МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА ЗАСТОСУНКУ «ВІРТУАЛЬНА ПРОГУЛЯНКА»»**

Виконав: студент 5 курсу, групи 6.1229-з
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)
освітньої програми Комп'ютерні науки
(назва освітньої програми)
Федченко К.В.
(ініціали та прізвище)
Керівник доцент кафедри комп'ютерних наук, к.т.н., Добровольський Г.А.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
Рецензент професор кафедри програмної інженерії, доцент, к.ф.-м.н. Кудін О.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра комп'ютерних наук

Рівень вищої освіти бакалавр

Спеціальність 122 Комп'ютерні науки
(шифр і назва)

Освітня програма Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук, д.т.н., доцент

_____ Шило Г.М.
(підпис)

" ____ " _____ 2024 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Федченку Костянтину Вікторовичу
(прізвище, ім'я та по-батькові)

1. Тема роботи Розробка застосунку «Віртуальна прогулянка»

керівник роботи Добровольський Геннадій Анатолійович, к.т.н.
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 21 » грудня 2023 року № 2180-с
та додатком до наказу ЗНУ №376-с від « 29 » лютого 2024 року

2. Строк подання студентом роботи 15.05.2024

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Розробка сайту для ліцею с технологією REACT.

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

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 04.03.2023

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	04.03.2024	
2.	Збір вихідних даних.	09.03.2024	
3.	Обробка методичних та теоретичних	20.03.2024	
	джерел.		
4.	Розробка першого та другого розділу.	26.03.2024	
5.	Розробка третього розділу.	04.04.2024	
6.	Оформлення та нормоконтроль	10.05.2024	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	30.05.2024	

Студент _____ К. В. Федченко
(підпис) (ініціали та прізвище)

Керівник роботи _____ Г. А. Добровольський
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.Г.Спиця
(підпис) (ініціали та прізвище)

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧКИ

CV	(Computer Vision) Комп’ютерний зір
DNN	(DotNetNuke) це система управління контентом сайту, або глибокі нейронні мережі
OCR	Оптичне розпізнавання символів
AR	(Augmented reality) систем та доповненої реальності
VR	(Virtual reality) систем і віртуальної реальності
ШІ	(Artificial intelligence, AI) штучний інтелект
ПЗ	(Software) програмне забезпечення
ШПФ	Швидке перетворення Фур’є

ЗМІСТ

РЕФЕРАТ

Кваліфікаційна робота бакалавра “Розробка застосунку “Віртуальна прогулянка””: 50 с., 5 табл., 14 рис.,17 джерел, 2 додатки.

ВІРТУАЛЬНА РЕАЛЬНІСТЬ (ВР), ГЛИБОКЕ НАВЧАННЯ, КОМП’ЮТЕРНИЙ ЗІР, OPEN CV, ШТУЧНИЙ ІНТЕЛЕКТ (ШІ),

Об’єкт дослідження – застосунок “Віртуальна прогулянка”.

Мета роботи − реалізувати простий та швидкий вимірювач швидкості зміни зображення, який може автономно працювати на мобільній платформі, та поєднати його із виконавчим механізмом.

Методи дослідження – моделювання, проектування, програмний, аналітичний.

Реалізація вимірювання швидкості зміни зображення передбачає використання відеокамери для вимірювання частоти рухів людини. Швидкість віртуальної ходьби вважається пропорційною кількості рухів за одиницю часу.

Це робиться шляхом зміни швидкості відтворення відео, яке показує рух навколо бігової доріжки. Швидкість відтворення змінюється кожні 0,5 секунди і розраховується за кількістю кадрів. Чим швидше рухається людина, тим більше кадрів відтворюється. Обробка зображення використовується для порівняння поточного зображення з попереднім, а для прискорення часу обробки застосовується швидке перетворення Фур’є з децимацією. Структура програмного забезпечення передбачає згладжування кадрів, знаходження зображення Фур’є згладженого кадру, обчислення відносної відстані та підрахунок рухів, якщо відстань перевищує певний поріг. Про швидкість об’єкта спостереження можна судити, підрахувавши ці події в одиницю часу. Алгоритми включають захоплення кадру, його згладжування, отримання зображення Фур’є, обчислення різниці між поточним і попереднім зображеннями Фур’є та збільшення лічильника moveCounter, якщо різниця між лічильниками перевищує порогове значення. Алгоритм відтворення швидкості перевіряє відсутність руху протягом однієї секунди та зупиняє

таймер відтворення або пропорційно встановлює частоту кадрів. Третій алгоритм просто показує кадри на весь екран, а швидкість визначається частотою спрацьовування події таймера.

Реалізація програмного коду передбачає контрольну форму, де можна вибрати трек і розпочати віртуальну прогулянку. Застосунок працює асинхронно з трьома алгоритмами, що працюють одночасно.

SUMMARY

The bachelor's qualification work "Development of the application "Virtual walk"": 50 pages, 14 figures, 5 tables, 17 references, 2 supplements

ARTIFICIAL INTELLIGENCE (AI), DEEP LEARNING, COMPUTER VISION, OPEN CV, VIRTUAL REALITY (VR)

The object of the study is the "Virtual Walk" application.

The goal of the work is to implement a simple and fast image change rate meter that can work autonomously on a mobile platform and connect it to an executive mechanism.

Research methods − modeling, design, software, analytical.

The implementation of the measurement of the rate of change of the image involves the use of a video camera to measure the frequency of human movements. The speed of virtual walking is considered proportional to the number of movements per unit of time.

This is done by changing the playback speed of a video that shows movement around the treadmill. The playback speed changes every 0.5 seconds and is calculated by the number of frames. The faster a person moves, the more frames are played. Image processing is used to compare the current image with the previous one, and Fourier transform with decimation is applied to speed up processing time. The framework of the software involves smoothing the frames, finding the Fourier image of the smoothed frame, calculating the relative distance, and counting movements if the distance exceeds a certain threshold. The speed of the object of observation can be judged by counting these events per unit of time. The algorithms include capturing a frame, smoothing it, obtaining a Fourier image, calculating the difference between the current and previous Fourier images, and incrementing the moveCounter if the difference between the counters exceeds a threshold value. The speed playback algorithm checks for no motion for one second and stops the playback timer or sets the frame rate proportionally. The third algorithm simply

shows the frames to the entire screen, and the speed is determined by the frequency of the timer event.

The implementation of the software code involves a control form where you can select a track and start a virtual walk. The application works asynchronously with three algorithms running simultaneously.

ВСТУП

Розвиток інформаційних технологій в даний час зосереджений на відеоінформаційних технологіях, які обробляють і використовують зображення. Актуальність відеоінформації зумовлена вимогами систем штучного інтелекту щодо візуальної навігації та аналізу сцен, пошуку об’єктів та оцінки їхніх форм і характеристик. Ці можливості мають вирішальне значення не тільки для промислового застосування, але й для повсякденного домашнього використання. З урахуванням тенденцій, що розвиваються, у майбутньому очікується значне зростання сфери відеоінформації.

1. ЩО ТАКЕ КОМП'ЮТЕРНИЙ ЗІР?

Ви коли-небудь думали, як ми можемо розуміти те, що бачимо? Подібно до того, як ми бачимо, як хтось рухається, усвідомлюємо ми це чи ні, використовуючи необхідні знання, наш мозок розуміє, що відбувається, і зберігає це у вигляді інформації. Уявіть, що ми дивимося на щось і втрачаємо свідомість та втрачаємо це знання. Страшно навіть уявити. У даній роботі детально розглянуто ситуацію. того, як до нашого мозку інтерпретується зображення, які ми бачимо.

Ідея передати комп'ютеру людський інтелект та інстинкти видається досить простою. Можливо, тому що її вирішують і дуже маленькі діти, людина схильна забувати про обмеження комп'ютерів у порівнянні з нашими біологічними можливостями. Складність зорового сприйняття нескінченно варіюється і завжди динамічна у разі самої людини, а про комп'ютерний інтелект.

Історично розвиток комп'ютерного зору розпочався ще у 50-х роках. Було виділено три основні завдання: скопіювати принципи роботи людського ока (складно), скопіювати пристрій зорової кори головного мозку (дуже складно), зімітувати роботу інших частин мозку (мабуть, найскладніша проблема).

Наш мозок має здатність ідентифікувати об'єкт, обробляти дані та вирішувати, що робити, виконуючи таким чином складне завдання за частки секунди. Мозок працює за допомогою картин, які, скажімо так, "бачить" наш розум. Більшість мозку використовується саме для зору і цей процес відбувається навіть на клітинному рівні. Мільярди клітин працюють спільно, щоб виділити якісь зразки з хаотичного сигналу від сітківки.

Якщо в ньому є якась контрастна лінія під певним кутом чи швидкий рух у якомусь напрямку, нейрони починають рухатись. Мережі вищого рівня перетворюють розпізнані зразки на метазразки: наприклад, "круглий об'єкт", "рух вгору". До роботи підключається наступна мережа: "коло біле з червоними

лініями”. “Об’єкт збільшується в розмірах”. Їх цих простих, але доповнюють одне одного описів складається вся картина.

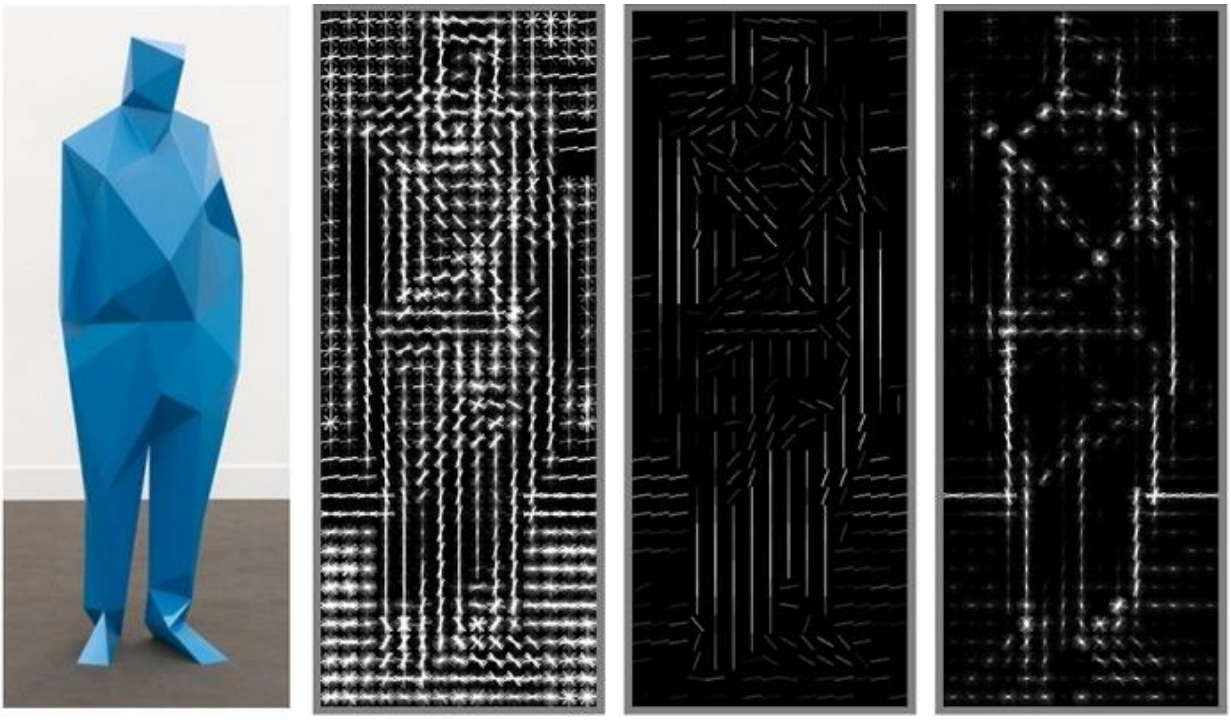

Рисунок 1.1 − “Гістограма спрямованих градієнтів” знаходить межі та інші параметри, працюючи за таким же принципом, як і області мозку, які відповідають за зір.

Ранні дослідження у сфері комп’ютерного зору вважали ці взаємозв’язки неймовірно складними. На думку вчених, взаємозв’язок вибудовувався “згори донизу” – книга схожа на те, отже, потрібно шукати такий зразок. Машина виглядає так і рухається таким чином. Для деяких об’єктів у контрольованих ситуаціях такий метод спрацьовував. Але з його допомогою неможливо описати кожен об’єкт навколо вас під різним кутом, за будь-якого освітлення, руху та інших факторів. Мета полягає в тому, щоб дозволити комп’ютерам робити те саме. Отже, цю область можна назвати об’єднанням штучного інтелекту і машинного навчання, яке включає алгоритми навчання і спеціалізовані методи для інтерпретації того, що бачить комп’ютер.

Серйозні дослідження у цій галузі розпочалися у 1960-х та 1970-х роках, коли були створені перші алгоритми для виявлення основних шаблонів у зображеннях. Однак саме поява машинного навчання та нейронних мереж стала справжнім поштовхом для розвитку цієї сфери.

Спочатку загадкова ідея, над якою досі міркують технологічні гіганти, вважалася досить простою для студентського літнього проекту тими самими людьми, які були піонерами штучного інтелекту. Технологія саме комп'ютерного зору бере початок у 1959 році, коли американський інженер Рассел Кірш [7] (жартівливо названий "винахідником пікселя", і який заклав свою "цеглинку" в основи супутникової фотографії, комп'ютерної томографії, віртуальної реальності та Instagram) з Національного бюро стандартів, відсканував зображення свого сина на комп'ютер. Подібність малолітнього сина Кірша стала першим цифровим зображенням у всій своїй зернистій красі, і воно започаткувало абсолютно нову галузь інформатики та розвитку ШІ (штучного інтелекту). Р. Кірш був визнаний розробником першого цифрового сканера зображень. У 1966 році, Сеймур Пейперт і Марвін Мінськ [8, 9] із групи штучного інтелекту (ШІ) Массачусетського технологічного інституту розпочали проект, метою якого було створення системи, здатної аналізувати сцену та ідентифікувати об'єкти на ній. Через кілька років Ларрі Дж. Робертс написав свою докторську дисертацію про здатність використовувати двовимірні зображення для отримання тривимірної інформації про тверді предмети [4]. Його робота заклала курс на десятиліття прогресу та позширила його славу як батька Інтернету. Завдяки цим піонерам комп'ютерні інженери у всьому світі шукали нові способи перетворення реальних зображень у дані, які комп'ютер може розпізнавати, сортувати, обробляти та реагувати на них. У 1980 році був представлений неокогнітрон, рання версія сучасної CNN Куніхіко Фукусіма [5]. На початку 1990-х років відеоспостереження з'явилося у банкоматах, а менш ніж через десятиліття дослідники Массачусетського технологічного інституту [8, 9] представили першу систему розпізнавання обличчя у реальному часі. Саме з цього моменту стало можливим дослідникам, інженерам та розробникам прискорити темп, намагаючись створити найкращі рішення для комп'ютерного зору. Корпорації такі як Google, Facebook, Apple, Amazon та навіть міжнародні уряди ввійшли до сферу розробки технологій комп'ютерного зору від розпізнавання облич до безпілотних автомобілів.

1.1 Глибоке навчання

Наука, що лежить в основі комп'ютерного зору, обертається навколо штучних нейронних мереж. Простими словами алгоритми, натхненні людським мозком, які навчаються, використовуючи великі обсяги наборів даних, щоб максимально точно клонувати людські інстинкти. Ці алгоритми мають чудову точність, у деяких завданнях навіть перевищуючи людський рівень. Глибокий зір, що є лише підмножиною глибокого навчання, є рушійною силою комп'ютерного зору.

Від селфі до пейзажних зображень, сьогодні ми наповнені різними фотографіями і це лише невелика кількість завантажених зображень. Подумайте, яке число вийде, якщо порахувати усі зображення, що зберігаються в телефонах, переглядаються відеоролики на YouTube та надсилаються спам-повідомлення, і це щодня. Знову ж таки, це лише частина – комунікації, засоби масової інформації та розваги, Інтернет речей активно роблять свій внесок. Цей широко доступний візуальний контент вимагає аналізу та розуміння, і комп'ютерний зір допомагає в цьому, навчаючи машини "бачити" ці зображення та відео.

Складне програмне забезпечення вимагає, по-перше, обчислювальних ресурсів і по-друге – енергії, тому актуальною є розробка простих автономних додатків із мінімальними вимогами до програмного та апаратного забезпечення.

Комп'ютерний зір − (Computer Vision, CV) − це область штучного інтелекту, що допомагає комп'ютерам і системам розуміти відео та зображення шляхом розпізнавання візуальних образів та виявлення об'єктів інформації. В алгоритмах комп'ютерного зору використовуються як класичні методи машинного навчання, так і глибокі нейронні мережі.

Комп'ютерний зір дозволяє машинам бачити, спостерігати і розуміти. Комплексні рішення на базі Computer Vision вже активно використовують у різних сферах бізнесу та виробництва.

Рисунок 1.2 – Демонстрація виявлення об’єктів.

Обробка зображень

Обробка зображення – це метод виконання деяких операцій із зображенням, щоб отримати покращене зображення або витягти з нього деяку корисну інформацію. Обробка зображень виконує аналіз та обробку оцифрованого зображення (дуже часто з метою покращення його якості).

Обробка зображення має такі етапи:

- Імпорт зображення.
- Аналіз та обробка зображення.
- Видача результату (змінене зображення або звіт).

Прикладами обробки зображень є читання зображення, вилучення значень RGB пікселя, зміна розміру зображення, обертання зображення, малювання прямокутника та ін.

Як працює комп’ютерний зір?

Комп’ютер ніколи не буде побачити так, як ми, тому що комп’ютерам бракує очей, щоб сприймати та передавати вхідні дані в мозок. Тому технологія комп’ютерного зору спирається на складну. симфонію даних та алгоритмів, які відображають те, як людські очі сприймають зображення та передають їх у мозок.

Важливо відзначити, що ми все ще не зовсім розуміємо, як працює людський мозок. Більшість людей мають елементарне розуміння того, що очі отримують інформацію, перекладають її та передають повідомлення нашому

мозку. Проте нейробіологи можуть сказати вам, що людський зір є набагато складнішим і що ми все ще маємо обмежене розуміння того, як працює наш мозок.

Ці обмеження в розумінні передаються інженеру комп'ютерного зору, який намагається навчити комп'ютер бачити. Дані та алгоритми, які використовуються для навчання комп'ютера "бачити" та інтерпретувати зображення, залишаються обмеженими нашим розумінням того, як людські очі та мозок взаємодіють.

Комп'ютерний зір працює аналогічно роботі нашого мозку та очей: щоб отримати будь-яку інформацію, наші очі спочатку фіксують це зображення, а потім відправляють цей сигнал у наш мозок. Потім наш мозок обробляє ці сигнальні дані і перетворює їх на значну повну інформацію про об'єкт, а потім розпізнає/класифікує цей об'єкт на основі його властивостей.

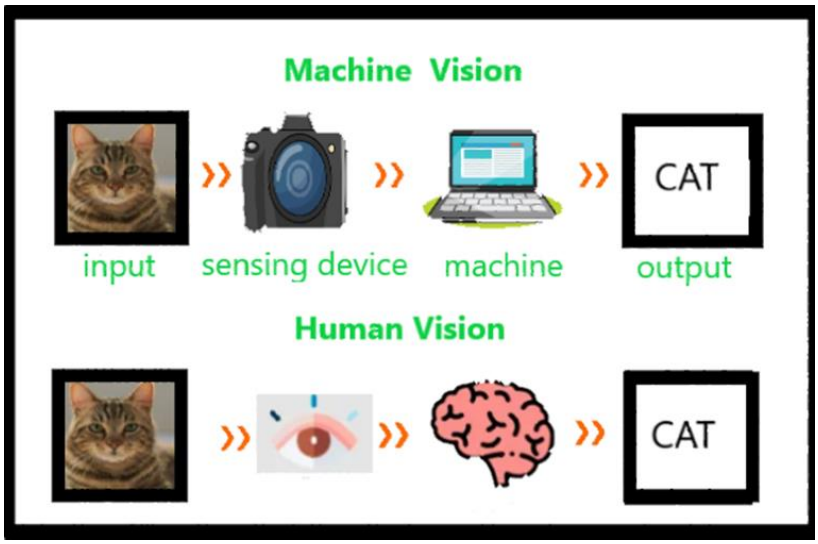

Рисунок 1.3 – Класифікація об'єкта комп'ютером та людиною

Подібно до роботи з комп'ютерним зором, в CV використовується камера для захоплення об'єктів, а потім вона обробляє ці візуальні дані за допомогою деяких алгоритмів розпізнавання образів і на основі цієї властивості ідентифікує об'єкт. Але перш ніж передати невідомі дані пристрою, програма навчається на величезній кількості даних із візуальним маркуванням. Ці позначені дані дозволяють пристрою аналізувати різні закономірності у всіх точках даних і

можуть співвідноситися з цими мітками. **Наприклад,** комп'ютеру надають аудіодані тисяч пісень птахів. У цьому випадку комп'ютер навчається на цих даних, аналізує кожен звук, висоту звуку, тривалість кожної ноти, ритм тощо і, отже, ідентифікує закономірності, схожі на спів птахів, і генерує модель. В результаті ця модель розпізнавання звуку тепер може точно визначати, чи містить звук птахів спів чи ні для кожного вхідного звуку.

Таблиця 1 − Еволюція комп'ютерного зору

№	Часовий період	Еволюція комп'ютерного зору
1.	**2**	**3**
1	2010-2015 рр.	− розробка алгоритмів глибокого навчання. пізнаване зображення. − впровадження згорткових нейронних мереж (CNN) для класифікації зображень. − використання комп'ютерного зору в автономних транспортних засобах для виявлення об'єктів та навігації.
2	2015-2020 рр.	− досягнення у виявленні об'єктів у реальному часі за допомогою таких систем, як YOLO ("Ви дивіться лише один раз"). − у технології розпізнавання осіб, що використовується в різних програмах, таких як розблокування смартфонів та спостереження. − інтеграція комп'ютерного зору до систем доповненої реальності (AR) та віртуальної реальності (VR). − використання комп'ютерного зору у медичній візуалізації для діагностики захворювань.
3	2020-2025 рр. (прогноз)	− подальші досягнення в області виявлення об'єктів та розпізнавання зображень у реальному часі. − більш складне використання комп'ютерного зору в автономних транспортних засобах. − ширше використання комп'ютерного зору у охороні здоров'я для раннього виявлення та лікування захворювань. − інтеграція комп'ютерного зору у більшу кількість споживчих товарів, таких як пристрої для розумного будинку.

1.2 Галузь застосування комп'ютерного зору

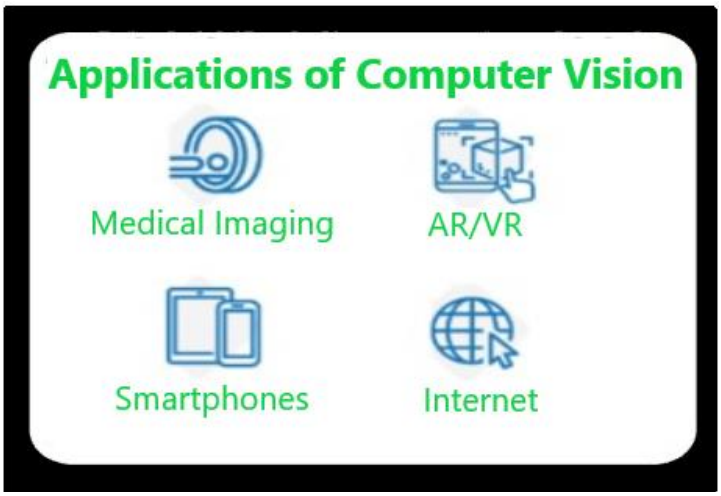

Рисунок 1.4 − Галузь застосування CV

Охорона здоров’я. У медицині моделі комп’ютерного зору використовуються для автоматизації аналізу медичних зображень, виявлення захворювань і відхилень, таких як рентгенівські знімки, МРТ та КТ, що дозволяє точно діагностувати та класифікувати різні захворювання. На перспективу застосування комп’ютерного зору надасть можливість глибшого впровадження у медичну сферу, де системи зможуть діагностувати різні захворювання на основі візуальних даних та забезпечувати більш точне лікування пацієнтів. Лікарі та хірурги зможуть проходити навчання медичним навичкам за допомогою віртуальних платформ, що дозволяє їм безпечно імітувати операції та процедури перед тим, як братися за реальні випадки.

Створення розумних міст. Нарешті, комп’ютерний зір відіграватиме важливу роль у концепції розумних міст: для моніторингу транспортного руху, управління енергоспоживанням, забезпечення безпеки та оптимізації міської інфраструктури.

Автомобільна промисловість. У сфері транспорту системи комп’ютерного зору застосовуються для виявлення та запобігання аваріям, моніторингу дорожнього руху та розпізнаванню номерних знаків. У безпілотних автомобілях комп’ютерний зір використовується виявлення об’єктів, утримання смуги руху і розпізнавання дорожніх знаків. Це допомагає зробити автономне водіння безпечним та ефективним. Також CV можуть використовуватися в системах, що дозволяють автономним транспортним засобам визначати маршрути та взаємодіяти з довкіллям. CV основа всіх систем автопілотів. Найкращий приклад – автопілот Tesla. А покращення здібностей систем розпізнавання навколишнього середовища та їх адаптацію до різних умовам підвищить безпека та ефективність таких технологій.

Роздрібна торгівля. Комп’ютерний зір використовується в роздрібній торгівлі для управління запасами, запобігання крадіжкам та аналізу поведінки клієнтів. CV може автоматично відстежувати товари на полицях і відстежувати переміщення покупців, а менеджер отримує аналітичні звіти в режимі реального часу. CV також допомагає розпізнавати на фотографіях цінники, визначити ціни

та співвідношення її зі схемою розкладки товару званої планограмою. Також CV може знаходити на фото промоакції та визначати їх актуальність. Одна з причин необхідності автоматизації − підсвічування на устаткуванні (світлодіодна стрічка), яка засвічує фото та заважає розпізнавати товари. Автоматизація фото торговельного обладнання (реалограм) допоможе прискорити процес оцінки реалограм та уникнути людського фактор А.

Сільське господарство. У сільському господарстві комп'ютерний зір використовується для моніторингу посівів та виявлення хвороб. Це допомагає виявити хворі рослини та області, що потребують більшої уваги. Тобто фермери, які займаються великою виробничою діяльністю, можуть оптимізувати свої справи за допомогою програмного забезпечення комп'ютерного зору, яке слідкуватиме за тваринами та посівами. Найлегше своєчасно виявляти зараження комахами та спалахи хвороб, відстежувати врожайність та оптимізувати свою команду. Фермери, які працюють з дефіцитом персоналу, можуть автоматизувати різні дії, включаючи збирання врожаю, прополку та посів.

Виробництво. В області виробництва моделі комп'ютерного зору застосовуються для автоматизації контролю якості, при виявленні дефектів виробництва продукції, забезпечуючи високу точність у виявленні дефектів продукції, яку важко помітити людським оком. Крім того, роботи, оснащені системами CV, здатні швидко і точно виконувати завдання щодо сортування та пакування товарів, спираючись на візуальну інформацію, що підвищує ефективність виробничих процесів.

Безпека та спостереження. Комп'ютерний зір використовується у камерах відеоспостереження для виявлення підозрілих дій, розпізнавання облич та відстеження об'єктів. Він може попередити працівників служби безпеки під час виявлення загрози. Крім того, розпізнавання осіб з використанням комп'ютерного зору може використовуватися для ідентифікації особи та забезпечення контролю доступу у зонах безпеки. Розвиток моделей комп'ютерного зору буде спрямовано також на підвищення ефективності систем

безпеки, включаючи розробку складних методів розпізнавання осіб, виявлення незвичайного поведінки та запобігання можливих інцидентів.

Доповнена та віртуальна реальність. В AR та VR комп'ютерний зір використовується для відстеження рухів користувача та взаємодії з віртуальним середовищем. Це допомагає створити більш захоплюючий досвід. Технології відстеження руху очей за допомогою комп'ютерного зору використовується не тільки в ігрових ноутбуках, а й у звичайних та корпоративних комп'ютерах, для того щоб ними могли керувати люди, які не можуть скористатися руками.

Смартфони: всі фотофільтри (включаючи фільтри анімації в соціальних мережах), сканери QR-кодів, побудова панорам, комп'ютерна фотографія, детектори облич, детектори зображень, такі як (Google Lens, Night Sight), які ми використовуємо, є програмами комп'ютерного зору.

Інтернет/соціальні мережі. Комп'ютерний зір використовується в соціальних мережах для розпізнавання зображень, пошук зображень, картографування, підписи до фотографій, категоризація відео та багато іншого. Ідентифікація об'єктів, місць та людей на зображеннях та надавати відповідні теги.

Дрони. У дронах комп'ютерний зір використовується для навігації та відстеження об'єктів. Це допомагає уникати перешкод та відстежувати цілі. Крім військових місій, радіокеровані дрони отримали визнання у цивільній сфері та успішно застосовуються:

- в умовах дикої природи для розвідки та збору даних. Наприклад, дрони зі штучним інтелектом, які розпочали патрулювання узбережжя Австралії. Мета безпілотників Little Ripper – виявити у прибережних водах плаваючу акулу (при цьому не сплутати її з дельфіном чи серфінгістом) та надіслати сигнал оператору на берег;
- сільському господарстві для відстеження стану культур та для обробки посівів. Хороші результати показав, наприклад, український стартап Kray Technologies, який представив агродрон для обслуговування від 300 до 500 га на день;

- у логістиці для доставки дрібних вантажів на стадії “останнього кілометра. Першовідкривачами в цій галузі стала американська компанія Amazon, яка запропонувала не лише доставляти дрібні посилки за допомогою радіокерованих дронів, але й подала заявку на патент багаторівневої парковки- хаба для безпілотників;
- у сфері охорони правопорядку для контролю переміщення та відстеження окремих осіб та техніки. Прикладом може бути поліцейський підрозділ дронів у Британії та тандемі наземного робота-патрульного та літаючого безпілотника в Сінгапурі.

Сфера застосування радіокерованих дронів постійно розширюється і вже навіть звичайні аматорські моделі квадрокоптерів мають функції професійних дронів.

Спорт. У спорті комп’ютерний зір використовується для відстеження гравців, аналізу ігор та створення яскравих моментів. Він може відстежувати рухи гравців та м’яча та надавати детальну статистику. Автоматизація суддівства. Автоматизовані системи суддівства можуть підтримати процес прийняття рішень під час ігор:

- у зменшення людських помилок: це мінімізує ймовірність того, що судді щось пропустять або неправильно судитимуть ігри;
- підвищена справедливість: гарантує, що всі команди будуть оцінюватися за однаковими стандартами;
- підвищена прозорість: пропонує фанатам та командам більш чітке розуміння процесу прийняття рішень;
- підтримка гравців і тренерів: у меншає кількість випадків несправедливих покарань чи пропущених порушень.

Популярні моделі Computer Vision

У галузі Computer Vision (CV) існує безліч популярних моделей, кожна з яких розроблена для вирішення конкретних завдань. Розглянемо деякі з найбільш відомих і широко використовуваних моделей Computer Vision:

Таблиця 2 − Моделі Computer Vision для обробки зображень

№	Моделі Computer Vision	Використання моделей комп'ютерного зору для обробки зображень
1	**2**	**3**
1	Convolutional Neural Networks (CNN)	CNN є одними з найпоширеніших моделей у CV. Завдяки своїй здатності точно виділяти ознаки з візуальних даних, вони успішно застосовуються для розпізнавання та класифікації зображень.
2	Inception (GoogLeNet)	Відома своєю інноваційною архітектурою, модель Inception використовує паралельні операції згортки різних розмірів для миттєвого вилучення ознак об'єктів із зображень.
3	OpenPose	Ця модель спеціалізується на виявленні та відстеженні ключових точок людського тіла, що робить її досить ефективною в сферах безпеки та віртуальної реальності.
4	Recurrent neural network (RNN)	RNN – вид нейронних мереж, де зв'язки між елементами утворюють спрямовану послідовність. Завдяки цьому з'являється можливість обробляти серії подій у часі або послідовні просторові ланцюжки. Модуль, що повторюється, в стандартній RNN складається з одного шару
5	Long short-term memory (LSTM)	Структура LSTM також нагадує ланцюжок, але модулі виглядають інакше. Замість одного шару нейронної мережі вони містять чотири, і ці шари взаємодіють особливим чином. Модель, що повторюється, в LSTM мережі складається з чотирьох взаємодіючих шарів.

Використання різних моделей комп'ютерного зору для обробки зображень відкриває безліч можливостей для використання в повсякденному житті. Воно покращує точність діагностики у медичних дослідженнях, забезпечує безпеку в технологіях автономного водіння, а також створює нові перспективи для більш ефективної аналітики та покращення візуального сприйняття навколишнього світу.

2. НАЯВНІ МЕТОДИ ТА ТЕХНОЛОГІЇ КОМП'ЮТЕРНОГО ЗОРУ

В останні кілька років технології комп'ютерного зору, також відома як аналіз відеоконтенту чи інтелектуальна відеоаналітика, приваблює зростаючий інтерес як з боку промисловості, так і в академічному світі. Завдяки популяризації глибокого навчання відеоаналітика дозволила автоматизувати завдання, які колись були винятковою компетенцією людей.

Нещодавні покращення в області відеоаналітики змінили правила гри: від додатків, що відстежують пробки та оповіщень у режимі реального часу, до таких, що аналізують потік клієнтів у роздрібній торгівлі для максимізації продажів, а також інші більш відомі сценарії, такі як міміка особи. розпізнавання або розумне паркування.

Камера відеоспостереження виявляє транспортні засоби в режимі реального часу, щоб розпізнавати певні події, такі як автомобільні аварії, та відповідним чином активувати оповіщення. У цій основній концепції відеоаналітики про те, як вона використовується в реальному світі для автоматизації процесів та отримання цінної інформації, а також про те, що слід враховувати при впровадженні інтелектуальних рішень відеоаналітики.

То що таке інтелектуальна відеоаналітика? Основна мета відеоаналітики – автоматичне розпізнавання тимчасових та просторових подій у відеороликах. Людина, що підозріло пересувається, недотримання дорожніх знаків, раптова поява полум'я і диму; це лише кілька прикладів того, що може виявити рішення для відеоаналітики.

Зазвичай, ці системи виконують моніторинг у режимі реального часу, при якому виявляються об'єкти, атрибути об'єктів, моделі руху або поведінка, пов'язана з контрольованим середовищем. Однак відеоаналітику також можна використовувати для аналізу історичних даних та отримання більш глибокої інформації. Це завдання судово-медичної експертизи може виявити тенденції та закономірності, які відповідають на такі бізнес-питання, як:

- Коли присутність покупців у моєму магазині досягає піку і який їхній віковий розподіл?

- Скільки разів проїжджає червоне світло та які номери автомобілів це роблять?

2.1 Програми в області відеоаналітики

Деякі програми в області відеоаналітики широко відомі широкому загалу. Одним із таких прикладів є відеоспостереження, завдання, яке існує вже близько 50 років. В принципі, ідея проста: стратегічно встановіть камери, щоб оператори могли контролювати те, що відбувається у кімнаті, зоні чи громадському просторі.

Однак на практиці це далеко не просте завдання. Оператор зазвичай відповідає більш ніж за одну камеру, і, як показали деякі дослідження, збільшення кількості камер, що підлягають контролю, негативно впливає на продуктивність оператора. Іншими словами, навіть якщо є велика кількість обладнання, що генерує сигнали, коли приходить час обробляти ці сигнали, утворюється вузьке місце через людські обмеження. Програмне забезпечення для аналізу відео може зробити істотний внесок, надаючи засоби точної обробки великих обсягів інформації.

Відеоаналітика з глибоким навчанням

Машинне навчання та, зокрема, вражаючий розвиток підходів глибокого навчання зробили революцію у відеоаналітиці/аналізі відеопотоку. Використання глибоких нейронних мереж (DNN) дозволило навчити системи відеоаналізу, що імітують поведінку людини, що призвело до зміни парадигми. Все почалося із систем, заснованих на класичних методах комп’ютерного зору (наприклад, спрацювання оповіщення, якщо зображення з камери стає занадто темним або різко змінюється), і перейшло до систем, здатних ідентифікувати конкретні об’єкти на зображенні та відстежувати їхній шлях.

Виявлення велосипедів

Виявлення велосипедів за допомогою набору інструментів глибокого навчання Luminoth. Наприклад, оптичне розпізнавання символів (OCR) десятиліттями використовувалося для вилучення тексту із зображень. У принципі, для розпізнавання номерного знака буде достатньо застосувати алгоритми оптичного розпізнавання символів безпосередньо до зображення номерного знака. У попередній парадигмі це могло спрацювати, якщо камера була розташована таким чином, що під час виконання OCR проводиться зйомка саме номерного знака.

Реальним застосуванням може бути розпізнавання номерних знаків на парковках, де камера розташована біля воріт і може знімати номерний знак, коли машина зупиняється. Однак постійно запускати розпізнавання зображень із дорожньої камери ненадійно: якщо розпізнавання тексту повертає результат, як ми можемо бути впевнені, що він дійсно відповідає номерному знаку?

У новій парадигмі моделі, засновані на глибокому навчанні, здатні визначати точну область зображення, де з’являються номерні знаки. Завдяки цій інформації OCR застосовується тільки до конкретного регіону, що розглядається, що призводить до надійних результатів.

Таблиця 3 − Приклади використання комп’ютерного зору

№	Програмне забезпечення	Обчислювальні ресурси	Виробник
1	**2**	**3**	**4**
1.	Технологія штучного сприйняття DriveNet	Технологія штучного сприйняття DriveNet від NVIDIA є самонавченим комп’ютерним зіром, що працює на основі нейронних мереж. З її допомогою лідари, радари, камери та ультразвукові датчики здатні розпізнавати оточення, дорожню розмітку, транспорт та багато іншого. Суперкомп’ютер NVIDIA Drive PX 2 вже є базовою платформою для безпілотників Tesla, Volvo, Audi, BMW та Mercedes-Benz	NVIDIA

Продовження 1. Таблиця 3

№	Програмне забезпечення	Обчислювальні ресурси	Виробник
1	**2**	**3**	**4**
2.	Штучний інтелект Co Pilot	Здатний розпізнавати обличчя, рухи губ, напрям погляду та мову. Завдяки розпізнаванню руху губ комп'ютер краще розумітиме вимовлені слова, а здатність вловлювати напрям погляду (аналіз очей, обличчя та положення голови) допоможе не тільки визначити, чи не заснув водій, а й врятує у важких ситуаціях, коли людина не помічає небезпека – наприклад, наближається позаду центром дороги мотоцикліста. Що стосується зчитування руху губ, то зараз мережі, що використовують глибинне навчання, здатні розпізнавати мову з точністю до 95%, тоді як людина розпізнає її з точністю 3%. Нині ця неймовірна здатність використовується для покращення розпізнавання мови в автомобілі, а саме у гучних ситуаціях.	NVIDIA
3.	Спеціальні камери в концепт-карі Chrysler Portal, розташовані за кермом	За допомогою комп'ютерного зору автомобіль може миттєво визначати особистість водія ще до того, як той сяде в машину, і завантажувати його улюблені музичні композиції, налаштувати крісло у потрібне положення, відрегулювати температуру тощо. Машина вміє розпізнавати не тільки водія, а й пасажирів, і так само автоматично регулювати сидіння і температуру і навіть розкривати шумопригнічуючі "кокони", в яких можна послухати улюблену музику. Такі здібності не тільки сподобаються власникам автомобілів, а й зможуть сильно вплинути на райдшерингові сервіси на кшталт Uber та Lyft.	
4.	Tobii Dynavox PCEye Mini	Являє собою пристрій розміром із кулькову ручку, який стане ідеальним та непомітним аксесуаром для планшетів та ноутбуків. Також ця технологія відстеження руху очей використовується в нових ігрових та звичайних ноутбуках Asus та смартфонах Huawei.	
5.	Технологія Google Tango	Ця технологія є комбінацією датчиків і ПЗ з комп'ютерним зором, яка може розпізнавати зображення, відео та навколишній світ у реальному часі за допомогою лінзи фотокамери.	Lenovo
6.	Changhong H2	Перший смартфон із вбудованим молекулярним сканером. Він збирає світло, що відбивається від об'єкта і розбивається на спектр, а потім аналізує його хімічний склад. Завдяки програмному забезпеченню, що використовує комп'ютерний зір, отримана інформація може використовуватися для різних цілей – від виписування ліків та підрахунку калорій до визначення стану шкіри та розрахунку рівня вгодованості.	

Продовження 2. Таблиця 3.

№	Програмне забезпечення	Обчислювальні ресурси	Виробник
1	**2**	**3**	**4**
7.	FLIR Duo та Duo R	Пристрої FLIR Duo та Duo R зовні нагадують GoPro або іншу екшн -камеру. Їх можна прикріпити до будь-якого дрону та відстежувати тепло в різних ділових та побутових ситуаціях – наприклад, можна виявити витік в ізоляції даху або вести повітряну топографічну зйомку полів та нафтових родовищ.	
8.	Intelligent Vision System від компанія ITR I	Система Intelligent Vision System, яка використовує глибинне навчання та комп'ютерний зір, щоб роботи могли розрізняти об'єкти різного розміру (фігурки, чашки) та визначати їхнє положення. Розпізнавши об'єкт, робот зможе взяти його та принести у потрібне місце. Такі навички відмінно використовуються для обслуговування столиків у ресторані або для гри в шахи.	
9.	Alexa від Amazon, Google Home та інші цифрові помічники та роботи, доступні на ринку, на зразок LG Hub та Kuri від Mayfield Robotics	Роботи мають базові навички комп'ютерного зору і можуть визначити, хто з ними розмовляє, або вигнати собаку з дивана.	
10.	Пристрій eyeSight's Singlecue Gen 2	Пристрій eyeSight's Singlecue Gen 2 використовує комп'ютерний зір (розпізнавання жестів, аналіз обличчя, визначення дій) і дозволяє керувати за допомогою жестів телевізором, розумною системою освітлення та холодильниками.	
11.	Пристрій FridgeCam	Елегантний пристрій FridgeCam від Smarter кріпиться до стінки холодильника і може визначати, коли закінчується термін придатності, повідомляти, що саме знаходиться в холодильнику, і навіть рекомендувати рецепти страв із вибраних продуктів. Працює з додатком, який аналізує зображення із вбудованої в холодильник камери та повідомляє, коли у вас закінчуються певні продукти	Smarter

3. РЕАЛІЗАЦІЯ ВИМІРЮВАННЯ ШВИДКОСТІ ЗМІНИ ЗОБРАЖЕННЯ

3.1 Загальний опис підходу

Вимірюється частота рухів людини з використанням відеокамери і тому змінюємо швидкість відтворення відео, на якому відображається рух навколо бігової доріжки.

Швидкість відтворення змінюється раз на 0,5 секунди (наприклад). Ця швидкість обчислюється як пропорційна кількості кадрі, тобто. чим швидше рухається людина, тим вважається що вона біжить, відповідно більше кадрів з треку відтворюється. Визначення частоти рухів відповідно залежить від обробки зображення, щоб зрізати поточне зображення з попереднім. Зрозуміло на обробку та порівняння кадрів бажано витрачати якнайменше час. Зрозуміло що чим менше масив пікселів, тим менше час його обробку. Щоб цього можна застосувати згортку на кшталт Фур'є-перетворення з децимацією, найкраще логарифмічну децимацію, так зване БПФ (швидке перетворення Фур'є). Тоді замість порівняння масиву 1000x1000 пікселів чисел ми отримуємо приблизно е порівняння чисел масивом 10x10. Що дає прискорення обробки 1000000x100 тобто у 10000 разів.

Порівняння зображень (і захоплення з камери), визначення швидкості відтворення та власне відтворення відео будемо робити у трьох окремих потоках. Навіщо організуємо подію від трьох незалежних таймерів для трьох незалежних процесів.

3.2 Структура програмного забезпечення

3.2.1 Математична модель розпізнавання руху

Математична модель розпізнавання руху передбачає такі дії:

Крок 1 – згладжування

Крок 2 – знаходимо Фур'є-образ згладженого кадру

Крок 3 − обчислюємо відносну відстань за формулою (2)

Крок 4 − якщо виявляється що ця сума більша за деяку величину, то зараховуємо рух.

Порахувавши описані вище події в одиницю часу ми можемо судити про деяку швидкість суб'єкта спостереження.

За формулою (1) математична модель розпізнавання руху має такий вигляд:

$$Outp_k = \frac{1}{N} \sum_{j}^{log_2 N} P_j cos \frac{2k * 2^j}{N^2} \qquad (1)$$

де $k = 1 \dots NF$, $a\ NF \ll N$

NF − розмір Фурье образу

N − об'єм зображення кадру

$Outp_k$ − Фур'є образ

P_j − оригінал масиву пікселя

Обчислюємо відносну відстань:

$$S = \sqrt{\frac{\sum_i (x_i - E_i)^2}{\sum_i E_i^2}} \qquad (2)$$

де x_i − це поточний Фур'є образ ,

E_i − еталонний (попередній) Фур'є образ

Формула (2) являє собою відносну різницю в частках одиниці між образами зображень x та E. Тоді можемо припустити, що швидкість віртуальної прогулянки пропорційна кількості рухів об'єкта, що спостерігається в одиницю часу. Це може бути використане для визначення швидкості відтворення треку віртуальної прогулянки.

3.2.2 Алгоритми

1-й Алгоритм захоплення камери руху такий:

− захоплення камери кадру;

- згладжування;
- отримання Фур'є-образу поточного кадру;
- обчислення різниці між поточним та попереднім Фур'є-образами;
- і, якщо ця різниця більше ніж константа `threshold`, то збільшуємо лічильник `moveCounter` на 1.

2 -й алгоритм. Захоплення швидкості `SpeedTimer`.

Операція, що викликається, відбувається один раз на секунду (1 сек.). Якщо `moveCount=0` (об'єкт не рухається), тобто немає руху протягом однієї секунди (1 сек.), то таймер відтворення зупиняється (stop -кадр). Інакше частота відтворення кадру встановлюється пропорційно. Період TimerFilm – зворотньо пропорційний.

Власне 3-й алгоритм `TimerFilm` дуже простий.

Взяти з фільму черговий кадр та показати його на весь екран. Чим частіше таймер викликає подію, тим швидше швидкість віртуальної прогулянки. Важлива примітка – всі три алгоритми працюють асинхронно, тобто одночасно.

3.2.3 Реалізація програмного коду застосунку "Віртуальна прогулянка"

Застосунок є керуючою формою, за допомогою якої можна вибирати трек і відповідно запускати віртуальну прогулянку.

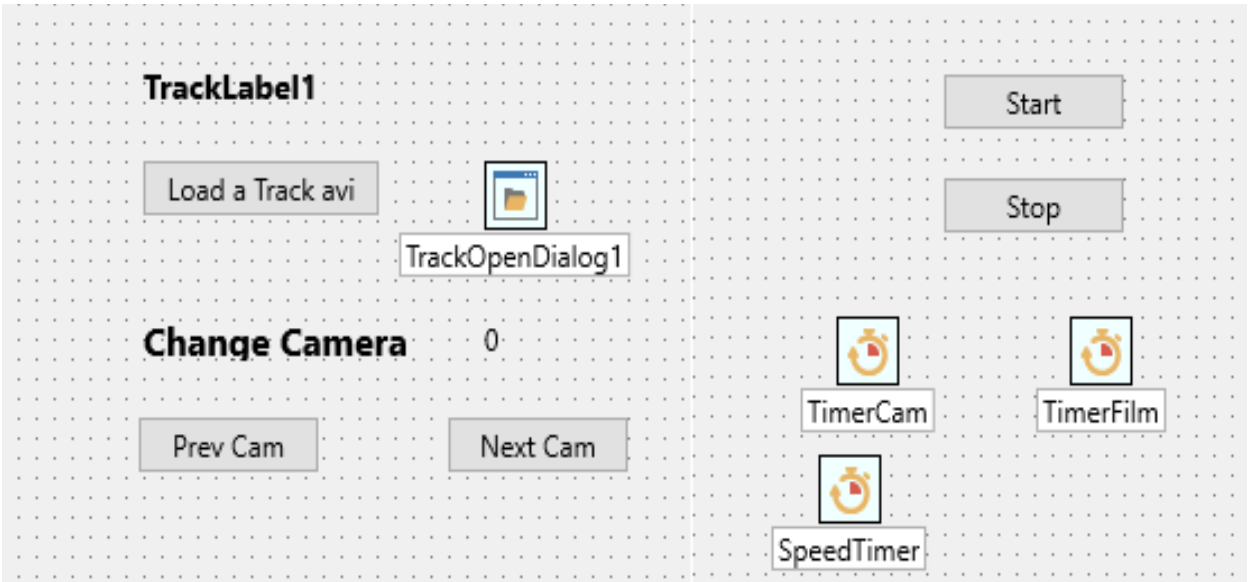

Рисунок 3.1 – Інтерфейс форми

Опис роботи кожного елемент форми.

Start

Рисунок 3.2 − Start

Програмний код кнопки Start виглядає так:

```
procedure TForm1.btnStartClick(Sender: TObject );
begin
 StartVirtualTour ();
end;

procedure TForm1.StartVirtualTour;
begin
InitCV;
moveCounter:=0;
  TimerFilm.Enabled:=true; // запуск потоків таймерів
 TimerCam.Enabled:=true;
 SpeedTimer.Enabled:=true;
// WindowState:= TWindowState.wsMinimized;
      cvNamedWindow (VIDEO_WIN,CV_WINDOW_NORMAL); // створення вікна відтворення, розгорнутого на весь екран
cvSetWindowProperty(VIDEO_WIN,CV_WND_PROP_FULLSCREEN,CV_WINDOW_FULLSCREEN);
      cvSetMouseCallback ( VIDEO_WIN,OnMouseKey );
      SendToBack;
end;
```

Stop

Рисунок 3.3 − Stop

За аналогією програмний код кнопки Stop виглядає так:

```
procedure TForm1.btnStopClick(Sender: TObject );
begin
StoptVirtualTour (); // Зупинка всіх таймерів, закриття всіх вікон
 Application.Terminate;
Halt
end;

procedure TForm1.StoptVirtualTour;
begin
 SpeedTimer.Enabled:=false;
 TimerFilm.Enabled:=false;
 TimerCam.Enabled:=False;
// cvDestroyWindow (VIDEO_WIN);
  cvDestroyAllWindows;
  BringToFront
end;
```

Таймер TimerCam

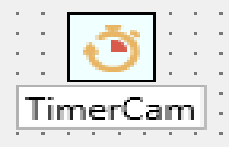

Рисунок 3.4 – TimerCam

TimerCam – це таймер за подією від якого здійснюється відеозахоплення. Подія по таймеру із захоплення камери TimerCam відбувається так:

```
procedure TForm1.TimerCamTimer(Sender: TObject );
begin

 CaptuireFrame; // захоплення кадру з камери та згладжування
  FastFurieTransform
(capframe^.imageData,fframes[fcount],frame^.imageSize,FOURIE_IMAGESIZE);
// Обчислення Фур'є образу

  inc ( fcount );
if( fcount >1) then fcount:=0;
if (length( fframes [0])>0)and(length( fframes [1])>0) then
begin
      var ImageDif:= CalcImageDifPercent (); // Обчислення відмінності між
fframes
      if ImageDif >THRESHOLD then inc ( moveCounter ); // якщо кадри різняться,
то виявлено рух
{$IFDEF SHOWCAPTURE}
// Висновок налагоджувальної інформації // Друк поверх кадру
    var  ffont: = cvFont (0.25);
    cvInitFont ( @ ffont, CV_FONT_HERSHEY_COMPLEX,1.0, 1.0, 0, 1, CV_AA);
    // використовуючи шрифт виводимо на картинку текст
     var pt:= cvPoint ( 40, 40 );
     var  valtxt:= pCVChar ( AnsiString ( FloatToStr ( ImageDif ) + ' + FloatToStr (
moveCounter ) + # 0));
// показуємо
      cvPutText ( capframe, valtxt, pt, @ffont, CV_RGB(150, 0, 150));
//************
{$ENDIF}
end;
{$IFDEF SHOWCAPTURE}
if Assigned( capframe ) then cvShowImage ('capture', capframe );
  cvReleaseImage ( capframe );
{$ENDIF}
end;
```

Таймер SpeedTimer

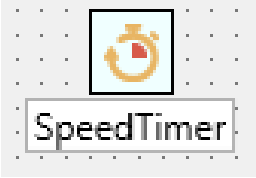

Рисунок 3.5 − SpeedTimer

Таймер SpeedTimer бере кількість рухів за одиницю часу та обчислює інтервал для таймера TimerFilm.

```
procedure TForm 1. SpeedTimerTimer ( Sender: TObject ); // визначає швидкість анімації в залежно від руху об’єкта

begin
if moveCounter =0 then TimerFilm.Enabled:=false // відсутні рухи − анімацію призупинено
else
begin

 speedPercent:=1.0* moveCounter * SpeedTimer.Interval / TimerCam.Interval;

 var interval: = coef / moveCounter / (1000.0 / SpeedTimer.Interval ); // визначаємо нову швидкість відтворення
  TimerFilm.Interval:= round(interval );
  TimerFilm.Enabled:=true;
 moveCounter:=0;
end;
end;

procedure TForm1.TimerFilmTimer(Sender: TObject );
begin
// ShowFrame
    ShowFilmFrame
end;

end.
```

Таймер TimerFilm

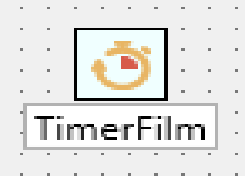

Рисунок 3.6 – TimerFilm

TimerFilm – це таймер, який відповідає за відтворення відеофайлу

Обробник події таймера для відтворення кадру у відео

```
procedure TForm1.ShowFilmFrame;
begin
if Filmcapture = nil then exit; //захист від дурня, перевірка аргументів, щоб не було вильотів,т.к. nil - це порожній покажчик
    var fframe:= cvQueryFrame ( Filmcapture ); // беремо черговий кадр із фільму
    if fframe =nil then
        cvSetCaptureProperty ( Filmcapture, CV_CAP_PROP_POS_FRAMES,0);

    cvShowImage (VIDEO_WIN, fframe ); // показуємо кадр

end;
```

Також передбачено налаштування на пристрій відеозахоплення.

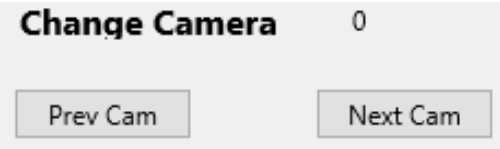

Рисунок 3.7 – Робота камери

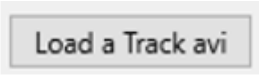

Рисунок 3.8 – Налаштування пристрою відеозахоплення

Кнопка NextCam - Перебираємо камери циклічно, поки не знайдеться робоча, відповідно **кнопка PrevCam -** перебирає в зворотний бік.

```
procedure TForm1.Button2Click(Sender: TObject ); //
begin
repeat
if Assigned(capture) then cvReleaseCapture ( capture);
inc (CVCAP);
capture:= cvCreateCameraCapture (CVCAP);
until Assigned(capture) or (CVCAP>1200);
lblCamNum.Text:= IntToStr (CVCAP)
end;

procedure TForm1.btnLoadClick(Sender: TObject );
begin
if TrackOpenDialog1.Execute then
if TrackOpenDialog1.FileName<>'' then
begin
  FilmName:=TrackOpenDialog1.FileName;
end;
end;

function TForm1.CalcImageDifPercent(): real;
begin
// fframes [ fccount ]
if (length( fframes [0])>0)and(length( fframes [1])>0) then
begin
Result:= 10* Svertka ( fframes [0], fframes [1]);
// if result<THRESHOLD then Result:=0;
  end
else
Result:=0;

end;
```

При натисканні цієї кнопки з'являється можливість вибору треку.

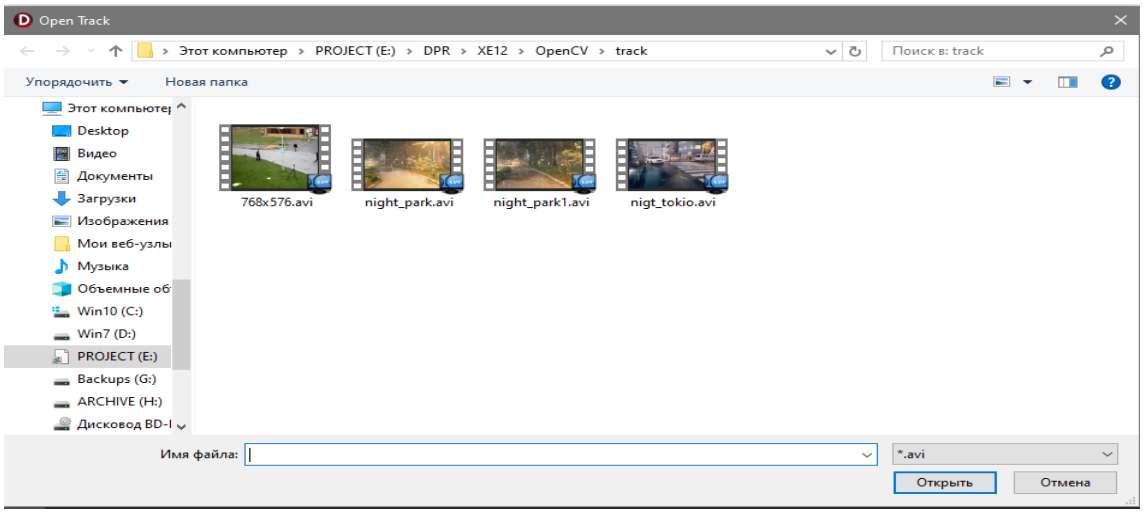

Рисунок 3.9 – Процес вибору треку

ВИСНОВКИ

У сучасному світі, де технології відіграють все більш важливу роль у повсякденному житті людей, створення застосунків, що забезпечують нові можливості та задовольняють різноманітні потреби користувачів, постають ключовими завданнями для розробників. Застосунок “Віртуальна прогулянка” являє собою одну з таких можливостей.

У процесі розробки такого застосунку враховано низку факторів, починаючи від визначення цілей і закінчуючи вибором − Open CV як відповідної технології та середовища програмування Embarcadera RAD Studio Community Edition. Одним з ключових моментів є розуміння потреб користувачів та створення інтерфейсу, який би максимально задовольняв очікування, включаючи розробку зручної навігації, доступ до інформації, що цікавить, а також можливість інтерактивної взаємодії з віртуальним оточенням за допомогою веб-камери.

Однією з головних переваг віртуальних прогулянок є їхня доступність. Користувачі можуть досліджувати різні місця, будь то культурні пам’ятки, природні пам’ятки або історичні об’єкти, не виходячи з дому. Це особливо важливо для освітніх та спортивних закладів, які можуть використовувати застосунок “Віртуальна прогулянка” як додатковий освітній та розважальний ресурс.

У першому розділі кваліфікаційної роботи проаналізовано комп’ютерний зір, його типи та підрозділи, а також сфери в яких він активно застосовується.

В другому розділі проаналізовано наявні методи та технології комп’ютерного зору для подальшого використання отриманих даних у роботі.

Третій розділ присвячений практичним питанням реалізації вимірювання швидкості зміни зображення, описано загальний підхід та структуру програмного забезпечення. Також описано математичну модель розпізнавання руху та алгоритми.

Розробка програми “Віртуальна прогулянка” також включає ряд викликів. Один із головних – це складність створення реалістичного віртуального оточення. Для досягнення цієї мети застосували передові технології в галузі віртуальної реальності, використано фрагмент проходження з комп’ютерної гри Ghostwire: Tokyo, що забезпечило високий рівень графіки.

З урахуванням правильного розвитку підходу, розуміння потреб користувачів та використання передових технологій, така програма може стати не тільки популярним та затребуваним продуктом, але й інструментом, що сприятиме збагаченню знань, культурному розвитку та розвазі мільйонів людей по всьому світу.

ПЕРЕЛІК ПОСИЛАНЬ

1. Antonini G. (Ed.). Optical Flow Estimation and Applications. InTech, 2013. 204 p.
2. Barber D. Bayesian Reasoning and Machine Learning. Cambridge University Press, 2012. 688 p.
3. Kaehler A., and Bradski, G. Learning OpenCV 4: Computer Vision в C++ and OpenCV Library. O'Reilly Media, 2019. 832 p.
4. Lawrence G. Roberts. Machine perception of three-dimensional solids. Thesis (Ph. D.) Massachusetts Institute of Technology, Dept. of Electrical Engineering, 1963. P. 82. URL: https://dspace.mit.edu/handle/1721.1/11589 (дата звернення 12.04.2024)
5. Fukushima K., and Okada M., and Hiroshige K. Neocognitron with dual C-cell layers. Neural Networks, 1994. pp. 41-47.
6. Lv Z., and Zhang, J. Visual Perception and Robotic Manipulation: 3D Object Recognition, Tracking and Hand-Eye Coordination. World Scientific, 2019. 220 нар.
7. Рассел А. Кірш (англ. Russell A. Kirsch; червня 20, 1929 – серпня 11, 2020) американський інженер Національного бюро стандартів (нині відомому як Національний інститут стандартів та технологій). URL: https:// en. Wikipedia. org / wiki / Russell _ Kirsch
8. Марвін Лі Мінські (англ. Marvin Lee Minsky; 9 серпня 1927 − 24 січня 2016) − американський дослідник у галузі штучного інтелекту, співзасновник лабораторії штучного інтелекту Массачусетського технологічного Інституту, автор праць з штучного інтелекту та філософії. URL: https:// ru. Wikipedia. org / wiki / % D 0% 9 F % D 0% B 5% D 0% B 9% D 0% BF% D 0% B 5% D 1%80% D 1%82,_% D 0% A 1% D 0% B 5% D 0% B 9% D 0% BC % D 1%83% D 1%80
9. Сеймур Пейперт (англ. Seymour Papert; 1 березня 1928, Преторія, Південна Африка − 31 липня 2016, Блу Гілл, Мен, США) – видатний математик,

програміст, психолог і педагог. Один із основоположників теорії штучного інтелекту, творець мови Logo (1968). URL: https://uk.wikipedia.org/wiki/%D0%A1%D0%B5%D0%B9%D0%BC%D1%83%D1%80_%D0%9F%D0%B5%D0%B9%D0%BF%D0%B5%D1%80%D1%82

10. Shanmugamani R. Deep Learning для Computer Vision. Packt Publishing, 2019. 664 p.
11. Tan DJ. Motion Detection and Object Tracking Using OpenCV. Packt Publishing, 2015. 282 p.
12. Smith J., and Jones A. Motion Detection Techniques in Computer Vision. International Journal of Computer Vision, 2020. Vol. 25 (3), pp. 112-130.
13. Kim S., and Park M. Deep Learning Approaches для Motion Detection in Video Sequences. Neural Networks, 20 18. Vol. 32 (4), pp. 567-580.
14. Huang L., and Liu G. Functional motion detection заснований на artificial intelligence. The Journal of Supercomputing, 2021. Vol. 78, pp. 4290-4329.
15. Patel S., and Gupta R. Bayesian Inference Methods for Motion Detection in Dynamic Environments. Pattern Recognition Letters, 2015. Vol. 29 (4), pp. 567-578.
16. Tang Y., and Huang Y., and Wang H., and Wang C, and Guo Q, and Yao W. Framework for artificial intelligence analysis in large-scale power grids based on digital simulation. CSEE J Power Energy Syst, 2018. Vol. 4 (4) pp. 459-468.
17. Wang L., and Zhang H. Зображення з оптичним Flow Estimation Methods for Motion Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2019. Vol. 41 (5), pp. 1234-1256.

Додаток А

Код програми unMain

```
unit unMain;
{.$DEFINE SHOWCAPTURE}
interface

uses
  System.SysUtils, System.Types, System.UITypes, System.Classes,
System.Variants,
  FMX.Types, FMX.Controls, FMX.Forms, FMX.Graphics, FMX.Dialogs,
  FMX.Controls.Presentation, FMX.StdCtrls,
  ocv.highgui_c,
  ocv.core_c,
  ocv.core.types_c,
  ocv.utils,
  ocv.imgproc_c, ocv.imgproc.types_c,
windows,
   FMX.Objects, FrameComparement;
// uResourcePaths;

const
VIDEO_WIN='original';
CV_CAP = 0;
TourName = '..\..\track\night_park.avi';
// TourName = '..\..\track\768x576.avi';
THRESHOLD=0.09;
FOURIE_IMAGESIZE=1024;
coef = 1.0 * 70; //speed

type
TForm1 = class( TForm )
   btnStart: TButton;
   btnStop: TButton;
TimerCam: TTimer;
TimerFilm: TTimer;
SpeedTimer: TTimer;
TrackOpenDialog1: TOpenDialog;
TrackLabel1: TLabel;
   btnLoad: TButton;
Button1: TButton;
Button2: TButton;
```

```
Label1: TLabel;
   lblCamNum: TLabel;
procedure btnStartClick (Sender: TObject );
procedure FormDestroy (Sender: TObject );
procedure TimerCamTimer (Sender: TObject );
procedure TimerFilmTimer (Sender: TObject );
procedure FormCreate (Sender: TObject );
procedure btnStopClick (Sender: TObject );
procedure SpeedTimerTimer (Sender: TObject );
procedure btnLoadClick (Sender: TObject );
procedure Button2Click(Sender: TObject );
procedure Button1Click(Sender: TObject );
private
   FFilmName: string;
procedure SetFilmName ( const Value: string);

{Private declarations}
public
capture: PCvCapture;
   Filmcapture: PCvCapture;
width: Double;
height: Double;
frame, capframe: PIplImage;

{Public declarations}
   speedPercent:real;

   fframes:array [0..1] of TRealArray;
   fcount:integer;
   moveCounter:integer;

    CVCAP:integer;
property FilmName:string read FFilmName write SetFilmName;
PROCEDURE InitCV ();
PROCEDURE StartVirtualTour ();
PROCEDURE StoptVirtualTour ();
procedure CaptuireFrame;
procedure ShowFrame;
procedure ShowFilmFrame;
function CalcImageDifPercent ():real;
end;

var
Form1: TForm1;
```

```
implementation

{$ R *. fmx }

procedure TForm1.btnStartClick(Sender: TObject );
begin
 StartVirtualTour ();
end;

procedure TForm1.btnStopClick(Sender: TObject );
begin
StoptVirtualTour ();
 Application.Terminate;
Halt
end;

procedure TForm1.Button1Click(Sender: TObject );
begin
repeat
if Assigned(capture) then cvReleaseCapture ( capture);
dec (CVCAP);
capture:= cvCreateCameraCapture (CVCAP);
until Assigned(capture) or (CVCAP=0);
lblCamNum.Text:= IntToStr (CVCAP)
end;

procedure TForm1.Button2Click(Sender: TObject );
begin
repeat
if Assigned(capture) then cvReleaseCapture ( capture);
inc (CVCAP);
capture:= cvCreateCameraCapture (CVCAP);
until Assigned(capture) or (CVCAP>1200);
lblCamNum.Text:= IntToStr (CVCAP)
end;

procedure TForm1.btnLoadClick(Sender: TObject );
begin
if TrackOpenDialog1.Execute then
if TrackOpenDialog1.FileName<>'' then
begin
   FilmName:=TrackOpenDialog1.FileName;
end;
```

```
end;

function TForm1.CalcImageDifPercent(): real;
begin
// fframes [ fccount ]
if (length( fframes [0])>0)and(length( fframes [1])>0) then
begin
Result:= 10* Svertka ( fframes [0], fframes [1]);
// if result<THRESHOLD then Result:=0;
end
else
Result:=0;

end;

procedure TForm1.CaptuireFrame;
begin

frame:= cvQueryFrame (capture);
if Assigned(frame) then
begin
// if not Assigned( capframe ) then
        capframe:= cvCreateImage ( cvGetSize (frame), frame^.depth, frame^.
nChannels );
// cvCopyImage (frame, capframe );
      cvSmooth (frame, capframe, CV_GAUSSIAN, 15, 15);
// cvCopy (frame, capframe );
// cvReleaseImage (frame);
end;



end;

procedure TForm1.FormCreate(Sender: TObject );
begin
CVCAP: = 0;
 speedPercent:=0.0;
 fcount: = 0;
 FilmName:= TourName;
// InitCV;
end;

procedure TForm1.FormDestroy(Sender: TObject );
```

```
begin
if Assigned( capframe ) then
     cvReleaseImage ( capframe );
if Assigned( Filmcapture ) then
       cvReleaseCapture ( Filmcapture );
   cvReleaseCapture (capture);

  cvDestroyAllWindows;
end;

procedure TForm1.InitCV;
begin
capture:= cvCreateCameraCapture (CV_CAP);
// if not Assigned(capture) then Halt;
// дізнаємось ширину і ви з того кадру
width:= cvGetCaptureProperty (capture, CV_CAP_PROP_FRAME_WIDTH);
height:= cvGetCaptureProperty (capture,
CV_CAP_PROP_FRAME_HEIGHT);
     capframe:= nil;

if FileExists ( FilmName ) then
begin
     var fname:= pCVChar ( AnsiString ( FilmName ));
     Filmcapture: = cvCreateFileCapture ( fname );

end;
end;

procedure TForm1.SetFilmName( const Value: string);
begin
  FFilmName:= Value;
TrackLabel1.Text:= FFilmName;
//TrackOpenDialog1.FileName:= FFilmName
end;

procedure TForm1.ShowFilmFrame;
begin
if Filmcapture =nil then exit;

     var fframe:= cvQueryFrame ( Filmcapture );
if fframe =nil then
          cvSetCaptureProperty ( Filmcapture,
CV_CAP_PROP_POS_FRAMES,0);
      // Тут можна вставити
```

```
// процедуру обробки


// показуємо кадр
      cvShowImage (VIDEO_WIN, fframe );
// cvReleaseImage ( fframe )

end;

procedure TForm1.ShowFrame;
begin
// показуємо
if Assigned( capframe ) then
       cvShowImage ('capture', capframe );
end;




procedure OnMouseKey (event: Integer; x, y, flags: Integer; param: Pointer);
cdecl;
begin
case event of
CV_EVENT_MOUSEMOVE:;
CV_EVENT_LBUTTONUP,CV_EVENT_RBUTTONUP:
Form1.StoptVirtualTour;
end
end;

procedure TForm1.StartVirtualTour;
begin
InitCV;
moveCounter:=0;
  TimerFilm.Enabled:=true;
 TimerCam.Enabled:=true;
 SpeedTimer.Enabled:=true;
// WindowState:= TWindowState.wsMinimized;
       cvNamedWindow (VIDEO_WIN,CV_WINDOW_NORMAL);
cvSetWindowProperty(VIDEO_WIN,CV_WND_PROP_FULLSCREEN,CV_
WINDOW_FULLSCREEN);
      cvSetMouseCallback ( VIDEO_WIN,OnMouseKey );
      SendToBack;
end;
```

```
procedure TForm1.StoptVirtualTour;
begin
 SpeedTimer.Enabled:=false;
 TimerFilm.Enabled:=false;
 TimerCam.Enabled:=False;
// cvDestroyWindow (VIDEO_WIN);
  cvDestroyAllWindows;
  BringToFront
end;

procedure TForm1.TimerCamTimer(Sender: TObject );
begin

 CaptuireFrame;
  FastFurieTransform
(capframe^.imageData,fframes[fcount],frame^.imageSize,FOURIE_IMAGESIZE);

  inc ( fcount );
if( fcount >1) then fcount:=0;
if (length( fframes [0])>0)and(length( fframes [1])>0) then
begin
     var ImageDif:= CalcImageDifPercent ();
if ImageDif >THRESHOLD then inc ( moveCounter );
{$IFDEF SHOWCAPTURE}
// Друк поверх кадру
   var  ffont: = cvFont (0.25);
   cvInitFont ( @ ffont, CV_FONT_HERSHEY_COMPLEX,1.0, 1.0, 0, 1,
CV_AA);
    // використовуючи шрифт виводимо на картинку текст
     var pt: = cvPoint (40, 40);
     var  valtxt:= pCVChar ( AnsiString ( FloatToStr ( ImageDif ) + ' + FloatToStr
( moveCounter ) + # 0));
// показуємо
    cvPutText ( capframe, valtxt, pt, @ffont, CV_RGB(150, 0, 150));
//**********
{$ENDIF}
end;
{$IFDEF SHOWCAPTURE}
if Assigned( capframe ) then cvShowImage ('capture', capframe );
  cvReleaseImage ( capframe );
{$ENDIF}
end;
```

```
procedure TForm1.SpeedTimerTimer(Sender: TObject ); // визначає кількість рухів за секунду
begin
if moveCounter =0 then TimerFilm.Enabled:=false
else
begin

 speedPercent:=1.0* moveCounter * SpeedTimer.Interval / TimerCam.Interval;

 var interval: = coef / moveCounter / (1000.0 / SpeedTimer.Interval );
  TimerFilm.Interval:= round(interval );
  TimerFilm.Enabled:=true;
 moveCounter:=0;
end;
{
if interval>1 then
begin
        TimerFilm.Interval:= round(interval);
        TimerFilm.Enabled:=true;
end
else TimerFilm.Enabled:=false;
}
end;

procedure TForm1.TimerFilmTimer(Sender: TObject );
begin
// ShowFrame
   ShowFilmFrame
end;

end.
```

Додаток Б:

Код програми FrameComparement

```
unit FrameComparement;

interface
uses ocv.core.types_c;
type TRealArray = array of real;

процедура FastFurieTransform ( X:PByte;
  var Outp:TRealArray;
N:integer; //input size
  NF:integer //output size
);

function Svertka ( const X,Etalon:TRealArray ):real;
implementation

процедура FastFurieTransform ( X:PByte;
  var Outp:TRealArray;
N:integer; //input size
  NF:integer //output size
);
    var i,j,k,L:integer;
       Max,S,sqM,sqN:real;
       stj,Pixel:integer;
begin
    SetLength ( Outp, NF );
    sqN:=NF*N;
for k:= 0 to NF-1 do
begin

s:=0;


// for j:= 0 to (N-1) div 2 do
j:=0;
      stj:=1;
while j<N div 3 do

begin
        var pX:=X+3*j;
Pixel:= pX ^+(pX+1)^+(pX+2)^;
S:=S+Pixel*cos(2*k*j/sqN);
```

```
        stj:= stj *2;
j:=stj;
end;
       Outp [k]: = S / N;

end;
end;

function MaxXValue ( const Outp:TRealArray ):real;
    var k,l,nf,mf:integer;
        Max:real;
begin
max: = Outp [0];
nf: = high ( Outp );

for k:= 0 to NF-1 do
if Max< Outp [k] then Max: = Outp [k];

result:=Max;
end;

function Svertka ( const X,Etalon:TRealArray ):real;
     var i,j,N:integer;
        maxX,MAXE,S,Lx:real;
begin
if (X=nil) or (Etalon=nil) then exit(0);
if (Length(x)=0) or (Length(Etalon)=0) then exit(0);

s:=0; Lx: = 0;
N: = Length (X);
// maxX:= MaxXValue (X);
// MAXE: = MaxXValue (Etalon);
// maxX: = maxX * maxX;
// MAXE: = MAXE * MAXE;
for i:= 0 to N-1 do
begin
Lx: = Lx + sqr (Etalon [i]);
s:=s+ sqr (X[i]-Etalon[i])
// s:=s+abs( X[i]-Etalon[i])/abs(e
end;
якщо Lx>0 then Result:= sqrt (s/Lx) else Result:=1;
     end;
end.
```