Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

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

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

РОЗРОБКА ТЕЛЕГРАМ-БОТА ДЛЯ РОЗПІЗНАВАННЯ ТЕКСТУ НА ЗОБРАЖЕННЯХ ТА ЙОГО КОНВЕРТУВАННЯ
(тема)

Виконав:
студент 4 курсу, групи ІТІНФ-18-2
Гунько А. С.
(прізвище, ініціали)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник ст. викл. Кіношенко Д.К.
(посада, прізвище, ініціали)

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

2022 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти перший (бакалаврський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Гуньку Антону Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка телеграм-бота для розпізнавання тексту на зображеннях та його конвертування

затверджена наказом університету від 16 травня 2022 року № 541Ст

2. Термін подання студентом роботи до екзаменаційної комісії 27 травня 2022 р.

3. Вихідні дані до роботи: теоретичні відомості про методи обробки зображень, матеріали про розробку телеграм-ботів, дані інтернет-мережі, бібліотека розпізнавання текстів з відкритим кодом Tesseract, мови програмування Java та Python

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд технологій обробки зображень.
2. Аналіз існуючих методів обробки зображень.
3. Огляд підходів до синтезу мовлення.
4. Огляд підходів до розробки телеграм-ботів.
5. Програмна реалізація телеграм-бота для розпізнавання тексту на зображеннях та його конвертування з використанням бібліотеки Tesseract.
6. Тестування розробленої моделі.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність проблеми розпізнавання тексту, постановка задачі, навчання нейронної мережі із використанням бібліотеки Tesseract, тестові зображення, найважливіші частини коду програми, аналіз результатів моделювання, висновки.

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

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

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	18.04.2022	
2	Аналіз завдання, підбір літератури	19.04.22-21.04.22	
3	Дослідження існуючих методів вирішення задачі розпізнавання тексту	22.04.22-25.04.22	
4	Огляд існуючих алгоритмів розпізнавання тексту	26.04.22-30.04.22	
5	Проектування застосунку	01.05.22-05.05.22	
6	Програмна реалізація	06.05.22-23.05.22	
7	Оформлення пояснювальної записки	24.05.22-26.05.22	
8	Перевірка на плагіат	05.06.22	
9	Рецензування	06.06.22	
10	Підготовка презентації та доповіді	07.06.22-12.06.22	
11	Занесення роботи в електронний архів	12.06.22	
12	Попередній захист кваліфікаційної роботи	13.06.22	

Дата видачі завдання 18 квітня 2022 р.

Студент _____
(підпис)

Керівник роботи _____ ____ст. викл. Кіношенко Д.К._
(підпис) (посада, прізвище, ініціал)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 49 с., 26 рис., 2 дод., 30 джерел.

РОЗПІЗНАВАННЯ ТЕКСТУ, РОЗПІЗНАВАННЯ ТЕКСТУ З ЗОБРАЖЕНЬ, МЕТОДИ РОЗПІЗНАВАННЯ ТЕКСТУ, НЕЙРОННІ МЕРЕЖІ, ОПТИЧНЕ РОЗПІЗНАВАННЯ ТЕКСТУ, ТЕЛЕГРАМ-БОТ.

Об’єктом роботи є набір зображень з текстом.

Метою роботи є розробка застосунку для розпізнавання тексту з зображень, ідентифікації мови цього тексту та його озвучення на ідентифікованій мові.

Використано метод оптичного розпізнавання тексту. Проведено дослідження за допомогою бібліотек Tess4j для розпізнавання тексту, Polyglot для ідентифікації мови та Gtts для озвучення тексту.

У результаті роботи здійснена програмна реалізація системи у вигляді телеграм-боту для розпізнавання тексту з зображень та його конвертації у аудіо та текстовий формати.

TEXT RECOGNITION, TEXT RECOGNITION FROM IMAGES, TEXT RECOGNITION METHODS, NEURAL NETWORKS, OPTICAL RECOGNITION, TELEGRAM-BOT.

The subject of the work is a set of images with text.

The aim of the work is to develop an application for text recognition from images, language identification of this text and its sounding in the identified language.

The method of optical text recognition is used. Research has been conducted using the Tess4j libraries for text recognition, Polyglot for language identification and Gtts for text sounding.

As a result of the work the software implementation of the system in the form of a telegram-bot for text recognition from images and its conversion into audio and text formats was carried out.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

OCR – Optical Character Recognition (Оптичне розпізнавання символів)

ICR – Intelligent Character Recognition (Інтелектуальне розпізнавання символів)

IWR – Intelligent Word Recognition (Інтелектуальне розпізнавання слів)

OMR – Optical Mark Recognition (Оптичне розпізнавання знаків)

TTS – Text-To-Speech

gTTS – Google Text-To-Speech

REST – Representational State Transfer

ВСТУП

Розпізнавання тексту є одним із напрямків розпізнавання образів. Цей процес є дуже складною задачею в теоретичному й практичному планах, незважаючи на те, що з нею досить легко може впоратись людина. Вкрай складно реалізувати систему, яка зможе ефективно та безпомилково виконувати цей процес [1]. Під розпізнаванням розуміється співвіднесення зображення об'єкта, його образу, набору ознак самому об'єкту. Найпопулярнішими прикладами розпізнавання тексту є:

– оцифрування тексту зображень для подальшої роботи з цифровими аналогами;

– обробка анкетних бланків;

– розпізнавання номерів машин.

Ця тема залишається досить актуальною і на сьогоднішній день, оскільки до сих пір немає універсального рішення, яке змогло б безпомилкового виконувати розпізнавання тексту з зображень та документів.

1 ОГЛЯД СИСТЕМИ РОЗПІЗНАВАННЯ ТЕКСТУ ТА СИНТЕЗУ МОВЛЕННЯ

1.1 Розпізнавання тексту

Оптичне розпізнавання символів – це технологія, яка дозволяє розпізнавати друкований або рукописний текст всередині цифрових зображень та документів. Основний процес включає дослідження тексту з об'єкта з якого відбуватиметься розпізнавання та переклад символів у код ASCII, який можна використовувати для обробки даних надалі цифровим пристроєм [2].

Зображення зазвичай фіксуються за допомогою комп’ютерних сканерів або цифрових камер. Їхня якість відіграє важливу роль у визначенні кількості помилок при розпізнаванні тексту. Наприклад, система OCR може не розпізнавати текст, якщо джерело введення фізично несправне, старе або містить будь-які спотворення, такі як розриви або плями.

Існує два типи систем оптичного розпізнавання символів:

– автономна система OCR: витягує дані з відсканованих зображень через оптичні сканери та камери;

– онлайн-система OCR – використовує спеціальні дигітайзери для запису введення користувача в режимі реального часу відповідно до порядку літер, рухів та швидкості письма.

Технічно кажучи, кожна система OCR проходить процес послідовних етапів перетворення тексту з зображення у комп’ютерний цифровий текст (рис. 1.1):

– етап отримання зображення: фіксує вхідний документ;

– етап попередньої обробки: покращує якість зображення, шляхом відокремлення тексту для розпізнавання від заднього фону;

– етап виділення та класифікації ознак: витягує подібні об’єкти з вхідного документа й групує їх у класи, які пізніше розпізнаються як символи й слова;

– етап постобробки: уточнює вихідний текст OCR шляхом виправлення лінгвістичних помилок.

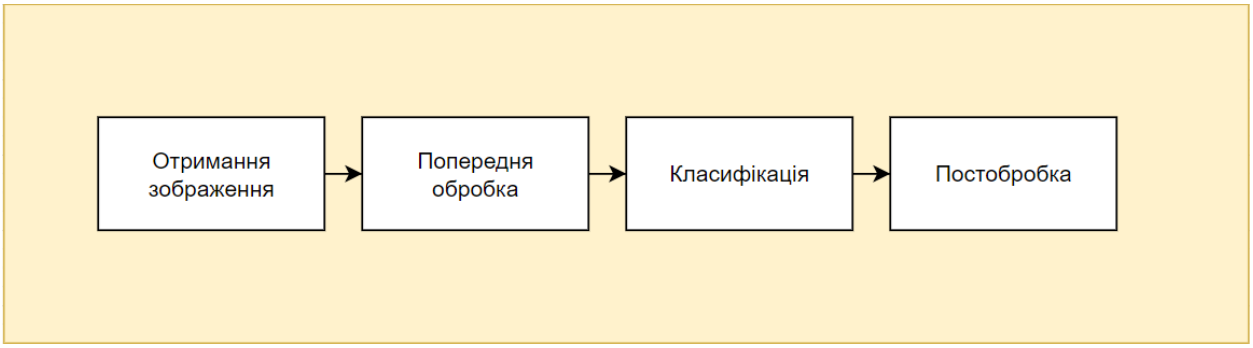

Рисунок 1.1 – Етапи оптичного розпізнавання тексту

Технологія OCR досить часто піддається помилкам та недолікам. Рівень помилок може буду досить високим, якщо, наприклад, папір, що сканується, має численні дефекти (поганий фізичний стан, якість друку, старі папери тощо). Коли система OCR не розпізнає символ, це призводить до створення орфографічних помилок у вихідному тексті [3]. Наприклад, символ «В» може бути неправильно перетворений на число «8», символ «Ь» в число «6», символ «О» в число «0» тощо. Щоб вирішити цю проблему, люди можуть вручну переглядати та виправляти вихідний текст після процесу розпізнавання тексту. До певної міри ця процедура вважається дорогою, трудомісткою та схильною до помилок. Кращим підходом може бути автоматизація виправлення орфографічних помилок за допомогою програмного забезпечення. Це рішення полягає у використанні пошукового словника для знаходження та виправлення слів з помилкою. Хоча ця техніка намагається вирішити проблему – вона насправді створює нову, ще більш незручну. Фактично, словниковий підхід намагається розглядати слово з помилкою окремо, у тому сенсі, що він не враховує контекст, у якому вона сталася. З цієї причини були запропоновані

методи виправлення помилок на основі лінгвістичного контексту для їх виявлення та виправлення. Як результат, такий підхід значно покращує якість розпізнавання тексту [4].

Очевидно, що всі вищезгадані методи все ж мають загальний недолік: вони всі вимагають інтеграції величезного словника масивних термінів, який охоплює майже кожне слово цільової мови. Крім того, цей словник має включати власні іменники, назви країн і місць, наукову термінологію та технічні ключові слова. Нарешті, зміст цього словника слід постійно оновлювати, щоб включати нові слова, що з’являються в мові.

1.2 Види розпізнавання тексту

Існує декілька видів розпізнавання тексту. Кожен з них відрізняється не лише способом зчитування даних з документів або зображень, але й самим форматом документа [5].

1.2.1 Інтелектуальне розпізнавання слів

Інтелектуальне розпізнавання слів – це технологія, яка дозволяє розпізнавати рукописний текст з документів та зображень, шляхом розпізнавання цілих слів. IWR використовує штучний інтелект для розпізнавання цілих слів, а не окремих символів, на відміну від ICR, який обробляє слова посимвольно [6]. Наприклад, при використанні ICR для вилучення слова «весна», система розпізнає «в», «е», «с», «н», «а». IWR зіставлятиме літери зі словником, щоб витягти все слово «весна» на основі розпізнавання образів і різних алгоритмів. Технологія автоматично виправляє орфографічні помилки, порівнюючи це слово зі словником, та дає нечітку відповідність йому відповідно до структури речення. Якщо слово написано з

помилкою, наприклад «апльсин», воно виправить його на «апельсин» під час обробки. IWR зменшує кількість символьних помилок, пов'язаних з цими механізмами, і ідеально підходить для обробки реальних документів, які містять переважно дані довільної форми [7].

1.2.2 Інтелектуальне розпізнавання символів

Інтелектуальне розпізнавання символів – це технологія, яка дозволяє розпізнавати рукописний текст з документів та зображень посимвольно.

Більшість програмного забезпечення ICR має систему самонавчання, яку називають нейронною мережею, яка автоматично оновлює базу даних розпізнавання для нових шаблонів рукописного введення. Оскільки цей процес бере участь у розпізнаванні рукописного тексту, рівень точності за деяких обставин може бути не дуже високим, але може досягати більше 97% показників точності при читанні тексту в структурованих формах. Часто для досягнення цих високих показників розпізнавання в програмному забезпеченні використовується кілька механізмів зчитування, і кожному з них надається право голосу для визначення поточного механізму зчитування символів. У числових полях перевага надається механізмам, призначеним для читання чисел, тоді як в альфа-полях двигуни, як призначені для читання рукописних літер [8].

Інтелектуальне розпізнавання слів може розпізнавати та витягувати не тільки друковану інформацію, а й рукописний текст. ICR розпізнає слова на рівні символів, тоді як IWR працює з повними словами або фразами.

1.2.3 Оптичне розпізнавання знаків

Оптичне розпізнавання знаків – це технологія збору введених людьми даних шляхом розпізнавання позначок або візерунків у документі.

Ця популярна та високоточна технологія розпізнавання використовується для збору даних із тестів, опитувань, бюлетенів та багато інших видів форм [9].

Оптичне розпізнавання марок дає змогу респонденту вибрати відповідь на запитання за допомогою заповнення «бульбашки» або «помітки», пов’язаної з вибором відповіді. Наприклад, на зображенні праворуч респондент вибрав «Відмінно» у полі, щоб вказати, що місце розташування було «Відмінне».

На додаток до питань із кількома варіантами, OMR можна використовувати для отримання імен, ідентифікаційних номерів та інших даних, які не мають варіантів (рис. 1.2). У цьому випадку респондент заповнив своє ім’я як «KP CRABTREE». Збір такого типу даних за допомогою OMR набагато точніший, ніж спроба розпізнати рукописне ім’я респондента за допомогою технології ICR [10].

OMR підтримує декілька форматів полів:

– поле з багатьма варіантами: є декілька варіантів, але вибирається лише один;

– сітка: бульбашки або лінії налаштовуються у форматі сітки, щоб користувач міг заповнити номер телефону, ім’я, ідентифікаційний номер тощо;

– поле с логічним значенням: потрібно відповісти (так чи ні) на всі питання, що підходять;

– бінарне поле: потрібно відповісти (так чи ні) лише на одне питання;

– поле з пунктирними лініями.

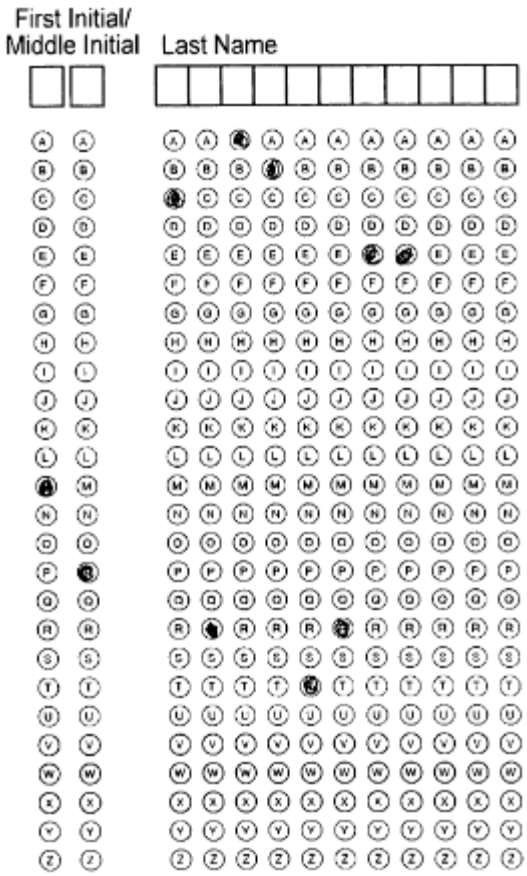

Рисунок 1.2 – Форма для заповнення ім’я

У OMR є деякі недоліки та обмеження. Якщо користувач хоче зібрати велику кількість тексту, OMR ускладнює збір даних. Також існує ймовірність відсутності даних у процесі сканування, а неправильні або ненумеровані сторінки можуть призвести до того, що вони будуть скановані в неправильному порядку. Крім того, якщо не діють заходи безпеки, сторінку можна повторно сканувати, надаючи повторювані дані [11].

У результаті широкого поширення та простоти використання OMR стандартизовані іспити можуть складатися переважно з питань із кількома відповідями, що змінює характер того, що перевіряється. Хоча технологія OMR існує вже давно, вона продовжує розвиватися і на даний момент є найпопулярнішим у світі рішенням для обробки різних типів форм.

1.3 Синтез мовлення

Синтез мовлення – це штучне виробництво людського мовлення. Комп’ютерна система, яка використовується для цієї мети, називається синтезатором мовлення або «мовним комп’ютером», і може бути реалізована в програмних або апаратних продуктах [12].

Синтезоване мовлення можна створити шляхом об’єднання фрагментів записаного мовлення, які зберігаються в базі даних. Системи відрізняються за розміром збережених мовних одиниць. Наприклад, система, яка зберігає фони або дифони, забезпечує найбільший діапазон виводу, але може мати погану чіткість. Для конкретних прикладів використання зберігання цілих слів або речень дозволяє отримати високоякісний вихід. Крім того, синтезатор може включати модель голосового тракту та інші характеристики людського голосу, щоб створити повністю синтетичний голосовий вихід [13].

Якість синтезатора мовлення оцінюється за його схожістю з людським голосом і його здатністю бути зрозумілим. Розбірлива програма перетворення тексту в мовлення дозволяє людям з вадами зору або читанням слухати написані слова на електронних пристроях. Багато комп’ютерних операційних систем та смартфонів мають встановлений синтезатор мовлення за замовчуванням.

Система перетворення тексту в мовлення або «двигун» складається з двох частин:

– front-end;

– back-end.

Front-end має два основні завдання. По-перше, він перетворює необроблений текст, що містить символи, такі як числа та скорочення, в слова. Наприклад, результатом обробки символу «2» буде слово «два». Цей процес часто називають нормалізацією тексту, попередньою обробкою або токенізацією. Потім Front-end призначає фонетичні транскрипції кожному слову, а також розділяє та позначає текст на інтонаційна одиниці, такі як фрази

та речення. Процес присвоєння словам фонетичної транскрипції називається перетворенням тексту в фонему. Фонетичні транскрипції та інтонаційна інформація разом складають символічне мовне уявлення, яке є результатом обробки Front-end частини. Back-end, який часто називають синтезатором, потім перетворює символічне мовне уявлення у звук. У деяких системах ця частина включає обчислення цільової інтонації, яка потім накладається на вихідну мову [14].

1.4 Постановка задачі

Таким чином, розпізнавання тексту з зображень та його конвертація в аудіо та текстовий формати є актуальним завданням на сьогоднішній день, особливо у якості телеграм-бота, де не потрібно завантажувати окремі додатки до вашого гаджету, щоб мати доступ до цього функціоналу. Також це може використовуватися людьми з обмеженими можливостями, щоб мати змогу почути та зрозуміти озвучений текст з будь-якого зображення. У зв’язку з цим ставиться завдання пошуку технологій та розробка алгоритму для розпізнавання тесту з зображень та його конвертацію в аудіо формат.

Об’єктом роботи є набір зображень з текстом.

Метою роботи є розробка застосунку для розпізнавання тексту з зображень, ідентифікації мови цього тексту та його озвучення на ідентифікованій мові.

Для досягнення мети необхідно вирішити такі завдання:

– провести аналіз існуючих підходів до розпізнавання тексту;

– розробити алгоритм для розпізнавання тексту та його конвертацію у аудіо формат;

– провести аналіз існуючих підходів та бібліотек щодо синтезу мовлення;

– реалізувати телеграм-бота для завантаження зображень та отримання результатів.

2 МАТЕМАТИЧНА МОДЕЛЬ РОЗПІЗНАВАННЯ ТЕКСТУ НА ЗОБРАЖЕННЯХ

2.1 Попередня обробка

Серед усіх етапів OCR етап попередньої обробки є найважливішим з них, оскільки точність системи OCR дуже залежить від того, наскільки добре він виконаний. Основна його мета полягає в тому, щоб максимально підвищити шанси на успішне розпізнавання, відокремивши текст для розпізнавання від заднього фону [15].

2.1.1 Бінаризація зображення

Бінаризація зображення – це процес перетворення кольорового або сірого зображення в чорно-біле або бінарне зображення, яке складається лише з двоих пікселів – чорного та білого (рис. 2.1). Бінаризація виконується для відокремлення тексту або будь-якого іншого об'єкта на зображенні від фону для подальших маніпуляцій із ним. Цей процес є необхідним, оскільки більшість алгоритмів розпізнавання працюють лише з бінарними зображеннями [16-17].

Найпростішим способом бінаризації є використання порогового значення. Якщо значення пікселя перевищує поріг, він розглядається як білий піксель, інакше як чорний піксель. Зазвичай у якості порога використовується значення 127, оскільки це рівно половина діапазону пікселів 0–255.

Але ця стратегія не завжди може давати бажані результати. У тих випадках, коли умови освітлення не є однорідними на зображенні, цей метод не працює.

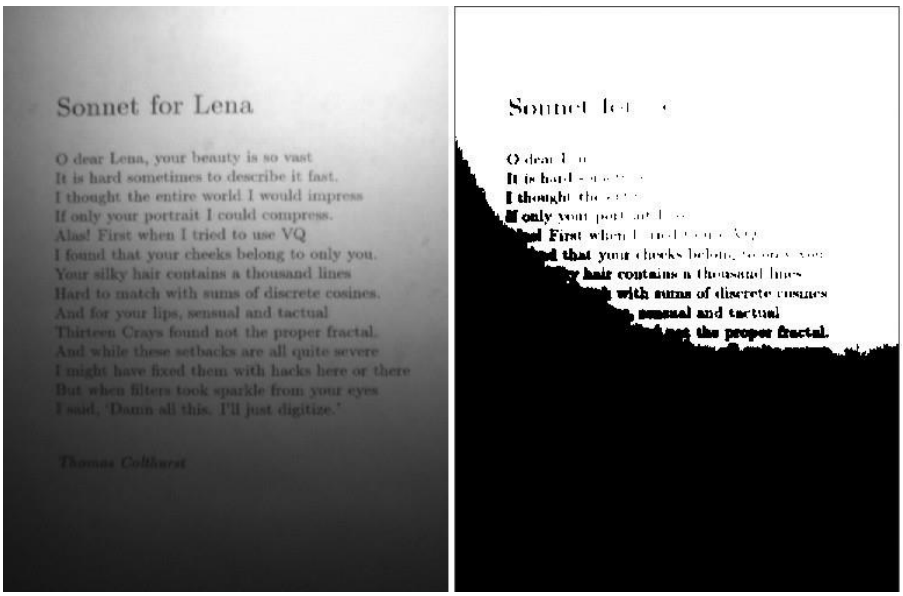

Рисунок 2.1 – Результат бінаризації зображення

Отже, вирішальна частина бінаризації – це визначення порогу. Це можна зробити за допомогою різних технік.

Першої з них буде техніка під назвою «Локальний екстремум». Використовуючи цю стратегію, ми матимемо різні порогові значення для різних частин зображення, залежно від навколишніх умов освітлення тощо.

$$C(i,j) = \frac{I_{max} - I_{min}}{I_{max} - I_{min} + E}, \tag{1.1}$$

де C(i,j) - порогове значення для визначеної області на зображенні;

Imax - максимальне значення пікселя на зображенні;

Imin - мінімальне значення пікселя на зображенні.

Далі йде «Метод Оцу». Цей метод автоматично обчислює порогове значення на всьому зображенні, враховуючи його різні характеристики, такі як умови освітлення, контраст, різкість тощо (рис. 2.2).

Рисунок 2.2 – Результат бінаризації зображення за допомогою методу Оцу

Останньою технікою є «Адаптивне порогове значення». Цей метод дає поріг для невеликої частини зображення залежно від характеристик його місцевості та сусідів, тобто не існує єдиного фіксованого порогу для всього зображення, але кожна невелика частина зображення має різний поріг, а також дає плавний перехід (рис. 2.3).

Рисунок 2.3 – Результат бінаризації зображення за допомогою адаптивного порогового значення

2.1.2 Корекція нахилу зображення

Відповідним зображенням для розпізнавання тексту вважається те, в якому лінії документа, вздовж яких йдуть окремі символи, є горизонтальними

або вертикальними залежно від мови. Ухилення може бути навмисним, щоб підкреслити важливі деталі в документі, чи ненавмисним через ті чи інші умови отримання зображення [18]. У будь-якому випадку воно має бути усунено, інакше точність наступних процесів сегментації та класифікації різко впаде. Корекція ухилення складається з вирішення двох завдань:

– визначення кута нахилу;

– вирівнювання зображення.

Зазвичай розрізняють два типи нахилу в документах: глобальний і локальний. Перша категорія є найпопулярнішою при дослідженнях, для неї було запропоновано безліч методів, проте далеко не всі проблеми вирішені, особливо для документів з малюнками або з шрифтами різних розмірів. Виділення глобальних ухилень поділяють на чотири основні категорії методів:

– проекційний профіль;

– перетворення Хафа;

– групування найближчих сусідів;

– крос-кореляція між перерізами.

Метод проекційного профілю є найбільш широко використовуваним способом визначення кута перекосу на зображенні [19]. У діапазоні очікуваних кутів обчислюються окремі проекційні профілі, з яких вибирається той, який найбільш яскраво відображає різницю між особливостями ліній з пікселями знаків та ліній з пікселями фону (рис. 2.4).

В свою чергу підхід з урахуванням перетворень Хафа націлений на виявлення ліній і кривих на цифрових зображення. Використовується у припущенні, що лінії, що характеризують спрямованість тексту, містять досить велику кількість пікселів. Щоб підвищити обчислювальну ефективність, застосовуються різновиди базової ідеї використання перетворення Хафа, які скорочують кількість точок у просторі образів.

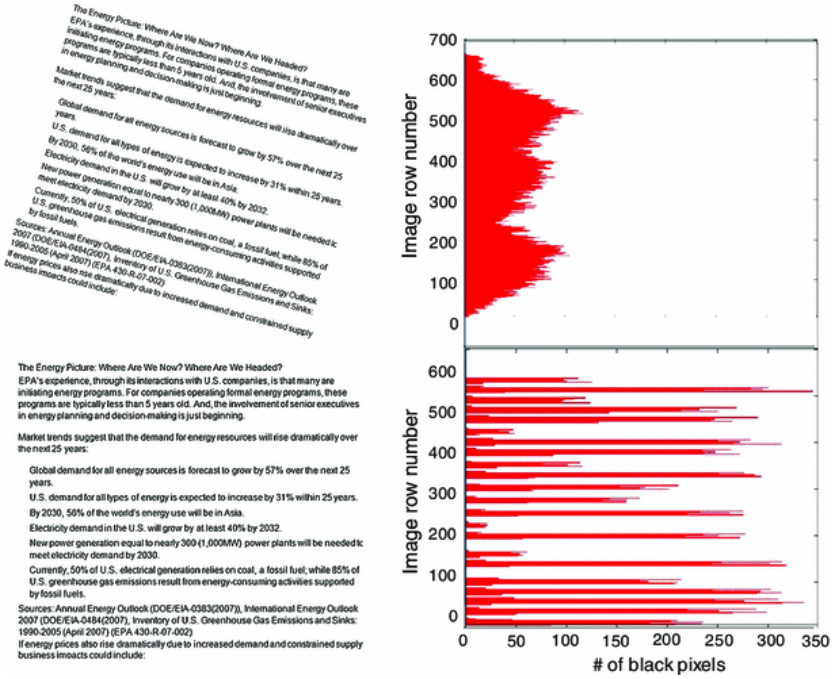

Рисунок 2.4 – Результат роботи методи проекційного профілю

2.1.3 Усунення шуму

Шум – це небажана інформація, яка погіршує якість зображення. Він відображається у вигляді зерен та має багато специфічних форм [20].

Шум солі і перцю, також відомий, як імпульсний шум – є формою шуму, який іноді можна побачити на цифрових зображеннях. Він може бути викликаний різкими та раптовими порушеннями сигналу зображення і відображається як білі і чорні пікселі, що рідко зустрічаються (рис. 2.5).

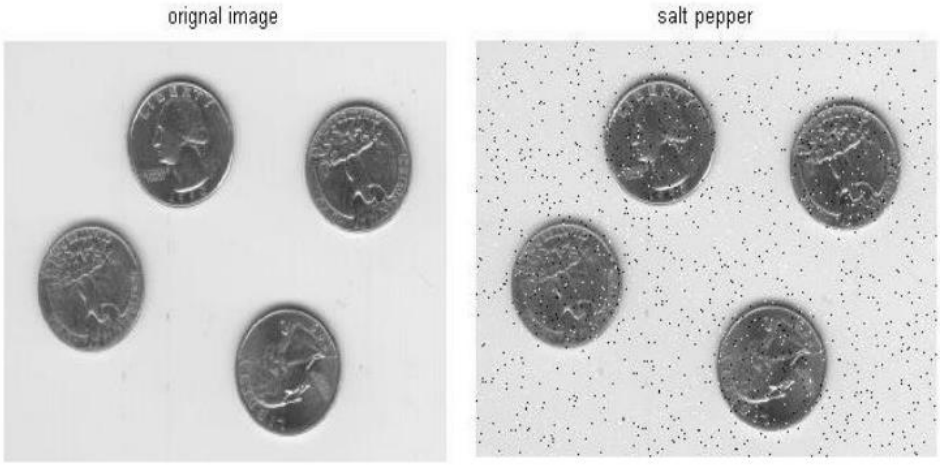

Рисунок 2.5 – Шум солі і перцю

Дробовий шум – це шум, який виникає, коли кількість фотонів, що сприймаються датчиком, недостатня для надання статистичної інформації, яку можна виявити (рис. 2.6).

Рисунок 2.6 – Дробовий шум

Гауссівський шум – це статистичний шум, що має щільність ймовірності, рівну густини ймовірності нормального розподілу (рис. 2.7). Значення, які може приймати такий шум, мають гауссівський розподіл [21].

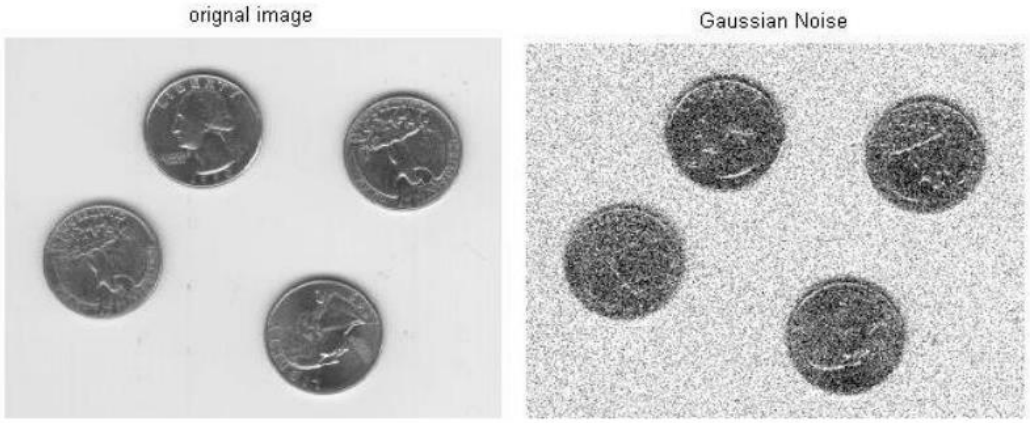

Рисунок 2.7 – Гауссівський шум

Скляний шум (Speckle Noise) – це мультиплікативний шум на відміну від гауссівського шуму та шуму солі і перцю. Він може бути змодельований шляхом випадкових множень значення на значення пікселів зображення: *P = I + n * I, де P – скляний шум, I – вхідне зображення, n – рівномірний шум зображення* (рис. 2.8).

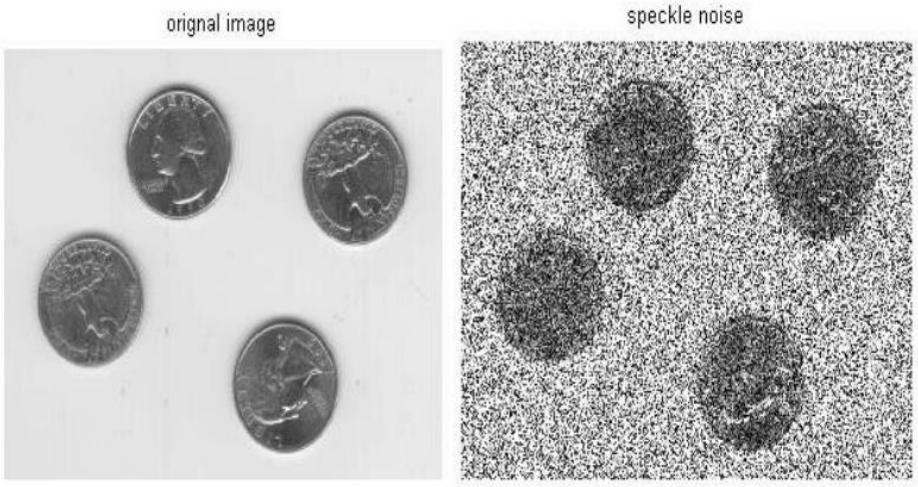

Рисунок 2.8 – Скляний шум

Існує два типи підходів до видалення шуму: лінійна фільтрація та нелінійна фільтрація.

Лінійна фільтрація відбувається, коли операція, що виконується для кожного пікселя, є простою математичною операцією зі скаляром, результат

якої подібний для всіх пікселів. Наприклад, якщо ми множимо інтенсивність кожного пікселя на 2, то все зображення посилюється в два рази, що означає, що ми фактично помножили матрицю зображення на 2.

З іншого боку, у нелінійній фільтрації, загальний вплив на зображення неможливо передбачити лише за операцією, що виконується з кожним пікселем. Наприклад, зведення в квадрат кожного пікселя не те саме, що зведення в квадрат матриці зображення [22].

Позбавлення зображення від шумів є дуже важливим етапом при обробці зображень, завданням якого є видалення шуму з зображення таким чином, щоб оригінальне зображення було видно.

2.2 Розпізнавання тексту

Першим кроком розпізнавання тексту є використання сканера для обробки фізичної форми документа. Після копіювання всіх сторінок програмне забезпечення OCR перетворює документ у чорно-білу версію. Відскановане або растрове зображення потім аналізується на наявність світлих і темних областей, де темні області визначаються як символи, які потрібно розпізнати, а світлі ділянки визначаються як фон [23].

Темні ділянки потім обробляються, щоб знайти букви алфавіту або цифри. Програми розпізнавання тексту можуть відрізнятися за своїми техніками, але зазвичай передбачають розпізнавання по одному символу, слову або блоку тексту за раз. Потім символи ідентифікуються за допомогою одного з двох алгоритмів:

– розпізнавання візерунків;

– виявлення функції.

Виявлення функції – програми розпізнавання тексту застосовують правила щодо особливостей певної літери чи цифри, щоб розпізнати символи у відсканованому документі. Характеристики можуть включати кількість

нахилених ліній, перехрещених ліній або кривих у символі для порівняння. Наприклад, велика літера «А» може зберігатися як дві діагональні лінії, які перетинаються з горизонтальною лінією посередині [24].

Розпізнавання візерунків – програми розпізнавання тексту подають приклади тексту в різних шрифтах і форматах, які потім використовуються для порівняння та розпізнавання символів у відсканованому документі чи зображенні.

Коли символ ідентифікований, він перетворюється в код ASCII, який може використовуватися комп'ютерними системами для подальших маніпуляцій.

2.3 Постобробка

Постобробка є останнім кроком, який виконується в серії етапів обробки OCR. Головною його метою є виявлення та виправлення мовних помилок у вихідному тексті OCR після того, як вхідне зображення було відскановано та повністю оброблено [25]. По суті, існує два типи помилок розпізнавання тексту: помилка неіснуючого слова (non-word errors) і помилка існуючого слова (real-word errors). Помилка в написанні неіснуючого слова – це слово, яке розпізнається системою OCR, однак він не відповідає жодному запису в лексиконі. Наприклад, коли «Гарна погода» розпізнається системою OCR, як «Гярня погода», тоді кажуть, що «Гярня» є помилкою неіснуючого слова, оскільки слова «Гярня» не існує в українській мові. На відміну від цього, помилка в написанні існуючого слова — це слово, яке розпізнається системою OCR і відповідає запису в лексиконі, хоча воно граматично неправильне щодо речення, в якому воно відбулося. Наприклад, коли «Як пройшов твій день» розпізнається системою OCR як «Як пройшов твого день», тоді «твого» вважається помилкою реального слова, оскільки «твого» хоча синтаксично правильне (таке слово існує в українській мові), його використання в реченні

граматично неправильне. Як правило, помилки існуючих та неіснуючих слів підпадають під три класи помилок:

– помилки видалення;

– помилки вставки;

– помилки заміни.

Помилка видалення виникає, коли один або кілька символів відкидається або видаляється зсередини оригінального слова. Наприклад, неправильне розпізнавання слова «Телефон» як «Тлефон», «Телфон», «Тлфон» і т.д.

Помилка вставки виникає, коли до вихідного слова додається один або кілька додаткових символів. Наприклад, неправильне розпізнавання слова «Телефон» як «Теелефон» [26].

Помилка заміни виникає, коли один або кілька символів випадково замінюються в оригінальному слові, наприклад зміна символу «ф» у «Телефон» на «д».

Поганий стан паперів, що обробляються, є однією з найпопулярнішою причиною неточності або некоректності роботи OCR. Тому для виявлення та виправлення помилок OCR було запропоновано незліченну кількість підходів і алгоритмів постобробки. В цілому їх можна розділити на три великі категорії:

– ручне виправлення помилок;

– виправлення помилок на основі словника;

– виправлення помилок на основі контексту.

Інтуїтивно, найпростіший спосіб виправити помилки після OCR – це найняти групу людей, які будуть редагувати текст після обробки вручну. Він вимагає безперервного ручного втручання людини. Також такий спосіб вважається схильним до помилок, оскільки люди можуть ненавмисно допускати ряд з них. Крім того, ручна корекція в деякій мірі розглядається як трудомістка та дорога практика [27].

У невпинних зусиллях знайти спосіб краще виявляти та виправляти слова з помилками в тексті OCR, дослідники задумали методологію

виправлення помилок на основі словника, також відому як виправлення лексичних помилок. У цьому підході використовується лексикон або пошуковий словник для перевірки орфографії розпізнаних слів, які він потім виправляє, якщо слова написані з помилкою [28].

Існує кілька нетривіальних алгоритмів виправлення помилок на основі словника, одним з яких є алгоритм зіставлення рядків, який зважує слова в тексті за допомогою метрики відстані, що представляє різні витрати. Кандидат на виправлення з найменшою відстанню щодо слова з помилкою найкраще підходить як виправлення. Інший алгоритм продемонстрував, що використання синтаксичних властивостей мови та N-грами може прискорити процес генерування кандидатів на корекцію та, в кінцевому рахунку, підбору найкращого відповідного кандидата. Сад Джуро запропонував метод виправлення помилок, заснований на навчанні шаблонів, при якому спочатку генерується список кандидатів на виправлення лексичних помилок, потім вибирається найбільш ймовірний з них на основі словникового запасу та граматичних характеристик, що оточують слово з помилкою [29].

Гіпотетично, методи виправлення помилок на основі словника є досить успішними. Однак вони не можуть виправляти помилки на основі контексту слова в реченні. З іншого боку, методи на основі контексту на це здатні. Вони здійснюють виявлення та виправлення слів с помилками на основі їх граматичного та семантичного контексту. Це дозволило б вирішити наступну дилему виправлення помилок реальних слів, наприклад, у реченні «Як ти день», оскільки відповідно до контексту, в якому зустрічається «ти», навряд чи буде особистий займенник після іменника, скоріше, швидше за все мати присвійний займенник, за яким стоїть іменник [30].

3 КОМП'ЮТЕРНА МОДЕЛЬ ТЕЛЕГРАМ-БОТА ДЛЯ РОЗПІЗНАВАННЯ ТЕКСТУ НА ЗОБРАЖЕННЯХ ТА ЙОГО КОНВЕРТУВАННЯ

3.1 Архітектура застосунку

У рамках роботи було виконано декомпозицію задачі, в результаті якої було прийнято рішення розбити застосунок на 4 окремих сервіси (рис. 3.1). Задача кожного з них – чітко виконувати свою роботу і нічого більше. Такий підхід дозволить легко робіти зміни та інтеграції інших сервісів й функцій у застосунок у майбутньому легше та швидше. Також така архітектура дозволяє обирати найбільш підходящі технології для вирішення різних типів задач та не обмежує нас у виборі мови програмування або фреймворку.

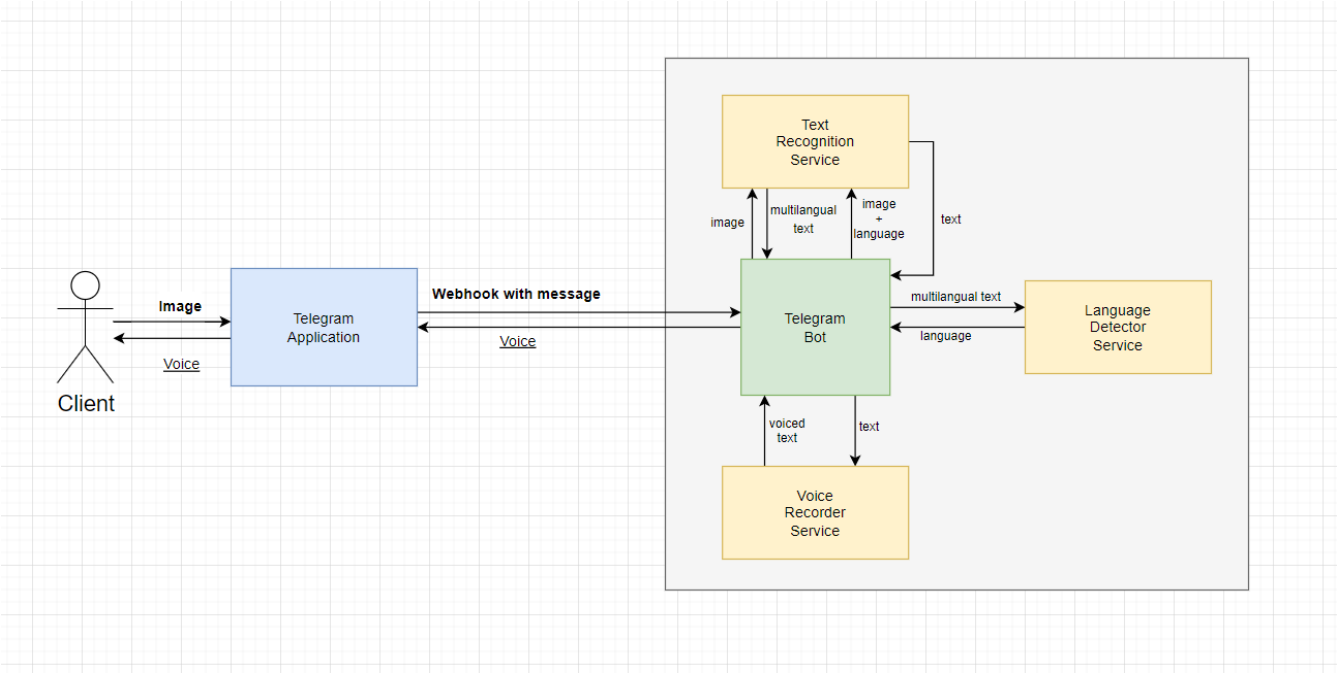

Рисунок 3.1 – Сервіси застосунку з конвертації тексту з зображення у текстовий та аудіо формати

До набору сервісів, з яких складається застосунок відносяться:

– телеграм-бот: відповідає за обробку повідомлень від користувача та виступає у якості оркестратора, який делегує підзадачі до інших сервісів;

– сервіс із розпізнавання тексту на зображеннях: відповідає за розпізнавання тексту з зображення;

– сервіс з визначення мови: відповідає за визначення найбільш вірогідної мови у тексті;

– сервіс з озвучення тексту: відповідає за створення аудіо файлу та озвучення тексту.

Даний підхід має назву «Мікросервісна архітектура». Це архітектурний стиль розробки застосунків, який дозволяє розділяти велику програму на менші незалежні частини, причому кожна частина має свою власну сферу відповідальності. Для того, щоб обробити один запит користувача, застосунок з використанням мікросервісної архітектури може викликати багато внутрішніх мікросервісів для складання відповіді.

Застосунки, створені з використанням даної архітектури, мають наступні характеристики: невеликі за розміром, націленість на виконання однієї задачі та незалежність один від одного.

До плюсів даної архітектури можна віднести:

– незалежність від мови програмування та технологій, які використовуються у розробці;

– простота додавання нового функціоналу;

– стійкість до помилок;

– незалежне масштабування (наприклад, під час пікового навантаження можна збільшити кількість екземплярів одного з сервісів);

– простий у розгортанні;

– ефективне використання ресурсів.

Комунікація між сервісами буде відбуватися через протокол HTTP використовуючи REST підхід. REST – це архітектурний стиль, метою якого є підвищення продуктивності, масштабованості, простоти, портативності та надійності системи. Це досягається завдяки дотриманню принципів REST:

– клієнт-сервер;
– відсутність стану;
– кешування;
– однорідний інтерфейс;
– шари абстракції;
– запитування коду.

3.2 Обґрунтування вибору технологій для реалізації сервісів

Наступним кроком після декомпозиції заді було визначення технологій та мов програмування для написання сервісів.

У якості програмного засобу для написання нашого застосунку було прийнято рішення використовувати IntelliJ IDEA. Це одна з найпопулярніших інтегрованих середовищ розробки, яка використовується для написання коду на багатьох мов програмування (Java, Python, Kotlin тощо). Серед особливостей IntelliJ IDEA можна віднести наступне:

– допомога в кодуванні (автозаповнення коду, зручна навігація по коду);
– вбудована інтеграція для багатьох інструментів та технологій (Gradle, Maven, Git);
– можливість розширення базового функціоналу (екосистема плагінів);
– підтримка багатьох мов програмування.

У якості інструменту для збірки проекту буде використовуватись Gradle. Це інструмент автоматизації збірки для розробки багатомовного програмного забезпечення. Він контролює процес розробки від компіляції та пакування до тестування, розгортання та публікації. До списку мов програмування, що підтримується, входять: Java, Kotlin, Groovy, Scala, C++ та JavaScript. У нашому випадку ми будемо використовувати цей інструмент для збірки Java

проектів. Також Gradle поширюється як програмне забезпечення з відкритим кодом за ліцензією Apache 2.0 і вперше був випущений у 2008 році.

3.2.1 Телеграм-бот

Існує велика кількість бібліотек для написання телеграм-ботів. У кожної з них є свої недоліки та переваги: деякі з них підтримують лімітовану кількість функцій телеграму, деякі доступні лише на 1 мові програмування тощо. У якості такої бібліотеки було прийнято рішення використовувати «telegrambots» (рис. 3.2), яка є офіційною бібліотекою від розробників месенджеру Telegram та яка має велику спільноту серед розробників, та підтримується багатьма мовами програмування (Java, Python, JavaScript і т.д.). Вона дає доступ до більшості функцій телеграму, серед який: обробка повідомлені від користувача, відправка файлів різних форматів та розмірів тощо.

```
dependencies {
    implementation group: 'org.springframework.cloud', name: 'spring-cloud-starter-openfeign', version: "${version_openfeign}"
    implementation group: 'org.telegram', name: 'telegrambots', version: "${version_telegrambots}"
    implementation group: 'org.apache.commons', name: 'commons-lang3', version: "${version_apache_commons_lang3}"
}
```

Рисунок 3.2 – Використання бібліотеки «telegrambots»

У якості мови програмування для написання боту було обрано Java, так як вона чудово підходить для написання серверних застосунків та надає гарну продуктивність. Також вона використовує строгу типізацію, що значно зменшує ймовірність появи багів.

У якості фреймворку для створення сервісу було обрано Spring Boot. Це є однією з найпопулярніших бібліотек мови Java, яка має підтримку до інтеграції з багатьма технологіями та з під коробки використовую Tomcat server для розвертання застосунку.

3.2.2 Сервіс із розпізнавання тексту на зображеннях

У якості програмного засобу для розпізнавання тексту на зображеннях було прийнято рішення використовувати Tesseract OCR. Tesseract – це двигун оптичного розпізнавання символів, який є однією з найпопулярніший OCR-бібліотек. Він використовує нейронні мережі для пошуку та розпізнавання тексту на зображеннях. Tesseract шукає шаблони в пікселях, літерах, словах та реченнях, використовує двоетапний підхід, який також називається адаптивним розпізнаванням. Потрібна 1 ітерація за даними для розпізнавання символів, потім друга ітерація для заповнення літер, в яких він не був впевнений, літерами, які, швидше за все, відповідають даному слову або контексту речення.

У якості мови програмування для написання сервісу буде також використовуватись Java. Через той факт, що Tesseract має лише консольний інтерфейс для користувача, сервіс буде взаємодіяти с цим двигуном через оболонку, яка написана на мові програмування Java та має назву «tess4j» (рис. 3.3).

У якості фреймворку для написання сервісів було також обрано Spring Boot.

```
dependencies {
    implementation group: 'org.springframework.boot', name: 'spring-boot-starter-aop'
    implementation group: 'org.springframework.boot', name: 'spring-boot-starter-cache'
    implementation group: 'com.github.ben-manes.caffeine', name: 'caffeine', version: "${version_caffeine}"
    implementation group: 'net.sourceforge.tess4j', name: 'tess4j', version: "${version_tess4j}"
}
```

Рисунок 3.3 – Використання бібліотеки «tess4j»

3.2.3 Сервіс з визначення мови

У якості інструменту з визначення мови спочатку планувалось використовувати якийсь безкоштовний сервіс, але майже всі з них мають обмеження на кількість перекладів у безкоштовній версії й також обмеження на кількість запитів у секунду. Переглянувши доступні бібліотеки на різних мовах програмування було помічено, що Python однозначно маж перевагу у цьому плані, порівнюючи, наприклад, з Java. Бібліотеки на Python надають більш простий інтерфейс для взаємодії з бібліотекою і також мають кращу продуктивність ніж бібліотеки на Java.

Виходячи з цього, було прийнято рішення використовувати Python, як основну мову програмування для написання цього сервісу, а також бібліотеку «polyglot», яка має найкращу продуктивність, порівнюючи її з іншими бібліотеками (рис. 3.4).

Library	Time
polyglot	3.67 s
fasttext	6.41
cld3	14 s
langid	1min 8s
langdetect	2min 53s
chardet	4min 36s

Рисунок 3.4 – Порівняння бібліотек для ідентифікації мови

У якості фреймворку для написання сервісу з визначення мови було обрано Flask. Це фреймворк для створення веб-застосунків мовою програмування Python, що використовує набір інструментів Werkzeug, а також

шаблонізатор Jinja2. Належить до категорії так званих мікрофреймворків – мінімалістичних каркасів веб-застосунків, які свідомо надають лише самі базові можливості.

3.2.4 Сервіс з озвучення тексту

У якості мови програмування для написання сервісу, була обрана мова Python, на якій написана велика кількість бібліотек для озвучення тексту.

Для озвучення тексту було прийнято рішення використовувати бібліотеку під назвою «gTTS». Google Text-to-Speech – це бібліотека, яка написана на мові Python, взаємодіє з API синтезу мовлення від Google Translate.

У якості фреймворку для написання сервісу з озвучення тексту було також обрано Flask.

3.3 Інструкція користувача

Для того, щоб почати користуватися телеграм-ботом, потрібно завантажити застосунок під назвою «Telegram». Зробити це можна на будь-якій платформі: Windows, MacOS, Android, IOS або Linux.

Виконавши цей крок, потрібно написати у пошуку месенджера наступний текст: «Image text converter» (рис. 3.5).

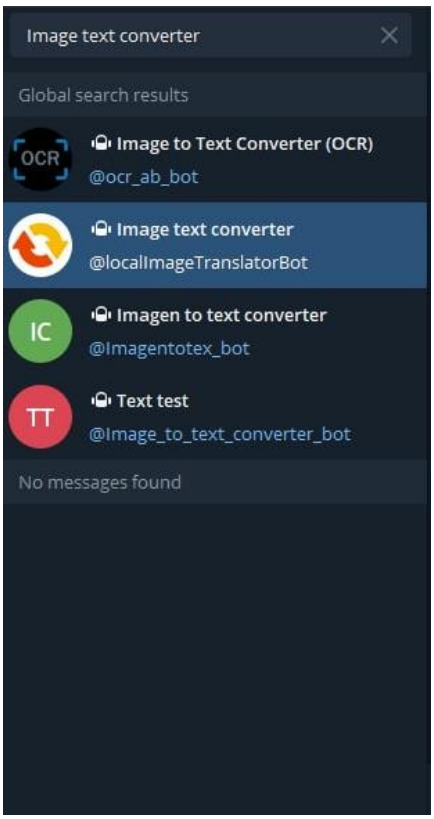

Рисунок 3.5 – Приклад пошуку телеграм-бота для розпізнавання тексту серед інших ботів

У результаті пошуку ви повинні побачити декілька варіантів. Вам потрібно вибрати варіант з наступним логотипом (рис.3.6). Зробивши це, Телеграм відкриє чат з ботом для розпізнавання тексту і відобразить початкове привітання (рис. 3.7).

Рисунок 3.6 – Логотип телеграм-бота для розпізнавання тексту

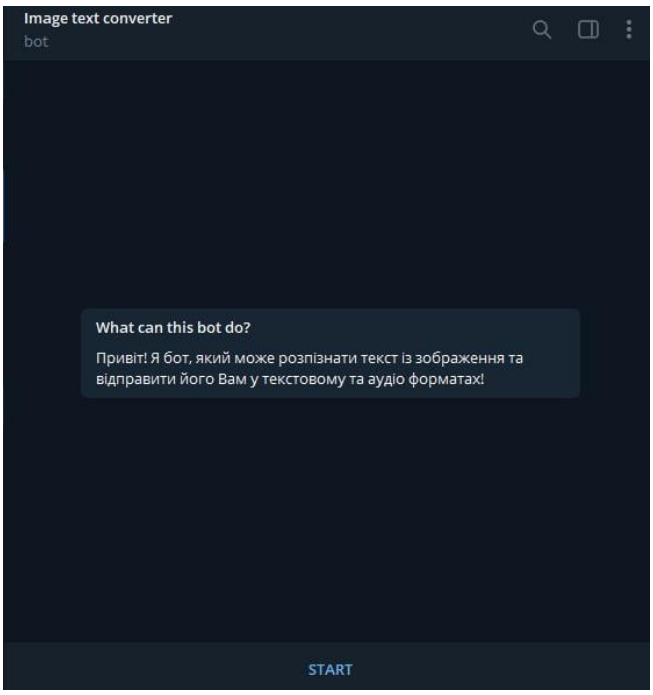

Рисунок 3.7 – Початкове привітання телеграм-бота для розпізнавання тексту

Для того, щоб телеграм-бот почав розпізнавати текст з вашого зображення, натисніть на кнопку «START», після чого боту можна буде відправляти зображення для конвертації (рис. 3.8).

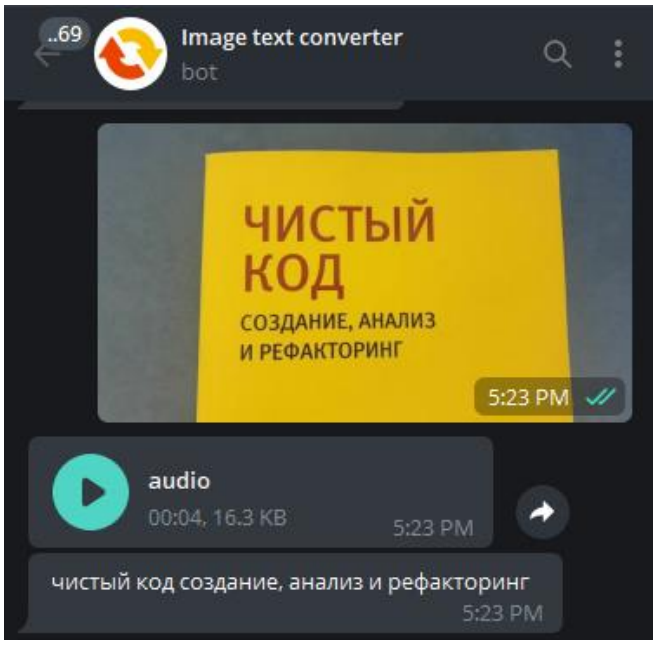

Рисунок 3.8 – Приклад використання телеграм-бота для розпізнавання тексту

3.4 Тестування розробленої моделі

До списку обмежень застосунку відносяться:

– підтримка тільки стиснених зображень;

– підтримка наступних мов: українська, російська та англійська.

У якості тестування було розглянуто 4 сценарії, які демонструють більшість можливих сценаріїв використання телеграм-бота:

– обробка скріншота зображення з текстом;

– обробка зображення з текстом, зробленого на камеру, з рівномірним освітленням;

– обробка зображення з текстом, зробленого на камеру, з нерівномірним освітленням;

– обробка зображення без тексту.

У якості результату першого сценарію було отримано оцифрований текст у аудіо та текстовому форматах, який повністю співпадає з тим, що є на зображенні (рис. 3.9).

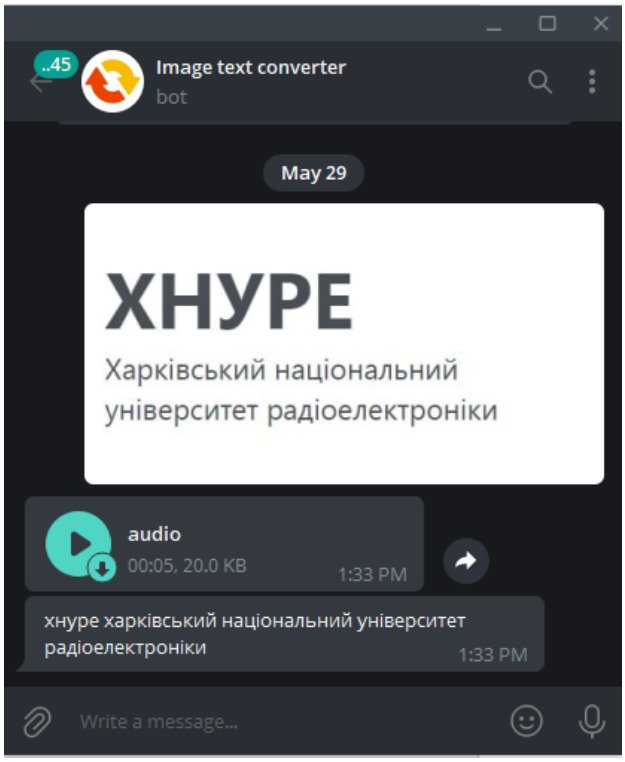

Рисунок 3.9 – Скріншот зображення з надрукованим текстом

Результатом другого сценарію також став оцифрований текст у аудіо та текстовому форматах, який повністю співпадає з текстом на зображенні (рис. 3.10).

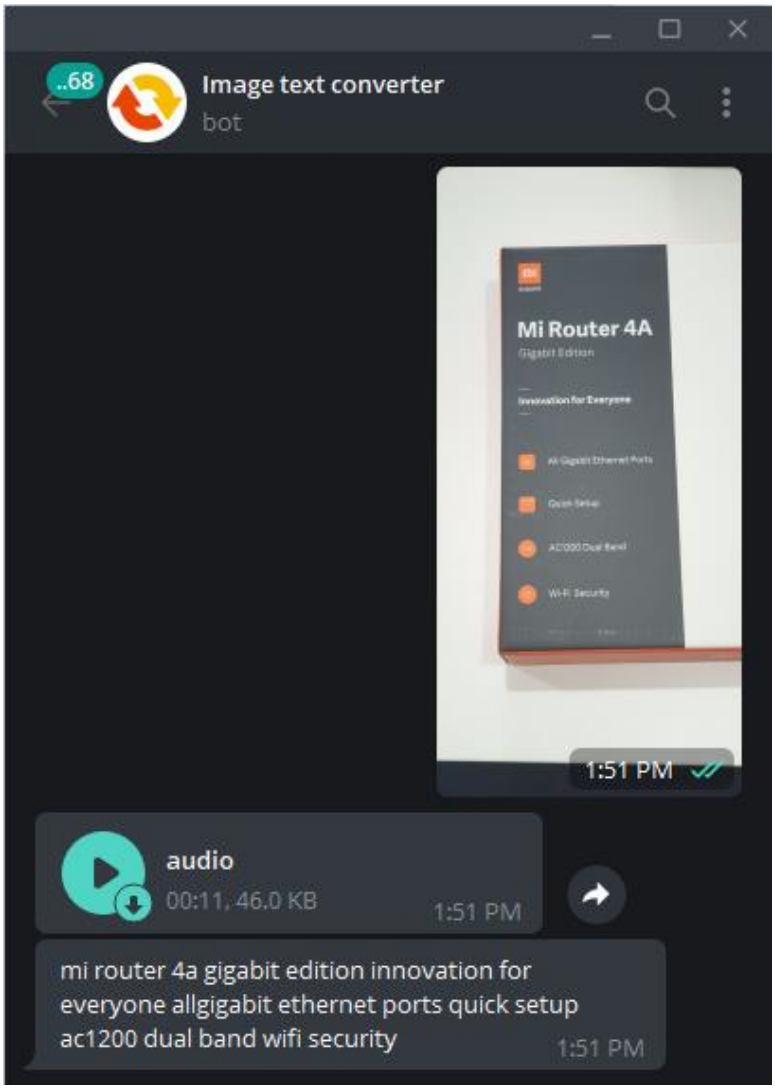

Рисунок 3.10 – Зображення з камери з рівномірним освітленням

У якості результату третього сценарію було отримано оцифрований текст, на якому присутні деякі артефакти («коодд» замість «код»), але, незважаючи на це, більшість тексту було розпізнана коректно (рис. 3.11).

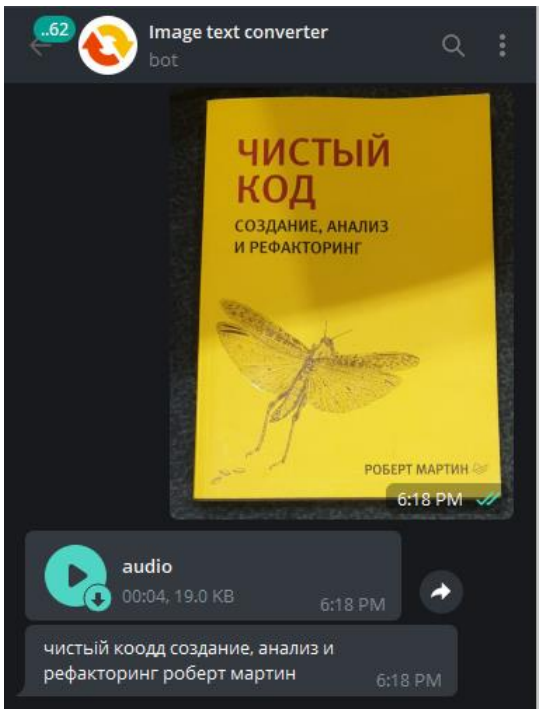

Рисунок 3.11 – Зображення з камери з нерівномірним освітленням

У якості результату четвертого сценарію було отримане повідомлення про те, що застосунок не може розпізнати текст з зображення (рис. 3.12). Такий результат роботи телеграм-бота є логічним, бо на зображенні дійсно його немає.

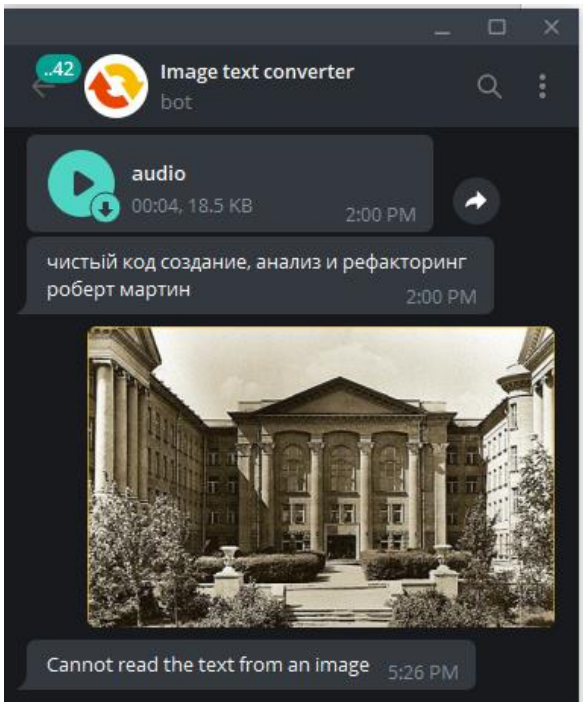

Рисунок 3.12 – Зображення без тексту

ВИСНОВКИ

У рамках кваліфікаційної роботи була здійснена програмна реалізація застосунку для розпізнавання тексту з зображень, ідентифікації мови цього тексту та його озвучення на ідентифікованій мові у вигляді телеграм-бота. Було проведено дослідження за допомогою бібліотек Tess4j для розпізнавання тексту, Polyglot для ідентифікації мови та Gtts для озвучення тексту.

Результатом проведеного дослідження є виконання всіх поставлених цілей:

– проведений аналіз існуючих підходів до розпізнавання тексту;

– створений алгоритм для розпізнавання тексту та його конвертацію у аудіо формат;

– проведений аналіз існуючих підходів та бібліотек щодо синтезу мовлення;

– створена реалізація телеграм-бота із використанням вище вказаного алгоритму.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Kinoshenko, D., Mashtalir, V., Yegorova, E., & Vinarsky, V. (2005, July). Hierarchical partitions for content image retrieval from large-scale database. In International Workshop on Machine Learning and Data Mining in Pattern Recognition (pp. 445-455). Springer, Berlin, Heidelberg.

2. Путятін, Є. П., Гороховатський, В. О., & Матат, О. О. (2006). Методи та алгоритми комп'ютерного зору: навч. посібник.

3. Творошенко, І. С. (2021). Технології прийняття рішень в інформаційних системах: навч. посібник. *Харків: ХНУРЕ*.

4. Гороховатський, В. О., & Творошенко, І. С. (2021). Методи інтелектуального аналізу та оброблення даних: навч. посібник.

5. Кобилін, О. А., & Творошенко, І. С. (2021). Методи цифрової обробки зображень: навч. посібник. *Харків: ХНУРЕ*.

6. Kinoshenko, D., Mashtalir, V., & Shlyakhov, V. (2007). A partition metric for clustering features analysis.

7. Kinoshenko, D., Mashtalir, V., & Yegorova, E. (2006). CLUSTERING METHOD FOR FAST CONTENTBASED IMAGE RETRIEVAL. In Computer Vision and Graphics (pp. 946-952). Springer, Dordrecht.

8. Kobylin O., Gorokhovatskyi V., Tvoroshenko I., and Peredrii O. (2020) The application of non-parametric statistics methods in image classifiers based on structural description components, Telecommunications and Radio Engineering, 79(10), pp. 855-863.

9. Daradkeh, Y.I., Tvoroshenko, I., Gorokhovatskyi, V., Latiff, L.A., and Ahmad, N. (2021) Development of Effective Methods for Structural Image Recognition Using the Principles of Data Granulation and Apparatus of Fuzzy Logic, IEEE Access, 9, pp. 13417-13428.

10. Daradkeh, Y.I., Gorokhovatskyi, V., Tvoroshenko, I., Gadetska, S., and Al-Dhaifallah, M. (2021) Methods of Classification of Images on the Basis of the

Values of Statistical Distributions for the Composition of Structural Description Components, IEEE Access, 9, pp. 92964-92973.

11. Gorokhovatskyi, V.O., Tvoroshenko, I.S., and Peredrii O.O. (2020) Image classification method modification based on model of logic processing of bit description weights vector, Telecommunications and Radio Engineering, 79(1), pp. 59-69.

12. Gorokhovatskyi, V., Gorokhovatskyi, O., Yevgenyi, P., & Olena, P. (2018). Quantization of the Space of Structural Image Features as a Way to.

13. Tvoroshenko, I., & Kukharchuk, V. (2021). Current state of development of applications for recognition of faces in the image and frames of video captures.

14. The flow of improved BRISK algorithm. URL: https://www.researchgate.net/figure/the-flow-of-improved-BRISK-algorithm_fig1_328946366 (дата звернення 24.04.2022).

15. Bodyanskiy, Y., Kinoshenko, D., Mashtalir, S., & Mikhnova, O. (2012). On-line video segmentation using methods of fault detection in multidimensional time sequences. International Journal of Electronic Commerce Studies, 3(1), 1-20.

16. Chupikov, A., Kinoshenko, D., Mashtalir, V., & Shcherbinin, K. (2006, July). Image retrieval with segmentation-based query. In International Workshop on Adaptive Multimedia Retrieval (pp. 207-221). Springer, Berlin, Heidelberg.

17. Kinoshenko, D., Mashtalir, V., Orlov, A., & Yegorova, E. (2003, September). Method of creating of functional invariants under one-parameter geometric image transformations. In Joint Pattern Recognition Symposium (pp. 68-75). Springer, Berlin, Heidelberg.

18. Kinoshenko, D., Mashtalir, S., Stephan, A., & Vinarski, V. (1993). Neural Network Segmentation Of Video Via Time Series Analysis. INFORMATION THEORIES & APPLICATIONS, 232.

19. Kinoshenko, D., Mashtalir, V., Shlyakhov, V., & Yegorova, E. (2012). nested Partitions Properties for spatial content Image retrieval. In Multimedia Storage and Retrieval Innovations for Digital Library Systems (pp. 240-269). IGI Global.

20. Kinoshenko, D., Mashtalir, V., Shlyakhov, V., & Yegorova, E. (2011). Metrical properties of nested partitions for image retrieval. In Machine Learning Techniques for Adaptive Multimedia Retrieval: Technologies Applications and Perspectives (pp. 18-49). IGI Global.

21. Kinoshenko, D., Mashtalir, V., & Yegorova, E. (2010). Block-Diagonal Forms of Distance Matrices for Partition Based Image Retrieval. In Pattern Recognition Recent Advances. IntechOpen.

22. Rakun, J., Berk, P., Stajnko, D., Ocepek, M., & Lakota, M. (2012). Digital image processing approach to fingerprint authentication. DAAAM International Scientific Book, 517-527.

23. Warkari, K. D. S., & Kshirsagar, U. A. (2015). Design of Point Processing Algorithm using Hardware Co-simulation for Digital Image Processing Application. International Journal of Electronics, Communication and Soft Computing Science & Engineering (IJECSCSE), 4, 70.

24. Turrisi, S. (2017). Motion blur compensation to improve the accuracy of Digital Image Correlation measurements.

25. Costarelli, D., Seracini, M., & Vinti, G. (2020). A comparison between the sampling Kantorovich algorithm for digital image processing with some interpolation and quasi-interpolation methods. Applied Mathematics and Computation, 374, 125046.

26. Shinde, B. S., & Dani, A. R. (2012). The origins of digital image processing & application areas in digital image processing medical images. IOSR Journal of Engineering, 1(1), 066-071.

27. Fuchigami, A., Yoneyama, S., Goto, K., & Ishimura, K. (2022). Evaluation of Thermal Deformation of Fastening Structure of Carbon Fiber Reinforced Plastic and Aluminum Using Stereo Image Correlation. In Thermomechanics & Infrared Imaging, Inverse Problem Methodologies, Mechanics of Additive & Advanced Manufactured Materials, and Advancements in Optical Methods & Digital Image Correlation, Volume 4 (pp. 41-44). Springer, Cham.

28. MONCLOU CHAPARRO, J. E. N. N. I. F. E. R. (2017). A first approach to study the thermal annealing effect of an object made of poly-lactic acid (PLA) produced by fused deposition modeling (FDM) technology.

29. Xi, X., Wang, F., & Liu, Y. (2013, December). Improved Criminisi algorithm based on a new Priority Function with the gray entropy. In 2013 Ninth International Conference on Computational Intelligence and Security (pp. 214-218). IEEE.

30. Pal, K., Ghosh, G., & Bhattacharya, M. (2012, December). Retrieval of hidden infected region using biomedical image watermarking for tele-diagnosis to ensure better treatment. In 2012 5th International Conference on Computers and Devices for Communication (CODEC) (pp. 1-4). IEEE.