

Міністерство освіти і науки України  
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук  
(повна назва)

Кафедра Програмної інженерії  
(повна назва)

**АТЕСТАЦІЙНА РОБОТА**  
**Пояснювальна записка**

другий (магістерський)  
(рівень вищої освіти)

Дослідження методів розпізнавання образів для E-commerce систем  
(тема)

Виконав: студент 2 курсу, групи ІПЗм-17-2  
спеціальності 121- Інженерія програмного  
забезпечення  
(код і повна назва спеціальності)  
освітньо-професійної програми Інженерія  
програмного забезпечення

Гаврилов В.С.  
(прізвище, ініціали)

Керівник доц. Ревенчук І.А.  
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри, проф. 3.В.Дудар

2019 р.

Харківський національний університет радіоелектроніки

Факультет \_\_\_\_\_ Комп'ютерних наук \_\_\_\_\_

Кафедра \_\_\_\_\_ Програмної інженерії \_\_\_\_\_

Рівень вищої освіти \_\_\_\_\_ другий (магістерський) \_\_\_\_\_

Спеціальність \_\_\_\_\_ 121- Інженерія програмного забезпечення \_\_\_\_\_  
(код і повна назва)

Освітньо-наукова програма \_\_\_\_\_ Інженерія програмного забезпечення \_\_\_\_\_  
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри \_\_\_\_\_  
(підпис)

« \_\_\_\_ » \_\_\_\_\_ 20 \_\_\_\_ р.

**ЗАВДАННЯ**  
НА АТЕСТАЦІЙНУ РОБОТУ

студентові \_\_\_\_\_ Гаврилову Володимирі Сергійовичу \_\_\_\_\_  
(прізвище, ім'я, по батькові)

1. Тема роботи \_\_\_\_\_ Дослідження методів розпізнавання образів для E-commerce систем \_\_\_\_\_

затверджена наказом по університету від “ \_\_\_\_ ” \_\_\_\_\_ 20 \_\_\_\_ р № \_\_\_\_\_  
заповнюється вручну після отримання наказу

2. Термін подання студентом роботи до екзаменаційної комісії \_\_\_\_\_ 201 \_\_\_\_ р.

3. Вихідні дані до роботи методи аналізу зображень, пояснювальна записка, програмна система для розпізнавання образів в системах E-commerce. Використовувати ОС Windows, середовище об'єктно-орієнтованого проектування

4. Перелік питань, що потрібно опрацювати в роботі мета роботи, аналіз проблемної галузі і постановка задачі, огляд методів машинного зору, використання машинного навчання і розпізнавання образів, реалізація програмної системи

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) діаграма послідовностей, use-case діаграма, діаграма класів, схеми архітектури системи, інтерфейс системи, слайди презентації

## 6. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спецчастина	доц. Ревенчук І.А.		

## КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз проблемної області		
2	Розробка моделі предметної галузі		
3	Розробка структури зберігання даних		
4	Створення коду програми		
5	Тестування і налагодження програми		
6	Підготовка пояснювальної записки		
7	Підготовка презентації та доповіді		
8	Попередній захист		
9	Нормоконтроль, рецензування		
10	Занесення диплома в електронний архів		
11	Допуск до захисту у зав. кафедри		

Дата видачі завдання \_\_\_\_\_ 2019р.

Студент гр.ІПЗм-17-2 \_\_\_\_\_ Гаврилов В.С.

Керівник роботи \_\_\_\_\_ доц. Ревенчук І.А.

## РЕФЕРАТ / ABSTRACT

Пояснювальна записка до атестаційної роботи: 70 с., 39 рис., 2 табл., 1 додаток, 22 джерела.

### ПРОГРАМНА СИСТЕМА, ПРОГРАМНИЙ ПРОДУКТ, РОЗПІЗНАВАННЯ ОБРАЗІВ, ЕЛЕКТРОННА ТОРГІВЛЯ

Об'єкт – методи розпізнавання образів, програми машинного зору, системи машинного навчання.

Мета роботи – проектування та створення програмної системи, яка дозволяє розпізнавати та ідентифікувати товар за його зображенням, визначати його категорію та знаходити за артикулом в електронних магазинах.

В роботі проведений аналіз предметної галузі, досліджені основні методи та фреймворки машинного зору, проведений аналіз актуальних технологій, розроблена програма із використанням знань, набутих під час дослідження.

### SOFTWARE SYSTEM, SOFTWARE PRODUCT, PATTERN RECOGNITION, E-COMMERCE

The object of the development are pattern recognition methods, computer vision software, machine learning systems.

The aim of the task is to design and create a software system that allows you to recognize and identify the product by its image, define its category and find it by stock keeping unit in electronic stores.

As a result, the analysis of the subject area was carried out, the basic methods and computer vision frameworks were investigated, the analysis of the current technologies was conducted, a program with the use of knowledge gained during the research was developed.

## ЗМІСТ

Вступ.....	6
1 Аналіз предметної галузі та постановка задачі .....	9
1.1 Історія та розвиток комп'ютерного розпізнавання образів .....	9
1.2 Огляд існуючих програм розпізнавання образів .....	11
1.3 Фреймфорки для машинного навчання.....	15
1.4 Аналіз існуючих бібліотек машинного зору.....	18
1.5 Постановка задачі.....	21
2 Опис досліджень та прийнятих програмних рішень .....	22
2.1 Аналіз підходів з порівняння ознак об'єктів на зображенні.....	22
2.2 SIFT (Scale Invariant Feature Transform) .....	23
2.3 SURF (Speeded up Robust Feature) .....	25
2.4 Відмінності між SIFT та SURF .....	29
2.5 Результати порівняльного аналізу .....	30
2.6 Вибір методів та технологій для аналізу зображень .....	38
3 Проектування програмної системи .....	42
3.1 UML проектування програмної системи .....	42
3.2 Архітектура веб-додатку.....	46
4 Опис програмної реалізації.....	48
4.1 Вибір засобів розробки .....	48
4.2 Опис програмної системи .....	51
Висновки.....	59
Перелік джерел посилання .....	61
Додаток А Слайди презентації .....	63

## ВСТУП

Однією з найбільш характерних здібностей людського зору є швидке впізнавання знайомих об'єктів, речей, людей, зокрема - розпізнавання знайомих предметів за їхніми зображеннями. Уміння розпізнавати образи формується у людини з дитинства. В останній час все популярнішою стає ідея надання таких можливостей програмним системам.

Машинний зір, що представляє собою порівняно молоду галузь, вже сьогодні має дуже велике поширення у всіх без винятку сферах життєдіяльності людини: від систем обробки текстових друкованих документів і аж до визначення осіб людини і жестів.

Область дослідження в сфері комп'ютерного зору досить різноманітна і динамічно розвивається. Багато з розроблюваних методів і способів обробки зображень досі перебувають на стадії дослідження, хоча все частіше деякі цих методів успішно знаходять своє застосування в комерційних продуктах, де вони часто складають частину більшої системи, здатної вирішувати складні завдання. У більшості практичних алгоритмів застосувань технічного зору комп'ютери запрограмовані для вирішення певного кола завдань, але ті методи, які засновані на знаннях, стають все більш загальними.

Компанії в різних секторах, такі як електронна комерція, автомобільна промисловість, охорона здоров'я та цифрові ігри, швидко впроваджують розпізнавання зображень.

Згідно з доповіддю MarketsandMarkets, ринок розпізнавання зображень поділяється на апаратні засоби, програмне забезпечення та послуги. Апаратний сегмент, в якому переважають смартфони і сканери, може зіграти величезну роль у зростанні ринку розпізнавання зображень.

Розпізнавання зображень відноситься до технологій, які визначають місця, логотипи, людей, об'єкти, будівлі та кілька інших змінних у зображеннях. Користувачі обмінюються великою кількістю даних через програми, соціальні

мережі та веб-сайти. Крім того, мобільні телефони, оснащені камерами, ведуть до створення безмежної кількості цифрових зображень і відео. Великий обсяг цифрових даних використовується компаніями для забезпечення кращих і розумніших послуг для людей, які мають доступ до неї.

Розпізнавання зображень є частиною комп'ютерного зору та процесу ідентифікації та виявлення об'єкта або атрибута в цифровому відео або зображенні. Комп'ютерне бачення - це більш широкий термін, який включає методи збору, обробки та аналізу даних з реального світу. Дані є високовимірними і дають чисельну або символічну інформацію у вигляді рішень.

Крім розпізнавання зображень, комп'ютерне бачення також включає виявлення подій, розпізнавання об'єктів, навчання, реконструкцію зображень та відеоспостереження.

В час високої популярності E-commerce[1] систем програми комп'ютерного зору отримали ще один напрям розвитку. E-commerce - це електронна комерція, тобто особлива галузь економіки, в яку входять торговельні та фінансові операції, що здійснюються з використанням комп'ютерних мереж, а також бізнес-процеси, що проводяться за допомогою таких угод.

Відомо, що останнім часом все більша кількість угод виконується дистанційно, за допомогою електронних пристроїв. Це викликано тим, що електронні платежі робити зручніше і швидше, ніж платежі готівкою.

Все більше число людей воліють обирати та придбати товар дистанційно. Але часто виникає ситуація, коли важко ідентифікувати необхідний товар, маючи лише його зображення. В таких ситуаціях було б дуже доцільно скористуватися відповідною програмою для розпізнавання предметів за їх образом.

В якості основної технології при розробці програмного забезпечення для розпізнавання образів повинна виступати одна з наявних технологій розпізнавання зображень. Є велика кількість бібліотек для цієї технології. Кожна з них має свої недоліки і переваги.

Необхідно досягти гнучкості та високої точності розпізнавання, адже електронна комерція охоплює нескінченну кількість продукції всіх видів. А із

урахуванням постійно зростаючої кількості нової продукції, програма підтримує розширюваний набір оброблюваних даних та включає в себе нейронну мережу для класифікації об'єктів на зображенні.

Все вищесказане визначило актуальність теми роботи - використання найбільш придатного методу в задачах розпізнавання образів для E-commerce систем.

Отже, тема та засоби для виконання роботи є актуальними. Метою роботи є проектування та створення програмної системи, яка дозволяє розпізнавати та ідентифікувати товар за його зображенням, визначати його категорію та знаходити за артикулом в електронних магазинах.



# 1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

## 1.1 Історія та розвиток комп'ютерного розпізнавання образів

Розпізнавання образів – одне найпоширеніших завдань, яке намагаються вирішувати в сучасних програмних системах та додатках. Можна стверджувати, машинне розпізнавання образів увійшло в повсякденне життя. Facebook розпізнає обличчя на фотографіях, Twitter блокує підозрілі фотографії без участі адміністраторів, OneDrive виявляє заборонений контент за допомогою технології PhotoDNA, а сервіси і додатки на базі API Microsoft Project Oxford однаково легко визначають по фотографії настрій людини або породи собак.

Рівень впровадження цієї технології є найвищим у сфері електронної комерції, включаючи пошук і рекламу. Розпізнавання зображень може перетворити смартфон у віртуальний каталог. Технологія розпізнавання із великим успіхом може бути використана в програмних додатках для ідентифікації конкретних продуктів. Це дає змогу більш інтерактивно взаємодіяти зі світом, роблячи все навколо придатним для пошуку.

Технологія дозволяє підвищити рівень електронної комерції. Завдяки функціям ідентифікування об'єктів потенційні покупці можуть здійснювати порівняння продуктів у реальному часі без відвідування веб-сайту.

Всі ці та безліч інших функцій комп'ютерного зору стали можливими завдяки напрямку штучного інтелекту, який отримав назву «глибинне навчання». Хоча ця технологія зараз в тренді, їй всього кілька років, і останні експерименти, проведені Microsoft Research, переконують, що вона прибуває в початковій стадії розвитку і не розкрила більшої частини своїх можливостей. Інакше кажучи, глибинному навчанню є куди рухатись.

Революція в машинному зорі назрівала протягом багатьох років. Поворотним пунктом став 2012 рік, коли дослідники штучного інтелекту з Університету Торонто виграли конкурс ImageNet - змагання машин в розпізнаванні образів. У конкурсі перемагає система, точніше інших ідентифікує будь-які об'єкти: котів,

автомобілі або хмари. Учасники команди, що перемогла (Алекс Крижевський (Alex Krizhevsky) і професор Джефф Хінтон (Geoff Hinton) використовували глибинні нейронні мережі - технологію розпізнавання образів, в основі якої не жорсткі алгоритми, закладені розробниками, а навчання системи, засноване на послідовній ідентифікації величезної кількості зображень.

Перемога команди Торонто відкрила перед глибинним навчанням нові перспективи. У наступні роки всі глобальні інтернет-компанії, включаючи Facebook, Google, Twitter і Microsoft, стали застосовувати схожі технології для побудови систем машинного зору. Скоро вони справлялися з розпізнаванням зображень не гірше, а часом і краще людей.

Нейронні мережі за допомогою апаратних і програмних засобів імітують павутину нейронів в людському мозку. Ця ідея з'явилася ще в 80-і роки, але в 2012 році Крижевський і Хінтон вдосконалили її реалізацію, побудувавши мережу, що використовує ресурси графічних процесорів (відеокарт). Провідні світові ІТ-компанії сьогодні будують на цих процесорах свої системи розпізнавання образів, що застосовуються для самих різних завдань: від пошуку картинок в Інтернеті - до забезпечення безпеки.

Створення пристроїв, які виконують функції розпізнавання різних об'єктів, у багатьох випадках здатне помітно полегшити людині життя. Такі вдосконалення дозволяють значно розширити можливості різних систем, що виконують складні інформаційно-логічні завдання. А з розвитком технологій все більшу популярність отримують різноманітні види E-commerce. Таким чином, більше 30% всіх онлайн-транзакцій здійснюються за допомогою мобільних пристроїв. І ця цифра продовжує зростати. Сучасні покупці надають перевагу покупкам зі своїх екранів, в першу чергу смартфонів.

Варто зауважити, що із появою на ринку величезної кількості товарів стало майже неможливо знати назву всіх найменувань та навіть видів цих товарів. В цій сфері значну допомогу можуть надати програми із технологією розпізнавання образів, що здатні до машинного навчання. Вони дозволять маючи лише

зображення предмета з легкістю його ідентифікувати, підказати його вартість або місце продажу.

## 1.2 Огляд існуючих програм розпізнавання образів

Як було зазначено вище обрана тема є актуальною на сьогодні і на близьке майбутнє. Щоб мати більше уявлення про роботу систем розпізнавання було виділено декілька програмних продуктів для розгляду.

Google Lens[2] – це додаток, що дозволяє знімати зображення за допомогою камери смартфона, а потім здійснювати пошук на основі зображень в Інтернеті. Він працює так само, як зворотний пошук Google зображень, пропонуючи користувачам посилання на сторінки, статті Вікіпедії та інші відповідні ресурси, пов'язані з зображенням. В основі цього додатку лежить нейронна мережа від Google та величезна база Google зображень.

Коли направляєте камеру телефону на об'єкт, Google Lens намагатиметься ідентифікувати об'єкт або прочитати ярлики та текст і показати відповідні результати пошуку та інформацію. Наприклад, при наведенні камери пристрою на мітку Wi-Fi, яка містить назву мережі та пароль, вона автоматично підключається до сканованого джерела Wi-Fi. На рисунку 1.1 показано, як програма розпізнає рослини, підключається до мережі за допомогою штрих-коду та визначає місця за вивісками.

Google Lens також інтегрований з додатками Google Photos і Google Assistant. Служба схожа на Google Goggles, попередню програму, яка функціонувала аналогічно, але з меншими можливостями.

Google Lens використовує просунуті програми глибокого навчання і набір інструментів аналізу зображень (доступних на Google Play). Останнім часом програма отримала можливість копіювати текст, здійснювати дзвінки на розпізнані

телефонні номери, ідентифікувати об'єкти, рослини, тварин, книги, фільми та орієнтири.

Додаток є досить потужним інструментом але не дуже добре розпізнає специфічні предмети, що рідко зустрічаються. Це в першу чергу пов'язано із тим, що користувачі не мають можливості власноруч розширювати набір даних Google Lens. Програма не здатна здійснювати пошук ідентифікованих товарів у онлайн магазинах, а також підтримується лише флагманськими смартфонами на базі Android із версією операційної системи 6.0 і вище.

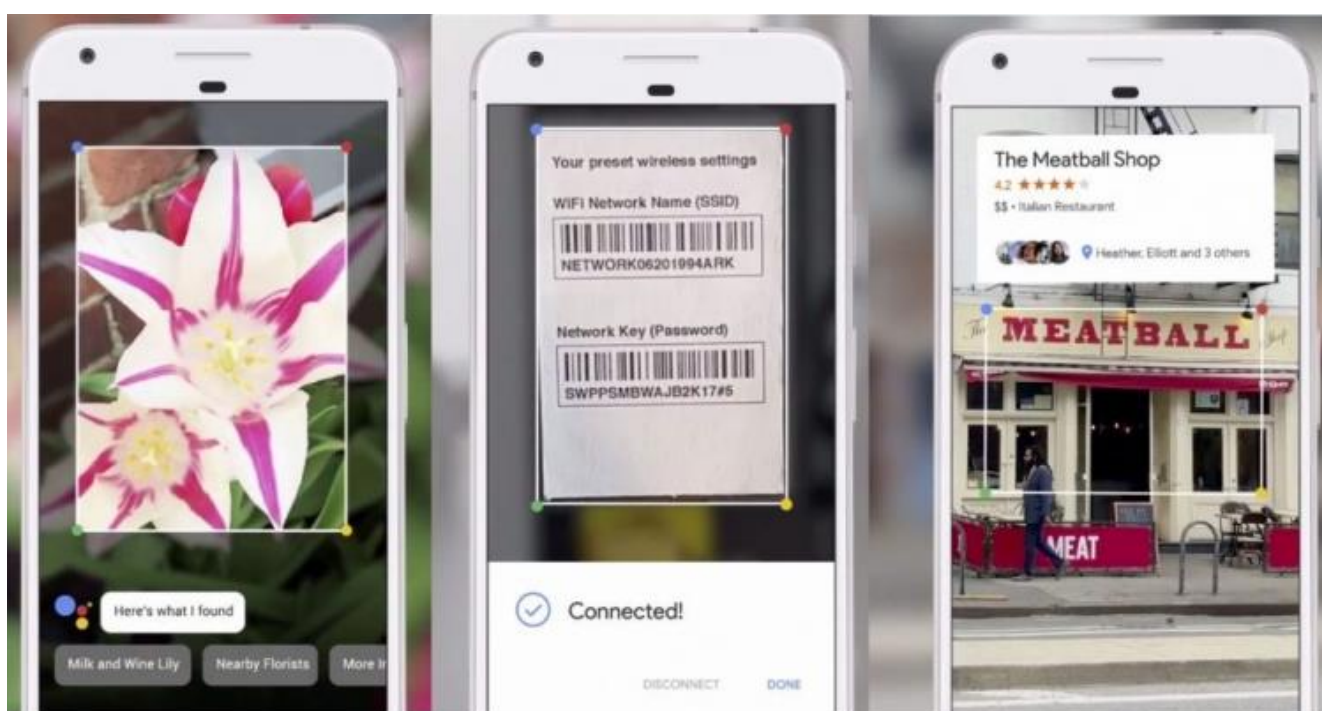


Рисунок 1.1 – Робота додатку Google Lens

Flow Powered by Amazon[3] - додаток із технологією доповненої реальності, який дозволяє шукати інформацію про елементи навколо.

Flow може визначити десятки мільйонів продуктів, включаючи книги, DVD-диски та упаковані предмети домашнього ужитку, такі як коробка пластивців або кава.

Додаток також вміє сканувати номери телефонів, URL-адреси, адреси електронної пошти та візитні картки, а потім швидко набирати, відкривати або додавати інформацію до ваших контактів.

Щоб розпочати роботу із додатком достатньо навести камеру на:

- обкладинки для книг;
- відео ігри;
- DVD-диски та компакт-диски;
- упаковані товари, такі як ігри та іграшки;
- коробка пластівців та ін.
- Flow також декодуватиме:
- штрих-коди UPC;
- QR-коди;
- номери телефонів, веб-адреси та адреси електронної пошти;
- візитні картки.

Приклад сканування товару за кодом представлено на рисунку 1.2.

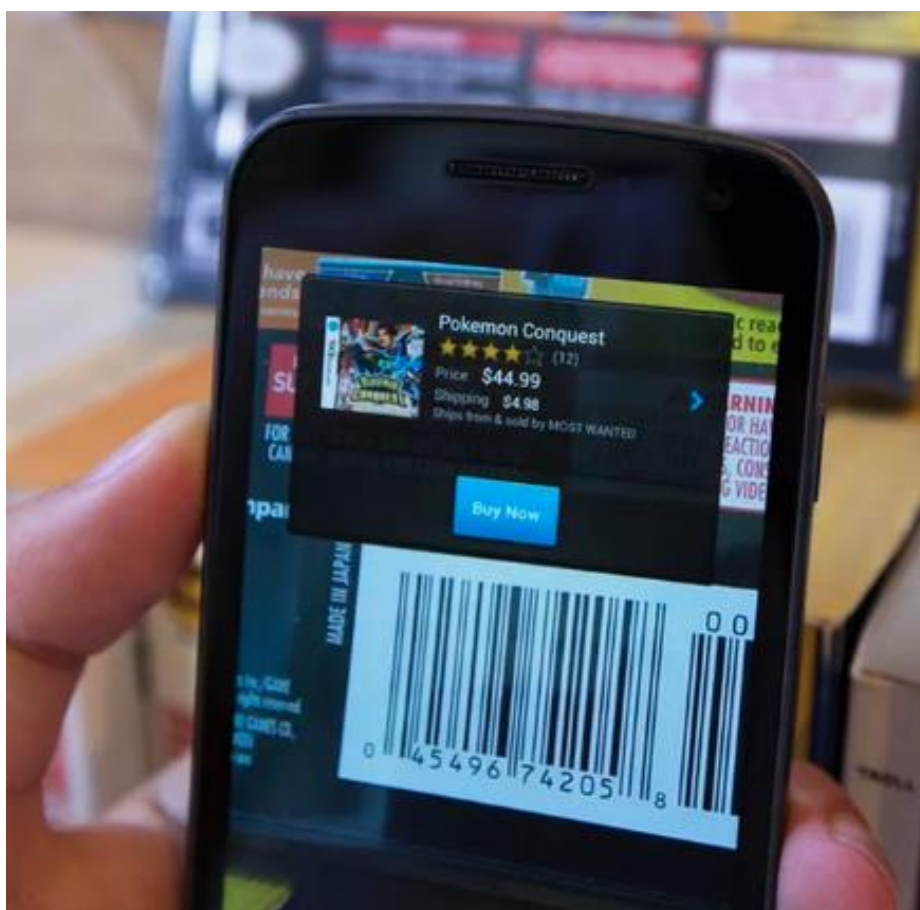


Рисунок 1.2 – Сканування за допомогою додатку Flow Powered by Amazon

Головним недоліком додатку є те, що повноцінне використання можливе лише на території Сполучених Штатів Америки, а товари шукаються лише в каталозі Amazon.

Vivino[4] - це найбільший в світі інтернет-ринок вина і найпопулярніша винна програма, створена спільнотою мільйонів користувачів.

Унікальний досвід Vivino для купівлі вина використовує дані спільноти, щоб запропонувати індивідуальні рекомендації щодо вина, зробити відкриття вина та його придбання веселим, доступним та легким для всіх винних любителів. Програма Vivino доступна для завантаження на пристроях Android і Apple. Станом на 2019 рік, Vivino має базу вин, що містить більше 10 мільйонів різних вин, і мав 35 мільйонів користувачів. У липні 2013 року в базі даних було 1 млн. вин.

Все, що потрібно - сфотографувати етикетку або завантажити готову фотографію в додаток. Програма проаналізує її і зіставити зі своєю базою даних (див. рис. 1.3). Якщо цього не відбулося автоматично, необхідні дані в каталог занесуть розробники.

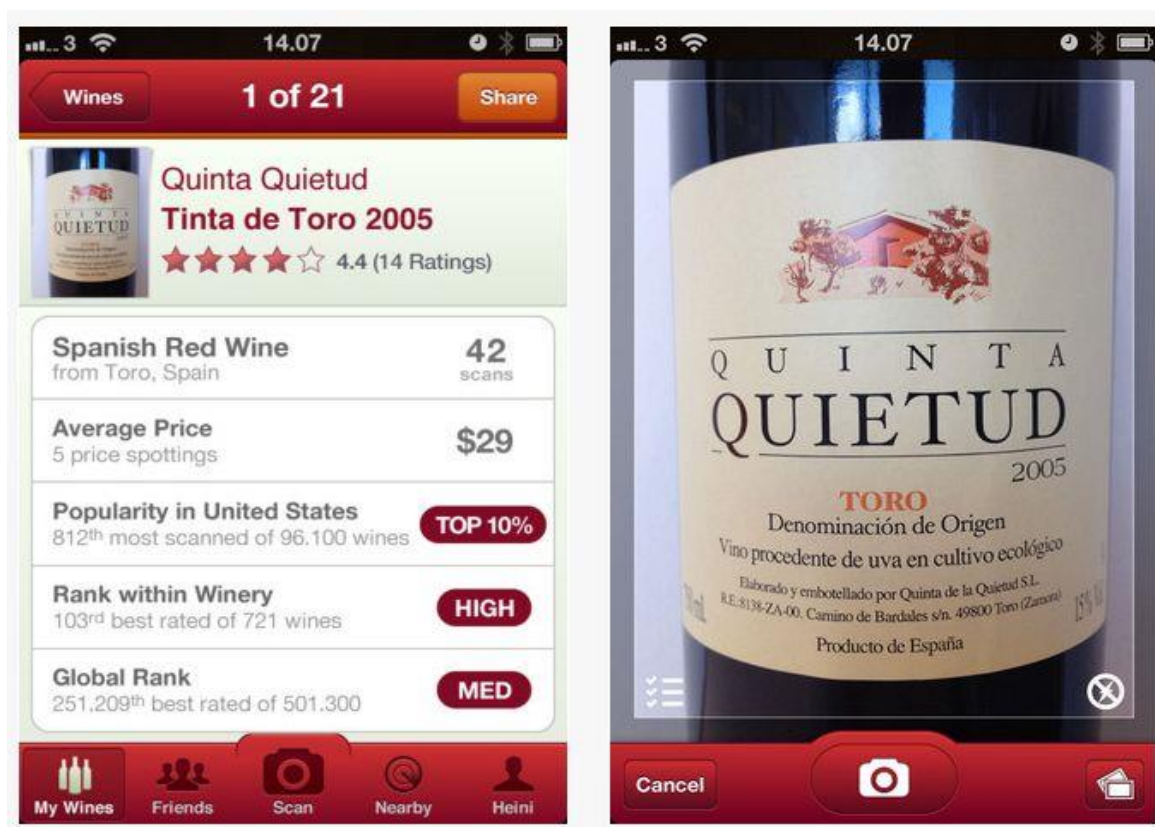


Рисунок 1.3 – Ідентифікація вина додатком Vivino

У додатку використовується гео-локація, тому користувач буде завжди проінформований, чи є в прилеглому супермаркеті або ресторані цікавляче його вино. Завдяки Vivino можна дізнатися вартість практично будь-якого вина.

Додаток розроблений на основі UGC(user-generated content), тобто дані в програму вносять користувачі, тому база даних Vivino, яка на сьогоднішній день містить вже понад 500 тисяч вин, майже 15 тисяч виноробів, 800 регіонів і 48 країн, стає дедалі більше.

З головних мінусів - розпізнавати і аналізувати можна тільки вина. Ніяких інших об'єктів програма не знаходить.

Жодна із сучасних програмних реалізацій не є ідеальною, у них є певні недоліки. До основних можна віднести недостатню підтримку у деяких країнах за рахунок того, що нейронні мережі не навчалися або не мали достатньої кількості даних. У вирішенні цієї проблеми допомогли б дані завантажені користувачами. Також багато з доступних на даний момент рішень прив'язані до баз конкретних магазинів або корпорацій, що сильно звужує коло пропозицій товарів.

### 1.3 Фреймворки для машинного навчання

Жодна сучасна система, націлена на розпізнавання великого різноманіття об'єктів, не обходиться без технологій машинного навчання, а саме глибинного навчання, що ґрунтується на навчанні ознак даних. Деякі з архітектур глибинного навчання знайшли примінення у реалізації глибинних нейронних мереж.

В даний час існує близько десятка різновидів нейронних мереж. Одним з найбільш широко використовуваних варіантів є мережа, побудована на багат шаровому перцептроні, яка дозволяє класифікувати подане на вхід зображення/сигнал відповідно до попереднього налаштування/навчання мережі.

Навчаються нейронні мережі на наборі навчальних прикладів. Суть навчання зводиться до налаштування міжнейронних зв'язків в процесі рішення

оптимізаційної задачі методом градієнтного спуску. В процесі навчання нейронної мережі відбувається автоматичне вилучення ключових ознак, визначення їх важливості та побудова взаємозв'язків між ними. Передбачається, що навчена нейронна мережа зможе застосувати досвід, отриманий в процесі навчання, на невідомі образи за рахунок узагальнюючих здібностей.

Зі збільшенням популярності глибинних нейронних мереж почало з'являтися чимало фреймворків для машинного навчання. Для кращого розуміння зробимо порівняльну характеристику найпотужніших з них.

Tensor Flow[5] - це розроблена компанією Google надійна платформа з відкритим кодом, що підтримує глибоке навчання, доступ до якої можна отримати навіть зі смартфона.

Tensor Flow є відмінним інструментом для створення і розробки статистичних програм. Оскільки фреймворк пропонує розподілене навчання - всі моделі штучного інтелекту навчаються набагато ефективніше на будь-якому рівні абстракції, який обере користувач.

До особливостей можна віднести масштабований мультипрограмний інтерфейс для комфортного програмування, постійний розвиток платформи, за рахунок величезної спільноти ентузіастів і відкритого коду. Також платформа надає великі і добре задокументовані мануали для людей.

Із переваг: фреймворк здатний розвивати величезну обчислювальну потужність. Звідси випливає, що він може використовуватися фактично на будь-якому CPU і GPU. Система використовує обчислювальну графічну абстракцію для створення моделей штучного інтелекту.

Проте Tensor Flow має свої недоліки. Для прийняття рішення або прогнозування, фреймворк передає вхідні дані через кілька вузлів - цей процес займе у вас багато часу.

Другим розглянемо продукт компанії Microsoft. Microsoft CNTK - це швидкий і універсальний фреймворк з відкритим кодом, заснований на нейронних мережах з підтримкою тексту, повідомлень і ремоделювання голосу. Платформа



вдає із себе ефективну середу масштабування, за рахунок швидкої загальної оцінки комп'ютера з дотриманням якості оцінювання.

Microsoft CNTK інтегрований з більшою частиною масивів даних, що робить цей фреймворк одним з кращих у своєму роді, в тому числі для таких проектів як: Skype, Cortana і т. Д. Крім усього іншого фреймворк, представляє з себе простий і зрозумілий інструмент, що сприяє ефективній роботі з ним.

Фреймворк добре оптимізований, що забезпечує високу ефективність, масштабованість, відмінну швидкість роботи і високорівневу інтеграцію

Платформа включає в себе різні компоненти, такі як: налаштування гіперпараметра, контроль моделей і їх посилення, CNN, RNN і т. Д.

Фреймворк ефективно використовує обчислювальні потужності комп'ютера для забезпечення кращої працездатності фреймворка

Так як платформа підтримує мови Python і C ++, фреймворк дозволяє працювати з декількома сервісами одночасно, що, в свою чергу, значно прискорює процес навчання. Microsoft CNTK розроблявся з урахуванням останніх подій в світі штучного інтелекту. Архітектура Microsoft CNTK підтримує GAN, RNN і CNN. Все це дозволяє, за допомогою розподіленого навчання, ефективно навчати моделі штучного інтелекту.

Щодо недоліків, то у фреймворку відсутні панель візуалізації і підтримка мобільного архітектури мікропроцесорів ARM.

Ще одним поширеним фреймворком є Caffe. Це платформа, що включає в себе встановлені набори обучуваних нейронних мереж. Цей фреймворк відомий своїми можливостями обробки зображень, також в платформу була включена підтримка пакета прикладного ПЗ MATLAB.

Всі моделі фреймворка мають відкритий вихідний код, він забезпечує високу швидкість і ефективність роботи. Завдяки відкритому вихідному коду, у платформи існує активна спільнота, яке обговорює, модифікує і спільно використовує код фреймворка. Caffe підтримує мови C, C ++ і Python, а також CNN (технологія вигнутих нейронних мереж).

Проте Caffe не здатний обробляти комплексні масиви даних.

Останнім але не менш популярним фреймворком є Theano. Завдяки використанню графічних процесорів (GPU), замість центральних процесорів (CPU) - моделі штучного інтелекту на основі Theano досягають високої точності в обчислювальних операціях, що вимагають великої обчислювальної потужності. Theano здатний забезпечити високу продуктивність обчислення багатовимірних масивів даних.

Фреймворк заснований на мові програмування Python, який давно зарекомендував себе в задачах, для вирішення яких потрібна швидкі обробка і відповідь. Процес оцінки виразів протікає швидше, через динамічної генерації коду. Theano забезпечує чудову точність, навіть при мінімальних значеннях.

Модульне тестування є важливою особливістю Theano[6] - це дозволяє користувачеві самостійно перевіряти свій код, а також легко виявляти і діагностувати помилки. Фреймворк Theano забезпечує ефективну підтримку всіх додатків з інтенсивним використанням даних, але вимагає об'єднання з іншими бібліотеками. Платформа відмінно оптимізована для роботи як з CPU, так і з GPU.

Головним недоліком є те, що для поточної версії Theano не запланований випуск оновлень і додавання функціоналу.

Серед наведених актуальних фреймворків найкращим вибором для даної роботи є Tensor Flow, завдяки його активному розвитку та наявності інтерфейсів розробки для багатьох технологій та мов програмування.

#### 1.4 Аналіз існуючих бібліотек машинного зору

Для порівняльного аналізу було обрано найпопулярніші на даний момент бібліотеки машинного зору.

Intel Open Computer Vision library або OpenCV[7] написана на мові високого рівня (C / C++) і містить алгоритми для: інтерпретації зображень, калібрування камери за зразком, усунення оптичних спотворень, визначення подібності, аналіз

переміщення об'єкта, визначення форми об'єкту та стеження за об'єктом, 3D-реконструкція, сегментація об'єкта, розпізнавання жестів тощо.

Ця бібліотека дуже популярна за рахунок своєї відкритості та можливості безкоштовно використовувати як в навчальних, так і комерційних цілях.

OpenCV - це набір типів даних, функцій і класів для обробки зображень алгоритмами комп'ютерного зору.

Бібліотека поділяється на основні модулі, наведені нижче.

Перший модуль – `sxcore`, ядро що містить базові структури даних і алгоритми:

- базові операції над багатовимірними числовими масивами;
- матрична алгебра, математичні функції, генератори випадкових чисел;
- запис / відновлення структур даних в / з XML;
- базові функції 2D графіки.

Другий модуль – `CV`, модуль обробки зображень і комп'ютерного зору:

- базові операції над зображеннями (фільтрація, геометричні перетворення, перетворення колірних просторів і т. д.);
- аналіз зображень (вибір відмінних ознак, морфологія, пошук контурів, гістограми);
- аналіз руху, спостереження за об'єктами;
- виявлення об'єктів, зокрема осіб;
- калібрування камер, елементи відновлення просторової структури.

Наступною складовою Highgui - модуль для введення / виведення зображень і відео, створення призначеного для користувача інтерфейсу, він також виконує наступні функції:

- захоплення відео з камер і з відео файлів, читання / запис статичних зображень;
- функції для організації простого UI (всі демонстративні додатки використовують HighGUI).

Модуль `Svaux`, що містить експериментальні і застарілі функції:

- просторовий зір: стерео калібрації, безпосередньо калібрації;

- пошук стерео-відповідності, кліки в графах;
- знаходження і опис рис об'єктів.

Ще одна часто використовувана бібліотека - Matlab. Matlab є прекрасним інструментом для створення додатків для обробки зображень і широко використовується в дослідженнях. Причина в тому, що Matlab дозволяє швидко створювати прототипи.

Інший цікавий аспект полягає в тому, що код Matlab досить стислий, порівняно з C ++, що полегшує читання і налагодження. Він вирішує помилки перед виконанням, пропонуючи кілька способів зробити код швидше.

З іншого боку, Matlab є платним інструментом. Крім того, він може бути досить повільним під час виконання. Matlab не пристосований до використання в якості інструменту розробки у фактичному виробничому середовищі, оскільки він був побудований для створення прототипів і досліджень.

Наступна бібліотека для порівняння - PCL (Point Cloud Library). Це всеосяжна відкрита бібліотека для n-D Point Clouds і обробки 3D геометрії. Бібліотека містить численні висококласні алгоритми: фільтрації, функції оцінки, реконструкції поверхні, реєстрації, підгонки моделей, сегментації та ін. Написана на C ++ та знаходиться у вільному доступі.

PCL є популярним вибором при вирішенні задач таких як тривимірній комп'ютерний зір.

Ці алгоритми використовують, наприклад, для сприйняття оточення в робототехніці, щоб фільтрувати шум, зшивати тривимірні точкові хмари разом, сегментувати відповідні частини навколишньої середовища, витягувати ключові точки і обчислювати дескриптори для розпізнавання об'єктів у світі на основі їх геометричного виду, створювати поверхні з точкових хмар і візуалізувати їх.

Головним недоліком PCL є те, що ця бібліотека не оптимізована для роботи із 2D-зображенням в таких задачах як порівняння та розпізнавання образів.

Згідно результатів аналізу, на даний момент найбільш гнучким та потужним інструментом у створенні систем машинного зору є бібліотека OpenCV.

## 1.5 Постановка задачі

Метою атестаційної роботи є дослідження засобів та технологій розпізнавання образів на зображеннях та їх застосування у системах E-commerce. Результатом дослідження повинна стати розробка веб-додатку з використанням технології машинного зору та застосування знань, здобутих під час дослідницької роботи.

Для забезпечення роботи веб-додатку необхідна наявність у користувача пристрою з доступом до мережі Інтернет та щонайменше наступними системними вимогами: Internet Explorer 11, Google Chrome 49, Safari 10, Firefox 52, Android 4.2 або iOS 8.

Для максимально ефективної обробки візуальної інформації та успішної ідентифікації відсканованого товару в системах E-commerce необхідно реалізувати:

- підходи та методи ідентифікації об'єктів за зображенням;
- систему із можливістю глибинного машинного навчання;
- функції перетворення зображення для підвищення точності та швидкості порівняння образів.

Функціонал додатку повинен бути наступний:

- авторизація користувача в системі;
- пошук товарів за зображенням;
- можливість додавання зображень до каталогів розпізнавання;
- перехід за посиланням на сторінку інтернет-магазину зі знайденим товаром.

Відповідно до аналізу предметної галузі, задовільнення потреб кінцевих користувачів E-commerce систем є пріоритетним напрямом діяльності, а системи машинного зору користуються високим попитом, отже аналіз та розробка систем машинного зору для ідентифікації товару та визначення його вартості і місця продажу є актуальною.

## **2 ОПИС ДОСЛІДЖЕНЬ ТА ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ**

### **2.1 Аналіз підходів з порівняння ознак об'єктів на зображенні**

Зображення представляє складну інформацію у простому вигляді. Кожне зображення має свої ключові особливості, за якими ми можемо ідентифікувати об'єкти, що на ньому знаходяться. Ці особливості несуть основну інформацію. Це можуть бути точки, лінії, краї та багато інших ознак.

Є галузі досліджень, що займаються реєстрацією зображень, відстеженням об'єктів та їх вилученням тощо, де потрібно виявляти відповідні характеристики зображень.

Особливості об'єктів мають бути знайдені незалежно від обертання, масштабування, перетворення, освітленості, шумних і розмитих зображень.

Пошук відповідних зображень об'єкта за ключовими точками - це дуже складна робота. Ці ознаки повинні відповідати одна іншій під різними ракурсами та не залежати від кольорової гами.

Існує безліч алгоритмів, які використовуються для виявлення та відповідності функцій, такі як SIFT (Scale Invariant Feature Transform)[8], SURF(Speeded up Robust Feature)[9], FAST, ORB. SIFT і SURF є найбільш надійними і використовуваними методами для виявлення відповідностей зображень та їх особливостей.

Співпадіння підбираються на основі пошуку мінімальної порогової відстані. Відстань можна знайти за допомогою евклідової відстані, відстані Манхеттена тощо. Якщо відстань між двома точками менше мінімальної порогової відстані, ключові точки формують пару відповідностей.

Точки характеристик застосовуються для знаходження матриці гомографічного перетворення, вони знаходяться за допомогою RANSAC(RANdom SAmple Consensus).

## 2.2 SIFT (Scale Invariant Feature Transform)

В SIFT виконуються наступні основні кроки для виявлення та узгодження ключових точок.

Крок 1. Виявлення екстремумів масштабу простору.

Для знаходження функції унікальності, виконується перший етап, що здійснює пошук у просторі масштабу за допомогою функції Difference of Gaussian (DoG), визначаючи потенційні точки інтересу, які є інваріантними до масштабу та орієнтації. Простір масштабу зображення визначається як  $L(x, y, \sigma)$  (рівняння 1), який утворює згортку змінної шкали  $G(x, y, \sigma)$  (рівняння 2) з вхідним зображенням  $I(x, y)$ :

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad (1)$$

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (2)$$

Для виявлення ефективних і надійних ключових точок екстремуми масштабу простору знаходяться в множині DoG.  $D(x, y, \sigma)$ , який може бути обчислений з різниці двох сусідніх шкал, розділених постійним мультиплікативним коефіцієнтом  $k$  (рівняння 3):

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma) \quad (3)$$

Для кожної октави масштабного простору початкове зображення неодноразово згортається з гаусіями для отримання набору масштабованих зображень ліворуч (див. рис. 2.1).

Суміжні гаусові зображення віднімаються для отримання відмінностей гаусових зображень праворуч. Після кожної октави, гаусове зображення зменшується в 2 рази і процес повторюється.

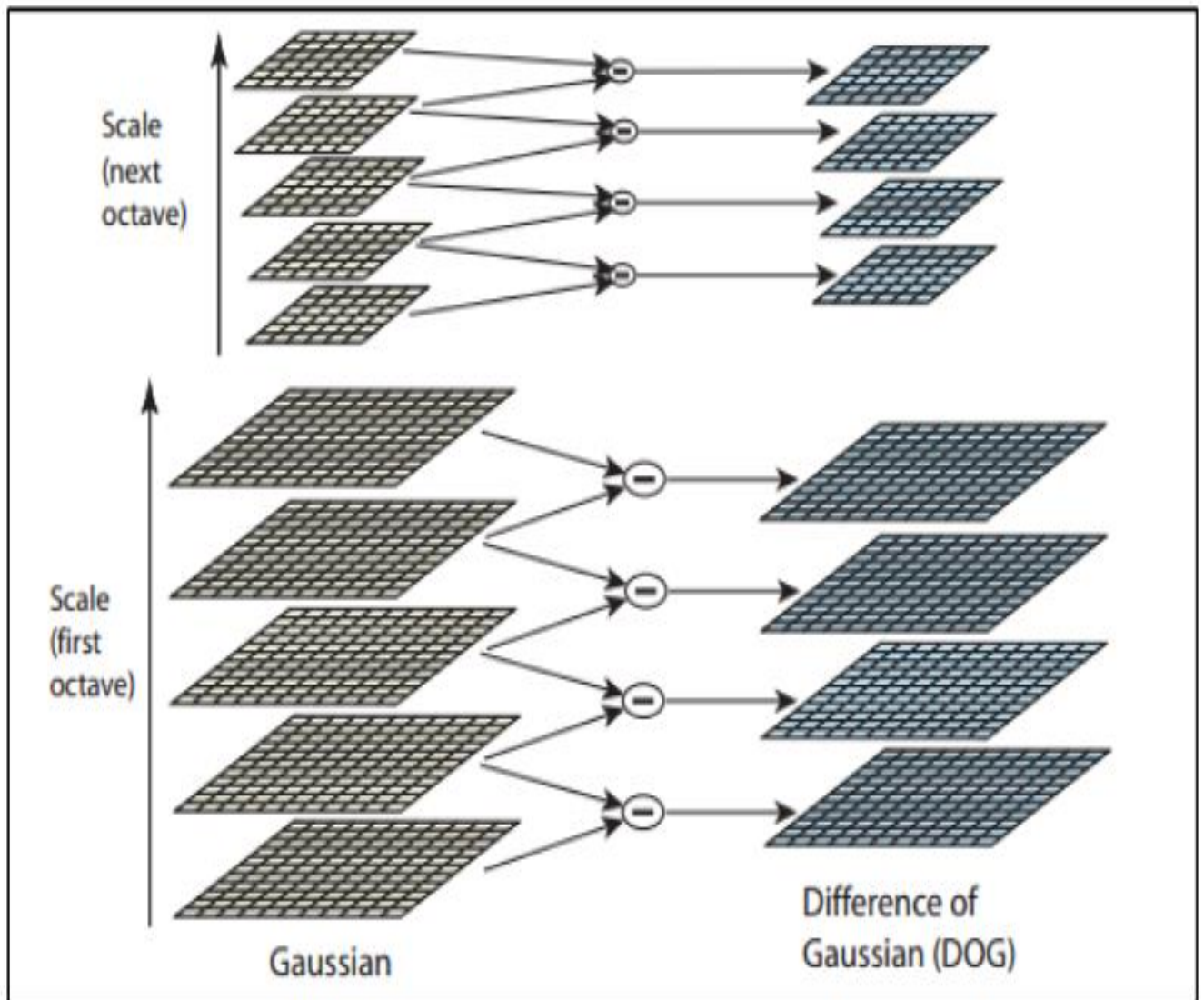


Рисунок 2.1 – Різниця гаусіанів

Крок 2. Локалізація ключової точки.

На місці кожної ключової точки розміщується детальна модель для визначення позиції та масштабу.

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \quad (4)$$

$$\theta(x, y) = \tan^{-1} \left( \frac{(L(x, y+1) - L(x, y-1))}{(L(x+1, y) - L(x-1, y))} \right) \quad (5)$$

Крок 3. Опис ключових точок.

Локальні градієнти зображення вимірюються в обраному масштабі в області навколо кожної ключової точки. Вони трансформуються в подання, що дозволяє не зважати на значні рівні локальних спотворень форми і зміни освітленості.



## 2.3 SURF (Speeded up Robust Feature)

Детектор і дескриптор SURF не тільки виконують роботу швидше, але і перший є більш повторюваним, а останній більш розпізнавальним.

Детектори на базі гесіанів є більш стабільними та повторюваними, ніж їхні аналоги на основі Харріса [10], і виявлено, що такі приблизні розрахунки, такі як DoG, можуть принести швидкість з низькою втратою точності при порівнянні ключових точок.

У SURF є два основних кроки.

Крок 1. Пошук ключової точки.

SURF узгоджує оригінальне зображення із інтегральним зображенням. Інтегральне зображення, яке підсумовує таблиці областей, є проміжним поданням зображення.

Інтегральне зображення - це сума значень інтенсивності всіх пікселів у оригінальному вхідному зображенні.

Прямокутна область зображення, утворена початковою точкою  $O = (0, 0)$  і будь-якою точкою  $X = (x, y)$ . Цей метод забезпечує швидке обчислення згорткових box-фільтрів.

$$I_{\Sigma}(X) = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(i, j) \quad (6)$$

На основі інтегрального зображення виконуються лише три операції (додавання або віднімання), необхідні для розрахунку суми інтенсивності значення пікселів будь-якої прямокутної області.

При використанні інтегральних зображень для обчислення суми інтенсивностей усередині прямокутної області будь-якого розміру потрібно лише три додавання та чотири доступи до пам'яті (див. рис. 2.2).

Інтегральне зображення складається з box-фільтром. Box-фільтр є приблизним фільтром гаусового фільтра.

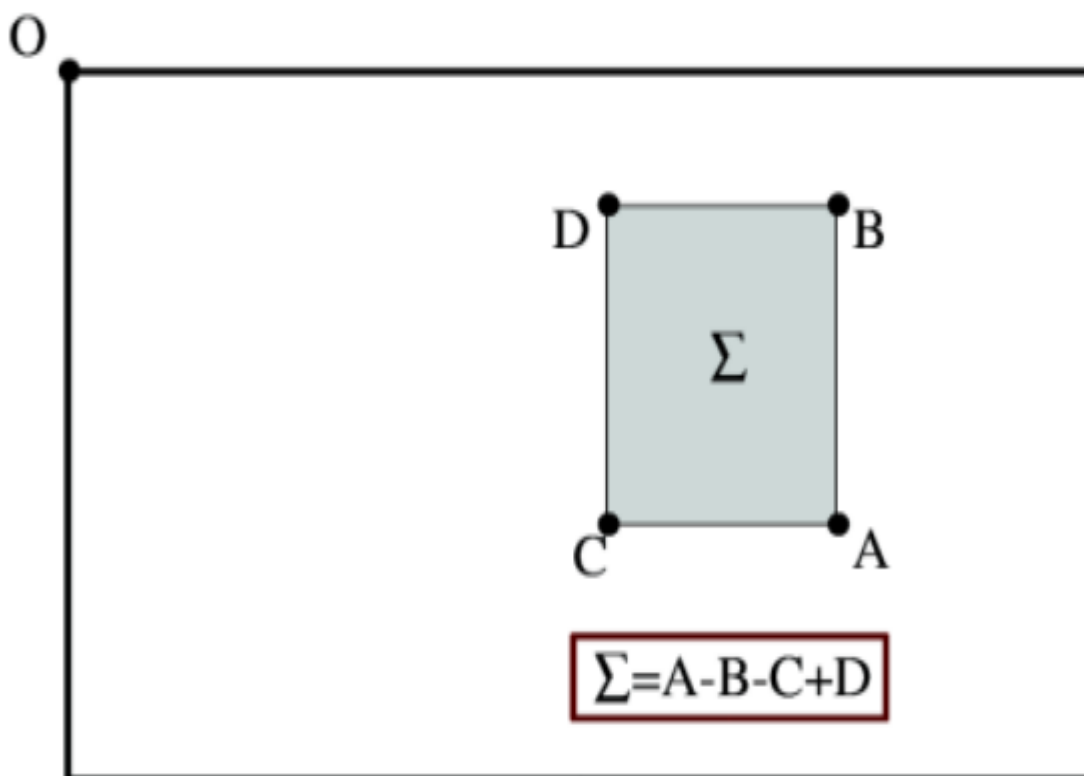


Рисунок 2.2 – Розрахунок суми інтенсивностей

Матриця гессіанів  $\mathcal{H}(X, \sigma)$ , (див. рівняння 7) де  $X = (x, y)$  зображення  $I$ , при масштабі  $\sigma$  визначається наступним чином:

$$\mathcal{H}(X, \sigma) = \begin{bmatrix} L_{xx}(X, \sigma) & L_{xy}(X, \sigma) \\ L_{xy}(X, \sigma) & L_{yy}(X, \sigma) \end{bmatrix} \quad (7)$$

Крок 2. Опис ключової точки.

Для інваріантності до обертання зображення ми визначаємо відтворювану орієнтацію для ключових точок. Через це спочатку розраховуються вейвлет-відповіді Хаара [11] в  $x$  і  $y$  напрямку в межах кругової зони радіуса  $6s$  навколо ключової точки, з масштабом  $s$  (крок вибірки залежить від  $s$ ), при якому виявлена ключова точка.

Розмір вейвлету залежить від масштабу, і його сторона має довжину  $4s$ . Для обчислення відповіді у напрямку  $x$  або  $y$  у будь-якому масштабі необхідно лише шість операцій.

На рисунку 2.3 зображені вейвлет-фільтри Хаара для обчислення відповідей у напрямку  $x$  (ліворуч) і  $y$  (справа). Темні частини мають вагу  $-1$  і легкі частини  $+1$ .

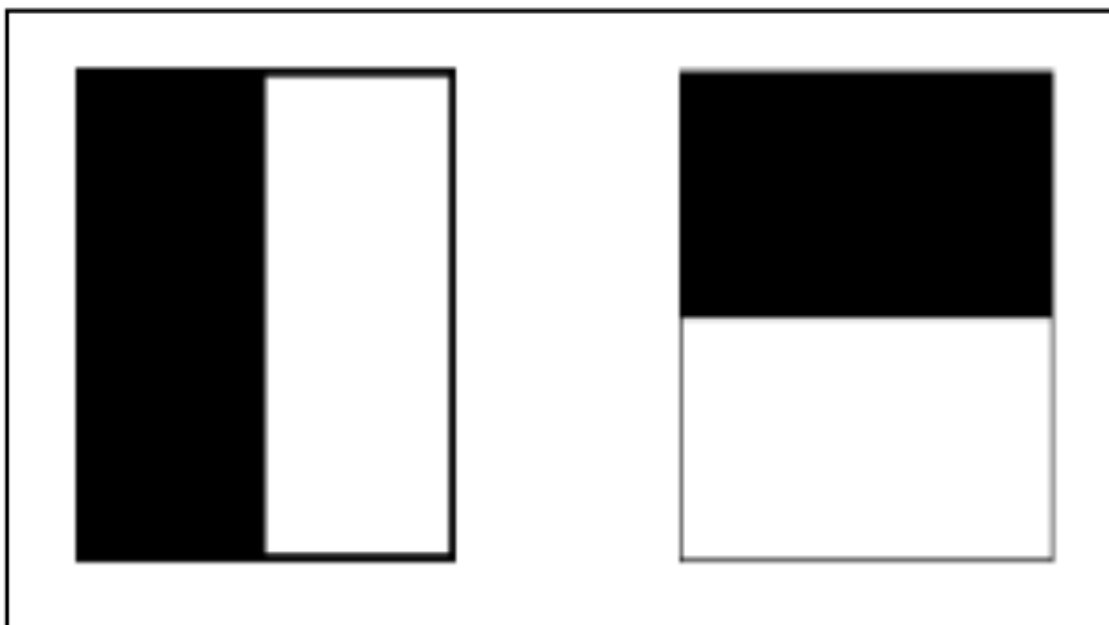


Рисунок 2.3 – Вейвлет-фільтри Хаара

Після того вейвлет-відповіді обчислюються і зважуються з гаусіаною  $\sigma = 2s$  з центром в точці інтересу. Відповіді представлені у вигляді точок у просторі з горизонтальною силою відгуку вздовж абсциси і по вертикалі ординату. Знаходиться максимальна сума всіх відповідей, яка є вейвлет-відповіддю у кожному вікні (орієнтація вікна  $\pi / 3$ ) (див. рис. 2.4).

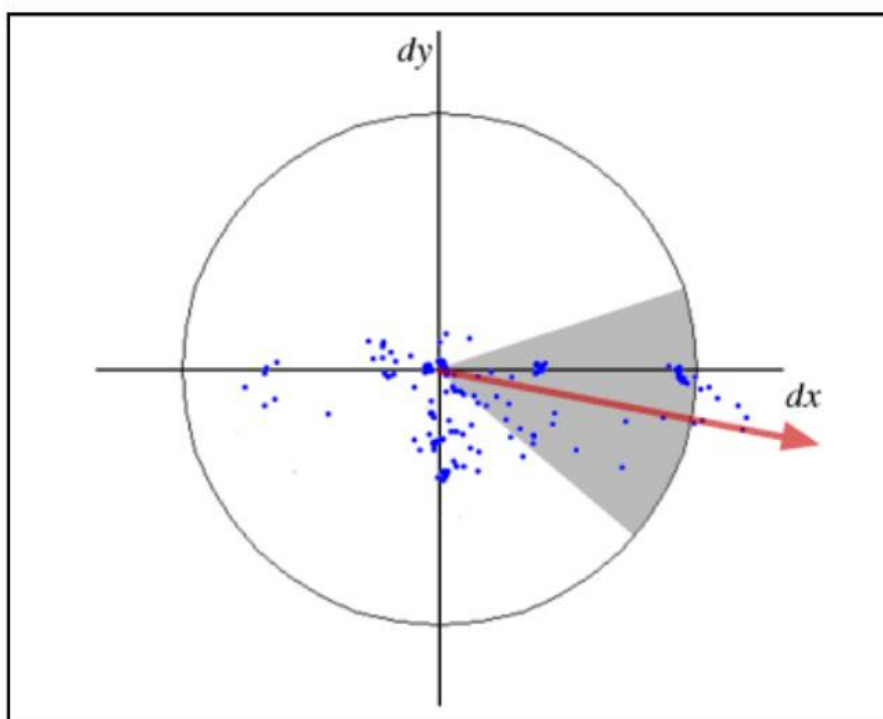


Рисунок 2.4 – Задання орієнтації

Підсумовуються горизонтальні та вертикальні відповіді у вікні. З цих двох горизонтальних і вертикальних, підсумовані відповіді потім дають вектор локальної орієнтації. Орієнтація ключової точки може бути визначена шляхом знаходження найдовшого вектора серед усіх вікон.

Для вилучення дескриптора, квадратична область, розмір якої дорівнює  $20s$ , будується на ключових точках. Приклади таких квадратів області показані на рисунку 2.5.



Рисунок 2.5 – Визначені орієнтовані квадрати ключових областей

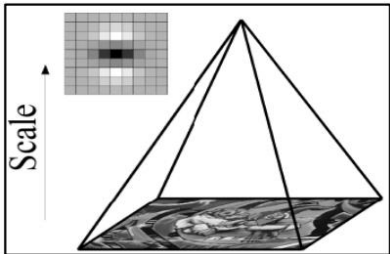
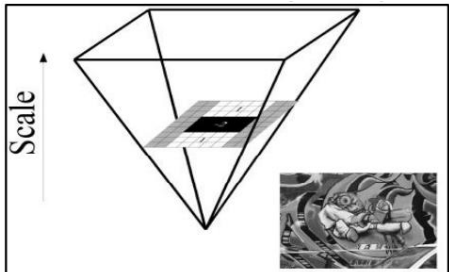
Вейвлет-відповіді  $dx$  і  $dy$  підсумовуються по кожному субрегіону і утворюють перший набір записів у векторі ознак. Для того, щоб ввести інформацію про полярність змін інтенсивності, витягується сума абсолютних значень відповідей,  $|dx| + |dy|$ , кожний субрегіон має вектор 4-мірного дескриптора  $V$  для своєї основної структури інтенсивності  $V = (\sum dx, \sum dy, \sum |dx|, \sum |dy|)$ . Об'єднання всіх  $4 \times 4$  субрегіонів дає в результаті дескрипторний вектор довжини 64.

Вейвлет-відповіді інваріантні до змін в освітленні (зміщення) і інваріантності до контрасту (масштабний коефіцієнт), що досягається за рахунок перетворення дескриптора в одиничний вектор.

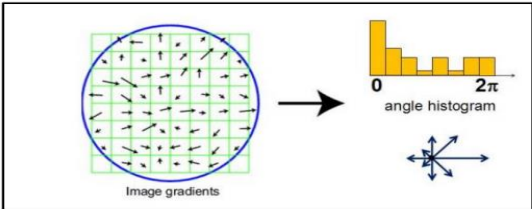
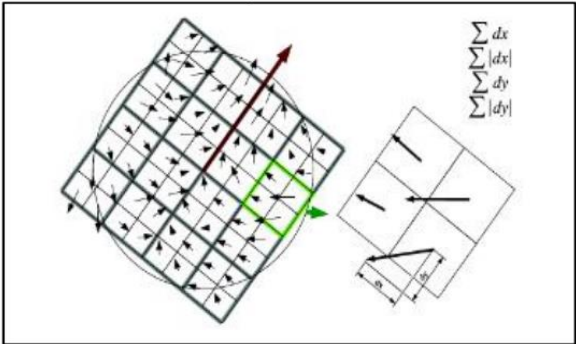
## 2.4 Відмінності між SIFT та SURF

У попередніх розділах були делально розглянуті методи порівняння зображень SIFT та SURF. Найважливішу інформацію про особливості даних алгоритмів наведено у таблиці 2.1.

Таблиця 2.1 - Порівняльна характеристика методів SIFT і SURF

	SIFT	SURF
Робота з масштабом зображень	<p>Difference of Gaussian (DoG) згортання з різними розмірами зображення з однаковим розміром фільтра (див. рис. 2.6).</p>  <p>Рисунок 2.6 - Фіксований фільтр складається з зображеннями вибірки</p>	<p>(Laplacian of Gaussian (LoG)) – відмінний розмір box-фільтра складається з інтегральним зображенням (див. рис. 2.7).</p>  <p>Рисунок 2.7 - Фіксоване зображення складається з фільтрами вибірки</p>
Пошук ключових точок	<p>Використовуючи визначення локальних екстремумів, застосовується немаксимальне стискання і виключаються краї з матриці Гессе.</p>	<p>Визначаються ключові точки за матрицею Гессе і немаксимальним стисканням.</p>

Кінець таблиці 2.1

	SIFT	SURF
Орієнтація	Величина градієнта зображення і орієнтації відбираються навколо розташування ключової точки, використовуючи масштаб ключової точки для вибору рівня розмиття Гауса для зображення.	Рухоме вікно орієнтації розміром $\pi/3$ визначає головну орієнтацію зваженого вейвлета Хаара.
Дескриптор	<p>Дескриптор ключової точки дозволяє значно змінювати градієнт позицій шляхом створення гістограм орієнтації у регіонах розміром 4x4. На рисунку 2.8 показано 8 напрямків для кожної орієнтаційної гістограми з довжиною кожної стрілки, що відповідає величині запису гістограми.</p>  <p>Рисунок 2.8 – Задання орієнтації в SIFT</p>	<p>Орієнтаційна квадратична сітка з квадратних секцій розмірами 4x4 (див. рис. 2.9) складається над ключовою точкою. Для кожного квадрата, вейвлет-відповіді обчислюються з 5x5 зразків.</p> <p>Дескриптор SURF - <math>V = (dx, \sum dy, \sum  dx , \sum  dy )</math>.</p>  <p>Рисунок 2.9 – Задання орієнтації в SURF</p>
Розмір дескриптора	128 біт	64 біта

Таблиця наглядно демонструє основні характеристики алгоритмів SIFT і SURF, такі як робота із масштабом та орієнтацією зображення, властивості дескрипторів та їх розмір.

## 2.5 Результати порівняльного аналізу

Для порівняння продуктивності методів було використано OpenCV 2.4.9 [12] налаштований на Microsoft Visual Studio 2013 і Windows 8 OS. Тестове зображення було модифіковане за допомогою графічних редакторів та збережене в декількох варіантах, наведених нижче.

На рисунку 2.10 представлено оригінальне зображення, на основі якого буде здійснене порівняння стійкості методів до спотворень картинки.



Рисунок 2.10 – Оригінальне зображення

Наступні рисунки демонструють всі основні види спотворень основного зображення.

Першим видом модифікації зображення є його обернення на 90 градусів зі зміною співвідношення сторін (див. рис. 2.11).





Рисунок 2.11 – Обернене на 90 градусів зображення

Наступним зображенням для порівняння було обрано фрагмент оригінального зображення, збільшений у декілька разів (див. рис. 2.12).



Рисунок 2.12 – Фрагмент оригінального зображення із іншим масштабом



Ще один зразок для аналізу – розмитий варіант оригінального зображення, показаний на рисунку 2.13.



Рисунок 2.13 – Розмите зображення

На рисунку 2.14 продемонстровано зразок для порівняння зі зміненою кольоровою гамою.



Рисунок 2.14 – Зображення із іншим відтінком

Ще один вид редагування зображення, який було використано – це зміна насиченості кольорів, тобто яскравості зображення (див. рис. 2.15).



Рисунок 2.15 – Зображення з відмінною насиченістю кольору

Також потрібно порівняти методи аналізу зображень на стійкість до викривлення зображень. Так, на рисунку 2.16 показано зразок картинки, яку було деформовано за двома осями та повернено на 60 градусів.

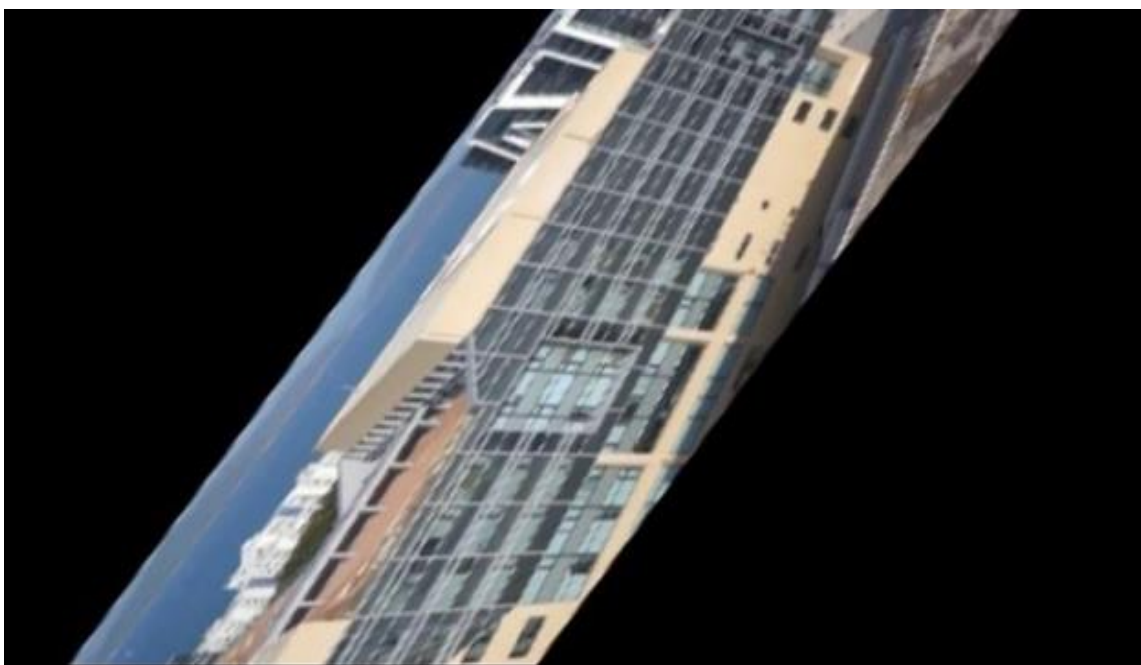


Рисунок 2.16 – Викривлене зображення

Останнім видом спотворення в даному порівнянні є створення кольорових шумів (див. рис. 2.17)



Рисунок 2.17 – Зображення із шумом

SURF і SIFT застосовуються на оригінальному зображенні з різними типами зображень. У таблиці 2.2 пояснюється, як багато точок ознак знаходяться в зображенні 1 і зображенні 2, скільки пар ознак було утворено з двох зображень і як багато часу використовується для вилучення і узгодження пар зображення.

Таблиця 2.2 - Порівняння продуктивності методів SIFT і SURF

Зображення	SIFT				SURF			
	Знайдено особливостей у зображенні1	Знайдено особливостей у зображенні2	Кількість співпадінь	Час виконання (мс)	Знайдено особливостей у зображенні1	Знайдено особливостей у зображенні2	Кількість співпадінь	Час виконання (мс)
Зображення (2.10) та (2.11)	3001	3043	215	5235.25	878	886	309	1354.27
Зображення (2.10) та (2.12)	3001	823	153	3511.85	878	325	32	989.488



Кінець таблиці 2.2

Зображення	SIFT				SURF			
	Знайдено особливостей у зображенні1	Знайдено особливостей у зображенні2	Кількість співпадінь	Час виконання (мс)	Знайдено особливостей у зображенні1	Знайдено особливостей у зображенні2	Кількість співпадінь	Час виконання (мс)
Зображення (2.10) та (2.13)	3001	862	29	3626.46	878	356	217	1008.99
Зображення (2.10) та (2.14)	3001	823	153	4075.62	878	895	169	1215.27
Зображення (2.10) та (2.15)	3001	2164	479	4368.75	878	225	220	973.853
Зображення (2.10) та (2.16)	3001	1567	266	4263.84	878	1767	565	935.151
Зображення (2.10) та (2.17)	3001	3315	23	5223.4	878	992	57	1279.22

Інформація з таблиці представлена у вигляді графіків. На рисунку 2.18 показано кількість знайдених ознак у зображенні за допомогою SIFT і SURF.

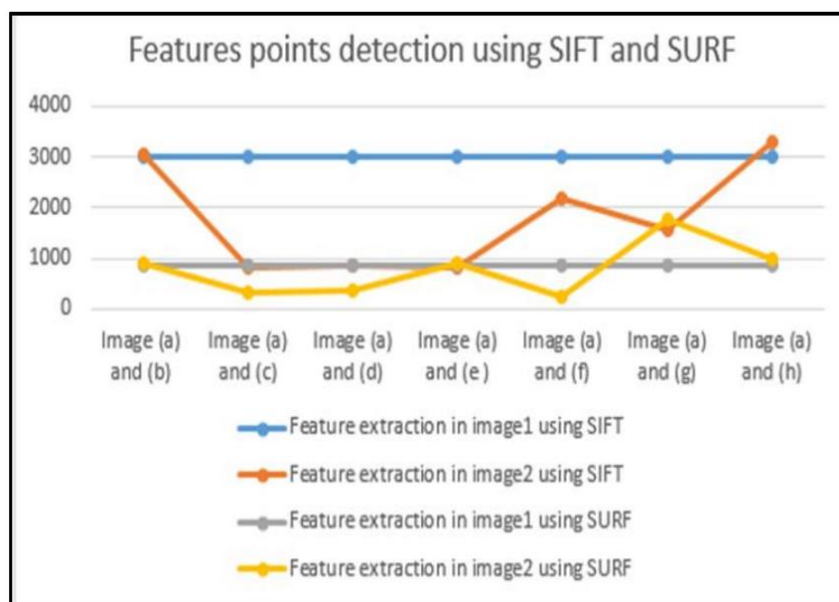


Рисунок 2.18 - Кількість виявлених особливостей за допомогою SIFT і SURF

На рисунку 2.19 показано, скільки відповідних точок ознак узгоджується за допомогою виявлення точок особливості SIFT і SURF.

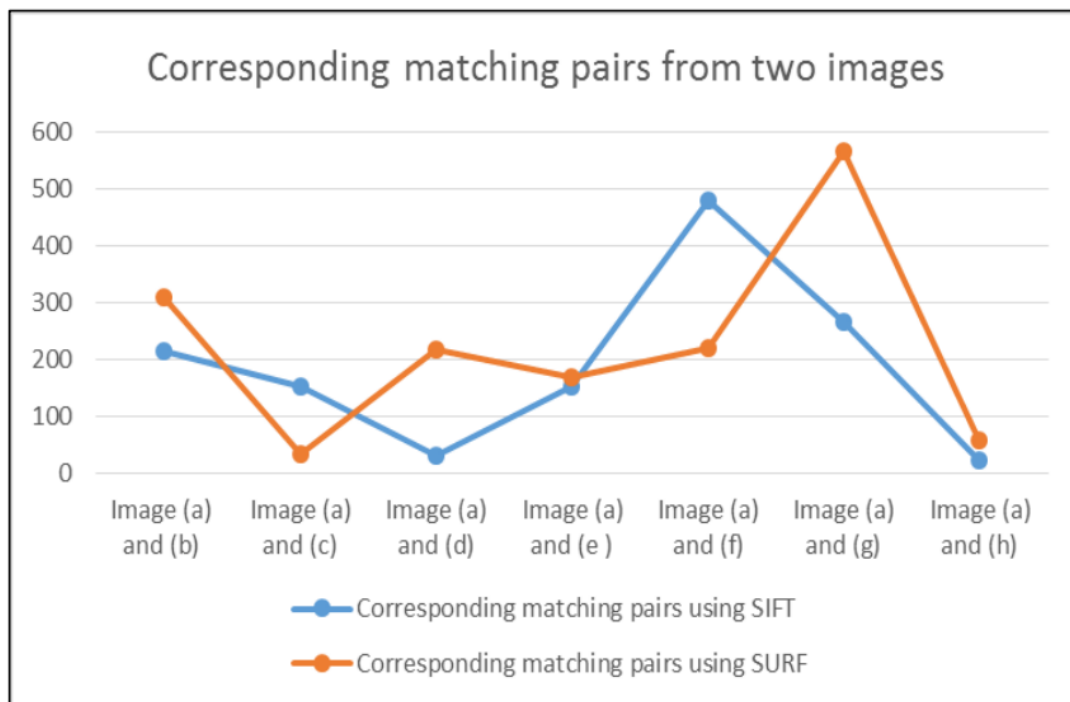


Рисунок 2.19 - Кількість утворених пар ознак за допомогою SIFT і SURF

На рисунку 2.20 показано загальний час вилучення для пошуку та зіставлення точок за допомогою SIFT і SURF.

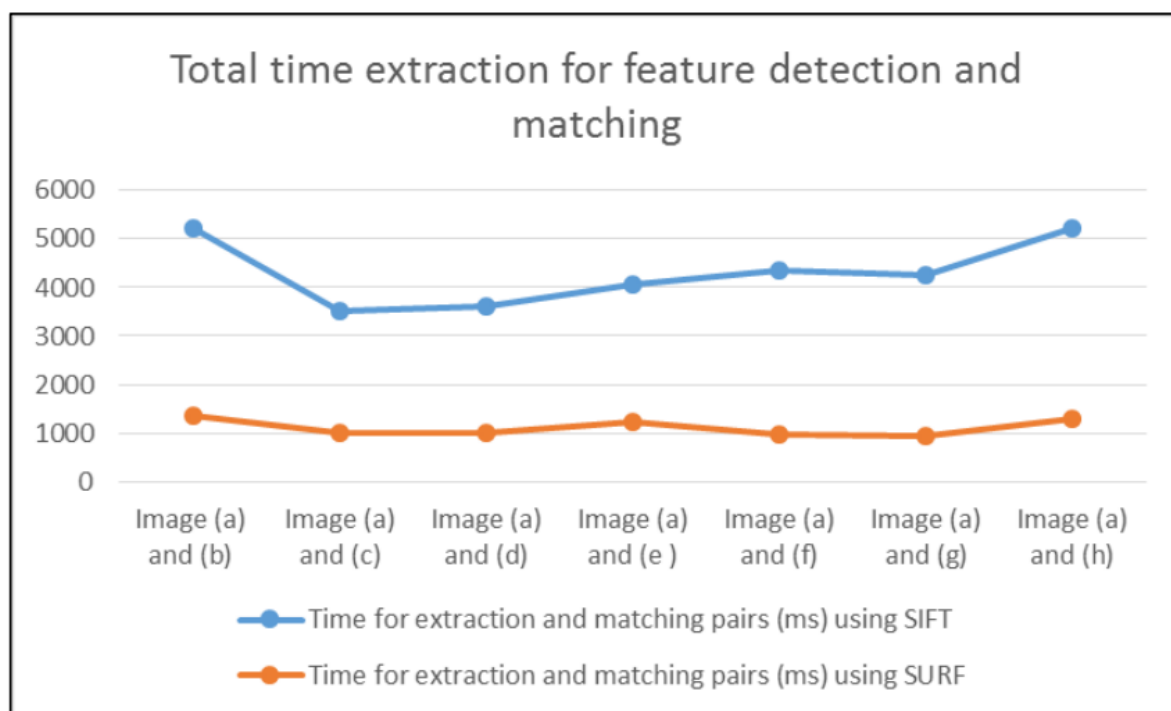


Рисунок 2.20 – Час обробки зображень методами SIFT і SURF

З порівняння двох методів ми маємо наступні результати:

- SURF забезпечує кращий результат відносно відповідних пар об'єктів у ротації, при розмитті зображення, різниці відтінків, різних деформацій, трансформацій і шумів, ніж SIFT;
- SIFT забезпечує кращий результат у різних масштабах і насиченості зображення, ніж SURF;
- SURF в 3 рази швидше, ніж SIFT.

Можна зробити висновок, що SIFT і SURF є надійними методами для виявлення відповідностей у зображеннях. Обидва інваріантні до повороту, масштабу, розмиття, області освітлення, викривлення та шуму. Проте SURF має кращий результат за більшістю показників: у ротації, розмитті, викривленні, шумах RGB і часі, ніж SIFT.

## 2.6 Вибір методів та технологій для аналізу зображень

За результатами тестування було обрано бібліотеку з відкритим кодом OpenCV, але з урахуванням обраних технологій розробки (C#/.NET) застосовано адаптований фреймворк Emgu CV на мові C#, що є обгорткою оригінальної бібліотеки OpenCV.

Враховуючи великий обсяг даних для порівняння у програмній системі для розпізнавання образів, було прийнято рішення застосувати глибинне навчання на базі фреймворку Tensor Flow із використання вже готових моделей машинного навчання. Але варто зазначити, що для більш якісної роботи потрібно тривалий час навчати нейронну мережу на великих об'ємах даних.

Інструменти Tensor Flow [13] дозволяють виділити об'єкт на фото та віднести його до певної категорії (див. рис. 2.21).

Початкова обробка зображення дозволяє значно звузити коло пошуку зображень до однієї категорії. Також тепер ми можемо виділити об'єкт на фото,

відкинувши зайву частину зображення. Чим менший розмір зображення, тим більша швидкість його аналізу.

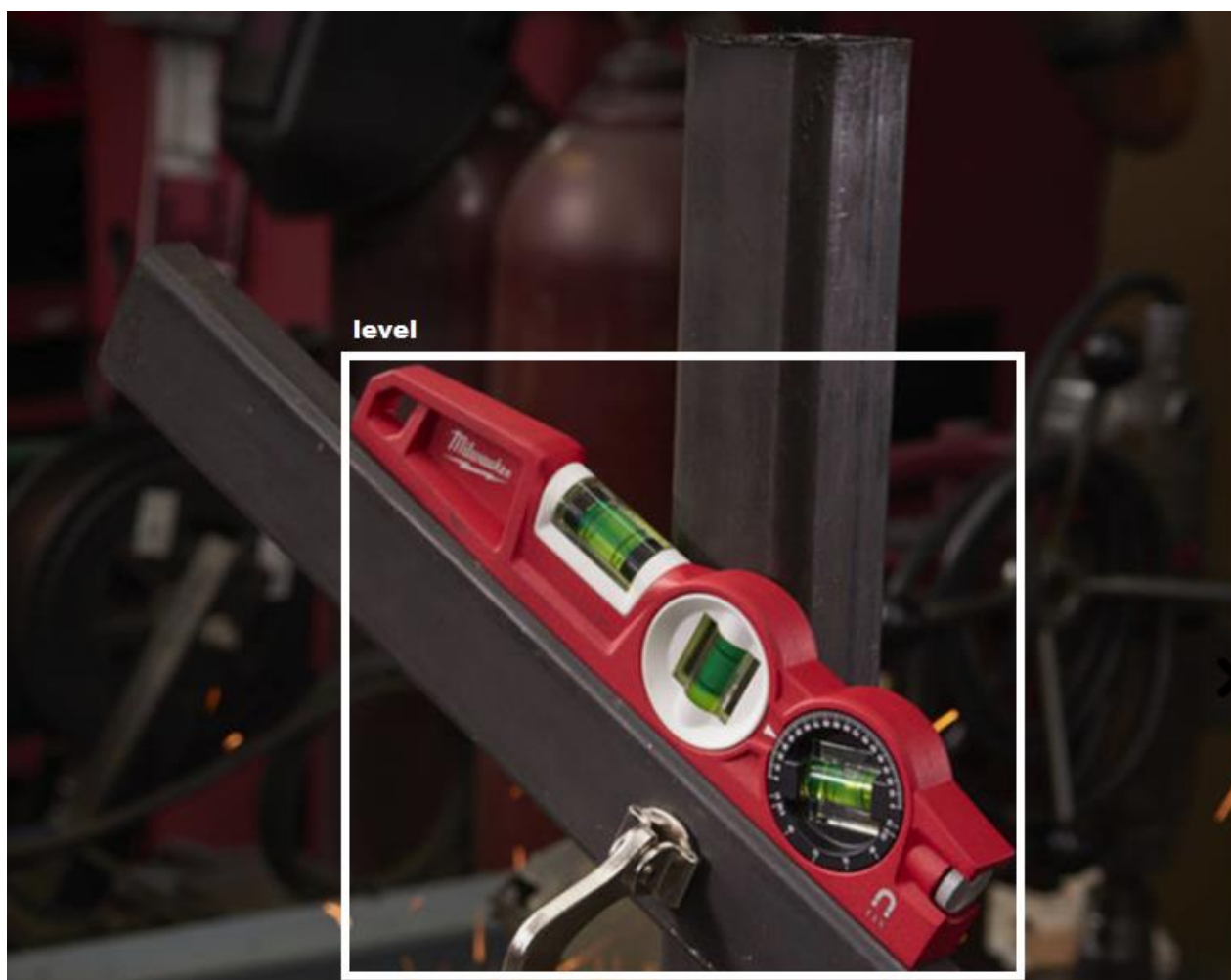


Рисунок 2.21 – Класифікація об'єктів завдяки Tensor Flow

Подальша обробка потребує використання спеціальних алгоритмів для стабілізації зображення [14], усунення шуму на фото.

Одним з перших пунктів цієї обробки є перетворення зображення в чорно-біле, після чого усунення з зображення шуму, або розмиття за допомогою розмивання Гауса.

Розмивання Гауса [15] - один з часто застосовуваних фільтрів розмиття зображень, в основі якого лежить нормальний (Гаусовий) розподіл, що застосовується для вирахування перетворень.

Рівняння розподілу Гауса для  $N$  вимірів має вигляд:

$$G(r) = \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} e^{\frac{-r^2}{2\sigma^2}}$$

де  $r$  — це радіус розмиття, для якого  $r^2 = u^2 + v^2$ ,

$\sigma$  — стандартне відхилення розподілу Гауса.

У випадку двох вимірів ця формула задає концентричну поверхню з розподілом Гауса від центральної точки. Ті пікселі, розподілення яких відмінне від нуля, застосовуються для побудови матриці згортання [16], що використовується безпосередньо для обробки вихідного зображення. Значення кольору кожного пікселя стає усередненим по оточенню. Центральний піксель має найбільшу вагу (має найвище значення), а сусідні пікселі меншу вагу, в залежності від відстані до них.

Метод розмивання Гауса у Emgu CV має вигляд:

```
public static void GaussianBlur(  
    IInputArray src,  
    IOutputArray dst,  
    Size ksize,  
    double sigmaX,  
    double sigmaY = 0,  
    BorderType borderType = BorderType.Reflect101)
```

Наступним етапом буде порівняння зображення із застосування методу SURF. Метод Speeded Up Robust Features (SURF) позитивно зарекомендував себе в задачі пошуку зображень, 3D реконструкції, при порівнянні зображень.

Метод SURF вирішує дві задачі - пошук особливих точок і створення їх дескрипторів (описувальних елементів, інваріантного та змінного масштабу). Крім того, сам пошук ключових точок теж повинен володіти інваріантністю, тобто повернутий об'єкт повинен володіти тим самим набором ключових точок, що і зразок.



Цей метод реалізується за допомогою інструментів Emgu CV (див. рис. 2.22).

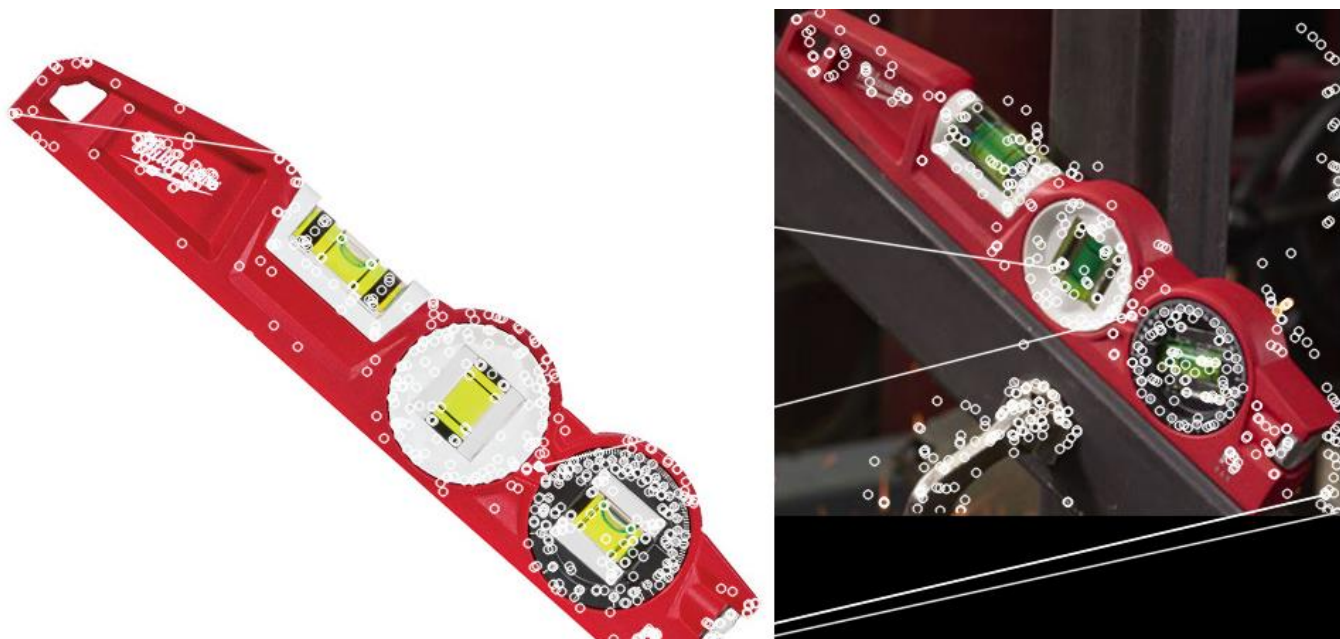


Рисунок 2.22 – Використання методу SURF за допомогою Emgu CV

Кожному зображенню в базі відповідає артикул товару. Тому знайшовши зображення, що найбільше відповідає шуканому об'єкту, користувач зможе легко знайти цей товар. У випадку успішного пошуку за артикулом користувачу буде надано інформацію про товар, таку як його назва, вартість та наявність, а також посилання на сайт онлайн магазину із даним товаром.

## 3 ПРОЕКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

### 3.1 UML проектування програмної системи

UML [17] – це аббревіатура, що означає "Єдина мова моделювання". Іншими словами, UML – це сучасний підхід до моделювання, візуалізації, опису та документування програмного забезпечення. Він допомагає розкрити взаємозв'язок між діями користувача та відгуком програмної системи, взаємозв'язком між різними компонентами системи, тощо.

UML був створений в результаті хаосу, що обертався навколо розробки програмного забезпечення та документації. У 1990-х роках існувало кілька різних способів представлення та документування програмних систем. Виникла потреба більш уніфікованого способу візуального представлення цих систем, і в результаті в 1994-1996 роках UML розробляли три програмні інженери, що працювали в Rational Software. Пізніше він був прийнятий як стандарт у 1997 році і з тих пір залишається стандартом, отримуючи лише кілька оновлень.

Діаграми UML [18] використовуються для передачі різних аспектів і характеристик системи. Проте вони використовуються для високорівневого опису системи і не можуть передати усі деталі, необхідні для повноцінної розробки програмного забезпечення.

В результаті проектування програмного додатку було розроблено декілька UML діаграм для того, щоб описати його роботу на високому рівні. Дані діаграми буде розглянуто нижче.

Для побудови UML діаграм було використано онлайн ресурс Draw.io.

Use case діаграма відображає взаємодію користувача з системою та функції системи, доступні для того чи іншого актора. В розробленій системі є один актор, котрий може авторизуватися в системі щоб відкрити доступ до додатку. Усі функції додатку доступні лише при наявності стабільного підключення до мережі Інтернет. Основними функціями є: розпізнавання об'єктів на зображенні та знаходження їх у онлайн магазинах, додавання нових товарів до бази пошуку, оскарження

існуючих даних у випадку їх невідповідності, перегляд історії пошуку. Діаграма зображена на рисунку 3.1.

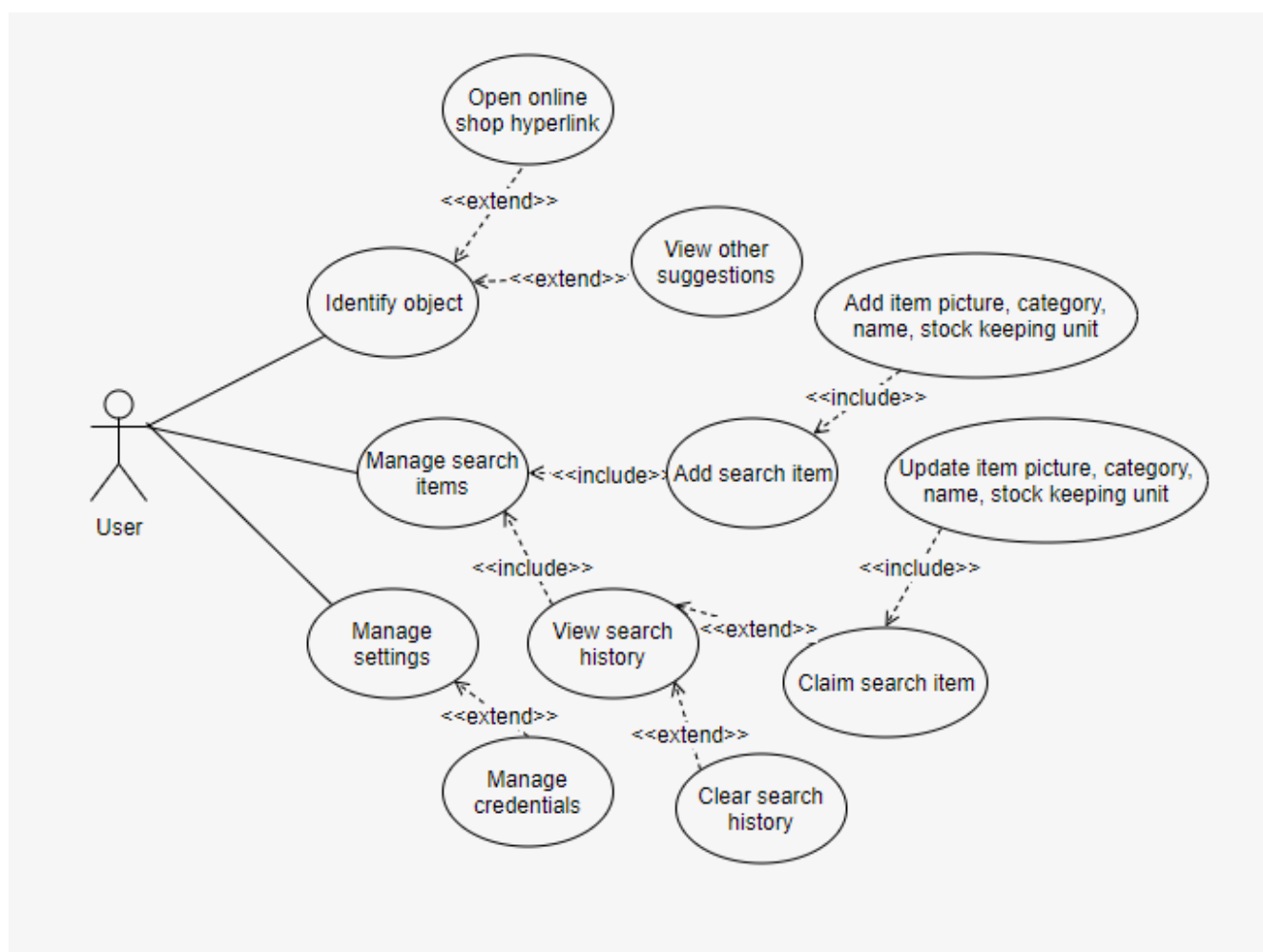


Рисунок 3.1 – Use case діаграма

На Use Case діаграмі зображено весь основний функціонал мобільного додатку та вказано, які функції розширюють інші. Наприклад, як видно на діаграмі, користувач може відкрити посилання на товар в магазині після його ідентифікації.

Користувач також може видаляти історію пошуку. В цьому випадку, усі дані про попередньо відскановані товари буде видалено з профіля.

Діаграма послідовностей відображає те, що відбувається поза очима користувача, коли додаток намагається розпізнати об'єкт. Відразу після автентифікації, користувач бачить інтерфейс програми, де має можливість завантажити фото для пошуку. Модель запускає Recognition Service, котрий

використовує SURF-метод для порівняння зображень. На даному етапі працюють функції комп'ютерного бачення. Діаграму наведено на рисунку 3.2.

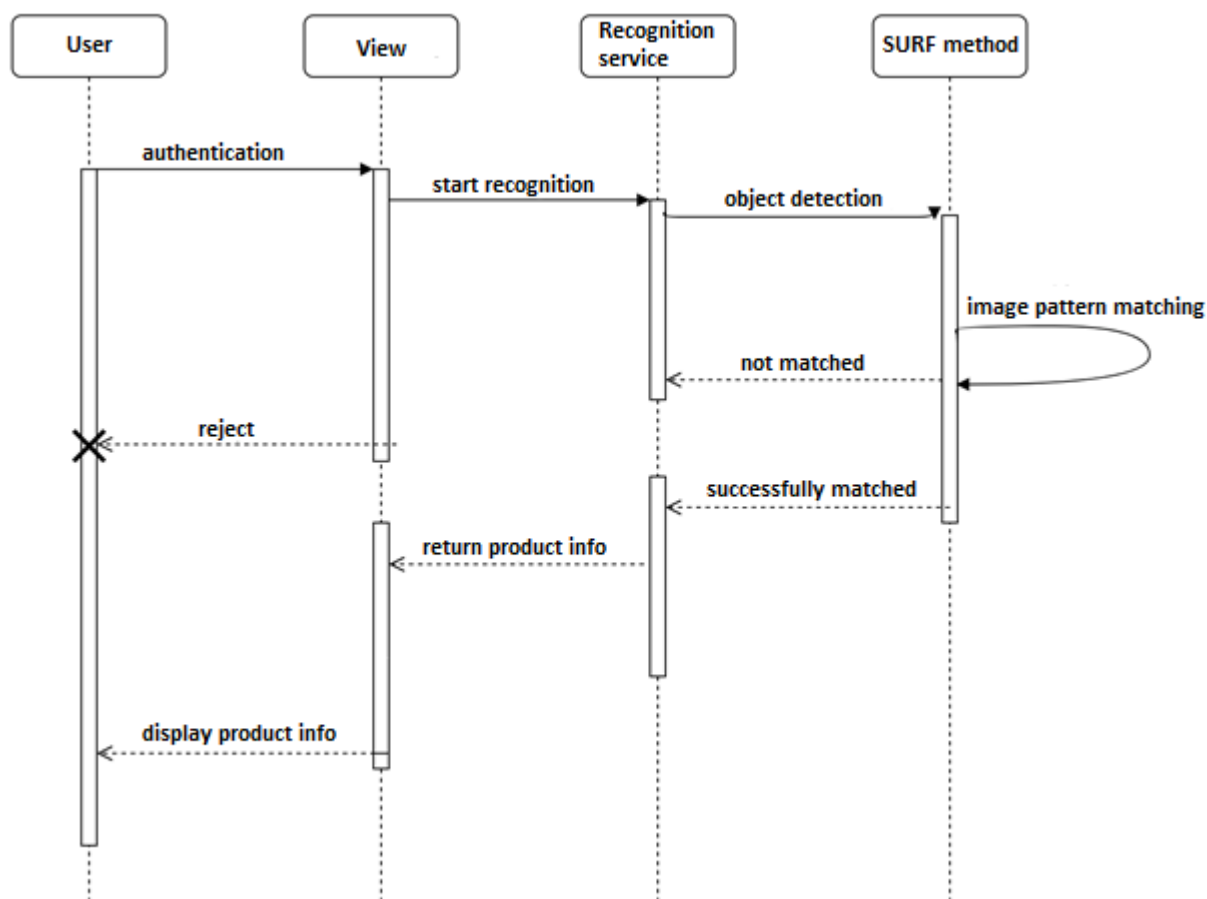


Рисунок 3.2 – Діаграма послідовності

Якщо об'єкт розпізнався, то дані про нього передаються до Recognition Service, котрий формує набір інформації про цей об'єкт та відображує її в інтерфейсі програми.

Наступним кроком було створення діаграми класів (див. рис. 3.3). Ця діаграма відображає основні моделі системи та залежності між ними.

Більшість службових класів та класів роботи з базою даних не були включені до діаграми, бо вони мають службове значення та не допомагають зрозуміти систему на високому рівні.

Діаграма включає в себе клас ProductService, котрий дозволяє виконувати основні операції над продуктами, такі як їх додавання, редагування та видалення.

Ще однією важливою складовою є сервіс для розпізнавання та порівняння зображень.

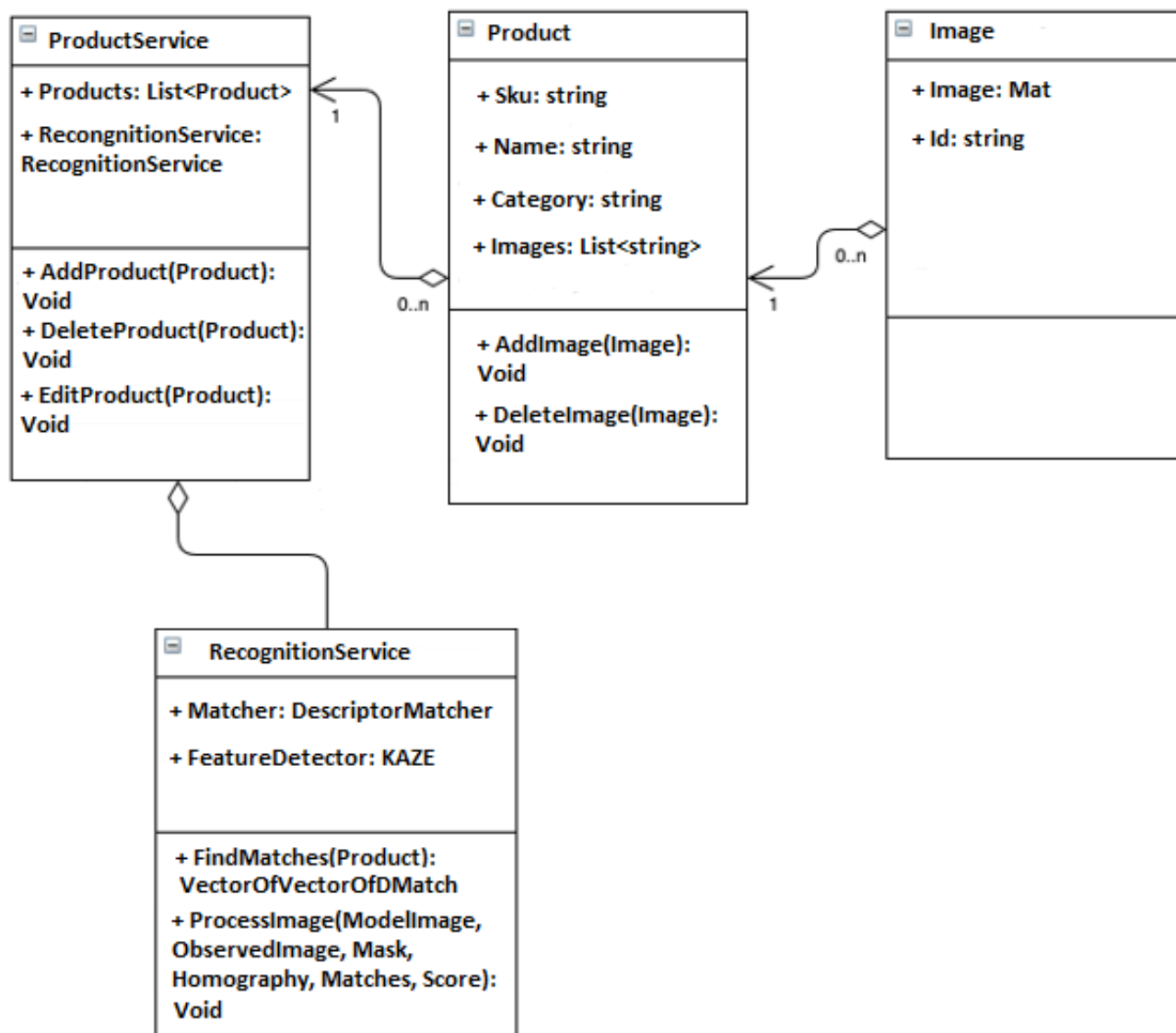


Рисунок 3.3 – Діаграма класів

Клас `Product` дозволяє додавати або видаляти зображення, пов'язані із конкретним продуктом. Цей клас відповідає безпосередньо за той об'єкт, котрий буде розпізнаватися та включає в себе уся інформацію, котру треба буде додавати на екран після успішного розпізнавання.

### 3.2 Архітектура веб-додатку

Для реалізації сервісу вирішено використовувати клієнт-серверну архітектуру. Ця архітектура є одним із архітектурних шаблонів програмного забезпечення та є домінуючою концепцією у створенні розподілених мережових додатків і передбачає взаємодію та обмін даними між ними. На рисунку 3.4 наведене схематичне зображення архітектури системи із вказаними технологіями, що були застосовані на кожному з рівнів.

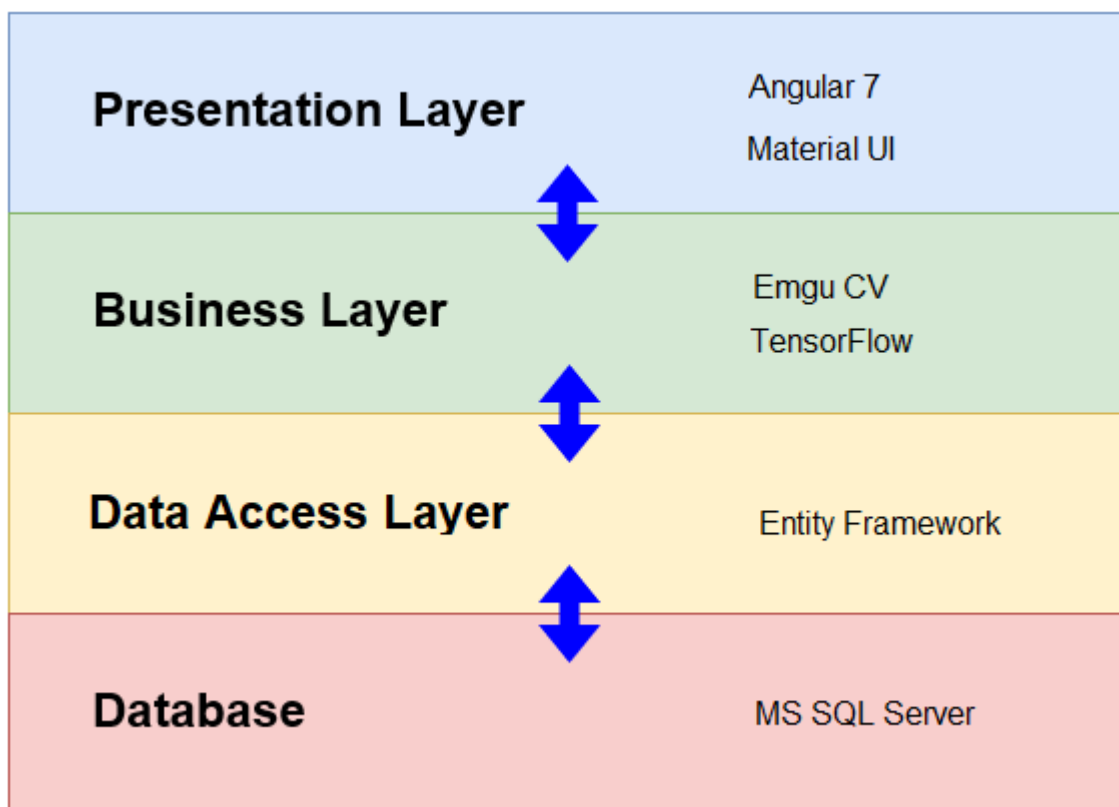


Рисунок 3.4 – Архітектура системи

Presentation layer - це той рівень, з яким безпосередньо взаємодіє користувач. Цей рівень включає компоненти для користувача інтерфейсу, механізм отримання даних від користувача. На даному рівні розташовані всі ті компоненти, які складають призначений для користувача інтерфейс (стилі, статичні сторінки html, javascript), а також контролери, об'єкти контексту запиту. Основою цього рівня

лежить проект, створений із застосуванням фреймворку Angular та дизайну Material UI.

Business layer (рівень бізнес-логіки) містить набір компонентів, які відповідають за обробку отриманих від рівня уявлень даних, реалізує всю необхідну логіку додатка, все обчислення, взаємодіє з базою даних і передає результат обробки. Саме на цьому рівні відбувається розпізнавання зображень.

Data Access layer (рівень доступу до даних) зберігає моделі, що описують використовувані сутності, також тут розміщуються специфічні класи для роботи з різними технологіями доступу до даних, наприклад, клас контексту даних Entity Framework. Тут також зберігаються репозиторії, через які рівень бізнес-логіки взаємодіє з базою даних.

## 4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

### 4.1 Вибір засобів розробки

Під час планування та моделювання програмного продукту було визначено, що проект має складатися з двох частин: сервер, на якому будуть зберігатися дані, та інтерфейс веб-додатку, розроблений із застосуванням Angular, що взаємодіє із сервером за допомогою Web API 2 - контролерів.

Для розробки серверу використовується технологія створення Web-застосувань мовою C#[19] з використанням ASP MVC 5 фреймворку. C# – об'єктно-орієнтована мова програмування, розроблена в 1998-2001 роках групою інженерів під керівництвом Андерса Хейлсберга в компанії Microsoft як мова розробки додатків для платформи Microsoft .NET Framework[20] і згодом була стандартизована як ECMA-334 і ISO / IEC 23270.

Для розробки використовується середовище Visual Studio Professional 2017. Visual Studio - продукт компанії Microsoft, що включає інтегроване середовище розробки програмного забезпечення і ряд інших інструментальних засобів. Дані продукти дозволяють розробляти як консольні додатки, так і додатки з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також веб-сайти, веб-додатки, веб-служби як в рідному, так і в керованому кодах для всіх платформ, підтримуваних Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework і Silverlight.

Visual Studio включає в себе редактор вихідного коду з підтримкою технології IntelliSense і можливістю найпростішого рефакторінга коду. Вбудований відладчик може працювати як відладчик рівня вихідного коду, так і відладчик машинного рівня. Решта вбудованих інструментів включають в себе редактор форм для спрощення створення графічного інтерфейсу додатку, веб-редактор, дизайнер класів і дизайнер схеми бази даних. Visual Studio дозволяє створювати і підключати сторонні додатки (плагіни) для розширення функціональності практично на



кожному рівні, включаючи додавання підтримки систем контролю версій вихідного коду (як, наприклад, Subversion і Visual SourceSafe), додавання нових наборів інструментів (наприклад, для редагування і візуального проектування коду на предметно-орієнтованих мовах програмування) або інструментів для інших аспектів процесу розробки програмного забезпечення (наприклад, клієнт Team Explorer для роботи з Team Foundation Server).

Для виклику методів сервера використовуються REST контролери з фреймворку ASP.NET MVC[21] , що дозволяє отримувати та відправляти різноформатні дані по протоколу HTTP.

На сьогоднішній день прийнято використовувати REST – (скор. від англ. RepresentationalStateTransfer – «передача репрезентативного стану») – метод взаємодії компонентів розподіленого додатка в мережі Інтернет, при якому виклик віддаленої процедури являє собою звичайний HTTP-запит, а необхідні дані передаються як параметри запиту.

У свою чергу HTTP – протокол передачі даних, що використовується в комп'ютерних мережах. Назва скорочена від Hyper Text Transfer Protocol, протокол передачі гіпер-текстових документів.

HTTP – протокол прикладного рівня, схожими на нього є FTP і SMTP. Обмін повідомленнями йде за звичайною схемою «запит-відповідь».

Для збереження даних програмного продукту була використана СКБД Microsoft SQL Server – система керування базами даних, дуже поширений вибір для додатків, розроблених із застосуванням платформи .NET.

Основна причина, чому Microsoft SQL Server є фаворитом розробників і адміністраторів віртуалізації, так це простота у використанні. MSSQL поставляється з відмінними інструментами, які заощадають багато часу в цих областях - інструменти, такі як SQL Server Profiler, SQL Server Management Studio, BI інструменти та бази даних Tuning Advisor.

Налаштовується майже все, від установки на віртуальну машину до початкового написання і редагування запитів, все це неймовірно легко з MSSQL - особливо в порівнянні з іншими продуктами SQL. Якщо є проблеми в будь-якій

стадії розвитку, існує онлайн-підтримка та документації у додаток до живої підтримки продукту, в той час як варіанти підтримки для інших продуктів SQL не є настільки надійними.

Доступ і маніпуляція даними в базі здійснюється засобами ORM Entity Framework, метою якого є звільнення розробника від значних типових завдань із програмування взаємодії з базою даних. Розробник може використовувати Entity Framework як при розробці з нуля, так і для вже існуючої бази даних.

Entity Framework піклується про зв'язок класів з таблицями бази даних (і типів даних мови програмування із типами даних SQL), і надає засоби автоматичної побудови SQL запитів й зчитування/запису даних, і може значно зменшити час розробки, який зазвичай витрачається на ручне написання типового SQL коду. Entity Framework генерує SQL виклики і звільняє розробника від ручної обробки результуючого набору даних, конвертації об'єктів і забезпечення сумісності із різними базами даних.

Сервер реалізований за допомогою наступних технологій:

- .NET Framework 4.5.1;
- ASP.NET MVC 5;
- Web API 2.

Сервіс представляє собою single page application – додаток, що містить лише одну HTML сторінку, а подальша взаємодія виконується за допомогою динамічно завантажуваних HTML, CSS та JavaScript із використанням Angular. При цьому роутинг конфігурується клієнтом додатку, який отримує та відправляє запити до сервера.

Angular представляє фреймворк від компанії Google для створення клієнтських додатків. Перш за все він націлений на розробку SPA-рішень (Single Page Application), тобто односторінкових додатків. В цьому плані Angular є спадкоємцем іншого фреймворка AngularJS. У той же час Angular це не нова версія AngularJS, а принципової новий фреймворк.

Angular надає таку функціональність, як двостороннє зв'язування, що дозволяє динамічно змінювати дані в одному місці інтерфейсу при зміні даних

моделі в іншому, шаблони, маршрутизація і так далі. Однією з ключових особливостей Angular є те, що він використовує в якості мови програмування TypeScript.

Окрім цього, для створення інтерфейса [22] був використаний сучасний фреймворк Material Design.

Material Design - це мова дизайну для веб- і мобільних додатків, який був розроблений Google в 2014 році. Material Design спрощує розробникам налаштування UI, зберігаючи при цьому зручний інтерфейс додатків. Material Design надає добре організований формат і гнучкість.

Angular Material складається з набору попередньо встановлених компонентів Angular. На відміну від Bootstrap, який надає компоненти, які можна використовувати будь-яким способом, Angular Material прагне забезпечити розширений і послідовний інтерфейс. У той же час він дає можливість контролювати, як поведуться різні компоненти.

Фронт-енд додатку в повній мірі може функціонувати на таких браузерах: Google Chrome, Internet Explorer, Opera, Mozilla Firefox та інших браузерах, версії яких підтримують JavaScript, CSS3 та HTML5.

Керування життєвим циклом об'єктів на сервері здійснюється за допомогою Інверсії управління.

Якщо описувати модель взаємодії усіх компонентів системи, то можна зрозуміти, що головною ланкою є сервер з REST-сервісом.

## 4.2 Опис програмної системи

Як вже було неодноразово сказано, програмна система складається з двох компонентів: серверу та веб-клієнта.

На сервері реалізований REST-сервіс, за допомогою якого веб-клієнт використовує функції системи. Методи, що вертають дані, вертають їх у форматі JSON.

Під час розробки програмної системи однією з цілей було створити максимально простий та зрозумілий інтерфейс веб-клієнта, щоб у користувача не виникало труднощів із завантаженням зображень та пошуком товарів.

Веб-клієнт повністю відповідає принципам юзабіліті з точки зору інтерфейсу. Якщо користувач ще не авторизувався у системі, його буде перенаправлено до сторінки автентифікації (див. рис. 4.1).

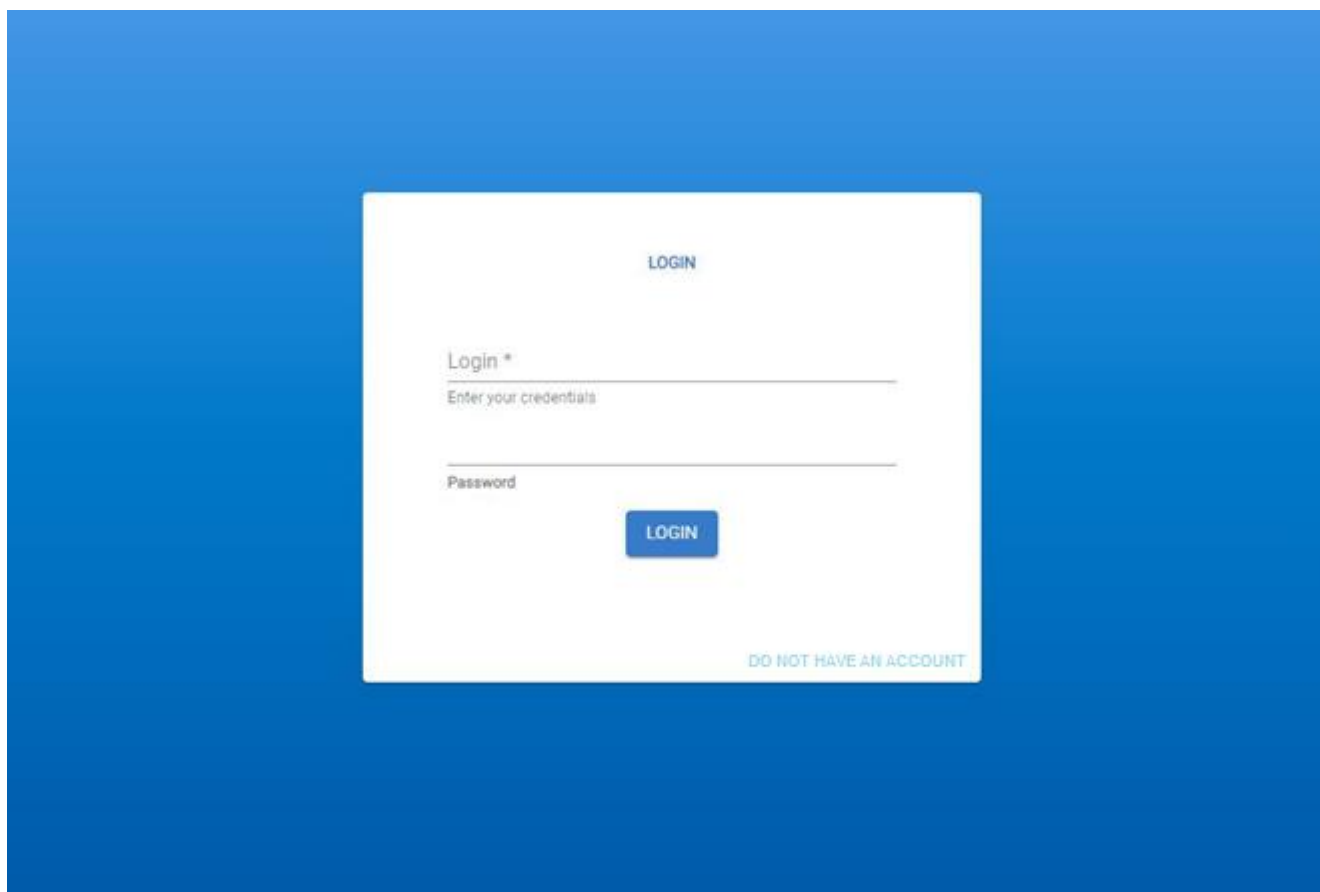


Рисунок 4.1 – Сторінка автентифікації

Ця сторінка є формою для вводу даних. Користувач обов'язково повинен ввести свій логін та пароль від акаунту. Тут присутня валідація, що не дозволить користувачеві вести некоректні дані. Валідація присутня як на сервері, так і на клієнті. У випадку введення невірних даних користувача буде повідомлено за

допомогою спливаючого вікна та підсвічення невірно заповнених полів. Якщо користувач ще не має аккаунту в системі, його необхідно зареєструвати.

Після авторизації користувач потрапляє на головну сторінку (див. рис. 4.2) зі списком попередніх пошуків товару.

Цей список надає основну інформацію про товар, який було ідентифіковано, а саме:

- назву;
- категорію, що була визначена нейронною мережею при аналізі зображення;
- артикул товару, необхідний для пошуку його в онлайн магазинах.

Окрім даних про товар, в історії користувачу надається можливість відкрити відповідний об'єкт на сторінці інтернет магазину, заповнити форму на редагування, якщо були знайдені помилку в назві, артикулі або невірно визначена категорія. Також користувач може окремо видаляти записи з історії пошуку.

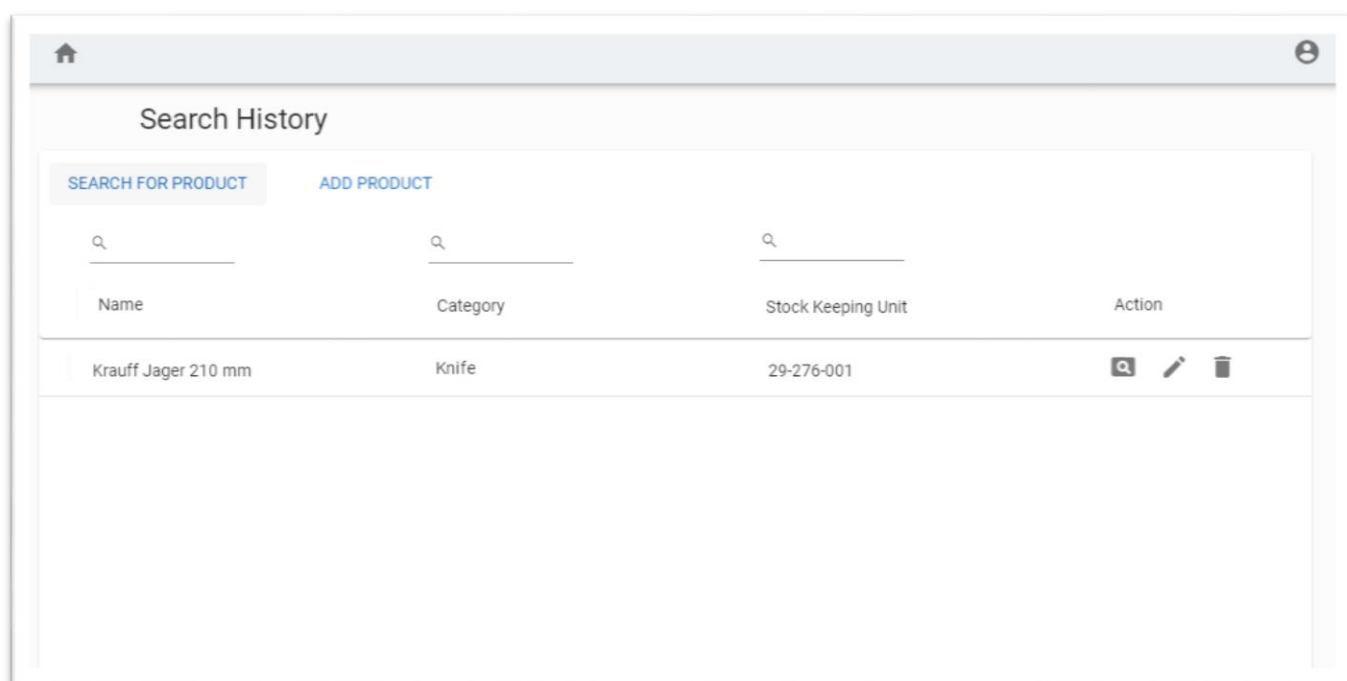


Рисунок 4.2 – Головна сторінка

Головна сторінка дає користувачу доступ до всіх основних функцій системи, а саме до пошуку та додавання нових товарів до бази пошуку.

При натисненні на кнопку “Search for product” користувачу відкриється діалог із формою для завантаження зображення (див. рис. 4.3).

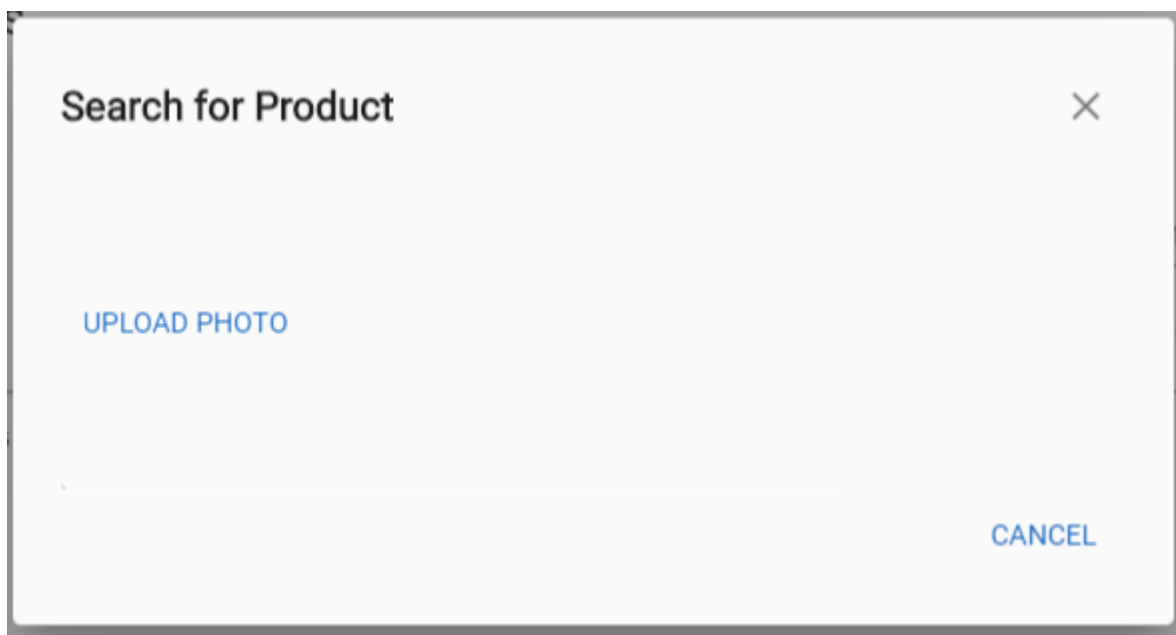


Рисунок 4.3 – Діалог пошуку продукту до завантаження зображення

Після успішного завантаження почнеться процес обробки зображення. По-перше буде виділено ключовий об’єкт на фото, а все зайве відкинуто. Цю роботу виконає нейронна мережа, яка також класифікує об’єкт за типом. Завдяки цьому зображення для порівняння будуть обиратися лише з визначеної категорії. Це значно зменшує кількість операцій на сервері та в декілька разів прискорює роботу програми. Наступним етапом є оптимізація зображення. На рисунку 4.4 наведено фрагмент коду, який виконує оптимізацію зображення для подальшої обробки.

```
private void OptimizeImage(Mat modelImage, IOutputArray outputArray)
{
    → var blackAndWhiteImage = new Image<Gray, byte>(modelImage.Bitmap).Mat;
    → CvInvoke.GaussianBlur(blackAndWhiteImage, outputArray, new Size(7, 7), 0);
    →
}
```

Рисунок 4.4 – Метод із оптимізації зображення

Метод `OptimizeImage` виконує дві основних дії. Перша – це приведення кольорового зображення до чорно-білої гами. Другим та найважливішим кроком є гаусове розмиття. Головна роль розмиття – це усунення шумів із зображення. Таким чином, у результаті виконання цього методу змінній `outputArray` буде задано масив пікселів, готових до аналізу.

Після цього зображення порівнюється із іншими зразками методом SURF. На рисунку 4.5 зображений перший етап аналізу зображення – пошук ключових точок та дескрипторів.

```
modelKeyPoints = new VectorOfKeyPoint();
surf.DetectAndCompute(image, null, modelKeyPoints, modelDescriptors, false);
```

Рисунок 4.5 – Фрагмент коду із пошуком ключових точок та дескрипторів

В даному випадку використовується вбудований в бібліотеку EmguCV клас SURF. Виклик методу `DetectAndCompute` визначає дескриптори для відповідного зображення, яке було передано параметром. Наступним кроком є порівняння дескрипторів. Фрагмент коду, що за це відповідає, показано на рисунку 4.6.

```
BFMatcher matcher = new BFMatcher(DistanceType.L2);
matcher.Add(modelDescriptors);
indices = new Matrix<int>(observedDescriptors.Rows, k);
using (VectorOfVectorOfDMatch dist = new VectorOfVectorOfDMatch())
{
    matcher.KnnMatch(observedDescriptors, dist, k, null);
    mask = new Matrix<byte>(dist.Size, 1);
    mask.SetValue(255);
    Features2DToolbox.VoteForUniqueness(dist, uniquenessThreshold, mask.Mat);
}
```

Рисунок 4.6 – Порівняння дескрипторів

В результаті такого порівняння залишаються лише унікальні дескриптори, за якими ми можемо аналізувати зображення.

Порівнявши дескриптори зображення та відкинувши ненадійні, ми маємо все необхідне для останнього етапу обробки дескрипторів ключових точок – перевірку

їх відповідності незалежно від масштабу та кута повернення зображення. Відповідний фрагмент коду показано на рисунку 4.7.

```
nonZeroCount = Features2DToolbox.VoteForSizeAndOrientation(modelKeyPoints,
    ....observedKeyPoints, indices, mask, 1.5, 20);
```

Рисунок 4.7 – Перевірка точок на стійкість до масштабування та повернення зображення

Після зваження всіх залишившихся дескрипторів найкраще співпадіння буде відображене в діалозі пошуку (див. рис. 4.8).

Після знаходження відповідного зображення в першій вкладці діалогу буде виведено порівняння зображень: завантажене користувачем – ліворуч, а знайдене відповідне йому – праворуч. Нижче буде опис товару, та кнопки для відображення ще декількох співпадінь та закриття діалогу.

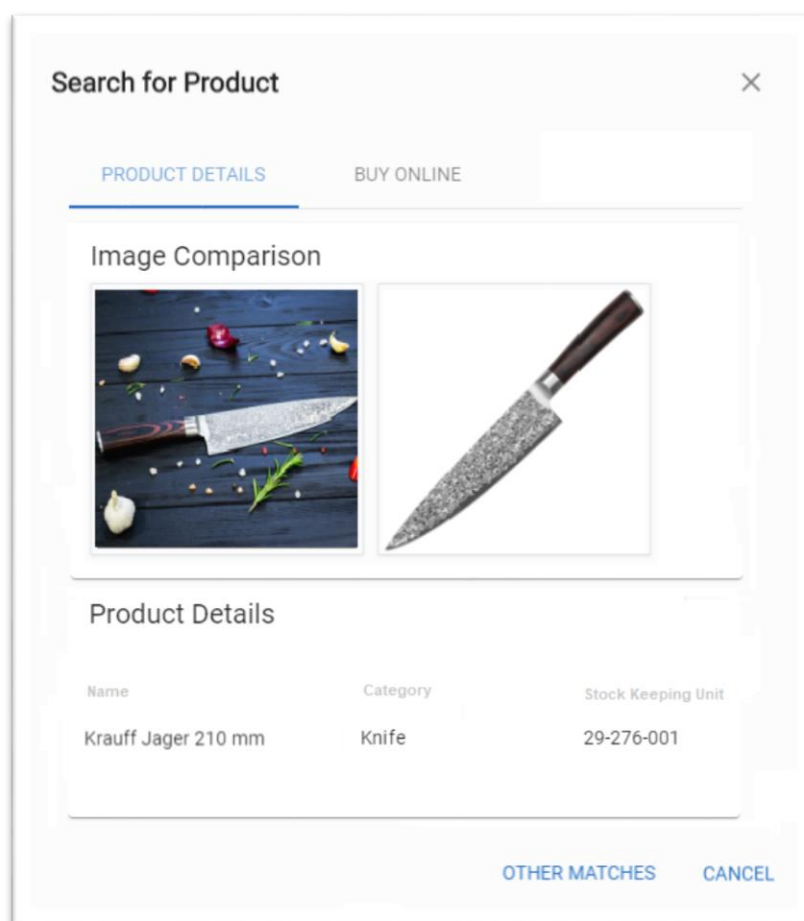


Рисунок 4.8 – Діалог пошуку зі знайденим співпадінням



На другій вкладці (див. рис. 4.9) знаходяться посилання на онлайн магазини та коротка інформація про даний товар.

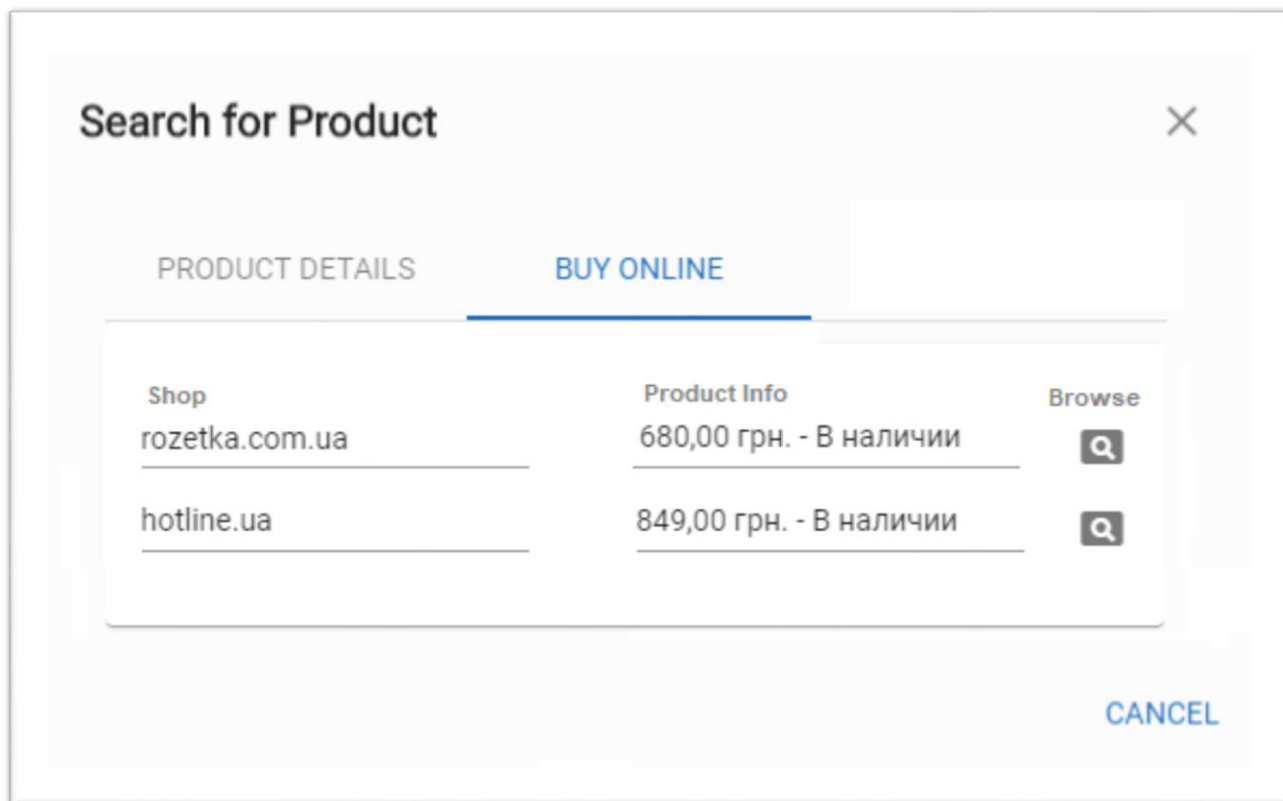


Рисунок 4.9 – Вкладка із посиланнями на магазини

Окрім наведених функцій користувач також може очистити список пошуку товарів, вибравши відповідний пункт у меню, що відкривається при натисканні на значок користувача на верхній панелі (див. рис. 4.10).

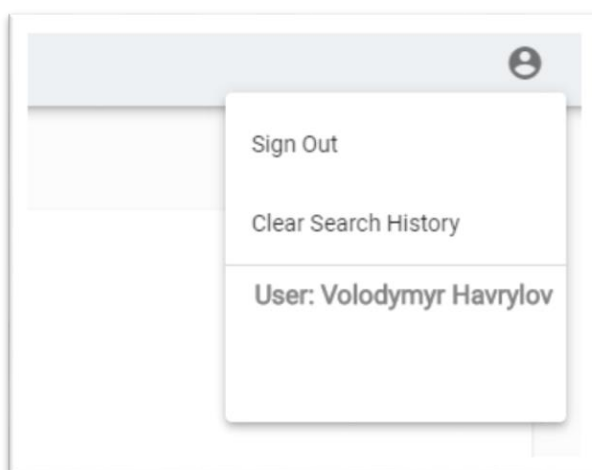


Рисунок 4.10 – Меню користувача

В цьому меню також знаходиться кнопка виходу з поточного акаунту та відображений власник дійсної сесії.

Після виходу з сервісу користувача буде перенаправлено на сторінку логіну, а всі операції з перегляду історії, відвідування інтернет-магазинів чи пошуку товарів будуть недоступні.

Слід також зазначити, що завантаження веб-сторінок є повністю автоматичним завдяки двосторонньому зв'язуванню об'єктів в Angular. Це дозволить користувачу завжди бачити актуальну інформацію на своєму моніторі та стежити за станом списку товарів без потреби постійно перевантажувати сторінку в браузері.

## ВИСНОВКИ

В процесі роботи був проведений аналіз предметної галузі програм для розпізнавання образів та їх використання у E-commerce. Були досліджені основні фреймворки та технології, що застосовуються у сучасних системах машинного зору, а саме:

- фреймворки для машинного навчання (Tensor Flow, Microsoft CNTK, Caffe, Theano);
- бібліотеки машинного зору (Open CV, Matlab, PCL);
- алгоритми перетворення та порівняння зображень.

Це дослідження відіграло важливу роль у виборі ефективних методів розпізнавання а також побудови розширюваного програмного продукту із можливістю ідентифікації товару в інтернет магазинах за зображенням.

У результаті виконання роботи був розроблений програмний продукт, що дозволяє на основі зображення ідентифікувати та шукати товар в E-commerce платформах, такий як інтернет магазини.

Система базується на клієнт-серверній архітектурі, тому складається з двох окремих частин: сервера та веб-клієнта.

Сервер було розроблено з використанням мови програмування C# у середовищі Microsoft Visual Studio 2017 Professional, клієнтська частина була написана переважно на мові TypeScript із застосуванням фреймворку Angular. Сервер розроблявся з використанням фреймворку ASP.NET MVC 5. Для бази даних була застосована СКБД Microsoft SQL Server 2016.

В результаті виконання роботи було виконано наступні задачі:

- розроблено сервер з сервісами, які надають змогу працювати одночасно різними типами клієнтів з системою;
- розроблено веб-клієнт з достатнім функціоналом та зручним інтерфейсом.

В результаті розробки поставлену задачу було цілком виконано. Програмний продукт має зрозумілий інтерфейс для користувачів, не викликає труднощів з відправкою даних на сервер.

Під час виконання роботи були набуті навички з застосування фреймворків глибинного навчання і технологій машинного зору. Також були поглиблені знання стосовно проектування баз даних, вивчено і втілено багато нового стосовно розробки клієнтських засобів на мові Typescript.

Програмний продукт навіть після реалізації усіх поставлених задач, для комерційного запуску має дороблюватися. Перш за все, необхідно доробити додатковий функціонал та створити комерційну складову проекту.

У подальшому планується розширити функціональну складову розробленої системи шляхом додавання наступних можливостей:

- налаштувати механізм оскарження інформації про товар, що не відповідає дійсності;
- ввести можливість придбати товар не покидаючи сторінку веб-додатку;
- розробити утиліту для додаткового навчання нейронної мережі.

Другорядним завданням стоїть реалізація клієнтів для різних мобільних платформ (iOS, OS Android, Windows Phone), це дозволить покрити більшість сучасних смартфонів та планшетів. Також, необхідно доробити локалізацію.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. What is ecommerce/ DynamicWebs. [Електронний ресурс]. – Режим доступу: <http://www.dynamicwebs.com.au/tutorials/e-commerce.htm>
2. Google Lens: What is it and how does it work? [Електронний ресурс]: Chris Hall, Britta O'Boyle, 2018. – Режим доступу : <https://www.pocket-lint.com/apps/news/google/141075-what-is-google-lens-and-how-does-it-work-and-which-devices-have-it>
3. Flow powered by Amazon App [Електронний ресурс]. – Режим доступу : <https://www.businesswire.com/news/home/20111102007116/en/Flow-Powered-Amazon-App>
4. Vivino – how not to forget your favourite wine [Електронний ресурс]: Радужная Н., 2012. – Режим доступу : <https://lifel hacker.ru/vivino/>
5. Lieder, Itay, Tom Hope , Yehezkel S. Resheff Learning TensorFlow: A Guide to Building Deep Learning Systems. Gravenstein Highway North, Sebastopol: O Reilly, 2017.-244p.
6. Theano [Електронний ресурс]. – Режим доступу : <http://www.deeplearning.net/software/theano/>.
7. Bradski, Gary, Adrian Kaehler. Learning OpenCV. Gravenstein Highway North, Sebastopol: O Reilly, 2008.-556p.
8. Lecarme, Oliver. The Book of GIMP: A Complete Guide to Nearly Everything. San Francisco, Calif.: William Pollock, 2013.-658p.
9. SURF: Speeded Up Robust Features [Електронний ресурс]: Herbert Bay, Tinne Tuytelaars, and Luc VanGool. – Режим доступу: <https://www.vision.ee.ethz.ch/~surf/eccv06.pdf>
10. Distortion Invariant Object Recognition [Електронний ресурс] – Режим доступу <http://ieeexplore.ieee.org/document/210173/>
11. Image Analysis With Convolutional Neural Networks [Електронний ресурс] – Режим доступу <http://ieeexplore.ieee.org/document/554195/>

12. Введение в OpenCV [Электронный ресурс] – Режим доступа <http://robocraft.ru/blog/computervision/264.html>
13. Image Analysis With Convolutional Neural Networks, Stefan Duffner, 2007 [Электронный ресурс] – Режим доступа <https://pdfs.semanticscholar.org/dbb7/f37fb9b41d1aa862aaf2d2e721a470fd2c57.pdf>
14. The analysis of the existing approaches to object recognition [Электронный ресурс] – Режим доступа <http://developers-club.com/posts/238129/>
15. Аналіз існуючих підходів до розпізнавання об'єктів [Електронний ресурс] – Режим доступу <http://it-ua.info/news/2014/09/25/analz-snuyuchih-pdhodv-do-rozpznnavannya.html>
16. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 24, Issue 7, 971–987 (2002)
17. Фаулер, М. UML. Основы [Текст] : пер. с англ. А.: Петухов/ М. Фаулер, К. Скотт. - СПб.: Символ, 2017. - 184 с.
18. Мюлер Джордж Р. Проектирование баз данных и UML – Москва: Лори, 2013. – 432 с.
19. Пугачев, С. Разработка приложений для Windows на языке С# [Текст] / С.В. Пугачов, А.М. Шерієв, К.А. Кичинський. – СПб.: БХВ-Петербург, 2013. – 416 с.
20. Рихтер, Дж. CLR via C# Программирование на платформе Microsoft .NET Framework 4.0 на языке C# [Текст] / Дж. Рихтер, пер. с англ. Радченко И., Рузмайкина И. – СПб. Питер, 2013. – 428 с..
21. Шилдт, Г. C#4.0: Полное руководство [Текст] / Г. Шилдт. ; пер. с англ. И. Берштейн. – М.: ООО «И.Д. Вильямс», 2011. – 356 с.
22. Этапы разработки пользовательского интерфейса [Электронный ресурс] – Режим доступа <http://www.4stud.info/user-interfaces/stages-of-development-user-interface.html>