Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

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

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

ВИКОРИСТАННЯ ТА ДОСЛІДЖЕННЯ МОРФОЛОГІЧНИХ ОПЕРАЦІЙ OPENCV
(тема)

Виконав:
студент 4 курсу, групи ІТІНФ-18-1

Федорченко Є. В.
(прізвище, ініціали)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Кузьомін О. Я.
(посада, прізвище, ініціали)

Допускається до захисту

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

2022 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти перший (бакалаврський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

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

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Федорченку Євгену Віталійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Використання та дослідження морфологічних операцій OpenCV

затверджена наказом університету від 20 травня 2021 року No 663СТ

2. Термін подання студентом роботи до екзаменаційної комісії 24 травня 2021 р.

3. Вихідні дані до роботи Науково-методична та науково-технічна література, матеріали конференцій, дані інтернет-мережі, відкрита програмна бібліотека комп'ютерного зору OpenCV

4. Перелік питань, що потрібно опрацювати в роботі
1. Аналіз точності та ефективності існуючих методів та алгоритмів розпізнавання об'єктів
2.Розробка модифікованого методу розпізнавання шляхом використання морфологічних операцій
3. Розробка програмного забезпечення для розпізнавання об'єктів з використанням модифікованого методу

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність проблеми, мета роботи, постановка задачі, етапи виконання роботи, результати тестування, перспективи подальшої роботи

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

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

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	18.04.2022	
2	Аналіз завдання, підбір літератури	18.04.22-25.04.22	
3	Аналіз літератури з досліджуваної проблеми	26.04.22-27.04.22	
4	Огляд існуючих алгоритмів попередньої обробки зображень	28.04.22-29.04.22	
5	Огляд існуючих алгоритмів розпізнавання образів	30.04.22-02.05.22	
6	Програмна реалізація	03.05.22-20.05.22	
7	Оформлення пояснювальної записки	20.05.22-05.06.22	
8	Перевірка на плагіат	09.06.22	
9	Рецензування	10.06.22	
10	Підготовка презентації та доповіді	11.06.22-13.06.22	
11	Занесення роботи в електронний архів	14.06.22	
12	Попередній захист кваліфікаційної роботи	14.06.22	

Дата видачі завдання 18 квітня 2022 р.

Студент _____

(підпис)

Керівник роботи _____ доц. Кузьомін О. Я._____

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 65 с., 5 табл., 22 рис., 24 джерело.

АЛГОРИТМ ЛУКАСА – КАНАДЕ, АЛГОРИТМ HORN–SCHUNCK.

Об’єктом роботи є комп’ютерний зір для розпізнавання об’єктів за даними відеопотоку.

Метою роботи є розробка методів, що базуються на використані модифікованого методу ідентифікації для підвищення якості розпізнавання об’єктів

Використано методи комбінованого використання фільтрації шуму, гама-корекції, морфологічних операцій та використання багатопоточності для розпізнавання. Досліджено метод детектування об’єктів за допомогою модифікованого методу ідентифікації. У результаті роботи здійснена програмна реалізація системи для розпізнавання об’єктів для різних варіантів неякісних вхідних даних в режимі реального часу.

LUCAS-CANADA ALGORITHM, HORN-SCHUNCK ALGORITHM.

The object of the work is a computer vision for recognition of objects based on video stream data.

The aim of the work is to develop methods based on the modified identification method to improve the quality of recognition of objects

Methods of combined use of noise filtration, gamma-correction, morphological operations and use of baggage flow for recognition were used. The method of detecting objects by means of modified identification method was investigated. As a result of the work, a program implementation of the system for recognition of objects for different variants of unequal input data in real time mode was carried out.

ЗМІСТ

ВСТУП

Для роботизований пристроїв визначення об’єктів, що рухаються є важливою задачею котра використовує комп’ютерний зір. Отримання, обробка, аналіз цифрового зображення та його розуміння це включає в себе комп’ютерний зір.

Це робиться за допомогою статичних методів та моделей побудованих за допомогою теорії статистичного навчання, геометрії, фізики, статистики. Завдання розробки систем відеоспостереження є важливою задачею на сьогоднішній день і впливає на життя людей. З появою дешевих цифрових відеореєгістраторів з’явилася можливість обробляти данні комп’ютером для виконування таких задач як охорона периметра або території об’єкта, розпізнавання руху або осіб. Для запобігання та виявлення злочинів та моніторингу дорожнього руху широко використовуються системи відеоспостереження. Завдяки появі потужних комп’ютерів та недорогих високоякісних камер та великих потреб у автоматичному аналізі відео збільшився інтерес до алгоритмів супроводження руху. Основи відеоаналітики це розпізнавання рухомого об’єкта, відстеження його на кожному кадрі та аналіз його поведінки. Алгоритм стеження за об’єктом може використовуватися у таких задачах:

– розпізнавання на основі руху;

– автоматизоване спостереження, що виявляє підозрілу активність;

– взаємодія людина-комп’ютер, наприклад, слідкування за поглядом для ввода даних, розпізнавання жестів;

– навігація машин, що пов’язана з планування маршруту та уникання перешкод.

Актуальність роботи полягає в наступному: покращення алгоритму пошуку об’єктів відеопотоку, використовуючи багатопоточність для

розпізнавання, фільтрації шуму, гама-корекції, морфологічні операції комбінованим методом.

Відстежування об’єктів є важкою тому що:

– нестача інформації через проекцію трьохвимірного світу на зображення;
– шум в зображеннях;
– складні рухи об’єктів;
– часткова або повна оклюзія об’єктів;
– зміна освітлення;
– потреба у визначенні об’єктів у реальному часі.

Завдяки тому що на сьогоднішній день є потужна обчислювальна техніка, котра дозволяє обробляти великі потоки даних в реальному часі, динамічні методи обробки зображень розвиваються з великою швидкістю.

Це підвищує актуальність автоматизованих методів динамічної сегментації для розпізнавання рухомих об'єктів.

Мета роботи – вдосконалити процес розпізнавання об’єктів відеопотоку використовуючи морфологічні операції, фільтрацію шуму та зчитуванням даних у окремий потік. Для досягнення мети потрібно:

– розглянути методи розпізнавання та виділення об'єктів;
– обрати метод розпізнавання відповідно до вимог;
– розробити програмне забезпечення для розпізнавання об'єкта в відеопотоці;
– провести тест розробленої програми за допомогою використання різних неякісних вхідних даних.

1 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

1.1 Особливості і проблематика

При створені та використані системи відеоспостереження виникають проблеми з некоректним визначенням потрібних об’єктів. Об’єкти на задньому фоні із-за схожості з цільовими об’єктами спостереження можуть призвести к некоректному визначенню об’єктів, що як ми очікуємо, будуть рухатись. На рис. 1.1 зображено приклад помилкового визначення об’єкту.

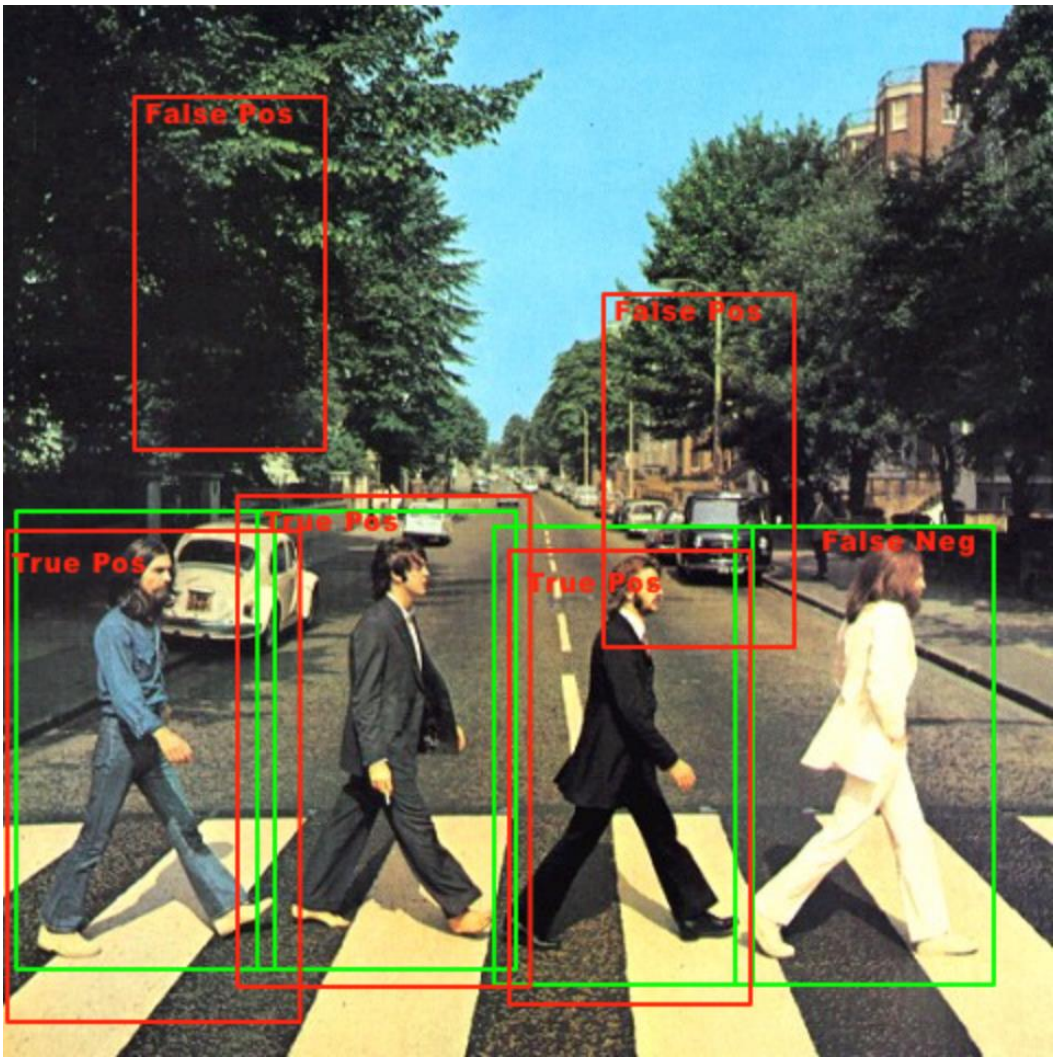

Рисунок 1.1 – Помилкове визначення об’єктів

Важко відстежити потрібний об’єкт, вигляд котрого змінився в об’єктиві із-за таких факторів:

– якщо наприклад об’єкт крутиться, він змінює свій вигляд та позицію;

– зміна параметрів освітлення таких як інтенсивність, колір, напрямок, може впливати на вигляд об’єкту. Це може впливати на подальше відстеження об’єкту;

– якість матриці камери напряму впливає на кількість шуму в отриманих зображеннях;

– ціль спостереження може бути перекрита іншими об’єктами.

Перекриття можуть виникати коли ціль рухаються за іншими об’єктами або інші об’єкти, що рухаються, перекривають об’єкт спостереження;

1.2 Способи зображення об’єктів

Об’єктом, що відстежуються може бути будь-що визначене важливим для подальшого аналізу. Об’єкт визначаються за їх формою та виглядом. Широко використовують наступні методи представлення об’єкту відстеження, зображено на рис. 1.2:

– об’єкти представляють точкою, або набір точок(рис. 1.2(b)).

– краще підходить для стеженням за малими областями зображення;

– форма представлена геометричною фігурою, наприклад еліпс або прямокутник(рис. 1.2(c, d)). Призначається для відстеження твердих об’єктів;

– об’єкт позначається контуром на границях об’єкту (рис. 1.2(g, h)). Всередині контуру силует об’єкту (рис. 1.2(i)). Призначається для відстеження нетвердих об’єктів;

– об’єкт складаються з декількох об’єктів котрими являються частини тіла з’єднані суглобами (рис. 1.2(e)). Використовується для відстеження об’єктів з суглобами.

– скелетна модель (рис. 1.2(f)). Дуже поширена модель для відстеження та розпізнавання об’єктів.

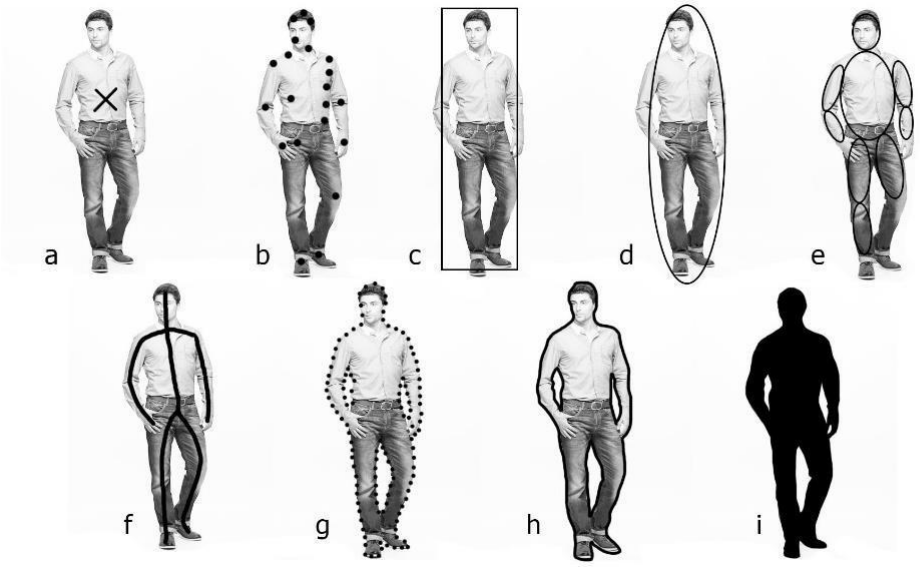

Рисунок 1.2 – Способи виділення об’єкта: a – центр об’єкта, b – особливі точки, c – форма об’єкта у вигляді прямокутника, d – об’єкт у формі еліпса, e – поєднання фігур, f – скелетна модель, g та h – контур об’єкта, i – силует

Для виділення об’єктів частіше використовують такі методи:

– параметризація щільності ймовірностей вигляду об’єкту за методом зміщування за Гаусом або за Гаусом. Щільності ймовірностей вигляду об’єкту може бути і непараметризована, наприклад вікно Парзена;

– з примітивних силуетів та форм формувати шаблони. Ці шаблони несуть просторову інформацію та зовнішній вигляд. Доречні коли об’єкт нерухомий;

– активна модель вигляду. Генерується одночасно форма і вигляд об’єкта. Об’єкт у моделі уявляє собою набір орієнтирів. Вектор вигляду робиться для кожного орієнтира і визначається збіганням кольору, градієнту, текстури;

– багаторакурсна модель описує об’єкт з різних ракурсів.

1.3 Виділення характерних рис для відстежування

У відстежуванні об’єктів ключовим являється виділення коректних рис об’єкту. Зазвичай об’єкт потрібен мати унікальні візуальні риси, щоб об’єкт можна було виділити на фоні інших об’єктів. Представлення об’єкта тісно пов’язано з виділенням його характерних рис. Характерними рисами можуть бути границі об’єкту для представлення контура. Основні візуальні риси:

– видимий колір, може бути представлений в RGB чи HSV (Hue, Saturation, Value);

– інтенсивність зображення змінюється кордонами об’єкта. Ці зміни знаходять за допомогою розпізнавання границь. Низька чутливість до змін світла - важлива властивість границь. Одним з найпоширеніших є детектор Canny;

– оптичний потік – це представлення видимого сліду руху об'єктів, поверхонь, і граней візуальної сцени, що спостерігається під час відносного руху між спостерігачем і сцени. Поширеними алгоритми знаходження оптичного потоку: Хорна, Лукас-Канаде;

– текстура – це міра зміни інтенсивності поверхні, яка визначає так характеристики як рівність та постійність.

1.4 Оптичний поток і його алгоритми

При аналізі руху нерідко проводять дослідження оптичного потоку. Є три етапи виділення динамічних об'єктів: попередня стабілізація параметрів зображення, виділення рухомих точок або областей, співставлення рухомих точок і областей між кадрами. Алгоритми знаходження руху виділяють рухомі точки та області. Розрахунок оптичного потоку(ОП) зазвичай проводять глобальними і локальними диференційними алгоритмами.

Для швидкого розрахунку оптичного потоку використовують локальні диференційні алгоритми, але точність при використанні цього методу може буди недостатньою для стабільного виділення рухомих об’єктів.

При використані глобальних методів розрахунку оптичний поток є більш точним, але із-зі високої складності розрахунку не завжди цей метод може бути використаний у реальних задачах.

Алгоритми оптичного потоку також використовуються для аналізу структури сцени, та 3D-руху об’єктів і спостерігача відносно сцени. Також оптичний поток може використовуватися при вивчені структури об’єктів.

В робототехніці використовують оптичний потік для вирішення таких задач: розпізнавання та слідкування за об’єктом, визначення руху та для навігації робота.

Послідовності впорядкованих зображень можуть бути представлені як дискретне зміщення рухомого об’єкту, або миттєву швидкість зображення.

При розрахунку оптичного потоку ставлять таку задачу: по даним про яскравість пікселей оцінити зміщення зображення в кожній точці на послідовних кадрах. Популярні методи розрахунку можна виділити у 3 групи: методи, основані на співставленні середовища, диференційні методи та розрахунок фазової кореляції.

Диференційні методи це роботи Лукаса і Канаде та Хорна і Шунка.

Алгоритм Лукаса – Канаде один з алгоритмів, що часто використовують в задачах комп’ютерного зору. Це локальний диференційний метод розрахунку оптичного потоку.

Метод спирається на те, що в локальному околі пікселя, що розглядається, потік є суттєво сталий, і для всіх пікселів у цьому околі розв’язує рівняння оптичного потоку методом найменших квадратів.

Поєднуючи інформацію з кількох сусідніх пікселів, метод Лукаса-Канаде в більшості може розв’язувати невизначеність рівняння оптичного потоку. Цей метод менш чутливий до різного шуму в зображені при порівняні з потоковими методами. Але проблемою цього методу є те, що оскільки він

локальний, то не здатен всередині потіку однорідних областей зображення надавати інформацію.

Головним принципом методе Лукаса-Канаде є припущення, що зміщення об'єктів на зображені між сценами є невеликим і приблизно сталим в межах около точки p, яку розглядають. При цьому можна вважати, що для всіх пікселів у межах вікна з центром в p виконується рівняння оптичного потоку.

Вектор локального потоку(Vx, Vy) зображення мусить задовольняти

$$\begin{cases} \mathrm{I_x}(q_1)\mathrm{Vx}+\mathrm{I_y}(q_1)\mathrm{Vy} = -\mathrm{I_t}(q_1) \\ \mathrm{I_x}(q_2)\mathrm{Vx}+\mathrm{I_y}(q_2)\mathrm{Vy} = -\mathrm{I_t}(q_2) \\ \mathrm{I_x}(q_n)\mathrm{Vx}+\mathrm{I_y}(q_n)\mathrm{Vy} = -\mathrm{I_t}(q_n), \end{cases} \quad (1.1)$$

де q1,q2,...,qn це пікселі у вікні, а $\mathrm{I_x}(q_i)$, $\mathrm{I_y}(q_i)$, $\mathrm{I}_t(q_i)$ частина похідного зображення I за положенням за x, y та часом t, оцінювані в точці qi у поточний момент часу .

Це рівняння також може бути записане у матричному вигляді Av = b, де

$$A = \begin{bmatrix} \mathrm{I_x}(q_1) & \mathrm{I_y}(q_1) \\ \mathrm{I_x}(q_2) & \mathrm{I_y}(q_2) \\ \mathrm{I_x}(q_n) & \mathrm{I_y}(q_n) \end{bmatrix} \quad v = \begin{bmatrix} V_x \\ V_y \end{bmatrix} \quad b = \begin{bmatrix} -\mathrm{I_t}(q_1) \\ -\mathrm{I_t}(q_2) \\ -\mathrm{I_t}(q_n) \end{bmatrix}. \quad (1.2)$$

Особливість цієї системи рівнянь є те, що воно має більше рівнянь ніж невідомих. За допомогою принципу найменших квадратів отримуємо компромісний розв'язок за методом Лукаса-Канаде. Він розв'язує систему 2X2 $A^T A_v = A^T b$ або $v = (A^T A)^{-1} A^T b$

Де A^T це транспонування матриці A. Тобто, він обчислює

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i \mathrm{I_x}(q_i)^2 & \sum_i \mathrm{I_x}(q_i)\mathrm{I_y}(q_i) \\ \sum_i \mathrm{I_y}(q_i)\mathrm{I_x}(q_i) & \sum_i \mathrm{I_y}(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i \mathrm{I_x}(q_i)\mathrm{I_y}(q_i) \\ -\sum_i \mathrm{I_y}(q_i)\mathrm{I_x}(q_i) \end{bmatrix}. \quad (1.3)$$

де середня матриця в цьому рівнянні це обернена матриця.(рис. 1.3)

Рисунок 1.3 – Приклад визначення оптичного потоку методом Лукаса-Канаде

Алгоритм Хорна-Шунка використовує глобальну функцію енергії для оцінки оптичного потоку:

$$E = \iint [(Ixu + Iyv + It)2 + \alpha 2(|| u||2 + || v||2)]dxdy. \quad (1.4)$$

де v(r) це векторне поле котрим визначається оптичний потік. При вирішені рівняння Ейлера-Лагранжа можна знайти екстремум енергії.

Метод Хорна-Шунка більш глобальний, ніж метод Лукаса-Канаде. Основним являється те, що на зображенні оптичний потік достатньо гладкий. Від того самого рівняння:

$$I_x dx + I_y dy + I_t dt = 0. \quad (1.5)$$

Перехід к функціоналу:

$$(I_x dx + I_y dy + I_t dt)^2 + \alpha^2(|\partial u|^2 + |\partial v|^2). \quad (1.6)$$

Додаємо вимогу на відсутність значних зміщень з коефіцієнтом a. Це призводить нас до системи з двох рівнянь:

$$I_x^2 u + I_x I_y v = \alpha^2 \quad u - I_x I_t, \tag{1.7}$$

$$I_x I_y u + Iy^2 v = \alpha^2 \quad v - I_y I_t. \tag{1.8}$$

В рівняннях лапсасіан пропонують розрахувати наближено:

$\Delta u =< u > -u;$ - різниця з середнім значенням. Записуємо систему рівнянь для кожного пікселя і вирішуємо ітеративно:

$$u_{n+1} = u_n - I_x(I_x u_n + I_y v_n + I_t)/(I_x^2 + I_y^2 + \alpha^2), \tag{1.9}$$

$$v_{n+1} = v_n - I_y(I_x u_n + I_y v_n + I_t)/(I_x^2 + I_y^2 + \alpha^2). \tag{1.10}$$

В алгоритмі також іноді використовують масштабування з коефіцієнтом 0.65. Для зображень більше 105 пікселів алгоритм Хорна-Шунка підходить погано із-за великої складності.

Його часова складність має порядок Θ(SN), де S — кількість пікселей у зображенні, N — кількість ітерацій метода Якобі або Гауса–Зейделя.

Знаходження оптичного потоку це проблема, що вивчається не перше десятиліття, і методи обробки оптичного потоку продовжують вдосконалювати. При близькому розгляді це дуже складна проблема, а від точності визначення руху об’єкту залежить ефективність багатьох алгоритмів. (рис. 1.4)

Рисунок 1.5 – Приклад визначення оптичного потоку методом Хорна-Шунка

1.5 Методи виявлення об’єктів

Для виявлення об’єктів будують набор правил, яким повинен відповідати об’єкт, що виявляється, на фрагменті зображення. Об’єкт може виявлятись за допомогою заданих шаблонів. При використані шаблону перевіряється кожна область зображення на відповідність шаблону.

Існує багато алгоритмів виявлення об’єктів. З них можна виділити деякі актуальні методи. Розглянемо недоліки та переваги цих алгоритмів.

Важливу роль у вирішені задач розпізнавання та виділення об’єктів в системах комп’ютерного зору грає сегментація зображень. Є такі вимоги до оброблюваних областей:

– однорідність області, тобто внутрішня частика області не повинні мати складних елементів чи великої кількості отворів, які не відносяться до області;

– велика різниця за характеристиками між суміжними областями, щодо яких вони вважаються однорідними;

– явні межі сегментів. При правильній сегментації кінцевий результат подальшого аналізу буде значно кращім.

Алгоритм Віола-Джонса дозволяє виявляти об’єкти у реальному часі та може застосовуватись для рішення різних задач, але ціль його розробки була виявлення обличь.

Цей алгоритм має 4 етапи:

– вибір функцій, схожих на Хаар;

– створення цілісного образу;

– запуск тренувань AdaBoost;

– створення каскадів класифікаторів.

Алгоритм розглядає невеликі субрегіони і пробує знайти на них обличчя, розглядаючи регіони на наявність конкретної особливості. Оскільки на зображені може міститись безліч граней різного розміру, потрібно перевіряти велику кількість різних масштабів та положень. Для виявлення обличь розробники алгоритму використовували функцію Хаар. Метод засновано на таких принципах:

– інтегральне представлення зображення. При використані цього методу можливо швидко обчислювати необхідні об’єкти. Інтегральне представлення зображення – матриця, яка співпадає по розмірам з розмірами вхідного зображення в пікселях. Елементи матриці уявляють собою суму інтенсивності пікселів, які знаходяться лівіше і вище даного елемента;

– пошук потрібного об’єкту робиться за допомогою використання ознак, схожих на ознаки Хаара;

– вибираються ознаки алгоритмом бустингу;

– побудова композиції алгоритмів машинного навчання, у якій кожен наступний алгоритм виправляє недоліки попереднього;

– класифікатор, що робить висновок чи являється об’єкт, що розглядається, об’єктом який треба виявити;

– каскади ознак використовують при виявлені потрібного об’єкту. Ідеєю каскаду ознак є побудова послідовності класифікаторів, в яких наступний класифікатор намагається врахувати помилки попередніх;

– метод скануючого вікна використовується для зображень: зображення сканується вікном пошуку, до кожного вікна застосовуються класифікатори.

Переваги:

– популярність і розповсюдженість алгоритму;

– завдяки використанню каскадного класифікатора досягається велика швидкість виявлення;

– у порівняні з більш повільними алгоритмами, висока точність знаходження і низький процент помилок у виявлені;

Недоліки:

– для навчання необхідна велика вибірка та багато часу;

– обмеження на положення об'єкта при знаходженні.

Цей алгоритм порівнює ділянки зображення з шаблоном. Застосування цього алгоритму доречно у галузі робототехніки та виробництві. При розбіжності масштабу або повороті об’єкт виявлено не буде.

Це простий метод, в якому ми знаходимо найбільш схожу на шаблон область. Коли змін масштабу та повороту немає цей метод працює добре. Метод працює за принципом:

– зображення накладається на шаблон;

– зображення переміщається по шаблону в пошуку збігів;

Коли зображення змінює положення, розраховується збіг з шаблоном та записується у матрицю результатів.

Після закінчення порівнянь співпадіння знаходяться в глобальних мінімумах або максимумах, залежить від методу. Деякі методи використовують метод найменших квадратів, кореляції та інше. Цей алгоритм найпростіший, але и точність його низька.(рис 1.6)

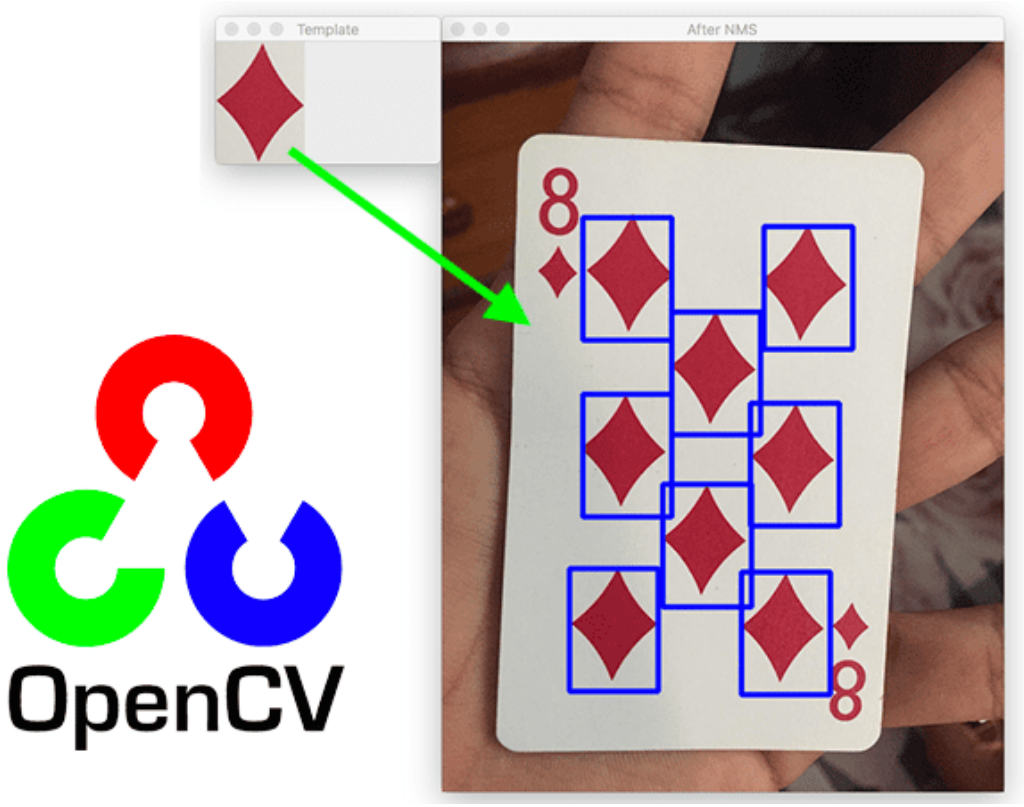

Рисунок 1.6 – Пошук об'єкта методом порівняння з шаблоном

У задачах в області комп'ютерного зору і обробки зображень однією з важливих задач є знаходження переднього плану. Це допомагає знаходити зміни в послідовності зображень. Метод, що дозволяє вилучити передній фон, називається віднімання фону. (рис 1.7)

Рисунок 1.7 – Визначення об’єктів методом віднімання заднього фону

1.6 Постановка задачі

Актуальність теми. В останній час широке розповсюдження отримує відеоаналітика - технологія, яка використовує комп'ютерний зір для автоматизованого збору інформації у послідовності кадрів, які отримуються з відеокамер в реальному часі чи з відеозаписів. Дана технологія може бути застосована в відеоспостереженні, системах безпеки, транспорті.

Об’єктом роботи є розробка програмного забезпечення для відеоаналізу з модифікованим методом розпізнавання об’єктів.

Метою роботи є розробка програмного забезпечення із застосуванням методів попередньої обробки зображення, спрямованого на покращення якості розпізнавання об’єктів.

Але сучасні методи відеоаналітики мають низку точність розпізнавання об’єктів в умовах зміни освітлення або великої кількості оптичних шумів.

Для підвищення якості розпізнавання об’єктів, необхідно розробити модифікований метод розпізнавання об’єктів.

2 ВІДСТЕЖЕННЯ ОБ'ЄКТІВ У ВІДЕОПОТОЦІ

2.1 Відстеження об'єктів у відеопотоці

Задача виявлення рухомих об'єктів – виділення ділянок зображення в відеопотоці, які мають подібні ознаки з заданим об'єктом. Ознаки задаються залежно від алгоритму. Трекінг – визначення положення динамічного об'єкта в реальному часі за допомогою камери. Положення рухомих об'єктів встановлюється алгоритмом, який аналізує відеопотік.

Найбільшою проблемою систем відстеження є зіставлення об'єкта на різних кадрах, особливо при високій швидкості об'єкту у порівняні з кадровою частотою. Ці системи використовують модулі руху, які можуть описати змінення вигляду об'єкта при різноманітних рухах. Прикладами таких моделей є:

– відстеження 2D об'єктів, модулю руху є двомірне перетворення об'єкту;

– якщо об'єкт твердий та трьохвимірний тоді модулю руху буде його вигляд при різних ракурсах;

– для стиснення відео, ключові кадри (англ. key frames) розділяють на макроблоки (англ. macroblocks). Модель руху в цьому випадку є розрив ключових кадрів, де макроблоки перетворюють використовуючи вектори руху, отриманого з параметрів руху;

– зображення деформовного об'єкта може бути покрите сіткою (англ. mesh), рух об'єкта задають положенням вершин цієї сітки.

Алгоритми відстеження для виконання головного завдання аналізує відео для оцінки параметрів руху заданого об'єкта. За цими параметрами характеризується положення об'єкта.

Модулі трекінгу у більшості випадків пов'язані з детекторами руху. Для визначення траєкторії руху об'єкта аналізується кожен кадр відео, на якому рухаються об'єкти. У кадрі може бути велика кількість рухомих об'єктів отже

необхідно не тільки будувати траєкторію об’єктів але і розрізняти самі об’єкти та їх рухи.

Відстежувати траєкторію об’єкту легше всього двома камерами. Проводиться аналіз швидкості, напрямку руху та їх розмір, і розраховується найбільш ймовірні переміщення. З цього складають траєкторію об’єкту.

Об'єкти в кадрі можуть переміщатися по-різному: вони можуть перекриватись, і із-за цього об'єкти можуть зникати та з’являтись знов та об'єднуватися або об’єкти можуть різко змінювати напрямок руху. Це значно ускладнює задачу побудови траєкторії.

Для побудов траєкторії у таких випадках двох камер недостатньо, бо має велику похибку. Для підвищення якості відстеження використовують технологію аналізу послідовності кадрів і безперервного постобработки отриманих результатів.

Для аналізу переходів об’єкту з одного стану в інший програмою виконується побудова графів. Для розуміння якому об’єкту які переміщення відповідають, аналізують положення, колір та напрям руху об’єкта. У результаті виходить набір найімовірніших рухів об’єкту, що утворює траєкторію. Різниця методів полягає в тому, що коли обробка відбувається послідовно кадр за кадром є можливість врахувати не тільки поточний стан об’єкту, а і історію його переходів. Це дозволяє значно покращити точність у складних ситуаціях перетину руху, зникнень і виникнення об'єкта.

Цей алгоритм добре працює, коли кожен об’єкт рухається окремо, але коли об’єкти рухаються у скупченнях цей алгоритм працює дуже погано.

Коли потрібен аналіз масових переміщень нерідко використовують кореляційні методи. Для реалізації цього методу оператору необхідно задати область кадру, переміщення якої буде будуватися, програма шукає цю область на наступних сценах і визначає траєкторію. Пошуковою областю може виступати як об’єкт на який зреагував детектор руху так і об’єкт виявлений класифікатором. Програма складає гистограмму квітів виділеної області,

зазначає особливі точки (характерні кути, відстані), потім відбувається їх пошук на наступних кадрах.

Із-за високої ресурсомісткості яка може бути в десятки, а в окремих випадках сотні раз більша за методах заснованих на детекторах руху не завжди цей метод можливо застосувати.

Також недоліком методу є те, що він будує траєкторії тільки заданих об’єктив. Кореляційний метод не застосуємо для сцен з високою інтенсивністю руху.

Передбачає установку синхронізованих відеокамер, які спостерігають за пов'язаними областями. Коли відстежується об’єкт і він зникає з поля зору однієї камери і переходить у поле зору іншої камери програма це фіксує та продовжує розрахунок траєкторії. Якщо не були відкалібровані камери або обладнання має різні характеристики точність цього методу дуже сильно падає. При збільшені кількості переходів між камерами імовірність помилки дуже сильно збільшується.

Заснований на роботі інтерактивного пошуку за прикметами і не вимагає установки спеціального обладнання. Для реалізації цього методу оператору необхідно задати час переходу між камерами, вибрати зразок об’єкта, що відстежується або самостійно задати параметри об’єкту. Програма знайде усі схожі об’єкти та оператору потрібно самостійно вибрати потрібній об’єкт. Далі відбувається пошук об’єкту на камерах і відбувається аналіз траєкторії. Цей аналіз подається у вигляді набору треків серед яких оператор вибирає потрібний. Цей спосіб потребує багато часу та уваги оператора, але має дуже велику точність результатів.

Багатокамерний аналіз дуже перспективна тема для вдосконалення та цікава для великої кількості розробників. Сучасні технології межкамерного стеження дозволяють відзначати на планах траєкторії переходів від однієї камери до іншої, а також будувати шляхи переміщення об'єктів на плані в рамках однієї камери - переносити переміщення об'єкта з відео на план.

Такий аналіз відео може використовуватись для таких задач як підрахунок відвідування, контроль входу в зону та перебування в ній та побудови плане теплових карт інтенсивності руху.

Перенесення траєкторії з відео на план є складною задачею з ряду причин:

– головна складність це необхідність зв'язати площині спостереження камери і плану;

– прив'язка поля зору камери до області на плані;

– відрізняти один об’єкт від іншого;

– спотворення;

Велика кількість наукових робіт присвячена завданню відстеження об’єктів. Серед методів розпізнавання розпізнають чотири основні класи: відстеження за характерними ознаками, відстеження областей, відстеження за моделлю, відстеження по активному контуру.

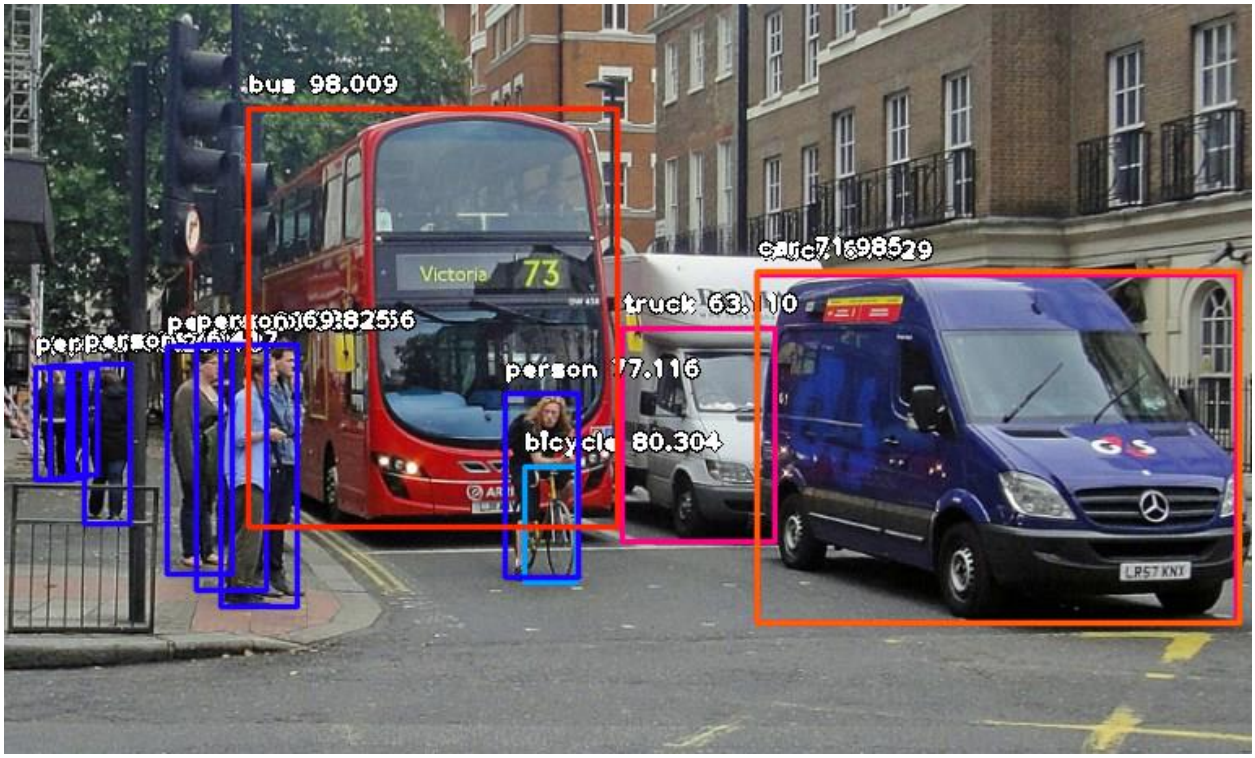

Рисунок 2.1 - Приклад відстеження багатьох об'єктів

Методи відстеження областей відстежують об'єкти по зміні областей кадру, відповідних рухомим об'єктам. При використані цього методу зазвичай задній фін динамічно оновлюється, а області руху виділяють методом віднімання фону.

Метод відстеження областей добре працює, коли аналізує сцени с невеликою кількістю рухомих об’єктів на постійному фоні. Якщо же фон динамічно змінюється і об’єкти перекривають один одного, тоді ефективність цього методу значно падає і він може використовуватись тільки як попередня обробка відео.

Цей вид відстеження характеризується як визначення об’єктів по точках покадрово. Задача відстеження точок може бути складною, через перекриття об’єкту, неправильних визначень, входження і виходу об’єкта. Цей метод має дві різні категорії: статистичний і детерміністичний. Детерміністичні методи для супроводження точок визначаються ціною асоціації кожного об’єкта на попередньому кадрі з одним об’єктом на теперішньому кадрі, використовуючи набір рухових обмежень.

Для використання статистичного методу відстеження об’єктів використовують ряд обмежень:

- незначні зміни швидкості;
- жорсткість;
- загальні рухи;
- максимальна швидкість;
- близькість.

Методи відстеження по активному контуру відстежують об'єкти шляхом подання їх обрисів у вигляді обмежуючих контурів і динамічного оновлення цих контурів на наступних кадрах. Метою цих методів є пряме вилучення форми об'єктів. Порівняно з методами відстеження областей, ці методи дають більш повний опис об’єкту.

Методи відстеження за активним контуром описують об’єкти набагато легше та ефективніше ніж методи відстеження областей, що дозволяє

зменшити обчислювальну складність. Вони добре працюють в складних умовах, коли на кадрі є спотворення та об’єкти перекривають один одного. Але проблемою цього методу є велика чутливість до точності початкового відстеження, що робить запуск автоматичної системи відстеження більш складним.

Методи відстеження за характерними ознаками виконують відстеження об'єктів шляхом вилучення характерних елементів кадру, оцінки їх кількісних параметрів і наступного порівняння з характерними ознаками інших кадрів. Методи відстеження за характерними ознаками розділяють на три види, які залежать від виду ознак, що використовуються: методи на основі локальних ознак, методи на основі глобальних ознак та методи на основі графів залежностей. Локальними ознаками вважають: точки стику контурів областей та криволінійні та криволінійні сегменти. Глобальними ознаками вважаються: розподіл кольору та яскравості на кадрі, а також центр ваги. Методи на основі графів залежностей використовують для відстеження відстані та геометричних відносин між характерними ознаками.

Методи добре підходять для відстеження декількох рухомих об’єктів у режимі реального часу.

Методи відстеження за характерними ознаками можуть правильно працювати, коли рухомі об’єкти перекривають один одного, використовуючи інформацію про рух об'єкта, локальні ознаки і графи залежностей. Але у цих методів є значні обмеження. Невисока точність відстеження на основі двомірних ознак, бо при відстеженні об’єкт змінює зовнішній вид залежно від рухів та зміни точки зору.

Методи відстеження по моделі відстежують об'єкти шляхом зіставлення ділянок зображення проекцій тривимірних моделей цих об'єктів, складених по апріорним даним. Моделі зазвичай будуються попередньо за допомогою ручних вимірювань, інструментів автоматизації проектних робіт або методів комп'ютерного зору.

Такі методи мають ряд переваг у порівняні з іншими методами відстеження.

По-перше, ці методи досить стійкі за рахунок використання апріорних знань про тривимірні контури або поверхні об'єктів. Завдяки чому досягається більша за інші методи ефективність при перекриті об’єктів і зближення кількох осередків руху на зображенні. По-друге, метод здатен отримувати природним чином тривимірне положення об’єкту, завдяки встановленню геометричної відповідності між двовимірними координатами зображення і тривимірними просторовими координатами шляхом калібрування камери. По-третє, ці методи можуть використовуватися навіть в умовах, коли об’єкт рухається швидко відносно кадрової частоти.

Головним недоліком методів є потреба будувати модель та висока обчислювальна складність.

В умовах низької точності початкового відстеження та перекриття об’єктів один одним використання методів відстеження по областям або по активному контуру значно обмежені.

Методи відстеження по моделі незважаючи на свою високу точність не можуть застосовуватись для задач у реальному часі через велику обчислювальну складність.

Будь які методи відстеження потребують механізми визначення об’єктів на кадрах, або при першому з’явленні в відеопотоці. Зазвичай використовують інформацію з одиночного кадру. Але алгоритми можуть використовувати і тимчасову інформацію, що була підрахована з послідовності кадрів.

2.2 Проблема перекриття при відстежуванні об'єктів

На проблемі перекриття об'єктів варто зупинитися докладніше. В дійсності об’єкти можуть частково перекриватись або зникати з поля зору на невизначений час, а трекер повинен продовжувати відстеження цілі в звичайному режимі.

Перекриття об’єктів можуть значно погіршити якість відстеження. Найчастіше, ситуація перекриття повинна бути визначена до того, як ціль буде точно локалізована в кадрі шляхом маскування перекритою частиною, в той час, як визначити перекриту частину цілі можна буде тільки після того, як ціль буде локалізована в кадрі. Зробимо огляд існуючих варіантів розв'язання проблеми.

У книзі Jepson D., Fleet DJ, El-Maraghi Robust Online Appearance Models for Visual Tracking пропонується алгоритм адаптивного трекінгу, що використовує для представлення мети три компонента-розподілу: стабільний S (той, який навчають весь період часу), перехідний W (за два кадри) і компонент викидів L. При цьому перекриття характеризуються компонентом викидів, які мають рівномірний розподіл. Алгоритм адаптує модель до повільних змін цілі, навчаючи таким чином стабільний компонент. Він досить надійний і використовується для оцінки руху, а перехідний - як додаткове обмеження. Обидва цих компонента мають розподіл Гаусса.

Для навчання моделі використовується EM-алгоритм, який адаптовано до використання в реальному часі. Важливим є те, що трекер може реалізовувати будь які ознаки зображення. У цій книзі пропонується застосування комплексних коефіцієнтів спеціального смугового фільтра для обробки зображень керованої піраміди (steerable pyramid). Завдяки цій моделі забезпечується стабільність алгоритму в умовах зміни просторового оточення та масштабу. Недоліки цього алгоритму:

– не враховує кольори;
– працює тільки з об’єктами, що рухаються повільно;

– має низьку ефективність, якщо об’єкт повністю перекривається та коли в оточені є схожі об’єкти.

Інший підхід визначає перекриття пікселів при перевищенні помилки деякого порога. Такі алгоритми добре працюють в умовах, коли очікувані параметри перекриття збігається за статистичним.

Але в більшості випадків такі очікування не виправдовуються тому, що в реальних ситуаціях перекриття може мати схожі з цілю ознаки або перекривати її на великий проміжок часу.

Виявлення перекриття завдяки зміні руху окремих областей

У книзі Hariharakrishnan K., Schonfeld D. Fast object tracking using adaptive block matching представляється блоковий контурний трекер. Цей алгоритм ситуації перекриття об’єктів аналізується за допомогою порівняння параметрів руху між блоками та межами зображення, які не можуть бути компенсовані рухом(наступними кадріми). Ціль представляється маскою, яка ініціалізується в кілька кроків: спочатку виконується багатозначна сегментація за чотирма напрямками, а потім - пошук області однаково рухомих сегментів (таким чином, аналізується просторово-часовий контекст об'єкта).

Алгоритм очікує, що об’єкт рухається з невеликою швидкістю, і аналізується тільки один кадр з трьох. В процесі роботи алгоритм шукає блоки зображення, що належать цілі, границям мети та задньому фону.

Далі робота йде з отриманими блоками. Рух між кадрами визначається шляхом застосування афінних перетворень до початкових блокам, якими ініціалізувалася ціль трекінгу. Відмінною особливістю алгоритму є розгляд подій перекриття і звільнення від перекриття, як взаємнозворотніх (для визначення звільнення від перекриття, поточний кадр послідовності компенсується рухом попереднього (наступного) кадру для отримання відкритих (перекритих) областей). У разі звільнення від перекриття, відкриті (перекриті) області, що володіють запропонованими характеристиками руху, ідентичні (розрізняються) з об'єктом. Алгоритм шукає такі подібності

(відмінності) руху. Якщо в якійсь області рух припинився (або відрізняється від рухом цілі), значить, ця область перекрита. Область позначається неперекритою, якщо вона відновлює рух.

Відновлення руху визначається шляхом кластеризації векторів руху областей цілі і обчисленням відстані від вектора руху розглянутої області до центра цього кластера. Якщо рух якийсь із областей відновилося, маска оновлюється.

Алгоритм контентно-адаптивного прогресивного аналізу перекриттів *(CAPOA).*

Відмінність даного алгоритму від звичайних є те, що він враховує не тільки зовнішні характеристики цілі, а поетапно аналізує ситуацію перекриття враховуючи при цьому просторово-часовий контекст. При цьому отримана інформація двічі перевіряється на відповідність цілі і обмеженням переміщення - це дозволяє краще розділяти цільовий об'єкт з об’єктами-перекриттями. Алгоритм використовує зіставлення з адаптивним шаблоном для вирішення проблеми помилкового визначення положення цілі, при перекриті. Проблема перекриття вирішена завдяки порівнянні з адаптованим шаблоном (алгоритм VMTM), де неперекритих частина мети використовується для установки цілі з помилкового положення в правильне. В результаті отримаємо трекер, що є стійким до більшості видів перекриттів.

У початковий момент часу ціль вручну виділяється визначенням області інтересу (ROI). Коли приходить новий кадр, апроксимація ROI знаходиться з першого співпадіння за зразком. Однак місце розташування цілі, визначене по першому співпадінню, може бути помилковим, так як використовує шаблон, сформований відповідно до ситуації з перекриттями на попередньому кадрі. Пошук відповідності шаблону здійснюється перетворенням системи координат для відображення припущенного шаблона на кадр і знаходження області, що найкраще узгоджується з шаблоном. При цьому перекриття вже є повністю замаскованими завдяки CAPOA.

Алгоритм CAPOA отримує на вхід чотири аргументи: ROI, ціль, попередній кадр і попередню карту викидів. На виході алгоритм видає карту викидів і оновлену маску шаблону.

Визначення перекриттів відбувається на блоках, а не на окремих пікселях, так як просторовий контекст відіграє важливу роль у визначенні перекриттів (особливо, якщо перекриття має схожість з метою).

Для знаходження компромісу між надійністю і роздільною здатністю, використовується прогресивна процедура сканування. ROI сканується багато разів, і з кожним проходом розмір блоку зменшується вдвічі.

При цьому аналізуються тільки ті блоки, в яких ситуація з перекриттями не була вирішена на попередньому кроці. Сканування закінчується, коли ситуація з перекриттями в ROI стає повністю визначеною. Для того, щоб визначити перекриття, використовуються попередні кадри – для кожного блоку шукається схожий на нього блок на попередньому кадрі.

Якщо це необхідно, пошук триває до тих пір, поки ситуація з перекриттями для даного блоку не буде точно встановлена (пошук може дійти до першого кадру, де ситуація чітко визначена для кожного блоку). Ситуація з перекриттями на даному кадрі задається бінарною картою викидів. При скануванні ROI зі старої карти викидів генерується нова, тимчасова. Якщо неможливо однозначно визначити, перекритий блок чи ні, то використовується оцінка руху для даного блоку.

Модель цілі на кожному кроці алгоритму при необхідності масштабується і адаптується за допомогою фільтра Калмана до змін у відстежуваному об'єкті. Основне завдання алгоритму VMTM полягає в підвищенні точності визначення місцеположення цілі. Це досягається пошуком локального мінімуму спеціальної функції, яка враховує попередню карту викидів. Таким чином, уникають перешкоди, пов'язані з можливими перекриттями. Даний метод дає в цілому хороші показники, однак він не справляється з повними перекриттями і об'єктами-перекриттями, ідентичними відстежуваному об'єкту.

Спільне використання трекера і детектора TLD (Tracking-Learning-Detection) демонструє хороші показники і має комерційний успіх. Розроблений для довгострокового відстеження будь-якого класу цілей. На відміну від алгоритмів, що спеціалізуються тільки на відстеженні об'єкта або тільки на покадровому пошуку об'єкта, TLD використовує обидві ідеї і скріплює їх стадією навчання. TLD складається з трекера, аналізуючого рух цілі від кадру до кадру і детектора, незалежно скануючого кожен кадр послідовності.

При цьому детектор може робити помилки двох типів: помилкові спрацьовування і помилкові відмови.

Навчальний компонент відстежує стан і трекера, і детектора і генерує навчальну вибірку для зменшення числа помилок останнього. При цьому навчальний компонент передбачає можливість одночасного виходу з ладу і трекера, і детектора. В процесі навчання детектор навчається визначати більш широкий діапазон варіацій представлення цілі і чіткіше розділяти ціль від заднього фону. Особливий інтерес представляє запропонований навчальний компонент (P-N Learning). Основна ідея полягає в використанні двох експертів: P-експерта, який визначає тільки помилкові відмови, і N-експерта - помилкові спрацьовування.

Незважаючи на те, що самі експерти допускають помилки, їх незалежне використання компенсує ці помилки. Детектор включає в себе бінарний класифікатор, скануючий вікно і модель цілі, що складається з навчальної вибірки прикладів представлення цілі. Кожен елемент вибірки обробляється незалежно від інших. При наявності N вузлів в регулярній сітці сканування виходить 2N комбінацій для даного кадру. При цьому може бути ситуація, коли одній цілі відповідає відразу кілька місць розташування, що неможливо, так як порушується зв'язність руху. В цьому випадку виникає «помилкове спрацьовування».

Саме зв'язність руху є одним з головних параметрів для визначення помилок детектора. P- експерт використовує тимчасову зв'язність і передбачає, що вона рухається по траєкторії.

Він застосовує результати роботи трекера для оцінки переміщення цілі і, якщо детектор відбраковує найбільш ймовірне (з точки зору трекера) місце розташування цілі, генерує відповідну навчальну вибірку. Вона генерується з поточної цілі за допомогою переміщення, масштабування і поворотів (близько 100 екземплярів).

N-експерт використовує просторову зв'язність мети і передбачає, що ціль перебуває тільки в єдиному місці.

Він також застосовує дані з трекера і серед результатів детектора знаходить найкращий, який оновлює стан трекера, а решту додає до вибірки, як негативні екземпляри. Як було згадано вище, в якості моделі цілі використовується колекція негативних і позитивних екземплярів - областей зображення. Подібність між двома екземплярами розраховується як нормалізована кореляція. Вона ж і виступає критерієм в класифікаторі по найближчому сусіду (Nearest Neighbor).

Саме цей класифікатор приймає рішення про оновлення моделі. Детектор використовує скануюче вікно для пошуку відповідності за шаблоном. При великій кількості шаблонів пошук може зайняти дуже багато часу.

Тому використовується каскад з трьох класифікаторів: по різниці моделей, класифікатор ансамблів і по найближчому сусіду. На практиці до останнього класифікатора доходить близько 50 екземплярів, а таку кількість вже можна швидко обробити за допомогою NN-класифікатора. В якості трекера використовується Median-Flow трекер, який є розширенням трекера Лукаса - Канада (Lucas - Kanade), що працює з оптичними потоками. Розширення включає в себе визначення факту втрати мети: в цьому випадку трекер просто не повертає розташування (обмежувальну рамку).

Факт втрати мети встановлюється, якщо різниця зміщення деякої точки цілі і медіани всіх точок перевищує заданий поріг. Це може відбуватися при швидкому переміщенні або перекритті цілі. Якщо на певному етапі ні детектор, ні трекер не видає розташування цілі, вона вважається втраченою.

В іншому випадку вибирається найкращий результат (шляхом розрахунку кореляції з шаблоном).

При такому підході детектор спирається на вже відомі шаблони, але може знайти ціль, якщо вона різко змінила розміщення або вийшла з довгого перекриття, а трекер вносить нові шаблони, оновлюючи базис детектора, і враховує просторово-часовий контекст.

При всіх своїх перевагах, метод має такі недоліки:

– погана обробка ситуації поворотів цілі;

– навчання тільки детектора (трекер залишається статичним), немає можливості відстеження декількох цілей;

– слабка підтримка рухомих каркасних об'єктів.

Найбільш цікавим і комплексним методом з розглянутих є TLD. Ідея паралельного застосування трекера, що реалізує принцип просторово часової безперервності траєкторії цілі і детектора, який здійснює пошук за шаблоном, дозволить використовувати сильні і нівелювати слабкі сторони обох підходів. Застосування простого детектора із пошуком по всьому кадру природно для вирішення проблеми повного перекриття мети, з подальшим її виникненнім а невідомих координатах.

У той же час, такий підхід не дозволяє легко адаптуватися до змін цілі і провокує велику кількість помилок (помилкових спрацьовувань і відмов).

Застосування простого трекера, що відслідковує ціль по її траєкторії і використовує інформацію з попередніх кадрів, дозволяє адаптуватися до змін зовнішнього вигляду цілі і уникнути помилок, пов'язаних з помилковими спрацьовуваннями (якщо вибрані відповідні поставленому завданню ознаки), проте є абсолютно непридатним для боротьби з повними тривалими

перекриттями. Тому, для побудови комплексного методу, що успішно вирішує проблему перекриттів, необхідно використовувати обидва підходи.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ПІДСИСТЕМИ

3.1 Інструментарій розробки

Мета роботи збільшення швидкості обробки кадрів для для інтелектуального робота в режимі реального часу та вдосконалення процесу розпізнавання об’єктів за рахунок використання операцій переробки кадрів. Для створення програми використовується середовище розробки PyCharm з використанням мови програмування Python та бібліотеки OpenCV (рис. 3.1).

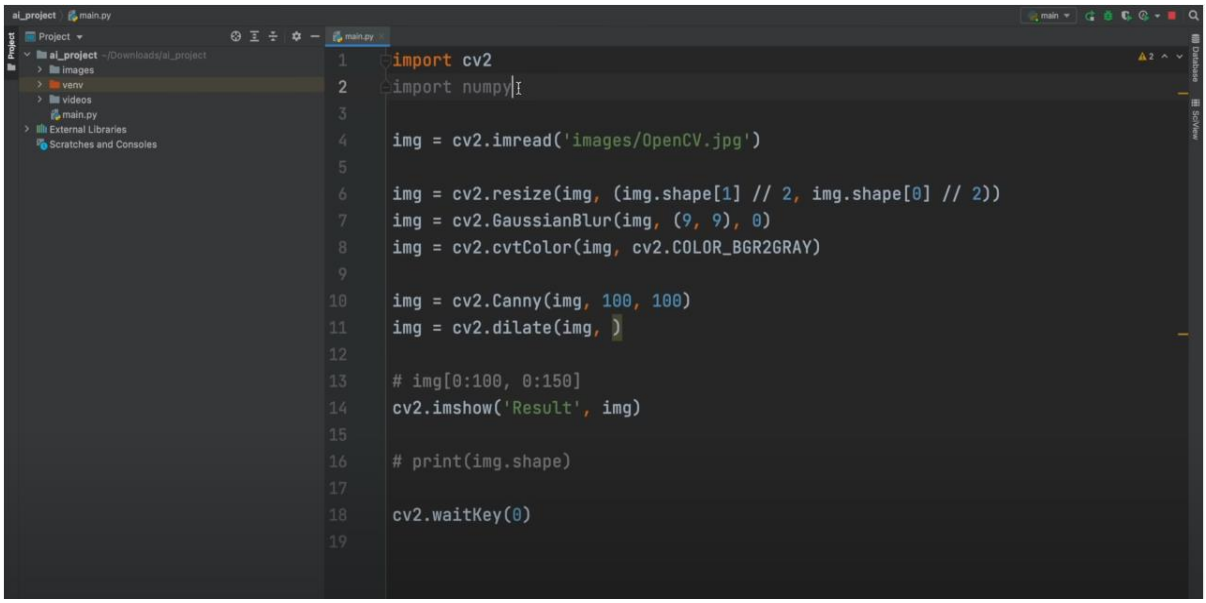

Рисунок 3.1 – Середовище розробки

Операційні засоби, які можуть використовуватись при розробці:

Windows 10 – операційна система для персональних комп’ютерів, розроблена корпорацією Microsoft в рамках сімейства Windows NT після Windows 8.1. Особливість системи у тому, що вона здатна стати єдиною для різних пристроїв, таких як консолі Xbox, персональні комп’ютери, смартфони та інше. Для Windows 10 випускаються оновлення протягом усього циклу підтримки.

За статистикою сайту ain, ця операційна система займає перше місце в світі за популярністю серед операційних систем, що використовуються.

Python – мова програмування високого рівня, яка досить проста в використані та має ефективний підхід до об’єктно-орієнтованого програмування. Це інтерпретована мова з динамічною обробкою типів та елегантним синтаксисом, що робить цю мову кращою для написання скриптів та розробки прикладних програм при невеликих затратах часу в великій кількості галузей та платформ. Python – універсальна мова, яка може бути використана для самих різних цілей таких як програмування GUI , оброблення текстів, бази даних, швидка розробка шаблонів програмування Internet і Web додатків – Webсерверів, клієнтських, серверних та серверів додотків. Можливість переносити програми на комп’ютери з різними операційними системами та іншою архітектурою, простота запису алгоритмів, можливість створити код з достатньою швидкістю виконання все це переваги мови Python. Ця мова програмування є зручною тому, що це мова високого рівня яка має підтримку модульності та структурного програмування. Python одна з популярних мов програмування завдяки своїй універсальності та гнучкості. Python – досить нова мова, створена в 1990 році. Її розробив Гвідо ван Росум у голландському інституті CWI приймаючи участь у проєкті створення мови ABC. Призначенням цієї мови було заміна мови BASIC в навчанні студентів основних концепцій програмування. Разом із розробкою основного проекту Гвідо написав інтерпретатор іншої простої мови. При розробці інтерпретатора були застосовані деякі принципи мови ABC. Мову було названо у честь англійського колективу коміків "Monty Python’s Flying Circus". Під цим іменем почалося її розповсюдження в мережі Internet.

Мова зацікавила велику кількість людей і завдяки цьому швидко розвивалася. Із невеликого інтерпретатора без об’єктноорієнтованого програмування и невеликою кількістю функцій Python розвився у велику мову програмування з великою кількістю функцій і підтримкою об’єктноорієнтованого програмування та продовжує свій розвиток і сьогодні.

Для більшості доступних платформ можливо знайти інтерпретатор Python. Усі інтерпретатори розповсюджуються безкоштовно. Python одна із найпопулярніших мов програмування у світі. Python - інтерпретована мова програмування, що створює байт-код для більш швидкої роботи.

PyCharm – середовище розробки для мови програмування Python. Середовище розробки має такі інструменти: графічний зневаджувач, аналіз коду, інструмент запуску юніт-тестів і підтримує веб-розробку на Django. PyCharm це середа розробки створена компанією JetBrains на основі IntelliJ IDEA. Ця середа розробки має версії для Linux, Windows та Mac OS. Можливості середовища розробки:

– можливість знаходження помилок у коді та синтаксисі за допомогою статичного аналізу коду;

– можливості навігації серед коду та проектів: перехід між методами, класами, файлами та відображення файлової структури проекту.

– рефакторинг: введення констант та змінних, перейменування або витяг методу, підняття ті опускання методів;

– можливість вести веб-розробку використовуючи фреймворк Django;

– вбудований зневажувач для Python;

– наявність інструментів для юніт-тестування;

– можливість використовувати Google App Engine;

– підтримка систем контролю версій: загальний користувацький інтерфейс для Mercurial, Git, Subversion, Perforce і CVS з підтримкою списків змін та злиття.

Функціональність, вартість та умови використання PyCharm відрізняються залежно від варіанту ліцензії. PyCharm Professional Edition безкоштовна, но може бути використана тільки для освітніх цілей та проектів з відкритим кодом.

OpenCV (англ. Open Source Computer Vision Library) – бібліотека, що містить алгоритми комп’ютерного зору, чисельні алгоритми загального призначення та алгоритми обробки зображень. Ця бібліотека має відкритий

код. Язик на якому реалізована бібліотека це C/C++, може використовуватись для різних язиків програмування таких, як Java, Python, Matlab, Ruby та інших мов. Бібліотека містить набір класичних та практичних алгоритмів для комп'ютерного зору і машинного навчання. Всього у бібліотеці більше 2500 оптимізованих алгоритмів. Ці алгоритми використовують для таких задач:

– обробка та аналіз зображень;
– розпізнавання обличь;
– розпізнавання жестів;
– розпізнавання та відстеження переміщень камери;
– побудова 3D моделей;
– побудова 3D хмар точок зі стерео камер;
– створення зображень з високою роздільною здатністю за допомогою склейки зображень;
– ідентифікація об'єктів;
– розробка систем для взаємодії людини з комп'ютером;
– пошук зображень, які схожі на задане, у базі даних;
– трекінг відео;
– усування ефекту червоних очей при фотозйомці зі спалахом;
– відстеження руху очей;
– сегментація зображення;
– аналіз руху;

NymPy – бібліотека, яка містить алгоритми обчислень і має відкритий код. В ній записані такі структури даних, як матриці та багатовимірні масиви. Її використовують для виконання математичних операцій над масивами. Це можуть бути як статичні та алгебраїчні обчислення так і тригонометричні. В бібліотеці міститься велика кількість функцій перетворень, алгебраїчних та математичних функцій.

3.2 Аналіз бібліотеки OpenCV

OpenCV(Onen Source Computer Vision Library) бібліотека, що безкоштовна для використання як для навчальних цілей так і комерційних. Ця бібліотека підтримує наступні мови програмування: Java, C++ та Python. Операційні системи, що підтримуються біблеотекою: iOS, Android, Mac OS, Windows та Linux. Бібліотека створена на мові програмування C/C++ завдяки чому підтримує багатопоточність. У бібліотеці використовується OpenCL, завдяки чому можливо використовувати апаратне прискорення базової неоднорідної обчислювальної платформи. OpenCV використовують понад 50 тисяч користувачів спільноти та завантажили бібліотеку понад 15 мільйонів раз. OpenCV – це бібліотека з різноманітним набором готових методів, що значно спрощує задачу написання простої програми для розпізнавання. В бібліотеці містяться готові каскади, що робить задачу знаходження деяких об’єктів легкою.

В бібліотеці містяться алгоритми для таких задач: калібровка камер, визначення схожості, усунення шумів в зображеннях, інтерпретація зображень, сегментація зображень, аналіз жестів та переміщення об’єкту.

Модулі бібліотеки діляться на основні групи:

– модуль highgui. Цей модуль реалізує базовий функціонал.(функції для обчислення, базові структури, отримання та вивід зображення);

– модулі features2d, imgproc. Вони реалізують алгоритми обробки зображення(сегментація, пошук особливих точок, фільтрація, геометричні перетворення);

– модулі objdetect, calib3d, video(пошук положення в просторі, оптичний потік, калібровка камери, аналіз руху);

– модуль ml. Призначення цього модуля реалізація машинного навчання(нейроні мережі, наївний байесівський класифікатор, метод найближчих сусідів).

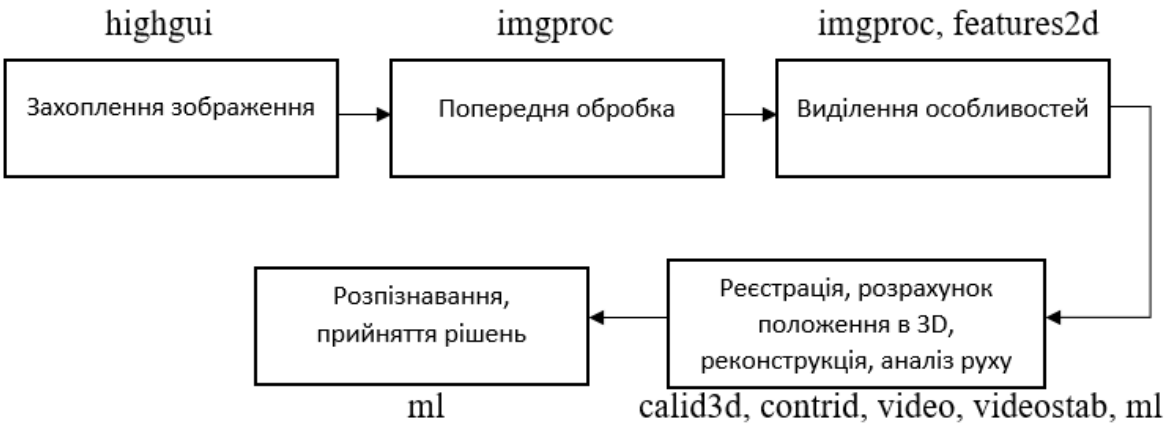

Рисунок 3.2 – Графічне представлення модулів бібліотеки openCV

Все починається з того, що модуль highgui захватує зображення. Відбувається зчитування зображення з камери або файлу.

Далі використовується модуль imgproc для здійснення попередньої обробки, це уявляє собою такі операції: вирівнювання яскравості, контрасту, видалення відблисків, тіней, усунення шуму.

При різному освітлені об’єкти виглядають по різному. При яскравому світлі красний колір може виглядати як оранжевий. А у похмуру погоду може виглядати бардовим. Для таких ситуацій використовується вирівнювання кольору. Попередня обробка використовує досить складні технології.

Після цього за допомогою модулів features2d, imgproc виділяємо особливості. При виконані завдання стеження це можуть бути пошук спеціальних точок на об’єкті, які легко відстежувати; при виконані детектування обличчя – обчислення опису кожного пікселя.

Наступним корком є за допомогою модулів features2d, imgproc провести сегментацію зображення та знаходження об’єкту, що цікавить нас. Наприклад коли об’єкт рухається, а камера ні, то можна використати алгоритм віднімання фону.

Далі при застосуванні модулів calib3d, contrib, video, stitching, videostab, ml обчислюємо розташування об’єкту в просторі 3D, проводимо аналіз та реконструкцію 3D структури, реєстрацію та інше.

Наприклад для виконання завдання склейки панорам зображень частини різних кадрів зіставляють для визначення потрібного перетворення. При відеоспостережені за цим встановлюється траєкторія об’єкта та інше.

Останнім кроком є використання модуля ml для розпізнавання та прийнятя рішення. При знаходженні тексту це визначення тексту, що собою уявляє текст та інше.

3.3 Морфологічні зміни зображення

Коли є проблема великої кількості шуму у зображені використовують морфологічні зміни зображення, завдяки цьому досягаються зміна форми ключових об’єктів.

Кроки потрібні для реалізація морфологічної операції наступні. Створюються масив однаковий по розміру з зображенням. В ньому зберігається результат морфологічної операції. Обирається вікно, розміром з структурний елемент. Воно рухається по зображені і для центру обирається максимальне чи мінімальне значення, в залежності від операції, із значень пікселів, які у межах вікна та значення записується в масив.

Розглянемо морфологічну операцію ерозія(звуження). При великій кількості випадкових вкраплень на зображені для позбавлення від них використовують операцію ерозії. Ідеєю операції є те, що при розмиті вкраплення усуваються, коли значимі регіони залишаються. При виконані цієї операції зменшуються границі форми. Розширення є операцією для потовщення. Використання операції усуває шум, дозволяє об’єднати області розділені шумом та збільшує границі для всіх форм на зображення. Виконаємо морфологічні зміни над зображенням.(рис 3.3)

Рисунок 3.3 – Оригінальне зображення

З метою видалення шуму напишемо функцію erosion()(рис. 3.4).

```
def erosian(img: np.ndarray, kernal: np.ndarray) -> np.ndarray:
    y = kernal.shape[0] // 2
    x = kernal.shape[1] // 2
    processed_image = np.copy(img)
    for i in range(y, img.shape[0] - y):
        for j in range(x, img.shape[1] - x):
            local_window = img[i-y:i+y+1, j-x:j+x+1]
            processed_image[i][j] = np.min(local_window[kernal])
    return processed_image
```

Рисунок 3.4 – Функція для виконання ерозії

Після використання функції отримаємо зображення(рис. 3.5).

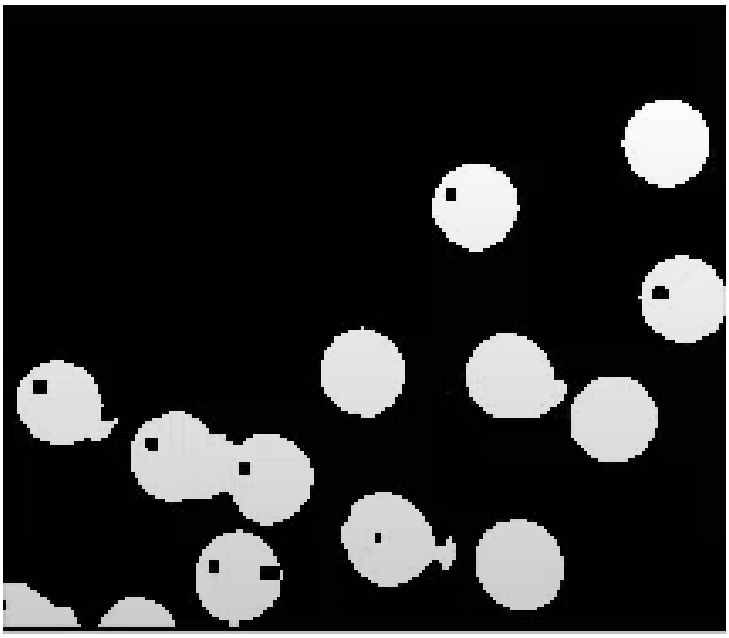

Рисунок 3.5 – Зображення після використання ерозії

При видалені з зображення елементів менших за структурні використовується операція розмикання, але ця операція нарощує контур. Для того щоб запобігти нарощення контуру використовується операція замикання. Реалізуємо операцію у вигляді функції dilatation()(рис. 3.6).

```
def dilate(img: np.ndarray, kernal: np.ndarray) -> np.ndarray:
    y = kernal.shape[0] // 2
    x = kernal.shape[1] // 2
    processed_image = np.copy(img)
    for i in range(y, img.shape[0] - y):
        for j in range(x, img.shape[1] - x):
            local_window = img[i-y:i+y+1, j-x:j+x+1]
            processed_image[i][j] = np.max(local_window[kernal])
    return processed_image
```

Рисунок 3.6 – Функція для виконання розширення

Після використання отримаємо таке зображення(рис. 3.7).

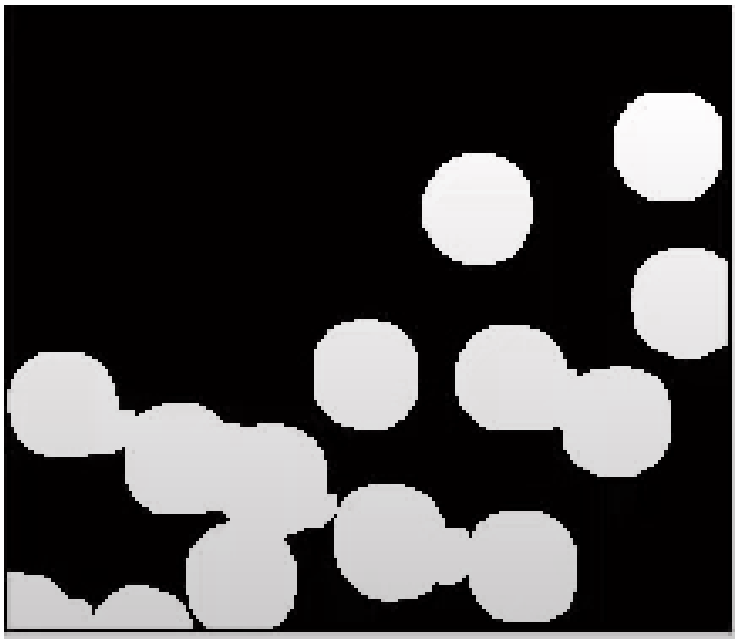

Рисунок 3.7 – Зображення після використання розширення

3.4 Модифікація алгоритму розпізнавання об'єктів

Важливою задачею аналізу зображення є автоматичне виділення рухомих об’єктів в відеопотоці. Є багато методів вирішення цієї задачі для нерухомої камери.

У роботі представлено метод мета якого вдосконалення процесу розпізнавання. Задача методу виділення об’єкту на зображені який належить до одного з 20 представлених класів.

```
# Labels of network.
classNames = {0: 'background'
      1: 'aeroplane', 2: 'bicycle', 3: 'bird', 4: 'boat',
      5: 'bottle', 6: 'bus', 7: 'car', 8: 'cat', 9: 'chair',
      10: 'cow', 11: 'diningtable', 12: 'dog', 13: 'horse',
      14: 'motorbike', 15: 'person', 16: 'pottedplant',
      17: 'speep', 18: 'sofa', 19: 'train', 20: 'tvmonitor'}
```

Рисунок 3.8 – Класи для ідентифікації об’єктів

Найбільшою проблемою систем стеження у реальному часі є великий обсяг даних, які треба проаналізувати, що треба обробити у невеликий проміжок часу та прийняти рішення. Спочатку об’єкт виявляється у реальному часі наступним кроком є розпізнавання. При розпізнавані об’єкту будемо використовувати алгоритм SSD(Single Shot MultiBox Detector).

Спочатку за допомогою модуля highgui захоплюємо зображення. Це зображення вводиться з відео або файлу. Потім використовуючи модуль imgproc виконуємо попередню обробку для видалення бліків та тіней, вирівнювання кольорів та усунення шумів. Наприклад об’єкти при різних вилах освітлення мають різний вигляд.

Функція fastNLMeansDenoisingColored() використовується для фільтрації шуму.

```
# прибираємо шум та зберігаємо в dst
dst = cv2.fastNlMeansDenoisingColored(frame, None, 10, 10, 7, 21)
```

Рисунок 3.9 – Фрагмент коду для зменшення шуму зображення

Для кольорових зображень використовують функцію fastNLMeansDenoisingColored(). Аргументи функції:

– h: параметр для визначення потужності фільтра. Чим більше це значення тим більше видаляється шуму але видаляються і деталі зображення.

– hForColorComponents параметр аналогічний h але призначений для використання на кольорових зображеннях.

– templateWindowSize: розмір вікна шаблону(має бути непарним числом)

– searchWindowSize: розмір вікна пошуку(має бути непарним числом)

Гамма-корекція популярний метод обробки зображень та використовується в більшості програм цієї області. Після гама-корекції можливо виявить об’єкти, що не було видно в темних або світлих областях. Реалізуємо функцію для гама-корекції для выдеопотоку:

```
def adjust_gamma(image, gamma=1.0):

    invGamma = 1.0 / gamma
    table = np.array([((i / 255.0) ** invGamma) * 255
        for i in np.atange(0, 256)].astype("unit8")

    return cv2.LUT(image, table)
```

Рисунок 3.10 – Фрагмент коду для гама-корекції кадру

Наступний крок є виконання морфологічної операції з метою підвищення точності визначення об’єктів та зменшення кількості шуму в зображенні. Виконуємо операції розмикання та замикання послідовно. При виконані операції розмикання видаляються елементи, що менші за структурний елемент, але виконання цієї операції викликає нарощення контуру. Для того щоб виправити це використовують операцію замикання.

Після виконанні зазначених операцій та методів виконується процес визначення об’єкту.

Умова використання алгоритму є нерухома камера. Алгоритм розроблено з цілю, щоб він міг проводити обробку зображень з частотою камери.

Розмір об’єкту повинен становити більше 10 пікселів як в висоту та і в ширину. Рухатись об’єкт може як періодично так і поступально, як по кривій так і по прямій траєкторії. Обмеження руху об’єкта не більше 100 пікселів за сцену.

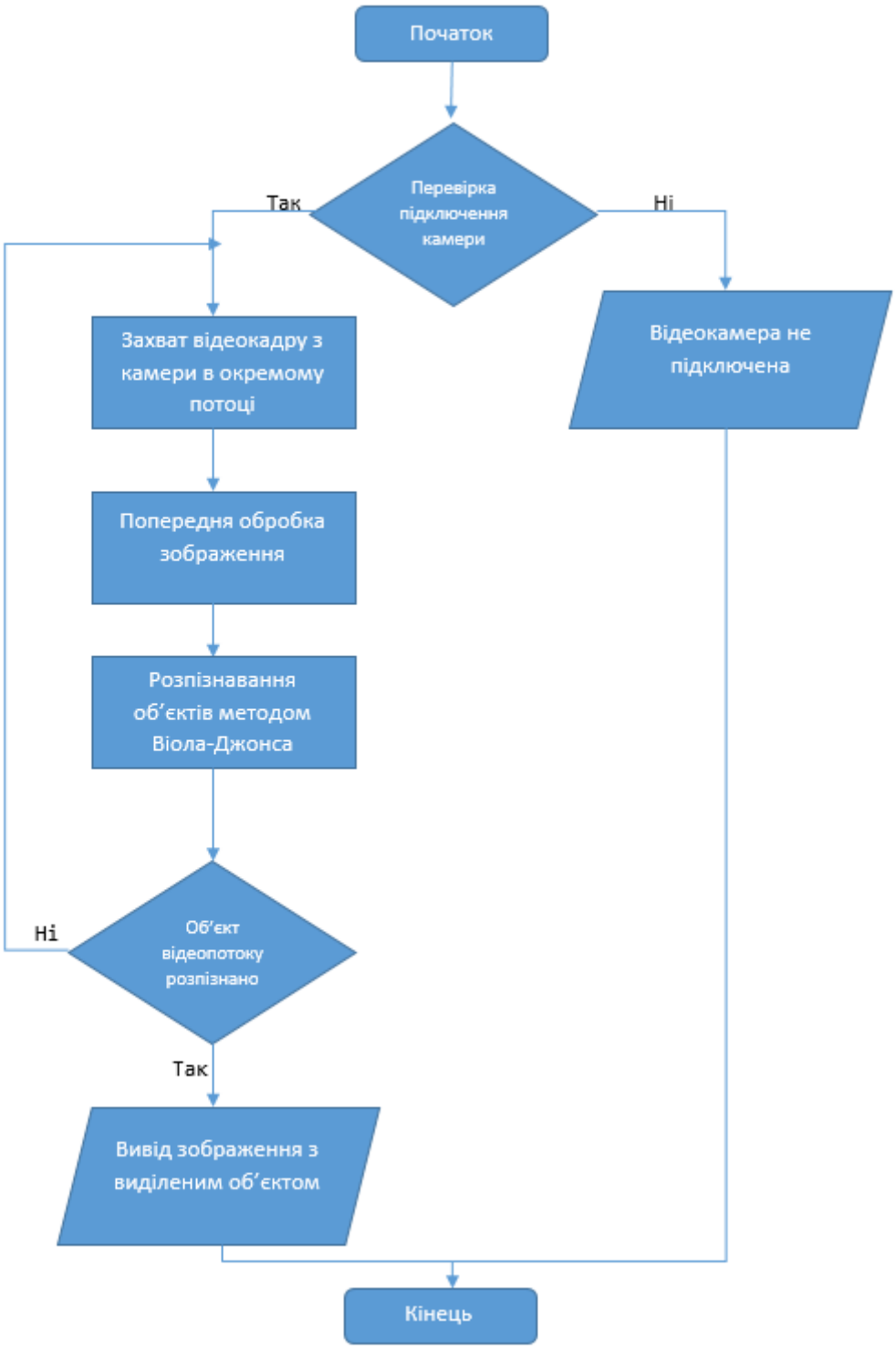

Рисунок 3.11 – Узагальнена схема роботи підсистеми

3.5 Підвищення якості відеотрекінгу за рахунок використання багатопоточності

Кадрова частота – це частота (швидкість), з якою пристрій формування зображення відображає послідовні зображення, що називаються кадрами.

Цей термін може використовуватись для камер, систем спостереження та комп’ютерної графіки. Цей термін був створений фотографом Едвардом Майбріджем, коли він експериментував з зйомками послідовно декількома фотокамерами рухомих об’єктів. Задається в кадрах в секунду(fps).

Багатопоточність це можливість працювати з кількома потоками паралельно або одночасно на багатопроцесорних системах. Використання багатопоточності призводить до збільшення швидкості програм за рахунок ефективного використання ресурсів комп’ютера у багатьох випадках.

Цілю використання багатопоточності є досягнення квазі-багатозадачності при виконані процесу, тобто всі потоки працюють на виконанням одного процесу. Всі потоки мають спільний адресний простір та дескриптори файлів. Кожен процес використовує що найменш один поток.

Багатопоточність це не багато-процесорність і не багатозадачність. Операційні системи як правило підтримують як багатопоточність так і багатозадачність.

Переваги багатопоточності:

– завдяки використанню загального адресного простору можливо значно спростити програми в багатьох випадках;

– збільшення швидкості роботи процесу, який використовує багатопоточність за рахунок розпоралелювання обчислень;

Створимо додатковий потік для зчитування зображення з камери з метою збільшення значення fps. У цей час головний потік обробляє поточний кадр.

Використовуючи функції OpenCV реалізуємо окремий потік для процесу вводу. Це збільшить fps програми, бо процес вводу/виводу в більшості випадків повільний. Програми обробки відео та комп’ютерного зору(особливо призначених для роботи в реальному часі) створюють велике навантаження на процесор. Процес вводу/виводу також можуть викликати велику навантаження на процесор.

Оскільки зчитування зображення з камери перенесено в інший потік, головний потік безперервно обробляє поточний кадр.

Спочатку створимо клас FPS для вимірювання кадрової частоти програми. Клас потрібен для перевірки впливу потоків на fps.

```
import datetime

class FPS:
      def  __init__(self):
            # зберігаємо час початку роботи, кількість кадрів, час закінчення
            self._start = None
            self._end = None
            self._numFrame = 0

      def start(self):
            # час початку
            self._start = datetime. datetime.now()
            return self

      def start(self):
            # час зікінчення
            self._end = datetime. datetime.now()

      def update(self)
            # інкрементатор для кожгої ітерації
            self._numFrames += 1

      def elapsed(self)
            # повертає час роботи програми
            return (self._end - self._start).total_seconds()

      def fps(self)
            # повертає кількість кадрів в секунду
            return self._numFrames / self.elapsed()
```

Рисунок 3.12 – Фрагмент коду для класа FPS

Для доступу к камері визначимо клас CamVideoStream. Для порівняння fps варіантів з багатопоточністю та без неї визначимо сценарій fps_demo.py.

Таблиця 3.1 Порівняння швидкості роботи системи з використанням багатопоточності та без багатопоточності

№ Випробування	fps	
	Однопоточна обробка кадрів	Багатопоточна обробка кадрів
1	24,45	72,03
2	24,47	71,94
3	24,46	72,09

```
from threading import Thread
import cv2

class CamVideoStream:
	def __init__(self, src=0):
		# ініціалізіруємо потік для зчитування кадрів з камери
		self.stream = cv2.VideoCapture(src)
		(self.grabbed, self.frame) = self.stream.read()

		# ініціалізіруємо змінну, що використовується для зупинки потоку
		self.stopped = False

		def start(self):
			# початок потоку, що зчитує відеопоток
			Thread(target= self.update, args=()).start()
			return self

		def update(self):
			# повторюємо цикл доки поток не завершиться
			while True:
				# якщо індикатор True завершуємо потік
				if self.stopped:
					return
				# якщо індикатор False продовжуємо зчитувати потік
				(self.grabbed, self.frame) = self.stream.read()

		def read(self):
			#повертаэмо зчитаний кадр
			return self.frame

		def stop(self):
			#індикатор того, що потік треба зупинити
			self.stopped = True
```

Рисунок 3.13 – Коду класу CamVideoStream

У результаті отримаємо, що впровадження багатопоточності збільшує fps приблизно в 3 рази.

1.6 Тестування системи

Протестуємо точність розпізнавання програмою рухомих об’єктів. Визначимо кількість шуму та скільки об’єктів буде визначено правильно. Програма не може ефективно працювати коли відбуваються зміни у фоні, якщо на відео іде сніг або коливаються дерева вони будуть розпізнаватись як об’єкт переднього плану. Зображення з поганою якістю викликають помилкові визначення об’єктів, що робить програму для таких зображень неефективною. Проаналізуємо результат роботи програми.

У відеопотоці розпізнається чотири об’єкта в режимі реального часу.

Рисунок 3.14 – Зображення роботи програми

У результаті програма виводить відео з розпізнаними об’єктами. Об’єкти, які були розпізнані програмою відмічені зеленим прямокутником, назвою та точністю розпізнавання.

У даному випадку програма відстежує рух чотирьох об’єктів, які рухаються по кривій траєкторії. Програма відстежує зміну положення об’єктів на кожному кадрі. Спочатку були правельно розпізнані чотири об’єкти з точністю 0,4012%, 0,6489%, 0,8771% та 0,8382%. На 12 секунді програма з високою точністю розпізнає ці об’єкти. Програма розпізнала з такою точністю об’єкти на 1,4,8,12 секунді(табл. 3.2).

Таблиця 3.2 – Точність розпізнавання об'єктів через рівні проміжки часу

Час, сек	Точність розпізнавання, %			
	Об'єкт 1	Об'єкт 2	Об'єкт 3	Об'єкт 4
1	40,12	64,89	87,71	83,82
4	82,69	90,43	46,41	75,85
8	86,94	92,17	67,28	91,90
12	78,16	95,95	75,30	96,65

Програма відповідає вимогам: всі об’єкти знайдені та правильно ідентифіковані. При приближені об’єкта точність його ідентифікації значно зростає. Наявність часткового перекриття об’єктів та шумів камери не завадили відстеженню об’єктів, що задовольняє очікування.

Розглянемо лінійний рух декількох об’єктів. В цьому прикладі відео містить сильний шум, що сильно впливає на точність розпізнавання. На першій секунді програма змогла встановити лише два з чотирьох об’єктів оскільки інші об’єкти перекриті більш ніж на половину. На 12 секунді було

визначено 3 з 4 об’єктів незважаючи на великий рівень шуму, рис. 3.15.

Рисунок 3.15 – Результат роботи програми

Точність розпізнавання об’єктів у прикладі табл. 3.3.

Таблиця 3.3 – Точність розпізнавання об’єктів через рівні проміжки часу

Час, сек	Точність розпізнавання, %			
	Об'єкт 1	Об'єкт 2	Об'єкт 3	Об'єкт 4
1	74,34	97,82	Не розпізнано	Не розпізнано
4	90,02	86,09	30,92	Не розпізнано
8	58,96	94,01	Не розпізнано	52,24
12	58,96	94,01	Не розпізнано	52,24

У результаті отримано задовільний результат, бо всі об’єкти, що були визначені, були правильно ідентифіковані. Як і в попередньому прикладі при наближені об’єкту точність його розпізнання підвищується. Через перекриття об’єктом № 1 об’єкт № 3 було розпізнано тільки на 4 секунді. Через великий рівень шуму об’єкт № 4 було розпізнано тільки на 8 та 12 секунді. Отже, усі

об’єкти були ідентифіковані правильно, незважаючи на перекриття та сильний шум.

Об’єкти рухаються повільно та послідовно. На відео змінюється інтенсивність освітлення. Проведемо аналіз розпізнавання та точності об’єктів у таких умовах.

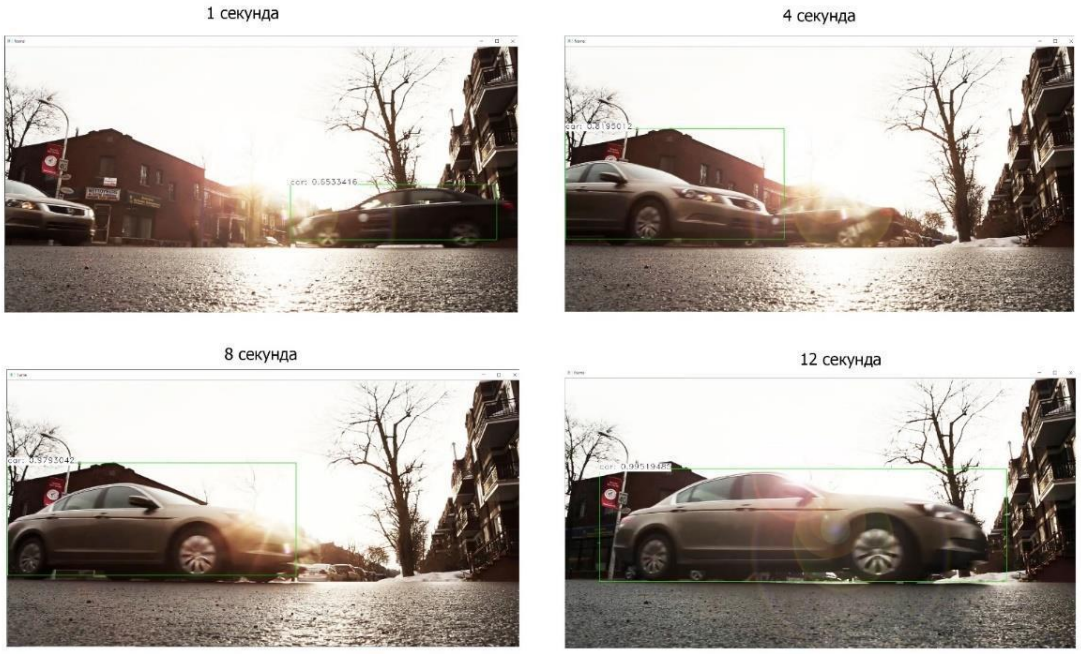

Рисунок 3.16 – Результат роботи програми

Програма розпізнала обидва об’єкти та відслідила зміну їх положення. Спочатку було розпізнано об’єкт № 1 з точністю 0.6533, коли об’єкт № 2 здебільше знаходився поза кадру. На 4, 8 та 12 секунді було розпізнано об’єкт № 2, але об’єкт № 1 не було розпізнано через перекриття об’єктом № 2.

Таблиця 3.4 – Точність розпізнавання через рівні проміжки часу

Час, сек	Точність розпізнавання, %	
	Об'єкт 1	Об'єкт 2
1	65,33	Не розпізнаний

Продовження таблиці 3.4

4	Не розпізнан	81,95
8	Не розпізнан	97,93
12	Не розпізнан	99,51

Після розробки модифікованого методу розпізнавання порівняємо звичайний метод та модифікований. Розрахуємо середній відсоток розпізнання об’єктів в різних умовах та порівняємо. Ці умови: ідеальних умовах розпізнавання, при частковому перекритті, зашумленій відеопослідовності та при значній зміні інтенсивності освітлення.

Таблиця 3.5 Порівняння існуючого алгоритму розпізнавання та алгоритму з попередньою обробкою кадрів

Умови тестування	Точність розпізнавання, %		
	Вихідний метод розпізнавання об'єктів	Модифікований метод розпізнавання об'єктів	Відсоток покращення
Ідеальні умови розпізнавання	92,14	92,14	Не змінився
Сильний шум	82,63	90,00	7.37
Зміна інтенисвності освітлення	55,12	63,45	8.33

В результаті можливо зробити висновок, що застосування методу комбінації морфологічних операцій, медіанної фільтрації та використання сегментації підвищує відсоток розпізнавання об’єктів у умовах сильного шуму чи зміни інтенсивності освітлення. Отже модифікований метод підвищує якість розпізнавання.

ВИСНОВКИ

У даній кваліфікаційній роботі був проведений аналіз методів розпізнавання та відстеження об’єктів, так же були розглянуті різні способи покращення розпізнавання. У результаті роботи було створено модифікований метод розпізнання об’єктів, який показує більш високу якість розпізнавання при неякісних вхідних даних. З чого можна зробити такі висновки:

– проведено аналіз різних методів розпізнавання та відстеження об’єктів;

– модифіковано алгоритм розпізнавання об’єктів у відео, за рахунок використання комбінованого методу обробки зображення, що використовує морфологічні операції, гама-корекцію, фільтрацію шуму, а також багатопоточності;

– розроблено програмне забезпечення на мові програмування Python для ідентифікації об’єктів;

– програмне забезпечення протестоване на різних неякісних вхідних даних.

ПЕРЕЛІК ПОСИЛАНЬ

1. OpenCV library. URL: https://opencv.org/ (дата звернення 24.05.2022).
2. NumPy. URL: http://www.numpy.org/ (дата звернення 25.05.2022).
3. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs / [L.-C. Chen, G. Papandreou, I. Kokkinos, et al.] URL: https://arxiv.org/pdf/1606.00915.pdf (дата звернення 26.05.2022).
4. Вьюгин В.В. Математические основы теории машинного обучения и прогнозирования / В.В. Вьюгин. – М.: МЦНМО, 2013. 387 с.
5. Лукьяница А. А. Обработка видеоизображений / А.А.Лукьяница, А.Г.Шишкин; Под ред. Поздняков С.А. — М.: Ай-Эс-Эс Пресс, 2009. 764 с.
6. Алпатов Б.А., Бабаян П.В., Балашов О.Е., Степашкин А.И. Методы автоматического обнаружения и сопровождения объектов. Обработка изображений и управление. М., Радиотехника, 2008. 176 с.
7. Гансалес Р., Вудс Р. Цифровая обработка изображений. М., Техносфера, 2005.1012 с.
8. Сойфер В. А. Методы компьютерной обработки изображений. М.,ФИЗМАТЛИТ, 2003. 784 с.
9. Кобилін, О. А., & Творошенко, І. С. (2021). Методи цифрової обробки зображень: навч. посібник. *Харків: ХНУРЕ*.
10. Путятін, Є. П., Гороховатський, В. О., & Матат, О. О. (2006). Методи та алгоритми комп'ютерного зору: навч. посібник.
11. Shi J., Tomasi C. Good features to track. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Seattle, USA, 1994, pp. 593–600.
12. Comaniciu D., Ramesh V., Meer P. Kernel-based object tracking. IEEE Transaction on Pattern Analysis and Machine Intelligence, 2003, pp. 564–577.

13. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs / [L.-C. Chen, G. Papandreou, I. Kokkinos, et al.]. URL: https://arxiv.org/pdf/1606.00915.pdf(дата звернення 27.05.2022).
14. Isard M., Blake A. Condensation — conditional density propagation for visual tracking. International Journal of Computer Vision, 1998, pp. 5–28.
15. Freeman W.T., Roth M. Orientation histograms for hand gesture recognition. Proceedings of the Workshop on Automatic Face and Gesture Recognition. Zurich, Switzerland, 1995, pp. 296–301.
16. Kwon J., Lee K.M. Visual Tracking Decomposition. Conference on Computer Vision and Pattern Recognition, 2010.
17. Babenko B., Yang M.-H., Belongie S. Visual Tracking with Online Multiple Instance Learning. Conference on Computer Vision and Pattern Recognition, 2009.
18. Lowe G. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 2004, vol. 60, no. 2, pp. 91–110.
19. Viola P., Jones M. Rapid object detection using a boosted cascade of simplefeatures. Conference on Computer Vision and Pattern Recognition, 2001.
20. Апальков И.В., Хрящев В.В. Удаление шума из изображений на основе нелинейных алгоритмов с использованием ранговой статистики.
21. Ярославский государственный университет имени П.Г. Демидова, 2007.
22. Automated Tests for Errors in Computer System ‘Environment’/ Oleksii Vasilenko, Oleksandr Kuzomin. // International Journal"Information Models and Analyses" Volume 7, Number 4. PP.2019. 373 – 384.
23. Intellectual Models and Means of the Biometric System Dynamics of Rinosinusite / Oleksii Vasilenko, Oleksandr Kuzomin, Tatyana Khripushina // International Journal "Information Models and Analyses" Volume 7, Number 4. PP.2019. 350 – 361.
24. Research of the intellectual system of knowledge search in Databases / Oleksii Vasilenko, Oleksandr Kuzomin, Bohdan Maliar //International Journal

"Information Models and Analyses" Volume 7, Number 4. PP.2019. 327 – 338.

25. Using of ontologies for building databases and knowledge bases for consequences management of emergency/Kuzomin, O., Dudka, O., Vasylenko, O., Radchenko, V., Lyashenko, V.//International Journal of Advanced Trends in Computer Science and Engineering, 2020, 9(4), pp. 5040-5045