ISSN 2786-5371 print
ISSN 2786-538X online
Технології та інжиніринг, № 3(20), 2024

https://doi.org/10.30857/2786-5371.2024.3.6

УДК 004.932.2

ПОСВІСТАК В. С., МІРОШНИЧЕНКО Д. В.
Київський національний університет технологій та дизайну, Україна

АНАЛІЗ ПРОБЛЕМ ВБУДОВАНИХ АЛГОРИТМІВ ВІДСТЕЖЕННЯ ОБ'ЄКТІВ OPENCV ПРИ ОБРОБЦІ КАДРІВ ІЗ БПЛА

Мета. *Дослідження та порівняння ефективності алгоритмів відстеження об'єктів OpenCV при обробці кадрів із типовими проблемами зображень, зроблених з камер дронів. Проведення експериментів для визначення переваг та недоліків, тенденцій до збоїв, проблем із швидкодією.*

Методика. *Для експериментів записані відео у симуляторі польотів на дроні для семи типових проблемних ситуацій. Коректну поведінку (фундаментальну істину) визначено шляхом ручного задання обмежувальної рамки для кожного кадру з допомогою Python-скрипту. Визначено типові проблемні ситуації: зміна кута камери, зміна позиції об'єкта на кадрі, наявність ідентичних об'єктів, рух об'єкта, спотворення зображення, поява перепони перед об'єктом, зміна масштабування об'єкта. Розроблено та застосовано Python-скрипти для автоматизації вимірювання чотирьох критеріїв ефективності роботи алгоритмів: індекс співпадіння обмежувальних рамок, відстань між центрами рамок, показник успішності, витрата часу на кожний кадр.*

Результати. *Досліджено ефективність роботи алгоритмів Boosting, CSRT, KCF, MedianFlow, MIL, MOSSE, TLD при обробці кадрів із проблемними ситуаціями, типовими для зображень із дронів. Проведено аналіз схильності до збоїв, подібностей роботи різних алгоритмів, проблем із швидкодією. Результати експериментів представлені у вигляді таблиці чисел, графіків та візуалізації роботи кожного алгоритму.*

Наукова новизна. *За результатами дослідження визначено тенденції алгоритмів до некоректної роботи при обробці кадрів, що містять проблеми, типові для зображень, зроблених із камери дрона.*

Практична значимість. *Результати можуть бути використані для пріоритетизації використання або комбінування окремих алгоритмів залежно від сфери застосування.*

Ключові слова: *відстеження об'єктів; визначення об'єктів; дронові технології; обробка зображень; OpenCV.*

Вступ. Відстеження об'єктів є однією із фундаментальних задач комп'ютерного зору, стоячи поруч із виявленням та визначенням об'єктів. Алгоритми відстеження об'єктів широко застосовуються у повсякденних задачах: контроль дотримання правил дорожнього руху, спортивна аналітика, доповнена реальність, відстеження пожеж тощо. Задачі відстеження об'єктів досліджуються вже десятки років, багато алгоритмів було створено достатньо давно, та виникають нові, більш досконалі алгоритми, проте поточний стан розвитку даної сфери далекий від ідеального.

Існують різні фактори, що можуть негативно впливати на ефективність алгоритмів, такі як погане освітлення, нестабільний фон, розмите зображення тощо [1]. Створення універсального алгоритму, який би працював однаково ефективно у різних умовах є майже неможливою задачею [2, 3].

Із розвитком дронових технологій алгоритми відстеження об'єктів набувають неабиякої актуальності, оскільки можуть застосовуватись у широкому спектрі задач – від простого підсвічування об'єктів на кадрі до комплексних програмних систем для автономного керування дроном. Через специфіку сфери застосування присутній набір викликів, що ускладнюють обробку зображень. Залежно від конкретного обладнання, може варіюватись якість зображення, присутність чи відсутність стабілізації відеозйомки.

Камери можуть бути чутливими до поганого освітлення, їх розмір обмежений, а передача даних здійснюється за допомогою радіозв'язку, що схильний до впливу зовнішніх

факторів на якість зображення (радіо- та фізичні перешкоди, погана погода тощо) [4]. Залежно від сфери застосування може бути присутня тенденція до оптимізації та зниження витрат, що також впливає на якість камери, радіопередавача, контролера, а в результаті і кінцевих зображень.

OpenCV – бібліотека з відкритим кодом, що містить у собі набір функцій та алгоритмів для комп'ютерного зору, обробки зображень та застосування машинного навчання. Вона широко використовується у робототехніці для задач із аналізом, обробкою зображень, відеоаналітикою, доповненою реальністю та для досліджень у галузі комп'ютерного зору. Про популярність даної бібліотеки свідчить понад 47 тисяч користувачів та приблизно 18 мільйонів скачувань [5]. OpenCV підтримує широкий спектр мов програмування, серед яких є C++, Python, Java тощо, а також містить у собі інструменти для використання переваг апаратного прискорення та багатоядерних систем.

Постановка завдання. Метою даного дослідження є проведення експериментів із алгоритмами відстеження об'єктів OpenCV при обробці кадрів із типовими проблемами зображень, зроблених з камер дронів (див. табл. 1), а також визначення тенденцій до збоїв, проблем із швидкодією та подібностей різних алгоритмів.

Таблиця 1

Проблемні ситуації

Абревіатура	Проблема	Опис
ANGL_CH	зміна кута камери	Під час польоту кут огляду камери відносно об'єкта може змінюватися, що призводить до зміни перспективи на зображенні
FST_MOT	стрімка зміна позиції об'єкта на кадрі	Об'єкти на зображенні можуть швидко змінювати своє положення через швидкість руху дрона або об'єкта, різкої зміни направлення польоту тощо
ID_OBJ	наявність ідентичних об'єктів	У випадку наявності декількох ідентичних об'єктів у кадрі можуть виникати проблеми у розрізненні між ними
MOV_OBJ	рух об'єкта	При русі об'єкта може змінюватись кут огляду, позиція на зображенні, його масштабування тощо
NOISE	спотворення зображення	Нестабільний радіосигнал або радіоперешкоди можуть призводити до спотворення зображень та появи білого шуму, що ускладнює розпізнавання відстежуваних об'єктів, особливо у комбінації із камерою невисокої якості
OBST	поява перепони перед об'єктом	Під час польоту можуть з'являтися перешкоди перед об'єктом, що призводять до часткового або повного закриття об'єкта на зображенні
SCALE	зміна масштабування об'єкта	Розмір об'єктів може змінюватись при стрімкому наближенні чи віддаленні дрона, що ускладнює їх розпізнавання

Результати дослідження будуть корисні при виборі алгоритмів до використання та їх комбінація, залежно від потенційних проблемних ситуацій. Очікується, що визначення переваг та недоліків алгоритмів допоможе оптимізувати витрату часу на обробку зображень у реальному часі та підвищити точність відстеження об'єктів. Це відкриє можливості для розробки більш стабільних систем, здатних адаптуватися до динамічних змін у зовнішньому середовищі.

Досліджені проблемні ситуації. Зображення з камер дронів можуть містити типові проблемні ситуації через особливості умов зйомки та характеристики подібних систем, що

можуть ускладнювати відстеження об'єктів. Було визначено поширені проблемні ситуації, кожній із яких було присвоєно відповідну абревіатуру, яка буде використовуватись в подальшому у цій роботі (див. табл. 1).

Відеозаписи із проблемними ситуаціями було зроблено у симуляторі FPV-дронів "Uncrashed" [6] і збережено у вигляді коротких відеофайлів (від 1,4 до 3 секунд) із частотою кадрів 30 на секунду. Даний симулятор містить різні карти, де можна відтворювати польоти на дроні у вільному режимі. Під час запису можуть бути присутні інші об'єкти на задньому фоні, включаючи рухомі, такі умови найбільш наближені до реальних. Було визначено окремі критерії для оцінки ефективності алгоритмів та для їх подальшого порівняння: індекс співпадіння істинної та фактичної обмежувальної рамки, відстань між центрами рамок, витрата часу на кадр, а також показник успішності. Проведено порівняння роботи алгоритмів в однакових умовах та продемонстровано їх тенденції до некоректної поведінки у проблемних ситуаціях.

Методи дослідження. Як було зазначено в аналогічних роботах, проведення експериментів із ефективністю алгоритмів відстеження об'єктів може бути досить складним завданням через різні фактори, такі як освітлення, зміна куту, нестабільність камери, спотворення та розмиття зображення тощо [3]. Вибір правильної метрики оцінювання та експериментальних даних необхідний для забезпечення того, щоб жодна важлива інформація про алгоритм не була втрачена. У випадку відстеження об'єктів критеріями якісної роботи алгоритму є точність обмежувальної рамки, показник успішності та швидкодія.

Досліджено алгоритми CSRT [7], KCF [8–10], MIL [11], Boosting [12], MedianFlow [13], MOSSE [14], TLD [15]. Експерименти були проведені з opencv-contrib-python 4.9.0.80 із Python інтерпретатором версії 3.11, на ОС Ubuntu 23.04 із версією ядра Linux 6.2.0-39-generic. Апаратні характеристики: AMD Ryzen 5 5600X, AMD Radeon RX 6700 XT, 32 GB RAM.

Визначення фундаментальної істини (ground truth). Для правильного вимірювання коректності роботи алгоритмів у проблемних ситуаціях основним викликом є наявність фундаментальної істини (ground truth). У даному дослідженні визначення коректних рамок відбувалось за допомогою скрипту для ручного визначення коректної обмежувальної рамки (або її відсутності) для об'єкта у кожному кадрі відео. Під час ручного задання прямокутника була можливість автоматично перевикористати рамку із попереднього кадру, яка завжди була відображена на поточному кадрі, таким чином мінімізуючи похибку через ручне введення (див. рис. 1). Задання прямокутника відбувалось якомога ближче до фізичних контурів об'єкта, у такий спосіб прямокутник доволі точно відображає позицію об'єкта, проте можливі незначні похибки. Дані зберігались у локальний файл у вигляді списку елементів формату (x1, y1, w, h), де (x1, y1) – координати початку рамки (ліворуч зверху), а (w, h) це її довжина та ширина. Загалом таким чином було оброблено 591 кадр. В подальшому істинна рамка буде зображена у вигляді зеленого прямокутника, а рамка, розрахована алгоритмами – у вигляді синього.

Точність обмежувальної рамки. Для вимірювання точності обмежувальної рамки проведено 2 окремих види експериментів, де були визначені показники BBMR (bounding box match ratio) – індекс перетину рамок, та CTCD – відстань між центрами рамок у пікселях.

Bounding box match ratio. Для визначення BBMR за основу було використано формулу коефіцієнта Жаккара – область (кількість пікселів) перетину рамок, поділена на область їх об'єднання:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|},$$

де A та B – прогнозована та очікувана обмежувальні рамки [16]. Для ініціалізації кожного алгоритму використовувався перший кадр із збереженого списку ground truth. У

випадках появи перепони перед об'єктом чи втрати об'єкта із кадру (див. рис. 2), використання оригінальної формули Жаккара не є можливим при відсутності обох рамок. В цьому випадку і перетин і об'єднання дорівнюють 0, але відсутність обох рамок є коректним результатом. Індекс BBMR враховує ситуацію, коли обидві рамки порожні, та у цьому випадку дорівнює 1 – див. рис. 3.

```
ground_truth_box = cv.selectROI(frame, showCrosshair=False)
if ground_truth_box == (0, 0, 0, 0):
    print(f"{TextColor.GREEN}No bounding box was selected. \n"
          f"Press <y> to confirm that it's expected to have no bounding box\n"
          f"Press <r> to re-use the previous bounding box\n"
          f"Press any other key to re-select the bounding box{TextColor.RESET}")
    key = cv.waitKey(0)
    if key == ord('y'):
        prev_bbox = ground_truth_box
        break
    elif key == ord('r'):
        ground_truth_box = prev_bbox
        break
else:
    prev_bbox = ground_truth_box
    break
```

Рис. 1. **Ручне введення обмежувальної рамки з допомогою функції selectROI у OpenCV**

Рис. 2. **Присутність та відсутність рамки, залежно від перепони (експеримент OBST)**

```
def bbox_match_ratio(bbox_a, bbox_b):
    if is_empty(bbox_a) or is_empty(bbox_b):
        if is_empty(bbox_a) and is_empty(bbox_b):
            return 1
        return 0

    return intersection_area(bbox_a, bbox_b) / union_area(bbox_a, bbox_b)
```

Рис. 3. **Визначення значення індексу співпадіння рамок (BBMR)**

Center to center distance. Оскільки у випадку відсутності їх перетину, коефіцієнт Жаккара однаково дорівнює 0, незалежно від того, як далеко поточна рамка знаходиться від очікуваної, необхідно ввести в експеримент ще один критерій. CTCD – дистанція між центрами обмежувальних рамок у пікселях (зображено на кадрі у вигляді червоної лінії – див. рис. 4).

Також можлива ситуація, коли центр рамки співпадає з очікуваним, але сама рамка має більший або менший за очікуваний розмір, зрівнюючи показник ефективності зміщеної та

збільшеної рамки (див. рис. 5). Завдяки вимірюванню CCTD жодна важлива інформація про експеримент не буде втрачена, про необхідність чого було зазначено вище.

Рис. 4. **Демонстрація необхідності визначення відстані між центрами рамок, у обох випадках BBMR = 0 (експеримент FST_MOT – алгоритм MIL)**

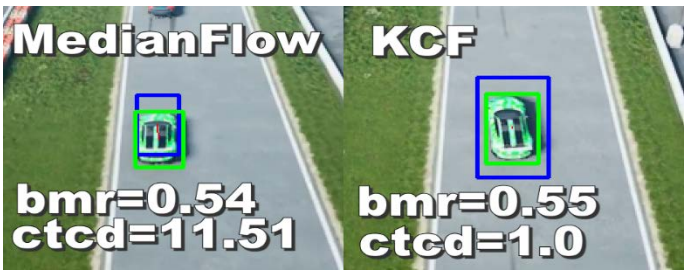

Рис. 5. **Демонстрація необхідності визначення відстані між центрами рамок, приблизно однаковий BBMR (експеримент MOV_OBJ)**

Виняткова ситуація – відсутність однієї із обмежувальних рамок, у цьому випадку індекс CTCD буде розрахований як максимально можлива дистанції – між точками (x=0, y=0) та (x=<ширина кадру>, y=<висота кадру>), тобто по діагоналі кадру. У випадку відсутності обох рамок дистанція буде дорівнювати 0, оскільки це очікуваний результат (а чим менша дистанція, тим вища коректність знайденої рамки).

Показник успішності. Для визначення показника успішності можливо використовувати вже розрахований BBMR. Для кожного кадру успішне визначення є таким, де BBMR > 0.5 [3]. Також для окремих проблемних випадків буде врахований показник BBMR > 0.25.

Швидкодія. Швидкодія алгоритмів при обробці кожного кадру була розрахована за допомогою бібліотеки timeit [17]. Її використання дозволяє уникнути поширених помилок із використанням немонотонних системних годинників, має зручний інтерфейс для вимірювання ізольованої частини коду та конфігурування кількості ітерацій вимірювання для уникнення похибки. У даному експерименті було використано 100 ітерацій (див. рис. 6).

```
# number_of_runs == 100 by default
time = timeit.timeit(lambda: tracker.update(processed_frame), number=number_of_runs)
time_in_ms = (time * 1000) / number_of_runs
```

Рис. 6. **Розрахунок витрати часу на кожний кадр**

Результати досліджень. Витрата часу та BBMR разом із CTCD були розраховані окремими python-скриптами почергово для кожного алгоритму, який зберігав всі дані для експорту локально. Для візуалізації результатів використовувалися графічні інструменти Statistics Kingdom [18, 19]. Із усіма результатами, набором експериментальних зразків, візуалізацією та скриптами розрахунку можна ознайомитись у GitHub репозиторії даної роботи [20].

Зміна кута камери. Експеримент проводився із облітанням автомобіля по колу на 90 градусів. Усі алгоритми впоралися із утриманням рамки на об’єкті, окрім KCF, який ближче до кінця відео втратив фокус (див. рис. 7). При зміні кута на фронтальний відносно автомобіля, прогнозована рамка зберігала свою минулу форму у вигляді горизонтального прямокутника, таким чином зменшуючи коефіцієнт Жаккара ближче до кінця відео.

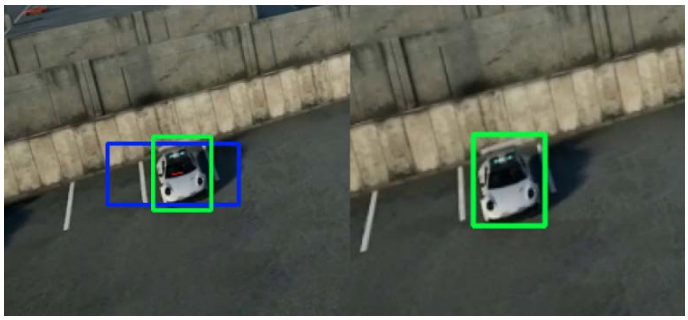

Рис. 7. **Втрата фокусу алгоритмом KCF ближче до кінця відео, демонстрація збереження форми прямокутника (ліворуч)**

MOSSE, MedianFlow – лідери за швидкодією, проте присутня тенденція прив’язки до обраної області, не дивлячись на те що при обертанні навколо об’єкта його позиція на кадрі змінюється (див. рис. 8). У випадку MedianFlow також зменшується прогнозована рамка. Це візуалізується гіршими CTCD результатами для даних алгоритмів (див. рис. 10).

Рис. 8. **Прив’язка MOSSE до пікселів на фоні (аналогічно із MOSSE)**

TLD мав тенденцію «стрибків» прямокутника під час обертання, проте фокусувався на об’єкті. Boosting, CSRT, KCF, MIL точно відстежували об’єкт, центруючи його у початковій формі – див. рис. 9.

Рис. 9. **Центрування об'єкта алгоритмом CSRT (аналогічно із Boosting, KCF, MIL)**

Стрімка зміна позиції об’єкта на кадрі. TLD найкраще впорався із фокусом на об’єкті, що різко змінює своє положення на кадрі, і майже весь час був сфокусований на правильній позиції. У MedianFlow були тенденції до фокусування на початку відео (коли швидкість не була такою стрімкою), але при подальшому русі та розмитті кадру він втрачав фокус. Інші алгоритмі прив'язувались до фіксованої позиції на кадрі і рухались разом із центром камери дрона при повороті, або втрачали фокус – див. рис. 11.

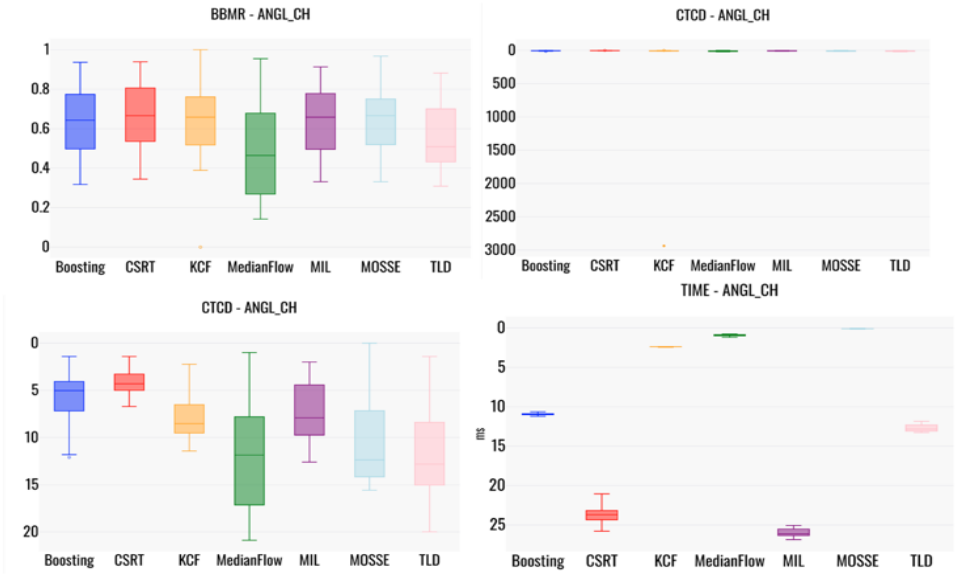

Рис. 10. **Показники ефективності алгоритмів при обробці кадрів із зміною кута камери (BBMR, CTCD, час обробки кадру)**

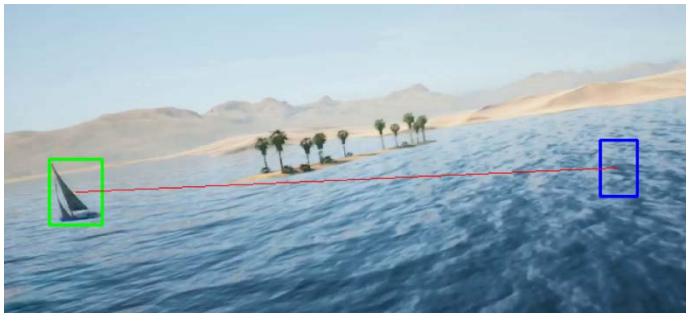

Рис. 11. **Втрата фокусу при стрімкому повороті камери**

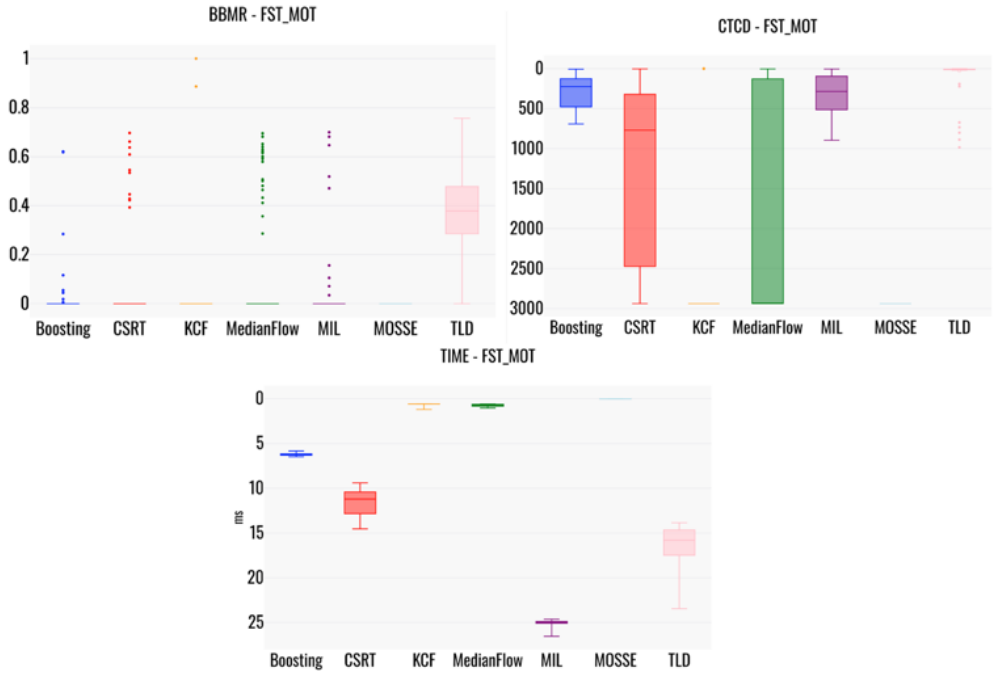

Рис. 12. **Показники ефективності алгоритмів при обробці кадрів із стрімкою зміною позиції об'єкта на кадрі (BBMR, CTCD, час обробки кадру)**

Наявність ідентичних об'єктів. Boosting, CSRT, KCF, MedianFlow відмінно тримали очікувану позицію, незважаючи на ідентичні об'єкти поряд (див. рис. 13). В середньому лідерами швидкодії були KCF (0.5 мс/кадр), MedianFlow (0.9 мс/кадр) та Boosting (4.4 мс/кадр). CSRT в середньому витрачав 23.3 мс/кадр.

Рис. 13. **Вдале фокусування алгоритмів Boosting, CSRT, KCF, MedianFlow**

MIL, TLD мали тенденцію фокусуватись на об'єктах, ідентичних до очікуваного, MIL тільки вертикалі, а TLD у випадковому режимі – див. рис. 14.

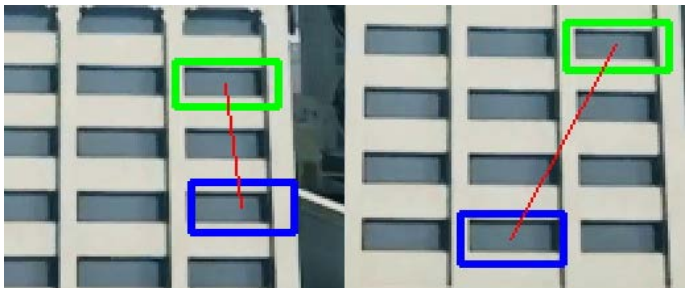

Рис. 14. **Фокусування на ідентичних об'єктах, MIL (ліворуч) та TLD (праворуч)**

MOSSE був поза кадром майже весь час, створивши лише один прямокутник із BBMR=0.07.

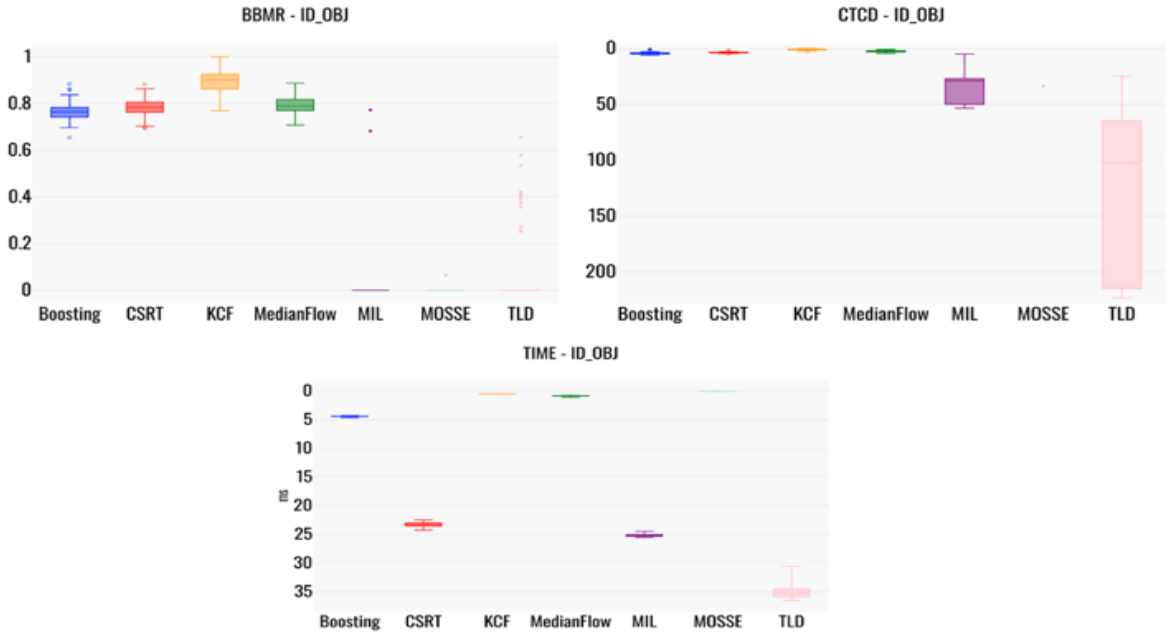

Рис. 15. **Показники ефективності алгоритмів при обробці кадрів із наявністю ідентичних об'єктів (BBMR, CTCD, час обробки кадру)**

Об'єкт, що рухається. Відео для цього експерименту (камера, що переслідує авто, що рухається) є певною комбінацією незначної зміни кута камери, розміру об'єкта та зміни його позиції. Авто добре виділяється на фоні дороги. Алгоритми показали високу ефективність у фокусуванні на очікуваному об'єкті. Boosting, CSRT, KCF, MOSSE зберігали більший розмір прямокутника по мірі віддалення авто, але точно центрували об'єкт – див. рис. 16.

Рис. 16. **Поведінка Boosting алгоритму (схожа на CSRT, KCF, MOSSE)**

MedianFlow правильно адаптував розмір рамки, але було зміщення догори (див. рис. 17). Загалом проявив себе найкраще із показником успішності 100%, для всіх кадрів BBMR був вищий за 0.5 (див. табл. 2).

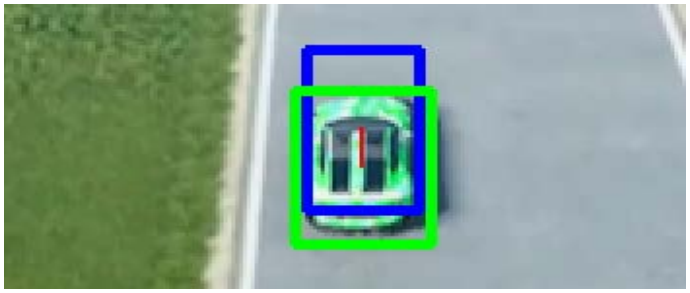

Рис. 17. **Зміщення догори (MedianFlow)**

Для MIL та TLD були характерні незначні горизонтально-вертикальні зміщення, проте TLD виявився точнішим за рахунок того, що адаптував розмір рамки по мірі зменшення об'єкта – див. рис. 18.

Рис. 18. **Зміщення рамки та адаптація розміру (MIL ліворуч, TLD праворуч)**

CSRT показав найгірші результати у витраті часу. Після середини вимірювання (~40й кадр, враховуючи 100 ітерацій вимірювання витрати часу для кожного з них) витрата часу зросла більше ніж вдвічі, ближче до кінця відео також було стрімке зростання до пікових

182 мс/кадр (див. рис. 19). Експеримент був повторений декілька разів, викликаючи подібну поведінку. Достеменна причина невідома, але на це може впливати конкретна реалізація алгоритму у бібліотеці OpenCV.

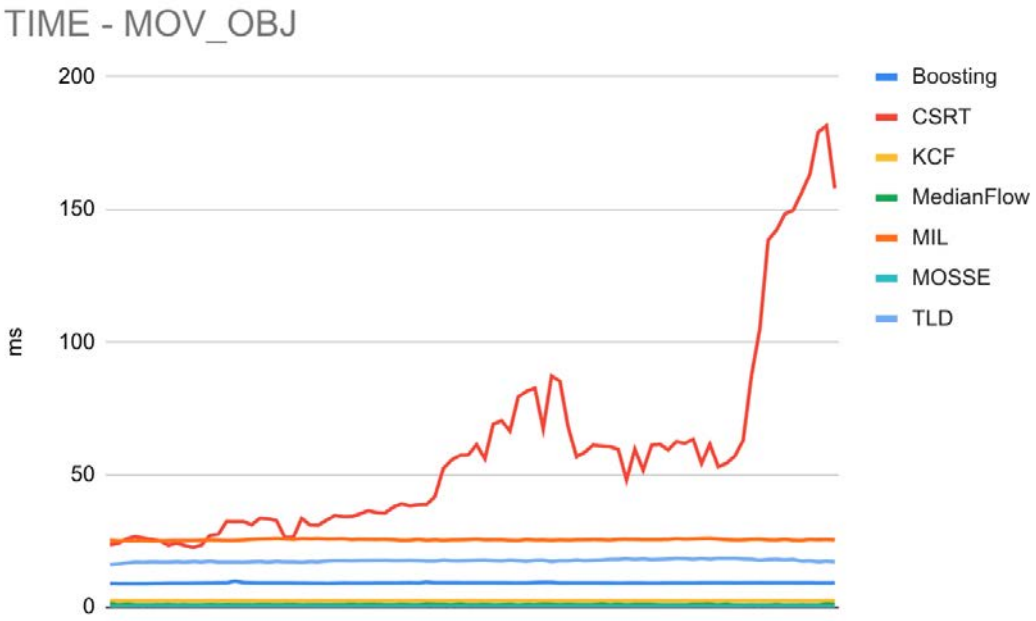

Рис. 19. **Витрата часу CSRT впродовж відео (кожен кадр вимірювався 100 разів)**

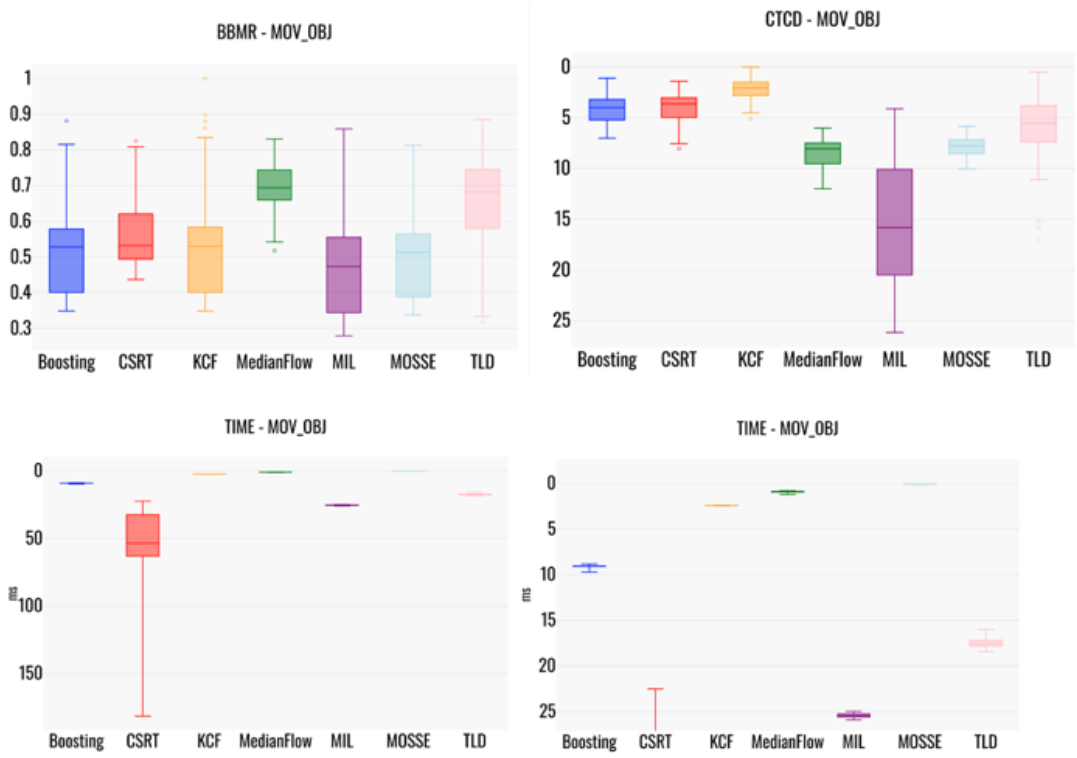

Рис. 20. **Показники ефективності алгоритмів при обробці кадрів із об'єктом, що рухається (BBMR, CTCD, час обробки кадру)**

Спотворення зображення. Boosting, CSRT, MIL показали хороший результат у фокусуванні на об’єкті при наявності білого шуму та періодичного зміщення об’єкта в сторону – див. рис. 21.

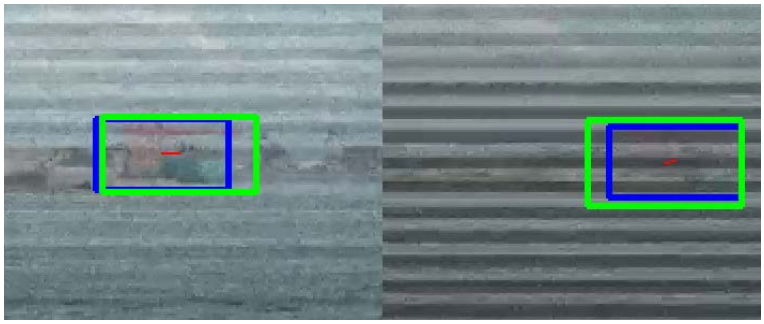

Рис. 21. **Збереження фокусу на зображенні при білому шумі та зміщенні в сторону (Boosting)**

KCF майже весь час був поза кадром, проте іноді коректно фокусувався на об’єкті після зниження рівню спотворення. MedianFlow частково був поза кадром, частково фокусувався на неправильній області у випадковому порядку, ні разу не сфокусувався правильно. MOSSE був поза кадром весь час. TLD зберігав фокус при відсутності спотворення, а під час спотворення фокусувався на неправильній області у випадковому порядку. Також мав тенденцію збільшувати витрату часу на кадр приблизно у 2–3 рази під час спотворення зображення (див. рис. 22).

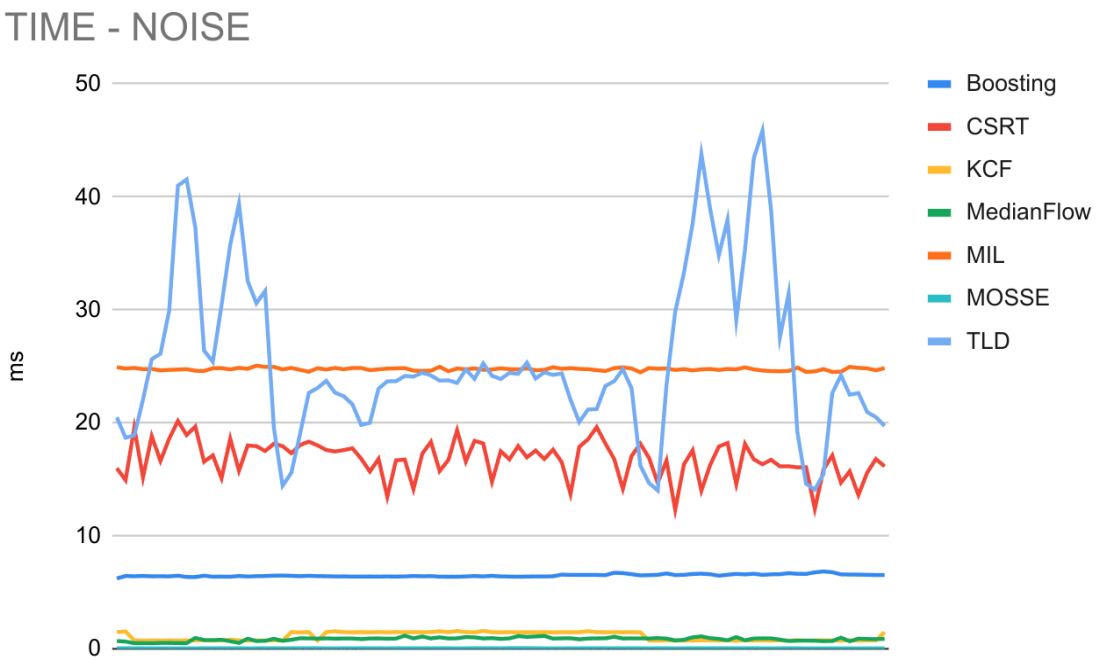

Рис. 22. **Збільшення витрат часу TLD під час спотворення зображення**

TLD складається із декількох етапів: відстеження, навчання, визначення об’єктів. Якщо вхідні дані мають низьку якість або велику кількість шуму, алгоритму може бути складно відстежувти об'єкти, що призводить до додаткових обчислень для коригування результатів, а відповідно і збільшення витрат часу – див. рис. 23.

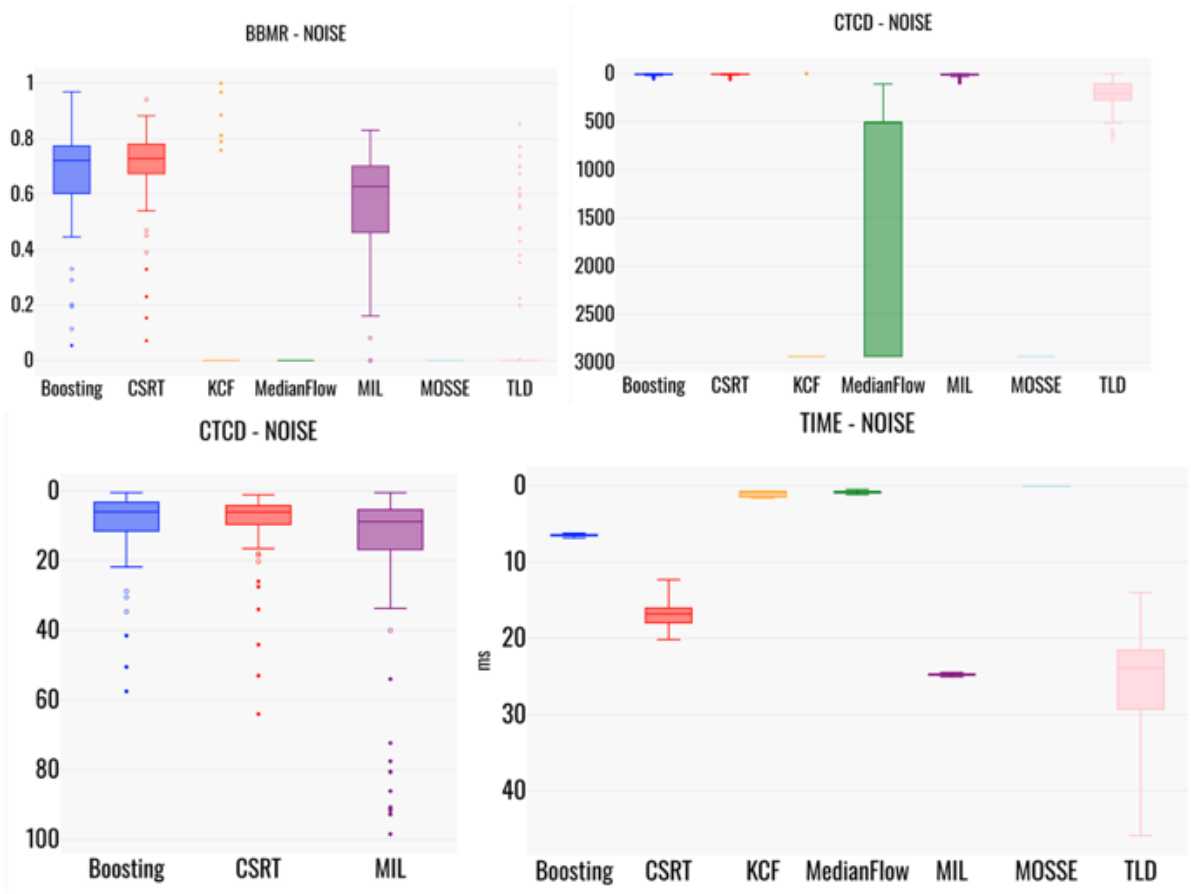

Рис. 23. **Показники ефективності алгоритмів при обробці кадрів із спотворенням зображення (BBMR, CTCD, час обробки кадру)**

Перепона перед об'єктом. У якості відео для експериментів було відтворено ситуацію коли при русі дрона на деякий час автомобіль перекривається стовпом, а потім знову з'являється у кадрі (див. рис. 24). Поведінку алгоритів при даному експерименті можна поділити на 3 категорії:

1. Точний початковий фокус, подальший фокус на перепоні після втрати об'єкта (CSRT, MIL, Boosting, MedianFlow).

2. Точний початковий фокус, коректна втрата фокусу після втрати об'єкта, неможливість сфокусуватись на об'єкті після його появи у кадрі (KCF, Mosse).

3. Точний початковий фокус, фокусування на випадкових об'єктах під час перепони у кадрі, відновлення фокусу на об'єкті після його появи у кадрі (TLD).

CSRT, MIL, Boosting не змогли коректно зреагувати на появу перепони і до кінця відео зберігали фокус на стовпі. Алгоритми KCF та Mosse коректно фокусувались на авто до появи стовпа, після чого фокусувались на довільних об'єктах у різних позиціях кадру до кінця відео (інші авто, вікна, паркани тощо). TLD також мав тенденцію довільного фокусування під час наявності перепони, але зміг відновити коректний фокус на авто після її зникнення.

Зміна масштабу об'єкта. Експериментальні дані: відео, де камера дрону стрімко наближається до авто, в кінці відео авто заповнює весь кадр. CSRT, MedianFlow вдало масштабували рамку (див. рис. 25). CSRT залишався в кадрі, зупинивши масштабування рамки ближче до кінця відео, зберігаючи однаковий розмір. MedianFlow втратив фокус, коли об'єкт майже заповнив екран.

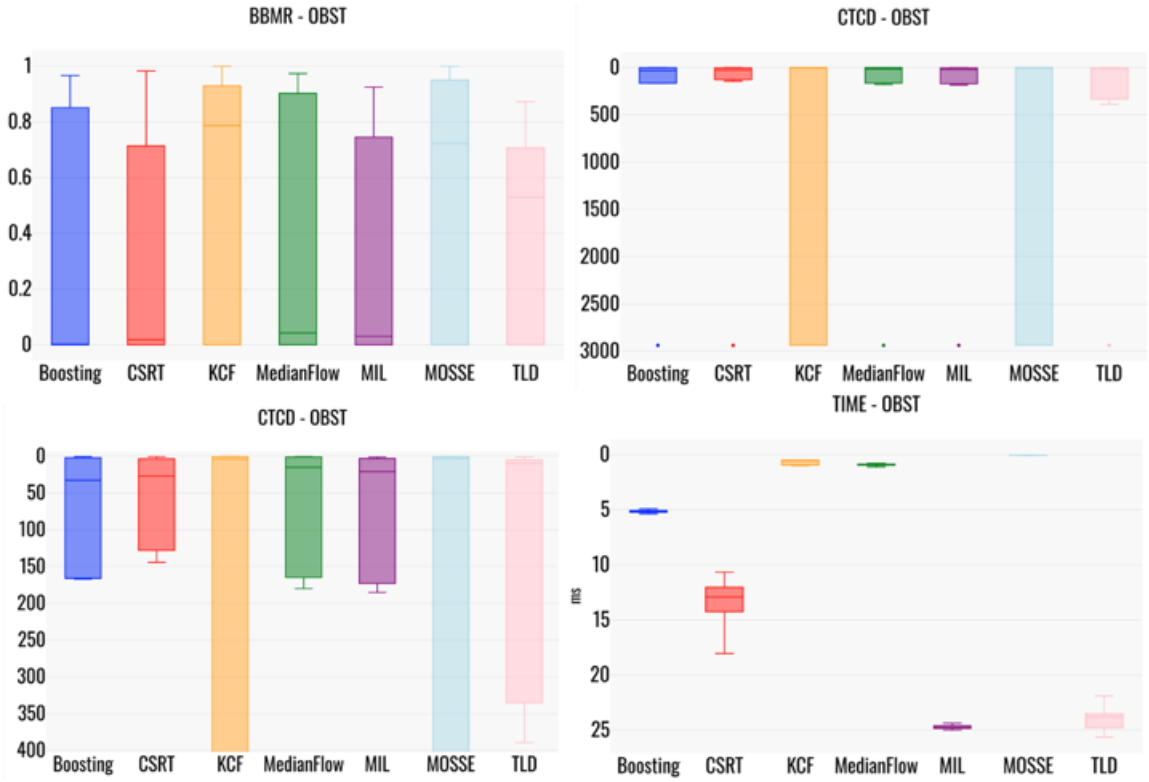

Рис. 24. **Показники ефективності алгоритмів при обробці кадрів із появою перепони перед об'єктом (BBMR, CTCD, час обробки кадру)**

Рис. 25. **Масштабування розміру рамки при наближенні (CSRT, приблизна ситуація в MedianFlow)**

TLD масштабував рамку, проте під час збільшення були незначні стрибки в різні сторони (див. рис. 26). Загалом вдало тримав приблизну позицію і фокус до самого кінця експериментального відео.

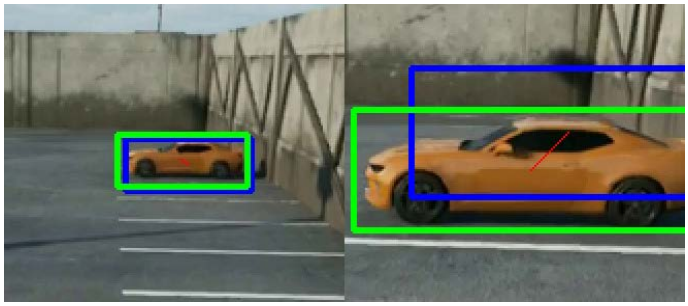

Рис. 26. **Стрибки під час збільшення (TLD)**

Boosting, MIL, MOSSE не масштабували розмір рамки. MOSSE був приблизно в центрі, але втратив фокус в кінці. Boosting тримав фокус, але змістився ліворуч. Для MIL були характерні стрибки всередині очікуваної рамки. KCF втратив фокус на ранньому етапі під час збільшення об'єкта. На рис. 27 зображено статистику роботи алгоритмів.

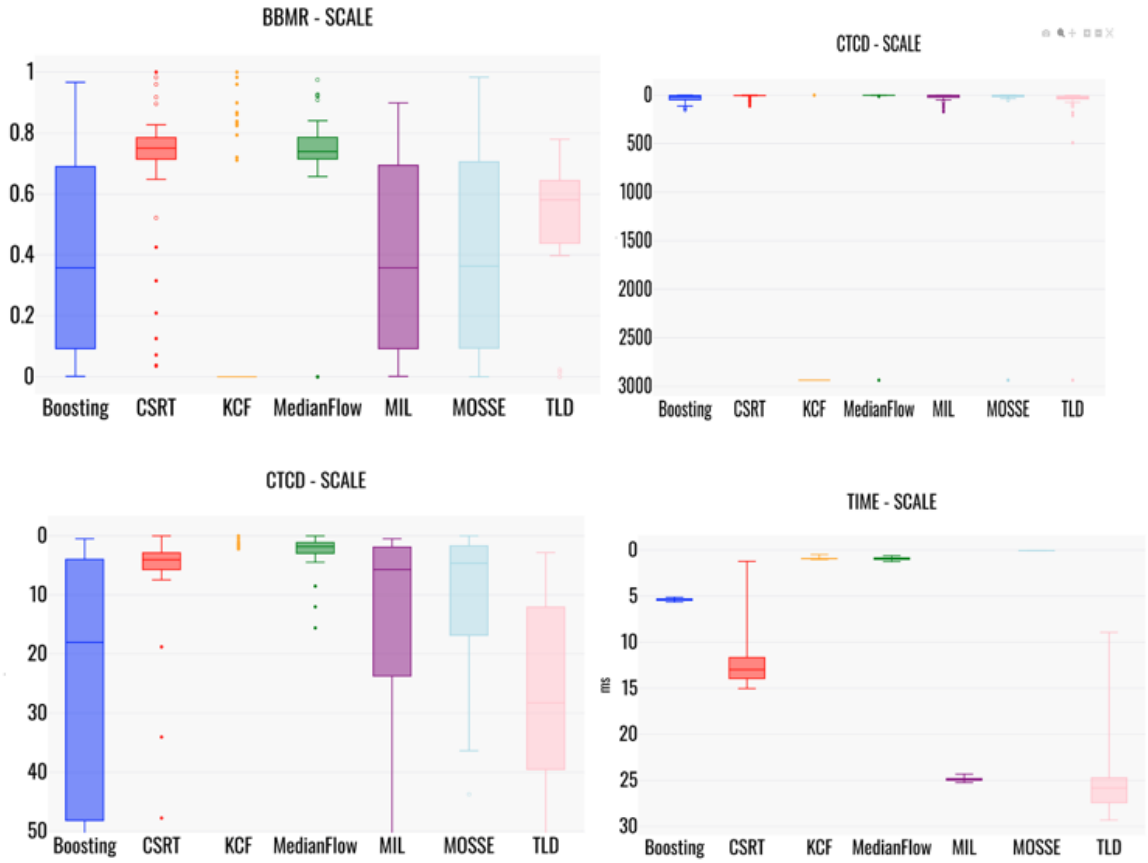

Рис. 27. **Показники ефективності алгоритмів при обробці кадрів із зміною масштабування об'єкта (BBMR, CTCD, час обробки кадру)**

Висновки. У даній роботі була досліджена поведінка 7 алгоритмів відстеження об'єктів OpenCV із поширеними проблемними ситуаціями на кадрах із БПЛА. Результати показали, що алгоритми мають свої переваги та недоліки, та їх ефективність може значно коливатися залежно від умов. Під час досліджень було виявлено, що певні алгоритми проявляють схожу поведінку у певних сценаріях. Алгоритми Boosting та CSRT показували схожу продуктивність у всіх експериментах крім останнього – зміни масштабу об'єкта. Також CSRT витрачав значно більше часу на обробку кадрів (у 2–5 разі більше). MIL, MOSSE, TLD погано справляються з відстеження, коли на кадрі присутні ідентичні об'єкти. TLD – єдиний алгоритм, який здатен відносно вдало фокусуватись на об'єкті при стрімкій зміні позиції та розмитті зображення, за умови, що відстежуваний об'єкт добре контрастує із фоном. Boosting, CSRT, MIL непогано утримують позицію об'єкта та його зміщення при спотворенні зображення, за умови, якщо в оригінальному кадрі без спотворень позиція на кадрі залишається відносно сталою. Із усіма результатами, набором експериментальних зразків, візуалізацією та скриптами розрахунку можна ознайомитись у GitHub репозиторії даної роботи [20].

Таблиця 2

Відсоток кадрів із BBMR >0.5 при обробці кадрів із проблемними ситуаціями

BBMR > 0.5 (%)	Boosting	CSRT	KCF	MedianFlow	MIL	MOSSE	TLD
ANGL_CH	74	78	76	47	74	76	51
FST_MOT	2	7	2	17	5	0	22
ID_OBJ	100	100	100	100	2	0	4
MOV_OBJ	60	68	61	100	41	56	82
NOISE	90	92	10	0	73	0	13
OBST	39	34	56	39	37	56	54
SCALE	40	88	23	87	40	40	62

Таблиця 3

Відсоток кадрів із BBMR >0.25 при обробці кадрів із проблемними ситуаціями

BBMR > 0.25 (%)	Boosting	CSRT	KCF	MedianFlow	MIL	MOSSE	TLD
ANGL_CH	100	100	90	79	100	100	100
FST_MOT	4	12	2	24	6	0	77
ID_OBJ	100	100	100	100	2	0	19
MOV_OBJ	100	100	100	100	100	100	100
NOISE	96	97	10	0	85	0	19
OBST	44	41	56	46	44	56	63
SCALE	58	92	23	87	58	58	95

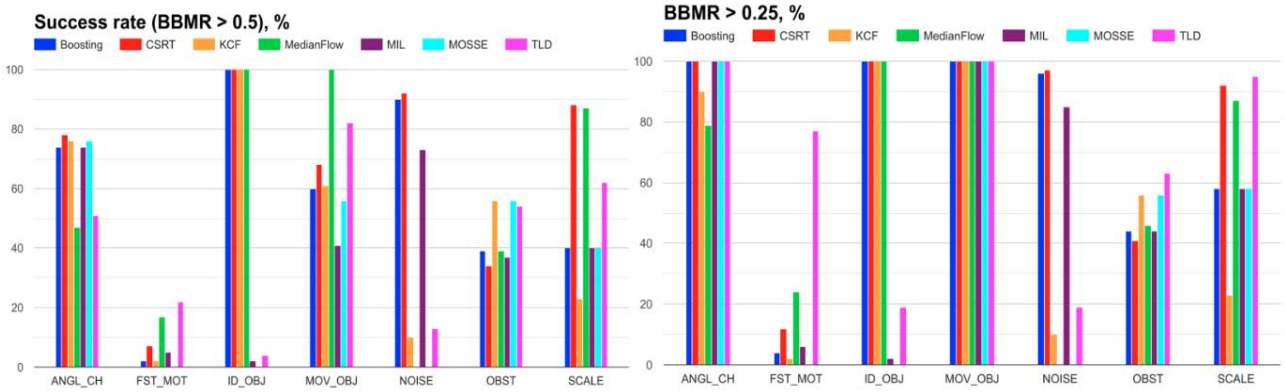

Рис. 28. **Показники успішності алгоритмів (BBMR > 0.5 (ліворуч)) та BBMR > 0.25 (праворуч) для усіх експериментів**

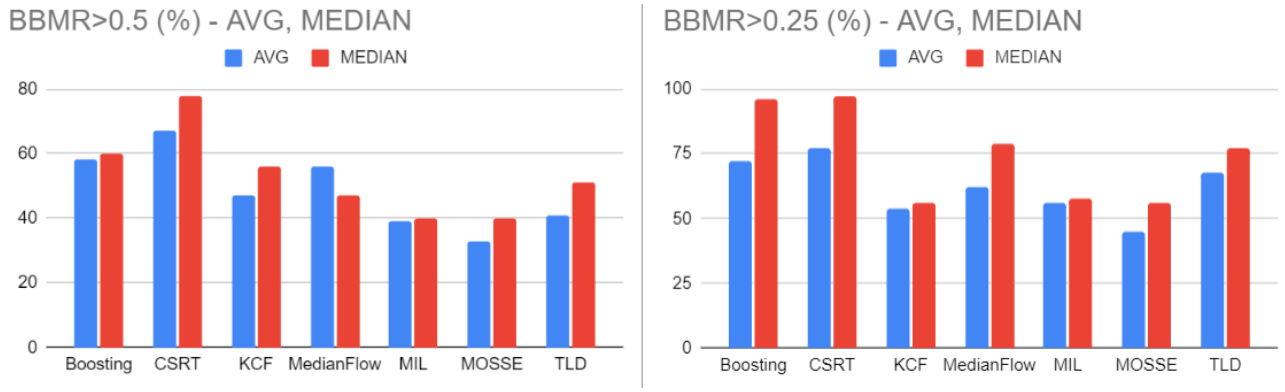

Рис. 29. **Середнє число та медіана показників BBMR > 0.5 (ліворуч) і BBMR > 0.25 (праворуч)**

Швидкодія CSRT та TLD більше залежала від зміни умов, ніж у інших алгоритмах, про це свідчить більший діапазон значень на графіках витрат часу. Перспективним покращенням швидкодії було б використання наявних оптимізацій у OpenCV, таких як багатоядерна паралелізація етапів роботи алгоритму та використання обчислювальної спроможності відеокарт [21].

Для досягнення кращих результатів у реальних умовах необхідно проводити дослідження можливості комбінації алгоритмів, їх реініціалізації в режимі реального часу, обробки зображень перед застосуванням алгоритмів. На рис. 28 зображено статистику успішності алгоритмів для різних експериментів – відсоток значень із індексом BBMR, більшим за 0.5 та 0.25. На рис. 29 зображено загальну статистику успішності алгоритмів для всіх експериментів.

References

1. Wu, Y., Lim, J., Yang, M.-H. (2013). Online Object Tracking: A Benchmark. *2013 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2013, P. 2411–2418. DOI: 10.1109/cvpr.2013.312.

2. Li, D., Liang, B., Zhang, W. (2014). Real-time moving vehicle detection, tracking, and counting system implemented with OpenCV. *2014 4th IEEE International Conference on Information Science and Technology*, Apr. 2014, P. 631–643. DOI: 10.1109/icist.2014.6920557.

3. Wu, Y., Lim, J., Yang, M.-H. (2015). Object Tracking Benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Sep. 01, 2015, Vol. 37, No. 9, P. 1834–1848. doi: 10.1109/tpami.2014.2388226.

4. RF Wireless Vendors and Resources | RF Wireless World. 10 Advantages And Disadvantages Of Radio Frequency (RF) Waves And Its Uses. URL: https://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-RF.html.

5. OpenCV. About OpenCV. URL: https://opencv.org/about.html.

6. Rochette, L. Uncrashed: FPV Drone Simulator on Steam. URL: https://store.steampowered.com/app/1682970/Uncrashed__FPV_Drone_Simulator/

7. Lukežič, A., Vojíř, T., Čehovin Zajc, L., Matas, J., Kristan, M. (2018). Discriminative Correlation Filter Tracker with Channel and Spatial Reliability. *International Journal of Computer Vision*, Vol. 126, No. 7, P. 671–688. DOI: 10.1007/s11263-017-1061-3.

8. Henriques, J. F., Caseiro, R., Martins, P., Batista, J. (2015). High-Speed Tracking with Kernelized Correlation Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Mar. 01, 2015, Vol. 37, No. 3, P. 583–596. DOI: 10.1109/tpami.2014.2345390.

Література

1. Wu Y., Lim J., Yang M.-H. Online Object Tracking: A Benchmark. *2013 IEEE Conference on Computer Vision and Pattern Recognition*. Jun. 2013. P. 2411–2418. DOI: 10.1109/cvpr.2013.312.

2. Li D., Liang B., Zhang W. Real-time moving vehicle detection, tracking, and counting system implemented with OpenCV. *2014 4th IEEE International Conference on Information Science and Technology*. Apr. 2014. P. 631–643. DOI: 10.1109/icist.2014.6920557.

3. Wu Y., Lim J., Yang M.-H. Object Tracking Benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Sep. 01, 2015. Vol. 37, No. 9. P. 1834–1848. DOI: 10.1109/tpami.2014.2388226.

4. 10 Advantages And Disadvantages Of Radio Frequency (RF) Waves And Its Uses. *RF Wireless Vendors and Resources | RF Wireless World*. URL: https://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-RF.html.

5. About OpenCV. *OpenCV*. https://opencv.org/about.html.

6. Rochette L. Uncrashed: FPV Drone Simulator on Steam. URL: https://store.steampowered.com/app/1682970/Uncrashed__FPV_Drone_Simulator/

7. Lukežič A., Vojíř T., Čehovin Zajc L., Matas J., Kristan M. Discriminative Correlation Filter Tracker with Channel and Spatial Reliability. *International Journal of Computer Vision*. Jan. 08, 2018. Vol. 126, No. 7. P. 671–688. DOI: 10.1007/s11263-017-1061-3.

8. Henriques J. F., Caseiro R., Martins P., Batista J. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

9. Danelljan, M., Khan, F. S., Felsberg, M., Van De Weijer, J. (2014). Adaptive Color Attributes for Real-Time Visual Tracking. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2014. DOI: 10.1109/cvpr.2014.143.

10. Henriques, J. F., Caseiro, R., Martins, P., Batista, J. (2012). Exploiting the Circulant Structure of Tracking-by-Detection with Kernels. *Computer Vision – ECCV* 2012. P. 702–715. DOI: 10.1007/978-3-642-33765-9_50.

11. Babenko, B., Yang, M.-H., Belongie, S. (2009). Visual tracking with online Multiple Instance Learning. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009. DOI: 10.1109/cvpr.2009.5206737.

12. Grabner, H., Grabner, M., Bischof, H. (2006). Real-Time Tracking via On-line Boosting. *Procedings of the British Machine Vision Conference 2006*. British Machine Vision Association, 2006. DOI: 10.5244/c.20.6.

13. Kalal, Z., Mikolajczyk, K., Matas, J. (2010). Forward-Backward Error: Automatic Detection of Tracking Failures. *2010 20th International Conference on Pattern Recognition*, IEEE, Aug. 2010. DOI: 10.1109/icpr.2010.675.

14. Bolme, D., Beveridge, J. R., Draper, B. A., Lui, Y. M. (2010). Visual object tracking using adaptive correlation filters. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2010. DOI: 10.1109/cvpr.2010.5539960.

15. Kalal, Z., Mikolajczyk, K., Matas J. (2012). Tracking-Learning-Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Jul. 2012, Vol. 34, No. 7, P. 1409–1422. DOI: 10.1109/tpami.2011.239.

16. Hancock, J. M. (2004). Jaccard Distance (Jaccard Index, Jaccard Similarity Coefficient). *Dictionary of Bioinformatics and Computational Biology*. Wiley, Oct. 15, 2004. DOI: 10.1002/9780471650126.dob0956.

17. timeit – Measure execution time of small code snippets – Python 3.9.0 documentation. *docs.python.org*. URL: https://docs.python.org/3/library/timeit.html.

18. Advanced box and whisker plot maker. *Statistics Kingdom*, 2017. URL: https://www.statskingdom.com/advanced-boxplot-maker.html.

Mar. 01, 2015. Vol. 37, No. 3. P. 583–596. DOI: 10.1109/tpami.2014.2345390.

9. Danelljan M., Khan F. S., Felsberg M., Van De Weijer J. Adaptive Color Attributes for Real-Time Visual Tracking. *2014 IEEE Conference on Computer Vision and Pattern Recognition*. Jun. 2014. DOI: 10.1109/cvpr.2014.143.

10. Henriques J. F., Caseiro R., Martins P., Batista J. Exploiting the Circulant Structure of Tracking-by-Detection with Kernels. *Computer Vision – ECCV 2012*. 2012. P. 702–715. DOI: 10.1007/978-3-642-33765-9_50.

11. Babenko B., Yang M.-H., Belongie S. Visual tracking with online Multiple Instance Learning. *2009 IEEE Conference on Computer Vision and Pattern Recognition*. Jun. 2009. DOI: 10.1109/cvpr.2009.5206737.

12. Grabner H., Grabner M., Bischof H. Real-Time Tracking via On-line Boosting. *Procedings of the British Machine Vision Conference 2006*. British Machine Vision Association, 2006. DOI: 10.5244/c.20.6.

13. Kalal Z., Mikolajczyk K., Matas J. Forward-Backward Error: Automatic Detection of Tracking Failures. *2010 20th International Conference on Pattern Recognition*, IEEE, Aug. 2010. DOI: 10.1109/icpr.2010.675.

14. Bolme D., Beveridge J. R., Draper B. A., Lui Y. M. Visual object tracking using adaptive correlation filters. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* Jun. 2010. DOI: 10.1109/cvpr.2010.5539960.

15. Kalal Z., Mikolajczyk K., Matas J. Tracking-Learning-Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. Jul. 2012. Vol. 34, No. 7. P. 1409–1422. DOI: 10.1109/tpami.2011.239.

16. Hancock J. M. Jaccard Distance (Jaccard Index, Jaccard Similarity Coefficient). *Dictionary of Bioinformatics and Computational Biology*. Wiley, Oct. 15, 2004. DOI: 10.1002/9780471650126.dob0956.

17. timeit – Measure execution time of small code snippets – Python 3.9.0 documentation. *docs.python.org*. URL: https://docs.python.org/3/library/timeit.html.

18. Advanced box and whisker plot maker. *Statistics Kingdom*, 2017. URL: https://www.statskingdom.com/advanced-boxplot-maker.html.

19. Chart maker. *Statistics Kingdom*, 2017. URL: https://www.statskingdom.com/chart-maker.html.

20. OpenCV object trackers analysis for UAV. *GitHub repository*, 2024. URL: https://github.com/valposv/opencv-obj-trackers-analysis-for-uav.

21. Janku, P., Koplik, K., Dulik, T., Szabo, I. (2016). Comparison of tracking algorithms implemented in OpenCV. *MATEC Web of Conferences*, Vol. 76, P. 4. DOI: 10.1051/matecconf/20167604031.

19. Chart maker. *Statistics Kingdom*, 2017. URL: https://www.statskingdom.com/chart-maker.html.

20. OpenCV object trackers analysis for UAV. *GitHub repository*, 2024. URL: https://github.com/valposv/opencv-obj-trackers-analysis-for-uav.

21. Janku P., Koplik K., Dulik T., Szabo I. Comparison of tracking algorithms implemented in OpenCV. *MATEC Web of Conferences*. 2016. Vol. 76. P. 4. DOI: 10.1051/matecconf/20167604031.

POSVISTAK VALERII
Postgraduate Student,
Faculty of Mechatronics and Computer Technologies,
Kyiv National University of Technologies and Design, Ukraine
https://orcid.org/0009-0001-7785-1378
E-mail: valposv@gmail.com

MIROSHNYCHENKO DMYTRO
Postgraduate Student,
Faculty of Mechatronics and Computer Technologies,
Kyiv National University of Technologies and Design, Ukraine
https://orcid.org/0009-0002-0904-1645
E-mail: dmiroshnycheko@gmail.com

POSVISTAK V. S., MIROSHNYCHENKO D. V.
Kyiv National University of Technologies and Design, Ukraine

ANALYSIS OF OPENCV OBJECT TRACKING ALGORITHMS WHEN PROCESSING UAV FRAMES

Purpose. *Investigation and comparison of OpenCV object tracking algorithms effectiveness when processing frames with typical problems for images from UAV cameras. Conducting experiments to determine advantages and disadvantages, failure tendencies, time performance issues.*

Methodology. *Experimental videos were recorded in a FPV-drone flight simulator for seven typical problems. Correct behavior (ground truth) was defined manually using Python scripts with a bounding box input. Typical problems were determined: angle change, object position change, presence of identical objects, moving object, image noise, obstacle in front of an object, object size scaling. Python scripts were developed and used to automate calculation of four effectiveness criteria: bounding box match rate, center to center distance, success rate, time spent per frame.*

Findings. *Investigated effectiveness of Boosting, CSRT, KCF, MedianFlow, MIL, Mosse, TLD algorithms when processing frames with problems typical for images from UAV cameras. Carried out an analysis of failure tendencies, similarities between different algorithms, time performance issues. The experiments results were presented as tables with raw values, charts and human-friendly visualization of each algorithm's work.*

Originality. *According to the results of the research, the tendencies of object tracking algorithms to work incorrectly when processing frames with problems typical for images taken from a UAV camera were determined.*

Practical value. *The results can be used for prioritization of certain object tracking algorithms usage or combination, depending on the field of application.*

Keywords*: object tracking; object detection; drone technologies; UAV; image processing; OpenCV.*