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 другий (магістерський)

ДОСЛІДЖЕННЯ МЕТОДІВ АВТОМАТИЧНОГО ФОРМУВАННЯ ПАНОРАМ З МНОЖИН ЗОБРАЖЕНЬ
(тема)

Виконав:
студент 2 курсу, групи ІНФм-22-2

Россіна Т.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 проф. Машталір В.П.
(посада, прізвище, ініціали)

Допускається до захисту

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

2024 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Россіній Тамарі Сергіївні
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження методів автоматичного формування панорам з множин зображень

затверджена наказом по університету від 3 листопада 2023 року № 1280Ст

2. Термін подання студентом роботи до екзаменаційної комісії 22 грудня 2023р.

3. Вихідні дані до роботи математичні моделі детектування та вибору ключових точок на зображеннях, теоретичні відомості про методи, що використовується для оцінки параметрів математичної моделі для набору спостережуваних даних, Affine Covariant Regions Datasets, OXFORD's Affine Covariant Regions Datasets.

4. Перелік питань, що потрібно опрацювати в роботі

1. Огляд існуючих методі та оцінка ефективності підходів до автоматичного формування панорам.

2. Розробка нового метода, спрямований на вдосконалення процесу автоматичного зшивання зображень.

3. Розробка експериментального дослідження з використанням реальних або синтетичних даних для об'єктивної оцінки ефективності.

4. Розробка програмного додатоку для порівняння методів.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність проблеми обраної теми, постановка задачі, обрані методи детектування та вибору ключових точок на зображеннях, тестові зображення.

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

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	03.11.2023	
2	Аналіз завдання, підбір літератури	4.11.23-6.11.23	
3	Аналіз літератури з досліджуваної проблеми	7.11.23-13.11.23	
4	Аналіз методів вибору ключових точок	13.11.23-16.11.23	
5	Розробка системи автоматичного формування панорам	17.11.23-22.11.23	
6	Програмна реалізація	22.11.23-27.11.23	
7	Оформлення пояснювальної записки	27.11.23-2.12.23	
8	Перевірка на плагіат	06.12.2023	
9	Рецензування	10.12.2023	
10	Підготовка презентації та доповіді	24.12.2023	
11	Занесення роботи в електронний архів	01.01.2024	
12	Попередній захист кваліфікаційної роботи	03.01.2024	

Дата видачі завдання 3 листопада 2023 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Машталір В.П.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 70 с., 2 табл., 29 рис., 45 джерел.

ПАНОРАМНІ ЗОБРАЖЕННЯ, КЛАСТЕРИЗАЦІЯ, SURF, SIFT, ORB, ЗШИВАННЯ ЗОБРАЖЕНЬ, RANSAC, БІБЛІОТЕКА OPEN CV, PANORAMA STITICHING.

Об’єктом дослідження є процес автоматичного об’єднання зображень для отримання панорами.

Метою дослідження є роботи є аналіз існуючих методів та розробка нових підходів для підвищення якості та ефективності формування панорам.

На основі проаналізованих досліджень було розроблено власний алгоритм розв’язання задачі автоматичного об’єднання зображень. При розробці даного алгоритму було враховано переваги та недоліки вже досліджених алгоритмів. Розроблений алгоритм зшивання зображень вимагає декілька кроків: виявлення ключових точок і вилучення локальних інваріантних дескрипторів, зіставлення дескрипторів між зображеннями, застосування RANSAC для оцінки матриці гомографії, і застосування зшивання з використанням матриці гомографії.

У результаті дослідження здійснена програмна реалізація системи.

PANORAMIC IMAGES, CLUSTERING, SURF, SIFT, ORB, IMAGE STITCHING, RANSAC, OPEN CV LIBRARY, PANORAMA STITCHING.

The object of research is the process of automatic image merging to obtain a panorama.

The aim of the research is to analyze existing methods and develop new approaches to improve the quality and efficiency of panorama formation.

Based on the analyzed research, we developed our own algorithm for solving the problem of automatic image fusion. When developing this algorithm, the advantages and disadvantages of the already studied algorithms were taken into account. The developed image fusion algorithm requires several steps: detection of key points and extraction of local invariant descriptors, comparison of descriptors between images, application of RANSAC to estimate the homography matrix, and application of fusion using the homography matrix.

As a result of the research, the system was programmatically implemented.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

SURF – Speeded-Up Robust Features

SIFT – Scale Invariant Feature Transform

ORB – Oriented FAST and Rotated BRIEF

FAST – Features from Accelerated Segment Test

BRIEF – Binary Robust Independent Elementary Features

BRISK – Binary Robust Invariant Scalable Keypoints

NN – Nearest Neighbor

k-NN – *k*-Nearest Neighbor

RANSAC – RANdom SAmple Consensus

ВСТУП

У сучасному світі, де розробники штучного інтелекту намагаються дати комп’ютерам можливість «бачити» так само, як і людина, виникає актуальне завдання дослідження та розробки алгоритмів для автоматичного формування панорам з множин зображень. Ця технологія важлива для створення віртуальних турів, карт, систем віртуальної реальності, тощо. Зростання популярності цифрових зображень визначає важливість розробки ефективних алгоритмів формування панорам для відображення якнайбільше деталей.

Панорамні зображення стають важливим елементом сучасної фотографії, що дозволяє створювати унікальні фотокарти, які вирізняються великим кутом огляду та можливістю вміщувати більше деталей, ніж традиційні методи фотографування. Вони перетворюють сприйняття міст, космосу та природи, занурюючи у новий світ цифрових зображень. Дана дослідницька робота спрямована на вдосконалення алгоритмів автоматичного формування панорам для досягнення високої якості та результатів.

Актуальність дослідження методів автоматичного формування панорам з множин зображень визначається низкою ключових факторів, що впливають на різноманітні сфери нашого сучасного життя. З підвищенням зацікавленості людей у фотографії та розвитком соціальних мереж попит на вражаючі та деталізовані панорамні зображення постійно зростає. Розвиток методів автоматичного формування панорам дозволить користувачам створювати унікальні та привабливі зображення без значних зусиль.

У галузі медичної діагностики створення панорамних зображень має важливе значення для аналізу та візуалізації медичних даних, таких як знімки з магнітно-резонансної томографії чи комп’ютерної томографії. Ефективні та точні методи формування панорам можуть поліпшити процес діагностики та аналізу. У робототехніці панорамні зображення можуть бути використані для поліпшення систем візуального сприйняття роботів, що може знайти застосування у роботах для автономної навігації, вирішенні завдань

у робототехнічній сфері. У сфері безпеки та нагляду за великими територіями панорамні зображення можуть допомагати в ефективному візуальному моніторингу та виявленні подій. В туризмі та геодезії формування панорам стає ключовим для створення вражаючих віртуальних турів та картографічних даних.

Дані аспекти підкреслюють важливість дослідження та вдосконалення методів автоматичного формування панорам, оскільки це дозволяє відповідати потребам різних галузей та розвивати технології для подальшого впровадження в практику.

1 СИСТЕМИ АНАЛІЗУ ТА ОБРОБКИ ЦИФРОВИХ ЗОБРАЖЕНЬ

1.1 Поняття про панорамні зображення

Останні досягнення в галузі високошвидкісних комунікацій та штучного інтелекту призвели до збільшення попиту на оновлення традиційних систем зображення. Панорамне зображення має ширший кут огляду порівняно з традиційними оптичними системами і може зафіксувати більше інформації про оточуюче середовище за один раз. Їх використовують в автономному водінні, машинному баченні, ендоскопічній медицині, супутниковому атмосферному моніторингу та багатьох інших сферах. Щодо якості зображення, панорамне зображення стикається з декількома загальними викликами, щоб задовольнити вимоги застосувань, таких як машинне бачення, включаючи роздільність, відсутність сліпих зон та якість зображення. Ще однією важливою вимогою є багатовимірне розуміння, де очікується, що панорамне зображення буде поєднано з розумними датчиками для запису, об’єднання та сприймання більш високовимірної інформації про оточуюче середовище. Крім того, панорамне зображення буде еволюціонувати в напрямку легких і малов’язких структур для застосувань у сценаріях, де обмежені простір та вага. Однак ці вимоги, як правило, передбачають вирішення різних компромісів, що робить проектування інструментів панорамного зображення особливо складним.

Панорамне зображення пережило свій перший бум у 1990-х роках, але цей бум поступово зменшився через відсутність кваліфікованого обладнання для виробництва та систем обробки цифрової інформації [1]. Протягом останнього десятиліття концепцію панорамного зображення переглянули і вона знову переживає бум. Новітні технології, такі як вільні форми поверхонь і метаповерхні перетворили системи панорамного зображення і пролили світло в цьому напрямку. Застосування цих нових оптичних технологій створило сильний імпульс для покращення продуктивності архітектур

оптичних систем панорамного зображення. Тим часом, пропозиція багатовимірної панорамної системи виявлення збагатила панорамне зображення та відіграла більш потужну роль у різних сферах застосування. Панорамна фотографія, також відома як широкоформатна фотографія – це спеціальна техніка, яка об’єднує кілька зображень з однієї камери в одну широку фотографію вертикальну або горизонтальну. Термін «панорама» в перекладі з грецької буквально означає «увесь огляд», і вперше його придумали митці, які хотіли зафіксувати широкий огляд пейзажу, а не лише його певну частину [2].

Перші панорамні фотографії робили шляхом простого вирівнювання друкованих версій плівки, що виходило не дуже добре, оскільки ідеально вирівняти фотографії було майже неможливо. З винаходом персональних комп’ютерів, прогресом у комп’ютерному програмному забезпеченні та цифровій фотографії тепер стало набагато легше з’єднувати цифрові зображення за допомогою спеціального програмного забезпечення. Фактично, використовуючи відповідну техніку фотографії та панорамне обладнання, тепер можна створювати майже ідеальні панорами з надзвичайно високою роздільною здатністю. Деякі фотографи навіть зшивають сотні зображень високої роздільної здатності, щоб створити гігантські «гігапіксельні» панорами.

На сьогоднішній день цифрова панорамна зйомка досить популярна і поширена не тільки серед пейзажистів, а й серед фотографів архітектури та міських пейзажів. Панорамна фотографія може бути досить складною та дорогою, залежно від цілей. Наприклад, створення панорамних зображень в архітектурній фотографії вимагає належного калібрування камери та об’єктива на спеціальному панорамному обладнанні, щоб запобігти кривим лініям, спотворенням і неправильному зшиванню близьких об’єктів. У той же час можна успішно робити чудові пейзажні панорамні зображення, не вкладаючи коштів у будь-яке обладнання для камери. Загалом, панорамна фотографія передбачає кілька фотографій і з’єднаних разом у програмному

забезпеченні. Приклад панорамного процесу, який об’єднує кілька фотографій в одне остаточне зображено на рисунку 1.1.

Рисунок 1.1 – Приклад панорамного процесу

Хоча слово «панорама» автоматично передбачає, що це буде широке горизонтальне або вертикальне зображення, але це не обов’язково. Якщо з’єднати кілька зображень і виявиться квадратне зображення, то це панорамне зображення високої роздільної здатності.

Ширококутні панорами – все, що виглядає як ширококутна фотографія, яка охоплює менше 180 градусів, горизонтально чи вертикально. Ширококутні панорами можуть навіть виглядати як звичайні зображення, за винятком того, що вони складаються з кількох фотографій і тому мають більшу роздільну здатність.

180 градусні панорами – панорами, які охоплюють 180 градусів зліва направо. Ці типи панорам виглядають дуже широкими, охоплюючи велику площу.

Сферичні панорами – також відомі як «планети». Це 360-градусні панорами, які перетворюються на квадратне сферичне зображення за допомогою спеціальної техніки постобробки. Зшивання зображень набуває

все більшої популярності завдяки різноманітним повсякденним застосуванням та науковому значенню.

Процес зшивання зображень передбачає поєднання двох або більше зображень, як засіб отримання ширшого поля зору, зберігаючи при цьому якомога більше інформації зі сцени. Традиційно два або більше зображень ретельно вирівнювали одне біля одного і вручну комбінували, щоб створити велике зображення [3, 4]. Ця техніка була не позбавлена обмежень, і спотворення об’єктива або дисбаланс кольору та щільності між зображеннями було звичайним явищем. Ці обмеження, зрештою, робили процес зшивання процес зшивання нестабільним, а в деяких випадках і ненадійним. Завдяки розвитку цифрових технологій, інструменти для редагування цифрових зображень стали широко доступними. Досягнення зробили процес зшивання набагато практичнішим і надійнішим завданням. Розробляються більш інноваційні та досконалі методи обробки, щоб не відставати від зростаючого обсягу даних, що створюються. Подібно до традиційної, ручної форми зшивання зображень, процес цифрового зшивання зображень має на меті об’єднати два або більше зображень для створення великого композитного зображення або панорами.

Процес цифрового зшивання зображень передбачає вирівнювання, змішування та зшивання разом декількох цифрових зображень, які містять ділянки, що перекриваються, в одне зображення з високою роздільною здатністю зображення з високою роздільною здатністю за допомогою складних алгоритмів [5].

Перший крок процесу, етап вирівнювання або розпізнавання, який передбачає порівняння сегментів зображень, що перекриваються, для знаходження ключових опорних точок між зображеннями. Комп’ютерний алгоритм знаходить відповідні ознаки між вхідними зображеннями. У деяких випадках, залежно від об’єкта, процес може бути складнішим, оскільки деякі опорні точки можуть виглядати схожими на інші, що може спричинити

невідповідності в процесі зшивання. Приклад вилучення ключових точок зображено на рисунку 1.2.

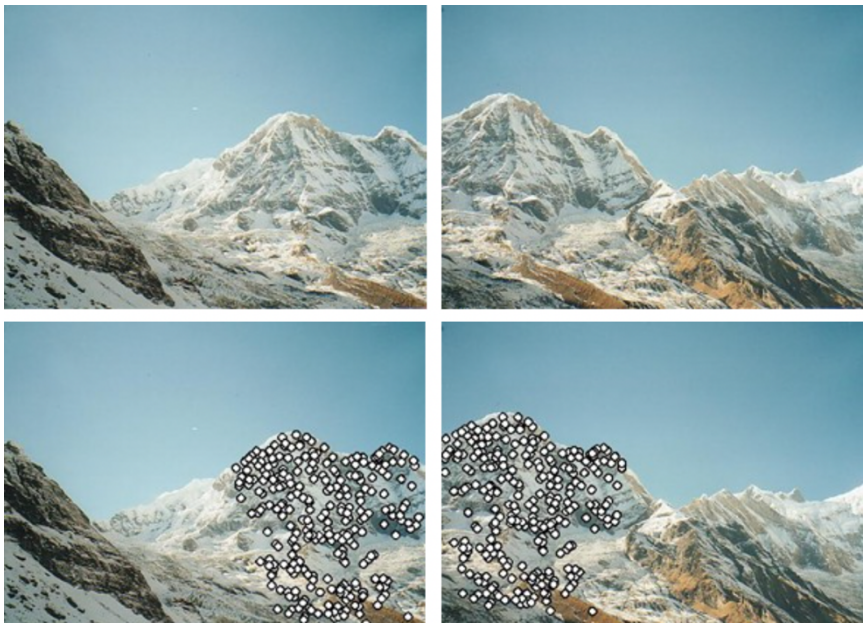


Рисунок 1.2 – Вилучення ключових точок і локальних інваріантів

Програма обчислює які зображення містять найбільше збігів, і згодом з’єднує ці зображення. Після вирівнювання зображень, що перекриваються, їх об’єднують так, щоб межі між сусідніми зображеннями були видимо безшовними, зберігаючи при цьому високу якість вихідних зображень. Злиття відіграє важливу роль у створенні безшовного кінцевого зображення, зменшуючи різницю в експозиції, радіальні спотворення, неспіввісність зображень або віньєтування між сусідніми зображеннями. Це композитне зображення з високою роздільною здатністю є кінцевим продуктом і називається панорамою [6]. Якщо вихідні зображення були зроблені з різних точок зору або з різних площин, отримана автоматизована панорама може містити артефакти, такі як неспівпадіння в процесі зшивання. У деяких випадках процес накладання може зменшити появу артефактів, на перший погляд панорама здається в ідеальному стані, але при детальному огляді на кінцевому панорамному зображенні все ще присутні неспівпадіння або розриви в зшиванні. Обмеження, пов’язані з панорамною зйомкою, вказують

на те, що оптимізація параметрів, пов’язаних з панорамною зйомкою, як у процесі захоплення, так і в процесі зшивання, була б важливою для покращення розмірної цілісності отриманих зображень. зображень.

1.2 Алгоритми пошуку особливих точок

Об’єкт можна ідентифікувати за кольором, текстурою, плямою, формою або будь-якою іншою ознакою. Ефективність системи розпізнавання об’єктів в основному залежить від особливих точок витягнутих з бази даних зображень. Розглянемо три алгоритми пошуку особливих точок, а саме: SIFT, SURF, та ORB.

Масштабне інваріантне перетворення ознак (SIFT) – це локальний детектор і дескриптор ключових точок, який був запропонований Девідом Лоу у 2004 році. Цей алгоритм виокремлює особливості точок враховуючи різні масштаби, обертання, освітлення та геометричні перетворення [7]. SIFT зарекомендував себе як найпоширеніший алгоритм розпізнавання об’єктів. Він отримує зображення на вході і видає набір ознак на виході. Вихідний вектор ознак SIFT має розмір (p×128), де p використовується для кількості виявлених ключових точок, а 128 для дескрипторів ключових точок. SIFT працює в чотири етапи:

Крок 1. Виявлення екстремумів у масштабному просторі.

Крок 2. Локалізація ключових точок.

Крок 3. Призначення орієнтації.

Крок 4. Дескриптор ключової точки.

SIFT будує піраміду з різною роздільною здатністю над вхідним зображенням. Спочатку застосовується метод різниці Гаусса застосовується для виявлення локальних екстремумів у просторовому масштабі. Переважні екстремуми розглядаються ключовими точками. По-друге, більш точне розташування ключових точок визначається за допомогою порогового

значення. На третьому етапі задається орієнтація для опису ключових точок з інваріантністю до повороту зображення [8–10]. На останньому етапі обчислюється набір з 128 дескрипторів ключових точок, які утворюють 16 блоків розміром 4×4, кожен з яких створює 8 гістограм. Приклад представлення дескриптора SIFT для фрагмента 16×16 пікселів і Гаусовій ваговій функції масиву дескрипторів розміру 4×4 зображено на рисунку 1.3.

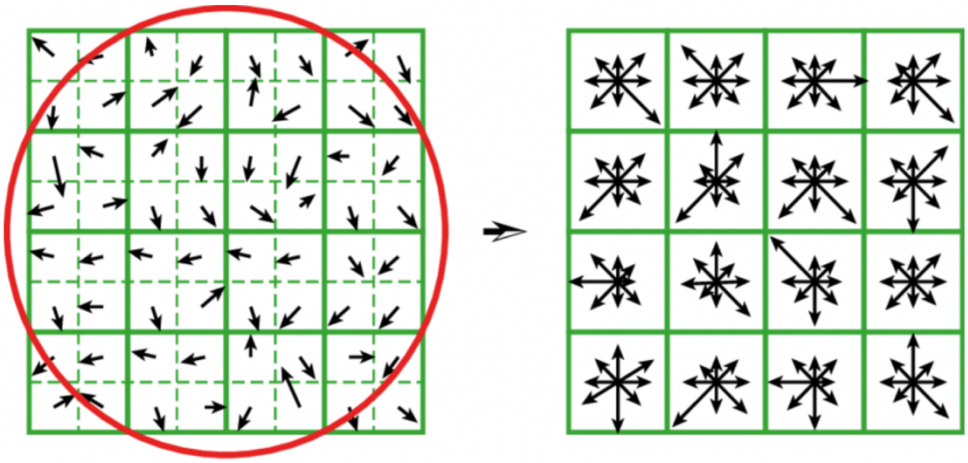


Рисунок 1.3 – Схематичне представлення дескриптора SIFT

Переваги SIFT – це результати для повернених і масштабованих зображень, він інваріантний до освітлення та точок зору, дає точніші результати, ніж інші дескриптори. Недоліки SIFT – це математично складний алгоритм, потрібно більше часу для обчислення вектора ознак, іноді не дає точних результатів на розмитих зображеннях, і не дає точних результатів для викривлених зображень.

Прискорена функція Robust Feature (SURF), введена Х. Беєм. Це прискорена версія SIFT. SURF також є локальним детектором і дескриптором ключових точок. Він виводить 64 або 128 дескрипторів ключових точок дескрипторів. SURF також працює в чотири етапи, як і SIFT, але з деякими змінами. У підході до виявлення точки інтересу використовується найпростіше наближення матриці Гессе. Інтегральне зображення або таблиця сумарної площі було представлено в 1984 році. Інтегральне зображення

використовується як швидкий і ефективний спосіб обчислення суми значень піксельних значень у зображенні – або прямокутній підмножині сітки. Він також може або в основному використовується для обчислення середньої інтенсивності в заданому зображенні.

$$I_{\Sigma}(x) = \sum_{i=0}^{i\leq x} \sum_{j=0}^{j\leq y} I(i,j). \tag{1.1}$$

SURF використовує матрицю Гессе через її хорошу продуктивність у часі обчислення та точності. Замість того, щоб використовувати іншу міру для вибору місця та масштабу (детектор Гессена-Лапласа), пошук покладається на визначник матриці Гессе для обох.

Для адаптації до будь-якого масштабу відфільтровано зображення за ядром Гаусса, тож задано точку x, матриця Гесса у масштабі σ визначається як:

$$H(x,\sigma) = \begin{bmatrix} L_{xx}(x,\sigma) & L_{xy}(x,\sigma) \\ L_{xy}(x,\sigma) & L_{yy}(x,\sigma) \end{bmatrix}. \tag{1.2}$$

Щоб обчислити визначник матриці Гесса, необхідно почати зі згортки з ядром Гаусса, а потім застосувати похідну другого порядку. Спираючись на успішне наближення Лоу за допомогою SIFT, SURF просуває наближення далі, використовуючи бокс-фільтри. Гаусові похідні другого порядку можна апроксимувати і обчислювати з низькими обчислювальними витратами, використовуючи цілісні зображення, незалежно від розміру, це один з факторів, що сприяє швидкості SURF [8, 11–13]. Масштабні простори зазвичай реалізуються як піраміди зображень.

Зображення багаторазово згладжуються за Гаусом, а потім піддискретизуються, щоб досягти вищого рівня піраміди. Завдяки використанню бокс-фільтрів та інтегральних зображень, SURF не повинен ітеративно застосовувати той самий фільтр до виходу попередньо

відфільтрованого шару, а може застосовувати такі фільтри будь-якого розміру з однаковою швидкістю безпосередньо на вихідному зображенні, і навіть паралельно. Таким чином, масштабний простір аналізується шляхом збільшення розміру фільтра (9×9, 15×15, 21×21, 27×27, тощо), а не шляхом ітеративного зменшення розміру зображення. Таким чином, для кожної нової октави розмір фільтра збільшується вдвічі, одночасно вдвічі збільшуються інтервали дискретизації для виділення точок інтересу, що дозволяє масштабувати фільтр з постійною вартістю. Для локалізації точок інтересу на зображенні та в різних масштабах застосовується не максимальне пригнічення в околі 3×3×3. Замість ітераційного зменшення розміру зображення, використання цілісних зображень дозволяє масштабувати фільтр за постійної вартості, це зображено на рисунку 1.4.

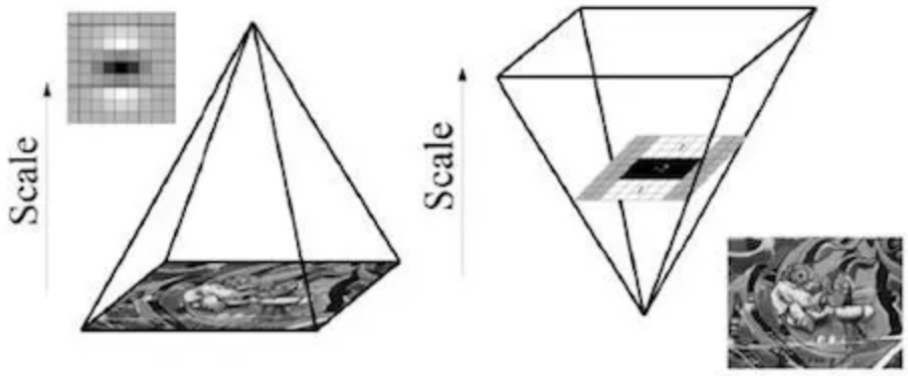


Рисунок 1.4 – Масштабування фільтра при постійній вартості на основі цілісних зображень

Створення дескриптора SURF відбувається у два кроки. Перший крок полягає у фіксації відтворюваної орієнтації на основі інформації з кругової області навколо ключової точки. Потім будується квадратна область, вирівняну за обраною орієнтацією, і витягується з неї дескриптор SURF. Для того, щоб бути інваріантним до обертання, SURF намагається визначити відтворювану орієнтацію для точок інтересу. Для цього алгоритм спочатку обчислює вейвлет відповіді Хаара в напрямках x та y, і це в круговій околиці радіусом $6s$ навколо ключової точки, де s – масштаб, в якому було виявлено

ключову точку. Крім того, крок дискретизації залежить від масштабу і вибирається рівним s, а вейвлет відповіді обчислюються на поточному масштабі s. Відповідно, на високих масштабах розмір вейвлетів є великим. Тому для швидкої фільтрації використовуються інтегральні зображення. При обчисленні суми вертикальних і горизонтальних вейвлет відповідей в області сканування, відбувається зміна орієнтацію сканування, а саме додавання $\pi/3$ і повторення обчислення, поки не знайдеться орієнтація з найбільшим значенням суми, ця орієнтація і є основною дескриптора ознаки (рис. 1.5).

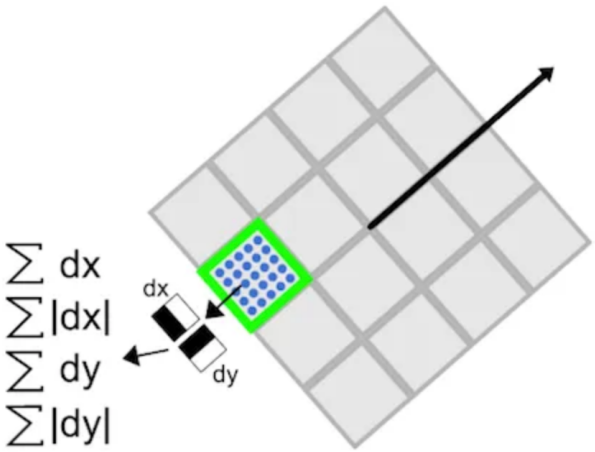


Рисунок 1.5 – Обчислення дескриптора особливої точки

Перший крок полягає у побудові квадратної області з центром y ключовій точці, орієнтованої за напрямом, який вже отримали вище. Розмір цього вікна становить $20s$. Потім область регулярно розбивається на менші квадратні підобласті розміром 4×4. Для кожного субрегіону обчислюється кілька простих функцій у точках вибірки з регулярним інтервалом 5×5.

Підвищення стійкості до геометричних деформацій і помилок локалізації, відгуки dx та dy спочатку зважуються за допомогою гаусівської функції з центром у ключовій точці. Потім вейвлет відповіді dx та dy підсумовуються по кожній підобласті і формують перший набір елементів вектора ознак. Для того, щоб додати інформацію про полярність змін інтенсивності, виділяється сума абсолютних значень відгуків, $|dx|$ та $|dy|$.

Таким чином, кожен субрегіон має чотиривимірний вектор-дескриптор v для своєї базової структури інтенсивності.

$$V = (\sum dx, \sum dy, \sum|dx|, \sum|dy|). \quad (1.3)$$

Це дає вектор дескриптора для всіх підобластей 4×4 довжиною 64 (у SIFT дескриптор – це вектор 128 D, тому SURF працює швидше, ніж SIFT).

Переваги SURF – він дає кращі результати в обертанні, розмитті та освітленні порівняно з SIFT, працює швидше, ніж SIFT. Довжина вектора ознак менша за довжину ознак, отриманих за допомогою SIFT. З недоліків не показує хороших результатів в області масштабування порівняно з SIFT.

Орієнтований швидкий та обернений BRIEF (ORB) був запропонований Етаном Рублі, Вінсентом Рабаудом, Куртом Коноліджем і Гері Брадскі в 2011 році у лабораторії OpenCV. ORB набагато швидший за SIFT та SURF. Він виконує вилучення ознак за допомогою детектора ключових точок FAST, детектора кутів Харріса та дескриптора ключових точок BRIEF [14]. Для заданого пікселя у масиві швидко порівнюється його яскравість з навколишніми 16 пікселями, які знаходяться у невеликому колі. Пікселі у колі потім сортуються на три класи: світліші, темніші або схожі. Якщо більше 8 пікселів темніші або світліші, то вони обираються як ключові точки. Таким чином, ключові точки, знайдені за допомогою алгоритму, дають інформацію про розташування визначальних ребер на зображенні. Однак алгоритму бракує компонента орієнтації та багатомасштабних функцій, тому алгоритм ORB використовує багатомасштабну піраміду зображень, яка представляє різні представлення одного і того ж зображення з різною роздільною здатністю. Кожен рівень піраміди містить зображення, зменшене порівняно з попереднім рівнем. Після того, як ORB сформував піраміду, він використовує алгоритм FAST для виявлення ключових точок на зображенні [15]. Виявляючи ключові точки на кожному рівні, ORB ефективно знаходить їх у різних масштабах,

роблячи піраміду частково інваріантною до масштабу. Після визначення ключових точок куля призначає кожній з них орієнтацію ліворуч або праворуч, залежно від того, як змінюються рівні інтенсивності навколо цієї ключової точки. Для визначення зміни інтенсивності куля використовує центроїд інтенсивності. Центроїд інтенсивності припускає, що інтенсивність кута зміщена від його центру, і цей вектор може бути використаний для визначення орієнтації.

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y). \tag{1.4}$$

Після обчислення орієнтації патча, можна повернути його на канонічний кут, а потім обчислити дескриптор, отримавши таким чином деяку інваріантність обертання.

BRIEF бере всі виявлені ключові точки з швидкого алгоритму і перетворює їх у бінарний вектор ознак для представлення об’єкта. Двійковий вектор ознак, також відомий як двійковий дескриптор ознак, складається виключно з 1 та 0. Кожна ключова точка описується вектором ознак, який є рядком довжиною 128-512 біт; BRIEF починається зі згладжування зображення за допомогою ядра Гаусса, щоб уникнути чутливості дескриптора до високочастотного шуму [16]. Вибирається випадкова пара пікселів з певної області навколо ключової точки. Ця область, відома як патч, являє собою квадрат з певною шириною і висотою, представлений пікселями. Перший піксель у парі випадковим чином витягується з гаусівського розподілу, зосередженого навколо ключової точки, зі стандартним відхиленням або розкидом сигми. Другий піксель у кожній випадковій парі вибирається з гаусівського розподілу з центром у першому пікселі, зі стандартним відхиленням sigma, помноженим на два. Якщо перший піксель яскравіший за другий, то відповідному біту присвоюється значення 1, інакше – 0.

Доки орієнтація ключової точки не змінюється в різних ракурсах, для обчислення її дескриптора буде використано правильний набір точок $S\theta$. ORB визначає алгоритм rBRIEF наступним чином:

Крок 1. Запустити кожен тест на всіх навчальних патчах.

Крок 2. Впорядкувати тести за їх віддаленістю від середнього значення 0,5 сформувавши вектор *T*.

Крок 3. Жадібний пошук. Помістити перший тест у вектор результатів *R* і видалити його з *T*. Натупний тест з *T* необхідно порівняти з усіма тестами в *R*. Якщо його абсолютна кореляція перевищує порогове значення, треба його відкинути, інакше додати до *R*, доки в *R* не залишиться 256 тестів.

Переваги ORB – витягує менше (тобто 32 дескриптори ознак), але більш значущі ознаки зображення, мультимедійні інструменти та програми, також потребує менших обчислювальних витрат порівняно з алгоритмами SIFT та SURF. Крім того, ORB є вільним від ліцензування порівняно з запатентованими алгоритмами SIFT і SURF. З недоліків – менш масштабно-інваріантний порівняно з SURF та не працює на розмитих та спотворених зображеннях.

1.3 Зіставлення особливих точок

Багато алгоритмів, як керованих, так і некерованих, використовують міри відстані. Ці міри, такі як евклідова відстань або косинусна подібність, часто можна знайти в таких алгоритмах, як *k*-nearest neighbors, UMAP, HDBSCAN тощо. Розуміння області мір відстані є більш важливим, наприклад, *k*-nearest neighbors – це метод, який часто використовується для навчання під наглядом. За замовчуванням він часто використовує евклідову відстань. Однак, що, якщо дані мають високу розмірність, то гаверсова відстань буде кращою альтернативою.

Евклідова відстань – це міра відстані, яку найкраще можна пояснити, як довжину відрізка, що з'єднує дві точки. Формула досить проста, оскільки відстань обчислюється за декартовими координатами точок за допомогою теореми Піфагора.

Хоча це загальна міра відстані, евклідова відстань не змінюється в масштабі, що означає, що обчислені відстані можуть спотворюватись залежно від одиниць об'єктів [17]. Як правило, перед використанням цієї міри відстані потрібно нормалізувати дані (рис. 1.6).

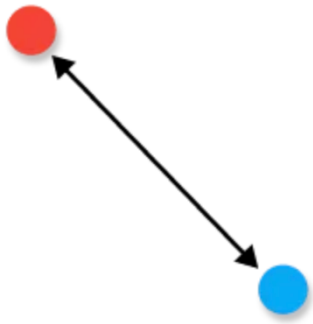


Рисунок 1.6 – Евклідова відстань

Крім того, зі збільшенням розмірності даних стає менш корисною евклідова відстань, бо це пов'язано з поняттям розмірності, яке пов'язане з уявленням про те, що вимірний простір не діє так, інтуїтивно очікуємо від двох чи трьох вимірного простору.

$$D(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad . \tag{1.5}$$

Евклідова відстань чудово працює, коли є дані низької розмірності, і важливо виміряти величину векторів. Такі методи, як *k*-nearest neighbors і HDBSCAN, показують чудові результати відразу, якщо евклідова відстань використовується для даних низької розмірності. Хоча було розроблено багато інших заходів для врахування недоліків евклідової відстані, вона все ще є однією з найбільш використовуваних мір відстані, бо є неймовірно інтуїтивно

зрозумілою у використанні, проста у застосуванні та показує чудові результати в багатьох випадках використання.

Косинусна подібність часто використовується як спосіб протистояти проблемі евклідової відстані з високою розмірністю. Косинусна подібність – це просто косинус кута між двома векторами [18–20]. Вона також має однаковий внутрішній добуток векторів, якщо вони були нормалізовані до довжини один. Два вектори з однаковою орієнтацією мають косинусну подібність одиницю, тоді як два вектори, діаметрально протилежні один одному, мають подібність мінус один (рис. 1.7).

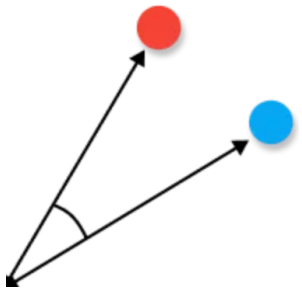


Рисунок 1.7 – Косинусна відстань

Одним з основних недоліків косинусної подібності є те, що не враховується величина векторів, а лише їх напрямок. На практиці це означає, що різниця у значеннях не враховується повністю. Якщо взяти, наприклад, систему рекомендацій, то косинусна подібність не враховує різницю в шкалі оцінок між різними користувачами.

$$D(x, y) = \cos(\theta) = \frac{xy}{\|x\| \ \|y\|} \quad . \tag{1.6}$$

Часто використовується косинусна подібність, коли є дані високої розмірності і коли величина векторів не має значення. Для аналізу тексту ця міра досить часто використовується, коли дані представлені у вигляді кількості слів. Наприклад, якщо слово зустрічається частіше в одному документі, ніж в іншому, це не обов’язково означає, що один документ більше

пов’язаний з цим словом. Може статися так, що документи мають неоднакову довжину і величина підрахунку має менше значення. Тоді найкраще використовувати косинусну схожість, яка не враховує величину.

Відстань Хеммінга – це кількісне вимірювання розбіжності між двома векторами шляхом підрахунку кількості різних значень, які вони містять. Зазвичай використовується для порівняння двійкових рядків еквівалентної довжини. Крім того, його можна застосовувати для порівняння схожості між рядками шляхом підрахунку символів, які відрізняються один від одного [21]. Однак, відстань Хеммінга стає складною для застосування, коли задача з двома векторами неоднакової довжини (рис. 1.8).

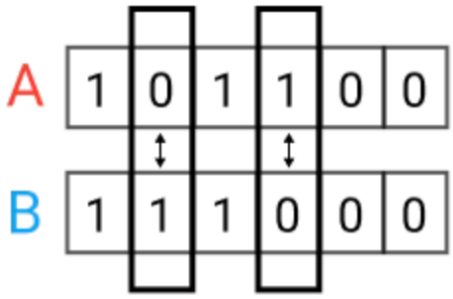


Рисунок 1.8 – Відстань Хеммінга

При порівнянні векторів однакової довжини важливо визначити позиції, в яких вони не збігаються. Однак, використовувана міра відстані не враховує фактичне значення, поки вони різні або рівні. Тому не рекомендується використовувати цю міру, коли величина є вирішальним фактором. Ця міра відстані зазвичай використовується для виправлення або виявлення помилок при передачі даних через комп’ютерні мережі. Відстань Хеммінга можна використовувати для оцінки помилок у двійкових словах, обчислюючи кількість спотворених бітів. Крім того, її можна використовувати для вимірювання категоріальної змінної відстані.

Манхеттена відстань, також відома як відстань таксі або відстань міських кварталів, обчислює відстань між векторами з дійсними значеннями. Цей метод розглядає вектори, які описують об’єкти на рівномірній сітці,

наприклад, на шаховій дошці. Манхеттена відстань відноситься до відстані між двома векторами, коли вони можуть рухатися тільки під прямим кутом. Діагональний рух не бере участі в обчисленні відстані [22]. Хоча відстань Манхеттена добре працює для даних високої розмірності, вона є менш інтуїтивно зрозумілою мірою порівняно з евклідовою відстанню, особливо для даних високої розмірності (рис. 1.9).

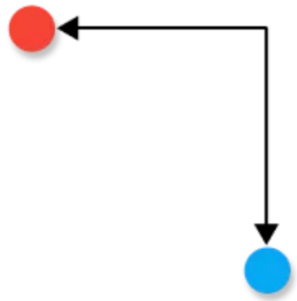


Рисунок 1.9 – Відстань Манхеттена

Крім того, вона часто дає більше значення відстані, ніж евклідова відстань, оскільки не враховує найкоротший шлях.

$$D(x, y) = \sum_{i=1}^{k} |x_i - y_i|. \quad (1.7)$$

При роботі з набором даних, що містить бінарні або дискретні атрибути, відстань Манхеттена, як правило, є більш придатною, оскільки вона враховує реалістичні шляхи, які можуть бути пройдені в межах цих атрибутів. На противагу цьому, евклідова відстань створить пряму лінію між двома векторами, що може бути нездійсненним у реальності.

Відстань Чебишева – це найбільша різниця між двома векторами в кожному координатному вимірі, інакше кажучи, максимальна відстань вздовж однієї осі. Її часто називають відстанню шахової дошки через її схожість з мінімальною кількістю ходів, необхідною для переміщення короля з однієї клітини на іншу.

Відстань Чебишева здебільшого застосовується до конкретних сценаріїв. Тому її не можна використовувати як загальну метрику відстані, таку як евклідова відстань або косинусна подібність [23]. Бажано застосовувати її лише тоді, коли вона однозначно підходить для вашої конкретної мети.

Відстань Чебишева можна використовувати для обчислення найменшої кількості ходів, необхідних для переходу з однієї клітинки в іншу (рис. 1.10).

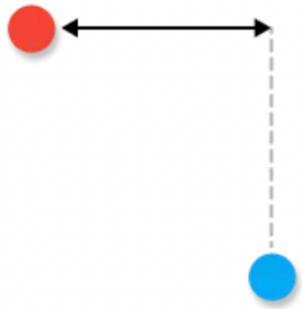


Рисунок 1.10 – Відстань Чебишева

У складській логістиці ця відстань широко використовується, оскільки вона точно імітує час, необхідний мостовому крану для транспортування об’єкта.

$$D(x, y) = \max_i(|x_i - y_i|). \quad (1.8)$$

Вимірювання відстані виявляється ключовим елементом в сфері машинного навчання та науки про дані, де аналіз та обробка інформації грають визначальну роль. Цей інструмент стає невід’ємною частиною завдань, пов’язаних з порівнянням та кластеризацією даних, сприяючи покращенню точності та ефективності аналізу. Вимірювання відстані в машинному навчанні дозволяє оцінити схожість між різними об’єктами або наборами даних, це стає важливим при вирішенні завдань класифікації, де необхідно визначити, до якого класу відноситься новий об’єкт на основі його схожості з вже відомими прикладами [24]. Метрики відстані, такі як евклідова, відстань

Манхеттена чи косинусна, використовуються для визначення відстаней та допомагають у розв’язанні таких завдань.

У сфері науки про дані вимірювання, відстані використовується для порівняння об’єктів або визначення ступеня схожості між різними наборами даних. Це допомагає виявляти закономірності, встановлювати зв’язки та виділяти ключові риси в даних. Наприклад, у задачах кластеризації вимірювання відстані використовується для групування подібних об’єктів в кластери, що дозволяє виявляти структуру та особливості набору даних.

Необхідність вимірювання відстані визначається також в контексті оптимізації алгоритмів машинного навчання та обробки даних. Використання відповідних метрик відстані може покращити якість моделей та результати аналізу, забезпечуючи більш точні та зрозумілі висновки.

1.4 Постановка задачі дослідження

Таким чином, дослідження методів автоматичного формування панорам з множин зображень має велике значення у сучасній області обробки зображень та комп’ютерного зору. Перш за все, це сприяє подоланню обмежень традиційних методів фотографії та відеозйомки, дозволяючи створювати широкі панорамні зображення з декількох джерел. Такі панорами надають можливість користувачам отримувати більше інформації та глибше відчувати простір оточуючого світу.

Об’єктом дослідження є процес автоматичного об’єднання зображень для отримання панорами.

Метою дослідження є роботи є аналіз існуючих методів та розробка нових підходів для підвищення якості та ефективності формування панорам.

Для досягнення мети необхідно вирішити такі завдання:

− провести аналіз існуючих методі та оцінити ефективність наявних підходів до автоматичного формування панорам;

- розробити новий метод, спрямований на вдосконалення процесу автоматичного зшивання;
- зробити експериментальне дослідження з використанням реальних або синтетичних даних для об’єктивної оцінки ефективності;
- реалізувати програмний застосунок для порівняння методів.

2 МАТЕМАТИЧНІ МОДЕЛІ НОРМАЛІЗАЦІЇ ЗОБРАЖЕНЬ НА ОСНОВІ ДЕСКРИПТОРІВ ЇХ ХАРАКТЕРНИХ ТОЧОК

2.1 Геометричні перетворення

Геометричні перетворення включають в себе низку основних операцій для зміни форми та положення двовимірних графічних об’єктів. До таких ключових процедур відносяться: переміщення початку координат, масштабування, обертання вісей координат. Ці перетворення грають важливу роль у візуалізації та моделюванні об’єктів в комп’ютерних програмах.

Комп’ютерний зір, у свою чергу, використовується для аналізу та обробки зображень на комп’ютері. Він дозволяє системам розпізнавання облич, відстеження руху та інші задачі, пов’язані з обробкою візуальної інформації. Комп’ютери використовують алгоритми обробки зображень для імітації способу, яким працює людський зір, що робить можливим взаємодію з об’єктами в цифровому середовищі. Поєднання геометричних перетворень та комп’ютерного зору важливо для створення реалістичних візуальних ефектів у графіці та відтворення тривимірних об’єктів на двовимірному екрані. Алгоритми геометричних перетворень визначають, як об’єкти змінюють своє положення та форму при взаємодії з користувачем або в результаті анімації. Будь-яке афінне перетворення можна виразити як комбінацію чотирьох базових перетворень, перелічених вище. У комп’ютерній графіці для запису цих перетворень використовується матриця. Комбінацію отримують шляхом перемноження матриць, що описують відповідні афінні перетворення [25].

Переміщення початку координат – ця операція дозволяє переміщувати початок координат, змінюючи положення об’єкта на площині без спотворення його форми.

$$[T] = \begin{bmatrix} 1 & 0 & \lambda \\ 0 & 1 & \mu \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.1)$$

Масштабування – процедура впливає на розмір об’єкта, збільшуючи або зменшуючи його на основі заданих параметрів масштабу.

$$[S] = \begin{bmatrix} \alpha & 0 & 0 \\ 0 & \delta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.2)$$

Операція обертання дозволяє змінювати орієнтацію об’єкта відносно осей координат, що є важливим для створення різноманітних ефектів та анімації.

$$[R] = \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} . \quad (2.3)$$

Компоненти матриці довільного афінного перетворення не мають чіткої геометричної інтерпретації. Щоб виконати конкретне перетворення, необхідно розкласти його на базові елементи, побудувати їхні матриці та перемножити їх у потрібній послідовності.

2.2 Алгоритм RANSAC

Абревіатура RANSAC (RANdom Simple Consensus) описує ітеративний метод, який оцінює параметри математичної моделі з набору даних, що містить викиди. Фішлер і Боллес вперше опублікували цей алгоритм у 1981 році [26]. Алгоритм RANSAC припускає, що всі дані складаються як з пропусків,

так і з викидів. Пропуски можна пояснити моделлю з певним набором значень параметрів, тоді як викиди ніколи не вписуються в цю модель. Іншим необхідним припущенням є те, що існує процедура, здатна оптимально оцінити параметри обраної моделі на основі наданих даних. Алгоритм RANSAC приймає на вхід набір значень спостережуваних даних, параметризовану модель, яка може відповідати спостереженням або пояснювати їх, і довірчі параметри. Гіпотетична модель інвайлерів будується шляхом підбору всіх вільних параметрів до інвайлерів. Всі інші дані оцінюються за цією моделлю. Додаткова точка вважається гіпотетичною, якщо вона добре вписується в оцінену модель. Оцінена модель вважається адекватною, коли достатня кількість точок була визначена як гіпотетичні вхідні дані. Модель повторно оцінюється з використанням усіх гіпотетичних вхідних даних, оскільки раніше вона оцінювалася лише з початковим набором гіпотетичних вхідних даних. Для оцінки моделі оцінюється похибка вхідних даних по відношенню до неї. Кожна ітерація призводить або до відкинутої моделі, з надто малою кількістю точок, класифікованих як вхідні дані, або до уточненої моделі з відповідною мірою похибки (рис. 2.1).

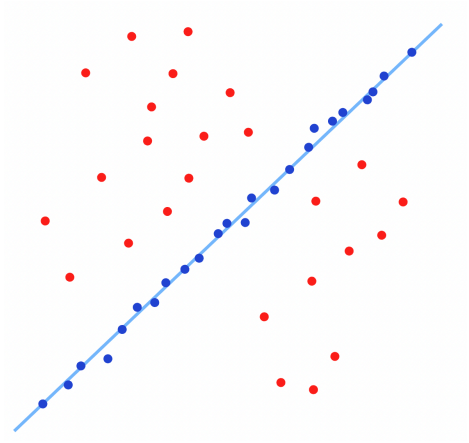


Рисунок 2.1 – Підігнана лінія викидів за допомогою RANSAC

Переваги RANSAC – це надійний алгоритм, який ефективно справляється зі значною кількістю шуму та викидів у даних. Він

універсальний, застосовується до будь-якої моделі, яку можна оцінити на основі підмножини даних [27–30]. Крім того, його відносно легко і ефективно реалізувати. RANSAC може точно визначити справжню модель, навіть якщо дані містять багато викидів (рис. 2.2).

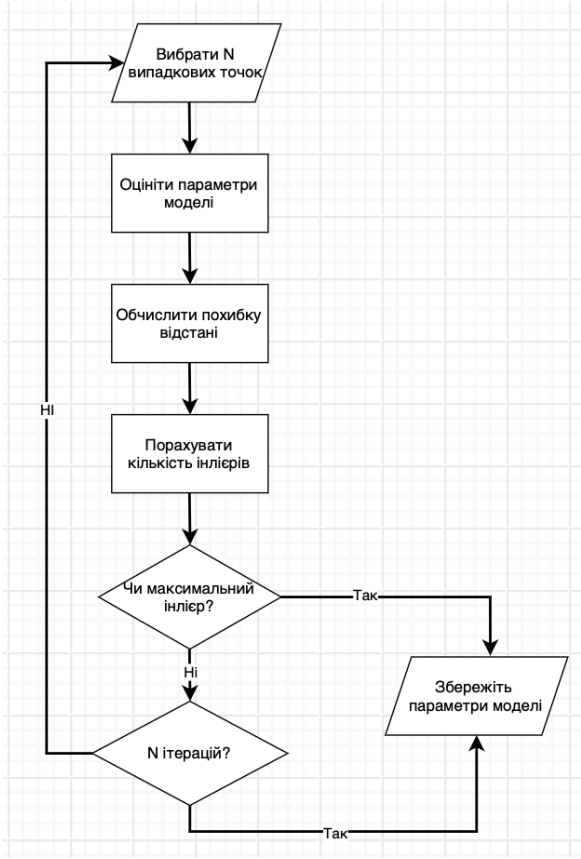


Рисунок 2.2 – Діаграма алгоритму RANSAC

З недоліків оскільки RANSAC є евристичним алгоритмом, він не може гарантувати оптимальне рішення. Вибір параметрів (n, k, t, d) може помітно вплинути на роботу алгоритму. Визначення найбільш підходящих значень цих параметрів може бути складним завданням. Початкова випадкова вибірка може спричинити чутливість алгоритму, що призводить до різних результатів для кожного прогону.

Алгоритм часто використовується в комп’ютерному зорі та робототехніці для таких завдань, як реєстрація зображень, розпізнавання

об’єктів і 3D-реконструкція. Він також використовується в машинному навчанні для таких завдань, як лінійна регресія і кластеризація. RANSAC особливо корисний у ситуаціях, коли дані містять велику кількість шуму або викидів, а інші алгоритми можуть не дати точних результатів.

Деякі конкретні випадки використання RANSAC включають припасування ліній та оцінка фундаментальної матриці. RANSAC можна використовувати для підгонки лінії до набору 2D або 3D точок, навіть якщо присутні викиди, що виявляється корисним у задачах комп’ютерного зору, таких як виявлення смуг руху в автономних транспортних засобах. Він також може оцінити фундаментальну матрицю, яка пов’язує відповідні точки на двох зображеннях. У стереозорі він може допомогти з 3D-реконструкцією та відстеженням об’єктів. Застосування RANSAC у розпізнаванні об’єктів має першорядне значення для зіставлення ознак з різних зображень і обчислення положення об’єкта [31]. У робототехніці RANSAC може відігравати ключову роль в операціях збирання та переміщення. Також, RANSAC може підігнати площину до набору 3D точок, незважаючи на наявність викидів. Це корисно в додатках комп’ютерної графіки, таких як рендеринг і 3D-моделювання.

Загалом, RANSAC є потужним алгоритмом для оцінювання робастних моделей за наявності викидів. Незважаючи на свої обмеження, він може слугувати цінним інструментом у різних додатках машинного навчання та комп’ютерного зору.

2.3 Програмне забезпечення для виділення об’єктів

Програмне забезпечення для вирішення задачі виділення об’єктів включає в себе два ключові компонента: бібліотеку OpenCV та фреймворк Tensorflow. OpenCV є потужним інструментом у галузі комп’ютерного зору та обробки зображень. Вона використовує каскадні методи визначення об’єктів, що дозволяє ефективно виділяти області інтересу та відокремлювати об’єкти

від фону [32]. Tensorflow, як відомий фреймворк для машинного навчання, відіграє визначальну роль у вирішенні задач виділення об’єктів, використання нейронних мереж типу R-CNN в Tensorflow забезпечує високу точність та надійність, цей підхід ефективно класифікує та локалізує об’єкти [33]. Об’єднання OpenCV та Tensorflow утворює потужний інструментарій для розв’язання завдань виділення об’єктів, сприяючи поєднанню переваг каскадних методів та сучасних підходів машинного навчання.

2.3.1 Бібліотека OpenCV

Спочатку розроблена компанією Intel, OpenCV (Open Source Computer Vision) – це безкоштовна крос-платформна бібліотека для обробки зображень у реальному часі. Її об’єктивність та універсальність застосування зробили OpenCV стандартом де-факто в галузі комп’ютерного зору. Станом на 2023 рік OpenCV залишається дуже затребуваною, з понад 29000 щотижневих завантажень.

Написана на мовах C та C++, і сумісна з найпопулярнішими операційними системами, включаючи GNU Linux, OS X, Windows, Android та iOS. Бібліотека OpenCV є вільно доступною під ліцензією Apache 2, з постійною розробкою для Python, Ruby, Matlab та інших мов. Вона містить понад 2500 алгоритмів, а також вичерпну документацію та приклади коду для комп’ютерного зору в реальному часі. З моменту дебюту в 2000 році під ліцензією BSD, а згодом під ліцензією Apache 2, OpenCV підтримує різноманітні програми, продукти та дослідницькі роботи. Ці програми охоплюють широкий спектр завдань, таких як зшивання зображень з камер для супутникових або веб-карт, вирівнювання сканів зображень, зменшення шуму в медичних зображеннях, аналіз об’єктів, забезпечення систем безпеки та виявлення вторгнень, проведення автоматичного моніторингу та процедур безпеки, проведення виробничого контролю ШІ, калібрування камер,

проведення оборонних та військових операцій, а також безпілотних повітряних, наземних та підводних апаратів.

OpenCV було розроблено для оптимальної ефективності та високої продуктивності в обробці складних завдань технічного зору. Таким чином, він надає пріоритет застосуванню штучного зору в реальному часі [34]. OpenCV написана оптимізованою мовою C, що дозволяє ефективно використовувати багатоядерні процесори. Основна мета OpenCV – запропонувати зручний фреймворк комп'ютерного зору, який дозволяє швидко створювати просунуті програми зору, надаючи понад 500 функцій, що охоплюють кілька областей зору. OpenCV часто використовується в різних сферах, включаючи контроль якості продукції, медичну візуалізацію, аналіз безпеки, людино-машинний інтерфейс, калібрування камер, стереозорість і роботизований зір. Широкі можливості бібліотеки обробки зображень полегшують обробку відеопотоків, зшивання зображень об'єднання декількох камер, калібрування камер і різноманітні завдання попередньої обробки зображень (рис. 2.3).

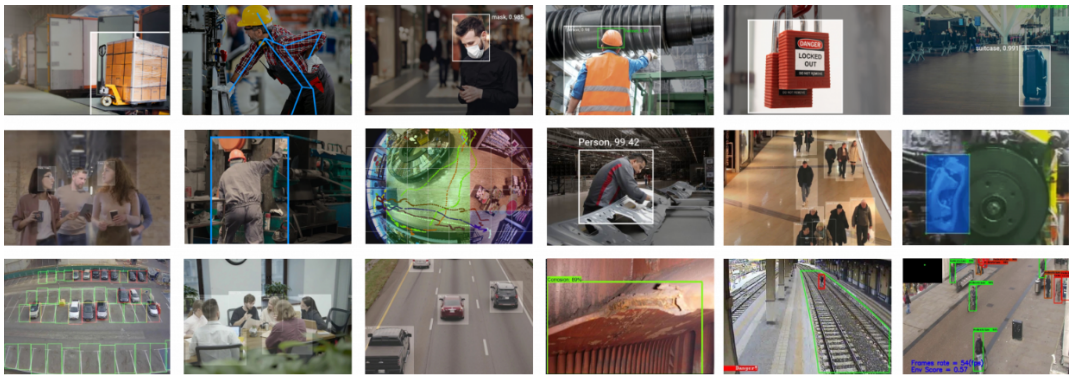


Рисунок 2.3 – Програми комп'ютерного зору, створені з використанням OpenCV

З 2011 року OpenCV включає комплексну багатоцільову бібліотеку ML, орієнтовану на статистичне розпізнавання образів і кластеризацію, що має вирішальне значення для комп'ютерного зору. Ця функція включає підтримку апаратного прискорення NVIDIA CUDA та графічних процесорів (GPU), а також Open Computing Language (OpenCL). Модуль OpenCV GPU дозволяє

явно керувати передачею даних між пам’яттю CPU і GPU. OpenCV використовується великими корпораціями та державними установами, серед яких Google, Toyota, IBM, Microsoft, SONY, Siemens та Facebook. Крім того, відомі стартапи, що спеціалізуються на комп’ютерному зорі, використовують OpenCV для розробки передових продуктів і рішень ШІ.

Численні дослідницькі установи, такі як Стенфорд, Массачусетський технологічний інститут, INRIA, Кембридж та КМУ, також використовують OpenCV. Потенційні можливості застосування комп’ютерного зору дуже широкі. Хоча багато хто знайомий з популярними кейсами використання в системах безпеки, відеоспостереження та безпілотних автомобілях, менше знають про застосування комп’ютерного зору в конкретних галузях, таких як промислове виробництво, ресторани та аналітика в роздрібній торгівлі. Завдяки швидкому розвитку комп’ютерного зору в останні роки, компанії з різних секторів можуть розробляти спеціалізовані програми комп’ютерного зору для вирішення конкретних завдань, таких як виявлення проблем з продуктом, підрахунок об’єктів і аналіз поведінки. OpenCV було створено в рамках дослідницької ініціативи Intel, спрямованої на розвиток ресурсоємних додатків [35–38].

Деякі провідні університетські групи, в тому числі MIT Media Lab, використовували добре розвинені, внутрішньо відкриті інфраструктури комп’ютерного зору для прискорення роботи над розробкою. Спільний код забезпечив значну перевагу в розробці користувацьких застосунків комп’ютерного зору без необхідності починати з нуля. Прискорення розвитку комерційного комп’ютерного зору за допомогою спільної інфраструктури завжди було головною метою команд розробників OpenCV в Intel. Надання портативного та оптимізованого за продуктивністю коду у вільний доступ сприяє розвитку комерційних додатків на основі технічного зору та збільшує попит на швидкі процесори, що є основним напрямком діяльності Intel. Перехід на швидші процесори приносить Intel більше прибутку, ніж продаж додаткового програмного забезпечення.

Причина, чому виробник обладнання, а не компанія-розробник програмного забезпечення, найімовірніше, створив вільний і відкритий код, пов’язана з його поточним станом. OpenCV отримує вигоду від численних внесків користувачів, і його центральна розробка більше не знаходиться в руках Intel. Додаткову підтримку надають Willow Garage, стартап у сфері комп’ютерного зору, та Itseez, який Intel придбала у 2016 році.

Останнім часом розробка без коду та з низьким рівнем коду з’явилася як інноваційний засіб для підприємств та організацій, що дозволяє прискорити та оптимізувати доставку та рішення. Розробка комп’ютерного зору передбачає складність, що вимагає ітеративних циклів проектування та розгортання. Отже, візуальна розробка та автоматизована інфраструктура розгортання технології без коду є надзвичайно корисними для оптимізації надання послуг комп’ютерного зору. Платформа комп’ютерного зору пропонує можливості OpenCV у вигляді модульних будівельних блоків, які пришвидшують процес створення додатків комп’ютерного зору, не вимагаючи написання коду з нуля. Це дозволяє командам швидше використовувати OpenCV, а також спрощує інтеграцію різних типів обладнання, таких як камери, периферійні комп’ютери та моделі машинного навчання. Технологія без коду долає розрив між досвідченими інженерами комп’ютерного зору та бізнес-командами, дозволяючи легко вносити зміни в рішення в міру розвитку технологій і зміни потреб бізнесу. Впровадження OpenCV без кодування усуває потребу в технічних знаннях і розкриває весь економічний потенціал комп’ютерного зору, зменшуючи ризики і знижуючи витрати.

Комп’ютерний зір – це ключова галузь технології штучного інтелекту, яка дозволяє комп’ютерам отримувати інформацію з цифрових зображень і відео для досягнення певних цілей. Визначення штучного інтелекту (ШІ) можуть відрізнятися, але основна ідея полягає у створенні машин, які імітують людський інтелект. Метою технологій машинного зору є відтворення людського зору шляхом інтерпретації цифрових зображень або відео з метою розуміння та ідентифікації об’єктів і сцен на них. Це досягається за допомогою

комбінації програмного та апаратного забезпечення, що імітує функції людської зорової системи. Розпізнавання образів, виділення ознак і обробка зображень – ось деякі з поширених методів, що застосовуються в машинному зорі. Нейронні мережі виявляють ті аспекти, які можуть ускладнити для людини процес сприйняття. Вони дозволяють виявляти та аналізувати нюанси, які легко можуть утекти з людської уваги. Застосування нейронних мереж включає в себе різноманітні сфери, такі як розпізнавання емоцій, де система може точно визначати вирази обличчя та пов'язані з ними емоційні стани зображено на рисунку 2.4.

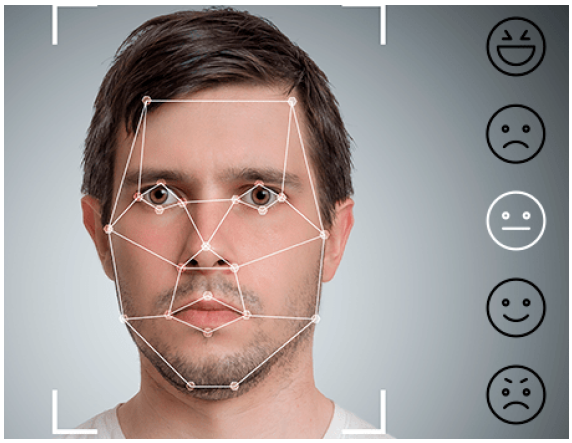


Рисунок 2.4 – Визначення виразу обличчя

Можливість ефективно цифрувати та автоматизувати операції, змушуючи комп'ютери бачити за допомогою штучного інтелекту та сприймати фізичний світ за допомогою зорових сенсорів, стала невід'ємною технологією. В останні роки технології машинного навчання, особливо глибокого навчання, успішно продемонстрували застосування комп'ютерного зору в різних галузях. Використовуючи технологію штучного зору, комп'ютери можуть розпізнавати обличчя, читати рукописний текст, розпізнавати об'єкти, класифікувати рухи людини, проводити автоматизований огляд або автоматично виявляти критичні ситуації за допомогою розпізнавання зображень [39].

Комп’ютерний зір є складним завданням, і отримання високоякісних результатів за розумну ціну має вирішальне значення для масштабованого комп’ютерного зору. При використанні камер дані зображення часто піддаються спотворенням або шуму, що виникають через фізичні фактори, такі як освітлення, віддзеркалення або рух, а також через недосконаліть об’єктива (спотворення, поле зору), сенсора або механічних налаштувань (кут, положення, висота). Щоб подолати виклики, які виникають у комп’ютерному зорі, розробники повинні побудувати складні конвеєри, які моделюють потік даних. Цей процес поєднує в собі різні завдання, починаючи від збору кадрів і закінчуючи попередньою обробкою (знебарвленням, фільтрацією, згладжуванням, тощо) і пропусканням їх через один або кілька алгоритмів машинного зору.

Сегментація зображень передбачає використання алгоритмів обробки зображень для поділу зображення на кілька окремих сегментів. Зазвичай сегментація використовується для спрощення, модифікації або покращення зображення, часто як частина завдань комп’ютерного зору. Використовуючи відповідні ключові точки, можна застосувати трансформацію перспективи та отримати остаточну панораму (рис. 2.5).

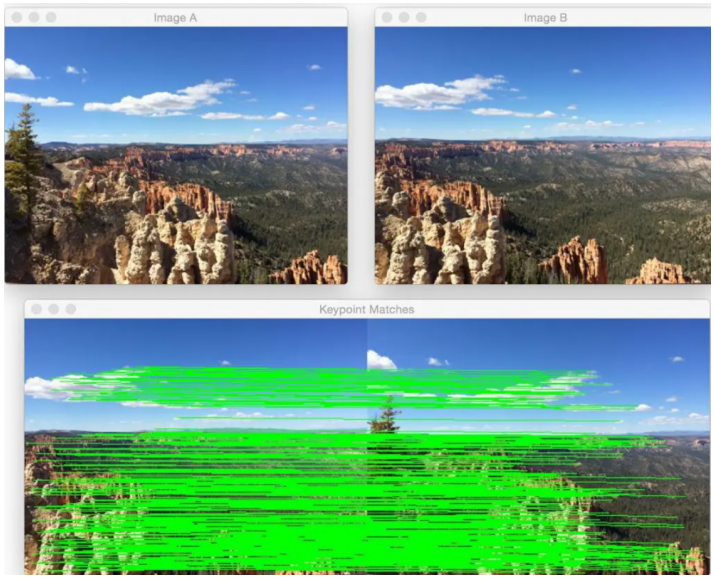


Рисунок 2.5 – Відповідність ключових точок між двома зображеннями

Практичне застосування є в автономному керуванні автомобілем, де сегментація зображень дає змогу виявляти та ідентифікувати дорогу. Розпізнавання поз і жестів людини за допомогою аналізу відео використовується для інтерпретації і розуміння фізичних жестів людини. Рухи тіла, рук та обличчя можуть бути ідентифіковані та класифіковані за попередньо визначеними категоріями. Аналіз рухів часто є складовою оцінки пози, коли рухи тіла досліджуються за допомогою еталонних ключових точок, таких як суглоби та кінцівки. Обчислення пози об’єкта дає змогу зрозуміти позиціювання об’єкта в тривимірному просторі, наприклад, його обертання. Автоматичне розпізнавання обличчя використовується для ідентифікації осіб шляхом виявлення рис обличчя та зіставлення їх з базою даних. FaceRecognizer від OpenCV пропонує ряд відомих алгоритмів, які зазвичай використовуються в практичних програмах розпізнавання облич.

Доповнена реальність (AR) дозволяє взаємодіяти в реальному часі між реальним і віртуальним світом. Концепція доповненої реальності спрямована на покращення фізичного оточення шляхом включення комп’ютерної перцептивної інформації.

У різних галузях промисловості комп’ютерний зір пропонує конкурентну перевагу, дозволяючи спростити складні і дорогі процедури за допомогою індивідуальних, інтегрованих і високопродуктивних додатків комп’ютерного зору.

2.3.2 Зшивання зображень в панораму

Для побудови панорами зображень будуть використані методи комп’ютерного зору та обробки зображень, включаючи виявлення ключових точок і локальних інваріантних дескрипторів, зіставлення ключових точок, RANSAC і викривлення перспективи. Оскільки існують значні відмінності у тому, як виявляються ключові точки та локальні інваріантні дескриптори

(наприклад, SIFT та SURF) у OpenCV 2.4.X та OpenCV 3.X, створено код, який працює з обома версіями, за умови, що OpenCV 3 було скомпільовано з підтримкою опції opencv_contrib.

Алгоритм зшивання панорам складається з чотирьох окремих кроків:

Крок 1. Визначення ключових точок за допомогою таких методів, як DoG або Harris та вилучити локальні інваріантні дескриптори з вхідних зображень.

Крок 2. Зіставлення дескрипторів між зображеннями.

Крок 3. Використування алгоритму RANSAC для оцінки матриці гомографії, використовуючи вектори ознак.

Крок 4. Застосування деформуючого перетворення, використовуючи матрицю гомографії, отриману на Кроці 3.

Процес зшивання потребує лише одного вхідного параметра, який називається *images* і вказує на пару зображень, які будуть об’єднані для створення панорамного зображення [40]. Також можна ввести додаткові параметри, зокрема *ratio*, що застосовується у тесті Девіда Лоу на співвідношення для зіставлення об’єктів, *reprojThresh* який позначає найбільше відхилення пікселів, дозволене алгоритмом RANSAC, і *showMatches*, бінарну змінну, яка вказує, чи потрібно показувати підсвічені ключові точки збігів, чи ні.

Код включає клас Stitcher, який призначений для генерації панорамних зображень з пари вхідних зображень. Процедура складається з декількох етапів, включаючи виявлення ключових точок, вилучення дескрипторів, зіставлення ключових точок, застосування тесту на коефіцієнт Лоу і обчислення матриці гомографії за допомогою алгоритму RANSAC. Код починається з розпакування списку з двох зображень у порядку зліва направо. Метод detectAndDescribe визначає ключові точки та витягує локальні інваріантні дескриптори, використовуючи алгоритм SIFT. Функція matchKeypoints зіставляє ознаки між зображеннями, використовуючи

вбудовану функцію з OpenCV для зіставлення дескрипторів. Збіги проходять тест коефіцієнта Лоу для усунення помилкових спрацьовувань.

Якщо знайдено достатню кількість збігів, обчислюється матриця гомографії. На основі цієї матриці генеруються зшиті зображення. Якщо визначено ключові точки збігів, їх можна візуалізувати. В іншому випадку повертається зшите зображення. Код містить допоміжні методи detectAndDescribe для виявлення та опису ключових точок, а також drawMatches для візуалізації відповідності ключових точок між двома зображеннями [41].

Зрештою, цей код забезпечує надійний алгоритм зшивання, який створює панорамні зображення шляхом виявлення ключових точок, зіставлення об’єктів та обчислення гомографій. Функціональність вмикається скриптом драйвера за допомогою аргументів командного рядка, які визначають шляхи вхідних зображень.

2.4 Бікубічна інтерполяція

Дослідження методів автоматичного формування панорам з множин зображень є актуальним напрямком в сучасній обробці зображень. У цьому контексті великою увагою користується проблема масштабування та повороту зображень, оскільки правильний вибір методу інтерполяції грає ключову роль у забезпеченні якісних результатів. Обрана стратегія з використуванням бікубічної інтерполяції для масштабування та повороту, бо цей метод визначається високою точністю та здатністю зберігати якість зображення навіть під час значних трансформацій [42]. Бікубічна інтерполяція сприяє плавному згладжуванню пікселів та зменшенню артефактів, що можуть виникнути при інших методах інтерполяції.

Бікубічна інтерполяція – це метод підвищення чіткості та збільшення цифрових зображень за допомогою кубічних сплайнів або інших

поліноміальних підходів. Вона часто використовується в комп'ютерних програмах для редагування зображень ретушерами та редакторами при збільшенні масштабу або передискретизації зображення. Інтерполяція передбачає спотворення пікселів від однієї сітки до іншої. Два найпоширеніші алгоритми інтерполяції – адаптивний і неадаптивний. Адаптивні методи залежать від типу інтерполяції, тоді як неадаптивні методи обробляють усі пікселі однаково. Пропрієтарні методи у спеціалізованому професійному програмному забезпеченні для редагування зображень, такому як Photozoom Pro та Adobe Photoshop, зазвичай використовують адаптивні алгоритми. До неадаптивних алгоритмів належать, зокрема, алгоритми найближчого сусіда, білінійні, бікубічні та сплайнові. Бікубічна інтерполяція зазвичай досягається за допомогою поліномів Лагранжа, кубічних сплайнів або алгоритмів кубічної згортки. Під час інтерполяції оцінюються невідомі дані на основі відомих даних. Наприклад, якщо зображення з роздільною здатністю 16 МП було знято цифровою дзеркальною камерою, відомі дані включають 4928×3264 пікселів або 3264×4928 пікселів залежно від орієнтації, горизонтальної чи вертикальної. Щоб збільшити розмір зображення до 24 Мп з роздільною здатністю 6000×4000 пікселів, апроксимуємо нові значення пікселів на основі навколишніх пікселів, це призводить до появи додаткових пікселів, яких раніше не було, а саме додаткові 8 мегапікселів.

При додаванні пікселів замість віднімання існує ризик втрати деталей. Щоб зберегти різкість і цілісність зображення, кожен піксель повинен бути апроксимований з навколишніми пікселями, щоб отримати максимально близьке значення [43]. Це схоже на дублювання пікселя, щоб заповнити простір, створений при збільшенні масштабу зображення.

$$p(x,y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j \ . \qquad (2.4)$$

Таким чином, всі значення мають бути приблизно такими ж, як і їхні найближчі сусіди. Потрібно додати вісім мільйонів додаткових пікселів до

зображення, щоб заповнити простір, який утворився під час масштабування. Ці пікселі повинні мати достатню точність для відтворення деталей і збереження різкості, що забезпечить чіткість зображення. Цей метод передбачає отримання значень точки $p(x, y)$ на сітці та використання інтерполяції для оцінки значення сусідньої точки. Хоча в процесі є додаткові кроки, це фундаментальна формула. Щоб створити інтерпольовану поверхню для зображення, потрібно визначити 16 коефіцієнтів $p(x, y)$.

Якщо кодування не є можливим, можна також використовувати програмне забезпечення з вбудованими функціями для редагування зображень. При повторній дискретизації зображення у Photoshop доступні методи бікубічної інтерполяції. Збільшення кількості пікселів у зображенні можливе, але є певні обмеження. Бікубічна інтерполяція ефективна для високих роздільних здатностей, але при перевищенні 24 мегапікселів у надвисокій роздільній здатності вона може давати неоптимальні результати. Бікубічна, біквадратна та білінійна інтерполяція – це методи згладжування та відтворення зображень, вони застосовуються для поліпшення якості збільшення чи зменшення зображень, заповнення пропусків пікселів (рис. 2.6).

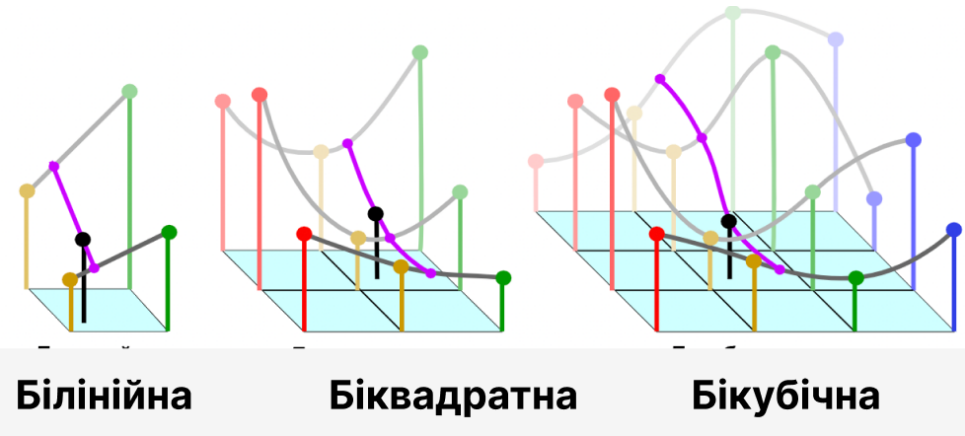


Рисунок 2.6 – Відповідність ключових точок між двома зображеннями

Причиною цього є те, що додавання пікселів вимагає збереження більшої кількості деталей, що алгоритм не завжди може точно виконати порівняно з вихідною роздільною здатністю зображення. Крім того, в

редагуванні зображень часто використовуються зображення у форматі JPEG, які часто стискаються. Якщо високий рівень стиснення для економії місця, можливо, проблема з відображенняи цінних деталей. Бажано вибрати найнижчий рівень стиснення, щоб зберегти більше деталей зображення при збільшенні масштабу. З іншого боку, зменшуючи кількість пікселів, є більше деталей і різкості для роботи, ніж збільшуючи кількість пікселів. Це пов’язано з тим, що відтворити деталі складніше, ніж видалити їх. При зменшенні масштабу якість вихідного зображення залишається високою, оскільки не додаються штучні деталі для заповнення прогалин.

Бікубічна інтерполяція використовується не лише для масштабування зображень, але й для відображення відео. Однак, хоча вона перевершує інші 2D-методи, вона має деякі недоліки, такі як перемасштабування (ореоли), відсікання, артефакти дзвону та надмірна різкість, які не завжди є візуально приємними [44]. Надмірне збільшення масштабу за допомогою бікубічної інтерполяції може призвести до незадовільних результатів. Однак сучасні методи надвисокої роздільної здатності, такі як SRCNN (згорткові нейронні мережі надвисокої роздільної здатності) або SRGAN (генеративні змагальні мережі надвисокої роздільної здатності) є більш ефективними у збереженні різкості та деталізації. При використанні цього методу в контексті автоматичного формування панорам важливо враховувати, що зображення можуть мати різні освітлення та кольорові властивості. Бікубічна інтерполяція, зберігаючи деталі та колір, допомагає уникнути втрати інформації під час обробки та забезпечує гармонійність у панорамному зображенні.

Враховуючи те, що формування панорами зазвичай включає в себе обробку великої кількості зображень, використання такого ефективного методу, як бікубічна інтерполяція, стає ключовим аспектом забезпечення ефективності та якості результата. Ця стратегія може виявитися особливо корисною при автоматизованих процесах обробки та аналізу зображень у великих обсягах даних.

3 РОЗРОБКА І РЕАЛІЗАЦІЯ АЛГОРИТМУ СТВОРЕННЯ ПАНОРАМНИХ ЗОБРАЖЕНЬ

3.1 Алгоритм створення панорамних зображень

Загальна блок-схема алгоритму включає наступні основні етапи: завантаження зображень, пошук особливих точок та отримання їх дескрипторів для поточної пари завантажених зображень за допомогою одного з методів, таких як SIRF чи SURF. Структура алгоритму представлена на рисунку 3.1.

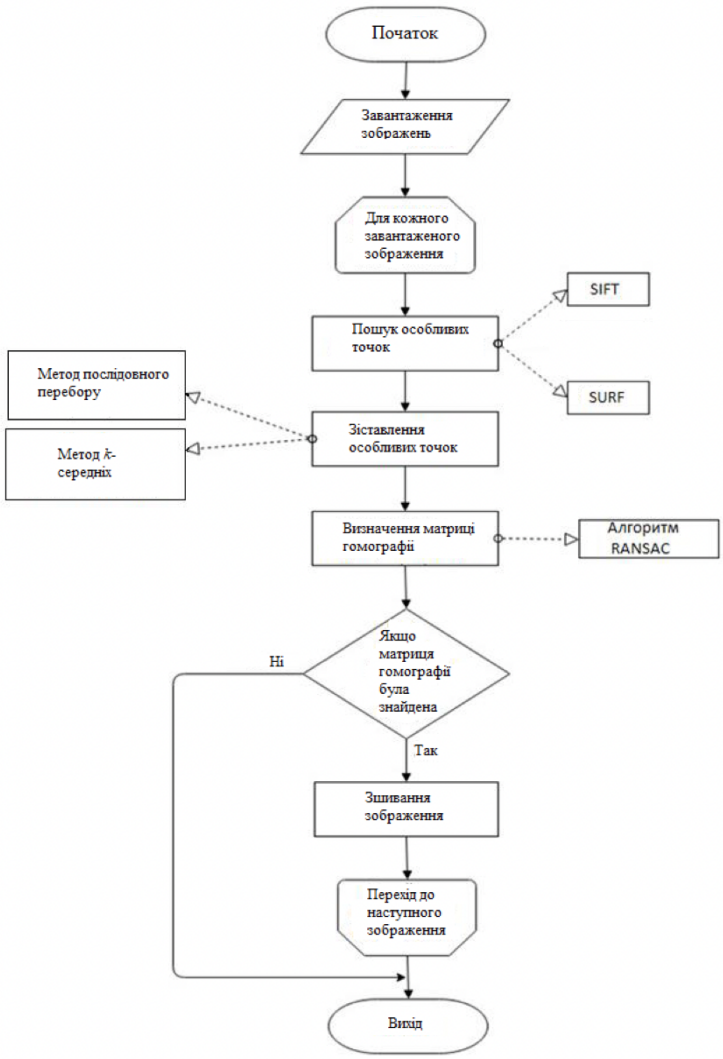


Рисунок 3.1 – Блок-схема

Зіставлення ключових точок є важливим кроком у виявленні схожих зображень або об’єктів, присутніх на одному зображенні при порівнянні з іншим. Для цього було досліджено методи послідовного пошуку та *k*-середніх.

Перший метод передбачає порівняння кожного дескриптора з першого набору з дескрипторами з другого набору, знаходження відстані між ними та визначення найближчого сусіда. Цей метод організовує різні комбінації для прискорення роботи з великими масивами даних.

Інший підхід передбачає побудову ієрархічного дерева *k*-середніх, який є найбільш ефективним для обробки великих обсягів даних. Отримавши матрицю гомографії, обчислюємо нові координати ключових точок другого знімка, об’єднуємо знімки і отримуємо панорамне зображення. Переходимо до наступного знімка. Якщо матриця гомографії не може бути отримана, створення панорами неможливе, і система завершує роботу.

3.2 Результати експериментів

Зіставлення зображень є основою багатьох комп’ютерних програм та програм машинного зору зокрема: Навігація роботів, оцінка положення, візуальна одометрія, візуальна одночасна локалізація і картографування, виявлення об’єктів, відстеження об’єктів, доповнена реальність, мозаїка зображень і зшивання панорам. Візуальна одометрія – це вимірювання відстані, пройденої мобільною системою на основі візуальної інформації. Одночасна локалізація та картографування на основі зору – це локалізація мобільного агента в середовищі з одночасним картографуванням його оточення. Оцінка положення виконується шляхом перетворення перспективи для зображення, отриманого в кінцевому положенні, по відношенню до зображення, отриманого в початковому положенні (рис. 3.2).

Рисунок 3.2 – Типові етапи оформлення зображення

Виявлення та відстеження об’єктів базується на виявленні та зіставленні спільних ознак між зображенням об’єкта та сценою. Зшивання або мозаїка зображень – це процес створення великомасштабного консолідованого зображення з декількох дрібномасштабних зображень, що перекриваються. Він широко застосовується в аеро, морській та супутниковій зйомці для отримання безперервного єдиного зображення предметного середовища.

Програмне забезпечення для створення панорам у мобільних телефонах і цифрових камерах є ще одним важливим застосуванням мозаїки. Всі ці програми вимагають наявності певної області перекриття (зазвичай понад 30%) між будь-якими двома послідовними зображеннями для зіставлення. Детектор ознак – це алгоритм, який виявляє на зображенні ознаки (також звані точками інтересу або ключовими точками). Об’єкти, як правило, виявляються у формі кутів, крапель, країв, з’єднань, ліній тощо. Виявлені об’єкти згодом описуються логічно різними способами на основі унікальних візерунків, які мають їхні сусідні пікселі. Цей процес називається описом ознак, оскільки він

описує кожну ознаку, присвоюючи їй відмітну ідентичність, що дозволяє їх ефективне розпізнавання для відповідності.

Деякі детектори ознак легко доступні з визначеним алгоритмом опису ознак, тоді як інші існують окремо. Однак окремі детектори ознак можуть поєднуватися з декількома типами відповідних дескрипторів ознак. SIFT, SURF, ORB і є одними з основних детекторів ознак масштабу, обертання та афінних інваріантів, кожен з яких має визначений дескриптор ознак і має власні переваги та недоліки (рис. 3.3).

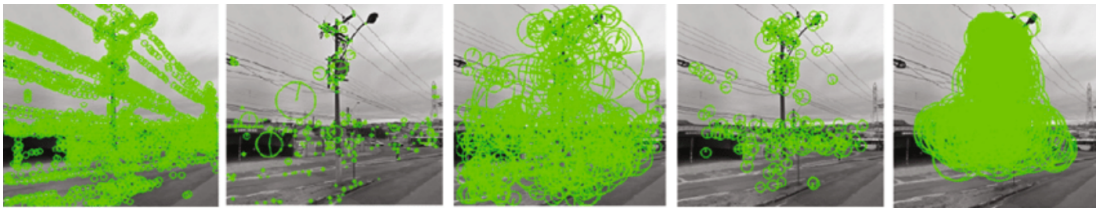


Рисунок 3.3 – Методи виділення ознак FAST, SIFT, SURF, BRISK, ORB

Після виявлення-опису ознак виконується співставлення ознак за допомогою L1-норми або L2-норми для рядкових дескрипторів (SIFT, SURF) та відстані Хеммінга для бінарних дескрипторів (ORB, BRISK).

Для зіставлення ознак можуть бути прийняті різні стратегії зіставлення, такі як зіставлення на основі порогових значень; зіставлення за найближчим сусідом; зіставлення за співвідношенням відстаней до найближчого сусіда. Неправильних збігів (або викидів) неможливо повністю уникнути на етапі зіставлення ознак, тому для точної підгонки моделі перетворення обов’язковим є наступний етап – відхилення викидів. Випадковий консенсус вибірки RANSAC, консенсус вибірки з MSAC та прогресивний консенсус вибірки PROSAC – це деякі з надійних імовірнісних методів, що використовуються для видалення викидів із зіставлених ознак та підбору функції перетворення в матриці гомографії. Ця матриця забезпечує перспективне перетворення другого зображення відносно першого. Потім на основі отриманої функції перетворення виконується реконструкція зображення, щоб вирівняти друге зображення відносно першого.

Реконструйована версія другого зображення накладається на еталонне зображення доти, доки не будуть перекриті всі відповідні точки. Ця велика консолідована версія менших зображень називається мозаїчним або зшитим зображенням.

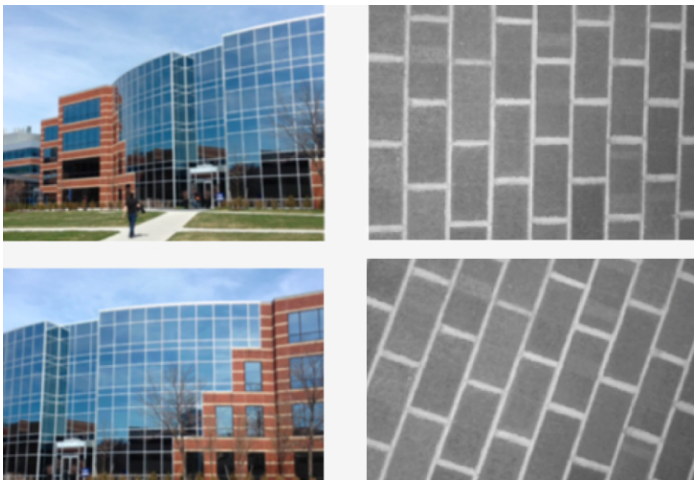
Алгоритми SIFT (краплі), SURF (краплі) [14], ORB (кути) і BRISK (кути) порівнюють для відповідності та реєстрації зображень. Продуктивність дескрипторів детектора ознак додатково оцінюється, щоб дослідити: який є більш інваріантним до масштабу, обертання та афінних змін. Щоб перевірити цю проблему, було виконано зіставлення зображень за допомогою цих дескрипторів детекторів функцій, щоб зіставити синтетично масштабовані версії від 5% до 500% і синтетично повернуті версії від 0° до 360° різних зображень з їхніми оригінальними версіями. Щоб дослідити точку зору або афінну інваріантність, було виконано зіставлення зображень для послідовності графіті та стіни.

Експерименти проводилися на різноманітних зображеннях, взятих із відомих наборів даних Оксфордського університету, MATLAB, VLFeat і OpenCV. Коефіцієнт відстані найближчого сусіда було застосовано як стратегію зіставлення ознак з алгоритмом грубого пошуку, тоді як RANSAC застосовано для підгонки моделей перетворення зображення (у формі гомографічних матриць) і для відхилення викидів. Для проведення експериментів, використовувався MATLAB-2017a з OpenCV 3.3. Технічні характеристики використовуваної комп’ютерної системи: процесор Intel(R) Core(TM) i7 @ 3,40 ГГц, та 16,00 ГБ оперативної пам’яті. SURF(64D), SURF(128D), ORB(1000) і BRISK(1000) представляють SURF з дескриптором 64-Floats, розширений SURF з дескриптором 128-Floats, обмежені детектори ORB і BRISK з верхньою межею для виявлення лише до найкращі 1000 балів, відповідно. Об’єкти OpenCV, які використовуються для дескрипторів детекторів функцій, усі інші параметри використовуються OpenCV за замовчуванням показано у таблиці 3.1.

Таблиця 3.1 – Налаштування OpenCV та розміри дескрипторів

Алгоритм	**OpenCV**	**Розмір дескриптора**
SIFT	cv.SIFT(‘ConstrastThreshold’,0.04,’Sigma’, 1.6)	128 Bytes
SIFT (128D)	cv.SURF(‘Extended’,true,’HessianThreshold’,100)	128 Floats
SURF(64D)	cv.SURF(‘HessianThreshold’,100)	64 Floats
ORB	cv.ORB(‘MaxFeatures’,100000)	32 Bytes
ORB (1000)	cv.ORB(‘MaxFeatures’,1000)	32 Bytes

Для цього дослідження було використано два набори даних. Набір даних A підготовлено шляхом вибору 6 пар зображень різноманітних сцен з різних еталонних наборів даних. Зображення Building показані на рисунку 3.4 а). Зображення Bricks, вибрані з інструментарію технічного зору MATLAB, показані на рисунку 3.4 б).



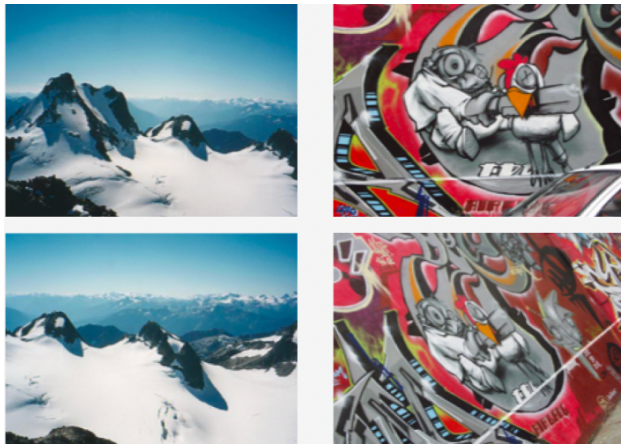
а) б)

Рисунок 3.4 – Зображення вибрані з MATLAB:
а) Building; б) Bricks

Експериментальні результати представлені у формі кількісного порівняння, часу виявлення ознаки-опису, часу зіставлення ознак, відхилення викидів і часу підгонки моделі, повторюваності та похибки у відновлених

результатах порівняно з фактичними значеннями. Зображення гори, взято з тестових даних OpenCV показана на рисунку 3.5 а).

Графіті, взято з бази даних Affine Covariant Regions Datasets Університету Оксфорда показана на рисунку 3.5 б).



а) б)

Рисунок 3.5 – Зображення з OpenCV та Affine Covariant Regions Datasets:
а) Mountains; б) Graffiti

Зображення Roofs, взято з даних бібліотеки VLFeat показана на рисунку 3.6 а). Зображення River, взято так само з даних бібліотеки VLFeat показана на рисунку 3.6 б).

а) б)

Рисунок 3.6 – Зображення з бібліотеки VLFeat:
а) Roofs; б) River

Набір даних B базується на 5 зображеннях, вибраних з набору даних афінно-коваріантних регіонів Університету Оксфорда, University of OXFORD’s Affine Covariant Regions Datasets (рис. 3.7).

Рисунок 3.7 – Набір даних B

Набір даних A, який складається з 6 пар зображень різноманітних сцен було використано для порівняння різних аспектів дескрипторів ознак для процесу реєстрації зображень, тоді як набір даних B було використано для дослідження здатності дескрипторів ознак бути інваріантними до масштабу та обертання. Крім того, для дослідження афінної інваріантності було використано послідовність графіті та послідовність стін з наборів даних афінно-коваріантних областей. Значення базової істини для перетворень зображень були використані для розрахунку та демонстрації похибки у відновлених результатах з кожною ознакою-детектором-дескриптором. Для оцінки інваріантності масштабу та повороту для кожного зображення в наборі даних в синтезовано базові істини шляхом зміни розміру та повороту зображення до відомих значень масштабу від 5% до 500% та повороту від 0° до 360°. Для масштабування та повороту зображень було обрано бікубічну інтерполяцію, оскільки це найточніший метод інтерполяції, який зберігає якість зображення в процесі перетворення. Для дослідження афінної

інваріантності було використано еталонні базові істини, надані в наборах даних Affine Covariant Regions Datasets для послідовностей графіті та стін. Виявлення ознак, зіставлення та накладання за допомогою SIFT, SURF, ORB показано на рисунку 3.8.

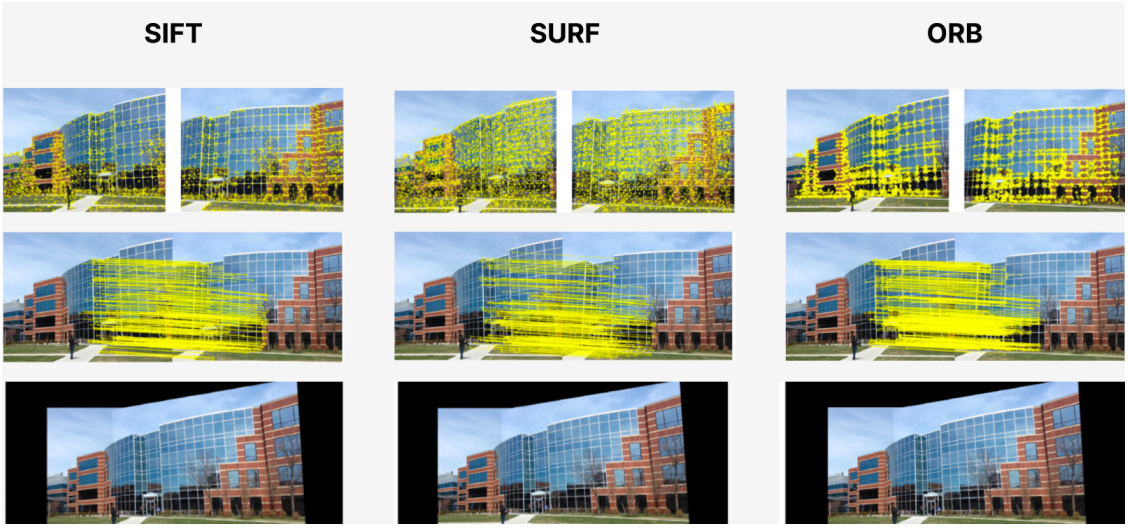


Рисунок 3.8 – Застосування методів накладання SIFT, SURF та ORB

Стратегія зіставлення ознак, прийнята для експериментів, базується на коефіцієнті відстані до найближчого сусіда, який використовувався Д. Г. Лоу для зіставлення ознак SIFT. У цій схемі зіставлення здійснюється пошук найближчого сусіда та другого найближчого сусіда для кожної ознаки-дескриптора з першого набору ознак, другому наборі ознак. Потім для кожної ознаки-дескриптора обчислюється відношення найближчого сусіда до другого найближчого сусіда і встановлюється певне порогове відношення для фільтрації бажаних збігів. Для експериментальних результатів, представлених у нижче, це порогове відношення дорівнює 0,7. L1-норму (також звану найменшими абсолютними відхиленнями) було використано для зіставлення дескрипторів SIFT, SURF, тоді як відстань Хеммінга було використано для зіставлення дескрипторів, ORB і BRISK.

У таблицях наведено результати кількісного порівняння та обчислювальні витрати на дескриптори ознак-детекторів для зіставлення зображень (рис. 3.9).

	Unnamed: 0	Features Detected in the	Unnamed: 2	Features Matched	Outliers Rejected	Feature Detection &	Unnamed: 6	Feature Matching Time (s)	Outlier Rejection & Homography Calculation Time (s)	Total Image Matching Time (s)
0	Algorithm	Image Pairs				Description Time (s)				
1		1st Image	2nd Image			1st Image	2nd Image			
2	Building Dataset									
3	SIFT	1907	2209	384.0	51.0	0.1812	0.198	0.1337	0.0057	0.5186
4	SURF(128D)	3988	4570	319.0	58.0	0.1657	0.1786	0.5439	0.0058	0.894
5	SURF(64D)	3988	4570	612.0	73.0	0.1625	0.1734	0.2956	0.0052	0.6367
6	ORB	5095	5345	854.0	149.0	0.0213	0.022	0.1586	0.0067	0.2086
7	ORB(1000)	1000	1000	237.0	21.0	0.0103	0.0101	0.0138	0.0049	0.0391
8	Bricks Dataset									
9	SIFT	1404	1405	427.0	16.0	0.1571	0.1585	0.068	0.0053	0.3889
10	SURF(128D)	2855	2332	140.0	34.0	0.1337	0.1191	0.2066	0.0051	0.4645
11	SURF(64D)	2855	2332	327.0	63.0	0.1307	0.1137	0.1173	0.0055	0.3672
12	ORB	978	942	323.0	17.0	0.0075	0.0079	0.0124	0.0049	0.0327
13	ORB(1000)	731	734	240.0	32.0	0.0067	0.0073	0.0088	0.005	0.0278
14	Mountain Dataset									
15	SIFT	1867	2033	170.0	45.0	0.1943	0.2017	0.1197	0.0047	0.5204
16	SURF(128D)	1890	2006	175.0	33.0	0.0979	0.113	0.1208	0.0051	0.3368
17	SURF(64D)	1890	2006	227.0	62.0	0.0978	0.1113	0.0689	0.0053	0.2833
18	ORB	4791	5006	340.0	78.0	0.0228	0.0236	0.1397	0.0052	0.1913
19	ORB(1000)	1000	1000	113.0	29.0	0.0118	0.0118	0.0117	0.0049	0.0402

Рисунок 3.9 – Результати для Building, Bricks та Mountain dataset

Алгоритм RANSAC з 2000 ітераціями і вірогідністю 99,5% був застосований для відкидання викидів і знаходження матриці гомографії. Це матриця 3×3, яка являє собою набір рівнянь, що задовольняють функцію перетворення першого зображення в друге. Повторюваність детектора ознак – це відсоток виявлених ознак, які витримують фотометричні або геометричні перетворення на зображенні. Повторюваність не пов’язана з дескрипторами і залежить лише від продуктивності частини детектування ознак, що входить до складу детектування-опису ознак. Він обчислюється на основі області перекриття (перетину) в парі зображень об’єкта. Детектор ознак з вищою повторюваністю для певного перетворення вважається надійнішим за інші, особливо для цього типу перетворень. Результати для dataset graffiti, roofs, river показано на рисунку 3.10.

	Unnamed: 0	Features Detected in the	Unnamed: 2	Features Matched	Outliers Rejected	Feature Detection &	Unnamed: 6	Feature Matching Time (s)	Outlier Rejection & Homography Calculation Time (s)	Total Image Matching Time (s)
0	Algorithm	Image Pairs				Description Time (s)				
1		1st Image	2nd Image			1st Image	2nd Image			
2	Graffiti Dataset									
3	SIFT	2654	3698	99.0	51.0	0.2858	0.3382	0.294	0.0073	0.9253
4	SURF(128D)	4802	5259	44.0	31.0	0.2166	0.2184	0.7399	0.0222	1.1971
5	SURF(64D)	4802	5259	62.0	42.0	0.2072	0.2142	0.3951	0.0169	0.8334
6	ORB	5527	7517	38.0	22.0	0.0277	0.0341	0.2129	0.0083	0.283
7	ORB(1000)	1000	1000	16.0	7.0	0.0146	0.0157	0.0114	0.0059	0.0476
8	Roofs Dataset									
9	SIFT	2303	3550	423.0	154.0	0.1983	0.2665	0.2475	0.0063	0.7186
10	SURF(128D)	2938	3830	171.0	95.0	0.1173	0.1523	0.3349	0.0084	0.6129
11	SURF(64D)	2938	3830	247.0	143.0	0.1165	0.1504	0.1847	0.009	0.4606
12	ORB	7660	11040	498.0	157.0	0.0296	0.0407	0.4131	0.0065	0.4899
13	ORB(1000)	1000	1000	91.0	32.0	0.0106	0.0113	0.0111	0.0055	0.0385
14	River Dataset									
15	SIFT	8619	9082	1322.0	192.0	0.6795	0.7083	0.2475	0.0092	3.6652
16	SURF(128D)	9434	10471	223.0	63.0	0.3768	0.4205	0.3349	0.0055	3.6649
17	SURF(64D)	9434	10471	386.0	66.0	0.3686	0.4091	0.1847	0.0049	2.2731
18	ORB	34645	35118	2400.0	553.0	0.1219	0.1276	0.4331	0.0092	5.64
19	ORB(1000)	1000	1000	40.0	6.0	0.0235	0.0235	0.0109	0.0046	0.0625

Рисунок 3.10 – Результати для Graffiti, Roofs та River dataset

Кожне значення часу, представлене в таблиці, є середнім значенням 100 вимірювань для мінімізації помилок, які виникають через збої в обробці. Синтетично згенеровані перетворення масштабу та обертання для зображень набору даних. Та доступні афінні перетворення для послідовностей графіті і стін відновлюються шляхом зіставлення зображень з кожним детектором-дескриптором ознак. Для обчислення похибок афінних перетворень зображення послідовностей графіті та стіна спочатку було деформовано відповідно до гомографій істинних та відновлених гомографій. Після цього було обчислено евклідову відстань окремо для кожної кутової точки відновленого зображення по відношенню до відповідної кутової точки на базовому зображенні

Можна зробити такі висновки:

– ORB виявляє найбільшу кількість ознак;

– SURF виявляє більше ознак, ніж SIFT;

– для SURF(64D) знайдено більше ознак порівняно з SURF(128D);

– об’єкти SIFT і SURF виявляються в розсіяному вигляді, як правило, по всьому зображенню, причому об’єкти SURF розташовані щільніше, ніж SIFT. ORB більш сконцентровані по кутах);

– ORB є найефективнішим дескриптором ознак з найменшими обчислювальними витратами;

– SURF(64D) має менші витрати на зіставлення ознак порівняно з SIFT. Це означає, що час пошуку ознак для SURF(64D) менший, ніж для SIFT(128D), але час пошуку ознак для SURF(128D) більший, ніж для SIFT(128D);

– ORB(1000) має найменшу вартість зіставлення ознак.

Обчислювальні витрати на одну точку на основі середніх значень для всіх пар зображень набору даних показано у таблиці 3.2.

Таблиця 3.2 – Витрати на одну точку

Algorithm	**Mean Feature-Detection- Description Time per Point (µs)**		**Mean Feature Matching Time per Point (µs)**
	1st Images	**2nd Images**	
SIFT	90,44	85,15	**142,02**
SURF(128D)	42,78	42,22	**168,55**
SURF(64D)	41,83	41,18	89,66
ORB	3,96	3,96	97,25
ORB(1000)	13,51	13,92	11,82

Повторюваність ORB для зіставлення зображень залишається стабільно високою, поки масштаб тестового зображення залишається в діапазоні від 60% до 150% по відношенню до еталонного зображення, але за межами цього діапазону повторюваність значно падає. Для обертання зображень ORB(1000), ORB, мають вищу повторюваність, ніж SIFT і SURF. SIFT виявився найточнішим детектором ознак-дескриптором для масштабу, повороту та афінних варіацій загалом.

Експериментальні результати надають багату інформацію та різні нові ідеї, які є цінними для прийняття важливих рішень в додатках на основі технічного зору. SIFT, SURF і BRISK виявилися найбільш масштабно-інваріантними детекторами ознак (на основі повторюваності), які пережили широкі варіації масштабу. ORB виявився найменш найменше масштабно-інваріантним. ORB(1000) є більш інваріантними до обертання, ніж інші. SIFT має вищу точність для поворотів зображень порівняно з іншими. Хоча ORB є найефективнішими алгоритмами, які можуть виявити величезну кількість ознак, час зіставлення для такої великої кількості ознак збільшує загальний час зіставлення зображення. ORB(1000) навпаки, виконує найшвидше зіставлення зображень, але точність при цьому знижується. Загальна точність SIFT виявилася найвищою для всіх типів геометричних перетворень, і SIFT визнано найточнішим алгоритмом.

Кількісне порівняння показало, що загальний порядок розташування детекторів-дескрипторів ознак за їх здатністю виявляти велику кількість ознак має такий вигляд: ORB, SURF, SIFT. Послідовність алгоритмів за обчислювальною ефективністю виявлення опису ознак на одну точку ознаки наступна: ORB, ORB(1000), SURF(64D), SURF(128D), SIFT. Порядок ефективного співставлення ознак для кожної точки ознаки наступний: ORB(1000), SURF(64D), ORB, SIFT, SURF(128D). Дескриптори-детектори ознак можна оцінити за швидкістю сумарного зіставлення зображень так: ORB(1000), SURF(64D), SIFT, ORB, SURF(128D).

3.3 Програмні модулі додатку автоматичного формування панорам

Для реалізації окремих функцій роботи програмної системи було використано цифрову бібліотеку OpenCV. Алгоритм, використаний у цьому дослідженні, можна порівняти з методом, представленим Брауном і Лоу у 2007 році. На відміну від попередніх підходів до зшивання зображень, які є

чутливими до послідовності вхідних зображень, метод Брауна та Лоу є більш стійким. Таким чином, він не нечутливий до варіацій у порядку, орієнтації, освітленості та шуму зображень, навіть якщо зображення були неправильно включені в панораму.

Крім того, завдяки використанню компенсації підсилення та змішування зображень, техніка зшивання зображень здатна генерувати панорамні зображення з більш візуально привабливим результатом. В OpenCV реалізовано метод, подібний до алгоритму Брауна та Лоу, за допомогою функцій cv2.createStitcher (OpenCV 3.x) та cv2.Stitcher_create (OpenCV 4), ця функція має єдиний параметр try_gpu, яка може збільшити пропускну здатність конвеєра зшивання зображень. Однак, підтримка графічних процесорів у OpenCV обмежена, і не вдалося домогтися ефективної роботи цього параметра. Тому необхідно завжди встановлювати його у значення False.

Лістинг 3.1 Реалізація Функція cv2.Stitcher_create у OpenCV 4:

```
Stitcher_create(...)
    Stitcher_create([, mode]) -> retval
    .   @brief Creates a Stitcher configured in one of the stitching
    .   modes.
    .
    .   @param mode Scenario for stitcher operation. This is usually
    .   determined by source of images to stitch and their transformation.
    .   Default parameters will be chosen for operation in given scenario.
```

Продовження лістингу 3.1

```
@return Stitcher class instance.

    OpenCV 3.x:
stitch(...) method of cv2.Stitcher instance
    stitch(images[, pano]) -> retval, pano
```

OpenCV 4.x:

stitch(...) method of cv2.Stitcher instance

stitch(images, masks[, pano]) -> retval, pano

. @brief These functions try to stitch the given images.

.

. @param images Input images.

. @param masks Masks for each input image specifying where to

. look for keypoints (optional).

. @param pano Final pano.

. @return Status code.

Ця функція отримує список вхідних зображень і намагається об’єднати їх у панораму. Отримане зображення панорами повертається у функцію, що його викликала. Змінна стану використовується для вказівки на успіх або невдачу процесу зшивання зображень, з чотирма можливими значеннями: 0, 1, 2, 3.

OK (0) означає успіх, тоді як ERR_NEED_MORE_IMGS (1) вказує на те, що для завершення панорами потрібні додаткові вхідні зображення. Зазвичай ця помилка виникає, коли на вхідних зображеннях виявлено недостатньо ключових точок.

ERR_HOMOGRAPHY_EST_FAIL = 2: ця помилка виникає, коли оцінка гомографії RANSAC є невдалою. Можливо, знадобиться більше зображень, або зображенням бракує унікальних текстур чи об’єктів для точної відповідності ключовим точкам.

ERR_CAMERA_PARAMS_ADJUST_FAIL = 3: така помилка пов’язана з труднощами у правильному визначенні внутрішніх та зовнішніх властивостей камери за вхідними зображеннями. Дві фотографії для тестування зшивання зображень показано на рисунку 3.11.

Рисунок 3.11 – Зображення для тестування

Задача полягає об’єднати ці два зображення в одне панорамне зображення. Щоб отримати бажаний результат, розглянемо основні операції обробки зображень, включаючи порогове значення, виділення контурів, морфологічні операції тощо.

По-перше, реалізовано 10-піксельну рамку з усіх боків зшитого зображення, щоб у подальшому можна було визначити контури всього контуру панорами. Потім створено версію зшитого зображення у відтінках сірого. Звідти сіре зображення буде пороговим, створено бінарне зображення панорами, де білі пікселі (255) представляють передній план, а чорні пікселі (0) – задній план.

Наступні кроки включають деформацію зображень на основі гомографії та їх злиття для створення бездоганного композитного зображення шляхом обчислення матриці гомографії на попередньому кроці. Цей процес забезпечує точне вирівнювання зшитих фотографій і плавні переходи між ними. Використовуючи розраховану гомографію, зображення можна деформувати Функція cv2.warpPerspective(), доступна в OpenCV, дозволяє деформувати зображення за допомогою перетворення гомографії. Функція потребує

вхідних параметрів, таких як оцінена матриця гомографії, розмір вхідного і вихідного зображень, а також вихідне або кінцеве зображення. Вихідне зображення матиме той самий розмір, що й еталонне. Щоб створити безшовне композитне зображення, викривлені зображення потрібно об’єднати разом.

Просте об’єднання зображень може призвести до появи помітних швів або різких змін у зображеннях. Для досягнення більш плавних ефектів змішування можна застосувати кілька технік, таких як альфа-змішування, розтушовування та багатосмугове змішування. Ці методи спрямовані на поступовий перехід інтенсивності пікселів між зображеннями, досягаючи плавного і реалістичного результату. Оптимальний метод накладання залежить від індивідуальних потреб і якостей фотографій, що зшиваються. Маючи порогове зображення, можемо застосувати виділення контурів, обчислити обмежувальну рамку найбільшого контуру і намалювати обмежувальну рамку (рис. 3.12).

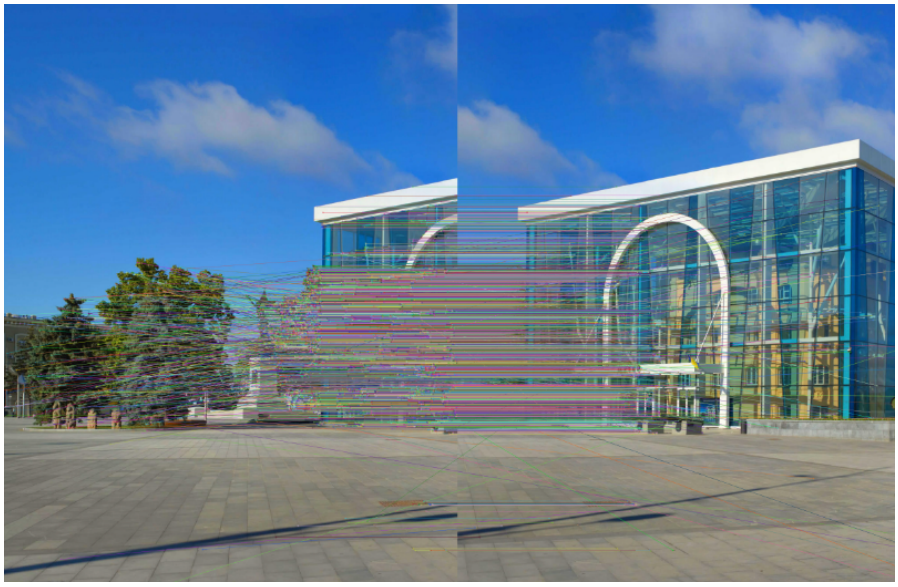


Рисунок 3.12 – Відповідність ключових точок між двома зображеннями

Функції OpenCV, здатні створювати точні, естетично привабливі панорами, одним із головних недоліків методу є те, що він абстрагує будь-який доступ до гомографічних матриць.cv2.createStitchercv2.Stitcher_create.

Лістинг 3.2 Реалізація виділення та аналізу контуру:

```
cnts = cv2.findContours(thresh.copy(),
cv2.RETR_EXTERNAL,<font></font>
                    cv2.CHAIN_APPROX_SIMPLE)<font></font>
                cnts = imutils.grab_contours(cnts)<font></font>
                c = max(cnts, key=cv2.contourArea)<font></font>
<font></font>
                mask = np.zeros(thresh.shape, dtype="uint8")<font></font>
                (x, y, w, h) = cv2.boundingRect(c)<font></font>
                cv2.rectangle(mask, (x, y), (x + w, y + h), 255, -1)<font></font>
```

Одне з припущень побудови панорами в реальному часі полягає в тому, що сама сцена не сильно змінюється з точки зору змісту. Після обчислення початкової гомографічну оцінки, необхідно лише час від часу перераховувати матрицю. Відсутність необхідності виконувати повномасштабне зіставлення ключових точок і оцінка RANSAC дає величезну швидкість під час побудови панорами, тому без доступу до необроблених гомографічних матриць було б важко використати алгоритм зшивання зображень OpenCV і перетворити його в режимі реального часу.

Метод зшивання зображень з використанням OpenCV і Python має широкий спектр застосування. Він чудово підходить для створення панорам у фотографії пейзажів, де потрібне об’єднання кількох знімків для отримання великого огляду. Також цей підхід можна застосувати в архітектурній зйомці, де важливо зберегти пропорції і перспективу будівель.

У сфері віртуальної реальності цей метод може використовуватися для створення панорамних зображень, забезпечуючи поглиблений досвід користувачів. Крім того, в робототехніці він може бути застосований для створення оглядів середовища, даючи змогу роботам ефективно сприймати і

взаємодіяти з навколишнім світом. Результат зшивання кількох зображень за допомогою OpenCV і Python (рис. 3.13).

Рисунок 3.13 – Результат зшивання двох зображень

ВИСНОВКИ

У рамках кваліфікаційної роботи було проведено високоінформативне дослідження у галузі автоматичного формування панорамних зображень. В роботі була зібрана база даних зображень, яка слугувала основою для наступних етапів дослідження. Існуючі алгоритми створення панорам включали детальний аналіз та порівняння існуючих методів пошуку ключових точок, таких як SIFT, SURF, ORB. Досліджено принципи функціонування алгоритму RANSAC для знаходження матриці перетворення, яка в свою чергу визначала оптимальне гомографічне з'єднання зображень.

Процес зшивання зображень був реалізований на основі знайденої матриці перетворення, а нові координати ключових точок другого зображення були розраховані для їхнього об'єднання з першим. Важливим етапом роботи було вивчення функціоналу бібліотеки OpenCV та вибір модулів, що оптимально відповідали задачам, поставленим перед дослідженням. Відмінною особливістю використання OpenCV було розглядання його методів для визначення об'єктів на зображеннях та можливостей покращення якості зображень з метою отримання кращих результатів роботи алгоритму.

На основі розробленого алгоритму було успішно реалізовано програмне забезпечення, яке включало в себе усі етапи обробки та об'єднання зображень у високоякісні панорами.

Результати дослідження апробовано у вигляді тез доповідей під час VI Міжнародна науково-практична конференція «Methodical and practical methods of creating inventions» [45].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Gao, S., Yang, K., Shi, H., Wang, K., & Bai, J. (2022). Review on panoramic imaging and its applications in scene understanding. IEEE Transactions on Instrumentation and Measurement, 71, 1-34.

2. Zhang, J., Yang, K., Ma, C., Reiß, S., Peng, K., & Stiefelhagen, R. (2022). Bending reality: Distortion-aware transformers for adapting to panoramic semantic segmentation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 16917-16927).

3. Yu, D., & Ji, S. (2019). Grid based spherical cnn for object detection from panoramic images. Sensors, 19(11), 2622.

4. Dias, M. J. M., Franco, A., Junqueira, J. L. C., Fayad, F. T., Pereira, P. H., & Oenning, A. C. (2020). Marginal bone loss in the second molar related to impacted mandibular third molars: comparison between panoramic images and cone beam computed tomography. Medicina Oral, Patologia Oral y Cirugia Bucal, 25(3), e395.

5. Кобилін, О. А., & Творошенко, І. С. (2021). Методи цифрової обробки зображень.

6. Wang, B., Zhong, S., Lee, T. L., Fancey, K. S., & Mi, J. (2020). Non-destructive testing and evaluation of composite materials/structures: A state-of-the-art review. Advances in mechanical engineering, 12(4), 1687814020913761.

7. Gupta, S., Thakur, K., & Kumar, M. (2021). 2D-human face recognition using SIFT and SURF descriptors of face's feature regions. The Visual Computer, 37, 447-456.

8. Ansari, S. (2019, February). A review on SIFT and SURF for underwater image feature detection and matching. In 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT) (pp. 1-4). IEEE.

9. Bakar, S. A., Jiang, X., Gui, X., Li, G., & Li, Z. (2020, October). Image stitching for chest digital radiography using the SIFT and SURF feature extraction

by RANSAC algorithm. In Journal of Physics: Conference Series (Vol. 1624, No. 4, p. 042023). IOP Publishing.

10. Maietta, R., Mihas, P., Swartout, K., Petruzzelli, J., & Hamilton, A. B. (2021). Sort and Sift, Think and Shift: Let the Data Be Your Guide An Applied Approach to Working With, Learning From, and Privileging Qualitative Data. Qualitative Report, 26(6).

11. Gupta, S., Thakur, K., & Kumar, M. (2021). 2D-human face recognition using SIFT and SURF descriptors of face's feature regions. The Visual Computer, 37, 447-456.

12. Katoch, S., Singh, V., & Tiwary, U. S. (2022). Indian Sign Language recognition system using SURF with SVM and CNN. Array, 14, 100141.

13. Wang, C., Zhang, Z., Li, Q., & Zhou, X. (2019). An image copy-move forgery detection method based on SURF and PCET. IEEE Access, 7, 170032-170047.

14. Taunk, K., De, S., Verma, S., & Swetapadma, A. (2019, May). A brief review of nearest neighbor algorithm for learning and classification. In 2019 international conference on intelligent computing and control systems (ICCS) (pp. 1255-1260). IEEE.

15. Gupta, S., Kumar, M., & Garg, A. (2019). Improved object recognition results using SIFT and ORB feature detector. Multimedia Tools and Applications, 78, 34157-34171.

16. Сидоренко, Ю. В., & Городецький, М. В. (2020). Аналіз роботи алгоритму інтерполяційної функції Гауса на елементарних алгебричних функціях. Сучасні проблеми моделювання, (19), 138-145.

17. Бодянський, Є. В., Шафроненко, А. Ю., & Климова, І. М. (2021). Метод адаптивної достовірної нечіткої кластеризації даних на основі еволюційного алгоритму. Збірник наукових праць Харківського національного університету Повітряних Сил, (2 (68)), 80-83.

18. Wang, L., Chen, P., Chen, L., & Mou, J. (2021). Ship AIS trajectory clustering: An HDBSCAN-based approach. Journal of Marine Science and Engineering, 9(6), 566.

19. Stewart, G., & Al-Khassaweneh, M. (2022). An implementation of the HDBSCAN* clustering algorithm. Applied Sciences, 12(5), 2405.

20. Park, K., Hong, J. S., & Kim, W. (2020). A methodology combining cosine similarity with classifier for text classification. Applied Artificial Intelligence, 34(5), 396-411.

21. Mashtalir, S., Mashtalir, V., & Stolbovyi, M. (2018, August). Representative based clustering of long multivariate sequences with different lengths. In 2018 IEEE second international conference on Data Stream Mining & Processing (DSMP) (pp. 545-548). IEEE.

22. Chiu, W. Y., Yen, G. G., & Juan, T. K. (2016). Minimum manhattan distance approach to multiple criteria decision making in multiobjective optimization problems. IEEE Transactions on Evolutionary Computation, 20(6), 972-985.

23. Klove, T., Lin, T. T., Tsai, S. C., & Tzeng, W. G. (2010). Permutation arrays under the Chebyshev distance. IEEE Transactions on Information Theory, 56(6), 2611-2617.

24. Mashtalir, S., Mashtalir, V., & Stolbovyi, M. (2017). Video shot boundary detection via sequential clustering. International Journal “Information Theories and Applications, 24(1), 50-59.

25. Руденко, В. О. (2020). Використання комп’ютерного зору для стеження за морганням очей.

26. Brachmann, E., & Rother, C. (2019). Neural-guided RANSAC: Learning where to sample model hypotheses. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 4322-4331).

27. Barath, D., Cavalli, L., & Pollefeys, M. (2022). Learning to find good models in RANSAC. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 15744-15753).

28. Shen, X., Darmon, F., Efros, A. A., & Aubry, M. (2020). Ransac-flow: generic two-stage image alignment. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16 (pp. 618-637). Springer International Publishing.

29. Li, J., Hu, Q., & Ai, M. (2021). Point cloud registration based on one-point ransac and scale-annealing biweight estimation. IEEE Transactions on Geoscience and Remote Sensing, 59(11), 9716-9729.

30. Чугаев, А. А. (2019). Дослідження питання нормалізації геометричних перетворень на основі аналізу дескрипторів характерних точок.

31. Yakovleva, O., & Nikolaieva, K. (2020). Research of descriptor based image normalization and comparative analysis of SURF, SIFT, BRISK, ORB, KAZE, AKAZE descriptors. Advanced Information Systems, 4(4), 89-101.

32. Abu-Jassar, A. T., Al-Sharo, Y. M., Lyashenko, V., & Sotnik, S. (2021). Some Features of Classifiers Implementation for Object Recognition in Specialized Computer systems. TEM Journal, 10(4).

33. Tvoroshenko, I., Gorokhovatskyi, V., Kobylin, O., & Tvoroshenko, A. (2023). Application of deep learning methods for recognizing and classifying culinary dishes in images.

34. Jocher, G., Stoken, A., Chaurasia, A., Borovec, J., Kwon, Y., Michael, K., ... & Thanh Minh, M. (2021). ultralytics/yolov5: v6. 0-YOLOv5n'Nano'models, Roboflow integration, TensorFlow export, OpenCV DNN support. Zenodo.

35. Christa, G. H., Jesica, J., Anisha, K., & Sagayam, K. M. (2021, May). CNN-based mask detection system using openCV and MobileNetV2. In 2021 3rd International Conference on Signal Processing and Communication (ICPSC) (pp. 115-119). IEEE.

36. Sigut, J., Castro, M., Arnay, R., & Sigut, M. (2020). OpenCV basics: A mobile application to support the teaching of computer vision concepts. IEEE Transactions on Education, 63(4), 328-335.

37. Gupta, N., Sharma, P., Deep, V., & Shukla, V. K. (2020, June). Automated attendance system using OpenCV. In 2020 8th International Conference on

Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO) (pp. 1226-1230). IEEE.

38. Bradski, G., & Kaehler, A. (2000). OpenCV. Dr. Dobb's journal of software tools, 3(2).

39. Mashtalir, S., & Mashtalir, V. (2016, August). Sequential temporal video segmentation via spatial image partitions. In 2016 IEEE First International Conference on Data Stream Mining & Processing (DSMP) (pp. 239-242). IEEE.

40. Ma, B., Zimmermann, T., Rohde, M., Winkelbach, S., He, F., Lindenmaier, W., & Dittmar, K. E. (2007). Use of autostitch for automatic stitching of microscope images. Micron, 38(5), 492-499.

41. Ніколаєва, К. Г. (2019). Розробка та дослідження методу нормалізації геометричних перетворень зображень на основі аналізу характерних точок.

42. Carlson, R. E., & Fritsch, F. N. (1985). Monotone piecewise bicubic interpolation. SIAM journal on numerical analysis, 22(2), 386-400.

43. Gao, S., & Gruev, V. (2011). Bilinear and bicubic interpolation methods for division of focal plane polarimeters. Optics express, 19(27), 26161-26173.

44. Ward, C. M., Harguess, J., Crabb, B., & Parameswaran, S. (2017, September). Image quality assessment for determining efficacy and limitations of Super-Resolution Convolutional Neural Network (SRCNN). In Applications of Digital Image Processing XL (Vol. 10396, pp. 19-30). SPIE.

45. Rossina, T. (2023). Methodical and practical methods of creating inventions.