МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

Шпак Володмир Олександрович

«Алгоритми сегментації цифрових кольорових зображень на основі теорії графів / Algorithm for segmentation of digital color images based on graph theory»

спеціальність: 123 - Комп'ютерна інженерія
освітньо-професійна програма - Комп'ютерна інженерія
Кваліфікаційна робота

Виконав студент групи КІм-21
В.О. Шпак
_____

Науковий керівник:
к.т.н., доц. Ю.М. Батько
_____

Кваліфікаційну роботу допущено
до захисту:

"___" _____ 20___ р.

Завідувач кафедри
_____Л. О. Дубчак

ТЕРНОПІЛЬ – 2024

АНОТАЦІЯ

Алгоритми сегментації цифрових кольорових зображень на основі теорії графів /

Кваліфікаційна робота зі спеціальності 123 «Комп'ютерна інженерія» освітнього ступеня «магістр» написана обсягом 86 сторінки та містить 25 ілюстрацій, 2 додатки та 50 джерел за переліком посилань.

Метою роботи є розробка алгоритму сегментації цифрових кольорових зображень на основі теорії графів.

Методи досліджень. Для розв'язання поставлених задач у кваліфікаційній роботі використано методи: попередньої обробки цифрових зображень (для для оцінки та обчислення параметрів роботи); алгоритми теорії графів (для розбиття зображення на оноріжні області); об'єктно-орієнтованого програмування (для програмної реалізації запропонованих алгоритмів та структури програмної системи).

Результати дослідження: алгоритм сегментації цифрових зображень з використанням елементів теорії графів, програмний додаток обрбки та аналізу растрових зображень на мові Java з використання бібліотеки OpenCV.

Результати роботи можуть бути використані в створенні складних систем обробки, опису та синтезу цифрових зображень, охоронних системах та системах діагностки тощо.

Орієнтовні напрямки розвитку досліджень: розроблення алгоритмів сегментації менш чутливих до зовнішніх шумів, створення нових програмних засобів та моделей для навчання.

КЛЮЧОВІ СЛОВА: РАСТРОВІ ЗОБРАЖЕННЯ, СЕГМЕНТАЦІЯ, ТЕОРІЯ ГРАФІВ, СИСТЕМИОБРОБКИ ЦИФРОВИХ ЗОБРАЖЕНЬ.

ANNOTATION

Shpak V. Algorithm for segmentation of digital color images based on graph theory – manuscript.

Qualification work on specialty 123 – Computer Engineering is 86 pages long and contains 25 illustrations, 1 tables, 2 appendices and 50 references.

The aim of the work is develop a segmentation algorithm for digital color images based on graph theory.

Research methods. To solve the tasks set in the qualification work, the following methods were used: pre-processing of digital images (for evaluating and calculating the parameters of the work); graph theory algorithms (for dividing the image into different regions); object-oriented programming (for software implementation of the proposed algorithms and the structure of the software system).

Research results: a segmentation algorithm for digital images using graph theory elements, a software application for processing and analyzing raster images in Java using the OpenCV library.

The results of the work can be used in creating complex systems for processing, describing and synthesizing digital images, security systems and diagnostic systems, etc.

Approximate directions of research development: development of segmentation algorithms less sensitive to external noise, creation of new software tools and models for training.

KEYWORDS: BITRAM IMAGES, SEGMENTATION, GRAPH THEORY, DIGITAL IMAGE PROCESSING SYSTEMS.

ЗМІСТ

ВСТУП

Актуальність роботи. В даний час обробка та аналіз зображень на основі засобів теорії графів широко використовується в різноманітних технічних рішеннях. Причина використання графів як одного з основних підходів полягає в тому, що вони підходить для розвитку ефективних методи та мають значну гнучкість у представленні різні види зображень. Крім того, вони можуть бути використані для розробки нових теоретичних основ та підвищення можливостей засобів обробки та аналізу цифрових зображень. Традиційні зображення на основі представлення у формі набору точок є ефективним та простим підходом для відображення різноманітних сцен. Проте кількість точок може бути значною, що негативно впливає на загальн розміри цифрових зобаржень, тому зрупування наборуоднорідних точок є ефективним засобом зменшення даного впливу. Дуже багатообіцяючим підходом є виділення графів із зображення, що є початковим етапом процесу аналізу зображень на основі графів. Перед процесом кодування зображення на основі графів виконується процес сегментації. Сегментація зображення створює набір однорідних областей, так що всі пікселі однієї області з'єднуються один з одним. Кожна область має набір пікселів, а всі пікселі в одній області пов'язані з набором функцій. Таке представлення використовує граф суміжності регіону, який є способом представлення зображення з формою регіону, представленого вершиною, і зв'язків регіону з його сусідами, представленими ребром. Визначення оптимальної кількості регіонів має вирішальне значення на етапі вилучення графів на основі результатів сегментації.

Останнім часом графи все частіше використовуються як спосіб представлення для обробки та аналізу зображень. Для представлення використовуються графи, що складається з сутностей, які описуються як вершини і зв'язки між сутностями, які описуються як ребра. В області розпізнавання образів ці уявлення використовувалися з кінця 1970-х років.

Даний підхід застосовувався як до 2D/3D зображень, форм, документів, символів так і для кодування людиноподібних або біологічних структур, семантичних мереж, соціальних та економічних мереж тощо.

Сегментація зображення є початковим етапом вилучення графа зображення. Це утворить однорідністьнас набір регіони. Кожен регіон має набір підключенихпікселі та всі пікселі в регіоні мають один набір функцій. Отримання дуже повної функції зображення, яка може повторити зображення все ще є складним завданням. Тому задача розробки та дослідження алгоритму сегментації цифрових зображень є актуальною.

Метою роботи є розробка алгоритму сегментації цифрових кольорових зображень на основі теорії графів.

Для досягнення даної мети ставились наступні завдання:

- проаналізувати технології створення та опису цифрових зобаржень;
- дослідити методи та алгоритми теорії графів;
- провести аналітичний огляд цифрових бібліотек та програмних засобів оброки цифрових зображень;
- проаналізувати методи та алгоритми сегментації цифрових зображень;
- розробити алгоритм сегментації цифрових зображень на основі теорії графів;
- реалізувати програмний засіб обробки та опису цифрових зображень, провести його тестування та порівняння з аналогами.

Об’єкт дослідження – процес обробки цифрових зображень.

Предмет дослідження – методи і алгоритми сегментації кольорових зобаржень.

Наукова новизна одержаних результатів визначається наступним чином:

- здійснено аналітичний огляд технологій сегментації цифрових кольорових зображень на основі порівняння точності отриманих результатів, що дозволило видітити переваги та недоліки використання підходів теорії графів для розділення цифрового зображення на однорідні області;

- розроблено алгоритм сегментації цифрових зображень на основі теорії графів, що дозволило спроектувати та провести аналіз внутрішньої архітектури програмного додатку обробки та опису цифрових зображень.

Практична цінність одержаних результатів полягає в тому, що:

- запропоновано принципи подубови внутрішньої структури програмного додатку обробки цифрових зображень на основі використання модульного підходу, що дозволило програмно реалізувати розроблені алгоритми та провести їх оцінку;

- програмно реалізовано систему обробки та опису цифрових зображень на основі запропонованої структури та алгоритмів сегментації, що дозволило провести тестування їх тестування та порівняння з аналогічними системами.

Публікації та апробація до випускної кваліфікаційної роботи. За результатами наукових досліджень, проведених у випускній кваліфікаційні роботі, підготовлено тези доповіді «Аналіз алгоритмів сегментації цифрових зображень» обсягом 2 сторінки на X Всеукраїнській науково-практичній конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі» (ІКСМ'2024), а також «Алгоритм обробки та аналізу біомедичного зображення» обсягом 2 сторінки на X Всеукраїнській науково-практичній конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі» (ІКСМ'2024).

В першому розділі досліджено та описана технології створення, кодування та опису цифрових зображень та описано можливості використання методів та алгоритмів теорії графів в процесі обробки цифрових зображень.

В другому розділі було розглянуто методи та лгоритми обробки цифрових зображень та запрооновано лагоритм сегментації цифрового зображення на основі використання засобів теорії графів.

В третьому розділі кваіфікаційної роботи наведено опис структури програмного додатку обробки цифрових зображень, проведено його тестування та порівняння з аналогами.

1 ТЕХНОЛОГІЇ СТВОРЕННЯ ТА ОБРОБКИ ЦИФОВИХ КОЛЬОРОВИХ ЗОБРАЖЕНЬ

1.1 Цифорові кольорові зображення

Цифрові зображення – це електронні «файли», створені, отримані, оброблені та збережені в цифровій формі. Ці зображення можуть бути отримані такими пристроями, як сканери, камери або смартфони шляхом фіксації деякої візуальної сцени (на папері, реальна сцена тощо). Їх також можна створити з нуля, наприклад, за допомогою комп'ютера або графічного планшета. У будь-якому випадку ці файли буде легко змінити за допомогою спеціального програмного забезпечення. Існує два види цифрових зображень: растрові зображення та векторні зображення. Растрові зображення з математичної точки зору – це зображення складається з масиву точок (пікселів). Ці точки можуть бути чорно-білими, сірими або кольоровими. З точки зору користувача фундаментальною характеристикою растрового зображення є максимально точне відображення реальності. Дані зображення можуть правдиво відобразити реальність, зробивши фотографії цифровою камерою або навіть смартфоном. Даний тип зображень використовується в основному для фотофіксації сцен які вимагають максимальної відповідності з реальністю. Дані зображення можуть предати усі дрібні деталі сцени, що в подальшому значно полегшує процеси обробки інформації наведеної в даному вигляді. Проте для досягнення високого рівня деталізаніх необідні великі обсяги пам'яті, що унеможливлює використання даного способу отримання, опрацювання та зберігання інформації в цифровому вигляді. Принцип роботи векторних зображень полягає в представленні даних зображення (вмісту) математичними (геометричними) формулами. На відміну від растрових зображень, векторні зображення найчастіше створюються без відношення до реальності. Найчастіше вони використовуються для зображення логотипів. Окремо слід відмітити, що для зберігання зображень в вектороному форматі використовуються набори

геометричних примітивів, що зацмають значноменше місце в пам’яті. Під час збільшення векторного зображення не відбувається втрати якості, на відміну від растрового зображення (рисунок 1.1).

а) б)

Рисунок 1.1 – Приклад масштабування цифрових зображень у растровому(а) та векторному (б) представленні

Об’єкт видимий лише в тому випадку, якщо наше око отримує світло від цього об’єкта. Сам об’єкт може бути джерелом світла або відбивати джерело світла. Фотографія – це запис цього світла на носій (плівку або цифровий сенсор). Фотографувати означає писати світлом. Світло – це електромагнітне випромінювання, яке сприймається оком і може надходити від природного (сонце, зірки) або штучного (лампочка) джерела або від об’єкта, що відбиває світло (наприклад, місяць, освітлений сонцем). Світло складається з фотонів (частинок), але воно має властивості хвилі. Подвійність хвиль частинок дозволяє краще зрозуміти аспекти фотографії. Той факт, що світло складається з фотонів, допомагає пояснити роботу цифрових датчиків, які вловлюють енергію випромінювання, що переноситься фотонами, перетворюючи її на електрони (електричну енергію). Той факт, що світло має властивості хвилі, дозволяє краще зрозуміти використання кольорових фільтрів або явище люмінесценції.

В однорідному прозорому середовищі світло поширюється по прямих лініях, які утворюють світлові промені. У 1666 році Ньютон за допомогою експерименту з призмою продемонстрував, що промінь білого світла можна

розбити на різні кольори: червоний, оранжевий, жовтий, зелений, синій і фіолетовий. Даний експеримент показав, що біле світло поділяється на складові і кожен промінь кольору, який утворює біле світло, відхиляється склом призми по-різному, як це спостерігається у веселки. Природне світло здається білим (рисунок 1.2), але насправді це світло складається з кількох кольорових випромінювань, видимі для людського ока довжини хвиль яких варіюються від 400 до 725 нм (1 нм = 10-9 м).

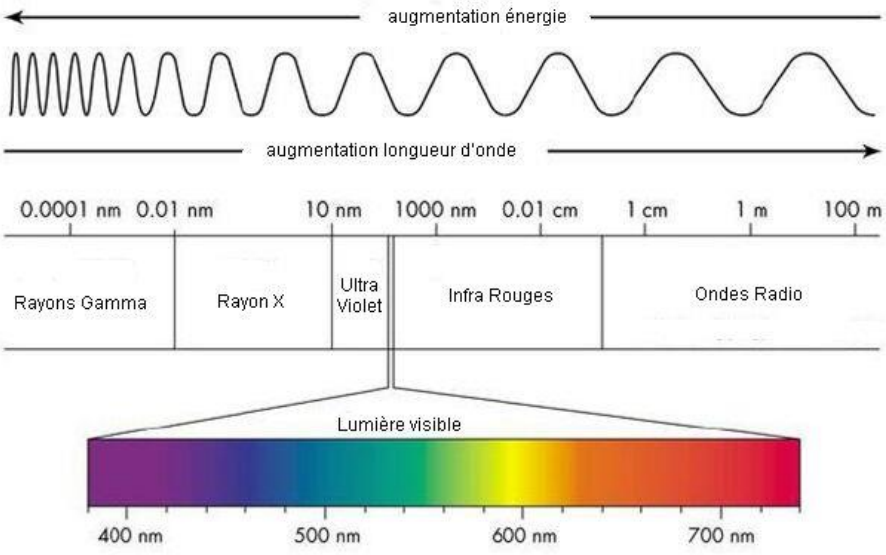

Рисунок 1.2 – Розподілення довжин світла у природі

По обидва боки видимого спектра знаходиться випромінювання, невидиме неозброєним оком, наприклад:

- Інфрачервоне (ІЧ), характерне для світла, що випромінюється гарячими тілами: Сонцем, людиною. Інфрачервоні промені мають довжину хвилі, більшу за довжину хвилі видимого спектра.

- Ультрафіолет (УФ), який зазвичай називають «холодним світлом», насправді випромінюють надзвичайно гарячі розжарені тіла, такі як сонце та зірки. Ці типи променів можуть використовуватися технічною та науковою поліцією для активації явища люмінесценції (волокна, люмінесцентні барвники).

Сенсори цифрових камер мають набагато кращу спектральну чутливість, ніж людське око. Насправді вони здатні записувати дані від ультрафіолетового та інфрачервоного (інфрачервоного) випромінювання.

Колірна температура. Світло, яке спостерігає людина, може набувати різних «відтінків». У той час як світло від електричної лампочки має жовтуватий відтінок, світло від літнього неба має блакитний відтінок. Ці різні відтінки, які може приймати джерело світла, відповідають «кольоровій температурі» світла. Це не має нічого спільного з температурою джерела світла. Це поняття колірної температури було визначено на основі теоретичної моделі «чорного тіла», яке при нагріванні випромінює певний відтінок світла. Нагріваючи це чорне тіло, воно червоніє, біліє, а потім синіє. Колірна шкала чорного тіла, нагрітого від 0 Кельвінів (0K = -273,15°C) до 22000K, змінюється від червоного до синього. Колір світла можна порівняти з кольором чорного тіла. Таким чином, колір світла, випромінюваного галогенною лампою, ідентичний кольору чорного тіла, нагрітого від 3000 К до 3200 К. Це поняття колірної температури є особливо важливим у фотографії для налаштування балансу білого.

Світло є формою енергії. З фізичної точки зору, це енергія, випромінювана джерелом, транспортована променем, отримана освітленим об’єктом. Воно характеризується своїм спектральним складом і енергією, яку воно транспортує. Коли людське око дивиться на джерела світла, промінь або предмети, вони можуть здаватися нам більш-менш яскравими. Фотометрія вимірює ці «кількості світла», що сприймається оком, випромінюється джерелом або відбивається об’єктом. У фотометрії необхідно враховувати чотири параметри:

- інтенсивність світлового променя;
- світловий потік;
- освітленість;
- яскравість.

Сила світла променя – це сила світлового потоку в даному напрямку. Одиниця вимірювання – кандела (кд). Світловий потік – це загальна кількість світла, випромінюваного (джерелом), транспортованого (променем) або

отриманого (об'єктом) за секунду. Одиницею вимірювання є люмен. Освітлення – це світло, яке отримує об'єкт, винесений на поверхню. Одиницею є люмен на $м^2$ або люкс. Яскравість – це те, що відбиває поверхня, освітлена світловим потоком. Одиницею є кандела на $м^2$ ($кд/м^2$)

Сила світла та світловий потік характеризують лише джерело, освітленість характеризує освітлену поверхню, а яскравість характеризує світло, яке сприймає око. У фотографії освітленість і яскравість особливо важливі для отримання знімка з хорошою експозицією.

Біле світло є додаванням або комбінацією кількох довжин хвиль (рисунок 1.3). Фотографія є адитивним синтезом, білий відновлюється за допомогою додавання основних кольорів червоного, зеленого та синього (RGB). З іншого підходу існує субстрактивний синтез, проекція здійснюється за допомогою білого світла, з якого частина випромінювання видаляється для створення очікуваних кольорів. Додавання додаткових кольорів блакитного, пурпурового та жовтого (CMY) створить чорний колір.

а) б)

Рисунок 1.3 – Приклад формування кольорів в залежності від типу накладання світлових хвиль: (а) адитивний синтез та (б) субстрактний синтез

На якість отриманих цифрових зображень впливають багато факторів, одним з основних є рівень освітленості. Окрім того, що він жалежить від інтенсивності джрежа освітлення, не менш важливим є вид освітлення. Виділяють чотири основних напрямки освітлення, що використовуються в процесі створення цифрових зображень (рисунок 1.4).

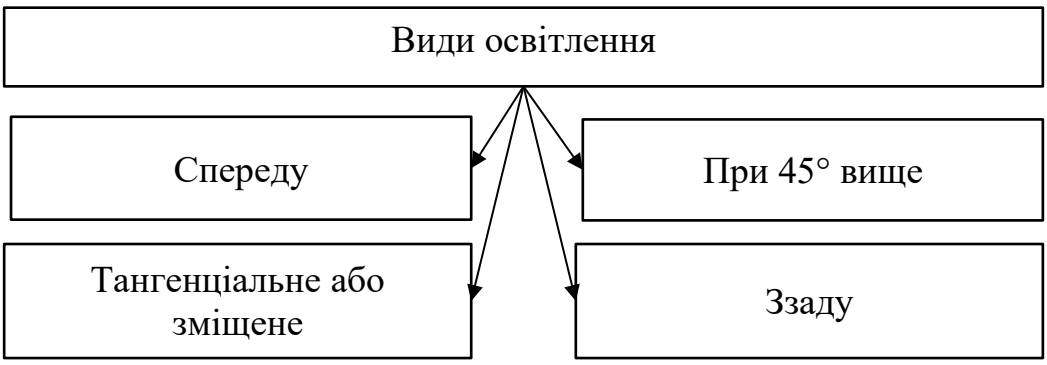

Рисунок 1.4 – Класифікація видів освітлення

Спереду – це освітленння при якому світло згладжує рельєф і дає «пласке» зображення. Щоб зменшити цей ефект, між світлом і об’єктом можна помістити розсіювач (кальку або легку тканину). При 45° вище дозволяє створити освітлення, яке набуває приємної рельєфності, а якщо тіні трохи різкі, їх можна пом'якшити відбивачем. Тангенціальне або зміщене. Цей тип освітлення створює тіні і тим самим підкреслює підсвічування рельєфів. Особливо підходить для фотографування слідів взуття або відбитків пальців з відкладенням матеріалу (або виявлених за допомогою ціаноакрилату). При використання освітлення ззаду світло створює контрове освітлення, і для правильної експозиції фотографії необхідно використовувати додатковий спалах.

Люмінесценція – це фізико-хімічне явище, яке широко використовується технічною та науковою поліцією. Люмінесценція характеризується випромінюванням світла попередньо збудженої речовини. Певні речовини мають здатність поглинати світло та повторно випромінювати його з іншою довжиною хвилі. Освітлена речовина зберігає частину енергії падаючого випромінювання E_1 і повертає нове, менш енергетичне випромінювання E_2 . Оскільки довжина хвилі випромінювання обернено пропорційна його енергії, довжина хвилі випущеного випромінювання λ_2 більша за довжину хвилі падаючого випромінювання λ_1 :

$$E_1 \geq E_2 => \lambda_1 \leq \lambda_2.$$

Втрата енергії зумовлена електронними переходами в освітленій речовині. Електрони, присутні в молекулах, фактично переходитимуть з основного стану в збуджений стан, а потім повертатимуться в основний стан, випромінюючи світло. Люмінесценція – це загальний термін, який охоплює явища фосфоресценції та флуоресценції. Флуоресценція – світіння, яке припиняється при припиненні джерела збудження. Це може, наприклад, статися, якщо слід сперми освітлюється джерелом світла з довжиною хвилі, близькою до 400 нм. Тоді можна тимчасово спостерігати нове світло з довжиною хвилі, близькою до 500 нм. Як тільки падаюче випромінювання припиняється, випромінюване також припиняється. З іншого боку, що стосується фосфоресценції, випромінювання світла продовжується, навіть якщо джерело падаючого світла зупинено. Це явище дає змогу розрізняти об’єкти в темряві без постійної подачі енергії, але наразі не становить інтересу для технічної та наукової сфер.

Фільтри розміщуються перед об’єктивами і призначені для ефекту зображення або для усунення технічної проблеми. З програмами для редагування цифрових зображень (Photoshop, Photofiltre, Gimp та інші) фільтри стають все менш корисними, але деякі фільтри залишаються цікавими:

- фільтр проти ультрафіолетового випромінювання. Дані фільтри не впливають на отримане зображення, але допомагають захистити лінзу від зовнішніх елементів, які можуть впливати на лінзу (пісок, удари, вода тощо)
- поляризаційні фільтри. Лінійні або кругові поляризаційні фільтри мають основний ефект усунення неметалевих відблисків (води, скла тощо). Проте будьте обережні з лінійними поляризаційними фільтрами, які можуть спричинити проблеми з експозицією або фокусуванням. Тому краще віддавати перевагу фільтрам з круговою поляризацією. Фотограф повинен зробити знімок під кутом 45° до падаючого світла.
- кольорові фільтри. Кольорові фільтри дозволяють освітлювати або затемнювати певні кольори. Ці фільтри пропускають довжини хвиль свого кольору та блокують частину інших світлових променів. Червоний фільтр,

наприклад, пропускатиме помаранчеве та червоне випромінювання та зупинятиме зелене та синьо-зелене випромінювання.

Цифрове зображення складається з набору точок, які називаються пікселями. Піксель (скорочення від PICture ELement) визначається як найменший складовий елемент растрового цифрового зображення. Для двотонового чорно-білого зображення піксель може бути закодований одним бітом кодування (0 для чорного або 1 для білого). Для зображень у відтінках сірого або кольорових піксель може бути закодований 2, 4, 8, 16, 24 або 32 бітами. Чим більше пікселів на зображенні, тим більша «чіткість» зображення. Зображення визначається як загальна кількість пікселів, що утворюють цифрове зображення. Чим більша кількість пікселів, тим краща якість необробленої фотографії. Кількість пікселів безпосередньо пов’язана з якістю сенсора та камери. Зображення дуже високої чіткості забезпечить багате на деталізацію зображення та полегшить можливість збільшення зображення. Кожна цифрова камера має свою оптимальну роздільну здатність (є пристрої високої чіткості з роздільною здатністю понад 50 Мп), але можна встановити інші, нижчі чіткості безпосередньо на пристрої (щоб не надто швидко насичувати карту пам’яті).

Якщо зображення має занадто низьку роздільну здатність, його збільшення може призвести до пікселізації зображення. Тому вибір визначення безпосередньо пов’язаний із тим, що необхідно зробити із зображенням. Проблема пікселізації часто виникає при перегляді слідчими службами зображень з камер відеоспостереження. Зображення, яке в основному має низьку чіткість, не зможе покращити якість деталей, навіть за допомогою програмного забезпечення для обробки зображень і операцій збільшення, зміни яскравості, кольорів або контрасту. Наприклад, зображення, отримані з відеоспостереження в міських центрах, хоч і дуже інформативні, але не часто дають змогу визначити реєстрацію транспортних засобів (хоча окремі методи іноді виправляють цю проблему). Збільшення пластини або покращення яскравості не призведе до створення інформації, яка спочатку не існує. Спочатку, якщо на вході є зображення з 32 пікселів з інформацією про кольоровість і яскравість для

кожного пікселя, було б ілюзорно думати, що створення 32 додаткових пікселів за допомогою програмного забезпечення дасть інформацію там, де її немає. Додаткові 32 пікселі справді можна створити, але вони будуть створені з даних існуючих пікселів. Це процес «інтерполяції». Цей процес суворо заборонений або настійно не рекомендується під час використання фотографій цифрових слідів, оскільки він змінює реальність зображення. Створені пікселі є «фіктивними», вони не відповідають дійсності зображення. Приклад застосування інтерполяції наведено на рисунку 1.5.

Рисунок 1.5 – Приклад використання інтерполяції растрового цифрового зображення

Розмір створюваного цифрового файлу безпосередньо залежатиме від чіткості зображення. Чим більше визначення, тим вищою буде початковий розмір. Розмір цифрового файлу також залежатиме від інших факторів, які можна регулювати безпосередньо на камері, як-от формат запису зображення та глибина кольору.

Формат зображення вказує на спосіб його збрігання та особливості обробки в процесі отримання. Формати можуть бути націлені на оптимізацію одного чи декількох характеристик, а тому під час вибору конкретного формату необхідно проаналізувати завдання які ставляться перед відповідними

цифровими зображеннями. Приклади нафбільш відомих форматів цифрових зображень, що використовуються у сучасній техніці наведено на рисунку 1.6.

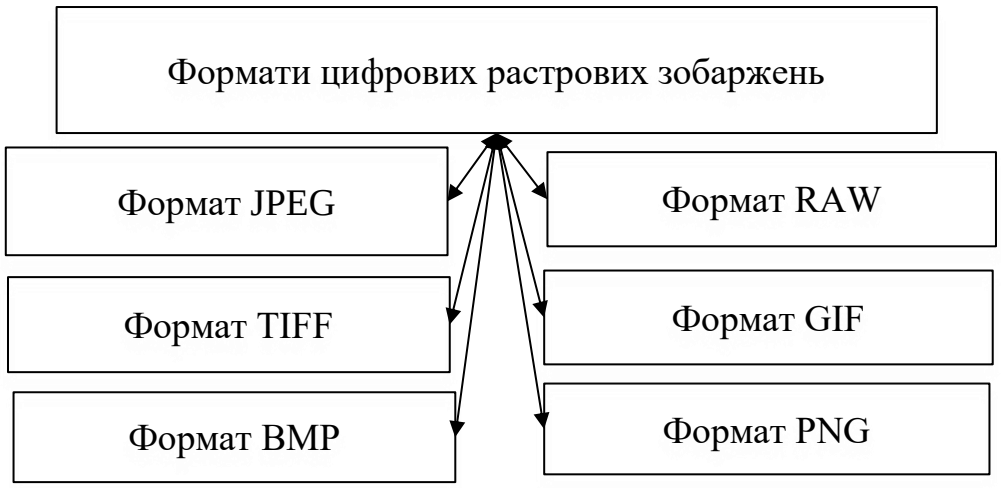

Рисунок 1.6 – Формати цифрових растрових зобаржень

JPEG формат належить до групи які стискають зображення. Він найчастіше використовується в цифровій фотографії, оскільки пропонує чудове співвідношення між вагою та якістю зображення. Цей формат можна використовувати з різними рівнями стиснення (високим, нормальним або базовим), кожен рівень зменшує розмір файлу, а також якість зображення. Як це працює, використання стиснення JPEG зменшує глибину кольору зображення та створює блоки одного кольору, усуваючи проміжні відтінки.

RAW – це «необроблений» формат, який передає інформацію безпосередньо з датчика. Формат RAW безпосередньо записує надану інформацію без її обробки. Це призводить до отримання якісного зображення, але значної ваги (яка іноді може перевищувати 50 МБ для пристроїв із роздільною здатністю понад 20 Мпікселів). Програмне забезпечення для обробки зображень пропонує можливість трансформувати файл RAW у зображення TIFF, GIF, BMP або навіть PNG.

Формат TIFF – це формат без стиснення або зі стисненням LZW, який відновлює зображення без втрати деталей чи якості. Недоліком цього формату є дуже великий розмір створюваного файлу.

Формат GIF пропонує стисне зображення, зберігаючи хорошу візуалізацію, але який залишається обмеженим щодо глибини кольору (обмежений до 256 кольорів). Цей формат дозволяє створювати анімовані зображення.

Формат BMP (або Bitmap) від Microsoft Windows за замовчуванням, який пропонує файл без стиснення і тому дуже важкий.

Формат PNG – це формат, який забезпечує чудове стиснення без втрат.

Вибір формату має здійснюватися відповідно до бажаної якості зображення, економії місця на носії запису, простоти ретушування зображення або навіть його довговічності.

Ще один важливий параметр – це роздільна здатність зображення. Роздільна здатність представляє кількість пікселів на одиницю площі. Це щільність. Вона виражається в dpi (точка на дюйм) або ppp (піксель на дюйм). Роздільна здатність набуває повного значення, коли зображення друкується на фізичному носії. Дюйм має розмір 2,54 см, що означає, що для роздільної здатності 16 пікселів на дюйм друковане зображення складається з 16 пікселів для ширини 2,54 см і 16 пікселів для висоти 2,54 см, тобто квадрат 16×16 = 256 пікселів (рисунок 1.7).

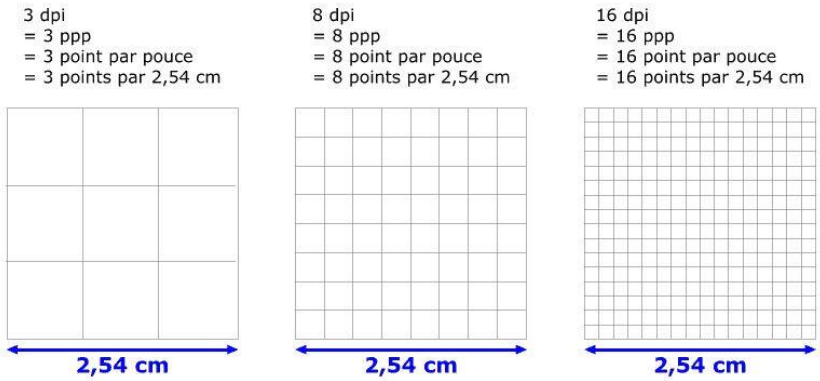

Рисунок 1.7 – Приклади залежності якості зображення від роздільної здатності

Роздільна здатність 300ppi означає, що лінія 2,54см складається з 300 пікселів, тому розмір пікселя становить приблизно 0,085мм (2,54см/300). З огляду на роздільну здатність ока (здатність візуально розділяти два різні об'єкти) ця роздільна здатність є необхідною, щоб запропонувати рендеринг, досить близький до класичної плівкової фотографії з форматом 10x15 см, і щоб око не розрізняло пікселі за перегляд надрукованого зображення з відстані понад 20 см. Сканери також здатні розрізняти більш-менш дрібні деталі. Коли сканер сканує зображення з роздільною здатністю 500 dpi, він здатний розрізняти 500 пікселів кожні 2,54 см.

Після налаштування чіткості та формату зображення глибина кольору є іншим параметром, який можна налаштувати. Можна налаштувати кількість тонів і кольорів. Така глибина дозволяє оптимізовувати розмір та якость цифрового зображення. Глибина кольору визначається в бітах. Це кількість тонів або кольорів, які може приймати піксель:

- коли глибина кольору становить 1 біт, кожен піксель зображення може мати два значення: 0 або 1. 0 відповідає чорному, а 1 – білому. Тоді створене зображення буде чорно-білим;
- коли інші біти використовуються для опису кожного пікселя, інші тони (сірі) потраплятимуть між чорним і білим. Отже, для роздільної здатності 2 біти 22 = 4 тони чорного, білого та 2 сірих тонів можна створити з кодуванням 00, 01, 10 і 11;
- для глибини кольору 8 біт (один байт) піксель може приймати 28 = 256 тонів сірого або кольору;
- створення кольорової фотографії здійснюється шляхом адитивного синтезу червоно-зеленого та синього кольорів (RGB). Кожен колір кодується 256 рівнями, і саме суміш цих трьох кольорів створить остаточний колір. Тому піксель кодується сумішшю 3 кольорів на 256 рівнях (2563), що становить приблизно 16 мільйонів кольорів;
- існують інші системи кольорів, наприклад CMYK (блакитний, пурпуровий, жовтий і чорний), що є субтрактивним синтезом, який

використовується для чотириколірного друку. Кодування виконується на 32 бітах.

Чим більше використовується кількість бітів, тим більше кольорів створеного цифрового зображення буде відповідати реальності. З іншого боку, розмір створеного образу буде більшим.

1.2 Основні визначення та поняття теорії графів

Теорія графів є потужним інструментом для схематизації моделей зв’язків і відносин між об’єктами. Вивчення графів почалося у 18 столітті з проблеми математичної цікавості, коли Ейлер поставив знамениту проблему Кенігсберзького мосту. Протягом останніх двох десятиліть теорія графів викликала експоненціальний інтерес головним чином завдяки своїй ролі моделі оптимізації та явних обчислень, що вимагають розробки та аналізу кількох алгоритмів. Механізми теорії графів займають провідні позиції в інформатиці, прикладній математиці (матричний числовий аналіз), біології, фізиці (електричні схеми), хімії тощо. Теорія графів стала одним із найефективніших інструментів для вирішення багатьох дискретних проблем. Використання наробок з даного напряму сприяє вирішенню багатьох конкретних проблем повсякденного життя. Таким чином, найкращий спосіб правильного засвоєння ця теорія повинна початися з простих графів, оскільки кожен орієнтований граф пов’язаний з неорієнтованим графом, опускаючи орієнтації, і будь-яка неорієнтована концепція застосовується до будь-якого типу графа, а отже можна здійснювати перехід між двома типами. Таким чином, оволодіння простими графами значно полегшує засвоєння орієнтованих графів, а отже і інших технологій в даному напряіму.

Дійсно, моделювання з використанням теорії графів є дуже ефективним для аналізу проблем, які включають набори об’єктів, між якими існують зв’язки.

Нам може бути цікаво, що таке визначення теорії графів і як вона втручається у вирішення проблем великих даних. Ми можемо побачити, як провести навчання даних, щоб знати цю теорію.

Граф визначається як набір елементів, які пов'язані один з одним. Їх геометричне представлення виконується за допомогою моделей, що складаються з точок (також званих вершинами або вузлами), з'єднаних кривими лініями (також званими ребрами, зв'язками або стрілками). Ребра можуть бути несиметричними і тоді розглядаються як стрілки або дуги. Коли ми вирішуємо орієнтувати їх і/або призначаємо їм вагу, кажуть, що графіки орієнтовані або зважені.

Простий граф $G= (V, E)$ – це дані двох множин V та E, де E – множина пар елементів $V,E \subseteq V(2)$. Елементи V називаються вершинами, а елементи E – ребрами. Ребро e визначається двома вершинами u і v, які називаються кінцями ребра e. У цьому випадку пишемо $e=[u, v]$ або просто $=uv$. Дві вершини u і v називають суміжними або сусідніми, якщо вони є кінцями одного ребра $=[u,v] \in E$. Два ребра називаються суміжними, якщо вони мають спільний кінець G, який називається порядком G. Позначимо його $|G|$. Кількість ребер G називається розміром G. Позначимо це через $t(G)$. Отже, $t(G) =|E|$. Кількість сусідів вершини $x \in V$ називається ступенем x і позначається $d(x)$. Вершина називається початковою, якщо $d(v) = 1$. Вершина називається вузлом, якщо $d(v) \geq 2$. Вершина степеня 0 називається ізольованою. Стійкою (або незалежною) частиною графа $G=(V,E)$ будемо називати будь-яку частину U вершин G, які попарно не є сусідніми. Простий граф $G= (V, E)$ дорівнює нулю, якщо V E порожні. простий граф G дитривіальний, якщо він допускає одну вершину і не має ребра, ми позначаємо його T_1. Тривіальним графом порядку n називаємо граф, що має n вершин і не має ребер, позначаємо O на Tn.

Потім теорія графів вивчає численні властивості цих представлень. Це наявність найкоротших шляхів, найменш дорогих шляхів, кількість перехресть на плані, проблеми забарвлення, окремі цикли тощо.

Класичне вивчення даних складається з представлення їх у вигляді таблиці, суміжної матриці. У традиційній системі баз даних (яка не використовує власний графік) нам доведеться перетинати всі дані в рядках і стовпцях. Це дає змогу ідентифікувати зв’язки перед бізнес-алгоритмом. Додаючи додаткові дані, традиційна система також передбачає інтеграцію стовпця та рядка в матрицю, щоб зробити цей перехресний аналіз можливим. На практиці обсяг даних, які потрібно обробити, експоненціальний порівняно з традиційною операцією. Обсяг даних швидко стає гігантським, навіть якщо він належить до підмножини менших бізнес-даних. Щоб вирішити проблему часу обчислення, фахівець з даних повинен вибрати дані для аналізу. Система не повинна бути заблокована або передбачати надмірно тривалий час обробки з результатами, отриманими лише після того, як сталося шахрайство.

Структуровані дані – це форма організації інформації, яка використовує теорію графів для створення зв’язків між різними елементами даних. У сфері обробки цифрових зображень або електронної комерції ці структуровані дані дозволяють пошуковим системам краще розуміти та контекстуалізувати поставлені задачі.

Використовуючи теорію графів, дані безпосередньо представлені «вузлом», без необхідності генерувати непотрібні та громіздкі дублікати. Щоб додати дані до графіка, нам не потрібно буде множити розмір бази в геометричній прогресії. Це дозволяє спеціалісту з даних обробляти всі корисні для нього дані, не обмежуючи продуктивність його системи аналізу. Простіше кажучи, теорія графів — це дуже старий математичний принцип.

Існує кілька типів графічних зображень які відповідають різним типам графів. Неорієнтовані графи – це ті, які не вказують жодного фіксованого напрямку між вузлами. У цьому випадку ребро від вузла *A* до *B* буде ідентичним ребру від *B* до *A*. Цей тип графа можна використовувати для представлення кожним вузлом фіксованого пункту призначення, а за ребрами – двонаправлені маршрути до них.

Орієнтовані графи дозволяють візуалізувати орієнтацію або напрямок між різними вузлами . Це означає, що коли ребро (зображене тут стрілкою) йде від вузла *A* до вузла *B*, а отже можемо рухатися лише від А до В. Протилежне буде можливим, лише якщо друга стрілка йде від В до А. Використовуючи приклад із пунктами призначення можемо отримати напрямок від міста А до міста В.

Зважені графіки мають ребра, які містять відповідну вагу та представляють наслідки реального світу. Цей тип графів може бути орієнтованим і неорієнтованим. У прикладі з пунктами призначення цією новою змінною може бути вартість транспортування з одного міста в інше, час у дорозі або навіть відстань, залежно від оброблених даних. Тому зважені графіки часто використовуються в програмуванні GPS і пошукових системах. Вони використовуються для планування подорожей і пропонують порівняння часу польоту та вартості.

Теорію графів можна використовувати для моделювання взаємозв'язків і процесів в інформаційних системах, фізичних, біологічних або навіть соціальних системах (рисунок 1.8).

Рисунок 1.8 – Сфери застосування елементів теорії графів

1.3 Програмні бібліотеки для обробки та аналізу цифрових зображень

OpenCV, або якщо у повному варіанті Open Source Computer Vision Library – це широковикористоввана бібліотека, яка містить функції комп’ютерного зору та машинного навчання, що поширюються відкритим кодом. Дана розробка пропонує понад 2500 алгоритмів, оптимізованих для різних завдань комп’ютерного зору, таких як обробка зображень, захоплення відео, аналіз руху та розпізнавання об’єктів. Розроблений для оптимізації операцій у реальному часі, OpenCV підтримує багатоядерну обробку та використовує ефективне апаратне прискорення.

OpenCV широко використовується в галузі штучного інтелекту та машинного навчання завдяки потужним можливостям обробки зображень і відео. Він полегшує розробку моделей машинного навчання, надаючи інструменти попередньої обробки даних, що має вирішальне значення для ефективного навчання моделей ШІ. Основні області застосування включають розпізнавання жестів, виявлення обличчя та відстеження об'єктів. Його сумісність з багатьма мовами програмування, такими як Python, C++ і Java, гарантує його доступність для широкої аудиторії. OpenCV є універсальним і використовується в багатьох секторах (рисунок 1.9).

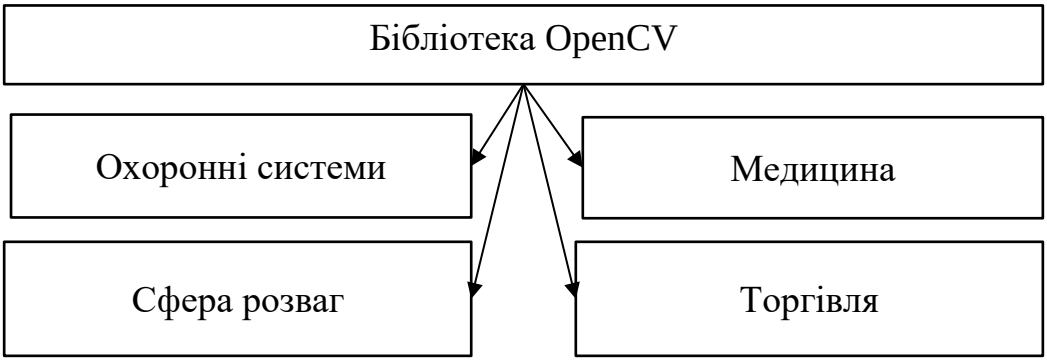

Рисунок 1.9 – Сфери застосування функцій бібліотеки OpenCV

Засоби даної бібліотеки допомагають аналізувати медичні зображення для діагностики, підвищуючи точність таких процедур, як виявлення пухлин або порушень. Також, використання даної бібліотеки підтримує технологію автономного водіння, вмикаючи функції розпізнавання зображень, що є критично важливими для навігації та безпеки. В роздрібній торгівлі дана бібліотека використовується для моніторингу та аналізу клієнтів за допомогою обробки відео в реальному часі для покращення безпеки та бізнес-стратегій.

OpenCV став наріжним каменем у розробці систем розпізнавання облич. Його ефективність у розпізнаванні обличчя в реальному часі пояснюється його здатністю вирішувати складні завдання, як-от визначення рис обличчя за допомогою каскадів Хаара або моделей глибокого навчання. Розпізнавання облич використовується в системах безпеки для швидкої та точної ідентифікації особи.

У контексті автономних транспортних засобів OpenCV допомагає з виявленням і класифікацією об’єктів, що має вирішальне значення для інтерпретації середовища. Технологія обробляє канали камери, щоб ідентифікувати світлофори, пішоходів та інші транспортні засоби, забезпечуючи безпечне водіння.

На відміну від подібних бібліотек, OpenCV оптимізовано для додатків у реальному часі та підтримує декілька платформ – Windows, Linux, Android та macOS. Його інтеграція з іншими фреймворками штучного інтелекту, такими як TensorFlow і PyTorchOpenCV, інтеграція з іншими фреймворками штучного інтелекту, такими як і , дозволяє розробляти широкі програми, які використовують сильні сторони різних технологій.

Для користувачів Ultralytics HUB інтеграція з OpenCV може покращити обчислювальні завдання, забезпечуючи безперебійний робочий процес від навчання моделі до розгортання. Ultralytics YOLO з OpenCV може покращити обчислювальні завдання, забезпечуючи безперебійний робочий процес від навчання моделі до розгортання.

OpenCV продовжує залишатися важливою бібліотекою для створення розширених програм штучного інтелекту, надаючи дослідникам і розробникам необхідні інструменти для розробки складних рішень для бачення. Його незмінне значення полягає в гнучкості та ефективності, які він привносить у різноманітні технологічні досягнення в різних секторах.

1.4 Постановка задач дослідження

В даному розділі було основну увагу було приділено дослідженню технологій створення цифрових зображень, виділення основних факторів, які впливають на характеристики майбутнього зображення під час його отримання за допомогою фотофіксуючої апаратури. Додатково було проаналізовано основні формати цифрових зображень, що широко використовуються у сучасних програмних розробках. Було проаналізовано основні елементи та сфери застосування теорії графів та досліджено види та особливості окремих параметрів графів та їхній вплив на кіцейвий аналіз отриманих даних. Додатково дуло розглянуто бібліотеку функцій для обробки та опису цифрових зображень OpenCV. На основі отриманих даних в кваліфікаційній робті було поставлено наступні завдання:

- проаналізувати технології створення та опису цифрових зобаржень;
- дослідити методи та алгоритми теорії графів;
- провести аналітичний огляд цифрових бібліотек та програмних засобів оброки цифрових зображень;
- проаналізувати методи та алгоритми сегментації цифрових зображень;
- розробити алгоритм сегментації цифрових зображень на основі теорії графів;
- реалізувати програмний засіб обробки та опису цифрових зображень, провести його тестування та порівняння з аналогами.

1.5 Висновки до розділу

Здійснено аналізу технологій створення цифрових зображень на основі дослідження параметрів процесу створення цифрових відображень сцен, що дозволило виділити основні фактори які впливають на якість вихідних цифрових зображень.

Досліджено методи та алгоритми теорії графів, на основі аналізу характеристик різних типів графів, що дозволило виділити основні сфери їх застосування.

Проведено аналітичний огляд цифрової бібліотеки оброки цифрових зображень OpencvCV, на основі дослідження її функціональних можливостей та використовуваних структур даних, що дозволило виділити основні функції для обробки цифрових зображень.

2 МЕТОДИ ТА АЛГОРИТМИ ОБРОБКИ ЦИФРОВИХ ЗОБРАЖЕНЬ

2.1 Алгоритми попередньої обробки цифрових зображень

У наш час велика кількість даних записується в цифровому форматі, тому для інтерпретації та аналізу майже всіх зображень потрібна цифрова обробка. Обробка цифрових зображень може використовувати різні процеси, включаючи форматування та корекцію даних, цифрове покращення для полегшення візуальної інтерпретації або навіть автоматичну класифікацію цілей і структур повністю комп'ютером. Цифрова обробка зображень вимагає, щоб дані були записані та доступні в цифровому форматі, придатному для зберігання на комп'ютерних сховищах даних.

Для обробки цифрових зображень, очевидно, потрібна комп'ютерна система (або система аналізу зображень), а також обладнання та програмне забезпечення для обробки даних. Програмні системи для обробки цифрових зображень, як правило виконують даних процес у декілька етапів (рисунок 2.1).

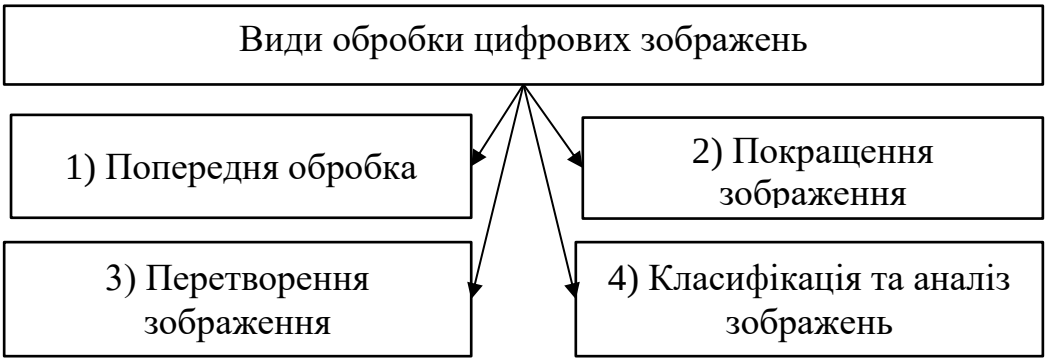

Рисунок 2.1 – Етапи обробки цифрових зображень

Операції, які зазвичай потрібні перед основним аналізом і вилученням інформації називаються функціями попередньої обробки. Операції попередньої обробки включають, серед іншого видалення шумів, що були отримані впроцесі оцифровування сцени або конвертація у більш зручніших формат. Функції

покращення призначені для покращення зовнішнього вигляду зображень для сприяння візуальній інтерпретації та аналізу. Функції покращення дозволяють розтягнути контраст, щоб збільшити тональне розрізнення між різними елементами в сцені, і просторову фільтрацію, щоб підсилити (або усунути) певні просторові візерунки в зображенні.

Перетворення зображень – це операції, подібні до покращення зображення. Однак у той час як покращення зображення зазвичай застосовується до однієї смуги даних за раз, трансформація зображення поєднує обробку даних із кількох спектральних смуг. Арифметичні операції (тобто додавання, віднімання, множення, ділення) виконуються для об'єднання та перетворення оригінальних стрічок у «нові» зображення, які чіткіше показують певні елементи сцени. Операції класифікації та аналізу зображень використовуються для цифрової ідентифікації та класифікації пікселів на зображенні. Класифікація зазвичай виконується в мультиспектральних базах даних (A), і цей процес надає кожному пікселю зображення певний клас або тему (B) на основі статистичних характеристик значення інтенсивності пікселя. Існують різноманітні підходи до чисельної класифікації. Операції класифікації широковикористовуються і різнх цифрових системах автоматизованої обробки даних, що в свою чергу накладає ряд вимог до якості та швидкості їх роботи. Проте, очевидиним є той факт, що на отриману класифікацію напряму впливають результати, що були отримані на попередніх етапах аналізу. Тому, при проектуванні програмних систем обробки цифрових зображень необхідно враховувати та максимально ефективно використовувати алгоритми попредньої обробки.

Процедура сегментації відбувається після етапів попредньої обробки, а ї результат залежить від отриманих результатів. Чим якісніше буде проведена попередня обробка тим більша ймовірність отримани якісний результат в процесі розділення зображення на однорідні області. На рисунку 2.2 наведено перелік операцій які проводяться для підвищення якості результатів сегментації.

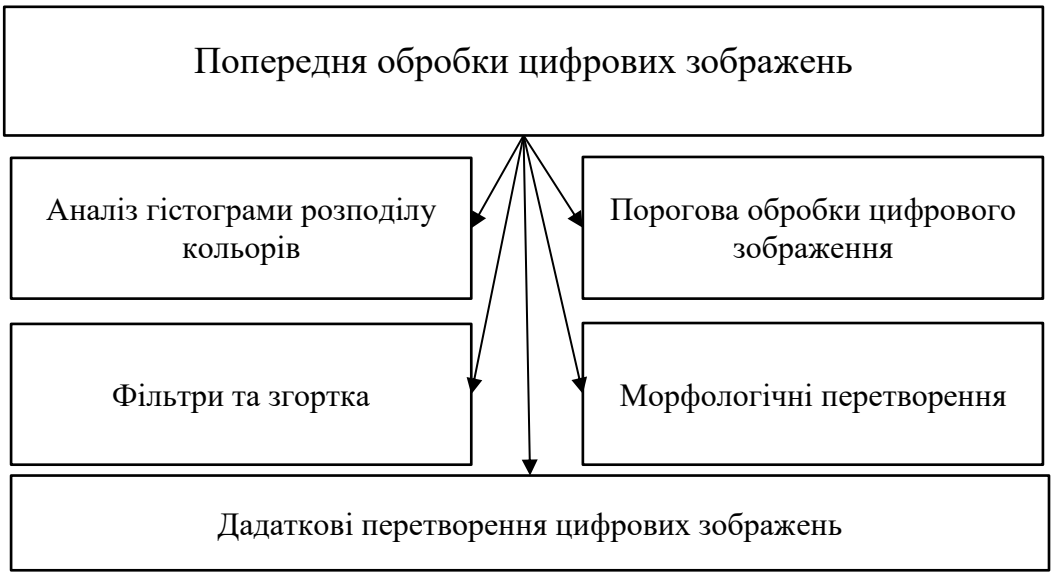

Рисунок 2.2 – Алгоритми попередьої обробки зображень

У цифровому кольоровому зображені кожен піксель є кортежем або інакше суперпозицією каналів кольорів (червоний, зелений і синій, тобто [*R*, *G*, *B*]), тому для отримання результатів обробки цифрових зображень необхідно проводити аналіз розподілу цих трьох основних кольорів у зображенні.

Гістограма зображення – це не графік, який відображає на осі абсцис різні значення каналу/пікселя, а на ординаті кількість каналів/піксель, які мають це значення (рисунок 2.3).

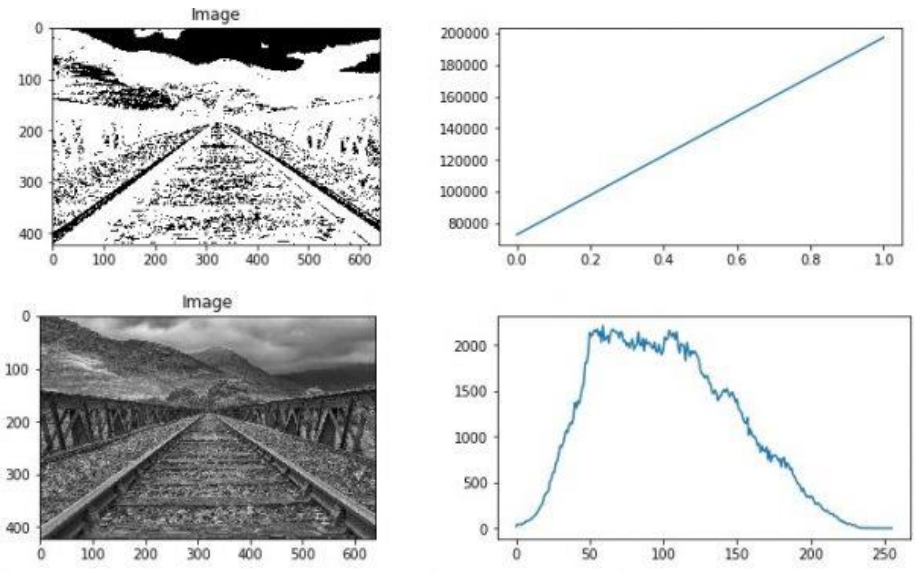

Рисунок 2.3 – Приклад гістограми розподілу значень яскравості

Дуже часто для створення чорно-білого зображення використовується техніка порогового визначення: це називається бінарним пороговим визначенням. У цих випадках здається логічним базувати його на основі середнього пікселів. Усе, що нижче середнього, буде встановлено на 0, а все, що вище, на 1. Іншим варіантом є використання порогового визначення Оцу. Даний підхід для визначення порогів використовує інформацію засновану на формі гістограми зображення.

Для кольорових зображень принцип порогового значення такий самий, але з тією різницею, що необхідно повторити операцію для кожного з каналів (RGB). Для прикладу візьмемо кольорове зображення та проведемо аналіз отриманої гістограми (рисунок 2.4).

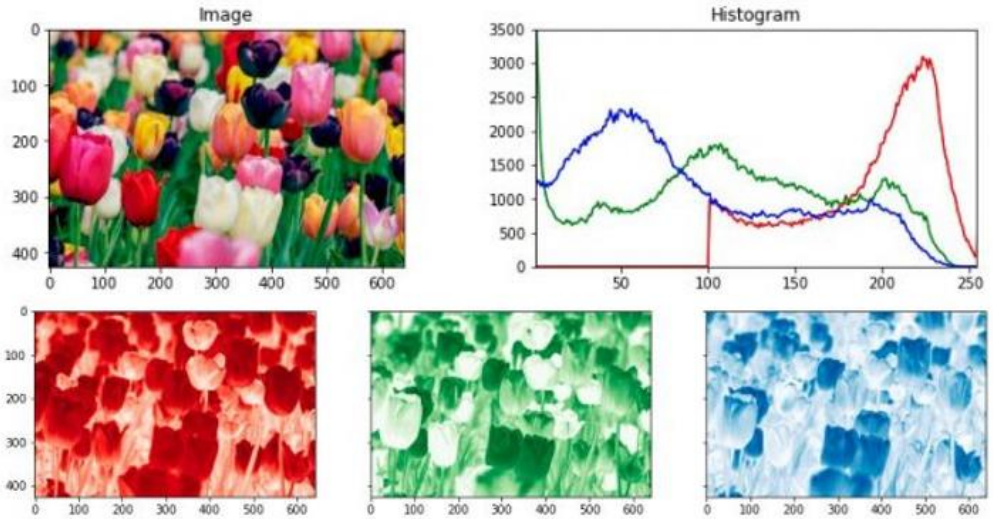

Рисунок 2.4 – Приклад аналізу гістограми розподілу за 3 кольоровими каналами

Гістограма цифрового зображення – це статистична крива**,** що відображає розподіл його пікселів відповідно до їх інтенсивності. Для чорно-білого зображення він вказує по осі абсцис рівень сірого (ціле число від 0 до 255), а по осі ординат кількість пікселів, що мають це значення. Коли гістограма нормалізована, вона вказує на ординату ймовірнісного розташування пікселя з відповідним рівнем сірого. Тоді інтенсивність пікселя розглядається як дискретна випадкова змінна.

Нормована кумулятивна гістограма обчислює відсоток пікселів зі значенням, меншим за заданий рівень сірого. Нормалізовану гістограму можна інтерпретувати як щільність ймовірності, а нормалізовану кумулятивну гістограму як функцію розподілу. Це дуже важливий інструмент для обробки зображень, оскільки його модифікація дозволяє регулювати динаміку рівнів сірого або кольорів зображення, щоб зробити його візуально приємнішим. Грубо кажучи, зліва чорні пікселі, праворуч білі пікселі, а посередині всі відтінки сірого.

Розтягування гістограм полягає у виправленні яскравості або експозиції зображення. Наприклад, якщо на зображеня є занадто темним або недотриманим, то більшість пікселів розташовано в лівій частині гістограми, у напрямку до низьких значень сірого. І навпаки, пікселі зображення, які надто світлі або перетримані , зосереджені в правій частині гістограми. З іншого боку, гістограма, пов’язана із зображенням із відносно хорошою експозицією, представляє розподіл пікселів по всьому інтервалу [0,255]. Таким чином, щоб виправити дефекти, пов’язані з експозицією зображення, достатньо просто розтягнути його гістограму. Мета полягає в тому, щоб розширити значення рівнів сірого погано експонованого зображення, в основному розподілених у підінтервалі.

Якість фотографії також може бути погіршена цифровим шумом, тобто випадковою появою зайвих «зерен». Це поширене явище в цифровій фотографії, викликане неправильним налаштуванням чутливості датчиків камери або обмеженням їх можливостей. Шум можна розглядати як зображення, що складається з пікселів, інтенсивність яких визначена випадковим чином. Оскільки зображення визначається або як функція, або як матриця, то можемо застосувати звичайні математичні операції, такі як додавання. Таким чином, додатковий шум, це коли зашумлене зображення є сумою вихідного зображення та шуму. Координати інтенсивності пікселів (x,y) у зашумленому зображенні задається співвідношенням:

$$I'(x,y)=I(x,y)+\eta(x,y)I'(x,y)=I(x,y)+\eta(x,y).$$

Класичним прикладом адитивного шуму є гауссовий шум, для якого інтенсивності вибираються випадковим чином відповідно до нормального закону. Для зменшення шуму на зображенні було розроблено декілька методів усунення шумів (або згладжування). Згладжування за допомогою усереднення є найбільш інтуїтивно зрозумілим рішенням.

Що робить зображення візуально неприємним, так це наявність певних пікселів, які не узгоджуються з іншими елементами представленої сцени: шум надав їм «спотворені» значення. У якісному зображенні піксель, як правило, має інтенсивність, відносно подібну до інтенсивності його сусідів. Тоді згладжування за допомогою усереднення полягає в заміні значення кожного пікселя середньою інтенсивністю його околиць. Околиця визначається квадратним вікном непарного розміру з центром у пікселі, який потрібно виправити.

Фільтрація є важливим аспектом обробки зображень, і одна з її головних цілей полягає в тому, щоб очистити зображення, усунувши якомога більше шуму.

Існують різні методи фільтрації залежно від типу шуму, який потрібно послабити. Згладжування за допомогою усереднення використовує лінійний фільтр і в цьому сенсі є частиною найпростішого класу фільтрації. Оператор називають лінійним, якщо він задовольняє такі дві властивості: адитивність та однорідність. В обробці зображень і навіть в загальній обробці сигналів лінійний фільтр – це система, яка перетворює зображення за допомогою лінійного оператора. Важливою властивістю лінійного фільтра є трансляційна інваріантність – модифікація пікселя залежить від його сусідства, а не від його положення на зображенні. Для згладжування шляхом усереднення використовується лінійний фільтр, який називається фільтром усереднення. Це згладжування ґрунтується на принципі, що кожен піксель у якісному зображенні повинен мати інтенсивність, подібну до інтенсивності його сусідів. Середнє арифметичне навіть припускає, що кожен сусід робить рівний внесок. Це може призвести до неузгодженості, особливо якщо кількість розглянутих сусідів

велика. Існує велика ймовірність того, що піксель буде виправлено залежно від пікселів, які описують різні об’єкти на зображенні, а отже, мають мало спільного з ним. Таким чином, можна покращити згладжування за допомогою усереднення, надаючи різну вагу пікселям у середньому, тобто усі коефіцієнти більше не мають однакові значення й залежать від положення сусіда відносно центрального пікселя.

Лінійний фільтр замінює значення кожного вхідного пікселя лінійною комбінацією інтенсивностей його сусідніх пікселів. Оператор для виконання цього перетворення називається продуктом згортки. Ось чому застосування лінійного фільтра також відоме як згорточна фільтрація . X є ядром згортки (або маскою), якщо це квадратна матриця непарного розміру $2k+2k+1$, що характеризує застосований лінійний фільтр. Вибір коефіцієнтів і розмір ядра згортки X дозволяє визначити безліч різноманітних лінійних фільтрів. Двома найпопулярнішими типами лінійних фільтрів є фільтри усереднення та фільтри Гауса. Фільтр Гаусса – це лінійний фільтр, елементи ядра згортки якого визначаються відповідно до щільності закону Гаусса з центром у двох вимірах.

Незважаючи на те, що лінійні фільтри легко розробити та реалізувати, особливо завдяки властивості роздільності, вони не завжди ефективно усувають шум, особливо імпульсивний шум (наприклад, «сіль і перець»). Ці обмеження спонукали до створення нелінійних фільтрів. Такі фільтри, роботу яких не можна визначити як продукт згортки. Прикладом нелінійного фільтра є медіанний фільтр , принцип якого близький до фільтра усереднення: значення кожного пікселя замінюється медіаною (а не середнім) його околиці.

Даний передік основних алгоритмів та технологій які використовують для покращення якості вхідних зображень після етапу їх квантування. Дані процеси не додають на зображення додаткову інформацію, скоріш навпаки, вони дозволяють видалити інформацію яка не є характерною для деякої області на вхідному зображенні. Проте, в деяких задачах, навпаки виникає необхідність пошуку елементів які важко виділити за допомогою простого відуального перегляду, наприклад вони мінімально відрізняють від своїх сусідів або мають

невеликий розмір. В такому випадку необхідно застосовувати алгоритми не для вдалення нетипових пікселів, а навпаки надавати їм більшої ваги.

Результат обробки цифрових зображень на даному етапі значно впливає на якість підсумкового аналізу, тому даний етап є важливий для систем з елементами обрбки та аналізу цифрових зображень.

2.2 Алгоритми сегментації цифрових кольорових зображень

Сегментація зображення – це операція обробки зображення, яка складається з виявлення та групування пікселів відповідно до критеріїв, зокрема інтенсивності або простору, таким чином зображення складається з однакових областей. Сегментація може, наприклад, показувати об’єкти, чітко відрізняючи їх від фону. У випадках, коли критерії поділяють пікселі на два набори, обробкою є бінаризація.

Сегментація зображення – це критично важливий процес у комп’ютерному зорі та обробці зображень, який передбачає поділ зображення на кілька сегментів або областей, часто для спрощення або зміни подання зображення на щось більш значуще та легше для аналізу. Метою сегментації є визначення місцезнаходження об’єктів і меж (ліній, кривих тощо) на зображеннях. Точніше, сегментація зображення призначає мітку кожному пікселю зображення таким чином, що пікселі з однаковою міткою мають певні характеристики.

Алгоритми написані як заміна знанням високого рівня, які люди використовують для ідентифікації об’єктів і структур. Алгоритми високорівневої сегментації (кожна область є семантичним об'єктом) знаходяться в стадії розробки в різних областях, пов'язаних з обробкою зображень.

Сегментація є важливим кроком в обробці зображень. Численні методи сегментації поділяються на три основні класи (рисунок 2.5):

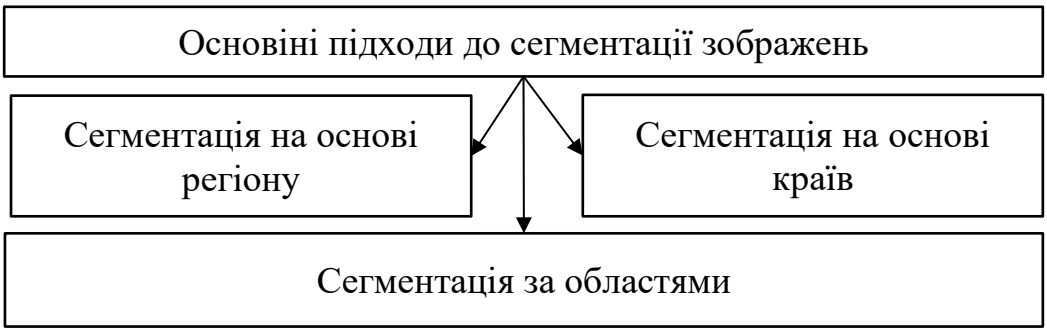

Рисунок 2.5 – Технології сегментації цифрових зображень

Сегментація на основі регіону. До даної групи відносяться: зростання регіону (egion-growing), декомпозиція/злиття (split and merge).

Сегментація на основі країв. Сегментація на основі класифікації або порогового значення пікселів відповідно до їх інтенсивності (classification або thresholding).

Сегментація за областями. Сегментація за областями полягає в пошуку територій з однаковими характеристиками, дотримуючись різних методів визначення територій (ідентифікація та визначення периметра). Методи, що належать до цього сімейства, безпосередньо маніпулюють областями, змінюючи їх ітеративно, поки очікувані умови не будуть задоволені. Або вони починаються з першого розділу зображення, який потім змінюється шляхом поділу або групування областей, і тоді говоримо про методи декомпозиції/злиття (або розділення та злиття англійською); або вони починаються з кількох регіонів, які ростуть шляхом об’єднання, доки не буде охоплено все зображення, і тоді говоримо про методи зростання регіонів. Також існують методи, засновані на спільному статистичному моделюванні закономірностей регіонів і рівнів сірого в кожному регіоні.

Алгоритми декомпозиції/злиття використовують специфічні характеристики кожної області (поверхня, інтенсивність світла, колориметрія, текстура тощо). При аналізі шукаємо пари регіонів-кандидатів на об’єднання та оцінюємо їх відповідно до впливу, який це злиття матиме на загальний вигляд зображення. Потім об’єднуємо пари областей з найкращим рейтингом і

повторюємо, доки характеристики зображення не відповідатимуть попередньо визначеній умові: кількість областей, яскравість, контраст або загальна текстура, або доки найкращі оцінки, призначені парам областей, більше не досягнуть певний поріг (в останньому випадку використовуємо алгоритм з функціональною мінімізацією).

Алгоритми збільшення регіону починаються з першого набору регіонів, який може бути обчислений автоматично (наприклад, мінімуми зображення) або наданий користувачем в інтерактивному режимі. Потім регіони збільшуються шляхом включення найбільш схожих пікселів відповідно до певного критерію, наприклад різниці між рівнем сірого пікселя, що розглядається, і середнім рівнем сірого регіону. До цієї категорії відносяться алгоритми сегментації водоподілу, розроблені в рамках математичної морфології.

Метод водоподілу заснований на аналогії із природними процесами, де вода стікає з вершин до западин, утворюючи басейни або водозбірні області. Застосовуючи цю аналогію до зображень, ідея полягає у поділі зображення на регіони, які представляють басейни. Для цього використовується градієнт зображення, що відображає зміни інтенсивностей пікселів. Градієнт зображення можна уявити як топографічну карту, де вершини представляють високі градієнти (сильні зміни інтенсивності), а долини – низькі градієнти.

Процедура методу водоподілу включає такі основні етапи:

- Обчислення градієнта зображення, що створює карту градієнтів, яка відображає зміни інтенсивності пікселів на зображенні.
- Маркування регіонів. На даному етапі задаються початкові маркери для різних регіонів. Це можуть бути локальні мінімуми градієнта або інші попередньо визначені точки.
- Заливка водою. Припускаємо, що вода починає заливати зображення з маркованих точок, заповнюючи басейни до зустрічі з межами інших басейнів. Процес продовжується, поки вода не заповнить усі басейни.

Для оцінки реузльтатів робти даного алгоритму проаналізуємо наведені на рисунку 2.6 приклади роботи даного алгоритму.

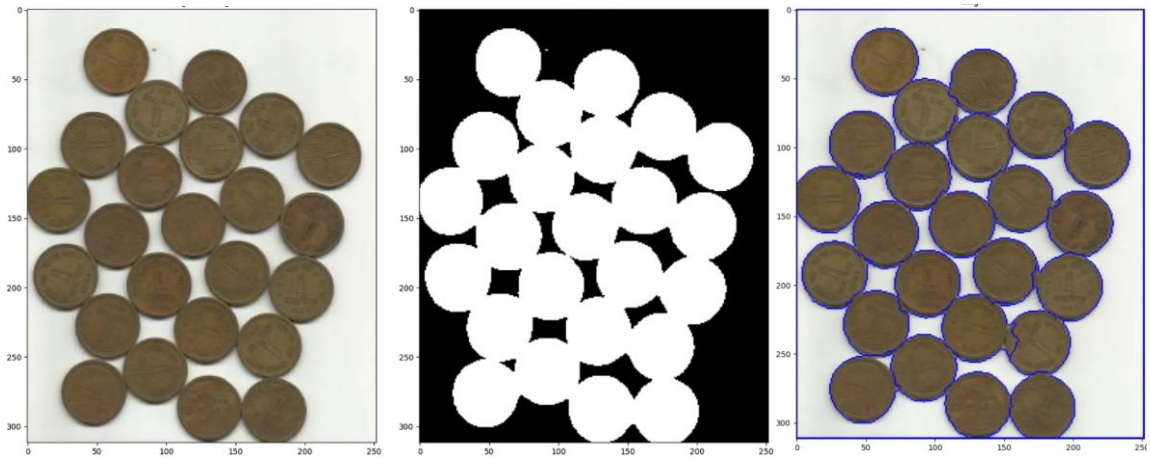

Рисунок 2.6 – Приклад сегментації зображення на основі методу водоподілу

Метод водоподілу має низку значних переваг. По-перше, він є інтуїтивно зрозумілим і базується на природних процесах. Окрім того, метод забезпечує точне виділення контурів об'єктів, що особливо корисно для медичних і біомедичних зображень, де висока точність сегментації є критично важливою.

Однак, метод має і свої недоліки. Основним обмеженням є його чутливість до шуму і дрібних деталей на зображенні, що може призвести до надмірної сегментації – утворення великої кількості малих сегментів. Це явище називається "пересегментацією". Для подолання цієї проблеми часто використовуються попередні кроки згладжування зображення або застосування маркерів для контролю процесу сегментації.

Для поліпшення результатів сегментації за допомогою методу водоподілу було розроблено кілька вдосконалень. Одним із підходів є використання попередньої фільтрації зображень для зменшення шуму, таких як гауссове розмивання або середнє фільтрування. Іншим підходом є введення маркерів для керованого водоподілу, де маркери визначаються заздалегідь на основі аналізу зображення або ручного втручання. Цей метод, відомий як маркований водоподіл, допомагає уникнути пересегментації та поліпшити точність сегментації.

Алгоритми, засновані на спільному статистичному моделюванні регіонів і рівнів сірого, зокрема ті, що спираються на приховані поля Маркова, базуються

на мінімізації функції ймовірності (або енергії). Ця функція одночасно враховує ймовірність приналежності пікселя до регіону з урахуванням його рівня сірого та регіони, до яких належать сусідні пікселі. Ця функція забезпечує компроміс між точністю початкового зображення та регулярністю сегментованих областей.

Сегментація контурів використовує помітні переходи між двома сусідніми областями. Старіші методи використовують оператори обробки зображень, такі як фільтр Canny, щоб виділити пікселі, які належать до краю. Метод Canny edge detection складається з кількох етапів, які забезпечують високу точність виявлення країв і зменшують рівень шуму на зображенні. Першим етапом є згладжування зображення за допомогою гауссового фільтра, що дозволяє зменшити вплив шуму і поліпшити якість виявлення країв. Після цього обчислюється градієнт зображення, що включає визначення напрямку і величини градієнта для кожного пікселя.Далі проводиться не максимальне пригнічення, яке зменшує товщину країв до одного пікселя, видаляючи незначущі пікселі. Завершальним етапом є подвійний поріг і трасування країв за гістерезисом, що дозволяє зберегти лише значущі краї, відкидаючи слабкі краї, що не з'єднані з сильними.

Метод Canny забезпечує високу точність і чіткість виявлення країв завдяки використанню гауссового згладжування та не максимального пригнічення. Окрім того він є стійким до шуму, що робить його особливо корисним для обробки зображень з високим рівнем шуму. Крім того, він дозволяє контролювати кількість виявлених країв за допомогою подвійного порогу, що забезпечує гнучкість у налаштуванні алгоритму для різних застосувань. Проте, основним обмеженням є його чутливість до параметрів, таких як розмір гауссового фільтра і значення порогів. Невдалий вибір параметрів може призвести до недостатнього або надмірного виявлення країв. Крім того, метод Canny не завжди ефективний для виявлення країв у зображеннях з дуже різноманітними текстурами, де може виникати проблема з розділенням значущих і незначущих країв.

Також можна використовувати моделі, що деформуються, за допомогою параметричних кривих (крива Без'є, сплайн тощо) або багатокутників (наприклад, бульбашковий алгоритм). Щоб почати процес, шукаємо помітні точки на зображенні, наприклад точки на перетині принаймні трьох сегментів. Такі точки називаються насінням.

Основний інтерес методів сегментації з використанням межового підходу полягає в тому, щоб мінімізувати кількість операцій, необхідних у разі ітерації процесу на серіях зображень, які мало відрізняються одне від одного (зокрема, у випадку відеозображень). Дійсно, коли контури областей знайдено на першому зображенні, застосування моделі, що деформується, до наступного зображення ефективніше, ніж перерахунок усього, якщо різниця між зображеннями незначна.

Сегментація за класифікацією або пороговим значенням. В даному підході починаємо із співвідношення, яке підтримує кожен піксель окремо, з інформацією, обчисленою для всього зображення, такою як середнє значення рівнів сірого всіх пікселів або медіана, що дає змогу побудувати *n* класів інтенсивності. Коли класи визначаються вибором порогу, то говоримо про порогове значення. Пікселі, що належать до одного класу та є спорідненими, утворюють області зображення.

Так званий метод Оцу (або мінімізація внутрішньокласового варіанту) є класичним методом знаходження порогу за зображенням у градаціях сірого. Метод K-means дозволяє розділити пікселі зображення на кілька класів інтенсивності; його також можна використовувати на кольорових зображеннях і на більших наборах.

Метод K-means є алгоритмом кластеризації, який використовує визначене число кластерів K для поділу даних на групи. В контексті сегментації зображень, цей метод використовує інтенсивності пікселів або кольорову інформацію для групування пікселів. Алгоритм K-means включає такі основні етапи:

- Вибір числа кластерів K.
- Ініціалізація центроїдів кластерів випадковим чином.

- Призначення кожного пікселя зображення до найближчого центроїду кластеру.
- Перерахунок центроїдів як середнього значення всіх пікселів у кластері.
- Повторення кроків 3-4 до конвергенції, тобто до моменту, коли зміни в положенні центроїдів стають незначними.

Однією з основних переваг K-means є його простота та швидкість виконання. Алгоритм легко реалізується та добре працює на великих наборах даних. Крім того, K-means є детерміністичним, що означає, що при однакових початкових умовах він завжди буде давати однаковий результат. Однак, метод має і суттєві обмеження. По-перше, результат сегментації сильно залежить від вибору початкових центроїдів кластерів. Невдалі ініціалізації можуть призвести до локальних мінімумів та некоректної сегментації. По-друге, K-means вимагає знання числа кластерів *K* наперед, що не завжди є можливим. Третє обмеження полягає в тому, що K-means може бути неефективним при обробці зображень з неоднорідними структурами або з високим рівнем шуму.

Для подолання обмежень стандартного методу K-means було розроблено кілька вдосконалень. Одним із таких підходів є використання алгоритму K-means++ для ініціалізації центроїдів, що дозволяє уникнути проблем з локальними мінімумами. Іншим підходом є алгоритм Fuzzy C-means, який дозволяє кожному пікселю належати до кількох кластерів із певною ймовірністю, що дозволяє поліпшити якість сегментації в разі нечітких меж між регіонами.

Метод K-means знайшов широке застосування в різних областях, таких як медична діагностика, обробка супутникових зображень, автоматичне розпізнавання об'єктів та багато інших. Наприклад, у медичній діагностиці K-means використовується для сегментації томографічних зображень, що дозволяє виділяти аномальні регіони, такі як пухлини. В обробці супутникових зображень цей метод допомагає виявляти різні типи покриття землі, що є важливим для екологічного моніторингу та сільського господарства. Сегментація цифрових

зображень за допомогою методу K-means є ефективним та широко використовуваним підходом у комп'ютерному зорі та обробці зображень. Незважаючи на певні обмеження, такі як залежність від початкових умов та необхідність знання числа кластерів, метод K-means залишається популярним завдяки своїй простоті, швидкості та можливості вдосконалення через різні модифікації. Завдяки своєму широкому застосуванню в різних галузях, цей метод відіграє важливу роль у сучасних технологіях аналізу зображень.

Сегментація за підходом машинного навчання. Методи, засновані на нейронних мережах, внесли значний внесок у сегментацію зображень. Згорточні нейронні мережі (CNN) стали революційними у цьому процесі завдяки їх здатності навчатися складним характеристикам необроблених даних зображень. Використовуючи численні шари згорткових фільтрів для виявлення об'єктів, які потім проходять через пов'язані шари для класифікації, CNN забезпечують високу точність і здатність виконувати складні завдання сегментації. Однак, ці мережі вимагають великих обсягів позначених даних і значних обчислювальних ресурсів. Повністю згорточні мережі (FCN) є розширенням CNN, призначеним спеціально для семантичної сегментації. Вони замінюють повністю пов'язані шари на згорткові для створення щільних прогнозів, забезпечуючи сегментацію на рівні пікселів і можливість обробки вхідних даних довільних розмірів, але також потребують значних обчислювальних ресурсів і великих обсягів позначених даних. U-Net є спеціалізованим типом CNN, розробленим для сегментації біомедичних зображень, але широко застосовується і в інших сферах. Ця модель поєднує скорочувальний шлях (кодер) і розширений шлях (декодер) із пропускними з'єднаннями для захоплення контексту і точної локалізації. U-Net особливо ефективна для сегментації медичних зображень і роботи з невеликими наборами даних, однак, також потребує інтенсивних обчислень і може бути неефективною при роботі з дуже малими наборами даних.

Теорія графів представляє зображення як граф, де вершини відповідають пікселям або регіонам, а ребра визначають зв'язки між ними. Вага ребра зазвичай визначається за допомогою метрики подібності, що враховує інтенсивності

пікселів або інші атрибути, такі як колір або текстура. Метою сегментації є розбиття графа на підграфи, які відповідають значущим об'єктам або областям на зображенні. Одним із популярних методів сегментації на основі теорії графів є метод мінімального розрізу (min-cut). Цей метод полягає у знаходженні такого розрізу графа, який мінімізує сумарну вагу розрізаних ребер. У контексті сегментації зображень, це відповідає знаходженню меж між сегментами, які мають мінімальну подібність між собою. Процедура методу мінімального розрізу включає побудову графа з зображення, де вершини відповідають пікселям, а ребра зважені за допомогою метрики подібності. Потім використовується алгоритм для знаходження мінімального розрізу, такий як алгоритм Форда-Фалкерсона або алгоритм Дініца, щоб розділити граф на дві частини. Цей процес можна повторювати для подальшого розбиття на підграфи, доки не будуть досягнуті бажані сегменти. Метод нормалізованого розрізу (normalized cut) є вдосконаленням методу мінімального розрізу, який враховує не лише вагу розрізаних ребер, а й внутрішню подібність сегментів. Це дозволяє уникнути переваги малих сегментів, яка часто виникає при використанні мінімального розрізу. Процедура нормалізованого розрізу включає обчислення міри нормалізованого розрізу, яка враховує відношення між вагою розрізаних ребер та сумарною вагою ребер у кожному сегменті. Використовуються алгоритми, такі як спектральна кластеризація, для знаходження оптимального розрізу, що мінімізує цю міру. Це дозволяє отримувати більш збалансовані та значущі сегменти на зображенні.

Метод рандомізованих прогулянок (random walks) є ще одним підходом до сегментації зображень на основі теорії графів. Цей метод використовує моделювання випадкових прогулянок на графі для визначення ймовірності того, що піксель належить до певного сегмента. Процедура методу рандомізованих прогулянок включає побудову графа з зображення та визначення початкових маркерів для кожного сегмента. Потім моделюються випадкові прогулянки, які починаються з цих маркерів і продовжуються до досягнення ймовірності зупинки. Пікселі призначаються до сегмента з найвищою ймовірністю

досягнення від відповідного маркера, що забезпечує м'яке і адаптивне розбиття зображення на сегменти.

Метод графових зрізів (graph cuts) є узагальненням енергетичних методів і включає використання різних варіантів енергетичних функцій для розв'язання задач сегментації. Цей метод забезпечує гнучкість у виборі енергетичної функції, що дозволяє адаптувати підхід до різних типів зображень та задач. Процедура методу графових зрізів включає формулювання енергетичної функції, яка складається з двох частин: терміну даних, який враховує подібність між пікселями, і терміну згладжування, який мінімізує довжину меж між сегментами. Використовуються алгоритми для знаходження мінімального розрізу, що забезпечує оптимальне розбиття зображення на сегменти з урахуванням якості та гладкості меж.

Сегментація зображень - це різноманітне поле з численними алгоритмами, призначеними для вирішення різноманітних завдань і додатків. Кожен алгоритм має свої сильні та слабкі сторони, тому важливо вибрати правильний метод на основі конкретних вимог поставленого завдання. Незалежно від того, чи йдеться про простоту порогових методів, точність методів на основі ребер, інтуїтивність методів на основі регіонів, адаптивність методів кластеризації, потужність нейронних мереж чи оптимальність методів на основі графів, існує техніка сегментації, яка підходить для будь-яких потреб в обробці зображень і комп'ютерному зорі.

2.3 Алгоритм сегментації на основі теорії графів

Провівши порівняльний аналіз відомих алгоритмів сегментації та враховуючи особливості характеристик цифрових зображень було запропоновано алгоритм сегментації на основі використання елементів теорії графів. В даному випадку зображення буде представлятись як набір вершин

графа, яким відповідатимуть точки, а ребра будуть вказувати на сусідство окремих точок. Вага кожного ребра буде визначатись як різниця значень кольору між відповідними точками. Відповідно чим менша вага ребра тим більш подібними є сусідні точки і при розрізання графу дані точки мають більшу ймовірність потрапити в одну область. Такий підхід дозволяє провести розділення вхідного цифрового зображення на однорідні області з подальшим їх об'єднанням. Запропонований підхід в загальному можна описати за допомого наступних етапів. На першому кроці будується граф для представлення зображення, де кожен піксель є вузлом. Отже, необхідно визначити ваги ребер між вузлами. Дана оцінка базується на подібності чи відмінності між пікселями, як-от різниця в кольорі чи значеннях інтенсивності. Потім розбиваємо граф на непересічні області, використовуючи алгоритм поділу графа. Метою цього алгоритму є мінімізація ваги ребер між сегментами. На останньому етапі вдосконалюємо сегменти, об'єднуючи або розділяючи їх на основі кількох критеріїв, таких як розмір, форма чи текстура. На рисунку 2.7 проілюстровано результати основних етапів обробки цифрового зображення.

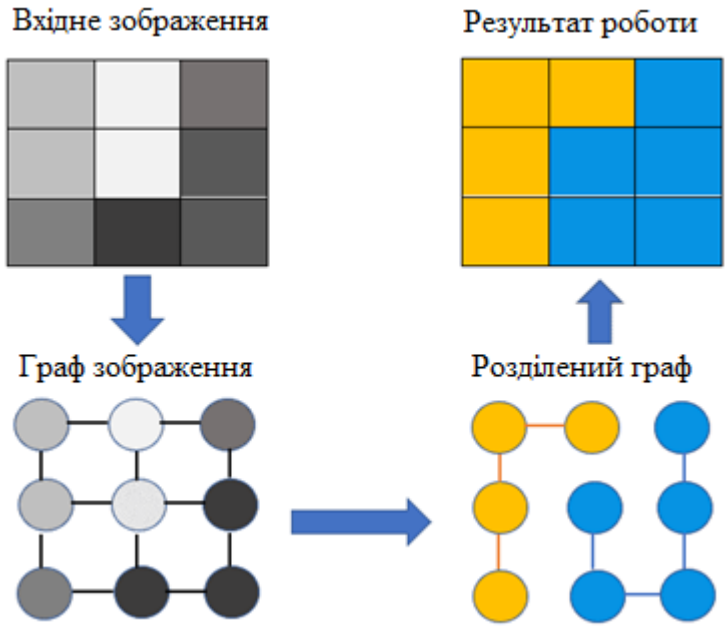

Рисунок 2.7 – Схема сегментації цифрового зображежння на основі теорії графів

Більш детально послідовність кроків алгоритму сегментації цифрових кольорових зображень наведено у вигляді блок-схеми (рисунок 2.8).

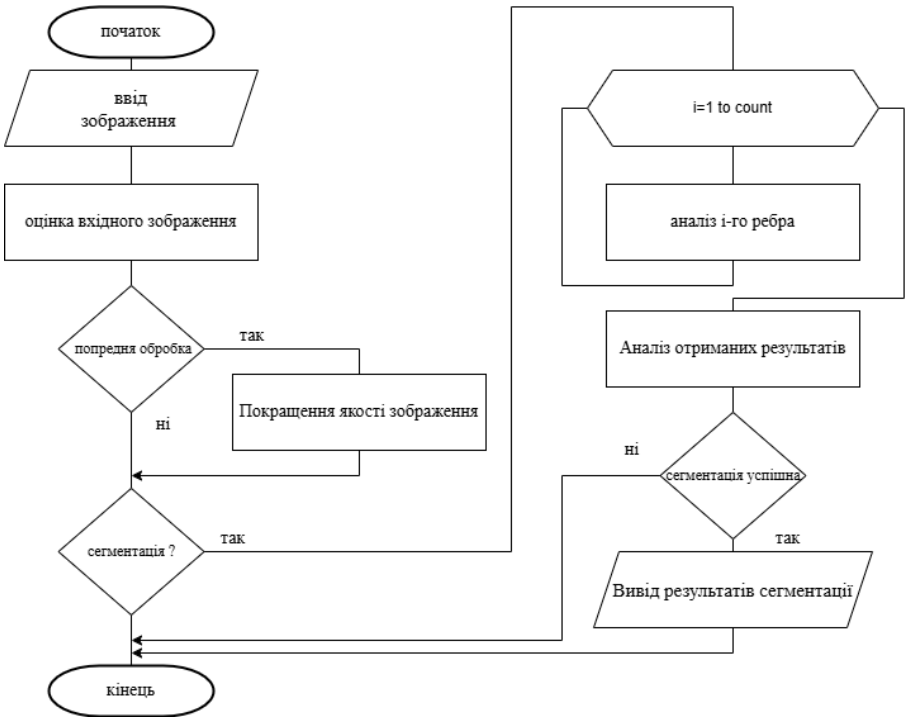

Рисунок 2.8 – Блок-схема алгоритму сегментації кольорових зображень на основі теорії графів

Даний алгоитм дозволяє проводити сегментацію цифрових зображень у напівавтоматичному резимі, що значно спрощує його використання у програмних системах обробки цифрових зображень. Окрім того в ньому передбачено етап попередньої обробки, що дозволить покращити рівень вхідного зображення та зменшити кількість шумів. Етап попередньої обробки вимагає додаткових часових затримок, проте це може бути компенсовано за рахунок скорочення ачсу на етапі сегментації. Послідовність кроків запропонованого алгоритму:

1) Отримання вхідного цифрового зображення *I*.

2) Проведення попередньої оцінки вхідного зображення з метою виявлення рівня зашумленості (велика кількість пар сусідніх пікселів, що значно відрізняються за кольором).

3) Якщо зображення малозашуслене, то переходимо на крок 4 інакше проодимо фільтрацію вхідного зображення з метою згладжування різких перепадів між сусідніми пікселями шляхом усереднення значення пікселів в околі.

4) Формування представлення вхідного зображення *I* у вигляді графа *G* (матриця суміжностей).

5) Визначення ваг для усіх ребер графа *G* шляхом обчислення відстні між кольорами пікселів в просторі RGB.

6) Видалення з графа ребер які мають значення більше за середні для цілого графа.

7) Обчислення характеристик утворених підграфів шляхом підрахунку кількості пікселів, які належать до кожної з підобластей.

8) Групування малих підграфів в більші.

9) Вивід результатів сегментації зображення.

Запропонований алгоритм може опрацьовувати зображення різної складності, дозволяє проводити процедури покращення якості вхідного зображення, працює з зображеннями різних типів та форматів. Серед недоліків можна виділити залежність результатів роботи від рівня зашумленості .

2.4 Висновки до розділу

Проведено аналіз відомих методів та алгоритмів сегментації цифрових кольорових зображень, на основі базових принципів розподулі пікселів по одноріжнихм областям, що дало можливість виділити групу алгоритмів сегментації на основі використання елементів теорії графів.

Запропоновано алгоритм сегментації цифрових кольорових зображень на основі використання елементів теорії графів, що дозволило зменшити вплив попередньої обробки на результати сегментації зображень.

3 ПРОГРАМНИЙ ДОДАТОК ОБРОБКИ ТА ОПИСУ ЦИФРОВИХ КОЛЬОРОВИХ ЗОБРАЖЕНЬ

3.1 Структура реалізованого програмного додатку

Для більшості гравців у секторі аналіз цифрових зображень складається з розрізання файлу на цікаві об’єкти (сегментація), обчислення його «цифрового підпису» відповідно до серії дескрипторів (форми, текстури, кольору тощо). індексувати файл і, можливо, здійснити розпізнавання зображень за допомогою функцій для вивчення категорій об'єктів, інтеграції варіацій зовнішнього вигляду та збагачення бази даних. Потім можна запустити пошук шляхом порівняння цифрових підписів в Інтернеті або в базі даних, тому ця технологія надає дві основні групи послуг: керування вмістом (пошук), найчастіше в індексованих базах даних (комерційні каталоги, банки зображень тощо).), що допомагає збільшити ймовірність покупки, а також можна здійснювати модерацію вмісту (фільтрування) для неконтрольованих потоків зображень (сайти спільнот, програмне забезпечення безпеки тощо). під час завантаження веб-сторінок або в службах обміну повідомленнями.

Програмні системи для аналізу цифрових зображень можна класифікувати відповідно до етапів процесу аналізу, від отримання зображення до інтерпретації результатів. Цей процес може включати кілька субдоменів, таких як попередня обробка, сегментація, вилучення ознак і сам аналіз. Попередня обробка, наприклад, складається з виправлення дефектів зображення, покращення якості зображення або навіть зменшення шуму. Сегментація етап поділу зображення на різні регіони або важливі об’єкти, що полегшує аналіз. Вилучення ознак дозволяє нам отримати корисні дескриптори, які потім будуть використовуватися для таких завдань, як розпізнавання образів або класифікація. Нарешті, аналіз зображень передбачає застосування алгоритмів для інтерпретації результатів, наприклад, виявлення об’єктів або розпізнавання образів.

Розробка програмних систем для аналізу цифрових зображень спирається на комбінацію методів з кількох дисциплін. Основні методи, які використовуються в цій галузі, включають трансформацію зображення, алгоритми фільтрації, підходи на основі нейронної мережі та статистичні методи. Наприклад, перетворення Фур'є часто використовується для обробки зображень у частотній області, дозволяючи маніпулювати певними характеристиками, такими як текстура або певні частоти, які не будуть помітні в просторовій області. Алгоритми фільтрації, такі як згладжування або фільтри виявлення країв, допомагають покращити якість зображення та підготувати дані для подальших кроків у процесі аналізу. Нейронні мережі, а точніше згорточні нейронні мережі, зробили революцію в аналізі зображень, дозволивши класифікувати та розпізнавати об'єкти набагато точніше та швидше, ніж раніше. Ці мережі здатні вивчати складні функції з великої кількості позначених даних, що дозволяє їм вирішувати такі проблеми, як розпізнавання обличчя, виявлення захворювань на рентгенівських знімках і автономне водіння.

Одним із перших кроків у розробці програми є моделювання архітектури програмного забезпечення. Архітектура визначає фундаментальну структуру програми, включаючи розподіл компонентів і їх взаємодію. Традиційно архітектурні шаблони в основному базувалися на монолітному підході, коли додаток будувався як нерозривне ціле. Однак зі збільшенням складності додатків і необхідністю зробити додатки більш модульними та гнучкими, архітектура мікросервісів набула популярності. Архітектура мікросервісів дозволяє розкласти програму на кілька незалежних служб, кожна з яких відповідає за певну функцію. Такий підхід полегшує розробку та розгортання масштабованих програм, оскільки кожен мікросервіс можна оновлювати незалежно, не впливаючи на всю систему. Однак архітектура мікросервісів створює проблеми з точки зору керування зв'язками між службами, забезпечення узгодженості даних і керування розподіленими розгортаннями.

При розробці програмних додатків технології програмування також відіграють вирішальну роль. Вибір мови програмування зазвичай залежить від

типу програми, яка розробляється, і вимог до продуктивності, безпеки та сумісності. Наприклад, для додатків, які потребують інтенсивної обробки даних або наукових обчислень, можна віддати перевагу такій мові як java, завдяки її ефективності та спеціалізованим бібліотекам. Ще один важливий вимір розробки програмного забезпечення полягає у використанні фреймворків. Фреймворки – це набори інструментів і бібліотек, які забезпечують базову структуру для розробки програм. Вони допомагають оптимізувати процес розробки, пропонуючи заздалегідь розроблені рішення для повторюваних проблем, таких як керування базами даних, автентифікація користувачів або маршрутизація запитів. Наприклад, у сфері веб-розробки такі фреймворки, як React або Angular для інтерфейсу та Django або Ruby on Rails для серверу, широко використовуються для прискорення розробки додатків. Ці інструменти дозволяють більше зосередитися на бізнес-логіці та конкретній функціональності програми, а не на переписуванні основної функціональності з кожним проектом.

Однак, незважаючи на численні переваги сучасних технологій розробки програмного забезпечення, деякі проблеми залишаються. Одна з головних проблем полягає в управлінні зростаючою складністю програм. Завдяки інтеграції багатьох служб, мікросервісів і технологій підтримувати, тестувати та розгортати програми стає все важче. Команди розробників повинні постійно розвиватися, щоб запроваджувати нові практики та інструменти для керування цією складністю. Крім того, інтеграція нових технологій, таких як штучний інтелект і машинне навчання, вводить нові проблеми в дизайні та продуктивності, що вимагає спеціальних знань для ефективного використання цих інструментів.

Проведений аналізу програмний засобів обрбки зображень та цифрових і бліотек функцій дозволив видітити основні архітектурні рішення які використовуються при програмній реалізації систем даного типу. Для програмної реалізації розролений алгоритмів та для вирішення задач, що були поставлені в кваліфікаційні роботі було розроблену молульну архітектура

програмного додатку, обрбки растрових кольорових зображень. Внутрішня структура програмного додатку наведена на рисунку 3.1.

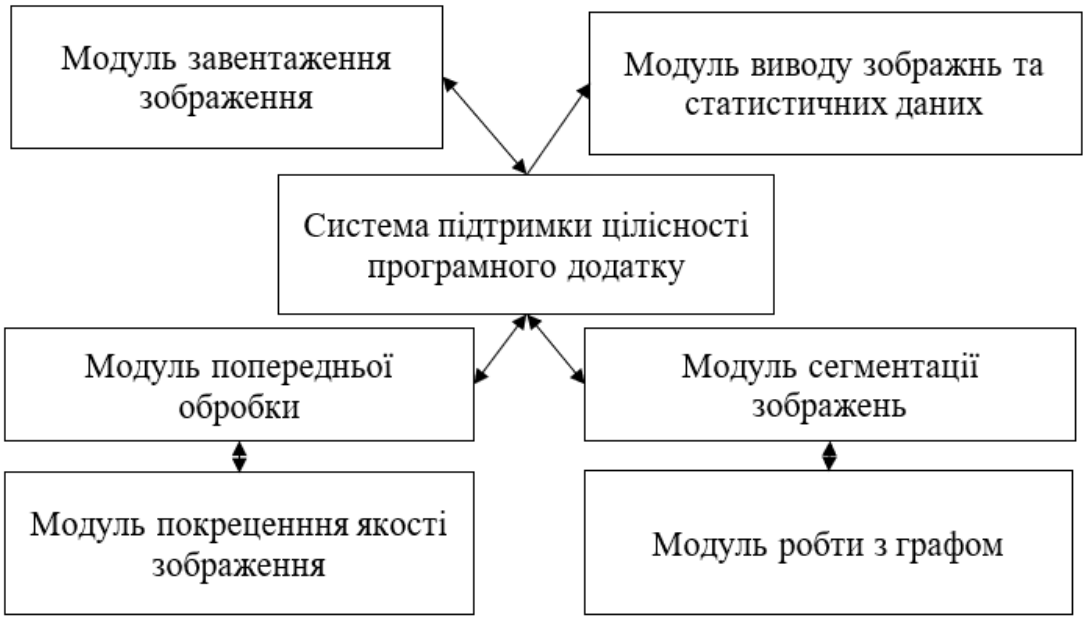

Рисунок 3.1 – Узагальнена структура програмного додатку обрбки цифрових зображень з використанням елементів теорії графів

Запропонована структура програмної системи, спрямованої на підтримку цілісності програмного додатку. Вона складається з декількох модулів, кожен із яких виконує специфічну функцію. Даний набір ілюструє тільки основні архітектурні рішення, що були запропоновані в проуесі проектування. Кожен модуль є набором множини функцій, які дозволяють організовувати виконання поставлених завдань не залежно від типу вхідного зображення. В структуру програного додатку входять настіпні модулі:

Модуль завантаження зображення Цей модуль відповідає за отримання вхідних даних у вигляді зображень. Він забезпечує зчитування зображень із зовнішніх джерел, таких як файли, бази даних чи мережеві ресурси. Завантаження може відуватись як з деякого сховища даних так і отримувати з фотофіксуючої апаратури в процесі робти програми. Моливість отримання зображень в режимі реального часу підвищує функціональні можливості програмного забезпечення та дозволяє більш ефективно використовувати

функціональні можливості програмної реалізації. Ця частина системи необхідна для подальшої обробки інформації.

Модуль попередньої обробки виконує попередню обробку завантажених зображень. Це може включати нормалізацію розмірів, фільтрацію шумів, усунення артефактів і виконання базових перетворень, таких як зміна формату або масштабування. Даний модуль доданий в архітектуру програмної системи з метою проведення оцінки вхідного зображення та встановлення параметрів робти системи в цілому. Даний модуль пропонує користувачу набір стандартних параметрів, які мжуть бути відкоректовані в процесі роти як в ручному так і в автоматичному режимах. Попередня обробка забезпечує стандартизацію даних перед їх аналізом.

Модуль покращення якості зображення реалізує функції поліпшення якості вхідних зображень. Модуль може включати алгоритми підвищення чіткості, корекції яскравості та контрасту, а також методи відновлення зображень у разі їх пошкодження. Даний модуль є допоміжним, а його функції будуть використовуватись у випадках, якщо вхідне зображення буде містити велику кількість шумів. Рівень засумленості буде визначати шляхом обчислення відносноїо кількісті різких перепадів рівня яскравості між сусідніми точками. Якщо даний показник буде високим, то на зображені присутня велика кількість дрібних елементів або велика кількість сторонніх шусів. Це дозволяє підвищити точність подальших обчислювальних процедур.

Модуль сегментації зображень відповідає за виділення окремих об'єктів або областей на зображенні. В даному модулі буде реалізовано функції розбиття растрового зображення на підобласті, з подоальшим їх маркування. В модулі буде реалізовано алгоритм сегментації на основі використання елементів теорії графів. Проте додатково було додато алгоритми сегментації які реалізовані в та інгегровані в бібліотеку OpenCV. Даний підхід був реалізований для підвищення функціональних можливостей розроленого програмного додатку. В загальному, модуль містить функції класифікацію пікселів, створення масок сегментації або розподіл зображення на логічні частини для подальшого аналізу.

Модуль роботи з графом. Даний елемент архітектури програмного додатку дозволяє використовувати елементи теорії графів для проведеня допоміжних маніпуляцій з растровим зображенням, що представлене у вигляді графів. Він може забезпечувати створення, аналіз та обробку графових структур, які використовуються для моделювання складних взаємозв'язків між об'єктами на зображеннях. Даний модуль включає в себе дві основні функції, а саме створення опису цифрового зображення у вигляді звізного графа та розділення головного графа на множину підграфів. Опис цифрового зображення у вигляді графа здійснюється на основі обчислення відмінностей кольорів між сусідніми пікселіми. В даному випадку значення кольору берельсяся за всіма трьома кольоровими каналами. Такий підхід дозволяє визначити відмінність між двома кольорми як відстань між двома точками в тьохвимірному просторі. Така мірі різниці дозволяє визначати степінь відмінності незважаюючи на значення окремих кольорів в точці на зображенні.

Модуль виводу зображень та статистичних даних. Завершальний етап обробки даних виконує модуль, який відповідає за виведення оброблених зображень, а також представлення статистичних даних, отриманих у процесі аналізу. Він може генерувати звіти, візуалізації або метадані для користувачів або інших систем.

Система підтримки цілісності програмного додатку Уся описана функціональність інтегрована в рамках системи підтримки цілісності, яка забезпечує узгоджену роботу всіх модулів, а також гарантує точність, надійність і стабільність виконання основних процесів. Це ядро системи виконує управління потоком даних між модулями та контроль правильності їхньої роботи.

Модулі взаємодіють між собою за принципом послідовної передачі даних від завантаження зображень до їхнього виводу. Деякі модулі, як-от модулі сегментації зображень і роботи з графом, можуть взаємодіяти безпосередньо, якщо це необхідно для виконання специфічних завдань. Запропонована структура є прикладом добре організованої багаторівневої архітектури, яка

дозволяє виконувати складні завдання обробки зображень ефективно та модульно.

Після етапу розробки архітектури програмного додатку було здійснено процедури моделювання для того, щоб переконатись у правильності обраної стратегії та напрямків подальшої роботи. Моделювання допомагає визначити вимоги, виявити можливі проблеми та відкорегувати процес роботи до початку реалізації програмного забезпечення. Важливість моделювання в проектуванні програмних систем полягає у кількох ключових аспектах. Однією з головних переваг моделювання є можливість зрозуміло визначити вимоги до системи та ефективно комунікувати їх між різними учасниками проекту. Застосування діаграм і моделей дозволяє візуалізувати структуру та поведінку системи, що сприяє кращому розумінню вимог замовником, розробниками та іншими зацікавленими сторонами. Це знижує ймовірність виникнення непорозумінь і помилок на початкових етапах проекту, забезпечуючи більш точне виконання завдань. Проектування програмних систем часто включає роботу з великою кількістю компонентів та взаємодій між ними. Моделювання допомагає структурувати цю складність, розділяючи систему на окремі модулі та визначаючи їх взаємодію. Такі моделі, як діаграми класів, діаграми станів та діаграми послідовностей, надають зрозумілий і структурований підхід до аналізу та проектування системи. Це полегшує виявлення потенційних проблем та їхнє вирішення на ранніх етапах розробки.

Процес модулювання було проведено у декілька етапів. На кожному з них було одсліджено різні елементи запропонованої структури з метою проведення оптимізації та виділення еллментів, що можна виокристати повторно. На першому етапі було досліджено можливості які надає програмна система для користувачів. Зокрема було згруповано та досліджено групи функцій, які можуть застосовувати користувачі на різних етапах під час взаємодії з програмним додатком. Для ілюстрації процесу модулювання було використано діаграму прецедентів UML яку наведено на рисунку 3.2.

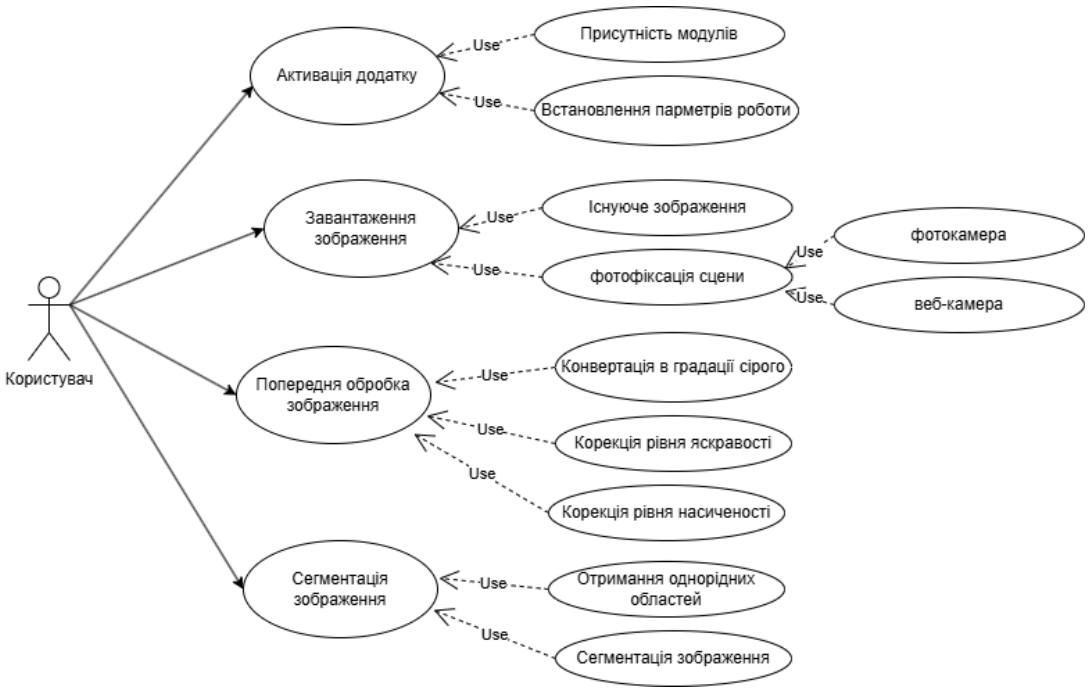

Рисунок 3.2 – Діаграма прецедентів програмної системи

Процес моделювання програних систем заметою виділення основних можливостей по взаємодії користувача з програмою дозволяє більш ефективно спроектувати рівні доступу до різних модулів. Якщо рівень доступу дуде занадто великий то, в кінцевому результаті користувач змушений буде витрачати час на вивчення додаткових операцій по взаємодії з системою. І навпаки, якщо програмна система буде максимально автоматизована в процесі своєї роботи, то зменшується її функціональні можливості по оробці вхідних зображень, оскільки користувач не матиме можливості вносити корекції в процес її роботи. Проведено модулювання показало, що користувачам системи надано доступ до різних елементів робти програмної системи, проте дані функції модуть замінюватись автоматичними рішеннями. Інтеграція автоматичних рішень в робти програмної системи значно підвищує рівень її практичності та функцональності за рахунок зменшення часу на опанування програмним додатком користувачами. З іншої сторони, користувачі з високим рівнем володіння комп’ютерної технікою та знаннями в галузі обробки цифрових зображень можуть вносити корективи в роцес робти програмного додатку, що в кінцевому результаті підвищить рівень точності результатів сегментації.

Для отримання більш цілісного уявлення про внутрішню структуру програмного додатку та процесів обміну даними між окремими модулями було проведено відповідне моделювання. Для відображення результатів процедури моделювання було використано механізм моделі послідовності, яка входить в групу UML. З результати моделювання представлені на рисунку 3.3.

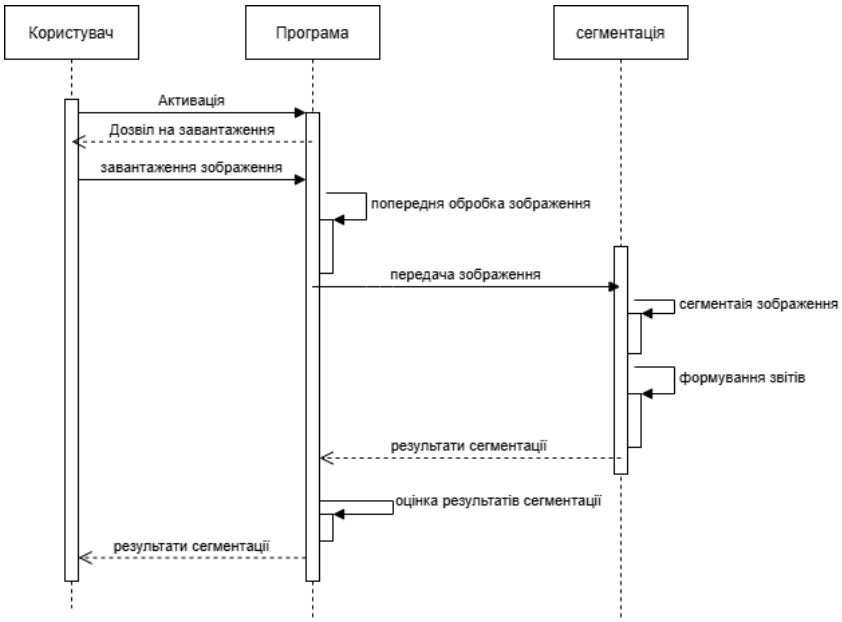

Рисунок 3.3 – Діаграма послідовності програмної системи

Процес моделювання включав дослідження послідовності етапів, які проходить цифрове зображення в процесі оброки та часові затримки, що відбуваються на кожному з етапів. Діаграма послідовності доволяє провести комплексну оцінку з метою виявлення внутрішніх конфліктів та можливих помилкових систуацій. В результаті проведеного моделювання можна зробити висновки, що запропонована архутектура та набір інтерфейсів дозволяє виконувати оставлені завдання без винекнення помилкових ситуацій. Зображення в середині програмного додатку передаються почергово між окремими модулями, окремі функції запускаються послідовно і тільки після закінцення поередньої обробки. Внутрішня система перекодування інформації у зручний формат для обробки різних блоків дозволяє проводити процедуру аналізу без додаткомих помилок.

Для програмної реалізації було розроблено набір класів, які дозволяють зберігати інформацію про цифрове зображення, його параметри. Додатково було спроектовано клас для робти з графами, який містить набір методів для безпосередньох робти з графами тта методи для переконвертування вхідного растрового зображення у представлення за допомогою графа. Дані класи сформували ієрархію класів, що у повній мірі дозволило реалізувати внутрішню архітетуру програмного додатку. Ієрархія запропонованих класів програмного додатку наведена нв рисунку 3.4.

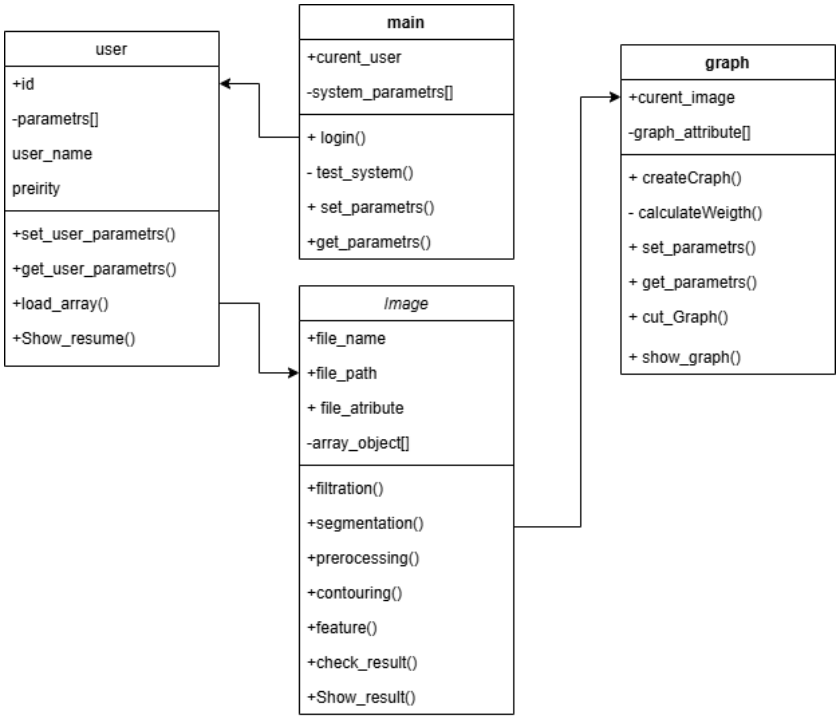

Рисунок 3.4 – Діаграма класів програмної розробки

Запропонований набір класів доповнюється набором стандартних класів які використовуються в мові java та бібліотеці OpenCV для роботи з файловою системою та засобами візуалізації цифрових зображень. Поєднання даних класів дозволяє реалізувати повнихй комплекс задач по отриманні, попередній обробці, аналізу, збереженню цифрових зобсрових зображень в середині реалізуваного програмного рішення. При дослідженні особлива увага була приділена класу для роботи з графами, оскільки він є базовим під час реалізації алгоритму

сегментації. Повнота методів та внутрішніх параметрів у повній мірі дозволила реалізувати розроблений алгорим сегментації. Проте процес поделювання показав деяку надлишковість допоміжних параметрів, які мають невеликий вплив на процс сегментації. Після проведення додаткових досліджень їх було вирішено залишити, оскільки вони займають невеликий об'єм пам'яті, але мжуть бути використаними в подальших модифікаціях програмного забезпечення.

Проектований програмний додаток реалізує можливості обробки цифрових растрових зображень, а отже більшу частину часу, яку користувачі будуть витрачати для обробки даних вони будуть безпосередньо взаємодіяти з активними керуючими елементами програмної розробки. Тому для підвищення рівня зручності користування програмним додаток було розроблено графічний інтерфейс користувача який наведено на рисунку 3.5.

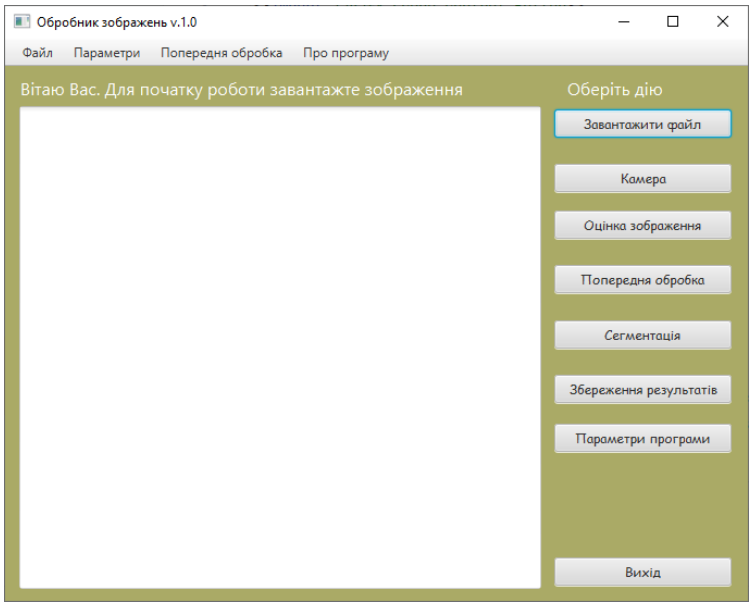

Рисунок 3.5 – Графічне вікно програмної обробки растрових зображень

В процесі проектування графічного інтерфейсу були враховані особливості подулови взаємодії користувачів з програмами даного типу, а також максимально збільшено область для візуалізації результатів робти програмного додатку. Такий підхід дозволив розробити простий, інтуїтивно зрозумілий інтерфейс, який можна швидко опанувати та виконувати посталені задачі.

3.2 Функції програмного модуля обробки растрових зображень

Обробка зображень – це набір операцій, які призводять до модифікації цифрових зображень. Ці зміни, як правило, стосуються кольору пікселів. Для роботи з цими даними використовується кілька методів, метою яких є покращення цих даних для отримання кращої читабельності зображення. Java – це кросплатформна мова програмування, відома багатством своїх API і портативністю. По суті, Java була розроблена для написання динамічних веб-сторінок із інтеграцією аплетів на клієнті, які взаємодіють із сервлетами на серверах. Ця орієнтація на Інтернет змінилася завдяки великій кількості класів Java (3780 класів у Java SE 6). Java також присутня в області комп'ютерної графіки. Саме завдяки її 2D Java API користувачі можуть написати потужне програмне забезпечення для обробки та аналізу зображень. Java 2D – це API, що складається з набору класів, призначених для створення зображень і двовимірних зображень.Це дозволяє: малювати лінії, прямокутники та будь-які інші геометричні фігури; складати форми суцільними кольорами або градієнтами та додавати текстури; додавати текст, призначати йому різні шрифти та контролювати його відтворення; малювати зображення, можливо, виконуючи операції фільтрації.

Щоб працювати із зображеннями в Java 2D, вам потрібно знати два важливі класи API (рисунок 3.6):

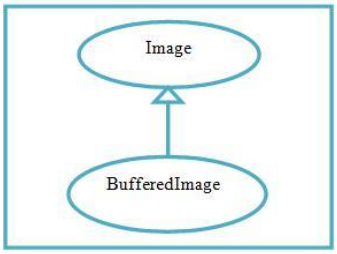

Рисунок 3.6 – Ієрархія класів для роботи з 2D зображеннями

java.awt.Image – це суперклас, наданий у JDK з версії 1.0. Це абстрактний клас, який дозволяє представляти зображення у формі прямокутника з пікселів. Обмеження цього класу полягає в тому, що він не дозволяє доступу до пікселів. Тому він непридатний для обробки зображень напряму.

java.awt.image.BufferedImage додано до JDK, починаючи з версії 2. Він успадковує клас Image і реалізує його інтерфейси, щоб дозволити вам досліджувати внутрішню частину завантажених зображень і працювати безпосередньо з його даними. Таким чином користувачі можуть з об’єкта BufferedImage відновити кольори пікселів і змінити їхні кольори. Як випливає з назви, BufferedImage є буферним зображенням. Цей клас керує зображенням у пам'яті та надає методи для інтерпретації пікселів. Якщо необхідно обробити зображення, то робота проводиться з об’єктами BufferedImage. На рисуноку 3.7 наведено структуру об’єкта BufferedImage складається з двох частин:

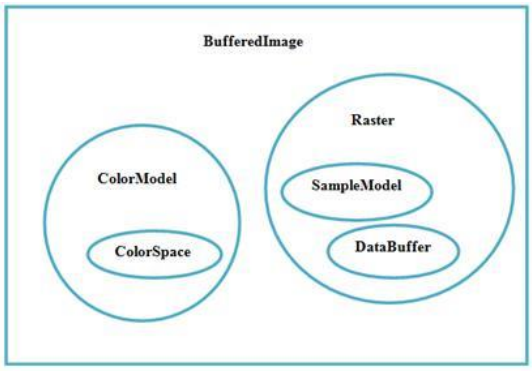

Рисунок 3.7 – Внутрішня структура об’єкта класу BufferedImage

java.awt.image.ColorModel - це клас, що визначає, як інтерпретувати кольори. ColorModel здатний переводити значення даних із растру в об’єкт java.awt.Color.

java.awt.image.Raster містить дані зображення та може представляти їх у вигляді масиву значень пікселів. Крім того, він зберігає дані зображення в пам’яті та надає методи доступу до певних пікселів у зображенні. А також дозволяє редагувати дані зображення та створювати більше його екземплярів.

Реалізоване програмне забезпечення, написане з використанням технічних можливостей мови Java дозволяє застосовувати певні операції до растрових зображень. Серед основний функцій:

- читати файли зображень і виводити їх на екран;
- зберігати зображення після редагування на диск;
- застосувати набір фільтрів (візуалізувати зображення з рівнем сірого або двійковим чином,затемнення кольорів пікселів);
- збільшити/зменшити зображення.

Програма складається з двох класів: Frame та PanDessin. Клас Frame описує головне вікно програми.Конструктор цього класу дозволяє створити три меню:

- меню «Файл» містить підменю «Відкрити», яке дозволяє відкрити файл зображення, що зберігається на диску, і друге підменю «Зберегти», яке дозволяє зберегти зображення на диску після зміни;
- меню «Фільтр» містить п’ять підменю, які представляють основні способи обробки зображень, завантажених із диска;
- Меню «Змінити розмір» містить підменю «Збільшити» та «Зменшити», що дозволяє відповідно, як вказує назва кожного, збільшувати або зменшувати розмір завантаженого зображення.

Клас PanDessin є похідним класом від класу JPanel. Він представляє панель, де відображатимуться зображення, завантажені програмою. Він допускає унікальний атрибут monImage типу BufferedImage, який у будь-який час представляє завантажене зображення, що відображається на панелі. Цей клас представляє ядро програми. Його методи дозволяють змінювати зображення:

```
protected void reduireImage(){}
protected void agrandirImage(){}
protected void imageConvolue(){}
protected void imageEclaircie(){}
protected void imageSombre(){}
protected void imageBinaire(){}
protected void imageEnNiveauGris(){}
protected void modifierImage(){}
protected void ajouterImage(File fichierImage){}
protected BufferedImage getImagePanneau(){}
```

Операції обробки запускаються після натискання відповідної команди в рядку меню. Наприклад, якщо необхідно відкрити зображення як початкову операцію, то треба вибрати меню «Файл», а потім клацнути підменю «Відкрити» та так само для інших операцій. Створення меню необхідно виконати в конструкторі класу Cadre. Остання інструкція конструктора дозволяє нам пов'язати об'єкт класу PanDrawing з об'єктом Frame, що дозволяє додати панель до головного вікна. Тоді ця панель буде областю відображення зображення. Відповідно до принципу програмування делегування, натискання одного з підменю викликає метод actionPerformed(). Залежно від обраного підменю, цей метод викликає відповідний метод у класі PanDrawing. Метод addImage () класу PanDessin дозволяє читати файл зображення. Він приймає як параметр об'єкт типу java.io.File, який відповідає файлу зображення:

```
protected void ajouterImage(File fichierImage)
{
    monImage = ImageIO.read(fichierImage);
    repaint();
}
```

Об'єкт Imagefile є об'єктом класу java.io.File. Цей об'єкт можна отримати за допомогою конструктора класу File, який приймає шлях до файлу як параметр. Статичний метод read() класу ImageIO дозволяє завантажувати зображення в пам'ять із цього файлу, зв'язуючи з ним об'єкт BufferedImage. Цей метод очікує, поки зображення повністю завантажиться, перш ніж повертати об'єкт BufferedImage.

- ImageIO – це клас у пакеті javax.imageio. Цей пакет містить базові класи та інтерфейси для роботи з файлами зображень. Ми знаходимо такі класи:

- ImageIO використовується для опису вмісту файлів зображень, включаючи метадані;

- ImageReader, ImageReadParam і ImageTypeSpecifier для керування процесом читання зображення.

- ImageWriter і ImageWriteParam для керування процесом написання зображення;
- ImageTranscoder клас для перекодування між форматами;
- IIOException щоб повідомити про помилки.

Формати .jpg або .gif є зовнішніми форматами, які використовуються для представлення цифрових зображень. Щоб використовувати ці зображення в Java, необхідно перетворити ці формати у внутрішній формат. Це перетворення виконується класами пакета javax.imageio, зокрема класами ImageReader і ImageTranscoder . Фактично, підтримуваними форматами є gif, png, jpeg, bmp, wbmp.

Клас ImageIO вибирає відповідний зчитувач на основі типу файлу. Для цього він може перевірити розширення файлу та контрольне число, яке знаходиться в перших байтах і яке ідентифікує формат файлу. Якщо не вдається знайти відповідний зчитувач або зчитувач не може декодувати вміст файлу, метод read() повертає null. Інші формати, такі як tiff, можна обробляти, встановлюючи файли jar, що містять плагіни, які після встановлення дозволять автоматично розпізнавати новий формат без будь-яких змін у коді програми. Кожен формат зображення має свої переваги та недоліки:

В процесі робти можна використовувати метод getReaderFormatNames(), щоб дізнатися, які формати підтримуються для читання. Після прочитання файлу зображення за допомогою методу read() і отримання об’єкта BufferedImage для його представлення в пам’яті його можна відобразити на нашій панелі. Метод paintComponent() панелі спеціалізується на відображенні графічних елементів, які знаходяться на поверхні панелі. Цей метод автоматично викликається, коли нам потрібно відобразити або повторно відобразити вікно програми. Тому потрібно використовувати цей метод, щоб бути впевненим, що вхідне зображення відображатиметься повторно під час оновлення. Щоб відобразити вхідне зображення виокристовується метод drawImage(Image img, int x, int y, ImageObserver observer) класу java.awt.Graphics. Перший параметр методу має бути типу Image або похідного типу. У нашому випадку об’єкт myImage має тип

BufferedImage, він посилається на зображення в пам’яті. Значення x і y відображають координати верхньої лівої точки зображення. Зображення можна намалювати на поверхні панелі, оскільки вона також може бути поверхнею для малювання двовимірних форм або навіть зображень. За допомогою меню «Змінити розмір» можна зменшити або збільшити розмір зображення. Тому необхідно зробити трансформацію зображення.

```
    protected void reduireImage()
    {
        BufferedImage imageReduite = new
BufferedImage((int)(monImage.getWidth()*0.5),(int)(
monImage.getHeight()*0.5), monImage.getType());
        AffineTransform reduire =
AffineTransform.getScaleInstance(0.5, 0.5);
        int interpolation = AffineTransformOp.TYPE_BICUBIC;
        AffineTransformOp retaillerImage = new
AffineTransformOp(reduire, interpolation);
        retaillerImage.filter(monImage, imageReduite );
        monImage = imageReduite ;
        repaint();
    }
```

Масив пікселів зображення матеріалізується об'єктом типу java.awt.image.Raste. Використовуючи об’єкт екземпляра цього класу, можна отримати інформацію про колір пікселів. Повернений колір пікселя зберігається в масиві colorRGB розміром 3. Перше поле містить значення червоного компонента, друге для зеленого компонента і останнє поле для синього відтінку. Отримана інформація про тип зображення, який належить до типу BufferedImage.TYPE_INT_RGB , тобто колірна модель rgb без прозорості. На наступному кроці використовуємо клас ColorModel, який є компонентом класу BuferedImage і дозволяє представити колірну модель, що використовується для кожного зображення. Один із способів зробити це – використати метод getDataElements() , який повертає колір пікселя, не вимагаючи від коритсувача вказувати використану колірну модель. Інший способів зробити це використовувати метод setDataElements() замість метод getDataElements() для зміни пікселів.

Бінарне зображення складається з пікселів, які мають лише один із двох кольорів – чорний або білий. Теоретично існує кілька методів візуалізації бінарного зображення. Якщо використати простий метод порогового значення, то значення кожного пікселя $P(x, y)$ порівнюється з пороговим значенням S, і якщо це значення більше за S, піксель приймає значення 1 (чорний), інакше він приймає значення 0 (білий). Працюючи з BufferedImage, можна створити бінарне зображення, просто використовуючи тип BufferedImage.TYPE_BYTE_BINARY. Проте використання тільки одного порогу не завжди якісно дозволяє обрацювати вхідне зображення, тому доцільніше виколристовувати підхід за атоматичним обчисленням декількох порогових значень. Однак перед двоїковим перетворенням зображення ви повинні змінити його на рівень сірого за допомогою типу BufferedImage.TYPE_BYTE_GRAY, щоб мати один колірний компонент.

Інший спосіб обробки цифрових зображень це за допомогою фільтрації. Згортка – це дія використання матриці цілих чисел, яку також називають маскою або ядром, для множення значень кольору певного пікселя зображення на цю маску. Принцип алгоритму полягає в посиленні значення пікселя P і кожного з n пікселів, які його оточують, на відповідне значення в ядрі. Мета полягає в тому, щоб додати всі результати так, щоб піксель P приймав значення кінцевого результату.В роботі використовується вінкно розміром 3x3 точок. Приклад використовуваної маски та результат роботи наведено на рисунку 3.8.

0.1f, 0.1f, 0.1f,
0.1f, 0.2f, 0.1f,
0.1f, 0.1f, 0.1f

а) Ядро маски б) Вхідне зображення в) Вихідне зображення

Рисунок 3.8 – Приклади фільтрації цифрового зображення

Операція множення або, точніше, математична згортка виконується об'єктом класу ConvolveOp . Якщо необідно застосувати інші операції, такі як сегментація, розмиття або градієнт, необхідно змінити коефіцієнти матриці згортки, визначені змінною float[ ] blurmask. Наприклад, щоб отримати градієнт Sobel, необхідно використати настуні маски.

```
float[ ] maskGradientX =
{   -1f, 0f, 1f ,
    -2f, 0f, 2f ,
    -1f, 0f, 1f };
float[ ] maskGradientY =
{   1f, 2f, 1f,
    0f, 0f, 0f,
    -1f, -2f, -1f};
```

Використання стандартних можливостей мови програмування Java в поєднанні з використанням сучасних алгоритмів обробки цифрових зображень дозволило реалізувати запропонований алгоритм сегментації растрових зображень на основі використання елементів теорії графів. Внурішну функції дозволяють опрацьовувати цифрове зображення як на етапі попередньої обробки для підвищення рівня якості вхідного зображення так і вже безпосередньо для проведення процедури рзбиття вхідного зображення на однорідні області. На завершальному етапі використовують засоби для відображення отриманих результатів. Додатково користувач має можливість отримати статистичні дані про процеси, які відбувались в автоматичному режимі та провести їх корекцію у випадку якщо отриманий результат не задовільняє поставленим перед користувчем задач. Серед основних функцій, що використуються в прцесі аналізу цифрового зображенння є функції фільтрації, порогової обробки, формування графу та визначення його параметрів. Для процесу сегментації важливими є функції розрізання основного графа на підграфи, а також об’єднання невеликих за розміром підграфів для отримання множини однорідних областей. Дані функції вимагають додаткових порогових значень які користувач може відкоректоввати в процесі роботи для покращення результатів.

3.3 Тестування та аналіз реалізованого програмного модуля

Для можливості оцінки ефективності програної розробки було проведено ряд тестових експериментів. Процес тестування проводився на компютері, які міє середні/офісні технічні характеристики, для того щоб уникнути ситуації, коли наефективність роботи програмного додатку значно впливають параметри апаратного забезпечення на якому воно виконується. Вибір інтегрованої відеокарти базувався на мінімізації впливу її робти на кінцевий час обробки цифрового зображення. Перелік параметрів комп'ютера, що був використаний на основному етапі тестування наведено на рисунку 3.9.

Назва параметра	Значення
Процесор (CPU)	Intel Core i5-12400 (6 ядер, 12 потоків, базова частота 2.5 ГГц, Turbo до 4.4 ГГц
Оперативна пам'ять (RAM)	Corsair Vengeance LPX 8GB (2x4GB) DDR4 3200 MHz
Система зберігання (вінчестер)	SSD Kingston A2000 500GB NVMe
Відеокарта (GPU)	Інтегрована графіка Intel UHD 730 (для Intel Core i5-12400)
Материнська плата	ASUS Prime B560M-A
Монітор	Dell P2419H 24" Full HD IPS
Операційна система	Windows 11 Pro
Клавіатура та мишка	Logitech K120 (клавіатура) та Logitech M100 (мишка)
Периферійні пристрої (камера)	Logitech C920 HD Pro Webcam
Живлення	Corsair CV450, 450W
Мережа	TP-Link Archer T3U Plus (Wi-Fi адаптер)
Вартість	900$

Рисунок 3.9 – Параметри робочої станції для тестування

Для дослідження можливостей програмного додатку при рообті з різними вхідними ображеннями було сформовано декілька тестових вибірок, цифрові зображення в яких мали характерні особливості. Зокрема були виділені наступні групи зображень:

- Елементарні(бінарні зображення) – дана група зображень характеризується відсутність різноманітності кольорів, кількість об'єктів на таких зображеннях 1-5 штук. Границі чітко прослідковуються. Окремі елементи не дотикаються та однозначно ідентифікуються. Прикладом таких зображень може бути фото конвеєрного полотна.

- Прості зображення – на даних зображеннях міститься велика кількість різнх об’єктів, велика гама кольорів, елементи на зображенні можуть дотикатись, проте відмінність між різними об’єктами чітко прослідковується.
- Складні зображення характеризуються наявністю великої кількості об’єктів які можуть дотикатись/накладатись одне на одного, колірні відмінності між сусідніми обєктами можуть бути мінімальними, границі не завжди омжна однозначно ідентифікувати за допомогою візуальної оцінки.

Даний набір тестових зображень може у повній мірі продемонструвати можливості програмного забезпечення під час роботи з растровими зображеннями. В результаті проведених тестувань розроблена програма коректно провела обробку усіх запропонованих зображень, час на опрацювання одного зображення не первищував 1с, критичні позибки під час візуальної оцінки результатів робти не було виявлено. Якщо аналізумати кожну тестову вибірку зображень окремо, то можна зробити висновки, що елементарні зображення були сегментовані в режимі реальнго часу, без жодних похибок. Алгоритм чітко та однозначно міг рзділити основний граф на однорідні за кольором підграфи. Під час опрацювання зображень даного типу програма не задаваладодаткових запитань та працювала в автоматичному режимі. Під час опрацювання зображень другої тетової групи було помічено збільшення часових затримок в процесі сегментації. Результати сегментації візуально не відрізнялись від очікуваних, проте в деяких випадках програма зверталась за допомогою до користувача для підтвердження/корекції параметрів її роботи. Помилкових результатів сегментації не було виявлено. Результати сегментації складних зображень під час деяких експериментів містили хибне спрацювання. У більшості випадів програма потребувала втручанн користувача. При коректній роботі користувача результат сегментації відповідали очікуваним відповідям.

Для більш детальної оцінки результатів робти програмного додатку було проведено порівняння результатів його робити алгоритмами-аналогами. Порівняння було здійснено на вище опрацьованих зображення, а самі алгоритми були взяті з бібліотеки OpenCV. (рисунок 3.10)

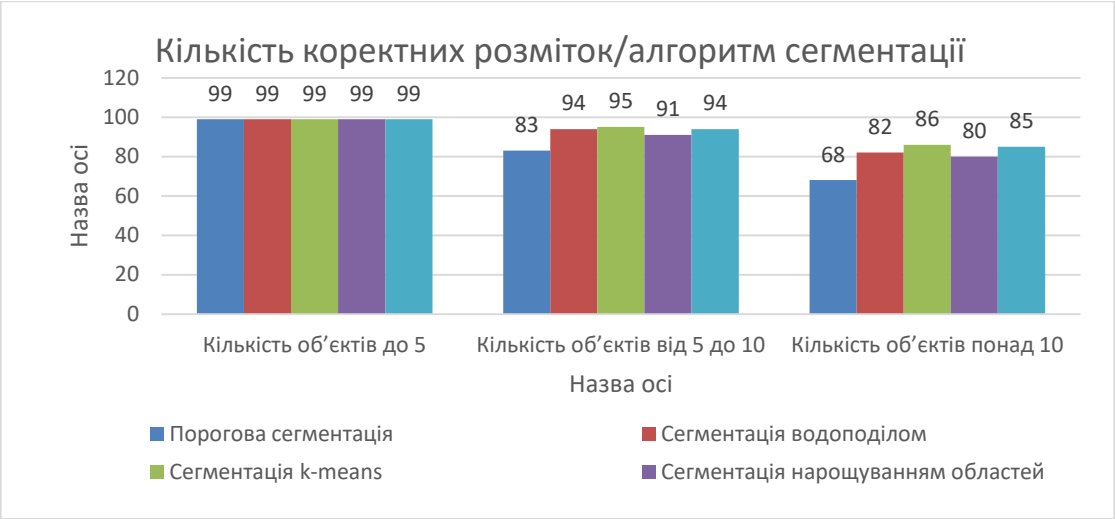

Рисунок 3.10 – Приклад порівняння розробленого алгоритму з аналогами

Проведене тестування та порівняння запропонованого алгоритму з аналогами показало наступні переваги розробленого програмного забезпечення:

- простота реалізації та швидкість роботи;
- можливість опрацювання зображень різного типу;
- можливість внесення корекцій в значення параметрів під часробти алгоритму.

3.4 Висновки до розділу

Розроблено та проведено дослідження внутрішньої архітектури програмного додатку обрбки цифрових зображень, що даломожливість в подальшому провести програмну реалізацю запропонованого алгоритму сегментації зображень на основі теорії графів.

Програмно реалізовано систему обрбки та аналізу цифрових зображень на основі використання елементів терії графів, що дозволило провести тестові дослідження на різнх типах растрових зображень та порівняти отримані результати з програмами- аналогами.

ВИСНОВКИ

В результаті проведених досліджень та на сонові використання елементів теорії гарфів в системи обрбки кольорових цифрових зображень можна зробити наступні висновки:

1. Здійснено аналізу технологій створення цифрових зображень на основі дослідження параметрів процесу створення цифрових відображень сцен, що дозволило виділити основні фактори які впливають на якість вихідних цифрових зображень.

2. Досліджено методи та алгоритми теорії графів, на основі аналізу характеристик різних типів графів, що дозволило виділити основні сфери їх застосування.

3. Проведено аналітичний огляд цифрової бібліотеки оброки цифрових зображень OpencvCV, на основі дослідження її функціональних можливостей та використовуваних структур даних, що дозволило виділити основні функції для обробки цифрових зображень.

4. Проведено аналіз відомих методів та алгоритмів сегментації цифрових кольорових зображень, на основі базових принципів розподулі пікселів по одноріжнихм областям, що дало можливість виділити групу алгоритмів сегментації на основі використання елементів теорії графів.

5. Запропоновано алгоритм сегментації цифрових кольорових зображень на основі використання елементів теорії графів, що дозволило зменшити вплив попередньої обробки на результати сегментації зображень.

6. Програмно реалізовано систему обрбки та аналізу цифрових зображень на основі використання елементів терії графів, що дозволило провести тестові дослідження на різнх типах растрових зображень та порівняти отримані результати з програмами- аналогами.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Шпак В.О., Патра В.М. Аналіз алгоритмів сегментації цифрових зображень. X Всеукраїнська науково-практична конференція «Інтелектуальні комп'ютерні системи та мережі» (ІКСМ'24). 05 листопада 2024, Тернопіль, Україна. Тернопіль: ЗУНУ, 2024.

2. Патра В.М., Шпак В.О. Алгоритм обробки та аналізу біомедичного зображення. X Всеукраїнська науково-практична конференція «Інтелектуальні комп'ютерні системи та мережі» (ІКСМ'24). 05 листопада 2024, Тернопіль, Україна. Тернопіль: ЗУНУ, 2024.

3. Дубчак Л. О., Гураль І. В. Методичні вказівки до оформлення курсових проектів, звітів про проходження практики, випускних кваліфікаційних робіт для студентів спеціальності «Комп'ютерна інженерія» / ред. О. М. Березький. Тернопіль : ТНЕУ, 2019. 33 с.

4. Дубчак Л. О., Мельник Г. М. Методичні рекомендації до виконання кваліфікаційної роботи з освітнього ступеня "Магістр". Спеціальність: 123 - Комп'ютерна інженерія. Магістерська програма — Комп'ютерна інженерія" / ред. О. М. Березький. Тернопіль: ЗУНУ, 2020. 32 с.

5. Kucharski, A.; Fabijańska, A. CNN-watershed: A watershed transform with predicted markers for corneal endothelium image segmentation. Biomed. Signal Process. Control 2021, 68, 102805.

6. Gonzalez, R.C.; Woods, R.E. Digital Image Processing; Prentice Hall: Upper Saddle River, NJ, USA, 2008.

7. Gostick, J.T. Versatile and efficient pore network extraction method using marker-based watershed segmentation. Phys. Rev. E 2017, 96, 023307.

8. Kornilov, A.S.; Safonov, I.V. An overview of watershed algorithm implementations in open source libraries. J. Imaging 2018, 4, 123.

9. Sun, Y., Hua, Y., Mou, L., Conditional GIS-Aware Network for Individual Building Segmentation in a VHR SAR Image. In 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS. – 2021. - pp. 432-435.

10. Jin, X., Gu, Y., Intrinsic Hyperspectral Image Decomposition With DSM Cues. IEEE Transactions on Geoscience and Remote Sensing, 60, - 2021.- 1-13.

11. Zhuang, L., Gao, L., Zhang, B., Fu, X., & Bioucas-Dias, J. M. Hyperspectral image denoising and anomaly detection based on low-rank and sparse representations. IEEE Transactions on Geoscience and Remote Sensing, 60. – 2020 - 1-17.

12. A. Arnab and P. H. Torr. Pixelwise instance segmentation with a dynamically instantiated network. In CVPR, 2017.

13. M. Bai and R. Urtasun. Deep watershed transform for instance segmentation. In CVPR, 2017.

14. H. Caesar, J. Uijlings, and V. Ferrari. COCO-Stuff: Thing and stuff classes in context. In CVPR, 2018. 75 10.L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. PAMI, 2018.

15. Image Segmentation: 10 Concepts, 5 Use Cases and a Hands-on Guide. URL: https://www.basic.ai/blog-post/image-segmentation:-10-concepts,-5-usecases-and-a-hands-on-guide-%7C-updated-2024#viewer-480aa (дата звернення 14.11.2024).

16. Borowiec, M. L., Dikow, R. B., Frandsen, P. B., McKeeken, A., Valentini, G., & White, A. E. (2022). Deep learning as a tool for ecology and evolution. Methods in Ecology and Evolution, 13(8), 1640-1660.

17. Human Pose Estimation using Keypoint RCNN in PyTorch. URL: https://learnopencv.com/human-pose-estimation-using-keypoint-rcnn-in-pytorch/ (дата звернення 14.11.2024).

18. Thresholding-Based Image Segmentation. URL: https://paimo.jodymaroni.com/thresholding-based-image-segmentation/ (дата звернення 14.11.2024).

19. Bogucharskiy, S., & Mashtalir, V. Image segmentation via X-means under overlapping classes. In 2015 Xth International Scientific and Technical Conference" Computer Sciences and Information Technologies"(CSIT) (pp. 45-47). IEEE.

20. Mashtalir, V., & Bogucharskiy, S. Covering image segmentation via matrix X-means and J-means clustering. Sensors & Transducers, 195(12), 56- 61.

21. Bogucharskiy, S., Mashtalir, V., & Mikhnova, O. Fuzzy JMeans image segmentation. In Advances in Data Science: proc. International Workshop and Networking Event, Poland, Holny Mejera (pp. 6-8).

22. Mashtalir, S., & Mashtalir, V. Spatio-temporal video segmentation. Advances in Spatio-Temporal Segmentation of Visual Data, 2020, 161-210.

23. Кобилін, О. А., & Творошенко, І. С. Методи цифрової обробки зображень, 2021, 72-75.

24. Chowdhary, C. L., & Acharjya, D. P. Segmentation and feature extraction in medical imaging: a systematic review. Procedia Computer Science, 167, 2020, 26-29.

25. OpenCV projects – Image segmentation with Watershed algorithm. URL: https://datahacker.rs/007-opencv-projects-image-segmentation-with-watershedalgorithm/ (дата звернення 17.10.2024).

26. Graph-cut segmentation principle. URL: https://www.researchgate.net/figure/Graph-cut-segmentationprinciple_ fig2_ 285228259 (дата звернення 11.09.2024).

27. Sultana, F., Sufian, A., & Dutta, P. Evolution of image segmentation using deep convolutional neural network: A survey. KnowledgeBased Systems, 2020.

28. Tvoroshenko, I., Gorokhovatskyi, V., Kobylin, O., & Tvoroshenko, A. Application of deep learning methods for recognizing and classifying culinary dishes in images, 2023, 59-62.

29. Thakur, N., Bhattacharjee, E., Jain, R., Acharya, B., & Hu, Y. C. Deep learning-based parking occupancy detection framework using ResNet and VGG-16. Multimedia Tools and Applications, 83(1), 2023, 1941-1964.

30. Bilonoh, B., Bodyanskiy, Y., Kolchygin, B., & Mashtalir, S. Tunable Activation Functions for Deep Neural Networks. In International Scientific Conference

“Intellectual Systems of Decision Making and Problem of Computational Intelligence” 2023, pp. 624-633.

31. Badrinarayanan, V., Kendall, A., & Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE transactions on pattern analysis and machine intelligence, 39(12), 2481-2495.

32. Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18, pp. 234-241.

33. Boonpook, W., Tan, Y., & Xu, B. Deep learning-based multifeature semantic segmentation in building extraction from images of UAV photogrammetry. International Journal of Remote Sensing, 42(1), 2021, 1-19.

34. Pomazan, V., Tvoroshenko, I., & Gorokhovatskyi, V. Handwritten character recognition models based on convolutional neural networks, 2023, 65-66.

35. Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence, 40(4), 2023, 834-848.

36. He, J., Wu, P., Tong, Y., Zhang, X., Lei, M., & Gao, J. Bearing fault diagnosis via improved one-dimensional multi-scale dilated CNN. Sensors, 21(21), 2023, 7319.

37. Wu, H., Zhang, J., Huang, K., Liang, K., & Yu, Y. Fastfcn: Rethinking dilated convolution in the backbone for semantic segmentation. arXiv preprint arXiv:1903.11816. 2019.

38. Takikawa, T., Acuna, D., Jampani, V., & Fidler, S. Gated-scnn: Gated shape cnns for semantic segmentation. In Proceedings of the IEEE/CVF international conference on computer vision. 2019. pp. 5229-5238.

39. Berezsky O.M. Texture segmentation of biomedical images based on spatial moments / O.M. Berezsky, Yu.M. Batko, G.M. Melnyk. - Proceedings of the 4th International Scientific-Technical Conference "Computer Science and Information

Technology 2009", (October 15-17, 2009, Lviv, Ukraine), "Tower and Co", - pp 42-45.

40. Berezsky O.M. Segmentation of Cytological and Histological Images of Breast Cancer Cells / O.M. Berezsky, Yu.M. Batko, G.M. Melnyk, S.O. Verbovyy, L. Haida. - Proceedings of the 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS'2015), 24-26 September 2015, Warsaw, Poland. – Warsaw, 2015. – V.1. – P. 287-292.

41. Chuanbo Wang Fully automatic wound segmentation with deep convolutional neural networks // Scientific reports. –2020. –Vol. 10, no.1.

42. Tapasvi B. A survey on semantic segmentation using deep learning techniques. Nternational journal of engineering research & technology. –Vol.9, no. 5.

43. Jianping Gou. Knowledge distillation:asurvey. International journal of computervision. –2021. –Vol. 129, no. 6. –P. 1789–1819.

44. Olga Russakovsky. ImageNet large scale visualrecognition challenge. International journal of computervision. –2015. –Vol. 115, no. 3. –P. 211–252.

45. Shervin Minaee. Image segmentation using deep learning: a survey. IEEE transactions on pattern analysis and machine intelligence. –2021. –P. 1.

46. Mennatullah Siam. RTSeg:real-time semantic segmentation comparative study. 25th IEEE internationalconference on image processing, Athens, 7–10. 2018.

47. Mohammad HesamHesamian. Deep learningtechniques for medical image segmentation: achievements and challenges. Journalofdigital imaging. –2019. –Vol. 32, no. 4. –P. 582–596.

48. Guotai Wang. Interactive medical image segmentation using deep learning with image-specific finetuning. IEEE transactions on medical imaging. –2018. –Vol. 37, no. 7. –P. 1562–1573.

49. Lai M. Deep learning for medical image segmentation [електронний ресурс] URL:https://arxiv.org/abs/1505.02000 (дата звернення 14.11.2024).

50. Chen Chen. Deep learning for cardiacimage segmentation: a review. Frontiers in cardiovascular medicine. –2020. –Vol. 7.