МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІМ. Ю.М. ПОТЕБНІ

КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ
АВТОМАТИЗОВАНИХ СИСТЕМ

Кваліфікаційна робота

другий (магістерський)
(рівень вищої освіти)

на тему Дослідження алгоритмів стабілізації відеопотоків

Виконав: студент 2 курсу, групи 8.1210-іпз
спеціальності 121 Інженерія програмного забезпечення
(код і назва спеціальності)
освітньої програми Інженерія програмного забезпечення
(код і назва освітньої програми)
Д. В. Кондратьєв
(ініціали та прізвище)
Керівник доцент, В. І. Попівщий
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)
Рецензент директор ТОВ «АЛЬТЕР ВІЖН ГРУП»
В. С. Тряпічко
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)

Запоріжжя
2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІМ. Ю.М. ПОТЕБНІ

Кафедра _____ програмного забезпечення автоматизованих систем
Рівень вищої освіти _____другий (магістерський)_____
Спеціальність ___121_Інженерія програмного забезпечення_____
(код та назва)
Освітня програма ___Інженерія програмного забезпечення
(код та назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри_ В.Г. Вербицький
"_01_" __вересня_____2021_року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

__Кондратьєву Давіду Віталійовичу__
(прізвище, ім'я, по батькові)
1.Тема роботи __Дослідження алгоритмів стабілізації відеопотоків_

керівник роботи ___Попівщий Василь Іванович, доцент_____
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом ЗНУ від ___"30" червня 2021 року № 974-с_____

2. Строк подання студентом кваліфікаційної роботи_____30.11.2021_____
3. Вихідні дані магістерської роботи
- комплект нормативних документів;
- технічне завдання до роботи.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
- огляд та збір літератури стосовно теми кваліфікаційної роботи;
- огляд та аналіз існуючих рішень та аналогів;
- дослідження алгоритмів стабілізації відеопотоків;
- створення програмного застосунку та його опис;
- перелік вимог для роботи програми;
- дослідження поставленої проблеми та розробка висновків та пропозицій.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
___слайдів презентації

6. Консультанти розділів магістерської роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата
		Завдання прийняв

7. Дата видачі завдання 01.09.2021

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів магістерської роботи	Примітка
1	Аналіз предметної області	02.09-05.09.21	виконано
2	Формулювання основної задачі дипломної роботи та узгодження її з науковим керівником	06.09-07.09.21	виконано
3	Аналіз існуючих методів рішення	08.09-12.09.21	виконано
4	Дослідження алгоритмів стабілізації відеопотоків	13.09-18.09.21	виконано
5	Узгодження подальших дій з науковим керівником	19.09-20.09.21	виконано
6	Аналіз теоретичних відомостей	21.09-25.09.21	виконано
7	Проектування етапів розробки застосунку	26.09-28.09.21	виконано
8	Узгодження етапів розробки застосунку з науковим керівником	29.09-30.09.21	виконано
9	Реалізація функціоналу застосунку для стабілізації відеопотоків	01.10-14.10.21	виконано
10	Представлення отриманих результатів науковому керівнику і узгодження плану подальшого дослідження	15.10-16.10.21	виконано
11	Проведення порівнянь застосунку з готовими програмними продуктами	17.10-23.10.21	виконано
12	Проведення аналізу можливостей розроблених програмних застосунків	24.10-09.11.21	виконано
13	Оформлення звіту	10.11-29.11.21	виконано

Студент _____ Д.В. Кондратьєв
(підпис) (прізвище та ініціали)

Керівник роботи _____ В.І. Попівщий
(підпис) (прізвище та ініціали)

Нормоконтроль пройдено

Нормоконтролер _____ І.А. Скрипник
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Сторінок: 106

Рисунків: 25

Таблиць: 8

Джерел: 38

Кондратьєв Давід Віталійович. Дослідження алгоритмів стабілізації відеопотоку.

Кваліфікаційна робота для здобуття ступеня вищої освіти магістра за спеціальністю 121 «Інженерія програмного забезпечення» наук. керівник В. І. Попівщий. Запоріжжя : ЗНУ, 2021. 106 с.

Мета і завдання дослідження полягають у вивченні існуючих сучасних програмних алгоритмів стабілізації відеопотоку, перевірка їх роботи в різних умовах, пошук оптимального рішення, використовуючи в якості вхідних даних готові файли з відеопотоками.

У процесі дослідження було розглянуто особливості існуючих рішень для програмного стабілізування відео, проаналізовано їх методи аналізування відеопотоку, визначення положення об’єктів. В результаті було розроблено програмний застосунок з реалізацією вибраного алгоритму для стабілізації відеопотоку на базі мови програмування Python, який ефективно компенсує отримані вектори локального руху камери та вирішує поставлену задачу. Окрім цього, було порівняно розроблений застосунок вже з готовими програмними продуктами та онлайн сервісом для стабілізації відеопотоку. Знайдено переваги і недоліки запропонованого підходу для певних умов відеопотоку.

Ключові слова: алгоритм, відеопотік, стабілізація.

SUMMARY

Pages: 106

Figures: 25

Tables: 8

Sources: 38

Kondratiev David. Research of algorithms of stabilization of video streams.

Qualification work for higher master's degree in specialty 121 — Software Engineering, supervison Vasilij Popivshchij. Engineering Educational Scientific Institute of Zaporizhia National University.

The purpose and objectives of the research are to study the existing modern software algorithms for stabilizing the video stream, testing their operation in different conditions, finding the optimal solution, using as input ready-made files with video streams.

In the course of the research the peculiarities of the existing solutions for software stabilization of video were considered, their methods of video stream analysis, determination of the position of objects were analyzed. As a result, a software application was developed with the implementation of the selected algorithm to stabilize the video stream based on the Python programming language, which effectively compensates for the received vectors of local camera movement and solves the problem. In addition, the application was compared with ready-made software products and online service to stabilize the video stream. The advantages and disadvantages of the proposed approach for certain conditions of the video stream are found.

Keyword : algorithm, video, stabilization

ЗМІСТ

АНОТАЦІЯ 4
SUMMARY 5
ЗМІСТ 6
ВСТУП 10
Актуальність теми 10
Мета і завдання дослідження 11
Об’єкт дослідження 11
Предмет дослідження 11
Методи дослідження 11
Наукова новизна одержаних результатів 12
Практичне значення одержаних результатів 12
Глосарій 12
РОЗДІЛ 1 АНАЛІЗ МЕТОДІВ ОЦІНКИ РУХУ ТА СТАБІЛІЗАЦІЇ ВІДЕОПОТОКУ 13
1.1 Аналіз методів стабілізації відеопотоку 13
1.1.1 Механічні та оптичні методи 14
1.1.2 Програмні методи стабілізації 17
1.2Аналіз методів оцінки руху при стабілізації відеопотоків 20
1.1.3 Блочні методи оцінку руху 20
1.1.4 Методи оцінки руху на основі особливих точок 23
1.1.5 Методи оптичного потоку 25
1.2 Аналіз методів компенсації небажаного руху 28
1.2.5 Низькочастотна фільтрація 29
1.2.6 Компенсація руху на основі особливих точок 29

1.2.7 Використання фільтра Калмана 30
1.3 Висновок з розділу 1 31
РОЗДІЛ 2 ЦИФРОВА СТАБІЛІЗАЦІЯ СТАТИЧНИХ І ДИНАМІЧНИХ СЦЕН У ВІДЕОПОТОКІ 34
2.1 Постановка задачі 35
2.2 Стабілізація статичних та динамічних сцен 36
2.2.1 Поділ відеопотоку на сцени 36
2.2.2 Пошук особливих точок 37
2.2.2.1 Властивості особливих точок 37
2.2.3 Детектор кутів 38
2.2.4 Знаходження кутів за допомогою детектора Моравець 40
2.2.5 Знаходження кутів за допомогою детектора Харріса 40
2.2.6 Відстеження руху об’єкту та аналіз його гістограми 43
2.3 Компенсація руху 44
2.4 Відновлення зображення 47
2.4.1 Переорієнтація кадру для відновлення динамічних сцен 48
2.5 Висновок до розділу 2 51
РОЗДІЛ 3 РОЗРОБКА АЛГОРИТМУ СТАБІЛІЗАЦІЇ ВІДЕОПОТОКУ ТА ЕКСПЕРИМЕНТАЛЬНІ РЕЗУЛЬТАТИ 53
3.1 Налаштування середовища для впровадження та розробки алгоритму стабілізації відео 53
3.2 Підготовка до розробки системи стабілізації відео 54
3.2.1 Встановлення середовище розробки IDLE 54
3.2.2 Встановлення бібліотеки OpenCV 55
3.2.3 Пікселі та кольорові простори 56
3.2.4 Обробка і знайомство з зображенням 58

3.2.5 Тестове знайомство з відео 61
3.2.6 Реалізація системи 63
3.2.7 Програмна архітектура 63
3.2.8 Реалізація основного функціоналу 65
3.3 Висновки з розділу 3 69
РОЗДІЛ 4 ДОСЛІДЖЕННЯ РЕЗУЛЬТАТІВ РОБОТИ АЛГОРИТМУ СТАБІЛІЗАЦІЇ ВІДЕОПОТОКУ 70
4.1 Результати експериментальних досліджень роботи алгоритму стабілізації статичних сцен з рухомими об’єктами 70
4.2 Результати експериментальних досліджень роботи алгоритму стабілізації динамічних сцен 74
4.3 Алгоритм стабілізації онлайн сервісу «123apps» 78
4.3.1 Результати експериментальних досліджень роботи алгоритму онлайн-сервісу 123apps для стабілізації статичних сцен 79
4.3.2 Результати експериментальних досліджень роботи алгоритму онлайн-сервісу 123apps для стабілізації динамічних сцен 80
4.4 Алгоритм стабілізації програмного забезпечення для відео редагування Adobe Premiere Pro 82
4.4.1 Результати експериментальних досліджень алгоритму програмного забезпечення Adobe Premiere Pro для стабілізації статичних сцен 83
4.4.2 Результати експериментальних досліджень алгоритму програмного забезпечення Adobe Premiere Pro для стабілізації динамічних сцен 84
4.5 Алгоритм стабілізації програмного забезпечення для відео редагування Movavi Video Editor 86

4.5.1 Результати експериментальних досліджень роботи алгоритму програмного забезпечення Movavi Video Editor для стабілізації статичних сцен з рухомими об’єктами 87

4.5.2 Результати експериментальних досліджень алгоритму програмного забезпечення Movavi Video Editor для стабілізації динамічних сцен 88

4.6 Критерії до порівнянь роботи всіх алгоритмів стабілізації відеопотоку 90

4.6.1 Результати порівнянь розмірів файлів відеопотоку до та після використання алгоритму стабілізації 91

4.6.2 Порівняння часу роботи алгоритмів стабілізації відеопотоку 92

4.6.3 Порівняння вимог для використання алгоритмів стабілізації відеопотоку 94

4.7 Висновки з розділу 4 95

ВИСНОВКИ 97

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 98

ВСТУП

Актуальність теми

Сьогодні відео є однією з невід'ємних інформаційних частин життя людини. Ми весь час переглядаємо якісь відеоролики в таких відомих сервісах як YouTube, Instagram, Tik-Tok. В останні роки ручні відеокамери починають зростати у популярності, дозволяючи кожному легко створювати персональні відеокадри, для цього більше не потрібно мати професійне обладнання, та мати навички роботи з цим обладнанням. Сьогодні у кожного з нас є смартфони в яких є вбудовані камери, які дають змогу робити фото та відео, що дуже сильно спрощує цей процес, незважаючи на те, що на такі пристрої впливають струси та різні механічні вібрації, що призводять до втрати якості відео.

Програмні алгоритми стабілізації відеопотоку якраз і можуть допомогти людині в таких ситуаціях, та ефективно виправити якість відео, і можуть бути впроваджені у різних сферах нашої життєдіяльності. Наприклад, такі алгоритми можна використовувати не лише вже на готових записаних відеороликах, ай в реальному часі, наприклад, як автомобільні регістратори. Такі алгоритми прибирають тремтіння з кадрів, що покращує якість відео, адже без тремтіння відео буде більш плавним, та без різких рухів. А якість зображення буде набагато кращою, що дасть змогу детальніше розрізняти об’єкти.

Сьогодні можна використовувати спеціальне обладнання, яке допоможе зберегти якість відео. Такі пристрої допомагають фізично уникати механічних вібрацій, це можуть бути штативи, або активні стабілізатори, стедіками, але такі методи дуже дорогі, оскільки вони базуються на складних датчиках та процесорах, які вимірюють тремтіння камери, аналізують її рух, та проводять різні програмні обчислення.

Мета і завдання дослідження

Мета і завдання дослідження полягають у вивченні існуючих сучасних програмних алгоритмів стабілізації відео, перевірка їх роботи в різних умовах, пошук оптимального рішення, використовуючи в якості вхідних даних готові файли з відеопотоками.

Для реалізації поставленої мети необхідно виконати наступні завдання:

- Проаналізувати особливості існуючих рішень для програмного стабілізування відео.
- Проаналізувати бібліотеки комп’ютерного зору.
- Вивчити методи визначення положення об’єктів.
- Дослідити методи для аналізування відео:
    - Знаходження особливих точок та їх властивості.
    - Аналіз оптичного потоку.
    - Використання маски (фонового затемнення) для знаходження рухомих об’єктів.
- Виконати програмну реалізацію вибраних алгоритмів стабілізації відео і дослідити їх ефективність.

Об’єкт дослідження

Об’єктом дослідження є програмні алгоритми стабілізації відео.

Предмет дослідження

Предметом дослідження є складність виконання алгоритмів та їх вплив на якість отриманого відеопотоку.

Методи дослідження

При виконанні роботи використовувалися методи теорії обробки інформації, методи аналітичної геометрії, теорія розпізнавання образів, теорія обробки сигналів, методи об'єктно-орієнтованого програмування.

Наукова новизна одержаних результатів

Запропоновано модифікацію алгоритму оптичного потоку, яка показала покращені результати.

Практичне значення одержаних результатів

Практичне значення одержаних результатів дослідження полягає у тому, що було розроблено програмний застосунок, який ілюструє роботу алгоритму стабілізації та ефективніше стабілізує відео з меншими затратами потужностей системи.

Глосарій

Виявлення ключових точок об'єкта (англ. Keypoint detection) — це метод розпізнавання конкретного об'єкту за певними параметрами (різка зміна контрасту, форми тощо).

Глибина кольору (англ. Color Depth) — це визначення у вигляді бітів, яке використовують для представлення кольору одного пікселя растрового зображення.

Оптичний потік (англ. Optical Flow) — це схема видимого руху об'єктів, поверхонь і граней у здоровій сцені, спричинюваного відносним рухом спостерігача та сцени.

NumPy — це бібліотека для підтримки високорівневих математичних функцій, призначених для роботи з багатовимірними масивами.

CMake — це кросплатформна система генерації файлів, необхідних для компіляції програмного забезпечення із вихідного коду.

OpenCV — це відкрита бібліотека комп'ютерного зору, що включає в себе алгоритми для оброки та перетворення зображень, функціонал для роботи з фото та відеопотоками, модулі для розпізнавання, виявлення ключових точок..

РОЗДІЛ 1 АНАЛІЗ МЕТОДІВ ОЦІНКИ РУХУ ТА СТАБІЛІЗАЦІЇ ВІДЕОПОТОКУ

У першому розділі ми оглянемо існуючі методи оцінки руху, відновлення зображень і стабілізації відеопотоку. Наведемо їх класифікації та опишемо порівняльні характеристики методів. Розглянемо функціональні характеристики найбільш відомих систем стабілізації відеопотку.

1.1 Аналіз методів стабілізації відеопотоку

Розрізняють три категорії методів стабілізації відеопотоку: механічні, оптичні і програмні. Класифікація методів стабілізації відеопотоку за категоріями приведена у таблиці 1.1.

Таблиця *1.1*

Класифікація методів стабілізації відеопотоку

Категорії	**Методи**
Механічні методи	Гіроскопи (Gyroscopes) Акселерометри (Accelerometers)
Оптичні методи	Використання призми зі змінним кутом (Vari-Angle Prism) Стабілізатор зображення (Image Stabilizer) Цифровий стабілізатор зображення (Electronic Image Stabilizer)
Програмні методи	Глобальна оцінка руху кадрів (Global motion estimation) Перетворення повного кадру (Full-frame stabilization) Підходи на основі особливостей (Feature base approaches) Згладжування траєкторії руху камери (Optimal camera paths)

Коротко розглянемо основні методи, які використовуються для стабілізації відеокамер при зйомці (механічні та оптичні методи), а також методи цифрової стабілізації відеопотоку (програмні методи).

Особливістю програмних методів є те, що вони мають можливість компенсації не тільки під час зйомки, але і в режимі відеоредагування.

1.1.1 Механічні та оптичні методи

Історично так склалося, що першими методами стабілізації відеопотоку були саме механічні методи. Розрізняють два види механічної стабілізації: це зміщення датчика для протидії руху камери і стабілізація корпусу камери. У першому випадку при обертанні камери, що викликає кутові помилки, гіроскоп управляє механізмом, який переміщує датчик, який поєднує проекцію сцени з площиною зображення [24]. Недоліком цього підходу є необхідність використання ширококутного об'єктиву, оскільки датчик переміщається під час роботи, що знижує ефективність системи.

У другому випадку стабілізується положення корпусу відеокамери. Таким чином нам не потрібно використовувати додаткові об'єктиви або модулі відеокамери, для цього використовуються зовнішні гіроскопи. Прикладом успішної реалізації даного підходу є система «Steadicam». Вона використовує стабілізуюче кріплення камери, яке механічно ізолює рух оператора від камери (дивіться рисунок 1.1), і дозволяє здійснювати плавну зйомку навіть при швидкому русі оператора по нерівній поверхні. Однак, цей механізм є великим і важким і вимагає навичок для досягнення кращих результатів зйомки.

Оптична стабілізація зображення (Optical Image Stabilization, OIS), це спосіб стабілізації оптичної системи відеопотоку, який дозволяє усувати тремтіння

камери до того, як зображення запишеться на ПЗЗ-матрицю (прилад із зарядним зв'язком). Такий спосіб дозволяє отримати відеопоток високої якості.

Рис.*1.1 Використання системи Steadicam*

Основним недоліком оптичної стабілізації є дуже висока вартість в порівнянні зі звичайними камерами, що не дозволяє застосовувати її в камерах попередніх поколінь [10]. Проте, відеокамери з оптичною стабілізацією дуже продуктивні: більшість вібрацій повністю усуваються, корекція відбувається миттєво, зберігається висока якість зображення. До найбільш відомих відноситься система, що має призму зі змінним кутом заломлення (Vari-Angle Prism, VAP), і стабілізатор зображення (Image Stabilization, IS) (дивіться рисунок 1.2).

VAP-система заснована на використанні спеціальної призми зі змінним кутом заломлення. Така призма складається з плоского скла, гнучко з'єднаного зі спеціальною плівкою, яка розширюється і стискається в міру необхідності. Простір між скляними пластинами заповнений силіконовим маслом з високим показником заломлення, тому за допомогою розширення і стиснення цих пластин мо-

жна варіювати кут заломлення. IS - система використовує оптичні лінзи, які зсуваються електронними магнітами перпендикулярно оптичної осі об'єктива так, що при русі об'єктива промені світла, відбиті від об'єкта, змінюють свою траєкторію по відношенню до оптичної осі [17]. Кут заломлення коригується шляхом переміщення групи лінз в площині, перпендикулярній оптичної осі.

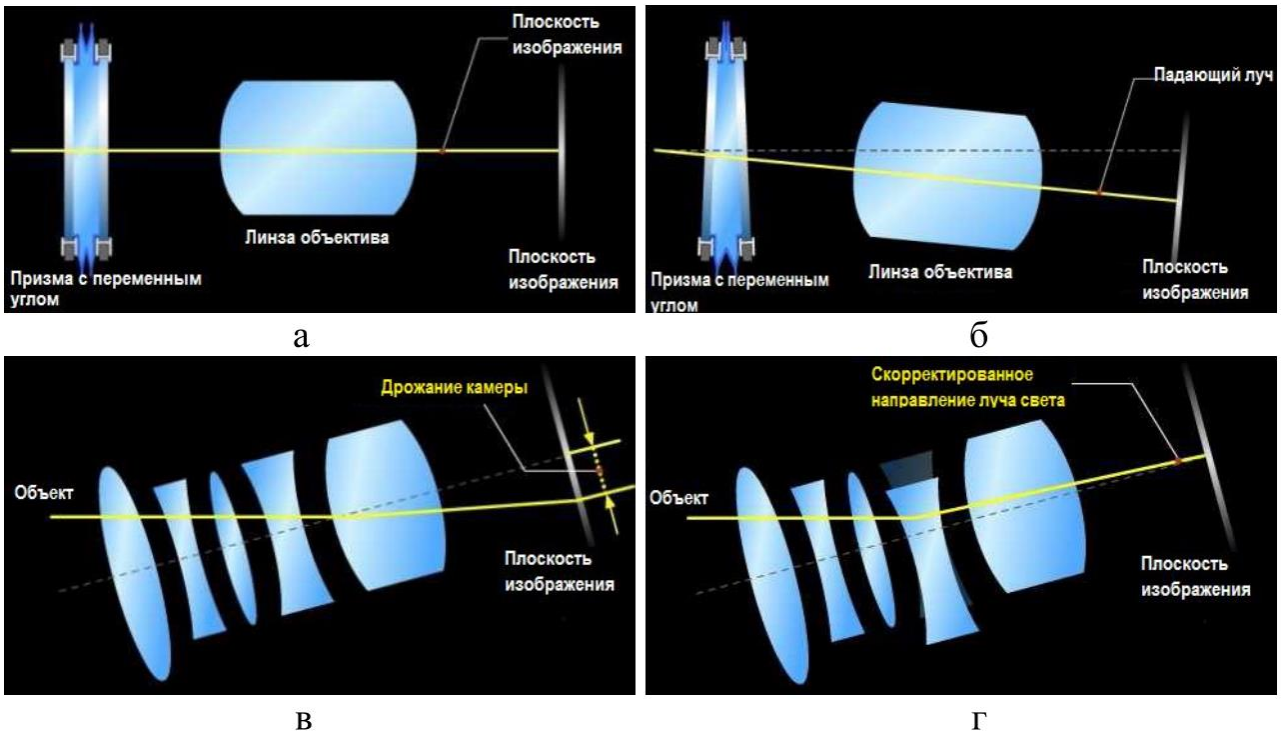


Рис.*1.2 Використання системи Steadicam: а) нерухома призма зі змінним кутом; б) деформація VAP, яку викликала вібрація світового променя; в) стабілізатор зображення IS без стабілізації; г) стабілізатор зображення IS із стабілізацією;*

Електронний стабілізатор зображень (Electronic Image Stabilization, EIS) використовується для контролю стабільності зображення. Якщо датчик системи ви-

являє тремтіння камери при потраплянні світла на ПЗЗ-матрицю, то система реагує переміщенням зображення таким чином, що воно залишається в колишньому положенні. Однак такий результат погіршує якість відеопослідовності, так як область матриці, яка використовується для виведення результуючого зображення, стає менше [24]. Цю проблему можна вирішити або за допомогою ПЗЗ-матриці більшого розміру або цифрового масштабування зображення. Крім того, рух об'єктів викликає розмитість і зернистість зображення, що особливо помітно в областях високої контрастності. Проте, EIS-системи мають перевагу щодо OIS-систем за рахунок зниження складності групи лінзи і ціни.

1.1.2 Програмні методи стабілізації

Застосування програмних методів стабілізації є найменш витратним підходом, що дозволяє виконувати стабілізацію як безпосередньо під час зйомки, так і підвищувати якість вже відзнятих відеоматеріалів. Основними етапами при стабілізації відеопотоку є оцінка руху, компенсація (згладжування) небажаного руху і приведення (відновлення) зображення до постійного стану (табл. 1.2).

Таблиця *1.2*

Етапи програмної стабілізації відеопотоку

Етапи	**Методи**
Оцінка руху	Блочно-порівняльні методи (Block-Matching Algorithm) Методи відстеження особливих точок (Feature Tracking) Методи оптичного потоку (Optical Flow)
Компенсація руху	Низькочастотна фільтрація (Low-pass filtering) Підходи на основі особливостей (Feature base approaches) Відновлення двовимірної сцени (2D scene reconstruction) Фільтр Калмана (Kalman Filtering) Пошук ідеальної траєкторії камери (Optimal camera paths) Відновлення трьовимірної сцени (3D scene reconstruction)
Відновлення зображення	Масштабування зображення (Zoom Image) Перерисовка границь кадру (Motion Inpainting) Перетворення зображення (Image Warping) Текстурні методи (Texture Reconstruction) Інтерполяція кадрів (Frame Interpolation) Переорієнтація кадрів (Video retargeting)

Застосування методів з різних категорій, зазначених в табл. 1.2, дозволяє вирішувати певні завдання стабілізації відеопотоку. Так, комбінація блочно-порівняльних методів, фільтра низьких частот і зміна розміру зображення найбільш часто застосовується для стабілізації відеопотоку статичних сцен [34]. Застосування методів з використання особливих точок дозволяє більш точно відновити плавну траєкторію руху камери [19,30]. Різні методи відновлення зображень були запропоновані в зв'язку з конкретними завданнями стабілізації [28, 33].

Традиційно стабілізація відеопотоку в 2D просторі (в площині зображень) складається з трьох етапів. По-перше, оцінюється модель руху між сусідніми кадрами, наприклад афінне або проекційне перетворення. По-друге, виконується низькочастотна фільтрація параметрів цієї моделі руху за часом. По-третє, використовується конвертування повного кадру на основі отриманих параметрів стабілізації для усунення високочастотного тремтіння камери [18]. Традиційно стабілізація відеопотоку в 2D просторі (в площині зображень) складається з трьох етапів. По-перше, оцінюється модель руху між сусідніми кадрами, наприклад фіне або проектне перетворення. По-друге, виконується низькочастотна фільтрація параметрів цієї моделі руху за часом. По-третє, використовується конвертування повного кадру на основі отриманих параметрів стабілізації для усунення високочастотного тремтіння камери [18]. Стабілізація в 2D-просторі може значно знизити тремтіння камери. Однак при цьому не вдається синтезувати ідеалізований шлях камери подібний до того, який можна виявити при професійній зйомці. При використанні 2D-методів відсутні знання про 3D-траєкторії руху вихідної камери, тому не можна уявити вид результуючої тривимірної траєкторії і оцінити, як сцена виглядала б з її застосуванням. Альтернативним способом є проекційне перетворення, яке являється апроксимацією тривимірної сцени, та використання низькочастотної фільтрації. Низькочастотна фільтрація може призвести до видимих артефактів на відеопотоці, в той час як при незначній фільтрації усувається тільки тремтіння, а згладжена траєкторія руху камери не будується. Розглянемо етапи програмної стабілізації відеопотоків Низькочастотна фільтрація може призвести до видимих артефактів на відеопотоці, в той час як при незначній фільтрації усувається тільки тремтіння, а згладжена траєкторія руху камери не будується. Розглянемо етапи програмної стабілізації відеопотоків більш докладно.

1.2Аналіз методів оцінки руху при стабілізації відеопотоків

Існує велика кількість різних методів оцінки руху відеопотоку. Однак в основному, використовуються такі методи:

- Блочні методи оцінки руху, що відрізняються високою швидкодією і достатньою надійністю при незначних змінах положення об'єктів в сцені за короткий час;
- Методи, засновані на точкових особливості, що дозволяють унікальним чином ідентифікувати особливі точки на зображенні для відстеження положення об'єктів.
- Метод оптичного потоку, що полягає в побудові і вирівнюванні поля векторів руху на основі яскравості інформації сусідніх кадрів.

1.1.3 Блочні методи оцінку руху

Найбільш численним з перерахованих вище груп, є група блочних методів (Block Matching Algorithm, BMA). Це обумовлено універсальністю, невисокою обчислювальною складністю і порівняно високою ефективністю алгоритмів цієї категорії. Не останню роль зіграла також простота їх апаратної реалізації [2, 23].

Схема блочної оцінки руху містить наступну послідовність дій. Зображення ділиться на певні блоки пікселів розмірами $N \times N$ (зазвичай 16х16 пікселів), значення інтенсивності якє визначається як $In(x, y)$, де x, y – координати пікселя, n — номер кадру.

Для кожного блоку в невеликій області $-S_x < d_x < S_{xi} - S_y < d_y < +S_y$, шукають найбільш схожі блоки на наступному кадрі $In + 1(x + d_x, y + d_y)$.

Подібність між блоками визначається мінімізацією функції помилки E, відповідно до використованої метрики [29]. Зазвичай застосовується три метрики

(формули (1.1) — (1.3)): абсолютних різниць (SAD, Sum of Absolute Differences), сума квадратичних відхилень (SSD, Sum of Squared Differences) і середнє значення різниць квадратів (MSD, Mean of Squared Differences).

$$E_{SAD(d_x,d_y)} = \sum_{x=1}^{N} \quad \sum_{y=1}^{N} \quad |I_{n+1}(x,y) - I_n(x+d_x, y+d_x)| \tag{1.1}$$

$$E_{SSD(d_x,d_y)} = \sum_{x=1}^{N} \quad \sum_{y=1}^{N} \quad (I_{n+1}(x,y) - I_n(x+d_x, y+d_x))^2 \tag{1.2}$$

$$E_{MSD(d_x,d_y)} = \frac{1}{B_{num} \times B_{num}} \sum_{x=1}^{N} \quad \sum_{y=1}^{N} \quad (I_{n+1}(x,y) - I_n(x+d_x, y+d_x))^2 \tag{1.3}$$

де B_{num} — кількість аналізованих блоків.

В роботі [3] запропоновано використовувати попередню фільтрацію векторів руху, отриманих за допомогою блочного алгоритму оцінки руху. Для підвищення точності оцінки руху автори пропонують власний критерій розрахунку помилки :

$$E = \frac{\sqrt{e_x^2 + e_x^2}}{const + mNorm};\ e_x = x_s - x_f;\ e_y = y_s - y_f;$$

$$mNorm = \frac{\sqrt{(x_f - x_i)^2 + (y_f - y_i)^2} + \sqrt{(x_s - x_i)^2 + (y_s - y_i)^2}}{2} \tag{1.4}$$

де (x_i, y_i) - координати центру блоку, пов'язаного з вектором руху в кадрі n; (x_f, y_f) - координати центру даного блоку в кадрі $(n+1)$, розрахованого за допомогою BMA; (X_s, Y_s) - координати центру блоку відповідно до афінного моделлю руху; $const$ – заданий параметр.

Ті ж автори у своїй іншій роботі [36] показують, що дана методика розрахунку помилки дає кращі результати, ніж метод найменших квадратів. За рахунок фільтрації неточних векторів руху, авторам вдалося досягти кращих результатів при наявності повороту камери. Отримавши результати досліджень, Puglisi і

Battiato розробили швидкий і точний метод стабілізації відеопотоку, заснований на методі блочних відповідностей. Цей результат був отриманий за рахунок використання методу повного пошуку та інтегрального розрахунку помилкової функції.

В роботі [6] автори пропонують новий метод оцінки руху, заснований на моделюванні критеріїв оцінки подібності блоків з використанням Гауссового розподілу. В цьому випадку оптимізація виконується за допомогою алгоритму максимізації очікування (Expectation Maximization algorithm, EM-алгоритм), заснованого на ітеративній оптимізації параметрів моделі та розширенням відстані Махаланобіса, що застосовується для оцінки відповідності між блоками для пошуку найбільш близьких блоків на сусідніх кадрах. Розглянемо модель Гаусового розподілу:

$$P\left(\frac{x}{\Theta_k}\right) = \sum_{i=1}^{k} i\, p\left(\frac{x}{\Theta_i}\right) = \sum_{i=1}^{k} a_i p\left(\frac{x}{\Theta_k}, \sum_i\right) \tag{1.5}$$

де k є числом компонентів, $(a_i \geq 0)$ пропорції компонентів, що задовольняють умові (1.6) та кожна щільність компонентів $p(\frac{x}{\Theta_i})$ є Функція Гауса ймовірної щільності:

$$\sum_{i=1}^{k} a_i = 1 \tag{1.6}$$

$$p\left(\frac{x}{\mu_i}, \sum_i\right) = \frac{1}{(2\pi)^{m.2}|E_i|^{1/2}} e^{-\left(\frac{1}{2}\right)(x-\mu_i)^{\tau}\sum_i^{-1}(x-\mu)} \tag{1.7}$$

де n – розмір вектора x, u_i – вектор значень, і $\sum_i$ – матриця коваріації, яка позитивно визначена. θ_k – набір з усіх параметрів в сумі, $\theta_k = (\theta_1, \dots, \theta_k, a_1, \dots, a_k)$.

Оцінка відстані між блоками здійснюється на основі значень їх ваги в розподілі. Мінімальна відстань між усіма трьома параметрами моделі відповідає найбільш схожим блоку. Автори провели оцінку розробленого методу при стабі-

лізації відеопотоку з критеріями середньоквадратичної помилки (SSD), абсолютних різниць (SAD), нормалізування взаємної кореляції (NCC, Normalized CrossCorrelation) і визначили, що запропонований критерій відповідності блоків дає найкращий результат за метрикою PSNR (Peak Signal to Noise Ratio). Однак їх метод залежить від існування коваріаційної матриці. На практиці інверсія коваріаційної матриці не завжди можлива, що ускладнює застосування запропонованого методу.

Блочні методи оцінки руху швидко і з достатньою точністю оцінюють зміщення об'єктів між кадрами, що дозволяє виконувати оцінку вектора глобального руху з високою ефективністю [7, 9]. Недоліками даних методів є вплив однорідних областей на результат оцінки руху, а також прив'язка до розміру пошукового блоку. Застосування блокових методів при стабілізації відеопотоку дозволяє здійснювати оцінку руху, але пов'язану з впливом негативних факторів при наявності рухомих об'єктів і зміні освітленості [23]. Крім того, оцінка руху в блоковому методі передбачає вибір моделі руху для подальшої оцінки глобального руху кадру [26, 27, 35].

1.1.4 Методи оцінки руху на основі особливих точок

Альтернативною методикою оцінки руху є пошук особливих точок, які дозволяють знаходити відповідності між положенням точок, кутів і фігур на подібних зображеннях. Алгоритми, що реалізують знаходження особливих точок можуть мати інваріативність до афінних перетворень, змін освітленості, розмиття, що дає можливість застосовувати їх для оцінки руху в складних умовах [17, 22].

У літературі з комп'ютерного зору описані численні точкові оператори, призначені для виділення особливих точок на зображенні. Точкові оператори можуть виділяти як окремі особливі точки в локальній 2D області, так і точки, що лежать на краях. Всі точкові оператори засновані на обчисленні деяких атрибутів і визначенні того, чи перевищують значення цих атрибутів порогові значення чи ні. Атрибути для кожної точки зазвичай обчислюються в невеликій локальній області пікселів. Число виявлених особливих точок залежить від порогового значення. Воно може визначатися адаптивно або встановлюватися заздалегідь. До найбільш відомих точкових детекторів можна віднести кутовий детектор Харріса, SIFT детектор (Scale Invariant Feature Transform) [25, 31], SURF детектор (Speeded Up Robust Feature) [5] і ряд інших.

Алгоритми стеження за особливими точками, в основному, спираються на роботу Б. Лукаса і Т. Канаді [32]. Згодом математична формула алгоритму була змінена, і стала основою для всіх подальших узагальнень з урахуванням афінних перетворень областей і освітленості. Шляхом заміни відповідних змінних на константи будь-який з модифікованих алгоритмів перетворюється в базовий алгоритм Лукаса-Канаді. Будь-який алгоритм знаходження особливих точок можна представити у вигляді такої послідовності дій:

- Знаходження статичного кадру з відеопотоку, на якому присутній вибраний об'єкт (автомобіль, людина і т. д.).
- Накладення сітки певного розміру, наприклад 16x16 пікселів, на зображення з подальшим розрахунком для кожного пікселя в кожному квадраті функції відгуку. При цьому функція відгуку різна для різних методів виділення особливостей.
- Визначення максимальної величини відгуку для кожного квадранта.
- Відбір знайдених точок по заданому граничному значенню.

Нехай J та K - два сусідніх кадра одного відеопотоку. Розглянемо ці два зображення як неперервні функції в двох измерениях. Треба відстежити відоме нам місцезнаходження точки $p_0 = [x_0, y_0]$ картинки J на рисунку K, знайшовши його зміщення $d = \left[d_x, d_y\right]T$. Відмінність K між особливістю на двох сусідніх кадрах з урахуванням області просторового вікна W обчислюється за формулою 1.8.

$$\varepsilon = \int_W [K(p_0) - J(p_0 - d)]^2 d_x d_y \tag{1.8}$$

Потрібно найти зміщення d, яке мінімізує величину E. Якщо зміщення не сходиться до нуля по декількох ітераціях, то особливість цієї точки вважається втраченою.

Знаходження особливих точок використовується в багатьох задачах комп'ютерного зору. Але цей метод потребує багато ресурсів. Його реалізація в режимі реального часу потребує додаткових програмно-апаратних рішень.

1.1.5 Методи оптичного потоку

Методи оптичного потоку представляються собою велику групу методів та їх модифікацій, в основі яких лежить рівняння процесів переносу різних середовищ:

$$\frac{d_f(x,y)}{\partial_t} + V \cdot \nabla f(x, y) = S \tag{1.9}$$

де $\frac{\partial f(x,y)}{\partial t}$ — похідна по часу функції $\frac{f(x,y)}{t}$ в просторовій області; S – змінна яскравості, несвідомих до просторового руху. Просторовий градієнт функції $\frac{f(x,y)}{t}$ знаходиться так:

$$\nabla f(x,y) = \left(\frac{\partial f(x,y)}{\partial x}, \frac{\partial f(x,y)}{\partial y}\right) \tag{1.10}$$

Рівняння (1.9) називається рівнянням оптичного потоку, а під переносним середовищем мається на увазі яскравість зображення $\frac{f(x,y)}{t}$. Задача полягає в знаходженні поля руху векторів на основі відомих знань функції яскравості в двох сусідніх кадрах. Наприклад, за вектор руху можна взяти вектор, який мінімізує праву частину рівняння (1.0) по області мікроблока. Якщо представить розрахунки в матричному вигляди, то можна використовувати тензорну алгебру для розрахунку руху векторів [1,11,14].

Суттєвим недоліком метода полягає в фізичному принципі причинності, згідно якому вплив в кожній точці поточного кадру залежить тільки від переміщень, які появилися з кінцевою швидкістю поширення з обмеженої просторової області та опорного кадру. Як наслідок, при великому модулю русі вектора, формули для похідних повинні включати значення з досить великої області опорного кадру. Збільшення значень $f(x,y)$ в формулах для похідних призводить до зростання обчислювальної складності, і метод перестає бути ефективним. Бажано, щоб модулі векторів руху мали малі значення в межах декількох пікселів. Зазвичай рух в відеопотокі оцінюється шляхом знаходження 3D-структурного тензора просторово-часового обсягу даних $J_S(p)$, який центрований до вектора p, в такий спосіб:

$$J_S(p)=\begin{bmatrix} \int_\Omega \frac{\partial I}{\partial x}\frac{\partial I}{\partial x}dq & \int_\Omega \frac{\partial I}{\partial x}\frac{\partial I}{\partial y}dq & \int_\Omega \frac{\partial I}{\partial x}\frac{\partial I}{\partial t}dq \\ \int_\Omega \frac{\partial I}{\partial y}\frac{\partial I}{\partial x}dq & \int_\Omega \frac{\partial I}{\partial y}\frac{\partial I}{\partial y}dq & \int_\Omega \frac{\partial I}{\partial y}\frac{\partial I}{\partial t}dq \\ \int_\Omega \frac{\partial I}{\partial t}\frac{\partial I}{\partial x}dq & \int_\Omega \frac{\partial I}{\partial t}\frac{\partial I}{\partial y}dq & \int_\Omega \frac{\partial I}{\partial t}\frac{\partial I}{\partial t}dq \end{bmatrix} \quad (1.12)$$

де $I(x, y)$ – функція яскравості зображення, I_x, I_y і I_t - приватні похідні по просторовим осях O_X і O_Y з тимчасової осі відповідно, (p, q) – локальний 3D -об'єм, центрований відносно вектора p, де q – локальна точка.

Найпростішим способом оцінки руху є обчислення сліду $trace(J_S(p))$ матриці $J_S(p)$ і порівняння його з граничним значенням:

$$\text{trace}(J_S(p)) = \int_\Omega ||\nabla I||^2 dq \quad (1.13)$$

Кожен з розглянутих підходів оцінки руху - блочні методи оцінки руху, методи оцінки руху на основі особливих точок, та методи оптичного потоку, - мають свою область застосування. Блочні методи оцінки руху найменш точні, але мають найбільшу швидкодію. Вони можуть бути успішно застосовані для оцінки небажаного руху в статичних сценах. Однак якщо потрібно відновити рухомий об'єкт, то доцільно використовувати метод оптичного потоку, заснованого на градієнтній інформації [12]. За допомогою методів оцінки руху особливих точок можна встановити тип руху складної динамічної сцени і знайти небажаний рух в такій сцені.

У розвиток базового методу оптичного потоку Cai і Walker [8] запропонували алгоритм відеостабілізації на основі вибору особливих точок і delta-оптичного потоку. Під delta-оптичним потоком авторами розуміється сума других похідних в 3D просторі з урахуванням масштабного множника K піраміди Лукаса-Канаді. Такий метод актуальний для швидко рухомих об'єктів, приклад таких об’єктів можна побачити на рисунку 1.3.

Рис.*1.3 Приклад стабілізації відеопотоку: а) (верхній ряд) нестабілізований відеопоток з великим об'єктом в кадрі; б) (нижній ряд) стабілізований відеопоток, з використанням зменшення розмірів кадрів;*

З наведеного прикладу видно, як суттєво можуть змінюватися положення границь кадрів для отримання стабілізованого відеоматеріалу. Вирівнювання границь виконується за допомогою процедури переорієнтації (retargeting) - обрізки і масштабування кадру [29].

1.2 Аналіз методів компенсації небажаного руху

Умовно методи компенсації небажаного руху можна розділити як методи, що оцінюють рух в 2D і 3D сценах. При цьому під 2D стабілізацією розуміється компенсація небажаних тремтінь щодо осей O_Xта O_Y. Для цього відбувається пошук вектор небажаного руху. 3D стабілізація враховує повороти об'єктів не тільки в площині зображення, але і в 3D просторі. При цьому будуються досить складні прогнозовані моделі, як правило, такі методи не передбачають роботу в реальному часі. До методів 2D стабілізації відносяться низькочастотна фільтрація, аналіз особливостей, відновлення 2D сцени, використання фільтра Калмана

в 3D просторі, пошук ідеальної траєкторії відеокамери, відновлення 3D сцени та ряд інших.

1.2.5 Низькочастотна фільтрація

Фільтр низьких частот першого порядку підсумовує рухи міжкадрових векторів для згладжування глобальної траєкторії руху за допомогою простих операцій в реальному часу. Такі фільтри можуть мати як кінцеві, так і нескінченні імпульсні характеристики. Вони добре працюють при невеликих зсувах камери. Однак при значних небажаних зрушення камери згладжена траєкторія руху будується з затримкою, та і в таких випадках потрібна додаткова фільтрація. Проте, низькочастотна фільтрація має широке застосування в цифровій стабілізації відеопотоку. Тому що є можливість аналізу всіх послідовних кадрів оригінальної відеопослідовності.

Модуль коригуючого вектора V_{cor} для n кадрів відеопослідовності розраховується за формулою (1.14) [13]

$$|V_{cor}(n)| = k \cdot |V_{cor}(n-1)| + |V_{inf}(n)| \tag{1.14}$$

де V_{inf} — вектор міжкадрового руху між кадрами n і $(n-1)$, k — компенсуючий коефіцієнт, значення якого вибирається рівним 0,995 при незначних небажаних вібраціях і рівним 0,9 при значних зсувах відеокамери.

1.2.6 Компенсація руху на основі особливих точок

Існує цілий напрям робіт, заснованих на застосуванні фільтра Калмана, коли в якості вимірювань використовуються особливі точки [4, 15, 20, 21]. Ці особливі точки зазвичай виступають як інваріантні ознаки SIFT (Scale Invariant Feature Transform) та SURF (Speeded-Up Robust Feature).

Одним з цікавих напрямків є відеостабілізація на основі SIFT ознак і нечіткої кластеризації. Автори знаходять використовують орієнтацію SIFT-ознак від кадру до кадру, тому цей метод має назву SIFTME (SIFT-Motion Estimation). Вважається, що рух задається афінною моделлю. Для поділу локального і глобального руху в сцені використовується нечітка кластеризація, яка складається з двох кроків. На першому кроці застосовується метод k-середніх. На другому кроці використовуються правила нечіткої логіки, якщо значення k занадто великі (надмірно велика кількість кластерів). Далі застосовується фільтр Калмана для прогнозування глобального руху.

1.2.7 Використання фільтра Калмана

Призначення фільтра Калмана пов'язано з оцінкою стану лінійних динамічних систем у вигляді:

$$S(t+1) = H \times S(t) + w(t) \tag{1.14}$$

де $S(t+1)$ і $S(t)$ — стани системи в моменти часу t і $(t+1)$, H — фільтр Калмана, $w(t)$ — шумова складова, яка зазвичай вважається розподіленою по нормальному закону.

Фільтр Калмана містить дві фази: фазу передбачення (оновлення рівнянь) і фазу коригування (уточнення прогнозованих значень на першій фазі на основі реальних отриманих оцінок), іншими словами, відбувається апріорне і апостеріорне оцінювання. Робота фільтра Калмана докладно представлена в дуже великій кількості джерел для різних застосувань в теорії керування, оцінці динамічних процесів в конкретних сферах людської діяльності, та зокрема, в теорії цифрової обробки зображень для супроводу візуальних об'єктів [37, 38] .

В даному випадку це оцінки, які були отримані в [13] на основі фільтра Калмана, для моделі з постійною швидкістю (вираз (1.15)) та моделі з постійним прискоренням (вираз (1.16)) об'єктів інтересу. Вони мають такий вигляд:

$$V_{cor}(n) = H_{X_{klm}}(n) - V_{X_{inf}}(n) \tag{1.15}$$

$$V_{cor}(n) = H_{klm}(n) - V_{inf}(n) + V_{cor}(n-1) \tag{1.16}$$

де $H_{X_{klm}}$ – складова фільтра Калмана по осі O_X, $V_{X_{inf}}$ – складова міжкадрової різниці по осі O_X, H_{klm} – фільтр Калмана для кадру n.

Фільтр Калмана є дорогим методом компенсації небажаного руху особливо при використанні цього методу у динамічних сценах, коли крім небажаного руху відеокамери є візуальні об'єкти, що рухаються з різною швидкістю і прискоренням. Фільтр Калмана, як і всі методи супроводу, не захищений від зривів при різкій зміні параметрів руху об'єктів або зйомки і в таких випадках потребує повторної ініціалізації.

1.3 Висновок з розділу 1

У першому розділі було розглянуто існуючі методи оцінки і компенсації руху, призначені для вирішення завдання стабілізації відеопотоку.

Методи оцінки руху поділяються на три основні групи, кожна з яких має свої переваги й недоліки. Методи блокової оцінки руху мають високу швидкість роботи в порівнянні з іншими алгоритмами, але більш низькою точністю оцінки. Існують різні модифікації даних методів, що дозволяють уникнути повного перебору області пошуку. Серед метрик, за якими визначається схожість блоків, найбільш відомі такі: сума абсолютних різниць (SAD), сума квадратичних відхилень

(SSD) і середнє значення квадратів різниць (MSD). Методи для розрахунку відповідності блоків є найбільш підходящими для завдання оцінки руху камери, оскільки не потрібно відстежувати зміну конкретних об'єктів.

Методи пошуку особливих точок дозволяють здійснити відстеження положення об'єктів на кількох кадрах, але мають високу обчислювальну складністю. Найбільш важливою відмінністю від інших методів оцінки руху є інваріантність особливих точок до різних змінних значень пікселів. В цілому, методи оцінки особливих точок володіють більш високою точністю для оцінки руху об'єктів в кадрі, але не дають переваги при оцінці глобального руху.

Методи оптичного потоку засновані на знаходженні поля векторів руху на підставі яскравості зображення. Метод оптичного потоку може використовуватися для виявлення рухомих об'єктів при русі камери, але в обчислювальному відношенні він складний і не може бути застосований до повних відеопотоків в реальному часі без спеціалізованих ЕОМ.

Програмні методи стабілізації складаються з трьох етапів: оцінка руху, стабілізація руху і відновлення зображення. Метою етапу оцінки руху є розрахунок вектора глобального руху кадру, при цьому можуть застосовуватися будь-які методи оцінки локального руху. На етапі компенсації руху потрібно розрахувати значення стабілізованого вектора руху. Залежно від наявності статичних або динамічних сцен, необхідно привести положення кадру в відповідність з опорними кадрами відеопотоку, або оцінити траєкторію руху камери і виконати її стабілізацію таким чином, щоб відеопоток виглядав так, як нібито його отримали при професійній зйомці. Для цього застосовується низькочастотна фільтрація, фільтр Калмана або тривимірна оцінка реальної траєкторії руху камери. Останнім етапом стабілізації є відновлення зображення. Більшість систем і алгоритмів пропонують просте масштабування кадру для приховування областей, яких немає на

результативних кадрах після стабілізації. Альтернативними методиками є інтерполяція області зображення на основі опорних кадрів або відокремлення зображення на основі сусідніх пікселів.

Отже було детально проаналізовано існуючі програмні засоби, які можуть вирішити завдання зі стабілізацією відеопотоку з плавним рухом. Було з’ясовано, що програми для відеоредагування орієнтовані на роботу в інтерактивному режимі, їх можливості по автоматичній обробці обмежені, корекція відеопотоку вимагає активної участі оператора.

РОЗДІЛ 2 ЦИФРОВА СТАБІЛІЗАЦІЯ СТАТИЧНИХ І ДИНАМІЧНИХ СЦЕН У ВІДЕОПОТОКІ

Аналіз існуючих підходів та методів стабілізації відеопотоку показав необхідність поділу алгоритмів на статичні і динамічні сцени. Складність завдання полягає в тому, що необхідно враховувати запланований рух камери, алгоритм повинен володіти стійкістю до наявності декількох рухомих об'єктів, зміни освітленості, повороту і зсуву камери. Обмеження, що пред'являються до вхідних відеопотоків, представлені в таблиці 2.1.

Таблиця *2.1*

Обмеження і умови, що пред'являються до вхідних кадрів відеопотоку

Обмеження	**Критерії**
Вид об’єкта	Зображення об’єктів невеликого розміру на передньому плані
Розмір обєктів	Не більше 30% від площі кадру
Тривалість відеопотоку	Без обмежень
Тип руху в сценах	Швидке, повільне
Рух камери	Відсутній, присутній, незначне масштабування

2.1 Постановка задачі

Стабілізацію відеопотоку можна розділити на наступні етапи:

1. Попередня обробка сцени відепотоку
    a. Поділ відеопотоку на сцени.
    b. Оцінка розмитості кадрів і підвищення чіткості розмитих кадрів.
2. Оцінка траєкторії руху:
    a. Оцінка локального руху методами особливих точок.
        i. Використання детектора Моравець для знаходження особливих точок.
        ii. Використання детектора Харріса для знаходження особливих точок.
    b. Побудова гістограми руху об’єкту за допомогою алгоритму Meanshift для уточнення оцінки локального руху.
3. Компенсація руху:
    a. Розрахунок коефіцієнта стабілізації в залежності від оригінального руху камери.
4. Відновлення зображення:
    a. Переорієнтація зображення для динамічних сцен, що містять об'єкти в кадрі які нас цікавлять.

У другому розділі детально розглядаються розроблені методи знаходження і компенсації небажаного руху камери для статичних і динамічних сцен, а також алгоритмічна реалізація кожного з представлених етапів.

2.2 Стабілізація статичних та динамічних сцен

Для статичних сцен оцінка руху виконується на основі методу відповідності блоків, як найбільш підходящого по швидкості і точності роботи. Однак при наявності змін в освітленості сцени і декількох рухомих об'єктів потрібно виконувати більш точну оцінку глобального руху, так як деякі локальні вектори можуть не відображати реальний рух камери, а бути схильними до впливу негативних факторів.

Якість оцінки руху в методах, заснованих на пошуку локальних векторів можна підвищити декількома способами. Для розрахунку параметрів перетворення кадру пропонується використовувати тільки достовірні вектори, які описують саме рух камери, але не об'єктів в кадрі [2]. Для розрахунку глобального вектора руху, застосовується двовимірна лінійна модель, що враховує афінне перетворення.

2.2.1 Поділ відеопотоку на сцени

Для підвищення якості відеоматеріалу спочатку необхідно виконати розподіл відеопотоку на сцени. В такому випадку обробка кожної сцени буде виконуватися незалежно. Це необхідно для виключення впливу кадрів, що належать сусідній сцені, при виконанні стабілізації або використанні тимчасових фільтрів.

Запропонований в роботі [5] алгоритм розподілу відеопотоку на сцени використовує двохпрохідний аналіз відеопослідовності. На першому проході оцінюються різні параметри повної відеопослідовності: такі, як гістограма кадру, конфігурація і кількість особливих точок, розташування колірних боків і інші па-

раметри. На другому проході встановлюються адаптивні пороги параметрів відеопослідовності, при перевищенні яких виконується поділ послідовності на сцени.

2.2.2 Пошук особливих точок

Для отримання з зображення деякої інтерпретуємої інформації необхідно прив’язатися до локальних особливостей зображення. На зображенні можна виділити особливі точки. Особлива точка m — це точка зображення, околиця якої $O(m)$ можна відрізнити від околиці особливої точки $O_2(m)$. В якості околиці на зображенні в більшості алгоритмів береться прямокутне вікно, яке складає розмір 5x5 пікселів. Процес визначення особливих точок досягається шляхом використання детектора і дескриптора.

2.2.2.1 Властивості особливих точок

- Відмінність - особлива точка повинна явно виділятися на тлі і бути відмінною в своїй околиці.
- Інваріантність - визначення особливої точки повинно бути незалежно до афінних перетворень.
- Стабільність - визначення особливої точки повинно бути стійким до шумів і помилок.
- Унікальність - крім локальної відмінності, особлива точка повинна володіти глобальною унікальністю для поліпшення розрізнення повторюваних патернів.
- Повторюваність - особлива точка знаходиться в одному і тому ж місці сцени або об'єкта зображення, незважаючи на зміни точки огляду і освітленості.

- Локальність - особлива точка повинна займати невелику область зображення, щоб зменшити ймовірність чутливості до геометричних і фотометричних знятих в різних точках огляду між двома зображеннями.
- Кількість - число виявлених особливих точок має бути досить великим, так щоб їх вистачило для виявлення навіть невеликих об'єктів. Однак оптимальна кількість особливих точок залежить від предметної області. В ідеалі кількість виявлених особливих точок має адаптивно визначатися з використанням простого і інтуїтивного порога. Щільність розташування особливих точок повинна відображати інформаційний вміст зображення, щоб забезпечити його компактне представлення.
- Точність - виявлені особливі точки повинні точно локалізуватися, як в оригінальному документі, так і взятому в іншому масштабі.
- Ефективність - час виявлення особливих точок на зображенні має бути допустимим в критичних за часом додатках.

2.2.3 Детектор кутів

Кути – це особливі точки, які визначають межу між різними об'єктами і або частинами одного і того ж об'єкта. По-іншому можна сказати, що кути - це точка, в області якої інтенсивність змінюється відносно центру (x, y). Кути визначаються по координатам і змінам яскравості навколишніх точок зображення.

Головна властивість таких точок полягає в тому, що в області навколо кута у градієнта зображення переважають два домінуючих напрямки, що робить їх помітними. Градієнт – це векторна величина, що показує напрямок найшвидшого зростання функції інтенсивності зображення $I(x, y)$. Оскільки зображення дискретне, то вектор градієнта визначається через приватні похідні по осі x і y через інтенсивність змін сусідніх точок зображення.

Залежно від кількості пересічних граней існують різні види куточків: L-, Y- (або T-), і X- зв'язкові. Різні кутові детектори по-різному реагують на кожен з таких видів куточків.

Рис. *2.2.3 Всі види куточків*

Підходи до визначення особливих точок можна розділити на 3 категорії:

1. Попередня Засновані на інтенсивності зображення: особливі точки обчислюються безпосередньо з значень інтенсивності пікселів зображення.

2. Засновані на інтенсивності зображення: особливі точки обчислюються безпосередньо з значень інтенсивності пікселів зображення.

3. Використовують контури зображення: методи витягають контури і шукають місця з максимальним значенням кривизни або роблять полігональну апроксимацію контурів і визначають точку перетину. Ці методи чутливі до околиць перетинів, оскільки витяг часто може бути неправильним в тих місцях, де перетинаються 3 або більше країв.

4. На основі використання моделі: використовуються моделі з інтенсивністю в якості параметрів, які підлаштовуються до зображень-шаблоні. Мають обмежене застосування з особливими точками спеціальних видів (наприклад, L-зв'язковими кутами), залежать від використовуваних шаблонів.

2.2.4 Знаходження кутів за допомогою детектора Моравець

Детектор Моравець - найпростіший з існуючих. Автор розглядає зміну яскравості квадратного вікна W (зазвичай розміру 3x3, 5x5, 7x7 пікселів) відносно цікавої точки при зсуві вікна W на 1 піксель в 8-ми напрямках (горизонтальних, вертикальних і діагональних). Алгоритм:

1. Для кожного пікселя (x, y) на зображенні обчислюється змінна інтенсивності

$$V_{u,v}(x,y) = \sum_{\forall a,b \in w}(I(x+u+a, y+v+b) - I(x+a, y+b))^2,$$

$$(u,v) \in \{(1,0),(1,1),(0,1),(-1,1),(-1,0),(-1,-1),(0,-1),(1,-1)\}. \quad (2.1)$$

2. Будується карта ймовірності знаходження кутів в кожному пікселі (x, y) зображення за допомогою обчислення оціночної функції. Тобто визначається напрямок, якому відповідала би найменша зміна інтенсивності, тому що кут повинен мати суміжні ребра.

3. Відсікаємо пікселі, в яких значення $C(x, y)$ нижче порогового значення T.

Видаляємо повторювані кути за допомогою застосування процедури пошуку локальних максимумів функції відгуку. Всі отримані ненульові елементи карти відповідають кутах на зображенні.

2.2.5 Знаходження кутів за допомогою детектора Харріса

Як показали дослідження, найбільш оптимальним детектором L-зв'язкових кутів є широко відомий детектор Харріса. Харріс і Стефенс поліпшили детектор Моравець, ввівши анізотропію в усіх напрямках, тобто розглядають похідні яскравості зображення для дослідження змін яскравості по безлічі напрямків. Для

цього подивимось на зображення і розглянемо вікно W (зазвичай розмір вікна дорівнює 5x5 пікселів, але може залежати від розміру зображення) в центрі (x, y), а також його зрушення на (u, v).

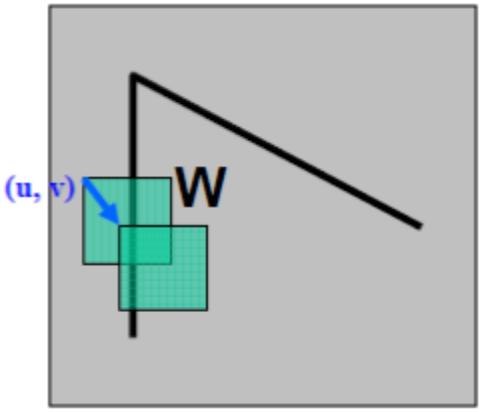


Рис.*2.2.5 Приклад роботи детектора Харріса*

Тоді зважена різниця суми квадрата між зрушеним і вихідним вікном (тобто зміна околиці точки (x, y) при зсуві на (u, v) дорівнює:

$$E(u,v) = \sum_{(x,y)\in W} w(x,y)(I(x+u, y+v) - I(x,y))^2 \approx$$

$$\sum_{(x,y)\in W} w(x,y)(I_x(x,y)u + I_y(x,y)v)^2 \approx (xy)M\begin{pmatrix}x\\y\end{pmatrix} \qquad (2.2)$$

де $w(x, y)$ - вагова функція (зазвичай використовується функція Гаусса або бінарне вікно).

Рис.*2.2.5.1. Вагова функція*

M - автокореляційна матриця:

$$M = \sum_{(u,v)\in W} w(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (2.3)$$

Кут характеризується великими змінами функції $E(x,y)$ по всіх можливих напрямах (x,y), що еквівалентно великим по модулю власним значенням матриці M. Розташування власних значень наведено на наступному малюнку.

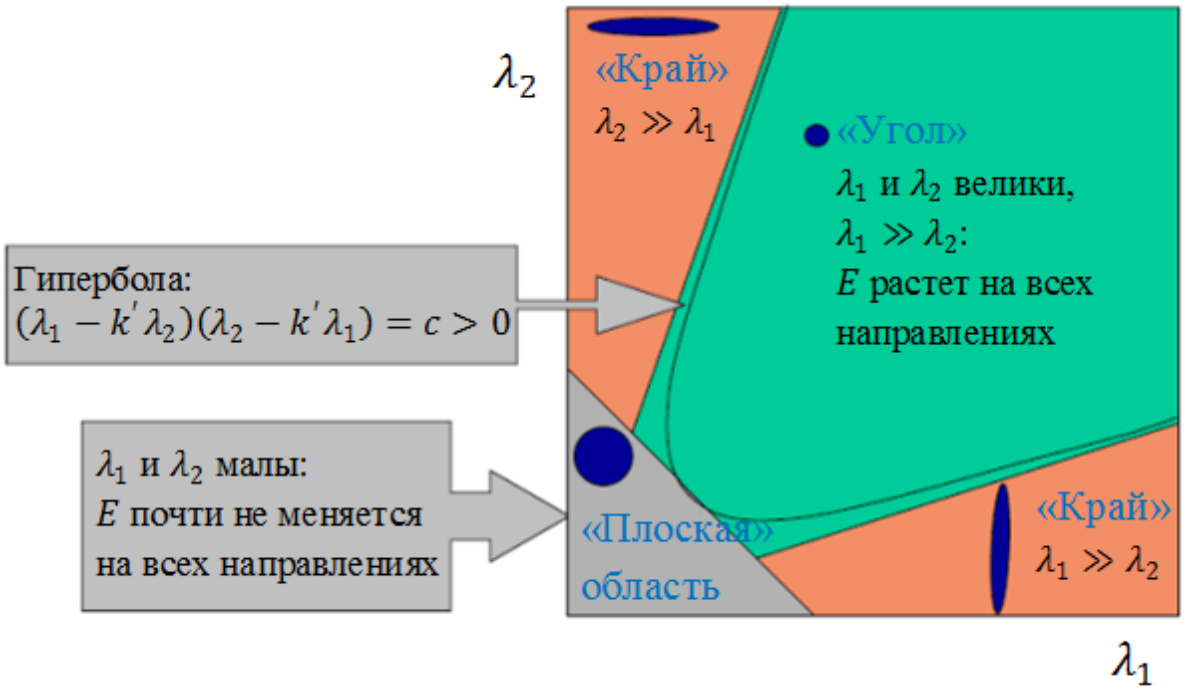


Рис.*2.2.5.2 Розташування власних значень*

Оскільки безпосередньо вважати власні значення є трудомісткою завданням, Харрісом і Стефеном була запропонована міра відгуку:

$$R = detM - k(trM)^2 > k \quad (2.4)$$

де k - емпірична константа, $k \in [0{,}04; 0{,}06]$.

Таким чином, значення R позитивне для кутових особливих точок. Потім проводиться відсікання точок по знайденому порогу R (тобто ті точки, у яких значення R менше деякого порога, виключаються з розгляду). Далі знаходяться локальні максимуми функції відгуку по околиці заданого радіусу і вибираються в якості кутових особливих точок.

2.2.6 Відстеження руху об’єкту та аналіз його гістограми

Представимо що у нас є набір точок. Нам дається невелике вікно (може бути коло) і нам треба перемістити це вікно в область максимальної щільності пікселів (або максимальної кількості точок). Дивіться на рисунку 2.2.6:

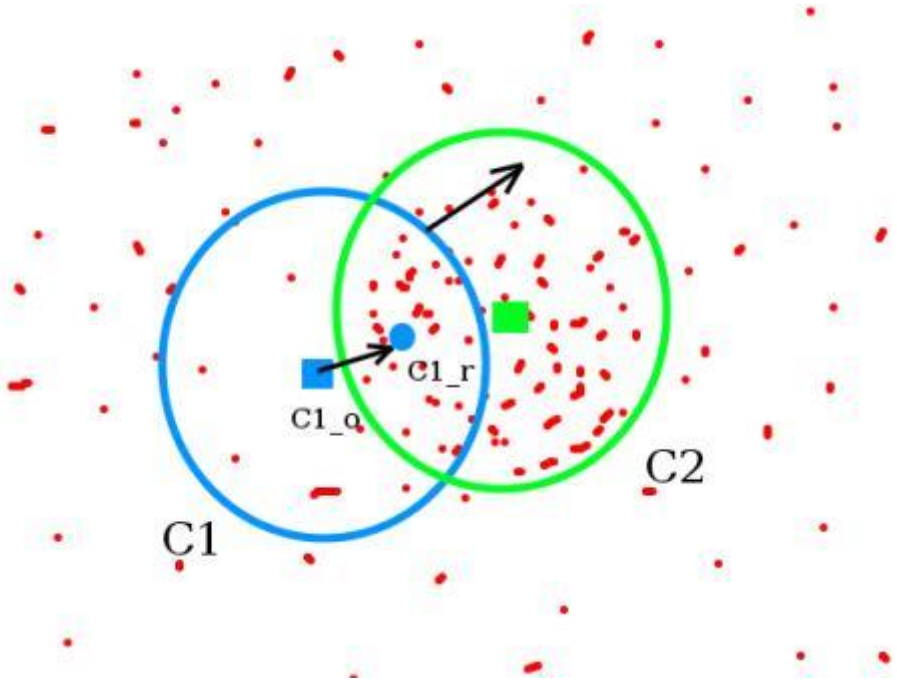


Рис.*2.2.6 Демонстрація вікон*

Початкове вікно показано блакитним колом з назвою "C1". Його початковий центр позначений синім прямокутником, названим "C1_o". Але якщо ви знайдете центроїд точок всередині цього вікна, ви отримаєте точку "C1_r" (позначену маленьким синім колом), яка є справжнім центроїдом вікна. Напевно вони не збігаються.

Тож перемістимо своє вікно так, щоб коло нового вікна збігалося з попереднім центроїдом. Знову знайдемо новий центроїд. Швидше за все, вони не збігатимуться.

Тож перемістимо його ще раз і продовжимо ітерації до тих пір, поки центр вікна та його центроїд потраплять в одне і те ж місце (або в межах невеликої бажаної помилки).

Отже, нарешті, ми отримаємо вікно з максимальним розподілом пікселів. Він позначений зеленим кружком, що має назву «C2». Як ви можете бачити на зображенні, він має максимальну кількість точок.

2.3 Компенсація руху

На етапі компенсації тремтіння необхідно знайти стабілізоване положення кадру на основі відомих глобальних векторів руху. Для цього в разі статичної сцени потрібно вибрати опорні кадри відеопотоку і відносно них провести компенсацію тремтіння в поточному кадрі. Пропонований алгоритм використовує фільтр низьких частот для видалення компонентів які викликають тремтіння[13].

Для динамічних сцен, розрахування вектору глобального руху кадру може складатися з двох основних компонентів: запланований рух (наприклад, панора-

мування камери) і не запланований рух. Якісний алгоритм корекції повинен видаляти тільки незапланований рух, не зачіпаючи при цьому запланований рух камери.

Припускаючи, що незапланований рух відповідає високочастотній компоненті, алгоритм використовує фільтр низьких частот для видалення небажаних моментів руху. Стабілізаційний вектор руху (SMV, Smoothing Motion Vector) отримується за допомогою низькочастотної фільтрації, яка зберігає запланований рух камери. Цей метод обчислює SMV у вигляді рівняння регресії першого порядку за формулою:

$$SMV_n = a \times SMV_{n-1} + (1 - a) \times GMV_n \quad (2.5)$$

де n – номер поточного кадру, $0 \leq a \leq 1$ – варіаційний параметр.

Фільтр низьких частот першого порядку може бути використаний в системах реального часу, його застосування вимагає мало пам'яті. При цьому згладжування руху буде візуально задовільним при виборі відповідного значення α. Параметр α можна розглядати як фактор згладжування: більший коефіцієнт згладжування призводить до плавного руху, але призводить до більшої затримки при запланованому русі камери. Запропоновано адаптивне налаштування параметра α, засноване на величині глобального зсуву камери на попередніх тридцяти кадрах, розраховане за формулою:

$$GDiff_n = \sum_{i=n-30}^{n} |GLV_i - GLV_{i-1}| \quad (2.6)$$

де n - номер поточного кадру, GLV_i – вектор глобального руху кадру i.

В якості вхідних параметрів для нечіткої моделі, яка виконує підбір коефіцієнта згладжування, використовується:

a. сума модулів глобальних векторів руху за попередні 30 кадрів відеопослідовності $GFiff_n$.

b. число екстремумів в функції глобального руху кадру, яке показує інтенсивність тремтіння камери, за останні 30 кадрів відеопослідовності Next.

Чим більші значення приймають вхідні параметри, тим сильніше потрібно стабілізувати відеопоток, а це означає, що потрібно вибирати більше значень для стабілізації коефіцієнта. Для розрахунку чисельного значення застосовується оператор максимуму з двох вхідних параметрів.

На рис. 2.3 представлений графік, що показує рівень вихідного руху аналізованої відеопослідовності, і розрахований при адаптивному підстроюванні параметра α стабілізуючого вектору (SMV_n).

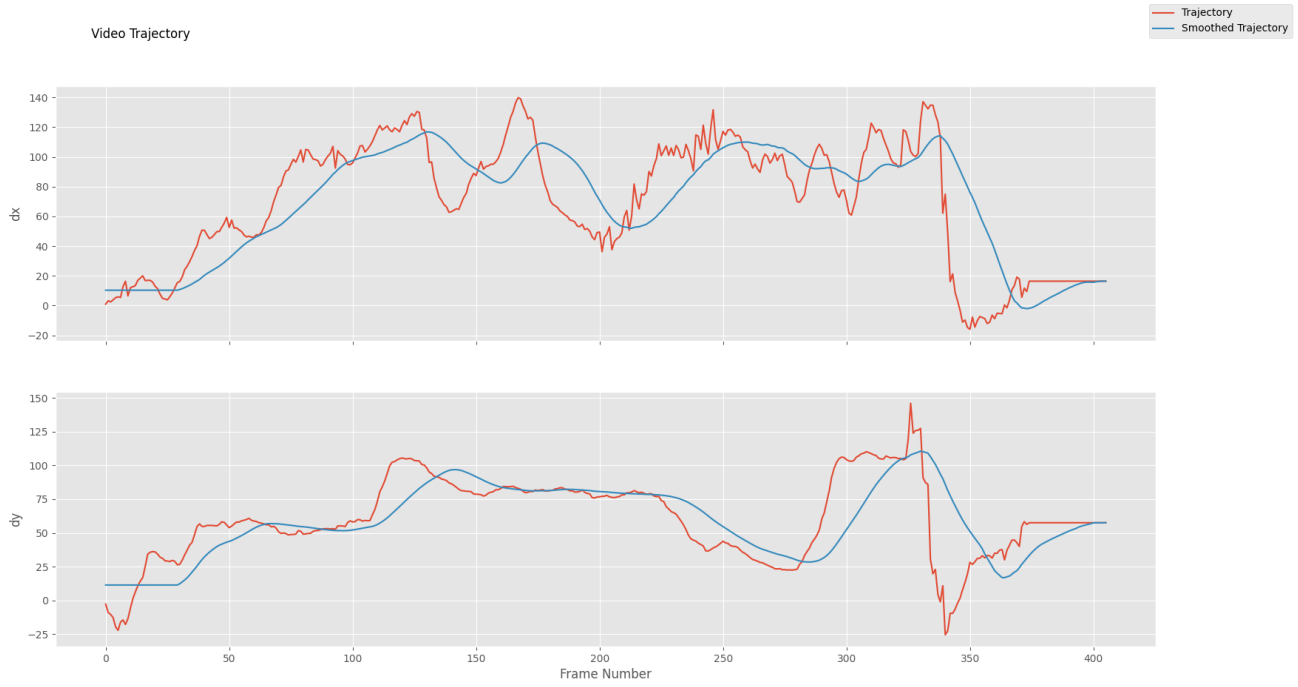


Рис.*2.3 Графік розрахунку стабілізаційного вектора для відеопотоку «train_clip.mp4»*

Після розрахунку стабілізації α, стабілізуючий вектор SMV_n обчислюється за формулою (2.5). Обчислення значення вектора незапланованого руху (UMV_n, Unwanted Motion Vector) здійснюється за формулою:

$$UMV_n = GMV_n - SMV_n \quad (2.7)$$

Для відновлення поточного кадру до його стабілізованого положення, потрібно змінити його позицію на векторі накопичення незапланованого руху (AMV_k, Accumulated Motion Vector), що розраховується за формулою (2.8), оскільки стабілізоване положення кадру визначається на підставі попередніх кадрів, які враховуються з останньої зміни сцени.

$$AMV_n = \sum_{i=kfr-1}^{n} UMV_i \quad (2.8)$$

де $(kfr - 1)$ – номер першого кадру з моменту останньої зміни сцени.

2.4 Відновлення зображення

При стабілізації кадр повинен містити тільки ту частину зображення, яка міститься на всіх послідовних кадрах сцени. Для цього розраховується вектор накопиченого руху (AMV_n), величина якого визначає, яку частину оригінального кадру необхідно виключити при отриманні стабілізованого зображення [10]. Застосування описаного методу представлено на рисунку 2.4.

В даному прикладі вихідна відеопослідовність має розміри кадрів 854×480, а стабілізована – 540×405. Крім необхідності збільшення масштабу зображення, що призводить до погіршення якості, втрати в кадрі складають 27% від розміру зображення [13].

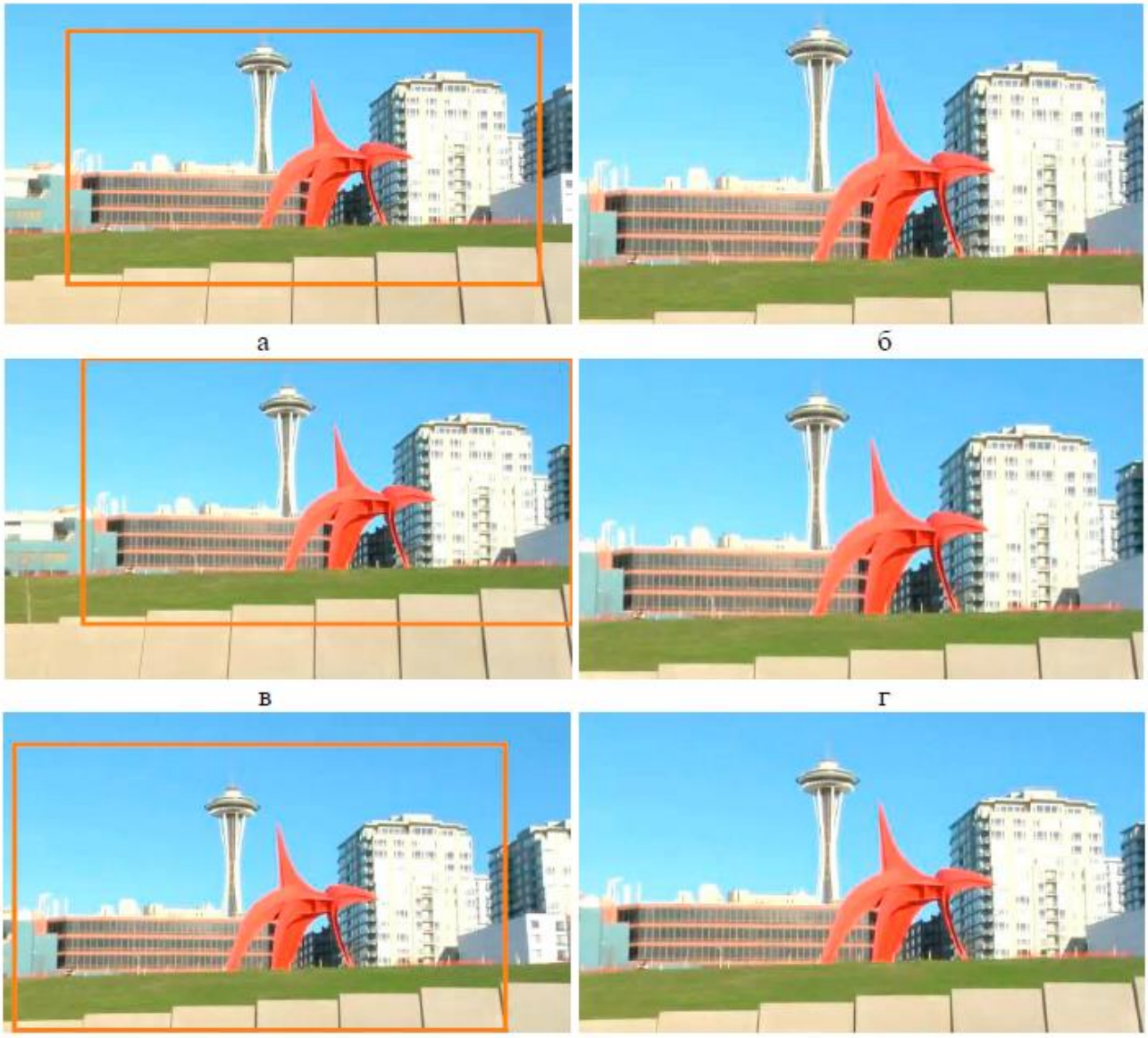


Рис.*2.4 Приклад обрізки країв зображення під час виконання стабілізації «gleicher4.avi»: а), в), д) – оригінальні кадри 1, 8, 19 відеопотоку; б), г), е) – ціж самі кадри після стабілізації відеопотоку;*

2.4.1 Переорієнтація кадру для відновлення динамічних сцен

У динамічних сценах застосування інтерполяції кадрів для відновлення границь зображення часто буває неможливо, оскільки необхідні для відновлення

ділянки кадри можуть не повторюватися на декількох кадрах відеоптоку в зв'язку з швидкою зміною сцени. Тому пропонується альтернативний варіант відновлення зображення, заснований на відстеженні об'єктів в області яка нас інтересує. Пропонується здійснювати переорієнтацію кадру на основі наявності рухомого об'єкту в центрі кадру.

Метою переорієнтації відеопотку є масштабування кадру з урахуванням збереження стабільності, що містять характерні і значущі об'єкти. Останні підходи в цій області засновані на побудові карти глибини об'єктів [5, 24]. Використовуючи фундаментальну матрицю обмежень і провівши кластеризацію на відстежуючих об’єктах, можна розрахувати глибину розташування об'єктів переднього плану в сцені і далі використовувати цю інформацію для переорієнтації кадру. Такий підхід заснований на такому принципі, що глядачі звертають увагу на рухомі об'єкти переднього плану, що є розумним припущенням в межах накладених обмежень.

Застосування переорієнтації кадру дозволяє уникнути артефактів, що виникають на границях кадру при стабілізації відеопотоку динамічних сцен, пов'язаних з розмиванням границь, або використання інформації з попередніх кадрів (рис. 2.4.1).

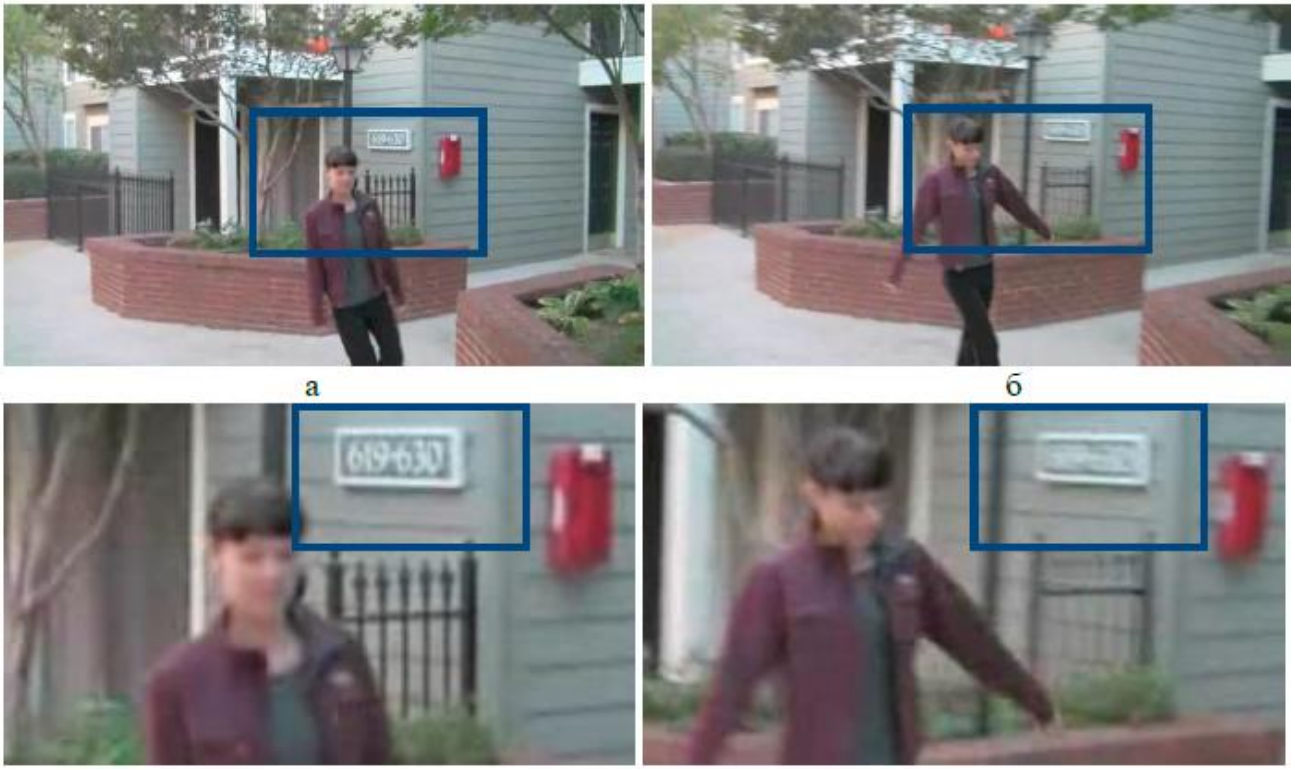


Рис.*2.4.1 Приклад стабілізованої динамічної сцени: а), б), в) – оригінальні кадри 230, 245, 260 відеопотоку «sam1.avi»; г), д), у) – ціж самі кадри при стабілізації с використання алгоритм переорієнтації зображення;*

Алгоритм переорієнтації включає наступні етапи:

1. Оцінка наявності обраного об'єкта в кадрі.
2. Виділення особливих точок для відстеження об'єкта.
3. Розрахунок параметрів масштабування на основі векторів глобального руху останніх 30 кадрів.
4. Застосування масштабування зображення.
5. Переміщення положення об'єкта до центру зображення.

2.5 Висновок до розділу 2

У другому розділі наведені етапи вирішення поставленого завдання.

Для компенсації руху застосовується низькочастотний фільтр першого порядку, що дозволяє розрахувати більш прямий вектор руху при стабілізації відеопотоку. Обчислення вирівнювання вектора проводиться з урахуванням запропонованого параметра вирівнювання, значення якого заснована на інтенсивності тремтіння попередніх 30 кадрів сцени. Адаптивне підстроювання стабілізуючого об’єкта дозволяє синтезувати гладку траєкторію руху камери після стабілізації.

Більшість відомих систем стабілізації використовують метод масштабування зображення, як найбільш простий метод відновлення зображення, проте це призводить до погіршення якості зображення і втрати частини інформації. Розроблено метод відновлення границь кадру в складних статичних сценах при стабілізації відеопотоку на основі інтерполяції опорних кадрів. Введена функція, що дозволяє здійснювати розрахунок інтерпольованого значення пікселя кадру з урахуванням значень пікселів попередніх кадрів і розрахунку рухомих векторів, що вказують на відповідні пікселі. Таким чином, отримане зображення будується з урахуванням зміщення опорних кадрів, а також стабілізованого положення поточного кадру, що дозволяє уникнути артефактів при відновленні кадру.

Для динамічних сцен відновлення границь кадру часто буває неможливо в зв'язку з високою швидкістю руху камери і неможливістю відтворити об’єкти які більше не повторювались. Тому застосовується переорієнтація зображення до обраного об'єкта, що дозволяє зберегти візуальну якість зображення, при цьому підвищивши зручність відстеження обраного об'єкту для оператора. Алгоритм заснований на припущенні, що на відеопотокі присутній об'єкт, що рухається в центрі кадру, положення якого потрібно зберегти. Відстеження обраного об'єкта

здійснюється на підставі особливих точок. Розрахунок параметрів масштабування зображення і переорієнтація зображення виконується з урахуванням глобальних векторів руху попередніх кадрів відеопотоку.

У другому розділі описані методи і алгоритми, розроблені для створення програмної система для стабілізації відеопотоку. Представлений алгоритм стабілізації відеопотоку статичних і динамічних сцен, що дозволяє компенсувати тремтіння камери, що підвищує візуальне якість зображення, а також відновленням границь кадру статичних сцен.

РОЗДІЛ 3 РОЗРОБКА АЛГОРИТМУ СТАБІЛІЗАЦІЇ ВІДЕОПОТОКУ ТА ЕКСПЕРИМЕНТАЛЬНІ РЕЗУЛЬТАТИ

У третьому розділі розглядається практична апробація розробленого алгоритму стабілізації відеопотоку. Розглядається структурна схема розробленого програмного забезпечення і призначення його модулів. Наводяться результати тестування відеопотоку, що містять статичні сцени з рухомими об'єктами та динамічні сцени.

Весь алгоритм можна розділити на декілька модулів, які послідовно виконують свою роботу, це модуль попередньої обробки, оцінки руху, компенсації руху, перетворення кадру, оцінки результатів.

3.1 Налаштування середовища для впровадження та розробки алгоритму стабілізації відео

Першим етапом, необхідним для роботи з відеопотоком, є налаштування робочого середовища. Кінцевим пристроєм для впровадження системи використовувався власний персональний комп’ютер. На якому відбувались наступні етапи роботи: налаштування середовища, розробка алгоритму для стабілізації відеопотоку, тестування роботи системи, проведення досліджень і порівнянь створеної системи із такими готовими системами як онлайн сервіс 123apps, Adobe After Effects 2020, та Movavi Video Editor Plus. Цей персональний комп’ютер був наділений наступними технічними характеристиками:

1. Чотирьох ядерний процесор Intel Xeon E3-1280 з тактовою частотою 3.6 ГГц.
2. Графічний процесор NVIDIA GeForce GTX 1070 8Gb.

3. 16 ГБ оперативної пам’яті.
4. Операційна система Windows 10 версія 20H2.

3.2 Підготовка до розробки системи стабілізації відео

Для початку роботи спочатку необхідно було підготувати середовище для розробки системи стабілізації відеопотоку, отже було виконано наступні кроки:

3.2.1 Встановлення середовище розробки IDLE

Незважаючи на те, що в основі нашої системи за більшість процесів буде відповідати бібліотека OpenCV, яка на даний час являється однією із найкращих і доступних варіантів для роботи із комп’ютерним зором, а вона була написана на мові програмування C++, все ж таки для реалізації роботи системи було вирішено використовувати середовище розробки IDLE (Integrated Development and Learning Environment) для розробки на мові програмування Python.

Перевагами такого кроку стали суттєві зміни при першому налаштуванні системі. Для використання бібліотеки OpenCV на її рідній мові програмування C++, потрібно було провести набагато більше додаткових кроків.

По-перше, це завантаження, встановлення та налаштування середовища розробки, а саме Microsoft Visual Studio 2019, та встановленням додаткових модулів для роботи з мовою програмування C++.

По-друге, самостійно фреймворк Microsoft Visual Studio 2019 не може встановити всі потрібні для нього модулі із бібліотеки OpenCV. Для цього спочатку треба встановити кросплатформову утиліту, для автоматичної збірки програми із

вихідного коду, під назвою CMake. Хоча більш правильніше треба сказати не збірку програми, а лише генеруванням потрібних файлів для збірки програми безпосередньо вже самим середовищем розробки.

Основною перевагою такого метода можна виділити те, що нам непотрібно встановлювати всі модулі бібліотеки OpenCV, а можна вибрати лише ті, які знадобляться нам в майбутньому для стабільної роботи програми. На відміну від утиліти CMake, в середовищі розробки IDLE є внутрішньо влаштована система управління пакетами, під назвою *pip*, яка автоматично може встановити та керувати програмними пакетами. Перевагою цієї утиліти є те, що користувач може легко відкрити каталог програмного забезпечення в *Python Package Index*, та з легкістю знайти потрібний йому модуль і встановити лише однією командою. Таким чином, після встановлення середовища розробки IDLE, для встановлення основних модулів бібліотеки OpenCV можна скористатися командою `pip install opencv-python`, а для встановлення всіх модулів цієї бібліотеки можна скористатися командою `pip install opencv-contrib-python`, що саме й було зроблено.

3.2.2 Встановлення бібліотеки OpenCV

Ми будемо працювати з графікою, отже для цього треба використовувати комп'ютерний зір. Однією із найвідоміших бібліотек, яка працює з комп'ютерним зором являється бібліотека OpenCV.

Це бібліотека з відкритим вихідним кодом розроблялась в центрі розробок програмного забезпечення компанії Intel. Була призначена для аналізу, класифікації та оброки відео-фото зображень, інтерпретація зображень, калібрування камери, усунення оптичних перешкод, визначення подібності, аналіз переміщення

об’єкт, визначення форми об’єкта та стеження за об’єктом, 3D-реконструкція, сегментація об’єкта, розпізнавання жесті тощо. Написана на мові високого рівня програмування C та C++. Також має досить велику популярність завдяки своїй відкритості та можливості користуватися нею безплатно.

3.2.3 Пікселі та кольорові простори

Перед тим як перейти до роботи з бібліотекою комп’ютерного зору OpenCV, спочатку треба розібратись з теорією. Кожне будь-яке зображення складається з пікселів. Піксель — це блок який будує зображення. Якщо ми представимо зображення у вигляді сітки, то кожен квадрат в сітці містить один піксель, де точка з координатою (0,0) відповідає верхньому лівому куту зображення. Наприклад, представимо, що маємо зображення розміром 400×300 пікселів. Це означає, що наша сітка складається з чотирьохсот рядків та трьохсот стовбців. Всього на нашому зображенні міститься 400×300 = пікселів.

В більшості зображень пікселі представлені двома способами: в відтінках сірого та в кольорову просторі RGB.

RGB (абревіатура Red, Green, Blue — червоний, зелений, синій) або ЧЗС — це адаптивна модель кольорового простору, яка описує спосіб кодування кольору для кольоровідтворення за допомогою трьох основних кольорів. Це обумовлено особливостями фізіології сприйняття кольору сітківкою ока.

В відтінках сірого кожен піксель має значення від 0 до 255, де 0 відповідає чорному кольору, та 255 відповідно відповідає білому. А значення між ними, а саме від 0 до 255, відповідає за різні відтінки сірого, де значення ближче до 0 більш темніше, а значення ближче до 255 являється більш світлішим (рис. 3).

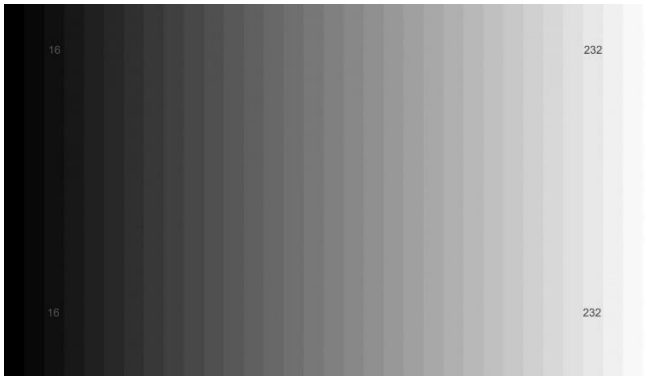


Рис. *3 Глибина кольору*

Кольорові пікселі зазвичай представлені в кольоровому просторі ЧЗС — в якому перше значення відповідає за червоний компонент, другий за зелений, та останній відповідає за синій колір. Кожен із трьох компонентів представлений цілим числом в діапазоні від 0 до 255 включаючи, це число вказує як «багато» кольору знаходиться. Виходячи з того, що кожна компонента представлена у вигляді діапазону від 0 до 255, то для того, щоб представити насиченість кожного кольору нам буде достатньо лише восьми-бітного цілого без знакового числа. Далі ми об’єднуємо значення всіх трьох компонентів і приводимо до одного вигляду. Наприклад, якщо ми хочемо отримати білий колір, то кожна компонента із простору ЧЗС повинна дорівнювати 255, в якому червоний відповідає значенню 255, зелений — 255, синій — 255. А для того, щоб отримати чорний колір, кожен із компонентів ЧЗС повинен дорівнювати 0, в якому червоний відповідає значенню 0, зелений — 255, синій — 255. На рисунку 4 зображено приміри для кожного кольору.

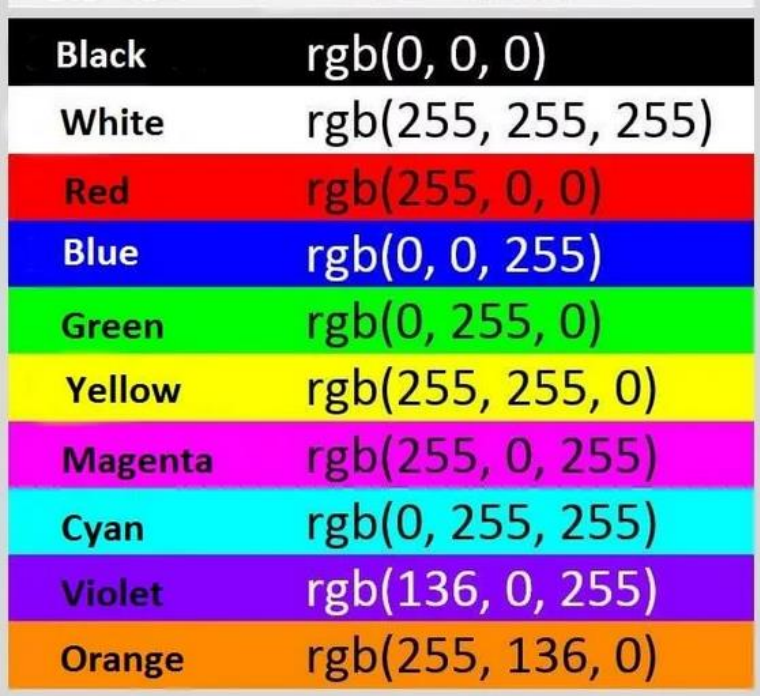


Рис. *4 Графічне кодування ЧЗС кольору*

3.2.4 Обробка і знайомство з зображенням

Трішки розібравшись з теоретичною частиною, можна перейти до практичної частини. Перше що необхідно зробити — це імпортувати бібліотеку наступною командою `(import cv2)`.

Для завантаження зображення ми використовуємо функцію `(cv2.imread())`, де першим аргументом потрібно вказати шлях до зображення, а другим вказати кольоровий простір, в якому ми хочемо зчитати наше зображення, але цей аргумент не являється обов'язковим.

Для зчитування зображення в ЧЗС просторі, треба використовувати модуль – cv2.IMREAD_COLOR, а для відтінків сірого кольору – cv2,IMREAD_GRAYSCALE, дивіться приклад використання на рисунку 5. По замовчу-

ванню цей аргумент приймає значення cv2.IMREAD_COLOR. Дана функція повертає 2D (для зображення в сірих відтінках), або 3D (для кольорового зображення) масив NumPy.

NumPy (Numeric Python) — це модуль з відкритим вихідним кодом який було написано для мови програмування Python. Вона представляє загальні математичні та числові операції у вигляді попередньої компіляції, та швидких функціях. Цей модуль забезпечує функціонал який можна зрівняти з функціоналом MatLab. NumPy надає базові методи для маніпуляцій з великими масивами та матрицями.

SciPy (Scientific Python) — являється додатковим пакетом до модулю NumPy, який розширює його функціонал, та доповнює його додатковими важливими алгоритмами, такими як мінімізація, перетворення Фур'є, регресія, та інші прикладні математичні техніки.

Для кольорового зображення, масив має форму: висота × ширина × 3, де 3 — це байти, по одному байту на кожну із компоненту. В зображення лише із сірими відтінками все набагато легше, там лише масив має форму: висота × ширину.

За допомогою функції `cv.imshow()` можна відобразити зображення на нашому екрані. В якості першого аргументу ми передаємо назву нашого вікна, а другим аргументом буде зображення, яке ми завантажили з диску, але якщо не вказати функцію `cv2.WaitKey()`, то зображення моментально закриється.

Рис. *5 Використання сірого фільтру*

Також, за допомогою функції `cv2.imwrite()` ми можемо записати зображення в файл в відомих форматах зображення, таких як: jpg, png, tiff, jpeg, bmp, тощо.

Для того, щоб визначити висоту, ширину та кількість каналів зображення, можна використовувати атрибут `shape`. Важливо пам’ятати, що зображення у сірих відтінках неможливо буде подивитися на кількість каналів зображення, так як дані цього зображення будуть представлені у вигляді 2D масиву.

Щоб отримати доступ до значення пікселю, треба просто вказати його *X* та *Y* координату. Також, треба пам’ятати, що бібліотека OpenCV зберігає канали ЧЗС в оберненому порядку, коли ми думаємо про червоний, зелений і синій, то OpenCV зберігає їх в порядку синього, зеленого та червоного кольору.

3.2.5 Тестове знайомство з відео

Тепер, коли ознайомлення з функціоналом бібліотеки OpenCV для зображення було завершено, настав час ознайомитися з основами взаємодії цієї системи в роботі з відеопотоком. Насправді, робота з відеопотоком мало чим відрізняється від роботи з зображенням, оскільки відеопоток складається із кадрів, які й являються тими ж самими зображеннями.

Лістинг *1 Підключення бібліотек та тестування роботи з відеопотоками*

```
import cv2
import numpy as np
Cap = cv2.VideoCapture(“test.mp4”)
While True:
Ret, frame = cap.read()
Cv2.imshow(‘video feed’, frame)
If cv2.waitKey(1) & 0xFF == ord(‘q’)
Break
Gray = cv2.cvtColor(frame, cv2.COLOR_BGRA2GARY)
Cv2.imshow(‘gray feed’, gray)
Fourcc = cv2.VideoWriter_fourcc(*’XVID’)
Out = cv2.VideWriter(‘output.avi’, fourcc, 20.0,
(640,480))
```

Лістинг 1 демонтсрує нам підключення бібліотек та перше тестування роботи з відеопотоками. Отже, для початку імпортуємо до нашого проекту потрібні нам бібліотеки `cv2` та `numpy as np`**.** Тепер створюємо об’єкт `VideoCapture`**,** який якраз-таки і відповідає за «захват» відео.

В першому аргументі цієї функції ми вказуємо на назву нашого відеофайлу. Для того, щоб отримувати із джерела кадр за кадром, створюємо цикл

Функція `cap.read()` повертає логічне значення та кадр із зображенням. Якщо кадр було зчитано правильно, то повертається `True`.

Функція `cv2.imshow()` використовується для відображення поточного кадру із відеопотоку.

Оскільки наш цикл являється нескінченним, то нам потрібно добавити щось, що буде його переривати, для цього добавимо наступний метод. Спочатку цей фрагмент коду може показатися дивним, але після того як ми в ньому розберемся, все стану дедалі легше.

Функція `waiKey(0)` повертає значення **-1**, коли введення взагалі не відбувається. Як тільки відбувається якась подія, тобто прикладом можна назвати натискання кнопки, воно повертає 32-розрядне цільове число.

В цьому сценарії `0xFF` — це 8-бітний двійковий код 11111111**,** оскільки він являється представником символу «q». В такому випадку ми отримаємо ціле число менше 255, звідси випливає, що порівнюючи ціле число зі значенням `ord(char)`**,** ми можемо перевірити подію, яка відповідає за натискання на клавішу, та перервати цикл. Пам’ятаєте як ми перетворювали кольорове зображення в лише відтінки сірого? Теж саме можна робити і з відео. Для цього треба добавити декілька незначних змін до нашого циклу. Функція `cv2.cvtColor()` відповідає за конвертацію кольорового простору, та конвертує наше поточне кольорове зображення у відтінки сірого. Тепер ми можемо навіть зберегти наше відео в новому кольоровому просторі, за допомогою модулю `VideoWriter`**.**

Об’єкт `VideoWriter` відповідає за збереження нашого нового відеопотоку. Розглянемо його аргументи, перший аргумент відповідає за назву вихідного

файлу, а саме ми надаємо назву нашому вихідному відеофайлу. Другим аргументом являється на перший погляд невідомий об’єкт `fourcc` — це 4 –байтовий код, який використовується для вказання відеокодеку. Третій аргумент відповідає за поточну кількість кадрів в секунду FPS, і останнім буде розмір вихідного кадра, в нашому випадку це 640 пікселів ширина × на 480 пікселів висоти. Але для того, щоб зберігати наше нове відео, це все треба робити в раніше створеному циклі `while`.

3.2.6 Реалізація системи

На основі сучасних підходів стабілізації відео було вирішено будувати систему на основі створення оптимальних шляхів камери для створення стабілізованого відео шляхом видалення небажаних рухів. Обчислювати траєкторію руху, яка складається з постійних лінійних сегментів, що імітують рухи камери. Алгоритм базується на структурі лінійного програмування щоб мінімізувати похідні які були отримані в результаті побудови 2D сцени руху камери, включаючи додаткові обмеження на шлях камери. Такий підхід дозволяє швидко і легко стабілізувати відео завдяки тому, що він може працювати як пост-процес для відео використовуючи його в фотоапаратах, відеокамерах, та просто використовуючи вже готові відеофайли.

3.2.7 Програмна архітектура

На рисунку 6 зображена діаграма компонентів розробленої системи. Ця система має окремі компоненти які представляють собою модулі.

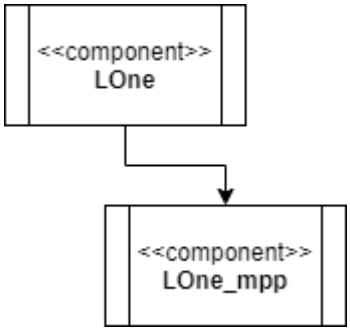


Рис. *6 Діаграма компонентів системи*

Кожен модуль відповідає за певну частину функціоналу системи:

1. Модуль LOne відповідає за наступні етапи:

a. Етап попередньої обробки — на цьому етапі виконується корекція кольору та яскравості зображення, це дозволяє знизити вплив негативних факторів, зв’язаних із змінною освітленості при подальшої обробки відеопотоку. В випадку з динамічною сценою виконується алгоритм для усунення розмиття зображення.

b. Етап оцінки руху — даний етап включає в себе алгоритм ініціалізації параметрів відеопотоку, який дозволяє виконати вибір оброблюваних блоків для поточної сцени. На наступних кадрах оцінка руху виконується тільки для деяких блоків зображення. Після успішного пошуку вектору руху, будується чітка модель для оцінки достовірності вектору руху, який показує, чи являється об’єкт рухомим, чи частиною фону. Для цього виконується кластеризація вектору руху і будується афінна модель руху.

2. Модуль LOne_mpp відповідає за компенсацію та стабілізацію руху, виконується оцінка вже обробленого вектору руху. Будується нечітка модель, яка на основі параметрів руху в використовує послідні тридцять кадрів відеопотоку і

виконує розрахунки параметра що згладжує. Використання такого вектору с адаптивним параметром згладжуваності дозволяє усунути рухи камери в відеопотоку під час динамічних сцен.

Основний функціонал всієї системи реалізований за допомогою лише двох класів.

3.2.8 Реалізація основного функціоналу

Лістинг 2 демонструє стабілізування траєкторії початкового руху зі стабілізованим рухом. Функція `def stabilize` спочатку приймає всю кількість кадрів `n` і перетворює їх у пари. Такі пари являються вхідними даними які повертають стабілізовані параметри траєкторії руху камери, ці стабілізовані параметри є компенсованою версією перетворень `B(t).` Тепер таки перетворення можна застосувати для стабілізації траєкторії.

Лістинг *2 Стабілізування траєкторії руху*

```
    def stabilize(F_transforms, frame_shape,
first_window=True, prev_frame_Bt=None, crop_ratio=0.8):
        prob = lpp.LpProblem("stabilize", lpp.LpMinimize)
        n_frames = len(F_transforms)
        corner_points = get_crop_window(frame_shape,
crop_ratio)
        e1 = lpp.LpVariable.dicts("e1", ((i, j) for i in
range(n_frames) for j in range(N)), lowBound=0.0)
        e2 = lpp.LpVariable.dicts("e2", ((i, j) for i in
range(n_frames) for j in range(N)), lowBound=0.0)
        e3 = lpp.LpVariable.dicts("e3", ((i, j) for i in
range(n_frames) for j in range(N)), lowBound=0.0)
```

```
        p = lpp.LpVariable.dicts("p", ((i, j) for i in
range(n_frames) for j in range(N)))
        prob += w1 * lpp.lpSum([e1[i, j] * c1[j] for i in
range(n_frames) for j in range(N)]) + \
                w2 * lpp.lpSum([e2[i, j] * c2[j] for i in
range(n_frames) for j in range(N)]) + \
                w3 * lpp.lpSum([e3[i, j] * c3[j] for i in
range(n_frames) for j in range(N)])
```

Такі вхідні пари тримають в собі об'єкт за яким ми слідкуємо. Це кадри які позначаються кутовими точками $C_i = (C_{i^x,} C_{i^y,})$.

Лістинг 3 демонструє реалізацію функціоналу по роботі з різними перешкодами руху кадру в відеопотокі. Першою перешкодою для стабілізації відео є приближення та віддалення об'єкту в кадрі, тому щоб нам це не заважало новий шлях камери $P(t)$ повинен зберігати початковий та запланований рух камери. Якщо кадр містить сегменти з приближенням камери, то новий оптимальний компенсований шлях також має слідувати за цим рухом, але вже більш плавніше, для цього треба обмежити наскільки наш $B(t)$ кадр може відхилятися від вихідного шляху, щоб зберегти початковий вигляд відеопотоку, тому на афінну частину параметризації $p(t)$ накладаємо суровімежі: $0.9 \leq a_t, d_t \leq 1.1, -0.1 \leq b_t,$
$c_t \leq 0.1, -0.05 \leq b_c + c_t \leq 0.05, -0.1 \leq a_t - d_t \leq 0.1.$

Перші два обмеження обмежують діапазон для можливих змін кадру, а саме масштабування і обертання, а останні два задають жорсткість для афінного перетворення, обмежуючи нерівномірне масштабування та величину перекосу. Тому в кожному випадку ми маємо верхню (ub) і нижню межу (lb), яка буде відповідною лінійною комбінаціє для $p(t)$, заданої U.

$$lb \leq Up_t \leq ub \tag{7}$$

Лістинг 3 *Функціонал по роботі з перешкодами руху*

```
        for t1 in range(n_frames):
            prob += p[t1, 2] >= 0.9
            prob += p[t1, 2] <= 1.1
            prob += p[t1, 3] >= -0.1
            prob += p[t1, 3] <= 0.1
            prob += p[t1, 4] >= -0.1
            prob += p[t1, 4] <= 0.1
            prob += p[t1, 5] >= 0.9
            prob += p[t1, 5] <= 1.1
            prob += p[t1, 3] + p[t1, 4] >= -0.1
            prob += p[t1, 3] + p[t1, 4] <= 0.1
            prob += p[t1, 2] - p[t1, 5] >= -0.05
            prob += p[t1, 2] - p[t1, 5] <= 0.05
            for (cx, cy) in corner_points:
                prob += p[t1, 0] + p[t1, 2] * cx + p[t1, 3] *
cy >= 0
                prob += p[t1, 0] + p[t1, 2] * cx + p[t1, 3] *
cy <= frame_shape[1]
                prob += p[t1, 1] + p[t1, 4] * cx + p[t1, 5] *
cy >= 0
                prob += p[t1, 1] + p[t1, 4] * cx + p[t1, 5] *
cy <= frame_shape[0]
```

Лістинг 4 демонструє реалізацію отримання між-кадрових перетворень. Для цього першим ділом потрібно знайти той об’єкт, який задовольняє нашим вимогам використавши функцію `cv.goodFeaturesToTrack` та відновивши

шлях руху цієї точки за допомогою оптичного потіку `cv.calcOpticalFlowPyrLK`, який відстежує наші точки на кожному кадрі. Отримавши всі кадри с рухомим об'єктом, будуємо попарно матриці $F(t)$ кадрів. Для афінного перетворення використаємо шести осьову свободу руху. Це можливість фізичного тіла створювати геометричні рухи, а саме рухатися вперед і назад, верх і вниз, вліво та вправо, та здійснювати Ейлерові повороти. Перетворення знаходиться за формулою

$$F_t = A(x; p_t) = \begin{pmatrix} a_t & b_t \\ c_t & d_t \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} dx_t \\ dy_t \end{pmatrix} \tag{3.8}$$

де $p(t)$ являється параметризованим вектором $(dx_t, dy_t, a_t, b_t, c_t, d_t)^T$.

Але так як ми будуємо 2D шлях для нашого відеопотоку, то аналогічним вирішенням будем використати подібність для чотирьох осьової моделі руху, для її розрахунку встановлюємо нові параметри, а саме $a_t = d_t$ та $b_t = -c_t$.

Лістинг 4 *Реалізація отримання між-кадрових перетворень*

```
    def get_inter_frame_transforms(cap, F_transforms,
prev_gray):
        n_frames = F_transforms.shape[0]
        for i in range(n_frames):
            prev_pts = cv.goodFeaturesToTrack(prev_gray,
maxCorners=200, qualityLevel=0.01,
                                              minDistance=30,
blockSize=3)
            success, curr = cap.read()
            if not success:
                break
```

```
            curr_gray = cv.cvtColor(curr, cv.COLOR_BGR2GRAY)
            curr_pts, status, err =
cv.calcOpticalFlowPyrLK(prev_gray, curr_gray, prev_pts, None)
            idx = np.where(status == 1)[0]
            prev_pts = prev_pts[idx]
            curr_pts = curr_pts[idx]
            m, _ = cv.estimateAffine2D(curr_pts, prev_pts)  #
will only work with OpenCV-3 or less
            F_transforms[i + 1, :, :2] = m.T
            prev_gray = curr_gray
        return
```

3.3 Висновки з розділу 3

В третій главі розглядається розроблене експериментальне програмне забезпечення для стабілізації відеопотоку.

Програмне забезпечення має модульну організацію и складається з двох модулів. Модулі які реалізують роботу алгоритмів: попередня обробка кадру, оцінки руху, компенсації руху, перетворення кадру. Було більш детально розглянуто схему функціонування вказаних модулів. Проведено тестування алгоритму в експериментальних дослідженнях для стабілізації відеопотоку, які містять в собі статичні сцени з рухомими об’єктами в кадрі, тремтінням і коливання камери, та змінна освітленості.

РОЗДІЛ 4 ДОСЛІДЖЕННЯ РЕЗУЛЬТАТІВ РОБОТИ АЛГОРИТМУ СТАБІЛІЗАЦІЇ ВІДЕОПОТОКУ

4.1 Результати експериментальних досліджень роботи алгоритму стабілізації статичних сцен з рухомими об’єктами

Для оцінки ефективності алгоритму оцінку руху використовувались відеопотоки наведені у таблиці 3.

Таблиця *3*

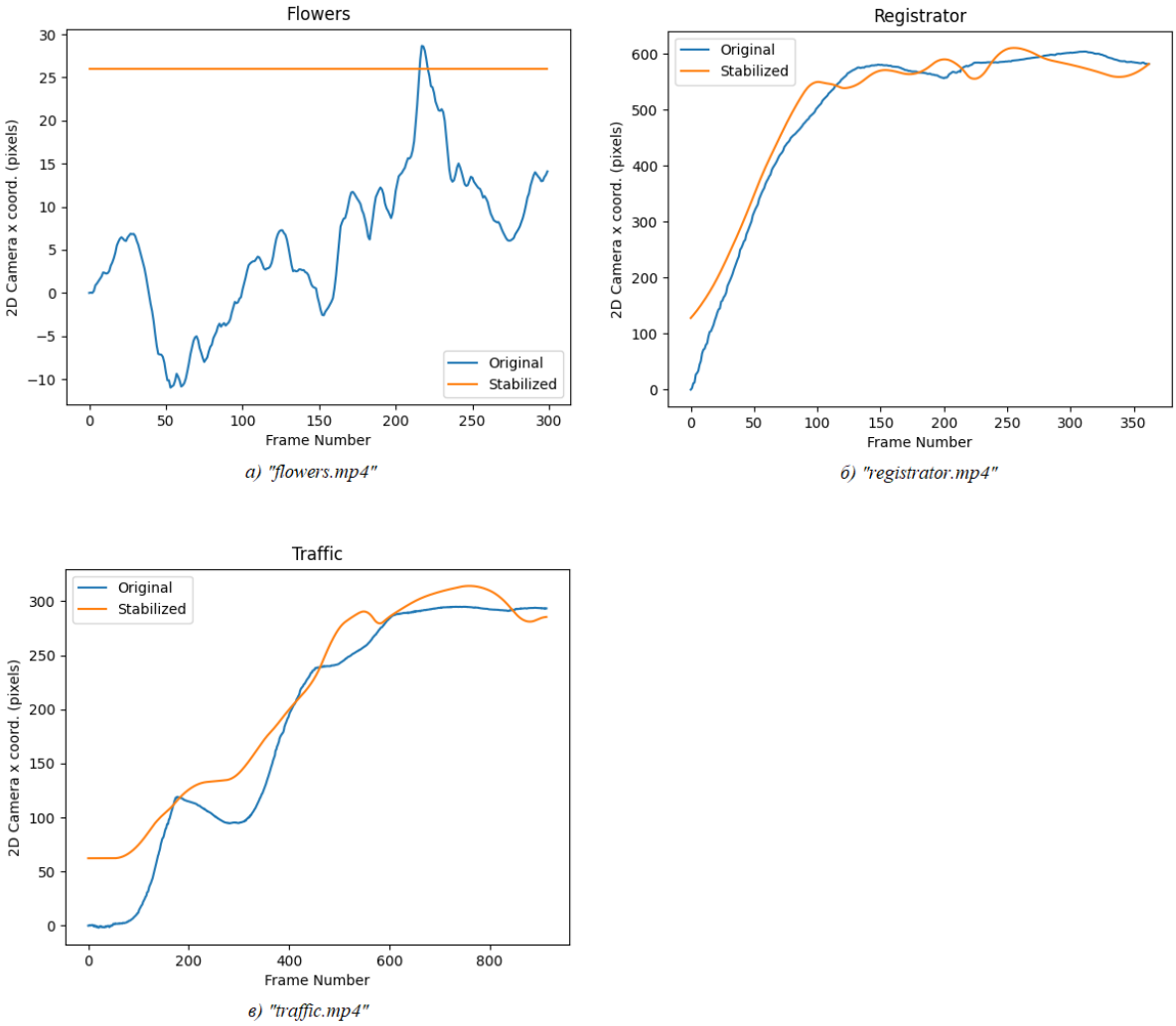
Опис відеопотоку статичних сцен

Назва	**Розширення екрану**	**Кількість кадрів**	**Тип руху**
Bicycle Ride	1280×720	157	Статична камера, вібраційні мікро-коливання камери, рухомі об’єкти
Flowers	720×480	300	Нерухома камера, об’єкти з різкими рухами
Registrator	1280×720	363	Нерухома камера, сильна вібрація, об’єкти з різними видами руху
Traffic	640×360	914	Нерухома камера, повільно рухомі об’єкти

Виконано тестування методу оцінки руху з використання методу видалення небажаних рухів, з мінімізуванням похідних.

При проведені досліджень на відеопотоках які містять на кадрах об’єкти з повільним та швидким рухом, було з’ясовано, що алгоритм, дозволяє вести обробку тільки тих блоків, у яких локальні вектори рухи достовірні і при покращенні ефективності стабілізації відеопотоку зі швидким рухом об’єктів не дає негативного ефекту. При цьому швидкість роботи алгоритму оцінку руху для відеопотоку підвищується на 50-80%, так як для 90% кадрів сцен потребують лише розрахунок локального вектору руху лише на 30-60% блоку кадра. Дослідження відбувались для різних відеопотоків с розширенням екрану від 640×360 до 1280×720, що містять рухомі об’єкти на передньому плані, проективні перетворення, та досить значне тремтіння камери.

Результати досліджень для зміщення локального X представлені в таблиці на рисунку 3.10.

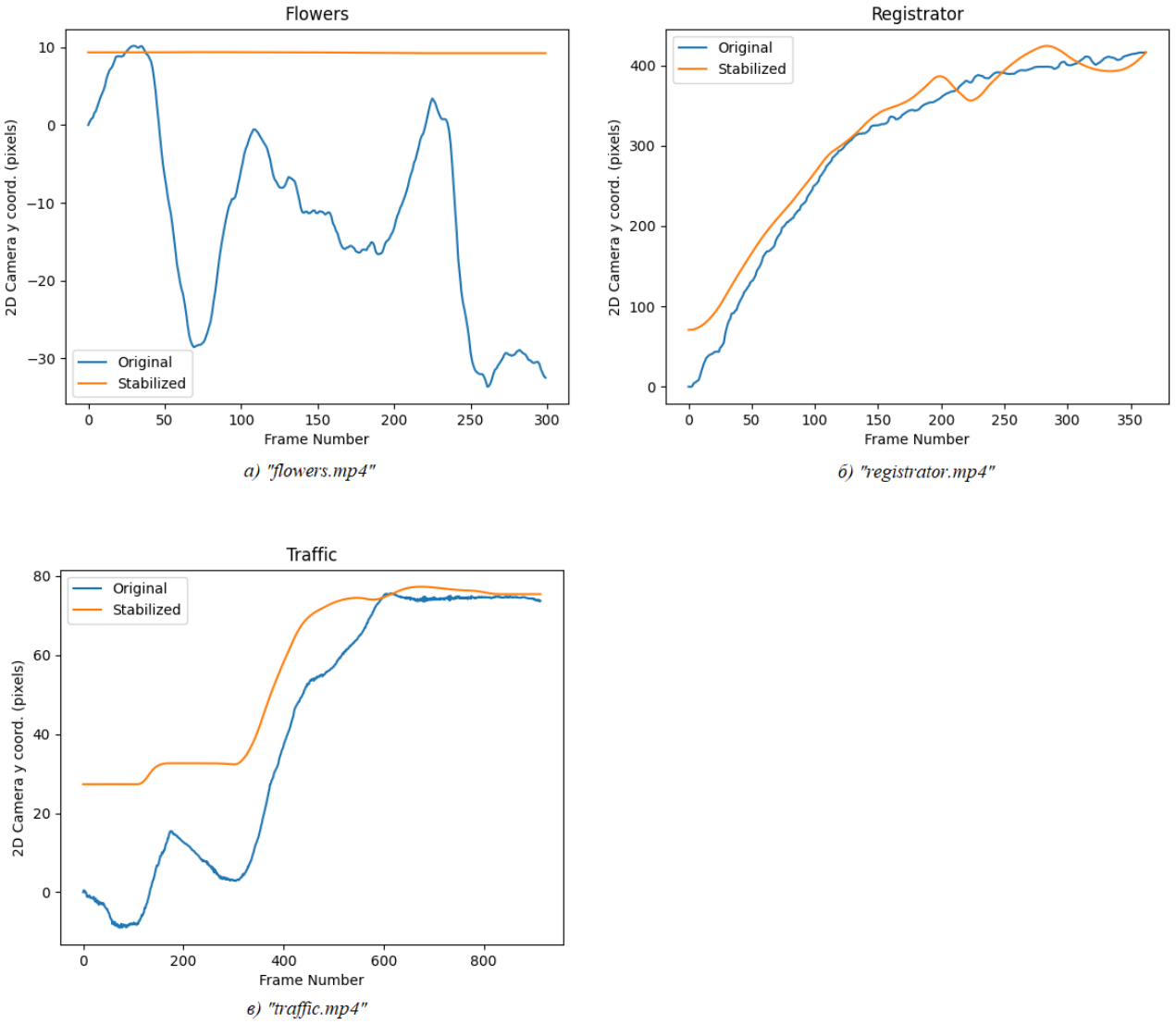


а) "flowers.mp4" *б) "registrator.mp4"*

в) "traffic.mp4"

Рис.*3.10 Результати початкового та стабілізованого X вектору*

Результати досліджень для зміщення локального руху вектору *Y* представлені на рисунку 3.11



а) "flowers.mp4" *б) "registrator.mp4"*

в) "traffic.mp4"

Рис. *3.11 Результати початкового та стабілізованого Y вектору*

Найбільшу свою ефективність алгоритм продемонстрував під час роботи стабілізації відеопотоку в «flowers.mp4», який містить рухомі об’єкти на передньому кадрі, при цьому ще й має середнє розширення екрану. Це дозволяє об’єднати велике число векторів в рухомому блоці, які вказують на задній фон зображення або об’єкт переднього плану, і не робити обробку більшості пікселів

вектору руху які не дозволяють провести оцінку глобального руху камери, наприклад таких, де однорідний фон займає більшу частину відеопотоку. На основі отриманих локальних векторів руху розраховується компенсаційний вектор руху. Значення компенсованого вектору руху залежить від точності оцінки локального вектору руху. Вибір векторів руху, лише тих які описують зміщення камери, дозволяють покращити точність розрахунків компенсованого вектору руху, що покращує якість стабілізації відеопотоку.

Було проаналізовано чотири відеопотока, які містять в собі статичні сцени довжиною більше ніж 1500 кадрів. На рисунку 3.10 та 3.11 представлені результати стабілізації відеопотоку із статичними сцена з використанням видалення небажаних рухів.

Для відеопотоку з статичними сценами оцінка руху с використанням нечіткої моделі більш точно відображає глобальний рух кадру, оскільки він не враховує рух об’єктів переднього плану.

4.2 Результати експериментальних досліджень роботи алгоритму стабілізації динамічних сцен

Для оцінки якості стабілізації динамічних сцен проводилась для семи відеофайлів наведених у таблиці 4, які містять різкі рухи та тремтіння камери.

Таблиця *4*

Опис відеопотоків з динамічними сценами

Назва	**Розширення екрану**	**Кількість кадрів**	**Тип руху**
C63	1920×1080	138	Різки рухи камери, зміна ракурсу, складна об’ємна сцена, швидко рухомі об’єкти
Dane Green Man	1280×720	379	Рух камери, рухомі об’єкти
MTB	1280×720	848	Різкі рухи камери, інтенсивні вібрації, динамічна складна сцена
Parkour	1920×1080	86	Різкі рухи камери, різкі рухи об’єктів
Train	854×480	376	Рухи камери, плавні рухи об’єкту
Walk in NY	1920×1080	511	Різкі рухи камери, велика кількість об’єктів
Walk in Park	3840×2160	323	Рухи камери, зміна ракурсу, велика кількість об’єктів

На рисунку 3.12. представлені результати оцінки початкового і компенсованого руху для відеопотоків з динамічними сценами по *X* вектору.

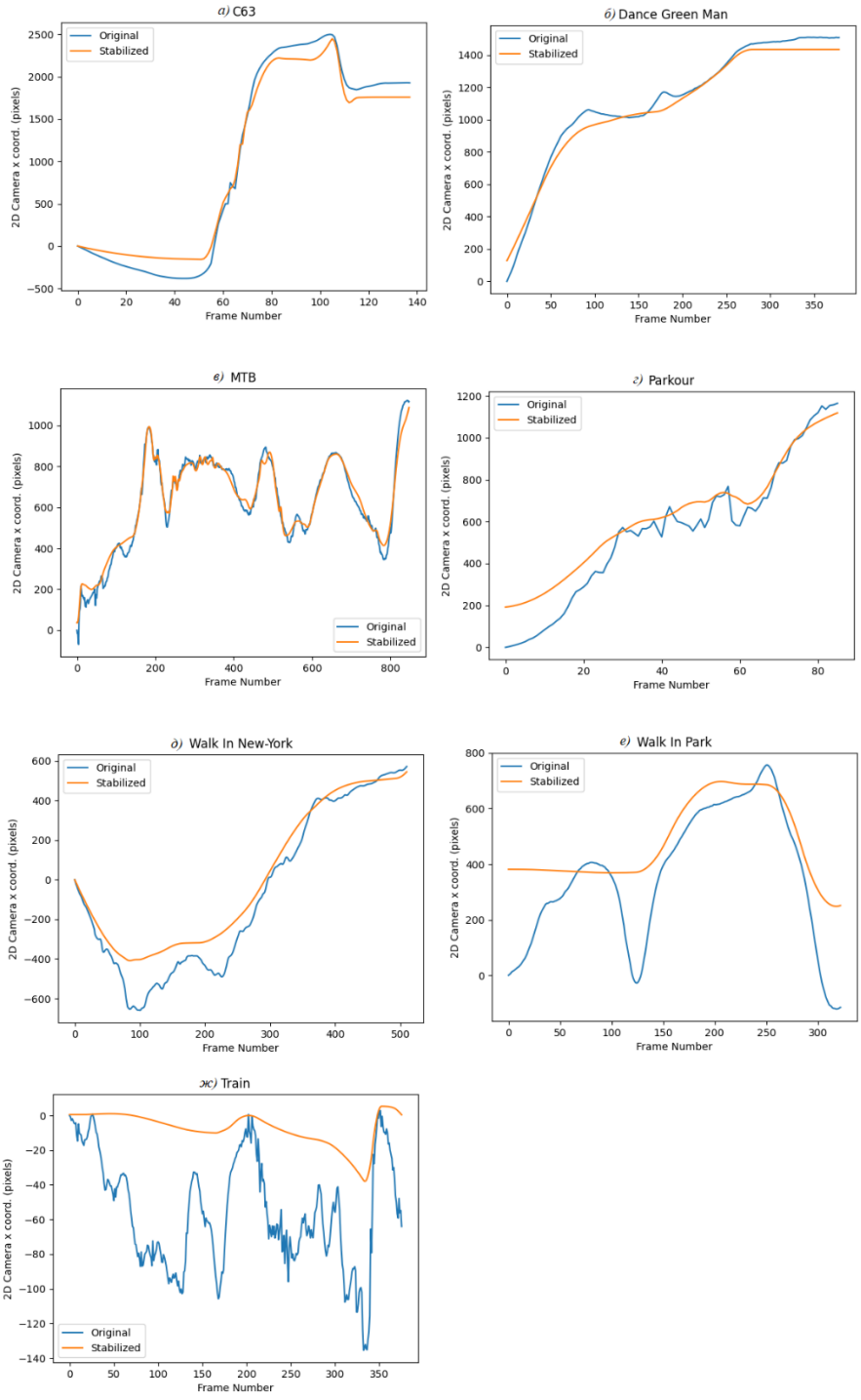


Рис.*3.12 Результати початкового та стабілізованого руху вектору X*

На рисунку 3.13. представлені результати оцінки початкового і компенсованого руху для відеопотоків з динамічними сценами по *Y* вектору.

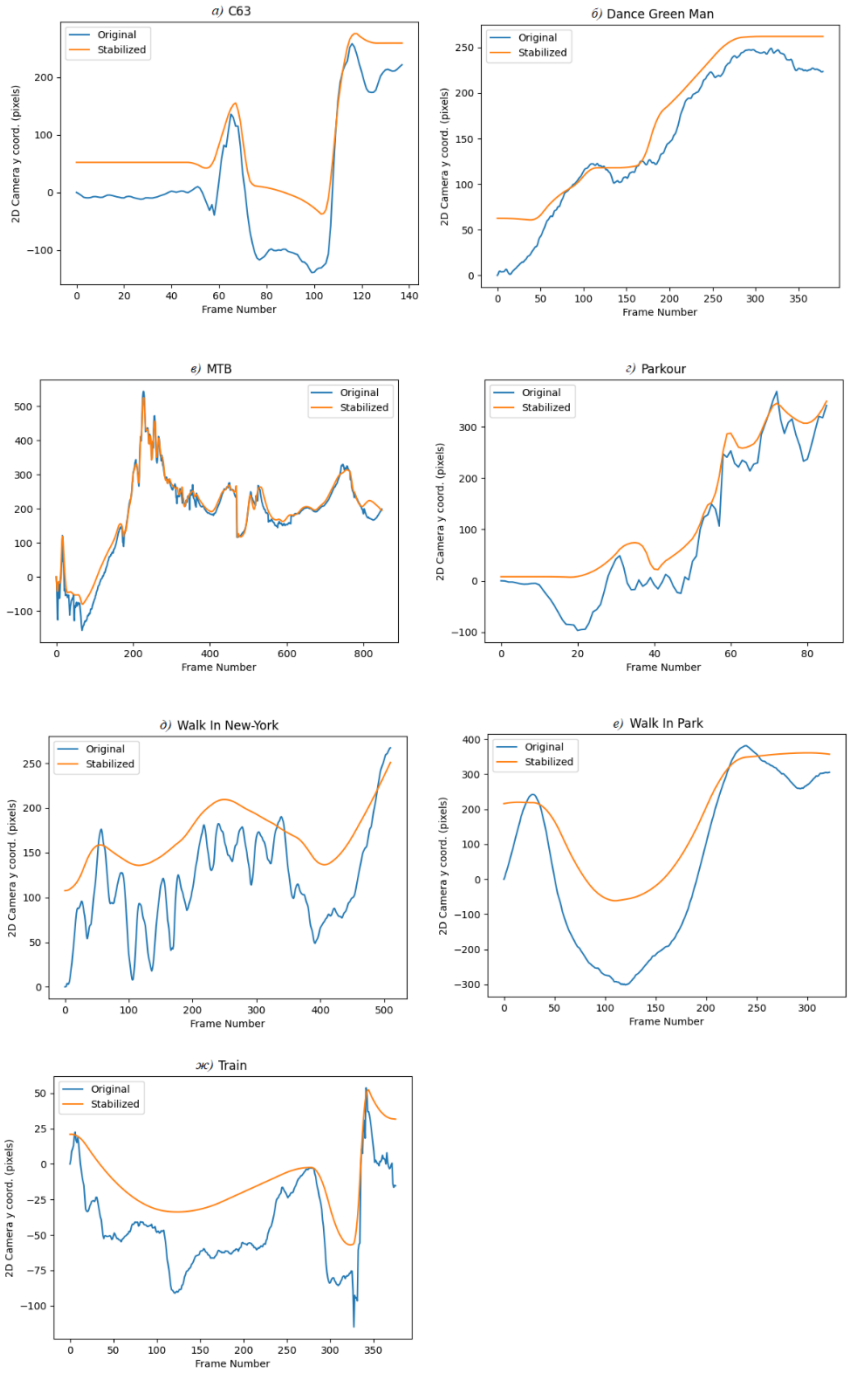


Рис. *3.13 Результати початкового та стабілізованого руху вектору Y*

Було проаналізовано сім відеопотоків, які містять динамічні сцени тривалістю більше ніж 2600 кадрів з різними видами руху камери, такими як: переміщення, поворот, масштабування, та рухомі об’єкти на передньому плані. На рисунку 3.12 та 3.13 представлені результати стабілізації відеопотоку із динамічними сценами з використанням видалення небажаних рухів.

4.3 Алгоритм стабілізації онлайн сервісу «123apps»

На сьогоднішній день онлайн-сервіс «123apps» являється найпопулярнішим онлайн-сервісом який надає послуги для редагування відео, добавляти різні ефекти, змінювати параметри та стабілізувати відео.

В основі алгоритму стабілізації онлайн-сервісу для стабілізації відеопотоку являється автоматичне застосування обмежень для рухів камери. Алгоритм під назвою L1 стабілізує рух камери шляхом видалення небажаних рухів. Основною метою — є обчислення траєкторії камери, яка складається з постійних, лінійних і параболічних сегментів, що імітують рухи камери, які використовуються професійними кінематографістами. Алгоритм базується на структурі лінійного програмування, щоб мінімізувати першу, другу і третю похідні результуючого шляху камери. Такий метод дозволяє стабілізувати відео за межами звичайної фільтрації каналів камери, яка лише погіршує сцени з високочастотним тремтінням. Стабілізування та трансформування відео відбувається за допомогою додавання додаткових обмежень безпосередньо на шлях руху камери. Такий підхід є гарною заміною дорогому обладнанню, оскільки він не вимагає взаємодії з користувачем або використання дорогої 3D – реконструкції сцені. А працює як пост-процес для відео, який можна використовувати як окреме програмне забезпечення, вбудувати в будь-яку камеру, або використовувати як онлайн-сервіс.

4.3.1 Результати експериментальних досліджень роботи алгоритму онлайн-сервісу 123apps для стабілізації статичних сцен

Для оцінки ефективності стабілізації відеопотоку за допомогою алгоритму онлайн-сервісу 123app використовувались відеопотоки які наведені у таблиці 3.

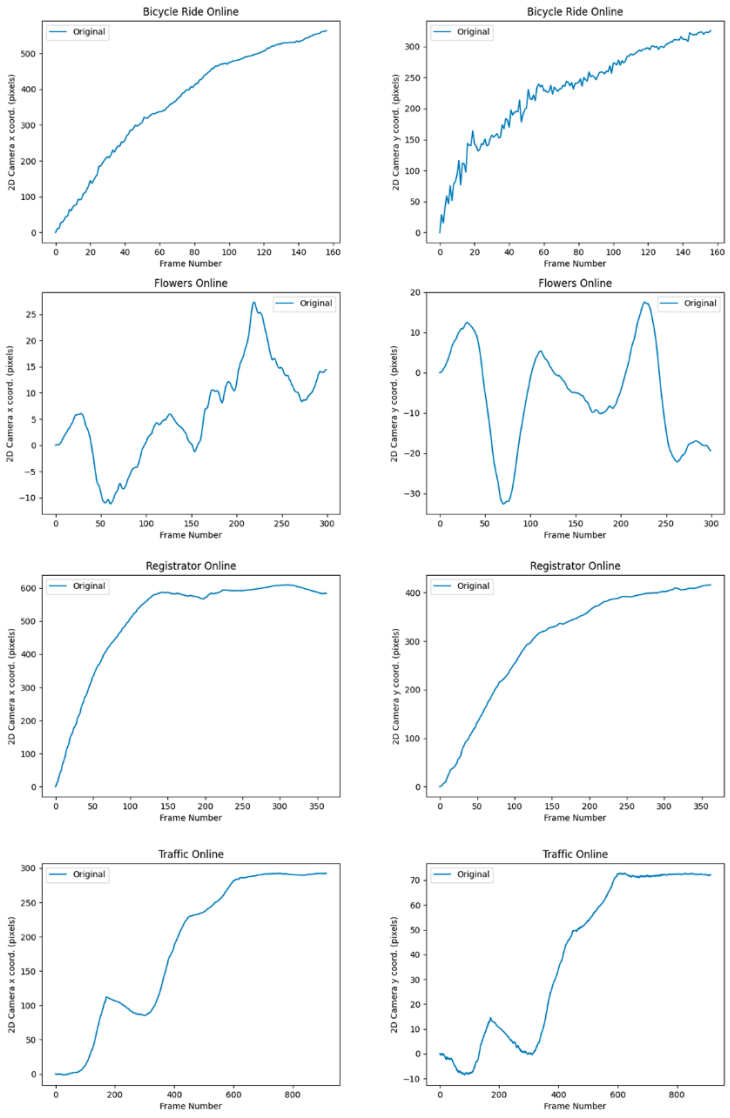


Рис. *4 Стабілізовані вектори руху статичних сцен за допомогою 123apps*

При проведені досліджень на відеопотоках які містять на кадрах об’єкти з повільним та швидким рухом, було з’ясовано, що алгоритм досить погано обробляє блоки з вібраційними рухами. Це можна побачити по гістограмі на рисунку 4 такого відеопотоку «Bicycle Ride». По вектору *Y* можна побачити що існують сильні перепади в вектору руху, які не сильно відрізняються від початкового оригіналу. Показуючи нам що ефективність такого алгоритму в даному випадку успішна в 2-5% і не рекомендується до використання в таких сценах. При цьому швидкість роботи алгоритму досить низька. Дослідження відбувались для різних відеопотоків з розширенням екрану від 640×360 до 1280×720, що містять рухомі об’єкти на передньому плані та задньому, проективні перетворення, та досить значне тремтіння камери.

Найбільшу свою ефективність алгоритм продемонстрував під час роботи стабілізації *X* вектору руху у відеопотоку «Bicycle Ride», який містить велику кількість об’єктів на сцені, при цьому ще й має досить високе розширення екрану.

4.3.2 Результати експериментальних досліджень роботи алгоритму онлайн-сервісу 123apps для стабілізації динамічних сцен

Для оцінки ефективності стабілізації відеопотоку з динамічними сценами за допомогою алгоритму онлайн-сервісу 123apps використовувались відеопотоки наведені у таблиці 4.

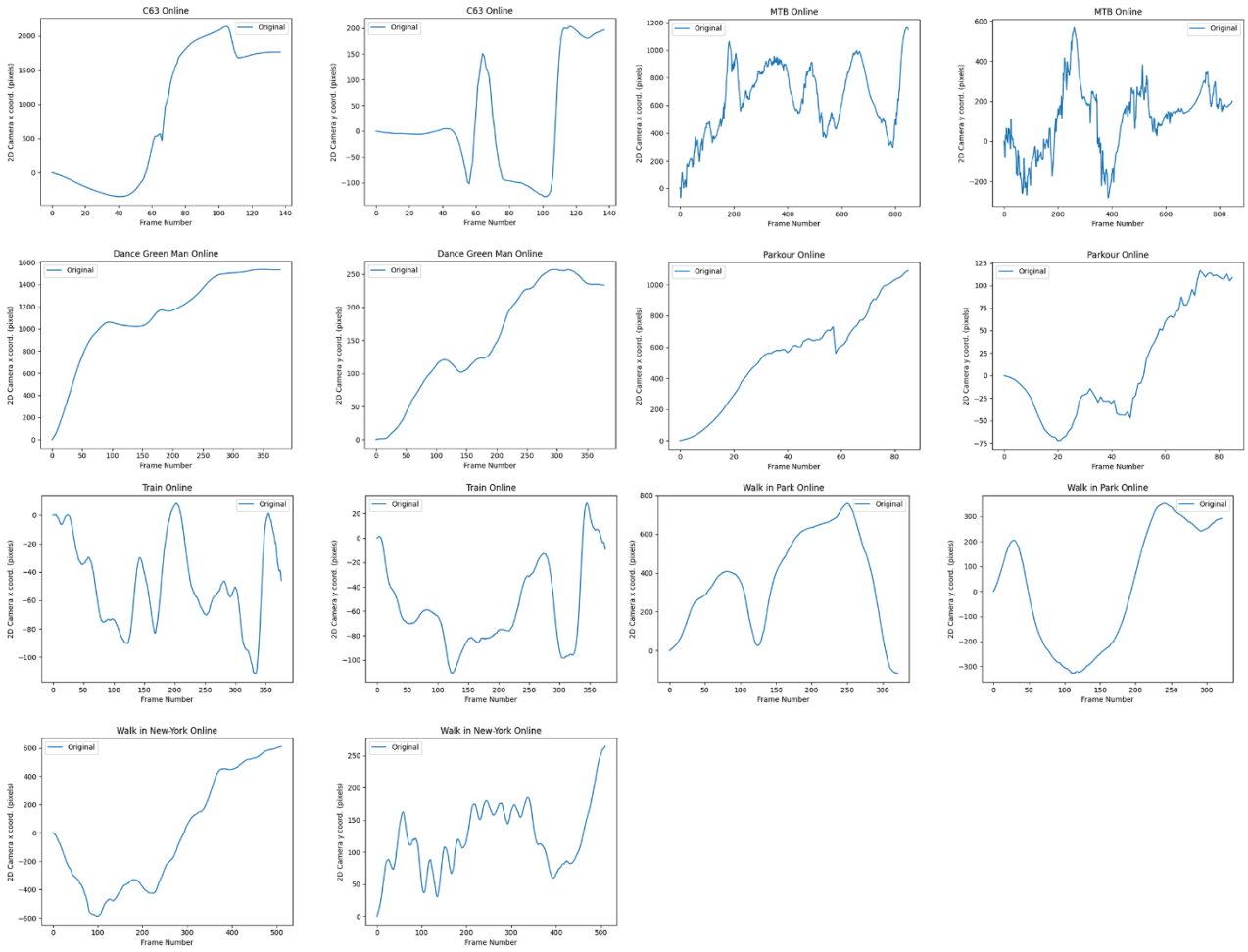


Рис. *5 Стабілізовані вектори рухи динамічних сцен за допомогою 123apps*

При проведені досліджень на відеопотоках що алгоритм досить погано оброблює відео з різкими рухами камери, а в деяких випадках робить відепоток ще гірше ніж оригінал. Це можна побачити у випадку з відеопотоком «MTB», у якому гарно видно, що початковий вектор руху *Y* має мінімальне та максимальне значення руху -150 і 550 відповідно. А новий компенсований рух отриманий алгоритмом має мінімальне та максимальне значення руху -250 та 580 відповідно. Що явно вказує на те, що результат погіршився приблизно 70-90%. І являється неприпустимим для використання в таких вимогах. Та не досить добре працює із поворотами камери, порівнявши кадри 0-80 у відеопотоку «Walk in Park», можна

побачити що результат не змінився, або змінився у дуже маленькому діапазону. Який вказує на те, що гарний результат буде лише на синтетичних тестах, та графіках, і не видно буде для звичайного користувача. Отже його ефективність в такому випадку скасовується.

4.4 Алгоритм стабілізації програмного забезпечення для відео редагування Adobe Premiere Pro

Adobe Premiere Pro — це одне із найвідоміших програмних забезпечень, що було випущено компанією Adobe System, яке дозволяє редагувати відео і динамічні зображення, створювати та розроблювати відео композиції. Нажаль інформація про те як саме працює алгоритм для стабілізації відео засекречена, тому знайти її в відкритому доступі неможливо. Але опираючись на деякі джерела та виходячи з самої назви алгоритму «Warp Stabilizer», основною метою роботи алгоритму є Деформація.

Тобто, виконується деформація сусідніх кадрів $\{I_{n^s}\}_{n\in\Omega_k}$ для вирівнювання цільових кадрів I_{k^t} у простору віртуальної камери. Отримавши деформоване поле від цільового кадру до ключового кадру $F_{k^t\to k^s}$ і за допомогою оцінки оптичного потоку від ключового кадру до сусідніх кадрів $\{F_{k^s\to n^s}\}_{n\in\Omega_k}$, відкривається можливість обчислити поле для деформації від цільового кадру до сусіднього кадру $\{F_{k^t\to n^s}\}_{n\in\Omega_k}$ за допомогою ланцюжка векторів потоку.

Таким чином проводиться деформація сусідніх кадрів I_{n^s} для вирівнювання їх із цільовими кадрами I_{k^t} яке використовують обернене деформування. Деякі пікселі в цільовому кадрі не видно в сусідніх кадрах через оклюзію або виходу кадру за межі екрану. Тому проводяться розрахунки для маски видимості

$\{M_{n^8}\}_{n\in\Omega_k}$ для кожного сусіднього кадру, щоб вказати, чи є піксель дійсним у вихідному кадрі чи ні.

4.4.1 Результати експериментальних досліджень роботи алгоритму програмного забезпечення Adobe Premiere Pro для стабілізації статичних сцен

Для оцінки ефективності стабілізації відеопотоку за допомогою алгоритму програмного забезпечення Adobe Premiere Pro використовувались відеопотоки наведені у таблиці 3.

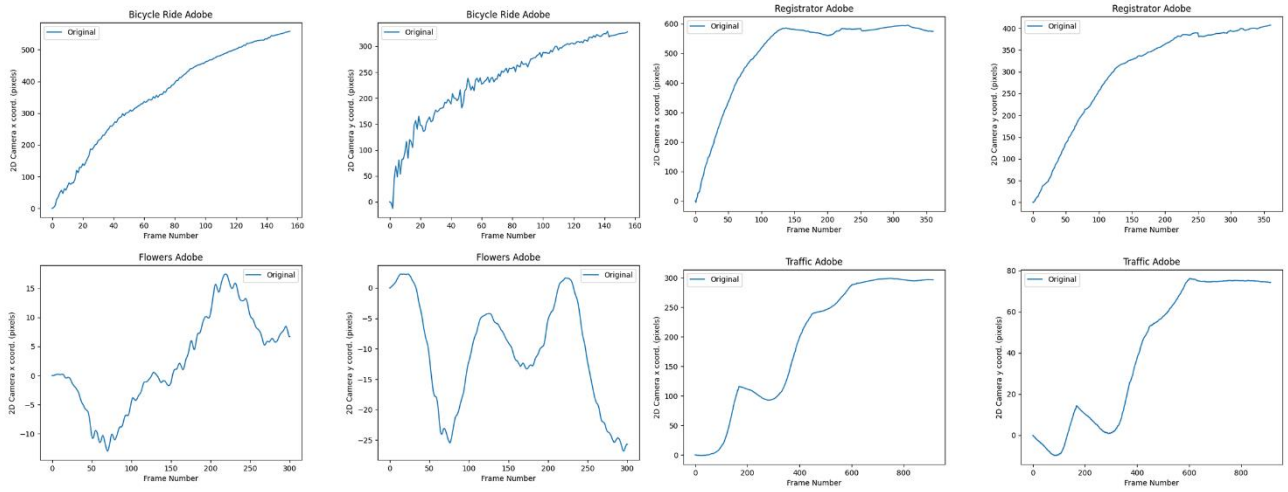


Рис. 6 *Результати стабілізованих векторів руху статичних за допомогою Adobe Premiere Pro*

Після проведень досліджень на відеопотоках, було проаналізовано отримані графіки які можна побачити на рисунку 6. Алгоритм добре впорався зі своїм завданням. Компенсовані гістограми показують, що рух об’єктів став більш плавніше, а граничні результати руху стали меншими. В відеопотоку «Flowers» добре видно на відрізку 200-250 кадрів, що границю руху було зменшено на 30-

50%. Також, треба виділити і роботу алгоритму з коливанням, які дуже гарно видно на графіку відеопотоку «Bicycle Ride», в оригіналі можна побачити різкі коливання по вектору X, та їх відсутність на компенсованому вектору. Такі результати показують нам, що рух в дійсності став більш рівномірним, лише з мікроколиваннями, які не будуть впливати на якість перегляду відеопотоку. Хоча швидкість виконання в такому випадку на відміну від інших була довше в середньому значенні на 65-70%.

4.4.2 Результати експериментальних досліджень роботи алгоритму програмного забезпечення Adobe Premiere Pro для стабілізації динамічних сцен

Для оцінки ефективності стабілізації відеопотоку за допомогою алгоритму програмного забезпечення Adobe Premiere Pro використовувались відеопотоки наведені у таблиці 4.

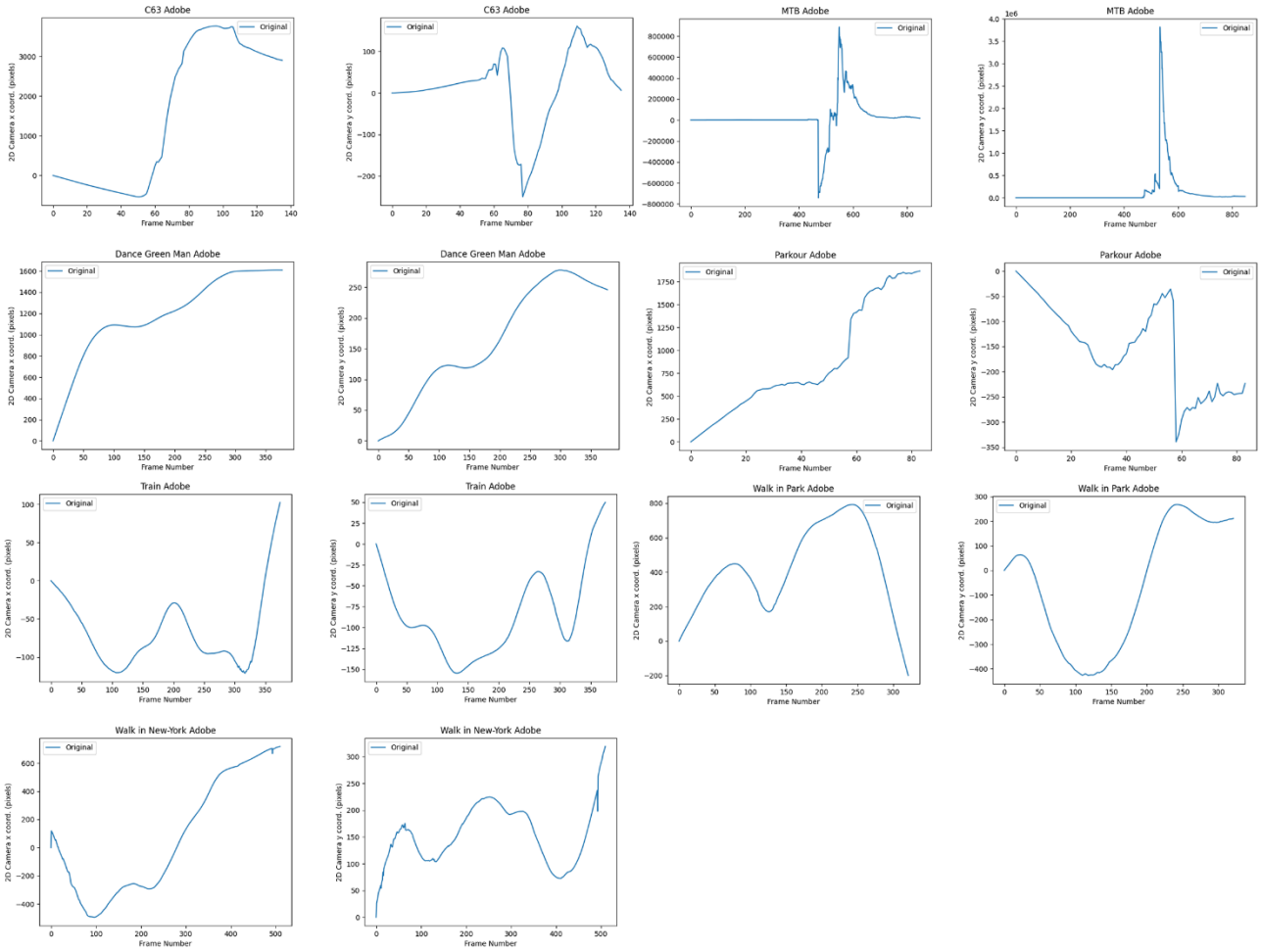


Рис. *7 Результати стабілізованих векторів руху динамічних сцен за допомогою Adobe Premiere Pro*

Під проведення досліджень було складно вирішити кінцевий результат роботи цього алгоритму. Треба пам’ятати ще й те, що цей алгоритм має автоматичне масштабування, обрізку та деформацію кадрів. Дивлячись на гістограми векторів руху зображені на рисунку 7, то можна побачити що в деяких випадках, а саме в відеопотоку «MTB» результат по вектору *X* показав результати руху більше в сімдесят раз більше ніж початковий рух.

Також різка зміна ракурсу камери в відеотопокі «C63» теж дала про себе знати. Але такий момент являється тяжким і різким для усіх алгоритмів, адже

вони працюють з цілим відеопотоком, тому будують маршрут для всього потоку. А цей момент можна вважати провокаційний, який «підкидає» важіль для роботи алгоритму.

Тому, в цілому стабілізація динамічних сцен, а саме компенсація рухів, була виконана на досить високому рівні. При цьому швидкість роботи алгоритму стабілізації відеопотоку була довше в два рази на відміну від інших алгоритмів.

Найбільшу свою ефективність алгоритм продемонстрував під час роботи стабілізації відеопотоку в «Walk in New-York», який містить великі коливання камери, та рухомі об’єкти на передньому плані.

4.5 Алгоритм стабілізації програмного забезпечення для відео редагування Movavi Video Editor

Програмне забезпечення Movavi Video Editor — являється сімейством відеоредакторів для нелінійного редагування відео, та займає середній рівень по технічному потенціалу. Нажаль, інформацію про роботу алгоритму для стабілізації відео зайти в відкритому доступі неможливо. Та під час використання стабілізації користувач може вибрати наступні параметри:

- Точність — впливає на точність аналізу відео. Чим вища точність, тим кращій результат, але це також займає більше часу.
- Тремтіння — користувач може самостійно встановити наскільки сильне тремтіння буде видно на відео.
- Радіус — коли відео стабілізується, кожен об’єкт коригується за допомогою пікселів навколишнього середовища. Радіус впливає на те, наскільки великою буде ця площа. Для відео з динамічними сценами рекомендується використовувати менший радіус щоб уникнути змішування об’єктів і зберегти більше деталей.

- Згладжування — обмежує прискорення камери. Для відео з статичними сценами, або більш-менш статичними, рекомендується ставити вищі значення, ніж для сцен з динамічними сценам. Оскільки надмірне згладжування може обмежити панорамування.
- Обрізка краю — після компенсації руху по краях можуть з’явитися деякі розмиті ділянки, або деформовані об’єкти.

4.5.1 Результати експериментальних досліджень роботи алгоритму програмного забезпечення Movavi Video Editor для стабілізації статичних сцен з рухомими об’єктами

Для оцінки ефективності стабілізації відеопотоку за допомогою алгоритму програмного забезпечення Movavi Video Editor використовувались відеопотоки наведені у таблиці 3.

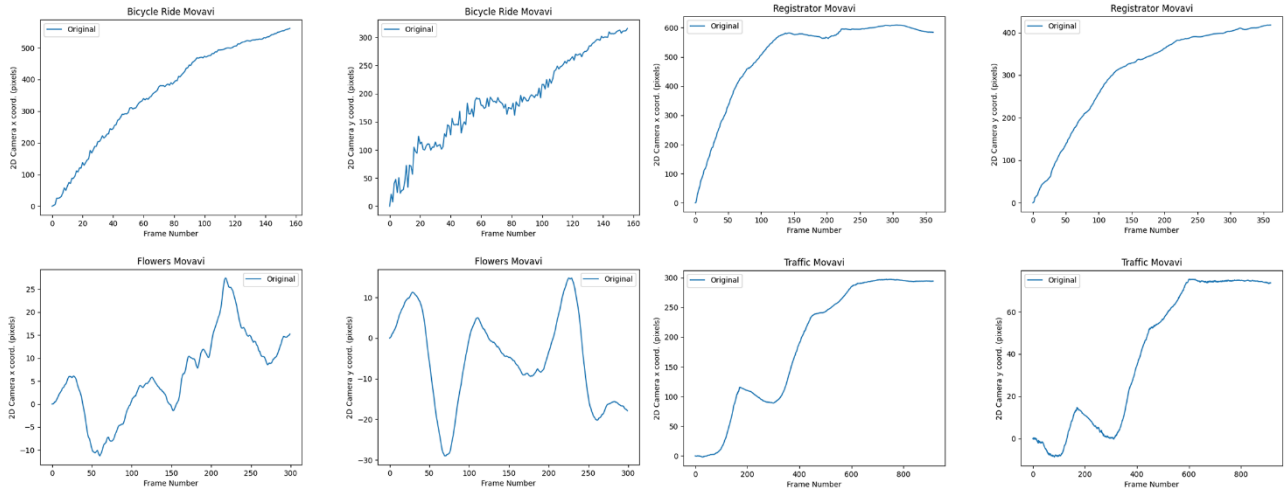


Рис. *8 Результати стабілізованих векторів руху статичних сцен за допомогою Movavi Video Editor*

Після отриманих результатів було що алгоритм не показує якихось колосальних результатів, хоча графіки на рисунку 8 говорять про те, що все таки рух камери дійсно був компенсований. Що зробило відеопоток більш приємним для перегляду. А у випадку із відеопотоком «Traffic» в якому камера статична, та рухомі лише об’єкти, зміни були дуже не значні, які видно лише на графіку, та не видно оком переглядача. Це говорить нам про те, що даний алгоритм може лише трішки «поправити» рух камери, але не повністю його перебудувати, що є не дуже гарним рішенням у певних випадках.

4.5.2 Результати експериментальних досліджень роботи алгоритму програмного забезпечення Movavi Video Editor для стабілізації динамічних сцен

Для оцінки ефективності стабілізації відеопотоку за допомогою алгоритму програмного забезпечення Movavi Video Editor використовувались відеопотоки наведені у таблиці 4.

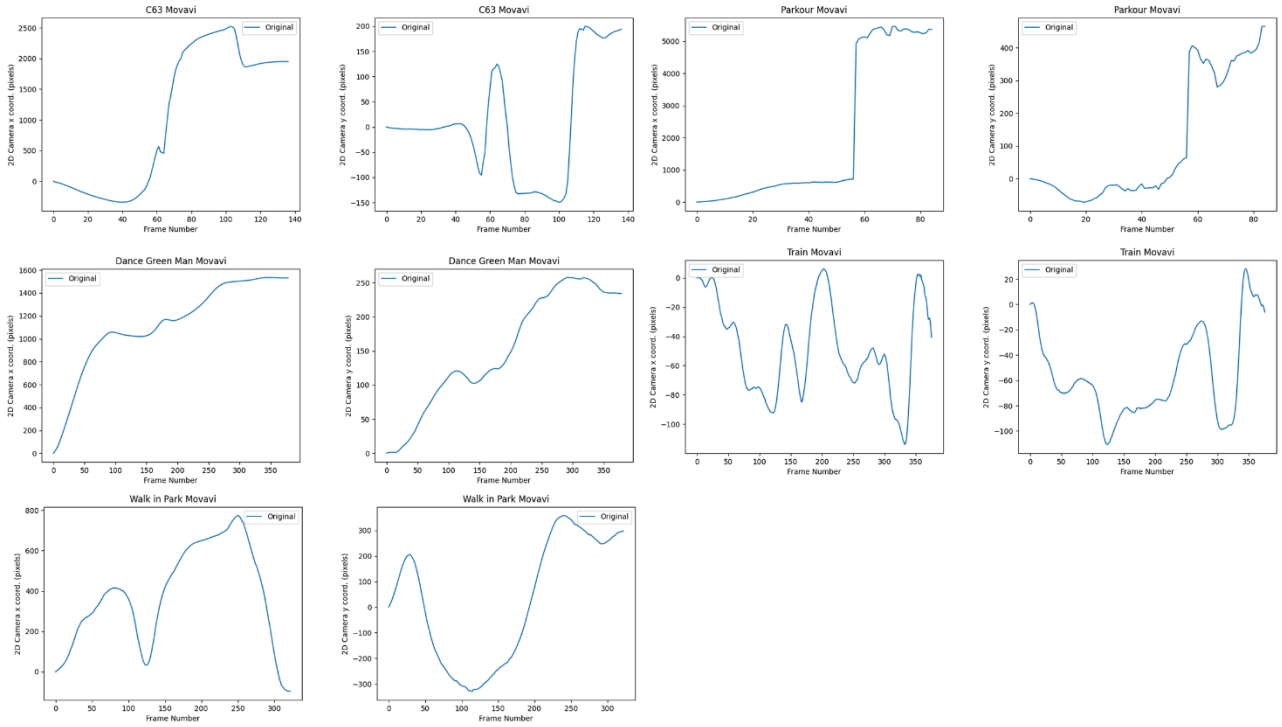


Рис. 9 *Результати стабілізованих векторів руху динамічних сцен за допомогою Movavi Video Editor*

При проведені досліджень на відеопотоках які містять на кадрах об’єкти з швидким рухом, різкими рухами камери та зміно камери, було з’ясовано, що алгоритм, непогано обробляє такі відеопотоки. Хоча в деяких випадках, в такому як у відеоролику «Parkour» на рисунку 9 можна побачити різку зміну положення камери на двох векторах, не відомо що призвело алгоритм до таких кроків. Адже пригадаємо те, що інформація про роботу даного алгоритму відсутня.

Найбільш ефективний успіх цей алгоритм продемонстрував під час роботи стабілізації відеопотоку «Train», який містить рухомий об’єкт на передньому плані, та різкі коливання камери. А компенсований рух досить плавний, та майже не має різких змін, таких як мікро-вібрацій, та мікро-коливань.

4.6 Критерії до порівнянь роботи всіх алгоритмів стабілізації відеопотоку

Для того щоб правильно порівняти роботу усіх алгоритмів, треба чітко визначити рамки з якими ми будемо працювати, та критерії до оцінювання.

Критерії для оцінювання були наступними:

- Порівняння розмірів відеофайлів початкового та кінцевого результату
- Час роботи алгортиму
- Вимоги до апаратної частини

Для використання алгоритмів застосовувався один набір відеофайлів з відеопотоками. Всі тестування проводились в однакових умовах, та на одному персональному комп’ютері.

Цей персональний комп’ютер був наділений наступними технічними характеристиками:

1. Чотирьох ядерний процесор Intel Xeon E3-1280 v2 з тактовою частотою 3.6 ГГц.
2. Графічний процесор NVIDIA GeForce GTX 1070 8Gb.
3. 16 ГБ оперативної пам’яті.
4. Операційна система Windows 10 версія 20H2.

В програмних забезпеченнях використовувались автоматично обрані параметри для роботи алгоритму. Гістограми векторів руху камери кожного відепотоку будувались за допомогою одного алгоритму. Нажаль, алгоритм стабілізації який використовувався в програмному забезпеченню Movavi Video Editor давав збій і програму автоматично закінчувала свою роботу. З’ясувати що на це впливало не вдалося, тому в результатах кінцевих оцінок, відеофайли які не вдалося стабілізувати за допомогою цього програмного забезпечення, автоматично були і видалені при підсумках у других алгоритмах програмного забезпечення.

4.6.1 Результати порівнянь розмірів файлів відеопотоку до та після використання алгоритму стабілізації

Важливою частиною використання такого алгоритму є кінцевий розмір отриманого відеофайлу. Адже не завжди у звичайного користувача є вільне місце для збереження певних відеофайлів, на відміну від професійних компаній відеозйомок. Такі компанії можуть мати приватні сервери та зберігати там велику кількість терабайтів відеопотоку.

А після використання таких алгоритмів, розміри деяких відеофайлів можуть збільшитися в десятки, а то і сотню разів від початкового розміру. Отже, під час проведення дослідження роботи алгоритмів стабілізації відеопотоку було отримано наступні результати яких наведені у таблиці 5.

Таблиця *5*

Порівняння розмірів відеофайлів до та після використання алгоритму стабілізації відеопотоку

Назва відеофайлу	**Початковий розмір у мегабайтах**	**Розмір після використання алгоритму у мегабайтах**			
		Розроблений застосунок	**123apps**	**Adobe Premiere Pro**	**Movavi Video Editor**
Bicycle Ride	3,2	7,6	5,4	6,5	3,0
C63	2,4	2,4	5,8	6,6	5,9
Dance Green Man	4,4	14,1	9,6	15,6	6,8
Flowers	2,2	3,6	2,3	12,2	2,2
MTB	8,3	33,8	23,8	33,8	-
Parkour	2,5	6,1	5,7	4,4	3,9
Registrator	5,4	20,4	15,6	17,5	7,9
Traffic	1,9	4,1	4,1	29,6	4,7
Train	4,2	6,6	5,9	18,0	4,0
Walk in NY	20,0	38,2	21,1	10,2	-
Walk in Park	16,5	161,5	116,4	15,6	41,4

Отримавши результати можна побачити що за цим критерієм найкраще справився саме алгоритм який використовується в програмному забезпечені Movavi Video Editor, а найгіршим результатом буде використання L1 алгоритму.

4.6.2 Порівняння часу роботи алгоритмів стабілізації відеопотоку

На час роботи алгоритму впливає достатньо велика кількість різних факторів, це і освітленість кожного кадру, кількість об’єктів у кадрі, розміри кадру, процент масштабування, рівень компенсації руху та багато чого іншого. Довільним результатом вважається такий алгоритм, який виконує свою роботу за час тривалості всього відеопотоку. Чим менше, тим набагато краще, а якщо значення вже більше від оригіналу, то це означає що або алгоритму досить тяжко працювати в таких умовах. Тому їх прийнято відкидати від використання.

У таблиці 6 наведені результати часу виконання алгоритмів.

Таблиця *6*

Порівняння часу виконання алгоритму стабілізації відеопотоку

Назва	**Час виконання алгоритму у секундах**			
	Розроблений застосунок	**123apps**	**Adobe Premiere Pro**	**Movavi Video Editor**
Bicycle Ride	8,3	5,7	10,2	5,0
C63	12,0	9,2	35,6	11,0
Dance Green Man	20,0	7,2	31,0	11,3
Flowers	62,8	15,9	103,9	5,8
MTB	7,4	9,8	24,3	-
Parkour	13,0	4,1	19,0	12,0
Registrator	52,5	39,0	130,0	6,4
Traffic	106,4	76,7	313,5	-
Train	22,9	17,0	33,0	394,0
Walk in NY	61,2	12,8	26,5	11,8
Walk in Park	7,5	5,4	14,0	8,3

Отримавши результати вирисовується така картинка, що найшвидшим є використання алгоритму програмного забезпечення Movavi Video Editor, найбільше часу для виконання алгоритму знадобилось для програмного забезпечення Adobe Premiere Pro, його час більше майже в сорок раз.

4.6.3 Порівняння вимог для використання алгоритмів стабілізації відеопотоку

Алгоритми стабілізації відеопотоку використовуються в різних випадках. Як же говорилося на початку, це можуть бути недорогі прилади які мають змогу знімати відео, або професійні відеокамери, ціна яких перевищує за десятки тисяч доларів. А ще потрібно купувати спеціальне обладнання для стабілізування самої камери, яке також коштує немаленьку суму. Тож саме в таких випадках нам і допомагають такі алгоритми. Але які ж вимоги до апаратної системи мають такі алгоритми? Розглянемо таблицю 7.

Таблиця *7*

Вимоги апаратної частини для використання алгоритму стабілізації відеопотоку

Назва системи	**Вимоги**
Розроблений застосунок	Основною вимогою для використання такої системи, є встановлений компілятор Python та додаткові бібліотеки. Встановлений двох ядерний процесор з потужністю 2.0 ГГц, та 16 ГБ пам’яті на жорсткому диску для збереження кешу.
123apps	Пристрій з доступом до інтернет-мережі. Підтримка Web 2.0
Adobe Premiere Pro	Ліцензійна версія програми. Двох ядерний процесор з потужністю 2.0 ГГЦ. Оперативна пам’ять від 4 ГБ.
Movavi Video Editor	Ліцензійна версія програми. Двох ядерний процесор з потужністю 2.0 ГГЦ. Оперативна пам’ять від 6 ГБ.

Тож як можна побачити, найдоступнішим варіантом являється використання онлайн сервісу 123apps, за допомогою якого можна стабілізувати відео навіть з мобільного телефону, чи любого другого пристрою. А самим тяжким для звичайного користувача буде використання методу L1, адже для його роботи треба попередньо значну кількість кроків. А відсутність інтерфейсу взагалі може налякати любого користувача.

4.7 Висновки з розділу 4

В четвертій главі проводяться експериментальні дослідження розробленого програмного забезпечення, застосування онлайн сервісу, та інших програмних забезпечень для стабілізації відеопотоку.

Було проаналізовані відеопотоки з статичними та динамічними сценами, які відрізняються різними видами руху камери: це переміщення камери, нахил та її поворот. Оцінка алгоритму проводилась на відеопотоках сприятливих для тестування алгоритмів стабілізації.

Результати показують що розроблене програмне забезпечення добре справляється з поставленим завданням, та має один з найкращих результатів. Основними перевагами можна виділити наступні моменті: швидкий час виконання роботи алгоритму, висока якість компенсованого шляху руху камери. Але це програмне забезпечення має й вагові недоліки, основними з яких можна виділити: досить великий розмір вихідного файлу на відміну початкового.

Також непогані результати показали програмне забезпечення Adobe Premiere Pro та онлайн сервіс 123apps. Які зі своїм завданням справляються на високому рівні, а вихідні результати мають високий рівень компенсованого руху. А час роботи онлайн сервісу взагалі має найкращі результати, в цьому випадку він

займає кріпке перше місце, й зносить усіх конкурентів на своєму місці. А результати вектору компенсованого руху в більшості випадків мають близькі значення які можна отримати за допомогою програмного забезпечення Adobe Premiere Pro. Також цей сервіс має найменші вимоги, а значить для звичайного користувача він підійде найкраще.

Найгірші результати нажаль показало програмне забезпечення Movavi Video Editor. Вихідний вектор компенсованого руху були на непоганому рівні, а в деяких випадках показав кращі результати ніж попередні два забезпечення. Час виконання також не уступає всім. А от найгіршим нажаль стало те, що він повністю відмовляється працювати з відеопотоками які мають високочастотні коливання камери. А це є неприпустимим для такого рівня задачі.

Отже, для отримання кінцевого результату знадобилося багато часу. Після отримання всіх метрик, було чітко зрозуміло, що найкращий результат показує алгоритм стабілізації відеопотоку в онлайн сервісі 123apps. Швидкість роботи і отримані результати займають високий рівень, а низькі вимоги до апарату виводять його на перше місце. Для використання опитним користувачам найкращим результатом буде використання власноруч розробленого програмного забезпечення та Adobe Premiere Pro. Адже вони мають велику кількість опцій, тому для кожної сцени їх можна налаштовувати власноруч.

На останньому місці залишається алгоритм який використовується в програмному забезпечені Movavi Video Editor, хоча його швидкодія і результати являються задовільними. Але неможливість обробити деякі відео з високочастотними коливаннями являється неприпустимим для такого типу завдання. Отже, перед тим як його використовувати потрібно впевнитися що він зможе з ним працювати.

ВИСНОВКИ

В роботі досліджувались алгоритми стабілізації відеопотоків важких статичних та динамічних сцен. Було виконано наступні етапи:

- Досліджено існуючі підходи та методи стабілізації відеопотоку, розглянуто методи знаходження і компенсації небажаного руху камери, їх складність, обмеження та умови в яких вони працюють. Розібрано їх етапи роботи: ознайомлено з попередньою обробкою сцені відеопотоку, оцінка траєкторії руху, засоби які використовуються для компенсації руху, та як відновлюється зображення у кінцевому етапі.
- Розроблено готову систему для стабілізації відеопотоку використовуючи метод на основі створення оптимальних шляхів камери для створення компенсованого руху вектору камери який базується на структурі лінійного програмування шляхом видалення небажаних рухів які були отримані в результаті побудованої двох мірної сцени руху камери. Проведено тестові для знаходження оптимальних параметрів розробленого програмного забезпечення.
- Досліджено результати роботи розробленого програмного забезпечення з іншими готовими продуктами: використано онлайн сервіс 123apps, програмні забезпечення Adobe Premiere Pro та Movavi Video Editor. Знайдено їх переваги і недоліки в роботі алгоритму стабілізації відеопотоку, перевірено стабільність роботи в різних умовах, знайдено пошук оптимального рішення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Almeida J., Minetto R., Almeida T.A., Da S. Torres R., Leite N.J. Robust estimation of camera motion using optical flow models // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 5875 LNCS (PART 1), 2009, P. 435–446.

2. Bao, X., Zhou, D., Peilin, L., Goto, S. An advanced hierarchical motion estimation scheme with lossless frame recompression for ultra high definition video coding // IEEE International Conference on Multimedia and Expo, P. 820-825, 2010.

3. Battiato S., Puglisi G., A robust video stabilization system by adaptive motion vectors filtering, in Proceeding of ICIAP. Modena (Italy), 2008, P. 435-440.

4. Battiato S., Gallo G., Puglisi G., Scellato S. SIFT Features Tracking for Video Stabilization, Proceedings of the 14th International Conference on Image Analysis and Processing, 2007. P. 825-830.

5. Bay H., Ess A., Tuytelaars T., Van Gool L. Speeded-Up Robust Features (SURF) // Computer Vision and Image Understanding. 2008. Vol. 110. P. 346 – 359.

6. Boudlal A., Nsiri B. Aboutajdine D. Modeling of Video Sequences by Gaussian Mixture: Application in Motion Estimation by Block Matching Method // EURASIP Journal on Advances in Signal Processing, 2010.

7. Cai J., Pan W. On fast and accurate block-based motion estimation algorithms using particle swarm optimization // Information Sciences, 197, 2012, P. 53–64

8. Cai J., Walker R. Robust video stabilisation algorithm using feature point selection and delta optical flow. IET Computer Vision, Vol. 3, No.4, 2009. P. 176–188.

9. Choi, K.-S., Ko, S.-J. Hierarchical motion estimation algorithm using reliable motion adoption // Electronics Letters, vol. 46, no. 12, P. 835 - 837, 2010.

10. Chun J. B., Jung H., Kyung C.M. Suppressing rolling-shutter distortion of cmos image sensors by motion vector detection. Consumer Electronics, IEEE Transactions on 54, (4) 2008, P. 1479 –1487.

11. Dani A.P., Dixon W.E. Single camera structure and motion estimation // Lecture Notes in Control and Information Sciences, 401, 2010. P. 209–229.

12. Donovan P. O., Using Optical flow for Stabilizing Image Sequence, CMPT400 (Honours Thesis Course), 2005, P. 1-10.

13. Erturk S. Real-Time Digital Image Stabilization Using Kalman Filters. Real-Time Imaging, Vol. 8, 2002. P. 317–328.

14. Favorskaya M. Motion Estimation for Object Analysis and Detection in Videos. In Handbook «Advances in reasoning-based image processing, analysis and intelligent systems: Conventional and intelligent paradigms», Springer-Verlag, Berlin Heidelberg, 2012, P. 211–253.

15. Favorskaya, M., Pyankov D., Popov A. Motion estimations based on invariant moments for frames interpolation in stereovision. Procedia Computer Science 22, 2013, P 1102 – 1111.

16. Flusser J., Suk T. Degraded image analysis: an invariant approach// IEEE Trans. Pattern Anal. Mach. Intell.,1998. Vol. 20. P. 590-603.

17. Forssen P.E., Ringaby E. Rectifying rolling shutter video from hand-held devices. In CVPR, 2010, P. 507–514.

18. Gleicher M., Liu, F. Re-cinematography: Improving the camerawork of casual video. ACM Transactions on Multimed. 5(1), 2008, P. 1–28.

19. Grundmann M. Auto-directed video stabilization with robust L1 optimal camera paths / M. Grundmann, V. Kwatra, I. Essa // In Proc. CVPR 2011, P 225-232.

20. Hu R., Shi R., Shen I., Chen W. Video Stabilization using Scale invariant Features, Proceedings of the 11th International Conference Information Visualization, 2007. P. 871–877.

21. Huang K.-Y., Tsai Y.-M., Tsai C.-C., Chen L.-G. Video Stabilization for Vehicular Applications using Surf-like Descriptor and KD-tree. 17th IEEE Int. Conf. on Image Processing, 2010. P. 3517–3520.

22. Hui, L., Peijun, D., Weichang, Z., Lianpeng, Z., Huasheng, S. Image registration based on corner detection and affine transformation //Image and Signal Processing (CISP), P. 2184-2188, 2010.

23. Ismail, Y., McNeelly, J., Shaaban, M., Bayoumi, M.A. Enhanced efficient Diamond Search algorithm for fast block motion estimation // IEEE International Symposium on Circuits and Systems, P. 3198-3201, 2009.

24. Karpenko A., Jacobs D., Baek J., Levoy M. Digital Video Stabilization and Rolling Shutter Correction using Gyroscopes // Stanford University Computer Science Tech Report CSTR 2011, P. 1-7.

25. Ke Y., Sukthankar R. PCA-SIFT: A More Distinctive Representation for Local Image Descriptors. In Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Washington, 2004, P. 506-513.

26. Kumar, S., Azartash, H., Biswas, M. Truong Nguyen Real-Time Affine Global Motion Estimation Using Phase Correlation and its Application for Digital Image Stabilization // IEEE Transactions on Image Processing, 20(12), P. 3406 - 3418.

27. Lakshman H., Schwarz H., Wiegand T. Adaptive Motion model selection using a cubic spline based estimation framework IEEE International Conference on Image Processing, Hong Kong, 2010, P. 805 - 808.

28. Liu C., Freeman W. T.A high-quality video denoising algorithm based on reliable motion estimation. In Proc. ECCV 2010, P 706–719.

29. Liu F., Gleicher M. Video Retargeting: Automating Pan and Scan. ACM Multimedia 2006, 2006, P. 241–250.

30. Liu F., Gleicher M., Jin H., Agarwala A. Content-preserving warps for 3D video stabilization. In ACM Transactions on Graphics (Proceedings of SIGGRAPH), 28(3), 2009, P. 225–231.

31. Lowe, D.G. Distinctive Image Features from Scale-Invariant Keypoints // International Journal of Computer Vision, 2004. Vol. 60, No. 2, P. 91–110.

32. Lucas B.D., Kanade T. An Iterative Image Registration Technique with an Application to Stereo Vision // International Joint Conference on Artificial Intelligence, 1981. P. 674–679.

33. Matsushita Y., Ofek E., Tang X., Shum H. Full-frame video stabilization in Proc. IEEE Int. Conf. Comput. Vision Pattern Recognition, San Diego, CA, 2005, P. 50–57.

34. Matsushita Y., Ofek E., Ge W., Tang X., Shum H.-Y. Full-Frame Video Stabilization with Motion Inpainting. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 28, no. 7, 2006. P. 1150–1163.

35. Mihaylova L., Brasnett P., Canagarajan N. Bull D. Object Tracking by Particle Filtering Techniques in Video Sequences; In: Advances and Challenges in Multisensor Data and Information. NATO Security Through Science Series, 8. Netherlands: IOS Press, 2007. P. 260–268.

36. Puglisi G., Battiato S. Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on Date of Conference: 20-25 June 2011. P. 50 – 57.

37. Han Z.J., Ye Q.X., Jiao J.B. Online feature evaluation for object tracking using Kalman Filter. 19th Int. Conf. on Pattern Recognition (ICPR 2008), 2008. P. 1–4.

38. Litvin A., Konrad J., Karl W.C. Probabilistic video stabilization using Kalman filtering and mosaicking. Image and Video Communication and Processing 2003, SPIE-IS&T Electronic Imaging, SPIE, Vol. 5022, 2003. P. 663-674.

Декларація

академічної доброчесності

здобувача ступеня вищої освіти ЗНУ

Я, Кондратьєв Давід Віталійович, студент__2__курсу, форми навчання денної, Інженерного навчально-наукового інституту, спеціальність 121 Інженерія програмного забезпечення, адреса електронної пошти se216-09@stu.zsea.edu.ua, — підтверджую, що написана мною кваліфікаційна робота на тему «**Дослідження алгоритмів стабілізації відеопотоків**» відповідає вимогам академічної доброчесності та не містить порушень, що визначені у ст.42 Закону України «Про освіту», зі змістом яких ознайомлений.

- заявляю, що надана мною для перевірки електронна версія роботи є ідентичної її друкованій версії;

згоден/згодна на перевірку моєї роботи на відповідність критеріям академічної доброчесності у будь-який спосіб, у тому числі а допомогою інтернет-систем, а також на архівування моєї роботи в базі даних цієї системи.

Дата 30.11.2021 Підпис_____Кондратьєв Давід Віталійович

(студент)

Дата 30.11.2021 Підпис_____Попівщий Василь Іванович

(науковий керівник)