

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ
ЗАСТОСУНКУ ВІДСТЕЖЕННЯ ОБ'ЄКТІВ НА ВІДЕО З FPV-ДРОНА
RESEARCH AND SOFTWARE IMPLEMENTATION OF THE
APPLICATION FOR OBJECT TRACKING ON VIDEO FROM AN FPV DRONE

Виконав(ла): студент(ка) 2 курсу, групи КНТ-133М

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

СУДАКОВ В.Р.

(ПРИЗВИЩЕ та ініціали)

Керівник СУББОТІН С.О.

(ПРИЗВИЩЕ та ініціали)

Рецензент КОЦУР М.І.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2024 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

СУДАКОВА Володимира Романовича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація застосунку відстеження об'єктів на відео з FPV-дрона. Research and Software Implementation of the Application for Object Tracking on Video from an FPV Drone

керівник проєкту (роботи) д.т.н., професор, СУББОТІН Сергій Олександрович,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “ 07 ” жовтня 2024 року № 398

2. Строк подання студентом проєкту (роботи) 02 грудня 2024 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	СУББОТІН С.О., завідувач кафедри		
Нормоконтроль	КАЛІНІНА М.В., асистент		

7. Дата видачі завдання “9” вересня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка структури програми.	3 тиждень	Розділ 3
5	Розробка програми.	4-7 тижні	Розділи 3,4
6	Тестування та експериментальне дослідження програмного забезпечення.	8 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	9-10 тижні	Додатки
8	Нормоконтроль та рецензування.	11 тиждень	
9	Захист роботи.	12 тиждень	

Студент(ка)

(підпис) Володимир СУДАКОВ
(Імя ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Сергій СУББОТІН
(Імя ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
92 с., 4 табл., 23 рис., 3 дод., 20 джерел.

C++, QT, ВІДСТЕЖЕННЯ ОБ'ЄКТІВ, МОВА ПРОГРАМУВАННЯ,
НЕЙРОННА МЕРЕЖА, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, РОЗПІЗНАВАННЯ.

Об'єкт дослідження – процес розробки програмного забезпечення для відстеження об'єктів.

Предмет дослідження – програмні засоби для відстеження об'єктів.

Мета роботи – дослідження та розробка програмного забезпечення для відстеження об'єктів на відео з FPV-дрона.

Матеріали, методи та технічні засоби: мова програмування C++, середовище розробки Qt.

Результати. Розроблено програмне забезпечення для відстеження об'єктів на відео з FPV-дрона за допомогою мови програмування C++ та середовища розробки Qt.

Практична цінність роботи полягає у розробці програмного забезпечення, що забезпечує відстеження об'єктів на відео.

Висновки. Виконано проєктування програмного забезпечення для відстеження об'єктів на відео. Розроблено програмне забезпечення для відстеження об'єктів на відео. Здійснено тестування розробленого програмного забезпечення для відстеження об'єктів на відео.

Галузь використання – відстеження об'єктів.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 92 pages, 4 tables, 23 figures, 3 appendixes, 20 sources.

C++, QT, OBJECT TRACKING, PROGRAMMING LANGUAGE, NEURAL NETWORK, SOFTWARE, RECOGNITION.

The object of research is the process of developing software for object tracking.

The subject of the research is software for object tracking.

The purpose of this work is to study and develop software for tracking objects on video from an FPV drone.

Materials, methods and technical tools: C++ programming language, Qt development environment.

Results. The software tools for tracking objects on video from an FPV drone using the C++ programming language and the Qt development environment have been developed.

The practical value of the work lies in the development of software that provides tracking of objects on video.

Conclusions. The software for tracking objects on video has been designed. The software for tracking objects on video has been developed. The developed software for tracking objects on video has been tested.

Scope of use – object tracking.

ЗМІСТ

	С.
Перелік скорочень та умовних познач	8
Вступ.....	9
1 Аналіз предметної області.....	11
1.1 Програмне забезпечення для відстеження об'єктів	11
1.2 Аналіз програмного забезпечення для відстеження об'єктів	18
1.3 Висновки за розділом 1	28
2 Матеріали і методи	30
2.1 Вибір мови програмування	30
2.2 Вибір середовища розробки для створення програмного забезпечення для відстеження об'єктів	32
2.3 Вибір засобів синтезу розпізнавальних моделей для відстеження об'єктів.....	35
2.4 Висновки за розділом 2	42
3 Опис програми	45
3.1 Структура програмного забезпечення для відстеження об'єктів.....	45
3.2 Функціонування програмного забезпечення для відстеження об'єктів ..	47
3.3 Розробка бази даних програмного забезпечення для відстеження об'єктів	51
3.4 Проєктування інтерфейсу програмного забезпечення для відстеження об'єктів	53
3.5 Висновки за розділом 3	55
4 Експлуатація, тестування та експериментальне дослідження програми.....	57
4.1 Призначення й умови застосування програми	57
4.2 Характеристики програми для відстеження об'єктів	58
4.3 Інструкція по експлуатації програми.....	59
4.3.1 Звернення до програми.....	59
4.3.2 Вхідні й вихідні дані	59
4.3.3 Повідомлення.....	60

4.4 Виконання програмного забезпечення для відстеження об'єктів.....	60
4.5 Тестування програмного забезпечення для відстеження об'єктів	62
4.6 Висновки за розділом 4	63
Висновки	64
Перелік джерел посилань	68
Додаток А Технічне завдання	70
Додаток Б Фрагмент тексту програми	74
Додаток В Слайди презентації	85

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

CNN – Convolutional Neural Networks;

FPV – First Person View;

GAN – Generative Adversarial Networks;

GRU – Gated Recurrent Units;

LSTM – Long Short-Term Memory;

RNN – Recurrent Neural Networks;

ВО – відстеження об'єктів;

ПЗ – програмне забезпечення.

ВСТУП

Програмне забезпечення для відстеження об'єктів на відео з FPV-дрона є надзвичайно актуальним через його широкий спектр застосувань та важливість у багатьох сферах. Дрони та відповідне програмне забезпечення все частіше використовуються не лише для розваг, а й у важливих завданнях, таких як військова справа, інспекція інфраструктури, моніторинг територій, рятувальні операції та кінозйомка. Інтеграція автоматичного відстеження об'єктів значно розширює їх функціональність, дозволяючи ефективніше виконувати складні завдання [1], [2].

У рятувальних операціях програмні засоби для відстеження об'єктів допомагають швидше знаходити людей у важкодоступних районах або в екстремальних умовах. В інспекції інфраструктури засоби з відстеженням об'єктів забезпечують можливість автоматичного аналізу мостів, трубопроводів чи електромереж. У сільському господарстві такі програмні системи стають незамінними для моніторингу посівів, тварин або конкретних ділянок. У сфері безпеки дрони з відповідним програмним забезпеченням дозволяють ефективно патрулювати території та виявляти підозрілі активності, а в кінематографії забезпечують динамічну зйомку, автоматично тримаючи об'єкт у центрі кадру [3], [4].

Сучасний розвиток технологій штучного інтелекту та комп'ютерного зору робить можливим використання потужних алгоритмів навіть на компактних пристроях при наявності відповідного програмного забезпечення [5], [6].

Проте деякі програми для відстеження об'єктів на відео можуть мати низку недоліків, які впливають на їхню ефективність і практичність. Одним із головних викликів є висока потреба в обчислювальних ресурсах, оскільки алгоритми відстеження, особливо ті, що базуються на глибокому навчанні, вимагають значної потужності процесора або графічного процесора. Це створює обмеження для їх використання на компактних дронах, де апаратні

ресурси обмежені. Крім того, програмні засоби часто вимагають спеціальної настройки для роботи з різними сценаріями, що може бути складним для недосвідчених користувачів. Ще одним викликом є енергоспоживання, а також те, що використання складних алгоритмів скорочує час роботи дрона через швидкий розряд батареї.

Тому актуальною є розробка програмного забезпечення для відстеження об'єктів на відео. У дипломній кваліфікаційній роботі магістра розв'язується актуальне завдання дослідження та розробки програмного забезпечення для відстеження об'єктів на відео з FPV-дрона.

Для досягнення поставленої мети у кваліфікаційній роботі магістра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для відстеження об'єктів;
- розробити метод спрощення синтезованих згорткових нейромережевих моделей для відстеження об'єктів на відео;
- здійснити проєктування програмного забезпечення для відстеження об'єктів;
- створити програмне забезпечення для відстеження об'єктів;
- виконати тестування розробленого програмного забезпечення для відстеження об'єктів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Програмне забезпечення для відстеження об'єктів

Відстеження об'єктів на відеоданих є процесом ідентифікації, локалізації та відстеження об'єктів у кожному кадрі відеопотоку. Ця задача полягає в аналізі послідовності зображень з метою визначення траєкторії руху об'єктів у просторі та часі. Зазвичай використовуються алгоритми комп'ютерного зору та машинного навчання, включаючи згорткові нейронні мережі, для обробки відеопотоку [1], [2].

Процес відстеження об'єктів починається з виявлення об'єктів на одному або кількох початкових кадрах. Далі, система використовує інформацію про їхні особливості, такі як форма, текстура або колір, для ідентифікації об'єкта на наступних кадрах. Можуть застосовуватися методи прогнозування траєкторій руху, наприклад, фільтри Калмана або Байєсові моделі, для корекції позицій об'єктів у випадках тимчасових перешкод [1], [2].

Для ефективного відстеження застосовуються різні нейронні архітектури. Зокрема, рекурентні нейронні мережі забезпечують облік послідовності кадрів, а згорткові мережі відповідають за виявлення об'єктів у кожному кадрі. Інтеграція цих технологій дозволяє досягти високої точності навіть у динамічних умовах [1], [2].

Відстеження об'єктів широко використовується в таких сферах, як автономні транспортні засоби, моніторинг безпеки, спортивний аналіз, робототехніка та медицина. Ефективність цього процесу залежить від точності алгоритмів, обчислювальних ресурсів та якості відеоданих [1], [2].

Відстеження об'єктів є важливим завданням у сфері комп'ютерного зору, що спрямоване на визначення місця розташування цільового об'єкта на кожному кадрі відеопотоку. Це завдання знаходить застосування у багатьох галузях, таких як робототехніка, системи безпеки, автономні транспортні засоби, інтерфейси між людиною та машиною, технології доповненої реальності та інші [1], [2].

Завдання відстеження єдиного об'єкта полягає у виявленні розташування визначеного об'єкта на всіх послідовних відеокадрах після його первинної ідентифікації. На початковому кадрі користувач задає обмежувальну рамку, яка вказує на цільовий об'єкт. Далі модель машинного навчання проводить пошук цього об'єкта на наступних кадрах, використовуючи інформацію з першого кадру як зразок. Початкова інформація про об'єкт позначається як шаблон, тоді як новий кадр розглядається як область, де здійснюється пошук [1], [2].

Нейромережеві моделі такого типу розроблені для універсального відстеження будь-якого об'єкта, виходячи з його початкового визначення. Їх називають класово-агностичними, оскільки вони не залежать від конкретної категорії об'єкта. Ці моделі можуть ефективно інтегруватися в інструменти для відеоанотацій, значно спрощуючи процес ручного маркування шляхом автоматизації відстеження вибраних об'єктів [1], [2].

Відстеження кількох об'єктів полягає у виявленні та визначенні траєкторій об'єктів, що належать до заздалегідь визначених категорій, на відео. Цей процес орієнтований на роботу з об'єктами певних класів, які були включені до навчальної вибірки моделі. Модель автоматично аналізує кожен кадр, розпізнає задані класи об'єктів і не дозволяє користувачу вносити коригування чи надавати зворотний зв'язок під час роботи [1], [2].

Такий підхід рідко використовується для створення анотацій відео, але він часто застосовується в реальних сценаріях, де моделі використовуються для аналізу відео, а потім результати обробляються для отримання конкретних даних. Наприклад, це може бути підрахунок об'єктів або виявлення нестандартної поведінки [1], [2].

Відстеження кількох об'єктів включає два основні етапи. На першому етапі виконується автоматичне розпізнавання об'єктів у кожному кадрі. Далі здійснюється процес об'єднання розпізнаних об'єктів із сусідніх кадрів у суцільні траєкторії, що пов'язано з проблемою асоціації або повторної ідентифікації [1], [2].

Деякі нейронні моделі здатні одночасно виконувати як виявлення об'єктів, так і об'єднання об'єктів в траєкторії в межах однієї архітектури. Цей підхід забезпечує високу швидкість роботи, але вимагає великих витрат часу і ресурсів на створення позначених відеоданих для навчання. Крім того, додавання нових класів для відстеження вимагає масштабної роботи з усією навчальною вибіркою, що створює складнощі в адаптації таких нейромоделей [1], [2].

Більш практичним підходом є розподіл процесів виявлення та повторної ідентифікації між різними неймережевими моделями. Цей метод, відомий як парадигма відстеження через виявлення, передбачає використання однієї моделі для побудови обмежувальних рамок у кожному кадрі, а окремий алгоритм потім пов'язує ці рамки у суцільні траєкторії об'єктів. Така архітектура дозволяє легко вдосконалювати окремі компоненти системи, інтегруючи новітні технології [1], [2].

Наприклад, модель виявлення може бути навчена на специфічному наборі даних зображень, після чого її результати застосовуються для аналізу відео. Використання двох окремих моделей знижує продуктивність у порівнянні з наскрізними архітектурами, оскільки обчислювальні ресурси витрачаються на запуск кожної моделі окремо [1], [2].

Переваги розділеного підходу полягають у його гнучкості та простоті навчання. Кожен елемент можна змінювати незалежно, що дозволяє швидко впроваджувати оновлення. Недоліком є потреба в більшому обсязі ресурсів для досягнення необхідної продуктивності [1], [2].

Сегментація об'єктів у відео нагадує процес відстеження одного об'єкта, але з використанням масок. Спочатку необхідно позначити маску для об'єкта на початковому кадрі, після чого модель займається сегментацією та стеженням за об'єктом у подальших кадрах [3], [4].

Задачі виявлення та відстеження об'єктів мають велике значення для вирішення сучасних практичних завдань у різних галузях, таких як сільське господарство, транспорт та робототехніка. У цьому контексті важливо

розуміти, що включають ці поняття, і вміти застосовувати їх у програмному коді та для візуалізації [3], [4].

Виявлення об'єктів обмежується аналізом одного кадру. Алгоритм такого типу працює із зображенням і визначає лише конкретні об'єкти, які відповідають заданим параметрам. У випадку відстеження об'єктів розглядається відеопослідовність. Тут алгоритм має завдання стежити за об'єктом протягом усього відео, забезпечуючи його ідентифікацію та відокремлення від інших [3], [4].

Особливу роль у таких завданнях відіграють механізми призначення унікальних ідентифікаторів об'єктам, використання фільтрів для прогнозування положення, наприклад, фільтра Калмана, або методів для аналізу руху, як у випадку з оптичним потоком. Такі підходи дозволяють ефективно обробляти динамічні сцени та забезпечують точність у визначенні траєкторії руху об'єктів [3], [4].

Методи та алгоритми, що використовуються для задач виявлення та відстеження об'єктів, охоплюють різноманітні підходи, спрямовані на вирішення таких завдань із високою точністю та продуктивністю [3], [4].

Оптичний потік, який визначається як векторне поле, що відображає видимий рух об'єктів між двома послідовними кадрами, забезпечує інструментарій для аналізу переміщень. Він допомагає оцінити траєкторію руху, наприклад, м'яча, через кілька кадрів, а також знаходить застосування у відеостабілізації, стисненні відео або структурному аналізі. Цей підхід дає змогу візуалізувати напрямок і швидкість руху об'єктів [3], [4].

Модель YOLOv8, яка є однією з найновіших реалізацій сімейства YOLO (You Only Look Once), розроблена для таких завдань, як виявлення, сегментація та класифікація об'єктів. Вона поєднує високу точність із зручністю використання, завдяки наявності кількох модифікацій, що підходять для різних сценаріїв і вимог до продуктивності. Ця модель варіюється за рівнем точності (mAP) та швидкістю, залежно від обраного варіанта. Наприклад, більш продуктивна версія може забезпечити точні результати, але

вимагатиме більше ресурсів, тоді як швидка, менш ресурсомістка модифікація може жертвувати деякими показниками точності [3], [4].

Алгоритм SORT представляє рішення для двовимірного відстеження кількох об'єктів у відеопослідовностях. Його відмінністю є мінімалізм і простота впровадження, що робить його зручним для багатьох сценаріїв. Цей алгоритм часто використовується як основа для складніших рішень, таких як DeepSort [3], [4].

Згорткові нейронні мережі відіграють важливу роль у задачах, пов'язаних із виявленням і відстеженням. Використання згорткових і об'єднувальних шарів дозволяє виділяти ключові ознаки з вхідних даних, зменшуючи їх просторовий розмір, що полегшує їх подальшу обробку. Такі нейронні мережі забезпечують розпізнавання важливих характеристик, таких як контури об'єктів, і дає змогу створювати ефективні моделі для аналізу зображень та відео [3], [4].

Оптичний потік визначається як переміщення пікселів між двома сусідніми кадрами відео, що виникає через зміну положення об'єкта чи рух камери. Основною задачею цього підходу є обчислення векторів зміщення, які відображають рух пікселів, що дозволяє зрозуміти траєкторію їх переміщення. Це дає змогу аналізувати не лише рух окремих пікселів, але й отримувати загальну картину переміщень у сцені [3], [4].

Обчислення оптичного потоку базується на формуванні щільного векторного поля, де кожен піксель отримує свій власний вектор зміщення. Такий підхід є основою для багатьох завдань комп'ютерного зору, як-от виявлення руху, навігація роботів, розпізнавання об'єктів та їхнє відстеження. Він дає змогу визначати напрямок і швидкість руху пікселів, що особливо корисно для аналізу динаміки об'єктів у відеопослідовності [3], [4].

Відстеження об'єктів у відео належить до основних проблем комп'ютерного зору, де потрібно не лише знаходити об'єкти в сцені, але й впізнавати їх у кожному кадрі, забезпечуючи їхню ідентифікацію навіть за наявності руху. Для вирішення цих завдань створено багато алгоритмів, які

використовуються як основа для подальших розробок. Наприклад, метод метод пропонує відстежувати особливі точки шляхом мінімізації різниці між картами інтенсивності пікселів у послідовних кадрах, застосовуючи афінні перетворення. Інші підходи базуються на аналізі градієнтів, що також дозволяє ефективно працювати з оптичним потоком і відстеженням [3], [4].

Відстеження об'єктів у відео знайшло широке застосування в різних сферах, таких як ігри, наукові дослідження на мікроскопічному рівні, системи спостереження, розслідування злочинів, контроль запасів, моніторинг виробничих процесів і допоміжні технології для водіїв. Ця технологія використовується у багатьох галузях, однак її впровадження в природних умовах або у специфічних відеоматеріалах часто супроводжується значними викликами [3], [4].

Програмні системи відстеження не завжди здатні працювати ефективно без адаптації до конкретних умов. Вплив освітлення, унікальні особливості об'єктів, які відслідковуються, а також шумові перешкоди та інші відхилення у відео вимагають тонкого налаштування алгоритмів. У більшості випадків для успішного вирішення завдання відстеження потрібна розробка індивідуального програмного рішення, що відповідає специфіці кожної ситуації [3], [4].

У наш час різні організації займаються розробкою програмного забезпечення для обробки зображень, комп'ютерного зору та машинного навчання, створюючи унікальні рішення для задач відеовідстеження. Технології комп'ютерного зору успішно впроваджені в багатьох галузях. Програмне забезпечення аналізує відео в реальному часі, передбачаючи рухи та поведінку тварин за допомогою вдосконалених алгоритмів [3], [4].

Розроблені алгоритми дозволяють не лише визначати та стежити за об'єктами в кадрі, але й впізнавати їх у складних умовах, наприклад, коли вони частково приховані, відображаються з кількох камер або знаходяться серед подібних об'єктів. Системи також застосовуються для підрахунку об'єктів, вимірювання відстаней між ними, збору статистичних даних із

відеоматеріалів, а також для виявлення дефектів у матеріалах, як-от мікросхеми. Такі рішення дозволяють ефективно аналізувати відео, що значно скорочує витрати часу і ресурсів, а в деяких випадках навіть сприяє збереженню людських життів [3], [4].

Процес створення програмного забезпечення для відстеження об'єктів на відеоданих включає кілька етапів, які забезпечують його ефективну функціональність та відповідність технічним вимогам [4], [5].

Перший етап передбачає аналіз вимог, де визначаються цілі розробки, основні функції системи, апаратні та програмні обмеження, а також ключові характеристики, як-от точність відстеження, швидкість обробки та сумісність із різними форматами відеоданих [4], [5].

На другому етапі здійснюється розробка архітектури програмного забезпечення. Вибирається модульна структура, яка включає компоненти для обробки відео, синтезу та оптимізації нейромережових моделей, комунікації між пристроями, а також зберігання та обробки даних. Обирається відповідна технологічна платформа, наприклад, C++ із використанням фреймворку Qt для інтеграції компонентів [4], [5].

Третій етап включає збирання даних для навчання та тестування. Формується навчальна вибірка, яка містить відеодані з розміченими об'єктами, а також набори для перевірки та оцінювання моделі [4], [5].

Четвертий етап — це розробка алгоритмів. Виконується синтез нейромережової або іншої моделі для детекції та відстеження об'єктів, оптимізація моделі для роботи на обмежених ресурсах, а також інтеграція методів передобробки відео для зменшення шуму та підвищення точності аналізу [4], [5].

На п'ятому етапі програмне забезпечення розробляється, тестується та налагоджується. Перевіряється робота кожного модуля окремо, а також функціонування всієї системи в реальних умовах. Оптимізація коду дозволяє досягти балансу між продуктивністю та ефективністю використання апаратних ресурсів [5], [6].

На завершальному етапі програмне забезпечення розгортається на цільових пристроях, таких як дрони чи наземні станції. Здійснюється навчання користувачів та надається технічна документація, включаючи інструкції для налаштування й експлуатації [5], [6].

Процес завершується підтримкою та оновленням програмного забезпечення, що включає впровадження нових функцій, вдосконалення алгоритмів та вирішення можливих проблем під час експлуатації [6], [7].

Визначено, що відстеження об'єктів на відеоданих є процесом ідентифікації, локалізації та відстеження об'єктів у кожному кадрі відеопотоку. Ця задача полягає в аналізі послідовності зображень з метою визначення об'єктів у просторі та часі. Зазвичай використовуються алгоритми комп'ютерного зору та машинного навчання, включаючи згорткові нейронні мережі, для обробки відеопотоку. Відзначено, що процес створення програмного забезпечення для відстеження об'єктів на відеоданих включає кілька етапів, які забезпечують його ефективну функціональність та відповідність технічним вимогам.

1.2 Аналіз програмного забезпечення для відстеження об'єктів

Проаналізуємо програмне забезпечення для підтримки процесу відстеження об'єктів [8]-[13].

Програмне забезпечення SynthEyes (рис. 1.1) є спеціалізованою програмою, яка створена для відстеження об'єктів у відео, забезпечуючи точність і контроль у роботі з тривимірною камерою та сценою. Цей програмний інструмент використовується для аналізу руху камери у відеоматеріалах і створення бездоганного поєднання комп'ютерних елементів із реальними кадрами. Основною метою SynthEyes є надання широких можливостей для 3D-відстеження, включаючи точкове, планарне та відстеження тривимірних об'єктів [8], [9], [11].

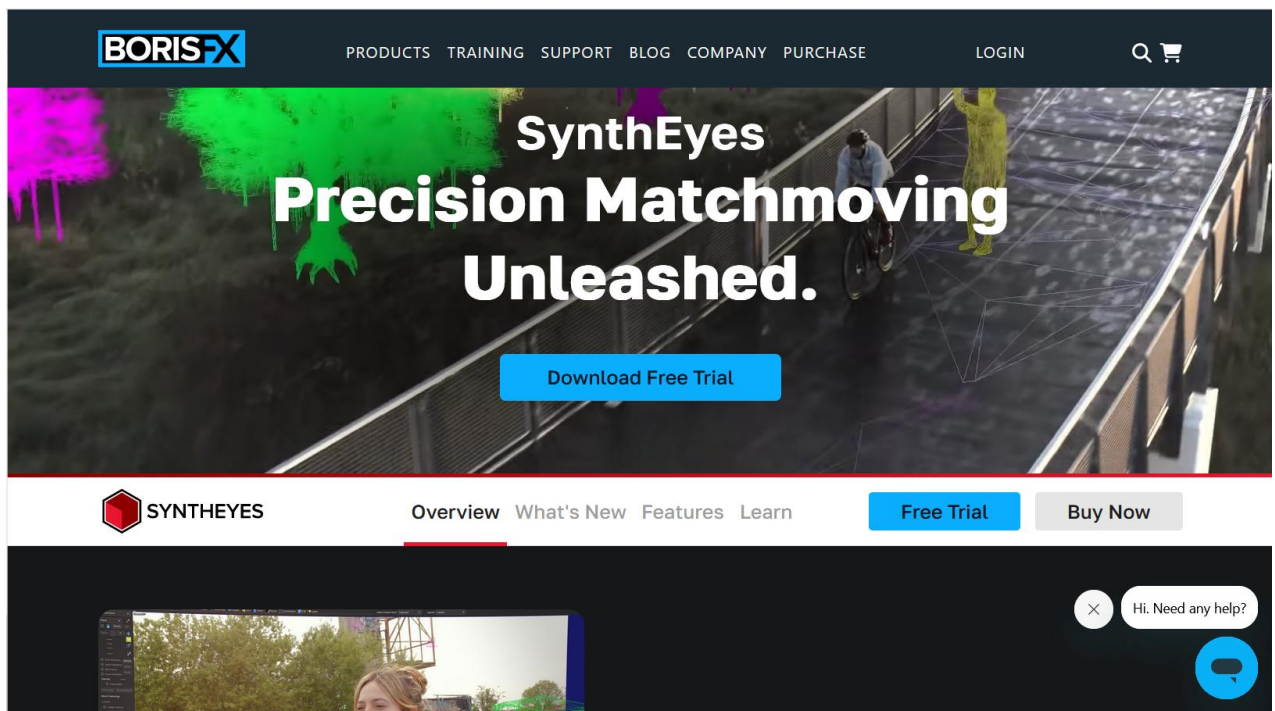


Рисунок 1.1 – Програмне забезпечення SynthEyes [11]

SynthEyes підтримує функції автоматизованого та ручного відстеження, дозволяючи адаптувати процес відстеження об'єктів під потреби користувача. Завдяки цьому застосунку, відстеження об'єктів і камер стає ефективним навіть у випадках із заплутаними рухами або складними спотвореннями лінз. Цей інструмент є важливим для професійного 3D-відстеження об'єктів, забезпечуючи стабілізацію, аналіз геометрії, точне калібрування лінз і масштабування сцени. SynthEyes легко інтегрується з іншими програмами для 3D-моделювання та компонування, такими як Blender, Maya, Nuke, After Effects та багатьма іншими. Завдяки підтримці різноманітних форматів, як-от USD, FBX та OBJ, програмне забезпечення стає універсальним рішенням для роботи з різними типами відеоматеріалів [8], [9], [11].

Застосування програмного забезпечення SynthEyes охоплює всі аспекти відстеження у відео: від базового вирівнювання сцени до складних завдань реконструкції об'єктів і геометрії. Програмне забезпечення SynthEyes надає інструменти для врахування таких нюансів, як анаморфні спотворення або ефекти риб'ячого ока, що робить його незамінним для досягнення високої реалістичності візуальних проєктів [8], [9], [11].

Основними характеристиками програмного забезпечення SynthEyes є такі [8], [9], [11]:

- відстеження камер та об'єктів: програма SynthEyes забезпечує високоточне тривимірне відстеження руху камер і об'єктів у відеоматеріалах, підтримуючи як автоматичне, так і ручне налаштування процесу [8], [9], [11];

- відстеження різних типів об'єктів: програмне забезпечення SynthEyes дозволяє працювати з точковим, планарним і тривимірним відстеженням, забезпечуючи гнучкість у роботі з різними видами кадрів [8], [9], [11];

- калібрування лінз: програмне забезпечення SynthEyes включає розширені інструменти для калібрування лінз, враховуючи ефекти спотворення, такі як анаморфне, радіальне або риб'яче око, що сприяє реалістичності кінцевого результату [8], [9], [11];

- стабілізація відео: інструменти стабілізації програмного забезпечення SynthEyes дозволяють усувати небажані коливання камери, роблячи відео плавним і придатним для подальшої обробки [8], [9], [11];

- реконструкція сцен: SynthEyes може автоматично створювати тривимірні моделі сцен із відеоматеріалів, спрощуючи процес інтеграції віртуальних елементів у реальне середовище [8], [9], [11];

- експорт у 3D-програми: програмне забезпечення SynthEyes підтримує інтеграцію з провідними інструментами для 3D-моделювання та компонування, такими як Maya, Blender, Houdini, Nuke, After Effects тощо, використовуючи формати USD, FBX, OBJ та інші [8], [9], [11];

- робота з різними типами матеріалів: програмне забезпечення SynthEyes може працювати з відео будь-якої складності, включаючи сцени з низьким освітленням, складними текстурами або великою кількістю рухомих об'єктів [8], [9], [11];

- інструменти вирівнювання сцени: набір функцій програмного забезпечення SynthEyes для налаштування масштабів і координат сцени дозволяє досягати точності та коректності при створенні тривимірних моделей об'єктів [8], [9], [11];

– швидка обробка: завдяки оптимізованим алгоритмам програмне забезпечення SynthEyes забезпечує високу швидкість роботи навіть із великими та складними відеопослідовностями [8], [9], [11];

– гнучкість та налаштування: програмне забезпечення SynthEyes дозволяє адаптуватися до потреб кожного користувача завдяки широкому вибору інструментів і можливостей, які можуть бути налаштовані для досягнення необхідного результату [8], [9], [11].

Серед недоліків програмного забезпечення SynthEyes можна відзначити складність освоєння, оскільки для новачків у сфері відстеження об'єктів та роботи з тривимірними сценами програмне забезпечення SynthEyes може здаватися складним через багатofункціональність та велику кількість налаштувань. Крім того дизайн інтерфейсу програмного забезпечення SynthEyes може здаватися застарілим або неінтуїтивним для тих, хто звик працювати з сучасними та більш візуально привабливими програмами. Іншим недоліком є те, що обробка складних сцен, особливо з високою роздільною здатністю відео, потребує потужного апаратного забезпечення, що може бути обмеженням для користувачів із менш продуктивними системами. Для індивідуальних користувачів або невеликих студій вартість ліцензії може бути значною, що робить програму SynthEyes менш доступною для широкого кола користувачів [8], [9], [11].

Програмне забезпечення Mocha Pro (рис. 1.2) є потужним програмним забезпеченням для відстеження руху на відео, яке надає користувачам велику кількість інструментів для точного та ефективного виконання завдань. Цей програмний інструмент підходить для тих, хто працює з відео та потребує високої точності при відстеженні об'єктів, зокрема для відстеження руху, площинного трекінгу та роботи з деформованими об'єктами. Програма Mocha Pro дозволяє відстежувати різні типи об'єктів на відео, включаючи складні поверхні, завдяки технології Power Mesh, яка забезпечує високу точність навіть у випадках, коли об'єкти змінюють свою форму [12].



Рисунок 1.2 – Програмне забезпечення Mocha Pro [12]

Програмне забезпечення Mocha Pro надає різноманітні інструменти, серед яких є можливість відстеження масштабів сітки, злиття і точного визначення діапазону рухомих елементів у кадрі. Завдяки потужним інструментам для калібрування об'єктива та експорту 3D-даних, Mocha Pro дозволяє легко інтегрувати відстежені елементи в проєкти для подальшої обробки та візуалізації. Це програмне забезпечення також дозволяє виконувати ротоскопування з мінімальною кількістю ключових кадрів, що значно пришвидшує процес і зменшує час, необхідний для завершення проєкту [12].

Програмне забезпечення Mocha Pro має виняткову сумісність із великою кількістю нелінійних редакторів, що дає можливість ефективно використовувати його в різноманітних робочих процесах. Це забезпечує універсальність і доступність для широкого кола користувачів. Маючи ці можливості, Mocha Pro є одним з найкращих виборів для тих, хто шукає найвищу якість і точність при відстеженні об'єктів на відео. Програмне забезпечення Mocha Pro

Програмне забезпечення Mocha Pro надає найкращі результати у своєму класі, забезпечуючи високу ефективність при виконанні складних завдань. Системи управління рухом в програмному забезпеченні Mocha Pro простіші в освоєнні порівняно з іншими подібними інструментами, що робить його зручним і доступним для використання. Однак, з огляду на вищу ціну, воно може бути не найекономічнішим варіантом, хоча відповідна якість виправдовує витрати [12].

Програмне забезпечення Mocha Pro має такі особливості [12]:

- площинне відстеження: програмне забезпечення Mocha Pro надає високу точність при відстеженні руху об'єктів у відео, особливо при роботі з плоскими та складними поверхнями [12];

- Power Mesh: програмне забезпечення Mocha Pro дозволяє ефективно відстежувати деформовані поверхні та органічні об'єкти, що робить програму потужним програмним інструментом для аналізу руху об'єктів, що змінюють свою форму [12];

- автоматизація ротоскопування: програмне забезпечення Mocha Pro надає можливість мінімізації кількості ключових кадрів при ротоскопуванні, що дозволяє прискорити процес відстеження об'єктів [12];

- експорт даних: програмне забезпечення Mocha Pro підтримує експорт даних відстеження, ротаційних форм, 3D-даних та калібрування об'єктів, що дає можливість інтегрувати відстеження з іншими програмами для створення візуальних ефектів [12];

- відстеження злиття та масштабу: програмне забезпечення Mocha Pro дозволяє здійснювати точне відстеження руху об'єктів у складних ситуаціях, зокрема при злитті або зміні масштабу об'єктів [12];

- сумісність з нелінійними редакторами: програмне забезпечення Mocha Pro підтримує широкий діапазон нелінійних редакторів, що забезпечує гнучкість у використанні для різних виробничих процесів [12];

– оптимізовані інструменти: програмне забезпечення Mocha Pro надає прості та інтуїтивно зрозумілі інструменти для відстеження об'єктів, що зменшує складність роботи навіть з складними відео матеріалами [12].

Серед недоліків програмного забезпечення Mocha Pro можна відзначити те, що його вартість є досить високою, що може бути недоступним для деяких користувачів, особливо тих, хто працює з обмеженим бюджетом. Окрім цього, хоча програмне забезпечення пропонує потужні функції, його інтерфейс і інструменти можуть бути складними для новачків, що вимагає додаткового часу на освоєння. Для ефективної роботи з великими відеофайлами та складними сценами потрібне потужне апаратне забезпечення, що також може стати додатковим витратним фактором. Крім того, безкоштовна версія Mocha Pro має обмежену кількість доступних функцій, що робить її малоефективною для професіоналів, які потребують всіх можливостей програми [12].

Програмне забезпечення Blender (рис. 1.3) є програмою з відкритим кодом, що має велику кількість інструментів для створення 3D-вмісту, а також для відстеження об'єктів на відео. Однією з ключових можливостей програмного забезпечення Blender є точне відстеження руху в 2D і 3D середовищах, що дає змогу додавати ефекти та анімацію до відео з високою точністю. Інтерфейс програмного забезпечення Blender дозволяє здійснювати відстеження камери, що дає змогу створювати візуальні ефекти на основі руху в реальному часі, а також коригувати невеликі нерівності та тремтіння в записах. Для здійснення відстеження об'єктів на відео програмне забезпечення Blender пропонує надійні функції, що дозволяють користувачам обробляти відеоматеріали, додавати 2D-елементи, такі як тексти або піктограми, а також коригувати рухи для досягнення плавності в сценах. Користувач програмного забезпечення Blender може з легкістю створювати 3D-сцени за допомогою маркерів відстеження, що дозволяє додавати візуальні об'єкти до реального відео з максимальною точністю [8], [9], [13].

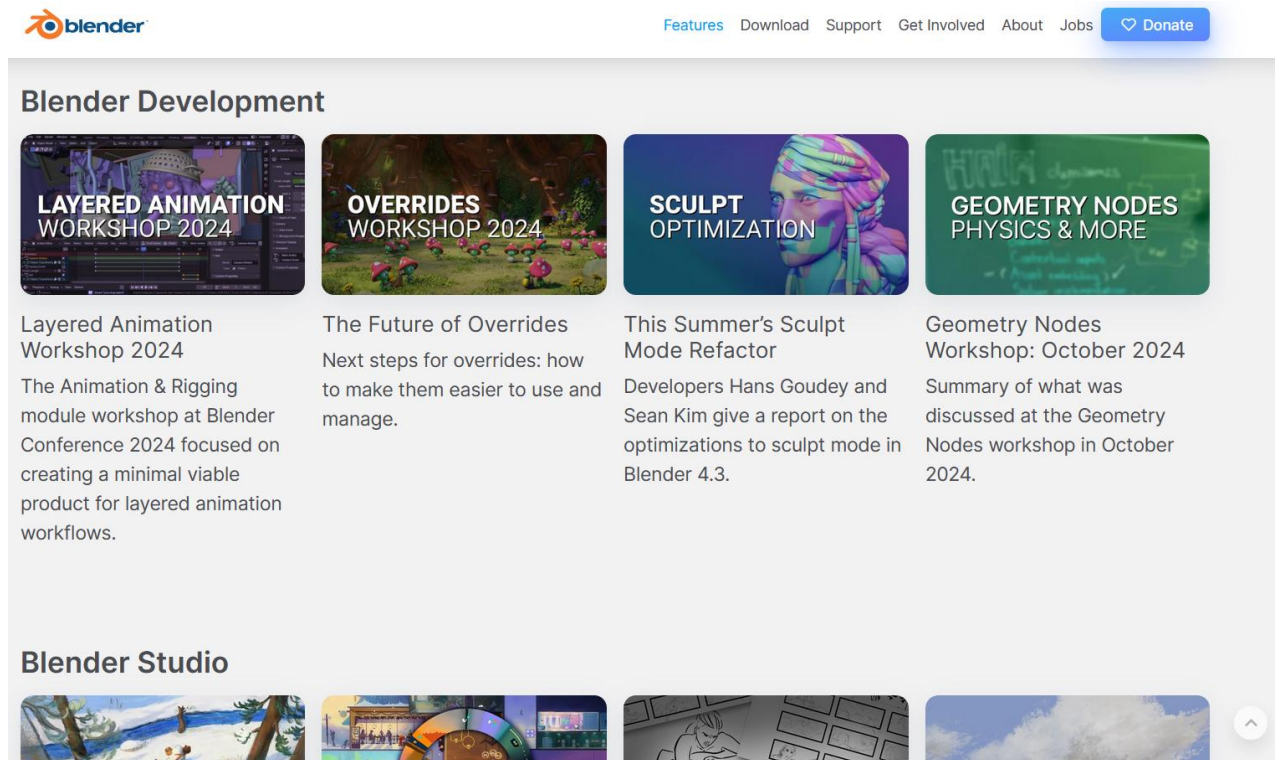


Рисунок 1.3 – Програмне забезпечення Blender [13]

Програмне забезпечення Blender дозволяє перетворювати статичні відеофрагменти в динамічні сцени, які поєднують реальний і віртуальний контент [8], [9], [13].

Відстеження об'єктів у Blender також дозволяє вносити корекції на різних етапах обробки відео, включаючи створення плавних анімацій об'єктів та камери, а також усунення проблем з рухами. Це робить програмне забезпечення відмінним вибором для художників та професіоналів, які потребують потужних можливостей відстеження для створення ефектів і анімацій у своїх проєктах [8], [9], [13].

Програмне забезпечення Blender має такі особливості [8], [9], [13]:

- відстеження об'єктів і руху в 2D та 3D: програмне забезпечення Blender підтримує як 2D, так і 3D відстеження об'єктів, що дозволяє точніше працювати з елементами відео та створювати інтерактивні сцени [8], [9], [13];

- маркерне відстеження: за допомогою маркерів можна точно визначити положення об'єктів у кадрі, що дозволяє здійснювати точну інтеграцію 3D-об'єктів у відео [8], [9], [13];

- створення 3D-сцен на основі відстежених маркерів: після того, як маркери відстежені, програмне забезпечення Blender дозволяє створити 3D-сцену, де можна додавати 3D-об'єкти, що ідеально інтегруються в реальне відео [8], [9], [13];

- додавання 2D елементів до відео: програмне забезпечення Blender дозволяє додавати 2D-елементи, такі як текст чи піктограми, до конкретних точок відстежуваного об'єкта для покращення візуальних ефектів [8], [9], [13];

- стабілізація відео: програмне забезпечення Blender дозволяє усунути тремтіння та нерівності у відео, що покращує якість зображення при обробці матеріалів [8], [9], [13];

- розширене 3D-модельовання та анімація: програмне забезпечення Blender забезпечує потужні інструменти для 3D-модельовання, текстуровання, скульптингу та створення складної анімації [8], [9], [13];

- рендеринг у реальному часі: за допомогою рендеринг-двигунів Cycles і Eevee програмне забезпечення Blender здатне створювати фотореалістичні зображення та анімації в режимі реального часу [8], [9], [13];

- інтуїтивно зрозумілий інтерфейс: хоча програмне забезпечення Blender має велику кількість функцій, його інтерфейс налаштовується під потреби користувача, що дозволяє легко освоїти програму [8], [9], [13];

- активна спільнота та додаткові ресурси: програмне забезпечення Blender має велику спільноту, де можна знайти безліч підручників, плагінів і доповнень для розширення її функціональності [8], [9], [13].

Серед недоліків програмного забезпечення Blender можна відзначити його складний інтерфейс, що може бути важким для освоєння початківцями. Хоча Blender пропонує потужні інструменти для відстеження об'єктів на відео, освоєння всіх можливостей програмного забезпечення потребує значного часу та зусиль. Для обробки складних сцен або відео з великою кількістю відстежуваних елементів може знадобитися потужне обладнання, оскільки високі вимоги до ресурсів можуть вплинути на швидкість роботи програми. Крім того, деякі розширені функції можуть вимагати додаткового

налаштування або навчання, що не завжди є зручним для користувачів, які шукають швидке рішення [8], [9], [13].

Порівняльну характеристику програмного забезпечення для підтримки процесу відстеження об'єктів наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення для підтримки процесу відстеження об'єктів

Критерій порівняння	SynthEyes [11]	Mocha Pro [12]	Blender [13]
Автоматичне відстеження об'єктів	+	+	+—
Обробка складних сцен	+—	+	+—
Складність використання	+—	+—	+
Відкритий код	—	—	+
Висока точність відстеження об'єктів	+	+	—
Підтримка великої кількості відео форматів	+	+—	+

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато програмних засобів для підтримки процесу відстеження об'єктів. Проте деякі програмні засоби для відстеження об'єктів на відео можуть мати низку недоліків, які впливають на їхню ефективність і практичність. Одним із головних викликів є висока потреба в обчислювальних ресурсах, оскільки алгоритми відстеження, особливо ті, що базуються на глибокому навчанні, вимагають значної потужності процесора або графічного процесора. Це створює обмеження для їх використання на компактних дронах, де апаратні ресурси обмежені. Крім того, програмні засоби часто вимагають спеціальної настройки для роботи з різними сценаріями, що може бути

складним для недосвідчених користувачів. Ще одним викликом є енергоспоживання, а також те, що використання складних алгоритмів скорочує час роботи дрона через швидкий розряд батареї. Тому актуальною є розробка програмного забезпечення для відстеження об'єктів.

При розробці програмного забезпечення для відстеження об'єктів необхідно забезпечити такі функціональні вимоги:

- підтримка можливості завдання даних для синтезу моделей для відстеження об'єктів;
- підтримка можливості оптимізації синтезованих моделей для відстеження об'єктів;
- можливість відстеження об'єктів на відео даних;
- можливість відображення відстежених об'єктів на графічних зображеннях.

1.3 Висновки за розділом 1

Визначено, що відстеження об'єктів на відеоданих є процесом ідентифікації, локалізації та відстеження об'єктів у кожному кадрі відеопотоку. Ця задача полягає в аналізі послідовності зображень з метою визначення об'єктів у просторі та часі. Зазвичай використовуються алгоритми комп'ютерного зору та машинного навчання, включаючи згорткові нейронні мережі, для обробки відеопотоку. Відзначено, що процес створення програмного забезпечення для відстеження об'єктів на відеоданих включає кілька етапів, які забезпечують його ефективну функціональність та відповідність технічним вимогам.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для відстеження об'єктів. Проте деякі програмні засоби для відстеження об'єктів на відео можуть мати низку недоліків, які впливають на їхню ефективність і практичність. Одним із головних викликів є висока потреба в обчислювальних ресурсах, оскільки

алгоритми відстеження, особливо ті, що базуються на глибокому навчанні, вимагають значної потужності процесора або графічного процесора. Це створює обмеження для їх використання на компактних дронах, де апаратні ресурси обмежені. Крім того, програмні засоби часто вимагають спеціальної настройки для роботи з різними сценаріями, що може бути складним для недосвідчених користувачів. Ще одним викликом є енергоспоживання, а також те, що використання складних алгоритмів скорочує час роботи дрона через швидкий розряд батареї. Тому актуальною є розробка програмного забезпечення для відстеження об'єктів.

Сформульовано функціональні вимоги до програмного забезпечення для відстеження об'єктів.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування

При розробці програмного забезпечення для відстеження об'єктів було використано мову програмування C++ [14], [15].

Використання мови програмування C++ для розробки програмного забезпечення для відстеження об'єктів на відео з FPV-дрона є доцільним з кількох причин [14], [15].

Висока продуктивність: C++ дозволяє ефективно працювати з апаратними ресурсами, забезпечуючи високу швидкість виконання коду. Це є критично важливим для обробки відеопотоків у реальному часі, де необхідно виконувати складні обчислення з мінімальною затримкою [14], [15].

Оптимізація використання ресурсів: FPV-дрони зазвичай мають обмежені обчислювальні ресурси та енергоспоживання. C++ надає розробникам можливість тонкого налаштування та оптимізації коду для максимально ефективного використання процесора та пам'яті [14], [15].

Підтримка бібліотек та фреймворків: C++ має широкий спектр бібліотек, які можуть бути використані для розробки програмного забезпечення для обробки зображень та відео, таких як OpenCV для комп'ютерного зору, TensorRT для прискорення роботи нейронних мереж та інші інструменти для роботи з апаратним прискоренням [14], [15].

Реалізація алгоритмів у реальному часі: завдяки своїй низькорівневій природі C++ забезпечує мінімальні затримки при обробці даних, що є важливим для відстеження об'єктів у відеопотоці [14], [15].

Кросплатформеність: програмне забезпечення, написане на C++, може бути зкомпільовано та запущено на різних платформах, включаючи вбудовані системи дронів, що спрощує інтеграцію програмного забезпечення з різними пристроями [14], [15].

Можливості роботи з апаратними компонентами: C++ забезпечує низькорівневий доступ до апаратного забезпечення, що дозволяє взаємодіяти з

камерами, сенсорами та іншими компонентами дронів для забезпечення ефективного оброблення вхідних даних [14], [15].

Зручність роботи з паралельними обчисленнями: C++ підтримує паралельне програмування через бібліотеки та стандарти, такі як OpenMP, Intel TBB та стандартні багатопоточні функції (std::thread), що дозволяє ефективно використовувати багатоядерні процесори для виконання інтенсивних обчислень [14], [15].

Отже, мова C++ є оптимальним вибором для розробки програмного забезпечення для відстеження об'єктів на відео з FPV-дрона завдяки своїй високій продуктивності, можливостям оптимізації та інтеграції з апаратними компонентами, що дозволяє забезпечити ефективну роботу системи навіть в умовах обмежених ресурсів дрона.

Обґрунтування вибору мови програмування для створення програмного забезпечення для відстеження об'єктів наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для створення програмного забезпечення для відстеження об'єктів

Критерій порівняння мов програмування	Мова програмування		
	C++	Python	Ruby
Висока продуктивність	+	+-	-
Оптимізація використання ресурсів	+	+-	+-
Реалізація алгоритмів у реальному часі	+	+	-
Зручність для створення ПЗ для відстеження об'єктів	+	-	-
Можливості роботи з апаратними компонентами	+	+-	+-

Отже, для реалізації програмного забезпечення для відстеження об'єктів обрано мову програмування C++, яка дозволяє ефективно працювати з апаратними ресурсами, забезпечуючи високу швидкість виконання коду. Це є критично важливим для обробки відеопотоків у реальному часі, де необхідно виконувати складні обчислення з мінімальною затримкою. Крім того, C++ має широкий спектр бібліотек, які можуть бути використані для розробки програмного забезпечення для обробки зображень та відео. Іншою перевагою є те, що завдяки своїй низькорівневій природі C++ забезпечує мінімальні затримки при обробці даних, що є важливим для відстеження об'єктів у відеопотоці. C++ забезпечує низькорівневий доступ до апаратного забезпечення, що дозволяє взаємодіяти з камерами, сенсорами та іншими компонентами дронів для забезпечення ефективного оброблення вхідних даних.

2.2 Вибір середовища розробки для створення програмного забезпечення для відстеження об'єктів

В якості середовища розробки для створення програмного забезпечення для відстеження об'єктів було обрано середовище Qt [16], [17].

Використання середовища розробки Qt для створення програмного забезпечення для відстеження об'єктів на відео з FPV-дрона є доцільним завдяки низці переваг, які забезпечують ефективність розробки, продуктивність та зручність для користувачів [16], [17].

Середовище розробки Qt дозволяє створювати програмне забезпечення, яке може працювати на різних операційних системах (Windows, Linux, macOS, Android, iOS) без значних змін у коді. Це важливо для дронів, які можуть працювати на різних вбудованих системах [16], [17].

Середовище розробки Qt надає потужний набір інструментів і бібліотек для побудови графічного інтерфейсу користувача, що значно скорочує час розробки. Візуальні елементи управління для FPV-дронів, такі як

відображення відеопотоку, інформації про об'єкти та налаштування системи, легко створюються за допомогою Qt Widgets або Qt Quick [16], [17].

Qt забезпечує ефективну роботу навіть у системах з обмеженими ресурсами завдяки оптимізації свого коду. Це дозволяє реалізовувати обробку відеопотоків у реальному часі без значного навантаження на апаратне забезпечення дрона [16], [17].

Підтримка роботи з мультимедіа: Qt Multimedia надає зручні інструменти для роботи з відео, що важливо для обробки відеопотоків із камери дрона. Вона дозволяє інтегрувати в програму відеопотоки, відображати кадри в режимі реального часу та обробляти їх [16], [17].

Гнучкість у розробці модульної архітектури: Qt дозволяє легко розділяти програму на модулі, наприклад, модуль для обробки відео, модуль для роботи з нейронними мережами та модуль для відображення даних, що спрощує підтримку та розвиток проєкту [16], [17].

Qt розроблено на основі C++ і дозволяє ефективно поєднувати низькорівневу обробку даних із високорівневими елементами інтерфейсу користувача. Це спрощує реалізацію складних алгоритмів, таких як нейромереві моделі для відстеження об'єктів [16], [17].

Підтримка роботи з мережею: середовище розробки Qt дозволяє легко реалізувати передачу даних між дроном та наземною станцією, наприклад, для передавання даних відеопотоку, телеметрії або результатів відстеження об'єктів [16], [17].

Інструменти для налагодження та тестування: середовище розробки Qt Creator, інтегроване середовище розробки, надає розширені інструменти для налагодження, аналізу продуктивності та тестування, що робить процес розробки зручнішим [16], [17].

Можливість створення адаптивного інтерфейсу. За допомогою Qt Quick можна розробити адаптивний інтерфейс, який буде однаково зручним для різних типів екранів і пристроїв, включаючи мобільні додатки для управління дроном [16], [17].

Загалом, використання Qt у проєктах для FPV-дронів є доцільним завдяки його універсальності, продуктивності та можливості швидкої розробки складних систем із графічним інтерфейсом і підтримкою мультимедійних функцій. Це сприяє створенню надійного програмного забезпечення для відстеження об'єктів на відео, що працює ефективно навіть на обмежених апаратних ресурсах [16], [17].

Обґрунтування вибору середовища розробки для створення програмного забезпечення для відстеження об'єктів наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору середовища розробки для створення програмного забезпечення для відстеження об'єктів

Критерій порівняння середовищ розробки	Середовища розробки		
	Qt	Visual Studio	Atom
Швидкість розробки	+	+	+—
Продуктивність	+	+—	+
Зручність інтеграції з C++	+	+	—
Підтримка роботи з мультимедіа	+	+	+—
Гнучкість у розробці модульної архітектури	+	+—	+

Таким чином, для створення програмного забезпечення для відстеження об'єктів обрано середовище розробки Qt, що дозволяє створювати програмне забезпечення, яке може працювати на різних операційних системах без значних змін у коді, що є важливим для роботи з дронами, які можуть працювати на різних вбудованих системах. Крім того, середовище Qt забезпечує ефективну роботу навіть у системах з обмеженими ресурсами завдяки оптимізації свого коду.

2.3 Вибір засобів синтезу розпізнавальних моделей для відстеження об'єктів

З метою можливості розробки програмного забезпечення для відстеження об'єктів необхідно мати розпізнавальні моделі [18]-[20].

Для відстеження об'єктів на відео доцільним є використання нейронних мереж завдяки їхній здатності ефективно вирішувати складні завдання аналізу зображень і відеопотоків. Основними перевагами нейронних мереж у цьому контексті є [18]-[20]:

- висока точність та адаптивність. Нейронні мережі, особливо згорткові (Convolutional Neural Networks, CNN), мають потужність для автоматичного виділення складних ознак об'єктів на відео. Вони здатні адаптуватися до різних умов, таких як зміни освітлення, ракурсу або розмиття, забезпечуючи високу точність відстеження [18]-[20];

- можливість роботи в реальному часі. Завдяки сучасним апаратним прискорювачам (наприклад, графічним процесорам або спеціалізованим чипам, таким як NVIDIA Jetson), нейронні мережі можуть обробляти відеопотоки в реальному часі. Це критично важливо для FPV-дронів, де затримка в обробці даних може призвести до втрати об'єкта [18]-[20];

- робота зі складними сценаріями. Нейронні мережі можуть розпізнавати та відстежувати об'єкти навіть у складних умовах, таких як велика кількість перешкод, багатозадачність (одночасне відстеження кількох об'єктів) або змінне тло. Наприклад, методи на основі Recurrent Neural Networks (RNN) або Transformer-моделей можуть враховувати тимчасові залежності у відеопотоку для більш стабільного відстеження [18]-[20];

- автоматичне навчання. Завдяки здатності до самонавчання нейронні мережі можуть бути навчені розпізнавати різні типи об'єктів, використовуючи великі набори даних. Це значно зменшує потребу в ручній обробці та налаштуванні алгоритмів [18]-[20];

– розширюваність й інтеграція. Нейронні мережі легко інтегруються з іншими технологіями, такими як оптичний потік, системи позиціонування та 3D-реконструкція сцени. Це дозволяє створювати комплексні системи для відстеження, що враховують не лише положення об'єкта, але й його швидкість, траєкторію та поведінку [18]-[20];

– універсальність. Одні й ті самі моделі нейронних мереж можуть бути використані для різних завдань, таких як класифікація об'єктів, їх сегментація чи прогнозування руху. Це робить нейронні мережі універсальним інструментом у відеоаналітиці [18]-[20];

– висока стійкість до шуму. Нейронні мережі мають вбудовану стійкість до шуму в даних, що робить їх ефективними навіть за наявності перешкод у відеопотоці, таких як низька якість зображення чи перешкоди від інших джерел сигналу [18]-[20].

Таким чином, використання нейронних мереж для відстеження об'єктів на відео забезпечує високу точність, продуктивність та універсальність у широкому спектрі сценаріїв. Це робить їх ідеальним вибором для сучасних систем на базі FPV-дронів [18]-[20].

Проаналізуємо моделі на основі нейронних мереж, які можна використовувати для відстеження об'єктів на відео [18]-[20].

Згорткові нейронні мережі (Convolutional Neural Networks, CNN) є основою для аналізу зображень і відеопотоків завдяки їхній здатності виділяти просторові особливості об'єктів. Згорткові нейронні мережі використовуються для розпізнавання об'єктів на кожному кадрі відео, виділення ключових точок і класифікації. Приклади моделей: YOLO (You Only Look Once), SSD (Single Shot Detector), Faster R-CNN. Особливості: ефективність у реальному часі (наприклад, YOLO), точність для складних сцен із багатьма об'єктами [18]-[20].

Рекурентні нейронні мережі (Recurrent Neural Networks, RNN) враховують часові залежності у відео, що є важливим для відстеження об'єктів на основі їхнього руху. Рекурентні нейронні мережі використовуються для

відстеження об'єктів, коли їх положення змінюється з часом (наприклад, прогнозування траєкторії). Особливості: враховують контекст попередніх кадрів, використовуються у поєднанні із CNN для обробки послідовностей кадрів [18]-[20].

Двонаправлені повторювані мережі (Bidirectional RNN) враховують як попередні, так і наступні кадри, що дозволяє більш точно передбачати рух об'єкта. Застосовуються для завдань, де важливо враховувати майбутній контекст, наприклад, у ретроспективному аналізі [18]-[20].

Long Short-Term Memory (LSTM) та Gated Recurrent Units (GRU) є модифікаціями RNN, які краще зберігають довготривалі часові залежності, що важливо для тривалого відстеження об'єктів. Такі моделі дозволяють виконувати відстеження об'єктів у складних відео з великою кількістю кадрів. Особливості: висока точність у завданнях із довгими послідовностями, стійкість до втрати важливих часових залежностей [18]-[20].

Мережі з трансформерами (Transformers), такі як Vision Transformers (ViT) або DETR (DEtection TRansformer), демонструють чудові результати в задачах обробки зображень і відео. Застосування: відстеження кількох об'єктів одночасно з урахуванням складних просторових і часових зв'язків. Особливості: ефективне використання обчислювальних ресурсів, покращення точності у складних динамічних сценах [18]-[20].

Генеративно-суперницькі мережі (Generative Adversarial Networks, GAN) можуть використовуватись для підвищення якості кадрів у відео або відновлення втрачених частин зображення, що може допомогти в стабільному відстеженні об'єктів. Застосування: відстеження об'єктів у відео з низькою якістю або відновлення об'єктів, які частково зникли з кадру [18]-[20].

Реальні моделі для обробки відео (3D CNN) враховують як просторові, так і часові аспекти, обробляючи послідовність кадрів як єдиний тривимірний масив. Такі моделі застосовуються для аналізу коротких відеофрагментів або послідовностей. Особливості: складна обробка відео з високою деталізацією.

Для відстеження об'єктів на відео можуть використовуватися CNN у поєднанні з RNN, LSTM чи трансформерами. Це забезпечує баланс між точністю, продуктивністю та адаптивністю до змінних умов. Вибір типу нейронної мережі залежить від конкретного завдання, доступних ресурсів та вимог до швидкодії й точності [18]-[20].

Порівняння видів нейронних мереж для відстеження об'єктів на відео наведено у табл. 2.3.

Таблиця 2.3 – Порівняння видів нейронних мереж для відстеження об'єктів на відео

Критерій порівняння нейронних мереж	Нейронні мережі для обробки зображень				
	CNN	RNN	LSTM/ GRU	3D CNN	Trans- formers (DETR/ ViT)
Точність у розпізнаванні об'єктів	+	—	+—	+	+
Обробка просторових ознак	+	—	—	+	+
Швидкість роботи в реальному часі	+	+—	+—	—	+—
Адаптація до зміни умов (тло, освітлення)	+	+—	+—	+	+
Складність архітектури	+	+	+—	—	—
Вимоги до апаратних ресурсів	+	+	+—	—	—

Визначено, що згорткові нейронні мережі є найефективнішими для відстеження об'єктів на відео завдяки їхній здатності швидко та точно обробляти просторові ознаки. Вони забезпечують високу продуктивність у

реальному часі, що робить їх оптимальним вибором для відстеження об'єктів за допомогою FPV-дронів та інших завдань, де потрібна швидкість і точність. Хоча RNN і LSTM краще працюють із часовими залежностями, їх швидкодія поступається CNN. У сучасних завданнях трансформери демонструють високу точність і адаптивність, але їхня складність і вимоги до апаратного забезпечення роблять їх менш практичними в багатьох реальних сценаріях.

Не зважаючи на те, що згорткові нейронні мережі широко використовуються у багатьох задачах комп'ютерного зору, але їх використання на пристроях з обмеженими ресурсами, таких як FPV-дрони, стикається зі значними труднощами, зокрема [18]-[20]:

- висока складність і вимоги до ресурсів. CNN часто мають велику кількість параметрів і операцій, що впливає на споживання обчислювальних ресурсів. На дронах обчислювальні потужності обмежені через компактний розмір, низьку вагу, обмеження енергоспоживання та тепловиділення. Зменшення складності моделі дозволяє значно скоротити обчислювальні витрати, забезпечуючи можливість її роботи в реальному часі навіть на обмежених апаратних платформах;

- висока енергозалежність. Робота з повнорозмірними CNN вимагає високопродуктивних процесорів або графічних чипів, які споживають багато енергії. Це обмежує час польоту дрона, що критично важливо для автономних операцій. Оптимізація зменшує кількість операцій і параметрів, що знижує енергоспоживання моделі, збільшуючи тривалість польоту дрона;

- впровадження в реальному часі. Для ефективного відстеження об'єктів на FPV-дронах потрібна висока швидкість обробки, адже затримки можуть призводити до втрати об'єкта або помилок у навігації. Спрощення моделі дозволяє знизити час обробки кадру, що забезпечує реальну роботу в режимі реального часу;

- збереження точності при зменшенні розміру. Деякі параметри CNN мають надлишкову інформацію або мало впливають на результат. Завдяки сучасним методам оптимізації можна значно зменшити розмір моделі без

істотної втрати точності. Використання методів прунингу (pruning), квантування (quantization) та знєколонізації (knowledge distillation) дозволяє зберігати точність відстеження при суттєвому зменшенні моделі;

– легкість інтеграції в обмежені платформи. FPV-дрони зазвичай використовують енергоефективні вбудовані платформи, такі як NVIDIA Jetson Nano, Google Coral або навіть ARM-процесори. Повнорозмірні CNN часто перевищують можливості таких платформ. Оптимізовані моделі легко інтегруються у ці пристрої, забезпечуючи стабільну роботу без надмірних вимог до обладнання.

Таким чином, оптимізація моделей на основі CNN для їх використання на FPV-дронах є критично важливою через обмеження в ресурсах, енергії та обчислювальній потужності. Завдяки методам скорочення можна зменшити модель, зберігаючи її функціональність, забезпечуючи реальну роботу у складних динамічних умовах. Оптимізовані CNN дозволяють FPV-дронам ефективно виконувати завдання відстеження об'єктів у реальному часі без значного збільшення вартості чи ваги системи.

Тому з метою зменшення складності нейромережових моделей при збереженні високої точності розпізнавання пропонується метод спрощення синтезованих згорткових нейромережових моделей для відстеження об'єктів на відео. Запропонований метод ґрунтується на поєднанні наступних підходів:

– прунинг (Pruning) – передбачає видалення малозначущих нейронів або зв'язків у моделі. Наприклад, фільтри або вузли, які мають малі ваги або незначну активність, можуть бути вилучені без суттєвого впливу на якість моделі. Процедура: виконати аналіз важливості фільтрів та шарів, видалити найменш значущі фільтри або зв'язки, перенавчити модель, щоб відновити точність після скорочення. Результат: зменшення кількості обчислень і пам'яті, необхідних для виконання задачі;

– квантування (Quantization) – передбачає зменшення розрядності параметрів і обчислень. Наприклад, замість 32-бітної точності використовуються 8-бітні або навіть 4-бітні представлення чисел. Процедура:

перетворити ваги та активації на меншу розрядність під час навчання; використати апаратно-орієнтовані оптимізації, сумісні з низькорозрядними обчисленнями. Результат: модель стає більш енергоефективною та швидшою без значної втрати точності;

- Knowledge Distillation – використання складної (великої) моделі ("вчителя") для навчання спрощеної ("студентської") моделі. "Студентська" модель навчається імітувати прогнози "вчителя", що дозволяє досягти подібної точності за меншої складності. Процедура: навчити складну модель на основному наборі даних, використати її виходи для навчання меншої моделі, параметри меншої моделі оптимізуються для ефективного використання обмежених ресурсів. Результат: менша модель успішно зберігає ключові характеристики великої, але працює швидше;

- архітектурні оптимізації – створення або адаптація архітектури CNN для задач з обмеженими ресурсами. Наприклад, використання легковагових архітектур, таких як MobileNet, SqueezeNet або ShuffleNet. Процедура: замінити стандартні згорткові шари на глибокі згортки (depthwise separable convolutions), скоротити кількість фільтрів у шарах, адаптувати модель до конкретної платформи (FPGA, NVIDIA Jetson тощо). Результат: спрощена архітектура з мінімальними втратами продуктивності;

- аугментація даних і регуляризація – навчання моделі з використанням розширеного набору даних і методів регуляризації (наприклад, Dropout або Batch Normalization) для покращення узагальнюючої здатності спрощеної моделі.

Отже розроблений метод спрощення синтезованих згорткових нейромережевих моделей для відстеження об'єктів на відео передбачає виконання таких етапів:

- здійснити базове навчання моделі для отримання максимальної точності;

- послідовно застосувати прунинг і квантування для зменшення моделі;

- використати "вчитель-студент" підхід для перенесення знань у спрощену модель;
- додатково провести оптимізацію архітектури для сумісності з цільовими пристроями дрона.

Оптимізована за допомогою запропонованого методу модель буде достатньо компактною для використання на FPV-дронах, забезпечуючи реальну роботу у режимі реального часу. Це дозволить досягти високої точності відстеження об'єктів при мінімальному споживанні енергії та обчислювальних ресурсів.

Таким чином, розроблено метод спрощення синтезованих згорткових нейромережових моделей для відстеження об'єктів на відео, який включає етапи прунингу, квантування, передачі знань та архітектурних оптимізацій. Унікальність запропонованого методу полягає в комплексному застосуванні кількох підходів для зменшення складності моделі при збереженні високої продуктивності та точності. Цей метод дозволяє синтезувати та програмно реалізувати ефективні моделі для відстеження об'єктів на ресурсозалежних пристроях, таких як дрони.

2.4 Висновки за розділом 2

Для реалізації програмного забезпечення для відстеження об'єктів обрано мову програмування C++, яка дозволяє ефективно працювати з апаратними ресурсами, забезпечуючи високу швидкість виконання коду. Це є критично важливим для обробки відеопотоків у реальному часі, де необхідно виконувати складні обчислення з мінімальною затримкою. Крім того, C++ має широкий спектр бібліотек, які можуть бути використані для розробки програмного забезпечення для обробки зображень та відео. Іншою перевагою є те, що завдяки своїй низькорівневій природі C++ забезпечує мінімальні затримки при обробці даних, що є важливим для відстеження об'єктів у відеопотоці. C++ забезпечує низькорівневий доступ до апаратного

забезпечення, що дозволяє взаємодіяти з камерами, сенсорами та іншими компонентами дронів для забезпечення ефективного оброблення вхідних даних.

Для створення програмного забезпечення для відстеження об'єктів обрано середовище розробки Qt, що дозволяє створювати програмне забезпечення, яке може працювати на різних операційних системах без значних змін у коді, що є важливим для роботи з дронами, які можуть працювати на різних вбудованих системах. Крім того, середовище Qt забезпечує ефективну роботу навіть у системах з обмеженими ресурсами завдяки оптимізації свого коду.

Визначено, що згорткові нейронні мережі є найефективнішими для відстеження об'єктів на відео завдяки їхній здатності швидко та точно обробляти просторові ознаки. Вони забезпечують високу продуктивність у реальному часі, що робить їх оптимальним вибором для відстеження об'єктів за допомогою FPV-дронів та інших завдань, де потрібна швидкість і точність. Хоча RNN і LSTM краще працюють із часовими залежностями, їх швидкодія поступається CNN. У сучасних завданнях трансформери демонструють високу точність і адаптивність, але їхня складність і вимоги до апаратного забезпечення роблять їх менш практичними в багатьох реальних сценаріях.

Показано, що оптимізація моделей на основі CNN для їх використання на FPV-дронах є критично важливою через обмеження в ресурсах, енергії та обчислювальній потужності. Завдяки методам скорочення можна зменшити модель, зберігаючи її функціональність, забезпечуючи реальну роботу у складних динамічних умовах. Оптимізовані CNN дозволяють FPV-дронам ефективно виконувати завдання відстеження об'єктів у реальному часі без значного збільшення вартості чи ваги системи.

Розроблено метод спрощення синтезованих згорткових нейромережевих моделей для відстеження об'єктів на відео, який включає етапи прунингу, квантування, передачі знань та архітектурних оптимізацій. Унікальність запропонованого методу полягає в комплексному застосуванні

кількох підходів для зменшення складності моделі при збереженні високої продуктивності та точності. Цей метод дозволяє синтезувати та програмно реалізувати ефективні моделі для відстеження об'єктів на ресурсозалежних пристроях, таких як дрони.

3 ОПИС ПРОГРАМИ

3.1 Структура програмного забезпечення для відстеження об'єктів

Структуру програмного забезпечення для відстеження об'єктів наведено на рис. 3.1.



Рисунок 3.1 – Структура програмного забезпечення для відстеження об'єктів

Структура програмного забезпечення для відстеження об'єктів складається з шести основних модулів, кожен з яких виконує специфічну роль в процесі обробки відео та відстеження об'єктів.

Модуль обробки відео забезпечує захоплення, попередню обробку та підготовку відео для подальшої обробки. Він включає в себе фільтрацію, зміну розміру, корекцію яскравості та контрастності, а також інші попередні етапи. Основними компонентами модулю обробки відео є засоби для захоплення кадрів з відео або FPV-дрона, засоби попередньої обробки зображень для видалення шумів, а також засоби для виділення ключових фреймів або областей інтересу.

Модуль відстеження об'єктів відповідає за визначення положення об'єктів на кожному кадрі та їх відстеження в реальному часі. Враховуються такі параметри, як зміни положення, орієнтації та розміру об'єктів, а також їх траєкторії, що дозволяє точно відстежувати об'єкти на відео. Основними компонентами модулю відстеження об'єктів є засоби для визначення області інтересу для кожного об'єкта, засоби відстеження на основі алгоритмів, а також засоби врахування швидкості та траєкторії руху об'єктів.

Модуль детекції об'єктів відповідає за виявлення об'єктів на кожному кадрі відео за допомогою моделей машинного навчання, зокрема згорткових нейронних мереж для локалізації об'єктів. Основними компонентами цього модулю є попередньо навчені моделі для класифікації та локалізації об'єктів на зображеннях, засоби для аналізу контексту та фону для зменшення кількості хибних спрацьовувань, засоби інтеграції з іншими модулями для визначення місцеположення об'єкта в просторі.

Модуль синтезу нейромережевої моделі для детекції об'єктів займається розробкою та налаштуванням нейромережових моделей, здатних виявляти об'єкти на відео. Підготовка та аугментація даних для тренування моделі, а також її навчання на відповідних вибірках здійснюється в цьому модулі. В результаті генерується модель, яка здатна точно та швидко визначати об'єкти на кадрах відео.

Модуль оптимізації нейромережевої моделі для детекції об'єктів призначений для покращення ефективності синтезованої нейромережевої моделі. Використовуються запропоновані методи спрощення синтезованих моделей, що використовує прунинг, квантування, перенавчання та оптимізацію параметрів, що дозволяє зменшити розмір моделі, скоротити час її виконання та забезпечити ефективне використання ресурсів.

Модуль взаємодії з користувачем та візуалізації створює інтерфейс для користувача, надаючи можливість переглядати результати відстеження об'єктів. Цей модуль забезпечує виведення інформації про відстежені об'єкти у вигляді графічних позначок на відео, візуалізація траєкторій руху та налаштування параметрів відстеження здійснюється саме через цей модуль.

Ці шість модулів забезпечують повний цикл обробки та відстеження об'єктів на відео, що дозволяє ефективно застосовувати систему в реальних умовах, таких як відстеження об'єктів на відео з дронів.

3.2 Функціонування програмного забезпечення для відстеження об'єктів

Функціонування програмного забезпечення для відстеження об'єктів відбувається у двох основних фазах, кожна з яких виконує специфічні завдання для забезпечення ефективного аналізу відеопотоку та визначення об'єктів.

Перша фаза функціонування програмного забезпечення для відстеження об'єктів призначена для синтезу та оптимізації нейромережевої моделі.

Схему функціонування фази навчання програмного забезпечення для відстеження об'єктів подамо за допомогою схеми, зображеної на рис. 3.2.



Рисунок 3.2 – Схема функціонування фази навчання програмного забезпечення для відстеження об'єктів

На цьому етапі використовується навчальна вибірка даних, що містить відео або зображення з розміченими об'єктами. Процес починається з вибору архітектури нейронної мережі залежно від вимог до точності та швидкості. Для реалізації цього проєкту використовувалися згорткові нейронні мережі. Після цього дані вибірки піддаються попередній обробці, яка може включати

аугментацію (обертання, масштабування, зміна яскравості тощо) для підвищення варіативності. Далі виконується навчання моделі шляхом мінімізації функції втрат з використанням оптимізаційних алгоритмів. Після навчання модель проходить етап оптимізації за допомогою запропонованого методу, під час якого використовуються прунінг, квантування та перенавчання. Оптимізація дозволяє зменшити розмір моделі, скоротити час її виконання та забезпечити її придатність до роботи на пристроях із обмеженими ресурсами, таких як дрони. На завершення етапу оптимізована модель готова до інтеграції в систему.

Друга фаза функціонування програмного забезпечення полягає у відстеженні об'єктів, що передбачає обробку відеопотоків у реальному часі за допомогою синтезованої та оптимізованої моделі. Кожен відеокادر, отриманий з камери дрона або іншого пристрою, спершу проходить попередню обробку, яка включає стабілізацію, нормалізацію кольорів та зміну розміру. Оптимізована нейромережева модель потім аналізує кожен кадр, визначаючи положення та тип об'єктів. Алгоритми відстеження, вбудовані в програмне забезпечення, забезпечують збереження траєкторії руху об'єктів, прогнозують їхні позиції на наступних кадрах і коригують помилки. Результати аналізу передаються в модуль візуалізації, який відображає виявлені об'єкти на відео з використанням графічних позначок, а також надає користувачеві доступ до ключових метрик, таких як швидкість обробки та кількість відстежених об'єктів.

Схему функціонування фази відстеження об'єктів за допомогою синтезованої моделі та програмного забезпечення подамо у вигляді схеми, зображеної на рис. 3.3.



Рисунок 3.3 – Схема функціонування фази відстеження об'єктів за допомогою синтезованої моделі та програмного забезпечення

Ці дві фази забезпечують повний цикл функціонування програмного забезпечення, починаючи від побудови моделі до її застосування для обробки відеопотоку. Такий підхід дозволяє досягти високої точності, ефективності та адаптивності системи для практичного використання.

3.3 Розробка бази даних програмного забезпечення для відстеження об'єктів

База даних програмного забезпечення для відстеження об'єктів призначена для підтримки ефективного управління різними типами інформації, зокрема навчальними вибірками, параметрами моделі, результатами відстеження. Для цього використовується багаторівнева структура збереження, яка поєднує кілька форматів і технологій для оптимізації доступу до даних.

Враховуючи особливості програмного забезпечення для відстеження об'єктів, доцільно зберігати такі дані:

- навчальні вибірки. Для збереження великих обсягів навчальних даних (зображень або відео) можуть використовуватися файлові системи або спеціалізовані сховища. Метадані, включаючи розмітку (координати об'єктів і їх класи), можуть бути збережені у форматі JSON або в баі даних;

- параметри нейромережових моделей. Після навчання або оптимізації моделі її архітектура і ваги зберігаються у форматах, що підтримуються відповідними фреймворками. Це дозволяє зберегти сумісність моделі для використання на різних платформах;

- результати відстеження об'єктів. Інформація про відстежені об'єкти (наприклад, їх координати, класи, траєкторії) можуть зберігатися у форматах CSV, JSON або у реляційній базі даних (MySQL). Це спрощує доступ до даних для подальшого аналізу чи візуалізації;

- метадані відеопотоку. Для кожного відеопотоку зберігаються параметри, такі як роздільна здатність, частота кадрів, джерело даних (FPV-дрон), час запису та унікальні ідентифікатори. Ця інформація може бути записана в реляційній базі даних або у спеціальних файлах опису.

- логування процесів. Для відстеження роботи програмного забезпечення створюється система логування, яка записує помилки, обчислювальну ефективність та ключові події.

База даних програмного забезпечення для відстеження об'єктів є структурованою системою, яка зберігає та організовує дані, необхідні для навчання, оптимізації, функціонування та аналізу результатів роботи системи. Вона складається з кількох взаємопов'язаних таблиць, кожна з яких відповідає за зберігання певного типу інформації.

Таблиця "Навчальні дані" містить посилання на навчальні вибірки, які використовуються для створення нейромережових моделей. Поля:

- id: унікальний ідентифікатор вибірки;
- назва_файлу: ім'я файлу зображення або відео;
- розмітка: JSON-поле для збереження координат об'єктів і їх класів;
- джерело: вказівка на джерело вибірки (наприклад, FPV-дрон, публічна база).

Таблиця "Моделі нейромереж" містить інформацію про архітектуру, параметри та результати роботи нейромережових моделей. Поля:

- id: унікальний ідентифікатор моделі;
- назва: назва моделі;
- архітектура: JSON-поле з описом архітектури моделі;
- ваги: посилання на файл із збереженими вагами;
- дата_створення: час і дата створення моделі;
- ефективність: метрики продуктивності моделі (точність, швидкість).

Таблиця "Відеопотоки" зберігає відео, які обробляються або були оброблені програмним забезпеченням. Поля:

- id: унікальний ідентифікатор відеопотоку;
- назва_файлу: ім'я файлу з відео;
- роздільна_здатність: роздільна здатність відео;
- джерело: унікальний ідентифікатор дрона або іншого джерела;
- дата_запису: час і дата запису.

Таблиця "Результати відстеження" зберігає інформацію про об'єкти, відстежені на відео. Поля:

- id: унікальний ідентифікатор запису;

- відео_id: посилання на таблицю "Відеопотоки";
- об'єкт_id: ідентифікатор відстеженого об'єкта;
- координати: JSON-поле з координатами об'єкта на кадрі;
- клас: класифікація об'єкта (наприклад, транспортний засіб, людина);
- час: мітка часу кадру.

Таблиця "Користувачі" призначена для зберігання інформації про операторів або аналітиків. Поля:

- id: унікальний ідентифікатор користувача;
- ім'я: ім'я користувача;
- роль: роль користувача в системі (оператор, аналітик);
- історія_дій: JSON-поле для зберігання історії взаємодії із системою.

Таблиця "Логи та моніторинг" зберігає дані про виконання процесів у програмному забезпеченні. Поля:

- id: унікальний ідентифікатор запису;
- тип: тип події (помилка, успішна операція);
- час: час події;
- деталі: опис події або додаткова інформація.

Для збереження великих даних (зображень, відео) використовуються зовнішні сховища, тоді як метадані зберігаються в реляційній базі даних (MySQL). JSON-поля надають гнучкість для збереження структурованих даних, які можуть відрізнятися за розміром та складністю.

Ця структура забезпечує збереження всіх необхідних даних для роботи системи відстеження об'єктів, що дозволяє ефективно навчати моделі, обробляти відеопотоки та аналізувати результати.

3.4 Проєктування інтерфейсу програмного забезпечення для відстеження об'єктів

Проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для відстеження об'єктів виконано з урахуванням вимог

технічного завдання. При розробленні програмного забезпечення для відстеження об'єктів враховані функціональні вимоги до програми:

- підтримка можливості завдання даних для синтезу моделей для відстеження об'єктів;
- підтримка можливості оптимізації синтезованих моделей для відстеження об'єктів;
- можливість відстеження об'єктів на відео даних;
- можливість відображення відстежених об'єктів на графічних зображеннях.

Інтерфейс форми синтезу та оптимізації нейромережових моделей для відстеження об'єктів наведено на рис. 3.4.

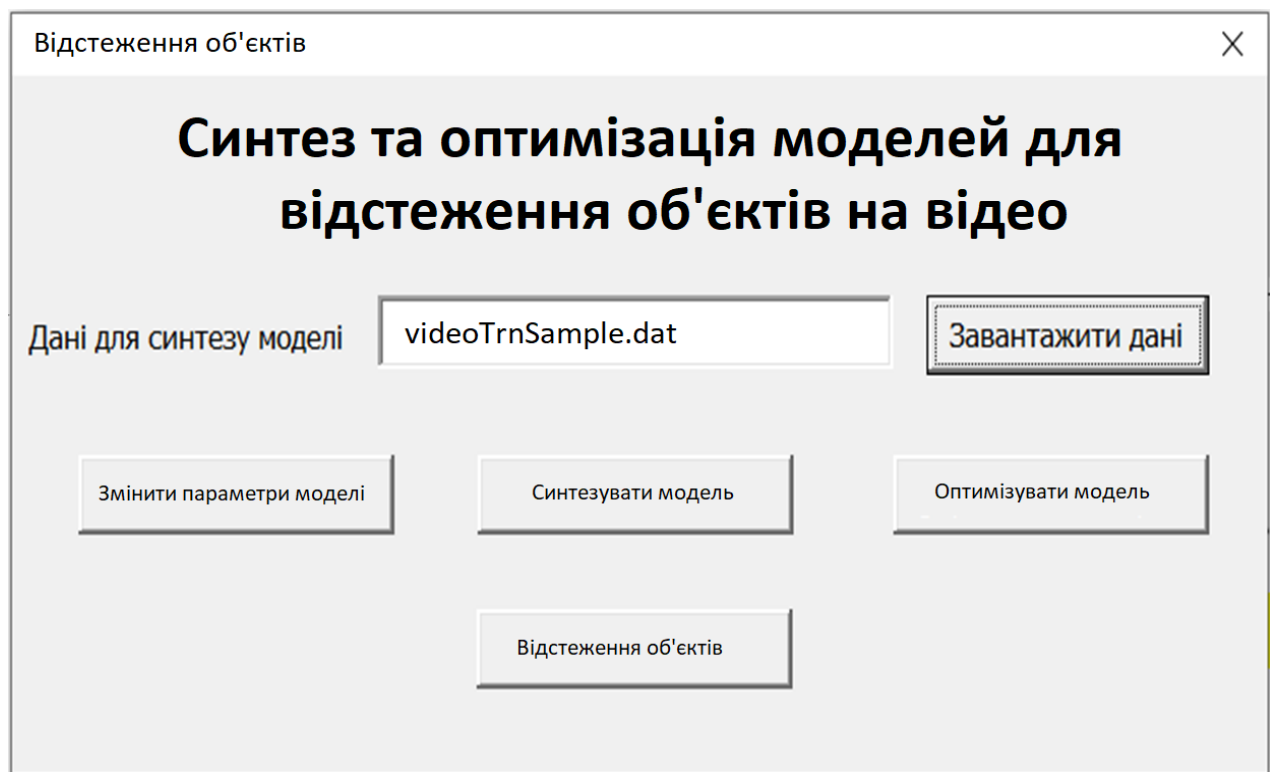


Рисунок 3.4 – Форма синтезу та оптимізації нейромережових моделей для відстеження об'єктів

Інтерфейс форми відстеження об'єктів на відеоданих наведено на рис. 3.5.



Рисунок 3.5 – Форма відстеження об'єктів на відеоданих

3.5 Висновки за розділом 3

Розроблено програмне забезпечення для відстеження об'єктів.

Запропоновано структуру програмного забезпечення для відстеження об'єктів. Визначено, що структура програмного забезпечення для відстеження об'єктів складається з шести основних модулів, кожен з яких виконує специфічну роль в процесі обробки відео та відстеження об'єктів. Основними модулями програмного забезпечення для відстеження об'єктів є такі: модуль обробки відео, модуль відстеження об'єктів, модуль детекції об'єктів, модуль синтезу нейромережевої моделі, модуль оптимізації нейромережевої моделі для детекції об'єктів, модуль взаємодії з користувачем та візуалізації.

Розроблено базу даних програмного забезпечення для відстеження об'єктів, яка призначена для підтримки ефективного управління різними типами інформації, зокрема навчальними вибірками, параметрами моделі, результатами відстеження. Для цього використовується багаторівнева структура збереження, яка поєднує кілька форматів і технологій для оптимізації доступу до даних.

Описано функціонування програмного забезпечення для відстеження об'єктів. Визначено, що функціонування програмного забезпечення для відстеження об'єктів відбувається у двох основних фазах, кожна з яких виконує специфічні завдання для забезпечення ефективного аналізу відеопотоку та визначення об'єктів. Перша фаза призначена для синтезу та оптимізації нейромережевої моделі. Друга фаза функціонування програмного забезпечення полягає у відстеженні об'єктів, що передбачає обробку відеопотоків у реальному часі за допомогою синтезованої та оптимізованої моделі.

Описано особливості реалізації програмного забезпечення для відстеження об'єктів. Виконано проектування інтерфейсу взаємодії користувача з програмним забезпеченням для відстеження об'єктів.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програма призначена для підтримки процесу відстеження об'єктів. Програма написана на мові C++, яка дозволяє ефективно працювати з апаратними ресурсами, забезпечуючи високу швидкість виконання коду. Крім того, C++ забезпечує низькорівневий доступ до апаратного забезпечення, що дозволяє взаємодіяти з камерами, сенсорами та іншими компонентами дронів для забезпечення ефективного оброблення вхідних даних. В якості середовища розробки для створення програмного забезпечення для відстеження об'єктів обрано Qt, що дозволяє створювати програмне забезпечення, яке може працювати на різних операційних системах без значних змін у коді, що є важливим для роботи з дронами, які можуть працювати на різних вбудованих системах. Крім того, середовище Qt забезпечує ефективну роботу навіть у системах з обмеженими ресурсами завдяки оптимізації свого коду.

Програмне забезпечення для підтримки процесу відстеження об'єктів забезпечує виконання таких функцій:

- підтримка можливості завдання даних для синтезу моделей для відстеження об'єктів;
- підтримка можливості оптимізації синтезованих моделей для відстеження об'єктів;
- можливість відстеження об'єктів на відео даних;
- можливість відображення відстежених об'єктів на графічних зображеннях.

Для експлуатації програмного забезпечення для відстеження об'єктів необхідно таке апаратне та програмне забезпечення:

- на наземній станції, де відбуваються розрахунки, пов'язані з синтезом та оптимізацією нейромережових моделей для відстеження об'єктів, необхідна

оперативна пам'ять об'ємом 1 Гб і більше, не менше 500 Мб вільного простору на запам'ятовуючому пристрою апаратного засобу, процесор із тактовою частотою не менше 2 ГГц;

- операційна система на борту дрона та наземної станції;
- комунікаційне програмне забезпечення для передачі відеопотоку;
- дрон або інший пристрій з камерою високої роздільної здатності для захоплення відеопотоку.

Функціональні характеристики розробленого програмного забезпечення для відстеження об'єктів наведено у технічному завданні (додаток А). Фрагмент тексту програмного забезпечення для відстеження об'єктів наведено у додатку Б.

4.2 Характеристики програми для відстеження об'єктів

Програмне забезпечення для відстеження об'єктів складається з шести основних модулів, кожен з яких виконує специфічну роль в процесі обробки відео та відстеження об'єктів. Основними модулями програмного забезпечення для відстеження об'єктів є такі: модуль обробки відео, модуль відстеження об'єктів, модуль детекції об'єктів, модуль синтезу нейромережевої моделі, модуль оптимізації нейромережевої моделі для детекції об'єктів, модуль взаємодії з користувачем та візуалізації.

База даних програмного забезпечення для відстеження об'єктів призначена для підтримки ефективного управління різними типами інформації, зокрема навчальними вибірками, параметрами моделі, результатами відстеження. Для цього використовується багаторівнева структура збереження, яка поєднує кілька форматів і технологій для оптимізації доступу до даних.

Функціонування програмного забезпечення для відстеження об'єктів відбувається у двох основних фазах, кожна з яких виконує специфічні завдання для забезпечення ефективного аналізу відеопотоку та визначення

об'єктів. Перша фаза призначена для синтезу та оптимізації нейромережевої моделі. Друга фаза функціонування програмного забезпечення полягає у відстеженні об'єктів, що передбачає обробку відеопотоків у реальному часі за допомогою синтезованої та оптимізованої моделі.

Програмне забезпечення для відстеження об'єктів характеризується здатністю обробляти відеопотік у реальному часі, забезпечуючи виявлення та трекінг об'єктів на основі нейромережевих алгоритмів. Використовується оптимізована архітектура згорткових нейронних мереж, що дозволяє адаптуватися до обмежених обчислювальних ресурсів. Інтерфейс програмного забезпечення є інтуїтивно зрозумілим, з адаптивним дизайном, що дозволяє працювати на різних системах. Підтримується модульна архітектура, що сприяє легкому оновленню та розширенню функціональності.

Забезпечується прийнятна точність та надійність відстеження об'єктів, навіть у складних умовах, таких як погане освітлення або наявність перешкод. Програмне забезпечення має низьке енергоспоживання.

У програмному забезпеченні використовуються підходи до оптимізації синтезованих нейромережевих моделей для відстеження об'єктів, що дозволяє значно скоротити час обробки даних без втрати якості результатів.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Для запуску програмного забезпечення для відстеження об'єктів на наземній станції необхідно запустити відповідний виконуваний файл.

4.3.2 Вхідні й вихідні дані

Вхідними даними до програмного забезпечення для відстеження об'єктів є інформація про відео дані для синтезу моделі, синтезовані та оптимізовані для використання моделі, відео дані для відстеження об'єктів.

Вихідними даними програмного забезпечення для відстеження об'єктів є синтезовані та оптимізовані моделі для відстеженні об'єктів, відстежені об'єкти на заданих відео даних.

4.3.3 Повідомлення

Користувач програмного забезпечення для відстеження об'єктів може отримати такі повідомлення: повідомлення про успішне завершення синтезу або оптимізації нейромережевої моделі для відстеження об'єктів.

4.4 Виконання програмного забезпечення для відстеження об'єктів

Після запуску програмного забезпечення для відстеження об'єктів користувачу відображається форма для реалізації першої фази використання програми (рис. 4.1), призначеної для синтезу та оптимізації нейромережевої моделі для відстеження об'єктів. Користувач повинен обрати файл з навчальними даними. За замовченням синтезується модель на основі згорткової нейронної мережі. У випадку необхідності зміни типу моделі та її параметрів користувачу необхідно натиснути кнопку Змінити параметри моделі. Для синтезу та оптимізації нейромережевих моделей для відстеження об'єктів на основі заданої вибірки даних користувачу необхідно натиснути кнопки Синтезувати модель та Оптимізувати модель, відповідно. Після синтезу або оптимізації моделі створюється файл з її архітектурою та параметрами.

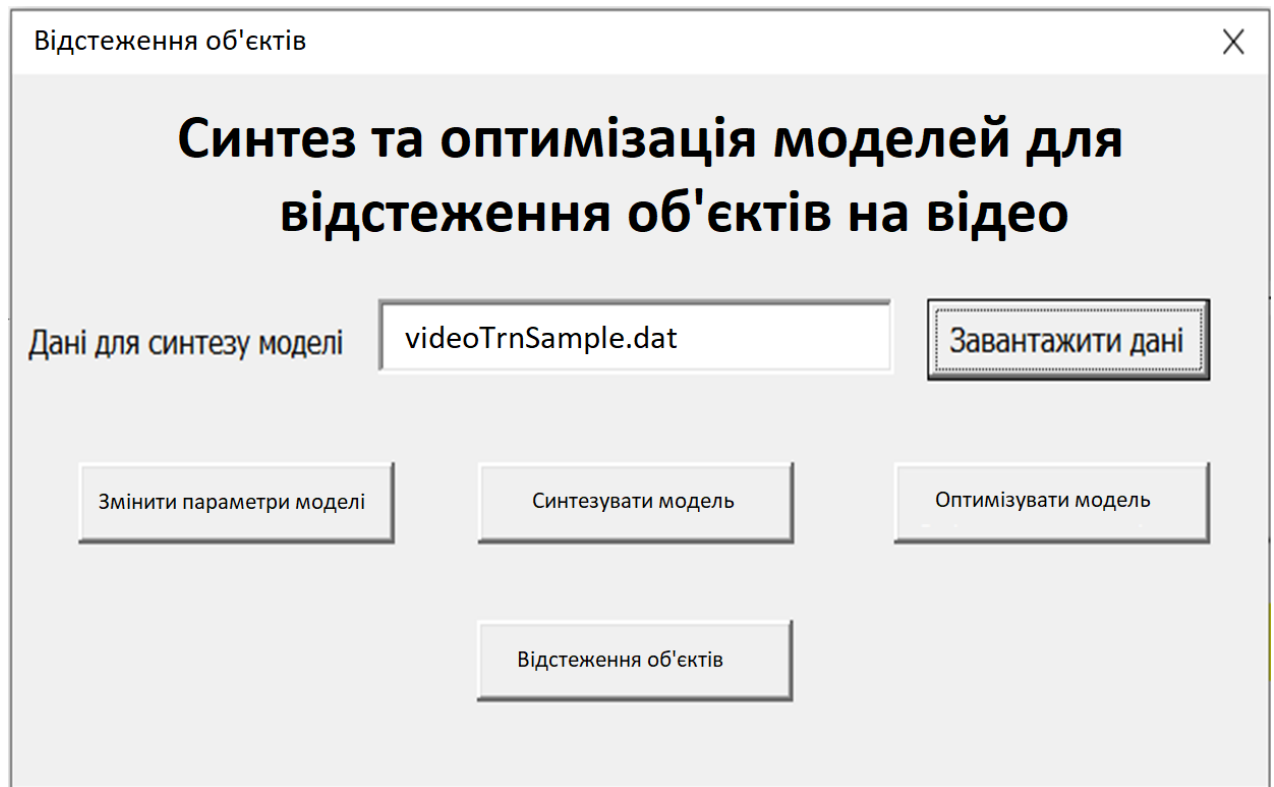


Рисунок 4.1 – Форма синтезу та оптимізації нейромережових моделей для відстеження об'єктів

Маючи синтезовану та оптимізовану модель можна переходити до її використання для відстеження об'єктів. Модель може використовуватися на дронах або інших пристроях, здатних отримувати відеопотоки з камер відео спостереження. Для перевірки ефективності роботи програми та синтезованої моделі реалізовано візуальний інтерфейс у вигляді форми користувача (рис. 4.2), де відображено відео дані та результати відстеження об'єктів.

Користувач повинен обрати файл з відео даними для відстеження об'єктів. Після цього необхідно натиснути кнопку Почати відстеження, внаслідок чого у відповідній області форми буде відображено відеопотік з рамками на розпізнаних програмою об'єктах. При цьому потік повинен містити такі ж типи об'єктів (наприклад, автомобілі), за якими відбувалося навчання моделі. У випадку необхідності використання іншої моделі необхідно натиснути кнопку Змінити модель та обрати нову модель. Результати відстеження об'єктів (відеопотік з розпізнаними об'єктами) можна

зберегти за допомогою відповідної кнопки. При необхідності синтезу нової моделі користувачу треба натиснути кнопку **Перейти до синтезу моделі**, внаслідок чого відкриється форма першої фази для синтезу та оптимізації розпізнавальної моделі.



Рисунок 4.2 – Форма відстеження об'єктів на відеоданих

4.5 Тестування програмного забезпечення для відстеження об'єктів

Виконано тестування програмного забезпечення для відстеження об'єктів.

Виявлено, що програма для підтримки процесу відстеження об'єктів функціонує правильно та злагоджено. Вона реалізує всі функціональні вимоги

та успішно виконує свою основну задачу підтримки процесу відстеження об'єктів.

Розроблене програмне забезпечення дозволяє забезпечувати автоматизацію процесів, пов'язаних з підтримкою процесу відстеження об'єктів.

4.6 Висновки за розділом 4

Описано програмне забезпечення для відстеження об'єктів. Виконано тестування розробленого програмного забезпечення для відстеження об'єктів. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою процесу відстеження об'єктів.

ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено процес розробки програмного забезпечення для відстеження об'єктів.

Визначено, що відстеження об'єктів на відеоданих є процесом ідентифікації, локалізації та відстеження об'єктів у кожному кадрі відеопотоку. Ця задача полягає в аналізі послідовності зображень з метою визначення об'єктів у просторі та часі. Зазвичай використовуються алгоритми комп'ютерного зору та машинного навчання, включаючи згорткові нейронні мережі, для обробки відеопотоку. Відзначено, що процес створення програмного забезпечення для відстеження об'єктів на відеоданих включає кілька етапів, які забезпечують його ефективну функціональність та відповідність технічним вимогам.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для відстеження об'єктів. Проте деякі програмні засоби для відстеження об'єктів на відео можуть мати низку недоліків, які впливають на їхню ефективність і практичність. Одним із головних викликів є висока потреба в обчислювальних ресурсах, оскільки алгоритми відстеження, особливо ті, що базуються на глибокому навчанні, вимагають значної потужності процесора або графічного процесора. Це створює обмеження для їх використання на компактних дронах, де апаратні ресурси обмежені. Крім того, програмні засоби часто вимагають спеціальної настройки для роботи з різними сценаріями, що може бути складним для недосвідчених користувачів. Ще одним викликом є енергоспоживання, а також те, що використання складних алгоритмів скорочує час роботи дрона через швидкий розряд батареї. Тому актуальною є розробка програмного забезпечення для відстеження об'єктів.

Сформульовано функціональні вимоги до програмного забезпечення для відстеження об'єктів.

Для реалізації програмного забезпечення для відстеження об'єктів обрано мову програмування C++, яка дозволяє ефективно працювати з апаратними ресурсами, забезпечуючи високу швидкість виконання коду. Це є критично важливим для обробки відеопотоків у реальному часі, де необхідно виконувати складні обчислення з мінімальною затримкою. Крім того, C++ має широкий спектр бібліотек, які можуть бути використані для розробки програмного забезпечення для обробки зображень та відео. Іншою перевагою є те, що завдяки своїй низькорівневій природі C++ забезпечує мінімальні затримки при обробці даних, що є важливим для відстеження об'єктів у відеопотоці. C++ забезпечує низькорівневий доступ до апаратного забезпечення, що дозволяє взаємодіяти з камерами, сенсорами та іншими компонентами дронів для забезпечення ефективного оброблення вхідних даних.

Для створення програмного забезпечення для відстеження об'єктів обрано середовище розробки Qt, що дозволяє створювати програмне забезпечення, яке може працювати на різних операційних системах без значних змін у коді, що є важливим для роботи з дронами, які можуть працювати на різних вбудованих системах. Крім того, середовище Qt забезпечує ефективну роботу навіть у системах з обмеженими ресурсами завдяки оптимізації свого коду.

Визначено, що згорткові нейронні мережі є найефективнішими для відстеження об'єктів на відео завдяки їхній здатності швидко та точно обробляти просторові ознаки. Вони забезпечують високу продуктивність у реальному часі, що робить їх оптимальним вибором для відстеження об'єктів за допомогою FPV-дронів та інших завдань, де потрібна швидкість і точність. Хоча RNN і LSTM краще працюють із часовими залежностями, їх швидкодія поступається CNN. У сучасних завданнях трансформери демонструють високу точність і адаптивність, але їхня складність і вимоги до апаратного забезпечення роблять їх менш практичними в багатьох реальних сценаріях.

Показано, що оптимізація моделей на основі CNN для їх використання на FPV-дронах є критично важливою через обмеження в ресурсах, енергії та обчислювальній потужності. Завдяки методам скорочення можна зменшити модель, зберігаючи її функціональність, забезпечуючи реальну роботу у складних динамічних умовах. Оптимізовані CNN дозволяють FPV-дронам ефективно виконувати завдання відстеження об'єктів у реальному часі без значного збільшення вартості чи ваги системи.

Новизна роботи полягає у тому, що розроблено метод спрощення синтезованих згорткових нейромережових моделей для відстеження об'єктів на відео, який включає етапи прунингу, квантування, передачі знань та архітектурних оптимізацій. Унікальність запропонованого методу полягає в комплексному застосуванні кількох підходів для зменшення складності моделі при збереженні високої продуктивності та точності. Цей метод дозволяє синтезувати та програмно реалізувати ефективні моделі для відстеження об'єктів на ресурсозалежних пристроях, таких як дрони.

Практична цінність результатів роботи полягає у тому, що розроблено програмне забезпечення для відстеження об'єктів.

Запропоновано структуру програмного забезпечення для відстеження об'єктів. Визначено, що структура програмного забезпечення для відстеження об'єктів складається з шести основних модулів, кожен з яких виконує специфічну роль в процесі обробки відео та відстеження об'єктів. Основними модулями програмного забезпечення для відстеження об'єктів є такі: модуль обробки відео, модуль відстеження об'єктів, модуль детекції об'єктів, модуль синтезу нейромережової моделі, модуль оптимізації нейромережової моделі для детекції об'єктів, модуль взаємодії з користувачем та візуалізації.

Розроблено базу даних програмного забезпечення для відстеження об'єктів, яка призначена для підтримки ефективного управління різними типами інформації, зокрема навчальними вибірками, параметрами моделі, результатами відстеження. Для цього використовується багаторівнева

структура збереження, яка поєднує кілька форматів і технологій для оптимізації доступу до даних.

Описано функціонування програмного забезпечення для відстеження об'єктів. Визначено, що функціонування програмного забезпечення для відстеження об'єктів відбувається у двох основних фазах, кожна з яких виконує специфічні завдання для забезпечення ефективного аналізу відеопотоку та визначення об'єктів. Перша фаза призначена для синтезу та оптимізації нейромережевої моделі. Друга фаза функціонування програмного забезпечення полягає у відстеженні об'єктів, що передбачає обробку відеопотоків у реальному часі за допомогою синтезованої та оптимізованої моделі.

Описано особливості реалізації програмного забезпечення для відстеження об'єктів. Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для відстеження об'єктів.

Виконано тестування розробленого програмного забезпечення для відстеження об'єктів. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою процесу відстеження об'єктів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Complete Guide to Object Tracking: Best AI Models, Tools and Methods [Electronic resource]. – Access mode: <https://supervisely.com/blog/complete-guide-to-object-tracking-best-ai-models-tools-and-methods-in-2023/>.
2. Video processing, Object detection & Tracking [Electronic resource]. – Access mode: https://stonesoup.readthedocs.io/en/v0.1b3/auto_demos/Video_Processing.html.
3. Video Object Tracking with Optical Flow and Yolo [Electronic resource]. – Access mode: <https://medium.com/softplus-publication/video-object-tracking-with-yolov8-and-sort-library-e28444b189aa>.
4. Object tracking in Movie Edit Pro [Electronic resource]. – Access mode: <https://www.magix.com/int/magazine/video/object-tracking/>.
5. Object tracking in videos [Electronic resource]. – Access mode: <https://www.rsipvision.com/object-tracking-in-videos/>.
6. Motion Tracking Video Editor [Electronic resource]. – Access mode: <https://www.kapwing.com/tools/pinning>.
7. Multi-object tracking [Electronic resource]. – Access mode: <https://github.com/swarm-lab/trackR>.
8. Top 7 Open-Source Object Tracking Tools [Electronic resource]. – Access mode: <https://blog.roboflow.com/top-object-tracking-software/>.
9. Paid and Free Motion Tracking Software [Electronic resource]. – Access mode: <https://www.videoproc.com/video-editor/motion-tracking-software.htm>.
10. Best Motion Tracking Software [Electronic resource]. – Access mode: <https://www.movavi.com/learning-portal/motion-tracking-software.html>.
11. SynthEyes [Electronic resource]. – Access mode: <https://borisfx.com/products/syntheyes/>.
12. Mocha Pro [Electronic resource]. – Access mode: <https://borisfx.com/products/mocha-pro/?collection=mocha-pro>.

13. Blender [Electronic resource]. – Access mode: <https://www.blender.org/>.
14. Introduction to C++ programming [Electronic resource]. – Access mode: <https://cpp-lang.net/learn/>.
15. C++ Programming Language [Electronic resource]. – Access mode: <https://www.geeksforgeeks.org/c-plus-plus/>.
16. Qt Documentation [Electronic resource]. – Access mode: <https://doc.qt.io/>.
17. The Qt Project [Electronic resource]. – Access mode: <https://contribute.qt-project.org/>.
18. What Is Pattern Recognition? Working, Types, and Applications [Electronic resource]. – Access mode: <https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-pattern-recognition/>.
19. What is a Neural Network [Electronic resource]. – Access mode: <https://www.cloudflare.com/ru-ru/learning/ai/what-is-neural-network/>.
20. Object Detection with Neural Networks [Electronic resource]. – Access mode: <https://towardsdatascience.com/object-detection-with-neural-networks-a4e2c46b4491>.

ДОДАТОК А
Технічне завдання

Вступ

Програмне забезпечення може використовуватися для підтримки процесу для відстеження об'єктів.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Дослідження та програмна реалізація застосунку відстеження об'єктів на відео з FPV-дрона», затверджене наказом Національного університету «Запорізька політехніка» № 398 від 07 жовтня 2024 р.

A.2 Призначення розробки

Програмний продукт призначений для підтримки процесу відстеження об'єктів на відео.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до функціональних характеристик

Програмне забезпечення для підтримки процесу відстеження об'єктів забезпечує виконання таких функцій:

- підтримка можливості завдання даних для синтезу моделей для відстеження об'єктів;
- підтримка можливості оптимізації синтезованих моделей для відстеження об'єктів;
- можливість відстеження об'єктів на відео даних;
- можливість відображення відстежених об'єктів на графічних зображеннях.

А.3.2 Вимоги до інтерфейсу програми

Інтерфейс програмного забезпечення для відстеження об'єктів повинен бути зручним для користувачів. Повинна бути забезпечена можливість роботи в візуальному режимі.

А.3.3 Вимоги до надійності

Програмне забезпечення для відстеження об'єктів повинно забезпечити надійне функціонування.

А.3.4 Умови експлуатації

Для експлуатації програмного забезпечення для відстеження об'єктів необхідна наявність персонального комп'ютера.

А.3.5 Вимоги до складу та параметрів технічний засобів

Для експлуатації програмного забезпечення для відстеження об'єктів необхідно таке апаратне та програмне забезпечення:

- на наземній станції, де відбуваються розрахунки, пов'язані з синтезом та оптимізацією нейромережових моделей для відстеження об'єктів, необхідна оперативна пам'ять об'ємом 1 Гб і більше, не менше 500 Мб вільного простору на запам'ятовуючому пристрою апаратного засобу, процесор із тактовою частотою не менше 2 ГГц;
- операційна система на борту дрона та наземної станції;
- комунікаційне програмне забезпечення для передачі відеопотоку;

– дрон або інший пристрій з камерою високої роздільної здатності для захоплення відеопотоку.

A.3.6 Вимоги до маркування і пакування

Програма для підтримки процесу відстеження об'єктів може бути записана на будь-якому носії інформації.

На пакуванні повинна бути назва програми – «Програмне забезпечення для відстеження об'єктів на відео».

A.4 Вхідні дані до роботи

Вхідними даними до програмного забезпечення для відстеження об'єктів є інформація про відео дані для синтезу моделі, синтезовані та оптимізовані для використання моделі, відео дані для відстеження об'єктів.

Вихідними даними програмного забезпечення для відстеження об'єктів є синтезовані та оптимізовані моделі для відстеження об'єктів, відстежені об'єкти на заданих відео даних.

ДОДАТОК Б
Фрагмент тексту програми

```

class VdFram {
public:
    int dovg;
    int vysot;
    strd::vector<uit> pxls;

    VdFram(int w, int h) : dovg(w), vysot(h) {
        pxls.resize(w * h * 3);
    }

    void clear() {
        strd::memset(pxls.data(), 0, pxls.size());
    }
};

class VdProcessingModule {
private:
    strd::ifstream vdFile;
    int dovg;
    int vysot;
    int framSz;
    bool isOpn;

public:
    VdProcessingModule() : dovg(640), vysot(480), framSz(dovg *
vysot * 3), isOpn(false) {}

    bool openVd(const strd::string& vdPath) {
        vdFile.open(vdPath, strd::ios::binary);
        if (!vdFile.is_open()) {
            strd::cerr << "Osrt opening vd file." << strd::endl;
            return false;
        }
        isOpn = true;
        return true;
    }

    void closeVd() {
        if (isOpn) {
            vdFile.close();
            isOpn = false;
        }
    }

    bool readFgram(VdFram& fgram) {
        if (!isOpn || vdFile.eof()) {
            return false;
        }
    }

```

```

    }

    vdFile.read(reinterpret_cast<char*>(fgram.pxls.data()),
framSz);
    if (vdFile.gcount() != framSz) {
        return false;
    }

    return true;
}

void processFgram(VdFram& fgram) {
    for (int i = 0; i < fgram.dovg * fgram.vysot; i++) {
        uit r = fgram.pxls[i * 3];
        uit g = fgram.pxls[i * 3 + 1];
        uit b = fgram.pxls[i * 3 + 2];
        uit gray = static_cast<uit>((r + g + b) / 3);

        fgram.pxls[i * 3] = gray;
        fgram.pxls[i * 3 + 1] = gray;
        fgram.pxls[i * 3 + 2] = gray;
    }
}

void displayFgram(const VdFram& fgram) {
    strd::cout << "Fgram processed: " << fgram.dovg << "x" <<
fgram.vysot << strd::endl;
}
};

class NrlNtwrk {
public:
    NrlNtwrk(int inptsz, int hddnsz, int outptsz) {
        this->inptsz = inptsz;
        this->hddnsz = hddnsz;
        this->outptsz = outptsz;

        srand(time(0));

        hddn_wghts.resize(inptsz, strd::vector<double>(hddnsz));
        for (int i = 0; i < inptsz; ++i) {
            for (int j = 0; j < hddnsz; ++j) {
                hddn_wghts[i][j] = (rand() % 100) / 100.0 - 0.5;
            }
        }
    }
};

```

```

        outpt_wghts.resize(hddnsz,
strd::vector<double>(outptsz));
        for (int i = 0; i < hddnsz; ++i) {
            for (int j = 0; j < outptsz; ++j) {
                outpt_wghts[i][j] = (rand() % 100) / 100.0 - 0.5;
            }
        }

        hddn_bias.resize(hddnsz, 0);
        outpt_bias.resize(outptsz, 0);
    }

    double sgmd(double x) {
        return 1.0 / (1.0 + exp(-x));
    }

    strd::vector<double> predict(const strd::vector<double>&
input) {
        // Вхід до прихованого шару
        strd::vector<double> hddn_shar_outpt(hddnsz);
        for (int i = 0; i < hddnsz; ++i) {
            hddn_shar_outpt[i] = 0;
            for (int j = 0; j < inptsz; ++j) {
                hddn_shar_outpt[i] += input[j] *
hddn_wghts[j][i];
            }
            hddn_shar_outpt[i] += hddn_bias[i];
            hddn_shar_outpt[i] = sgmd(hddn_shar_outpt[i]);
        }

        strd::vector<double> outpt_shar_outpt(outptsz);
        for (int i = 0; i < outptsz; ++i) {
            outpt_shar_outpt[i] = 0;
            for (int j = 0; j < hddnsz; ++j) {
                outpt_shar_outpt[i] += hddn_shar_outpt[j] *
outpt_wghts[j][i];
            }
            outpt_shar_outpt[i] += outpt_bias[i];
            outpt_shar_outpt[i] = sgmd(outpt_shar_outpt[i]);
        }

        return outpt_shar_outpt;
    }

    void train(const strd::vector<strd::vector<double>>& inputs,
                const strd::vector<strd::vector<double>>& trgt,
                int iters, double lrnrt) {

```

```

for (int iter = 0; iter < iters; ++iter) {
    for (size_t i = 0; i < inputs.size(); ++i) {
        strd::vector<double> input = inputs[i];
        strd::vector<double> trgt = trgt[i];

        strd::vector<double> outpt = predict(input);

        strd::vector<double> outpt_osrts(outptsz);
        for (int j = 0; j < outptsz; ++j) {
            outpt_osrts[j] = trgt[j] - outpt[j];
        }

        for (int j = 0; j < hddnsz; ++j) {
            for (int k = 0; k < outptsz; ++k) {
                double dltr = outpt_osrts[k] * outpt[k] *
(1 - outpt[k]) * hddn_shar_outpt[j];
                outpt_wghts[j][k] += lnrnt * dltr;
            }
        }

        for (int k = 0; k < outptsz; ++k) {
            outpt_bias[k] += lnrnt * outpt_osrts[k];
        }

        strd::vector<double> hddn_osrts(hddnsz);
        for (int j = 0; j < hddnsz; ++j) {
            hddn_osrts[j] = 0;
            for (int k = 0; k < outptsz; ++k) {
                hddn_osrts[j] += outpt_osrts[k] *
outpt_wghts[j][k];
            }
            hddn_osrts[j] *= hddn_shar_outpt[j] * (1 -
hddn_shar_outpt[j]);
        }

        for (int j = 0; j < inptsz; ++j) {
            for (int k = 0; k < hddnsz; ++k) {
                double dltr = hddn_osrts[k] * input[j];
                hddn_wghts[j][k] += lnrnt * dltr;
            }
        }

        for (int k = 0; k < hddnsz; ++k) {
            hddn_bias[k] += lnrnt * hddn_osrts[k];
        }
    }
}

```

```

        strd::cout << "Iter " << iter + 1 << "/" << iters <<
" completed.\n";
    }
    hddn_wghts.resize(inptsz, strd::vector<double>(hddnsz));
    for (int i = 0; i < inptsz; ++i) {
        for (int j = 0; j < hddnsz; ++j) {
            hddn_wghts[i][j] = (rand() % 100) / 100.0 - 0.5;
        }
    }

    outpt_wghts.resize(hddnsz,
strd::vector<double>(outptsz));
    for (int i = 0; i < hddnsz; ++i) {
        for (int j = 0; j < outptsz; ++j) {
            outpt_wghts[i][j] = (rand() % 100) / 100.0 - 0.5;
        }
    }

    hddn_bias.resize(hddnsz, 0);
    outpt_bias.resize(outptsz, 0);
}

strd::vector<double> predict(const strd::vector<double>&
input) {
    // Вхід до прихованого шару
    strd::vector<double> hddn_shar_outpt(hddnsz);
    for (int i = 0; i < hddnsz; ++i) {
        hddn_shar_outpt[i] = 0;
        for (int j = 0; j < inptsz; ++j) {
            hddn_shar_outpt[i] += input[j] *
hddn_wghts[j][i];
        }
        hddn_shar_outpt[i] += hddn_bias[i];
        hddn_shar_outpt[i] = sgmd(hddn_shar_outpt[i]);
    }

    strd::vector<double> outpt_shar_outpt(outptsz);
    for (int i = 0; i < outptsz; ++i) {
        outpt_shar_outpt[i] = 0;
        for (int j = 0; j < hddnsz; ++j) {
            outpt_shar_outpt[i] += hddn_shar_outpt[j] *
outpt_wghts[j][i];
        }
        outpt_shar_outpt[i] += outpt_bias[i];
        outpt_shar_outpt[i] = sgmd(outpt_shar_outpt[i]);
    }

    return outpt_shar_outpt;
}

```

```

    }

    void optimizeModel(const strd::vector<strd::vector<double>>&
inputs,
                                const strd::vector<strd::vector<double>>&
trgts,
                                int iters, double lrnrt) {

    for (int iter = 0; iter < iters; ++iter) {
        for (size_t i = 0; i < inputs.size(); ++i) {
            strd::vector<double> input = inputs[i];
            strd::vector<double> trgt = trgts[i];

            strd::vector<double> outpt = predict(input);

            strd::vector<double> outpt_osrts(outptsz);
            for (int j = 0; j < outptsz; ++j) {
                outpt_osrts[j] = trgt[j] - outpt[j];
            }

            for (int j = 0; j < hddnsz; ++j) {
                for (int k = 0; k < outptsz; ++k) {
                    double dltr = outpt_osrts[k] * outpt[k] *
(1 - outpt[k]) * hddn_shar_outpt[j];
                    outpt_wghts[j][k] += lrnrt * dltr;
                }
            }

            for (int k = 0; k < outptsz; ++k) {
                outpt_bias[k] += lrnrt * outpt_osrts[k];
            }

            strd::vector<double> hddn_osrts(hddnsz);
            for (int j = 0; j < hddnsz; ++j) {
                hddn_osrts[j] = 0;
                for (int k = 0; k < outptsz; ++k) {
                    hddn_osrts[j] += outpt_osrts[k] *
outpt_wghts[j][k];
                }
                hddn_osrts[j] *= hddn_shar_outpt[j] * (1 -
hddn_shar_outpt[j]);
            }

            for (int j = 0; j < inptsz; ++j) {
                for (int k = 0; k < hddnsz; ++k) {
                    double dltr = hddn_osrts[k] * input[j];
                    hddn_wghts[j][k] += lrnrt * dltr;
                }
            }
        }
    }
}

```

```

        }

        for (int k = 0; k < hddnsz; ++k) {
            hddn_bias[k] += lrnrt * hddn_osrts[k];
        }
    }

    strd::cout << "Iter " << iter + 1 << "/" << iters <<
    " completed.\n";
    }
}

struct Pxl {
    uit r, g, b;
};

double pxlDstc(const Pxl& pr1, const Pxl& pr2) {
    return sqrt(pow(pr1.r - pr2.r, 2) + pow(pr1.g - pr2.g, 2) +
    pow(pr1.b - pr2.b, 2));
}

class ObjectTracker {
public:
    strd::vector<strd::vector<Pxl>> prvis_fgram;
    strd::vector<strd::vector<Pxl>> krnt_fgram;

    int objktx, objkty;

    ObjectTracker(int dovlg, int vysot) {
        prvis_fgram.resize(vysot, strd::vector<Pxl>(dovlg));
        krnt_fgram.resize(vysot, strd::vector<Pxl>(dovlg));
    }

    void initialize(const strd::vector<strd::vector<Pxl>>& fgram,
    int x, int y) {
        prvis_fgram = fgram;
        objktx = x;
        objkty = y;
    }

    void trackObject(const strd::vector<strd::vector<Pxl>>&
    fgram) {
        krnt_fgram = fgram;
        double min_dstc =
    strd::numeric_limits<double>::infinity();
        int new_x = objktx, new_y = objkty;

        for (int y = 0; y < krnt_fgram.size(); ++y) {

```

```

        for (int x = 0; x < krnt_fgram[0].size(); ++x) {
            double dstc =
pxlDstc(prvis_fgram[objkty][objktx], krnt_fgram[y][x]);
            if (dstc < min_dstc) {
                min_dstc = dstc;
                new_x = x;
                new_y = y;
            }
        }

        objktx = new_x;
        objkty = new_y;

        prvis_fgram = krnt_fgram;
    }

    strd::pair<int, int> getObjectPosition() {
        return {objktx, objkty};
    }
};

class VdFram {
public:
    strd::vector<strd::vector<Pxl>> fgram;
    int dovg, vysot;

    VdFram(int w, int h) : dovg(w), vysot(h) {
        fgram.resize(vysot, strd::vector<Pxl>(dovg));
    }

    void generateRandomFgram() {
        for (int y = 0; y < vysot; ++y) {
            for (int x = 0; x < dovg; ++x) {
                fgram[y][x] = {static_cast<uit>(rand() % 256),
static_cast<uit>(rand() % 256), static_cast<uit>(rand() % 256)};
            }
        }
    }

    void displayFgram() {
        for (int y = 0; y < vysot; ++y) {
            for (int x = 0; x < dovg; ++x) {
                char smbl = (fgram[y][x].r > 128) ? '#' : '.';
                strd::cout << smbl;
            }
            strd::cout << strd::endl;
        }
    }
};

```

```

        }
    }
};

class UsrIntrctn {
public:
    int usrx, usry;

    UsrIntrctn() : usrx(0), usry(0) {}

    void promptUserForObjectPosition() {
        strd::cout << "(x y): ";
        strd::cin >> usrx >> usry;
    }

    void updateObjectPosition() {
        strd::cout << "new (x y): ";
        strd::cin >> usrx >> usry;
    }

    strd::pair<int, int> getUserPosition() {
        return {usrx, usry};
    }
};

class ObjectVisualization {
public:
    void drawTrackedObject(const VdFram& fgram, int x, int y) {
        strd::cout << "Visualization:" << strd::endl;
        fgram.displayFgram();

        strd::cout << "position: (" << x << ", " << y << ")" <<
strd::endl;
    }
};

int main() {
    const int dovg = 20;
    const int vysot = 10;

    VdFram fgram(dovg, vysot);
    UsrIntrctn user;
    ObjectVisualization vslzr;

    fgram.generateRandomFgram();

    user.promptUserForObjectPosition();
    auto [objktx, objkty] = user.getUserPosition();

```

```
vslzr.drawTrackedObject(fgram, objktx, objkty);

user.updateObjectPosition();
auto [new_x, new_y] = user.getUserPosition();

vslzr.drawTrackedObject(fgram, new_x, new_y);

return 0;
}
```

ДОДАТОК В
Слайди презентації

**Міністерство освіти і науки України
Національний університет «Запорізька політехніка»
Кафедра програмних засобів**

Дослідження та програмна реалізація застосування відстеження об'єктів на відео з FPV-дрона

Виконав: Володимир СУДАКОВ

ст. групи КНТ-133м

Керівник: Сергій СУББОТІН

д.т.н., професор, завідувач кафедри програмних засобів

Рисунок В.1 – Слайд 1

Об'єкт, предмет та мета роботи

- Об'єкт дослідження – процес розробки програмного забезпечення для відстеження об'єктів.
- Предмет дослідження – програмні засоби для відстеження об'єктів.
- Метою роботи є дослідження та розробка програмного забезпечення для відстеження об'єктів на відео з FPV-дрона.

Рисунок В.2 – Слайд 2

Завдання роботи

Для досягнення поставленої мети у кваліфікаційній роботі магістра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для відстеження об'єктів;
- розробити метод спрощення синтезованих згорткових нейромережових моделей для відстеження об'єктів на відео;
- здійснити проєктування програмного забезпечення для відстеження об'єктів;
- створити програмне забезпечення для відстеження об'єктів;
- виконати тестування розробленого програмного забезпечення для відстеження об'єктів.

3

Рисунок В.3 – Слайд 3

Порівняння існуючих аналогів

Критерій порівняння	SynthEyes	Mocha Pro	Blender
Автоматичне відстеження об'єктів	+	+	+—
Обробка складних сцен	+—	+	+—
Складність використання	+—	+—	+
Відкритий код	—	—	+
Висока точність відстеження об'єктів	+	+	—
Підтримка великої кількості відео форматів	+	+—	+

4

Рисунок В.4 – Слайд 4

Порівняння мов програмування

Критерій порівняння мов програмування	Мова програмування		
	C++	Python	Ruby
Висока продуктивність	+	+-	-
Оптимізація використання ресурсів	+	+-	+-
Реалізація алгоритмів у реальному часі	+	+	-
Зручність для створення ПЗ для відстеження об'єктів	+	-	-
Можливості роботи з апаратними компонентами	+	+-	+-

5

Рисунок В.5 – Слайд 5

Порівняння середовищ розробки

Критерій порівняння середовищ розробки	Середовища розробки		
	Qt	Visual Studio	Atom
Швидкість розробки	+	+	+-
Продуктивність	+	+-	+
Зручність інтеграції з C++	+	+	-
Підтримка роботи з мультимедіа	+	+	+-
Гнучкість у розробці модульної архітектури	+	+-	+

6

Рисунок В.6 – Слайд 6

Порівняння видів нейронних мереж для відстеження об'єктів на відео

Критерій порівняння нейронних мереж	Нейронні мережі для обробки зображень				
	CNN	RNN	LSTM/GRU	3D CNN	Trans-formers (DETR/ViT)
Точність у розпізнаванні об'єктів	+	–	+-	+	+
Обробка просторових ознак	+	–	–	+	+
Швидкість роботи в реальному часі	+	+-	+-	–	+-
Адаптація до зміни умов (тло, освітлення)	+	+-	+-	+	+
Складність архітектури	+	+	+-	–	–
Вимоги до апаратних ресурсів	+	+	+-	–	–

Рисунок В.7 – Слайд 7

Метод спрощення синтезованих згорткових НМ для відстеження об'єктів

Метод спрощення синтезованих згорткових нейромережевих моделей для відстеження об'єктів на відео:

- здійснити базове навчання моделі для отримання максимальної точності
- послідовно застосувати прунинг і квантування для зменшення моделі
- використати підхід Knowledge Distillation для перенесення знань у спрощену модель
- додатково провести оптимізацію архітектури для сумісності з цільовими пристроями дрона.

Рисунок В.8 – Слайд 8

Структура програми



9

Рисунок В.9 – Слайд 9

Функціонування програми

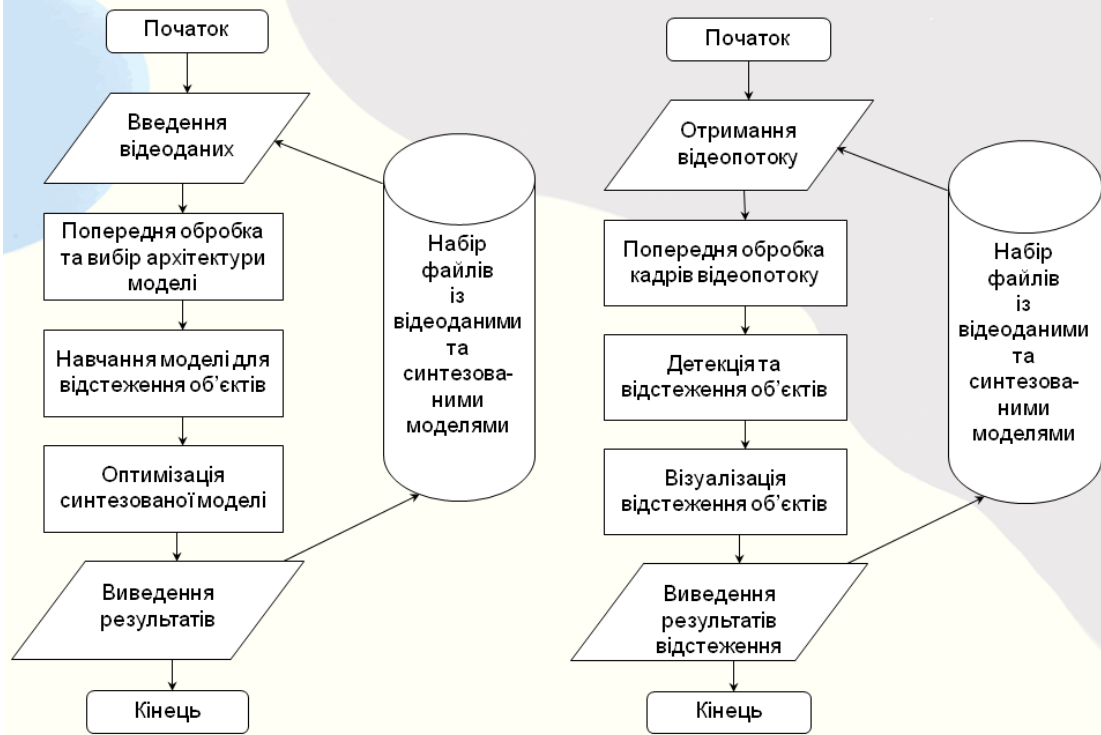


Рисунок В.10 – Слайд 10

Робота з програмою. Форма синтезу та оптимізації нейромережевих моделей

Відстеження об'єктів

Синтез та оптимізація моделей для відстеження об'єктів на відео

Дані для синтезу моделі:

11

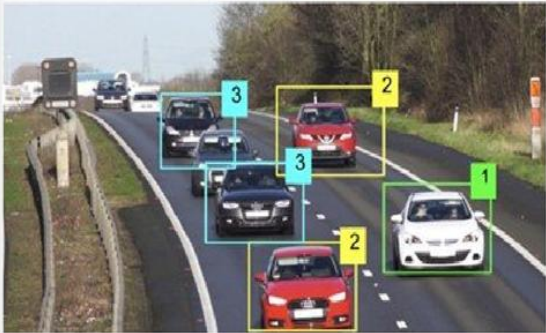
Рисунок В.11 – Слайд 11

Робота з програмою. Форма відстеження об'єктів на відеоданих

Відстеження об'єктів

Відстеження об'єктів за синтезованою моделлю

Дані для відстеження:



12

Рисунок В.12 – Слайд 12

Висновки

В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено процес розробки програмного забезпечення для відстеження об'єктів.

Сформульовано функціональні вимоги до програмного забезпечення для відстеження об'єктів. Для реалізації програмного забезпечення для відстеження об'єктів обрано мову програмування C++ та середовище розробки Qt.

Новизна роботи полягає у тому, що розроблено метод спрощення синтезованих згорткових нейромережових моделей для відстеження об'єктів на відео, який включає етапи прунингу, квантування, передачі знань та архітектурних оптимізацій.

Практична цінність результатів роботи полягає у тому, що розроблено програмне забезпечення для відстеження об'єктів.

Розроблене програмне забезпечення є зручним для використання та виконує всі необхідні функції.

Усі завдання дипломної кваліфікаційної роботи магістра повністю виконано.

13