Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інтелектуальних інформаційних технологій та автоматизації
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Інформаційна технологія інтелектуальної обробки зображень»

Виконав: студент 2-го курсу, групи КН-21м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
Побережник В. Р.
(прізвище та ініціали)

Керівник: к.т.н., доц. каф. КН
Колодний В. В.
(прізвище та ініціали)
«15» 12 2022 р.

Опонент: д.т.н., проф. каф. ПЗ
Жемецький М. С.
(прізвище та ініціали)
«15» 12 2022 р.

Допущено до захисту
Завідувач кафедри КН
д.т.н., проф. Яровий А. А.
(прізвище та ініціали)
«16» 12 2022 р.

Вінниця ВНТУ - 2022 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань – 12 – «Інформаційні технології»
Спеціальність – 122 – «Комп'ютерні науки»
Освітньо-професійна програма – «Системи штучного інтелекту»

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
Д.т.н., проф. Яровий А.А.
14.09 **2022 року**

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Побережнику В'ячеславу Руслановичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Інформаційна технологія інтелектуальної обробки зображень» керівник роботи к.т.н., доц. каф. КН, Колодний В. В.
затверджені наказом вищого навчального закладу від «14» 09 2022 року №203
2. Строк подання студентом роботи 18 листопада 2022 року
3. Вихідні дані до роботи: мова програмування – ООП; максимальна швидкодія обробки зображення – не більше 3 с; формат зображення – jpg; мінімальний об'єм тестової вибірки – 4 од.; інтерфейс користувача – інтуїтивно зрозумілий.
4. Зміст текстової частини: вступ, обгрунтування доцільності розробки інформаційної технології інтелектуальної обробки зображень, моделювання інформаційної технології інтелектуальної обробки зображень, структурна організація та особливості програмної реалізації інформаційної технології інтелектуальної обробки зображень, економічна частина, висновки, список використаних джерел, додатки.
5. Перелік ілюстративного матеріалу (з точним зазначенням обв'язкових креслень): структурна схема інформаційної технології; схема моделі функціонування інформаційної технології; схема загального алгоритму функціонування інформаційної технології; схема алгоритму розбиття зображення на пікселі; UML-діаграма класів; загальний вигляд інтерейсного вікна програми; загальний вигляд інтерейсного вікна для гістограми яскравості тестового зображення.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціалита посада консультанта	Підпис, дата	
		завдання видав	викон прий
1-3	Колодний В. В. к.т.н., доц. каф. КН	17.05.2022	14.12.202
4	Нікіфорова Л. О. к.т.н. доцент каф. ЕПВМ	14.09.2022	12.12.202

7. Дата видачі завдання 14. 09 2022 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	При-мітк
1	Обґрунтування доцільності розробки інформаційної технології інтелектуальної обробки зображень	14.09.2022р. – 1.10.2022р	Розділ 1
2	Моделювання інформаційної технології інтелектуальної обробки зображень	2.10.2022р – 16.10.2022р	Розділ 2
3	Структурна організація та особливості програмної реалізації інформаційної технології інтелектуальної обробки зображень	17.10.2022р – 07.11.2022р	Розділ 3
4	Підготовка економічної частини	08.11.2022р – 07.12.2022р.	Розділ 4
5	Апробація та/або впровадження результатів дослідження	23.11.2022р. – 01.12.2022р	тези доп
6	Оформлення пояснювальної записки, графічного матеріалу та презентації	02.12.2022р. – 14.12.2022р	Пояснюв записка, графічни матеріал презента

Студент _____ (підпис) Побережник В. Р.

Керівник роботи _____ (підпис) Колодний В. В.

АНОТАЦІЯ

УДК 519.87

Побережник В. Р. Інформаційна технологія інтелектуальної обробки зображень. Магістерська кваліфікаційна робота зі спеціальності 122 – Комп'ютерні науки, освітня програма – Комп'ютерні науки. Вінниця: ВНТУ, 2022. 110 с.

На укр. мові. Бібліогр.: 39 назв; рис.: 18; табл. 7.

Дана магістерська кваліфікаційна робота присвячена розробці інформаційної технології інтелектуальної обробки зображень. Виконано аналіз сучасних програм-аналогів, які використовуються для обробки зображень та побудови гістограм, наведено коротку порівняльну характеристику знайдених програм-аналогів, досліджено методи, які можуть бути використані для реалізації поставленої задачі. Проведено обґрунування моделі інтелектуальної обробки зображень. Як базовий метод запропоновано застосувати метод побудови гістограм на основі аналізу спектру зображення. Розроблено алгоритм інтелектуальної обробки зображень та відповідне програмне забезпечення на мові програмування Java, в середовищі IntelliJ IDEA. Аналіз роботи програмного забезпечення показав підвищення швидкодії обробки зображень.

Графічна частина складається з 7 плакатів із результатами моделювання.

У розділі економічної частини здійснено оцінювання комерційного потенціалу розробки інформаційної технології інтелектуальної обробки зображень, проведено оцінювання комерційного потенціалу розробки, спрогнозовано витрати на виконання наукової роботи та впровадження результатів, які склали 777 871,62 грн, розраховано період окупності – 1,28 роки.

Ключові слова: інформаційна технологія, метод аналізу спектру, гістограми, обробка зображень.

ABSTRACT

Poberezhnyk V. R. Information technology of intelligent image processing. Master's degree in specialty 122 - Computer Science, educational program – Computer Science. Vinnytsia: VNTU, 2022. 112 p.

In Ukrainian language. Bibliogr .: 39 titles; fig .: 18; table 7.

This master's thesis is devoted to the development of information technology of intelligent image processing. An analysis of modern analog programs used for image processing and histogram construction is performed, a brief comparative description of the found analog programs is given, and methods that can be used to implement the given task are investigated. Grounding of the model of intelligent image processing was carried out. As a basic method, it is proposed to use the method of constructing histograms based on the analysis of the image spectrum. An intelligent image processing algorithm and corresponding software in the Java programming language, in the IntelliJ IDEA environment, were developed. Analysis of the software's performance showed an increase in the speed of image processing.

The graphic part consists of 7 posters with simulation results.

In the section of the economic part, an assessment of the commercial potential of the development of information technology of intelligent image processing was carried out, an assessment of the commercial potential of the development was carried out, the costs of carrying out scientific work and the implementation of the results were forecast, which amounted to UAH 777,871.62, the payback period was calculated 1.28 years.

Keywords: information technology, spectrum analysis method, histograms, image processing.

ЗМІСТ

ВСТУП

Актуальність теми дослідження. На сьогоднішній день отримання, обробка та подальше використання цифрових зображень відіграють важливу роль у наукових дослідженнях, промисловості, інформаційних системах та медицині. Багато прикладів використання цифрових вказують на те, що вони використовуються у всіх сферах, де використовуються інформаційні технології.

Розвиток інформаційних технологій призводить до активного розвитку методів цифрової обробки сигналів. Цей процес посилюється інтеграцією сучасних комп’ютерних і телекомунікаційних технологій. Особливого розвитку в сучасних умовах набувають методи цифрової обробки зображень, оскільки вони складають значну частину загального трафіку мультисервісних мереж. Доречним і актуальним науково-практичним завданням є діяльність, пов’язана з удосконаленням наявних та розробкою нових методів інтелектуальної обробки зображень.

Інтелектуальна обробка зображень є важливою областю застосування сучасних комп'ютерних технологій. Цифрова обробка зображень є одним із пріоритетних напрямів науки і техніки. Це пояснюється тим, що зображення використовуються як засіб отримання візуальної інформації в системах спостереження, технічного зору, відеотелефонії, телебачення, автономних інтелектуальних системах, телемедицині тощо. Візуальна якість сприйняття зображення, стиснення даних для зберігання та передачі по каналах зв'язку, а також аналіз, розпізнавання та інтерпретація візуальних зображень для прийняття рішень відіграють додаткову важливішу роль.

Так як на сьогоднішній день існують тільки громіздкі системи обробки зображення, які мають певне призначення та спеціалізований функціонал для подальшої роботи з зображенням, багато з яких мають певні недоліки щодо швидкості роботи, або ж не мають безкоштовної версії, тому розробка інформаційної технології інтелектуальної обробки зображень є актуальною.

Зв'язок роботи з науковими програмами, планами, темами. Магістерська кваліфікаційна робота виконана відповідно до напрямку наукових досліджень кафедри комп'ютерних наук Вінницького національного технічного університету 22 К1 «Моделі, методи, технології та пристрої інтелектуальних інформаційних систем управління, економіки, навчання та комунікацій» та плану наукової та навчально-методичної роботи кафедри.

Мета та завдання дослідження. Метою дослідження є підвищення швидкодії обробки цифрових зображень за допомогою розробки інформаційної технології обробки графічних даних, що поєднує в собі метод вирівнювання гістограми на основі нечіткої логіки та аналізу спектрів зображень.

Для досягнення поставленої мети необхідно розв'язати наступні **задачі**:

– провести аналіз програм аналогів, систем-аналогів та обґрунтування доцільності розробки інформаційної технології інтелектуальної обробки зображень;

– розробити математичну модель інтелектуальної обробки зображень;

– розробити структуру інформаційної технології інтелектуальної обробки зображень;

– здійснити програмну реалізацію інформаційної технології інтелектуальної обробки зображень;

– провести тестування програми та проаналізувати отримані результати;

– економічно обґрунтувати доцільність розробки інформаційної технології інтелектуальної обробки зображень.

Об'єкт дослідження – це процес інтелектуальної обробки зображень.

Предмет дослідження – програмні засоби інтелектуальної обробки зображень.

Методи дослідження. У роботі використано такі методи наукових досліджень: метод системного аналізу для аналізу структури інформаційної системи; методи аналізу гістограм; методи сегментації, методи аналізу спектрів зображень; методи об'єктно-орієнтованого програмування для автоматизації розрахунків.

Наукова новизна одержаних результатів полягає в наступному:

удосконалено модель інтелектуальної обробки зображень, яка відрізняється від відомих поєднанням методу вирівнювання гістограми на основі нечіткої логіки та методу аналізу спектрів зображень за допомогою перетфорення Фур'є, що дозволяє максимально задовольнити потреби та максимально швидко здійснити обробку зображення.

Практичне значення одержаних результатів полягає у такому:

1. Удосконалено алгоритм обробки зображень, який використовує поєднання методу вирівнювання гістограми та методу аналізу спектів зображень, що забезпечує підвищення швидкодії обробки цифрових зображень.

2. Розроблено програмне забезпечення інформаційної технології інтелектуальної обробки зображень, що може бути використана для подальшої обробки фотографій або використання цієї технології у фото-програмах для кращого автофокусування.

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується строгістю постановки задач, коректним застосуванням математичних методів під час доведення наукових положень, строгим виведенням аналітичних співвідношень, порівнянням результатів з відомими та збіжністю результатів математичного моделювання з результатами, що отримані під час впровадження розроблених програмних засобів.

Особистий внесок здобувача. Результати даної магістерської кваліфікаційної роботи отримані самостійно. В публікації у співавторстві здобувачу належить дослідження перспектив інформаційної технології інтелектуальної обробки зображень [1].

Апробація результатів роботи. Результати досліджень було апробовано на «Всеукраїнська науково-практична інтернет-конференція студентів, аспірантів та молодих науковців "МОЛОДЬ В НАУЦІ: ДОСЛІДЖЕННЯ, ПРОБЛЕМИ, ПЕРСПЕКТИВИ" (2023)»[1].

Публікації. За основними результатами досліджень опубліковано тези доповіді на концференції [1].

1 ОБҐРУНТУВАННЯ ДОЦІЛЬНОСТІ РОЗРОБКИ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ІНТЕЛЕКТУАЛЬНОЇ ОБРОБКИ ЗОБРАЖЕНЬ

1.1 Особливості застосування обробки зображень

Проблема обробки зображень, розпізнавання, ідентифікації та виявлення їх достовірності постає з особливою гостротою з постійним збільшенням обсягів інформаційних потоків і підвищенням вимог до динаміки їх обробки в реальному часі. Зі збільшенням розмірності зображень, які потрібно ідентифікувати, обчислювальна складність зростає експоненціально.

Розглянемо найбільш популярні аспекти використання технологій і методів цифрової обробки зображень, їх значимість і практичне застосування. Аналіз сфер застосування методів обробки цифрових зображень виявив, що вони проникають практично в усі види соціального життя людини.

Компресія цифрових зображень (стиснення, компактне представлення) є одним з найважливіших завдань обробки цифрових зображень, вона пов’язана з потребою заощадження пам’яті на фізичних носіях. Ось приклади реального застосування компресії зображень:

– передача стиснених зображень на штучному супутнику Землі у процесі встановлення сеансу зв’язку для збільшення кількості інформації;

– зниження обсягів зображень з метою швидкого відкриття веб-сторінок;

– компактне подання зображень оптико-електронними приладами (фотоапарати, камери) для заощадження дискового простору;

– використання архіваторів зображень для зберігання архівних, історичних, художніх та інших документів;

– компресія зображень як довільних кадрів відеоряду з метою зменшення обсягів медіа та зниження вимог до якості каналів зв’язку.

Фільтрування шумів є одним із завдань ідентифікації об'єктів у військовій справі, важлива для підвищення чіткості зображення в цифровому телебаченні, фільтрації сигналів у належних блоках сучасної апаратури [2].

Веб-дизайн і комп'ютерна графіка популярні при розробці штучної графіки (комп'ютерні ігри, анімація, інтерфейс веб-сторінок) і, наприклад, розробки експертних систем для автоматизованих військових програм управління, критичних прикладних об'єктів (керування атомними електростанціями, залізницею і повітряним транспортом, центрами космічних польотів).

Розпізнавання образів є актуальним для військової справи, в час коли вірна ідентифікація ворожого об’єкта здатна істотно змінити хід бойових дій чи місцевих операції, має місце в працях археології та архівознавства, впливає на відтворення документальних знахідок, що мають історичну цінність, а також при картографуванні території та транслювання інформації зі штучних супутників Землі.

Обробка цифрових зображень – це галузь обчислювальної техніки, яка активно розвивається та включає як апаратне, так і програмне забезпечення.

Цифрове зображення – модель реального або синтетичного зображення, яка зберігається у файлі на комп’ютерному носії у виді набору кодів (фігур).

Сфера діяльності, пов'язана з аналізом та обробкою цифрових зображень, називається комп'ютерною графікою [3].

Галузі, пов'язані з цифровою обробкою зображень, – це теорія інформації, теорія оптимального прийому сигналу та теорія розпізнавання зображень. Поняття «сигнал» уподібнюється поняттям «дані» та «інформація».

Сигнал – це одиниця інформації, що надає дані про фізичні характеристики, стан або поведінку розглянутої фізичної програми, об’єкта чи середовища.

Під сигналом ми маємо на увазі конкретну змінну, яка налає або включає певний тип даних, і яку можна, наприклад, передати, відобразити на екрані або виконати певними діями.

Обробка сигналів зображення здійснюється з метою зміни якості зображень, надання зображенню нових властивостей, аналізу інформації, що міститься в зображенні, або зменшення потоку сигналів, що передує передачі або запису сигналів зображення [4].

Незважаючи на тип зображень всю різноманітність принципів і методів обробки зображень розділяють на наступні напрямки [5]:

– відновлення та покращення зображень;

– аналіз зображення (розпізнавання зображення та аналіз сцени);

– синтез зображення;

– кодування сигналів зображення.

Розглядаючи перший напрямок, мають на увазі зміну контрастності, видалення шумів, уточнення меж об’єктів, виправлення кольори.

Коли ми говоримо про вдосконалення іміджу, то маємо на увазі редагування його властивостей, що має на меті приведення його до більш комфортного суб’єктивного враження про такий образ, а не приведення до надмірної схожості з дійсним образом.

За допомогою другого напряму ідентифікуються об'єкти, присутні в шуканій сцені, оцінюється пов’язаність фрагментів зображення і визначаються характеристики зображених об'єктів.

Останнім часом надзвичайного розповсюдження отримав синтез зображень. Методи і способи синтезу іміджу використовуються в абсолютно різних сферах діяльності.

Так, наприклад, здійснюється синтез тривимірних зображень з плоских фотографій поверхні Землі або поверхні інших космічних об'єктів з метою вивчення властивостей цих об'єктів.

Синтез здійснюється при двовимірному або тривимірному моделюванні об'єктів при автоматизованому проектуванні будівель і транспортних засобів.

Процес кодування сигналу зображення виконується для зменшення потоку сигналу, потрібного для записування або передавання інформації про зображення.

Реєстрацією зображення називають процес перетворення деяких наборів даних в єдину систему координат. Такими даними можуть виступати серії фотографій, інформація з різних датчиків, моменти часу, глибина або точки спостереження [6].

Алгоритми, що реалізовують реєстрацію зображень застосовуються в комп’ютерному зорі, медичних зображеннях, військових програмах для машинного розпізнавання цілей, а також для організації та аналізу супутникових знімків. Запис має на меті надання можливості порівняння або інтегрування даних, отриманих з різних записуючих пристроїв.

Сучасні інформаційні системи мають можливість передачі, зберігання та обробки зображень здебільшого в цифровій формі, проте первинні зображення існують переважно як безперервні двовимірні поля розподілу яскравості та кольорів.

Трансформація первинних зображень у цифрові сигнали – обов'язкова операція, коли пропонується використовувати цифрову обробку, передачу та зберігання. Такі зміни вміщуєть дві процедури, що виконуються синхронно.

Перший має на увазі заміну неперервного зображення набором переривчастих елементів і називається дискретизацією, а другий замінює безперервний розподіл яскравості і кольору набором хвильових значень для всіх елементів зображення і називається квантуванням [7].

Двовимірність зображення порівняно зі звичайними одновимірними сигналами надають додаткові можливості для оптимізації потоку цифрових сигналів для зменшення кількості цифрових даних.

Будова поля зображення, що утворюється в результаті елементарного розбивання або синтезу зображення, називають растровою. Сучасна термінологія застосовує назву елемента зображення «піксель» або «піксел», а в іноземній літературі зустрічається кілька відповідних назв, утворених від поєднання слів picture element (елемент зображення) – pictel, pixel, pel.

На практиці використовується дискретизація за допомогою прямокутної рамки та квантування рівномірної світності. Такий підхід використовується не тільки в наслідок простоти у виконанні відповідних операцій, а ще й у зв’язку з необхідністю виконання інших операцій, пов’язаних із перетвореннями зображення.

За умови, що в кінцевому вигляді використовується прямокутний растр, сигнали відцифрованого зображення представлені за допомогою матриці (однотонне зображення) чи кількох матриць (різнобарвне зображення), рядки і стовпці в таких матрицях включають квантовані значення параметрів еквівалентних елементів дискретного зображення.

1.2 Дослідження проблем сучасних методів обробки зображень

Цифрове зображення – це дискретний простір, що складається з дрібних елементів поверхні, які називаються пікселями. Кожен з цих елементів містить значення або набір значень, що кодують рівень інтенсивності в кожній позиції. Цифрове зображення можна отримати за допомогою великої кількості різних пристроїв, таких як камера, МРТ-апарат або будь-який пристрій з датчиком, здатним фіксувати інтенсивність світла [8].

Обробка зображень для автоматичного аналізу. Обсяг інформації, що підлягає аналізу, настільки великий, що візуальні методи вивчення та оцінки не можуть задовольнити не тільки дослідників, а й потреби практичних завдань. Вивчення багатьох спектральних характеристик при обробці дані являють собою складну операцію вимірювання та процедуру обчислення. У таких випадках використовується багатозонна реєстрація просторово суміщених зображень. Багатозонні телевізійні системи давно використовуються в астрофізиці та метеорології для аналізу аерокосмічних зображень.

Перетворення зображень, призначених для автоматичного аналізу, зазвичай включає процедури збереження в пам'яті, обробки із затримкою та визначення значущих параметрів. У процесі автоматичної обробки зображення досліджуваного об'єкта формується перелік параметрів, часто в матричній формі або у вигляді стилізованого зображення (напівавтоматичний аналіз). Перелік значущих параметрів формується в залежності від конкретних завдань.

Аналіз багатьох джерел дозволяє виділити найбільш часто використовувані методи обробки: операція згортки в просторовій області;

фільтрація в просторово-частотній області; корекція затінення (вирівнювання яскравості в полі зображення); нелінійне амплітудне перетворення сигналу зображення; операція зіставлення з порогом; бінаризація зображення; рангова фільтрація; процедури локального усереднення; градієнтні перетворення; інтерполяція зображень у просторовій області; інверсія зображення; аналіз логічних зв'язків в зображенні; підсумовування і віднімання зображень; знайти крайні на малюнку. В окрему групу можна виділити геометричні перетворення зображень: масштабні перетворення (збільшення, зменшення), поворот. Процедури функціонального перетворення: перетворення Фур'є; косинус; пазуха; Трансформація Адамара та ін.

Цифрове зображення містить дуже великий обсяг комп’ютерної інформації, яку зазвичай необхідно обробити за один раз. У типовому цифровому зображенні, що містить 512 рядків по 512 пікселів кожен, для кожного пікселя потрібно 8 біт (1 байт) машинної пам’яті. Це відповідає 262 144 байтам пам'яті комп'ютера. Оскільки для порівняння часто необхідно зберігати в пам’яті кілька зображень одночасно, необхідна пам’ять може бути в кілька разів більшою. З цією метою більшість систем обробки зображень мають два різні пристрої цифрової пам’яті: пам’ять цифрових зображень для зберігання одного чи кількох цифрових зображень, що підлягають обробці, і основну пам’ять, яка використовується для зберігання програм та інших даних.

Якщо головний комп’ютер не може зберегти все цифрове зображення у власній пам’яті, він переміщує невеликі частини зображення між пристроєм зберігання даних і власною пам’яттю для запуску програм обробки. При великих зображеннях або недостатніх ресурсах комп'ютера цей процес може тривати досить довго. затримка, спричинена обмеженнями головного комп’ютера, може бути неприйнятною, якщо потрібна обробка зображень у реальному часі.

Набагато більш швидку обробку можна реалізувати в комп'ютерах спеціального призначення з використанням спеціальних матричних або відеопроцесорів, здатних обробляти дуже великі цифрові масиви. Такі процесори можуть дуже швидко виконувати обмежені типи арифметичних або логічних

операцій над даними в своїй пам’яті. У разі використання матричного процесора для обробки цифрових зображень головний комп’ютер лише керує його роботою, а не безпосередньо керує пам’яттю зображень.

Як правило, матричний процесор і керована ним пам'ять цифрових зображень разом з додатковими областями пам'яті для зберігання таблиць пошуку та інших даних, що використовуються в процесі обробки цифрових зображень, об'єднані в один пристрій. Фізичні пристрої, які використовуються для зберігання цифрових зображень, але не працюють з ними, називаються буферами кадрів.

Етапи обробки цифрових зображень можна розділити на дві широкі категорії: методи, вхідними та вихідними даних яких є зображення, і методи, вхідними даними яких можуть бути зображення, але вихідними є атрибути, отримані з цих зображень.

Отримання зображень є першим процесом обробки цифрових зображень. Зауважимо, що отримання може бути таким же простим, як отримання зображення, яке вже є в цифровій формі. Як правило, етап отримання зображення включає попередню обробку, таку як масштабування.

Наступним кроком є покращення зображення, яке є однією з найпростіших і найпривабливіших областей цифрової обробки зображень. По суті, ідея методів покращення полягає в тому, щоб виділити деталі, які затемнені, або просто виділити певні цікаві риси зображення. Знайомий приклад покращення – коли ми збільшуємо контрастність зображення, тому що «воно виглядає краще». Важливо пам’ятати, що покращення – це дуже суб’єктивна сфера обробки зображень.

Відновлення зображень – це сфера, яка також стосується покращення зовнішнього вигляду зображення. Однак, на відміну від покращення, яке є суб’єктивним, відновлення зображення є об’єктивним у тому сенсі, що методи відновлення, як правило, базуються на математичних або імовірнісних моделях погіршення зображення. Покращення, з іншого боку, базується на суб’єктивних уподобаннях людини щодо того, що є «хорошим» результатом покращення.

Обробка кольорових зображень – це сфера, яка набуває все більшого значення через значне зростання використання цифрових зображень в Інтернеті. Обробка кольорових зображень передбачає вивчення основних понять у колірних моделях і базову обробку кольорів у цифровій області. Колір зображення можна використовувати як основу для виділення цікавих особливостей зображення [9].

Вейвлети є основою для представлення зображень із різними ступенями роздільної здатності. Зокрема, вейвлети можна використовувати для стиснення даних зображення та для пірамідального представлення, у якому зображення послідовно поділяються на менші області.

Стиснення, як випливає з назви, має справу з методами зменшення пам’яті, необхідної для збереження зображення, або пропускної здатності, необхідної для його передачі Хоча за останнє десятиліття технологія зберігання значно покращилася, цього не можна сказати про пропускну здатність. Особливо це стосується використання Інтернету, яке характеризується значним графічним вмістом. Стиснення зображень знайоме (можливо, ненавмисно) більшості користувачів комп’ютерів у формі розширень файлів зображень, таких як розширення файлу jpg, яке використовується в стандарті стиснення зображень JPEG (Joint Photographic Experts Group).

Морфологічна обробка стосується інструментів для вилучення корисних компонентів зображення у представленні та описі форми. Морфологічна обробка зображень є початком переходу від процесів, які виводять зображення, до процесів, які виводять атрибути зображення. Процедури сегментації поділяють зображення на його складові частини або об’єкти. Загалом, автономна сегментація є одним із найскладніших завдань у цифровій обробці зображень.

Жорстка процедура сегментації наближає процес до успішного вирішення проблем із зображенням, які вимагають індивідуальної ідентифікації об’єктів. З іншого боку, слабкі або помилкові алгоритми сегментації майже завжди гарантують кінець кінцем невдачу. Загалом, чим точніша ця сегментація, тим більша ймовірність успіху розпізнавання. Подання та опис майже завжди

слідують за виходом етапу сегментації, який зазвичай є необробленими піксельними даними, що становлять або межу області (тобто набору пікселів, що розділяють одна область зображення з іншої) або всі точки в самій області. У будь-якому випадку необхідно перетворити дані у форму, придатну для комп’ютерної обробки. Перше рішення, яке необхідно прийняти, полягає в тому, чи дані повинні бути представлені як межа чи як повна область. Зображення меж є доцільним, коли у центрі уваги знаходяться характеристики зовнішньої форми, такі як кути та вигини.

Регіональне представлення є доцільним, коли фокус зосереджений на внутрішніх властивостях, таких як текстура або скелетна форма. У деяких програмах ці подання доповнюють одне одного. Вибір представлення є лише частиною рішення для перетворення необроблених даних у форму, придатну для подальшої комп’ютерної обробки. Необхідно також вказати метод для опису даних, щоб виділити цікаві характеристики. Опис, також званий вибором ознак, має справу з вилученням атрибутів, які призводять до деякої цікавої кількісної інформації або є базовими для диференціації одного класу об’єктів від іншого.

Розпізнавання – це процес, який призначає мітку (наприклад, «транспортний засіб») об’єкту на основі його дескриптори. Тема «Розпізнавання» стосується методів розпізнавання окремих об’єктів на зображенні [10].

Обробку зображень можна використовувати для покращення якості зображення, видалення небажаних об’єктів із зображення або навіть створення нових зображень з нуля. Наприклад, за допомогою обробки зображень можна видалити фон із зображення людини, залишивши лише об’єкт на передньому плані.

Обробка зображень – це величезна та складна галузь із багатьма різними алгоритмами та методами, які можна використовувати для досягнення різних результатів. У цьому розділі ми зосередимося на деяких із найпоширеніших завдань обробки зображень і способах їх виконання.

Одним із найпоширеніших завдань обробки зображень є покращення зображення або покращення якості зображення. Він має ключове застосування в задачах комп'ютерного зору, дистанційного зондування та спостереження. Одним із поширених підходів є налаштування контрастності та яскравості зображення.

Контраст – це різниця в яскравості між найсвітлішою та найтемнішою ділянками зображення. Збільшуючи контрастність, можна збільшити загальну яскравість зображення, що полегшує його перегляд. Яскравість — це загальна освітленість або темність зображення. Збільшуючи яскравість, зображення можна зробити світлішим, що полегшить його перегляд. Як контрастність, так і яскравість можна налаштувати автоматично за допомогою більшості програм для редагування зображень або їх можна налаштувати вручну [11].

Однак налаштування контрастності та яскравості зображення є елементарними операціями. Іноді зображення з ідеальним контрастом і яскравістю при підвищенні масштабу стає розмитим через меншу кількість пікселів на квадратний дюйм (щільність пікселів). Щоб вирішити цю проблему, використовується відносно нова та набагато вдосконаленіша концепція супер-роздільності зображення, згідно з якою зображення високої роздільної здатності отримують із аналогів із низькою роздільною здатністю. Для цього широко використовуються методи глибокого навчання.

Наприклад, найпершим прикладом використання глибокого навчання для вирішення проблеми супер-роздільності є модель SRCNN, де зображення з низькою роздільною здатністю спочатку масштабується за допомогою традиційної бікубічної інтерполяції, а потім використовується як вхідні дані для моделі CNN. Нелінійне відображення в CNN виділяє ділянки, що перекриваються, із вхідного зображення, а шар згортки накладається на витягнуті ділянки для отримання реконструйованого зображення високої роздільної здатності [12].

Ще одним завданням обробки є відновлення зображень. Якість зображень може погіршитися з кількох причин, особливо фотографій з епохи, коли хмарне

зберігання не було таким звичним явищем. Наприклад, на зображеннях, відсканованих із друкованих копій, зроблених за допомогою старих миттєвих камер, часто з’являються подряпини.

Реставрація зображень особливо захоплююча, оскільки передові методи в цій галузі можуть потенційно відновити пошкоджені історичні документи. Потужні алгоритми відновлення зображень на основі глибокого навчання можуть виявити великі фрагменти відсутньої інформації з розірваних документів.

До цієї категорії, наприклад, відноситься малювання зображень, і це процес заповнення відсутніх пікселів у зображенні. Це можна зробити за допомогою алгоритму синтезу текстур, який синтезує нові текстури для заповнення відсутніх пікселів. Однак моделі на основі глибокого навчання є фактичним вибором через їхні можливості розпізнавання образів.

Сегментація зображення – це процес поділу зображення на кілька сегментів або областей. Кожен сегмент представляє інший об’єкт на зображенні, і сегментація зображення часто використовується як етап попередньої обробки для виявлення об’єкта.

Існує багато різних алгоритмів, які можна використовувати для сегментації зображення, але одним із найпоширеніших підходів є використання порогових значень. Двійкове порогове визначення, наприклад, є процесом перетворення зображення у двійкове зображення, де кожен піксель є чорним або білим. Порогове значення вибирається таким чином, що всі пікселі з рівнем яскравості нижче порогового значення стають чорними, а всі пікселі з рівнем яскравості вище порогового значення стають білими. Це призводить до того, що об’єкти на зображенні сегментуються, оскільки тепер вони представлені чіткими чорними та білими областями.

Сучасні методи використовують автоматизовані алгоритми сегментації зображень із використанням глибокого навчання як для бінарних, так і для задач сегментації з кількома мітками. Наприклад, PFNet або Positioning and Focus Network – це модель на основі CNN, яка вирішує проблему сегментації

замаскованих об’єктів. Він складається з двох ключових модулів: модуля позиціонування (PM), призначеного для виявлення об’єктів (імітує хижаків, які намагаються визначити грубе положення здобичі); і модуль фокусування (FM), призначений для виконання процесу ідентифікації в хижацтві для уточнення початкових результатів сегментації шляхом фокусування на неоднозначних областях.

Впровадження методів обробки зображень мало величезний вплив на багато технологічних організацій. Ось деякі з найбільш корисних переваг обробки зображень, незалежно від сфери діяльності [13]:

- цифрове зображення може бути доступним у будь-якому бажаному форматі (покращене зображення, рентгенівський знімок, фотонегатив тощо);
- це допомагає покращити зображення для інтерпретації людиною;
- інформацію можна обробляти та витягувати із зображень для машинної інтерпретації;
- пікселі на зображенні можна регулювати до будь-якої бажаної щільності та контрастності;
- зображення можна легко зберігати та отримувати;
- це дозволяє легко передавати зображення в електронному вигляді стороннім постачальникам.

Отже, наразі є велика кількість завдань, що вирішують методи обробки зображень у багатьох сферах. В даному підрозділі розглянуто деякі з найважливіших методів обробки зображень і популярні методи на основі глибокого навчання, які вирішують завдання, від стиснення й покращення зображення до синтезу зображення.

1.3 Огляд відомих методів, що застосовуються для обробки зображень

Використання певних методів, налаштувань і режимів часто базується на досвіді, суб'єктивному сприйнятті та художніх здібностях.

Серед процесів обробки зображень можна виділити дві групи [14]:

– попозиційна обробка;

– кореляційна обробка.

При використанні методів обробки першої групи результат обробки будь-якої точки кадру зображення залежить тільки від опорного значення характерного параметра первинного зображення в цій же точці. Очевидною перевагою таких процедур є простота виконання, вони призводять до значного суб'єктивного поліпшення якості зору. Елементарна обробка зображень використовується як завершальний крок у складному процесі обробки зображень.

Друга група процедур заснована на тому, що між елементами цілого зображення або його окремими фрагментами існують зв'язки - кореляція.

Для отримання результату перетворення для кожної точки зображення говоримо в даному випадку про дані про параметри певної множини точок первинного зображення, розташованих навколо оброблюваної точки.

Суть елементарної обробки зображення полягає у встановленні деякої функціональної залежності між кінцевим еталонним значенням сигналу зображення та його первинним значенням або статистичною характеристикою.

Нехай $x(i,j) = x_{i,j}$ та $y(i,j) = y_{i,j}$ – значення яскравості вхідного та вихідного сигналів зображення в точці, яка має координати, що відповідають i-му відліку в напрямку вертикальної осі та j-му відліку в напрямку горизонтальної осі.

Поелементна обробка заснована на тому, що між значеннями яскравості існує однозначна функціональна залежність [15]:

$$y_{i,j} = f_{i,j}(x_{i,j}), \tag{1.1}$$

що дає можливість визначити значення кінцевого сигналу за значеннями первинного сигналу.

Параметри функції $f_{i,j}(\cdot)$, що описують процес обробки, можуть залежати від поточних координат. Якщо така залежність існує, то виконана обробка вважається неоднорідною.

У більшості процедур, які знайшли практичне застосування, проводять поелементну рівномірну обробку, при цьому індекси i і j виразу (1.1) можуть бути відсутні:

$$y = f(x). \tag{1.2}$$

Залежність між яскравістю елементів вхідного та вихідного зображень буде визначатися функцією (1.2), яка є однаковою для всіх точок зображення.

Елементарні процедури обробки можна розділити на:

– процедури, пов’язані зі зміною контрастності зображення;

– процедури бінаризації зображення.

Контрастність зображення – це параметр, рівний відношенню від максимальної до мінімальної яскравості в полі зображення.

Контрастність зображення – параметр, який визначає різницю яскравості від середнього рівня і чисельно дорівнює відношенню різниці між максимальною і мінімальною яскравістю до їх суми [16]. Часто кінцеве значення вказується у відсотках.

Основним завданням контрастного методу є покращення межі налаштування між динамічним діапазоном цифрового зображення та екраном, на якому виконується візуалізація. Наприклад, для цифрового відображення кожного з показань зображення дозволено 1 біт пам’яті пристрою, з яким ми працюємо, тоді вихідні або вхідні сигнали приймають лише одне значення 0..255.Тому те, що слідує за діапазоном (0..255) використовується як робочий діапазон; значення 0 відповідає рівню чорного, а 255 відповідає білому кольору.

Здійснення комп’ютерної обробки зображень можливе після трансформування сигналу зображення з аналогової у цифрову форму. Забезпечення ефективності обробки залежить від адекватності моделі, що описує зображення, необхідної для розробки алгоритмів обробки. Моделлю зображення називають функціональну систему, що описує основні характеристики зображення: функцію яскравості, що зображає зміну яскравості в площині

зображення, просторові спектри та спектральну інтенсивність зображень, функцію автокореляції. Канал зображення містить оптичну систему, оптико-електричний перетворювач, аналого-цифровий перетворювач (АЦП) та цифрову обробку сигналу зображення [17].

Існує кілька класифікацій методів, але більшість із них базуються на наступних двох властивостях сигналу яскравості – розривності та однорідності.

Методи, засновані на яскравості, включають визначення лінійної точки та різниці. При пошуку точок і ліній за допомогою спеціальних масок організовується відповідний пошук. Похідні та градієнти функцій яскравості використовуються як методи розрізнення відмінностей. Ці методи базуються на більш загальних ідеях. На даний момент основні методи сегментації зображень включають наступні класи (методи згруповані від найпростіших до найбільш трудомістких):

1. Морфологічні методи – в основному використовуються для роботи з бінарними зображеннями (чорно-білими).

2. Граничні методи – мають інтуїтивно зрозумілі властивості та прості у реалізації. Існує кілька основних типів сегментації за пороговим значенням, але лише два основних типи: метод оптимального порогового значення та метод адаптивного порогового значення.

3. Методи створення області – це алгоритми, які рекурсивно виконують процес групування пікселів в області відповідно до заздалегідь визначених критеріїв.

4. Текстурні методи – засновані на аналізі дифузних властивостей (колір, відображення) поверхні об'єкта. Методи, представлені в цій категорії, являють собою набори складних операторів, які можуть звести процес виявлення поверхні до простого завдання розрізнення рівнів яскравості.

Цифрові зображення можуть бути чутливі до різних типів шуму, який може виникнути внаслідок методів захоплення зображення, технологій передачі інформації та методів оцифрування даних. Видалення різного роду шумів із зображень називається фільтрацією [18]. Фільтрація замінює властивості

яскравості кожної точки зображення в цифровому форматі різним значенням яскравості, яке визнається найменш спотвореною перешкодою. Існує частотна та просторова фільтрація.

Частотні методи перетворень зображень засновані на ідеї перетворення Фур'є, зміст якої полягає у поданні вихідної функції як суми тригонометричних функцій різних частот, помножених на задані коефіцієнти. Якщо функція періодична, таке подання називається рядом Фур'є. В іншому випадку неперіодична функція з кінцевою площею під діаграмою може бути виражена як інтеграл тригонометричних функцій, помножений на вагову функцію. Цей варіант називається перетворенням Фур'є і в більшості практичних задач корисніший за ряд Фур'є. Важливою властивістю є те, що функція, представлена перетворенням Фур'є, може бути повернута до початкового стану після перетворень. Це дозволяє вам обробити функцію в частотній області, а потім повернутися у початковий стан, не втрачаючи жодної інформації. Перетворення Фур'є також можуть бути використані для вирішення проблем фільтрації зображень.

У практичному застосуванні реалізація частотних підходів може бути подібною до методів просторової фільтрації [19]. Методи просторового покращення зображення застосовуються до цифрових зображень, які представлені у вигляді двовимірних матриць. Принцип просторових алгоритмів полягає у застосуванні спеціальних операторів до кожної точки вихідного зображення. Оператори - це прямокутні або квадратні матриці, які називаються масками, ядрами або вікнами. Найчастіше маска являє собою невеликий двовимірний масив, і методи вдосконалення, засновані на цьому підході, часто називають обробкою маски або фільтруванням маски. Просторова фільтрація є найкращим рішенням у випадках, коли присутня лише адитивна складова шуму.

Моделі імовірнісних образів часто застосовують для опису зображень. У цьому випадку зображення аналізується у вигляді випадкової функції просторових координат (x, y) і часу t. Випадковий процес називають стаціонарним у ширшому розумінні, якщо він має постійні значення

математичного сподівання та дисперсії, в той час коли функція автокореляції залежить не від координат, а від їх різниці. Випадковий процес називають нерухомим, якщо його n-мірна щільність ймовірності переміщення є постійною. Випадковий процес описується розподілом щільності ймовірності яскравості на зображенні за просторовими координатами протягом фіксованого часу t.

Після визначення математичного сподівання (середнє, математичне очікування), стаціонарний процес обчислюється в ширшому розумінні

$$Mf = \xi = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y)p(x,y)dxdy = const. \qquad (1.3)$$

Аналіз спектра зображення широко використовується при обробці зображень. Спектр зображення можна отримати, використовуючи перетворення Фур'є.

$$F(u,v) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} I_1(x,y)e^{-j(ux+vy)}dxdy, \qquad (1.4)$$

Функція $e^{-j(ux+vy)}dxdy$ при сталому значенні просторових частот зображає плоску хвилю в площині зображення. За використання такого аналізу є можливість визначити також амплітуду і фазу спектрів. Обернене перетворення Фур'є допомагає відновити зображення за його спектром, а інтенсивність зображення відповідає розподілу енергії по просторових частотах.

Безперервне зображення (I (x, y)) – коли координати x і y можуть приймати будь-які значення, а дискретним – коли x і y визначені лише для певного набору значень. Як правило, зразок зображення знаходиться не в часовій області, а в просторі, і частота дискретизації вказує, скільки зразків зображення вимірюється на одиницю довжини в кожній координаті. Величина частоти дискретизації в цьому випадку становить 1 / M, а крок дискретизації виражається в одиницях довжини.

$$\delta[k,l] = \begin{cases} 1, k = 0, l = 0, \\ 0, k \neq 0, l \neq 0. \end{cases} \quad (1.5)$$

$$I_1[n,m] = \sum_{n=-\infty}^{+\infty} \sum_{m=-\infty}^{+\infty} I_1(x,y)\delta(k-n, l-m), k, l \in Z, \quad (1.6)$$

Реальні програми дискретного образу містять кінцеву кількість елементів, які звуться пікселями:

$$I_1[n,m], n = \overline{0, N-1}, m = \overline{0, M-1}\,. \quad (1.7)$$

Якщо для отримання яскравості пікселя обробленого зображення використовується яскравість лише одного пікселя вихідного зображення, це перетворення градації.

Зв'язок вхідних та вихідних зображень для корекції яскравості пікселів для різних значень гами (> 1 та <1) наведено на рисунку 1.2.

Гістограмою розподілу яскравості цифрового зображення називають дискретну функцію, що описує частоту рівня сірого на зображенні і представляється у вигляді графіки, абсциса якої містить кількість рівнів сірого у порядку зростання (інтенсивності) та містить вісь ординат – кількість сірих пікселів (частота інтенсивності) [20]. Гістограма має у складі n стовпців для напівтонового зображення n = 256.

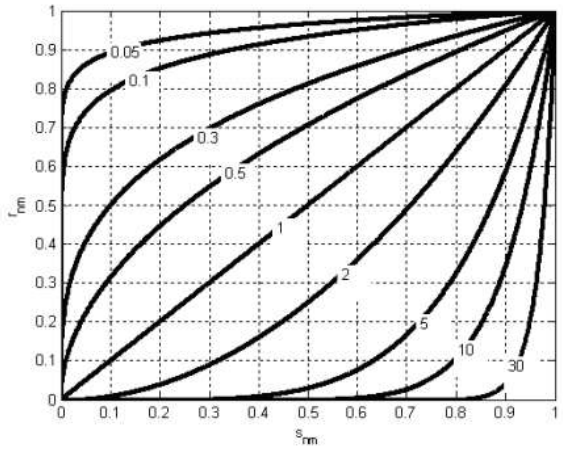


Рисунок 1.1 – Гамма-корекція зображення

З точки зору зорового сприйняття людини, оптимальними є зображення, елементи яких мають рівномірний розподіл світлих кольорів. Покращена ультразвукова візуалізація дозволяє досягти більш високого вирівнювання, яке забезпечує рівномірну здатність створювати яскраве зображення.

Для середньої фільтрації використовують сусідство відповідного пікселя вихідного зображення для отримання яскравості пікселя вихідного зображення. Порядок їх від найменшого до найбільшого визначається значеннями яскравості оточуючих пікселів. Для цієї послідовності існує медіана, тобто визначається, який піксель знаходиться в точці послідовності, що відповідає половині його довжини. Наприклад, якщо є близько 9 пікселів, медіаною є піксель, який є п'ятим у рейтингу. Яскравість цього пікселя – це значення яскравості пікселів відфільтрованого зображення [21].

1.4 Аналіз відомих програмних реалізацій для обробки зображень

Існує велика кількість застосувань обробки зображень у різноманітному спектрі людської діяльності – від інтерпретації сцен дистанційного зондування до інтерпретації біомедичних зображень. Програми обробки зображень допомагають отримати деякі кількісні властивості.

Програми для обробки зображень відрізняються за сферами їх застосування. Наприклад, існують програми, які використовують в медицині, для сфери дорожнього руху, для відновлення зображень та розпізнавання облич.

Обробка зображень широко використовується в медичних дослідженнях і дозволяє складати більш ефективні та точні плани лікування. Наприклад, його можна використовувати для раннього виявлення раку молочної залози за допомогою складного алгоритму виявлення вузликів під час сканування молочної залози. Оскільки медичне використання потребує

висококваліфікованих процесорів зображень, ці програми потребують значного впровадження та оцінки, перш ніж їх можна буде прийняти до використання.

У випадку датчиків дорожнього руху ми використовуємо систему обробки відеозображення або VIPS. Він складається з а) системи захоплення зображення, б) телекомунікаційної системи та в) системи обробки зображень. Під час зйомки відео VIPS має кілька зон виявлення, які видають сигнал «увімкнено», коли транспортний засіб в'їжджає в зону, а потім видають сигнал «вимкнено», коли транспортний засіб виходить із зони виявлення. Ці зони виявлення можуть бути налаштовані для кількох смуг і використовуватися для виявлення руху на конкретній станції. Крім того, він може автоматично записувати номерний знак транспортного засобу, розрізняти тип транспортного засобу, стежити за швидкістю водія на трасі та багато іншого.

Обробку зображення можна використовувати для відновлення та заповнення відсутніх або пошкоджених частин зображення. Це передбачає використання систем обробки зображень, які пройшли інтенсивне навчання з наявними наборами даних фотографій для створення нових версій старих і пошкоджених фотографій.

Одним із найпоширеніших застосувань обробки зображень, які ми використовуємо сьогодні, є розпізнавання облич. Він дотримується алгоритмів глибокого навчання, коли машина спочатку навчається конкретним характеристикам людських облич, таким як форма обличчя, відстань між очима тощо. Після навчання машини цим рисам людського обличчя вона почне приймати всі предмети на зображенні, які нагадують людське обличчя. Розпізнавання обличчя є життєво важливим інструментом, який використовується для безпеки, біометрії та навіть фільтрів, доступних у більшості програм соціальних мереж.

Наведемо приклад застосунків, які використовують для обробки зображень.

IC Measure – програмне забезпечення, що відображає тональний розподіл кольорів не лише зображень, але і відео. Крім того, воно також підтримує зображення різних форматів [22].

IC Measure – це безкоштовне програмне забезпечення для гістограм зображень для Windows. В основному це програмне забезпечення для вимірювання зображень, за допомогою якого користувачі можуть вимірювати відстань та форми, присутні на зображеннях. Крім того, його також можна використовувати для аналізу різних аспектів відео- та аудіофайлів. Тепер для візуалізації та аналізу тонального розподілу кольорів, присутніх на зображенні, користувачі можуть використовувати його вбудовану гістограму. На відміну від більшості інших програм для створення гістограм зображень, це програмне забезпечення також дозволяє користувачам переглядати в режимі реального часу тональний розподіл кольорів відео на його гістограмі. Ще однією хорошою особливістю цього програмного забезпечення є його здатність підтримувати зображення різних форматів, таких як BMP, JPEG, JFIF, PNG, TIFF тощо. Тепер дізнайтеся про основні функції цього програмного забезпечення. Приклад роботи даного програмного забезпечення наведено на рисунку 1.2.

Основними рисами прогрмами є наявність роботи з гістограмами: Користувачі можуть відкрити гістограму, перейшовши на вкладку Вид. Використовуючи цю гістограму, користувачі можуть переглядати значення розподілу кольорів RGB, присутні на зображенні, у вигляді гістограми. За замовчуванням ця гістограма використовує лінійну шкалу для відображення розподілу кольорів. Хоча користувачі можуть вибрати логарифмічну шкалу для перегляду розподілу кольорів RGB зображення.

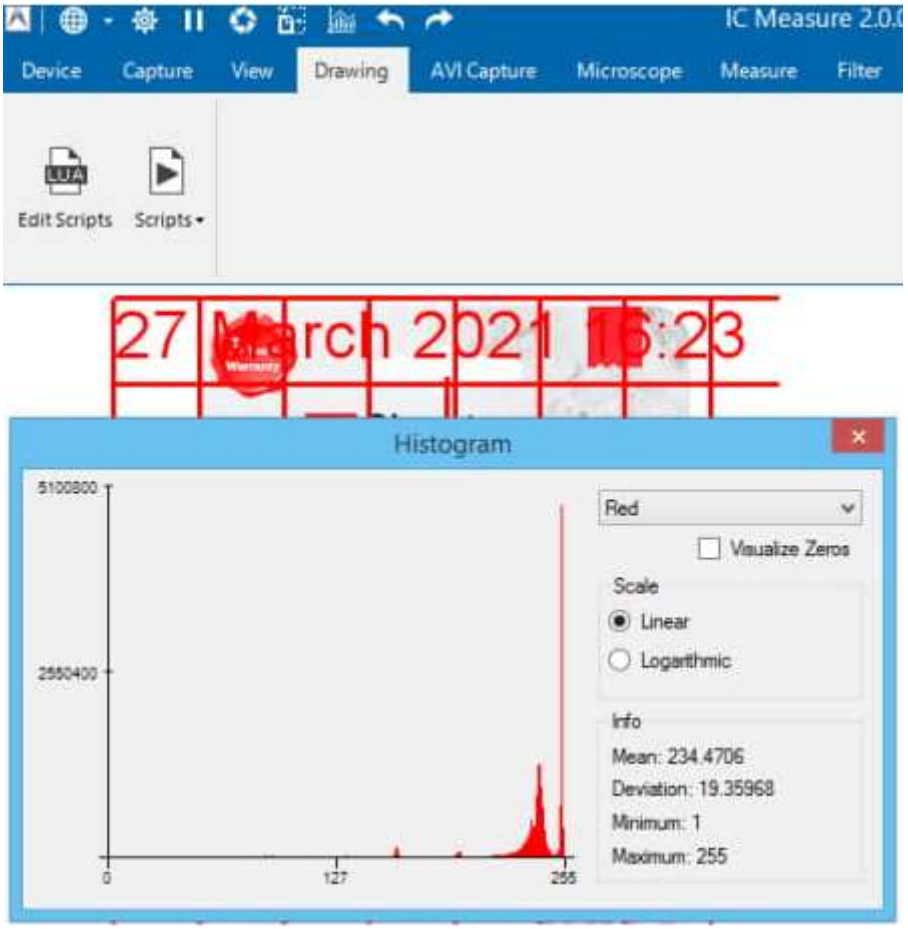


Рисунок 1.2 – Загальний вигляд інтерфейсного вікна програми IC Measure

Інформація гістограми показує середнє значення та відхилення значень тонального розподілу кольору.

Такоє в програмі присутня підтримка послідовності відео та зображень: У цьому програмному забезпеченні користувачі можуть також вводити послідовність зображень та відеофайлів для перегляду та аналізу значень розподілу кольорів.

GemIdent (рис. 1.3) – це інтерактивна програма розпізнавання зображень, яка визначає цікаві області на зображеннях і фотографіях [23]. Він спеціально розроблений для зображень із невеликою кількістю кольорів, де цікаві об’єкти виглядають схожими з невеликими варіаціями.

GemIdent також містить інструменти аналізу даних для дослідження просторових зв’язків між ідентифікованими об’єктами.

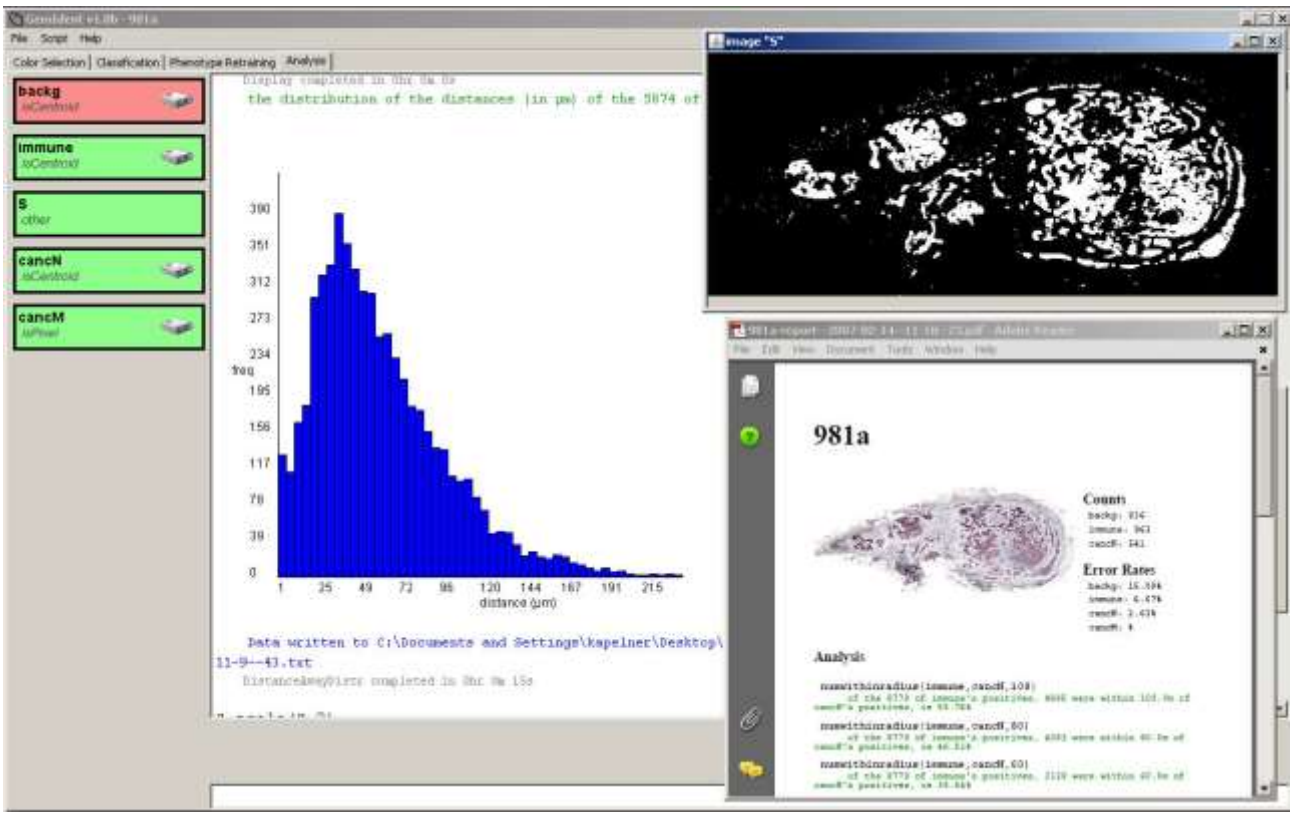


Рисунок 1.3 – Загальний вигляд інтерфейсного вікна програми GemIdent

GemIdent використовує контрольоване навчання для автоматизованого визначення цікавих областей на зображеннях. Таким чином, користувач повинен виконати значний обсяг роботи, спочатку надавши відповідні кольори, а потім вказавши приклади самих об’єктів або регіонів, а також негативи (створення навчального набору).

Коли користувач натискає на піксель, генерується багато балів, використовуючи інформацію про навколишній колір за допомогою генерації атрибута Mahalanobis Ring Score. Потім ці оцінки використовуються для створення випадкового класифікатора лісу машинного навчання, який потім класифікуватиме пікселі на будь-якому зображенні.

Після класифікації можуть бути помилки. Користувач може повернутися до навчання і вказати на конкретні помилки, а потім перекласифікувати. Ці ітерації навчання-класифікації-перепідготовки-перекласифікації (вважаються інтерактивним посиленням) можуть призвести до високоточної сегментації.

Jmicrovision (рис. 1.4) – програма аналізу зображень, що має наступні можливості [24]:

1. Дозволяє провести лінійні вимірювання і довжин довільних кривих.
2. Забезпечує вимірювання площ довільних фігур, та інших геометричних характеристик (овальність, периметр і т.д.).
3. Виділяє елементи структури за кольором.

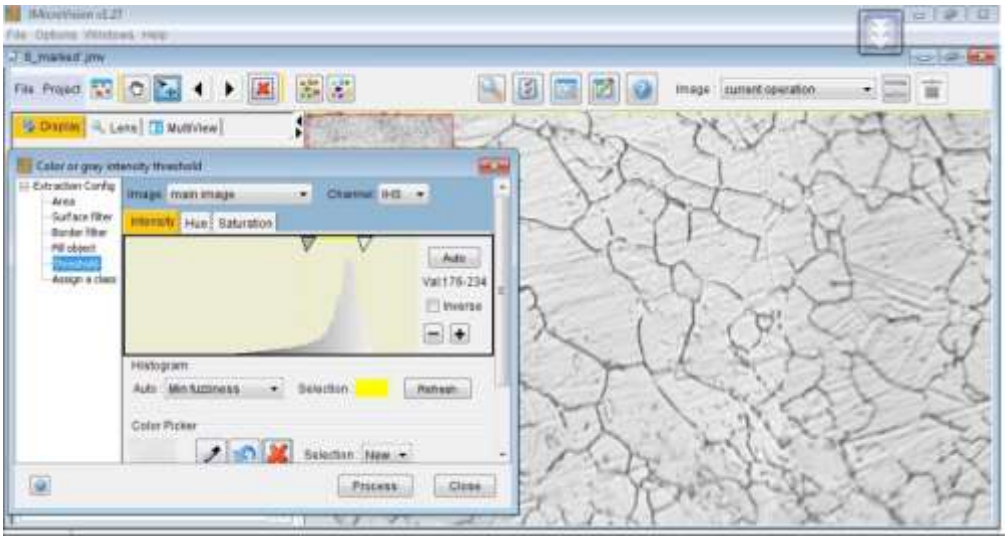


Рисунок 1.4 – Загальний вигляд інтерфейсного вікна програми Jmicrovision

Програмне забезпечення для спектроскопії ENLIGHTEN має всі функції, необхідні для швидкого та легкого збору спектрів комбінаційного розсіювання, поглинання, пропускання, відбиття та випромінювання зі спектрометрів Wasatch Photonics [25]. Його використання забезпечує легке створення безлічі дивовижних фрактальних образів (рис. 1.5).

Функції та можливості ENLIGHTEN:

- вимірювання в діапазоні, відображення/пропускання, поглинання;
- отримання даних з кількох спектрометрів одночасно;
- перегляд мініатюри останніх вимірювань на панелі «збережені спектри»;
- легке накладання збережених даних на живі вимірювання для легкого порівняння;

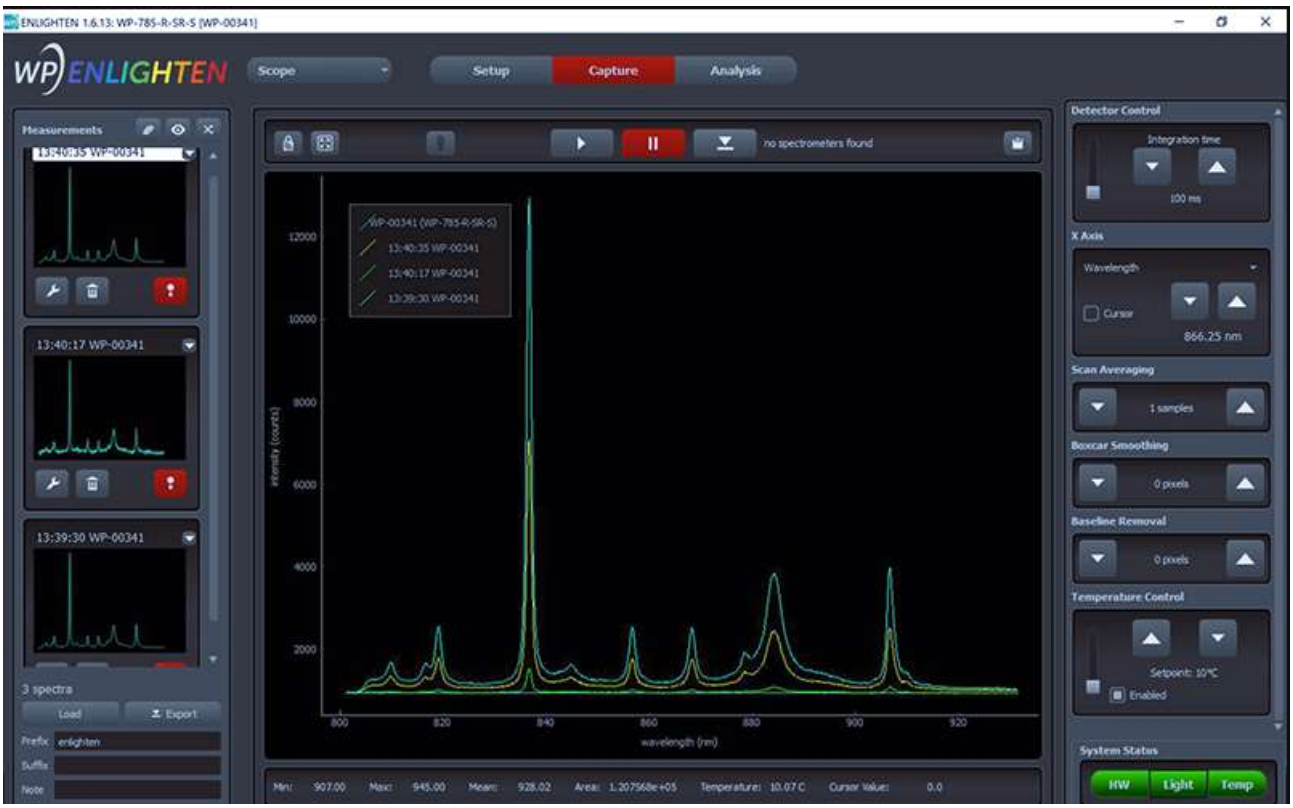


Рисунок 1.5 – Загальний вигляд інтерфейсного вікна програми ENLIGHTEN

- копіювання та вставка даних з буфера обміну в Excel та інші програми;
- гнучкі та інтуїтивно зрозумілі налаштування дисплея, а також курсор і пошук піків;
- пакетний збір даних, безперервний або плановий;
- корекція базової лінії за фіксовану область інтересу для флуоресціюючих зразків комбінаційного розсіювання;
- керування інтегрованими лазерами за допомогою програмного забезпечення.

Photomania – це ще одне безкоштовне програмне забезпечення для гістограм зображень для Windows [26]. Це ще одне хороше програмне забезпечення для гістограм зображень, яке показує комбінований розподіл кольорів RGB зображення. Окрім цього, він також дозволяє користувачам переглядати розподіл окремих кольорів зображення, наприклад розподіл червоного кольору, розподіл синього кольору тощо. Крім того, він також забезпечує підтримку зображень різних форматів, таких як JPG, BMP, PNG, TIF,

ICO , і більше. Користувачі також можуть використовувати це програмне забезпечення для перегляду та покращення вхідних зображень. Приклад роботи програми наведено на рисунку 1.6

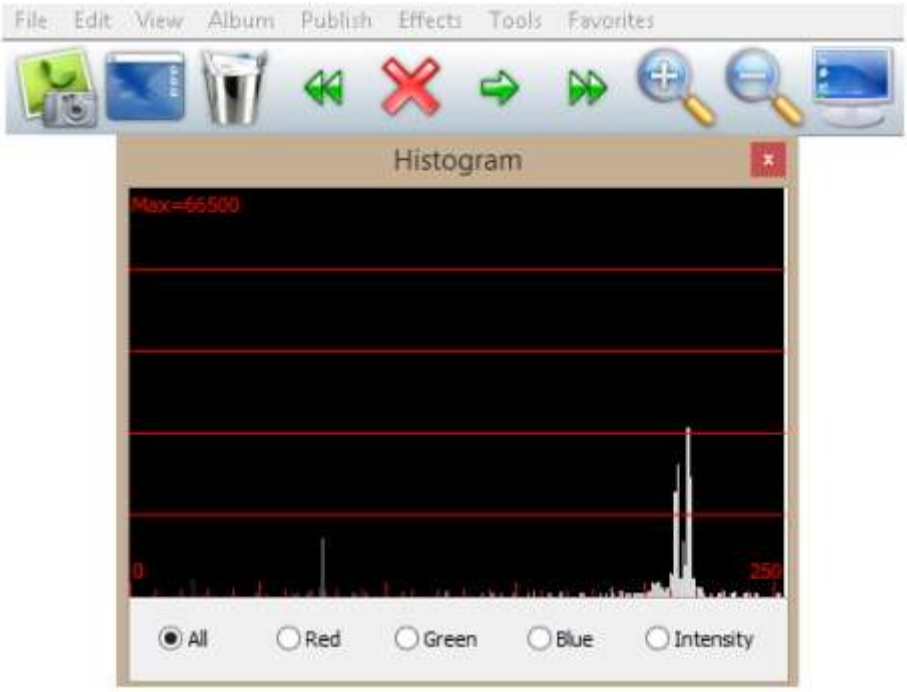


Рисунок 1.6 – Приклад роботи програми Photomania

Тепер розглянемо основні функції цього програмного забезпечення:

- Інструмент гістограми: цей інструмент присутній у розділі Інструменти. Після завантаження зображення користувачі можуть запустити цей інструмент, щоб переглянути розподіл кольорів RGB у вигляді гістограми. Це також дозволяє користувачам переглядати інтенсивність окремих кольорів RGB за допомогою параметра Intensity.

- Ефекти: за допомогою ефектів зображення користувачі можуть регулювати яскравість, насиченість, колірну температуру тощо властивості зображення. Він також пропонує інструменти для застосування попередньо визначених ефектів (зернистість плівки, рельєф, мозаїка, подвійне бачення тощо) до вхідних зображень.

- Слайд-шоу: використовуйте для створення та запуску слайд-шоу, що складається з власних зображень.

- Експорт зображень у PDF: за допомогою нього користувачі можуть швидко конвертувати зображення різних форматів у PDF.

Серед проаналізованих програмних систем, IC Measure, Jmicrovision та Photomania є максимально наближеними за своїми характеристиками до розроблюваної інтелектуальної системи обробки зображень. Отже, необхідно обрати ці системи як прототип.

1.5 Постановка задачі дослідження

Робота присвячена реалізації інформаційної технології інтелектуальної обробки зображень. Отже, необхідно програмно вирішити задачу обробки зображень. Інформаційна технологія розроблена для додатків, які спеціалізуються на обробці зображень і потребують швидкого інструменту для перетворення та аналізу даних. Таким чином, інформаційна технологія обробляє зображення, реалізує гістограми RGB-каналів та яскравість для подальшої обробки фотографій або використання цієї технології у фото-програмах для кращого автофокусування. Інформаційна технологія також містить обробку зображень, яка полягає у зміні яскравості зображення, щоб мати можливість продовжувати працювати з його обробкою.

Дана інформаційна технологія повинна містити зручну та зрозумілу реалізацію для кінцевого користувача, не потребувати спеціальних навичок та знань у галузі алгоритмів обробки зображень. Вхідні дані повинні бути імпортовані в програму у вигляді вхідного зображення, а вихідні дані повинні характеризуватися чіткістю подання. Максимально можлива швидкість обробки зображень важлива для користувача створеної інформаційної технології. Інформаційна технологія повинна автоматично обробляти вхідну інформацію та виводити результат у доступному форматі.

Як вже зазначалося, розроблена інформаційна технологія в основному орієнтований на швидкість. Швидкість повинна бути вищою, ніж у аналогових

програм. Це показник хорошої роботи інтелектуальної системи і, отже, можливості її подальшого використання.

Основними вимогами до інформаційної технології слід зазначити наявність головної сторінки з меню, можливість перегляду гістограми вхідного зображення, можливість вибору каналу, по якому слід будувати гістограму, можливість завантаження зображення безпосередньо з програми.

Отже, потрібно розробити інформаційну технологію, в якій буде реалізовано алгоритм обробки зображення.

1.5 Висновок до розділу 1

В даному розділі було досліджено особливості застосування обробки зображень та доцільність розробки інформаційної технології інтелектуальної обробки зображень. Також розглянуто проблеми, присутні в сучасних методах обробки зображень. Було проведено аналіз сучасних програм-аналогів, які використовуються для обробки зображень та побудови гістограм. Було наведено короткий опис основних функцій, які виконують дані програми. Також було досліджено методи, що можуть бути використані для обробки зображень, для чого запропоновано використати методи аобудови гістограми зображення за допомогою аналізу спектрів та ортогональних операторів Собеля. Здійснено постановку задачі.

2 МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ІНТЕЛЕКТУАЛЬНОЇ ОБРОБКИ ЗОБРАЖЕНЬ

2.1 Обґрунтування вибору методу обробки зображень

Більшість сучасних наукових досліджень у різних галузях використовують результати вимірювань як первинні дані. При цьому діапазон можливих методів вимірювання постійно розширюється, зокрема завдяки стрімкому розвитку технічних засобів збору, передачі, обробки та зберігання даних. Забезпечення автоматизованого збору та обробки великих обсягів інформації дозволило за останні роки значно розширити сферу застосування методів вимірювань, заснованих на реєстрації та поглибленому аналізі зображень. Незважаючи на те, що зображення містять значний обсяг інформації про досліджуваний об’єкт або процес, їх використання в метрологічній практиці потребує синтезу відповідних інформаційних технологій для їх обробки, аналізу та інтерпретації.

Різноманітність об'єктів, дослідження яких базується на отриманні візуальних даних, на даний момент не дозволяє запропонувати універсальні методи обробки зображень і виділення їх інформативних характеристик. Тому при вирішенні багатьох прикладних задач використовуються візуальні оцінки характеристик зображення. Зрозуміло, що такі оцінки мають суб'єктивний характер і обмежені можливості отримання кількісних значень інформативних характеристик. Алгоритми аналізу зображень часто будуються як автоматизоване програмне забезпечення, аналогічне експертним оцінкам.

Для виявлення окремих деталей або фрагментів зображень розроблено низку методів забарвлення, граничної різкості, а також алгоритмів нелінійного підвищення яскравості [27], але вони також можуть бути використані для обмеженого кола завдань.

Одним із найпоширеніших методів аналізу зображення є побудова гістограми зображення [28]. Гістограма яскравості зображення – це дискретна

функція, яка визначає кількість пікселів, що характеризується відповідною яскравістю для рівня яскравості.

Напівтонові зображення можна охарактеризувати на основі побудови однієї гістограми, яка охоплює всі пікселі зображення. Сьогодні в більшості програм обробки та аналізу зображень цифрові напівтонові зображення представлені 256 різними відтінками яскравості, які стандартно кодуються як цілі числа від 0 (чорний) до 255 (білий). Всі проміжні значення між крайніми 0 і 255 є відтінками сірого. Навпаки, кольорове зображення не можна представити на основі використання однієї гістограми. Для аналізу кольорових зображень зазвичай будують три гістограми, що відповідають палітрі RGB (тобто три гістограми розподілу яскравості для червоного, зеленого і синього кольорів) [29]. Окрім гістограм яскравості, також використовуються гістограми контрастності, які є дискретною функцією, що вказує на кількість пікселів з певним рівнем контрастності.

Перевагами використання гістограми для цифрової обробки та аналізу зображень є те, що вона забезпечує інтегральну оцінку параметрів зображення з урахуванням яскравості (або контрастності) кожного пікселя.

Гістограми є універсальним інструментом для аналізу, який можна застосувати практично до будь-якого цифрового зображення. Гістограми використовуються на різних етапах обробки візуальних даних: як на етапі попередньої обробки [30], так і на етапі виділення інформативних ознак та інтерпретації зображення. Однак, оскільки гістограма охоплює всі пікселі зображення, вона містить велику кількість параметрів, а саме 256 значень кількості пікселів з певною яскравістю. Це призводить до ускладнення можливості експертного аналізу гістограм людиною та виділення інформативних ознак із візуальних даних на етапі інтерпретації зображення.

Наприклад, при аналізі газорозрядних радіаційних зображень проб води, записаних на рентгенівську плівку, враховуються експертні оцінки самого зображення та отриманої гістограми. Оскільки 256 кількісних параметрів гістограми (ординати висоти її стовпців) не можуть бути проаналізовані

експертом, необхідно скоротити обсяг інформації. Зменшення обсягу кількісної інформації досягається поділом гістограми на окремі ділянки (ділянки). Недоліком цього методу є суб'єктивність і нечіткість поділу на зони.

Другим прикладом, який демонструє необхідність зменшення кількості кількісної інформації про інформативні властивості зображень, може служити завдання класифікації (поділ зображень на окремі класи або групи для об’єктів зі схожими властивостями). У цьому випадку кількісна інформація про 256 піксельних ординатах відповідної яскравості, отримана при побудові гістограми, може виявитися зайвою навіть для побудови автоматизованої системи програмної класифікації зображень матиме суттєво найменшу кількість числові інформативні ознаки [31].

Гістограма інтенсивності H зображення – це графік або таблиця всіх можливих значень інтенсивності (рівня сірого), розташованих у порядку зростання, і кількості пікселів зображення з відповідними значеннями інтенсивності. Таким чином, як набір, гістограма інтенсивності зображення може бути записана як $H = \{ h(k) \mid k = 0, 1, 2, \dots, L-1 \}$. Тут H – зведена гістограма, а $h(k)$ – значення гістограми при інтенсивності «k» (що є кількістю пікселів зображення $n(k)$, що мають значення інтенсивності «k»), а «L» – загальна кількість різні рівні інтенсивності, які може мати зображення для квантування інтенсивності, що використовується. Отже, $h(k) = n(k)$. Сукупна гістограма інтенсивності C зображення – це графік або таблиця всіх можливих значень інтенсивності, розташованих у порядку зростання, і кількості пікселів зображення, які мають інтенсивність до відповідного значення інтенсивності включно.

Таким чином, сукупна гістограма інтенсивності зображення може бути записана як $C = \{ c(k) \mid k = 0, 1, 2, \dots, L-1 \}$. Тут C – сукупна сукупна гістограма, а $c(k)$ – сукупне значення гістограми при інтенсивності «k» (що є загальною кількістю пікселів зображення зі значенням інтенсивності «k» або менше), а «L» – загальна кількість кількість різних рівнів інтенсивності, які може мати зображення для квантування інтенсивності, що використовується.

Трансформація інтенсивності в контексті цифрової обробки зображень – це техніка або операція точкової (піксельної) обробки, під час якої інтенсивності пікселів у зображенні перетворюються (перепризначаються) на різні інтенсивності на основі деякого відображення інтенсивності введення-виведення. Це робиться для того, щоб модифікувати вихідне (дане) зображення для досягнення бажаного кінцевого результату «покращення» оригінального зображення певним чином для конкретного застосування. Слід зазначити, що в перетворенні інтенсивності (точкова обробка/поліпшення зображення) існують різні підкатегорії, такі як розтягування (або модифікація) контрасту, вирівнювання гістограми та специфікація гістограми або збіг, щоб згадати декілька.

В результаті всіх поелементних перетворень відбувається зміна закону розподілу ймовірностей яскравості пікселя, який описує зображення. Розглянемо механізм цієї зміни на прикладі довільного перетворення монотонного характеру.

Нехай функція перетворення $y = f(x)$, а її єдина обернена функція $x = \varphi(y)$. Припустимо, що випадкова величина x розподілена відповідно до щільності ймовірності $\omega_x(x)$.

Нехай Δx – довільний малий інтервал значень випадкової величини x, а Δy – відповідний інтервал перетвореної випадкової величини y.

Пошук значення x в інтервалі Δx визначає пошук значення y в інтервалі Δy, тобто ймовірнісну еквівалентність цих двох подій.

Враховуючи малий розмір двох інтервалів, можна записати таку рівність:

$$w_x(x)|\Delta x| \approx w_y|\Delta Y|. \qquad (2.1)$$

Модулі y (2.1) враховують залежність ймовірностей від абсолютних довжин інтервалів (незалежність знаків Δx, Δy).

Щоб обчислити щільність ймовірності перетвореного значення, замінимо його вираз оберненою функцією замість x і виконаємо граничний перехід $\Delta x \to 0$, (що зумовлює $\Delta y \to 0$).

В результаті можна написати:

$$w_y(y) = w_x(\varphi(y)) \cdot |\frac{d\varphi(y)}{dy}|. \quad (2.2)$$

Цей вираз використовується для обчислення щільності ймовірності результату перетворення, яке не збігається з щільністю розподілу вихідної випадкової величини. Зрозуміло, що вид закону розподілу щільності ймовірностей залежить від характеристики перетворення, оскільки вираз (2.2) містить обернену функцію перетворення та її похідну.

Відносини набувають більш складної форми, коли перетворення описується невзаємно однозначною функцією.

У результаті кожного перетворення щільність ймовірності кінцевого зображення не збігатиметься з щільністю ймовірності вихідного зображення.

Неважко переконатися, що в умовах лінійного контрасту вигляд закону розподілу щільності ймовірності збережеться, однак параметри щільності ймовірності перетвореного зображення будуть іншими.

Перетворення щільності ймовірності передбачає знання інтегрального розподілу вихідного зображення. Достовірних відомостей про нього, як правило, немає.

Використання аналітичних наближень для опису функцій розподілу також недоцільно, оскільки їх невеликі відхилення від фактичних розподілів можуть спричинити значну різницю в кінцевих результатах.

Обробка зображень трансформації розподілу здійснюється в два етапи.

На першому кроці вимірюється гістограма вхідного зображення.

Для цифрового зображення, шкала яскравості якого становить від 0 до 255, гістограма являє собою масив із 256 чисел, кожне з яких вказує на кількість пікселів певної яскравості в даному кадрі.

Щоб знайти оцінку ймовірності розподілу яскравості пікселів на зображенні, необхідно розділити всі числа в цій таблиці на загальний розмір вибірки, який дорівнює загальній кількості пікселів на зображенні.

Позначимо цю оцінку через $w_x^*(j), 0 \leq j \leq 255.$

Тоді оцінка інтегрального розподілу буде визначатися за формулою:

$$F^*(j) = \sum_{i=1}^{j} w_x^*(i). \quad (2.3)$$

На другому етапі безпосередньо виконується нелінійне перетворення, що гарантує необхідні властивості вихідного зображення. Під час цього перетворення замість невідомого інтегрального розподілу використовується його оцінка на основі гістограми.

Методи поелементного перетворення зображення, метою яких є модифікація законів розподілу, називають методами гістограм. Зокрема, перетворення, в результаті якого кінцеве зображення має рівномірний розподіл, називається вирівнюванням (вирівнюванням) гістограм.

Процедури перетворення гістограм можна застосовувати як до зображення в цілому, так і до окремих його фрагментів.

Застосування цієї процедури до окремих фрагментів зображення може бути корисним для обробки нестаціонарних зображень, зміст яких суттєво відрізняється за ознаками в різних областях. У цьому випадку кращого ефекту можна досягти, застосувавши обробку гістограми до окремих ділянок зображення.

Використання співвідношень (2.1 – 2.3), справедливих для зображень із неперервним розподілом яскравості, не зовсім коректно для цифрових зображень.

Слід враховувати, що в результаті обробки не вдається отримати ідеальний розподіл ймовірностей вихідного зображення, тому корисно контролювати його гістограму.

2.2 Розробка математичної моделі інтелектуальної обробки зображень

Створення гістограми для кольорових зображень необхідне на кожному з каналів, а також для зображень у відтінках сірого. Гістограмою називають схему розподілу напівтонового зображення, що зображає яскравість уздовж горизонтальної осі та відносну кількість пікселів із сталими значеннями яскравості вздовж вертикальної осі [32].

Щоб побудувати гістограму зображення, необхідно слідувати наступному алгоритму.

– формування масиву даних (зазвичай масив [0..255]);

– виберір потрібного колірного каналу для кожного пікселя. Отримане значення має відповідати області індексу масиву, наприклад [0..255].

– гістограмою називають отриманий масив, елементи якого – висота стовпців.

Гістограма є розподілом кількості пікселів відповідно до їх інтенсивності, тому виникає необхідність проаналізувати зображення для визначення такого розподілу.

Розроблено функцію, яка приймає як вхідні дані двовимірний масив (цифрове зображення), що містить для кожного пікселя відповідне значення інтенсивності. Тип, який використовується для зберігання цих значень, може бути цілим або плаваючим. Функція перетворює цей двовимірний масив в одновимірний масив, що містить частоту кожного рівня інтенсивності.

Щоб мати можливість виконувати подальші перетворення зображення, необхідно нормалізувати гістограму. Нормалізація надасть розподілу властивості функції щільності ймовірності. Для отримання цього результату ми використали формулу, наведену в теоретичній частині.

Нормалізація гістограми – це техніка, яка полягає в перетворенні дискретного розподілу інтенсивностей у дискретний розподіл ймовірностей. Для цього нам потрібно розділити кожне значення гістограми на кількість пікселів. Оскільки цифрове зображення – це дискретний набір значень, який можна

розглядати як матрицю, і це еквівалентно розділенню кожного n_k на розмір масиву, який є добутком ширини на довжину зображення.

$$n_{kn} = \frac{n_k}{length \times width} = p_r(r_k) \tag{2.4}$$

Це можна записати в термінах математичного перетворення:

$$\begin{cases} [0, L-1] \to N \\ x \to Card(x) \end{cases} becomes \begin{cases} [0, L-1] \to [0,1] \\ x \to pd\ f(x) = \frac{Card(x)}{\sum_{i=0}^{L-1} Card(x_i)} \end{cases} \tag{2.5}$$

де Card означає потужність набору, тому в нашому випадку кількість пікселів.

Далі використаємо метод вирівнювання гістограми. Вирівнювання гістограми – це метод обробки зображень, щоб налаштувати контрастність зображення шляхом зміни розподілу інтенсивності гістограми. Метою цієї методики є надання лінійної тенденції кумулятивній функції ймовірності, пов’язаній із зображенням.

Обробка вирівнювання гістограми базується на використанні кумулятивної функції ймовірності (cdf). cdf – це кумулятивна сума всіх ймовірностей, що лежать у його області і визначається:

$$cfd(x) = \sum_{k=-\infty}^{x} P(k) \tag{2.6}$$

Ідея цієї обробки полягає в тому, щоб надати отриманому зображенню лінійну кумулятивну функцію розподілу. Дійсно, лінійна cdf пов’язана з рівномірною гістограмою, яку ми хочемо мати в результаті зображення.

З метою зберігання яскравості зображення та покращення локального контрасту запропоновано вирівнювання гістограми на основі нечіткої логіки. По-перше, нечітка гістограма створюється за допомогою нечіткої логіки, щоб краще впоратися з неточністю значень рівня сірого, і вона розділена на дві

підгістограми на основі медіанного значення вихідного зображення. Потім кожній гістограмі призначається новий динамічний діапазон.

Для цього необхідно виконати кілька перетворень. Перше з них – фазифікація та інтенсифікація. У фазифікації зображення інтенсивності рівнів сірого перетворюються на нечітку площину, значення якої коливається від 0 до 1. Зображення розміру $M \times N$ та рівня інтенсивності в діапазоні (0, L-1) можна розглядати як набір нечітких одиночних елементів у нотації нечіткого набору, кожен з яких має функція належності, що позначає ступінь наявності деякого рівня сірого. Нечітку матрицю F, що відповідає цьому зображенню, можна виразити як

$$F = \begin{bmatrix} \frac{\mu_{11}}{f_{11}} & \frac{\mu_{12}}{f_{12}} & \dots & \frac{\mu_{1N}}{f_{1N}} \\ \frac{\mu_{21}}{f_{21}} & \frac{\mu_{22}}{f_{22}} & \dots & \frac{\mu_{2N}}{f_{2N}} \\ \dots & \dots & \dots & \\ \frac{\mu_{M1}}{f_{M1}} & \frac{\mu_{M2}}{f_{M2}} & \dots & \frac{\mu_{MN}}{f_{MN}} \end{bmatrix}. \tag{2.7}$$

де $\mu_{ij} = 0$ позначає темний, $\mu_{ij} = 1$ позначає світлий. Будь-яке проміжне значення відноситься до ступеня максимального рівня сірого пікселя. Набір, що складається з усіх μ_{ij}, називається площиною нечітких властивостей зображення. Щоб зменшити кількість нечіткості зображення, посилення контрасту застосовується до нечіткого набору F для створення іншого нечіткого набору, функція приналежності якого виражається як

$$\mu_{F(ij)} = \begin{cases} 2 * (\mu_{ij})^2 & 0 \le \mu_{ij} \le 0.5, \\ 1 - (2 * \left(1 - \mu_{ij}\right)^2) & 0.5 < \mu_{ij} \le 1. \end{cases} \tag{2.8}$$

Щоб покращити зображення, ми зосереджуємося на посиленні контрастності. Це досягається за рахунок того, що темний піксель стає темнішим, а яскравий – яскравішим. Отже, нечітка гістограма обчислюється за

допомогою (2.8). Нечітка гістограма – це послідовність дійсних чисел $h(i), i \in (0,1 \dots, L-1)$, тут $h(i)$ вказано частоту появи сірих рівнів навколо i. Розглядаючи значення сірого $f(i,j)$ як нечітке число $\mu_{F(ij)}$, нечітка гістограма обчислюється як

$$F \leftarrow h(i) + \sum_i \sum_j \mu_{F(ij)}, \tag{2.9}$$

де $\mu_{F(ij)}$ нечітка функція належності.

Нечітка статистика здатна впоратися з неточністю значень сірого набагато краще, ніж класичні чіткі гістограми, таким чином створюючи плавну гістограму.

Більшість електронного обладнання отримує та відображає кольорові зображення. У цьому відношенні більший інтерес представляв би метод покращення кольорових зображень. Отже, підвищення контрастності можна легко поширити на кольорові зображення. Найочевидніший спосіб поширити запропоноване покращення контрастності градації сірого на кольорові зображення – застосувати цей метод лише до компонента яскравості та зберегти компоненти кольоровості. Цей метод дає кращі відчутні результати порівняно з іншими звичайними методами.

Аналіз спектрів зображень набув широкого поширення для обробки цифрових зображень. Спектр зображень визначається прямим двовимірним перетворенням Фур'є функції, яка описує зображення.

При неперервному зображенні його спектр за Фур'є має вигляд:

$$F(u,v) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} I_1(x,y) e^{-j(ux+vy)} dxdy, \tag{2.10}$$

де u,v – просторові частоти.

Оберненим перетворенням Фурьє для зображення називають

$$I_1(x,y) = \frac{1}{4\pi^2} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} F(u,v) e^{j(ux+vy)} dxdy, \tag{2.11}$$

При дискретному зображенні розмірністю M*N:

$$F[k,p] = \frac{1}{NM}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1} I_1[n,m]e^{-j2\pi(\frac{kn}{N}+\frac{pm}{M})}, \quad (2.12)$$

де k, p – номери гармонічних спектральних складових зображення, а

$$I_1[n,m] = \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} I_1[n,m]e^{-j2\pi(\frac{kn}{N}+\frac{pm}{M})}. \quad (2.13)$$

Так само як для одновимірного випадку, для комплексного спектру зображення можна отримати амплітудний та фазовий спектри:

$$|F(u,v)| = \sqrt{Re(F(u,v))^2 + Im(F(u,v))^2}. \quad (2.14)$$

Обробка зображень за допомогою використаних методів характеризується простотою реалізації та підходить для роботи з зображеннями, що забезпечує підвищення швидкості обробки зображень.

2.3 Проектування структури інформаційної технології інтелектуальної обробки зображенньь

Інтелектуальна обробка зображенньь потребує великої обчислювальної потужності. Вхідними даними інформаційної технології є зображення, яке необхідно обробити, проаналізувавши його спектр. Оскільки обробка здійснюється програмно, користувачеві потрібно лише завантажити зображення. Після отримання вхідних даних – відбувається його подальша розробка, після чого можна переглянути гістограми даного зображення. Інформаційна технологія здатна автоматично опрацьовувати всю доступну вхідну інформацію, а також вивести результат у доступному форматі, забезпечивши зручність та зрозумілість

подання інформації. Структурна схема інформаційної технології інтелектуальної обробки зображень наведено на рис. 2.1.

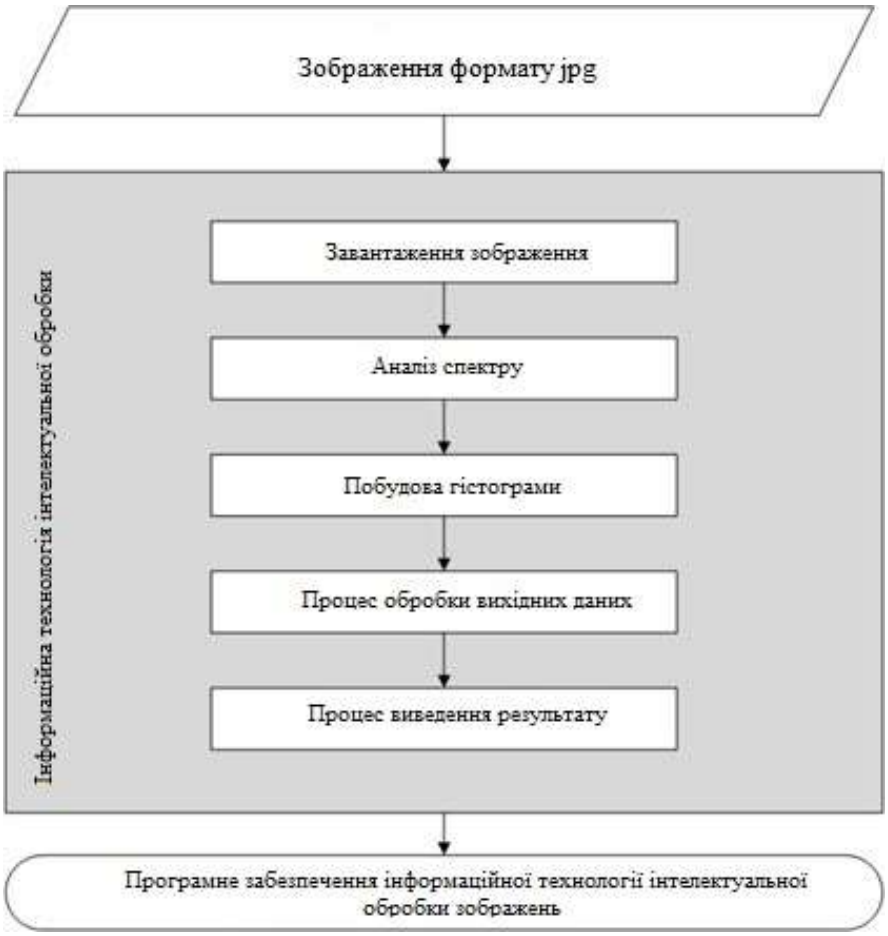


Рисунок 2.1 – Структурна схема інформаційної технології інтелектуальної обробки зображень

В інформаційній технології будуть представлені наступні модулі: модуль введення зображення, модуль обробки зображення, модуль інтерфейсу та модуль побудови гістограм. Схема моделі функціонування інформаційної технології інтелектуальної обробки зображень наведена на рисунку 2.2.

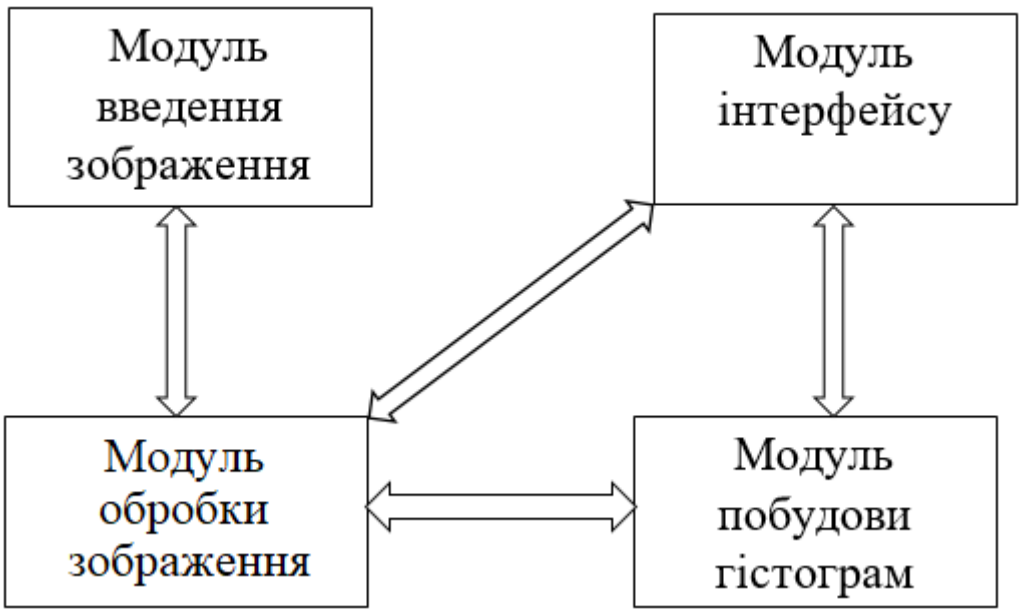


Рисунок 2.2 – Схема моделі функціонування інформаційної технології інтелектуальної обробки зображень

Завдяки графічному інтерфейсу, який має зв’язок з модулем обробки зображення та модулем побудови гістограм, маємо можливість переглянути та ввести всю необхідну інформацію для користувача. Цей модуль відповідає за коректне відображення результатів обробки та введення даних на екрані.

Модуль введення зображення імпортує зображення, які користувач планує обробити. Модуль введення зображення передає зображення модулю обробки зображення для подальшої роботи з ним.

Модуль обробки зображення отримує зображення через модуль введення зображення та містить функцію передачі результатів обробки до модуля побудови гістограм. Даний модуль є основним модулем програми та містить інтелектуальний алгоритм обробки зображення. Обробка проходить за алгоритмами, застосуваними при розробці математичної моделі інтелектуальної обробки зображення.

Завдяки останньому модулеві – модулеві побудови гістограм, відбувається перетворення зображення в масив даних, у якому елементи показані у вигляді стовпців гістограми. Модуль побудови гістограм пов’язаний з модулем інтерфейсу та модулем обробки зображення.

2.4 Розробка алгоритму інтелектуальної обробки зображень

На рисунку 2.3 наведено схему загального алгоритму функціонування інформаційної технології.

Загальний алгоритм функціонування інформаційної технології складається з наступних кроків:

1. Спочатку користувач вибірирає зображення, яке він планує обробити.

2. Далі відбувається завантаження зображення в інформаційну технологію інтелектуальної обробки зображень.

3. Після цього за допомогою графічного інтерфейсу користувача відбувається відображення головного меню програми.

4. Далі виконується перевірка на потребу виведення гістограми по одному з каналів. Якщо немає потреби виведення гістограми по одному з каналів, відбувається перехід до кроку 7.

5. На даному етапі відбувається попередня обробка зображення та аналіз його спектру.

6. Виведення гістограми по одному з вибраних каналів. Перехід до пункту 14.

7. Перевірка на необхідність виведення гістограми яскравості. Якщо користувач не бачить в цьому необхідності, відбувається перехід до пункту 10.

8. Обробка зображення.

9. Виведення гістограми яскравості. Перехід до пункту 14.

10. Обробка завантаженого зображення.

11. Побудова всіх гістограм.

12. Обробка гістограм.

13. Виведення результату.

14. Закінчення діалогу з системою.

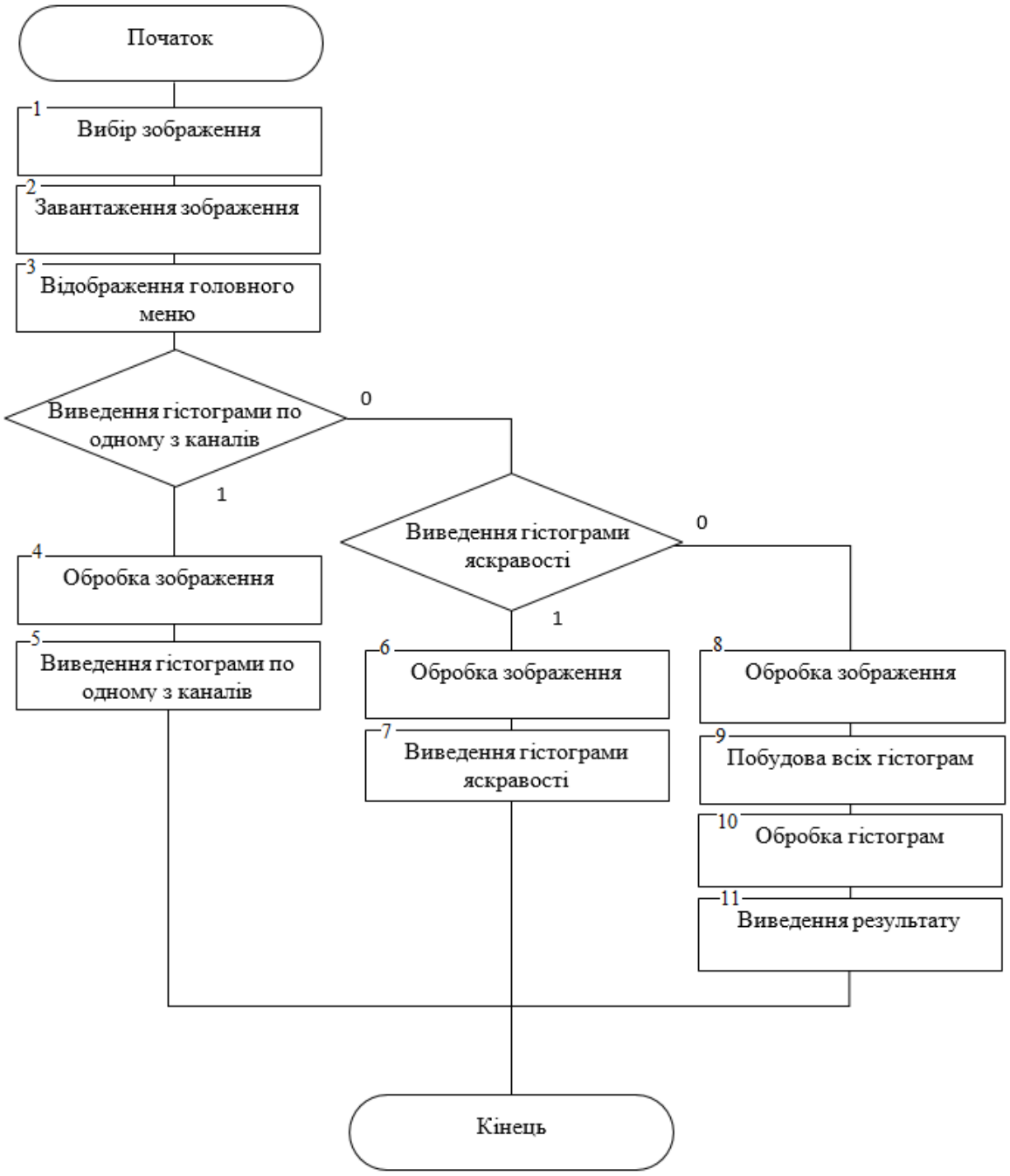


Рисунок 2.3 – Схема загального алгоритму функціонування інформаційної технології інтелектуальної обробки зображень

На рисунку 2.4 зображено схему алгоритму розбиття зображення на пікселі.

Даний алгоритм містить наступні кроки:

1. Розбиття зображення на набір пікселів.
2. На даному епапі відбувається перевірка на виконання обробки всіх пікселів. Якщо всі пікселі оброблені – завершується робота алгоритму.

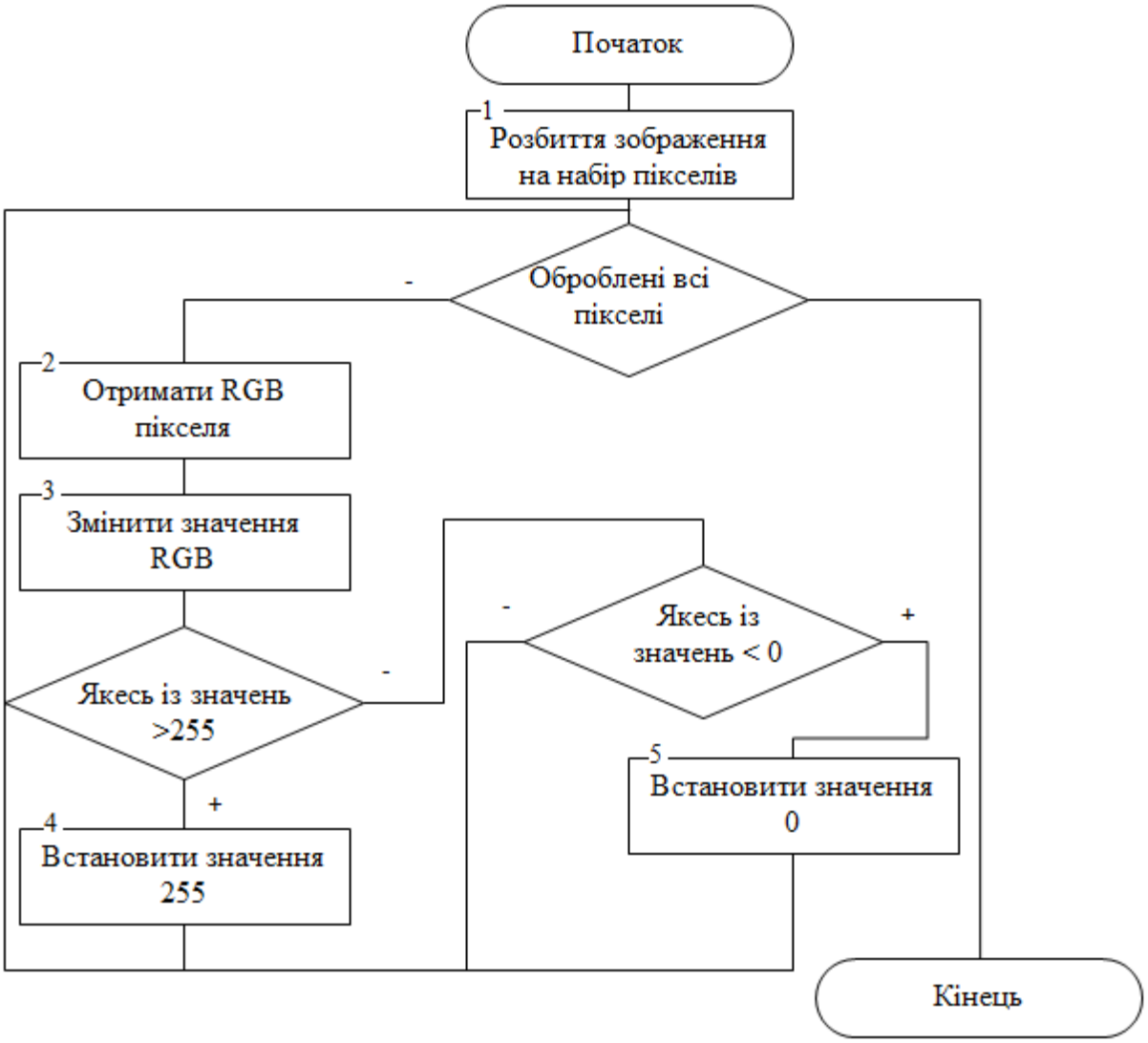


Рисунок 2.4 – Схема алгоритму розбиття зображення на пікселі

3. Отримання RGB пікселя.
4. Зміна значення RGB.
5. Перевірка на присвоєння пікселю значення більше ніж 255. Якщо значення менше – перехід до пункту 7.
6. Встановлення пікселю значення 255. Перехід до пункту 2.
7. Перевірка на присвоєння пікселю значення менше 0. Якщо значення більше – перехід до пункту 2.
8. Встановлення пікселю значення 0. Перехід до пункту 2.

2.5 Висновок до розділу 2

В другому розділі обґрунтовано використання методів побудови гістограми зображення, перевагами яких є те, що вони здатні швидко обробляти зображення. На основі вибраних методів було розроблено математичну модель, що поєднує в собі алгоритми аналізу спектру зображення та методу ортогональних операторів Собеля, що дозволяє максимально задовольнити потреби користувача та підвищити швидкість обробки зображення. Спроектовано структуру інформаційної технології інтелектуальної обробки зображення, наведено модель функціонування інформаційної технології. Розроблено алгоритми функціонування інформаційної технології, а також наведено алгоритм розбиття зображення на пікселі.

3 СТРУКТУРНА ОРГАНІЗАЦІЯ ТА ОСОБЛИВОСТІ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ІНТЕЛЕКТУАЛЬНОЇ ОБРОБКИ ЗОБРАЖЕНЬ

3.1 Обґрунтування вибору мови та середовища програмування

Мови та технології зазвичай розраховані на конкретні завдання та цілі. Наприклад, деякі мови використовуються тільки для розробки веб-сайтів, інші – тільки для написання програм для Windows, а деякі можуть поєднувати обидві функції. Для того, щоб зробити вибір якісної та об’єктивної мови програмування, необхідно порівняти найбільш актуальні мови. На сьогодні найбільш актуальними мовами для вирішення поставленої задачі є C# і Java.

Спочатку розглянемо мову програмування C #.

C# – об'єктно-орієнтована мова програмування з системою безпечного типу для платформи .NET [33]. Мова була розроблена Андерсом Ґалесбергом, Скоттом Вілтамутом і Пітером Голдом за підтримки Microsoft. Синтаксис мови C# подібний до мов програмування C++ і Java. C# має сувору статичну типізацію, підтримує поліморфізм, перевантаження операторів, посилання на функції-члени класу, властивості, події, атрибути, винятки та коментарі XML. C#, який дуже відрізняється від своїх попередників – C++, Delphi, Modules і Smalltalk – виключає певні шаблони, які спричиняють проблеми в розробці програмного забезпечення: так, C# не підтримує успадкування кількох класів (C++ підтримує) або виведення типу (на відміну від Haskell).

Коли мова йде про C#, слід виділити наступні переваги:

- використовує проміжний код, тому може працювати на будь-якому комп’ютері, незалежно від апаратного та програмного забезпечення системи;
- повністю об'єктно-орієнтований – можна писати тільки код для класів;
- є багато додаткових функцій, які допомагають програмувати;
- може автоматично звільняти непотрібну пам'ять;
- підтримує політику .NET.

- програма C# може складатися з одного або кількох файлів, що містять вихідний код мови програмування C#. Кожен із цих файлів має розширення .cs.

Недоліки включають поганий графічний інтерфейс платформи X; C# є внутрішньою частиною платформи .NET, тому сервер, на якому виконується програма, має бути на базі Windows; C# не дуже гнучкий, оскільки він в основному покладається на платформу .Net.

Далі ми розглянемо Java, об’єктно-орієнтовану мову програмування, розроблену компанією Sun Microsystems у 1991 році та офіційно випущену 23 травня 1995 року. Мова запозичує багато свого синтаксису з C і C++. Зокрема, він заснований на об'єктній моделі C++, але з модифікаціями. Виключає можливість певних конфліктних ситуацій, які можуть виникнути через помилки програмування, полегшує процес розробки об’єктно-орієнтованих програм. Набір операцій, які програміст повинен виконати на C/C++, делегується віртуальній машині. Java в першу чергу розроблена як незалежна від платформи мова з меншою кількістю апаратних функцій низького рівня, які можуть уповільнювати роботу програм порівняно з такими програмами, як C++. Java дозволяє при необхідності викликати підпрограми, написані на інших мовах програмування [34].

- При створенні мови програмування Java було п’ять початкових цілей:
- Синтаксис мови має бути «простим, об'єктно-орієнтованим і знайомим».
- Реалізація має бути «бездоганною та безпечною».
- «Архітектурна незалежність і портативність» повинні бути збережені.
- Висока продуктивність
- Мова має бути «інтерпретованою, багатопоточною, з динамічним зв’язуванням модулів» [35].

Є багато переваг використання Java як мови програмування. Далі обговоримо всі ці переваги одну за іншою.

1. Java проста. Проста мова програмування, яку легко вивчити та зрозуміти. Завдяки своєму простому коду Java є однією з найпростіших для вивчення та

впровадження мов програмування. Крім того, Java видаляє всі складні функції C і C++, такі як покажчики, структури, агрегати, і спрощує реалізацію коду.

2. Java – це об'єктно-орієнтована мова програмування. Однією з головних переваг Java є те, що це об'єктно-орієнтована мова програмування. Використання концепцій ООП робить Java простішою у реалізації та значно безпечнішою. Концепції ООП допомагають Java вирішувати реальні проблеми. Це також допомагає керувати великим кодом, розбиваючи його на іменовані менші частини.

3. Java є безпечною мовою. Такі мови, як C і C++, використовують покажчики для доступу до сховища. Це загроза безпеці, оскільки покажчики можуть призвести до неавторизованого доступу до пам’яті. Java також використовує концепції ООП, такі як інкапсуляція, абстракція та успадкування, щоб покращити безпеку та запобігти доступу неавторизованих користувачів.

4. Java не залежить від платформи. Java підтримує функцію WORA (Write Once Run Anywhere). Програми Java, написані на цій системі, можуть працювати на будь-якій іншій системі Java. Сумісність Java не залежить від операційної системи чи апаратного забезпечення, що робить платформу Java незалежною та надзвичайно гнучкою.

5. Java – мова програмування високого рівня. Програми Java написані людською мовою високого рівня. Вона схожа на англійську, але має легкий для запам’ятовуванння синтаксис. Java має інтерпретатор, який перекладає код на мову машинного рівня, щоб машина могла його зрозуміти.

6. Java підтримує функцію передачі. Java є портативною мовою. Це пояснюється тим, що Java не залежить від платформи і не вимагає спеціального обладнання для роботи. Фактично це робить Java сумісною майже з усіма можливими пристроями.

7. Java пропонує автоматичний збір сміття. У C або C++ ми повинні були звільнити пам'ять за допомогою програми. У Java JVM автоматично керує пам'яттю. Тому щоразу, коли об’єкт не посилається на клас і йому потрібно

назвати, JVM автоматично видаляє його з програми, тому нам не потрібно писати будь-який додатковий код. Тому Java підтримує автоматичне збирання сміття.

8. Java підтримує багатопотоковість. Пропускна здатність — це найменша можлива одиниця процесу. Багатопоточність є ключем до максимального використання ЦП. Java – це мова програмування, яка підтримує багатопотоковість. Java дозволяє нам запускати кілька потоків одночасно. Вони мають спільну пам’ять для підвищення ефективності та продуктивності програми. Нитки проходять незалежно один від одного.

9. Java регулярно отримує оновлення для виправлення помилок. Це робить Java однією з найстабільніших мов програмування. Оновлення негайно виправляють майже всі помилки. Тому важливо регулярно оновлювати Java.

10. Java забезпечує ефективну стратегію розподілу пам'яті. Java в основному ділить пам'ять на дві частини: область купи і область стеку. За потреби JVM виділяє пам’ять з однієї з двох частин. Це дозволяє ефективно керувати сховищем.

Отже, Java має кілька плюсів, однак вона теж не ідеальна. У Java також є свої недоліки.

1. Повільна і слабка продуктивність. Java споживає більше пам'яті, ніж рідні мови програмування, такі як C і C++. Java також повільніше в порівнянні з ними, що пов’язано з додатковою роботою інтерпретатора для перетворення коду на машинну мову. JVM виконує різні функції бекенда, які сповільнюють швидкість роботи програми. Оскільки Java підтримує автоматичне збирання сміття, вона завжди працює на стороні сервера, що впливає на продуктивність.

2. Java значно відстає, коли справа доходить до графічного інтерфейсу. Конструктор графічного інтерфейсу Java не може створити складний інтерфейс користувача. У Java є багато фреймворків графічного інтерфейсу, таких як Swing, SWT, JavaFX, JSF тощо. Сучасні мови, такі як Python, R, C# тощо, мають найкращі дизайнери графічного інтерфейсу.

3. Відсутність резервних коштів. Java абсолютно не має можливості зберігати дані користувача. В основному він зосереджений на зберіганні даних, але не захищений інструментами резервного копіювання.

4. Java займає більше пам'яті, ніж інші мови програмування, такі як C і C++. Погане керування пам’яттю Java. Java використовує збирач сміття, але це негативно впливає на продуктивність.

5. Детальний і складний код. Java має багато багатослівних і складних синтаксисів. Іноді ці складні синтаксиси важко запам’ятати. З цих причин багато програмістів віддають перевагу Python або C++ перед Java, оскільки вони мають відносно прості речення.

Отже, Java має як переваги, так і недоліки. Мінуси Java відносно менші в порівнянні з плюсами, тому її доцільність для використання при розробці інформаційної технології досить висока.

На основі описів двох мов програмування обрано Java для розробки даної інформаційної технології. Тому зробимо огляд програмного середовища для написання на обраній мові.

З метою розробки серверних програм Java зазвичай використовують три середовища розробки, а саме IntelliJ IDEA, Eclipse і NetBeans.

IntelliJ IDEA – це середовище розробки для створення програмних застосунків за допомогою Java. IntelliJ IDEA розроблялася компанією JetBrains [36]. Він доступний у вигляді ліцензованої версії спільноти Apache 2 і власної комерційної версії. Обидва середовища можна використовувати для розробки інформаційної технології. Однак саме IntelliJ IDEA було вибрано для реалізації інформаційної технології інтелектуальної обробки зображення.

IntelliJ IDEA надає рекомендації щодо завершення коду, аналізу коду та перевірених інструментів рефакторингу. Середовище має такі інструменти, як контроль версій, зв’язок з багатьма мовами і фреймворками. Воно може стежити за контекстом розробника і самостійно запускати потрібні інструменти.

Особливості IntelliJ IDEA:

- Розумний висновок: у ньому перераховані найбільш релевантні символи, які стосуються поточного контексту. Він безперервно переміщує новітні класи, методи тощо, збільшує список пропозицій. Відповідно, заповнення коду проводиться швидше.
- Аналіз потоку даних: дане середовище може аналізувати потік даних та передбачати можливий символ при виконанні.
- Мовне введення: можна легко інтегрувати фрагменти іншої мови у свій код Java, наприклад SQL.
- IntelliJ рекомендує глибокий та ефективний рефакторинг, так як він має необхідні знання про використання символів.
- Середовище пропонується з широким набором вбудованих інструментів, як наприклад GIT, контроль версій, декомпілятор, покриття, база даних SQL тощо.
- Наявний потужний компілятор, що може виявляти дублікати, запах коду тощо.
- Має надійну інтеграцію з серверами додатків.

Отже, завдяки наведеним перевагам, було вибрано мову програмування Java та середовище розробки IntelliJ IDEA.

3.2 Програмна реалізація інформаційної технології інтелектуальної обробки зображення

Програмування інформаційної технології інтелектуальної обробки зображень здійснюється мовою програмування Java. Розглянемо деякі бібліотеки, що використовуються в розробленій інформаційній технології.

Графічний інтерфейс користувача розроблено за допомогою зручного інструменту – бібліотеки Swing, яка була розроблена для забезпечення більш функціонального набору програмних компонентів, ніж її попередники. Swing – це графічна бібліотека [37], яка пропонує велику офіційну документацію,

гнучкість компонентів і відносно низький бар’єр для входу. Багато сучасних IDE включають графічні редактори для форм Swing.

Компоненти Swing написані на Java і повністю незалежні від платформи. Переваги також включають такі функції:

- вікна програми малюються на графічному процесорі. Це дозволяє анімацію в програмі;
- компоненти Swing реалізовані за принципом Model Look Controller. Це дозволяє швидко додати потрібну функціональність до існуючого компонента;
- безпека коду завдяки реалізації на Java;
- широкий спектр середовищ розробки: Eclipse, Intellij IDEA, NetBeans;
- широкий вибір бібліотек з додатковими компонентами;
- велика спільнота прихильників, ви можете знайти відповідь на будь-яке питання великого значення в розвитку, обмежене в часі.

Моделі, надані Swing, поділяються на дві категорії: моделі стану GUI та моделі даних програми. Перший визначає стан віджета, напр. В. натиснута чи ні кнопка. Цей шаблон можна використовувати поза контекстом GUI. Друга модель – це інтерфейс, який представляє кількісні дані, такі як дані діаграм або таблиці. Ці моделі даних забезпечують парадигму для відокремлення вмісту від зовнішнього вигляду.

Для роботи з бібліотекою необхідно імпортувати класи, що дозволять не тільки створювати графічний дизайн, але також відстежувати різні натискання на кнопки і виконувати дії після натискання.

```
import java.awt. *;
import java.awt.event. *;
import javax.swing. *;
```

Щоб працювати з багатьма методами, наш основний клас повинен наслідувати все від класу з назвою JFrame.

Після виконання успадкування можна отримати доступ до класів для створення полів, кнопок і інших елементів.

У Swing використовуються контейнери «Container» для угруповання декількох елементів в одну форму. Перед створенням контейнера необхідно встановити: назву вікна, його розміри і координати (x, y, w, h), а також встановили операцію при закритті вікна.

Далі створюється об'єкт на основі класу Container і прописується метод getContentPane (). Даний метод повертає контейнер верхнього рівня, в який поміщаються всі інші об'єкти на вікні.

Для розміщення об'єктів варто вибрати будь-якої шар. Наприклад, можна використаний шар GridLayout, що дозволяє розташовувати об'єкти в форматі сітки або ж таблиці.

Клас Java ImageIO є останнім класом, який належить пакету javax.imageio. Клас забезпечує зручний метод для читання та запису зображень і ароматизує просте кодування та декодування. Клас надає багато корисних методів, пов'язаних з обробкою зображень. Використовуючи клас, ми можемо працювати з популярними розширеннями зображень, такими як .jpg, .bmp, .gif, .png тощо.

Працюючи з бібліотеками введення / виведення зображень, можна помітити, що майже вся робота запитується через статичні методи класу ImageIO. Ці статичні методи – все, що потрібно для базового використання. Наприклад, щоб прочитати зображення, його місцезнаходження передається одному з наступних методів read класу ImageIO:

- read(File input);
- read(ImageInputStream stream);
- read(InputStream input);
- read(URL input).

Наступний фрагмент коду містить приклад використання Imageio.

```
import javax.imageio.ImageIO;
import java.io.File;
import java.awt.Color;
import java.awt.image.BufferedImage;
import java.io.IOException;

public class Image {
   public static void main(String[] args) {
      try {
```

```
            // Открываем изображение
            File file = Main.img;
            BufferedImage source = ImageIO.read(file);

            BufferedImage result = new BufferedImage(source.getWidth(), source.getHeight(),
source.getType());

            for (int x = 0; x < source.getWidth(); x++) {
               for (int y = 0; y < source.getHeight(); y++) {

                  Color color = new Color(source.getRGB(x, y));

                  int blue = color.getBlue();
                  int red = color.getRed();
                  int green = color.getGreen();

                  int grey = (int) (red * 0.299 + green * 0.587 + blue * 0.114);

                  int newRed = grey;
                  int newGreen = grey;
                  int newBlue = grey;

                  Color newColor = new Color(newRed, newGreen, newBlue);

                  result.setRGB(x, y, newColor.getRGB());
               }
            }
            File output = new File("katana_grey.jpg");
            ImageIO.write(result, "jpg", output);

         } catch (IOException e) {

            System.out.println("Файл не найден или не удалось сохранить");
         }
      }
   }
```

З фрагменту видно, що в ньому використовується підклас BufferedImage. Підклас BufferedImage описує зображення з доступним буфером даних зображення. BufferedImage складається з ColorModel і Raster даних зображення. Кількість і типи смуг у SampleModel растру мають відповідати кількості та типам, необхідним ColorModel для представлення її колірних і альфа-

компонентів. Усі об’єкти BufferedImage мають координату верхнього лівого кута (0, 0). Тому будь-який растр, який використовується для побудови BufferedImage, повинен мати min X=0 і minY=0.

Цей клас покладається на методи вибірки та налаштування даних Raster, а також на методи визначення кольорів ColorModel.

Також розглядається клас IOException. Він сигналізує про те, що сталася якась виняткова ситуація введення-виведення. Цей клас є загальним класом винятків, створених невдалими або перерваними операціями введення-виведення.

Клас File – це представлення Java шляху до файлу або каталогу. Оскільки імена файлів і каталогів мають різні формати на різних платформах, простого рядка не можна назвати. Клас File містить кілька методів для роботи з іменем шляху, видалення та перейменування файлів, створення нових каталогів, перерахування вмісту каталогу та визначення кількох загальних атрибутів файлів і каталогів.

Це абстрактне представлення файлів і шляхів до каталогів. Шлях, абстрактний чи у формі рядка, може бути абсолютним або відносним. Батьківський елемент абстрактного шляху може бути отриманий шляхом виклику методу getParent() цього класу. Перш за все, ми повинні створити об’єкт класу File, передавши йому ім’я файлу або ім’я каталогу. Файлова система може реалізовувати обмеження на певні операції над фактичним об'єктом файлової системи, такі як читання, запис і виконання. Ці обмеження спільно відомі як дозволи на доступ.

Екземпляри класу File незмінні; тобто після створення абстрактний шлях, представлений об’єктом File, ніколи не зміниться.

Клас Color є частиною пакета Java Abstract Window Toolkit (AWT). Клас Color створює колір, використовуючи задані значення RGBA, де RGBA означає RED, GREEN, BLUE, ALPHA, або використовуючи значення HSB, де HSB означає HUE, SATURATION, BRIcomponents. Значення для окремих компонентів RGBA коливається від 0 до 255 або від 0,0 до 0,1. Значення альфа

визначає непрозорість кольору, де 0 або 0,0 означає повністю прозорість, а 255 або 1,0 означає непрозорість.

Для побудови графіка з прогнозованою кількістю порушень правил дорожнього руху розроблено клас Line, що використовує абстрактні класи Point2D (визначення точки, яка представляє місце в координатному просторі (x, y)), Line2D (представлення відрізку лінії в просторі координат (x, y)) та Rectangle2D (опис прямокутника, визначеного розташуванням (x, y) і розмірністю (w x h)) [38].

Ці класи є лише абстрактними суперкласами усіх об’єктів, які зберігають 2D-координату. Фактичне представлення зберігання координат залишається для підкласу.

Для демонстрації розробки класу Line наведено наступний код

```
import java.awt.geom.Line2D;
import java.awt.geom.Point2D;
import java.awt.geom.Rectangle2D;
public class Line extends Line2D{

    private Point p1;
    private Point p2;

    public Line(){
        p1 = new Point();
        p2 = new Point();
    }

    public Line(double x1, double y1, double x2, double y2){
        p1 = new Point(x1, y1);
        p2 = new Point(x2, y2);
    }

    @Override
    public Rectangle2D getBounds2D() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public Point2D getP1() {
```

```
        // TODO Auto-generated method stub
        return p1;
    }

    @Override
    public Point2D getP2() {
        // TODO Auto-generated method stub
        return p2;
    }

    @Override
    public double getX1() {
        // TODO Auto-generated method stub
        return p1.getX();
    }

    @Override
    public double getX2() {
        // TODO Auto-generated method stub
        return p2.getX();
    }

    @Override
    public double getY1() {
        // TODO Auto-generated method stub
        return p1.getY();
    }

    @Override
    public double getY2() {
        // TODO Auto-generated method stub
        return p2.getY();
    }

    @Override
    public void setLine(double x1, double y1, double x2, double y2) {
        // TODO Auto-generated method stub
        p1.setLocation(x1, y1);
        p2.setLocation(x2, y2);

    }

}
```

Spring Framework (або коротше Spring) - це універсальний фреймворк з відкритим кодом для платформи Java [39].

Spring пропонує рішення багатьох проблем, з якими стикаються розробники та організації Java, які прагнуть створити інформаційну систему на основі платформи Java. Через його широку функціональність важко визначити основні конструктивні елементи, з яких вона складається. Незважаючи на широку інтеграцію, Spring не повністю інтегрований з платформою Java Enterprise, що є великою причиною його популярності.

Spring, мабуть, найвідоміший як джерело вдосконалень (функціональних можливостей), необхідних для ефективної розробки складних бізнес-додатків, які виходять за рамки високопродуктивних програмних моделей, що переважали в галузі в минулому. Ще однією перевагою є те, що раніше невикористані функції були введені в переважаючі сьогодні методи розробки навіть за межами платформи Java.

З цією технологією пов’язано два цікаві моменти:

По-перше, на відміну від багатьох інших платформ (таких як Apache Struts, які створюють лише Інтернет), Spring можна використовувати для побудови будь-яких програм Java (наприклад, окремих, веб-програм або Java Enterprise Edition (JEE)).

По-друге, легка вага не має нічого спільного з кількістю класів чи розміром розподілу, але визначає принцип всієї філософії стрибків - мінімальний вплив. Платформа Spring є легкою, оскільки вона максимально використовує свої основні функції.

3.3 Опис класів і функцій інформаційної технології інтелектуальної обробки зображень

На рисунку 3.4 зображено загальну UML-діаграму класів інтелектуальної системи обробки зображень.

Загальна UML-діаграма класів містить розроблені класи Main, MainG, Men, Gist, GraphicsPanel, Point, Line, Processing.

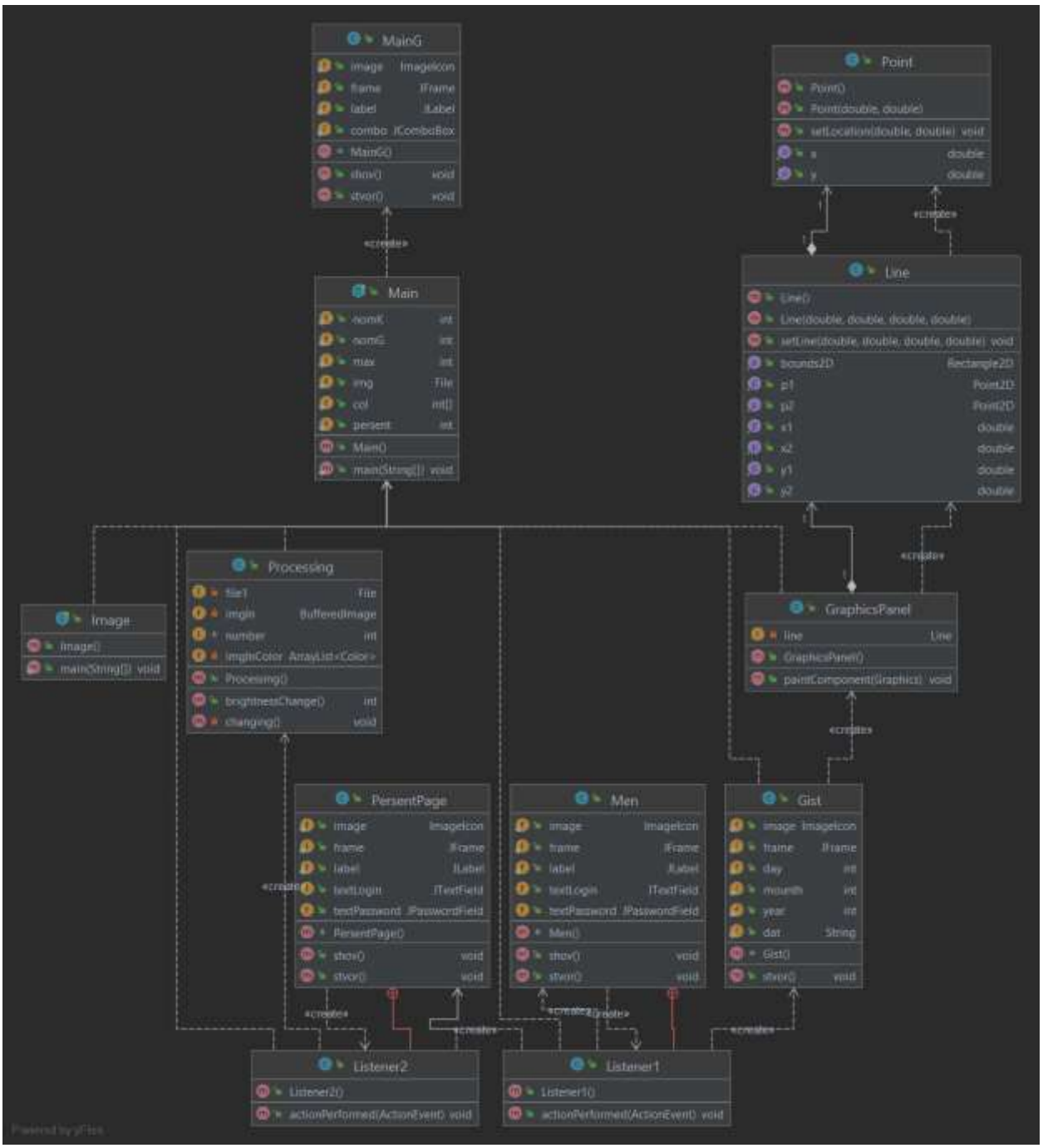


Рисунок 3.4 – Загальна UML-діаграма класів

Клас Main – основний клас програми що здійснює доступ до всіх інших функціональних класів.

У класі MainG відбувається відображення початкової активності програми, що містить кнопку для вибору зображення.

Клас Men відповідає за створення головного меню, що реалізовує перехід до кожного з пунктів обробки.

З допомогою класу Gist відбувається побудова гістограм для обробки зображення.

Клас GraphicsPanel – відповідає відображення гістограм.

Клас Point використовує абстрактний клас Point2D, що обчислює точку для представлення місця в координатному просторі (x, y).

Клас Line відповідає за використання абстрактних класів Point2D, Line2D та Rectangle2D для побудови гістограм.

Клас Processing відповідає за зміну яскравості зображення.

3.4 Тестування інформаційної технології інтелектуальної обробки зображень

Інформаційна технологія інтелектуальної обробки зображень була протестована. Було проведено 300 запусків програми, протестовано можливості її роботи. На початку роботи з програмою відкривається вікно початкової активності (рис. 3.2). Воно містить лише одну кнопку «Обрати зображення».

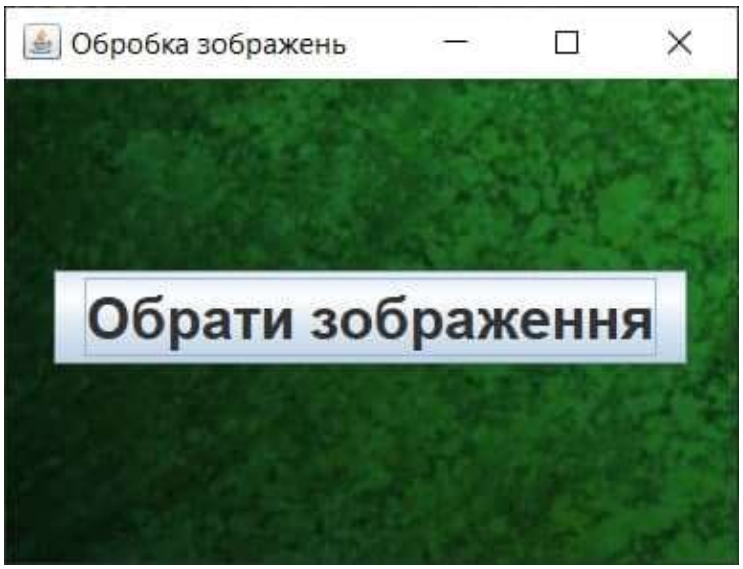


Рисунок 3.2 – Загальний вигляд інтерейсного вікна початкової активності

Далі потрібно обрати картинку з допомогою провідника (рис. 3.3), після чого натиснути кнопку «Open».

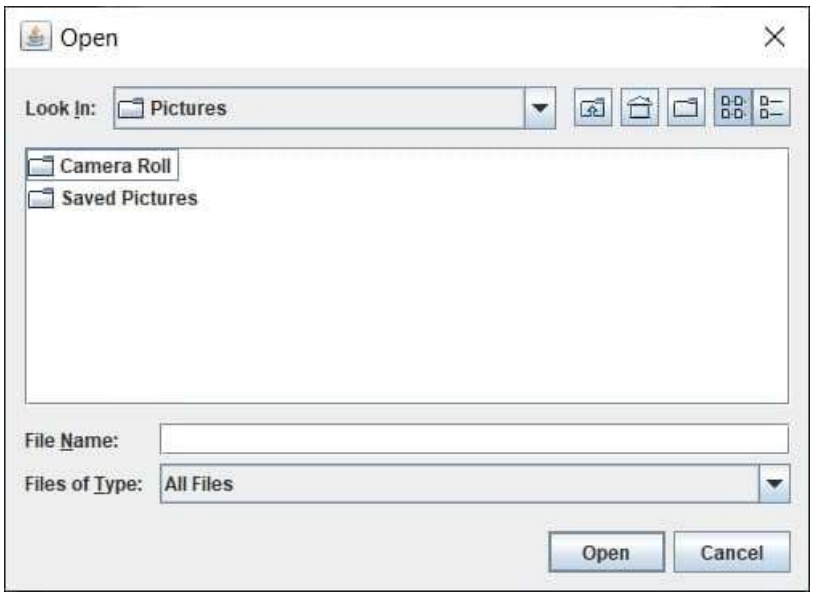


Рисунок 3.3 – Загальний вигляд інтерейсного вікна вибору зображення

Після того, як користувач обрав зображення, потрібно натиснути кнопку «Open». В результаті здійсниться перехід до головного меню обробки зображення, що дає змогу обрати один із запропонованих варіантів обробки зображення: «Гістограма по червоному каналу», «Гістограма по зеленому каналу», «Гістограма по синьому каналу», «Гістограма яскравості», «Змінити яскравість» (рис. 3.4).

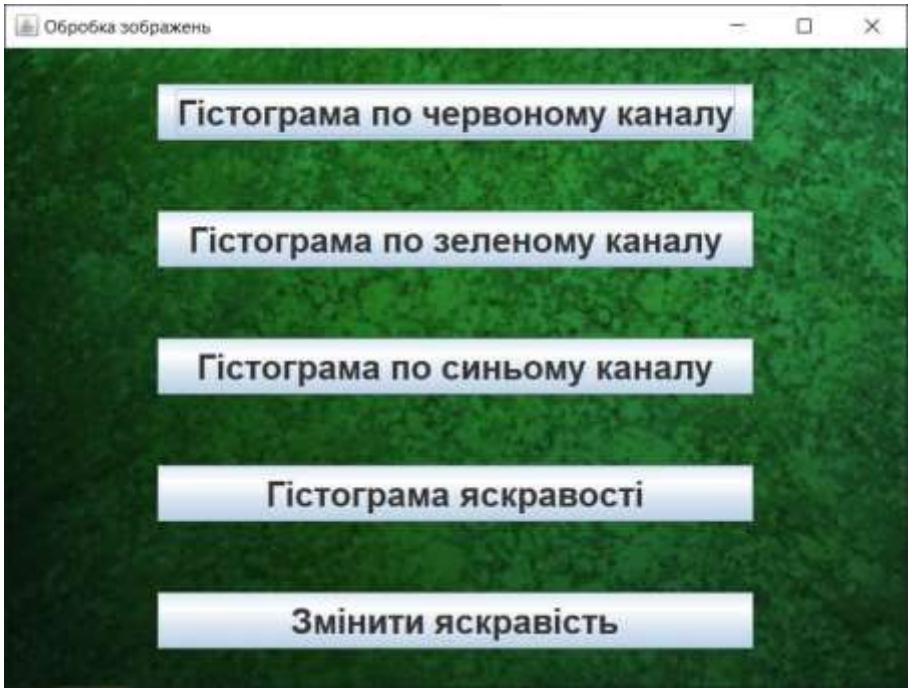


Рисунок 3.4 – Загальний вигляд інтерейсного вікна меню обробки зображення

Вибравши пункт «Гістограма по зеленому каналу» відбувається перехід до вікна з гістограмою, наведеного на рисунку 3.5. Вибравши пункт «Гістограма яскравості» відбувається перехід до вікна з відповідною гістограмою (рис. 3.6).

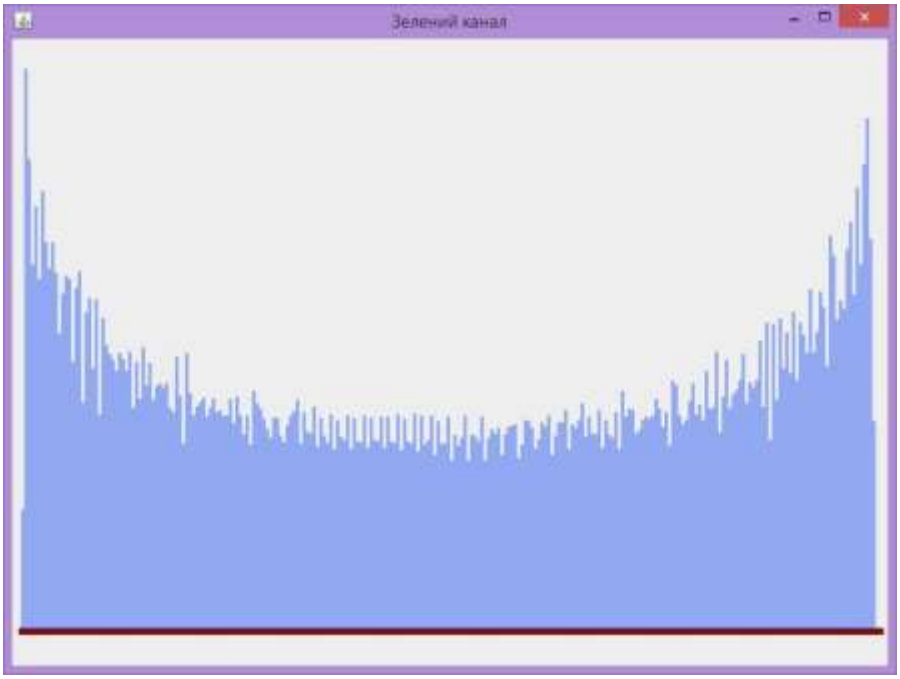


Рисунок 3.5 – Загальний вигляд інтерейсного вікна для гістограми по зеленому каналу

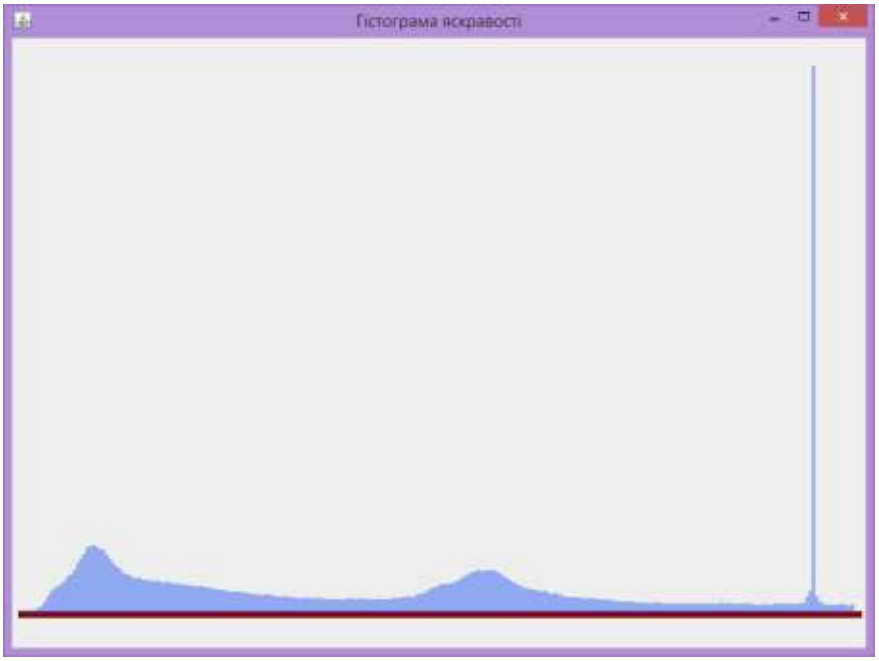


Рисунок 3.6 – Загальний вигляд інтерейсного вікна для гістограми яскравості тестового зображення

Вибравши останній пункт головного меню відбувається перехід до вікна для вводу числового значення зміни яскравості (рис. 3.7). Тут потрібно вводити числове значення, після чого натиснути кнопку «Далі» (рис. 3.8).

Рисунок 3.7 – Загальний вигляд інтерейсного вікна зміни яскравості

Рисунок 3.8 – Загальний вигляд інтерейсного вікна введення яскравості

Таблиця 3.1 містить порівняльні результати роботи розробленого додатку з програмами-аналогами IC Measure, Jmicrovision та Photomania. Розроблений додаток має вищу швидкість ніж розглянуті програми-аналоги.

Таблиця 3.1 – Порівняльний аналіз роботи програм-аналогів та розробленого додатку

Програма	Швидкість обробки
Розроблений додаток	0,5 секунди
Jmicrovision	0,9 секунди
IC Measure	0,9 секунди
Photomania	1,3 секунди

Отже, із таблиці 3.2 випливає, що розроблена інформаційна технологія інтелектуальної обробки зображень має на 0,5 с вищу швидкодію обробки, що означає досягнення мети.

3.5 Висновок до розділу 3

В третьому розділі обґрунтовано використання мови програмування Java, розглянуто її особливості, переваги та недоліки. Обґрунтовано вибір середовища програмування та наведено переваги роботи в ньому, обґрунтовано використання наведених бібліотек мови програмування Java, використаних для написання розробленої інформаційної технології інтелектуальної обробки зображень.

В даному розділі розроблено UML-діаграму класів програми. Програмно реалізовано та протестовано інформаційну технологію.

Проведене тестування показало підвищення швидкості обробки зображення порівняно з програмами аналогами, що довело досягнення мети. Тестування програми підтвердило очікувані результати та правильність роботи програми, довівши підвищення швидкодії обробки зображення на 0,5с.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту є оцінювання науково-технічного рівня та рівня комерційного потенціалу інформаційної технології інтелектуальної обробки зображень, створеної в результаті науково-технічної діяльності, тобто під час виконання магістерської кваліфікаційної роботи.

Аналогом може бути система «IC Measure», вартість використання якої складає 500 грн/місяць, тобто за три роки вартість використання даної програми складе 18000 грн.

Для проведення комерційного та технологічного аудиту залучають не менше 3-х незалежних експертів, якими можуть бути провідні викладачі випускової або спорідненої кафедри чи інші відомі фахівці. Не рекомендується залучати експертами керівника магістерської кваліфікаційної роботи та завідувача відповідної випускової кафедри.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням п’ятибальної системи оцінювання за 12-ма критеріями, наведеними в табл. 4.1.

Таблиця 4.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за п’ятибальною шкалою)					
	0	1	2	3	4
1	*2*	*3*	*4*	*5*	*6*
Технічна здійсненість концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах

Продовження таблиці 4.1

1	*2*	*3*	*4*	*5*	*6*
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижча за ціни аналогів	Ціна продукту значно нижча за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні не значні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування

Продовження таблиці 1.4

1	*2*	*3*	*4*	*5*	*6*
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5ти років. Термін окупності інвестицій більший 5ти років	Термін реалізації ідеї менший 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менший 3-х років. Термін окупності інвестицій менший 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що потребує значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту потребує незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь- які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці 4.2.

Таблиця 4.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	3	3	4
2. Ринкові переваги (наявність аналогів)	3	2	2
3. Ринкові переваги (ціна продукту)	3	4	3

Продовження таблиці 4.2

4. Ринкові переваги (технічні властивості)	4	3	4
5. Ринкові переваги (експлуатаційні витрати)	4	3	4
6. Ринкові перспективи (розмір ринку)	3	3	3
7. Ринкові перспективи (конкуренція)	4	3	3
8. Практична здійсненність (наявність фахівців)	4	4	4
9. Практична здійсненність (наявність фінансів)	3	3	4
10. Практична здійсненність (необхідність нових матеріалів)	4	4	3
11. Практична здійсненність (термін реалізації)	3	3	3
12. Практична здійсненність (розробка документів)	2	3	3
Сума балів	*40*	*38*	*40*
Середньоарифметична сума балів $СБ_c$	$\text{СБ}_\text{с} = \frac{\sum_i^3 \text{СБ}_i}{3} = 39{,}3$		

За результатами розрахунків, наведених в таблиці 4.2, робиться висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використовують рекомендації, наведені в табл. 4.3.

Таблиця 4.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вищий середнього
21…30	Середній
11…20	Нижчий середнього
0…10	Низький

Як видно з таблиці, рівень комерційного потенціалу розроблюваного нового програмного продукту є вищим середнього, що досягається за рахунок підвищення швидкодії обробки цифрових зображень шляхом удосконаленого

алгоритму обробки зображення, що поєднує в собі метод вирівнювання гістограми на основі нечіткої логіки та метод аналізу спектрів зображен.

Також, використання розробленої програми є значно дешевшим для кінцевого споживача.

4.2 Розрахунок витрат на здійснення науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної, дослідноконструкторської, конструкторсько-технологічної роботи, створенням дослідного зразка і здійсненням виробничих випробувань, під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуються за такими *статтями:*

- витрати на оплату праці;
- відрахування на соціальні заходи;
- матеріали;
- паливо та енергія для науково-виробничих цілей;
- витрати на службові відрядження;
- спецустаткування для наукових (експериментальних) робіт;
- програмне забезпечення для наукових (експериментальних) робіт;
- витрати на роботи, які виконують сторонні підприємства, установи і організації;
- інші витрати;
- накладні (загальновиробничі) витрати.

4.2.1 Витрати на оплату праці

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, лаборантам, робітникам, студентам, аспірантам та іншим

працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці, також будь-які види грошових і матеріальних доплат, які належать до елемента «Витрати на оплату праці».

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховують відповідно до посадових окладів працівників, за формулою:

$$З_о = \sum_i = 1 \frac{k \times M_{ni} \times t_i}{T_p}, \quad (4.1)$$

де k – кількість посад дослідників, залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – кількість днів роботи конкретного дослідника, дн.;

T_p – середня кількість робочих днів в місяці, $T_p=21…23$ дні. Проведені розрахунки бажано звести до таблиці.

Оберемо кількість робочих днів в місяці 22 дні.

Таблиця 4.4 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Кількість днів роботи	Витрати на заробітну плату, грн
Керівник проекту	30 000	1363,6	44	60000
Науковий співробітник	25 000	1136,4	44	50000
Всього				110000

Так як в даному випадку розробляється програмний продукт, то розробник виступає одночасно і основним робітником, і тестувальником розроблюваного програмного продукту.

Додаткова заробітна плата дослідників та робітників

Додаткова заробітна плата розраховується як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_д = (З_о + З_р) \times \frac{Н_{дод}}{100\%}, \qquad (4.2)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати.

$$З_д = (110\ 000 * 12\ \% / 100\ \%) = 13\ 200 \text{ (грн.)}$$

4.2.2 Відрахування на соціальні заходи

До статті «Відрахування на соціальні заходи» належать відрахування внеску на загальнообов'язкове державне соціальне страхування та для здійснення заходів щодо соціального захисту населення (*ЄСВ – єдиний соціальний внесок*).

Нарахування на заробітну плату дослідників та робітників розраховується як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$Н_з = (З_о + З_р + З_{дод}) \times \frac{Н_{зп}}{100\%}, \qquad (4.3)$$

де $Н_{зп}$ – норма нарахування на заробітну плату.

$$Н_з = (110\ 000 + 13\ 200) \cdot 22\ \% / 100\ \% = 27104 \text{ (грн.)}$$

4.2.3 Програмне забезпечення для наукових робіт

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних

засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення.

До балансової вартості програмного забезпечення входять витрати на його інсталяцію, тому ці витрати беруться додатково в розмірі 10…12% від вартості програмного забезпечення.

Балансову вартість програмного забезпечення розраховують за формулою:

$$В_{прг} = \sum_{i=1}^{k} Ц_{іпрг} \cdot С_{прг.і} \cdot К_і \quad (4.4)$$

де $Ц_{іпрг}$ – ціна придбання одиниці програмного засобу цього виду, грн; $С_{прг.і}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.; $К_і$ – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо ($К_і$ = 1,10…1,12), в даному дослідженні використаємо $К_і$ = 1,10; k – кількість найменувань програмних засобів.

Отримані результати необхідно звести до таблиці 4.5.

Таблиця 4.5 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Intellij Idea 2018	2	20000	40000
			40000

4.2.4 Амортизація обладнання, програмних засобів та приміщень

До статті «Амортизація обладнання, програмних засобів та приміщень» відносять амортизаційні відрахування по кожному виду обладнання, устаткування та інших приладів і пристроїв, а також програмного забезпечення

для проведення науково-дослідної роботи, *за його наявності* в дослідній організації або на підприємстві.

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо можуть бути розраховані з використанням прямолінійного методу амортизації за формулою:

$$А_{обл} = \frac{Ц_б}{Т_в} \times \frac{Т_{вик}}{12}, \quad (4.5)$$

$$А_{обл} = \frac{49999}{2} \times \frac{2}{12} = 4166{,}58$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$Т_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

Проведені розрахунки необхідно звести до таблиці 4.6.

Таблиця 4.6 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Ноутбук Lenovo IdeaPad Gaming 316IAH7 *49999* грн	49 999	2	2	4 166,58
Приміщення	800 000	20	2	6 666,67
Офісне обладнання (стіл 3500, крісло 3500)	7 000	5	2	233,33
Ліцензійна ОС, та спеціалізовані ліцензійні нематеріальні ресурси Intellij Idea	20 000	10	2	333,33
Всього				11 399,91

4.2.5 Паливо та енергія для науково-виробничих цілей

До статті «Паливо та енергія для науково-виробничих цілей» належать витрати на придбання у сторонніх підприємств, установ і організацій будь-якого палива, що витрачається з технологічною метою на проведення досліджень. Стаття формується у разі виконання енергоємних наукових досліджень за методом прямого внесення витрат і досягає значної питомої ваги у собівартості досліджень.

Витрати на силову електроенергію (B_e) розраховують за формулою:

$$B_e = \sum_{i=1} \frac{W_{yi} \times t_i \times Ц_e \times К_{впі}}{\eta}, \quad (4.6)$$

де $^{W}_{yi}$ – встановлена потужність обладнання на певному етапі розробки, кВт; t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; Тарифи на розподіл електроенергії для всіх енергорозподільних компаній встановлює Національна комісія з регулювання енергетики і комунальних послуг (НКРЕКП), вартість для юридичних осіб В = 6,10 грн/кВт;

$К_{впі}$ – коефіцієнт, що враховує використання потужності, $К_{впі} < 1$; η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

Проведені розрахунки необхідно звести до таблиці.

Таблиця 4.7 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Ноутбук ASUS ROG Strix G15	0.096	352	158,52
Освітлення	0.024	352	46,38
Всього			231,9

4.2.6 Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «*Інші витрати*» розраховуються як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = \left(З_о + З_р\right) \times \frac{Н_{ів}}{100\%}, \quad (4.7)$$

де $Н_{ів}$ – норма нарахування за статтею «*Інші витрати*».

$$I_в = 110\,000 * 60\% / 100\% = 66\,000 \text{ (грн.)}$$

4.2.7 Накладні

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов’язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов’язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов’язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «*Накладні (загальновиробничі) витрати*» розраховуються як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$В_{нзв} = \left(З_о + З_р\right) \times \frac{Н_{нзв}}{100\%}, \quad (4.8)$$

де $Н_{нзв}$ – норма нарахування за статтею «*Накладні (загальновиробничі) витрати*».

$Н_{нзв}$ = 110 000 * 110% / 100% = 121 000 (грн.)

Витрати на проведення науково-дослідної роботи розраховуються як сума всіх попередніх статей витрат за формулою:

$$В_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв}. \quad (4.9)$$

$В_{заг}$ = 60 000 + 50 000 +13 200 + 27 104 + 40 000 + 11 399,91 + 231,9 + 66 000 + 121 000 = 388 935,81 грн.

Загальні витрати *ЗВ* на завершення науково-дослідної (науковотехнічної) роботи та оформлення її результатів розраховуються за формулою:

$$ЗВ = \frac{В_{заг}}{\eta}, \quad (4.10)$$

де η – коефіцієнт, який характеризує етап (стадію) виконання науководослідної роботи. Так, якщо науково-технічна розробка знаходиться на стадії: науково-дослідних робіт, то η=0,1; технічного проектування, то η =0,2; розробки конструкторської документації, то η=0,3; розробки технологій, то η=0,4; розробки дослідного зразка, то η=0,5; розробки промислового зразка, то η=0,7; впровадження, то η=0,9.

ЗВ = 388 935,81 / 0,5 = 777 871,62 грн.

4.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

В ринкових умовах узагальнювальним позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку. Саме зростання чистого прибутку забезпечить потенційному інвестору надходження додаткових коштів, дозволить покращити фінансові результати його діяльності, підвищить конкурентоспроможність та може позитивно вплинути на ухвалення рішення щодо комерціалізації цієї розробки.

Для того, щоб розрахувати можливе зростання чистого прибутку у потенційного інвестора від можливого впровадження науково-технічної розробки необхідно:

а) вказати, з якого часу можуть бути впроваджені результати науково-технічної розробки;

б) зазначити, протягом скількох років після впровадження цієї науково-технічної розробки очікуються основні позитивні результати для потенційного інвестора (наприклад, протягом 4-х років після її впровадження);

в) кількісно оцінити величину існуючого та майбутнього попиту на цю або аналогічні чи подібні науково-технічні розробки та назвати основних суб’єктів (зацікавлених осіб) цього попиту;

г) визначити ціну реалізації на ринку науково-технічних розробок з аналогічними чи подібними функціями.

При розрахунку економічної ефективності потрібно обов’язково враховувати зміну вартості грошей у часі, оскільки від вкладення інвестицій до отримання прибутку минає чимало часу.

При оцінюванні ефективності інноваційних проектів передбачається розрахунок таких важливих показників:

- ✓ абсолютного економічного ефекту (чистого дисконтованого доходу);

- ✓ внутрішньої економічної дохідності (внутрішньої норми дохідності);
- ✓ терміну окупності (дисконтованого терміну окупності).

Аналізуючи напрямки проведення науково-технічних розробок, розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором можна об'єднати, враховуючи визначені ситуації з відповідними умовами.

4.3.1 Розробка чи суттєве вдосконалення інформаційної технології для використання масовим споживачем

В цьому випадку майбутній економічний ефект буде формуватися на основі таких даних:

$$\Delta П_i = (\pm \Delta Ц_0 \cdot N + Ц_0 \cdot \pm \Delta N)_i \cdot \lambda \cdot \rho \cdot \left(1 - \frac{\vartheta}{100}\right), \qquad (4.11)$$

де $\pm \Delta Ц_o$ – зміна вартості програмного продукту (зростання чи зниження) від впровадження результатів науково-технічної розробки в аналізовані періоди часу;

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки;

$Ц_o$ – основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки, $Ц_o = Ц_б \pm \Delta Ц_o$;

Цб – вартість програмного продукту у році до впровадження результатів розробки;

ΔN – збільшення кількості споживачів продукту, в аналізовані періоди часу, від покращення його певних характеристик;

λ – коефіцієнт, який враховує сплату податку на додану вартість. Ставка податку на додану вартість дорівнює 20%, а коефіцієнт $\lambda = 0{,}8333$.

p – коефіцієнт, який враховує рентабельність продукту;

ϑ – ставка податку на прибуток, у 2022 році ϑ =18%.

Припустимо, що при прогнозованій ціні 5500 грн. за програмний продукт, термін збільшення прибутку складе 3 роки. Після завершення розробки і її вдосконалення, можна буде підняти її ціну на 500 грн. Кількість проданих підписок на програмний продукт також збільшиться: протягом першого року – на 1000 шт., протягом другого року – на 1500 шт., протягом третього року на 3000 шт. До моменту впровадження результатів наукової розробки реалізації продукту не було:

$\Delta П_1$ = (0*500 + (55000 + 500)*1000)* 0,8333* 0,3) * (1 - 0,18) = 1 229 950,8 грн.

$\Delta П_2$ = (0*500 + (55000 + 500)*(1000 + 1500)* 0,8333* 0,3) * (1 - 0,18) = 3 074 877 грн.

$\Delta П_3$ = (0*500 + (55000 + 500)*(1000 + 1500 + 3000)* 0,8333* 0,3) * (1 - 0,18) = 6 764 729,4 грн.

Отже, комерційний ефект від реалізації результатів розробки за три роки складе 11 069 557,2 грн.

Розраховуємо приведену вартість збільшення всіх чистих прибутків ПП, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-дослідної розробки:

$$ПП = \sum_1^T \frac{\Delta П_i}{(1+\tau)^t}, \quad (4.12)$$

де $\Delta П_i$ збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої науково-дослідної роботи, грн;

Т – період часу, протягом якою виявляються результати впровадженої науково-дослідної роботи, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ = 0,05…0,15;

t – період часу (в роках).

Збільшення прибутку ми отримаємо починаючи з першого року:

ПП= (1 229 950,8/ (1 + 0,1)[1]) + (3 074 877 / (1 + 0,1)[2]) + (6 764 729,4 / (1 + 0,1)[3]) = 1 118 137,09 + 2 541 220,66 + 5 082 441,32 = 8 741 799,07 грн.

Далі розраховують величину початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{\text{інв}} \times \text{ЗВ}, \quad (4.13)$$

де $k_{\text{інв}}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо; зазвичай $k_{\text{інв}}$ = 2...5, але може бути і більшим;

ЗВ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

PV = 2 * 777 871,62 = 1 555 743,24 грн.

Тоді абсолютний економічний ефект $E_{\text{абс}}$ або чистий приведений дохід (NPV, Net Present Value) для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{\text{абс}} = \text{ПП} - \text{PV}, \quad (4.14)$$

$E_{\text{абс}}$ = 8 741 799,07 – 1 601 050,5 = 7 186 055,83 грн.

Оскільки $E_{\text{абс}} > 0$ то вкладання коштів на виконання та впровадження результатів даної науково-дослідної (науково-технічної) роботи може бути доцільним.

Для остаточного прийняття рішення з цього питання необхідно розрахувати внутрішню економічну дохідність або показник внутрішньої

норми дохідності (IRR, Internal Rate of Return) вкладених інвестицій та порівняти її з так званою бар'єрною ставкою дисконтування, яка визначає ту мінімальну внутрішню економічну дохідність, нижче якої інвестиції в будь-яку науково-технічну розробку вкладати буде економічно недоцільно.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_{\text{в}}$. Для цього використаємо формулу:

$$E_{\text{в}} = \sqrt[\text{Т}_{\text{ж}}]{1 + \frac{E_{\text{абс}}}{PV}} - 1, \qquad (4.15)$$

$\text{Т}_{\text{ж}}$ життєвий цикл наукової розробки, роки.

$$E_{\text{в}} = \sqrt[3]{(1 + 7\,186\,055{,}83 \div 1\,555\,743{,}24)} - 1 = 0{,}78$$

Визначимо мінімальну ставку дисконтування, яка у загальному вигляді визначається за формулою:

$$\tau = d + f, \qquad (4.16)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2022 році в Україні d = (0,23…0,25);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,5).

$$\tau_{min} = 0{,}25 + 0{,}05 = 0{,}29.$$

Так як $\text{Е}_{\text{в}} > \tau_{min}$, то інвестор може бути зацікавлений у фінансуванні даної наукової розробки.

Розрахуємо термін окупності вкладених у реалізацію наукового проекту інвестицій за формулою:

$$\text{Т}_{\text{ок}} = \frac{1}{\text{Е}_{\text{в}}}, \qquad (4.17)$$

$$T_{ок} = 1 / 0,78 = 1,28 \text{ р.}$$

Оскільки < 3-х років, а саме термін окупності рівний 1,28 роки, то фінансування даної наукової розробки є доцільним.

4.4 Висновок до розділу 4

Економічна частина даної роботи містить розрахунок витрат на розробку ої інформаційної технології інтелектуальної обробки зображень, сума яких складає 777 871,62 гривень. Було спрогнозовано орієнтовану величину витрат по кожній з статей витрат. Також розраховано чистий прибуток, який може отримати виробник від реалізації нового технічного рішення та економічний ефект при використанні даної розробки. В результаті аналізу розрахунків можна зробити висновок, що розроблений програмний продукт за ціною дешевший за аналог і є високо конкурентоспроможним.

ВИСНОВКИ

У ході виконання магістерської кваліфікаційної роботи реалізовано інформаційну технологію інтелектуальної обробки зображень. Для досягнення

мети були поставлені такі завдання: провести аналіз програм аналогів, систем-аналогів та обрґрунтуваня доцільності розробки інформаційної технології інтелектуальної обробки зображень, розробити математичну модель інтелектуальної обробки зображень; розробити структуру інформаційної технології інтелектуальної обробки зображень; здійснити програмну реалізацію інформаційної технології інтелектуальної обробки зображень; провести тестування програми та проаналізувати отримані результати; економічно обґрунтувати доцільність розробки інформаційної технології інтелектуальної обробки зображень, *які виконані в повному обсязі*.

Обґрунтовано доцільність розробки інформаційної технології інтелектуальної обробки зображень. Проведено аналіз сучасних аналогів та систем-аналогів, які виконують функції обробки зображень та побудови гістограм. Виявлено проблему відсутності інтуїтивно зрозумілого інтерфейсу, високої ціни за користування додатком, а також недостатньо високу швидкість роботи розглнутих програм, що довело актуальність розробки інформаційної технології інтелектуальної обробки зображень.

На основі аналізу існуючих моделей та методів, що застосовуються для вирішення поставленої задачі було обґрунтовано використання в інформаційній технології алгоритмів побудови гістограми, поєднавши алгоритми аналізу спектру зображення та методу вирівнювання гістограми на основі нечіткої логіки, що дозволяє максимально задовольнити потреби користувача та підвищити швидкість обробки зображення.

Було розроблено структуру інформаційної технології інтелектуального аналізу зображень та спроектовано схему алгоритму роботи розроблюваної програми. Розроблено алгоритми функціонування інформаційної технології та алгоритм розбиття зображення на пікселі.

Здійснено проектування та розробку інформаційної технології інтелектуальної обробки зображень. Обґрунтовано вибір мови програмування. Розробка інформаційної технології велася на об'єктно-орієнтованій мові програмування Java в середовищі розробки IntelliJ IDEA. Здійснено програмну

реалізацію інформаційної технології інтелектуальної обробки зображень, детально описано класи та функції, які використовуються в інформаційній технології.

Проведено тестування роботи інформаційної технології інтелектуальної обробки зображень та проаналізовано її результати. Тестування програми підтвердило очікувані результати та правильність роботи програми, довівши підвищення швидкодії обробки зображення на 0,5с.

Здійснено економічне обґрунтування доцільності розробки інформаційної технології інтелектуальної обробки зображень. Було спрогнозовано орієнтовану величину витрат по кожній з статей витрат. Також розраховано чистий прибуток, який може отримати виробник від реалізації нового технічного рішення, знайдено термін окупності витрат для виробника та економічний ефект для споживача при використанні такої розробки. В результаті аналізу розрахунків можна зробити висновок, що розробка у виробництві та використання дешевша за аналог і є висококонкурентоспроможною. Період окупності складе близько роки. Загальні витрати становлять 777 871,62 грн.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Побережник В. Р., Колодний В.В. «Information technology of intelligent image processing» в Матеріали конференції «Всеукраїнська науково-практична інтернет-конференція студентів, аспірантів та молодих

науковців ”МОЛОДЬ В НАУЦІ: ДОСЛІДЖЕННЯ, ПРОБЛЕМИ, ПЕРСПЕКТИВИ” (2023)» Вінниця, 2022. [Електронний ресурс]. Режим доступу:https://conferences.vntu.edu.ua/index.php/mn/mn2023/author/submission/16909. Дата звернення: листопад 2022.

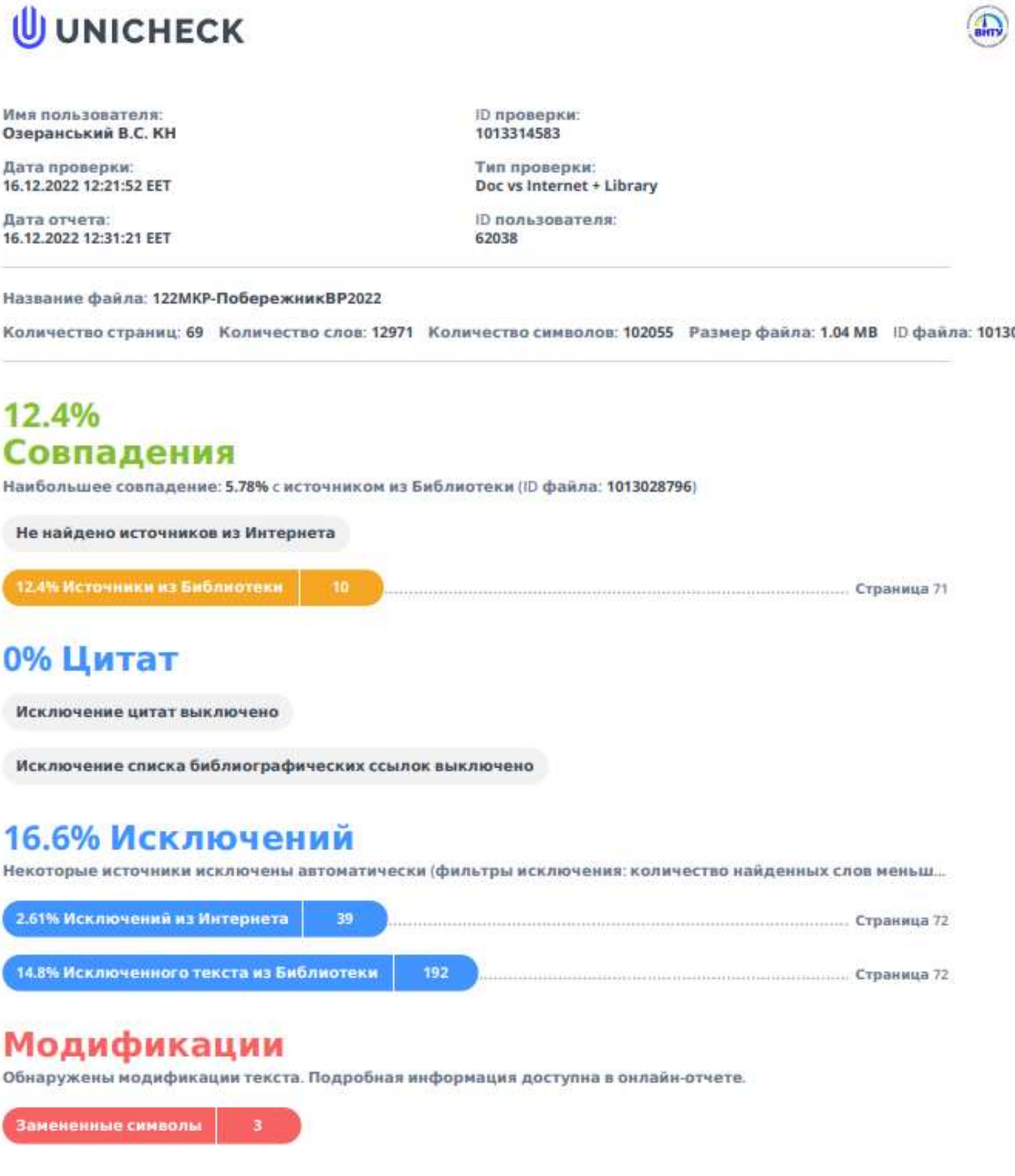
2. Талах М. В. Інтелектуальний аналіз сигналів та зображень : навч. посіб. Чернівці : ЧНУ ім. Ю. Федьковича, 2015. 326 с.
3. Суботін С.О. Подання й обробка знань у системах штучного інтелекту та підтримки прийняття рішень : навч. посіб. Запоріжжя : ЗНТУ, 2008. - 341 с.
4. Dash L., Chatterji B.N. Adaptive contrast enhancement and de-enhancement // Pattern Recognition, 1992. V. 24. № 4. P.289–302.
5. Зайченко Ю П. Основи проектування інтелектуальних систем : навч. посіб. Київ : Слово, 2004. 352 с.
6. Яровий А. А. Методологічні особливості побудови паралельно-ієрархічних та ієрарх-ієрархічних мереж на основі кластерних систем з розподіленою обробкою інформації : Оптико-електронні інформаційно-енергетичні технології. 2010. № 1 (19). С. 69 – 79.
7. Кутковецький В. Я. Розпізнавання образів : навч. посіб. Миколаїв : МДГУ ім. П. Могили, 2003. 196 с.
8. Березький О.М. Інтелектуальна система автоматизованої мікроскопії аналізу гістологічних та цитологічних зображень / О.М. Березький, О.Й. Піцун, П.Б. Лящинський, П.Б. Лящинський, Г.М. Мельник // Штучний інтелект. 2017. № 2. С. 129-141.
9. Sukop, M., & Ferenčík, P. (2020). Розробка інтелектуального роботизованого модуля з камерною системою. Технічні науки та технології, (3(17), 096–104.
10. Косаревич Р. Я. Обробка зображень методами штучного інтелекту : навч. посіб. Львів : НУЛП, 2015. 452 с.
11. Munteanu C., Rosa A. Gray-Scale Image Enhancement as an Automatic Process Driven by Evolution // IEEE Transactions on Systems, Man, and Cybernetics. – 2004. – V. 34. – P. 1292–1298.

12. Воробель Р.А. Цифровая обработка изображений на основе теории контрастности: дис. докт. техн. наук: 05.13.06. Львів, 1999. 369 с.
13. Kobylin, O., Gorokhovatskyi, V., Tvoroshenko, I., and Peredrii, O. The application of non-parametric statistics methods in image lassifiers based on structural description components. Telecommunications and Radio Engineering. 2020. 79(10), pp. 855–863.
14. Суботін С.О. Подання й обробка знань у системах штучного інтелекту та підтримкиприйняття рішень: навч. посіб. Запоріжжя : ЗНТУ, 2008. 341 с.
15. Творошенко І.С. Конспект лекцій з дисципліни «Цифрова обробка зображень». Харків: ХНУМГ ім. О.М. Бекетова, 2015. 75 с.
16. Volodymyr Gorokhovatskyi, and Iryna Tvoroshenko Image Classification Based on the Kohonen Network and the Data Space Modification. In CEUR Workshop Proceedings: Computer Modeling and Intelligent Systems (CMIS-2020). 2020. 2608. pp. 1013–1026.
17. Tvoroshenko, I.S., and Kramarenko, O.O. Software determination of the optimal route by geoinformation technologies. Radio Electronics Computer Science Control. 2019. Vol. 3. pp. 131–142.
18. Удовенко С.Г., Шамраєв А.А., Дудінова О.Б. Модифікований метод тональної корекції цифрових зображень. ІТ-Тренди: соціальні медіа, великі дані, штучний інтелект: тези доповідей 2-го Міжнародного форуму. Т. 1. Кременчук: Кременчуцький національний університет імені М. Остроградського, 2015. С. 8–9.
19. Шамраєв А.А., Шамраєва Е.О., Дудінова О.Б. Метод комп'ютерної обробки цифрових аерофотознімків. Системи обробки інформації. 2014. Вип. 7(123). С. 168–171.
20. Глухова Н.В. Виявлення інформативних ознак зображень на базі аналізу гістограм яскравості. Вчені записки ТНУ ім. В.І. Вернадського. Серія: технічні науки. Том 31 (70) №4, 2020. С. 75 – 80.
21. Ярема С.М., Гавва О.М. Етикетка: Навч. посіб. – К.: Ун-т «Україна», НУХТ, 2007. – 635с.

22. IC Measure - manual on-screen image measurement and image acquisition – [Електронний ресурс] – Режим доступу: https://www.theimaging source.com/support/downloads-for-windows/end-user-software/icmeasure/.
23. GemIdent 1.0 User Manual – [Електронний ресурс]. – Режим доступу: https://www.academia.edu/2788539/GemIdent_1_0_User_Manual.
24. User-friendly software for analyzing large images – [Електронний ресурс]. – Режим доступу: https://jmicrovision.github.io/.
25. Enlightenment Foundation Libraries – [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Enlightenment_Foundation_Libraries.
26. PhotoMania – [Електронний ресурс]. – Режим доступу: https://play.google.com/store/apps/details?id=com.photomania.app&hl=uk&gl=US&pli=1.
27. Глухова Н.В., Пісоцька Л.А. Інформаційна технологія для аналізу кольорових зображень газорозрядного випромінювання. Перспективні технології та прилади. № 12. 2018. С. 48–52.
28. Горошко А.В. Розрахунок допустимих значень параметрів об’єктів у випадку полімодальності їх ймовірнісних розподілів / А.В. Горошко, Ройзман В.П. Вібрації в техніці та технологіях. № 4(72). 2013. С. 19–27.
29. Борисенко Д. И. Методы поиска угловых особенностей на изображениях // Молодой ученый, 2011. №5. Т.1. С. 120-123.
30. Данілов В.Я., Кушніренко С.В. Теорія ймовірностей і математична статистика: навч. посіб. Кам’янець-Подільський: ПП Мошак М.І., 2009. 112 с.
31. Данілов В.Я., Кушніренко С.В. Математична статистика. навч. посіб. ВГЛ "Обрії", 2012. 152 с.
32. Творошенко І.С. Конспект лекцій з дисципліни «Цифрова обробка зображень». Харків: ХНУМГ ім. О.М. Бекетова, 2015. 75 с.
33. Шилдт Г. C# 4.0: повне керівництво: навч. посіб. Київ : Фабула, 2019, 448с.
34. Java Introduction – [Електронний ресурс] – Режим доступу: https://www.w3schools.com/java/java_intro.asp.

35. Сьєрра К. Бейтс Берт. Head First. Java. Київ : Фабула, 2022. 720 с.
36. Почему IntelliJ IDEA. Продуктивная разработка на Java – [Електронний ресурс] – Режим доступу: https://www.jetbrains.com/ru-ru/idea/.
37. Java Swing Tutorial – [Електронний ресурс] – Режим доступу: https://www.javatpoint.com/java-swing.
38. Графіка засобами Java – [Електронний ресурс] – Режим доступу: http://www.kievoit.ippo.kubg.edu.ua/kievoit/2016/Java_Picture/index.html.
39. Spring Framework – [Електронний ресурс] – Режим доступу: https://spring.io/projects/spring-framework.

Додаток А



UNICHECK

Имя пользователя:
Озеранський В.С. КН

ID проверки:
1013314583

Дата проверки:
16.12.2022 12:21:52 EET

Тип проверки:
Doc vs Internet + Library

Дата отчета:
16.12.2022 12:31:21 EET

ID пользователя:
62038

Название файла: 122МКР-ПобережникВР2022

Количество страниц: 69 Количество слов: 12971 Количество символов: 102055 Размер файла: 1.04 MB ID файла: 1013072981

12.4%
Совпадения

Наибольшее совпадение: 5.78% с источником из Библиотеки (ID файла: 1013028796)

Не найдено источников из Интернета

12.4% Источники из Библиотеки 10 ..... Страница 71

0% Цитат

Исключение цитат выключено

Исключение списка библиографических ссылок выключено

16.6% Исключений

Некоторые источники исключены автоматически (фильтры исключения: количество найденных слов меньш...

2.61% Исключений из Интернета 39 ..... Страница 72

14.8% Исключенного текста из Библиотеки 192 ..... Страница 72

Модификации

Обнаружены модификации текста. Подробная информация доступна в онлайн-отчете.

Замененные символы 3

Додат

Клас Main

```
import java.io.File;

public class Main {

	public static int nomK, nomG, max;
	public static File img;
	public static int col[] = new int[256];
	public static int persent;

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		MainG maG = new MainG();

		maG.stvor();

	}

}
```

Клас MainG

```
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.ResultSet;
import java.sql.SQLException;

import javax.swing.*;

public class MainG {
	public static ImageIcon image;
	public static JFrame frame;
	public static JLabel label;

	public static JComboBox combo;

	// this.EXIT_ON_CLOSE;
	MainG() {
		image = new ImageIcon(getClass().getResource("f1.jpg"));
	}

	public void stvor() {
		frame = new JFrame();
		frame.setTitle("Обробка зображень");
		frame.setSize(400, 300);
```

```
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.setLocationRelativeTo(null);

            Font font = new Font("TimesRoman", Font.BOLD, 30);
            Font font1 = new Font("TimesRoman", Font.BOLD, 24);
            // zaput.setBackground(Color.RED);
            label = new JLabel();
            label.setLayout(new GridBagLayout());
            label.setVisible(true);
            label.setBackground(Color.BLUE);
            label.setIcon(image);

            JButton btnAuto = new JButton();

            btnAuto.setText("Обрати зображення");
            btnAuto.setFont(font);
            label.add(btnAuto, new GridBagConstraints(0, 1, 1, 1, 0.0, 0.9,
GridBagConstraints.NORTH,
                        GridBagConstraints.HORIZONTAL, new Insets(100, 10, 10, 10), 0, 0));

            btnAuto.addActionListener(new Listener());

            frame.add(label);
            frame.setVisible(true);

        }

        public void shov() {
            frame.setVisible(false);
            frame.remove(label);

        }

        public class Listener implements ActionListener {

            @Override
            public void actionPerformed(ActionEvent JCom) {
                JFileChooser fileChooser = new JFileChooser();
                fileChooser.showDialog(label, "Open");

                Main.img=fileChooser.getSelectedFile();
                shov();

                Men m = new Men();
                m.stvor();
            }
        }

}
```

Клас Men

```
import java.awt.Color;
```

```
import java.awt.Font;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;

public class Men {
	public static ImageIcon image;
	public static JFrame frame;
	public static JLabel label;

	public JTextField textLogin;
	public JPasswordField textPassword;

	// this.EXIT_ON_CLOSE;
	Men() {
		image = new ImageIcon(getClass().getResource("f1.jpg"));
	}

	public void stvor() {
		frame = new JFrame();
		frame.setTitle("Обробка зображень");
		frame.setSize(800, 600);
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		frame.setLocationRelativeTo(null);

		Font font = new Font("TimesRoman", Font.BOLD, 30);
		Font font1 = new Font("TimesRoman", Font.BOLD, 24);
		// zaput.setBackground(Color.RED);
		label = new JLabel();
		label.setLayout(new GridBagLayout());
		label.setVisible(true);
		label.setBackground(Color.BLUE);
		label.setIcon(image);

		JButton btnAuto = new JButton();
		JButton btnReg = new JButton();
		btnAuto.setText("Гістограма по червоному каналу");
		btnAuto.setFont(font);
		label.add(btnAuto, new GridBagConstraints(0, 0, 1, 1, 0.0, 0.9,
GridBagConstraints.NORTH,
				GridBagConstraints.HORIZONTAL, new Insets(30, 10, 10, 10), 0, 0));

		btnReg.setText("Гістограма по зеленому каналу");
```

```
            btnReg.setFont(font);
            label.add(btnReg, new GridBagConstraints(0, 1, 1, 1, 0.0, 0.9,
GridBagConstraints.NORTH,
                    GridBagConstraints.HORIZONTAL, new Insets(30, 10, 10, 10), 0, 0));

            JButton btnReg1 = new JButton();
            btnReg1.setText("Гістограма по синьому каналу");
            btnReg1.setFont(font);
            label.add(btnReg1, new GridBagConstraints(0, 2, 1, 1, 0.0, 0.9,
GridBagConstraints.NORTH,
                    GridBagConstraints.HORIZONTAL, new Insets(30, 10, 10, 10), 0, 0));

            JButton btnReg2 = new JButton();
            btnReg2.setText("Гістограма яскравості");
            btnReg2.setFont(font);
            label.add(btnReg2, new GridBagConstraints(0, 3, 1, 1, 0.0, 0.9,
GridBagConstraints.NORTH,
                    GridBagConstraints.HORIZONTAL, new Insets(30, 10, 10, 10), 0, 0));

            JButton btnReg3 = new JButton();
            btnReg3.setText("Змінити яскравість");
            btnReg3.setFont(font);
            label.add(btnReg3, new GridBagConstraints(0, 4, 1, 1, 0.0, 0.9,
GridBagConstraints.NORTH,
                    GridBagConstraints.HORIZONTAL, new Insets(30, 10, 10, 10), 0, 0));
            btnAuto.addActionListener(new Listener1());
            btnReg.addActionListener(new Listener1());
            btnReg1.addActionListener(new Listener1());
            btnReg2.addActionListener(new Listener1());
            btnReg3.addActionListener(new Listener1());

            frame.add(label);
            frame.setVisible(true);

    }

    public void shov() {
            frame.setVisible(false);
            frame.remove(label);

    }

    public class Listener1 implements ActionListener {

            @Override
            public void actionPerformed(ActionEvent JCom) {
                    String button = ((JButton) JCom.getSource()).getText();
                    if (button.equals("Гістограма по червоному каналу")) {
                            Main.nomG = 1;
                            Men m = new Men();
                            m.shov();
                            Gist g = new Gist();
```

```
                        g.stvor();
                    } else {
                        if (button.equals("Гістограма по зеленому каналу")) {
                            Main.nomG = 2;
                            Men m = new Men();
                            m.shov();
                            Gist g = new Gist();
                            g.stvor();
                        } else {
                            if (button.equals("Гістограма по синьому каналу")) {
                                Main.nomG = 3;
                                Men m = new Men();
                                m.shov();
                                Gist g = new Gist();
                                g.stvor();
                            } else {
                                if (button.equals("Гістограма яскравості")) {
                                    Main.nomG = 4;
                                    Men m = new Men();
                                    m.shov();
                                    Gist g = new Gist();
                                    g.stvor();
                                } else {
                                    shov();
                                    new PersentPage().stvor();
//                                  Змінити яскравість
                                }
                            }
                        }
                    }
                }
            }
        }
```

Клас Gist

```
import java.awt.Color;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.Insets;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import java.sql.ResultSet;
import java.sql.SQLException;

import javax.imageio.ImageIO;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
```

```
public class Gist {
	public static ImageIcon image;
	public static JFrame frame;
	public static int day, mounth, year;
	public static String dat;

	Gist() {
		image = new ImageIcon(getClass().getResource("f1.jpg"));
		// setTitle("Прогнозування завантаженості сервера");
		// setSize(800, 600);
		// setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		// setLocationRelativeTo(null);
		// setVisible(true);
	}

	public void stvor() {
		frame = new JFrame();
		String s="";
		if(Main.nomG==1){
			s="Червоний канал";
		}
		if(Main.nomG==2){
			s="Зелений канал";
		}
		if(Main.nomG==3){
			s="Синій канал";
		}
		if(Main.nomG==4){
			s="Гістограма яскравості";
		}
		frame.setTitle(s);

		frame.setSize(800, 600);
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		frame.setLocationRelativeTo(null);
		frame.setLayout(new GridBagLayout());
		try {

			// Открываем изображение
			File file = Main.img;
			BufferedImage source = ImageIO.read(file);

			// Создаем новое пустое изображение, такого же размера
			BufferedImage result = new BufferedImage(source.getWidth(), source.getHeight(),
source.getType());

			// Делаем двойной цикл, чтобы обработать каждый пиксель
			for (int x = 0; x < source.getWidth(); x++) {
				for (int y = 0; y < source.getHeight(); y++) {

					// Получаем цвет текущего пикселя
					Color color = new Color(source.getRGB(x, y));
```

```
                        // Получаем каналы этого цвета
                        int blue = color.getBlue();
                        int red = color.getRed();
                        int green = color.getGreen();
                        if (Main.nomG == 1) {
                                Main.col[red]++;
                        }
                        if (Main.nomG == 2) {
                                Main.col[green]++;
                        }
                        if (Main.nomG == 3) {
                                Main.col[blue]++;
                        }
                        if (Main.nomG == 4) {
                                int y1 = (int) (0.2126 * red + 0.7152 * green + 0.0722 * blue);
                                Main.col[y1]++;
                        }

                }
        }
        Main.max = 0;
        for (int i = 0; i < 255; i++) {
                if (Main.col[i] > Main.max) {
                        Main.max = Main.col[i];
                }
        }

    } catch (IOException e) {

        // При открытии и сохранении файлов, может произойти неожиданный
        // случай.
        // И на этот случай у нас try catch
        System.out.println("Файл не найден или не удалось сохранить");
    }

    frame.setVisible(true);

    GraphicsPanel graphicsPanel = new GraphicsPanel();

    frame.add(graphicsPanel, new GridBagConstraints(0, 0, 1, 1, 1, 1,
GridBagConstraints.CENTER,
                    GridBagConstraints.BOTH, new Insets(0, 0, 0, 0), 0, 0));
    frame.setVisible(true);

  }

}
```

Клас GraphicsPanel

```
import java.awt.BasicStroke;
import java.awt.Color;
```

```
import java.awt.Font;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Rectangle;
import java.awt.geom.RoundRectangle2D.Double;
import java.util.Random;

import javax.swing.JPanel;

import org.w3c.dom.css.Rect;

public class GraphicsPanel extends JPanel {

	private Line line;

	public GraphicsPanel() {
		line = new Line(25, 100, 30, 20);
	}

	@Override
	public void paintComponent(Graphics g) {
		Graphics2D g2 = (Graphics2D) g;
		// g2.draw(line);
		Color newColor = new Color(139, 0, 0);
		g2.setStroke(new BasicStroke(6.0f));
		g2.setColor(newColor);
		g2.drawLine(9, 530, 777, 530);

		Font font1 = new Font("TimesRoman", Font.BOLD, 14);
		Font font = new Font("TimesRoman", Font.BOLD, 10);
		Font font2 = new Font("TimesRoman", Font.BOLD, 18);

		g2.setStroke(new BasicStroke(3.0f));
		Color newColor1 = new Color(0, 58, 250, 100);
		g2.setColor(newColor1);
		for (int i = 0; i < 255; i++) {
			float h = 500f / (float) Main.max * (float) Main.col[i];
			int hh = (int) h;
			g2.drawLine(9 + i * 3, 527, 9 + i * 3, 527 - hh);
			Main.col[i]=0;
		}
		// g2.setStroke(new BasicStroke(6.0f));
		/*
		 * g2.setColor(newColor1); g2.drawLine(30, 505, 750, 505);
		 * g2.drawLine(30, 480, 750, 480); g2.drawLine(30, 455, 750, 455);
		 * g2.drawLine(30, 430, 750, 430); g2.drawLine(30, 405, 750, 405);
		 * g2.drawLine(30, 380, 750, 380); g2.drawLine(30, 355, 750, 355);
		 * g2.drawLine(30, 330, 750, 330); g2.drawLine(30, 305, 750, 305);
		 * g2.drawLine(30, 280, 750, 280); g2.drawLine(30, 255, 750, 255);
		 * g2.drawLine(30, 230, 750, 230); g2.drawLine(30, 205, 750, 205);
		 * g2.drawLine(30, 180, 750, 180); g2.drawLine(30, 155, 750, 155);
		 * g2.drawLine(30, 130, 750, 130); g2.drawLine(30, 105, 750, 105);
```

```
                * g2.drawLine(30, 80, 750, 80); g2.drawLine(30, 55, 750, 55);
                * g2.setColor(Color.BLUE);
                */

               // g2.drawRect(30, 50, 250, 250);

               // g2.drawLine(750, 530, 750, 30);

        }

}
```

Клас Image

```
import javax.imageio.ImageIO;
import java.io.File;
import java.awt.Color;
import java.awt.image.BufferedImage;
import java.io.IOException;

public class Image {
   public static void main(String[] args) {
      try {

         // Открываем изображение
         File file = Main.img;
         BufferedImage source = ImageIO.read(file);

         // Создаем новое пустое изображение, такого же размера
         BufferedImage result = new BufferedImage(source.getWidth(), source.getHeight(),
source.getType());

         // Делаем двойной цикл, чтобы обработать каждый пиксель
         for (int x = 0; x < source.getWidth(); x++) {
            for (int y = 0; y < source.getHeight(); y++) {

               // Получаем цвет текущего пикселя
               Color color = new Color(source.getRGB(x, y));

               // Получаем каналы этого цвета
               int blue = color.getBlue();
               int red = color.getRed();
               int green = color.getGreen();

               // Применяем стандартный алгоритм для получения черно-белого изображения
               int grey = (int) (red * 0.299 + green * 0.587 + blue * 0.114);
               int newRed = grey;
               int newGreen = grey;
               int newBlue = grey;

               Color newColor = new Color(newRed, newGreen, newBlue);

               result.setRGB(x, y, newColor.getRGB());
            }
```

```
            }

            // Созраняем результат в новый файл
            File output = new File("katana_grey.jpg");
            ImageIO.write(result, "jpg", output);

        } catch (IOException e) {

            // При открытии и сохранении файлов, может произойти неожиданный случай.
            // И на этот случай у нас try catch
            System.out.println("Файл не найден или не удалось сохранить");
        }
    }
}
```

Клас Line

```
import java.awt.geom.Line2D;
import java.awt.geom.Point2D;
import java.awt.geom.Rectangle2D;

public class Line extends Line2D{

        private Point p1;
        private Point p2;

        public Line(){
                p1 = new Point();
                p2 = new Point();
        }

        public Line(double x1, double y1, double x2, double y2){
                p1 = new Point(x1, y1);
                p2 = new Point(x2, y2);
        }

        @Override
        public Rectangle2D getBounds2D() {
                // TODO Auto-generated method stub
                return null;
        }

        @Override
        public Point2D getP1() {
                // TODO Auto-generated method stub
                return p1;
        }

        @Override
        public Point2D getP2() {
                // TODO Auto-generated method stub
```

```
		return p2;
	}

	@Override
	public double getX1() {
		// TODO Auto-generated method stub
		return p1.getX();
	}

	@Override
	public double getX2() {
		// TODO Auto-generated method stub
		return p2.getX();
	}

	@Override
	public double getY1() {
		// TODO Auto-generated method stub
		return p1.getY();
	}

	@Override
	public double getY2() {
		// TODO Auto-generated method stub
		return p2.getY();
	}

	@Override
	public void setLine(double x1, double y1, double x2, double y2) {
		// TODO Auto-generated method stub
		p1.setLocation(x1, y1);
		p2.setLocation(x2, y2);

	}

}
```

Клас Point

```
import java.awt.geom.Point2D;

public class Point extends Point2D{
	private double x;
	private double y;

	public Point() {
		// TODO Auto-generated constructor stub
	}
```

```
        public Point(double x, double y) {
                this.x=x;
                this.y=y;
        }

        @Override
        public double getX() {
                // TODO Auto-generated method stub
                return x;
        }

        @Override
        public double getY() {
                // TODO Auto-generated method stub
                return y;
        }

        @Override
        public void setLocation(double arg0, double arg1) {
                // TODO Auto-generated method stub
                this.x=x;
                this.y=y;
        }

}
```

Клас Point

```
import javax.imageio.ImageIO;
import java.awt.*;
import java.awt.image.BufferedImage;
import java.io.*;
import java.util.ArrayList;
import java.util.Scanner;

public class Processing {

   private File file1;
   private BufferedImage imgIn;
   int number;
   private ArrayList<Color> imgInColor = new ArrayList<>();

   public Processing() {
      this.file1 = Main.img;
      try {
         imgIn = ImageIO.read(file1);
      } catch (IOException e) {
         e.printStackTrace();
      }
   }
```

```
public int brightnessChange() {
    System.out.println(number);
    System.out.println("f1");
    for (int i = 0; i < imgIn.getWidth(); i++) {
        for (int j = 0; j < imgIn.getHeight(); j++) {
            imgInColor.add(new Color(imgIn.getRGB(i, j)));
        }
    }


    System.out.println("f2");
    number += 12;
    System.out.println(number);

    int nu = 0;
    BufferedImage bf = new BufferedImage(imgIn.getWidth(), imgIn.getHeight(), 1);
    for (int i = 0; i < imgIn.getWidth(); i++) {
        for (int j = 0; j < imgIn.getHeight(); j++) {
            bf.setRGB(i, j, imgInColor.get(nu++).getRGB());
        }
    }
    changing();
    System.out.println("f4");
    try {
        ImageIO.write(bf, "png", file1);
    } catch (IOException e) {
        e.printStackTrace();
    }
    return number;
}
private void changing() {
    for (int i = 0; i < imgInColor.size(); i++) {
        int R = imgInColor.get(i).getRed();
        int G = imgInColor.get(i).getGreen();
        int B = imgInColor.get(i).getBlue();
        int newR = R + (R * Main.persent / 100);
        int newB = B + (B * Main.persent / 100);
        int newG = G + (G * Main.persent / 100);
        if (newR > 255) {
            newR = 255;
        }
        if (newR < 0) {
            newR = 0;
        }
        if (newG > 255) {
            newG = 255;
        }
        if (newG < 0) {
            newG = 0;
        }
        if (newB > 255) {
            newB = 255;
```

```
            }
            if (newB < 0) {
                newB = 0;
            }
            Color color1 = new Color(newR, newB, newG);
            imgInColor.set(i, color1);
        }
    }

}
```

Д

о

д

а

110

Додаток В (обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

«ІНТЕЛЕКТУАЛЬНА СИСТЕМА ОБРОБКИ ЗОБРАЖЕНЬ»

Виконав: студент 2-го курсу, групи КН-21м
Спеціальності 122 – «Комп`ютерні науки»
Побережник В. Р.
(прізвище та ініціали)

Керівник: д.т.н., доц. каф. КН
Колодний В.В.
(прізвище та ініціали)
«15» 12 2022 р.

Вінниця ВНТУ - 2022 рік

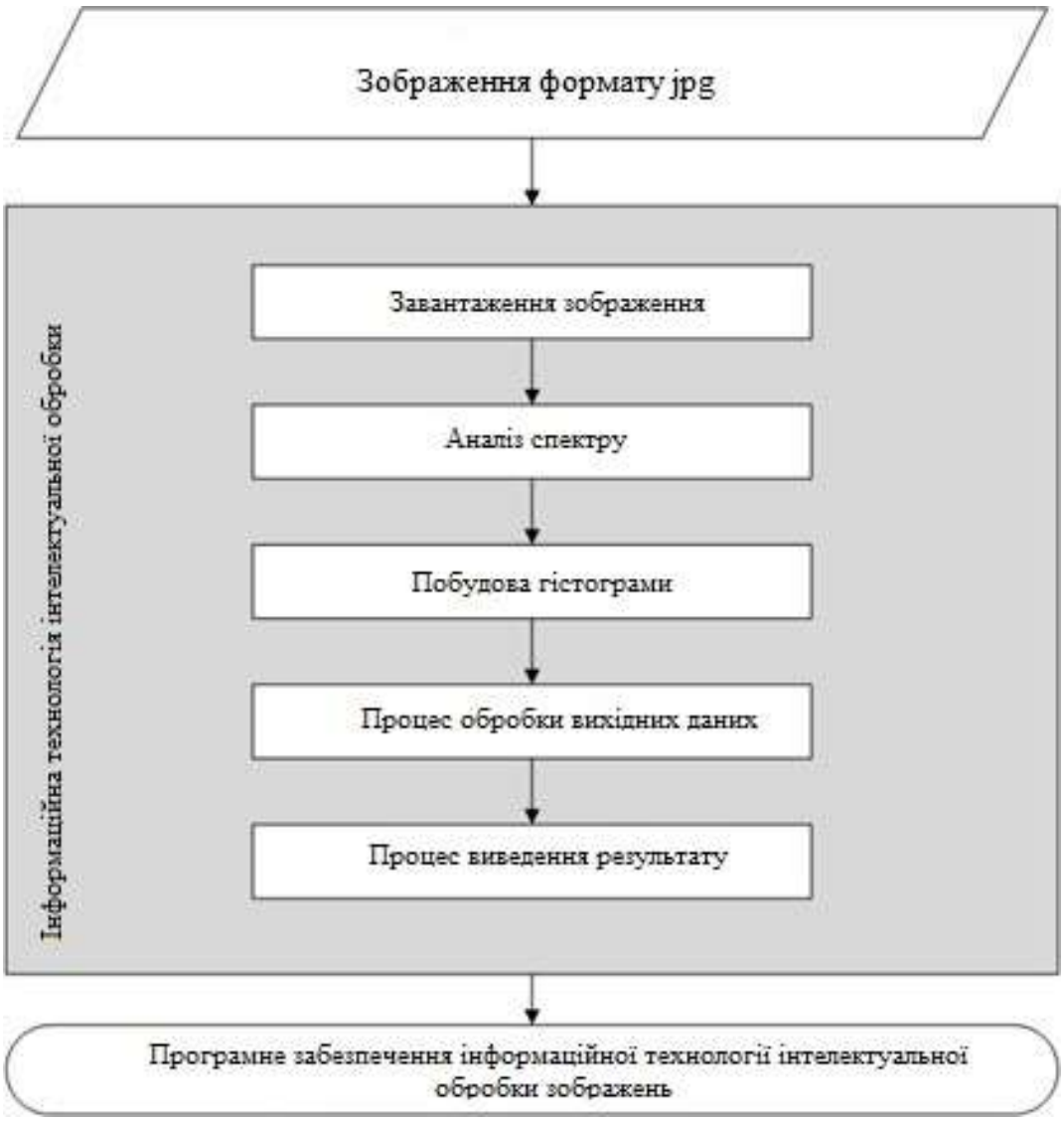


Рисунок В.1 – Структурна схема інформаційної технології

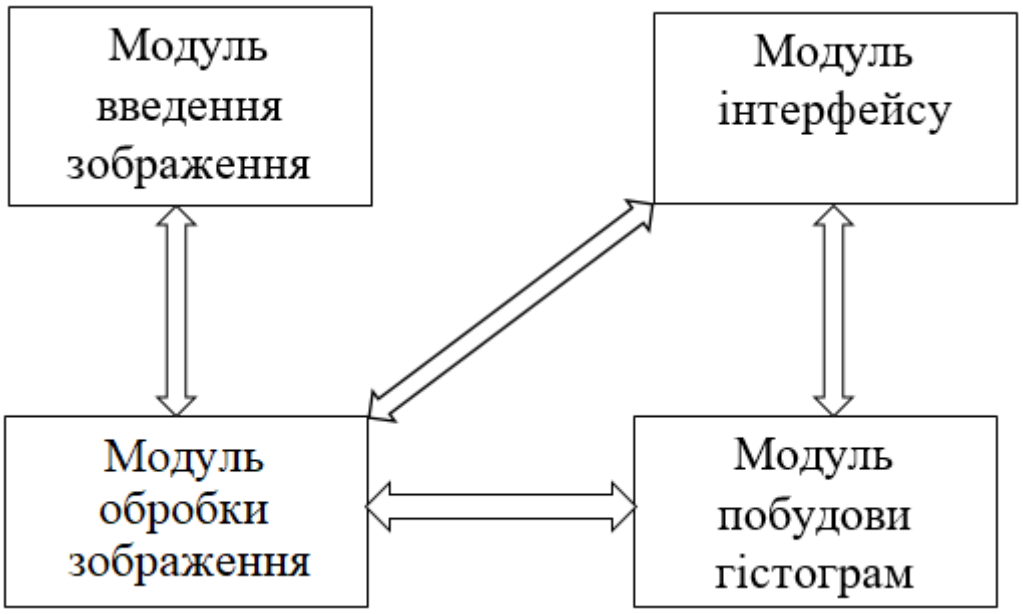


Рисунок В.2 – Схема моделі функціонування інформаційної технології

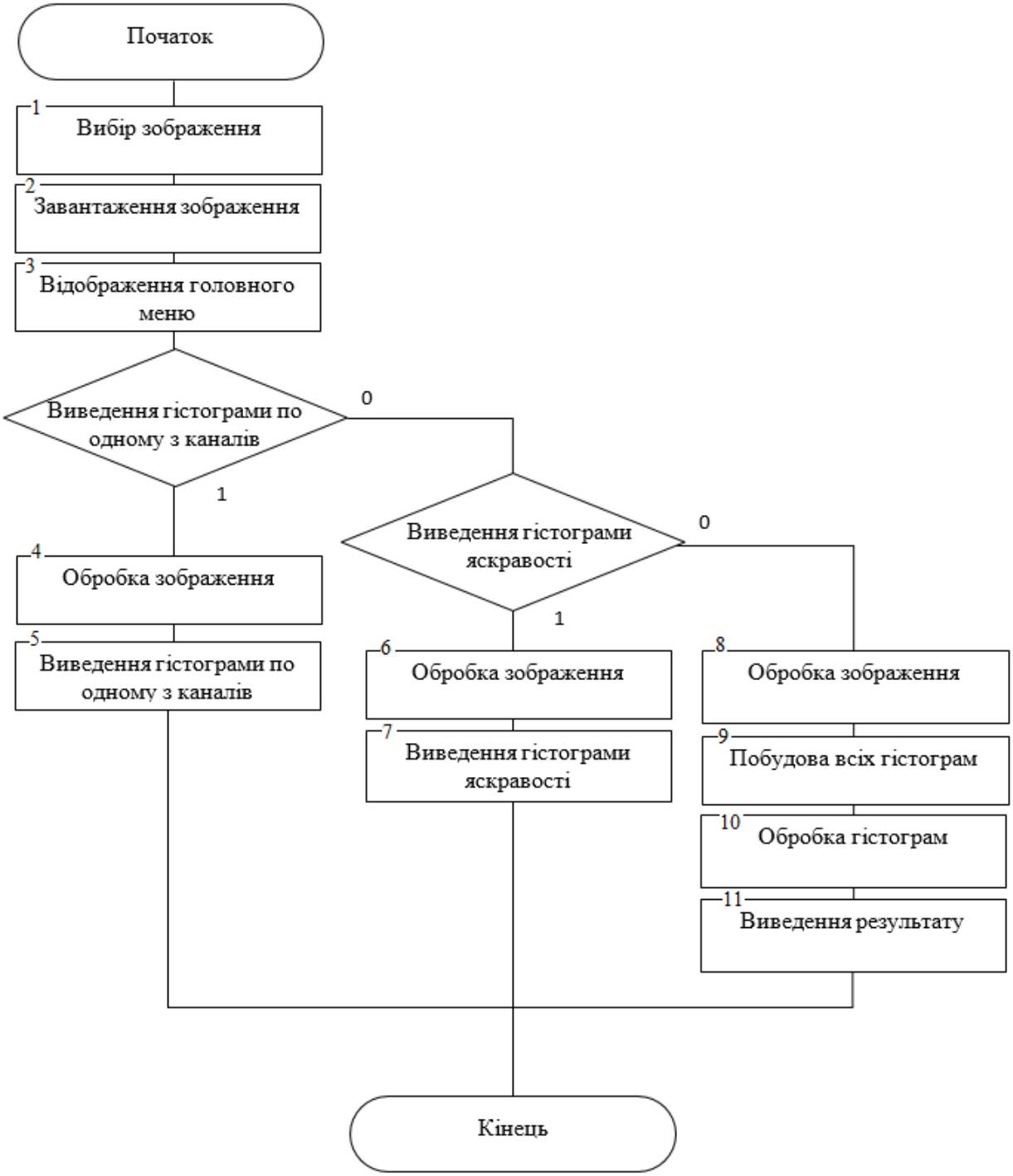


Рисунок В.3 – Схема загального алгоритму функціонування інформаційної технології

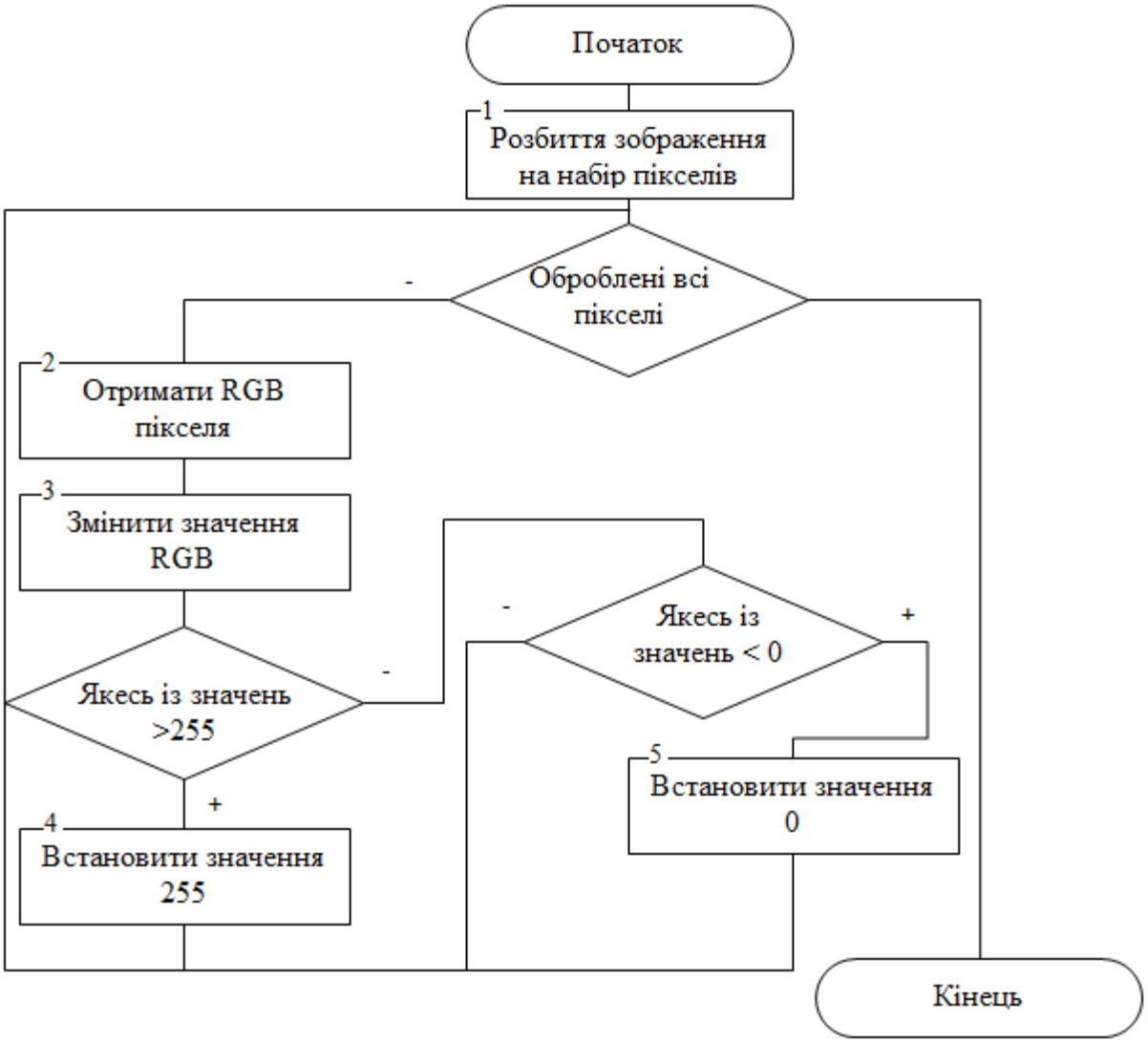


Рисунок В.4 – Схема алгоритму розбиття зображення на пікселі

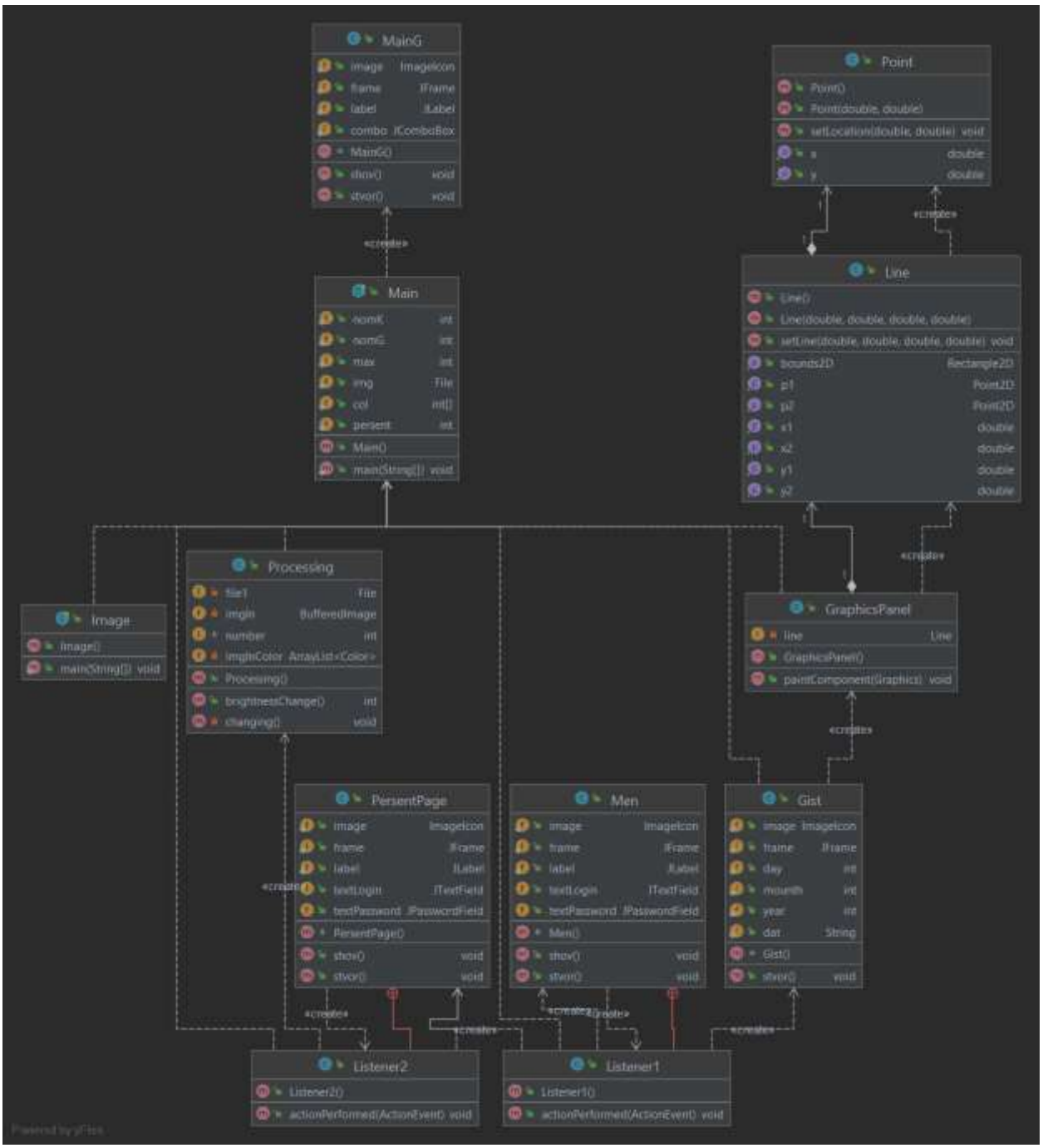


Рисунок В.5 – UML-діаграма класів

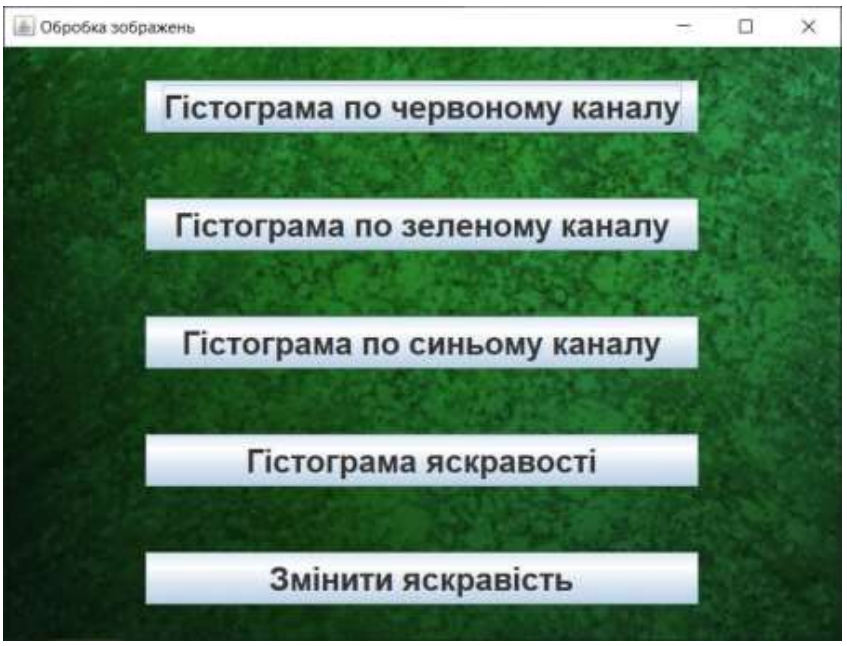


Рисунок В.6 – Загальний вигляд інтерейсного вікна програми

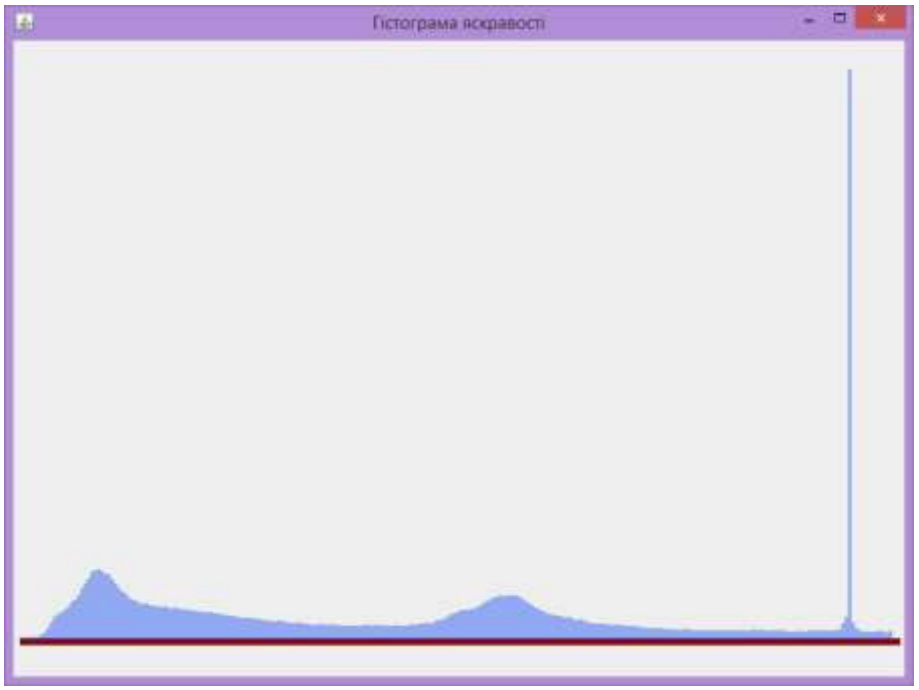


Рисунок В.7 – Загальний вигляд інтерейсного вікна для гістограми яскравості тестового зображення

Інструкція користувача

Після запуску програми відкривається вікно початкової активності (рис. Г.1). Аби почати роботу програми необхідно обрати зображення за допомогою кнопки «Обрати зображення», після чого відкриється вікно провідника (рис. Г.2), з допомогою якого буде завантажено зображення.

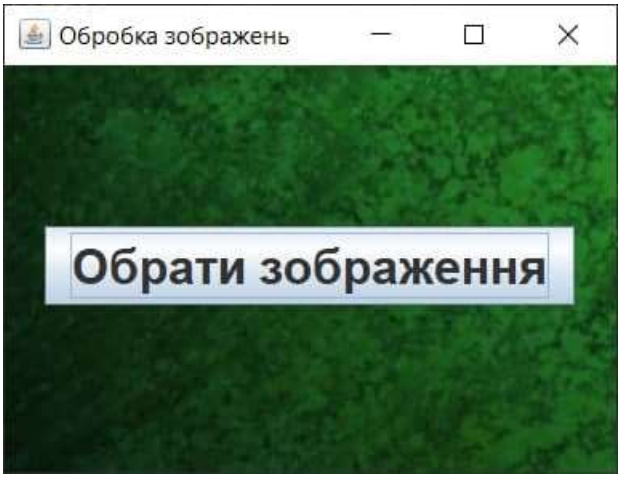


Рисунок Г.1 – Загальний вигляд інтерейсного вікна початкової активності

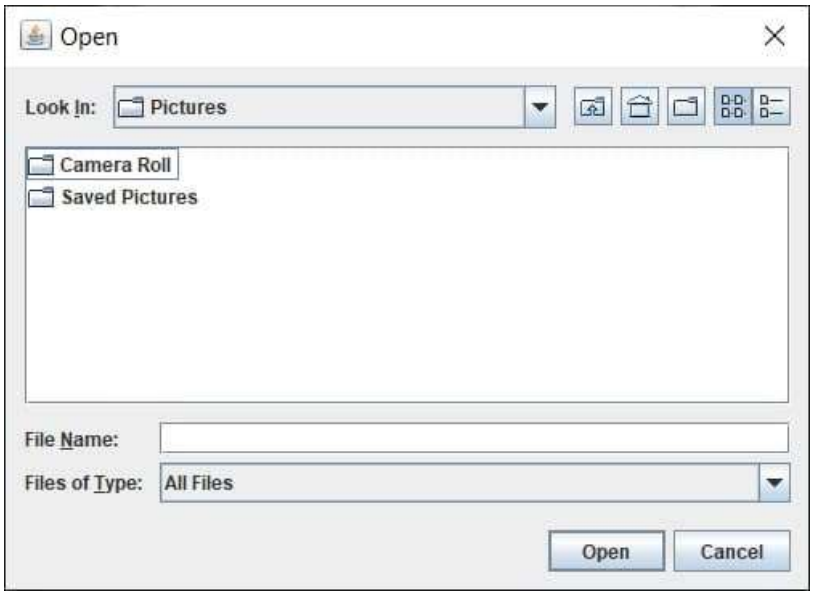


Рисунок Г.2 – Загальний вигляд інтерейсного вікна провідника

Після того як користувач відкрив зображення, відкривається меню обробки зображення, де необхідно обрати один з пунктів для подальшої обробки: «Гістограма по червоному каналу», «Гістограма по зеленому каналу», «Гістограма по синьому каналу», «Гістограма яскравості» та «Змінити яскравість» (рис Г.3). Для того, щоб змінити яскравість, необхідно ввести числове значення зміни яскравості в запропоноване поле (рис Г.4).

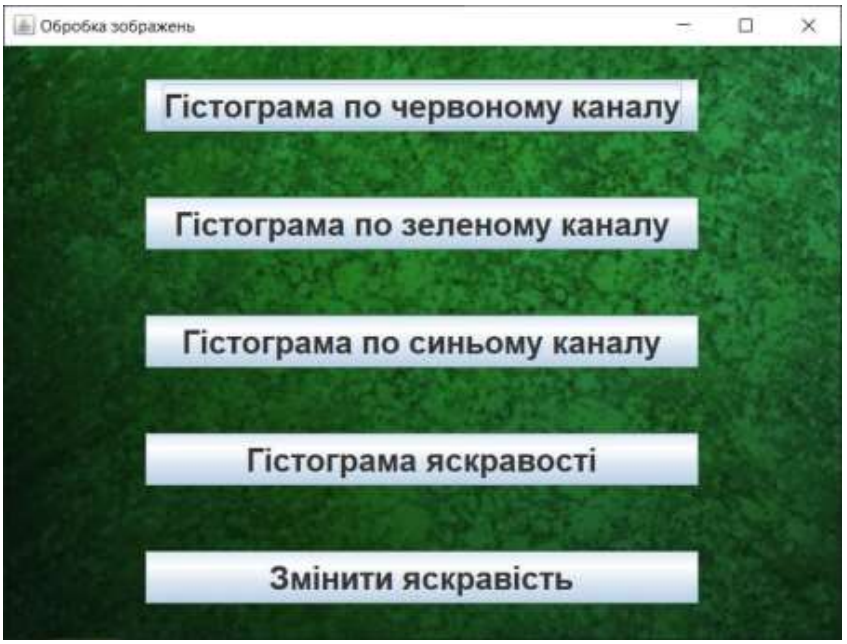


Рисунок Г.3 – Загальний вигляд інтерейсного вікна головного меню

Рисунок Г.4 – Загальний вигляд інтерейсного вікна введення значення яскравості