ДОДАТОК А

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ

StrikePlagiarism.com

Дата звіту 6/17/2025
Дата редагування ---

Звіт не був оцінений

Звіт подібності

метадані

Назва організації
Kharkiv National University of Radio Electronics
Заголовок
2025_М_ПІ_ІПЗм-23-1_Уткін_Ю_Є_скорочений
Автор Науковий керівник / Експерт
Уткін Юрій ЄвгеновичЄвген Кардаш
підрозділ
каф. ПІ

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

0.93% 0.93% КП 1

1.11% 1.11% КЦ

25	14878	111661
Довжина фрази для коефіцієнта подібності 2	Кількість слів	Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв	ß	1
Інтервали	A⟶	0
Мікропробіли	·	0
Білі знаки	ß	0
Парафрази (SmartMarks)	a	10

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://openarchive.nure.ua/server/api/core/bitstreams/4757476c-cf71-4a80-9572-6fbad4335a23/content	15 0.10 %
2	https://github.com/WongKinYiu/ScaledYOLOv4/blob/yolov4-large/detect.py	14 0.09 %
3	https://openarchive.nure.ua/server/api/core/bitstreams/4757476c-cf71-4a80-9572-6fbad4335a23/content	13 0.09 %
4	МКР ТРСА-21 Гірченко І. [redacted] 11/18/2024 National University "Lviv Politechnika" (NULP2)	11 0.07 %

ДОДАТОК Б

Слайди презентації

Дослідження методів рухової активності людини за допомогою дронів

ЮРІЙ УТКІН, ПЗПІ-23-1

Науковий керівник: проф. НАТАЛЯ БІЛОУС
20 червня 2025

1.

Мета роботи

- Об’єкт дослідження – методи детекції та донаведення дрону та необхідні аппаратні ресурси для цього.
- Предмет дослідження – методи та алгоритми детекції, класифікації руху об’єктів, трекінг об’єктів та алгоритм створення команд на embaded модулі.
- Мета роботи – дослідження методів детекції та класифікації руху об’єктів у реальному часі, трекінг об’єкту, аналіз їх ефективності та практичне створення алгоритму для створення команд для embaded.
- Методи досліджень – емпіричний аналіз, моделювання, порівняння, розробка та тестування.

2.

Проблематика

- Швидкість розвитку аппаратних прикладних систем типу дронів – не встигають за швидким розвитком технологій ШИ.
- Виклики сьогодення –технологічна війна;
- Неясність в доцільності інтеграцій моделей ШІ в компактні аппаратні платформи;
- Проблеми з результативнісью використання FPV дронів

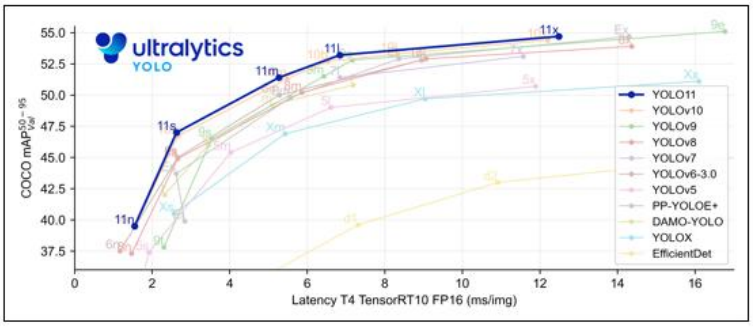
Постановка задачі

- аналіз існуючих методів виявлення та класифікації руху об’єктів;
- дослідження алгоритмів пошуку й відстеження об’єктів у відеопотоці;
- розробка прототипу інтегрованої системи для трекінгу об’єкту;
- тестування прототипу на основі апаратної платформи orange pi 5;
- аналіз результатів використання мінімальних ресурсів апаратного забезпечення.

Вибір технологій розробки

- DeepSORT
- PyCharm
- YOLOv8
- OpenCV
- YOLOv11

Вибір технологій детекції



Незважаючи на переваги новіших моделей – вибір тим не менш впав на мінімально необхідну відправну точку дослідження у вигляді YOLOv8.

Методика проведення дослідження(додаткові бібліотеки)

OpenCV (opencv-python) – бібліотека для роботи з зображеннями та відео, яка використовується для читання відеоданих, попередньої обробки кадрів та візуалізації результатів роботи YOLO.

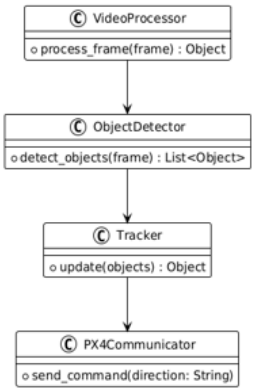
Deep SORT Realtime – бібліотека для трекінгу об’єктів у відеопотоці.

FilterPy – бібліотека, що реалізує алгоритм фільтра Калмана, який використовується для точнішого трекінгу й прогнозування руху об'єктів на відео.

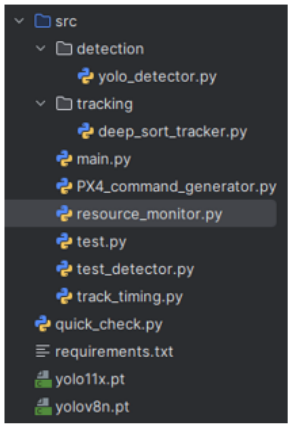
SciPy – бібліотека для наукових та інженерних розрахунків, яка підтримує допоміжні математичні операції, зокрема обчислення, які використовуються при обробці сигналів та відео.

psutil – бібліотека для моніторингу системних ресурсів, що застосовується для відстеження споживання процесорного часу.

Архітектура системи

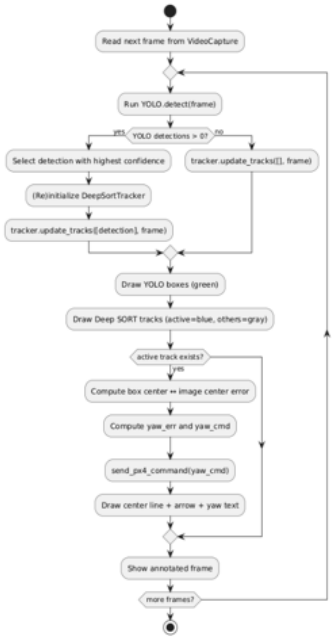


Архітектура системи у контексті технологічного стеку



Файлова структура додатку

Алгоритм пайплайну детекції та трекінгу



Алгоритм донаведення дрону

Формула керуючого сигналу виглядає як P-регулятор із врахуванням Δdx:

```
yaw_err = dx_px / frame_width * HFOV
delta_err = yaw_err - prev_yaw_err
yaw_cmd = kp * yaw_err + kd * delta_err
```

yaw_err – помилка зміщення об’єкта по горизонталі

HFOV – горизонтальне поле зору камери

kp і kd – налаштовані коефіцієнти

prev_yaw_err – попередня помилка

Якщо детекція на кадрі відсутня, але в попередньому кадрі об’єкт лежав ближче до краю зображення (наприклад, |dx_px| > 0.4·frame_width), вважаємо, що він вийшов із поля зору, і віддаємо команду «зависнути та обертатися на місці на 360°». Якщо ж об’єкт зник, перебуваючи близько до центру (| dx_px| ≤ 0.2·frame_width), віддаємо команду «зависнути на місці» без обертання.

Проведення експеременту

- спочатку було обрано та завантажено відеоджерела для тестування (записи з камери дрону);
- після чого протестовано скрипт, який для кожного відео здійснює кадр за кадром детекцію, працює паралельно із deep sort-трекер;
- одночасно фіксує час інференсу, кадрову статистику (загальну кількість кадрів і кадрів із детекцією) і апаратне навантаження (cpu, gpu, пам'ять);
- також зібрано ключові метрики продуктивності (fps, середній час детекції, відсоток кадрів із виявленими об'єктами, тривалість трекінгу при втраті об'єкта тощо);
- зібрані дані було оброблено та візуалізовано в графіках і таблицях, після чого проведено порівняльний аналіз результатів для тестової системи.

Вхідні дані

Для тестування було обрано відео з лінії зіткнення, на якому відсутні жорстокі кадри, показано слідкування за людиною яка переміщується у складних для детекції умовах, поміж дерев та ракурс задіяний при зйомці спочатку знаходиться збоку а потім слідкує майже чітко за обєктом

Кадри з тестових відео

Вхідні дані

Для відео з низькою якістю зображення ми свідомо обрали досить «м'які» налаштування YOLO,DeepSORT:

Нижчий поріг довіри --conf=0.2.

Помірний поріг NMS (Non-Maximum Suppression) --iou=0.45. (Intersection over Union)

```
class DeepSortTracker:  2 usages
    def __init__(
        self,
        max_iou_distance: float     = 0.7,
        max_age: int                = 30,
        n_init: int                 = 1,
        max_cosine_distance: float = 0.5,
        nms_max_overlap: float      = 1.0,
        half: bool                  = True,
        use_appearance: bool        = True
    ):
```

Трекінг:

- max_iou_distance = 0.7. Ми знизили його з типових ~0.9 до 0.7, щоб сильніше орієнтуватися на узгоджений рух (якщо дві рамки суттєво не накладаються, ми їх уже не «прив'язуємо»), але при цьому не втрачати трек, якщо об'єкт трохи змістився;
- max_age = 30.
- n_init = 1.
- max_cosine_distance = 0.5.
- nms_max_overlap = 1.0.
- half = True.
- use_appearance = True.

Аналіз результатів

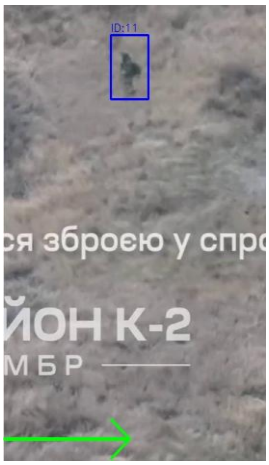
Елементи інтерфейсу системи

Відпрацювання детекції + трекінг

- Рамка детекції

- Рамка трекінгу

- Запропонований вектор руху дрона



Утримання трекінгом

Аналіз результатів

```
Кількість кадрів              : 638
FPS (від CAP_PROP_FPS)         : 30.00
Тривалість відео (секунд)      : 22.20
Загальний час детекції (сек)   : 6.380
Середній час/кадр детекції     : 0.01000 сек
Кадрів із виявленими об'єктами: 88 (13.8% від усіх кадрів)
```

Метричні показники YOLO 8

Оброблено 638 кадрів за загалом 6.38 с чистого часу детекції – це в середньому по 0.010 с (10 мс) на кадр.

При такій швидкості детекція виконується приблизно на 100 кадрів/сек (1 000 мс / 10 мс), що значно перевищує заявлені 30 FPS відео. Це означає, що детектор легко працює в реальному часі із запасом по потужності.

Також видно що об'єкт було знайдено лише в 88 кадрах з 638, тобто в 13.8 % випадків. Враховуючи якість відео та не знаходження обєкуту в полі зору камери певний час – це середній показник.

Аналіз результатів

```
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.84 param3=1 param4=1 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.83 param3=1 param4=1 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.85 param3=1 param4=1 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.84 param3=1 param4=1 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.84 param3=1 param4=1 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.83 param3=1 param4=1 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.83 param3=1 param4=1 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_CONDITION_YAW param1=0 param2=+2.82 param3=1 param4=1 param5=0 param6=0 param7=0
[PX4_CMD] MAV_CMD_DO_CHANGE_SPEED param1=1 param2=5.00 param3=-1 param4=0 param5=0 param6=0 param7=0
```

Команди корегування донаведення

- MAV_CMD_DO_CHANGE_SPEED — ідентифікатор команди зміни швидкості;
- param1=1 — тип швидкості: 0 = airspeed, 1 = ground speed, 2 = climb speed, 3 = descent speed;
- param2=5.00 — значення швидкості в м/с (тут 5 м/с);
- param3=-1 — throttle (процент подачі палива): –1 залишає поточне значення без зміни;

param4=0, param5=0, param6=0, param7=0 – зарезервовані місця для додаткових параметрів (не використовуються, тому 0).

Аналіз результатів

```
=== Resource Usage Summary ===
CPU time  – user: 116.52s, system: 24.83s, total: 141.34s
RAM peak  – 749.2 MiB
```

Результати виводу апаратного навантаження

На Orange Pi 5 є два класи ядер: чотири ARM Cortex-A76@2.4 GHz і чотири ARM Cortex-A55@1.8 GHz. Щоб порівняти їх із ядрами Zen 4, використовуємо орієнтовні пікові DMIPS-навантаження:

Cortex-A76 ≈ 3 DMIPS/MHz → 2 400 MHz × 3 ≈ 7 200 DMIPS на ядро

Cortex-A55 ≈ 2 DMIPS/MHz → 1 800 MHz × 2 ≈ 3 600 DMIPS на ядро

Zen 4 на 4.7 GHz ≈ 3.11 DMIPS/MHz → 4 700 MHz × 3.11 ≈ 14 617 DMIPS на ядро. Тоді відносна одноядерна потужність A76 ≈ 7 200/14 617 ≈ 0.49, A55 ≈ 3 600/14 617 ≈ 0.25, у той час як ядро Ryzen = 1.00.

Отож для нашого алгоритму, який на Ryzen виконувався за 20 с, Orange Pi 5 із цією архітектурою потребуватиме близько 45 с.

Висновки

- проведено аналіз існуючих методів виявлення руху об’єктів;
- Yolov 8 та DeepSORT обрано для детекції та трекінгу людини на кадрі;
- система оптимізована для роботи в умових обмежених обчислювальних ресурсів;
- впроваджено модуль донаведення дрона на основі руху знайденого об’єкта;
- платформу orange pi 5 виявлено як достатню для відстеження руху у реальному часі.

ДОДАТОК В

Апробація результатів роботи

Дослідження методів рухової активності людини за допомогою дронів

Уткін Ю. Є.
Науковий керівник – Професор Білоус Н. В.
Харківський національний університет радіоелектроніки
61166, Харків, просп. Науки, 14, каф. інформатики, тел. (057) 702-13-35
e-mail: yurii.utkin@nure.ua; nataliya.bilous@nure.ua

Актуальність дослідження визначається необхідністю створення високоефективних систем для аналізу динамічних об’єктів у складних умовах, характерних для військових операцій. Точне й своєчасне розпізнавання рухів об’єктів, таких як люди чи техніка, дозволяє знизити ризики, пов’язані з помилковими оцінками, та підвищити ефективність управління. Військові конфлікти підкреслюють важливість розвитку технологій, які дають змогу дронам працювати в реальному часі, забезпечуючи стратегічну перевагу. Використання машинного навчання, зокрема глибоких нейронних мереж і алгоритмів на кшталт YOLO або SSD[1], відкриває нові можливості для точного й швидкого аналізу руху в динамічних і небезпечних умовах.

Мета дослідження полягає у дослідженні методів детекції та класифікації руху об’єктів у реальному часі, аналізі їх ефективності та розробці інтегрованої системи для обробки відеопотоку з дронів у реальному часі.

Задача дослідження — оцінити, наскільки добре методи машинного навчання можуть ідентифікувати пошкоджені об'єкти на основі даних, зібраних дронами. Між точками дослідження планується розробити точно налаштовану систему, що працює в режимі реального часу на об'єктах комунальної техніки. «Особлива увага приділяється оцінці існуючих методів виявлення та сортуванню об'єктів, при цьому особлива увага приділяється їх налаштуванню з урахуванням можливостей розуміння дронів». Важливим елементом є розширення функцій апаратного прискорення для підвищення ефективності системи, а також методи зміни обсягу оптичних флуктуацій до порогу розпізнавання. Результати цього дослідження допоможуть створити прості та зрозумілі рішення для перегляду фільмів і подібних відео.

Процедури моніторингу включають в себе низку етапів, спрямованих на ретельне розслідування і створення ефективної системи ідентифікації предметів і класифікації літальних апаратів від дронів. «Попередній етап включає в себе збір, вивчення наукових статей, технічних записів, доступного коду, а також моніторинг алгоритмів пошуку і категоризації для ідентифікації об'єктів і оцінки сводів з використанням BlazePose, OpenPose, MediaPipe і YOLO». Наразі для визначення точності, швидкості та ефективності управління безпілотними літальними апаратами, включаючи розрахунки потужності та рівноваги, використовуються

численні життєздатні інструменти штучного інтелекту, їх інтеграція або розвиток відомих методів, а також емпіричні випробування на різних наборах даних.

Наступна дія складється з об'єднання вибраних методологій в єдину структуру для негайного розпізнавання і категоризації організацій та їх активів. «Надалі роботизоване середовище обґрунтовується на мікроконтролерах, включаючи Orange Pi 5+, Raspberry Pi 4 і RockPro, використовуючи інтеграцію обладнання NPU для оцінки ефективності використання ресурсів, продуктивності та енергоспоживання».

Заключний етап включає тестування готового прототипу в реальних умовах роботи на дроні, зокрема оцінку стабільності роботи, точності алгоритмів, швидкості реакції та коректності класифікації рухів у складних погодних умовах. У фіналі проводиться аналіз і порівняння результатів роботи системи з іншими відомими підходами або альтернативними реалізаціями для визначення її конкурентних переваг.

У нашому дослідженні YOLO (You Only Look Once) було обрано як основний метод ідентифікації шахраїв через його швидкість, точність і винахідливість в обробці, що є критично важливим для моніторингу дронів. YOLO полегшує розпізнавання об'єктів у режимі реального часу, що надзвичайно важливо для дронів з відеопотоком, оскільки на динамічні когнітивні процеси можуть впливати зміни освітлення, перешкоди та неспокійне навколишнє середовище. Порівняння з використанням різних підходів, таких як Accelerated R-CNN та Single Shot MultiBox Detector, показали швидкість та ефективність YOLO, що робить його придатним для розгортання на мініатюрних обчислювальних платформах.

Наступним кроком стало створення методу сортування для оцінки різних розпізнаних об'єктів. Для вищезгаданої мети ми обрали модель, засновану на навчанні, здатну групувати такі види діяльності, як біг підтюпцем, водіння автомобіля та малювання ліній, що потім розпізнається системою YOLO. Це дозволяє створити просту суміш для негайного розпізнавання та організації ефективних функцій.

Ми розробили модель, що складається з етапів виявлення, категоризації та анал.ізу мислення. Прототип був створений з використанням відомого набору даних аерофотозйомки (AS), збірки зображень та кінематографічних зображень, отриманих за допомогою повітряних пристроїв, пілотованих різними когнітивними об'єктами. Це допомагає їм адаптуватися до різної кількості світла, змін погоди та інших чинників, які можуть вплинути на те, наскільки добре вони розпізнають предмети.

Прототип пройшов перевірку на ПК з одним модулем, а також додатково вивчена установка Nvidia Jetson, яка є зразковою для цих завдань завдяки своїм характеристикам і енергозбереженню. Оцінка показала, що модель працює компетентно в реальному часі, з помітною точністю та мінімальним споживанням енергії, тим самим перевіряючи алгоритми, вибрані для роботи дрона.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Bilous, N., Malko, V., Moshenskyi, N. (2024). Search and Detection of People in the Water Using YOLO Architectures: A Comparative Analysis from YOLOv3 to YOLOv8. In: Szewczyk, R., Zieliński, C., Kaliczyńska, M., Bučinskas, V. (eds) Automation 2024: Advances in Automation, Robotics and Measurement Techniques. AUTOMATION 2024. Lecture Notes in Networks and Systems, vol 1219. Springer, Cham. https://doi.org/10.1007/978-3-031-78266-4_21
2. Bilous N, Malko V, Frohme M, Nechyporenko A. Comparison of CNN-Based Architectures for Detection of Different Object Classes. AI. 2024; 5(4):2300-2320. https://doi.org/10.3390/ai5040113