НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ННК "Інститут прикладного системного аналізу"
(повна назва інституту/факультету)

Кафедра Системного проектування
(повна назва кафедри)

«До захисту допущено»

Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)

"___"_____2016 р.

Дипломна робота

___першого (бакалаврського)_____**рівня вищої освіти**
(першого (бакалаврського), другого (магістерського))

зі спеціальності7.05010102, 8.05010102 Інформаційні технології проектування
_____7.05010103, 8.05010103 Системне проектування
(код та назва спеціальності)

на тему: Програма виділення та супроводження рухомих об'єктів у відео послідовності_____
_____

Виконав: студент 4 курсу, групи ДА-21
(шифр групи)

_____Аззуз Іскандар Джабра_____
(прізвище, ім'я, по батькові) (підпис)

Керівник_____ст. викладач Бритов О.А._____ ____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Нормоконтроль _____ст. викладач Бритов О.А._____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2016 року

Національний технічний університет України
«Київський політехнічний інститут»

Факультет (інститут) ННК "Інститут прикладного системного аналізу"
(повна назва)

Кафедра Системного проектування
(повна назва)

Рівень вищої освіти Перший(Бакалаврський) (перший (бакалаврський), другий (магістерський) або спеціаліста)

Спеціальність 7.05010102, 8.05010102 Інформаційні технології проектування
7.05010103, 8.05010103 Системне проектування
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)

«___»_____2016 р.

ЗАВДАННЯ на дипломний проект (роботу) студенту
Аззузу Іскандару Джабра
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи)) Програма виділення та супроводження рухомих об'єктів у відео послідовності

керівник проекту (роботи)) ст. викладач Британ О.А
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «12» травня 2016 р. № 50-ст

2. Строк подання студентом проекту (роботи) 08.06.2016

3. Вихідні дані до проекту (роботи) _____
Технології визначення та відстеження об'єктів на кольоровому відео.
Частота кадрів не більше 60 Гц. Нерухома камера.
Швидкісь руху об'єкту не більше 100 пікселів на кадр.
Характер руху: 1) поступальний по прямолінійній або криволінійній траєкторії; 2) періодичний (коливальний або по замкнутій траєкторії).
Мова програмування С++.
Передбачення положення об'єкту за допомогою фільтра Калмана.

4. Зміст розрахунково-пояснювальної записки (перелік завдань, які потрібно розробити)
    1. Проаналізувати існуючі рішення для визначення та супроводження об’єктів .
    2. Порівняти платформи розробки програми.
    3. Розробити алгоритм роботи та структуру програми.
    4. Реалізувати програму супроводження об'єктів в відео послідовності.
    5. Протестувати програму.
5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслеників, плакатів тощо)
    1. Блок-схема алгоритму роботи – плакат.
    2. Порівняння технологій відстеження об’єктів – плакат.
    3. Результати тестування програми – плакат.

Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Семенченко Н.В., професор		

6. Дата видачі завдання 01.02.2016

Календарний план

№ з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Отримання завдання	01.02.2016	
2	Збір інформації	15.02.2016	
3	Аналіз існуючих рішень	28.02.2016	
4	Вибір інструментів розробки	10.03.2016	
5	Розробка алгоритму рішенння задачі	15.03.2016	
6	Розробка програми супроводження об’єктів у відео послідовності	25.03.2016	
7	Тестування програми	25.04.2016	
8	Оформлення дипломної роботи	31.05.2016	
9	Отримання допуску до захисту та подача роботи в ДЕК	08.06.2016	

Студент _____ І.Д.Аззуз
(підпис) (ініціали, прізвище)

Керівник проекту (роботи) _____ О.А.Бритов
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

бакалаврської дипломної роботи Аззуза Іскандара Джабра
на тему «Програма виділення та супроводження рухомих об’єктів у відео послідовності»

Дана дипломна робота присвячена розробці програми визначення об’єктів та їх відстеження у відео послідовності. Приведено огляд алгоритмів та технологій для розпізнавання об’єктів і їх супроводження.

Досліджено сучасні методи та засоби для обробки зображення. Розглянуто готові реалізації та функції засобів в області комп’ютерного бачення.

Розроблено алгоритм та архітектуру програми, наведено основні переваги і недоліки. Реалізовано програму відстеження та супроводження об’єктів у відео послідовності. Проведено тестування для детальної демонстрації роботи програми, а також надані рекомендації по покращенню алгоритму та програми в цілому.

Загальний обсяг роботи: 75 сторінок, 35 рисунків, 6 таблиць, 11 посилань.

Ключові слова: комп’ютерний зір, OpenCV, модель заднього фону, детектор Кані, фільтр Калмана.

АННОТАЦИЯ

бакалаврской дипломной работы Аззуза Іскандара Джабра
на тему «Программа выделения и сопровождения движущихся объектов на видео последовательности»

Данная дипломная работа посвящена разработке программы определения объектов и их отслеживание на видео последовательности. Приведен обзор алгоритмов и технологий для распознавания объектов и их сопровождения.

Исследовано современные методы и средства для обработки изображения. Рассмотрено готовые реализации и функции средств в области компьютерного зрения.

Разработан алгоритм и архитектура программы, определены основные преимущества и недостатки. Реализована программа отслеживания и сопровождения объектов на видео последовательности. Проведены тестирования для детальной демонстрации работы программы, а также даны рекомендации по улучшению алгоритма и программы в целом.

Общий объем работы: 75 страниц, 35 рисунков, 6 таблиц, 11 ссылок.

Ключевые слова: компьютерное зрение, OpenCV, модель заднего фона, детектор Канні, фильтр Калмана.

ANNOTATION

of a bachelor’s degree work by Azzuz Iskandar
entitled «Software for detecting and tracking objects on video»

This bachelor's degree work is devoted to developing a programme of detecting and tracking objects.

This paper researched modern methods and tools for image processing. Work reviews the features and functionalities in the realm of computer vision.

The paper develops an algorithm and architecture of the programme and provides the main advantages and disadvantages. An application of tracking and accompanying the objects was released. Also, the paper conducts testing for a detailed demonstration of the programme performance. In addition, this work provides the recommendations for improving the algorithms and the programme.

The total amount of work: 75 pages, 35 figures, 6 tables, 11 references.

Keywords:computer vision, OpenCV, background substraction, Canny detector, Kalman`s filter.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

OpenCV – бібліотека комп’ютерного зору з відкритим кодом.

Tracking – відстеження об’єкта на відео послідовності.

Оклюзія – ситуація, в якій два об’єкти розташовані приблизно на одній лінії і один об’єкт, розташований ближче до віртуальної камери або вікн а перегляду, частково або повністю закриває видимість іншого об'єкта.

Скріншот – зображення, отримане комп'ютером, що відображає те, що бачить користувач на екрані монітора.

ВСТУП

Визначення об’єктів, що рухаються є важливою задачею у сфері комп’ютерного зору. Комп'ютерний зір включає в себе отримання цифрового зображення, його обробку, аналіз і розуміння зображень, використовуючи статистичні методи і моделі побудовані за допомогою фізики, геометрії, статистики і теорії статистичного навчання. Поширення потужних комп’ютерів, доступність високоякісних камер за невеликою ціною та збільшення потреб для автоматизованого аналізу відео зумовило великий інтерес в області алгоритмів супроводження руху. Є три основні ключові кроки в аналізі відео: розпізнавання об’єкта, що рухається, відстеження об’єкта кадр за кадром, і аналіз об’єктів для визначення їх поведінки. Таким чином, використання алгоритму стеження за об’єктом є доречним в таких задачах:

- Розпізнавання на основі руху, наприклад, рух людини за ходою, автоматичне розпізнавання об’єктів.
- Автоматизоване спостереження, що виявляє підозрілу активність.
- Взаємодія людина-комп’ютер, наприклад, розпізнавання жестів, слідкування за поглядом для вводу данник тощо.
- Навігація машин, що пов’язана з планування маршруту та уникання перешкод.

В найпростішій формі відстеження може бути визначений як проблема оцінки траєкторії об’єкта в площині зображення, що рухається навколо сцени. Іншими словами, відстежувач помічає постійним ярликом об’єкт, що рухається. Залежно від області, в якій використовується, вістежувач може надавати додаткову інформацію щодо об’єкта: орієнтацію, площу, розмір, форму. Відстежування об’єктів є складною задачою через такі фактори:

- Нестача інформаціх 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.

Мета роботи - реалізувати програму визначення та відстеження об’єктів. Для цього потрібно розглянути та проаналізувати існуючі засоби та рішеня в області визначення та відстеження об’єктів. Надати порівняння платформ, які мають готовий функціонал для обробки зображення. Розробити власний алгоритм та архітектуру програми.

1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

1.1 Особливості і проблематика

Основні проблеми, які виникають при створенні та роботі відстежувача, пов’язані з виглядом об’єкта та схожими іншими об’єктами на сцені.

Вигляд інших об'єктів може накладатимя на задній фон. В таких випадках, важко знайти на зображенні об’єкт, що як ми очікуємо, буде рухатися. На рисунку 1.1 зображено невизначеність кольору, що приводить до неправильної роботи відстежувача .[1]

Рисунок 1.1 – Помилкове визначення об’єктів.

Також важко визначити об'єкт, вигляд якого змінився в площині об’єктива через такі фактори

- Зміна позиції, об'єкт що рухається змінює свій вигляд на площині зображення, наприклад якщо крутиться
- Зміна освітлення, напрямок, інтенсивність, колір впливає на вигляд об'єкта. Так, зміна світла в глобальному плані викликає проблеми у сусідніх сценах. наприклад, коли хмари закривають сонце змінюється навколишнє світло, а також кут між нормалями до поверхні об’єкта і напрямком світла. Це впливає на те, як ми бачимо зображення через лінзу камери.
- Шум. Процес отримання зображення пов'язаний з певною долею шуму, який залежить від якості матриці камери.

- Перекриття. Спостерігати за ціллю, коли вона частково або повністю перекрита іншим об’єктом на зображенні. Перекриття виникають коли:
    - Ціль рухається за нерухомими об’єктами, наприклад, колона чи стіл.
    - Інші об’єкти, що рухаються, затуляють вид на ціль.

1.2 Способи зображення об’єктів

В задачі спостереження об’єкт може бути визначений як будь-що важливе для подальшого аналізу. Наприклад, люди, машини на дорозі, риби всередині акваріуму. Об’єкти представлені їх формами та виглядом. Способи представлення, що широко використовуються для відстеження, зображено на рисунку 1.2 [3]:

- Точки. Об’єкти представляють собою точки, або набір точок (Рисунок 1.1(b)). Підходить для стеження за невеликими регіонами на зображенні.
- Примітивні геометричні форми. Форма об’єкта представлена прямокутником, еліпсом тощо (Рисунок 1.1(c,d)). Рух об’єкта моделюється як переміщення, афінні чи проективні перетворення. Підходить для стеження за твердими об’єктами.
- Силует об’єкта та контур. Контур представляє собою границі об’єкта (Рисунок 1.1(g, h)). Силует знаходиться всередині контуру (Рисунок 1.1(i)). Підходить для стеження за нетвердими об’єктами.
- З’єднані між собою форми. З’єднані об’єкти складаються з частин тіла, що тримаються разом за рахунок суглобів. Наприклад, тіло людини - це з’єднаний об’єкт, що складається з торсу, ніг, рук, голови та суглобів (Рисунок 1.1(e)). Відношення між частинами регулюється моделями кінематичного руху.
- Скелетна модель. Ця модель часто використовується як форма для

розпізнавання об’єктів.

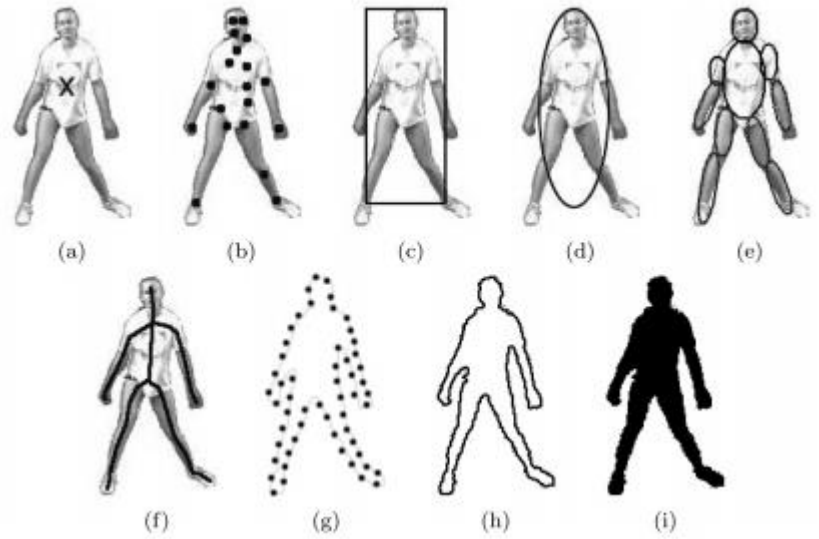

Рисунок 1.2 – Способи представлення об’єкта: a – центр об’єкта, особливі точки, c – форма об’єкта у вигляді прямокутник, d – об’єкт у формі еліпса, e – поєднання форм, f – скелетна модель, g та h – контур об’єкта, i – силует.

Також є ще декілька способів представити вигляд об’єкта. Найбільш широко використовуються такі:

- Щільність ймовірностей вигляду об’єкта. Може бути параметризована, наприклад, за Гаусом або змішування за Гаусом, чи непараметризована , як вікно Парзена
- Шаблони. Формуються з примітивних форм та силуетів. Їх перевага в тому, що вони несуть як просторову інформацію, так і про зовнішній вигляд. Використовуються, якщо положення об’єкта не змінюється.
- Активна модель вигляду. Генерується одночасно форма і вигляд об’єкта. В основному, об’єкт представляє собою набір орієнтирів. Для кожного з орієнтирів зберігається вектор вигляду: колір, текстура, градієнт.
- Багато-ракурсна модель. Ця модель описує об’єкт з різних ракурсів.

1.3 Виділення характерних рис для відстежування

Виділення правильних рис грає критичну роль у відстежу ванні об’єктів. В загальному випадку, характерні візуальні риси мають бути унікальними, щоб можна було виділити об’єкт відстежування порівняно з іншими. Виділення характерних рис тісно пов’язано з представленням об’єкта. Так границі об’єкта є характерними рисами для представлення контуром.

Основні візуальні риси:

- Колір. Видимий колір, може бути представлений в RGB чи HSV (Hue, Saturation, Value).
- Границі. Кордони об’єкта створюють значні зміни в інтенсивності зображення. Розпізнавання границь використовуються для знаходження цих змін. Важлива властивість границь – вони менш чутливі до змін світла порівняно з рисами кольору. Найбільш популярний 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 (наприклад, око людини або камера) і сцени. Популярні алгоритми знаходження оптичного потоку: Хорна, Лукас-Канаде.
- Текстура. Це міра зміни інтенсивності поверхні, яка визначає так характеристики як рівність та постійність.

1.4 Визначення об’єкта

Кожні методи відстеження потребують механізму визначення об’єкта на кадрах чи коли об’єкт вперше з’являється на відео. Загальний підхід у визначенні об’єктів – використання інформації з одиничного кадру. Але деякі алгоритми можуть використовувати тимчасову інформацію, яка була

підрахована з послідовності кадрів.

1.4.1 Детектор точок

Використовується для знаходження особливих точок на зображенні, які мають виразні текстури в їх відносних місцях локалізації. Бажана якість особливих точок постійна до змін навколишнього світла і позиції камери.

1. Оператор Моравека вираховує зміни варіацій інтенсивності у горизонтальному, вертикальному, діагональному та анти діагональному напрямках . Сила кута вираховується як найменша різниця квадратів між полем і його сусідами(у всіх напрямках). Зміна інтенсивності та точки зображені на рисунку 1.3:

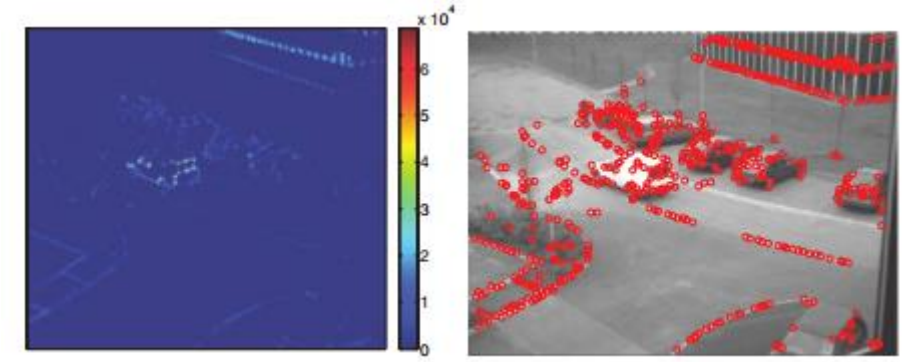

Рисунок 1.3. Особливі точки за допомогою оператора Моравека

2. Детектор Харріса розглядає похідну першого порядку від інтенсивності для визначення зміни яскравості в локальному околі точки. M – матриця, в якій записані зміни інтенсивності, вираховується для кожного пікселя.

$$R = \det M - k(trM)^2 > k,$$

$$M = \sum_{(u,v)\in W} w(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (1)$$

$$R = detM - k(trM)^2 > k \quad (2)$$

R – міра кута, де k – емпірична константа. Була запропонована, так як рахування власних чисел важка задача.

Таким чином значення R>0 для кутових точок. Далі відсікаються точки за пороговим значенням. Знаходяться локальні максимуми в околі заданого радіусу і вибираються в якості особливих точок.

3. Shi-Tomasi – використовується та ж сама матриця M. Але визначаються мінімальні власні числа матриці напряму.
4. SIFT. Будуються піраміди Гаусіан і визначаються екстремуми. Далі фільтруються особливі точки, наприклад, такі, що сконцентровані на границі. Визначається орієнтація ключових точок і будуються дескриптори.

1.4.2 Background subtraction

Визначення об’єктів можна досягнути за допомоги побудови представлення сцени, яка ще називається background model. Надалі шукаються відхилення від моделі для кожного нового кадру. Будь-які значні зміни в зоні зображення порівняно з моделлю означають, що об’єкт рухається. Значних покращень у побудові моделей є використання мультимодальних статистичних моделей для описання кольору кожного пікселя. Також використовують змішування по Гаусу для моделювання кольору пікселя. В цьому методі, пік сель в данному кадрі перевіряється порівняно з моделю фону, порівнюючи з кожним Гаусіаном в моделі до поки потрібний не буде знайдено. Якщо знайдено, то значення Гаусіана оновлюється, інакше додається новий Гаусіан зі значенням кольору поточного пікселя.

Рисунок 1.4 – Змішування по Гаусу для моделювання моделі заднього фону

Інший спосіб моделювання фону пропонує цілісний підхід використовуючи спектральний розклад матриць. Для к вхідних кадрів I (від одного до k), розмірів m на n, і матриця заднього фону B, розміру k на l, де l = m*n, тоді спектральний розклад для матриці B, $C=B^TB$. Отримані власні вектори охоплюють усі можливі освітлення, що знаходяться в полі зору. Розглянуто на зображені 1.5[4]:

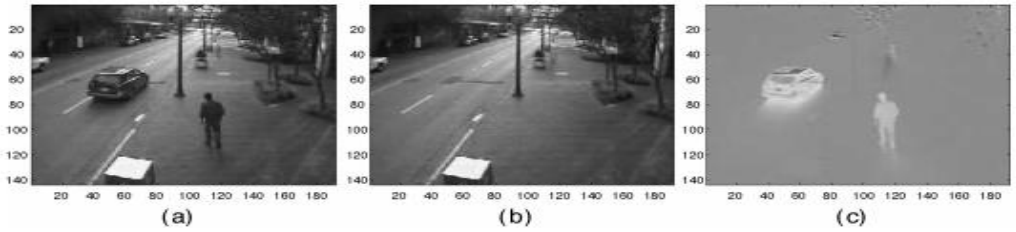

Рисунок 1.5 – Моделювання з використанням спектрального розкладу. Оригінальне зображення(a), отримане зображення заднього фону (b) , зображенні різниці (c)

Найважливішим недоліком моделювання заднього фону є те, що камера має мати фіксоване положення.

1.4.3 Сегментація

Ціль сегментації – розділити зображення на схожі зони. Кожен алгоритм вирішує 2 проблеми: критерій для гарного розбиття і метод для отримання ефективного розбиття.

Mean-Shift Clustering – Алгоритм базується на пошуку кластерів в поєднані простір+колір. На початку створюється велика кількість гіпотетичних кластерів випадковим чином. Далі центр кожного кластера переміщається до середнього значення даних, які лежать всередині еліпсоїда. Визначається mean-shift vector ітеративно поки не зміниться центр кластера. Алгоритм чутливий до вибору кольору і порогу мінімального розміру регіону, який утворює сегмент.

Сегментація зображення з використанням розрізу графа. Також може бут и сформульована як проблема розбиття графа, де вершини(пікселі), частково

розбиваються на N роз’єднаних під графів. Вага відрізаних граней між двома під графами називають розрізом. Ваги підраховуються як колір, яскравість чи схожість текстур між вузлами.

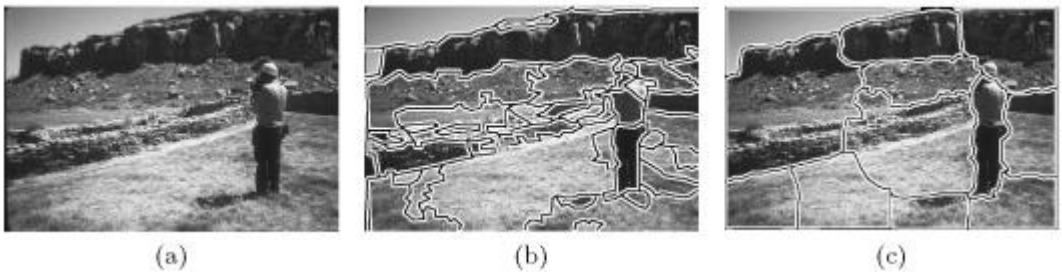

Рисунок 1.6. – Сегментація зображення(a) методом MeanShift(b), розрізом графа(c)

Активні контури. Сегментація зображення за допомогою виділення контурів близьких до граней об’єкта. Знаходиться контур , на якому деякий функціонал E – «енергія» - досягає мінімуму. Модель має динамічний характер.

1.4.4 Адаптивні системи розпізнавання зображення

Найбільш досконалі системи розпізнавання повинні вміти розпізнавати об’єкти і класифікувати їх за якимось ознаками. Наприклад, до якого класу відноситься об’єкт. В більш складних випадках, знайти лице заданої людини. Ці методи мають назву адаптивних тому, що вони здатні настроюватись, або як часто кажуть навчатись на даних, які обробляються.

Спочатку дані готуються для навчання - розміри всіх зображень мають бути однаковими. Далі важливим етапом є усунення надмірності .

Штучні нейронні мережі. Це універсальна нелінійна система, яка здатна настроювати свої параметри . Важливою здатністю є узагальненість, навчившись на деяких прикладах задач якогось класу, вона здатна давати розумні рішення задач цього класу.

Метод опорних векторів. Дозволяє знайти оптимальну гіперплощину, яка перпендикулярна найкоротшому відрізку, що з'єднує випуклі оболонки різних класів. Дані, що лежать на границі гіперплощини називають опорними векторами. В рамках розпізнавання об'єктів, ці класи відповідають класу об'єкта (позитивні зразки) та необ'єктним класам (негативні зразки). Якщо данні не можуть бути розділені гіперплощиною, то використовується нелінійна модель. Будується нелінійне відображення в інший багатомірний простір, де данні можуть бути розділені.[5]

1.5 Методи відстеження об'єктів

Ціллю відстеження об'єкта є створення траєкторії знаходження його позиції на кожному кадрі. Відстежувач також показує цілий регіон зображення, що займає об'єкт. Задача визначення об'єкта і встановлення взаємозв'язку між станами об'єкта на послідовності кадрів може вирішуватись окремо чи разом. Модель вибрана для представлення форми об'єкта накладає ліміт на способи відстеження.

1.5.1 Відстеження точок

Відстежування може бути сформульоване як визначення об'єктів по точках покадрово. Супроводження точок складна задача через наявність перекритів, неправильних визначень, входження і виходу об'єкта. Методи розділяються на дві різні категорії: детерміністичні і статистичні.

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. Супроводження визначається комбінацією наступних обмежень.

- Близькість
- Максимальна швидкість
- Незначні зміни швидкості (гладкий рух)

- Загальні рухи
- Жорсткість

Ці обмеження також можуть бути використані в контексті відстежування точок статистичними методами.

Статистичні методи. Зображення зняте матрицями камер завжди містить в собі шум. Більш того рух об'єктів зазнається випадковим ускладненням, наприклад, автомобіль, що виконує маневр. Теоретично найбільш оптимальним рішенням для знаходження послідовності станів точки є використання рекурсивного Байесового фільтра. Вирішення проходить в 2 етапи: передбачення та корекція.

Фільтр Калмана. Для відстеження одиночного об'єкта. Використовується для оцінки стану лінійної системи, де стани розподілені за Гаусом. На етапі екстраполяції використовується модель станів для передбачення нового стану змінних. Надалі етап уточнення вираховується відхилення вимірювання і оцінка відхилення. Фільтр Калмана широко використовується для відстеження об'єктів в зображеннях з шумом.[2]

Модель фільтру Калмана припускає, що

$$x_k = F_k x_{k-1} + B_k u_k + w_k, \quad (3)$$

де x_k – стан системи у момент часу k, F_k – модель переходу стану, що застосовується до попереднього стану x_{k-1}, B_k – моделль впливів керування, що застосовується до вектору керування u_k, w_k – шум процесу з нульовим середнім значенням коваріації Q_k.

$$z_k = H_k x_k + v_k \quad (4)$$

де H_k є моделлю спостереження, що відображає простір справжнього стану у спостережуваний простір, і v_k є шумом спостереження, що, як вважається, є гаусовим білим шумом з нульовим середнім значенням і з коваріацією R_k.

Передбачення:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1} + B_k u_k \quad (5)$$

де $\hat{x}_{k|k-1}$ – передбачена оцінка стану, B_k – модель впливів керування, що

застосовується до вектору керування u_k

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k \quad (6)$$

де $P_{k|k-1}$ – коваріація передбаченої (апріорної) оцінки, F_k – модель переходу стану, Q_k– матриця коваріації процесу шуму.

Уточнення:

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1} \quad (7)$$

де K_k – оптимальний передавальний коефіцієнт Калмана, H_k є моделлю спостереження, R_k – коваріація шуму спостереження.

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(z_k - H_k \hat{x}_{k|k-1}) \quad (8)$$

де $\hat{x}_{k|k}$ – оновлена (апостеріорна) оцінка стану, $\hat{x}_{k|k-1}$ – передбачена оцінка стану

$$P_{k|k} = P_{k|k-1} - P_{k|k-1} K_k H_k \quad (9)$$

Фільтр часток. Недолік фільтра Калмана є припущення, що змінні розподілені нормально. Інакше, фільтр Калмана дає слабкий результат. Цей недолік можна подолати, якщо використати фільтр часток. Густота ймовірності в час t представлена набором зразків {s(n) t : n = 1, ... , N} (частками) з вагами π(n). Ваги вказують на важливість зразка.

Multiple Hypothesis Tracking. Алгоритм здатний стежити за об’єктами, що були додані на сцену в процесі. Це ітеративний алгоритм. Для кожної гіпотези робиться прогноз позиції кожного об’єкта на наступному кадрі. Прогноз порівнюється зі справжніми вимірами. Кожна нова гіпотеза представляє набір відстежень на базі теперішніх вимірів.[4]

1.5.2 Kernel Tracking

Mean Shift. Використовує зважені гістограми для регіонів, що представляють собою регіони. Такий відстежував максимізує схожість за виглядом ітеративно, порівнюючи гістограми об’єктів Q, з вікном

гіпотетичного знаходження об’єкта P. Схожість гістограм визначається коефіцієнтом Бхатачарая:

$\sum^{u}{}_{b}$= 1 P(u)Q(u), $\sum_{b}^{u} P(u)Q(u)$, (10)

де b – кількість інтервалів, Q – гістограма об’єктів, P – вікно гіпотетичного знаходження об’єкта.

Метод Лукас-Канаде. Інший підхід, який базується на оптичному потоці. Ітеративно вираховується переміщення регіону центром, якого є ключові точки

Як тільки додається нова точка відстежувач підраховує якість супроводження нового регіону за допомогою афінного перетворення. Якщо сума різниці квадратів між регіонами мала, то точка надалі відстежується

2 ПОРІВНЯННЯ ПЛАТФОРМ РОЗРОБКИ ПРОГРАМИ

2.1 Платформа реалізації

Основним критерієм платформи є наявність бібліотеки для обробки зображення та відео. Тому вибір пав на мову кросплатформену мову програмування C++. Такою бібліотекою є OpenCV, яка має набір алгоритмів для комп'ютерного бачення, обробки зображення та численних алгоритмів загального призначення з відкритим кодом. Поширюється на найбільш популярні на сьогодні платформи – Windows, Linux, Mac, Android, IOS.

Окрім кросплатформеності сильною стороною мови програмування C++ є надійність та швидкість, що дозволяє

OpenCV містить в собі такі алгоритми: інтерпретація зображення, калібровка камери, усунення оптичних шумів, визначення подібності, аналіз преміщення об'єкта, сегментація зображення, аналіз жестів.

Основні модулі можна віднести до 4 груп:

- Модулі core, highgui, які реалізують базову функціональність(базові структури, математичні функції, лінійна алгебра, ввід/вивід зображення)
- Модулі imgproc, features2d для обробки зображення(фільтрації, геометричні перетворення, сегментація, пошук особливих точок)
- Модулі video, objdetect, calib3d (калібровка камери, аналіз руху, пошук положення в просторі, побудова карта глубин, оптичний потік)
- Модуль ml, який реалізує алгоритми машинного навчання(метод ближніх сусідів, наївний байесівський класифікатор, машина опорних векторів, нейронні мережі)

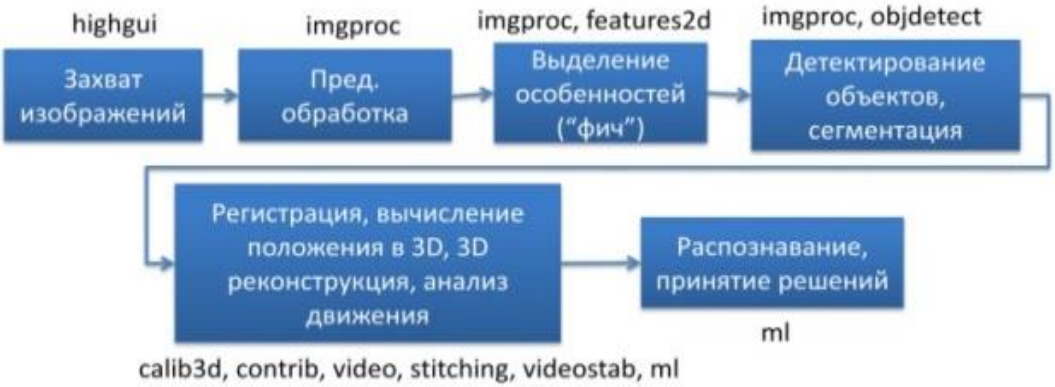

Рисунок 2.1 – Модулі бібліотеки OpenCV[11]

Все починається з захвату зображення (модуль highgui). Читаєте зображення з файлу або відео з потокової камери через мережевий протокол. Далі здійснюється попередня обробка (модуль imgproc), така, як усунення шуму, вирівнювання яскравості, контрасту, виділення і видалення відблисків, тіней. Наприклад, один і той же об'єкт при різному освітленні виглядає по-різному. У яскравому світлі червона машина, рух якої, необхідно відстежувати, буде яскраво-оранжевою. У похмуру погоду та ж машина буде виглядати червоно-рожевої. У цьому випадку на зображенні необхідно виконати вирівнювання кольору. Попередня обробка може бути простою, але може містити в собі цілу складну технологію.

Наступний етап - виділення особливостей (модулі imgproc, features2d). Наприклад, в завданны стеження за об'єктом це може бути пошук спеціальних точок на об'єкті, за якими легко спостерігати; для завдання детектування (тобто виявлення на зображенні) обличчя - обчислення опису кожного пікселя.

Далі відбувається детектування цікавих для нас об'єктів, виділення значущих частин, сегментація зображення (модулі imgproc, objdetect). Якщо, наприклад, камера нерухома, а зображення рухоме, можна використовувати алгоритми віднімання фону.

Після цього вирішується основне завдання, таке, як обчислення розташування об'єкта в 3d, реконструкція 3d структури, аналіз структури, реєстрацію і т. П. (Модулі calib3d, contrib, video, stitching, videostab, ml).

Наприклад в задачі склейки панорам зображень - це зіставлення частин різних кадрів, визначення потрібного перетворення. У задачі відеоспостереження це відновлення траєкторій об'єктів і т. д. Наприкінці відбувається розпізнавання і прийняття конкретних рішень (модуль ml). Наприклад, в системі відеоспостереження: з'явився небажаний об'єкт в кадрі чи ні. У задачі детектування тексту - визначити текст, що саме за текст і т. Д.

Ось коротко базове опис основних компонентів OpenCV і схеми роботи з бібліотекою. Тепер розглянемо бібліотеку більш докладно.

2.2 Попередня обробка зображення

Бібліотека OpenCV має досить багатий функціонал для обробки зображення. Розглянемо так функції та методи, які використовуються в області відстежування об’єктів.

Для створення моделі заднього фону використовують віднімання кадрів

cv::absdiff(frameTime1, frameTime2, frameForeground);

де frameTime1 – попередній кадр, frameTime2 – наступний кадр, frameForeground – результат.

За допомогою наступної функції можна вказати потрібні параметри для ігнорування шуму та малих змін. Наприклад, в діапазоні 15-255.

cv::threshold(frameForeground, frameForeground, 15, 255, cv::THRESH_BINARY);

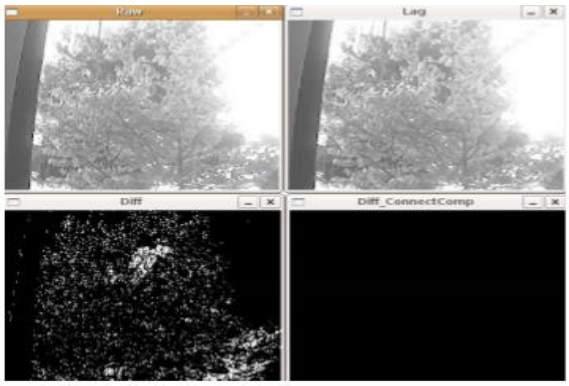

Рисунок 2.2 – Віднімання кадрів

За допомогою вбудованих класів BackgroundSubtractorMOg і BackgroundSubtractorMOg2 ініціалізується моделі для віднімання фону. Вони представляють собою дві різні моделі, але працюють за принципом Змішування Гаусіана для створення об’єкта віднімання фону.

```
pMOg->operator()(frame, fgMaskMOg);
pMOg2->operator()(frame, fgMaskMOg2);
```

Моделі оновлюється за рахунок вищезазначених методів. Тепер коли з’являються нові об’єкти, вони одразу стають частиною моделі.

2.2.1 Морфологічні зміни зображення

На метод віднімання фону впливає багато чинників. Його точність залежить від того як зображення обробляється. Одним із головних чинників є рівень шуму. Для вирішення цього недоліку використовується морфологічні зміни зображення, які змінюють форму ключових об’єктів на зображенні.

Над наступним зображенням виконаємо декілька морфологічних змін.[8]

Рисунок 2.3 – Оригінальне зображення

Розглянемо такий спосіб як ерозія. Це операція, яка зменшує границі форми

Рисунок 2.4 – Використання ерозії

void erode(inputImage, outputImage, element);

element вказує на те, яким чином будуть змінюватись границі.

Операція для потовщення називається розширення. Ця операція збільшує границі для всіх форм на зображення.

void dilate(inputImage, outputImage, element);

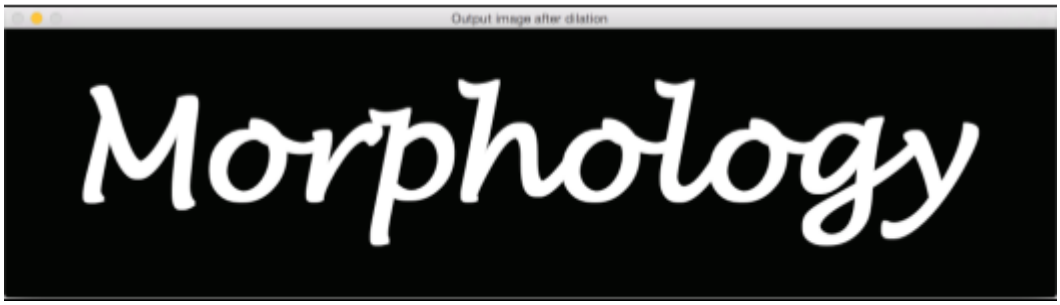

Рисунок 2.5 – Використання розширення

2.2.3 Визначення особливих точок

Пошук кутків - це техніка для визначення особливих точок на зображенні. Куток це в загальному випадку перетин двох граней. Куток є частим випаком особливих точок. В комп'ютерному баченні є популярна техніка визначення особливих точок, яка називається детектор кутків Харіса . Якщо запустити детектор Харіса, він буде виглядати так:

Для цього використовується функція:

void cornerHarris(InputArray src, OutputArray dst, int blockSize, int ksize, double k, int borderType=BORDER_DEFAULT)

Де параметри blocksize та k визначають кількість точок, які будуть знайдені.

Іноді потрібно визначити кутки з максимальною точністю. В opencv для цього використовується функція cornerSubPix(), яка надалі покращує кутки, що були знайдені з точністю сабпікселів. Спочатку знаходяться кутки Харіса і передається цетр цього кутка. Для цієї функції маємо визначити кількість ітерацій, щоб отримати потрібну точність. Також визначається розмір поля для

пошуку кутків. На рисунку 10 зображено кутки Харіса червоним кольором та знайдені сабпікселі за допомогою зазначеною функції.

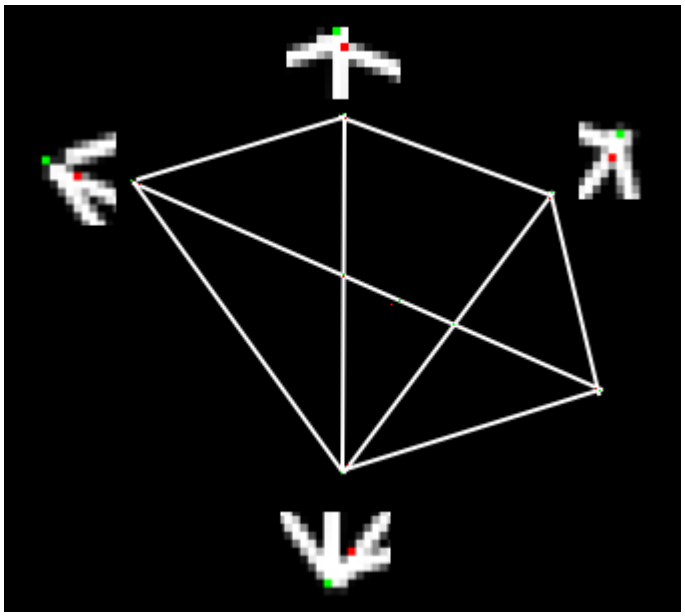

Рисунок 2.6 – Особливі точки знайдені за допомогою детектора Харіса та сабпікселі

Пошук кутків Харіса працює непогано в багатьох випадках, але може бути покращений. Близько шести років після публікаціх Харіса, Ши-Томасі покращили пошук кутків і назвали їх: «Гарні точки для відстежування». Вони виокристали іншу функцію розрахунку, що покращила загальну якість. Використовуючи функцію:

goodFeaturesToTrack(framegray, corners, numCorners, qualityThreshold, minDist, Mat(), blockSize, useHarrisDetector, k);

На рисунку 2.6 зображено особливі точки знайдені з виокристанням вищезазначеною функції.[8]

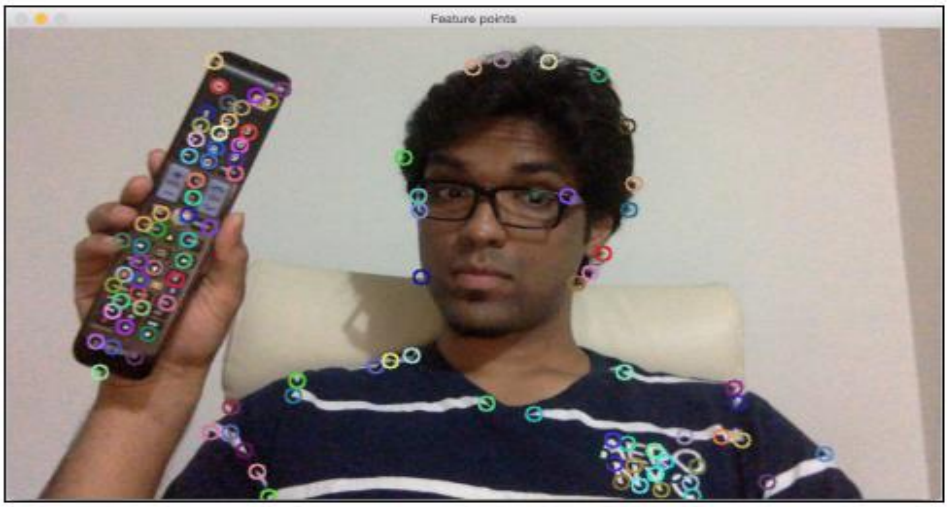

Рисунок 2.7 – Особливі точки знайдені за допомогою детектора Ши-Томасі

Кількість знайдених точок залежить від параметрів встановлених нами. Функція відсіює кутки, які нижчі за якістю, ніж встановлена. Також встановлюється мінімальна евклідова довжина між кутками.

2.2.3 Трекінг особливих точок

Трекінг особливих точок полягає у відстежування індивідуальних точок протягом кадрів у відео. Перевагою є те, що не потрібно визначати особливі точки в кожному кадрі. Ми можемо визначити їх один раз і відстежувати після цього. Це більш ефективний підхід, ніж визначення точок в кожному кадрі. Використаємо техніку оптичного потоку для відстеження точок. За допомогою функції void calcOpticalFlowLK(const CvArr* imgA, const CvArr* imgB, CvSize winSize, CvArr* velx, CvArr* vely);

Результатом роботи цієї функції є масив пік селів, для яких можливо підрахувати мінамальну похибку. В більшості випадків використовується покращений варіант цього алгоритму.

Пірамідальний алгоритм Лукас-Канаде. Завдякий покращенню, цей метод підраховує оптичний потік в піраміді.

Як бачимо цей алгоритм використовує особливі точки для відстежування і повертає показники того, як добре відстежується кожна точка.

void calcOpticalFlowPyrLK(const CvArr* imgA, const CvArr* imgB, CvArr* pyrA, CvArr* pyrB, CvPoint2D32f* featuresA, CvPoint2D32f* featuresB, int count, CvSize winSize, int level, char* status, float* track_error, CvTermCriteria criteria, int flags);

Ця функція містить багато вхідних параметрів. Перші два аргументи – це початкове та наступне зображення, кожні мають бути одно канальні та 8-бітні зображення. Наступні два аргументи буфери для збереження піраміди зображенняю. Далі передається масив точок, які будуть відстежуватися. Також вказується критерій закінчення. Це структура, що використовується у багатьох алгоритмах, що ітерують рішення.

cvTermCriteria(int type, // CV_TERMCRIT_ITER, CV_TERMCRIT_EPS, or both int max_iter, double epsilon);

Перший аргумент функції вказує на те, яким чином ми хочемо закінчити алгоритм: при досягненні необхідний кількості ітерацій чи коли результат роботи досяг малого значення меншого за epsilon.

Останнім аргументом є флаги CV_LKFLOW_PYR_A_READY, CV_LKFLOW_PYR_B_READY та CV_LKFLOW_INITIAL_GUESSES

Рисунок 2.8 – Знаходження особливих точок

Рисунок 2.9 – Відстеження особливих точок за допомогою пірамідального методу Лукаса-Канаде.

Розглянемо алгоритми mean-shift та camshift. В області комп'ютерного бачення використовуються в багатьох программах. Алгоритм знаходить відповідність розподілу набору точок.[9]

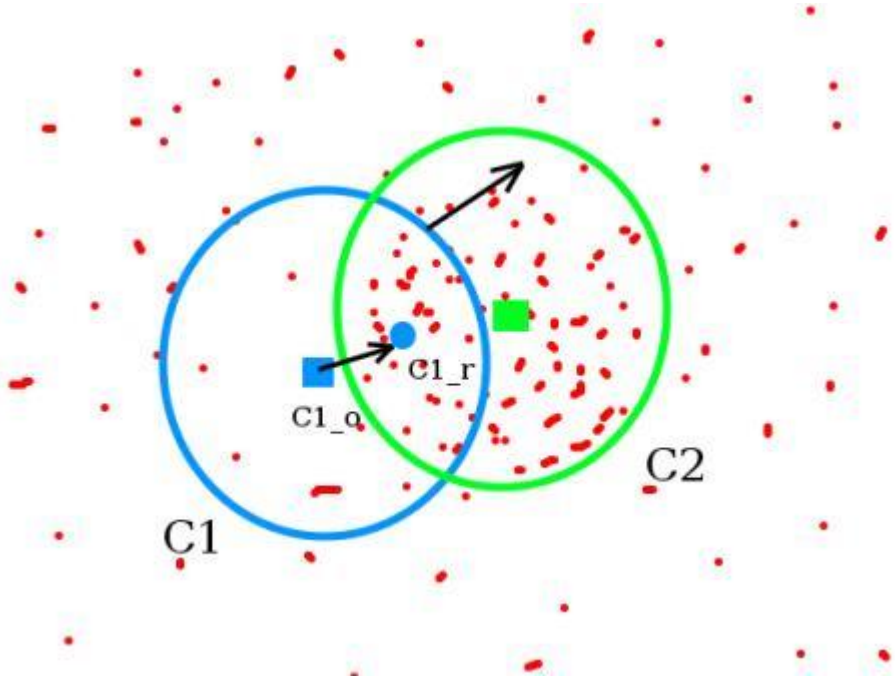

Рисунок 2.10 – Рух точок в бік більшої щільності пікселів.

int cvMeanShift(const CvArr* prob_image, CvRect window, CvTermCriteria criteria, CvConnectedComp* comp);

Розглянемо аргумети, що приймає функція, prob_image, представляє щільність можливих точок. Windows – вікно, в якому відбувається початковий пошук. Зв’язаний компонент містить розмір вікна та суму всіх пікселів.

На рисунку 2.9 зображено застосування даної функції[9]

Рисунок 2.11 – Відстеження позиції автомобіля за допомогою алгоритма meanshift

Пов’язаний з попереднім алгоритмом, Camshift трекер відрізняється тим, що вікно пошуку підлаштовується під розмір об’єкта. Якщо маємо добре-сегментований розподіл, наприклад лице, потім алгоритм автоматично підлаштовує розмір лиця людини, якщо вона приближається та віддаляється від камери.

int cvCamShift(const CvArr* prob_image, CvRect window, CvTermCriteria criteria, CvConnectedComp* comp, CvBox2D* box = NULL);

Перші 4 параметри такі ж самі, останній box зберігає новий розмір прямокутник. Для відстеження цей прямокутник підрахований в попередньому кадрі передається як розмір вікна у наступному кадрі[9].

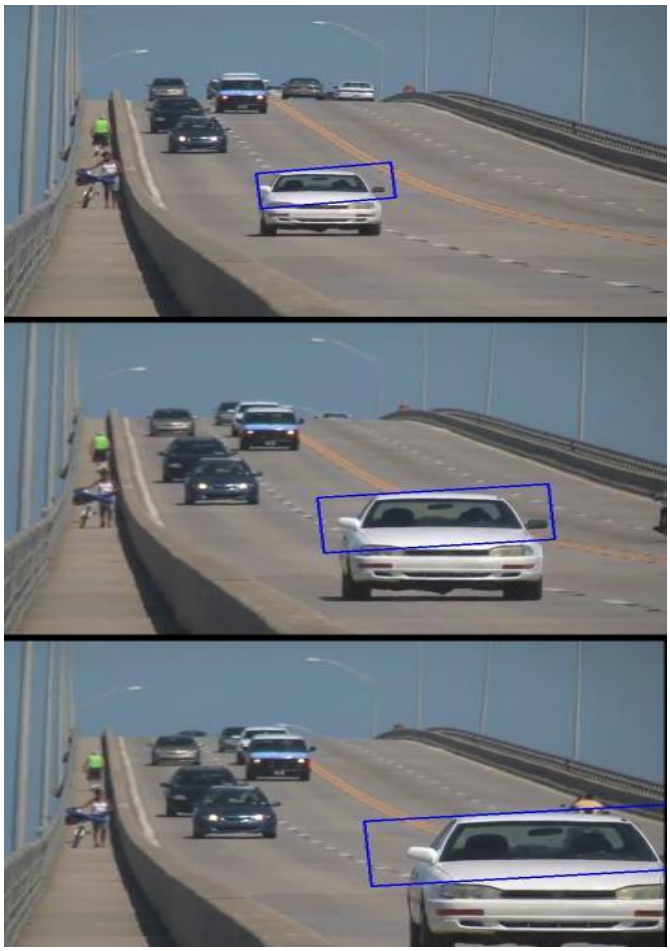

Рисунок 2.12 – Відстеження автомобіля за допомогою алгоритму Camshift. Розмір рамки змінюється в результаті того, що автомобіль наближається до камери

2.2.4 Оцінка відстеження

Якщо потрібно оцінити рух об’єкта таким шляхом, щоб отримати максимальний результат. Так, кумулятивний ефект багатьох вимірів дозволяє визначити траєкторію і оцінити її. Завдання поділяється на дві фази: передбачення,використовується інформація про попередній рух об’єктів для подальшого визначення позиції. У другій фазі, кореляції , передбачення на основі попередніх вимірів співставляються з результатом.

cvKalmanPredict(CvKalman* kalman, const CvMat* control = NULL); cvKalmanCorrect(CvKalman* kalman, CvMat* measured);

Ці дві функції реалізують фільтр Калмана. Передається матриця станів і для прогнозування викликається cvKalmanPredict() і надалі коригуються функієї cvKalmanCorrect. Після виконання цих двох функцій можливо відстежувати стан системи.

2.3 Matlab Image Processing i ComputerVision

Це система, яка постачає алгоритми, функції та додатки для розробки й імітації комп’ютерного бачення і систем відео обробки. Можливі виділення, визначення і співставлення, визначення об’єктів. Також можлива попередня обробка зображення, покращення ясності і видалення шумів та інших артефактів. Визначення кількості об’єктів на сцені та пошук особливих точок.

Ця система потребує встановлення програми Matlab, яка є платною. Можливий варіант використання протягом безкоштовного пробного періоду.

2.3.1 Способи обробки зображення

Серед методів покращення зображення можна виділити 3 ефективні техніки:

- Imadjust збільшує контрастність зображення, перетворюючи значення вхідної інтенсивності зображення.

- Histeq виконує гістограму вирівнювань. Ця функція покращує контрастність зображення, перетворюючи значення інтенсивності так, що гістограми вихідного зображення співпадає з гістограмою вказаною (нормального розподілу).
- Adapthisteq виконує контрастно-лімітоване адаптивне вирівнювання гістограм, порівнюючи з попередньою функцією, працює з малими регіонами даних (тайлами), а не з усім зображенням.[10]

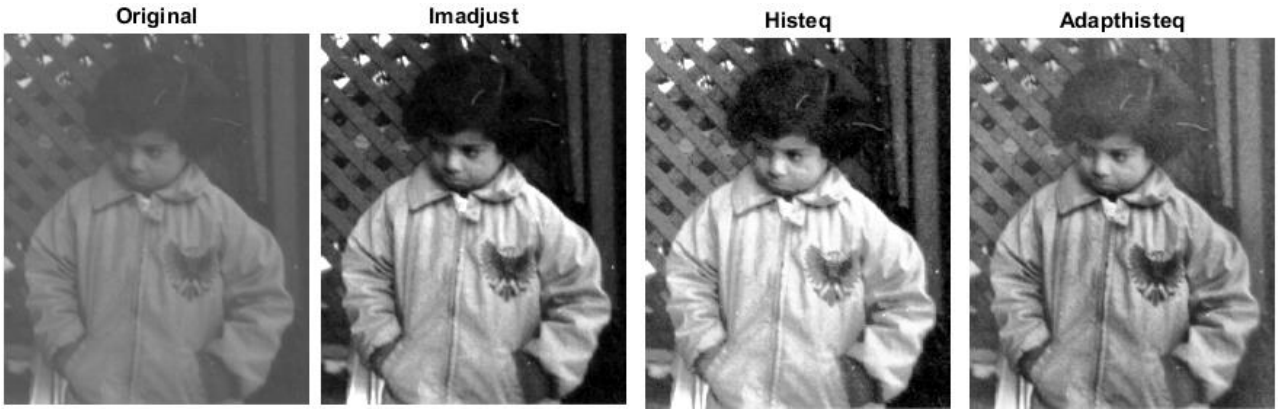

Рисунок 2.13 – Оригінальне зображення, оброблене imadjust, histeq та adapthisteq.

Морфологічні зміни зображення:

За допомогою функцій imdilate та imerode виконати операції розширення та ерозії. На базі них створено дві загальні операції:

- Створення скелету - bwmorph(BW1,'skel',Inf);

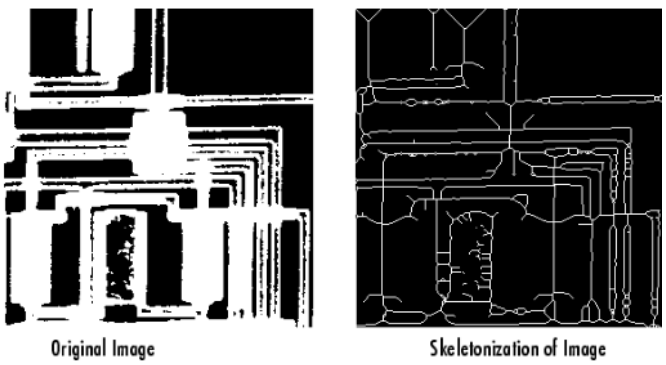

Рисунок 2.14 – Створення скелету зображення[10]

- Виділення периметрів - bwperim(BW1);

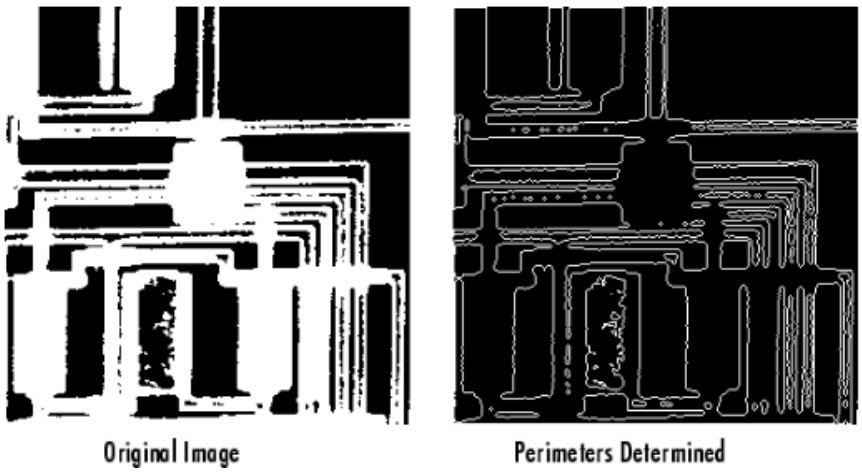

Рисунок 2.15 – Знаходження периметру об’єктів[10]

Серед функціоналу цього пакета: знаходження границь об’єктів, сегментація зображення, порогове зображення (thresholding).

2.3.2 Пакет Computer Vision System

Даний пакет включає в себе такі технології визначення особливих точок, пошук об’єктів та розпізнавання, відстеження об’єктів та оцінка руху, калібрування камери, побудова геометрії об’єкта та інші.

Алгоритми пошуку особливих точок FAST, Харіса, Ши-Томасі детектор кутків, SURF.

Наприклад, функція detectSURFFeatures () визначає особливі точки за алгоритмом Speed Up Robust Fearures.[10]

Рисунок 2.16 – Ключові точки SURF[10]

2.3.3 Визначення об’єктів

Computer Vision System Toolbox™ дає можливість використати різні підходи для визначення об’єктів, такі як відповідність шаблонів, аналіз регіонів, алгоритм Віола-Джонса, класифікація зображень.

Наприклад, об’єкт BlobAnalysis підраховує статистику поєднаних регіонів у бінарних зображеннях. Такоє статистикою є площа регіону та центр прямокутника, що описує її, осі еліпса, орієнтація і периметр.

Об’єкт ForegroundDetector порівнює кадри зображення у відтінках сірого для побудови моделі заднього фону. Для цього використовується моделі змішування Гауса. Серед параметрів функції: кількість кадрів для навчання, мінімальний поріг моделі заднього фону, кількість Гаусіан.

Для визначення прямокутника, що обмежує регіон у скупченнях використовується функція selectStrongestBbox()

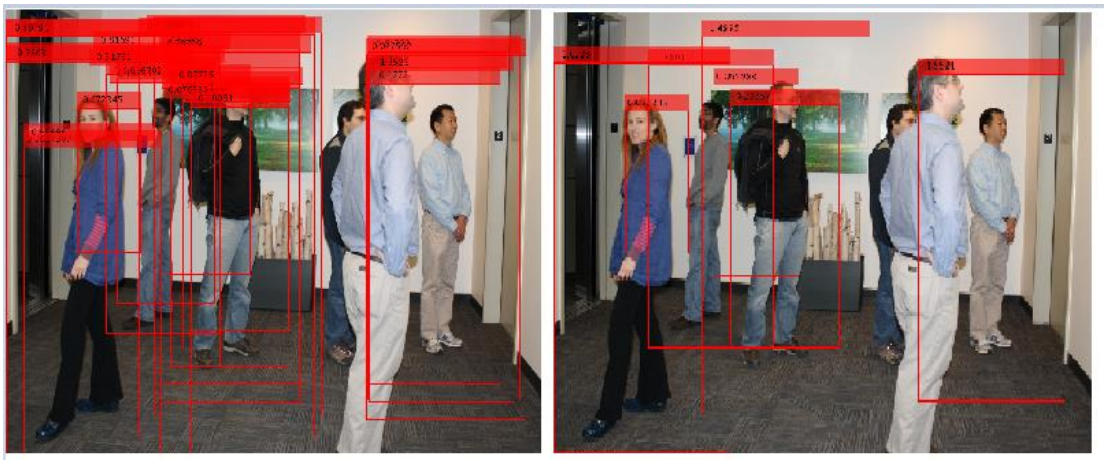

Рисунок 2.17 – Вибір найкращого прямокутника

Вістеження об’єктів та їх супроводження застосовують вже відомі попереднє алгоритми: оцінка руху за допомогою фільтра Калмана, алгоритма Канаде-Лукаса-Томасі, каскадного класифікатора.

Для відстеження об’єктів, використовуючи фільтр Калмана потрібно виконати такі дії:

- Створити об’єкт vision.KalmanFilter використовуючи configureKalmanfilter
- Застосувати методи predict та correct в послідовності, щоб зменшити присутність шуму в системі відстеження
- Застосувати метод predict, щоб оцінити положення об’єкта.

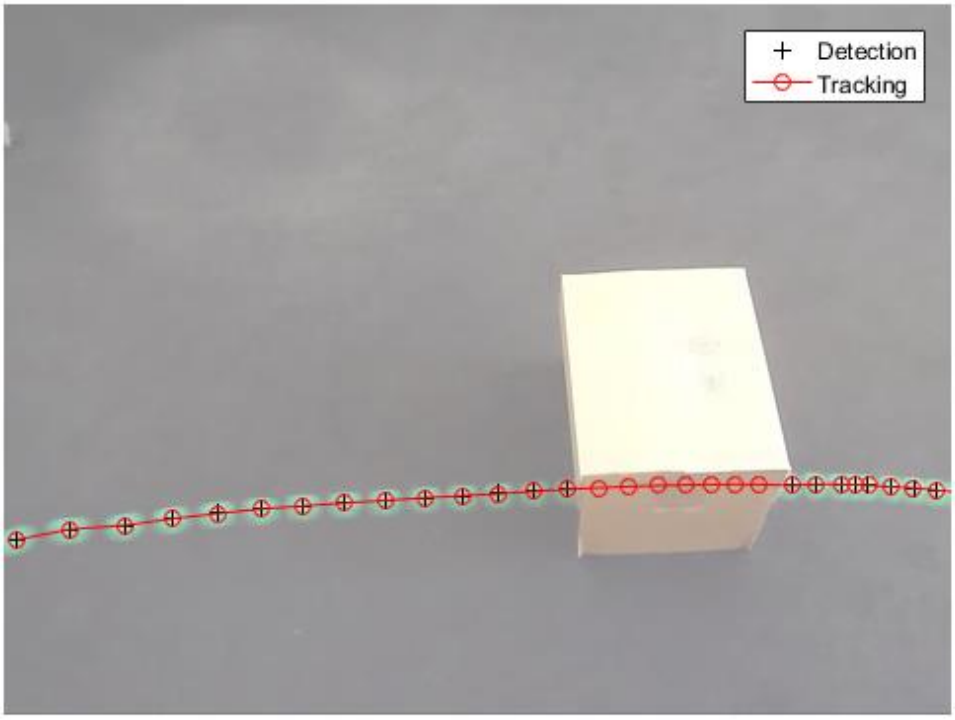

Рисунок 2.18 – Зображено рух об’єкта та оцінка положення (черовним) фільтром Калмана[10]

2.4 Засоби та платформа реалізації

Проаналізувавши алгоритми та методи зрозуміло, що перевага має надаватися мовам програмування високо рівня, які здатні продуктивно і швидко виконувати код. Так було обрано для розробки мову C++ в поєднанні з бібліотекою QT. QT — кросплатформовий інструментарій розробки програмного забезпечення (ПЗ) мовою програмування C++. Дозволяє запускати написане за його допомогою ПЗ на більшості сучасних операційних систем (ОС), просто компілюючи текст програми для кожної операційної системи без зміни сирцевого коду. Містить всі основні класи, які можуть бути потрібні для розробки прикладного програмного забезпечення, починаючи з елементів графічного інтерфейсу й закінчуючи класами для роботи з мережею,

базами даних, OpenGL, SVG і XML. Бібліотека дозволяє керувати нитями, працювати з мережею та забезпечує крос-платформовий доступ до файлів.

Qt також може бути використаним в багатьох інших мовах програмування: Ada (QtAda), C# (Qyoto/Kimono), Java (Qt Jambi), Qt Jambi, Node.js, Pascal, Perl, PHP (PHP-Qt), Ruby (QtRuby), та Python (PyQt,PySide).

Qt 5, який вийшов у грудні 2012, примітний модульною структурою та зміщенням акценту в бік використання для написання застосунків засобів декларативного опису інтерфейсу з визначенням логіки взаємодії з користувачем мовою JavaScript, у той час як застосування C++ позиціонується для реалізації критичних до часу виконання або надмірно складних частин програми, а також для створення нових модульних бекендів для Qt Quick. Незважаючи на багато істотних поліпшень і змін, Qt 5 зберігає базову зворотну сумісність із минулими випусками, підтримує повною мірою засоби для створення Qt-програм мовою C++ і містить майже всі компоненти Qt 4 (припинена підтримка давно застарілих елементів), більшість модулів колишнього Qt Mobility й деякі експериментальні елементи з Qt Labs.

Компоненти, що входять до складу пакета Qt 5.0:

Базові бібліотеки

- Qt Core
- Qt Network
- Qt Gui
- Qt Sql
- Qt Testlib
- Qt Widgets
- Qt QML
- Qt Quick
- Qt Multimedia
- Qt WebKit
- Qt WebKit Widgets

Доповнення

- Qt Xml
- Qt XmlPatterns
- Qt Svg
- Qt Concurrent
- Qt Printsupport
- Qt Dbus
- Qt OpenGL
- Qt ActiveQt
- Qt Graphical Effects
- Qt Script
- Qt Declarative
- Qt Image Formats

2.5 Висновки

Огляд існуючих платформ та бібліотек для аналізу зображень, визначення об’єктів та відстеження показав тенденції у розвитку в цій області. Реалізовано більшість сучасних алгоритмів та методів для роботи з зображенням та трекінгу об’єктів. Розглянуті платформи продовжують збільшувати функціонал з кожною новою версією. Ці технології бурхливо розвиваються в наш час так, як можуть знайти застосування у багатьох областях.

Аналіз показав, що пакет ImageProcess та ComputerVision Matlab має значну кількість реалізованих технологій та методів, більш простий у використанні та якісну документацію. З іншого боку стиль написання коду відрізняється від так мов програмування як C++ чи Python, що може викликати деякі незручності. Ліцензія для використання є платною. Програма написана у Matlab набагато повільніша та менш оптимізована у порівнянні з програмою на C++.

У порівнянні, бібліотека OpenCV(C++) також покриває велику кількість алгоритмів в області комп’ютерного бачення. Бібліотек оптимізована для продуктивного виконання на будь-якому пристрої – програми для вбудованої техніки або ж мобільні додатки. Бібліотека повністю безплатна і тому має значну підтримку з боку суспільства. Недоліками є слабка документація і невелика бібліотека машинного навчання, складність у налагодженні програм.

Отже, проаналізувавши альтернативи було визначено, що найбільш оптимальною для розробки програми є OpenCV. Це безкоштовна бібліотека, яка дозволяє створювати оптимізовані кросплатформені програми, постійно модернізується і є повністю безкоштовною.

3. РОЗРОБКА ПРОГРАМИ СУПРОВОДЖЕННЯ ОБ'ЄКТІВ НА ВІДЕО ПОСЛІДОВНОСТІ

3.1 Розробка алгоритму роботи програми

Було досліджено достатньо методів та технологій для визначення об'єктів. Найбільш продуктивними є пошук ключових точок, та використання методу віднімання фону (background substraction). Це метод, який будує модель фону і визначає бінарну маску – де 1 це об'єкт, що рухається, 0 – фон. Обрано рекурсивний алгоритм, який використовує інформацію про інтенсивність даного кадру, а значення маски вираховується як набір нормальних розподілів (за Гаусом). Для обробки кожного наступного кадру оновлюється математичне сподівання і середньоквадратичне відхилення. Коефіцієнт, який називається вагою характеризує наскільки часто змінюється піксель.

Для зниження шуму зображення і зменшення неточних визначень об'єктів, наступним кроком виконуються морфологічні операції. Послідовне використання операцій розмикання та замикання. Операція розмикання допомагає видалити з зображення елементи менші за структурний елемент, але це зумовлює нарощення контуру. Щоб уникнути цього застосовується операція замикання.

Вище зазначені методи й операції дозволяють безпосередньо перейти до визначення границь об'єктів. Застосуємо детектор границь Кенні. Цей алгоритм складається з 5 наступних кроків:

- Згладжування. Розмиття зображення для видалення шумів.
- Пошук градієнтів. Границі відмічаються там, де градієнт зображення набуває максимального значення.
- Відкидання не максимумів. Тільки локальні максимуми відмічаються як границі.
- Подвійна гранична фільтрація. Потенціальні границі визначаються

порогами.

- Трасування областей невизначеності. Визначаються найсильніші границі.

Виконання цього алгоритму дозволяє отримати набір контурів, які вже можна вважати конкретними об’єктами для відстеження. Одним з найважливіших кроків у відстеженні об’єктів є порівняння їх між собою. Мається на увазі визначення нових об’єктів або встановлення зв’язку між об’єктом у попередньому кадру й наступному. Використання Угорського алгоритму (Мункре) про призначення дозволяє за поліноміальний час визначити чи це новий об’єкт або ж об’єкт з попереднього кадру. Основним процесом алгоритму є побудова матриці коштів. Ця матриця представляє собою Евклідові відстані між об’єктами попереднього і наступного кадру. Основні кроки алгоритму:

1. Побудувати матрицю коштів.
2. Пошук мінімального елементу в кожному рядку і віднімання його від цього рядка.
3. Пошук мінімального елементу в кожному стовбці і віднімання його від цього стовбцю.
4. Пошук найбільшого сполучення пар.

В результаті отримуємо індекси нових та встановлених об’єктів. Останнім кроком роботи алгоритму супроводження об’єктів є передбачення наступного положення об’єкта і коригування вимірів за допомогою фільтра Калмана.

Для того, щоби використовувати фільтр Калмана для оцінювання внутрішнього стану процесу, маючи лише послідовність спостережень, необхідно змоделювати процес відповідно до моделі фільтру Калмана. Це означає задання наступних матриць: Fk, моделі переходу станів; Hk, моделі спостереження; Qk, коваріації шуму процесу; Rk, коваріації шуму спостереження; та іноді Bk, моделі керування, для кожного моменту часу, k, як описано нижче.

Алгоритм розроблено за умов, що камера, яка передає відео послідовність нерухома, а частота кадрів на більше 60 Гц. Розмір об’єктів, що визначаються не менше 10 пікселів висотою і шириною. Рух об’єкта поступальний по

прямолінійній або криволінійній тратторії, або періодичний. Швидкість руху не більше 100 пікселів на кадр.

3.2 Архітектура програми

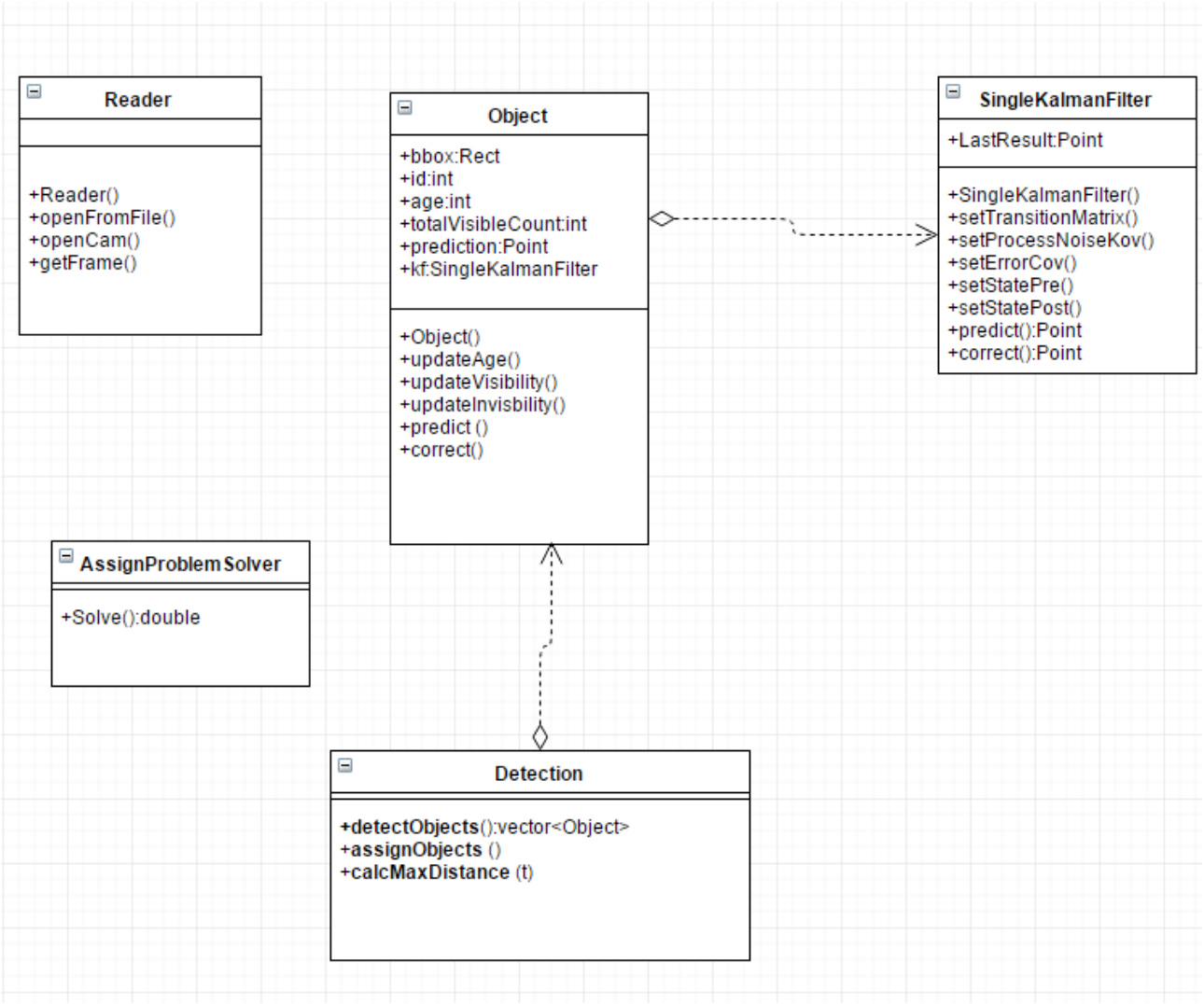

Рисунок 3.1 UML діаграма класів програми

Створення програми відбувається згідно з розробленою діаграмою класів програми. Розглянемо складові діаграми, щоб детальніше зрозуміти роботу програми.

Клас Reader відповідає за зчитування кадрів з відео послідовності. Можливі два варіанти використання: відкриття файлу або веб камери. Якщо обрано відкриття файлу з’являється діалогове вікно для вибору відео.

Бібліотека OpenCV гарантує підтримку більшості форматів та кодеків.

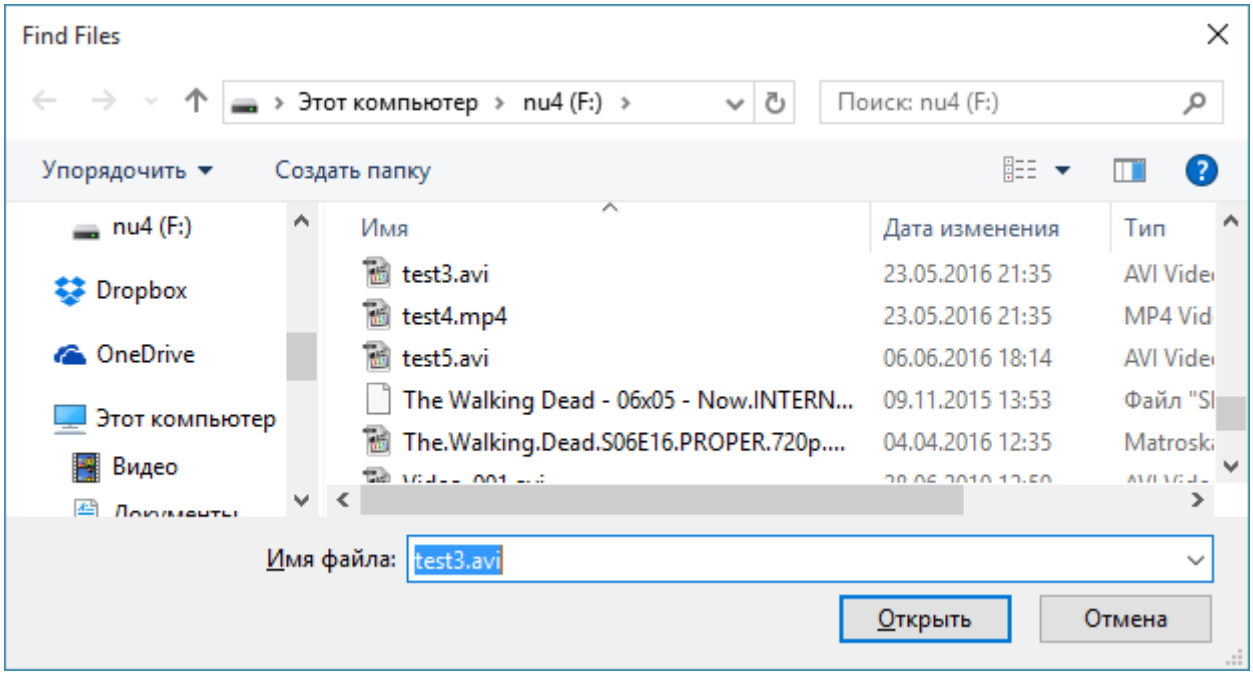

Рисунок 3.2 Діалогове вікно обрання відео

Після зчитування кадру, клас Detection відповідає за визначення і відстеження об’єктів. Обробку зображення, фільтрацію, створення моделі заднього фону, морфологічні операції та визначення об’єктів виконує метод detectObjects(Mat frame). Містить масив об’єктів. Для побудови моделі заднього фону зі змішуванням за Гаусом використовується функція cv::createBackgroundSubtractorMOG2(history,threshold,false), де history – кількість початкових зображень потрібних для створення моделі заднього фону, threshold – межа кількості зображень для побудови моделі.

Кожен об’єкт відео представляє собою екземпляр класу Object. Клас Object містить поля, які зберігають інформацію про вік об’єкта – скільки кадрів пройшло з моменту першого визначення, кількість кадрів, які програма відслідковувала об’єкт, положення. Для кожного об’єкта створюється фільтр Калмана, який представляє собою екземпляр класу SingleKalmanFilter, встановлюються всі параметри матриці коваріації шуму процесу, коваріації шуму спостереження. Методи predict() та correct() повертають, точку

положення об’єкта відповідно – передбачену та скориговану. Угорський алгоритм співставлення об’єктів реалізований у класі AssignProblemSolver.

Важливим є відсіювання об’єктів, які не підходять під параметри розробленого алгоритму. Одним з таким є кількість кадрів, поки програма не відстежує об’єкт. Цей параметр забезпечується пошуком найближчих об’єктів в попередньому і наступному кадру. Якщо вона більше за деяке значення dist_thres, то це означає, що об’єкт було загублено і він видаляється з масиву об’єктів. Щоб забезпечити універсальність програми, відстань вираховується як 10% від довжини діагоналі зображення. Що дає результат при відстеженні об’єктів на різних відео послідовностях.

3.3 Висновок

В розділі наведені деталі алгоритму та архітектури. Розробка програми визначення та відстеження об’єктів на відео послідовностях відбувається відповідно спроектованої архітектури. Наведено аргументи на користь обраних алгоритмів. Розроблено програму з використанням мови високого рівня C++ та інструментарію Qt. Для обробки зображень застосовано готові методи та функції бібліотеки OpenCV.

4. РЕЗУЛЬТАТИ ТЕСТУВАННЯ ПРОГРАМИ

4.1 Тестування програми

Для оцінки результатів роботи будемо розглядати наскільки точно програма розпізнає об’єкти, що рухаються. Тобто кількість правильно визначених об’єктів і шуму. Оскільки програма працює тільки з послідовностями зображень, які мають статичний фон. Деякі зміни у фоні, наприклад, коливання дерев чи снігопад зумовлюють їх розпізнавання як об’єктів переднього фону.[7] Неякісне зображення може привести до помилкових визначень об’єктів, що також дає негативний результат. Отже, емпіричним шляхом проаналізуємо результати роботи програми.

4.1.1 Приклад № 1

Даний приклад є ідеалізованим. Рівень якості зображення достатньо високий. Містить 2 об’єкти.

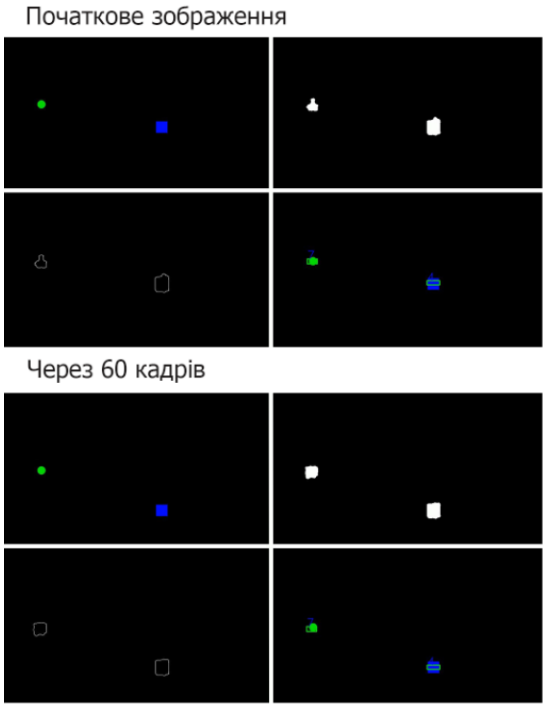

Рисунок 4.1 Ілюстрація результату роботи програми

По порядку програма виводить оригінальне зображення, отриману бінарну маску (зліва верхнє зображення), знайдені границі за допомогою визначення границь Кенні. Визначені об’єкти позначаються прямокутниками зеленого кольору і мають порядковий номер. Результати програми повністю співпадають з вимогами: знайдені об’єкти – це коло та прямокутник, що відповідно рухаються з постійною швидкість.

Отже, зрозуміло, що програма показує якісні результати на досить ідеалізованих прикладах.

4.1.2 Приклад № 2

Розглянемо більш реальний приклад.

Рисунок 4.2 – Зображення роботи програми

Розмір кадру даного зображення – 320 пікселів в довжину і 240 в ширину. Послідовність містить об’єкт, що рухається по криволінійнійній траекторії. Протягом усієї послідовності програма надає можливість бачити зміни положення об’єкта, що відслідковується. Отже, незважаючи на невеликі розміри об’єкта та шум реальної камери, результат даного прикладу повністю задовольняє очікування.

4.1.3 Приклад № 3

Розмір кадру даної послідовності зображення – 640 пікселів на 480. Нелінійний рух декількох об’єктів. Ця послідовність містить сильний шум, що негативно вплинув на результат роботи програми. Проаналізуємо якість розпізнавання і визначення об’єктів на цьому прикладі:

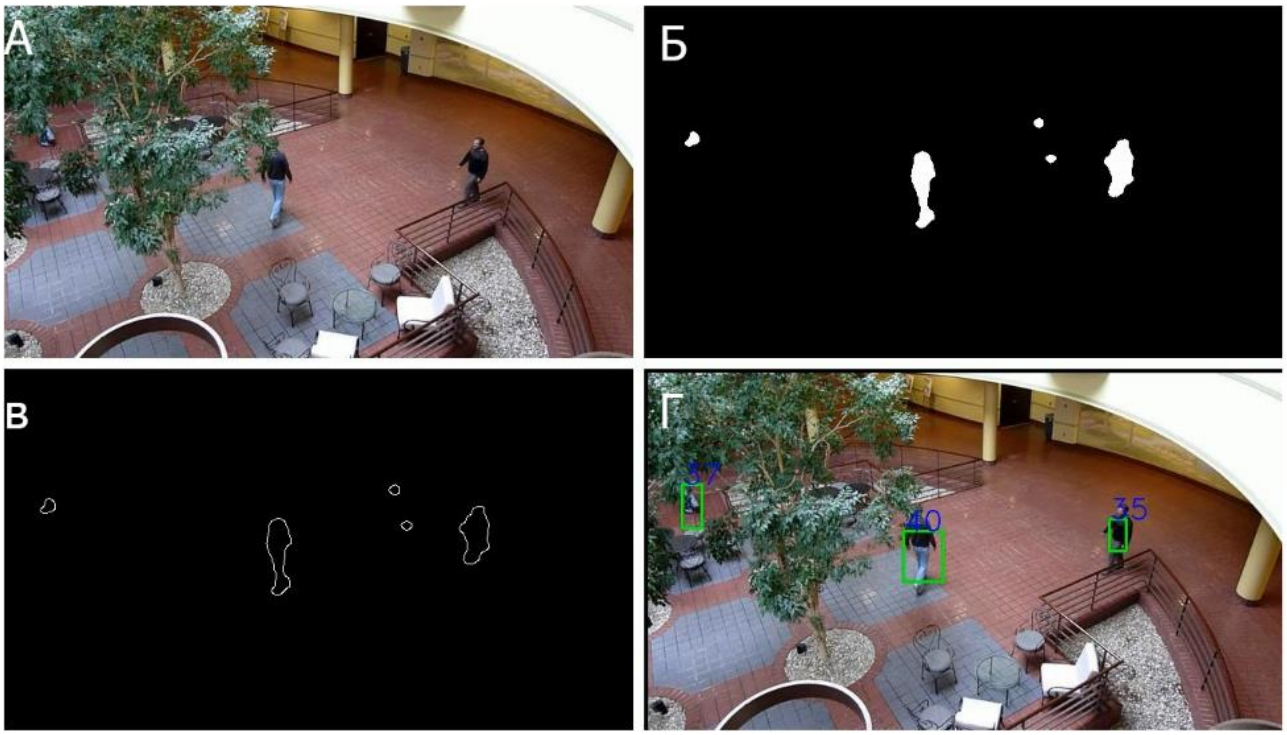

Рисунок 4.3 – Результат роботи програми для прикладу №3

Як бачимо з рисунку 4.3 програма чітко визначило 3 об’єкта, що рухаються навіть об’єкт, що знаходиться за перекриттям – об’єкт з порядковим

номером 37. На бінарній масці – зображення Б рисунку 4.3 помітні контури, що є шумом але за рахунок роботи алгоритму програми вони не визначаються як об’єкти, що рухаються. Надалі шум став більш інтенсивнішим, що вплинуло на роботу програми:

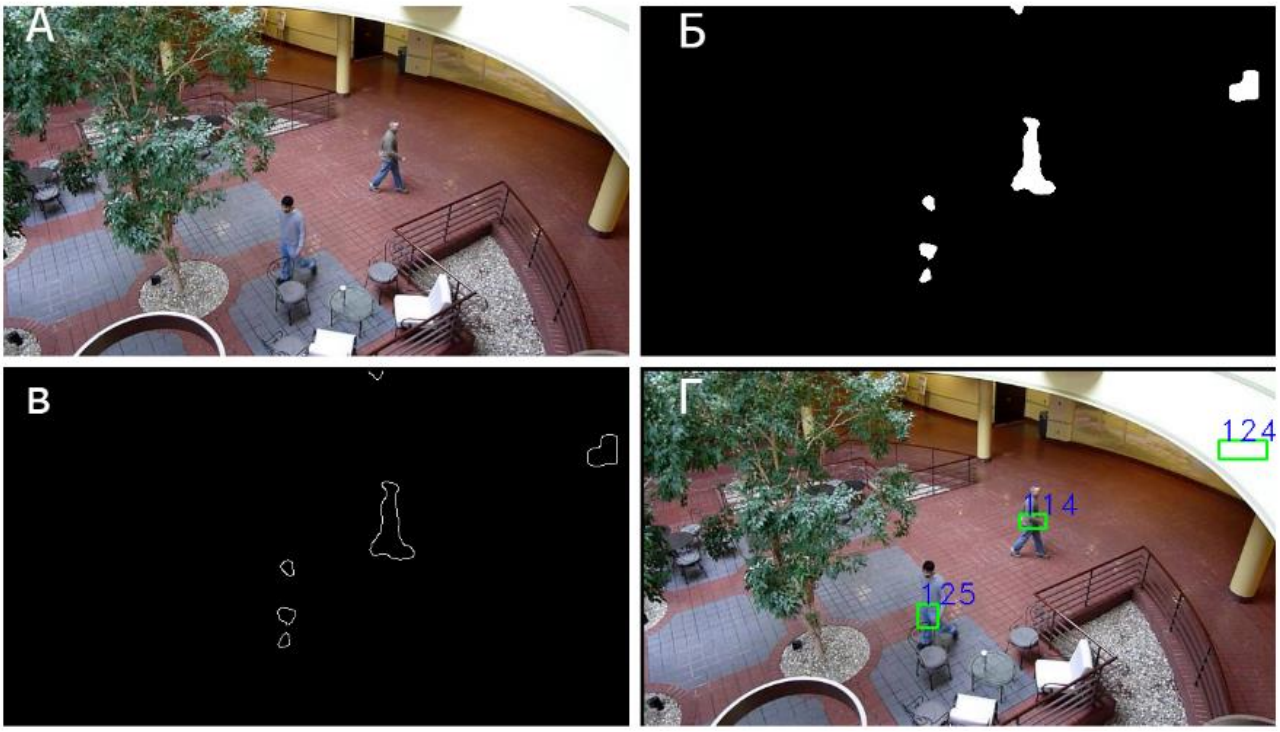

Рисунок 4.4 – Результат роботи програми для прикладу №3

На рисунку 4.4 (Г) бачимо неправильно визначений об’єкт з порядковим номером 124. Оскільки даний контур був наявний протягом необхідної кількості кадрів і його розмір досяг достатніх значень. Алгоритм визначив його як об’єкт, що рухається. Даний недолік виявився не одразу, адже інтенсивність шуму зростала і алгоритм віднімання фону за Гаусом прирівняв дані пікселі до переднього фону – на рисунку 4.4 (Б). Визначення інших об’єктів – порядкові номери 125 та 114 були вдалими та забезпечують коректну роботу програми в цілому.

4.3 Способи покращення результату

Основним недоліком алгоритму є розпізнавання шуму як об’єкта переднього плану. Це пов’язано зі способом створення моделі заднього фону і використання методу змішування моделей за Гаусом. Для того, щоб уникнути таких помилкових визначень можна використовувати різні способи фільтрації об’єктів:

- Для пошуку об’єктів використовувати співставлення за шаблоном. Кожне зображення порівнюється з зображенням шаблона для локалізація об’єкта. Такими шаблонами можуть бути силуети людини, машини тощо.
- Використання штучних нейронних мереж для класифікацій об’єктів на зображенні за їх параметрами. Наразі, використання нейронних мереж стає все більш популярною технікою для класифікацій, але вона потребують тренування на великій кількості наборів даних, щоб дати необхідний результат.
- Застосування багатокамерного трекінгу для співставлення об’єктів на різних кадрах одного моменту часу. Це потребує точної калі бровки камери. Дозволяє позбавитись недоліків оклюзії одного об’єкта іншим.
- Використання методів оптичного потоку для врахування впливу зміни освітлення, інтенсивності руху.
- Зменшення шуму зображення за рахунок більш якісного зображення камери та більшої кількості кадрів на секунду.

Майже кожний з розглянутих вище методів потребує збільшення навантаження на процесор для підрахування додаткових параметрів. Це суттєво знижує ефективність трекінгу об’єктів в реальному часі, але підвищує якість визначення і відстеження.

4.4 Висновки

Розроблено програмне забезпечення, яке відповідає завданню. Алгоритм програми успішно справляється з прямолінійним та криволінійним рухом об’єкті, незалежно від швидкості. Також було продемонстровано роботу з різними розмірами кадру та об’єкта. В цілому програма показує задовільні результати за умови, що послідовність забезпечена нерухомою камерою. Рівень шуму негативно впливає на визначення об’єктів і дає помилкові результати.

Помітно, що програма долає ускладнення з перекриттям одних об’єктів іншими. За допомогою лінійної динамічної моделі – фільтра Калмана – положення об’єкта прогнозується до того моменту поки він не з’явиться знову. Звісно якщо об’єкт не визначається протягом деякого часу, він видаляється.

Дане програмне забезпечення можна використовувати для ефективного визначення об’єктів у реальному часі. Розглянуто методи і технології для покращення результату роботи алгоритму програми.

5 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

У даному розділі проводиться оцінка основних характеристик програмного продукту, призначеного для виділення та відстеження об’єктів на відео послідовності. Інтерфейс програми був розроблений за допомогою мови програмування C++ у середовищі розробки QtCreator. Інтерфейс користувача створений за допомогою бібліотеки Qt.

Програмний продукт призначено для використання на персональних комп’ютерах під управлінням операційної системи Windows.

Нижче наведено аналіз різних варіантів реалізації модулю з метою вибору оптимальної, з огляду при цьому як на економічні фактори, так і на характеристики продукту, що впливають на продуктивність роботи і на його сумісність з апаратним забезпеченням. Для цього було використано апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) – це технологія, яка дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. Як прямі, так і побічні витрати розподіляються по продуктам та послугам у залежності від потрібних на кожному етапі виробництва обсягів ресурсів. Виконані на цих етапах дії у контексті метода ФВА називаються функціями.

Мета ФВА полягає у забезпеченні правильного розподілу ресурсів, виділених на виробництво продукції або надання послуг, на прямі та непрямі витрати. У даному випадку – аналізу функцій програмного продукту й виявлення усіх витрат на реалізацію цих функцій.

Фактично цей метод працює за таким алгоритмом:

– визначається послідовність функцій, необхідних для виробництва продукту. Спочатку – всі можливі, потім вони розподіляються по двом групам: ті, що впливають на вартість продукту і ті, що не впливають. На цьому ж етапі

оптимізується сама послідовність скороченням кроків, що не впливають на цінність і відповідно витрат.

– для кожної функції визначаються повні річні витрати й кількість робочих часів.

– для кожної функції на основі оцінок попереднього пункту визначається кількісна характеристика джерел витрат.

– після того, як для кожної функції будуть визначені їх джерела витрат, проводиться кінцевий розрахунок витрат на виробництво продукту.

5.1 Постановка задачі техніко-економічного аналізу

У роботі застосовується метод ФВА для проведення техніко-економічний аналізу розробки системи аналізу нелінійних нестаціонарних процесів. Оскільки основні проектні рішення стосуються всієї системи, кожна окрема підсистема має їм задовольняти. Тому фактичний аналіз представляє собою аналіз функцій програмного продукту, призначеного для збору, обробки та проведення аналізу гетероскедастичних процесів в економіці та фінансах.

Відповідно цьому варто обирати і систему показників якості програмного продукту.

Технічні вимоги до продукту наступні:

– програмний продукт повинен функціонувати на персональних комп’ютерах із стандартним набором компонент;

– забезпечувати високу швидкість обробки великих об’ємів даних у реальному часі;

– забезпечувати зручність і простоту взаємодії з користувачем або з розробником програмного забезпечення у випадку використовування його як модуля;

– передбачати мінімальні витрати на впровадження програмного продукту.

5.1.1 Обґрунтування функцій програмного продукту

Головна функція F_0– розробка програмного продукту, який визначає та відстежує об’єкти у відео послідовності. Виходячи з конкретної мети, можна виділити наступні основні функції ПП:

F_1 – вибір засобів для розробки;

F_2 – алгоритм відстеження об’єкта.

F_3 – відображення кінцевого результату

Кожна з основних функцій може мати декілька варіантів реалізації.

Функція F_1:

а) мова програмування С++;

б) Пакет Matlab;

Функція F_2:

а) створення власного алгоритму;

б) використання готових реалізацій.

Функція F_3:

а) відображення тільки кінцевого результату;

б) виведення всіх етапів обробки зображення.

5.1.2 Варіанти реалізації основних функцій

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1). На основі цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій (таблиця 4.1).

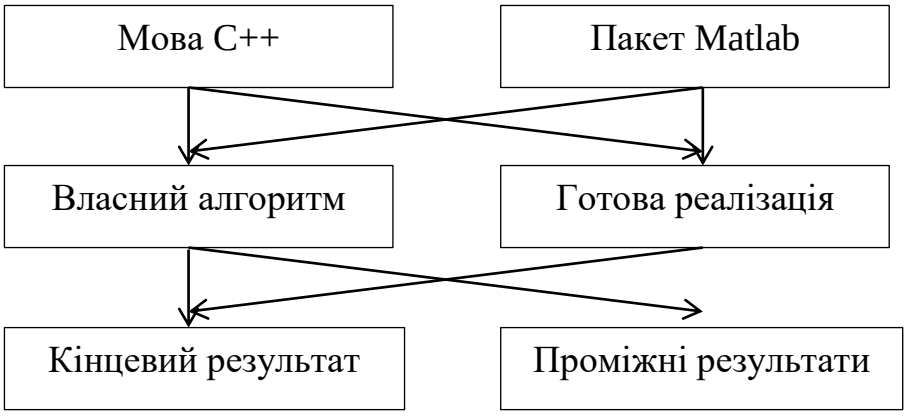

Рисунок 5.1 – Морфологічна карта

Морфологічна карта відображує всі можливі комбінації варіантів реалізації функцій, які складають повну множину варіантів ПП.

Таблиця 5.1 – Позитивно-негативна матриця

Основні функції	**Варіанти реалізації**	**Переваги**	**Недоліки**
F1	*А*	Більша швидкодія Кросплатформений	Займає більше часу при написанні коду
	Б	Простота створення	Низька швидкодія
F2	А	Унікальність	Більше часу на розробку
	Б	Простота створення	Не всі методи оптимізовано
F3	*А*	Більш оптимізована швидкість відображення	Менша наочність процесу
	Б	Більша наочність процесу	Ресурсомісткість

На основі аналізу позитивно-негативної матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Функція *F1:*

Оскільки розрахунки потребують більшої швидкодії, то час виконання є дуже важливим то варіант Б) має бути відкинутий.

Функція *F2:*

Варіант а) та б) вважаємо гідними розгляду

Функція *F3:*

Варіант а) та б) вважаємо гідними розгляду

Таким чином, будемо розглядати такі варіанти реалізації ПП:

1. F1а – F2а – F3а
2. F1а – F2б – F3а
3. F1а – F2а – F3б
4. F1а – F2б – F3б

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

На підставі даних про основні функції, що повинен реалізувати програмний продукт, вимог до нього, визначаються основні параметри виробу, що будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

– *X1* – швидкодія мови програмування;

– *X2* – об’єм оперативної пам’яті, потрібної для роботи програми

– *X3* – Процесорний час зайнятого програмою, %;

– *X4* – потенційний об’єм програмного коду.

5.1.3 Кількісна оцінка параметрів

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у табл. 4.2.

Таблиця 4.2 – Основні параметри ПП

Параметр	Позначення параметра	Гранично допустиме значення	Значення параметра	
			Середнє отримане	Досягаєме значення
Швидкодія мови програмування	**X1**	**19000**	**11000**	**2000**
Об’єм оперативної пам’яті	**X2**	**150**	**50**	**30**
Процесорний час зайнятого програмою, %;	**X3**	**25**	**10**	**5**
Потенційний об’єм програмного коду	**X4**	**2000**	**1500**	**8**

За даними таблиці 5.2 будуються графічні характеристики параметрів – рис. 5.2 – рис. 5.5.

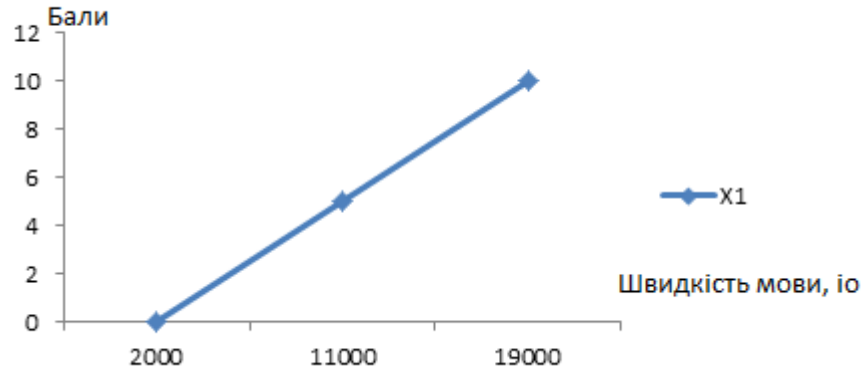

Рисунок 5.2 – X1, швидкодія мови програмування

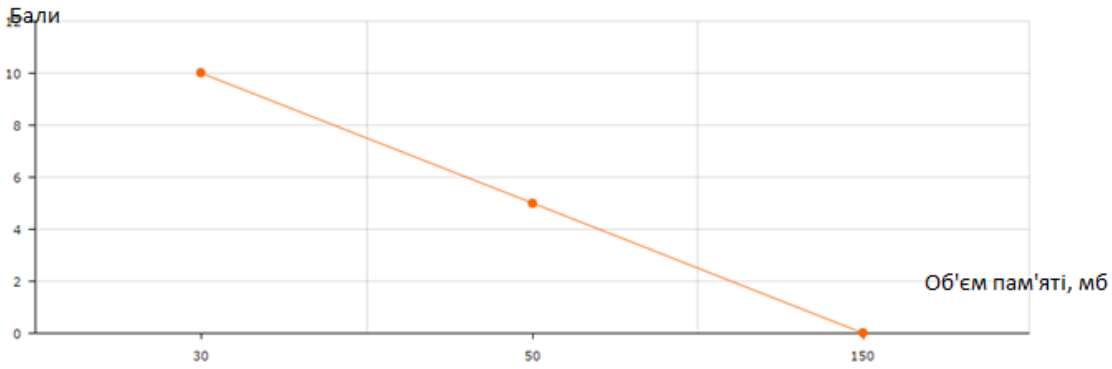

Рисунок 5.3 – X2, Об’єм оперативної пам’яті

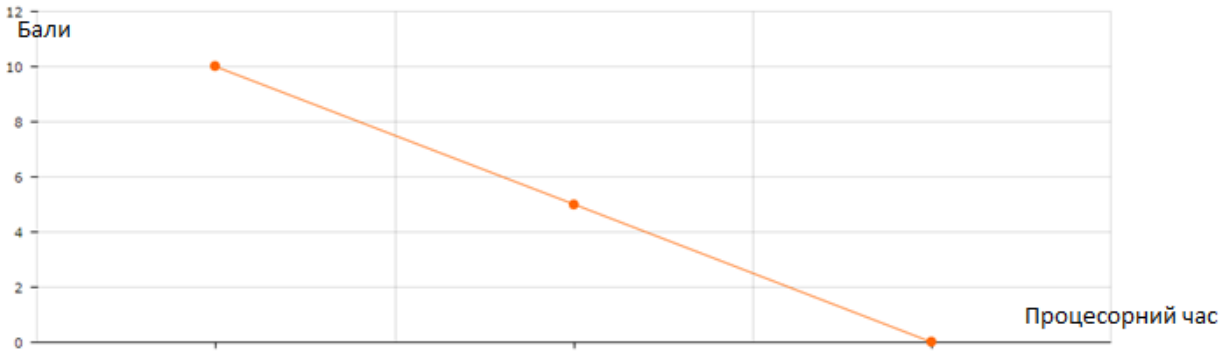

Рисунок 5.4 – X3, Процесорний час зайнятого програмою

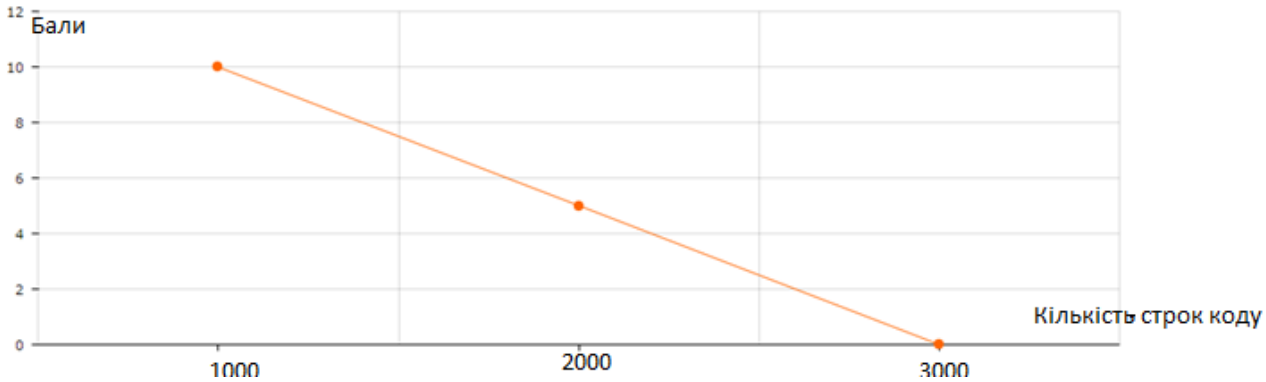

Рисунок 5.5 – X4, потенційний об’єм програмного коду

5.1.4 Аналіз експертного оцінювання параметрів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає найбільш точні результати при знаходженні параметрів моделей адаптивного прогнозування і обчислення прогнозних значень.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

– визначення рівня значимості параметра шляхом присвоєння різних рангів;

– перевірку придатності експертних оцінок для подальшого використання;

– визначення оцінки попарного пріоритету параметрів;

– обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у таблиці 5.3.

Таблиця 5.3 – Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	Оп/мс	4	4	3.5	4	4	3.5	3.5	26,5	-9	81
X2	Об’єм оперативної пам’яті	Мб	2	2.5	2.5	3	2	3	2.5	17	0,5	0,25
X3	Процесорний час зайнятого програмою,;	%	3	3	2.5	2.5	2.5	3	3	19,5	-2	4
X4	Потенційний об’єм програмного коду	кількість строк коду	1	1.5	1.5	1	1	0.5	1	7	10,5	110,25
	Разом		10	10	10	10	10	10	10	70	0	195,5

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70,$$

де N – число експертів, n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 17.5.$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г)загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 195{,}5.$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 195{,}5}{7^2(4^3 - 4)} = 0{,}8 > W_k = 0{,}67$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 5.4.

Таблиця 5.4 – Попарне порівняння параметрів

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	<	<	<	<	<	<	<	<	0,5
X1 і X3	<	<	<	<	<	<	<	<	0,5
X1 і X4	<	<	<	<	<	>	<	>	1,5
X2 і X3	>	>	=	>	>	=	>	>	1,5
X2 і X4	<	<	<	<	<	<	<	<	0.5
X3 і X4	<	<	<	<	<	<	<	<	0.5

З отриманих числових оцінок переваги складемо матрицю А=$\| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$K_{\text{вi}} = \frac{b_i}{\sum_{i=1}^{n} b_i}$,де $b_i = \sum_{i=1}^{N} a_{ij}$.

Відносні оцінки розраховуються декілька разів доти, поки наступні

значення не будуть незначно відрізнятися від попередніх (менше 2%).На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}$,де $b_i' = \sum_{i=1}^{N} a_{ij} b_j$.

Як видно з таблиці 5.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 5.5 – Розрахунок вагомості параметрів

Параметриx_i	Параметриx_j				Перша ітер.		Друга ітер.		Третя ітер	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$
X1	1,0	0,5	0,5	0.5	2.5	0,219	22,25	0,216	100	0,215
X2	1,5	1,0	1.5	0.5	4,5	0,281	27,25	0,282	124,25	0,283
X3	1,5	0.5	1,0	0.5	3.5	0,344	34,25	0,347	156	0,348
X4	1,5	1,5	1.5	1,0	5.5	0,156	14,25	0,155	64,75	0,154
Всього:					16	1	98	1	445	1

Як видно з таблиці, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

5.2 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 5.6):

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j} ,$$

де n – кількість параметрів;$K_{вi}$– коефіцієнт вагомості i–го параметра;B_i – оцінка i–го параметра в балах.

Таблиця 5.6 – Розрахунок показників рівня якості варіантів реалізації

основних функцій ПП

Основна функція	Варіант реалізації	Абсолютне значення параметру	Бальна оцінка параметру	Коефіцієнт вагомості параметру	Коефіцієнт якості
F1(x1)	а)	11000	5	0,215	1,075
F2(x4)	а)	1700	2.4	0,154	0,3696
	б)	1000	10	0,154	1,54
F3(x2,x3)	а)	60	4.5	0,283	1,2735
	б)	20	3.5	0,348	1,218

За даними з таблиці 5.6 за формулою

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}],$$

визначаємо рівень якості кожного з варіантів:

K_{K1} =1.075+0.3696+1.2735= 2,7181

K_{K2} =1.075+0.3696+1.218= 2,6626

K_{K3} =1.075+1.54+1.2735=3,888

K_{K4} =1.075+1.54 +1.218=3,833

Як видно з розрахунків, кращим є третій варіант, для якого коефіцієнт технічного рівня має найбільше значення.

5.3 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка алгоритму програми;

2. Розробка програмної оболонки;

Варіанти 1 і 2 містять додаткове завдання:

3. Написання власного модуля

Варіанти 2 і 4 містять додаткове завдання:

4. Виведення додаткової інформації.

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи В, завдання 3 – до групи Б, задача 4 – до групи Г. За складністю алгоритми, які використовуються в завданні 1 та 3 належать до групи 1; а в завданні 2 та 4 – до групи 3.

Для реалізації завдання 1 та 3використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних, завдання 4 використовує змінну інформацію.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань. Загальна трудомісткість обчислюється як

$$T_O = T_P \cdot K_П \cdot K_{СК} \cdot K_M \cdot K_{СТ} \cdot K_{СТ.М}, \quad (5.1)$$

де T_P – трудомісткість розробки ПП;$K_П$ – поправочний коефіцієнт;$K_{СК}$ – коефіцієнт на складність вхідної інформації; K_M – коефіцієнт рівня мови програмування;$K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм; $K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: T_P =90 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $K_П$ = 1.7. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх завдань рівний 1: $K_{СК}$ = 1. Тоді, за формулою 5.1, загальна трудомісткість програмування першого завдання дорівнює:

$T_1 = 90 \cdot 1.7 \cdot 0.7 = 107.1$ людино-днів.

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни В). Оскільки при розробці другого завдання

використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $К_{СТ}$ = 0.8. тобто $Т_Р$ =64 людино-днів, $К_П$ =0.6,$К_{СК}$ = 1,

$Т_2$ = 27 · 0,6 · 0.8= 12,96 людино-днів.

Для третього завдання використовується ступінь новизни Б, складність 1

$Т_Р$ =64 людино-днів, $К_П$ =1,021,$К_{СК}$ = 1,

$Т_3$ = 64 · 0,1,= 65,34 людино-днів.

Для четвертого завдання використовується ступінь новизни Г, складність 3. $Т_Р$ =8 людино-днів, $К_П$ =0,6,$К_{СК}$ = 1. Оскільки при розробці другого завдання використовуються стандартні модулі, врахуємо це за допомогою коефіцієнта $К_{СТ}$ = 0.8.

$Т_4$= 8 · 0,6*0,8= 3,84людино-днів.

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$Т_I$ = (107.1 + 12.96+65.34) · 8 = 1483.2 людино-годин;

$Т_{II}$ = (107.1 + 12.96 + 65.34+3.84) · 8 = 1513.92 людино-годин;

$Т_{III}$ = (107.1 + 12.96) · 8 = 960.5 людино-годин;

$Т_{IV}$ = (107.1 + 12.96+3.84) · 8 = 991.2 людино-годин;

Найбільш високу трудомісткість має варіант II.

В розробці беруть участь два програмісти з окладом 7000 грн., один фінансовий аналітик з окладом 9500грн. Визначимо зарплату за годину за формулою:

$$С_Ч = \frac{М}{T_m \cdot t} \text{ грн.},$$

де М – місячний оклад працівників;T_m – кількість робочих днів тиждень;t – кількість робочих годин в день.

$$С_ч = \frac{7000 + 7000 + 9500}{3 \cdot 21 \cdot 8} = 46{,}62 \text{ грн.}$$

Тоді, розрахуємо заробітну плату за формулою

$$С_{ЗП} = С_ч \cdot Т_i \cdot К_Д,$$

де $С_ч$– величина погодинної оплати праці програміста;$Т_i$ – трудомісткість відповідного завдання; $К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $С_{ЗП}$ = 46,62· 1483.2 · 1.2 = 82976,1408грн.

II. $С_{ЗП}$ = 46,62· 1513.92 · 1.2 = 84694.74048 грн.

III. $С_{ЗП}$ = 46,62· 960.5 · 1.2 = 53734.212 грн

IV. $С_{ЗП}$ = 46,62· 991.2 · 1.2 = 55451,6928 грн

Відрахування на єдиний соціальний внесок 22%:

I. $С_{ВІД}$ = $С_{ЗП}$ · 0.22 = 18254,75098грн.

II. $С_{ВІД}$ = $С_{ЗП}$ · 0.22 18632,84грн.

III. $С_{ЗП}$ = $С_{ЗП}$ · 0.22= 11821,53грн

IV. $С_{ЗП}$ = $С_{ЗП}$ · 0.22 = 12199,37грн

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного програміста з окладом 7000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$С_Г$ = 12·М·$К_З$ = 12 · 7000· 0,2 = 16800 грн.

З урахуванням додаткової заробітної плати:

$С_{ЗП}$ =$С_Г$· (1+ $К_З$) = 16800· (1 + 0.2) = 20160 грн.

Відрахування на єдиний соціальний внесок:

$С_{ВІД}$= $С_{ЗП}$ · 0.22 = 20160 * 0,22 = 4435,2грн.

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 10000 грн.

$С_А$ = $К_{ТМ}$· $К_А$·$Ц_{ПР}$ = 1.15 · 0.25 · 10000 = 2875 грн.,

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $К_А$– річна норма амортизації; $Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 10000 \cdot 0.05 = 575$ грн.,

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 8 – 16) \cdot 8 \cdot 0.9 = 1706.4$ годин,

де $Д_К$ – календарна кількість днів у році; $Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів; $Д_Р$ – кількість днів планових ремонтів устаткування; t –кількість робочих годин в день; $К_В$– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН} = 1706,4 \cdot 0,156 \cdot 2,0218 \cdot 2 = 1076,4$ грн.,

де $N_С$ – середньо-споживча потужність приладу; $К_З$– коефіцієнтом зайнятості приладу; $Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$С_Н = Ц_{ПР} \cdot 0.67 = 10000 \cdot 0,67 = 6700$ грн.

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н$$

$С_{ЕКС} = 20160 + 4435,2 + 2875 + 575 + 1076,4 + 6700 = 35821,6$грн.

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$С_{М-Г} = С_{ЕКС} / Т_{ЕФ} = 35821,6/1706,4 = 20,99$ грн/час.

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$С_М = С_{М-Г} \cdot Т$$

I. $С_М = 20,99 * 1483 = 31128,17$грн.;

II. $С_М$ = 20,99 *1513= 31757,87грн.;

III. $С_{ЗП}$ = 20,99 · 960= 20150,4 грн

IV. $С_{ЗП}$ = 20,99 · 991= 20801,09 грн

Накладні витрати складають 67% від заробітної плати:

$$С_Н = С_{ЗП} \cdot 0,67$$

I. $С_Н$ = 82976,1408 * 0,67 = 55594,01434грн.;

II. $С_Н$ = 84694,74048* 0,67 = 56745,47612грн.;

III. $С_Н$ = 53734,212 * 0.67= 36001,92204грн

IV. $С_Н$ = 55451,6928 * 0.67 = 37152,63418грн

Отже, вартість розробки ПП за варіантами становить:

$$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н$$

I. $С_{ПП}$ = 82976,1408+ 18254,75+ 18254,75+ 55594= 187953,0761 грн.;

II. $С_{ПП}$ = 84694,74048 + 18632,84291 + 31757,87 + 56745,47612 = 191830,9295 грн.;

III. $С_{ПП}$ = 53734,21 + 11821,526 + 20150,4 + 36001,92204 = 121708,06 грн

IV. $С_{ПП}$ = 55451,69+ 12199,37+ 20801,09+ 37152,63 = 125604,7894 грн

5.4 Вибір кращого варіанта ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$К_{ТЕРj} = К_{Кj} / С_{Фj},$$

$К_{ТЕР1}$ = 2,57 / 181258,59 = 1,44616·10^{-5};

$К_{ТЕР2}$ = 1,89 / 183561,43 = 1,38799·10^{-5};

$К_{ТЕР3}$ = 2,57 / 181258,59 = 3,19453·10^{-5};

$К_{ТЕР4}$ = 1,89 / 183561,43 = 3,05164·10^{-5};

Як бачимо, найбільш ефективним є перший варіант реалізації програми з

коефіцієнтом техніко-економічного рівня $К_{ТЕР3}$= 3,19453·10^{-5}.

5.5 Висновки

В даному розділі проведено повний функціонально-вартісний аналіз ПП, який було розроблено в рамках дипломного проекту. Процес аналізу можна умовно розділити на дві частини.

В першій з них проведено дослідження ПП з технічної точки зору: було визначено основні функції ПП та сформовано множину варіантів їх реалізації; на основі обчислених значень параметрів, а також експертних оцінок їх важливості було обчислено коефіцієнт технічного рівня, який і дав змогу визначити оптимальну з технічної точки зору альтернативу реалізації функцій ПП.

У другій частині ФВА вибрано найбільш економічно обґрунтовану альтернативу реалізації. Порівняння запропонованих варіантів реалізації в рамках даної частини виконувалось за коефіцієнтом ефективності, для обчислення якого були обчислені такі допоміжні параметри, як трудомісткість, витрати на заробітну плату, накладні витрати.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з чотирьох альтернатив оптимальнішою виявилась третя альтернатива. У неї виявився найкращий показник техніко-економічного рівня якості $К_{ТЕР}$ = 3,19453·10^{-5}

Цей варіант реалізації програмного продукту має такі параметри:

– мова програмування – C++;

– використання готових бібліотек;

– виведення тільки кінцевих результатів;

Даний варіант виконання програмного комплексу дає оптимальний результат і швидкодію

ВИСНОВКИ

В результаті виконання роботи була здійснена розробка програми визначення та супроводження об’єктів у відео послідовності. Дана програма може бути використана в системах безпеки, навігації машин, стеження за підозрілою активністю.

За останній час зроблений значний прогрес в області трекінгу об’єктів. Приводячи до значної кількості методів та технологій для визначення та відстеження об’єктів. Головним викликом для дослідників є розробка алгоритму, який би забезпечував результат незалежно від того як послідовність зображень була отримана – з веб камери або ж знято на якісну камеру. Такі відео частіше за все містять високий рівень шуму, неструктуровані, стиснуті.

Було проаналізовано методи та технології для визначення об’єктів. Розглянуто особливості супроводження об’єктів, принципи роботи алгоритмів.

Детально розглянуто засоби і платформи для обробки зображення та пакетів комп’ютерного зору. Наведено практичні рекомендації по роботі з зображенням та у сфері відстеження об’єктів. Проаналізовано ефективність використання бібліотеки OpenCV та пакетів Matlab.

Розроблено алгоритм роботи та архітектуру програми. Було обрано використання мови високого рівня C++ разом з інструментарієм Qt. Для обробки зображень та використання технологій комп’ютерного зору: створення моделі заднього фону, морфологічні операції та фільтр Калмана застосовувалась бібліотека OpenCV.

Тестування програми відбувалося на трьох прикладах, які показали переваги та недоліки роботи розробленого алгоритму. Надійна робота програма відповідає завданню розробки – нерухому камера, криволінійний або прямолінійний рух об’єктів з постійною швидкістю. Похибки визначення були продемонстровані на послідовності зображень з великою інтенсивністю шуму. Розглянуто способи уникнення недоліків і подальшого покращення роботи програми.

Отже, розвиток і тенденції в області комп’ютерного зору дають знаходять застосування в багатьох сферах. А саме визначення і відстеження об’єктів дає змогу аналізувати об’єкти для подальшої роботи з цими даними в залежності від використання.

ПЕРЕЛІК ПОСИЛАНЬ

1. Cavallaro A. Video tracking/ A.,E. Maggio, Cavallaro A. – Chichester, West Sussex, UK: Wiley, 2010.
2. Challa S. Fundamentals of object tracking/Chall S., Mark R. Morelande, Darko Mušicki // Cambridge UK: Cambridge University Press, 2011.
3. Martínez-Martín E. Robust motion detection in real-life scenarios./ A. Pobil, E. Martínez-Martín// – London: Springer, 2012.
4. Yilmaz A. Object tracking: A survey /A. Yilmaz, O. Javed and M. Shah// *ACM Comput. Surv*, 2006, vol. 38, no. 4,
5. Борисенко Д. И. Методы поиска угловых особенностей на изображениях // Молодой ученый, 2011. — №5. Т.1. — С. 120-123.
6. Компьтерное зрение.Современний подход/Форсайт Д., Понс Ж.,:Пер. с англ. – М.:Издательский дом «Вильямс», 2004. – 928с.
7. Цифровая обработка видеоизображений / А. А.Лукьяница, А.Г.Шишкин. – М.: «Ай-Эс-Эс Пресс», 2009. – 518с.
8. Joshi Prateek, Opencv by Example/ Pratek J. – S.l.: Packt Limited, 2016.
9. OpenCV documentation – Режим доступу: http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_video/py_meanshift/py_meanshift.html?highlight=camshift – Дата доступу: 25.05.2016.
10. Matlab documentation – Режим доступу: http://www.mathworks.com/examples/image/mw/images_product-ContrastEnhancementExample-contrast-enhancement-techniques – Дата доступу: 26.06.2016
11. ИНТУИТ – Режим доступу: http://www.intuit.ru/studies/courses/10621/1105/lecture/17985 – Дата доступу: 26.06.2016