm of superpixel image processing.

Methods of superpixel processing, combining areas, learning a mathematical model to perform superpixel segmentation are used. A study of the main methods of image processing, the algorithm of primary and superpixel image segmentation. The algorithm of formation of superpixel representation of the image is investigated, the algorithm of superpixel processing with the use of programmable logic integrated circuits is developed.

As a result of the conducted research, the superpixel algorithm of processing a video stream based on FPGA is offered.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 7
Вступ 10
1 Огляд основних методів обробки зображень 11
1.1 Підвищення контрасту 11
1.2 Згладжування шумів 12
1.3 Підкреслення меж 13
1.4 Медіанна фільтрація 14
1.5 Виділення об’єкта 15
1.6 Сегментація зображень 16
1.6.1 Порогові методи сегментації 17
1.6.2 Методи нарощування областей 19
1.6.3 Виділення контурів 20
1.6.4 Методи, що базуються на кластеризації 21
1.6.5 Методи з використанням гістограми 22
1.6.6 Методи розростання областей 23
1.6.7 Методи розрізу графа 24
1.6.8 Сегментація методом вододілу 24
1.7 Метод текстурних дескрипторів 26
1.8 Постановка задачі 30
2 Дослідження методу суперпіксельної обробки зображень 32
2.1 Оцінка суперпіксельної сегментації 33
2.2 Алгоритм формування суперпіксельного представлення зображення 36
2.3 Загальні вимоги до вмісту суперпікселів 40
2.4 Процедура об’єднання областей 43
2.5 Предикати однорідності при об’єднанні областей та перерахунок ознак 44

2.6 Ілюстрація застосування процедури укрупнення суперпікселів. .. 47
3 Реалізація методу суперпіксельної обробки з використанням FPGA 50
3.1 Обґрунтування вибору апаратного середовища 50
3.2 Обґрунтування вибору алгоритму і середовища програмної реалізації 53
3.3 Конструкції FPGA з використанням HDL 56
3.4 Реалізація обробки відеопотоку на основі FPGA 59
3.5 Метод навчання математичної моделі 60
Висновки 63
Перелік джерел посилання 64

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ГГц – одиниця виміру частоти (1 ГГц = 10^9 Гц)

МГц – одиниця виміру частоти (1 МГц = 10^6 Гц)

ПЗ – програмне забезпечення

ПК – персональний комп’ютер

ПЛІС – програмована логічна інтегральна схема

ЦП – центральний процесор

ALU – Arithmetic Logic Unit, блок процесора, який під керуванням пристрою керування служить для виконання арифметичних та логічних перетворень над даними

API – Application Programming Interface, опис способів, якими одна комп’ютерна програма може взаємодіяти з іншою програмою

ASIC – Application-Specific Integrated Circuit, інтегральна схема, спеціалізована для вирішення конкретного завдання

BIOS – Basic Input/Output System, набір мікропрограм, що реалізують API для роботи з апаратурою комп’ютера та підключеними до нього пристроями

BSDS – Bipolar Spectrum Diagnostic Scale, діагностична шкала біполярного спектру

CAD – Computer-Aided Design, програмне забезпечення, яке автоматизує працю інженера-конструктора та дозволяє вирішувати завдання проектування виробів та оформлення технічної документації за допомогою персонального комп’ютера

CPU – Central Processing Unit, центральний процесор, який відповідає за обробку всіх даних

DSP – Digital Signal Processors, цифровий сигнальний процесор

FPGA – Field-Programmable Gate Array, програмована користувачем вентильна матриця або програмована логічна інтегральна схема

GPU – Graphics Processing Unit, графічний процесор розташований на відеокарті, який виконує обробку 2D або 3D графіки

HDL – Hardware Description Language, мова опису апаратних засобів

HOG – Histogram of Oriented Gradients, гістограма спрямованих градієнтів

HSV – Hue, Saturation, Value; колірна модель, в якій координатами кольору є тон, насиченість та значення кольору

JPEG – Joint Photographic Experts Group, один з популярних растрових графічних форматів, що використовується для зберігання фотографій та подібних до них зображень

LAB – система задання кольорів, що використовує як параметри світлосилу, відношення зеленого до червоного та відношення синього до жовтого.

LVDS – Low Voltage Differential Signaling, метод передачі цифрових даних диференціальними сигналами з малими перепадами рівня

MAC – Multiply And Accumulate, клас команд DSP з двома операндами, зазвичай використовується для виконання основної операції скалярного твору у фільтрах з кінцевою імпульсною характеристикою

MOPS – Million Operations per Second, мільйон операцій за секунду

NTSC – National Television System Committee, система аналогового кольорового телебачення

OEM – Original Equipment Manufacturer, виробник комплектного обладнання

PCIe – Peripheral Component Interconnect Express, комп’ютерна шина, що використовує програмну модель шини PCI та високопродуктивний фізичний протокол, заснований на послідовній передачі даних

RGB – Red, Green, Blue; адитивна колірна модель, що описує спосіб кодування кольору для відтворення кольорів за допомогою трьох кольорів, які прийнято називати основними

ROI – Return on Investment, коефіцієнт рентабельності

SLD – Styled Layer Descriptor, стилізований дескриптор шару

SLIC – Software Licensing Description Table, вбудована в BIOS технологія опису ліцензування програмного забезпечення

SNIC – Simple Non-Iterative Clustering, фреймворк, спрямований на підвищення обчислювальної ефективності

USB – Universal Serial Bus, послідовний інтерфейс для підключення периферійних пристроїв до обчислювальної техніки

VHDL – VHSIC (Very High Speed Integrated Circuits) Hardware Description Language, мова опису апаратури інтегральних схем

VIP – Video Image Processing, обробка відеозображення

ВСТУП

Методи обробки зображень мають дуже велике значення, тому що вони з кожним роком все більше розвиваються та вдосконалюються. Системи обробки зображень стають популярними через доступність потужних комп’ютерів для персоналу, пристроїв великого розміру з достатнім обсягом пам’яті та графічного програмного забезпечення. Обробка зображень використовується в різних програмах, таких як дистанційне зондування, медична візуалізація, військова промисловість, обробка документів, графіка та ін. Звичайними етапами обробки зображень є сканування, збереження, покращення та інтерпретація зображень.

Суперпіксельні методи широко використовуються на стадії попередньої обробки в якості методів зниження обчислювальної складності за рахунок спрощення зображень при збереженні характеристик зображень у програмах комп’ютерного зору. Зазвичай суперпікселі генеруються на основі значень пікселів, а не з урахуванням характеристик зображення. В цій роботі мається за мету проаналізувати та реалізувати метод керування розмірами та формою створених суперпікселів, враховуючи вміст зображень, а саме скорочення числа елементів зображення у його розширеному суперпіксельному поданні та в результаті можливість зниження обчислювальної складності алгоритмів обробки.

Актуальність дослідження полягає у зростаючій потребі в більш ефективних методах обробки зображень, які можна було б застосовувати в різних додатках, таких як автоматичний візуальний огляд, сегментація зображення, відстеження об’єктів, що рухаються, мультимедійні обчислення, стиснення зображень та відео, пошук зображень на основі контенту та інших.

1 ОГЛЯД ОСНОВНИХ МЕТОДІВ ОБРОБКИ ЗОБРАЖЕНЬ

Обробка зображень є не тільки поліпшенням зорового сприйняття зображень, але й також класифікацією об’єктів, що виконується внаслідок аналізу зображень. Отже, перш ніж перейти до етапу аналізу, зображення повинне пройти певний етап препарування, який в свою чергу полягає у проведенні операцій для покращення візуальної якості (підвищення контрасту, підкреслення меж, усунення розмитості, фільтрація) та операцій з підготовки графічних зображень (виділення об’єкта, сегментація, виділення контурів).

1.1 Підвищення контрасту

Зазвичай слабкий контраст викликається невеликим динамічним діапазоном зміни яскравості, або занадто сильною нелінійністю під час передачі рівнів яскравості. Найбільш простим методом контрастування є функціональне відображення градацій яскравості f_{ij} в g_{ij}, тобто $g_{ij} = R(f_{ij})$. На практиці дуже часто використовуються лінійні функціональні відображення, наприклад, якщо під час фотографування або виготовлення фотографій в результаті нерівномірності освітлення виникають випадки, коли контраст різних частин зображення різний. У разі зміни контрасту використовуються адаптивні алгоритми контрастування. Гарним прикладом можна зазначити алгоритм локального посилення контрасту. Експериментальні дослідження підтверджують зазначену ефективність роботи алгоритму саме у тих випадках, коли на знімку присутні області з явно заниженим чи завищеним контрастом.

Суть алгоритму підвищення контрасту полягає у тому, що зображення сприймається на вході набором певної кількості локальних областей, а області

в свою чергу обробляються з урахуванням їх особистих характеристик. Обробка проводиться в наступній послідовності. Спочатку виконується обчислення коефіцієнту посилення зрізів щільності p для кожної локальної області зображення окремо. Після цього проводиться обробка кожного пікселя зображення. Якщо p прирівнюється до одиниці, то на локальну ділянку зображення не надається ніякого впливу (якщо p відмінно від одиниці, проводиться підвищення контрасту локальної ділянки). Спочатку контраст обчислюється в аналізованій точці щодо найближчого контуру. Потім значення відносного контрасту складається з одиницею і значення, яке отримується в результаті цього, в алгоритмі приймається як коефіцієнт посилення p, після чого проводиться обчислення за наступною формулою:

$$L'_i = p * L_i - int(p * L_i / L'_{max}) * L'_{max}, \quad (1.1)$$

де L'_i – нове значення яркості;

L_i – поточна яркість зображення, яке обробляється;

L'_{max} – необхідне максимальне значення яркості зображення, яке обробляється.

1.2 Згладжування шумів

На етапі оцифрування зображення піддаються впливу імпульсного та адитивного шумів.

Адитивний шум – це деякий випадковий сигнал, який додається до корисного сигналу на виході системи (наприклад, виникнення адитивного шуму внаслідок зернистості плівки).

Імпульсний шум, навпроти, характеризується впливом на корисний сигнал тільки для окремих випадкових точок (значення результуючого сигналу приймає випадкове значення в цих точках). Імпульсний шум є

характерним для цифрових систем передачі зображень та їх зберігання, саме тому виникає потреба подавління шуму під час процесу препарування зображення.

Найлегшим способом, щоб згладити шум на зображенні, є згладжування, а саме заміна для кожного елемента значення яркості середнім значенням, знайденим по його контуру:

$$f_{ij} = (1/p) * \Sigma k_1(f_{k1}), \quad (1.2)$$

де $f_{k1} \in S_p(f_{ij})$ – безліч точок, що належать до точки f_{ij} (включаючи і саму точку f_{ij});

p – число точок на контурі.

Метод, який було розглянуто вище, ефективно усуває імпульсний та адитивний шум для кожного елементу зображення.

1.3 Підкреслення меж

Методи згладжування зображень здатні дуже ефективно усувати шум. Суттєвим недоліком алгоритмів згладжування є змазаність зображення (пониження чіткості контурних елементів), при цьому величина нечіткості є пропорційною розміру маски, яка використовується для виконання операції згладжування. Для однозначного аналізу зображень є дуже важливим усунення змазаності з контурів об’єктів на зображенні, особливо при обчисленні геометричних характеристик структурних елементів. Це можна реалізувати через посилення різниці між градаціями яркості контурних елементів об’єкта та сусідніх елементів фону. В такому випадку для обробки зображень використовуються методи підкреслення меж.

Зазвичай підкреслення меж виконується методом високочастотної просторової фільтрації. Характеристики фільтрів задаються у вигляді маски, для якої середнє значення повинне дорівнювати нулю.

Ще одним методом підкреслення меж є так зване статичне диференціювання. Суть цього методу полягає в тому, що значення яркості кожного елемента ділиться на статистичну оцінку середньоквадратичного відхилення σ_F, тобто $g_{ij} = f_{ij}/\sigma_F$ (середньоквадратичне відхилення обчислюється в області елемента f_{ij}).

1.4 Медіанна фільтрація

Медіанна фільтрація належить до нелінійних методів обробки зображень та має деякі важливі переваги перед лінійною фільтрацією (яка є класичною процедурою згладжування): зберігає різкі перепади (кордони), ефективно згладжує імпульсний шум, не змінює яркість фону. Зображення часто мають піковий шум (наприклад, двійковий симетричний канал), і медіанна фільтрація ефективно знижує його.

Медіанний фільтр – це простий, але дуже потужний нелінійний фільтр. Він використовується задля зменшення кількості варіацій інтенсивності між одним пікселем та іншим пікселем. У цьому фільтрі значення пікселя замінюється середнім значенням. Медіана обчислюється шляхом сортування спочатку всіх значень пікселів у порядку зростання, а потім заміною пікселя, що обчислюється, на середнє значення пікселя. Оцінка продуктивності медіанних фільтрів зазвичай виконується шляхом вимірювання їх впливу на одновимірну оцінку розташування країв, двовимірне виявлення країв та збереження форми об’єкта, а також порівнянням з ефектами усередненого фільтра.

Медіанний фільтр розглядає кожен піксель зображення по черзі і звертає увагу на його найближчих сусідів, щоб вирішити, чи є вони

репрезентативними для свого оточення. Замість того, щоб просто замінювати значення пікселя середнім значенням сусідніх пікселів, медіанний фільтр замінює його на медіанне значення всіх значень. Медіана обчислюється шляхом сортування всіх значень пікселів в числовому порядку, а потім заміни пікселя, що розглядається, середнім значенням пікселя. (Якщо розглянута область містить парну кількість пікселів, використовується середнє значення двох середніх пікселів).

Медіанна фільтрація виконується шляхом руху деякої маски (апертури) вздовж дискретного зображення та заміною значення центрального елемента маски медіаною (середнім значенням впорядкованої послідовності) вихідних елементів усередині маски. У загальному випадку, маска може мати найрізноманітнішу форму, але, зазвичай, у практичних задачах найчастіше застосовується квадратна апертура розміром $(2k+1)*(2k+1)$, де $k = 1,2,\ldots$

1.5 Виділення об’єкта

Для виділення об’єкта існує безліч різних алгоритмів. Здатність алгоритмів функціонувати та видавати коректний результат залежить від різних параметрів вхідних даних. У зв’язку з цим виникає потреба використовувати не один алгоритм на вирішення поставленої задачі, а виявити кілька найефективніших і коректних алгоритмів із низки існуючих та комбінувати кілька алгоритмів задля досягнення найкращого результату.

Методи виділення об’єкта з відеопотоку можна розділити на три основні категорії: попіксельні, поблочні та засновані на мінімізації функціоналу енергії з усього зображення.

Попіксельні методи обробляють усі точки зображення незалежно. Зазвичай, спочатку вони будують колірну модель фону і під час роботи оцінюють, наскільки поточний колір пікселя їй відповідає. Подібні методи

зазвичай мають досить високу швидкість роботи, проте чутливі до шуму і не враховують пов’язаність розмітки (серед більшості пар сусідніх пікселів або обидва належать фону, або передньому плану).

Поблочні методи обробляють не окремі пікселі, а групи пікселів, об’єднані в блоки. Незважаючи на те, що використання таких методів дозволило в деяких випадках досягти високої швидкості роботи, подібний підхід навіть теоретично не дозволяє досягти ідеальної точності, через те що до фону чи переднього плану належить одночасно весь блок пікселів. Якщо блок містить кордон, то в таких областях межа об’єкта визначається свідомо неточно.

Методи, що засновані на мінімізації функціоналу енергії з усього зображення дозволяють використовувати інформацію з усього зображення в сукупності всіх ознак, включаючи інформацію про градієнт яскравості між сусідніми пікселями. Такий підхід дозволяє врахувати, наприклад, той факт, що межа об’єкта частіше поділяє пікселі, що сильно відрізняються за кольорами, ніж схожі за кольором. За рахунок обліку пікселів у сукупності ці методи виявляються менш чутливими до шуму в зображенні, ніж попіксельні методи, і дають більшу точність, ніж поблочні методи. Недоліком методів на основі функціоналу енергії є низька швидкість роботи, яка визначається низькою швидкістю методів мінімізації енергії.

1.6 Сегментація зображень

Під сегментацією зображень розглядається процес його розбору на складові, що мають змістовний сенс: об’єкти, їх межі або інші інформативні фрагменти, характерні геометричні особливості та інші. У розрізі автоматизації методів отримання зображень сегментацію потрібно розглядати як початковий та основний етап аналізу, який полягає у побудові формального

опису зображення, якість виконання котрого обумовлює успіх розв'язання задач інтерпретації та розпізнавання об'єктів.

У загальному випадку сегментація є операцією розбиття кінцевої множини площини, на котрій визначена функція вихідного зображення $f(x, y)$ на k непустих зв'язаних підмножин s_i $(i = \overline{1, k})$ відповідно до деякого предикату P, що визначається на множині $S = \{s_1, s_2, \dots, s_k\}$ і приймає істинні значення, коли будь-яка пара точок з кожного підмножини s_i задовольняє деякому критерію однорідності (як приклад, критерій однорідності, що заснований на оцінці максимальної різниці яскравості окремого пікселя і середнього значення яскравості, обчисленого по відповідній області).

1.6.1 Порогові методи сегментації

Порогова обробка – один з основних методів сегментації зображень завдяки власним інтуїтивно зрозумілим властивостям. Метод порогової обробки орієнтований на обробку зображень, в яких окремі однорідні області відрізняються середньої яскравістю. Найпоширенішим методом сегментації шляхом порогової обробки є бінарна сегментація, тобто коли у розпорядженні є два типи однорідних ділянок.

У такому випадку зображення обробляється за пікселями і перетворення кожного пікселя вхідного зображення $f_0(x, y)$ у вихідне $f_1(x, y)$ визначається з співвідношення:

$$f_1(x, y) = \begin{cases} z_0, \text{якщо } f_0(x, y) \leq T, \\ z_1, \text{якщо } f_0(x, y) > T, \end{cases} \quad (1.3)$$

де T – параметр обробки, званий порогом;

z_0 та z_1 – рівні вихідної яркості.

Обробка по пікселям, положення яких на зображенні не відіграє жодної ролі, називається точковою. Рівні z_0 та z_1 відіграють роль міток. За ними визначають, до котрого типу віднести цю точку: до H_0 чи H_1. Кажуть, що H_0 складається з фонових точок, а H_1 з точок інтересу. Як правило, рівні z_0 та z_1 відповідають рівням білого та чорного. Будемо називати класи H_1 класом об'єкта, а клас H_0 класом фону.

Звичайно сегментація може бути не тільки бінарною і в такому випадку існуючих класів більше, ніж два. Такий вид сегментації називається багаторівневим. Результуюче зображення не є бінарним, але воно складається із сегментів різної яскравості. Формально цю операцію можна записати так:

$$f_1(x,y) = \begin{cases} 1, \text{якщо } f_0(x,y) \in D_1; \\ 2, \text{якщо } f_0(x,y) \in D_2; \\ \dots \\ n, \text{якщо } f_0(x,y) \in D_n; \\ 0, \text{в інших випадках.} \end{cases} \tag{1.4}$$

де n – кількість рівнів;

D_i, $i = \overline{1,n}$ – класи зображення.

У такому випадку для кожного з класів має бути заданий відповідний поріг, який би відокремлював ці класи між собою. Бінарні зображення легше зберігати та обробляти, ніж зображення, в яких присутні багато рівнів яскравості.

Найскладнішим у пороговій обробці є процес визначення порога T. Поріг T часто записують як функцію, що має вигляд:

$$T = T(x, y, l(x, y), f), \tag{1.5}$$

де f – зображення;

$l(x, y)$ – деяка характеристика точки зображення (x, y), наприклад, середня яскравість навколо з центром у цій точці.

Якщо значення порога T залежить лише від f, тобто однаково всім точок зображення, такий поріг називають глобальним. Якщо поріг T залежить від просторових координат (x, y), такий поріг називається локальним. Якщо T залежить від характеристики $l(x, y)$, тоді такий поріг називається адаптивним. Таким чином, обробка вважається глобальною, якщо вона відноситься до всього зображення в цілому, та локальною, якщо вона відноситься до певної виділеної області.

Крім перерахованих алгоритмів розмежувань, існує ще безліч методів. Більшість з них є деякою сукупністю інших, але більшість з них, так чи інакше, базуються на аналізі гістограми вихідного зображення. Однак є й принципово інші підходи, які не торкаються аналізу гістограм у прямому вигляді або переходять від них до аналізу деяких інших функцій.

1.6.2 Методи нарощування областей

Методи цієї групи ґрунтуються на використанні локальних ознак зображення. Ідея методу нарощування областей полягає в аналізі спочатку стартової точки, потім її сусідніх точок відповідно до деякого критерію однорідності, і в подальшому зарахуванні проаналізованих точок в ту чи іншу групу (кількість стартових точок має дорівнювати кількості однорідних областей на зображенні). У більш ефективних варіантах методу в якості відправної точки використовуються не окремі пікселі, а розбиття зображення на ряд невеликих областей. Потім кожна область перевіряється на однорідність, і, якщо результат перевірки виявляється негативним, то відповідна область розбивається на більш дрібні ділянки. Процес триває доти, доки всі виділені області не пройдуть перевірку на однорідність. Після цього починається формування однорідних областей за допомогою нарощування.

Порогова сегментація та сегментація за критерієм однорідності на основі середньої яркості часто не дає бажаних результатів. Така сегментація зазвичай

призводить до появи значної кількості невеликих областей, які не мають реальних прототипів на зображенні. Найбільш ефективні результати дає сегментація за критерієм однорідності на основі текстури (або текстурних ознак).

1.6.3 Виділення контурів

Нерідко доводиться стикатися із завданням знаходження периметрів, кривизни, факторів форми, питомої поверхні об’єктів тощо. Усі ці завдання так чи інакше пов’язані з аналізом контурних елементів об’єктів.

Методи виділення контурів (меж) на зображенні можна розділити на три основні класи: методи високочастотної фільтрації, методи просторового диференціювання та методи функціональної апроксимації.

Спільним для всіх цих методів є прагнення розглядати межі в якості області різкого перепаду функції яскравості зображення $f(i,j)$, їх відрізняє введена математична модель поняття контуру і алгоритм пошуку граничних точок.

Відповідно до поставлених завдань до алгоритмів виділення контурів пред’являються наступні вимоги: виділені контури повинні бути утонченими, без розривів і замкнутими. Таким чином, процес виділення контурів дещо ускладняється у зв’язку з необхідністю застосовувати алгоритми потоншення та усунення розривів. Однак і це не завжди дає бажаного результату – у більшості випадків контури виходять незамкненими і, як наслідок, непридатними для процедур аналізу.

Розв’язати завдання можна, виконуючи оконтурювання алгоритмом простеження кордонів методом «жука», який дозволяє виділити замкнуті контури об’єктів. Суть алгоритму полягає в наступному: на об’єкті вибирається деяка стартова гранична точка і далі відбувається послідовне простеження контуру доти, доки не буде досягнуто стартової точки. У разі

простеження контуру за годинниковою стрілкою для досягнення стартової точки здійснюється за піксельним рухом праворуч, якщо піксель знаходиться поза об’єктом, і ліворуч, якщо на об’єкті.

Виділений в такий спосіб контур є замкнутим ланцюговим кодом, тобто послідовністю координат граничних точок об’єкта, що дуже зручно для вирішення поставлених задач.

1.6.4 Методи, що базуються на кластеризації

Метод K-середніх – це ітеративний метод, який використовується для розділення зображення на K кластерів [4-6]. Базовий алгоритм наведений нижче:

Крок 1. Вибрати K центрів кластерів, випадково чи на підставі деякої евристики.

Крок 2. Помістити кожен піксель зображення в кластер, центр якого є найближчим до цього пікселя.

Крок 3. Заново обчислити центри кластерів, усереднюючи всі пікселі в кластері.

Крок 4. Повторювати кроки 2 і 3 до моменту збіжності (наприклад, коли пікселі залишатимуться у тому ж кластері).

За відстань, зазвичай, береться сума квадратів або абсолютних значень різниць між пікселем і центром кластера. Різниця, частіше за все, заснована на кольорі, яскравості, текстурі та розташуванні пікселя, або на виваженій сумі цих факторів. K можна вибрати вручну, випадково або евристично. Цей алгоритм гарантовано сходиться, але він може призвести до оптимального рішення. Якість рішення залежить від початкової множини кластерів і значення K [8-10].

1.6.5 Методи з використанням гістограми

Методи з використанням гістограми є дуже ефективними, коли порівнюються з іншими методами сегментації зображень, тому що вони потребують лише одного проходу пікселів. У цьому методі гістограма обчислюється за всіма пікселями зображення та її мінімуми та максимуми використовуються, щоб знайти кластери на зображенні. Колір або яскравість можна використовувати при порівнянні.

Поліпшення цього – рекурсивно застосовувати його до кластерів на зображенні у тому, щоб поділити їх на дрібніші кластери. Процес повторюється поступово з меншими і ще меншими кластерами до того моменту, коли нові кластери перестають з’являтися взагалі.

Певним недоліком цього методу є те, що йому може бути важко знайти мінімуми та максимуми на зображенні. У цьому методі класифікація зображень схожа на метрику відстаней і метод зіставлення інтегрованих регіонів [13].

Підходи, засновані на використанні гістограм, можна також швидко адаптувати для декількох кадрів, зберігаючи їх перевагу в швидкості за рахунок одного проходу. Гістограма може бути побудована декількома способами, в яких розглядаються кілька кадрів. Той самий підхід, який використовується для одного кадру, може бути застосований для декількох кадрів, і після об’єднання результатів, мінімуми та максимуми, які складно виділяються, стають помітнішими. Гістограма також може бути застосована для кожного пікселя, де інформація використовується для визначення найчастішого кольору для певного положення пікселя. Цей підхід використовує сегментацію, засновану на об’єктах у русі, і нерухомому оточенні, що дає інший вид сегментації, корисний у відеотрекінгу [19].

1.6.6 Методи розростання областей

Першим був метод розростання областей під назвою SEEDS. Як вхідні дані цей метод приймає зображення і певний набір зернистості. Метод SEEDS відзначає об’єкти, які потрібно виділити. Області поступово розростаються, порівнюючи всі незайняті сусідні пікселі з областю. Різниця між яскравістю пікселя і середньою яскравістю області використовується як міра схожості. Піксель з найменшою такою різницею додається до відповідної області. Процес триває доки всі пікселі не будуть додані до одного з регіонів.

Метод розростання областей SEEDS вимагає додаткового введення. Результат сегментації залежить від вибору зернистості. Шум на зображенні може спричинити те, що область SEEDS погано розміщена. Метод розростання областей без використання SEEDS – це змінений алгоритм, який не потребує явної зернистості. Він починає свою роботу з однієї області A_1 – піксель, обраний на цьому етапі, впливає на кінцеву сегментацію. На кожній ітерації він розглядає сусідні пікселі так само, як метод розростання з використанням SEEDS. Але він відрізняється тим, що якщо мінімальна δ не менше, ніж заданий поріг T, він додається у відповідну область A_j. В іншому випадку піксель вважається сильно відмінним від усіх поточних областей A_i і створюється нова область A_{n+1}, що містить цей піксель.

Один із варіантів цього методу заснований на використанні яскравості пікселів. Середнє квадратичне відхилення, дисперсія області та яскравість пікселя-кандидата використовується для побудови тестової статистики. Якщо тестова статистика достатньо мала, то піксель додається до області, і середнє квадратичне відхилення з дисперсією області перераховується. Інакше піксель ігнорується і використовується для створення нової області.

1.6.7 Методи розрізу графа

Методи розрізу графа ефективно застосовуються для виконання сегментації зображень. Для цих методів зображення подається як зважений неорієнтований граф. У більшості випадків, піксель чи група пікселів асоціюються вершиною, а ваги ребер визначають схожість чи несхожість сусідніх пікселів. Після цього граф розрізається згідно з критерієм, що був створений для отримання «хороших» кластерів. Кожна частина вершин (пікселів), які одержуються внаслідок роботи цих алгоритмів, вважається об’єктом зображення.

Перед проведенням сегментації користувач вказує на вихідному зображенні маску зображення – прямокутну область, в якій знаходиться регіон, що цікавить користувача, а також маркери пікселів, що належать об’єкту і пікселям, що належать фону. Зазвичай піксель або група пікселів асоціюється з вершиною графа, а схожість або несхожість сусідніх пікселів, що визначається, наприклад, за допомогою обчислення зміни градієнта, визначає ваги ребер графа. Потім граф (зображення) розрізається згідно з критерієм (маскою), створеним для отримання бажаних кластерів. Кожна частина вершин (пікселів), одержувана цими алгоритмами, вважається об’єктом зображення. Якщо отриманий результат задовільний, що визначає користувач, алгоритм припиняється. Інакше він повторюється доти, доки не буде досягнуто бажаного результату.

1.6.8 Сегментація методом вододілу

У випадку сегментації методом вододілу зображення розглядається в якості певної карти місцевості, де значення яскравостей є значеннями висот відносно деякого рівня. Якщо цю область заповнювати водою, тоді утворюються так названі басейни. При подальшому заповненні водою ці

басейни об’єднуються. Місця об’єднання цих басейнів відзначаються як лінії вододілу.

У такій інтерпретації розглядаються точки трьох видів:

1. Локального мінімуму;
2. Точки, що є на схилі, тобто ті, з яких вода прямує в той самий зазначений локальний мінімум;
3. Точки локального максимуму, тобто ті, з яких вода прямує більш ніж в один мінімум.

Поділ предметів, що стикаються на зображенні, є одним з важливих завдань обробки зображень. Часто на вирішення цього завдання використовується так званий метод маркерного вододілу. При перетвореннях з допомогою цього методу необхідно визначити «водозбірні басейни» і «лінії вододілу» на зображенні шляхом обробки локальних областей залежно від власних характеристик яскравості.

Метод маркерного вододілу є одним із найбільш ефективних методів сегментації зображень.

Під час реалізації цього методу виконуються такі основні процедури:

Крок 1. Обчислюється функція сегментації. Вона стосується зображень, де об’єкти розміщені в темних областях і важко помітні.

Крок 2. Обчислюються маркери переднього плану зображень. Вони обчислюються внаслідок аналізу зв’язності пікселів кожного об’єкта.

Крок 3. Обчислюються фонові маркери. Вони є пікселями, які в свою чергу є частинами об’єктів.

Крок 4. Модифікуються функції сегментації на підставі значень розташування маркерів фону та маркерів переднього плану.

Одним з найважливіших застосувань при сегментації методом вододілу є виділення на фоні зображення однорідних за яскравістю об’єктів (у вигляді плям). Області, які характеризуються невеликими варіаціями яскравості, мають малі значення градієнта. Тому на практиці часто зустрічається ситуація,

коли сегментація методом вододілу застосовується не до самого зображення, а до його градієнта.

1.7 Метод текстурних дескрипторів

Наша зорова система здатна легко розпізнавати і розрізняти текстури. Набагато складніше охарактеризувати та виділити «дифузні» властивості текстури за допомогою точно визначених параметрів, які дозволяють комп’ютеру виконувати це завдання.

Довільна структура, яка поширюється на велику область зображення, безумовно, не розпізнається як текстура. Таким чином, основною властивістю текстури є мала елементарна структура, яка періодично або квазіперіодично повторюється в просторі. Таким чином, достатньо описати малу елементарну структуру та правила повторення. Останні задають характеристичний розмір текстури.

Аналіз текстури можна порівняти з аналізом структури твердих тіл – тематикою, що вивчається у фізиці твердих тіл, хімії та мінералогії. Фізики в області твердих тіл повинні визначити структуру, що повторюється, і розподіл атомів в елементарному осередку. Аналіз структури ускладнюється тим, що і структури, і періодичне повторення можуть виявляти суттєві випадкові флуктуації.

Текстури можна організовувати ієрархічно, тобто вони можуть виглядати зовсім по-різному у різних масштабах. Хорошим прикладом є текстура, показана на рисунку 1.1. У найдрібнішому масштабі увага фокусується на окремих нитках. Тоді характеристичним розміром є товщина ниток. Вони також мають переважну локальну орієнтацію.

Рисунок 1.1

На наступному більш грубому рівні розпізнаються петлі. Характеристичний розмір показує розмір петель. На цьому рівні локальна орієнтація добре розподілена.

І, нарешті, на максимально грубому рівні більше не розпізнаються окремі петлі, але спостерігаються складки тканини. Вони характеризуються ще одним характеристичним розміром, що показує період складок та їхню орієнтацію. Ці міркування наголошують на суттєвості великомасштабного текстурного аналізу. Отже, великомасштабні структури даних є важливими для текстурного аналізу.

Уявимо завдання, в якому необхідно розпізнати об’єкти, які випадково орієнтовані в зображенні. Нас не цікавить орієнтація об’єктів, але цікавить їх відмінність один від одного. Отже, текстурні параметри, які залежать від орієнтації, не становлять інтересу. Можна було б, як і раніше, використовувати їх, але тільки якщо об’єкти мають форму, яка потім дозволить визначити їхню орієнтацію. Можна використовувати такі аргументи для ознак, інваріантних щодо масштабу. Якщо об’єкти, що цікавлять нас, розташовані на різних відстанях від камери, текстурний параметр, який використовується для їх розпізнавання, повинен бути також інваріантним щодо масштабу. В іншому випадку розпізнавання об’єкта буде залежати від відстані. Однак, якщо текстура змінює свої характеристики в залежності від масштабу, текстурні

ознаки, інваріантні щодо масштабу, можуть не існувати взагалі. Тоді використання текстур для опису об’єктів на різних відстанях стає складним завданням.

Орієнтація поверхонь є ключовою ознакою іншого завдання обробки зображень – реконструкції тривимірної сцени по двовимірному зображенню. Якщо відомо, що поверхня об’єкта показує однорідну структуру, можна аналізувати орієнтацію та розміри текстури для знаходження орієнтації поверхні у просторі. Для цього необхідні характеристичні розміри та орієнтація текстури.

Текстурний аналіз є однією з тих областей в обробці зображень, щодо якої ще немає фундаментальних знань. Тому література містить багато різних емпіричних та напівемпіричних підходів до текстурного аналізу. Протилежно видається досить простий підхід до текстурного аналізу, який будує складні текстурні оператори по елементарним операторам.

Для текстурного аналізу використовуються лише чотири фундаментальні текстурні оператори:

- середнє значення;
- дисперсія;
- орієнтація;
- масштаб,

які застосовуються на різних рівнях ієрархічної структури послідовності обробки зображень. Так як було обчислено локальну орієнтацію і локальний масштаб, оператор усереднення і оператор дисперсії можна застосовувати знову, але цього разу не до середнього значення та дисперсії рівнів яскравості, а до локальної орієнтації та масштабу.

Ці чотири основні текстурні оператори можна згрупувати у два класи. Середнє значення та дисперсія є незалежними від повороту та масштабу, у той час як оператори орієнтації та масштабу лише визначають орієнтацію та масштаб відповідно. Цей важливий поділ між параметрами, інваріантними або неінваріантними щодо масштабу та повороту значно спрощує текстурний

аналіз. Значення цього підходу полягає в ортогональності набору параметрів, простоті та можливості їх застосування.

Один із найпростіших підходів, які застосовуються для опису текстури, полягає у використанні статистичних характеристик, що визначаються за гістограмою яскравості всього зображення або його області. Нехай z – випадкова величина, що відповідає яскравості елементів зображення, а $p(z_i), i = 0,1,2, \dots, L-1$ – її гістограма, де L позначає кількість різних рівнів яскравості. Тоді центральний момент порядку n випадкової величини z дорівнює:

$$\mu_n(z) = \sum_{i=0}^{L-1}(z_i - m)^n p(z_i), \tag{1.6}$$

де m – середнє значення z (середня яскравість зображення):

$$m = \sum_{i=0}^{L-1} z_i p(z_i). \tag{1.7}$$

З цих рівнянь видно, що $\mu_0 = 1$ та $\mu_1 = 0$. Для опису текстури особливо важливим є другий момент, тобто дисперсія $\sigma^2(z) = \mu_2(z)$. Вона є мірою яскравості контрасту, яку можна використовувати для побудови дескрипторів відносної гладкості. Наприклад, величина:

$$R = 1 - \frac{1}{1+\sigma^2(z)}, \tag{1.8}$$

дорівнює 0 для областей постійної яскравості (де нульова дисперсія) і наближається до 1 для великих значень $\sigma^2(z)$.

Третій момент

$$\mu_3(z) = \sum_{i=0}^{L-1}(z - m)^3 p(z_i), \tag{1.9}$$

є характеристикою асиметрії гістограми. Серед інших корисних характеристик текстури можна виділити однорідність:

$$U = \sum_{i=0}^{L-1} p^2(z_i), \tag{1.10}$$

та середню ентропію:

$$e = \sum_{i=0}^{L-1} p(z_i)\, log_2\, p(z_i). \tag{1.11}$$

Максимум значення $U = 1$ досягається для зображення, всі елементи якого мають однакову яскравість, і зменшується зі зростанням відмінностей у яскравості. Ентропія, в свою чергу, характеризує мінливість яскравості зображення. Вона дорівнює 0 для області з постійною яскравістю та максимальна у разі зростання флуктуації z.

1.8 Постановка задачі

Таким чином можна зробити висновок, що сегментація є актуальним завданням для обробки і розпізнавання зображень. Тому ставиться завдання розробки алгоритму суперпіксельної обробки, а саме поділу зображення на групи пов’язаних областей, які є однорідними і не перекриваються (група пікселів, які також називається суперпікселями). Однією з головних особливостей використання суперпікселів є продуктивність обчислень. Уособлення суперпікселів зменшує кількість залежностей зображення порівняно з уособленням пікселів, та надає локальну підтримку для визначення регіонів залежно від характеристик зображень.

Об’єктом дослідження є послідовність різноракурсних зображень об’єктів.

Метою дослідження є глибинний аналіз методів, що базуються на різних алгоритмах обробки зображень, які в свою чергу дозволяють ефективно виконувати суперпіксельну обробку.

Для досягнення мети необхідно вирішити такі завдання:

– провести аналіз існуючих методів обробки зображень;

– розробити алгоритм знаходження та виділення суперпікселів на базі існуючих методів обробки зображень;

– реалізувати алгоритм суперпіксельної обробки зображень з використанням програмованих вентильних логічних матриць (FPGA).

2 ДОСЛІДЖЕННЯ МЕТОДУ СУПЕРПІКСЕЛЬНОЇ ОБРОБКИ ЗОБРАЖЕНЬ

Суперпіксельна обробка завжди була популярним методом включення просторових апріорних точок у широкий спектр завдань комп'ютерного зору. Вона застосовується як для впливу на результуючі статистичні моделі, так і для обчислювальних цілей. Наприклад, для логічного висновку на основі розрізів графа широко використовувалися методи суперпікселів, щоб зменшити витрати на обчислення.

Метод суперпіксельної обробки дещо втратив популярність через достатньо швидку популяризацію глибоких нейронних мереж для класифікації, сегментації та багатьох інших завдань обробки зображень. Однак статистичні та обчислювальні переваги методу суперпікселів можуть дуже вдало застосовуватись у контексті глибоких мереж. Основна причина цього в тому, що глибокі нейронні мережі, призначені для завдань комп'ютерного зору, за останні роки стали дедалі глибшими та складнішими. Хоча це дозволяє підвищити точність розв'язання безлічі завдань, це також пов'язано з підвищеними обчислювальними витратами.

Тому береться за основну мета підвищити точність мереж за рахунок спрощення процесу сегментації, а не побудови складніших моделей. Пропонується зробити це за допомогою об'єднання суперпікселів. Такий рівень має дві основні переваги: групування інформації та введення апріорної інформації у сегментований висновок. Цей підхід сприяє сегментації, що зберігає кордони, які нам надають суперпікселі. Застосування алгоритму сегментації має за мету швидке формування суперпіксельного представлення зображення, в якому з високою точністю зберігалася б інформація про яскравість. При цьому немає гарантії, що кожен суперпіксель відповідає інтерпретованій області (об'єкту) зображення, бо швидше за все, ці об'єкти виявляться розбитими на деяке число сегментів. Більш того, окремі малі

області будуть створюватися для пікселів, які істотно відрізняються від сусідніх, коли, наприклад, зображення спотворюється імпульсними шумами. Для покращення відповідності суперпікселів об’єктам може бути застосована процедура укрупнення (об’єднання) областей.

2.1 Оцінка суперпіксельної сегментації

Як стверджують Рен і Малік [33], пікселі зображення не є природними об’єктами, а висока роздільна здатність сучасних камер означає, що багато пікселів будуть нести ту ж або дуже подібну інформацію, що й їхні сусіди, що робить їх зайвими. Зараз це ще більше вірно, ніж у 2003 році, коли була опублікована їх стаття, що робить алгоритми обробки зображень, які порівнюють пікселі, невиправдано дорогими для обчислень. Їх рішення полягало в тому, щоб розділити сегментацію на два етапи, спочатку розділивши зображення на невеликі сегменти, які краще відповідають природним об’єктам зображення, які називаються суперпікселями, а потім побудувавши сегменти з цих суперпікселів.

Інші дослідники показали, що суперпіксельна сегментація може бути потужним кроком попередньої обробки не тільки для сегментації зображень, наприклад, Міяма використовує її для збільшення швидкості стереоузгодження [34], а Гу та інші показали, що це може скоротити час виконання деяких задач класифікації [35].

Одним із добре використовуваних алгоритмів, які швидким та ефективним способом генерують суперпікселі хорошої якості, є Simple Linear Iterative Clustering, SLIC [36]. Він заснований на кластеризації K-середніх і ітеративно перевизначає кластери пікселів, щоб мінімізувати відстань від пікселя до кластера. Там, де центри кластерів спочатку розміщені на однорідній сітці, а відстані вимірюються як зважена сума просторової та

колірної відстані, тобто евклідові відстані в площині XY і колірному просторі CIELAB [37].

Іншим алгоритмом, який можна вважати найсучаснішим, є Simple Non-Iterative Clustering, SNIC [38]; це вдосконалений алгоритм дослідницької групи, яка розробила SLIC. У SNIC пікселі додаються до кластерів, і так само, як і SLIC, він починається з центрів кластерів на однорідній сітці. Після того, як піксель додається до кластера, центр кластера оновлюється до середньозваженого значення його поточної позиції та нового пікселя, що дорівнює загальному середньому всіх пікселів, доданих до кластера. Тоді всі сусіди нового пікселя, які ще не належать до кластера, поміщаються в чергу пріоритетів з пріоритетом на основі відстані до поточного центру кластера. Відстань обчислюється так само, як і в SLIC, а менша відстань означає більш високий пріоритет. Потім піксель з найвищим пріоритетом витягується з черги, і якщо він ще не належить до кластера, він додається до кластера, який поставив його в чергу. Цей цикл зчитування пікселя та ідентифікатора кластера з пріоритетної черги, призначення пікселя кластеру, а потім додавання сусідніх пікселів і ідентифікатора кластера читання до черги, продовжується, доки всі пікселі на зображенні не будуть додані до кластерів. У порівнянні зі SLIC, SNIC обчислює меншу відстань, використовує менше пам’яті та не виконує ітерацій, щоб знайти рішення, але пікселі відвідуються у випадковому порядку, і потрібна складна структура даних пріоритетної черги.

SEEDS, суперпікселі, витягнуті за допомогою енергійного відбору проб, розроблених Ван ден Берг та ін. [39] є ще одним алгоритмом суперпіксельної сегментації. Спочатку він розділяє всі пікселі на кластери суперпікселів, утворюючи сітку ідеальних квадратів. Потім відбувається обмін пікселями між кластерами на кордонах, щоб максимізувати енергетичну функцію. Енергетична функція – це функція, яка виражає повну енергію системи, у SEEDS енергія кластера зазвичай виражається як деяка функція подібності пікселів у кластері. Подібно до SLIC, цей алгоритм повторюється, поки не буде

виконана умова зупинки. Це може бути змінна або фіксована кількість ітерацій. Алгоритм можна зупинити на будь-якій ітерації і все одно надати дійсний результат. Якість сегментації покращується з кількістю ітерацій, але в порівнянні з алгоритмом, таким як SNIC, який повинен виконуватися від початку до кінця, щоб вивести суперпіксельну сегментацію, SEEDS можна перервати, щоб досягти кінцевого терміну, і все одно виводити суперпікселі.

Сегментація вододілу – це клас алгоритмів сегментації, який, на відміну від попередніх трьох алгоритмів, не визначається як кластеризація пікселів. Натомість ці алгоритми визначають межі між сегментами. Зображення перетворюється на відтінки сірого, а потім обробляється як топографічна карта. Хребти на цій карті є кордонами між сегментами. Швидкий алгоритм знаходження сегментів і кордонів є засіяним вододілом, запропонованим Мейєром [40]. Він працює шляхом вирощування кластерів із початкових точок і позначення пікселів, які сусідні пікселі з більш ніж одного кластера, як межі пікселів. Кластери вирощуються подібно до SNIC з використанням черги пріоритетів. Це швидкий спосіб сегментації зображення, але він не дає сегментів однакового розміру та форми, що є важливим аспектом використання їх як суперпікселів. Для вирішення цієї проблеми Нойберт і Протцель пропонують алгоритм компактного вододілу [41]. Показник відстані, який використовується при вирощуванні кластерів, тепер також враховує просторову відстань до початкової точки, роблячи сегменти набагато одноріднішими як за розміром, так і за формою.

У статтях, де представлені алгоритми суперпіксельної сегментації, що представлені вище в якості посилань, двома найбільш використовуваними метриками для порівняння та оцінки якості суперпіксельної сегментації є запам’ятовування кордонів і помилка недостатньої сегментації. Ці показники також використовуються в стандартизованій схемі оцінки суперпіксельної сегментації, введеної Нойбертом і Протцелем [42]. Відкликання кордонів – це міра того, наскільки добре краї між сегментами збігаються з кордонами в основній істині, вона розраховується як частка від того, скільки пікселів межі

в основній істині збігаються з межовим пікселем у сегментації та загальній кількості межових пікселів. Похибка недосегментації – це міра того, наскільки сегмент витікає за межі відповідної області в основній істині. Це можна обчислити кількома різними способами, але метод, запропонований Нойбертом і Протцелем, полягає в обчисленні розміру набору різниці і перекриття для кожного сегмента істини та пари суперпікселів, а саме додавання найменшого з цих значень з кожної пари, а потім розділення на загальну кількість пар, щоб отримати середнє. Для виконання таких тестів на основі істини потрібен набір даних із сегментованими зображеннями. Набори даних, які постійно з'являються у статтях про суперпікселі, є версіями набору даних сегментації Берклі, BSDS300 [43] і BSDS500 [44]. Оригінальний BSDS300 містить 300 зображень, а продовження, BSDS500, розширено 200 зображеннями. Усі зображення в наборах даних мають кілька основних істин від ручної сегментації людьми, у середньому для кожного зображення доступно п'ять різних сегментацій.

2.2 Алгоритм формування суперпіксельного представлення зображення

В основу матеріалу покладено алгоритм сегментації стосовно завдання компресії зображень. Він входить до класу подібних алгоритмів, що пізніше отримали загальну назву «методу нарощування областей», проте вигідно відрізняється від них низькою обчислювальною складністю (отримання суперпікселів здійснюється за один прохід за зображенням) і простотою налаштування (єдиний параметр). Для повноти картини коротко опишемо принцип його роботи у більш загальної формі.

Нехай $\{x(n_1, n_2)\}_{n_1, n_2=0}^{N_1-1, N_2-1}$ – матриця пікселів вихідного напівтонового зображення, n_1, n_2 – цілочисленні індекси, N_1, N_2 – розміри зображення. Позначимо $D_\Sigma = \{0 \le n_1 \le N_1 - 1, 0 \le n_2 \le N_2 - 1\}$ безліч цілих координат

усіх пікселів. Сегментація зображення означає розбиття цієї множини на M підмножин так, що

$$D_{\Sigma} = \bigcup_{m=0}^{M-1} Dm, D_m \cap D_n = \emptyset, \forall m \neq n, \quad (2.1)$$

причому кожна підмножина D_{m} утворює на площині зображення зв'язкову область, а пікселі, що входять до неї, забезпечують істинність деякого логічного предикату однорідності, визначеного на цій підмножині:

$$Q(D_m) = True, \ \forall m, Q(D_m \cup D_n) = False, \text{для } \forall \text{ смежных } D_m \text{ и } D_n. \quad (2.2)$$

Далі нехай пікселі покривають площину зображення квадратами одиничної площі. Пара пікселів вважається суміжною, якщо відповідні квадрати мають спільну сторону. Інакше кажучи, в описуваному алгоритмі використовується відомий принцип чотирьохзв'язностей областей. Зображення обробляється послідовно, у порядку зростання індексів пікселів (зліва направо у рядку та зверху вниз по рядках).

Обробка кожного пікселя x (n_1, n_2), що не належить першому рядку і першому стовпцю матриці (тобто при $n_1 \neq 0, n_2 \neq 0$), виконується за простою схемою, яка наведена на рисунку 2.1, де через A і B позначені підмножини (області), до яких належать пікселі, вже оброблені раніше і суміжні з поточним: $x(n_1 - 1, n_2)$ та $x(n_1, n_2 - 1)$ відповідно, а через (n_1, n_2) позначено область, що складається з одного поточного пікселя.

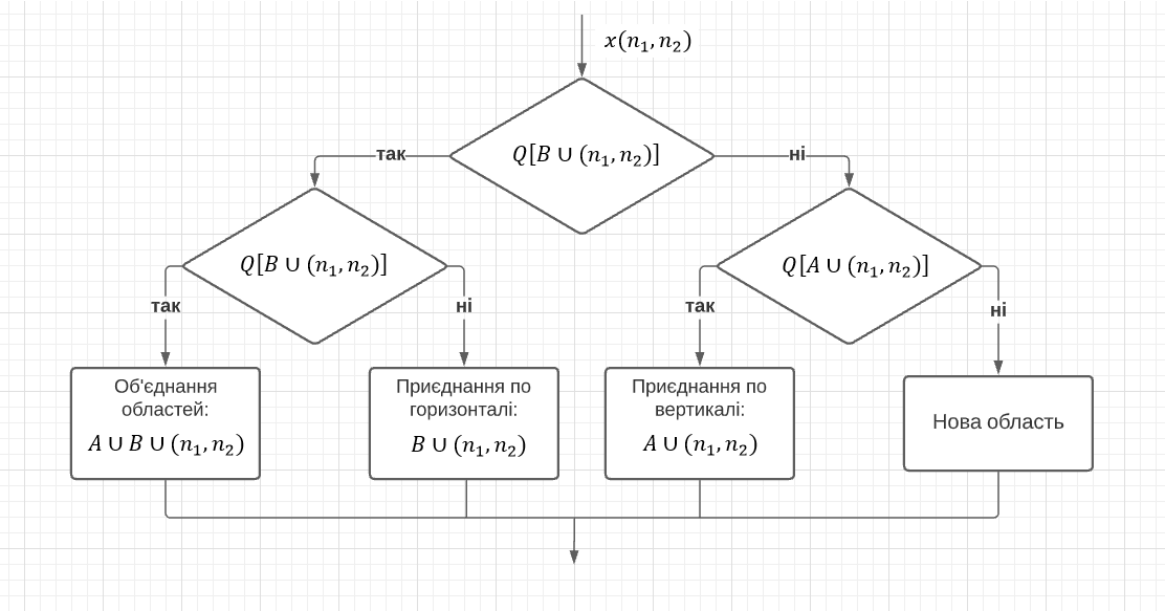


Рисунок 2.1 – Схема обробки кожного пікселя при сегментації зображення

На кожному кроці алгоритму вирішується, чи приєднати цей піксель до області A, до області B, чи він не належить ні A, ні B, а відкриває собою нову область, або ці області повинні бути об’єднані (разом з поточним пікселем). Особливості (додаткові спрощення) обробки пікселів першого рядка та першого стовпця матриці видаються очевидними. Описаний алгоритм працює з конкретним предикатом однорідності:

$$Q(D_m): \max_{(n_1,n_2)\epsilon D_m} x(n_1, n_2) - \min_{(n_1,n_2)\epsilon D_m} x(n_1, n_2) \leq 2\varepsilon_{max}, \qquad (2.3)$$

де ε_{max} – поріг сегментації (вхідний параметр алгоритму), що має сенс максимально допустимої похибки збереження значення кожного пікселя у сегментованому (суперпіксельному) поданні зображення.

Цей предикат є дуже ефективним. По-перше, він дуже просто обчислюється. По-друге, (і це головне) метою застосування алгоритму є збереження інформації, що міститься у вихідному зображенні, при істотному скороченні числа його елементів.

Проілюструємо це на прикладах тестових напівтонових зображень, показаних на рисунку 2.2. Кожне зображення містить напівтонові пікселі,

представлені в байтовому форматі (з цілими значеннями яскравості в діапазоні 0 ÷ 255).

(а) (б) (в)

Рисунок 2.2 – Тестові зображення: (а) «Зображення №1»; (б) «Зображення №2»; (в) Зображення №3

На рисунку 2.3 представлені експериментально отримані для цих зображень залежності скорочення числа елементів, досягнутого в результаті сегментації, від параметра ε_{max}, де $N = N_1 * N_2$.

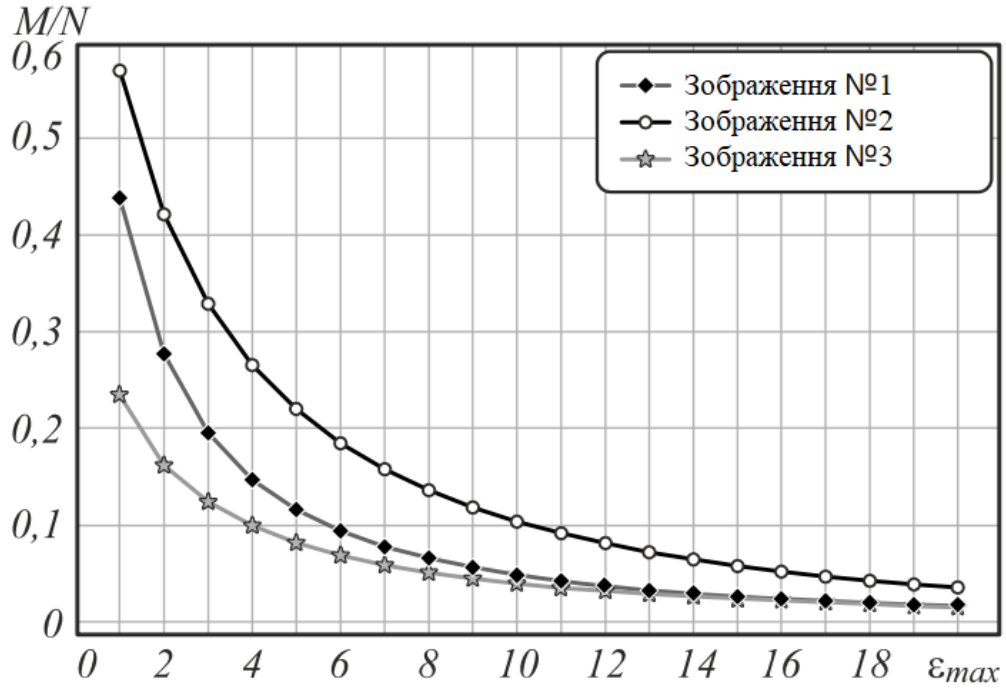


Рисунок 2.3 – Скорочення числа елементів опису зображень під час переходу до суперпіксельного подання

Видно, що суперпіксельне уявлення, яке формується описаним алгоритмом, різко скорочує число елементів опису зображень (вже при $\varepsilon_{max} = 2$ число елементів скорочується більш ніж на 50%), тобто дає ресурс підвищення обчислювальної ефективності їх подальшої обробки. Разом з тим воно зберігає досить високу інформативність вихідних растрових даних: похибка пікселів у байтовому форматі, що не перевищує 3-5 квантів функції яскравості, прийнятна для більшості практичних додатків, оскільки не породжує візуально помітних спотворень зображень, знаходиться на рівні шумів реєстрації та природних кількостей. зображених об'єктів. Узагальнення описаного алгоритму на випадок кольорових (багатокомпонентних) зображень є очевидним: просто в процесі сегментації потрібно контролювати істинність однорідності предикату для кожної компоненти.

2.3 Загальні вимоги до вмісту суперпікселів

У літературі, присвяченій роботі із зображеннями в суперпіксельному поданні, опис суперпікселів, як правило, обмежений вузьким набором ознак. Зазвичай він включає спектральні характеристики суперпікселів: математичне очікування яскравості, дисперсію яскравості, ентропію яскравості і гістограму її розподілу. Деякі автори доповнюють набір спектральних характеристик аналогічними ознаками, які розраховані в колірному просторі LAB або HSV. У своїх наукових роботах автори, крім спектральних характеристик, розглядають також геометричні ознаки: площа, периметр, компактність (показує близькість форми суперпікселя до кола), відношення площі суперпікселя до площі прямокутника, що його описує. Ознаки, що визначають текстуру суперпікселів, представлені в літературі як відгуки на різні фільтри: MR8, HOG, Гауса, Габора та локальних бінарних шаблонів. Існує така думка, що опис суперпікселів невеликим набором ознак занадто звужує можливості обробки зображень. У цій роботі пропонується розширити вміст суперпікселів

та включати в них безліч характеристик (ознак) зображення, що відповідає двом вимогам:

1. Низькій складності обчислення в процесі сегментації зображення;
2. Достатності для розв'язання широкого класу прикладних завдань (універсальності суперпіксельного представлення зображення, що формується).

Першу вимогу можна формалізувати наступним чином.

Нехай $F(D)$ – ознака, обчислена за деякою кількістю пікселів (за деякою областю) D. Для будь-яких областей D_{m}, що не перетинаються, D_{n} має існувати правило Φ, таке, що

$$F(D_m \cup D_n) = \Phi[F(D_m), F(D_n)]. \quad (2.4)$$

Умова (2.4) залишає найширший простір для вибору конкретних варіантів цього правила. Не претендуючи на їх вичерпний список, обмежимося декількома класами ознак, які найлегше обчислюються і є потенційно корисними для вирішення прикладних завдань.

«Екстремальні» ознаки:

$$F(D) = \max_{(n_1,n_2\epsilon D)} \varphi[n_1, n_2, x(n_1, n_2)], \quad (2.5)$$

або

$$F(D) = \min_{(n_1,n_2\epsilon D)} \varphi[n_1, n_2, x(n_1, n_2)]. \quad (2.6)$$

«Адитивні» ознаки:

$$F(D) = \sum_{(n_1,n_2\epsilon D)} \Psi[n_1, n_2, x(n_1, n_2)]. \quad (2.7)$$

У формулах (2.5) – (2.7) φ та Ψ – довільні функції координат і значень пікселів зображення в області, що розглядається.

«Векторні» ознаки:

$$F(D) = \left\{F^{(k)}(D)\right\}_{k=0}^{K-1}, \tag{2.8}$$

де K – число компонент $F^{(k)}$ вектора, котрими можуть братися будь-які скалярні ознаки, зокрема перелічені вище.

Характерною особливістю ознак цього є сталість K, однаковий зміст і порядок компонент ознак вектора всіх суперпікселів. Окремим випадком є ненормовані гістограми, тобто розподілу числа будь-яких параметрів зображення, що обчислюються по області суперпікселя в заданому діапазоні значень. Для них правило об'єднання (2.4) зводиться до покомпонентного об'єднання векторів.

«Множинні» ознаки:

$$F(D) = \left\{F^{(k)}(D)\right\}_{k=0}^{K_D-1}, \tag{2.9}$$

де K_D – число елементів $F^{(k)}$ деякої кінцевої множини, не обов'язково числової, визначеної на області D.

Склад елементів може змінюватися від області до області, а правило (2.4) має конкретний вид об'єднання множин:

$$F(D_m \cup D_n) = F(D_m) \cup F(D_n) = \left\{F^{(k)}(D_m)\right\}_{k=0}^{K_{Dm}-1} \cup \left\{F^{(k)}(D_n)\right\}_{k=0}^{K_{Dn}-1}. \tag{2.10}$$

Ознаки, задані у загальному вигляді виразами (2.5) – (2.9), обрані виключно з міркувань швидкого перерахунку при об'єднанні суперпікселів (зокрема, при формуванні алгоритмом, описаним у попередньому розділі). Дотримуючись термінології, називатимемо такі ознаки первинними

(базовими). Вони не обов'язково використовуються для обробки зображення. Проте за ними можна обчислити деякі вторинні (похідні) ознаки, які, хоч власними силами і задовольняють умові об'єднаності (2.4), але необхідні для вирішення різних прикладних завдань. Є доцільним конкретизувати базові ознаки, що становлять розширений вміст суперпікселів, пов'язавши їх із похідними ознаками, які, у свою чергу, випливають із завдань обробки зображень.

2.4 Процедура об'єднання областей

Зображення, представлене суперпікселями, традиційно описується неорієнтованим планарним графом суміжності, в якому вузли відповідають областям суперпікселів (містять їх ознаковий опис), а ребра вказують на суміжні області. У межах цього підходу немає необхідності спеціально створювати таку структуру даних, тому що граф суміжності задається опосередковано, через численні ознаки F_3 – списки сусідів для суперпікселів. Нехай в результаті первинної сегментації зображення розбите на M областей суперпікселів, проіндексованих таким самим числом унікальних (не повторюваних) міток. І нехай опис області D_{m} включає множинну ознаку з мітками, що вказують на K_{D_m} суміжних областей ($0 \le m \le M - 1$). Тоді найпростіша процедура укрупнення суперпікселів полягає в тому, що послідовно розглядаються всі суперпікселі і для кожного з них виконується перевірка всіх сусідів на можливість приєднання їх до поточного суперпікселю. Об'єднання поточної області D_{m} із суміжною областю D_n буде здійснюватися при істинності заданого предикату однорідності для укрупненої області:

$$Q(D_m \cup D_n) = True. \qquad (2.11)$$

Зауважимо, що максимальна кількість зазначених перевірок дорівнює подвоєному числу ребер графа суміжності (кожному ребру відповідають два «зустрічні» посилання в ознаках F_3 суміжних суперпікселів):

$$\sum_{m=0}^{M-1} K_{D_m} = 2R, \tag{2.12}$$

де R – число ребер графа.

На практиці перевірок може бути і менше, якщо динамічно коригувати (зменшувати) списки посилань у міру приєднання областей. Зауважимо також, що за формулою Ейлера для планарних графів

$$R \leq 3M - 6. \tag{2.13}$$

З (2.12), (2.13) можна зробити висновок, що обсяг обчислень при об’єднанні областей буде не швидше ніж лінійно та залежати від числа суперпікселів первинно сегментованого зображення. Враховуючи, що це число мало, можна очікувати, що процедура об’єднання не буде обчислювально складною.

2.5 Предикати однорідності при об’єднанні областей та перерахунок ознак

Конкретні форми предикату (2.1), що використовується при об’єднанні областей, можуть бути дуже різноманітні, включати в себе як прості операції порівняння з порогом будь-яких ознак областей, що об’єднуються, так і складні нейромережеві процедури класифікації з навчанням за «еталонною сегментацією» зображень, виконаної експертами.

Один із найпростіших предикатів однорідності (2.3) вже використовувався при первинній сегментації зображення. Для областей D_{m} і D_n, що об'єднуються, його можна переписати у вигляді:

$$Q(D_m \cup D_n): \max_{(n_1,n_2)} x(n_1,n_2) - \min_{(n_1,n_2)} x(n_1,n_2) \le 2\varepsilon_{max}, (n_1,n_2) \in D_m \cup D_n. \quad (2.14)$$

Якщо поріг ε_{max} збільшити в порівнянні з тим, що був прийнятий при первинній сегментації, цей предикат цілком може забезпечити укрупнення суперпікселів.

Однак він є надмірно суворим для отримання фінального суперпіксельного опису зображення, що не може забезпечити скорочення кількості елементів зображення, що продемонстровано на рисунку 2.3. Розглянемо два інших предикати, які будуть використані нижче в ілюстративних обчислювальних експериментах.

Результатом первинної сегментації є формування областей, у кожній з яких яскравість задається константою, що апроксимує пікселі вихідного зображення з похибкою в діапазоні $[-\varepsilon_{max}, \varepsilon_{max}]$. Однак у багатьох прикладних завданнях досить контролювати не максимальну, а середньоквадратичну помилку апроксимації, тобто при об'єднанні областей, замість (2.14), використовувати наступний предикат:

$$Q(D_m \cup D_n): \frac{\varepsilon^2(D_m \cup D_n)}{F_{12}(D_m \cup D_n)} \le \varepsilon_{\text{кв}}^2, \quad (2.15)$$

де $\varepsilon^2(D_m \cup D_n)$ – сумарна квадратична помилка апроксимації зображення в об'єднаній області, яка, зокрема, для апроксимації константою задається формулою (2.16), а для лінійної апроксимації – формулою (2.17),

$F_{12}(D_m \cup D_n)$ – площа об'єднаної області,

$\varepsilon_{\text{кв}}^2$ – допустима середньоквадратична помилка апроксимації.

$$\varepsilon^2 = F_{25} - \frac{F_{18}^2}{F_{12}}, \tag{2.16}$$

$$\varepsilon^2 = F_{25} - a_0 F_{18} - a_1 F_{19} - a_2 F_{20}. \tag{2.17}$$

Через шумові спотворення або локальні коливання яскравості вихідного зображення в результаті його первинної сегментації можуть виникати «зайві» суперпікселі малої площі. Для їх видалення пропонується застосувати процедуру злиття областей із предикатом однорідності наступного виду:

$$Q(D_m \cup D_n): F_{12}(D_m) \le S_{min} \vee F_{12}(D_n) \le S_{min}, \tag{2.18}$$

де S_{min} – допустима мінімальна площа суперпікселю.

Якщо при виконанні (2.18) для об'єднаної області дотримується істинність предикату (2.15), то перерахунок базових ознак виконується за загальною схемою: береться максимальне (або мінімальне) значення екстремальних ознак, сумарне значення для адитивних ознак і т.д. В іншому випадку вважаємо, що суперпіксель малої площі D_n, що підлягає об'єднанню з суперпікселем D_m, породжений шумом і, отже, його характеристики яскравості повинні бути проігноровані. При цьому виникають особливості перерахунку деяких базових ознак: дві екстремальні ознаки не змінюють значення:

$$\begin{aligned} F_1(D_m \cup D_n) &= F_1(D_m), \\ F_2(D_m \cup D_n) &= F_2(D_m), \end{aligned} \tag{2.19}$$

а пов'язані з яскравістю адитивні ознаки можуть бути перераховані з допущенням сталості яскравості вихідного зображення в межах укрупненого суперпікселя:

$$F_i(D_m \cup D_n) = \frac{F_{18}(D_m)}{F_{12}(D_m)} F_{i-6}(D_m \cup D_n), \quad (2.20)$$

$$F_{25}(D_m \cup D_n) = [\frac{F_{18}(D_m)}{F_{12}(D_m)}]^2 F_{12}(D_m \cup D_n). \quad (2.21)$$

2.6 Ілюстрація застосування процедури укрупнення суперпікселів.

Застосуємо процедуру укрупнення суперпікселів до тестових зображень, що пройшли первинну сегментацію (рис. 2.2). На рисунку 2.4 (а–в) представлені залежності числа суперпікселів, віднесеного до числа пікселів вихідного зображення, від середньоквадратичної помилки апроксимації $\varepsilon_{\text{кв}}$. Предикат однорідності (2.15) використовувався для двох видів апроксимації, які було розглянуто.

Також представлені аналогічні залежності, отримані після попереднього видалення з суперпіксельного зображення областей малої площі: використовувався предикат (2.18) при $S_{min} = 3$. Первинна сегментація зображень виконувалася при $\varepsilon_{max} = 5$. Для порівняння на зазначені графіки пунктиром додані залежності, взяті з рисунку 2.3 з перерахунком максимальної помилки до середньоквадратичної: $\varepsilon_{\text{кв}} = \varepsilon_{max} / \sqrt{3}$.

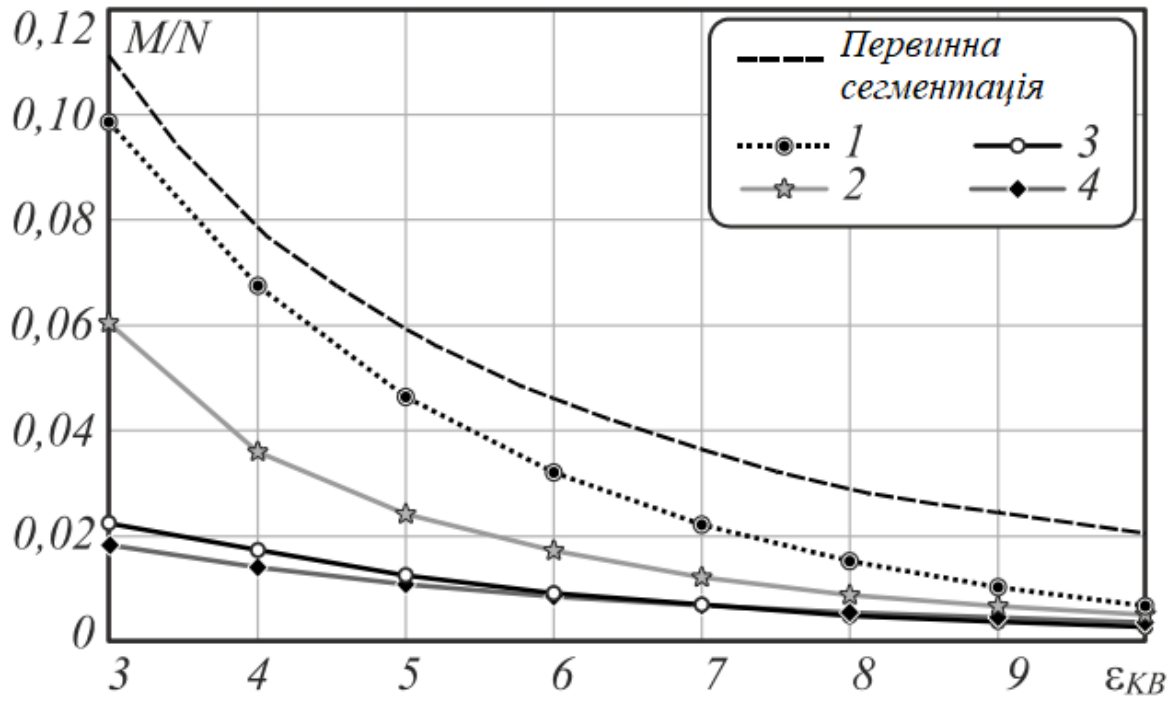
З наведених на рисунку 2.4 залежностей видно, що застосування запропонованої процедури укрупнення областей, отриманих після первинної сегментації, призводить до сильного скорочення їх числа за будь-яких значень середньоквадратичної помилки апроксимації.

При об’єднанні за предикатом однорідності (2.15) для апроксимації константою вже при $\varepsilon_{\text{кв}} = 3$ кількість отриманих областей для зображень «Зображення №1», «Зображення №3» (рис. 2.2 (а) та рис. 2.2 (в)), з переважно плавними змінами яскравості, становить менше 10% від загального числа пікселів, а для зображення «Зображення №2» (рис. 2.2 (б)), що містить велику кількість дрібних деталей, – менше 22%.

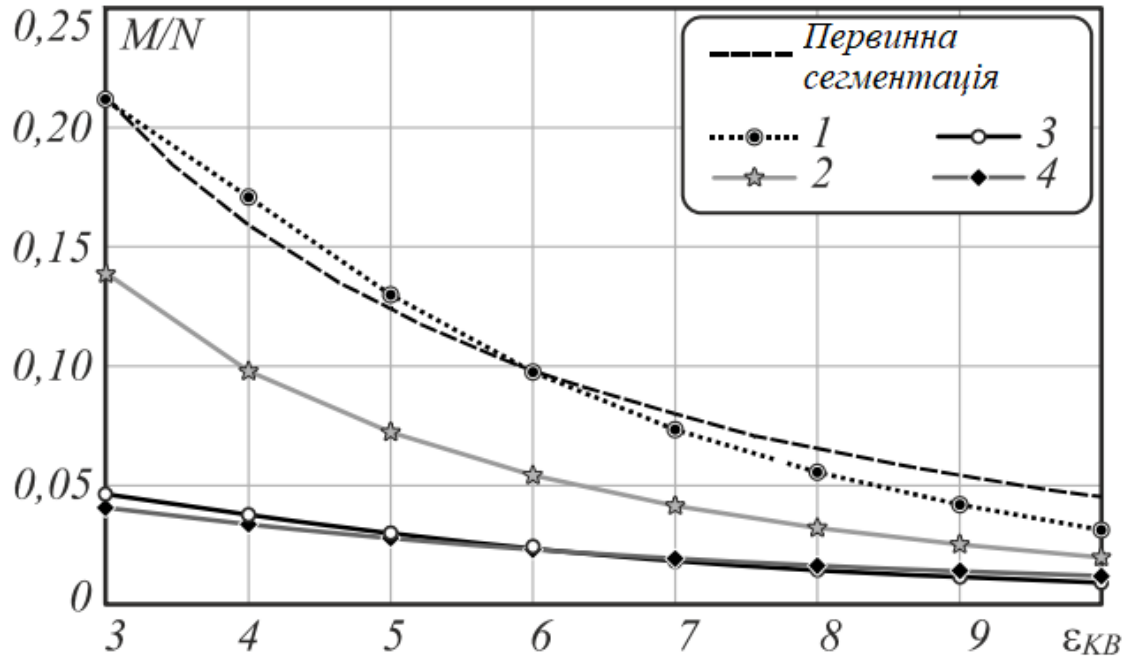
При лінійній апроксимації та тих же значеннях параметрів ε_{max} та $\varepsilon_{\text{кв}}$ кількість областей зображень «Зображення №1», «Зображення №3» становить менше 6% від загального числа пікселів, а зображення «Зображення №2» – менше 14%.

Попереднє видалення областей малої площі (породжених шумом або надмірною первинною сегментацією) за предикатом (2.18) дозволяє скоротити кількість елементів зображення, не викликаючи порушення меж об’єктів.

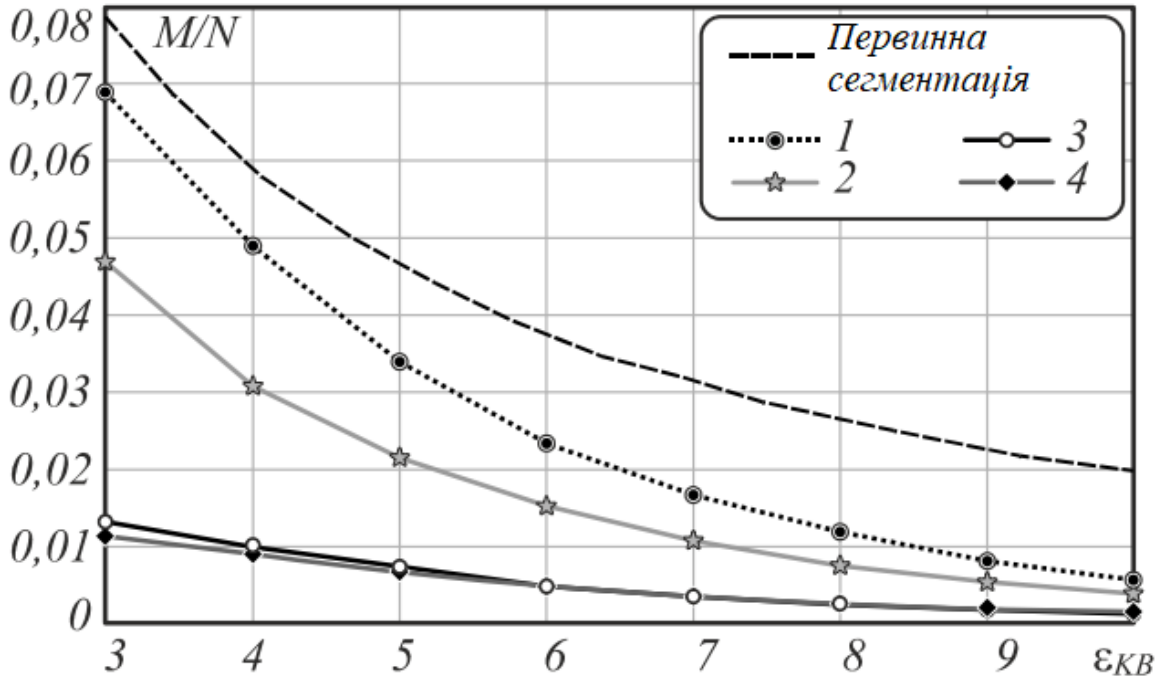
Видно, що результати виконання цієї операції при $S_{min} = 3$ для апроксимації константою та лінійної апроксимації відрізняються не більше, ніж на 1% від числа пікселів за будь-яких $\varepsilon_{\text{кв}}$.



(а) Для «Зображення №1»



(б) Для «Зображення №2»



(в) Для «Зображення №3»

Рисунок 2.4 (а-в) – Скорочення кількості елементів зображення після об’єднання суперпікселів (без видалення малих областей): 1 – апроксимація константою, 2 – лінійна апроксимація; після видалення малих областей ($S_{min} = 3$): 3 – апроксимація константою, 4 – лінійна апроксимація

3 РЕАЛІЗАЦІЯ МЕТОДУ СУПЕРПІКСЕЛЬНОЇ ОБРОБКИ З ВИКОРИСТАННЯМ FPGA

3.1 Обґрунтування вибору апаратного середовища

Для реалізації методу детектування дорожньої смуги була вибрана плата DE10-Nano фірми Terasic.

Field-programmable gate array (FPGA) – програмована мікросхема, в якій можна створити архітектуру з базових логічних елементів. Це дозволяє реалізувати попередню обробку зображення не програмно, навантажуючи ЦП комп’ютера, а апаратно – на FPGA у камері чи платі. Це підвищує стабільність системи машинного зору і збільшує швидкість роботи, а також знижує вимоги до продуктивності ЦП комп’ютера.

FPGA може встановлюватися безпосередньо в камеру, у плату захоплення, якщо ви використовуєте CoaXPress, CameraLink або оптоволокно, або встановлюватися окремою платою розширення для ПК.

FPGA у платі захоплення дозволяє програмувати плату захоплення на обробку зображення перед його збереженням буфер ПК для обробки вже програмними засобами. Дозволяє розвантажити ПК, знизити обсяг записуваних даних. Дозволяє працювати з кількома камерами, підключеними до плати захвату на одній FPGA. На цих платах часто реалізують стиснення зображення. Складність у цьому, що такі плати розраховані на швидкісні інтерфейси, такі як CameraLink, CoaXPress. Для GigE або USB камер таких плат немає.

Друга область, де широко використовуються FPGA, – це пристрої захоплення кадрів, знову ж таки, це часто робиться для перетворення даних або покращення зображення, наприклад, пристрої захоплення кадру Dalsa Xcelera зазвичай мають кілька різних прошивок, які можуть бути завантажені для виконання різних функцій, будь то перетворення шаблону Байера з сірою

шкалою для даних RGB або для покращення зображення, наприклад, для корекції плоского поля.

Більш просунуте використання FPGA у фрейм-граберах можна побачити на платах «Silicon Software Micro Enable 5 Ironman», архітектура в них використовує дві FPGA, одна для управління збором та передачею даних, віртуальними послідовними портами та зв’язком з камерою, а друга FPGA, запрограмована користувачем за допомогою графічного інструменту проектування для виконання критичних за часом, з високою пропускною здатністю та більш складних операцій, таких як фільтри Собеля, просте стиснення JPEG, високошвидкісне перетворення даних, визначення місця розташування великих двійкових об’єктів або лазерна тріангуляція.

У більшості камер машинного зору вже встановлено FPGA. За допомогою її в камерах реалізовано:

1. Робота з інтерфейсом I/O;
2. Різні режими ROI;
3. Зшивання рядків у лінійній камері.

Можна завантажувати аплет з готовим режимом, наприклад HDR, або самостійно конфігурувати роботу FPGA через спеціальне програмне забезпечення.

Однак доступ до програмування FPGA є не у всіх камерах. У камерах, де це можливо, можна реалізувати обробку зображень ще до того, як воно передається по інтерфейсу в ПК. Таким чином, можна полегшити роботу власного ПЗ, а також знизити навантаження на всю інфраструктуру, завдяки зменшенню кількості даних, що передаються і оброблюються.

Для прискорення роботи з кількома камерами можна використовувати FPGA плату встановлену в ПК. Вона встановлюється або у вигляді окремої плати розширення PCIe, або вже встановлена в деяких промислових ПК для машинного зору. Використання такого рішення також дозволить вам один раз запрограмувати FPGA на типову обробку знімків, а прискорювати обробку можна буде з кількох камер, надсилаючи на плату знімок з кожної камери.

ПЛІС (програмована вентильна матриця) – це інтегральна схема, що складається з набору «логічних блоків», «осередків введення-виведення» та «ресурсів міжз'єднання», це дозволяє реконфігурувати мікросхему для з'єднання входів та виходів (I/O) та логічних блоків разом безліччю різних способів [45].

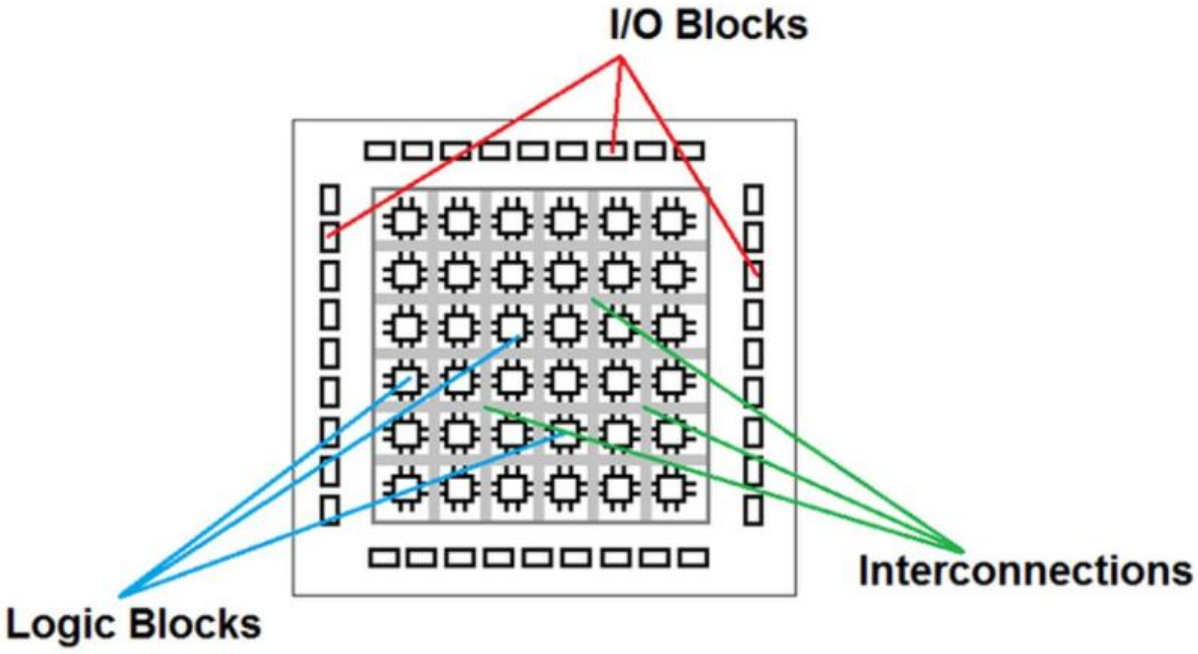


Рисунок 3.1 – Конфігурація FPGA

Кожен логічний блок традиційно може виконувати просту логічну операцію, таку як AND або XOR, і зазвичай містить деякий ступінь пам'яті, будь то простий тригер або складніший блок пам'яті. Логічні блоки перетворилися на блоки логічних функцій, що використовують таблиці пошуку всередині блоків для перемикання поточної функції, виконання завдань, таких як арифметичні операції.

ПЛІС не часто згадуються, але вони існують у безлічі обладнання, яке використовується у машинному зорі. Вони часто зустрічаються як частина серверної частини камери, що використовується для перетворення даних, що надходять від датчика, більш значущі дані, які будуть передаватися з камери. Як приклад можна подивитися на камери CVC GigE від STEMMER IMAGING,

вони використовують модуль Sony FCB як камеру (він сам по собі містить мікроконтролери і, можливо, FPGA), а інтерфейс модуля Sony з FPGA приймає сигнал LVDS з камери та перетворює його в потік даних передачі через Gigabit Ethernet. Крім перетворення даних, ядро FPGA запускає контролер пам'яті, використовуваний реалізації механізму повторної відправки даних.

Такий погляд на FPGA – лише частина того, що представляє з себе сучасна ПЛІС. Але справжня значущість ПЛІС, як і раніше, є основним атрибутом наявності ряду логічних блоків, які можна з'єднати разом для виконання масово паралельної обробки в реальному часі.

3.2 Обґрунтування вибору алгоритму і середовища програмної реалізації

Важко порівняти продуктивність процесора з FPGA. Порівняльний аналіз ЦП зазвичай визначається як тривалість виконання функцій і швидкість їх виконання, тоді як продуктивність FPGA базується на пропускній здатності даних. Сучасні процесори працюють у порядку ГГц, наприклад, сучасний процесор може мати чотири ядра, які працюють на частоті 2,4 ГГц кожне. У порівнянні з іншими, за швидкістю FPGA дуже повільно працює на частоті всього 62 МГц. Це виглядає як приголомшлива різниця і може викликати питання, як FPGA може бути достатньо швидкою, щоб мати цінність в обробці. Ось тут і з'являється перевага в паралельній обробці FPGA.

Замість того, щоб турбуватися про те, як швидко може працювати FPGA, нас цікавить швидкість передачі даних, яку вона може обробляти. Якщо подивитися на процес проходження потоку даних через FPGA, можна побачити, що швидкість є лише частиною рівняння. Дані надходять до FPGA, потім вони розбиваються для паралельної обробки, потім дані передаються в різні функції (налаштовані за допомогою логічних блоків). Потім FPGA

виводить оброблені дані. Все це відбувається в режимі реального часу (тобто у заздалегідь визначений проміжок часу з невеликою кількістю тремтіння).

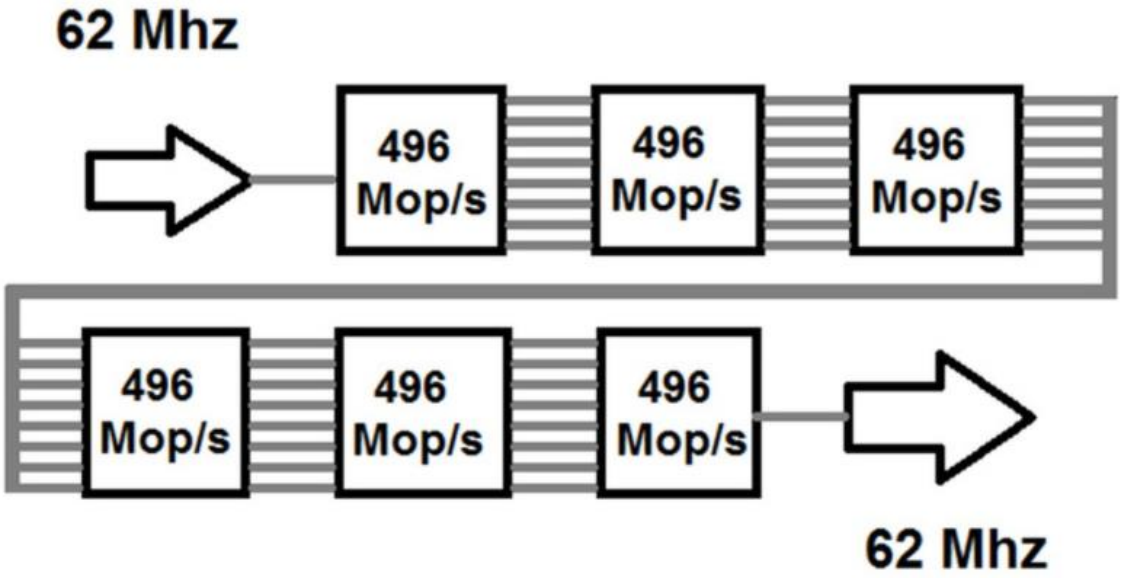


Рисунок 3.2 – Обробка потоку даних

Якщо подивитися на цей рисунок більш детально, то видно, що дані надходять зі швидкістю, яку видає камера (62 МГц). Потім розбиваються на 8 паралельних процесів, тож після цього швидкість передачі даних становить 496 MOPS (мільйон операцій в секунду). Потім дані проходять різні етапи обробки. Після цього паралелізм видаляється і виводиться в пам’ять зі стабільною швидкістю 62 МГц. Для додатків, де ця обробка відбувається на захваті кадрів, все це відбувається в режимі реального часу, без накладних витрат на ЦП, залишаючи ЦП виконувати інші функції.

FPGA можуть виконувати деякі функції дуже ефективно та дуже швидко, але є обмеження на те, що FPGA може робити. Найкраще місце для обробки FPGA – це потокова обробка, функції, які можуть виконуватися з потоком даних, не знаючи, як виглядають інші дані. Це включає перетворення Фур’є і матричні фільтри, нехай то фільтр високих і низьких частот, Собеля, розмиття, розширення. Все, що можна зробити з невеликою ділянкою даних, може бути дуже корисним для керування високими швидкостями передачі

даних, наприклад, якщо до зображення необхідно застосувати поріг, це дуже легко зробити за допомогою FPGA, бо вони мають саме ту перевагу, що надає процесору набагато менше даних для процесів.

Слабкість ПЛІС проявляється у випадках, коли їм доводиться мати справу з усім зображенням, ПЛІС зазвичай мають мало пам'яті в порівнянні з ЦП, вони не мають банків реєстрів, які можна використовувати для адресації, і вони не можуть маніпулювати даними. Мається на увазі, що виконання алгоритмів, що потребують довільного доступу до даних, є недоцільним. Також функції, у яких необхідно повторювати ітераційні операції, не є ідеальними для FPGA. Це не означає, що вони не можуть бути технічно реалізованими, якщо прийняти той факт, що на FPGA може бути створений віртуальний ЦП, то можна на цьому етапі виконувати ті ж функції, що і в системі на основі ЦП, але це потрібне додаткове обладнання для доступу до ПЛІС.

Завдяки зростаючій швидкості передачі даних, що досягається сучасними камерами, звернення до ПЛІС для отримання додаткової обчислювальної потужності може вирішити безліч проблем. Щоб подолати складність програмування ПЛІС, можна використовувати такі продукти як Visual Applets від Silicon Software, графічний інтерфейс для програмування ПЛІС на фрейм-граберах ME4, спрощуючи світ VHDL до середовища програмування на основі графічного інтерфейсу користувача.

Для більш підприємливих OEM-користувачів пропонується сумісне з GigE Vision IP-ядро для ПЛІС Xilinx та Altera, що забезпечує швидкий доступ до технологій на основі ПЛІС та інтерфейсу GigE.

FPGA і CPU не єдині методи обробки. Зараз дуже часто використовують графічний процесор (GPU) для розвантаження певної обробки, і це може бути дуже економічно ефективним рішенням. Цифрові сигнальні процесори (DSP) – ще один метод обробки даних в режимі реального часу. На рисунку 3.3 [45] наведено широкий огляд відмінностей між цими методами обробки.

FPGA processing	CPU Processing	GPU processing	DSP processing
Massive parallel processing	Sequential processing	Limited parallel processing	Limited parallel processing
Implementation is hardware	Implementation is software	Implementation is software (Imaging Lib's)	Implementation is software - proprietary
No or limited legacy due to "pipelined "process	Many I/O operations, load time etc.	Many I/O operations, load time etc.	Many I/O operations, load time etc.
Extremely fast execution Real – Time processing	Execution non-deterministic, varies with many dependencies	Execution fast Real – Time processing	Execution fast in certain areas, limited in others
Limited space	Nearly unlimited space for operations (programs)	Dependent on GPU chip (Limited)	
Not good at certain operations (floating point etc.)	Extremely versatile	Extremely strong in certain operations	Extremely strong in certain operations
Complicated to program (VHDL etc.)	With languages such as C++ very easy to program	With imaging libraries very easy to program	Not easy to program, proprietary development environment
Test / Debug difficult	Wide range of tools for debug	Some tools for debug available	Some tools for debug available
Need to interface to software	No need for special interfacing	No need for special interfacing	Needs to interface to software

Рисунок 3.3 – Порівняння методів обробки

FPGA, що використовуються для перетворення даних, широко поширені і, як правило, невидимі користувачеві, але коли вони виведені на передній план обробки, вони мають можливість розвантажувати потужність обробки від ЦП і можуть дозволити керувати надзвичайно високою пропускною здатністю, вони не підходять для всіх застосувань і не потрібні для деяких, але те, що вони пропонують, може зробити їх безцінним інструментом для машинного зору.

3.3 Конструкції FPGA з використанням HDL

Перші покоління FPGA були призначені лише для вузької області застосування, наприклад, дуже швидких цифрових систем на замовлення. Вартість мікросхеми та обмежені логічні ресурси були основними проблемами для цих FPGA. Незважаючи на обмежену кількість програмованих логічних ресурсів, апаратна реалізація цифрових систем забезпечила кращу

продуктивність для дуже швидких систем, де рішення на основі ЦП не могли задовольнити вимоги з точки зору обмежень часу.

Нині технологія FPGA досягла такої точки, що дозволяє вільно інтегрувати сотні жорстких ядер, блоків DSP і мільйони логічних елементів на одному чіпі. FPGA знаходиться в перерізі апаратної та програмної технології. Програмування FPGA фактично являє собою фізичну модель апаратного забезпечення. Тому для проектування на основі FPGA необхідні знання як апаратного, так і програмного забезпечення. Хоча інструменти CAD і автоматизація проектування спрощують програмування FPGA, залишається багато проблем і відкритих питань. Перш за все, побудова архітектури системи для ефективного використання ресурсів FPGA. Середовище проектування також має великий вплив на ефективне використання логічних ресурсів і час розробки. Іншим питанням додатків на основі FPGA є оцінка обчислювальної потужності.

Збільшуючи щільність логічних елементів FPGA, ефективне використання доступних ресурсів стає все більш складним. Мова програмування та середовище розробки мають значний вплив на ефективне використання логічної області, з одного боку, і скорочення часу розробки, з іншого. Щоб відповісти потребам швидкого створення прототипів нових систем на основі FPGA, постійно вводяться нові інструменти для програмування та перевірки конфігурації FPGA.

Мови опису обладнання (HDL) є стандартними інструментами для опису моделей, які будуть реалізовані на FPGA. Існує два основних текстових HDL: VHDL і Verilog. В основному обидва вони мають однаковий рівень складності. Однак їхні стилі програмування різні. Тому зазвичай тільки один з них використовується в будь-яких конструкціях залежно від умов інженерних груп. Вищезгадані інструменти програмування HDL забезпечують високу гнучкість для опису апаратних конфігурацій низького рівня. Тим не менш, недоліками низькорівневого програмування HDL є складність налагодження, перевірка та низька здатність працювати зі складністю проектування.

Хоча для програмування малих FPGA HDL є ефективними інструментами, що представляють високу гнучкість для опису фізичної реалізації, при роботі зі складними алгоритмами для цільових мікросхем FPGA з мільйонами програмованих логічних блоків вони займають дуже багато часу, і існує потреба в мові програмування, здатній представляти конфігурацію обладнання на більш високому рівні. Щоб не відставати від цього попиту, крім HDL, існують також мови конфігурації системного рівня та інструменти генерації коду, що дозволяють скоротити час розробки.

Інструменти SLD призначені для прискорення вбудованої розробки. Їх можна порівняти з мовами програмування високого рівня. Хоча VHDL корисний для розробки апаратної конфігурації та створення нових цифрових компонентів, інструменти SLD мають справу з автоматизацією проектування високого рівня та створенням нового дизайну шляхом з'єднання різних ядер IP (інтелектуальної власності), таких як процесори, пам'ять та мережеві контролери, інтерфейси зв'язку тощо. Компоненти індивідуально розроблені та оптимізовані в VHDL або Verilog, а потім можуть бути маршрутизовані в більшій конструкції за допомогою інструментів SLD.

Кількість і різноманітність ядер IP постійно збільшується. На додаток до зниження складності та уникнення збоїв у великих конструкціях FPGA, інструменти SLD дозволяють повторно використовувати дизайн. Прикладом інструментів SLD є Qsys, що надає Altera. Для вирішення розробки на основі FPGA для інженерії керування є деякі інші інструменти SLD, інтегровані в Simulink. Вони забезпечують зручне середовище проектування для розробки та перевірки алгоритмів керування. Прикладами таких SLD є Xilinx System Generator (XSG) і DSP Builder від Altera.

Інструменти генерації коду генерують коди HDL із звичайних програмних моделей. Дуже популярним середовищем розробки для інженерів управління є Simulink. Mathworks надає так званий HDL-кодер, який дозволяє конвертувати модель Simulink в коди VHDL або Verilog. Він має певну перевагу в порівнянні з іншими інструментами SLD: він демонструє більш

високу продуктивність для формування моделей, оскільки до створення HDL весь процес моделювання відбувався в середовищі Simulink. Крім того, згенерований код HDL відкритий для дизайнера і може бути поєднаний з іншими моделями HDL, закодованими вручну. І останньою, але не менш важливою, перевагою HDL-кодера є те, що він більш-менш універсальний і може застосовуватися для всіх сімейств FPGA, тоді як XSG і DSP Builder призначені лише для FPGA Xilinx і Altera відповідно.

3.4 Реалізація обробки відеопотоку на основі FPGA

Цифрові сигнальні процесори (DSP), спеціалізовані інтегральні схеми (ASIC) та графічні процесори (GPU) – це платформи, які зазвичай використовуються для реалізації алгоритмів обробки зображень і відео, що вимагають одночасних обчислень для декількох пікселів/кадрів. Якщо подивитися на архітектуру, типовий процесор TI DSP може мати два арифметичні логічні блоки (ALU) для виконання операцій множення та накопичення (MAC).

Для порівняння, FPGA може мати більше 200 блоків MAC, що обробляють пікселі паралельно. Деякі FPGA тепер мають виділені жорсткі блоки DSP/MAC для більш швидкої обробки. Використовуючи фреймворк обробки відео на основі FPGA, ми можемо забезпечити дослідження в режимі реального часу в гнучкому середовищі, що розвивається, заповненому зростаючим набором технологічних блоків.

Запропонована структура обробки відео використовує переваги вже доступного ТВ-декодера (ADV7181B) та цифро-аналоговий перетворювач для відео (ADV7123) для створення недорогого середовища для обробки відео. На рисунку 3.4 [45] представлена спрощена схема архітектури відеофреймворку. Вхідний сигнал є аналоговим композитним відеосигналом у форматі NTSC. ТВ-декодер перетворює вхідний сигнал в цифровий формат ITU-R BT.656.

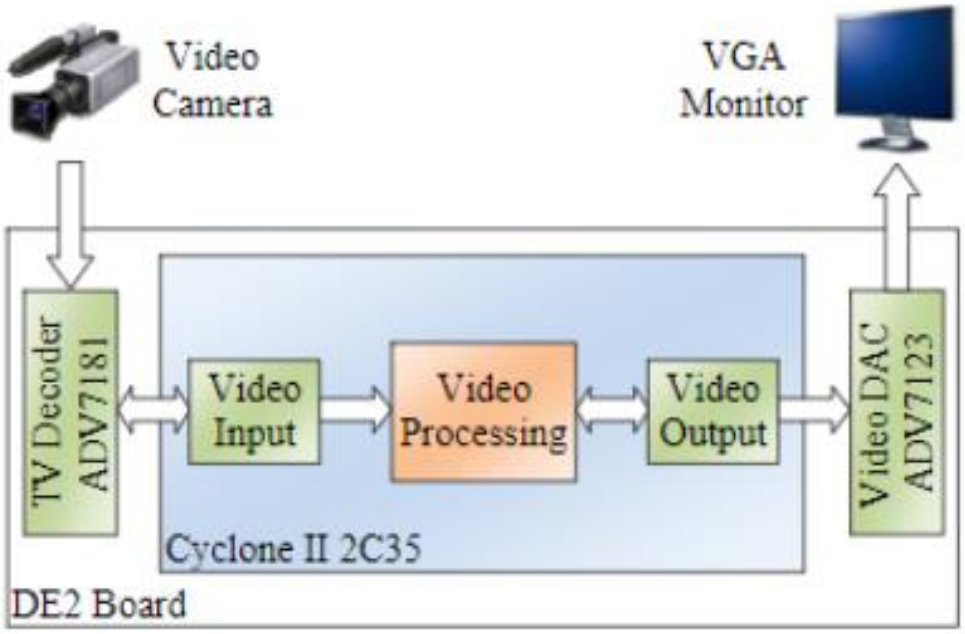


Рисунок 3.4 – Спрощена схема архітектури відеофреймворка

Модулі відеовходу та виведення засновані на демонстрації ТВ-боксу DE2, що поставляється з платою DE2 від Altera. Між цими двома модулями можуть бути розміщені оператори обробки відео, налаштовані для виконання конкретної програми в режимі реального часу.

3.5 Метод навчання математичної моделі

На даному етапі пропонується розглянути, яким чином буде побудована модель та навчатиметься на отриманих даних. Ця проблема вирішуватиметься як сегментування зображення на основі отриманої діаграми фіксацій. На вхід класифікаційна модель приймає «eye-tracking» дані у вигляді фіксацій людського ока та виводить просторову підтримку об’єкта, розташовуючи кожен піксель як об’єкт чи фон. Метод складається із двох частин: сегментація об’єкта та уточнення сегментації. На початковому етапі прогнозується оцінка розташування об’єкта, позначаючи кожен суперпіксель окремо.

Цей параметр визначає зв’язок між фіксаціями людського погляду та просторовим положенням об’єкта. Прогностичний параметр навчається на невеликому наборі зображень, та проанотований як фіксаціями, так і вручну рамками, що обмежують об’єкт. Виділення властивостей, за допомогою яких буде проходити навчання параметра, відбувається на цьому етапі. На виході утворюються значення кожного пікселя, які відповідають певній ймовірності їх розташування на об’єкті. Перший етап методу представлено на рисунку 3.5.

Після того, як проведено навчання класифікатора на невеликому блоці даних, виявлено, що є зв’язок між функціями фіксацій і тим фактом, чи розташований суперпіксель на цільовому об’єкті чи ні. Далі застосовуємо навчений класифікатор до великого набору даних 90%, у результаті чого утворюється маска м’якої сегментації. Цей процес означає вилучення об’єкта переднього плану з довільного безпосереднього зображення. Кожне значення пікселя в масці відповідає оцінній ймовірності того, що піксель знаходиться на об’єкті.

Навчання класифікатора суперпікселів відбувається окремо для кожного класу об’єкта, тому що зв’язок між фіксаціями та об’єктами може виявитися неоднорідним. Навчальна вибірка складатиметься із векторних ознак усіх суперпікселів. Кожен суперпіксель маркується відповідно до того, чи знаходиться всередині обмежувальна рамка чи ні.

На другому етапі отримання сегментів відбувається уточнення виходу м’якої сегментації M, зважаючи на парні залежності між сусідніми суперпікселями та покращуючи моделі зовнішнього вигляду. Нехай $I_S \in \{0,1\}$ – мітка для суперпікселів S, а L – мітка усіх I_S на зображенні. Використаємо бінарну функцію E попарної енергії, визначену над суперпікселями:

$$E(L) = \Sigma_S M_S(I_S) + \Sigma_{S,R} V(I_S, I_R). \quad (3.1)$$

Попарний потенціал V забезпечує гладкість, штрафуючи сусідні пікселі з різними мітками. Штраф залежить від колірного розмаїття між пікселями, він менший в областях з високою контрастністю (краї зображення).

Відзначимо, що за допомогою пристрою «eye-tracker» збирається велика кількість аспектів зображення. Також доведено, що точка фіксації погляду визначає найбільш точне положення об’єкта зображення. Тобто цей метод є дієвим методом навчання математичної моделі та виявляє всі важливі особливості, що дозволяють проводити ефективне сегментування цільового об’єкта.

ВИСНОВКИ

У рамках кваліфікаційної роботи був розроблений і проаналізований суперпіксельний метод обробки зображень за допомогою програмованих вентильних логічних матриць (FPGA).

В рамках даної роботи основну увагу приділено сегментації, а саме процесу поділу зображення на групи пов’язаних областей, які є однорідними і не перекриваються (група пікселів, які також називається суперпікселями). Отримані в результаті об’єкти полегшують демонстрацію зображення та визначення розташування меж і об’єктів, таких як криві, лінії. Перевагою обраного методу є те, що суперпікселі мають різні форми та розміри та можуть створювати сегменти, які містять в собі безліч дрібних частин, що зберігають велику кількість меж вихідного зображення.

Було сформульовано загальні вимоги до базових ознак, що швидко формуються при первинній сегментації. На підставі аналізу ряду типових завдань обробки та розпізнавання зображень введена в розгляд система, що складається з базових ознак, які в першу чергу слід включати в суперпікселі. Показано, як ці базові ознаки перераховуються у похідні ознаки, що безпосередньо використовуються при вирішенні конкретних прикладних завдань.

Для програмної реалізації обрано програмовані логічні інтегральні схеми FPGA, оскільки вони підходять для обробки відеосигналу з високою роздільною здатністю. Перевагою FPGA перед спеціалізованими сучасними інтегральними мікросхемами є те, що FPGA легко модернізувати і тому немає ризику, що система швидко застаріє.

Результати роботи було апробовано на X Міжнародній науково-практичній конференції «SCIENCE FOUNDATIONS OF MODERN SCIENCE AND PRACTICE», 23-26 листопада 2021 р., Афіни, Греція.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Rabotiahov, A., Kobylin, O., Dudar, Z., & Lyashenko, V. (2018, February). Bionic image segmentation of cytology samples method. In *2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)* (pp. 665-670). IEEE.

2. Работягов, А. В., Ляшенко, В. В., & Кобылин, О. А. (2016). Сегментация сложных изображений цитологических препаратов.

3. Lyashenko, V., Mohammad, A., & Kobylin, O. (2015). Experiments with Fusion of Images with Use of Wavelet Transformation in Problems of the Text Information Analysis.

4. Steinley, D. (2006). K-means clustering: a half-century synthesis. *British Journal of Mathematical and Statistical Psychology*, *59*(1), 1-34.

5. Huang, Z., & Ng, M. K. (1999). A fuzzy k-modes algorithm for clustering categorical data. *IEEE Transactions on Fuzzy Systems*, *7*(4), 446-452.

6. Xu, J., Han, J., Xiong, K., & Nie, F. (2016, July). Robust and Sparse Fuzzy K-Means Clustering. In *IJCAI* (pp. 2224-2230).

7. Деркач, О. І. (2016). Аналітична обробка текстової інформації за допомогою засобів кластеризації. *Young*, *34*(7).

8. Kobylin, O., Vyskrebentseva, S., & Petrova, R. (2019). Обробка даних, що містять пропуски в задачах кластеризації. *Системи управління, навігації та зв'язку. Збірник наукових праць*, *5*(57).

9. Little, R. J., & Rubin, D. B. (2019). *Statistical analysis with missing data* (Vol. 793). John Wiley & Sons.

10. Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR)*, *31*(3), 264-323.

11. Perret, B., Chierchia, G., Cousty, J., Guimarães, S. J. F., Kenmochi, Y., & Najman, L. (2019). Higra: Hierarchical graph analysis. *SoftwareX*, *10*, 100335.

12. Ackermann, M. R. (2009). *Algorithms for the Bregman k-Median problem* (Doctoral dissertation, University of Paderborn).

13. Khachumov, M. V. (2012). Distances, metrics and cluster analysis. *Scientific and Technical Information Processing*, *39*(6), 310-316.

14. Montgomery, D. C., Jennings, C. L., & Kulahci, M. (2015). *Introduction to time series analysis and forecasting*. John Wiley & Sons.

15. Zhang, J., Zhao, Z., Xue, Y., Chen, Z., Ma, X., & Zhou, Q. (2017). Time series analysis. *Handbook of Medical Statistics*, *269*.

16. Крашений, І. Е., Попов, А. О., Раміpез, Х., Горріз, Х. М., Крашеный, И. Э., Попов, А. А., ... & Горриз, Х. М. (2016). Використання методів кластеризації в системах нечіткого виводу для діагностики хвороби Альцгеймера на основі ПЕТ-зображень.

17. Штовба, С. Д. (2006). Побудова функцій належності нечітких множин за кластеризацією експериментальних даних. *Інформаційні технології та комп'ютерна інженерія*, (2), 92-95.

18. Bodyanskiy, Y. V., Tyshchenko, O. K., & Mashtalir, S. V. (2019, June). Fuzzy Clustering High-Dimensional Data Using Information Weighting. In *International Conference on Artificial Intelligence and Soft Computing* (pp. 385-395). Springer, Cham.

19. Oleg, K., Sergii, M., & Mykhailo, S. (2017, October). Video Clustering via Multidimensional Time-Series Analysis. In *Proceedings of the 9th International Conference on Information Management and Engineering* (pp. 60-63). ACM.

20. Mashtalir, S., Mashtalir, V., & Stolbovyi, M. (2017). Video shot boundary detection via sequential clustering. *International Journal "Information Theories and Applications*, *24*(1), 50-59.

21. Mashtalir, S., Mashtalir, V., & Stolbovyi, M. (2018, August). Representative Based Clustering of Long Multivariate Sequences with Different Lengths. In *2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)* (pp. 545-548). IEEE.

22. Bodyanskiy, Y., Kobylin, I., Rashkevych, Y., Vynokurova, O., & Peleshko, D. (2018, February). Hybrid fuzzy-clustering algorithm of unevenly and asynchronously spaced time series in computer engineering. In *2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)* (pp. 930-935). IEEE.

23. Bodyanskiy, Y., Vynokurova, O., Kobylin, I., & Kobylin, O. (2016). Adaptive fuzzy clustering of short time series with unevenly distributed observations in Data Stream Mining tasks. *Information Technology and Management Science*, *19*(1), 23-28.

24. Женбинг, Х., Бодянский, Е. В., Тыщенко, А. К., & Ткачев, В. Н. (2017). Fuzzy Clustering Data Arrays with Omitted Observations.

25. Kate, R. J. (2016). Using dynamic time warping distances as features for improved time series classification. *Data Mining and Knowledge Discovery*, *30*(2), 283-312.

26. Hu, Z., Mashtalir, S. V., Tyshchenko, O. K., & Stolbovyi, M. I. (2018). Clustering matrix sequences based on the iterative dynamic time deformation procedure. *International Journal of Intelligent Systems and Applications*, *10*(7), 66-73.

27. Wang, D., Lu, X., & Rinaldo, A. (2017). DBSCAN: Optimal Rates For Density Based Clustering. *arXiv preprint arXiv:1706.03113*.

28. Lyashenko V., Kobylin O., Selevko O. (2020) Wavelet Analysis and Contrast Modification in the Study of Cell Structures Images. International Journal of Advanced Trends in Computer Science and Engineering. 9(4). – 4701-4706.

29. Bodyanskiy, Y., Shafronenko, A., & Mashtalir, S. (2019, May). Online Robust Fuzzy Clustering of Data with Omissions Using Similarity Measure of Special Type. In *International Scientific Conference "Intellectual Systems of Decision Making and Problem of Computational Intelligence"* (pp. 637-646). Springer, Cham.

30. Mashtalir, S. V., Stolbovyi, M. I., & Yakovlev, S. V. (2019). Clustering Video Sequences by the Method of Harmonic k-Means. *Cybernetics and Systems Analysis*, *55*(2), 200-206.

31. Mashtalir, V., Ruban, I., & Levashenko, V. (Eds.). (2019). Advances in Spatio-Temporal Segmentation of Visual Data (Vol. 876). Springer Nature.

32. Kobylin, O., & Lyashenko, V. (2016). Contrast Modification as a Tool to Study the Structure of Blood Components.

33. Ren and Malik, "Learning a classification model for segmentation," in Proceedings Ninth IEEE International Conference on Computer Vision, Oct 2003, pp. 10–17 vol.1.

34. M. Miyama, "Fast stereo matching with super-pixels using one-way check and score filter," in 2017 7th IEEE International Conference on Control System, Computing and Engineering (ICCSCE), Nov 2017, pp. 278–283.

35. F. Gu, H. Zhang, and C. Wang, "A Classification Method for Polsar Images using SLIC Superpixel Segmentation and Deep Convolution Neural Network," in IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium, July 2018, pp. 6671–6674.

36. R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. S¨usstrunk, "Slic superpixels," 2010.

37. M. R. Luo, CIELAB. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 1–7. [Online]. Available: https://doi.org/10.1007/978-3-642-27851-8 11-1 (Accessed 2020-01-30).

38. R. Achanta and S. Susstrunk, "Superpixels and polygons using simple non-iterative clustering," 07 2017, pp. 4895–4904.

39. M. Van den Bergh, X. Boix, G. Roig, B. Capitani, and L. Van Gool, "SEEDS: Superpixels Extracted via Energy-Driven Sampling," in International Journal of Computer Vision, vol. 111, 10 2012.

40. F. Meyer, "Color image segmentation," in 1992 International Conference on Image Processing and its Applications, April 1992, pp. 303–306.

41. P. Neubert and P. Protzel, “Compact Watershed and Preemptive SLIC: On Improving Tradeoffs of Superpixel Segmentation Algorithms,” in Proceedings - International Conference on Pattern Recognition, 08 2014, pp. 996–1001.

42. P. Neubert and P. Protzel, “Superpixel benchmark and comparison,” in Forum Bildverarbeitung, 2012.

43. D. Martin, C. Fowlkes, D. Tal, and J. Malik, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics,” in Proc. 8th Int’l Conf. Computer Vision, vol. 2, July 2001, pp. 416–423.

44. P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, “Contour detection and hierarchical image segmentation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 5, pp. 898–916, May 2011.

45. E. Fykse, «Performance Comparison of GPU, DSP and FPGA implementations of image processing and computer vision algorithms in embedded systems», Norwegian University of Science and Technology, 2013.