Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютерних технологій
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Розроблення системи автоматизації управління цифровими зображеннями у комп'ютерно-інтегрованих технологіях
(тема)

Виконала:
здобувачка 4 року навчання,
групи АКТСІ -21-2
Аліна КОНЄВА
(власне ім'я, прізвище)

Спеціальність 151 Автоматизація та комп`ютерно-інтегровані технології
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Системна інженерія
(повна назва освітньої програми)

Керівник Світлана СОТНИК
(посада, власне ім'я, прізвище)

Допускається до захисту
Завідувач кафедри КІТАР Ігор НЕВЛЮДОВ
(підпис) (власне ім'я, прізвище)

2025 р.

Я, Конєва Аліна Ігорівна, як здобувачка вищої освіти ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавала і не одержувала недозволену допомогу під час підготовки кваліфікаційної роботи. Я не використовувала штучний інтелект для підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на віповідне джерело.

«31» травня 2025 р  Конєва А. І.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАР
Рівень вищої освіти перший (бакалаврський)
Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
Тип програми Освітньо-професійна
Освітня програма Системна інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)

«19» травня 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Конєвій Аліні Ігорівні
(прізвище, ім'я, по батькові)

1 Тема роботи Розроблення системи автоматизації управління цифровими зображеннями у комп'ютерно-інтегрованих технологіях
Затверджена наказом по університету від 19.05.2025 р. №391 Ст
2 Термін подання студентом роботи до екзаменаційної комісії 31.05.2025
3 Вихідні дані до роботи
3.1 Середовище розробки PyCharm
3.2 Мова програмування Python
3.3 Використання бібліотеки OpenCV
3.4 Використання бібліотеки Pillow
3.5 Використання бібліотеки PyTorch
4 Перелік питань, що потрібно опрацювати в роботі
4.1 Вступ
4.2 Аналіз технічного завдання
4.3 Проєктування системи автоматизації управління зображеннями
4.4 Програмна реалізація системи автоматизації управління зображеннями
4.5 Тестування та перевірка системи автоматизації управління зображеннями
4.6 Визначення можливостей інтеграції
4.6 Висновки та перелік джерел посилань

5 Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій

Демонстраційний матеріал, представлений у форматі презентації PowerPoint (*.ppt) 15 с. формату А4

6 Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№		Термін виконання етапів роботи	Приміт–ка
1	Аналіз технічного завдання	17.11.2024	Виконано
2	Проєктування системи автоматизації управління зображенням	03.12.2024	Виконано
3	Програмна реалізація системи автоматизації управління зображеннями	07.03.2025	Виконано
4	Тестування та перевірка системи автоматизації управління зображеннями	20.03.2025	Виконано
5	Визначення можливостей інтеграції	16.04.2025	Виконано
6	Тестування та перевірка системи автоматизації управління зображеннями	13.05.2025	Виконано
7	Висновки та перелік джерел посилань.	20.05.2025	Виконано

Дата видачі завдання 15.11.2024

Студент _____ Конєва А.І.
(підпис) (прізвище, ініціали)

Керівник роботи _____ доц. Сотник С. В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка містить: 143 с., 51 рис., 3 дод., 17 джерел.

КОМП’ЮТЕРНО-ІНТЕГРОВАНІ ТЕХНОЛОГІЇ, ЗОБРАЖЕННЯ, СИСТЕМА АВТОМАТИЗАЦІЇ, ШТУЧНИЙ ІНТЕЛЕКТ, АВТОМАТИЗАЦІЯ ОБРОБКИ, OPENCV, PYTHON, УПРАВЛІННЯ ФАЙЛАМИ.

Мета роботи – підвищення ефективності процесів управління цифровими зображеннями за рахунок розробки системи автоматизації, яка поєднує зручність користування, швидку обробку файлів та можливість інтеграції у вже існуючі робочі процеси.

Об'єкт розробки – процеси управління цифровим зображеннями у комп'ютерно-інтегрованих технологіях.

Предмет розробки – цифрові зображення у комп'ютерно-інтегрованих технологіях.

В першому розділі кваліфікаційної роботи бакалаврського рівня наведено аналіз сучасного підходу до управління зображеннями у комп’ютерно-інтегрованих технологіях.

В другому розділі детально описано процес проєктування системи автоматизації управління зображеннями у середовищі PyCharm за допомогою мови Python.

В третьому розділі описано розроблення системи автоматизації управління зображенням. Розглянуто можливості інтеграції додаткових модулей штучного інтелекту, взаємодії з периферійними пристроями.

Отримані результати можуть бути використані як основа для розробки та вдосконалення програмних засобів у сфері автоматизації обробки зображень.

ABSTRACT

Explanatory note: 143 pp., 51 fig., 3 app., 17 sources.

COMPUTER-INTEGRATED TECHNOLOGIES, IMAGES, AUTOMATION SYSTEM, ARTIFICIAL INTELLIGENCE, AUTOMATED PROCESSING, OPENCV, PYTHON, FILE MANAGEMENT.

The purpose of the study is to enhance the efficiency of digital image management processes by developing an automated system that combines user-friendly operation, fast file processing, and seamless integration into existing workflows.

Object of development is the processes of managing digital images within computer-integrated technologies.

The subject of development is digital images in computer-integrated technologies.

The bachelor's thesis work presents an analysis of modern approaches to managing digital images in computer-integrated technologies, substantiates the choice of architecture and development tools, and develops an automated image management system with a high degree of flexibility. It also examines the possibilities of integrating additional artificial intelligence modules and interaction with peripheral devices. The process of designing and implementing the automated image management system in the PyCharm environment using the Python programming language is described in detail.

The obtained results can serve as a foundation for the development or improvement of software tools in the field of automated image processing, as well as for implementation in industrial, scientific, or research processes where speed and accuracy of working with visual data are critical.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БПЛА – безпілотний літальний апарат;

БД – база даних;

КІТ – комп’ютерно-інтегровані технології;

ПК – персональний комп’ютер;

ЧБ – чорно-білий режим;

ШІ – штучний інтелект;

AI – artificial intelligence;

BMP – bitmap;

GUI – graphical user interface;

IDE – integrated development environment;

JPG/JPEG – joint photographic experts group;

JSON – javascript object notation;

OpenCV – open source computer vision library;

PNG – portable network graphics;

RGB – red, green, blue;

SCADA – supervisory control and data acquisition.

ВСТУП

Сьогодні цифрові зображення є невід’ємною частиною діяльності більшості сфер. Щороку зростає обсяг візуальних даних, які створюються камерами спостереження, дронами, мобільними пристроями, промисловими сенсорами та іншими джерелами. Такі дані потребують швидкої обробки для отримання інформації та її використання. На сучасному етапі розвитку інформаційних технологій автоматизація процесів управління та обробки зображень стає надзвичайно важливою у комп’ютерно-інтегрованих технологіях.

Існуючі рішення в цій сфері часто передбачають ручну обробку зображень або використовують обмежений набір інструментів, що не відповідає вимогам користувачів і виробничих процесів. Це створює потребу у розробленні програмних систем, здатних поєднувати зручне управління зображеннями, автоматичну обробку великих масивів даних і можливість інтеграції у вже існуючі автоматизовані процеси.

У таких умовах особливої актуальності набувають засоби, що роблять цей процес автоматизованим та зручним у використанні. Розробка такої системи вимагає врахування багатьох аспектів: забезпечення стабільної роботи навіть у складних сценаріях, інтеграції інтелектуальних алгоритмів для автоматизації роботи з зображеннями, а також створення інтуїтивно зрозумілого інтерфейсу для користувача.

Актуальність теми зумовлена і практичним значенням для широкого кола користувачів, які щоденно працюють з великими обсягами даних і прагнуть отримати швидкі та точні результати навіть без глибоких знань у сфері цифрової обробки. Водночас необхідно, щоб система мала здатність адаптуватися до різних сценаріїв використання.

Це має особливе значення для таких напрямів, як системи комп’ютеризованого виробництва, комп’ютеризованого керування технологічними процесами та автоматизованих систем контролю якості. Такі технології потребують зручності та адаптації для роботи з цифровими зображеннями, що дозволяє автоматизувати робочі процеси та підвищувати їхню ефективність.

Мета роботи – підвищення ефективності процесів управління цифровими зображеннями за рахунок розробки системи автоматизації, яка поєднує зручність користування, швидку обробку файлів та можливість інтеграції у вже існуючі робочі процеси.

Об'єкт розробки – процеси управління цифровим зображеннями у комп'ютерно-інтегрованих технологіях.

Предмет розробки – цифрові зображення у комп'ютерно-інтегрованих технологіях.

Для досягнення мети було визначено наступні завдання:

– провести огляд сучасних рішень у сфері автоматизації управління зображеннями, виділити їхні переваги та недоліки;

– проаналізувати технології та інструменти, що використовуються для розробки програмного забезпечення галузі комп’ютерно-інтегрованих технологій;

– розробити архітектуру програмної системи з урахуванням модульного підходу та можливості інтеграції в комп’ютерно-інтегровані технології;

– реалізувати функціонал обробки цифрових зображень із можливістю застосування автоматичної класифікації зображення, фільтрів та аналізу якості;

– розробити інтелектуальні модулі для автоматичного формування рекомендацій користувачу;

– оформити кваліфікаційну роботу згідно ДСТУ 3008:2015 [1], а також з методичними вказівками з підготовки й оформлення кваліфікаційної роботи здобувачами першого (бакалаврського) рівня вищої освіти спеціальності 151

«Автоматизація та комп’ютерно-інтегровані технології» освітньої програми «Системна інженерія» [2] з дипломним проєктуванням для студентів усіх форм навчання [3].

Отримані результати роботи можна віднести до Цілі сталого розвитку 9 “Промисловість, інновації та інфраструктура”, а саме п. 9.4 “Сприяти прискореному розвитку високо- та середньо-високотехнологічних секторів переробної промисловості, які формуються на основі використання ланцюгів «освіта – наука – виробництво» та кластерного підходу за напрямами розвиток інноваційної екосистеми”, індикатор 9.4.1.

За результатами роботи було опубліковано тези доповіді у збірнику університету [4] та наукову статтю у науково-технічному журналі [5 – 7].

1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ

1.1 Особливості цифрового зображення та його характеристики

У цифровому середовищі сьогодення візуальна інформація – це одна з найбільш значущих форм представлення даних, особливо в контексті швидкого динамічного розвитку індустрії. Системи автоматизації, що працюють із візуальними даними, використовують зображення будь-якого типу як ключове джерело інформації для аналізу, контролю та прийняття рішень. Візуальні елементи забезпечують відомостями про стан, форму, структуру про об'єкти та процеси. Ця інформація може бути передана як і людині, так і системі.

Цифрове зображення – це результат оцифрування оптичного сигналу та являє собою двовимірну дискретну матрицю пікселів, де кожному елементу відповідає числове значення, що відповідає за яскравість або колір відповідної точки зображення. Саме така структура дозволяє ефективно зберігати, передавати та обробляти візуальні дані в автоматизованих програмних системах. А саме: здійснювати над ними різноманітні математичні та логічні операції для подальшого аналізу, покращення чи автоматичного розпізнавання.

Основною складовою є піксель – найменша структурна одиниця, яка відображає колірну або експозиційну інформацію. Сукупність пікселів формує повне зображення, причому їх кількість і розташування визначають геометричні розміри зображення та рівень деталізації.

Цифрове зображення має ряд характеристик, що напряму впливають на якість та ефективність його подальшої обробки. Однією з надзвичайно важливих характеристик є роздільна здатність, яка визначається кількістю пікселів по горизонталі та вертикалі. Вона прямо впливає на деталізацію та чіткість картинки, що дуже важливо для будь-якої сфери автоматизації для сьогодення. Чим вища роздільна здатність, тим більше інформації може бути передано через

візуальний канал. Звісно, тим точніше відображаються дрібні деталі об'єкта чи сцени.

Не менш важливою характеристикою є глибина кольору. Вона описує кількість бітів, що виділяються для представлення кольору кожного пікселя. Наприклад, при глибині 8 біт можливо закодувати 256 відтінків сірого. Тим часом 24-бітне зображення (по 8 бітів на кожен з трьох кольорових каналів RGB) дозволяє відображати понад 16 мільйонів відтінків. Саме це є стандартом для більшості кольорових зображень у сучасних цифрових системах. Вибір глибини кольору визначається вимогами до точності відображення кольорової палітри та обсягом даних, що генеруються (рис. 1.1).

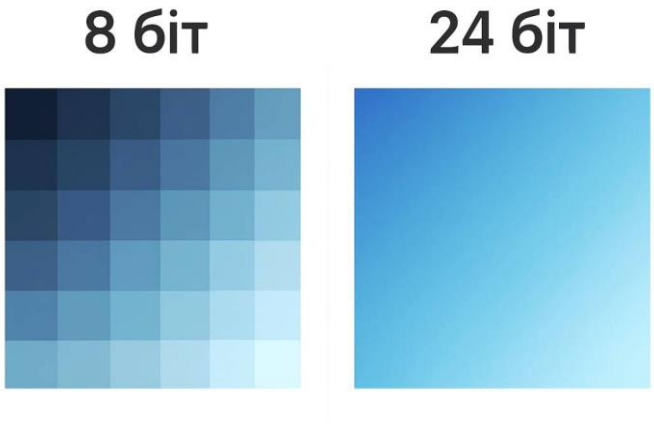

Рисунок 1.1 – Візуальне порівняння глибини кольору

Зображення можуть бути сформовані різними технічними засобами: цифровими камерами, сканерами, промисловими сенсорами, системами відеоспостереження, а також спеціалізованими медичними або науковими пристроями. Отримані візуальні дані зберігаються у різних файлових форматах, кожен з яких має свої переваги й недоліки залежно від завдань, що вирішуються.

Один із основних параметрів зображення є формат файлу. Він визначає спосіб кодування і зберігання зображення. Серед найбільш поширених форматів варто відокремити кілька основних. Забезпечення збереження зображення без втрати якості – це PNG, що також підтримує прозорість фону та розповсюджений для веб-ресурсів. BMP – це несжатий простий растровий формат, який

традиційно використовується у середовищах Windows та зберігає максимальну якість. Ці два формати часто використовуються для збереження креслень чи схем. Формат JPEG, зазвичай, характеризується меншою якістю, але дозволяє зменшити розмір файлу, що дуже важливо для передавання великої кількості зображень. Розширений професійний формат для збереження високоякісних файлів – це TIFF, який використовують там, де важлива максимальна точність передачі деталей та кольорів, що особливо корисно у поліграфії та наукових дослідженнях.

У реальних технічних системах зображення, зазвичай, зберігаються у форматах, що не тільки забезпечують компроміс між обсягом даних і якістю, а й також можуть містити метадані, що можуть описувати умови зйомки або параметри управління та обробки, тому правильне обрання формату зберігання надзвичайно важливе.

Крім того, при обробці зображень важливо враховувати такий параметр, як колір. Візуальне сприйняття кольору в цифровому середовищі вимагає точного визначення та обробки кольорів на рівні комп'ютерних систем, що забезпечується за допомогою відповідних математичних моделей. Одне з очевидних важливих аспектів кольору – це покращення сприйняття зображень людиною та психологічний вплив, але цей елемент напряму впливає на обробку та управління зображеннями.

Для точного представлення кольору в комп'ютерних системах використовуються спеціальні математичні моделі, що описують, як кольори кодуються та відображаються на екранах і в цифрових зображеннях. Ці моделі є основою для багатьох процесів обробки зображень, починаючи від їхнього створення і закінчуючи складними операціями, такими як корекція балансу білого, фільтрація кольорів, класифікація та сегментація зображень. Залежно від типу задачі обирається відповідна модель кольору.

Однією з найпоширеніших колірних моделей є RGB (Red, Green, Blue). Це адитивна модель кольору, що базується на трьох основних кольорах: червоному,

зеленому та синьому. Вона широко використовується в комп'ютерних моніторах, телевізорах та інших пристроях відображення. Обробка кольорів, зокрема інверсія, зміна контрасту чи переведення у монохромний режим, реалізована через маніпуляції саме з цими компонентами. У RGB-моделі кожен піксель зображення визначається трійкою значень, які вказують на інтенсивність кожного з кольорів (рис. 1.2).

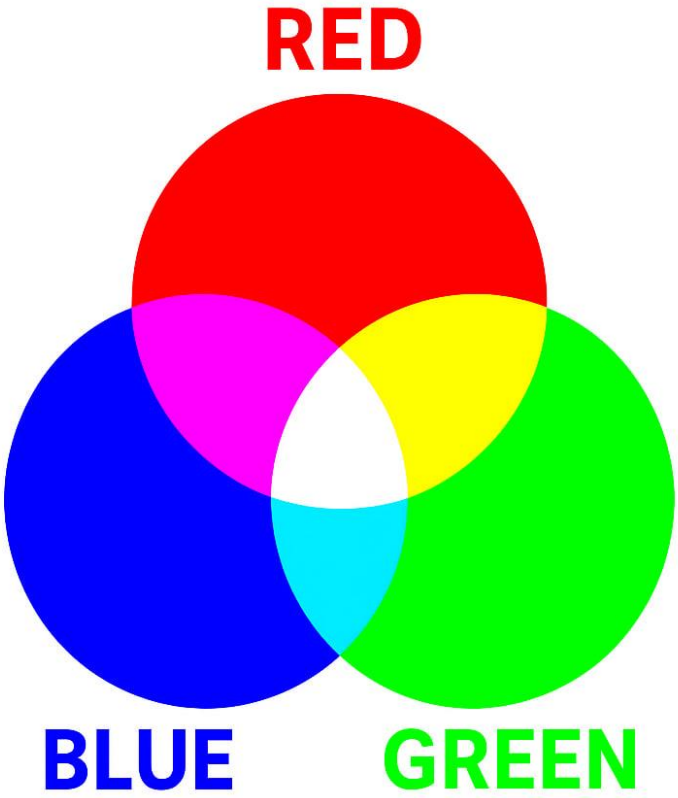

Рисунок 1.2 – Візуальна колірна модель RGB

CMYK (Cyan, Magenta, Yellow, Black) – субтрактивна колірна модель, що базується на принципі віднімання (поглинання) певних довжин хвиль світла з видимого спектру. Ця модель є основною в друкарській промисловості, де кольори утворюються через накладання різних відтінків чорнила на білий фон (рис. 1.3).

Рисунок 1.3 – Візуальна колірна модель CMYK

HSB/HSV (Hue, Saturation, Brightness/Value) – модель кольору, яка описує кольори через відтінок, насиченість і яскравість. Вона є інтуїтивно зрозумілою і широко використовується в графічних редакторах для маніпуляцій із кольорами. Це дозволяє зручніше налаштовувати кольори, зберігаючи природність сприйняття (рис. 1.4).

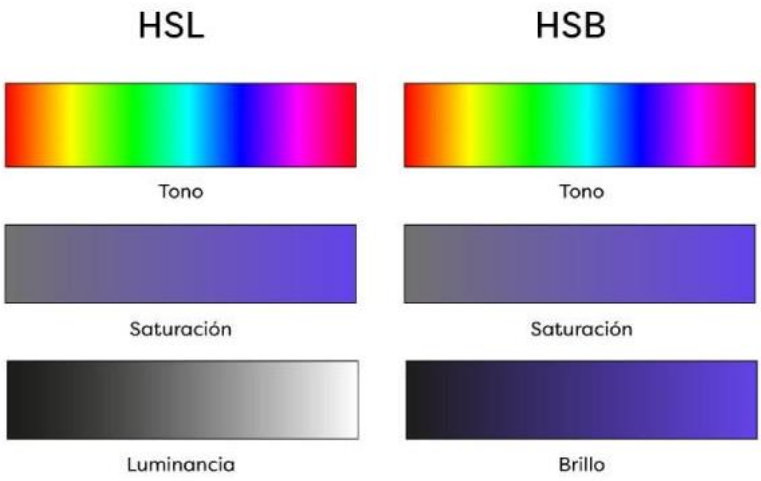

Рисунок 1.4 – Візуальна колірна модель HSB/HSV

Lab – модель кольору, яка була розроблена для точного відтворення кольорів незалежно від пристроїв і їх кольорових профілів. Вона використовує три компоненти: світлотінь (L), червоний-зелений (a) та синій-жовтий (b), що дозволяє точно визначати кольори в просторі, який відповідає людському сприйняттю кольорів (рис. 1.5).

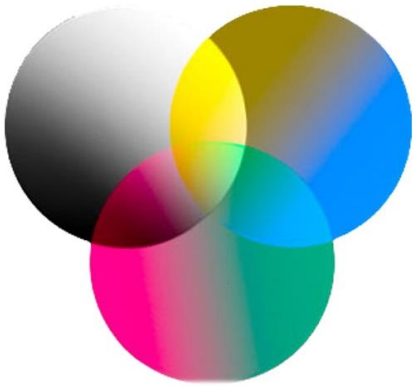

Рисунок 1.5 – Візуальна колірна модель Lab

Для точного представлення кольору важливо забезпечити коректне кодування кольорів, що залежить від вибору відповідної моделі. Це дозволяє зберігати і відображати зображення на різних пристроях, таких як монітори, принтери або проектори. Тобто це є гарантом забезпечення точності та відтворення кольорів без спотворень.

Оскільки різні пристрої та умови відображення вимагають гнучкості у роботі з кольором, важливою частиною управління візуальної інформації є саме обробка зображення. А одним із основних етапів обробки цифрових зображень є корекція кольору. Вона включає зміну яскравості, контрасту чи насиченості кольорів, щоб досягти бажаного візуального ефекту або підготувати зображення до подальших етапів обробки.

Часто колір використовується при сегментації зображень, де кольорові характеристики допомагають виділяти різні об'єкти на фотографії. Важливим етапом є також фільтрація та видалення шуму, коли фільтри очищують зображення або змінюють його кольорову гаму для покращення якості. Окремим

аспектом є класифікація об'єктів за їхніми кольоровими характеристиками. Це має велике значення в таких застосуваннях, як комп'ютерний зір, де розпізнавання кольору допомагає точніше ідентифікувати об'єкти або відслідковувати їх в реальному часі.

Крім кольорових характеристик, під час використання зображень в автоматизованих системах управління, особливо в задачах технічного зору, велике значення мають структурні характеристики: різкість меж, контрастність, рівень шуму, ступінь освітлення, чіткість контурів тощо.

Таким чином, цифрові зображення не лише є засобом візуалізації, але й важливим інструментом керування та аналізу. Розуміння структури зображень, їхніх параметрів та характеристик є необхідним етапом при розробці ефективних систем автоматизації обробки візуальних даних. Тобто, в інженерних програмах це не лише візуальний об’єкт, а структурований набір даних, який може бути автоматично проаналізований, трансформований або інтерпретований системою. Такий підхід є фундаментальним для побудови ефективних алгоритмів машинного зору.

У сфері комп’ютерно-інтегрованих технологій це ключовий форматом для передання даних для автоматизованого візуального контролю. Така технологія включає в себе контроль якості продукції, виявлення дефектів, ідентифікацію об'єктів та навігацію автономних систем, а вже використання алгоритмів машинного зору в автоматизованих системах дозволяє приймати рішення в реальному часі без участі людини. Тим самим графічні елементи напряму впливають на автоматизацію процесу.

1.2 Аналіз сфер використання візуальної інформації

Цифрові зображення дозволяють отримати важливі відомості про стан об’єктів, їх геометрію, поверхню, поведінку. Проте, традиційні підходи, у вигляді ручного управління графічною інформацією, що базуються на ручній

обробці, потребують втручання людини. Це є вкрай ресурсоємним, вимагає часу та експертної участі, що вже може ускладнювати оперативне реагування в режимі реального часу.

Саме тому автоматизація обробки цифрових зображень стає сьогодні важливою як інструмент підвищення ефективності та надійності. Системи автоматизованого управління зображеннями є складовою цифрової трансформації візуальних даних.

У контексті розвитку концепції «Індустрія 4.0» актуальності набувають технологічні платформи та продукти, що базуються на взаємодії між собою одного складного алгоритму. Сучасні підходи до автоматизації управління зображеннями базуються на тісній взаємодії технічних засобів фіксації зображень, програмних платформ обробки, алгоритмів аналізу та структурованого зберігання даних. Завдяки поєднанню цих елементів створюються системи, які не лише здатні замінити ручний контроль та автоматизувати рутинні операції, але й перевершують його за точністю і швидкістю. Це впливає на забезпечення стабільної якості процесів, тобто навантаження персоналу та надійність [6].

У промислових і технічних системах основною сферою застосування технологій обробки зображень є контроль якості. Такі системи відкривають можливість автоматично виявляти дефекти та деформації, неправильне компонування або інші відхилення від технічного стандарту. Завдяки безперервному візуальному контролю, інтегрованому безпосередньо у виробничі лінії, підприємства отримують можливість не лише уникати браку, але й значно скорочувати витрати на повторну перевірку чи ручний огляд продукції. Крім того, вони використовуються для калібрування механізмів через аналіз, моніторинг стану обладнання з метою виявлення зносу чи перегріву, а також для оцифровування технічної документації, креслень або маркувань.

Суттєву роль в цьому напрямі відіграють саме системи машинного зору. Вони дають змогу розпізнавати, інтерпретувати, аналізувати зображення,

зафіксовані високоточними камерами спостереження. Такі технології широко впроваджуються не тільки в промисловості, а й медицині та інших сферах, де критично важливим є точне виявлення дефектів, структури об'єктів та перевірка їх відповідності стандартам та заданим критеріям.

У медичній галузі цифрові зображення стали основою багатьох діагностичних рішень. Аналіз даних, отриманих за допомогою рентгенографії, комп’ютерної томографії, ультразвукової діагностики або МРТ, дедалі частіше здійснюється із залученням алгоритмів, що здатні виявляти патології на ранніх етапах, порівнювати динаміку змін, зберігати знімки в електронних картках пацієнтів та автоматично формувати медичні звіти. Застосування цих систем актуальне не лише в діагностиці, а й у телемедицині, де віддалений аналіз зображень забезпечує первинний скринінг пацієнтів без потреби фізичної присутності. У дерматології та стоматології, ці технології допомагають точно локалізувати ураження шкіри, дефекти емалі або інші порушення тканин [8].

У сфері цифрового маркетингу та електронної комерції обробка зображень дозволяє автоматизувати створення високоякісного контенту, забезпечити персоналізацію відображення товарів, проводити глибоку сегментацію аудиторії на основі візуальних переваг, а також адаптувати вигляд продукції відповідно до запитів користувача. Завдяки алгоритмам розпізнавання об’єктів, система здатна автоматично визначати товарні категорії на фотографіях, покращувати зображення для рекламних кампаній або проводити мінімальні тестування на основі емоційних реакцій користувачів. Такі підходи значно підвищують ефективність взаємодії з клієнтом і дозволяють компаніям створювати гнучкі та динамічні стратегії візуальної комунікації.

Не менш важливою є роль таких систем в інших прикладних сферах. В освітніх та архівних проєктах обробка сканованих документів дозволяє автоматизувати оцифрування старих текстів, карт, креслень, забезпечуючи збереження культурної спадщини в цифровому форматі. У сільському господарстві, наприклад, цифрові зображення, отримані з дронів або супутників,

використовуються для аналізу стану посівів, виявлення хвороб або зон, які потребують обробки.

Сучасні технології автоматизації управління зображеннями знайшли широке застосування в таких важливих сферах, як військова та правоохоронна сфера, де критичне значення має швидкість і точність аналізу візуальної інформації.

У сфері безпеки, а саме для правоохоронних органів, автоматизація обробки зображень має критичне значення задля забезпечення громадської безпеки та боротьби з кримінальністю. Камери відеоспостереження в поєднанні з алгоритмами розпізнавання поведінкових патернів дають змогу правоохоронцям своєчасно виявляти підозрілі дії у місцях великого скупчення людей, що сприяє запобіганню інцидентам та покращує якість розслідувань.

В Україні використовування камер спостереження в публічних місцях дозволяє використовувати алгоритми для автоматичного виявлення підозрілих осіб та аналізу поведінки людей у реальному часі. Це дає змогу правоохоронним органам швидше реагувати на потенційно небезпечні ситуації, знижуючи ризики для громадян. Крім того, ці технології сприяють розслідуванню кримінальних справ і виявленню доказів, що є важливим для забезпечення справедливості.

У військових цілях системи автоматизованого аналізу зображень активно використовуються для розпізнавання об'єктів: техніка, пересувань противника та інших загроз, інфраструктура, підозрілі об'єкти, що можуть становити загрозу. Наприклад, завдяки використанню супутникових зображень та алгоритмів обробки зображень можна оперативно виявляти переміщення ворожих сил, зміни на стратегічних об'єктах та, навіть, передбачати майбутні дії супротивника. В Україні ці технології стали незамінними для аналізу зображень, отриманих від безпілотних літальних апаратів (БПЛА), зо активно використовуються для моніторингу територій та оцінки ситуацій, зокрема на бойовій лінії зіткнення [9].

Особливо важливими є інтелектуальні алгоритми, що здатні адаптуватися до змінюваних умов, таких як зміни освітлення або маскувальні техніки, що

використовуються противником. Це дозволяє значно підвищити ефективність застосування автоматизованих систем навіть у складних умовах.

В Україні, особливо в умовах війни, використання зазначених технологій також значно підвищує ефективність оборонних заходів. Наприклад, системи розпізнавання об'єктів на основі зображень дозволяють оперативно виявляти важливі цілі на великій відстані, забезпечуючи своєчасне реагування військових. Застосування адаптивних алгоритмів, які здатні пристосовуватись до змін зовнішніх умов, маскування та погане освітлення зокрема, робить ці системи надзвичайно критично важливими, навіть, у складних бойових умовах.

Автоматизація управління зображеннями також використовує технології прогнозування для запобігання катастрофам. У надзвичайних ситуаціях, пов’язаних із природними катастрофами або техногенними інцидентами, такі системи допомагають оперативно оцінити масштаби загрози та надати актуальну інформацію для швидкого реагування служб порятунку та органів влади.

У виробничому середовищі машинний зір реалізується за допомогою застосування камер високої роздільної здатності, що мають змогу фіксувати зображення, та спеціалізованого програмного забезпечення, яке аналізує отримані дані за допомогою спеціально розроблених алгоритмів та дій. Ці рішення здатні автоматично виявляти похибки, контролювати параметри форм та розмірів, перевіряти маркування, а також стежити за перебігом процесів на виробничих лініях. Такі системи здатні автоматично виявляти дефекти продукції, контролювати геометричні параметри об’єктів, перевіряти правильність складання, здійснювати оптичне розпізнавання маркувань, штрихкодів, QR-кодів, а також відслідковувати рух і стан елементів виробничої лінії.

Із розвитком технологій, удосконаленням та постійною роботою над алгоритмами обробки та управління зображень, було незупинне інтегрування методів штучного інтелекту в системи машинного зору. Як приклад, нейронні мережі дають змогу розпізнавати складні образи та патерни, виявляти нові типи

дефектів та адаптувати системи до змін умов спостереження, середовища та продукції зокрема. Впровадження штучного інтелекту виявляється особливо ефективним в умовах, коли структура зображення є складною, непередбачуваною та має багато варіацій. Тобто там, де традиційні умови спостереження не мають змоги працювати ефективно.

В умовах промислового виробництва системи автоматизованого візуального контролю здатні використовувати камери задля фіксування у режимі реального часу. Алгоритми на основі згорткових нейронних мереж дозволяють здійснювати швидку та точну перевірку продукції, виявляючи навіть найменші дефекти.

Сьогодні сучасні автоматизовані системи управління зображеннями здатні до взаємодії з іншими елементами автоматизованих комплексів, а саме: з програмованими логічними контролерами, базами даних, Supervisory Control and Data Acquisition (SCADA) системами та інтерфейсами людина-машина. У цьому випадку цифрові зображення набувають значення вхідного сигнали, на основі якого формується автоматизоване рішення управління. Наприклад, при виявленні відхилень від стандарту візуального контролю, система може здійснити автоматичну зупинку обладнання, змінити параметри технологічного процесу або подати сигнал помилки оператору.

1.3 Аналіз аналогічних рішень

У межах розробки структури та алгоритму системи автоматизації для управління зображенням є доцільним проаналізувати вже існуючі програмні засоби, що використовують подібні функції. Це дозволяє визначити сильні та слабкі сторони вже готових рішень та визначити актуальні архітектурні рішення задля програмної реалізації системи.

Adobe Photoshop – один з популярних та актуальних редакторів зображень. Пропонує широкий арсенал інструментів для обробки, включаючи підтримку

розширених фільтрів, корекції кольору, шарів і масок. Хоча в останніх версіях з’явилися інструменти на базі штучного інтелекту, програма все ще переважно орієнтована на ручну обробку, що потребує професійних навичок і значного часу. Також тут лише частково реалізований аналіз на базі штучного інтелекту та працює задля художніх рішень, що не є підходящим для роботи з комп’ютерно-інтегрованими системами.

Luminar Neo (Skylum) – спеціалізоване рішення, яке реалізує автоматизовану обробку зображень із використанням технологій штучного інтелекту. Система орієнтована на кінцевих користувачів і пропонує функції автоматичного покращення неба, портретів, деталей пейзажів тощо. Проте її закритий характер унеможливлює інтеграцію у зовнішні системи або розширення функціоналу.

Topaz Photo AI (Topaz labs) – професійний інструмент, що використовує нейромережі для підвищення чіткості, видалення шумів та покращення роздільної здатності. Працює на базі моделей Real-Enhanced Super-Resolution Generative Adversarial Network (Real-ESRGAN), Denoise AI, Sharpen AI. Забезпечує високоякісні результати, однак, як і попередні системи, має закритий вихідний код і не дозволяє гнучкої адаптації до специфіки користувача.

GIMP (GNU image manipulation program) – безкоштовний редактор з широкими можливостями налаштування, підтримкою плагінів та скриптів. Дає змогу виконувати більшість базових операцій, однак не містить вбудованих засобів інтелектуального аналізу або рекомендацій на основі штучного інтелекту.

ImageJ – орієнтований на наукову та медичну візуалізацію. Забезпечує точні вимірювання, роботу з гістограмами та статистикою, однак не призначений для автоматичного покращення чи інтерпретації зображень.

Аналіз наявних програмних засобів свідчить про те, що існуючі комерційні системи часто є закритими та складними для інтеграції в спеціалізовані рішення.

З іншого боку, інструменти з відкритим кодом – гнучкі, але не мають вбудовані механізми штучного інтелекту та автоматизації.

На відміну від розглянутих аналогів, алгоритм нової системи автоматизації управління зображеннями, повинен поєднувати автоматизовані рішення, інтелектуальну обробку та адаптивність під різні рішення, зберігаючи при цьому відкриту архітектуру та нескладність впровадження. Завдяки цьому така система є перспективною та конкурентоспроможнім рішенням у галузі комп'ютерно-інтегрованих технологій.

1.4 Аналіз вимог та постановка задачі

У контексті зростання обсягів візуальних даних та потреби в їх ефективній обробці, особливого значення набуває створення програмних рішень, здатних забезпечити автоматизовану, швидку та якісну обробку цифрових зображень. Сучасні системи в цій сфері мають відповідати критеріям високої продуктивності, гнучкості налаштувань, інтуїтивного інтерфейсу та адаптивності до змісту зображень.

Надзвичайно актуальним буде створення системи для автоматизованого управління цифрових зображень, що поєднує традиційні методи фільтрації з інтелектуальним аналізом на основі характеристик і типу зображення. Реалізована система повинна забезпечувати інтеграцію класичних фільтрів з алгоритмами оцінки якості, класифікації сцен і генерації рекомендацій, залишаючись при цьому простою у використанні та придатною до адаптації під конкретні задачі та сценарії користувача. У межах реалізації потрібно передбачити інтеграцію класичних методів корекції з модулями інтелектуального аналізу, які визначають тип зображення, оцінюють його якість за низкою метрик і пропонують оптимальні дії щодо покращення.

Незважаючи на широке застосування засобів обробки через те, що у сучасних комп'ютерно-інтегрованих системах зображення дедалі частіше

виступають джерелом інформації для прийняття рішень, багато існуючих інструментів орієнтовані на досвідчених користувачів або потребують тривалого навчання. Це значно ускладнює їхнє впровадження у загальні виробничі або адміністративні процеси, де швидкість, простота та інтуїтивність мають ключове значення.

Саме тому основним завданням розробки є створення програмної системи, яка не лише автоматизує базові та інтелектуальні етапи обробки зображень, але й залишатиметься доступною та зрозумілою навіть для користувачів без спеціальної технічної підготовки.

Таким чином, постановка задачі передбачає розробку універсального, масштабованого програмного продукту, що поєднує засоби класичної обробки зображень, сучасні підходи до їх інтелектуального аналізу, рекомендаційного супроводу та управління візуальною інформацією у автоматизованих системах.

1.5 Огляд методів обробки зображення

Обробка цифрових зображень є невід’ємною частиною сучасних автоматизованих систем, що працюють з візуальними даними. Цей процес охоплює трансформацію, аналіз та інтерпретацію зображень з метою виділення потрібної візуальної інформації, покращення якості зображення або прийняття рішень у режимі реального часі. Завдяки застосуванню різноманітних алгоритмів, обробка зображення дозволяє покращити якість, чіткість деталей, виділити необхідні елементи, здійснити фільтрацію та оформити ключові особливості, трансформувати та аналізувати зображені об’єкти. Це надзвичайно важливо для коректної роботи систем технічного зору.

Цілі, з якими проводиться обробка зображень, можуть суттєво відрізнятися в залежності від конкретної поставленої задачі. З одного боку, у художніх чи дизайнерських застосуваннях мета може полягати у створенні певного візуального ефекту, тоді як у технічних або наукових – у підвищенні точності та

якості зображення. Візуальні покращення можуть включати корекцію контрасту, яскравості чи насиченості кольорів, тоді як технічна обробка спрямована на усунення шумів, артефактів або викривлень, спричинених оптичними чи передавальними спотвореннями. Вимірювання на зображеннях також впливає на цей процес. Це є актуальним для наукових досліджень та технічних аналізах.

У контексті комп’ютерно-інтегрованих технологій обробка зображень є основою для багатьох етапів роботи. Методи аналізу зображень формують базу для виявлення дефектів, автоматичної класифікації об'єктів, оцінки якості та структурування даних для подальшої обробки. Методи обробки зображень активно застосовуються в системах контролю якості, автоматичного виявлення дефектів, навігації автономних роботів, медичній діагностиці, а також в інтелектуальному відеоспостереженні. Від цього залежить точність і швидкість аналізу зображень, що часто визначає ефективність всього процесу. Наприклад, в автоматичних системах контролю якості використовується сегментація для виявлення дефектів на виробничих лініях, а в медицині для аналізу рентгенівських зображень для виявлення патологій.

Одним з ключових напрямів в усіх сферах застосування обробки вважається автоматизоване розпізнавання об’єктів. Це використовується буквально всюди: ідентифікація символів, відбитки пальців, розпізнавання обличчя, використання в системах безпеки, біометрії, автоматизованих контролерах та інших складових у складних системах аналізу.

Одним із основних напрямів застосування є покращення якості зображень, яке включає в себе такі етапи, як корекція контрасту, яскравості та чіткості, а також усунення небажаних спотворень, зокрема шумів або розмиття. Наприклад, для зменшення шуму можна використовувати медіанні фільтри, які дозволяють усунути випадкові спотворення без значної втрати деталей зображення. Збільшення контрасту зображення дозволяє краще виділити важливі елементи, що значно полегшує подальший аналіз.

Управління зображенням включає в себе значну кількість методів. Умовно їх можна поділити на просту процедуру базових низькорівневих та складний алгоритм високорівневих методів. Перші працюють із піксельними характеристиками, а другі спрямовані на інтерпретацію та структурування сцени. Корекція базових характеристик є первинною обробкою зображень, що починається з регулювання базових параметрів, які суттєво впливають на візуальну якість та подальшу роботу.

Яскравість являє собою показник інтенсивності світла в межах зображення. Оптимальний рівень цього параметру забезпечую належну видимість об’єктів у всіх ділянках сцени: і світлих, і темних. При надто низькій яскравості зображення стає затемненим та втрачає інформативність у тінях. При надмірному ж освітленні можна бачити пересвіти та засвічення, що призводить до втрати деталізації. Технічно регулювання яскравості реалізується шляхом зміщення інтенсивності пікселів у бік потрібного напрямку.

Контраст зображення має прямий вплив на візуальну інформацію. Він визначає різницю між темними та світлими ділянками зображення та є одним з важливих моментів покращення візуальної виразності об’єктів. Високий контраст покращує візуальне сприйняття структури об’єктів, однак у деяких випадках призводить до втрати нюансів у крайових зонах яскравості. Низький контраст, навпаки, робить зображення менш об’ємним і важким. Для регулювання контрасту використовуються як глобальні перетворення, так і адаптивні алгоритми. Часто це лінійні або нелінійні розтягування гістограми яскравості пікселів. Підвищення контрастності зображення може впливати на розпізнавання маркувань, покращення якості та забезпечення точного позиціонування.

Наявність дрібних деталей та вираженість контурів дуже часто пов’язано з регулюванням різкості. Для оцінки цієї характеристики часто використовують чутливі до змін оператори, що вимірюють інтенсивність змін яскравості. Широко застосовується метод дисперсії Лапласіана. Це алгоритм, який здійснює аналіз

другої похідної функції яскравості та виявляє зони з високими градієнтами. Високе значення цієї метрики свідчить про наявність чітких структур та меж, тоді як низьке значення вказує на розмиття або відсутність фокусу. Збільшення різкості досягається шляхом фільтрів посилення контурів або алгоритмів маскування нечіткості, що комбінують оригінал з посиленими межами.

Виявлення контурів та обробка градієнтів важлива для визначення структури. Градієнтний аналіз дозволяє знаходити межі на основі різких змін інтенсивності. Оператори Собеля, Преввіта, Робертса, Кенні використовуються від типу зображення. Метод виявлення контуру об’єкту фундаментальний для перевірки точності монтажу, аналізу форми, контроля геометрії деталей та орієнтації об’єктів у системах комп’ютерного зору [10].

Для виділення області інтересу, зображення переводять у бінарний формат, що можна назвати також двоколірним. Це називається порогова обробка або бінаризація. Такий метод дозволяє істотно спростити подальші алгоритми аналізу. Особливо корисна для автоматичної сегментації об’єктів на конвеєрі, контролю наявності деталі на об’єкті та попередньої обробки.

У процесі обробки можуть застосовуватись геометричні трансформації – масштабування, обертання, вирівнювання перспективи, що дозволяє коригувати розміщення чи розміри об'єктів, забезпечуючи максимально точні результати аналізу. Трансформації дозволяють зберігати важливу інформацію під час зміни розмірів або зміщення об'єктів на зображенні, що робить процес більш гнучким [11].

Не виняток, що початкове зображення часто містить шум. Це може ставатися через погане освітлення, погані налаштування камери або візуальної системи, електронні перешкоди, цифрові стискання, передавання через нестабільні канали. Шумом можна назвати випадкові варіації яскравості, що виникають через нестабільні умови та часто спотворюють дані. Вони можуть бути не тільки розподілені рівномірно по зображенню, а й у вигляді імпульсних поодиноких темних або світлих пікселів. Для оцінки рівня шуму доцільним є

застосування спектрального аналізу. За допомогою швидкого перетворення Фур’є зображення переводиться з простору пікселів у частотний домен, де стає можливим виявлення регулярних або випадкових високочастотних компонентів, характерних для шумів. У частотному представленні центральна область спектра відповідає низьким просторовим частотам, які пов’язані з великими структурами (фон, тло), тоді як периферійні області відповідають високим частотам – контурам, текстурам або шумам. Аналіз проводиться шляхом фільтрації та обчислення потужності високочастотних компонентів зони, розташованої за межами центрального кола спектра. Цей метод дає змогу оцінити рівень зашумленості зображення незалежно від візуального аналізу чи наявності еталонного зразка. Він широко застосовується в системах автоматичного контролю якості, де необхідно швидко й об’єктивно ідентифікувати вплив спотворень на результат подальшої обробки.

Сьогодні активно використовуються різні методи очищення зображення від шуму, в тому числі і через штучний інтелект. Зазвичай, використовують медіанні фільтри, гаусове розмиття та середньоарифметичні. Зниження цього показника може впливати на забезпечення стабільності алгоритмів аналізу.

Оскільки цифрові зображення займають значні обсяги пам’яті, що може сприйняти проблеми з архівами, важливим завданням є стиснення. Компресія цифрових зображень важлива в умовах обмеженого місця для зберігання та передачі даних. Це дозволяє зменшити розмір файлів, полегшуючи їх зберігання та передавання без значної втрати якості. Стиснення розміру зображення активно використовується в мультимедіа, архівах, а також у системах відеоспостереження та онлайн-сервісах, для швидкого завантаження веб-сторінок, зокрема [12].

У випадках зміни умов середовища актуальним є використання адаптивних алгоритми, що автоматично підлаштовують параметри обробки. Умови поганого освітлення зокрема відносяться до змін умов.

Ще одним важливим етапом є сегментація – процес поділу зображення на окремі логічні ділянки, кожна з яких має спільні властивості. Процес сегментації може базуватися на простих порогових значеннях, як зазначено раніше, або на більш складних методах кластеризації. Останні групують пікселі за подібністю ознак. Цей метод обробки зображення дозволяє працювати з окремими об’єктами на сцені незалежно, що дуже корисно для розділення об’єктів у складних сценах та підготовки вхідних даних для систем технічного зору [13].

Важливою складовою є також розпізнавання об'єктів. Це складний процес, що включає в себе не лише виявлення певних ознак, таких як форма або колір, а й автоматична класифікація об'єктів на основі виявлених особливостей. Для цього часто використовуються сучасні методи машинного навчання, зокрема нейронні мережі, які здатні вчитися на великих наборах даних і потім розпізнавати об'єкти в нових зображеннях. CNN є одним із найбільш ефективних інструментів для таких задач, оскільки вони здатні обробляти велику кількість візуальної інформації.

Поелементна обробка також є одним з методів управління. Вона передбачає зміну характеристик окремих пікселів чи ділянок, що дозволяє локально підвищувати якість зображення або посилювати важливі графічні елементи. Це може бути, наприклад, корекція контрасту або підсилення певних частин зображення для покращення його суб’єктивного сприйняття.

Всі ці методи обробки зображень є основою для створення складних автоматизованих систем. Застосування сучасних алгоритмів для аналізу зображень дозволяє значно підвищити ефективність роботи таких систем і забезпечити високу точність виконання завдань. Якість цифрового зображення відіграє визначну роль для коректної роботи всього функціоналу систем візуального контролю, технічного зору та автоматизованого моніторінгу. Від точності аналізу та достовірності візуальної інформації залежить ефективність, правильність розпізнавання та надійність рішень, яких було досягнуто в результаті.

Оцінка якості цифрового зображення важливий етап у системах автоматизованої обробки, особливо в тих випадках, коли подальші дії залежать від достовірності вхідної інформації. При аналізуванні графічних елементів розрізняють суб’єктивну та об’єктивну оцінку якості. Перша ґрунтується на візуальному враженні людини та може відрізнятися у різних людей, а друга базується на числових показниках. Це робиться через те, що головну увагу приділено саме технічним характеристикам. Метрики дозволяють кількісно визначити ступінь деталізації, контрастності, чіткості, а також рівня шуму та відхилень від еталону. Тому саме об’єктивні методі зазвичай використовують у автоматизованих алгоритмах.

Якісне зображення має відповідати низці параметрів: воно повинно містити достатньо деталей, мати чітко виражені контури об’єктів, правильний розподіл яскравості та збалансовану кольорову палітру. Важливо, щоб на зображенні не було надмірного шуму, який ускладнює аналіз і перешкоджає точному виділенню об’єктів. Рівень контрасту та яскравості також є визначальними: занадто темне або, навпаки, зображення з пересвітами втрачає свою інформативність. Що дуже впливає на загальну картину у складних сценах із великою кількістю деталей. Особливу увагу слід приділяти балансу білого – його порушення призводить до кольорових викривлень, що можуть спотворити результати подальшої інтерпретації. Також, якщо тіні занадто глибокі, вони можуть приховати важливу інформацію, а недостатня текстура ускладнює диференціацію поверхонь, що має поганий вплив у завданнях промислового контролю. Тобто, кожний показник обробки та управління зображень впливає на сприйняття візуальної інформації не тільки людини, а й систем.

Для оцінювання таких характеристик застосовуються математичні методи – спектральний аналіз, вирівнювання гістограмою, градієнтні моделі, локальні статистики. Спектральні методи, що дозволяють досліджувати частотну структуру зображення. Зокрема, через швидке перетворення Фур'є можна виявити перевищення високочастотних компонентів, які можуть бути пов’язані

з шумом або штучно підвищеною різкістю. Візуальний або автоматичний аналіз спектра дозволяє визначити, наскільки гармонійно розподілена енергія між низькими та високими частотами, та зробити висновок про баланс між деталізацією та чистотою зображення.

У багатьох випадках систем аналізу зображення порівнюються з еталонними або раніше збереженими шаблонами. Для цього використовуються комплексні метрики, що відображають схожість між зображеннями за рядом критеріїв. До них належать індекси, які враховують не лише яскравість і контраст, а й структурну відповідність між зображеннями. Ці індекси називають індексами подібності. Такі підходи дають змогу оцінити не просто загальний вигляд, а й ступінь збереження ключових ознак об’єкта.

1.6 Аналіз сучасних підходів до роботи над зображенням

У сучасних комп’ютерно-інтегрованих системах управління зображеннями вже давно вийшло за межі виключно технічної обробки. Сьогодні цей процес охоплює повний цикл роботи з візуальними даними, а саме: з моменту їх фіксації або завантаження до прийняття обґрунтованих рішень на основі результатів аналізу. Автоматизація цієї сфери відкриває можливості для зменшення залежності від людського втручання, забезпечення стабільності роботи в умовах змін середовища, а також для значного підвищення швидкості обробки та прийняття рішень.

Структура автоматизованих систем роботи із зображеннями, зазвичай, передбачає послідовне виконання кількох функціональних етапів. Першим із них є надходження графічних даних. Воно може здійснюватися з різних джерел – камер спостереження, сканерів, локальних або віддалених сховищ. На цьому рівні важливо забезпечити правильне масштабування, адаптацію форматів і перевірку параметрів для подальшої коректної обробки.

Далі виконується первинна підготовка зображення, яка включає такі дії, як корекція яскравості, видалення шумів, покращення контрастності або інші операції, що забезпечують стабільну основу для аналітичних процедур. Після цього система переходить до аналізу зображення – на цьому етапі відбувається автоматичне виявлення ознак, оцінка якості, класифікація об’єктів, пошук дефектів або інтерпретація окремих структур. Результати обробки можуть напряму впливати на логіку подальших дій: формування звітності, створення рекомендацій або передавання інформації до інших модулів .

Покращення якості цифрових зображень – одна з надзвичайно важливих задач у комп’ютерному зорі, що знаходить широке застосування в автоматизованих системах візуального контролю та управління. Ця задача може бути реалізована різними методами.

Традиційні методи ґрунтуються на використанні процедур фільтрації та трансформації, які допомагають зменшити шуми, підсилити корисну інформацію та налаштувати параметри яскравості й контрасту зображень. До них належать як лінійні, так і нелінійні фільтри згладжування (зокрема, середній, медіанний та гаусівський), які ефективно знижують рівень шуму. Для посилення контурів і деталізації зображення застосовуються фільтри різкості, такі як Unsharp Masking і Laplacian. Методи порогової та градієнтної обробки дають змогу виявляти межі об’єктів, спрощуючи сцену для подальших операцій. Окремо використовуються підходи до регулювання яскравості та контрасту, які спираються на перетворення гістограми. Адаптивний контраст, наприклад, за допомогою алгоритму CLAHE, дозволяє вдосконалити відображення деталей на локальних ділянках зображення. Перевагами цих підходів є їх обчислювальна простота, універсальність та швидкість виконання. Проте, їх застосування вимагає ручного підбору параметрів та часто не забезпечує високої якості в складних умовах.

Із стрімким зростанням обчислювальних можливостей та розвитком штучного інтелекту значна частина задач роботи над зображенням змістилась з класичних методів фільтрації до нейронних мереж.

Нейронні мережі сьогодні демонструють здатність відновлювати деталі. Глибоке навчання дозволяє не просто підвищувати якість зображення, а й приймати рішення на основі змісту сцени, підлаштовуючись під стан ситуації. Підвищення роздільної здатності зображень є одною з ключових в обробці зображень, що є реконструкцією низькоякісного зображення. Особлива ефективність використання таких технологій у промислових технологіях, де погані умови створення візуальної інформації – не рідкість.

Існує декілька методів реалізації підвищення якості. Це може бути як і реконструкція з одного зображення, так і об’єднання кількох кадрів. Є підходи, що спеціалізуються на портретах та обличчях, а є ті, що особливо ефективні для документації. Для кожного підходу є модель, яка вирішує поставлені задаче краще, ніж інші.

Серед сучасних моделей можна виділити:

– модель ESRGAN (Enhanced Super-Resolution GAN), особливість якої в натуральній деталізаціх та реалістичному вигляду зображень;

– модель Real-ESRGAN, що має високу стійкість до спотворень та більш реалізована до реальних зображень;

– модель SwinIR, що ефективна більш на складних прикладах та має високу якість та адаптивність до різних вхідних даних;

– модель RCAN (Residual Channel Attention Network), що добре зберігає важливі деталі, ефективна для структур;

– модель LapSRN (Laplacian Pyramid SR Network), яка краще працює з великими масштабами;

– модель FSRCNN (Fast Cuper-Resolution CNN) має низьке споживання ресурсів, швидку обробку, працює швидше за інші.

У автоматизованих системах має значення не лише покращення або фільтрація, а й інтерпретація та їх аналіз. Поєднання візуальної обробки з аналізом контексту забезпечує більшу гнучкість та адаптованість програмних рішень. Основу цьому підходу дає використання інструментів штучного інтелекту, які дозволяють системі самостійно здійснювати аналіз зображення.

Сьогодні існують різні рішення для цього підходу. Зокрема, модель Contrastive Language–Image Pretraining (CLIP) забезпечує єдиний простір для зіставлення зображень і текстових описів, що робить її зручною для універсальних сценаріїв використання. Модель Bootstrapped Language–Image Pretraining (BLIP) орієнтована на генерацію описів зображень і підтримує двосторонню відповідність між візуальною та текстовою інформацією. Для виявлення об'єктів без необхідності ручної розмітки використовується підхід Self-distillation with No Labels (DINO), тоді як Vision Transformer (ViT) застосовується для класифікації зображень і потребує навчання на спеціалізованих наборах даних. Модель Segment Anything Model (SAM) дозволяє проводити семантичну сегментацію об'єктів без попереднього тренування на визначених класах. Системи реального часу та відеоспостереження часто застосовують алгоритми YOLO-NAS, що дають змогу швидко виявляти та класифікувати об'єкти, хоча й з меншою точністю. Крім того, класичні згорткові нейронні мережі, такі як Residual Neural Network (ResNet) та Efficient Convolutional Neural Network (EfficientNet), знаходять широке застосування для глибокої класифікації зображень, забезпечуючи високу точність і стабільність результатів.

2 ПРОЄКТУВАННЯ СИСТЕМИ АВТОМАТИЗАЦІЇ УПРАВЛІННЯ ЗОБРАЖЕННЯМИ

2.1 Визначення ролі зображень в автоматизації та комп’ютерно-інтегрованих технологіях

Використання автоматизованих алгоритмів, здатних самостійно аналізувати, покращувати та класифікувати зображення, дозволяє істотно зменшити кількість людських помилок, підвищити швидкість ухвалення рішень та забезпечити стабільність роботи навіть у змінних або складних умовах. В умовах сучасного виробництва, автоматизація процесів якого розвивається з шаленою швидкістю, візуальні дані, що подаються у графічному вигляді, зображення зокрема, набувають особливої значущості. Сучасні комп’ютерно-інтегровані технології дозволяють використовувати такі візуальні матеріали як інструмент для аналізу, моніторингу та оптимізації процесів, що сприяє зростанню загальної продуктивності підприємства.

Технології, засновані на комп’ютерній інтеграції, задіяні в усіх етапах життєвого циклу продукції. Від створення концепції при проектуванні до її логістичних моментів, зокрема. Вони забезпечують єдину інформаційну взаємодію між усіма складовими виробничої системи. У цьому процесу і зображення відіграють, якщо не ключову, то надзвичайно важливу роль: замість суб'єктивного оцінювання оператором, вони застосовуються для постійного нагляду за процесами, ідентифікації дефектів, оцінки зовнішнього вигляду виробів, а також для швидкого виявлення об’єктів і підтримки оперативного прийняття рішень. Наприклад, під час візуального контролю якості система за допомогою камери фіксує навіть незначні відхилення у кольорі чи формі, після чого автоматично порівнює отримане зображення з контрольним зразком і визначає подальшу долю продукції. У цьому контексті важливе місце займають

спеціалізовані програмні рішення, здатні автоматизувати попередню обробку, аналіз та покращення зображень.

У сучасних виробничих системах аналіз зображень усе частіше відбувається безпосередньо на етапі технологічного процесу, з мінімальним або повністю відсутнім втручанням оператора. Такий підхід дозволяє досягти високої швидкості реагування та значно знижує ризики, пов’язані з людським фактором. Отримані з камер або сенсорів зображення можуть у режимі реального часу оброблятись програмним забезпеченням, після чого результати аналізу автоматично передаються до програмованих логічних контролерів (PLC) для подальшого управління обладнанням або системою.

Застосування програм у реальному часі забезпечує не лише оперативність, але й високу точність, що критично важливо для автоматизованих систем, де навіть незначні відхилення параметрів виробу можуть впливати на якість готової продукції або безпеку процесу.

Прикладом важливості є й логістичні процеси, системи обліку та відеоспостереження. Камери фіксують маркування товарів, аналізують правильність їхнього розміщення, а в разі відхилень – автоматично генерують сигнали до системи управління. Зображення використовуються для розпізнавання штрихкодів та QR-кодів, верифікації упаковки, виявлення небезпечних ситуацій або порушень на складах. Таким чином, зображення стають не лише засобом фіксації стану, а й інструментом оперативного реагування. Значущою перевагою для автоматизованої системи обробки є інтеграція з базами даних, звітними модулями та аналітичними платформами. Завдяки цьому стає можливим відстеження поточного стану виробництва, накопичення інформації з подальшою її обробкою для виявлення закономірностей або прогнозування потенційних збоїв [14].

Автоматизація управління цифровими зображеннями зазвичай включає реалізацію програмно-апаратних комплексів, які самостійно виконують обробку, збереження, аналіз і передачу візуальної інформації без участі оператора.

Завдяки інтеграції з базами даних, системами SCADA або логістичними платформами, результати аналізу можуть негайно використовуватись для керування технічними або бізнес-процесами.

Не можна обійти і тему упровадження технологій штучного інтелекту в системи цифрової обробки зображень. Це можна назвати якісним стрибком у напрямі розвитку автоматизованих платформ, де аналіз візуальної інформації більше не обмежується лише заздалегідь визначеними правилами. Моделі машинного навчання можуть бути використані для покращення якості, класифікації вмісту або надання рекомендацій щодо подальшої обробки.

Усе частіше такі системи розширюються функціоналом адаптивного навчання, де AI-алгоритми оновлюють свої моделі на основі нових зображень. Це дозволяє підвищувати точність класифікації об’єктів, виявлення небезпек або аномалій у реальному часі. Замість традиційних, суворо регламентованих підходів, застосовуються адаптивні моделі, здатні самостійно навчатися, узагальнювати нові шаблони та приймати рішення навіть за умов невизначеності або змін середовища.

Застосування технологій штучного інтелекту, зокрема нейронних мереж і методів машинного навчання, критично змінило підхід до обробки цифрових зображень у всіх сферах життя, комп’ютерно-інтегрованих системи зокрема. Сучасні алгоритми здатні проводити базову фільтрацію чи покращення якості, виконувати складні завдання: розпізнавання об’єктів, виявлення дефектів, пошук аномалій у зображенні, адаптації до нових умов середовища та прогнозування потенційних відхилень на основі накопичених даних. Тобто, покращити виконання багатьох існуючих функцій, які застосовувались до впровадження нових технологій.

Однією з найпотужніших у цій сфері є глибинне навчання, зокрема згорткові нейронні мережі. Convolutional Neural Networks (CNN) дозволяють з високою точністю ідентифікувати навіть малопомітні аномалії на поверхні виробів або передбачити можливі дефекти ще до їх виникнення, аналізуючи

послідовності зображень з виробничого процесу. Такі підходи істотно зменшують вплив людського фактору та сприяють впровадженню своєчасного обслуговування. Завдяки цим моделям стало можливим вирішувати завдання, що раніше вважалися надто складними для автоматизації. Зокрема, йдеться про підвищення роздільної здатності зображень із втратами, інтелектуальну реконструкцію пошкоджених ділянок, автоматичну колоризацію монохромних знімків, а також виявлення дефектів на складних фактурах, де традиційні методи часто дають збої.

Штучний інтелект виявляється надзвичайно цінним у сценаріях, де якість вхідного зображення залежить від динамічних зовнішніх умов: зміни освітлення, переміщення об’єктів, нечіткість через фокусування тощо. Класичні алгоритми в таких умовах швидко втрачають стабільність, тоді як AI-моделі, завдяки властивості до генералізації, здатні адаптуватися без необхідності ручного втручання. Поєднання класичних методів обробки з гнучкими алгоритмами штучного інтелекту дозволяє створити інтелектуальну систему нового покоління.

2.2 Вибір середовища розробки та інструментів

Система автоматизації управління зображеннями – це програмне рішення, що дозволяє користувачам здійснювати комплексну обробку візуальних даних за допомогою інтегрованих алгоритмів.

Під час створення програмного забезпечення для автоматизації процесів роботи з цифровими зображеннями постало завдання обрати таке програмне середовище, яке б одночасно забезпечувало гнучкість, стабільність, широкі можливості обробки та доступність у використанні. Провівши огляд сучасних технологічних рішень, було прийнято рішення зупинитися на мові програмування Python.

Для розробки системи було здійснено вибір середовища розробки PyCharm. PyCharm – це інтегроване середовище розробки (Integrated Development Environment, IDE), створене компанією JetBrains для розробки програмного забезпечення на мові Python. Це середовище забезпечує широким набором інструментів, включаючи засоби для налагодження коду, тестування, інтеграції з системами контролю. PyCharm сприяє швидкому створенню графічних інтерфейсів та інтерактивних додатків, що є важливим для реалізації програм для роботи з візуальними даними. У контексті обробки зображень це програмне середовище є зручним для роботи з бібліотеками. Є відстеження помилок та налагодження, можна детально проаналізувати роботу алгоритмів, перевірити значення змінних та оптимізувати код [15].

Вигляд базового вікна середовища можна побачити на рис. 2.1.

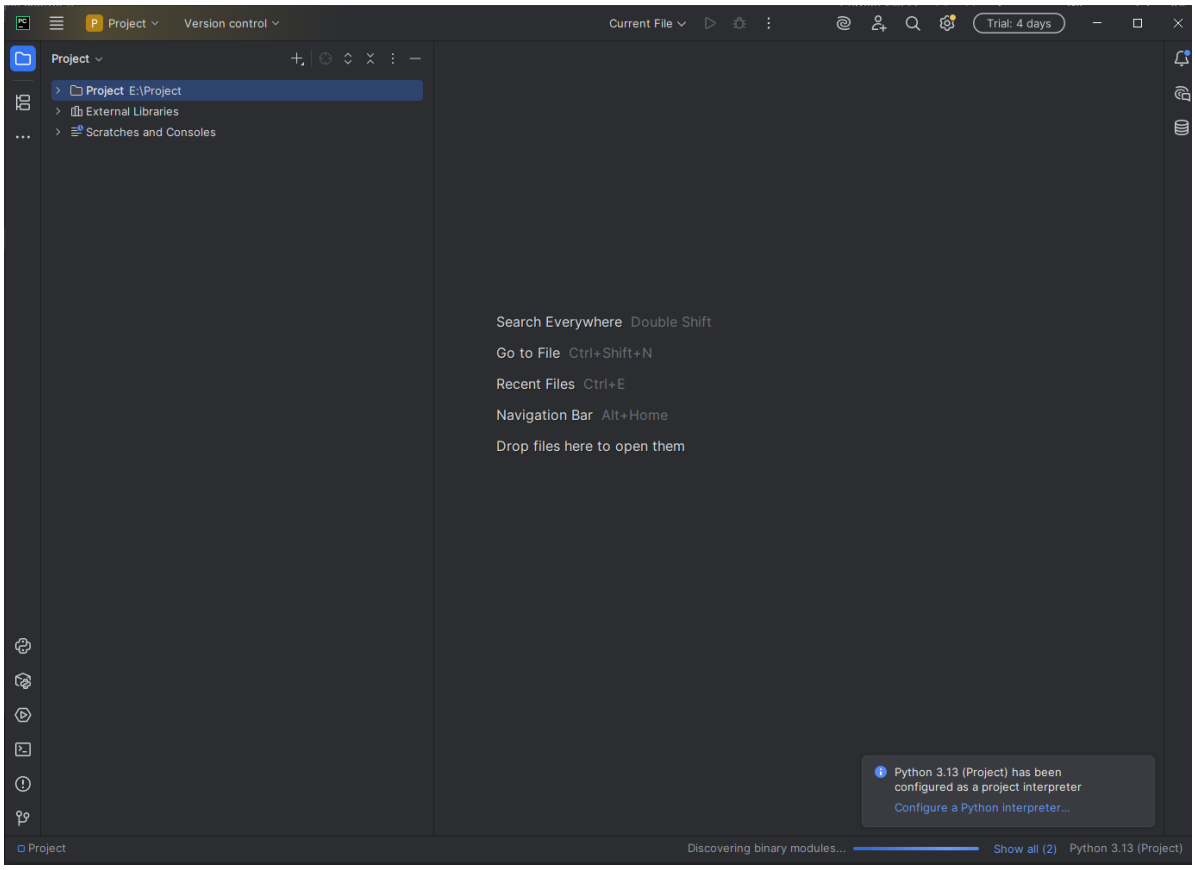

Рисунок 2.1 – Середовище розробки PyCharm

Таким чином, PyCharm стає не лише середовищем для написання коду, а й потужним інструментом для побудови, тестування та відлагодження програм для управління та обробки зображень, що сприяє зменшенню кількості помилок та підвищенню продуктивності розробника.

Для реалізації системи автоматизації обробки цифрових зображень було обрано мову програмування Python, оскільки вона поєднує простоту синтаксису, широку підтримку бібліотек для обробки зображень та інструменти інтеграції з моделями нейронних мереж. При використанні дозволяється швидко створювати прототипи, легко модифікувати логіку системи та забезпечує кросплатформенність розробленого застосунку.

Розробка програмного забезпечення для автоматизації управління зображеннями повинна реалізуватися з використанням ряду бібліотек, які забезпечують обробку візуальних даних та створення графічного інтерфейсу користувача.

Графічний інтерфейс користувача реалізовано за допомогою стандартної бібліотеки Tkinter, яка є інтегрованою в Python і не потребує додаткового встановлення. Вона забезпечує набір базових елементів управління, дозволяє швидко реалізувати інтерфейс з кнопками, панелями, слайдерами та меню, що особливо важливо для створення зручного середовища взаємодії з користувачем. Саме ця бібліотека обрана з огляду на її стабільність, нескладність в освоєнні та відсутність зовнішніх залежностей.

Для реалізації функцій обробки зображень використано бібліотеку Pillow, яка є розширеною версією класичної Python imaging library (PIL). Вона надає повний набір засобів для відкриття, редагування, збереження, конвертації, масштабування, фільтрації та редагування пікселів.

Модулі цієї бібліотеки: Image, ImageFilter, ImageEnhance, ImageOps, ImageTk. Використання цих модулів, що входять до складу Pillow, дозволяє ефективно реалізовувати основні обробки. Модуль Image виступає базовим інструментом для роботи із самими зображеннями, тоді як ImageTk дозволяє

інтегрувати ці зображення в інтерфейс Tkinter. Модуль ImageOps забезпечує базові перетворення, зокрема інверсію кольорів, а ImageFilter реалізує стандартні фільтри для згладжування та розмиття. Завдяки ImageEnhance досягається можливість налаштування контрасту, яскравості та чіткості, що дозволяє користувачу точно контролювати вигляд зображення.

OpenCV (Open source computer vision library) потужна бібліотека з відкритим кодом, призначена для обробки зображень і комп'ютерного зору. Вона широко використовується в академічних і промислових проєктах. Переваги такої адаптації в тому, що вона здатна реалізувати сотні алгоритмів для аналізу зображень, підтримує обробку в реальному часі, надає інструменти не тільки для роботи із зображеннями, а й відео, камерами, графікой, виявленням об'єктів та фільтрацією. Вона доповнила функціонал Pillow і дозволила реалізувати складніші аналітичні методи.

Одна з найпоширеніших бібліотек PyTorch ідеально вписується в концепцію фундаменту для створення та навчання глибоких нейронних мереж, що створена Facebook AI Research. Це гнучка модель обчислень, що має багато готових моделей та рішень. Інтегрується з бібліотекою torchvision, яка містить набори даних і моделей для зображень. Завдяки відкритій архітектурі бібліотеки torch та доступності попередньо натренованих моделей, вдалося адаптувати механізм для автоматичного розпізнавання типів сцен на зображеннях Contrastive language-image pretraining (CLIP), що розроблений OpenAI.

Окрему роль у структурі застосунку відіграють стандартні модулі Python. Для взаємодії з файловою системою використано модуль os, який забезпечує доступ до директорій, перевірку шляхів, а також збереження результатів обробки в обрані папки. Крім того, для повідомлення користувача про результати операцій використовується модуль messagebox, який входить до складу tkinter і відповідає за створення інформативних вікон із повідомленнями. Модуль threading застосовується для забезпечення асинхронного виконання тривалих операцій, зокрема фільтрації, аналізу або збереження зображень. У традиційному

сценарії графічний інтерфейс може зависнути під час обробки ресурсоємних задач. Впровадження багатопотоковості дозволяє виконувати такі задачі у фоновому режимі, зберігаючи при цьому чуйність інтерфейсу. Модуль json відповідає за структуроване збереження даних. Застосування формату JSON дозволяє не тільки зберігати ці дані у вигляді текстових файлів, а й інтегрувати результати з іншими програмами або системами.

Вибір саме таких інструментів забезпечує просту інсталяцію, гнучкість розширення функціоналу та відповідність технічному завданню системи. Поєднання цих елементів дозволяє реалізацію функціонального, гнучкого та зручного застосунку із можливістю повноцінної роботи з цифровими зображеннями.

2.3 Опис загальної архітектури системи автоматизації управління зображенням

Розроблена програма для автоматизації управління зображеннями повинна мати чітко визначену архітектурну структуру. Система побудована за модульною архітектурою, що дозволяє ефективно розділити функціональність між окремими блоками, забезпечити гнучкість масштабування та полегшити підтримку в майбутньому. Такий підхід сприяє ізоляції логіки, швидкому тестуванню окремих компонентів і простоті модернізації без впливу на інші частини системи.

Основна архітектурна модель побудована за принципом розділення на логічні компоненти, кожен з яких виконує свою окрему функцію.

Для кращого розуміння взаємодії було розроблено блок-схему (рис. 2.2).

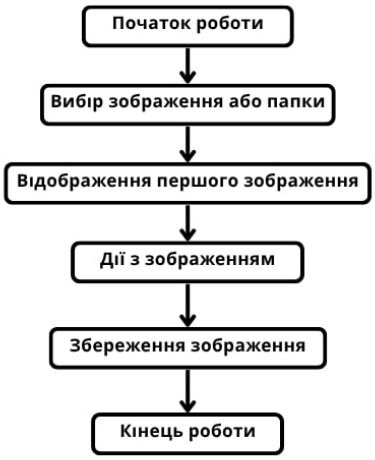

Рисунок 2.2 – Блок-схема взаємодії користувача з програмою

Вона забезпечує візуалізацію основних етапів роботи програми та відображає послідовність дій користувача. Складається з основних етапів, які формують логічний процес роботи з зображеннями у додатку.

Початок роботи: користувач запускає застосунок, що відкриває головне вікно з інтерфейсом. Це стартова точка роботи, яка надає доступ до основних функцій системи.

Вибір зображення або папки: система дозволяє обрати окремий файл або папку з зображеннями для пакетної обробки. Цей крок ініціює процес завантаження файлів у програму, що дозволяє відобразити їх у робочій області.

Відображення першого зображення: після завантаження перше зображення автоматично відображається у робочій зоні програми. Це дозволяє користувачеві одразу побачити поточне зображення, з яким будуть виконуватись сценарії.

Дії з зображенням: користувач може застосовувати до зображення різні операції. Цей етап є основним у процесі обробки зображень, оскільки реалізує ключові функціональні можливості програми. А саме: на цьому етапі користувач може взаємодіяти із зображенням за кількома напрямками: аналіз якості зображення, автоматичне визначення типу зображення, отримання згенерованих

рекомендацій під тип зображення, ручна обробка зображення, відслідковування модулю

Збереження зображення: після завершення обробки користувач може зберегти змінене зображення у обраному форматі та згенерувати звіт, який можна використовувати для документації.

Кінець роботи: завершення роботи із зображенням або вихід з програми. На цьому етапі користувач може закрити програму або повернутись до початку процесу для обробки іншого файлу.

З метою кращого розуміння загальної логіки роботи системи було розроблено схему загальної архітектури (рис. 2.3).

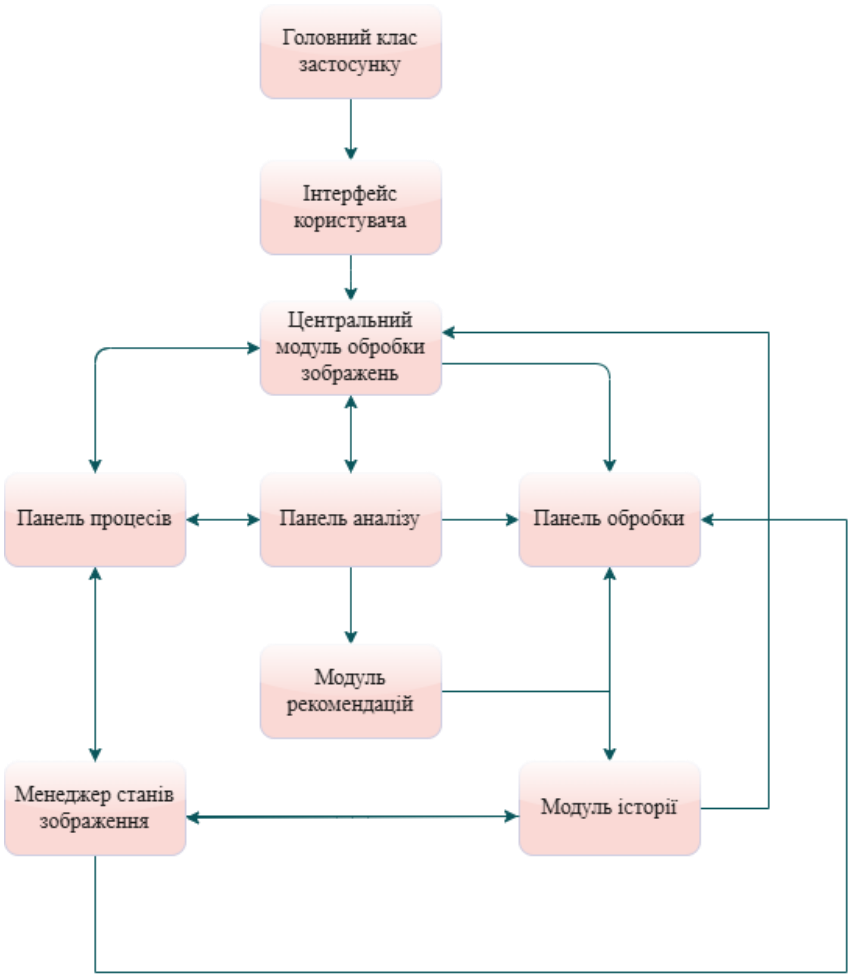

Рисунок 2.3 – Схема загальної архітектура системи

Схема демонструє взаємозв’язки між основними модулями системи та ілюструє послідовність обробки вхідних даних: від завантаження та обробки зображення до формування результатів та рекомендацій. Також візуально представлено концепцію гнучкості системи, де кожен модель передає інформацію в інший.

На верхньому рівні архітектури розташовано головний клас застосунку, який відповідає за ініціалізацію віконного середовища, побудову базового інтерфейсу та з’єднання між усіма функціональними компонентами.

Інтерфейс користувача реалізовано за допомогою бібліотеки, яка надає набір елементів для створення інтерактивних елементів управління. Інтерфейс структуровано у вигляді інтерактивних панелей зі зручним управлінням через кнопки, слайдери, меню та перемикачі.

Центральне місце займає модуль обробки зображень, що забезпечує виконання основних фільтраційних операцій. До його складу входять функції регулювання контрасту, яскравості, різкості, а також інструменти розмиття та перетворення кольору. Крім цього, реалізовано функції масштабування та обертання.

Кожна панель або функціональний модуль розроблені у вигляді окремого класу, що відповідає за свою частину логіки. Наприклад, панель процесів реалізує інструменти зміни яскравості, контрасту, фільтрації та трансформацій, Панель пресетів дозволяє зберігати та застосовувати типові набори обробки, а панель аналізу відповідає за розрахунок технічних параметрів якості зображення. Всі ці модулі незалежні, але взаємодіють через чітко визначені інтерфейси, що підвищує гнучкість системи.

Ключовим компонентом є менеджер станів зображення ImageStateManager, що зберігає оригінал, базову версію після застосування фільтрів, та поточне зображення з врахуванням масштабування, повороту або інших дій. Така структура дозволяє уникнути повторного застосування фільтрів при зміні масштабу та зберігає точність обробки.

Модуль аналізу якості виконує діагностику зображення за рядом технічних характеристик. Завдяки властивостям класифікації, цей модуль може визначати тип сцени без попереднього навчання: портрет, документ, інтер’єр, пейзаж, нічне фото тощо, що дозволить адаптувати поведінку системи до змісту зображення. На основі результатів аналізу та класифікації працює модуль рекомендацій, який формує індивідуальні підказки щодо покращення.

Додатково система містить модуль історії змін, який дає змогу відслідковувати та скасовувати дії користувача. Це реалізовано шляхом збереження серії кроків зі збереженими копіями зображення та відповідними параметрами.

Загалом архітектура орієнтована на ефективну автоматизовану роботу зі зображеннями, адаптивність до контексту та максимальну зручність для кінцевого користувача. Загальна архітектура застосунку є чітко структурованою та гнучкою до розширення. Модульний принцип дозволяє додавати нові панелі, фільтри або алгоритми без необхідності модифікації всієї системи. Це важливо в умовах подальшого розвитку та адаптації програмного забезпечення до нових завдань.

2.4 Розробка алгоритму роботи системи автоматизації управління зображенням

Функціонування розробленої системи базується на чіткій послідовності етапів. Тобто, це логіка взаємодії користувача з програмним інтерфейсом, в основі якої миттєва обробка запитів. Етапи забезпечують повний цикл взаємодії з цифровими зображеннями: від моменту їхнього відкриття до збереження результатів обробки та супровідної інформації. Після запуску програми відбувається ініціалізація всіх основних компонентів: віконного середовища, панелей керування, інструментів редагування та службових модулів. Структура інтерфейсу реалізована за принципом багатовіконного середовища з можливістю

інтерактивного перемикання між функціональними модулями без потреби у повторному завантаженні чи перезапуску.

На першому етапі користувач завантажує зображення або цілу серію файлів. Всі дані автоматично адаптуються під розміри робочої області перегляду, при цьому зберігаючи недоторканою оригінальну версію. Подальша обробка здійснюється лише з копією – базовим зображенням, що виключає ризик втрати початкових даних.

За потреби активується модуль аналізу якості, який проводить автоматизовану оцінку різкості, контрасту, балансу білого, рівня шуму та спектральних характеристик. У випадку увімкнення інтелектуального режиму, система виконує класифікацію типу зображення за допомогою моделі CLIP. Це дозволяє адаптувати межі оцінювання до контексту сцени. На основі зіставлення отриманих метрик із типом сцени програма формує індивідуальні рекомендації щодо покращення якості. Якщо певні параметри зображення виходять за межі встановлених нормативів для визначеного типу, система автоматично генерує адаптивні поради щодо покращення. Наприклад, при виявленні високого шуму в нічному фото буде запропоновано застосувати шумозаглушення, а при низькій різкості в документі – підвищити чіткість.

Далі користувач може перейти до активної фази редагування. Через графічний інтерфейс із зручними панелями та слайдерами застосовуються базові фільтри, такі як регулювання контрасту, яскравості, насиченості, обертання або масштабування. У разі потреби налаштування можуть бути збережені у вигляді пресетів, що дозволяє пришвидшити обробку серій подібних зображень. Після переходу до режиму редагування, користувач отримує доступ до повного набору фільтрів і трансформацій. Застосування кожного інструменту супроводжується реєстрацією змін у системному журналі, що дозволяє здійснювати відкат до попередніх кроків.

Після завершення редагування користувач має змогу зберегти результат у вигляді нового графічного файлу. Одночасно система формує JSON-файл зі

звітом, до якого включаються обчислені метрики, класифікаційна інформація та рекомендації. Це забезпечує можливість подальшого аналізу або аудиту обробки.

Для підвищення продуктивності реалізовано функцію пакетної обробки зображень. Програма здатна працювати у фоновому режимі, відслідковуючи появу нових файлів у заданій директорії. Кожне зображення автоматично аналізується, проходить обробку за наперед визначеним сценарієм і зберігається в окрему цільову папку. Такий підхід особливо ефективний при роботі з великими масивами візуальних даних.

Загальна структура алгоритму є модульною, адаптивною та розширюваною. Це дозволяє легко інтегрувати нові компоненти, алгоритми фільтрації чи методи аналізу, не змінюючи базову архітектуру. Система орієнтована на забезпечення гнучкості, стабільності та простоти у використанні як для досвідчених фахівців, так і для користувачів без попередньої технічної підготовки.

2.5 Теорія автоматичного управління у системі автоматизації управління зображеннями

Сучасні системи цифрової обробки зображень, що функціонують у рамках комп’ютерно-інтегрованих технологій, тісно пов’язані з принципами ТАУ. Вона забезпечує математичні методи опису, аналізу та синтезу динамічних систем.

Система автоматизації обробки цифрових зображень розглядається як динамічний контур управління, де об'єктом є зображення у процесі обробки, а керуючими діями можна вважати набір алгоритмічних впливів. Працює в замкнутому циклі: спочатку отримання вхідних даних, аналізування, рішення, управління та повторна перевірка.

Застосування методів ТАУ дозволяє формалізувати принципи роботи такої системи через поняття регульованої змінної (якість зображення за заданим

критерієм), сигналу управління (параметри обробки) та внутрішньої зворотної дії (автоматичний аналіз результату). Для цілей моделювання система може бути подана у вигляді структурної моделі із внутрішнім зворотним зв'язком, в якій об'єкт управління можна вважати зображення у поточному стані, а регулятор – це інтелектуальний блок AI-аналізу, що приймає рішення про тип сцени, рівень шуму, контрасту чи освітленості [16].

Було розглянуто спрощену модель системи, де передатна функція регулятора описує залежність між поточним станом зображення та бажаним результатом. Аналіз стійкості такої моделі може проводитись як за алгебраїчними, так і за частотними критеріями. Наприклад, у разі використання зворотного зв'язку у вигляді порівняння оцінки якості до та після обробки, можна визначити: чи призводить така дія до стабільного покращення результату, чи викликає коливальні процеси, що знижують ефективність.

Крім того, можливо здійснювати синтез параметрів обробки за критерієм мінімуму інтегральної помилки між поточним та бажаним станом зображення. Це дозволяє реалізувати оптимальну політику керування, що відповідає принципам ТАУ і зменшує відхилення від еталону при мінімальних змінах параметрів.

Система, реалізована в програмному середовищі Python, імітує поведінку адаптивного ПІ-регулятора з динамічним підбором коефіцієнтів, орієнтованих на якісну оцінку зображення. Таким чином, у процесі роботи система не тільки виконує обробку, а й активно управляє нею відповідно до заданої цілі – покращення візуальних характеристик з урахуванням змісту зображення.

На основі концепції теорії керування, поведінку управління зображенням можна описати операторними або частотними моделями, які враховують реакцію системи на зміну вхідних параметрів. Модулі системи автоматизації можна апроксимувати передатною функцією типу:

$$W(p) = \frac{K}{T_p + 1}, \tag{2.1}$$

де K – коефіцієнт підсилення (інтенсивність);

T – стала часу реакції (затримка в обробці).

$$K = 1 + \frac{intensity}{100}; \tag{2.2}$$

де K – коефіцієнт контрасту.

Розрахунок часу перехідного процесу:

$$t_{\text{перехід}} \approx 3\text{T} = 3 \cdot 0{,}5 = 1{,}5 \text{ с}. \tag{2.3}$$

Перевірка стійкості за критерієм Гурвіца (одночленне рівняння):

$$T_p + 1 = 0, \tag{2.4}$$

$$p = -\frac{1}{\text{T}} = -2. \tag{2.5}$$

Оскільки дійсна частина p від'ємна, система є стійкою.

Для підтвердження отриманої моделі модуля було виконано комп'ютерне моделювання перехідної характеристики за допомогою мови програмування Python. Скрипт описує експоненційний закон наростання вихідного сигналу, що характерний для інерційної ланки першого порядку.

Побудований графік демонструє плавне досягнення нового рівня підсилення контрастності протягом часу, що підтверджує коректність проведених розрахунків (рис. 2.4).

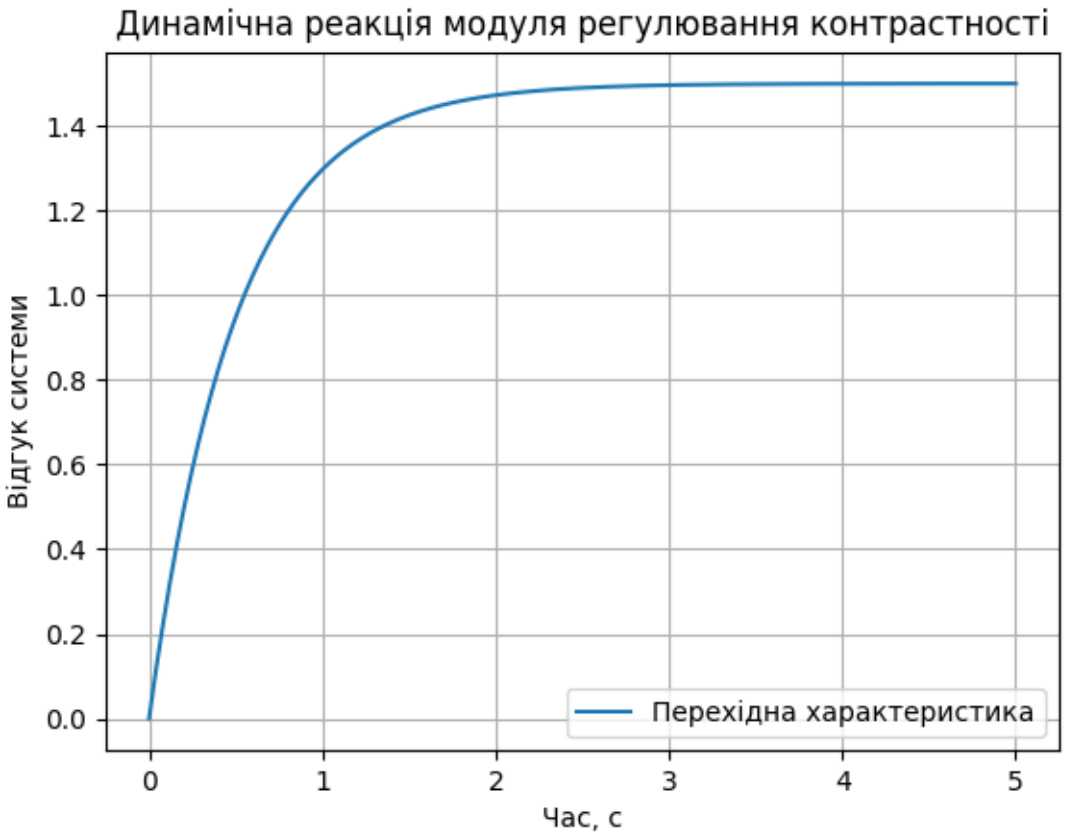

Рисунок 2.4 – Графік перехідної характеристики динамічної реакції модуля

Фрагмент коду, використаного для моделювання:

```
import matplotlib.pyplot as plt
import numpy as np

T = 0.5
K = 1.5
t = np.linspace(0, 5, 500)
y = K * (1 - np.exp(-t / T))

plt.plot(t, y, label="Перехідна характеристика")
plt.title("Динамічна реакція модуля регулювання контрастності")
plt.xlabel("Час, с")
```

```
plt.ylabel("Відгук системи")
plt.grid(True)
plt.legend()
plt.show()
```

Отже, застосування принципів теорії автоматичного управління дозволяє надати формальну основу для функціонування інтелектуальних редакторів зображень забезпечити адаптивність їх роботи та підвищити якість результатів за рахунок гнучкого та керованого підходу до кожного окремого випадку.

Можна зазначити, що використання таких принципів дозволяє реалізувати замкнуту систему керування зображенням, де кожен крок обробки є реакцією на актуальний стан. Завдяки цьому автоматична оптимізація цілком можлива. Також стійкість системи підвищується до впливів зовнішніх факторів. Різні типи шуму, змінення умов освітлення під час зйомки.

Впровадження принципів теорії автоматичного управління в систему автоматизації роботи над зображенням дозволяє підніти її на рівень інтелектуального керування, де програмне забезпечення не просто має алгоритм певних ій, а активно адаптується до конкретної ситуації.

Таким чином, це дозволяє підвищити точність контролю та зробити його максимально гнучким до складних та нестабільних вхідних сигнлів.

3 РОЗРОБКА СИСТЕМИ АВТОМАТИЗАЦІЇ ДЛЯ УПРАВЛІННЯ ЗОБРАЖЕННЯМИ

3.1 Реалізація модулів системи

3.1.1 Модуль інтерфейсу

Інтерфейс користувача – це центральний елемент у процесі взаємодії оператора з програмною системою. Він слугує засобом управління функціональними модулями та візуального контролю за результатами обробки.

На етапі проєктування інтерфейсу головною метою було досягнення інтуїтивності, логічної структури та функціональної гнучкості відповідно до сценаріїв використання. Для реалізації обрано бібліотеку Tkinter, яка входить до стандартного складу мови Python.

Підкресленням важливості інтерфейсу є і його візуальна ідентифікація. Тому для програми був розроблений графічний логотип (рис. 3.1).

Рисунок 3.1 – Логотип системи автоматизації управління цифровими зображеннями

У системі така іконка (рис. 3.1) була додана для відображення у заголовку вікна, на панелі завдань операційної системи. Це забезпечує не лише завершений вигляд інтерфейсу, а й дозволяє користувачеві швидше ідентифікувати програму серед інших відкритих вікон.

Графічний файл іконки збережено у форматі *.ico. Це сумісно із більшістю версій операційної системи Windows. Для його інтеграції у вікно застосунку використано стандартні засоби бібліотеки Tkinter.

Встановлення іконки виконується за допомогою наступного фрагмента коду:

```
icon_path = os.path.join("assets", "icon.ico")
self.root.iconbitmap(icon_path)
```

Логотип проєкту розроблений відповідно до тематики програмного забезпечення: зображення пов’язане з цифровою графікою та управлінням зображеннями за допомогою ПК. Виконаний в кольорах, що позитивно впливають на роботу користувача. Таким чином, логотип виконує декоративну роль та відображає зміст і функціональне призначення системи автоматизації.

Візуальний стиль плавно поєднується із організацією інтерфейсу, що складається з трьох основних зон.

Верхня частина вікна містить меню інструментів, у якому зосереджено основні дії: відкриття зображень, збереження результатів, виклик модулів обробки, запуск автоматичних сценаріїв та експорт технічних звітів. У центрі розташована зона виводу зображення, яка відображає актуальний стан файлу та автоматично оновлюється при внесенні змін. Праворуч розміщено функціональний блок, що динамічно змінюється в залежності від обраного режиму роботи: фільтрація, аналіз якості, застосування пресетів, рекомендації або перегляд історії змін.

Головне вікно розробленої системи на рис. 3.2.

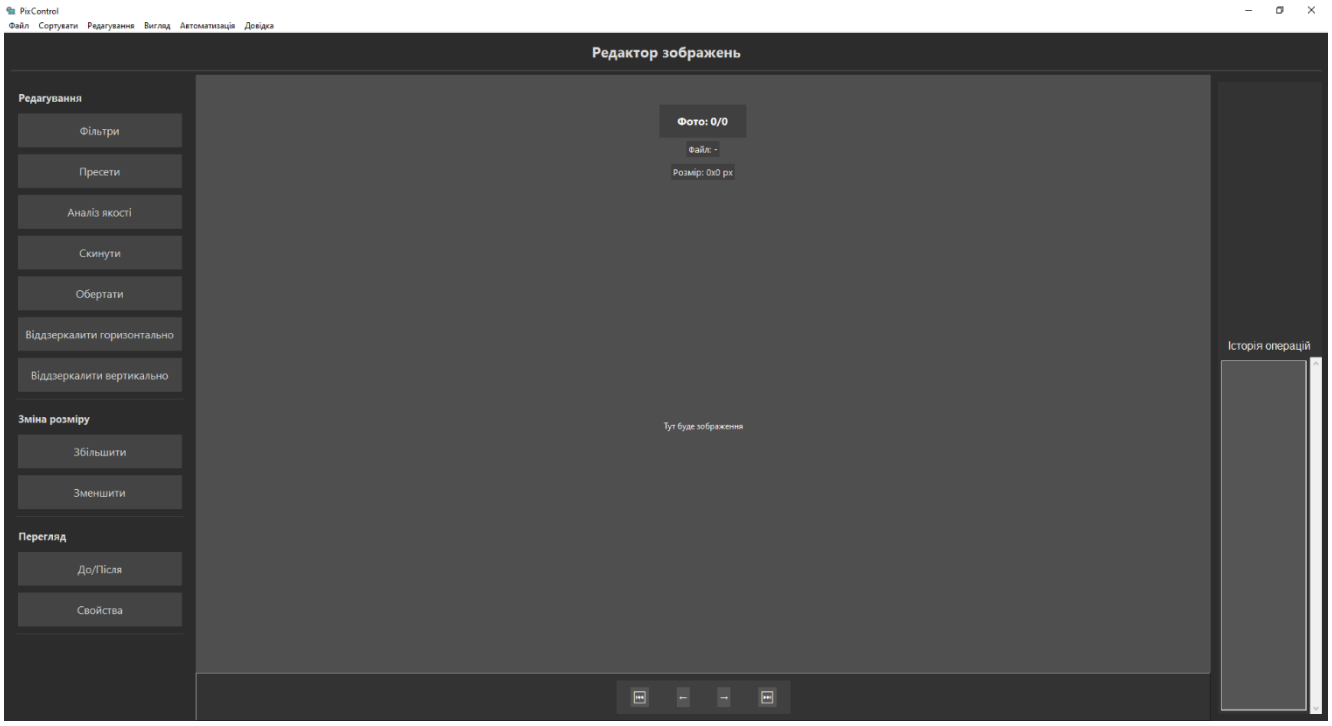

Рисунок 3.2 – Головне вікно програми

Передбачено підтримку адаптивної зміни розмірів вікна. Масштаб зображення та його положення розраховуються автоматично, що дозволяє зберігати правильне відображення без втрати якості або спотворення пропорцій. Для забезпечення доступу до розширених налаштувань реалізовано вертикальну область прокручування Scrollbar, яка дозволяє працювати навіть з великою кількістю керуючих елементів у правій панелі.

Кожна інтерактивна компонента інтерфейсу, зокрема кнопки та слайдери, має чітко визначене функціональне призначення та витримане мінімалістичне оформлення. Натискання на кнопку активує відповідний фільтр, а під нею з’являється панель налаштувань. Слайдер інтенсивності обраного фільтру дозволяє коригувати параметри в реальному часі.

Активовану панель фільтрів наведено на рис. 3.3.

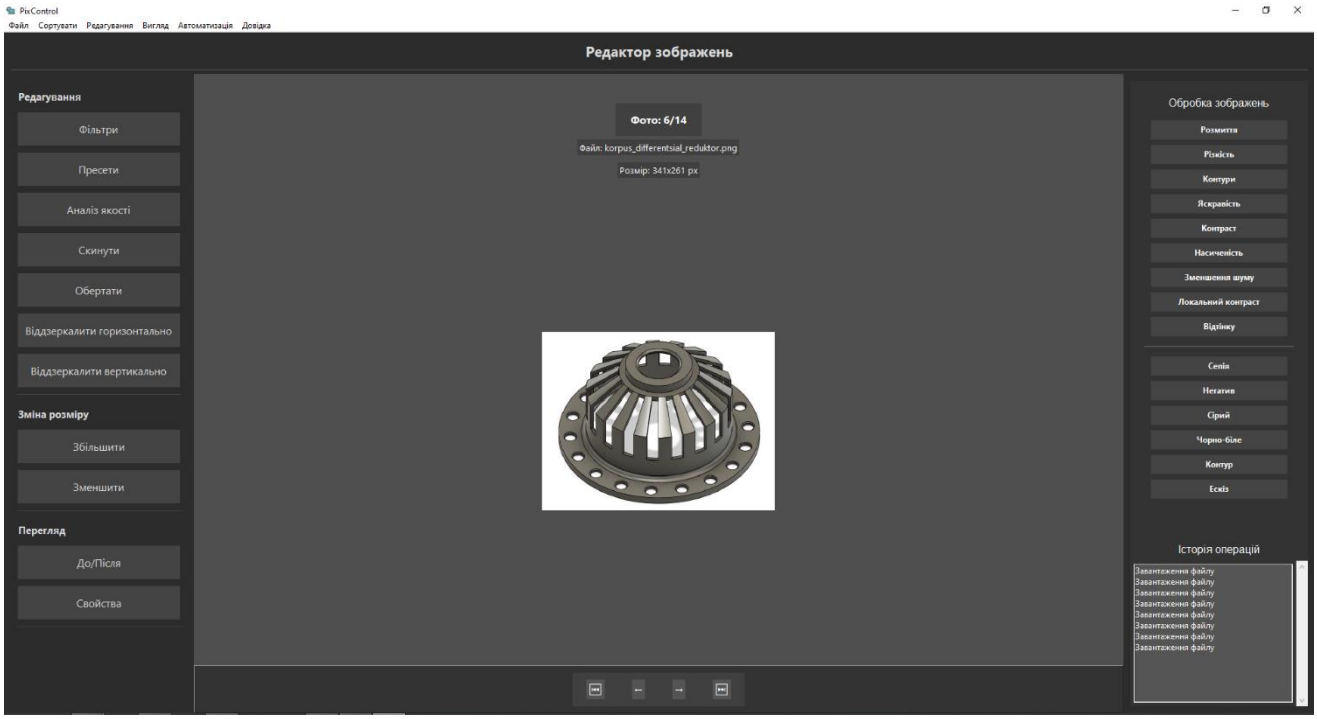

Рисунок 3.3 – Активована панель фільтрів

Повторне натискання приховує налаштування, що дозволяє зберігати компактність і зручність перегляду (рис. 3.4).

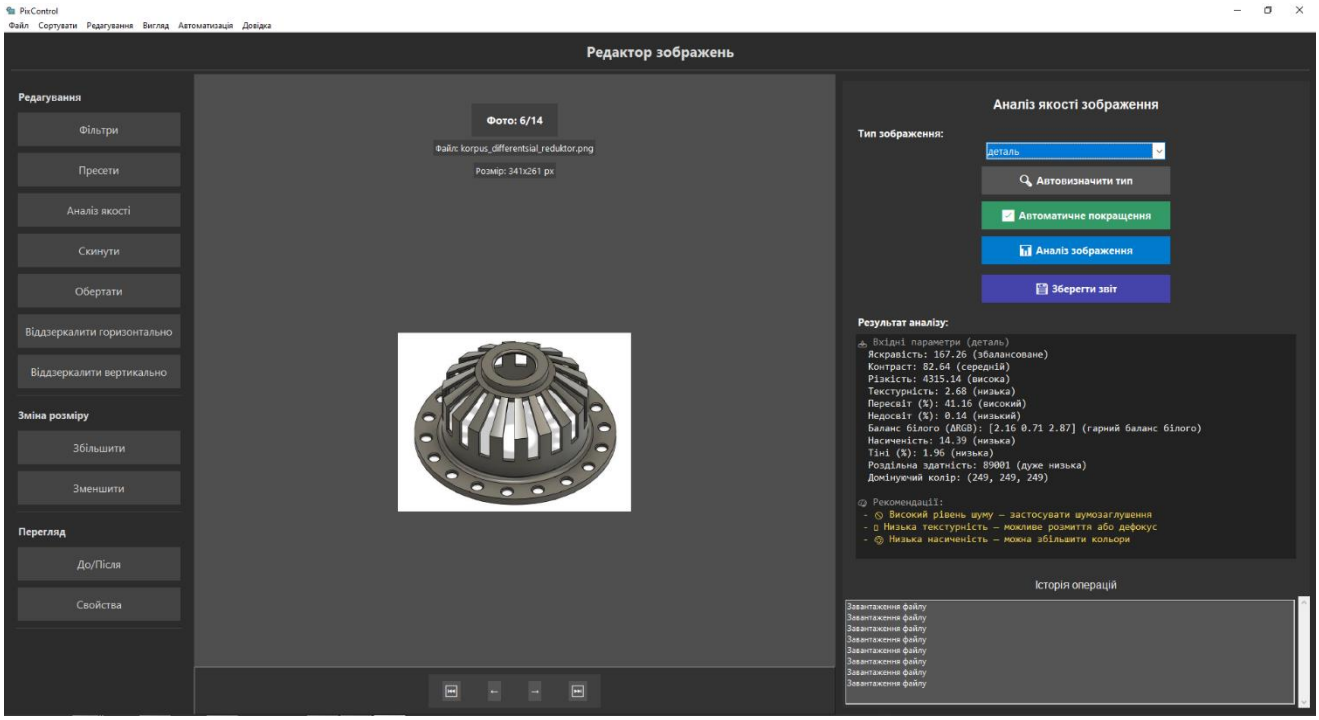

Рисунок 3.4 – Активована панель аналізу

Особливу увагу приділено обробці подій користувача. Всі функції виконуються асинхронно, або з використанням фонових потоків, що забезпечує стабільність програми навіть під час виконання складних великих операцій, таких як аналіз типу зображення або застосування інтенсивних фільтрів. Це дозволяє зберігати відгук інтерфейсу без зависання чи затримок.

Обробка запиту користувача на прикладі збереження обробленого зображення на рис. 3.5.

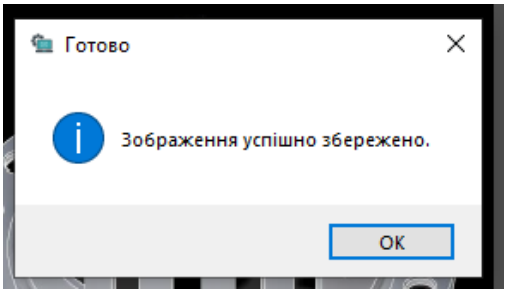

Рисунок 3.5 – Приклад збереження обробленого зображення

Оновлення даних у панелях реалізовано в динамічному режимі. Після кожної дії відбувається автоматичне оновлення відповідного інтерфейсного блоку без необхідності перезапуску програми чи окремих її частин. Така поведінка забезпечується внутрішнім механізмом обміну повідомленнями між модулями, організованим у межах об’єктно-орієнтованої архітектури.

Таким чином, реалізований графічний інтерфейс повністю відповідає вимогам сучасного прикладного програмного забезпечення: він поєднує функціональність, ергономічність, наочність і простоту використання. Його структура адаптована до потреб як технічно підготовлених фахівців, так і звичайних користувачів без попереднього досвіду в обробці зображень.

3.1.2 Модуль аналізу

Одна з основних складових системи автоматизації управління зображеннями є вбудовані модулі штучного інтелекту, що підвищують рівень

автоматизації та дозволяють зменшити вплив людського фактору при налаштуванні параметрів.

Модуль аналізу якості виконує функцію попереднього оцінювання вхідного зображення та надає користувачеві технічну інформацію, корисну для подальшого редагування. Його реалізовано у вигляді окремої панелі інтерфейсу, яка автоматично обчислює ключові метрики та виводить результати у табличному форматі з текстовими поясненнями та індикаторами візуального стану.

Архітектура побудована таким чином, що кожна функція аналізу працює автономно. Це забезпечує гнучкість системи, дозволяє легко адаптувати її під різні типи зображень та спрощує оновлення окремих частин без впливу на інші компоненти.

Панель аналізу зі всіма кнопками на рис. 3.6.

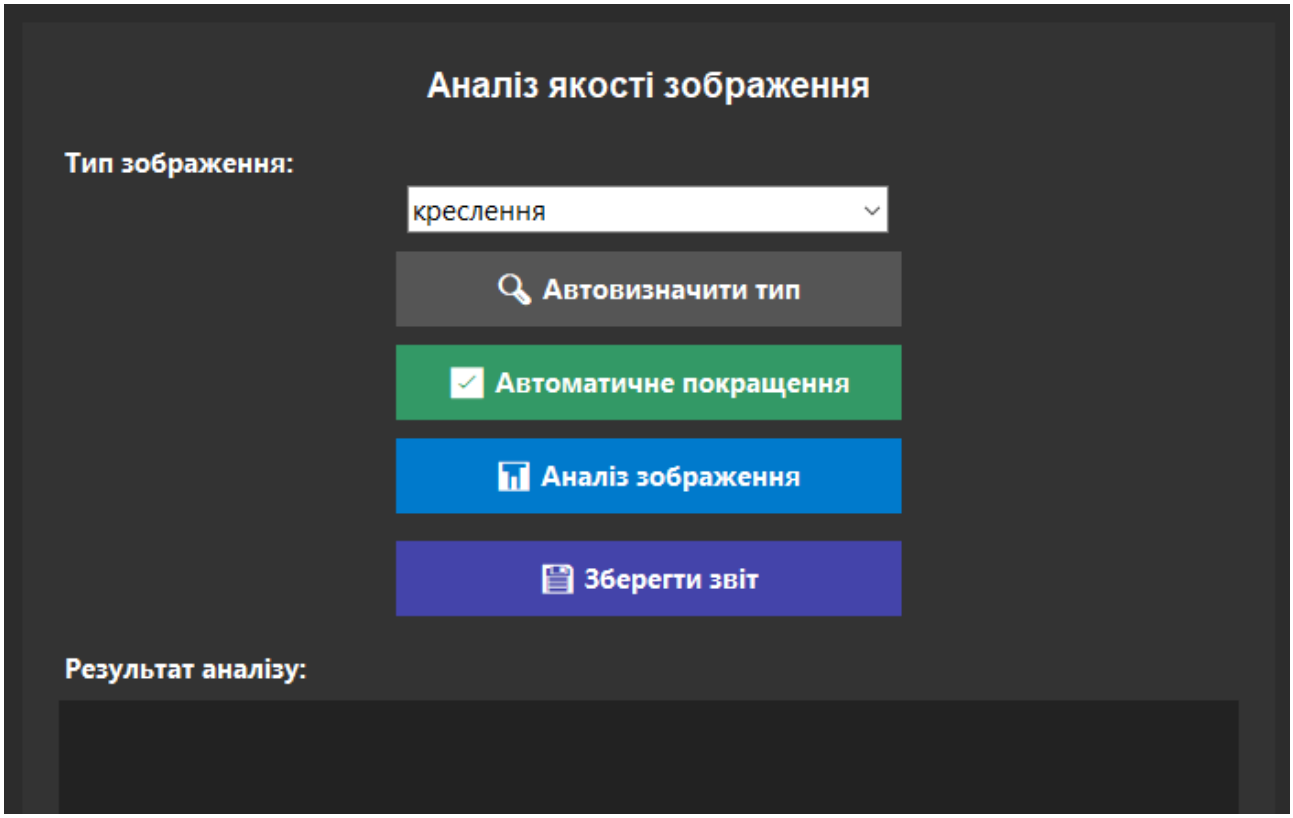

Рисунок 3.6 – Вигляд панелі аналізу якості зображення

Усі функції штучного інтелекту згруповано в окрему панель інтерфейсу, яка забезпечує доступ до результатів класифікації, рекомендацій і дозволяє запускати процес аналізу вручну або автоматично. Це впливає на інтуїтивність управління, адаптивність до різних сценаріїв використання та високу продуктивність при обробці великої кількості зображень.

До складу програмної системи входить модуль інтелектуального аналізу, який відповідає за автоматичне визначення типу зображення, оцінку його загального стану та формування рекомендацій щодо покращення. Його реалізація підвищує ступінь автономності системи, дозволяє адаптувати фільтри до змісту зображення та покращує взаємодію користувача з програмою.

У якості основи для класифікації зображень було обрано модель CLIP (Contrastive Language-Image Pretraining). Вона поєднує зображення з текстовими описами за допомогою спільного простору ознак. Завдяки цьому, система може автоматично розпізнавати зміст зображення без необхідності попереднього навчання на конкретному наборі класів (рис. 3.7).

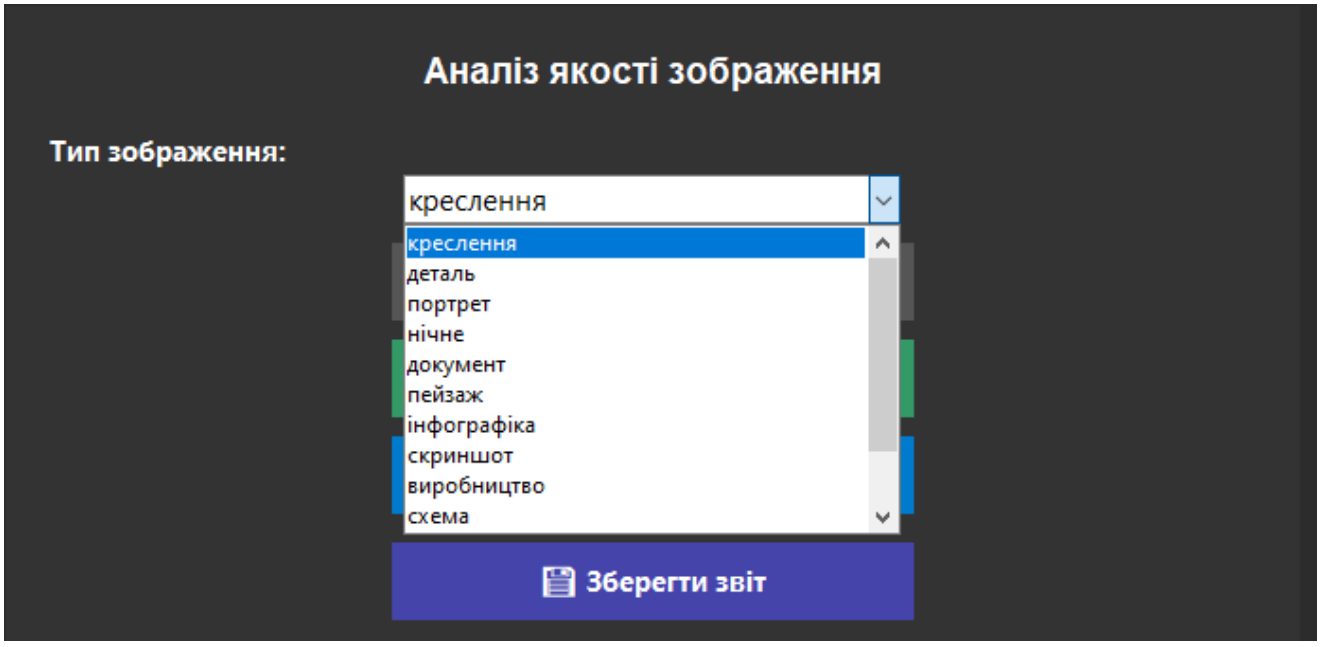

Рисунок 3.7 – Реалізовані типи зображення

В основі логіки лежить принцип порівняння вхідного зображення з заздалегідь підготовленим переліком текстових категорій, таких як портрет, пейзаж, документ, технічна деталь тощо. Це дозволяє визначити найбільш відповідну категорію та використовувати її як контекст для подальших операцій.

Алгоритм роботи класифікації типу зображень складається з деяких етапів. На початку зображення конвертується у формат, прийнятий для моделі. Далі CLIP одночасно обробляє вхідне зображення та перелік текстових описів, зазначених у програмі для різних типів зображень. Під час обробки модель обчислює схожість змісту зображення з описом, що заданий при технічній реалізації. Клас із найвищою ймовірністю вважається результатом класифікації (рис. 3.8).

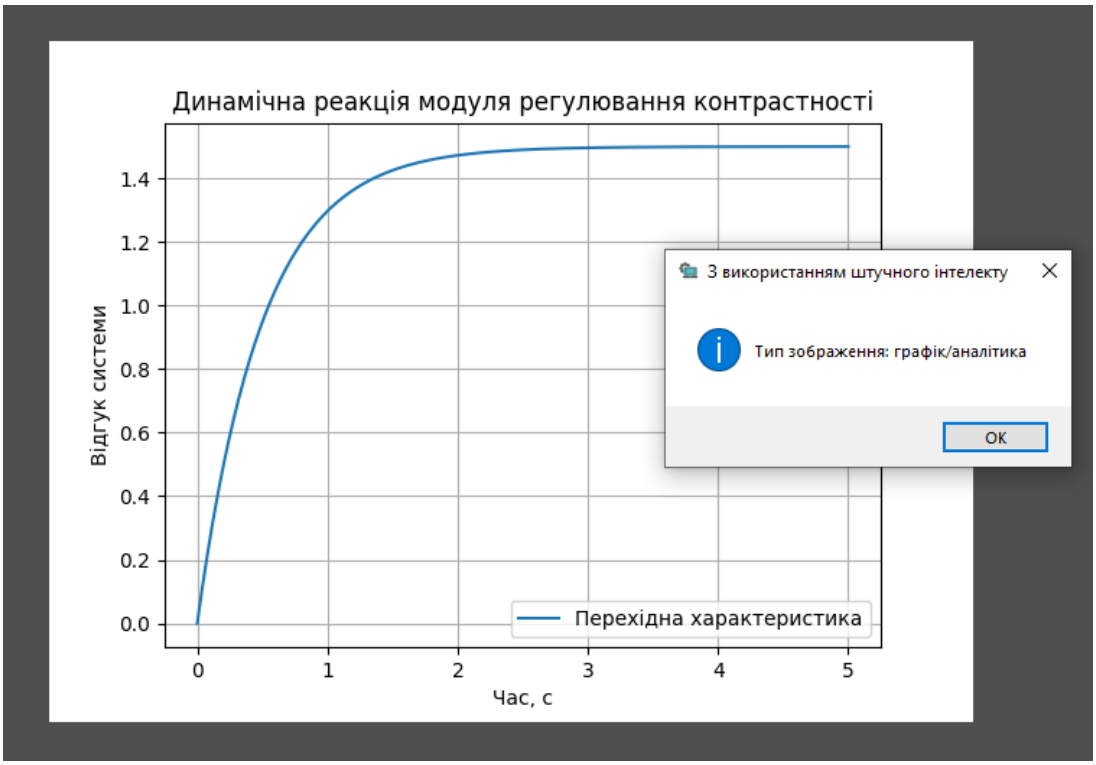

Рисунок 3.8 – Приклад роботи визначення типу зображень

Результати аналізу якості, згенеровані відповідним модулем, у поєднанні з інформацією про тип зображення, передаються до підсистеми генерації порад. Це важливий елемент, що реалізований у файлі ai_suggestions.py. Цей модуль

працює з принципом аналізу основних характеристик зображення та на основі саме цих даних система формує індивідуальний коментар та поради щодо оптимальних налаштувань зображення.

На рисунку 3.9 зображено процес автоматичного визначення типу зображення.

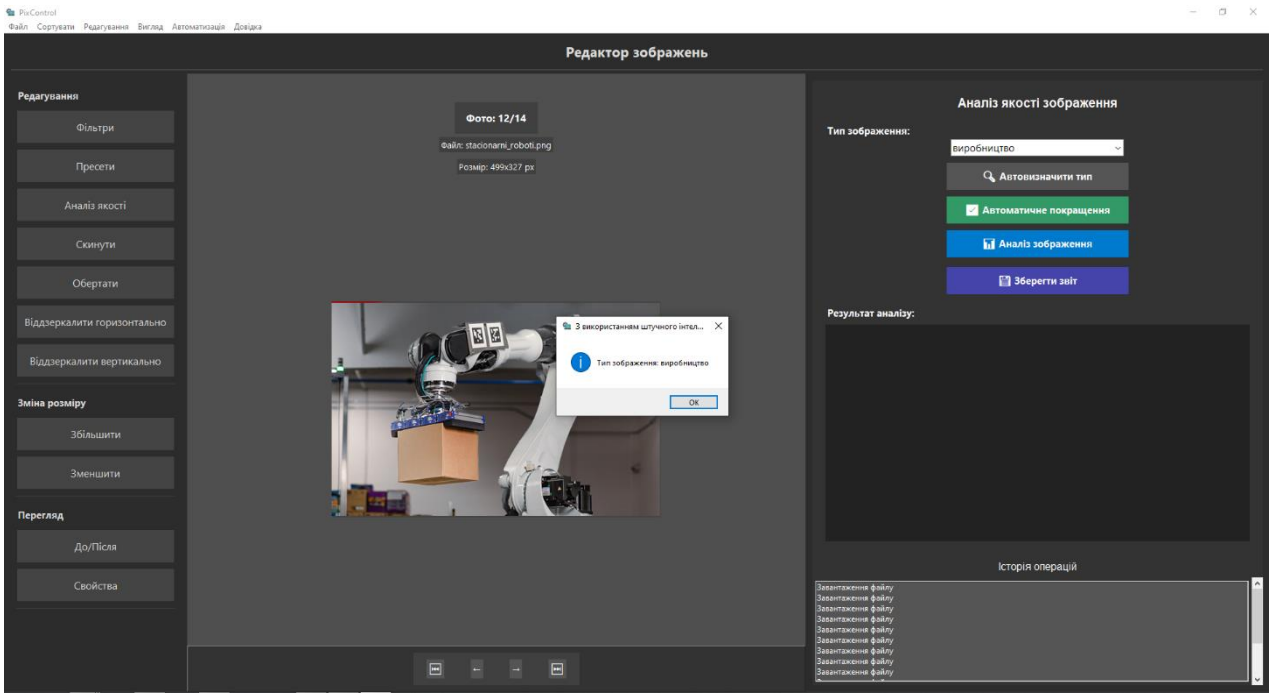

Рисунок 3.9 – Визначення типу зображення

Користувач завантажує фото та натискає «Автовизначити тип», після чого система коректно розпізнає, що зображено на знімку. За бажанням можна використовувати поради, що генеруються через «Аналіз зображення».

Система формулює короткий текстовий коментар, що містить оцінку технічного стану зображення та перелік рекомендованих дій. Наприклад, при зниженій різкості пропонується застосування відповідного фільтра підсилення, у разі виявлення низької контрастності відбудеться активація CLAHE або корекція яскравості.

Формування таких рекомендацій відбувається автоматично, тому користувачу не потрібно вручну аналізувати зображення чи шукати фільтр. У

разі того, що користувача не задовольняє ефект типових рекомендацій, він може здійснювати обробку зображення вручну.

Аналіз зображення разом із рекомендаціями на рис. 3.10.

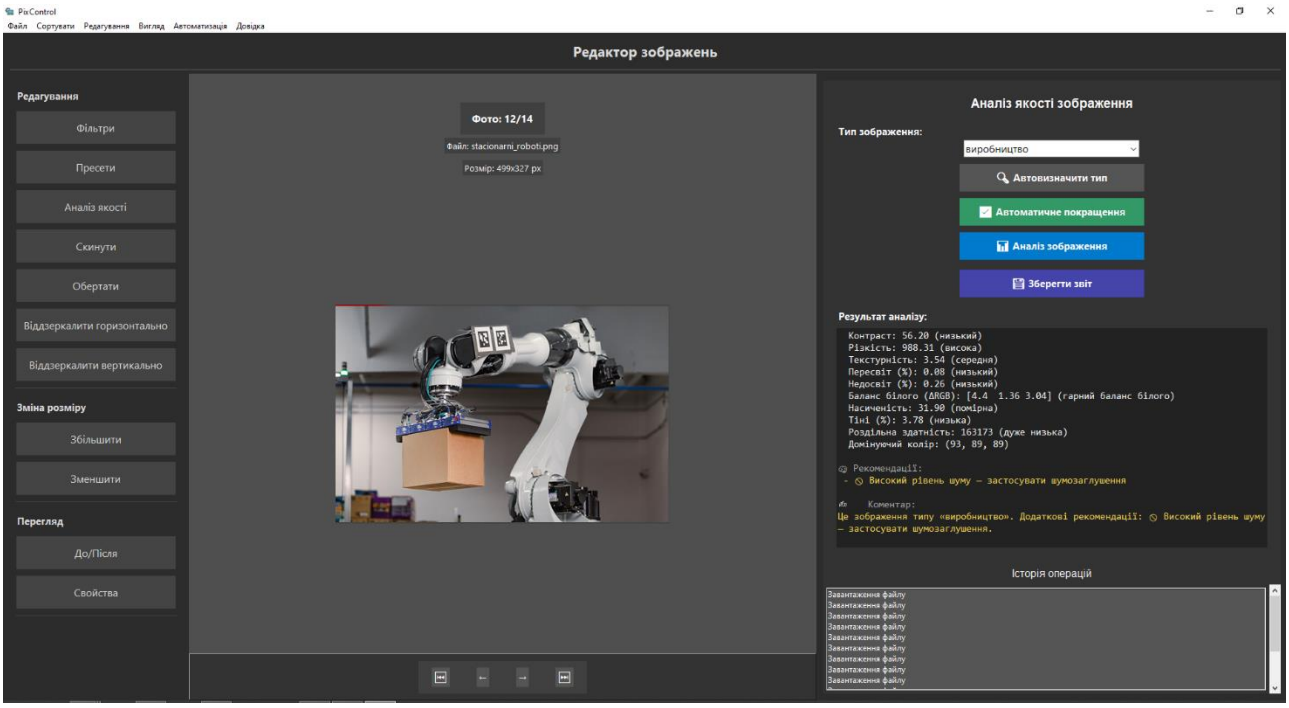

Рисунок 3.10 – Типові рекомендації у поєднанням з аналізом зображення

Під час обробки запиту застосовуються класичні методи цифрового аналізу. Зокрема, визначається різкість на основі оператора Лапласа, що дозволяє оцінити деталізацію зображення. Контрастність розраховується за статистичними характеристиками яскравості, що дає змогу визначити насиченість зображення. Для виявлення пересвітлених або затемнених ділянок аналізується розподіл пікселів з граничними значеннями. Баланс білого оцінюється шляхом порівняння середніх значень кольорових каналів, що дає змогу виявити відхилення від нейтрального тону. Крім того, виконується спектральний аналіз для виявлення шуму та розраховується глибина тіней, яка є критично важливою для технічних знімків і портретів.

Результати оцінювання доступні безпосередньо в інтерфейсі програми, а також можуть бути експортовані у вигляді текстового файлу, структури JSON

або автоматично згенерованого PDF-звіту. Користувач може повторно запускати аналіз після кожної зміни зображення, що дозволяє оперативно контролювати якість результату.

3.1.3 Модуль обробки

Модуль обробки – це один з головних компонентів системи автоматизації управління зображенням. Його функціонал реалізований в окремому файлі filters.py, що містить набір спеціалізованих функцій для покращення та перетворення різних типів зображень. Архітектурно реалізація базується на принципах інкапсуляції, простоти викликів, динамічного управління ресурсами та зручності масштабування.

Для забезпечення гнучкості та універсальності інтерфейсу всі фільтри викликаються через єдиний механізм: кожна функція зв’язується з відповідним повзунком через метод self.active_slider_function, а результат обробки застосовується через метод apply_and_update. Це дозволяє реалізувати незалежність від типу конкретного фільтра та значно спрощує внутрішню логіку керування змінами. На відміну від статичних рішень, слайдери для налаштування параметрів фільтрів не створюються заздалегідь. Вони формуються лише при виборі відповідної опції користувачем, що дозволяє зменшити навантаження на оперативну пам’ять і спростити підтримку інтерфейсу.

З огляду на те, що деякі фільтри можуть потребувати значних обчислювальних ресурсів, реалізовано механізм відкладеного запуску. Для цього використовується декоратор with_wait_message, який при перевищенні певного порогу часу автоматично виводить вікно очікування, що не блокує роботу інтерфейсу. Такий підхід дозволяє уникати зависання при обробці великих зображень.

Реалізація обробки включає в себе використання бібліотек Pillow та OpenCV. Перша обрана для базової роботи з зображеннями та кольорами, а друга

для реалізації складніших фільтрів, контурів, розмиття, покращення контрасту. Саме тому можна оптимально використовувати сильні сторони кожної бібліотеки. Кожна функція в модулі працює з об’єктом Image бібліотек або масивом NumPy, що дозволяє легко інтегрувати фільтри у загальний цикл обробки. Функції не змінюють оригінальне зображення, а повертають новий об’єкт, що забезпечує збереження початкового стану для подальшої роботи з історією змін.

Розглянемо процеси регулювання яскравості, що реалізовано функцією adjust_brightness. Вона призначена для коригування загальної яскравості зображення (рис. 3.11).

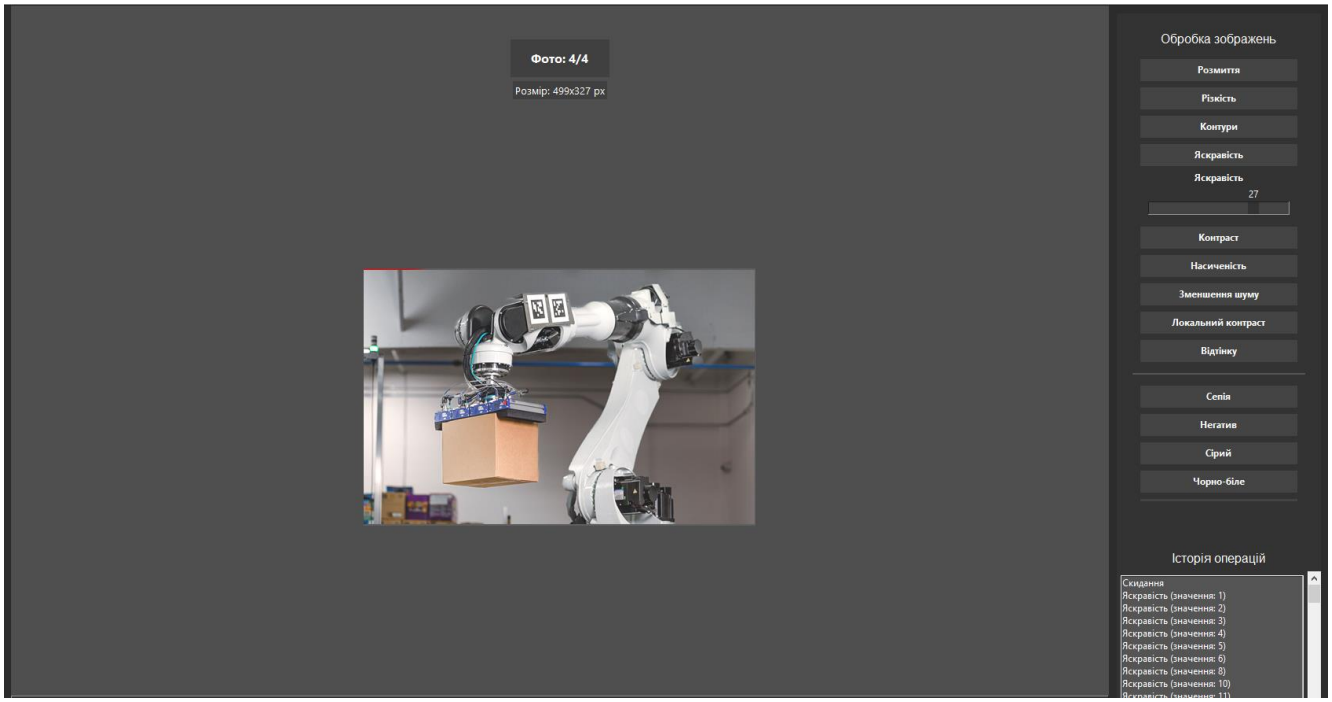

Рисунок 3.11 – Приклад налаштування яскравості

Приклад коду:

```
def adjust_brightness(image: Image.Image, intensity: int) -> Image.Image:
arr = np.array(image.convert("RGB"))
result = cv2.convertScaleAbs(arr, alpha=1.0, beta=intensity)
```

return Image.fromarray(result)

Функція яскравості реалізує лінійне зміщення інтенсивності пікселів за допомогою функції cv2.convertScaleAbs, яка забезпечує коректну обробку та автоматичне обмеження значень у діапазоні [0, 255]. Параметр intensity відповідає величині зміщення і передається як значення beta в алгоритм. При цьому коефіцієнт alpha дорівнює 1.0, тобто масштабування яскравості не змінюється, відбувається лише адитивна зміна.

Фільтр є особливо корисним у випадках недоекспонованих або пересвічених зображень, дозволяючи вручну налаштувати візуальний баланс світла без втрати контрастності. Працює швидко та стабільно завдяки використанню функцій OpenCV.

Розглянемо процеси регулювання контрасту через функцію adjust_contrast. Виконує лінійне масштабування яскравості пікселів відносно нуля, що дозволяє змінювати візуальний контраст зображення, а саме посилювати або зменшувати різницю між темними та світлими ділянками (рис. 3.12).

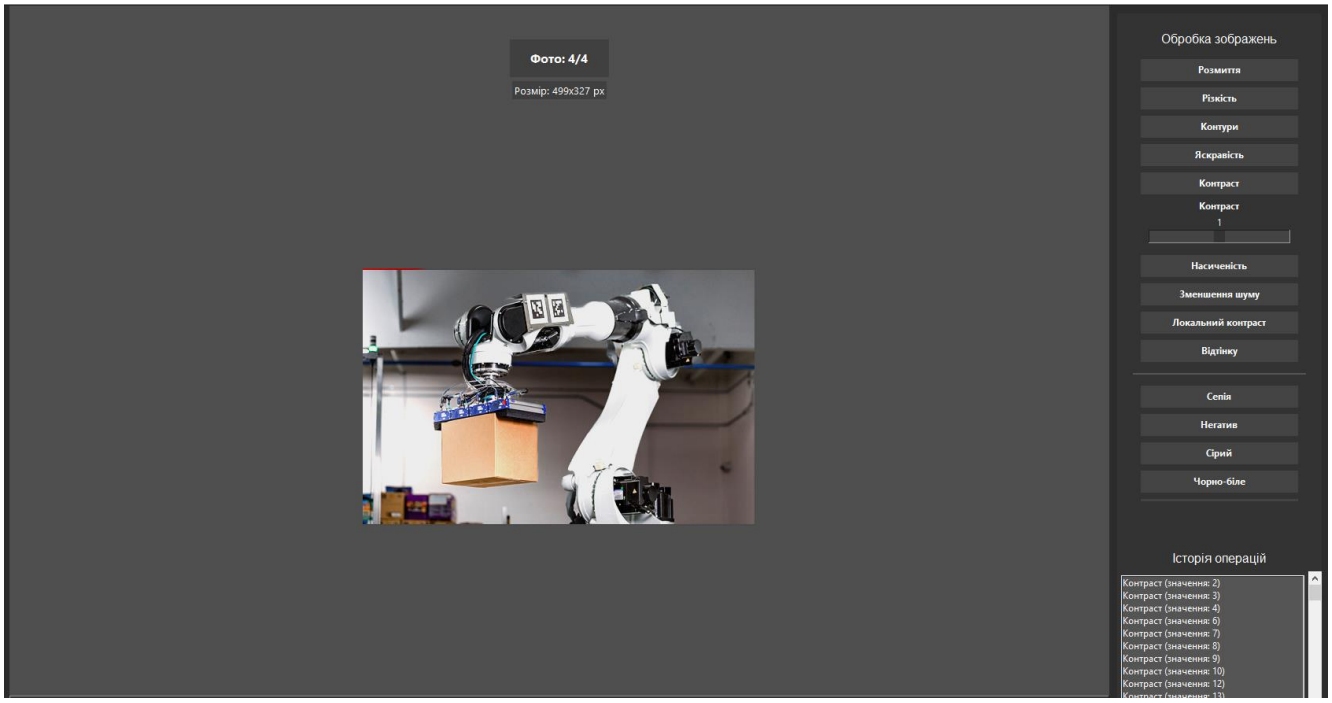

Рисунок 3.12 – Приклад налаштування контрасту

Приклад коду:

```
def adjust_contrast(image: Image.Image, intensity: int) -> Image.Image:
arr = np.array(image.convert("RGB"))
alpha = 1.0 + (intensity / 100.0)
result = cv2.convertScaleAbs(arr, alpha=alpha, beta=0)
return Image.fromarray(result)
```

Контрастність коригується шляхом зміни коефіцієнта alpha, що обчислюється за формулою 1.0 + intensity / 100.0. Це забезпечує гнучке керування від майже непомітних змін до сильного розширення діапазону яскравостей. Обробка здійснюється за допомогою функції cv2.convertScaleAbs, яка ефективно масштабує значення пікселів і автоматично виконує обмеження до допустимого діапазону [0, 255], що унеможливлює спотворення результату.

Фільтр застосовується у випадках, коли зображення виглядає "плоским" або тьмяним, а також для покращення розпізнаваності об'єктів у складному фоні.

Підвищення різкості реалізовано через apply_sharpen. Функція призначена для підвищення чіткості зображення шляхом посилення локальних контрастів на межах об'єктів. Застосування фільтра дозволяє зробити зображення більш деталізованим, покращуючи сприйняття текстур та контурів.

Приклад коду:

```
def apply_sharpen(image: Image.Image, intensity: int) -> Image.Image:
enhancer = ImageEnhance.Sharpness(image)
factor = 1.0 + intensity
return enhancer.enhance(factor)
```

Для реалізації використовувався вбудований засіб ImageEnhance.Sharpness з бібліотеки Pillow. Рівень посилення контролюється параметром intensity, який

трансформується у коефіцієнт factor за формулою 1.0 + intensity. Значення коефіцієнта більше за 1.0 підсилює різкість.

Фільтр є особливо корисним у випадках, коли зображення містить невиразні або злегка змазані деталі (рис. 3.13).

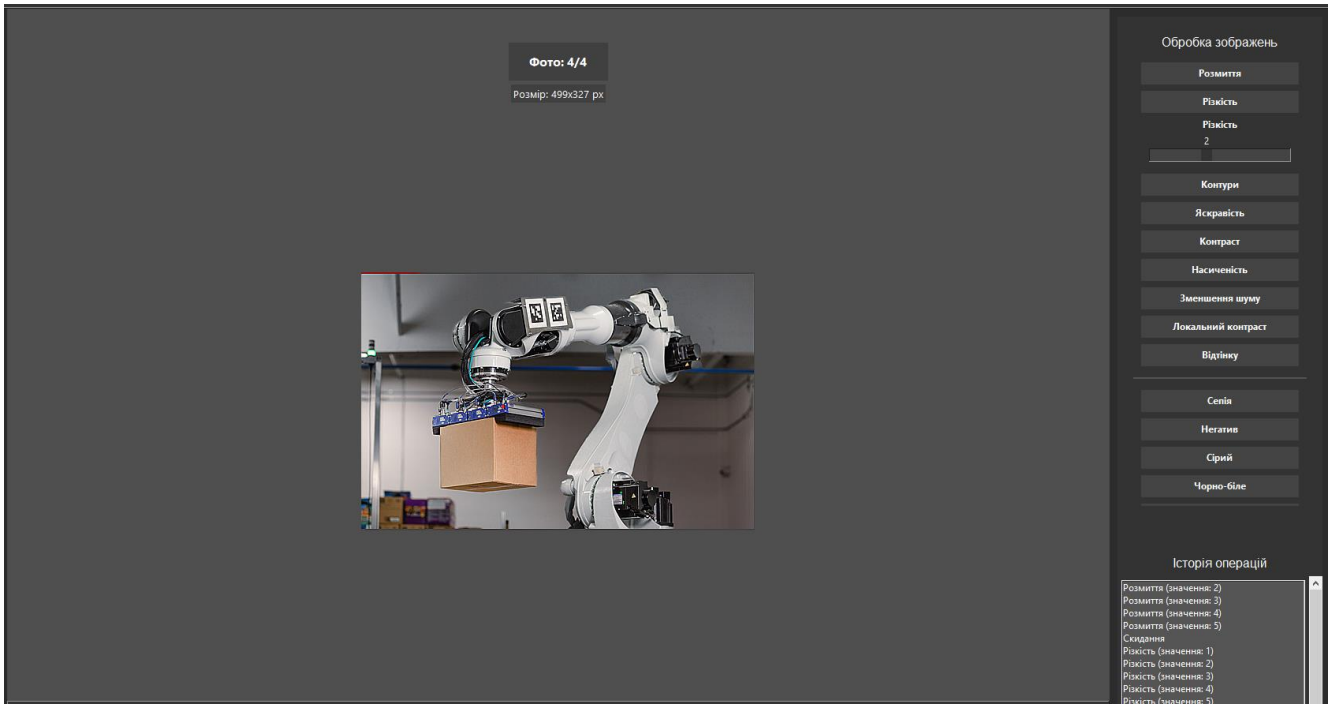

Рисунок 3.13 – Приклад налаштування різкості

Насиченість регулюється як adjust_saturation. Функція виконує зміну кольорової насиченості зображення без зміни яскравості чи контрасту.

Приклад коду:

```
def adjust_saturation(image: Image.Image, intensity: int) -> Image.Image:
arr = np.array(image.convert("RGB"))
hsv = cv2.cvtColor(arr, cv2.COLOR_RGB2HSV).astype(np.float32)
factor = 1 + intensity / 100
hsv[..., 1] = np.clip(hsv[..., 1] * factor, 0, 255)
result = cv2.cvtColor(hsv.astype(np.uint8), cv2.COLOR_HSV2RGB)
return Image.fromarray(result)
```

В цьому алгоритмі зображення спочатку перетворюється з простору RGB у HSV (відтінок, насиченість, яскравість), після чого масштабує компонент S (saturation) на заданий коефіцієнт factor, який визначається параметром intensity. Після зміни насиченості зображення знову конвертується у формат RGB для сумісності з іншими фільтрами програми. Також значення компонентів обрізаються в межах [0, 255].

Це потрібно, щоб уникнути спотворень. Завдяки зміні насиченості користувач може легко зробити зображення більш "живим", кольоровим або, навпаки, приглушити кольори. Метод працює швидко, стабільно і не вимагає складних обчислень (рис. 3.14).

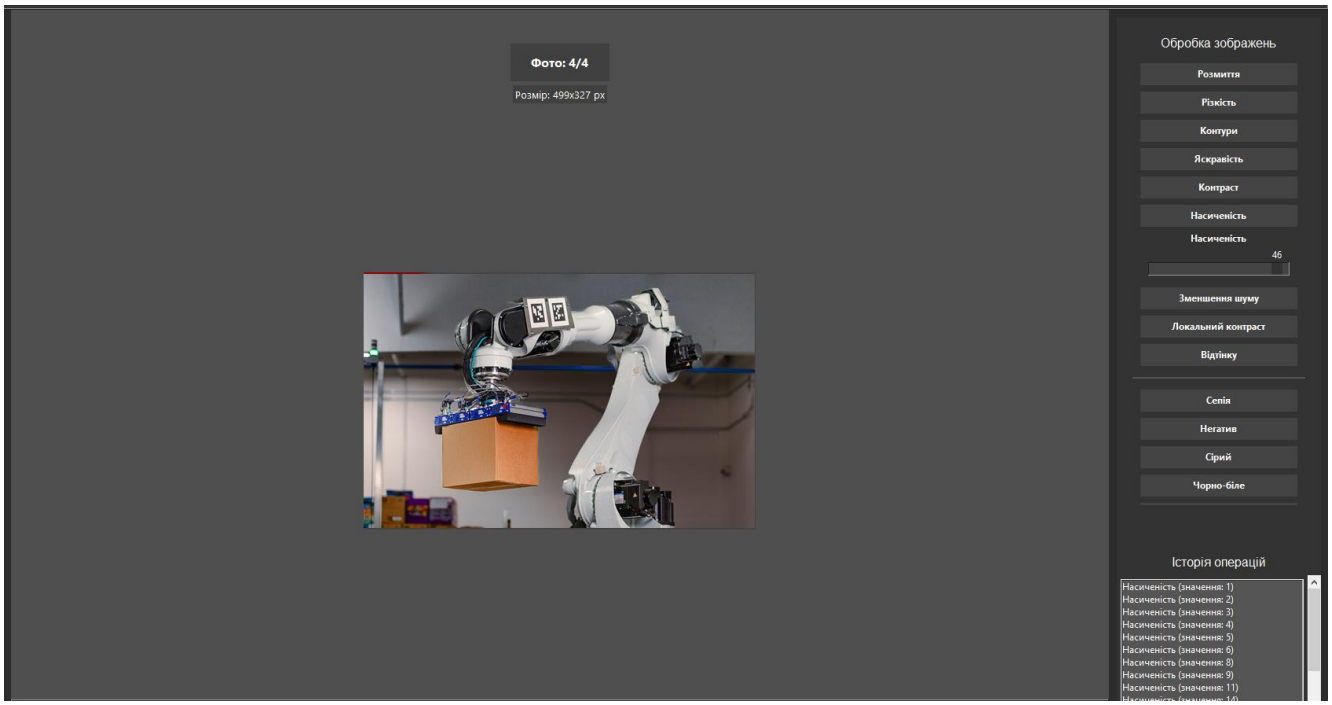

Рисунок 3.14 – Приклад налаштування насиченості

Регулювання градації сірого відбувається через apply_grayscale.

Функція перетворює зображення у відтінки сірого, що дозволяє спростити візуальну структуру та змінює кількість даних для подальшої оброки.

Приклад коду:

```
def apply_grayscale(image: Image.Image) -> Image.Image:
rgb = np.array(image.convert("RGB"), dtype=np.uint8)
gray = (0.299 * rgb[:, :, 0] + 0.587 * rgb[:, :, 1] + 0.114 * rgb[:, :,
2]).astype(np.uint8)
return Image.fromarray(gray, mode="L")
```

Використовує зважене середнє значень трьох каналів red, green, blue, Але важливо, що це не просто середнє арифметичне, а зелений колір найбільше впливає на сприйняття яскравості людським оком, тому його значення більше. Найменш впливає синій.

Далі отримане зображення повертається у відтінках сірого (режимі «L»), що зручно для подальшої обробки. Наприклад, якщо треба виділити контур або розпізнати текст, кольори не потрібні, бо там потрібна лише яскравість. Особливо корисно для використання в алгоритмах сегментації, контурного аналізу, розпізнавання символів або для попередньої обробки в комп’ютерному зорі.

Такий підхід дуже швидкий та ефективний, бо не потребує складних обчислень та швидко виконується на великій кількості зображень.(рис. 3.15).

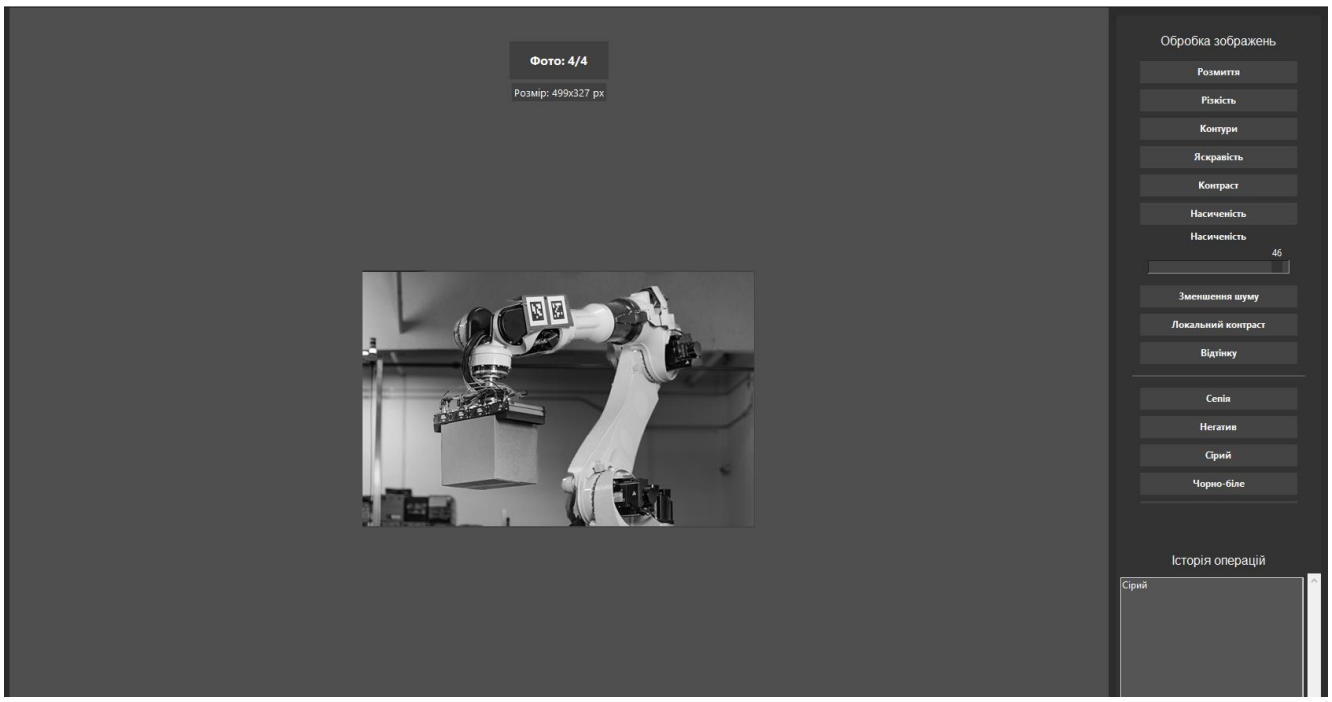

Рисунок 3.15 – Приклад фільтра «Сірий»

Бінаризація розроблена через apply_threshold. Функція виконує порогову обробку зображення, перетворюючи його у двокольорове, де кожен піксель стає або білим, або чорним. Це залежить від рівня яскравості (рис. 3.16).

Рисунок 3.16 – Приклад фільтра «Чорно-біле»

Приклад коду:

```
def apply_threshold(image: Image.Image, threshold: int) -> Image.Image:
gray = np.array(image.convert("L"), dtype=np.uint8)
binary = np.where(gray > threshold, 255, 0).astype(np.uint8)
return Image.fromarray(binary, mode="L")
```

Спочатку зображення переводиться у градації сірого, що дозволяє працювати лише з яскравістю без урахування кольору. Далі за допомогою NumPy для кожного пікселя перевіряється, чи перевищує його яскравість заданий поріг. Якщо перевищує, то піксель стає білим (255), якщо не перевищує, то він стає чорним (0). Результат формується як одноканальне зображення у режимі "L" (8-бітна глибина).

Бінаризація є важливою операцією у задачах виділення об’єктів, підготовці до морфологічних операцій, контролі наявності елементів на зображенні тощо.

Функція працює надзвичайно швидко та надає стабільні результати при правильно підібраному порозі.

Apply_denoise працює як зменшення шуму на цифровому зображенні. Функція реалізує фільтрацію зображення з метою зменшення шуму без значної втрати деталі.

У програмі використовується двосторонній фільтр, що дозволяє зберегти краї при розгладженні однородних ділянок

Приклад налаштування функції обробки зменшення шуму зображений на рис. 3.17.

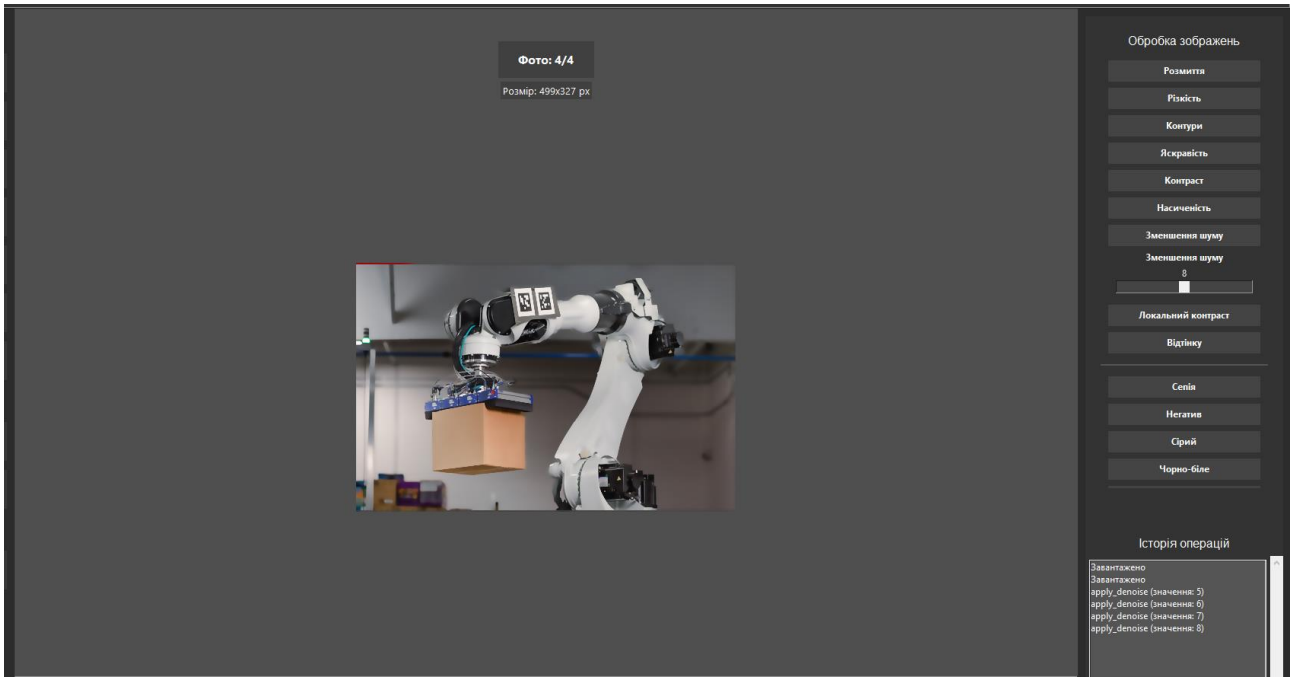

Рисунок 3.17 – Приклад налаштування зменшення шуму

Приклад коду:

```
def apply_denoise(image: Image.Image, intensity: int = 20) -> Image.Image:
arr = np.array(image.convert("RGB"))
result = cv2.bilateralFilter(arr, d=intensity, sigmaColor=75, sigmaSpace=75)
return Image.fromarray(result)
```

Фільтр cv2.bilateralFilter виконує згладжування з урахуванням sigmaColor, що є оцінкою схожесті кольорів пікселів, та sigmaSpace, що визначає близкість пікселів за розташуванням. Параметр intensity визначає розмір фільтра (діаметр області), у межах якої враховуються сусідні пікселі при згладжуванні.

Результатом є м’яке, чисте зображення без шумів і з природними переходами кольору.

Для виявлення та підсилення контурів застосовано метод apply_edges. Функція застосовує до зображення фільтр виявлення контурів із можливістю багаторазового підсилення результату. Це дозволяє виділити межі об’єктів на зображенні.

Приклад коду:

```
def apply_edges(image: Image.Image, intensity: int) -> Image.Image:
result = image
for _ in range(intensity):
result = result.filter(ImageFilter.FIND_EDGES)
return result
```

Функція використовує фільтр FIND_EDGES із бібліотеки Pillow, що реалізує один із класичних методів виявлення різких змін яскравості пікселів, тобто контурів. Ефект підсилюється циклічним повторенням, кількість регулюється параметром intensity. Тому завдяки такому підходу можна досягти плавного керування виразністю контурів. Також фільтр може бути корисним у попередній обробці зображень для подальшої сегментації (рис. 3.18).

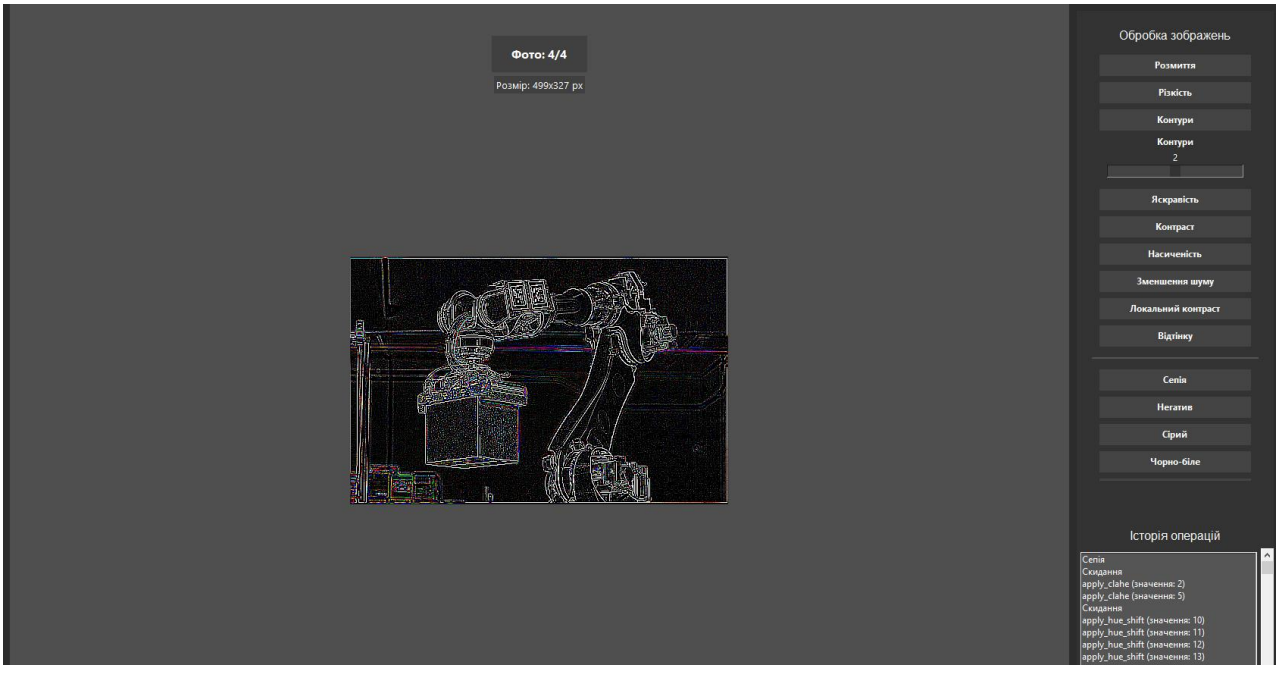

Рисунок 3.18 – Приклад виявлення контурів

Функція розмиття цифрового зображення створена як apply_blur. Алгоритм використовує фільтр Гауса – це спеціальний математичний спосіб згладжування зображень. Розмиває пікселі, беручи до уваги сусідні значення. (рис. 3.19).

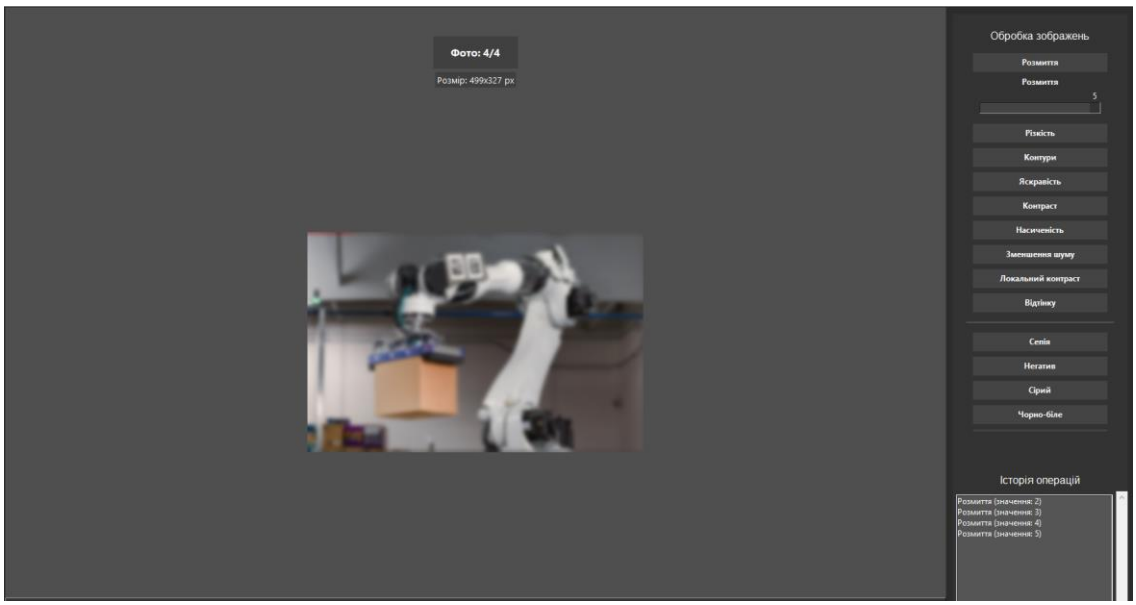

Рисунок 3.19 – Приклад налаштування розмиття

Форма самого фільтра (його ядра) задається гаусіанським розподілом, а сам фільтр працює тільки з непарними розмірами ядра. Розмір ядра обчислюється за формулою k = 2 · intensity + 1. Параметр intensity контролює ступінь розмиття. Ефект реалізований через cv2.GaussianBlur, яка застосовується до масиву пікселів. Тобто спочатку перетворюється у масив пікселів (NumPy), а потім знову повертається як нове зображення (Pillow).

Приклад коду функції:

```
def apply_blur(image: Image.Image, intensity: int) -> Image.Image:
  arr = np.array(image.convert("RGB"))
  k = 2 * intensity + 1
  blurred = cv2.GaussianBlur(arr, (k, k), 0)
  return Image.fromarray(blurred)
```

Негатив є функцією apply_negative. Це коли кожен колір перетворюється на свій протилежний за кольоровим спектром. Наприклад, чорний (0, 0, 0) стає білим (255, 255, 255), червоний – ціан, зелений – пурпуровим, синій – жовтим. Приклад коду:

```
def apply_negative(image: Image.Image) -> Image.Image:
arr = np.array(image.convert("RGB"), dtype=np.uint8)
inverted = 255 – arr
return Image.fromarray(inverted, mode="RGB")
```

Функція перетворює картинку у масив чисел, де кожен піксель описується трьома компонентами RGB. Інвертування кольорів корисне для візуального аналізу об'єктів на зображенні.(рис. 3.20).

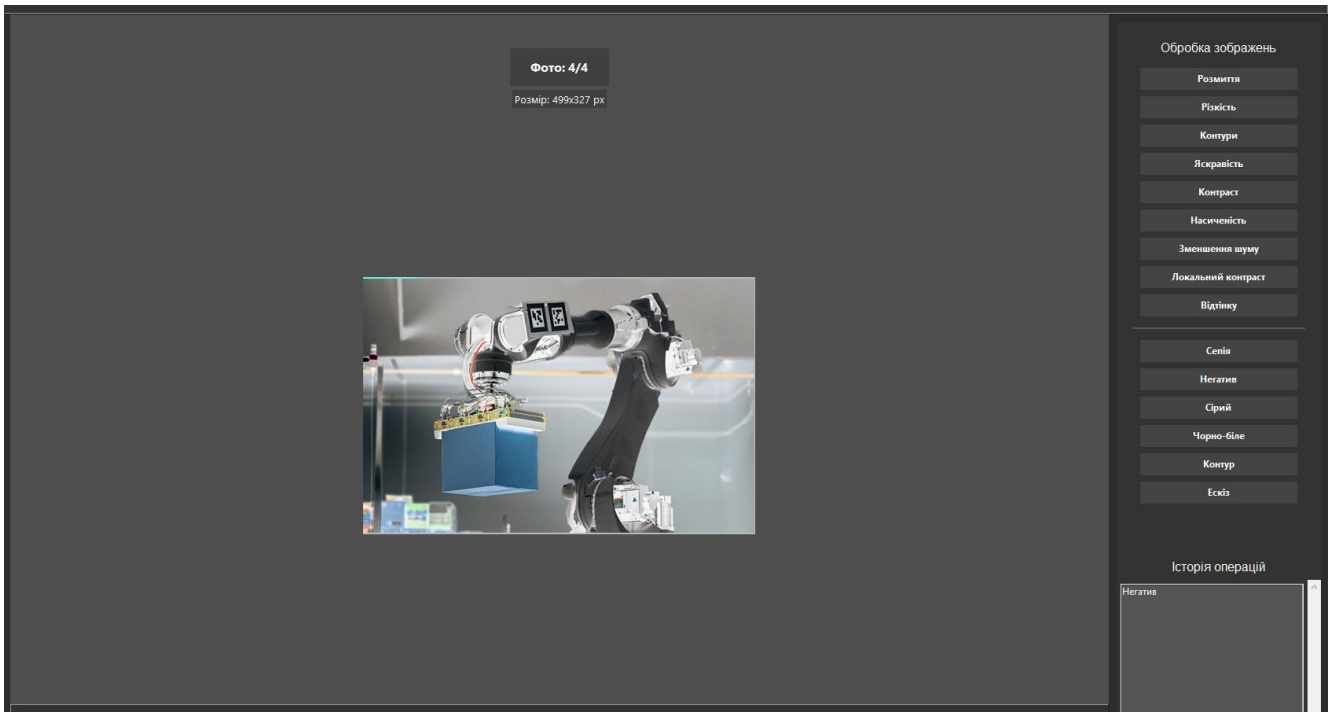

Рисунок 3.20 – Приклад фільтра «Негатив»

Розрахунок інверсії кольору через RGB наведений на рис. 3.21.

Колір	RGB	Інверсія	Інверсія RGB
Чорний	(0, 0, 0)	Білий	(255, 255, 255)
Білий	(255, 255, 255)	Чорний	(0, 0, 0)
Червоний	(255, 0, 0)	Ціан	(0, 255, 255)
Зелений	(0, 255, 0)	Пурпуровий	(255, 0, 255)
Синій	(0, 0, 255)	Жовтий	(255, 255, 0)
Сірий	(128, 128, 128)	Сірий	(127, 127, 127)
Темно-сірий	(64, 64, 64)	Світло-сірий	(191, 191, 191)
Світло-сірий	(192, 192, 192)	Темно-сірий	(63, 63, 63)

Рисунок 3.21 – Інверсії кольорів

Формула для інверсії кольору:

$$New\ color = (255 - R, 255 - G, 255 - B), \tag{3.1}$$

де R, G, B – вихідні компоненти кольору (значення від 0 до 255).

Сепія реалізована як apply_sepia. Функція застосовує до зображення ефект сепії, надаючи йому характерний теплий коричнево-жовтий відтінок, що імітує вигляд старих фотографій.

Для перетворення кожного пікселя використовується матриця 3×3, яка обчислює нові значення R, G та B через зважену суму початкових компонентів кольору. Це дозволяє сформувати ефект "пожовклого" зображення з приглушеними кольорами та посиленими теплими тонами. Завдяки використанню матричного множення ефект застосовується до всієї матриці зображення одночасно, що забезпечує високу продуктивність.

Після обробки значення обрізаються до діапазону [0, 255], щоб уникнути переповнення.

Цей фільтр дуже часто використовується сьогодні задля художньої обробки, але актуально його застосування і при відновлених старинних архівах.

Застосування сепії зображено на рис. 3.22.

Рисунок 3.22 – Приклад фільтра «Сепія»

Приклад коду:

```
def apply_sepia(image: Image.Image) -> Image.Image:
img = np.array(image.convert("RGB")).astype(np.float32)
sepia_matrix = np.array([
[0.393, 0.769, 0.189],
[0.349, 0.686, 0.168],
[0.272, 0.534, 0.131]
])
sepia_img = img @ sepia_matrix.T
sepia_img = np.clip(sepia_img, 0, 255).astype(np.uint8)
return Image.fromarray(sepia_img)
```

Локальний контраст реалізовано як apply_clahe. Функція покращує локальний контраст за допомогою алгоритму Contrast Limited Adaptive Histogram Equalization (CLAHE). На відміну від звичайного вирівнювання гістограми, цей метод працює локально на невеликих ділянках зображення, запобігаючи пересвітам та шумам.

Приклад коду:

```
def apply_clahe(image: Image.Image, clip_limit: float = 2.0) -> Image.Image:
import cv2
img = np.array(image.convert("RGB"))
lab = cv2.cvtColor(img, cv2.COLOR_RGB2LAB)
l, a, b = cv2.split(lab)
clahe = cv2.createCLAHE(clipLimit=clip_limit, tileGridSize=(8,8))
cl = clahe.apply(l)
merged = cv2.merge((cl, a, b))
result = cv2.cvtColor(merged, cv2.COLOR_LAB2RGB)
return Image.fromarray(result)
```

Спочатку зображення переводиться з RGB у колірний простір LAB, де L відповідає за яскравість, а A і B за кольоровість. Покращення застосовується лише до каналу L, після чого канали з’єднуються, та зображення повертається у формат RGB. Алгоритм CLAHE рівномірно розподіляє яскравість у межах локальних ділянок з обмеженням контрасту (через параметр clipLimit). Це дозволяє покращити деталізацію у затемнених або надто яскравих зонах без втрати природності. Застосування CLAHE є ефективним технічному візуальному контролі, а також у загальному покращенні якості фотографій, зроблених у складних умовах освітлення.

Ефект підвищення локального контрасту зображений на рис. 3.23.

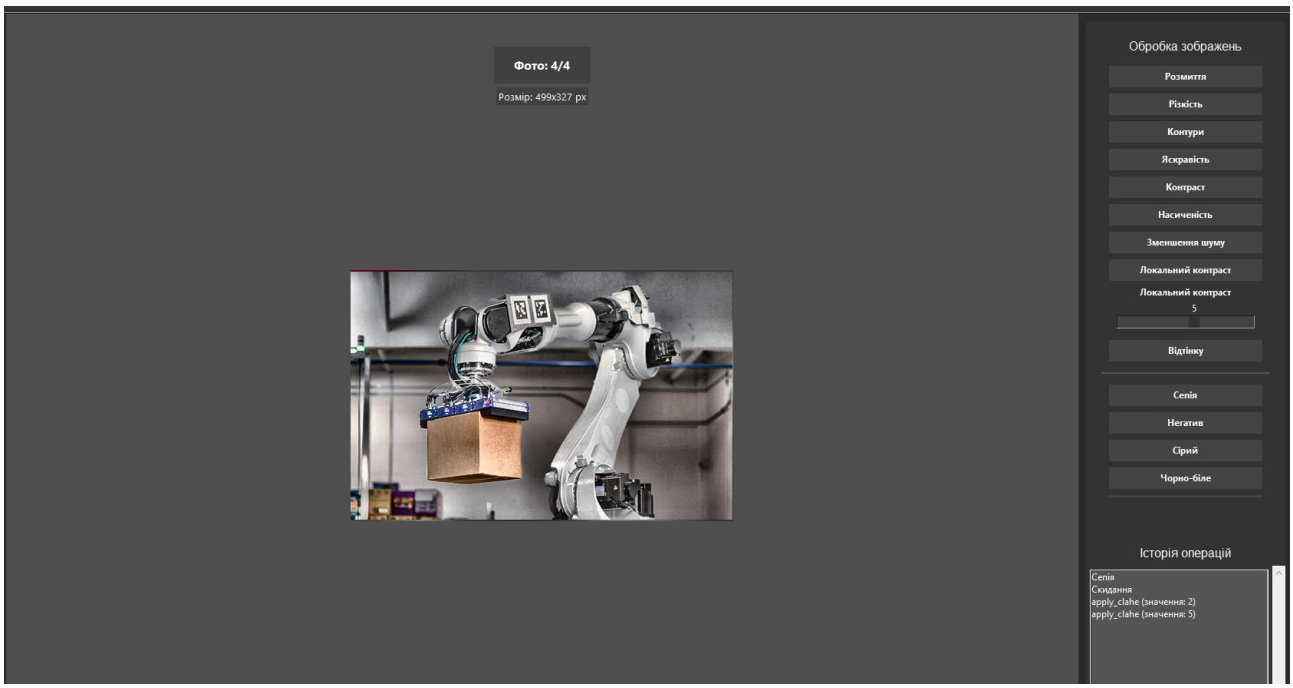

Рисунок 3.23 – Приклад налаштування локального контрасту

Зсув відтінку створений як функція apply_hue_shift. Така обробка зображення дозволяє змінювати колірний відтінок зображення без змін інших параметрів. Тому дає можливість фарбувати елементи в іншу кольору гаму штучно.

Приклад коду:

```
def apply_hue_shift(image: Image.Image, shift: int = 30) -> Image.Image:
arr = np.array(image.convert("RGB"))
hsv = cv2.cvtColor(arr, cv2.COLOR_RGB2HSV).astype(np.int32)
hsv[..., 0] = (hsv[..., 0] + shift) % 180
hsv = np.clip(hsv, 0, 255).astype(np.uint8)
result = cv2.cvtColor(hsv, cv2.COLOR_HSV2RGB)
return Image.fromarray(result)
```

Алгоритм виконує перетворення зображення з кольорової моделі RGB у HSV (відтінок, насиченість, яскравість), після чого модифікує лише компонент H (Hue), зміщуючи його значення на величину shift. Це дозволяє отримати той

самий візуальний образ, але з іншою домінантною палітрою. Оскільки колірний тон у HSV представляється у вигляді кута обертання по колу кольорів відбувається без втрати інформації. Після зміни зображення знову перетворюється у формат RGB для подальшого використання.

Зміна відтінку є корисною для генерації стилізованих варіантів зображень, а також у задачах доповнення даних для навчання моделей машинного навчання (рис. 3.24).

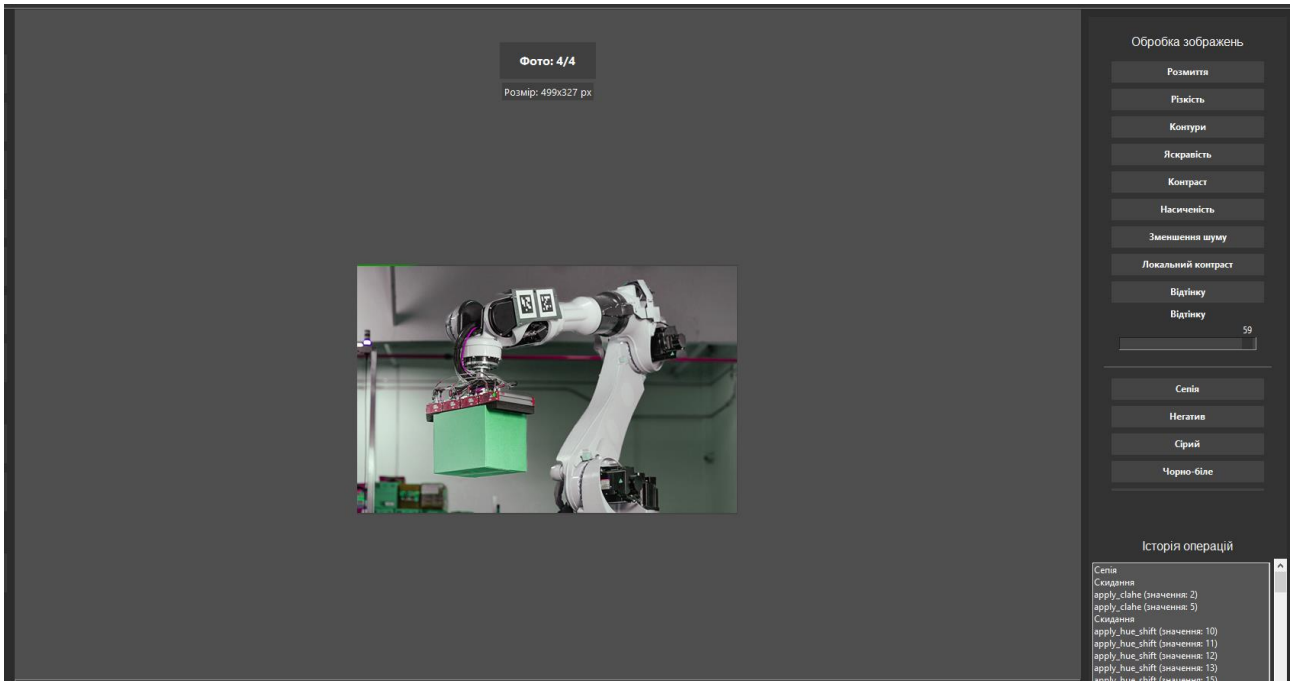

Рисунок 3.24 – Приклад налаштування відтінку

Постеризація представляє собою функцію apply_posterize. Застосовує ефект, зменшуючи кількість градацій кольору в зображенні. Це призводить до стилізованого вигляду з різкими переходами між тонами та зменшенням глибини кольору.

Робота цього ефекту зображена на рис. 3.25.

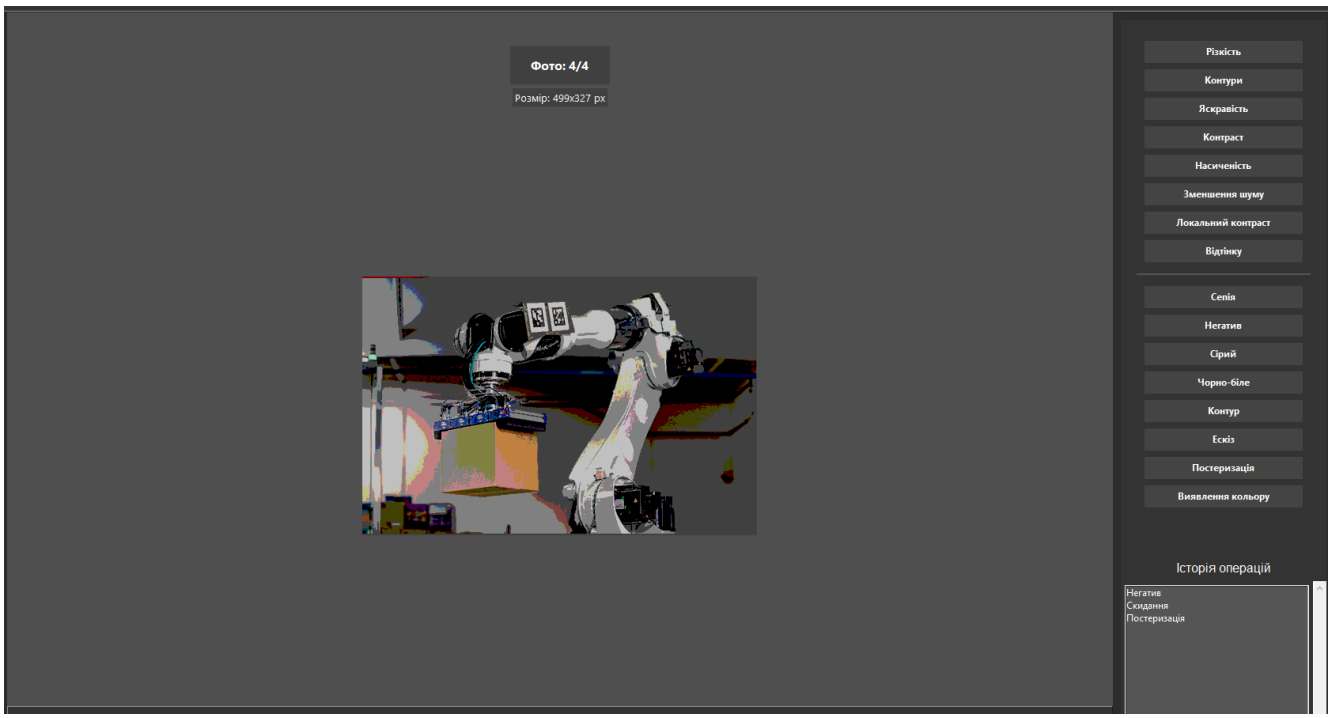

Рисунок 3.25 – Приклад фільтра «Постеризація»

Приклад коду:

```
def apply_posterize(image: Image.Image, levels: int = 4) -> Image.Image:
arr = np.array(image.convert("RGB"), dtype=np.uint8)
factor = 256 // levels
posterized = (arr // factor) * factor
return Image.fromarray(posterized)
```

Постеризація реалізується шляхом квантування кольорових значень: значення кожного пікселя округлюються до найближчого рівня, кратного обраному кроку. Це досягається простим цілочисельним діленням і множенням, що забезпечує високу швидкість обробки. Параметр levels визначає кількість рівнів градацій – чим менше рівнів, тим сильніший ефект. Наприклад, при levels = 2 зображення матиме лише два кольори на канал. Такий фільтр застосовується у графічному дизайні, арт-обробці, а також для підготовки зображень до задач сегментації, де важлива чітка межа між областями.

Ескіз створюється за допомогою apply_sketch. Імітує стилізоване перетворення зображення у вигляді створення його олівцем. При цьому зберігає основний контур та освітлення (рис. 3.26).

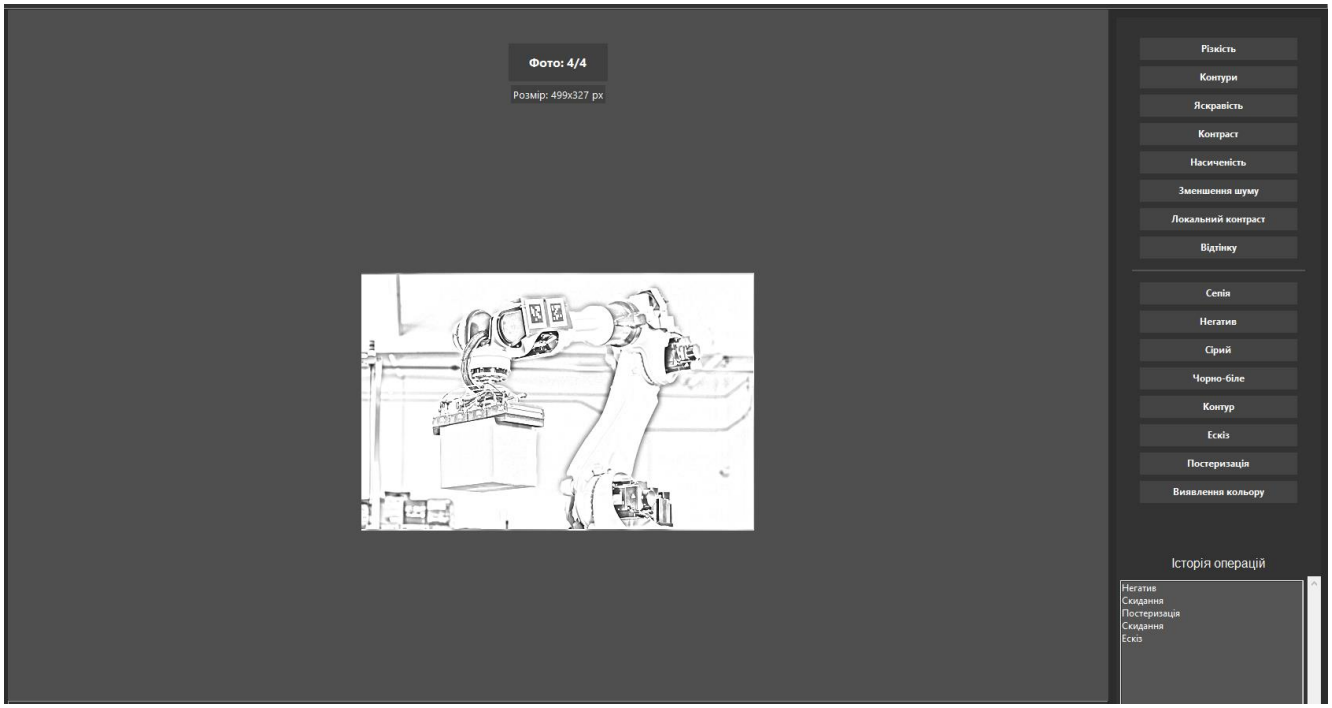

Рисунок 3.26 – Приклад фільтра «Ескіз»

Приклад коду:

```
def apply_sketch(image: Image.Image, intensity: int = 21) -> Image.Image:
img = np.array(image.convert("RGB"))
gray = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
inverted = 255 - gray
k = max(3, intensity | 1)
blurred = cv2.GaussianBlur(inverted, (k, k), 0)
sketch = cv2.divide(gray, 255 - blurred, scale=256)
return Image.fromarray(sketch)
```

Спочатку зображення перетворюється у відтінки сірого, після чого інвертується. Потім виконується згладжування методом Гауса з заданою інтенсивністю. На останньому кроці застосовується ділення початкового сірого зображення на розмиту версію інвертованого. Параметр intensity впливає на рівень розмиття та деталізацію ліній.

Такий підхід створює ефект тонких штрихів, подібний до малюнка олівцем. Фільтр застосовується для творчої обробки, візуалізації, підготовки контурів, а також у мобільних додатках для створення скетч-ефектів.

Виявлення контурів за алгоритмом Кенні реалізовано як apply_canny. Виконує детекцію контурів на зображенні за допомогою алгоритму, що базується на градієнтому аналізі яскравості.

Приклад коду:

```
def apply_canny(image: Image.Image, sigma=0.33) -> Image.Image:
img_cv = np.array(image.convert("L"))
v = np.median(img_cv)
lower = int(max(0, (1.0 - sigma) * v))
upper = int(min(255, (1.0 + sigma) * v))
edges = cv2.Canny(img_cv, lower, upper)
return Image.fromarray(edges)
```

Спочатку перетворюється у відтінки сірого, після чого визначається медіанне значення яскравості (v). Воно використовується для автоматичного розрахунку порогів lower і upper, які задають чутливість детектора до змін інтенсивності. Далі функція cv2.Canny виконує декілька етапів: згладжування зображення, знаходить зміни яскравості (градієнти), видаляє точки, які не належать до контурів та визначає самі контури за допомогою порогів.

Результатом є чітке зображення з тонкими білими лініями на чорному фоні, що представляють виявлені контури. Такий підхід широко

використовується в комп’ютерному зорі, розпізнаванні об’єктів, сегментації зображень і технічному контролі. Завдяки автоматичному підбору порогів за допомогою sigma, алгоритм адаптується до різних типів зображень без потреби ручного налаштування (рис. 3.27).

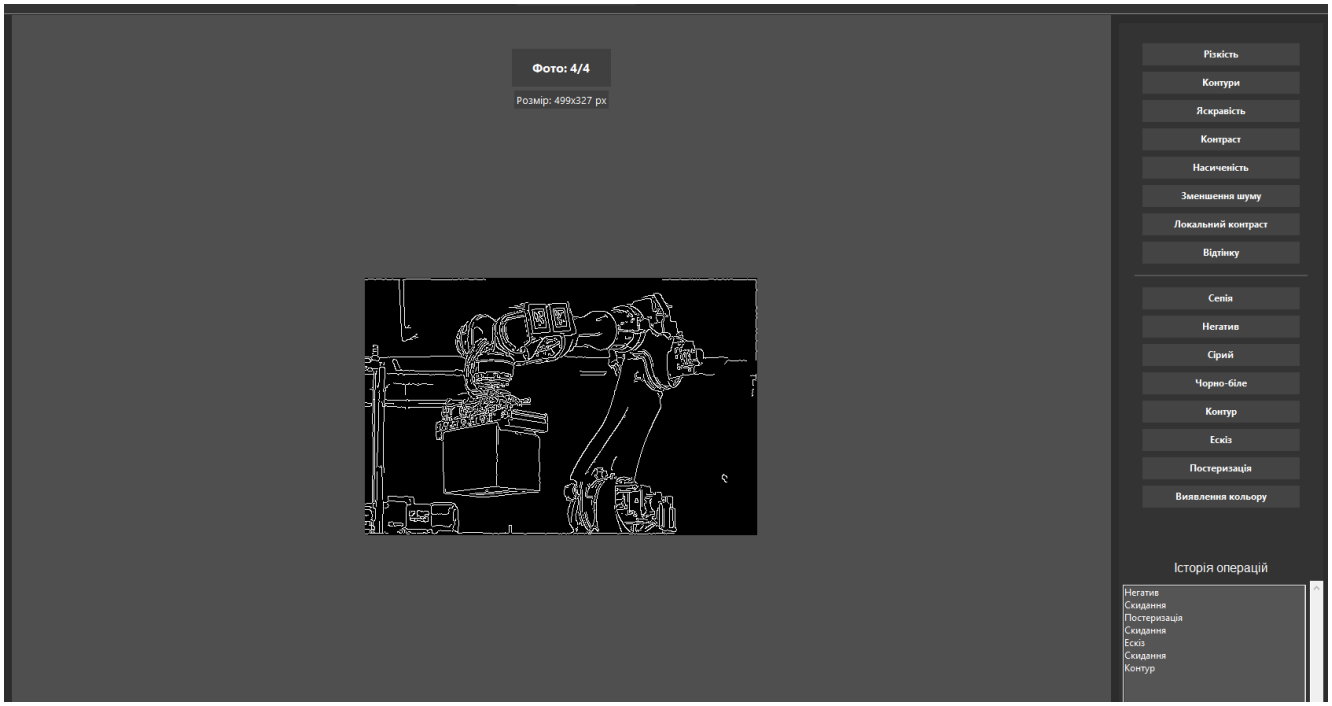

Рисунок 3.27 – Приклад виявлення контурів

Виділення заданого кольору працює через apply_color_detection. Функція дозволяє виділити на зображенні об’єкти певного кольору, а все інше «сховати». Це дозволяє сконцентрувати увагу на об'єктах заданого кольорового тону.

Приклад коду:

```
def apply_color_detection(image: Image.Image, color: str = "yellow") ->
Image.Image:
img = np.array(image.convert("RGB"))
hsv = cv2.cvtColor(img, cv2.COLOR_RGB2HSV)
color_ranges = {
"червоний": [
```

```
 (np.array([0, 100, 100]), np.array([10, 255, 255])),
 (np.array([160, 100, 100]), np.array([180, 255, 255]))
],
"помаранчевий": [(np.array([10, 100, 100]), np.array([20, 255, 255]))],
"жовтий": [(np.array([20, 100, 100]), np.array([30, 255, 255]))],
"зелений": [(np.array([40, 70, 70]), np.array([80, 255, 255]))],
"синій": [(np.array([100, 150, 0]), np.array([140, 255, 255]))],
"фіолетовий": [(np.array([130, 50, 50]), np.array([160, 255, 255]))],
"рожевий": [(np.array([160, 50, 50]), np.array([170, 255, 255]))],
"коричневий": [(np.array([10, 100, 20]), np.array([20, 255, 200]))]
}

ranges = color_ranges.get(color.lower(), [])
if not ranges:
return image
mask = cv2.inRange(hsv, ranges[0][0], ranges[0][1])
for r in ranges[1:]:
mask = cv2.bitwise_or(mask, cv2.inRange(hsv, r[0], r[1]))
result = cv2.bitwise_and(img, img, mask=mask)
return Image.fromarray(result)
```

Алгоритм починається з того, що зображення переводиться з RGB у HSV-формат. У цьому просторі кольори виділяються набагато чіткіше, оскільки вони представлені у вигляді відтінку, насиченості та яскравості. Далі для вибраного кольору створюється чорно-біле зображення, на якому білі пікселі відповідають потрібному кольору, а чорні за всі інші. Для кожного кольору задані один або кілька інтервалів значень HSV, щоб врахувати різні відтінки чи освітлення.

Якщо інтервалів кілька, вони об’єднуються через cv2.bitwise_or, щоб отримати єдину маску. Також застосовується cv2.bitwise_and, щоб із оригінального зображення залишити лише потрібний колір, а всі інші прибрати.

Пошук відтінку корисний у візуалізації, системах контролю якості, виділенні об’єктів конкретного кольору. (рис. 3.28).

Рисунок 3.28 – Приклад виявлення кольору синій

Для підвищення ефективності роботи користувача та автоматизації повторюваних операцій в системі реалізовано функціональність збереження і повторного застосування налаштувань обробки реалізовано механізм пресетів. Цей механізм дозволяє зберегти поточні параметри застосованих фільтрів як шаблон, який згодом може бути використаний для обробки інших зображень зі схожими характеристиками.

Функціональність пресетів реалізована в окремому модулі PresetsPanel, який надає засоби для створення, редагування, застосування та видалення шаблонів обробки.

Пресет містить набір ключ-значення у форматі json. Тобто він представляється у вигляді структурованого словника (dict), де ключами є назви фільтрів, а значеннями – їх параметри інтенсивності.

Приклад коду:

```
{
  "brightness": -10,
  "contrast": 25,
  "sharpen": 2,
  "denoise": 1
}
```

При збереженні система автоматично фіксує активні фільтри, що були застосовані до base_image, та їх налаштування. Дані записуються у *.json-файл, що зберігається у внутрішній структурі проєкту або у користувацькому каталозі. Користувач може у будь-який момент застосувати збережений шаблон до іншого зображення, після чого всі зазначені фільтри автоматично накладаються в тій самій послідовності з тими ж параметрами. Така технологія обробки часто використовується в умовах серійної обробки зображень, де потрібна стандартизація результатів.

Інтерфейс модуля побудований у вигляді списку на правій панелі, де кожен пресет представлений кнопкою. Натискання на неї викликає відповідну обробку. Передбачена також можливість перейменування та видалення існуючих шаблонів.

При застосуванні шаблону кожен активний фільтр викликається з відповідним значенням інтенсивності, аналогічно ручному режиму. Це дозволяє відтворити однакову обробку одразу на багатьох зображеннях. Важливо, зокрема, у задачах попередньої підготовки даних або у виробничих середовищах.

Ця функція працює як і з одним зображенням, так і до цілого масиву файлів у одній папці.

Вигляд панелі пресетів на рис. 3.29.

Рисунок 3.29 – Панель пресетів

Збереження відбувається у внутрішньому списку застосунку, з можливістю експорту у файл (формати *.json або *.txt можуть бути додані при потребі). Це дозволяє ділитися шаблонами між різними користувачами або створювати бібліотеки налаштувань.

Користувач може швидко змінити активний шаблон – система автоматично перераховує base_image та оновлює current_image у вікні перегляду. Це реалізовано без необхідності оновлення всієї програми чи перезавантаження зображення.

Таким чином, механізм пресетів є ефективним засобом для підвищення швидкості роботи, стандартизації обробки та зменшення впливу людського фактору в типових задачах.

3.1.4 Підтримка автоматизації обробки зображень

Задля мінімізації витрат часу на управління великого обсягу однотипних файлів у системі реалізовано режим пакетної обробки. Цей режим дозволяє одночасно обробляти не окремий файл, а цілу масив зображень, розташованих у визначеній папці. Таким чином, користувач отримує можливість виконувати однакові дії обробки для великої кількості зображень без необхідності відкривати та налаштовувати кожне зображення вручну (рис. 3.30).

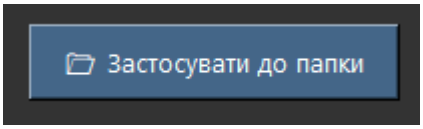

Рисунок 3.30 – Кнопка застосування пресетів до всіх файлів у папці

При використанні пакетного режиму користувач обирає папку з файлами, а далі застосовує до всієї папки однаковий набір налаштувань або пресетів. Така технологія широко використовується при потребі стандартної обробки для певних файлів. Особливо корисно для контролю якості продукції, обробці архівів документів. Це можуть бути фото схожих деталей, скановані файли, допомога зі звітами тощо (рис. 3.31).

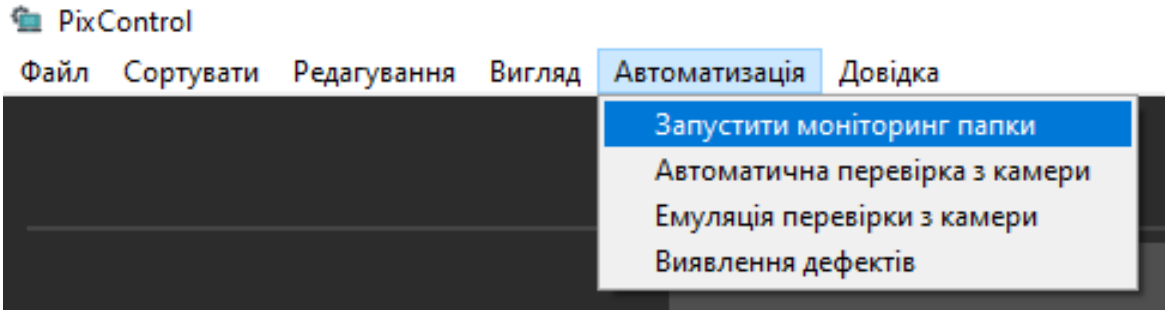

Рисунок 3.31 – Моніторинг обраної папки

У розробленій системі реалізовано механізм автоматичного відслідковування нових зображень, що потрапляють у визначену папку /input.

Для організації постійного відслідковування нових зображень передбачено використання модуля folder_watcher.py. Його логіка дозволяє реалізувати у майбутньому режим спостерігача, що відкриває нові можливості автоматичного застосування обробки до всіх нових файлів, які потрапляють у вхідну папку. Тобто це безперервна робота у режимі реального часу, що надзвичайно важливо для виробничих або адміністративних процесів (рис. 3.32).

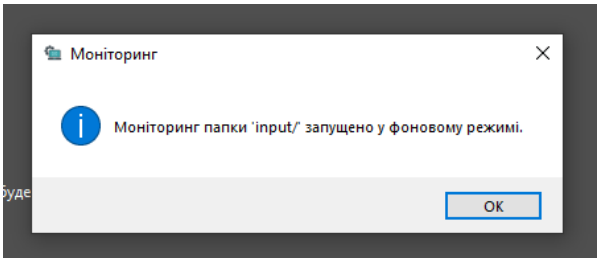

Рисунок 3.32 – Повідомлення про запуск моніторингу у фоновому режимі

Така функція мінімізує потребу в ручному втручанні, а саме – людський фактор. Користувач налаштовує умови обробки один раз, а далі система виконує їх автоматично.

Технологія особливо корисна на виробництві при отриманні фото з інспекційних камер у цеху. Система може відразу застосовувати потрібні налаштування параметрів зображення, автоматично сортує та зберігає у певну папку (рис. 3.33).

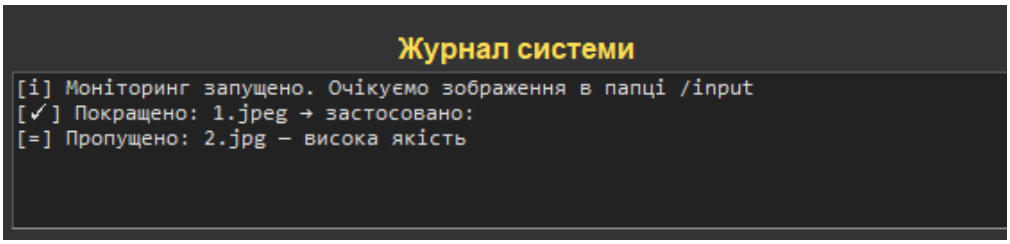

Рисунок 3.33 – Журнал системи

У журналі системі автоматично відображаються всі події з зображенням. Він включає в себе інформацію щодо початку моніторингу та очікування файлів

у папці. Підтверджує успішну обробку зображення або відмічає, що зображення залишається у такому ж вигляді через високу якість оригіналу.

Весь функціонал реалізовано у модулі folder_watcher.py. Він використовує бібліотеку watchdog, яка надає засоби для спостереження за змінами у файловій системі.

Приклад коду:

```
class ImageHandler(FileSystemEventHandler):
def on_created(self, event):
if event.is_directory:
return
filename = event.src_path
print(f"[✓] Новий файл: {filename}")
process_image(filename)

def start_monitoring(path="/input"):
    event_handler = ImageHandler()
    observer = Observer()
    observer.schedule(event_handler, path, recursive=False)
    observer.start()
    print("[i] Моніторинг запущено. Очікуємо зображення в папці:", path)
    try:
        while True:
            time.sleep(1)
    except KeyboardInterrupt:
        observer.stop()
    observer.join()
```

Створюється клас ImageHandler, що обробляє події файлової системи. А саме в цьому випадку – це створення нових файлів у папці.

При додаванні нового файлу спрацьовує метод on_created, де викликається функція process_image(filename). Це основний обробник із модуля обробки.

Функція start_monitoring ініціалізує моніторинг і запускає його у фоновому режимі.

3.1.5 Структура проєкту

Модуль main.py основний елемент програми. Цей файл відповідає за базову логіку, саме тут реалізовано запуск застосунку, а також поєднано всі необхідні модулі в єдину систему.

Тут здійснюється підключення необхідних бібліотек і зовнішніх модулів, зокрема, бібліотек для роботи з графікою (tkinter, Pillow, OpenCV), панелей керування (processing_panel, analysis_panel, history_panel), а також утиліт для обробки зображень і керування їх станом (image_state_manager, image_utils).

У межах цього модуля знаходиться головний клас застосунку – ImageEditorApp, який включає всі ключові методи й виклики для стабільної роботи програми (рис. 3.34).

```
# ---------- КЛАС ЗАСТОСУНКУ ----------

class ImageEditorApp:  1 usage
    def __init__(self, root):
        self.root = root
        self.root.title("PixControl")

        icon_path = os.path.join(os.path.dirname(__file__), "icon.ico")
        try:
            self.root.iconbitmap(icon_path)

        except Exception as e:
            print("Не вдалося встановити іконку:", e)
```

Рисунок 3.34 – Оголошення головного класу в main.py

Клас відповідає за налаштування головного вікна, а також за підключення панелей. До цього класу підключені панель обробки, аналізу, історії та інші. Крім того, у цьому класі реалізовано управління станами зображення та підготовку звітів або збереження результатів обробки.

3.2 Тестування та перевірка системи автоматизації управління зображеннями

З метою забезпечення надійності та стабільності розробленої системи був здійснений процес тестування. А саме: функціональна коректність, стійкість до помилок, швидкодія та зручність використання.

Була повністю перевірена логіка програми, а саме:

– відкриття усіх типів файлів;

– адаптивність до зміни вікна;

– наявність елементів керування та їх реакція на дії користувача;

– перевірка обробки некоректних сценаріїв;

– відповідність результатів управління зображеннями вимогам системи;

– відповідність результатів обробки очікуваним ефектам (рис. 3.35);

– автоматичного оновлення панелей після обробки зображення;

– точність обчислення метрик для аналізу зображення;

– передачі даних між аналізом якості та AI-рекомендаціями (рис. 3.36);

– ручне тестування класифікації зображень із очікуваними реальними зображеннями (рис. 3.37);

– застосування пресетів до нових зображень без втрати даних;

– коректність збереження зображень;

– коректність збереження звітів.

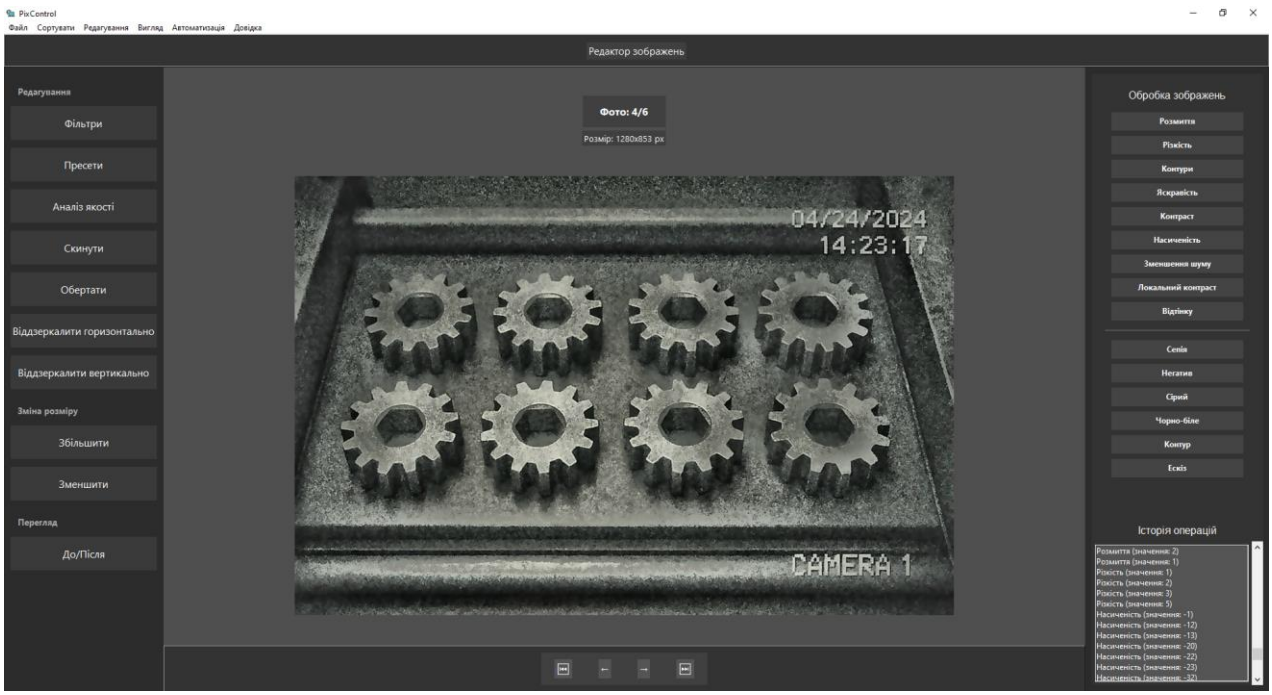

Рисунок 3.35 – Приклад ручного покращення знімку

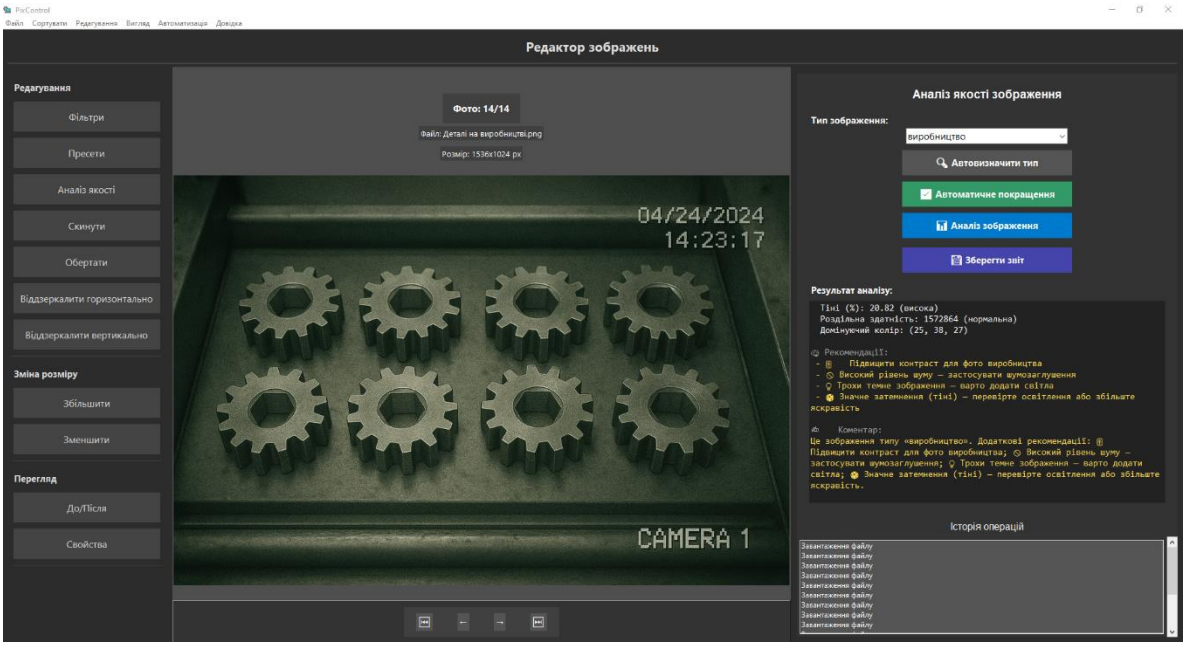

Рисунок 3.36 – Приклад аналізу якості цифрового зображення та рекомендацій щодо обробки

Приклад роботи системи автоматизації управління у вигляді коректного визначення типу зображення на рис. 3.37.

Рисунок 3.37 – Приклад роботи системи автоматизації управління зображеннями

Результати тестування підтвердили технічну придатність розробленої системи. Програма демонструє стабільну роботу в реальному середовищі, точність аналізу та зручність при управлінні великими обсягами файлів.

3.3 Визначення можливостей інтеграції

Завдяки тому, що система має модульну архітектуру, це дає змогу адаптувати її під нові задачі. Задля цього не потрібно буде втручатися суттєву у структуру, що полегшує впровадження її як і для особистого використання, так і задля процесів, де використовуються комп’ютерно-інтегровані технології.

У межах підсистеми фільтрації реалізовано принцип відкритості до доповнення. Наразі програма має всі необхідні функції обробки для базової роботи з цифровим зображенням. При бажанні додати нові варіації обробки зображення, це передбачає створення окремої функції, її зв’язування з

інтерфейсом та вбудовування у загальний алгоритм роботи. Система адаптована до індивідуальних потреб користувача або специфіки предметної області.

Крім того, платформа передбачає інтеграцію елементів штучного інтелекту. Зокрема, можливим є додавання алгоритмів для автоматичного виявлення дефектів на зображеннях, розпізнавання обличчя, або їх сегментації, що має особливу цінність у сфері технічного контролю, наукових досліджень та лабораторної діагностики (рис. 3.38).

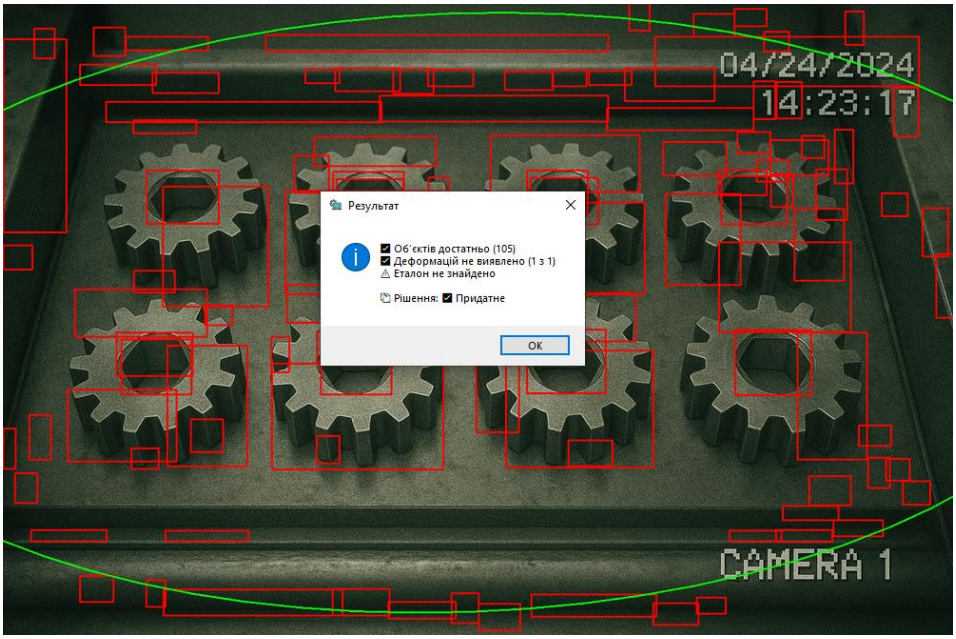

Рисунок 3.38 – Приклад інтеграції алгоритмів розпізнавання дефектів шестерень

Система також підтримує підключення зовнішніх пристроїв, що дозволяє розширити її функціональне поле. Наприклад, інтеграція зі сканером або камерою надає змогу здійснювати обробку зображень у режимі реального часу, що відкриває перспективи застосування у автоматизованих системах спостереження та контролю (рис. 3.39).

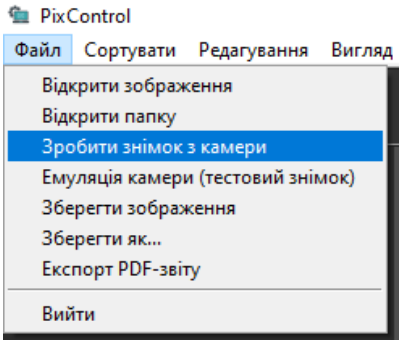

Рисунок 3.39 – Інтеграція з камерою

Наразі реалізована підтримка камери, що дає змогу зробити знімок прямо з камери та відслідковування модуля підключення завантажень зображень з камери. В майбутньому це дає можливість працювати з конвеєрною серією та будь-яким спостереженням в реальному часі.

З метою досягнення високої точності та надійності результатів було реалізовано модуль автоматизованої генерації PDF-звітів, що відображає ключові параметри досліджуваних зображень. Це рішення дозволило й забезпечити створення структурованої технічної документації відповідно до вимог сучасних інженерних практик. Ці дані представлено у компактній табличній формі, що значно полегшує їх порівняння та подальший аналіз у виробничих і дослідницьких завданнях (рис 3.40).

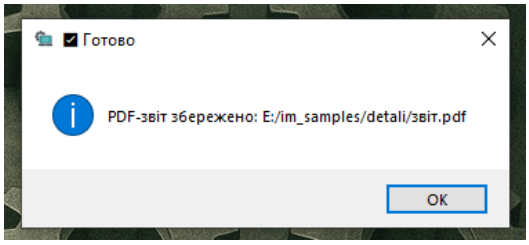

Рисунок 3.40 – Повідомлення про збереження звіту

Інтеграція цієї функції в загальну архітектуру програмного комплексу надала змогу суттєво спростити підготовку звітної документації (рис 3.42).

ТЕХНІЧНИЙ ЗВІТ АНАЛІЗУ ЗОБРАЖЕННЯ

Файл: Деталі на виробництві.png
Тип зображення: виробництво
Дата: 02.06.2025 15:28:44

Метрики зображення:

Яскравість	57.79 (темне)
Контраст	33.94 (низький)
Різкість	1514.19 (висока)
Текстурність	4.81 (середня)
Пересвіт	0.00 (низький)
Недосвіт	0.06 (низький)
Баланс білого (ΔRGB)	[2.19 8.46 6.26] (гарний баланс білого)
Насиченість	75.44 (висока)
Тіні	20.82 (висока)
Роздільна здатність	1572864 (нормальна)
Домінуючий колір	(25, 38, 27)

Рекомендації:

– □□ Підвищити контраст для фото виробництва
– □ Високий рівень шуму — застосувати шумозаглушення
– □ Трохи темне зображення — варто додати світла

AI-коментар:

Це зображення типу «виробництво». Виявлено проблеми: □ Високий рівень шуму — застосувати шумозаглушення. Рекомендовано: □□ Підвищити контраст для фото виробництва; □ Трохи темне зображення — варто додати світла.

Виконав(ла): _____ Дата: _____

1

Рисунок 3.41– Приклад PDF-звіту

Реалізація функції «Свойства» надало змогу спростити підготовку до роботи над зображенням, що важливо для інтеграції в процеси комп'ютерно-інтегрованих технологій (рис 3.42).

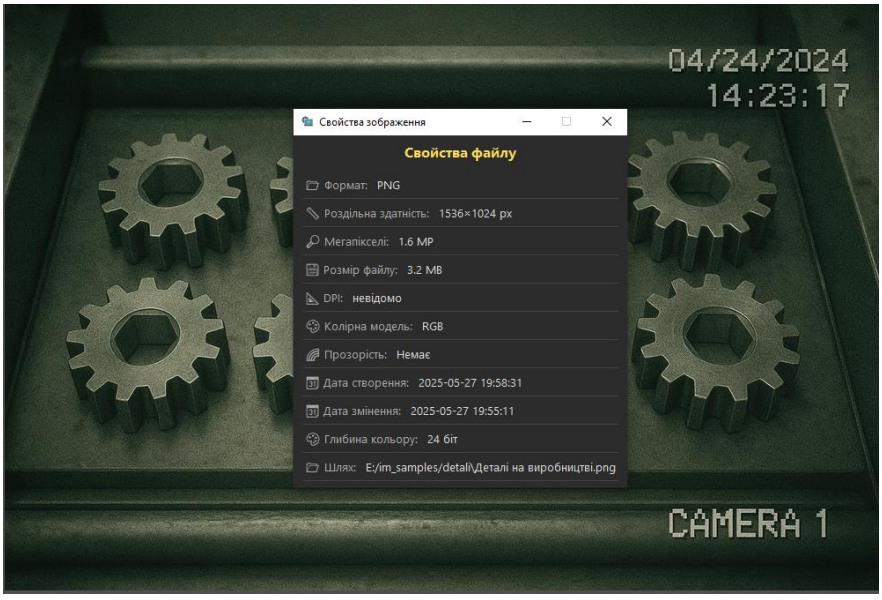

Рисунок 3.42– Функція «Свойства

3.4 Охорона праці

При процесі розробки програмного забезпечення для системи автоматизації основним робочим середовищем є персональний комп’ютер. При такій роботі стається монотонне статичне навантаження, тому що багато часу проводиться у положенні сидячі та зі значною напругою зору.

Основними аспектами, які впливають на умови праці спеціалістів в цій галузі, є вплив електромагнітного випромінювання, недостатній рівень освітлення робочої зони, значне навантаження на зоровий апарат під час роботи з монітором, а також психологічний і емоційний тиск. Крім того, негативний вплив справляє тривале перебування у статичній позі. Сукупність цих факторів може спричинити зниження загальної працездатності, перевтому очей, проблеми з опорно-руховим апаратом та розвиток професійного вигорання.

У процесі розробки та використання програмного забезпечення необхідно враховувати вимоги охорони праці, що забезпечують безпечні та комфортні умови для користувачів.

Згідно з чинними стандартами, такими як ДСТУ EN 527-1:2019 (вимоги до розмірів робочих столів), особлива увага приділяється організації робочого місця [17].

Робоче місце має бути обладнане з урахуванням ергономіки. Стіл повинен мати достатню площу не менше 1200 × 600 мм, а його поверхня матовою, щоб уникнути відблисків. Стілець повинен бути регульованим по висоті, з наявністю спинки та підлокітників для підтримки зручної пози користувача. Висота монітора має бути такою, щоб верхній край екрана знаходився на рівні очей або трохи нижче, а відстань до очей 50 см × 70 см.

Також потрібно забезпечити належне освітлення робочого місця. Рекомендована освітленість становить не менше 300 лк, з використанням природного освітлення або люмінесцентних ламп з розсіяним світлом. Необхідно також стежити за мікрокліматом у приміщенні: температура повітря

повинна бути в межах від 18 °С до 22 °С, вологість від 40 % до 60 %, а рівень шуму повинен не перевищувати 50 дБ.

З метою збереження здоров’я працівників передбачаються регламентовані перерви. Рекомендується робити короткі перерви тривалістю 5-10 хвилин після кожної години роботи за комп’ютером для зняття навантаження з очей та спини. Під час цих перерв варто виконувати вправи для очей, легкі фізичні розминки або змінювати вид діяльності.

Розроблене програмне забезпечення є безпечним у використанні та не містить елементів, які можуть становити фізичну небезпеку для користувача. Інтерфейс програми інтуїтивно зрозумілий і не створює додаткового навантаження чи стресу при взаємодії.

Завдяки автоматизації багатьох рутинних процесів та інтеграції AI-рекомендацій, система допомагає зменшити час, витрачений на ручну обробку, та підвищує загальну продуктивність користувача без негативного впливу на його здоров’я.

Таким чином, розробка та використання програмної системи відповідає чинним вимогам з охорони праці, що дозволяє створити безпечні та комфортні умови для роботи користувачів.

ВИСНОВКИ

У межах виконання кваліфікаційної роботи бакалавра було проведено розроблення системи автоматизації управління цифровими зображеннями, що відповідає сучасним вимогам до програмних рішень до засобів обробки зображень. В результаті досліджень та практичної реалізації вдалося досягти всіх поставлених задач.

У першому розділі проведено аналіз зображень як елемента системи автоматизації. Розглянуто характеристики зображень, які визначають процес управління, досліджено актуальні сфери використання. Було проаналізовано існуючі програмні рішення (Adobe Photoshop, GIMP, Topaz Photo AI, ImageJ) та підходи до обробки зображень (традиційні методи обробки та трансформації, сучасні ШІ-інструменти), сформовано вимоги до розроблюваної системи та основні завдання для її реалізації. У результаті дослідження було виявлено потребу у продукті, що поєднує комфортне управління та обробку файлів з можливістю автоматизованої інтеграції, не втрачаючи зручності для користувача у використанні.

Другий розділ присвячений проектуванню архітектури та визначенню основних інструментів розробки системи автоматизації управління зображенням, що стало фундаментом для подальшої програмної реалізації. Було визначено загальну структуру системи з акцентом на основні модулі: аналізу, обробки, історії змін, станів зображення та AI-модуль. Розроблено алгоритм роботи користувача в системі та описано особливості реалізації автоматизації у системі.

У третьому розділі детально опрацьований процес розроблення системи автоматизації управління цифровими зображеннями. Проєкт реалізований в окремих модулях, що зменшило навантаження на головний та відповідає принципам сучасної розробки програмного забезпечення. Інтегровано AI-

модулі, які розпізнають тип зображення за принципом поєднання змісту зображення та додаткового опису, який підлаштовується під використання системи, та пропонують рекомендації щодо покращення якості. Це дає потужний фундамент для майбутньої інтеграції у виробничий процес та налаштування системи під потреби користувача. Окремо розглянуто питання інтеграції у різні сфери: від автоматизації обробки цифрових зображень у промисловості до візуального аналізу. Програма легко адаптується під нові задачі та вимоги користувачів. Закладено основи для подальшого розвитку системи, передбачивши можливості інтеграції таких функцій, як розпізнавання об’єктів та виявлення дефектів. Крім того, реалізовано технічну можливість взаємодії з периферійними пристроями.

Унікальність розробленої системи автоматизації управління цифровими зображеннями полягає у поєднанні класичних методів обробки із сучасними AI-інструментами. Завдяки гнучким налаштуванням вона універсальна та легко адаптується до різних сфер застосування. Контроль якості у виробництві, інтеграція у візуальні сенсорні системи роботів для навігації, виявлення об’єктів, технічний аналіз, безпека та відеоспостереження, зокрема.

Загалом, отримані результати свідчать про успішне виконання поставленого технічного завдання. Розроблена система автоматизації управління цифровими зображеннями поєднує зручний інтерфейс, широкий функціонал та можливості автоматизації завдяки впровадженню штучного інтелекту. Це робить її ефективним інструментом для інженерних рішень у сфері комп’ютерно-інтегрованих технологій. Подальший розвиток дозволить розширити функціональні можливості та забезпечити ще більш глибоку інтеграцію у виробничі процеси.

ПЕРЕЛІК ПОСИЛАНЬ

1. ДСТУ 3008-15. Документація. Звіти у сфері науки та техніки структура та правила оформлення. Введ. 2015-06-22. К. Держстандарт України. – 2017. – 29 с.

2. Методичні вказівки з підготовки кваліфікаційної роботи для здобувачів першого (бакалаврського) рівня вищої освіти денної і заочної форми навчання спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» освітньої програми «Автоматизація та комп'ютерно-інтегровані технології» / Упоряд.: І.Ш. Невлюдов, О.І. Филипенко, О.В. Токарєва, С.П. Новоселов, О.В Сичова. – Харків: ХНУРЕ, 2023. – 64 с.

3. Навчальний посібник з підготовки кваліфікаційної роботи бакалавра для здобувачів вищої освіти денної і заочної форм навчання спеціальностей 151 «Автоматизація та комп'ютерно-інтегровані технології» та 174 «Автоматизація, комп'ютерно-інтегровані технології та робототехніка» освітньої програми «Системна інженерія» [Електронний ресурс] : навчальний посібник / І. Ш. Невлюдов, О. В. Токарєва, О. М. Цимбал, А. І. Бронніков ; М-во освіти і науки України, ХНУРЕ. – Харків : Видавництво Іванченка І. С. – 2023. – 218 с.

4. Konieva, A. Main trends in the development of automated image processing systems / A. Konieva, S. Sotnik // «Computer-integrated technologies, automation and robotics» CITAR-2025. – 2025. – 68- 72 с.

5. Конєва, А.І. Перспективи застосування сучасних систем автоматизації / А.І. Конєва // Автоматизація та приладобудування («Automation and Development of Electronic Devices» ADED-2022) [Електронний ресурс]: збірник студентських наукових статей / Харківський національний університет радіоелектроніки; [редкол.: І.Ш. Невлюдов та ін.]. – Харків : ХНУРЕ. – 2022. – Вип. 2. – 65-72 с.

6. Конєва, А. І. Аналіз особливостей технології Індустрії 4.0 / А. І. Конєва // «Automation and development of electronic devices» ADED-2021 Part 2. – 2021. – 85-88 с.

7. Конєва, А. І. Перспективи розвитку безпілотних систем / А.І. Конєва // Автоматизація та приладобудування («Automation and Development of Electronic Devices» ADED-2023 Part 2) [Електронний ресурс]: збірник студентських наукових статей / Харківський національний університет радіоелектроніки; [редкол.: І.Ш. Невлюдов та ін.]. – Харків : ХНУРЕ. – 2023. – Вип. 2. – 164-170 с.

8. Заболуєва, М. Ю. Автоматизація процесу сегментації ультразвукових зображень в медичній діагностиці. – 2024. – 85 с.

9. Юшко, О. В. Проблеми застосування методів розпізнавання та порівняння зображень під час навігації безпілотних літальних апаратів. Інфокомунікаційні та комп’ютерні технології . – 2024. – 116-120 с.

10. Методичні вказівки до лабораторних робіт з дисципліни «Технології мобільної робототехніки» / уклад. А.І. Бронніков. – Харків : ХНУРЕ. – 2021. – 52 с.

11. Перелигін, Б.В. Цифрова обробка зображень: конспект лекцій. Одеса:Одеський державний екологічний університет. – 2023. – 186 с.

12. Лавер, В.О. Обробка зображень: навч.-метод. посіб. / В.О. Лавер, О.М. Левчук. – Ужгород. – 2021. – 51 с.

13. Zaitsev, D. Огляд засобів ефективної сегментації зображень з використанням методів кластеризації даних. Системи управління, навігації та зв’язку. Збірник наукових праць. – 2024. – 77-81 с.

14. Невлюдов І.Ш. Комп’ютерно-інтегровані технології виробництва технічних засобів автоматизації. Частина 1: Підручник Харків. – 2021. – 604 с.

15. Методичні вказівки до лабораторних занять з дисципліни «Спеціалізовані мови програмування» для студентів спеціальності 151 Автоматизація та комп’ютерно-інтегровані технології, освітньо професійна

програма CI / Упоряд.: О.М. Цимбал, Д.В. Гурін, В.М. Гурін. – Харків: ХНУРЕ. – 2018. – 36 с.

16. Теорія автоматичного управління (збірник задач) [Текст]: навч. посіб. для студентів спеціальності 151 Автоматизація та комп'ютерно-інтегровані технології / І.Ш. Невлюдов, О.В. Токарєва; Харків. нац. ун-т радіоелектроніки. – Харків. – 2020. – 240 с.

17. ДСТУ EN 527-1:2019 Меблі офісні. Робочі столи. Частина 1. Розміри (EN 527-1:2011, IDT). – Київ: ДП «УкрНДНЦ», 2019. – 15 с.