Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра прикладної математики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Математичні моделі та методи трекінгу об'єктів за відеозображенням за допомогою безпілотного літального апарату
(тема)

Виконав:
здобувач _2_ року навчання, групи _САУм-24-1
Григорій РОМАНЕНКО
(Власне ім'я, ПРІЗВИЩЕ)
Спеціальність
124 Системний аналіз
(код і повна назва спеціальності)
Тип програми _освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма
Системний аналіз і управління
(повна назва освітньої програми)
Керівник доц. Валентин ЄСІЛЕВСЬКИЙ
(посада, Власне ім'я, ПРІЗВИЩЕ)

Допускається до захисту

Завідувач кафедри ПМ _____ Максим СИДОРОВ
(підпис) (Власне ім'я, ПРІЗВИЩЕ)

2025 р.

Харківський національний університет радіоелектроніки

Факультет _інформаційно-аналітичних технологій та менеджменту
Кафедра _прикладної математики
Рівень вищої освіти _другий (магістерський)
Спеціальність _124 Системний аналіз
(код і повна назва)
Тип програми _освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма _Системний аналіз і управління
(повна назва)

ЗАТВЕРДЖУЮ:

Завідувач кафедри _____
(підпис)
"_10_" _листопада_ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві _Романенку Григорію Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи _Математичні моделі та методи трекінгу об'єктів за відеозображенням за допомогою безпілотного літального апарату
затверджена наказом по університету від 10 листопада 2025 р. № 1027 Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 18 грудня 2025 р.

3. Вихідні дані до роботи _відеозапис і дані з сенсорів безпілотного літального апарату

4. Перелік питань, що потрібно опрацювати в роботі
1. Системний аналіз предметної області
2. Вибір і обґрунтування методу розв'язання
3. Програмна реалізація
4. Результати обчислювального експерименту
5. Аналіз можливих застосувань

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій

1. Актуальність теми роботи
2. Постановка задачі
3. Системний аналіз предметної області
4. Метод чисельного аналізу
5. Результати обчислювального експерименту

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Підбір та вивчення технічної літератури за темою роботи	10 – 16 листопада 2025 р.	виконано
2	Вибір та обґрунтування методу	17 – 23 листопада 2025 р.	виконано
3	Розробка алгоритму і програми	24 – 30 листопада 2025 р.	виконано
4	Проведення аналітичних досліджень та розрахунків	01 – 07 грудня 2025 р.	виконано
5	Робота над текстом пояснювальної записки	08 – 17 грудня 2025 р.	виконано
6	Представлення роботи на рецензію в ЕК	18 грудня 2025 р.	виконано

Дата видачі завдання _10_ листопада 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Валентин ЄСІЛЕВСЬКИЙ
(підпис) (посада, Власне ім'я, ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка: 49 с., 17 рис., 1 табл., 2 дод., 30 джерел.

ГЕОГРАФІЧНА ЛОКАЛІЗАЦІЯ, КОМП’ЮТЕРНИЙ ЗІР, НЕЙРОННА МЕРЕЖА, СЕНСОРИ БПЛА, ТРЕКІНГ ОБ’ЄКТІВ.

Об’єкт дослідження – алгоритми і моделі нейронних мереж для трекінгу об’єктів.

Мета роботи – дослідження моделей штучного інтелекту для вирішення задачі географічної локалізації об’єктів.

Методи дослідження – математичне моделювання, статистичний аналіз і інтелектуальна обробка даних.

Це дослідження було проведено з метою пошуку алгоритму географічної локалізації об'єктів на відео, знятому з безпілотного літального апарату (БПЛА). Виконано тренування моделі глибокого навчання для відстеження об'єктів. Реалізовано програмне забезпечення для обробки відеозаписів та телеметричних даних. Застосовано метод простого ковзного середнього для згладжування шуму даних датчиків. Актуальність теми зумовлена широким використанням БПЛА у цивільних та військових завданнях. Одним із можливих застосувань є створення карт місцевості.

ABSTRACT

Introductory note: 49 pages, 17 figures, 1 table, 2 appendixes, 30 sources.

COMPUTER VISION, GEOGRAPHICAL LOCALIZATION, NEURAL NETWORK, OBJECT TRACKING, UAV SENSORS.

Object of research – algorithms and neural network models for object tracking.

Purpose of work – research into artificial intelligence models for solving the problem of geographical localization of objects.

Methods of research – mathematical modeling, statistical analysis and data mining.

This research was done to find algorithm for geographical localization of objects on video taken from unmanned aerial vehicle (UAV). Trained deep learning model for object tracking. Implemented software application for processing of video records and telemetry data. Applied simple moving average method for noise smoothing of sensor data. The relevance of the topic is due to the widespread use of UAVs in civil and military tasks. One of the possible applications is the creation of terrain maps.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАК, ОДИНИЦЬ І ТЕРМІНІВ

БПЛА – безпілотний літальний апарат;

SMA – Simple Moving Average;

YOLO – You Only Look Once.

ВСТУП

Актуальність теми. Розробка алгоритмів трекінгу об'єктів для БПЛА має безпосередній вплив на підвищення ефективності військових і цивільних операцій. Одним з можливих застосувань є створення карт місцевості. Трекінг об'єктів дозволяє моніторити переміщення ворожої техніки та особового складу, що є необхідним для тактичного планування. Прикладом цивільного застосування може бути охорона об'єктів та периметрів**,** автоматичний трекінг вторгнень або підозрілої активності на великих територіях.

Мета і завдання кваліфікаційної роботи. Метою кваліфікаційної роботи є дослідження моделей штучного інтелекту для вирішення задачі географічної локалізації об’єктів. Для досягнення поставленої мети необхідно виконати наступні завдання:

– провести огляд і аналіз сучасного стану задачі трекінгу об’єктів;

– виконати постановку задачі трекінгу об’єктів;

– обрати найкращий метод розв’язання задачі трекінгу об’єктів;

– реалізувати алгоритм трекінгу об’єктів;

– створити набір даних для експериментів;

– провести попередню обробку в наборі даних;

– виконати географічну локалізацію знайдених об’єктів;

– провести серію експериментів для порівняльної оцінки розробленого алгоритму.

Об’єктом дослідження є алгоритми і моделі нейронних мереж для трекінгу об’єктів.

Предметом дослідження є математичні чисельні методи комп'ютерного зору для обробки відеозображення з камери БПЛА.

Методи дослідження. У кваліфікаційній роботі використовуються методи математичного моделювання, статистичного аналізу і інтелектуальної обробки даних.

Публікації. Результати, отримані у кваліфікаційній роботі, було представлено на 29-му Міжнародному молодіжному форумі «Радіоелектроніка та молодь у XXI столітті» (м. Харків 16 – 19 квітня 2025 р.) [1] і 4-й Міжнародній науково-практичній конференції «European Studies. Learning and Teaching in the World of Technologies» (Харків, Україна – Клуж-Напока, Румунія 12 листопада 2025 р.) [2].

1 СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ

1.1 Системний аналіз задачі трекінгу об’єктів

Задача трекінгу об’єктів є міждисциплінарною, оскільки охоплює питання комп’ютерного зору, обробки зображень та машинного навчання. Можливі сфери використання: спостереження за транспортом (моніторинг дорожнього руху), аграрний моніторинг (виявлення техніки, тварин), пошуково-рятувальні операції, військові та охоронні застосування.

Основною метою трекінгу об’єктів є визначення місцезнаходження об'єкта в суміжних відеокадрах [3].

На практиці задача трекінгу приймає одну з двох форм: трекінг вже ідентифікованих об’єктів і трекінг невідомих об’єктів, які в багатьох випадках ідентифікуються по характеру руху [4].

Для задачі географічної локалізації важливим є модуль GPS-навігації, необхідний для орієнтування у просторі, визначення розташування БПЛА. У світі існує кілька систем супутникової навігації, а саме: американська – GPS, європейська – Galileo, російська – ГЛОНАСС, китайська – BeiDou. Різниці для користувача практично жодної. Все, що необхідно знати про GPS у ключі навігації: що більше супутників він бачить, то точніше підраховує свою позицію. Існують приймачі, які працюють з однією або кількома системами, описаними вище. Приймачі, що працюють з кількома системами, бачать більшу кількість супутників і менш схильні до GPS-спуфінгу – методу радіоелектронної боротьби (РЕБ). При спуфінгу станція РЕБ глушить сигнали супутників і замінює їх своїми – фальшивими [5].

Основні компоненти систем розпізнавання та відслідковування об'єктів в БПЛА:

– камери та візуальні системи: є основним джерелом вхідних візуальних даних; вони дозволяють дрону бачити навколишнє середовище;

– датчики глибини, такі як LiDAR або глибинні камери, що надають додаткову інформацію про відстані до об'єктів;

– штучний інтелект та машинне навчання: ці методи використовуються для навчання моделей розпізнавання об'єктів; вони можуть виявляти та класифікувати об'єкти на зображеннях;

– алгоритми відслідковування, які дозволяють БПЛА визначати шлях руху та точно відслідковувати рух об'єктів.

Серед технічних обмежень можна виділити: обмежену обчислювальну потужність, похибки в даних з сенсорів.

1.2 Аналіз сценаріїв вирішення задачі трекінгу об’єктів

Існує два основних підходи до вирішення задачі трекінгу об’єктів: традиційні методи комп’ютерного зору і глибокі нейронні мережі.

Традиційні методи комп’ютерного зору:

– оптичний потік (Optical Flow, Lucas-Kanade, Farneback);

– методи на основі кореляції (KCF [6], MOSSE, CSRT).

Переваги: простота реалізації, швидкість, робота в реальному часі.

Недоліки: нестійкість до шумів, обмеження при швидких рухах, втрати при часткових перекриттях або зміні масштабу.

Глибокі нейронні мережі:

– Siamese-based трекери (SiamFC [7], SiamRPN [8], SiamMask [9]);

– детектор + трекер (YOLO + BoT-SORT);

– трансформерні архітектури (TransT [10], TrackFormer [11]).

Переваги: висока точність, адаптація до різних типів об’єктів.

Недоліки: потреба у великих обчислювальних ресурсах, залежність від навчання на великій вибірці.

Технічні обмеження апаратної частини безпосередньо впливають на вибір обчислювальної стратегії:

– бортова обробка (Edge Computing): вимагає використання легковагових та оптимізованих алгоритмів (наприклад, MobileNet-SSD, оптимізовані Siamese-мережі), що мають меншу точність, але забезпечують мінімальну затримку;

– наземна обробка: доступ до потужних GPU/CPU на наземній станції дозволяє застосовувати складні та точні алгоритми (наприклад, Transformer-based Trackers), але вносить затримку через передачу даних, що критично для керування БПЛА.

1.3 Змістовна та формальна постановка задачі

Змістовно задача полягає в тому, щоб, маючи послідовність кадрів відео, знятих БПЛА, ідентифікувати цільовий об’єкт у кожному кадрі та визначати його траєкторію руху.

Метою роботи є розробка та дослідження методів трекінгу об’єктів.

Для досягнення поставленої мети необхідно розв’язати такі підзадачі:

– проаналізувати існуючі алгоритми трекінгу об’єктів;

– визначити найбільш придатний алгоритм;

– створити набір даних для експериментів;

– провести попередню обробку в наборі даних;

– реалізувати алгоритм трекінгу;

– провести експериментальне дослідження ефективності алгоритму трекінгу;

– оцінити точність, швидкодію та стійкість системи до зовнішніх впливів.

Формальна постановка задачі. Нехай маємо послідовність кадрів відео:

$$I = \{I_1, I_2, I_3, ..., I_T\},$$

де I_t – кадр, отриманий у момент часу t;

T – загальна кількість кадрів.

У першому кадрі I_1 відома початкова позиція цільового об’єкта, що задається як прямокутна область:

$$B_1 = (x_1, y_1, w_1, h_1),$$

де x_1, y_1 – координати лівого верхнього кута;

w_1, h_1 – ширина та висота області.

Необхідно для кожного наступного кадру I_t визначити положення об’єкта B_t, таке, що функція подібності між зразком об’єкта O_1 та знайденим фрагментом у кадрі I_t максимізується:

$$B_t = \arg\max_{B \in \Omega} S\left(O_1, I_t(B)\right),$$

де S – функція подібності (наприклад, на основі кореляції, глибоких ознак чи відстані в ознаковому просторі);

Ω – множина всіх можливих областей у кадрі.

1.4 Постановка задач дослідження

Метою кваліфікаційної роботи є дослідження моделей штучного інтелекту для вирішення задачі географічної локалізації об’єктів. Для досягнення поставленої мети необхідно виконати наступні завдання:

– провести огляд і аналіз сучасного стану задачі трекінгу об’єктів;

– виконати постановку задачі трекінгу об’єктів;

– обрати найкращий метод розв’язання задачі трекінгу об’єктів;

– реалізувати алгоритм трекінгу об’єктів;

– створити набір даних для експериментів;

– провести попередню обробку в наборі даних;

– виконати географічну локалізацію знайдених об’єктів;

– провести серію експериментів для порівняльної оцінки розробленого алгоритму.

2 ВИБІР ТА ОБҐРУНТУВАННЯ МЕТОДУ РОЗВ’ЯЗАННЯ

2.1 Алгоритми трекінгу об’єктів

2.1.1 Оптичний потік

Оптичний потік – це закономірність руху об'єктів зображення між двома послідовними кадрами, спричинена рухом об'єкта або камери. Це двовимірне векторне поле, де кожен вектор є вектором зміщення, що показує рух точок від першого кадру до другого.

Оптичний потік працює на основі кількох припущень:

– інтенсивність пікселів об'єкта не змінюється між послідовними кадрами;

– сусідні пікселі здійснюють подібний рух.

Розглянемо піксель $I(x,y,t)$ у першому кадрі. Він рухається на відстань (dx,dy) у наступному кадрі, знятому після часу t. Отже, оскільки ці пікселі однакові, а інтенсивність не змінюється, можна сказати, що:

$$I(x,y,t) = I(x+dx, y+dy, t+dt).$$

Потім візьмемо апроксимацію правої частини рядом Тейлора, виділимо спільні члени та поділимо на dt, щоб отримати наступне рівняння:

$$f_x u + f_y v + f_t = 0,$$

де $f_x = \frac{\partial f}{\partial x}$, $f_y = \frac{\partial f}{\partial y}$;

$u = \frac{\partial x}{\partial t}$, $v = \frac{\partial y}{\partial t}$.

Наведене вище рівняння називається рівнянням оптичного потоку [12]. У ньому ми можемо знайти f_x та f_y, це градієнти зображення. Аналогічно, f_t – це градієнт з часом. Але (u,v) невідомо. Ми не можемо розв'язати це рівняння з двома невідомими змінними. Існує кілька методів для вирішення цієї задачі, один з них – метод Лукаса-Канаде.

Ми вже бачили припущення, що всі сусідні пікселі матимуть подібний рух. Метод Лукаса-Канаде бере ділянку 3x3 навколо точки. Таким чином, всі 9 точок мають однаковий рух. Ми можемо знайти (f_x, f_y, f_t) для цих 9 точок. Отже, тепер наша задача полягає у розв'язанні 9 рівнянь з двома невідомими змінними. Краще рішення отримуємо методом найменших квадратів. Нижче наведено остаточне рішення, яке є задачею з двох рівнянь та двох невідомих, та його розв'язання:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_i f_{x_i}{}^2 & \sum_i f_{x_i} f_{y_i} \\ \sum_i f_{x_i} f_{y_i} & \sum_i f_{y_i}{}^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i f_{x_i} f_{t_i} \\ -\sum_i f_{y_i} f_{t_i} \end{bmatrix}.$$

Отже, з точки зору користувача, ідея проста: ми даємо кілька точок для відстеження, отримуємо вектори оптичного потоку з цих точок. Але знову ж таки, є деякі проблеми. Досі ми мали справу з малими рухами, тому це не працює, коли є великий рух. Щоб вирішити це, ми використовуємо піраміди. Коли ми піднімаємося вгору по піраміді, малі рухи видаляються, а великі рухи стають малими. Тож, застосовуючи метод Лукаса-Канаде, ми отримуємо оптичний потік разом із масштабом. Приклад застосування методу Лукаса-Канаде показано на рисунку 2.1.

Рисунок 2.1 – Трекінг об’єктів за допомогою метода Лукаса-Канаде [13]

Метод Лукаса-Канаде обчислює оптичний потік для розрідженого набору ознак (у нашому прикладі, кутові точки, виявлені за допомогою алгоритму Ши-Томасі [14]). Існує інший метод для знаходження щільного оптичного потоку. Він обчислює оптичний потік для всіх точок у кадрі і базується на алгоритмі Гуннара Фарнебака (рис. 2.2) [15].

Рисунок 2.2 – Трекінг об’єктів за допомогою метода Фарнебака

2.1.2 Детектор + трекер

Алгоритм BoT-SORT [16] (рис. 2.3) являє собою значний прогрес у технології трекінгу об'єктів. Цей трекер базується на алгоритмі ByteTrack [17], водночас впроваджуючи три ключові покращення, які усувають давні обмеження в існуючих методах трекінгу шляхом детекції.

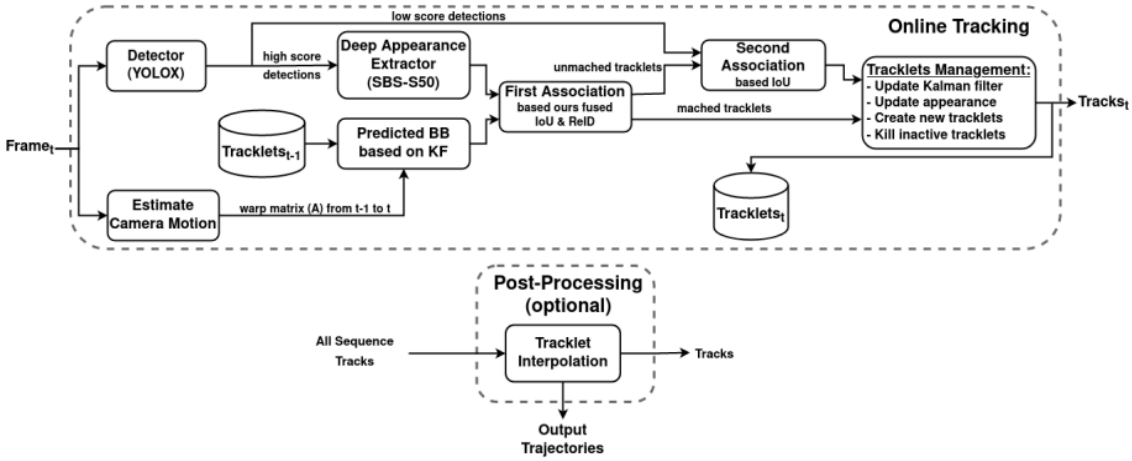


Рисунок 2.3 – Алгоритм роботи трекера BoT-SORT

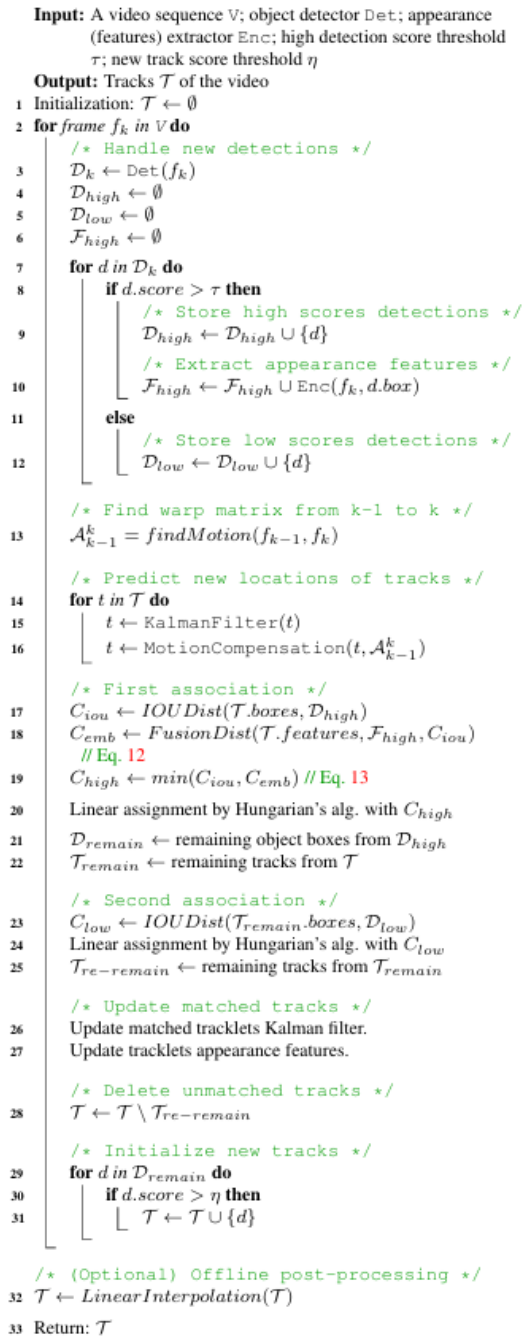
Проблеми попередніх алгоритмів трекінгу:

– неточне прогнозування рамки об’єкта: старіші алгоритми відстеження, такі як SORT та DeepSORT [18], використовували фільтр Калмана, який не оцінював ширину та висоту об'єктів; натомість вони оцінювали співвідношення сторін, що часто призводило до рамки, яка не дуже добре відповідала об'єкту;

– проблеми руху камери: більшість попередніх методів припускали, що камера нерухома; коли камера рухалася (наприклад, у відео з автомобілів або портативних пристроїв), ці трекери часто втрачали слід об'єктів або плутали їх ідентифікацію, що призводило до помилок;

– компроміс між точністю та ідентифікацією: багатьом трекерам доводилося вибирати між тим, щоб бути хорошими у виявленні об'єктів або зберігати той самий ідентифікатор для об'єкта в різних кадрах, але не обидва одночасно.

Псевдокод алгоритму BoT-SORT наведено на рисунку 2.4.



Input: A video sequence V; object detector Det; appearance (features) extractor Enc; high detection score threshold τ; new track score threshold η
Output: Tracks $\mathcal{T}$ of the video

Initialization: $\mathcal{T} \leftarrow \emptyset$
for *frame* f_k *in* V **do**
  /* Handle new detections */
  $\mathcal{D}_k \leftarrow \text{Det}(f_k)$
  $\mathcal{D}_{high} \leftarrow \emptyset$
  $\mathcal{D}_{low} \leftarrow \emptyset$
  $\mathcal{F}_{high} \leftarrow \emptyset$
  for d *in* $\mathcal{D}_k$ **do**
    if $d.score > \tau$ **then**
      /* Store high scores detections */
      $\mathcal{D}_{high} \leftarrow \mathcal{D}_{high} \cup \{d\}$
      /* Extract appearance features */
      $\mathcal{F}_{high} \leftarrow \mathcal{F}_{high} \cup \text{Enc}(f_k, d.box)$
    else
      /* Store low scores detections */
      $\mathcal{D}_{low} \leftarrow \mathcal{D}_{low} \cup \{d\}$
  /* Find warp matrix from k-1 to k */
  $\mathcal{A}^k_{k-1} = findMotion(f_{k-1}, f_k)$
  /* Predict new locations of tracks */
  for t *in* $\mathcal{T}$ **do**
    $t \leftarrow \text{KalmanFilter}(t)$
    $t \leftarrow \text{MotionCompensation}(t, \mathcal{A}^k_{k-1})$
  /* First association */
  $C_{iou} \leftarrow IOUDist(\mathcal{T}.boxes, \mathcal{D}_{high})$
  $C_{emb} \leftarrow FusionDist(\mathcal{T}.features, \mathcal{F}_{high}, C_{iou})$ // Eq. 12
  $C_{high} \leftarrow min(C_{iou}, C_{emb})$ // Eq. 13
  Linear assignment by Hungarian's alg. with C_{high}
  $\mathcal{D}_{remain} \leftarrow$ remaining object boxes from $\mathcal{D}_{high}$
  $\mathcal{T}_{remain} \leftarrow$ remaining tracks from $\mathcal{T}$
  /* Second association */
  $C_{low} \leftarrow IOUDist(\mathcal{T}_{remain}.boxes, \mathcal{D}_{low})$
  Linear assignment by Hungarian's alg. with C_{low}
  $\mathcal{T}_{re-remain} \leftarrow$ remaining tracks from $\mathcal{T}_{remain}$
  /* Update matched tracks */
  Update matched tracklets Kalman filter.
  Update tracklets appearance features.
  /* Delete unmatched tracks */
  $\mathcal{T} \leftarrow \mathcal{T} \setminus \mathcal{T}_{re-remain}$
  /* Initialize new tracks */
  for d *in* $\mathcal{D}_{remain}$ **do**
    if $d.score > \eta$ **then**
      $\mathcal{T} \leftarrow \mathcal{T} \cup \{d\}$
/* (Optional) Offline post-processing */
$\mathcal{T} \leftarrow LinearInterpolation(\mathcal{T})$
Return: $\mathcal{T}$

Рисунок 2.4 – Псевдокод алгоритму BoT-SORT

Як BoT-SORT вирішує ці проблеми:

– краще прогнозування рамки об’єкта: BoT-SORT змінює фільтр Калмана, щоб безпосередньо оцінювати як ширину, так і висоту об'єктів, а не лише їх співвідношення; це робить прогнозовані рамки набагато точніше;

– компенсація руху камери: BoT-SORT включає інтелектуальний крок, який визначає, як камера рухалася між кадрами; він коригує прогнозовані по-

ложення об'єктів, щоб врахувати цей рух камери, тому відстеження залишається точним, навіть якщо камера рухається;

– інтелектуальне зіставлення об'єктів: замість того, щоб просто поєднувати інформацію про рух та зовнішній вигляд, BoT-SORT використовує новий метод, який вибирає найкращий збіг на основі того, наскільки близькі об'єкти та наскільки вони схожі.

Експерименти проводилися на двох найпопулярніших наборах даних у галузі відстеження кількох об'єктів для виявлення та відстеження пішоходів у необмежених умовах: MOT17 та MOT20. MOT17 містить відеокадри, зняті як статичними, так і рухомими камерами. У той час як MOT20 містить сцени скупчення людей. На обох наборах даних BoT-SORT показав найкращі результати по більшості метрик серед алгоритмів трекінгу (рис. 2.5, 2.6).

Tracker	MOTA↑	IDF1↑	HOTA↑	FP↓	FN↓	IDs↓	FPS↑
Tube_TK [30]	63.0	58.6	48.0	27060	177483	4137	3.0
MOTR [55]	65.1	66.4	-	45486	149307	2049	-
CTracker [32]	66.6	57.4	49.0	22284	160491	5529	6.8
CenterTrack [62]	67.8	64.7	52.2	18498	160332	3039	17.5
QuasiDense [31]	68.7	66.3	53.9	26589	146643	3378	20.3
TraDes [49]	69.1	63.9	52.7	20892	150060	3555	17.5
MAT [18]	69.5	63.1	53.8	30660	138741	2844	9.0
SOTMOT [60]	71.0	71.9	-	39537	118983	5184	16.0
TransCenter [50]	73.2	62.2	54.5	23112	123738	4614	1.0
GSDT [45]	73.2	66.5	55.2	26397	120666	3891	4.9
Semi-TCL [23]	73.3	73.2	59.8	22944	124980	2790	-
FairMOT [59]	73.7	72.3	59.3	27507	117477	3303	25.9
RelationTrack [53]	73.8	74.7	61.0	27999	118623	1374	8.5
PermaTrackPr [42]	73.8	68.9	55.5	28998	115104	3699	11.9
CSTrack [24]	74.9	72.6	59.3	23847	114303	3567	15.8
TransTrack [41]	75.2	63.5	54.1	50157	86442	3603	10.0
FUFET [37]	76.2	68.0	57.9	32796	98475	3237	6.8
SiamMOT [25]	76.3	72.3	-	-	-	-	12.8
CorrTracker [44]	76.5	73.6	60.7	29808	99510	3369	15.6
TransMOT [10]	76.7	75.1	61.7	36231	93150	2346	9.6
ReMOT [52]	77.0	72.0	59.7	33204	93612	2853	1.8
MAATrack [40]	79.4	75.9	62.0	37320	77661	1452	**189.1**
OCSORT [9]	78.0	77.5	63.2	**15129**	107055	1950	29.0
StrongSORT++ [12]	79.6	79.5	64.4	27876	86205	**1194**	7.1
ByteTrack [58]	80.3	77.3	63.1	25491	**83721**	2196	29.6
BoT-SORT (ours)	**80.6**	79.5	64.6	22524	85398	1257	6.6
BoT-SORT-ReID (ours)	80.5	**80.2**	**65.0**	22521	86037	1212	4.5

Рисунок 2.5 – Порівняння алгоритмів трекінгу на наборі даних MOT17

Tracker	MOTA↑	IDF1↑	HOTA↑	FP↓	FN↓	IDs↓	FPS↑
MLT [57]	48.9	54.6	43.2	45660	216803	2187	3.7
FairMOT [59]	61.8	67.3	54.6	103440	88901	5243	13.2
TransCenter [50]	61.9	50.4	-	45895	146347	4653	1.0
TransTrack [41]	65.0	59.4	48.5	27197	150197	3608	7.2
CorrTracker [44]	65.2	69.1	-	79429	95855	5183	8.5
Semi-TCL [23]	65.2	70.1	55.3	61209	114709	4139	-
CSTrack [24]	66.6	68.6	54.0	25404	144358	3196	4.5
GSDT [45]	67.1	67.5	53.6	31913	135409	3131	0.9
SiamMOT [25]	67.1	69.1	-	-	-	-	4.3
RelationTrack [53]	67.2	70.5	56.5	61134	104597	4243	2.7
SOTMOT [60]	68.6	71.4	-	57064	101154	4209	8.5
MAATrack [40]	73.9	71.2	57.3	24942	108744	1331	14.7
OCSORT [9]	75.7	76.3	62.4	19067	105894	942	**18.7**
StrongSORT++ [12]	73.8	77.0	62.6	**16632**	117920	**770**	1.4
ByteTrack [58]	**77.8**	75.2	61.3	26249	87594	1223	17.5
BoT-SORT (ours)	77.7	76.3	62.6	22521	**86037**	1212	6.6
BoT-SORT-ReID (ours)	**77.8**	**77.5**	**63.3**	24638	88863	1257	2.4

Рисунок 2.6 – Порівняння алгоритмів трекінгу на наборі даних MOT20

Multiple Object Tracking Accuracy (MOTA) – точність.

False Positive (FP) – помилковий позитивний результат.

False Negative (FN) – помилковий негативний результат.

Frames per second (FPS) – швидкість.

2.2 Методи зменшення шуму в даних з сенсорів

2.2.1. Алгоритм простого ковзного середнього

Сенсори БПЛА можуть надавати неточні дані, особливо під час руху і поворотів, тому дані потребують додаткової попередньої обробки. Для згладжування шуму даних GPS-датчиків було використано алгоритм простого ковзного середнього (Simple Moving Average – SMA) [19]:

$$SMA = \frac{\sum_{i=1}^{n} L_i}{n},$$

де L_i – довгота або широта;

n – довжина згладжування або період SMA.

Значення періоду будо обрано експериментальним шляхом – 1 секунда, що відповідає 3 координатам від GPS-датчика (рис. 2.7). Результат застосування SMA можна побачити на рисунку 2.8.

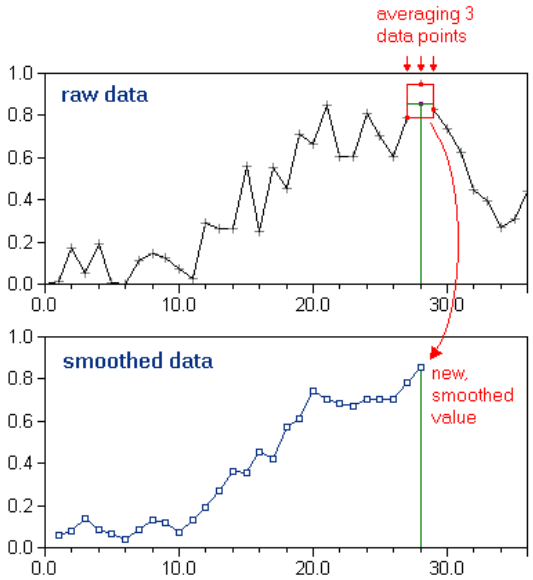


Рисунок 2.7 – Застосування SMA з вікном розміром 3 точки

Рисунок 2.8 – Трекінг об’єктів до та після застосування SMA

2.2.2. Фільтр Калмана

Фільтр Калмана був винайдений Рудольфом Калманом, який отримав Національну медаль науки 7 жовтня 2009 року. Фільтри Калмана вперше були використані під час космічної програми «Аполлон», яка відправила людей на Місяць, у космічних човнах NASA, підводних човнах ВМС США, а також у безпілотних аерокосмічних апаратах та озброєнні.

Давайте розглянемо приклад, чому нам може знадобитися фільтр Калмана. Припустимо, що ми хочемо відстежувати місцезнаходження БПЛА у певний момент часу t. Ми можемо використовувати для цього GPS, але точність може відрізнятися залежно від кількості доступних супутників та інших факторів. Ми можемо використовувати вбудований датчик для визначення пройденої відстані, але наш датчик може бути шумним. Тому ми можемо об'єднати вимірювання обох датчиків, щоб знайти оптимальну оцінку точного місцезнаходження БПЛА. Тобто, ми використовуємо фільтр Калмана, коли хочемо знайти найкращу оцінку станів, поєднуючи вимірювання з різних датчиків за наявності шуму [20].

Простіше кажучи, наша мета – визначити поточний стан системи, враховуючи як її основну динаміку, так і наявність недосконалих вимірювань.

Типи фільтрів Калмана. Вибір фільтра залежить від конкретних характеристик моделі системи та вимог до точності застосування. У деяких випадках може знадобитися провести експерименти або моделювання, щоб визначити, який фільтр найкраще працює в даній ситуації.

Лінійний фільтр Калмана [21]:

– припускає, що динаміка системи та моделі вимірювання є лінійними;

– оновлює оцінку стану та коваріацію безпосередньо за допомогою лінійних рівнянь;

– підходить для систем, які можна точно представити лінійними моделями;

– широко використовується в таких додатках, як системи відстеження,

навігації та керування.

Розширений фільтр Калмана [22]:

– розширює лінійний фільтр Калмана для обробки нелінійної системної динаміки;

– апроксимує нелінійні моделі за допомогою розкладання в ряд Тейлора першого порядку;

– потрібне обчислення матриць Якобіана для нелінійних моделей;

– використовується, коли динаміка системи або вимірювання демонструють нелінійну поведінку;

– зазвичай використовується в таких застосуваннях, як робототехніка, автономні транспортні засоби та аерокосмічні системи.

Коли ми відображаємо відфільтровані значення разом з істинним положенням та вимірюваннями, то спостерігаємо, що спочатку відфільтровані значення відповідають вимірюванням GPS, але з часом вони наближаються до істинного значення, навіть коли вимірювання GPS значно відхиляються, як показано жовтими стрілками на рисунку 2.9:

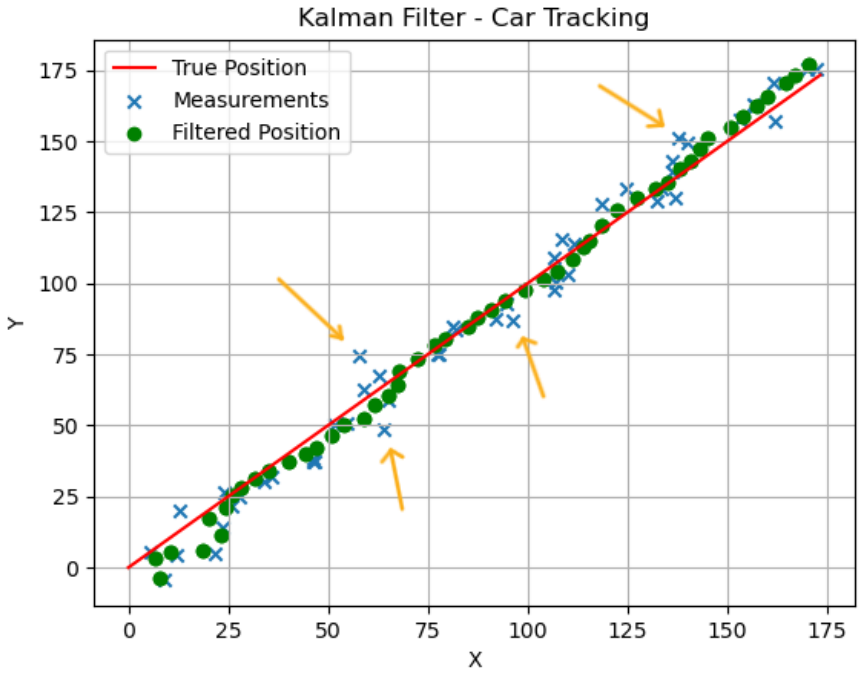


Рисунок 2.9 – Результат застосування фільтра Калмана до вимірів GPS

Висновки за розділом 2

В даному розділі були виконано:

– розглянуто існуючі алгоритми трекінгу об’єктів і методи зменшення шуму в даних з сенсорів;

– виконано порівняння алгоритмів трекінгу об’єктів по точності та іншим параметрам;

– обрано оптимальний алгоритм трекінгу і метод зменшення шуму в даних з сенсорів;

– показано алгоритм роботи трекера BoT-SORT;

– дослідження застосування фільтра Калмана до вимірів GPS.

Алгоритмом трекінгу було обрано детектор+трекер по причині можливості реідентифікації об’єктів після їх зникнення з поля зору. Алгоритм SMA успішно виконує зменшення шуму в даних з сенсорів.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Модель комп’ютерного зору YOLO

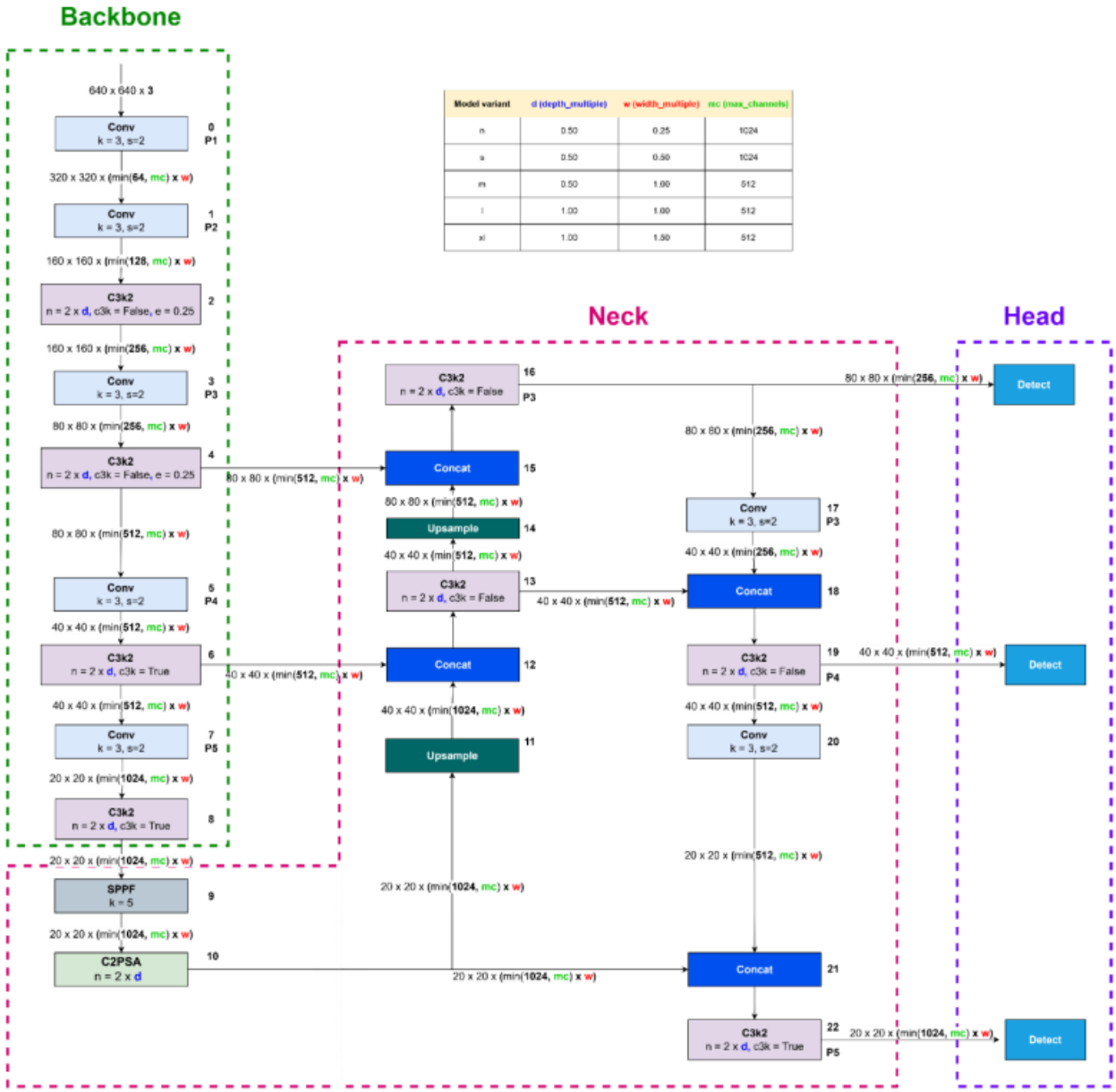
Ключовим проривом у виявленні об'єктів стала поява алгоритму You Only Look Once (YOLO) у 2015 році. Цей інноваційний підхід, як випливає з назви, обробляє все зображення за один прохід для виявлення об'єктів та їхнього розташування. Методологія YOLO відрізняється від традиційних двоетапних процесів виявлення, розглядаючи виявлення об'єктів як задачу регресії. Вона використовує єдину згорткову нейронну мережу для одночасного прогнозування рамок об’єктів та ймовірностей класів по всьому зображенню, що спрощує конвеєр виявлення порівняно з більш складними традиційними методами.

YOLOv11 – це остання версія серії YOLO, що базується на фундаменті, закладеному YOLOv1. Представлена на конференції YOLO Vision 2024 (YV24), YOLOv11 являє собою значний крок вперед у технології виявлення об'єктів у реальному часі. Ця нова версія вносить суттєві покращення як в архітектуру, так і в методології навчання, розширюючи межі точності, швидкості та ефективності.

Інноваційний дизайн YOLOv11 включає в себе передові методи визначення ознак, що дозволяє отримувати більше деталей, зберігаючи при цьому мінімальну кількість параметрів. Це призводить до підвищення точності в широкому спектрі завдань комп'ютерного зору, від виявлення об'єктів до класифікації. Крім того, YOLOv11 досягає значного приросту швидкості обробки, суттєво покращуючи можливості роботи в режимі реального часу.

По суті, архітектура YOLO складається з трьох фундаментальних компонентів (рис. 3.1). По-перше, backbone, служить основним виявлячем ознак, використовуючи згорткові нейронні мережі для перетворення зображень в багатомасштабні карти ознак. По-друге, компонент neck, діє як проміжний етап обробки, використовуючи спеціалізовані шари для агрегації та покращення представлень ознак у різних масштабах. По-третє, компонент head, функціонує як

механізм прогнозування, генеруючи кінцеві результати для локалізації та класифікації об'єктів на основі уточнених карт ознак [23].



Model variant	d (depth_multiple)	w (width_multiple)	mc (max_channels)
n	0.50	0.25	1024
s	0.50	0.50	1024
m	0.50	1.00	512
l	1.00	1.00	512
xl	1.00	1.50	512

Рисунок 3.1 – Архітектура моделі YOLOv11 [24]

Розмір вхідного зображення в YOLOv11 буде змінено зі збереженням співвідношення сторін зображення. Щоб зберегти співвідношення сторін, зображення буде доповнено сірими пікселями. Якщо вхідне зображення є квадратом, його розмір буде змінено без додавання.

Згортковий (Conv) блок (рис. 3.2) містить двовимірний згортковий шар,

двовимірну батч нормалізацію та функцію активації SILU.

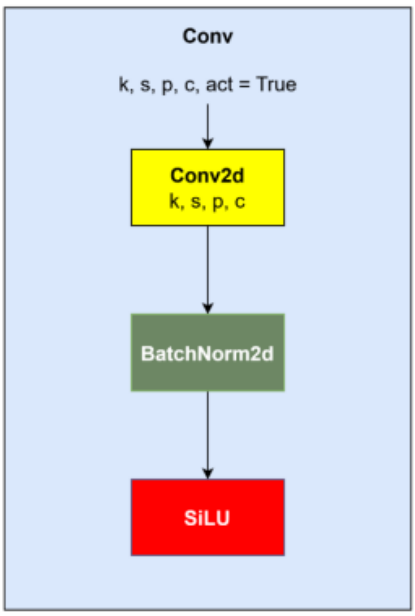


Рисунок 3.2 – Згортковий блок

У блоку детекції (рис. 3.3) класифікаційна голова та регресійна голова представляють різні архітектури. Ця диференціація була зумовлена різницею в обчислювальних витратах. Класифікаційна голова може бути вдвічі більшою, ніж регресійна головка.

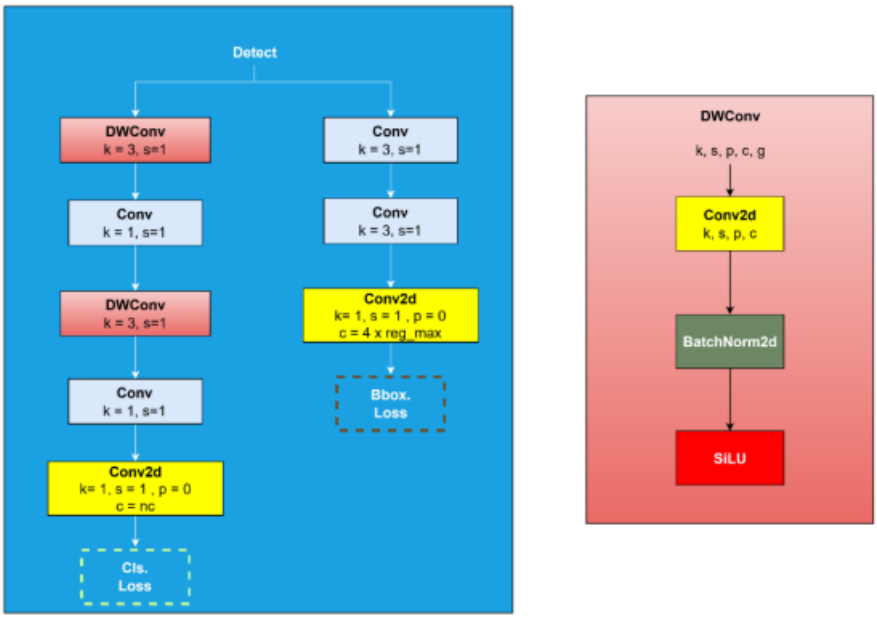


Рисунок 3.3 – Блок детекції

YOLOv11 підтримує широкий спектр завдань з візуалізації, демонструючи свою універсальність та потужність у різних застосуваннях. Ось огляд ключових завдань:

– виявлення об'єктів: YOLOv11 чудово справляється з ідентифікацією та локалізацією об'єктів на зображеннях або відеокадрах, забезпечуючи рамки для кожного виявленого об’єкта;

– сегментація екземплярів: виходячи за рамки простого виявлення, YOLOv11 може ідентифікувати та розділяти окремі об'єкти на зображенні аж до рівня пікселів;

– класифікація зображень: YOLOv11 здатна класифікувати зображення за заздалегідь визначеними категоріями;

– оцінка пози: модель може виявляти певні ключові точки на зображеннях або відеокадрах для відстеження рухів або поз;

– виявлення орієнтованих об'єктів: модель дає можливість виявлення об'єктів з кутом орієнтації, що дозволяє точніше локалізувати повернуті об'єкти;

– відстеження об'єктів: ідентифікує та відстежує шлях об'єктів у послідовності зображень або відеокадрів .

3.2 Алгоритм розв’язання задачі географічної локалізації об’єктів

Розрахунок геолокації об’єктів виконується використовуючи:

– координати об’єкта на зображенні (результати трекінг моделі);

– географічні координати БПЛА (довгота, широта);

– висоту польоту БПЛА;

– азимут (напрям польоту БПЛА);

– параметри камери (розміри сенсора, фокусна відстань, розміри зображення).

Для спрощення розрахунків камера зафіксована в напрямку перпендикулярно до поверхні землі.

Першим кроком розраховується відстань l від центра зображення до об’єкта використовуючи ground sampling distance (рис. 3.4) [25]:

$$GSD_w = \frac{FlightHeight \cdot SensorWidth}{FocalLength \cdot ImageWidth},$$

$$GSD_h = \frac{FlightHeight \cdot SensorHeight}{FocalLength \cdot ImageHeight},$$

$$l = \sqrt{\left(\Delta x \cdot GSD_w\right)^2 + \left(\Delta y \cdot GSD_h\right)^2}.$$

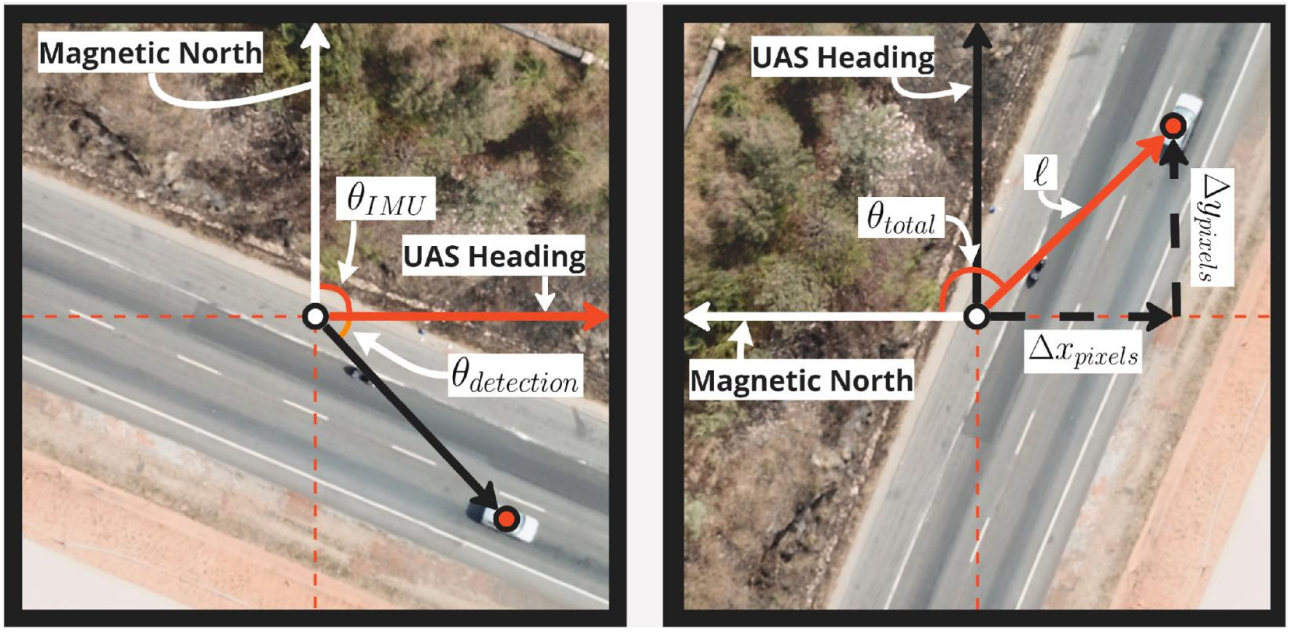


Рисунок 3.4 – Розрахунок геолокації об’єкта

Далі для визначення географічних координат об’єкта використовується Python бібліотека geographiclib.geodesic [26], яка розв’язує пряму проблему геодезії [27]:

$$Geodesic.WGS84.Direct(latitude_{uav}, longitude_{uav}, \alpha, l),$$

де α – азимут до об’єкта, що визначається відносно азимута БПЛА.

3.3 Опис програми

Програма написана мовою програмування Python із застосуванням бібліотек YOLO для розробки моделі, OpenCV для обробки вхідного відео файлу, Geographiclib для геодезичних розрахунків, Folium для побудови карти.

Вихідний код завантажено на Github [28].

Програма поділяється на декілька модулів:

– config: параметри тренування моделі, вхідні/вихідні файли;

– train: тренування моделі;

– tracker: зчитування вхідних файлів, застосування моделі, збереження результатів;

– geo: функції географічної локалізації і відображення на карті.

Результати виконання зберігаються в HTML файлі, який містить карту OpenStreetMap з нанесеними маршрутами БПЛА і знайдених об’єктів (рис. 3.5).

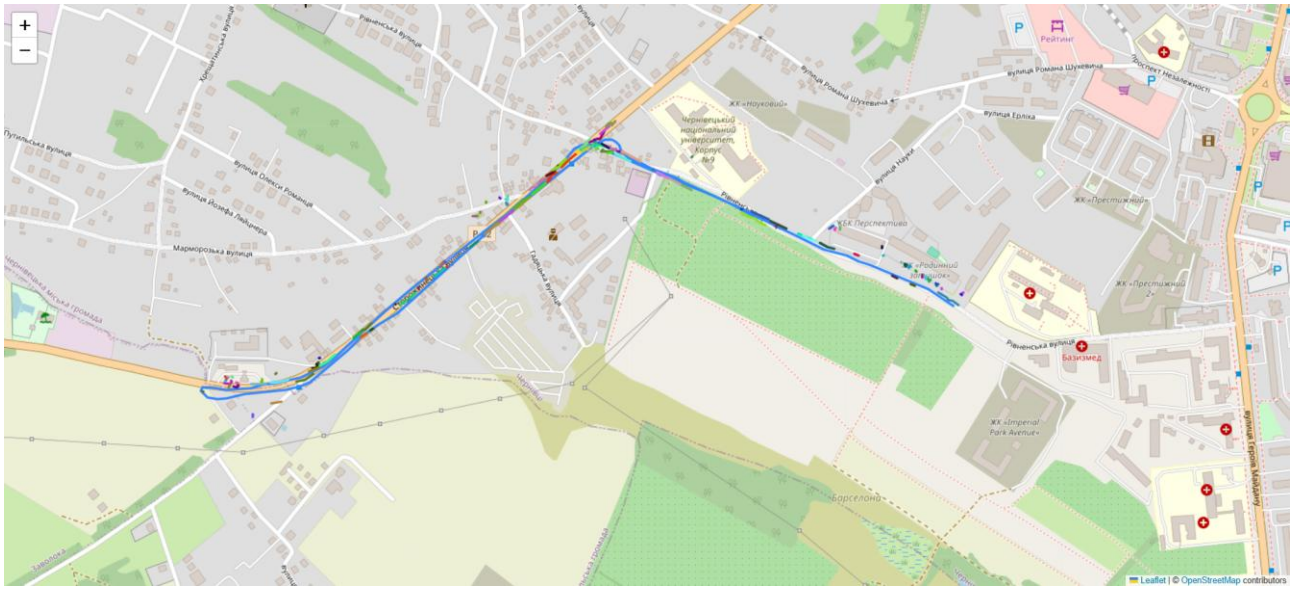


Рисунок 3.5 – Карта з маршрутом БПЛА і знайденими об’єктами

Висновки за розділом 3

У цьому розділі було детально розглянуто програмне середовище виконання практичної частини роботи, що включає опис алгоритму розв’язку задачі географічної локалізації об’єктів і розробку програмного забезпечення під алгоритм розв’язку задачі.

Програма розроблена на крос-платформній мові програмування Python із використанням популярних і потужних бібліотек для роботи із даними, тренування нейромереж, специфічних бібліотек для задач геодезії і візуалізації результатів.

Модель YOLO була обрана для трекінгу через простоту інтерфейсу тренування. Сенсори БПЛА дають можливість виконати географічну локалізацію об’єктів. Для візуалізації використано карту OpenStreetMap з нанесеними маршрутами БПЛА і знайденими об’єктами.

4 РЕЗУЛЬТАТИ ОБЧИСЛЮВАЛЬНОГО ЕКСПЕРИМЕНТУ ТА ЇХ АНАЛІЗ

4.1 Тренування моделі

Для тренування моделі був використаний набір даних з 500 розмічених зображень. Для розмітки використовувався сервіс Roboflow [29]. З метою збільшення набору даних було застосовано аугментації: поворот на 90˚ за годинниковою стрілкою (рис. 4.1), поворот на 90˚ проти годинникової стрілки, поворот на 180˚ (рис. 4.2).

Рисунок 4.1 – Аугментація поворот на 90˚ за годинниковою стрілкою

Рисунок 4.2 – Аугментація поворот на 180˚

Параметри тренування моделі [30] наведено в таблиці Б1, їх опис:

epoch – номер епохи, кількість проходів навчального алгоритму по повних набору даних;

train/box_loss – різниця між прогнозованими обмежувальними рамками та фактичними обмежувальними рамками об'єктів;

train/cls_loss – різниця між прогнозованими ймовірностями класів та фактичними мітками класів об'єктів;

train/dfl_loss – різниця між прогнозованими картами ознак та фактичними картами ознак об'єктів;

metrics/precision(B) – частка істинно позитивних виявлень серед усіх передбачених обмежувальних рамок;

metrics/recall(B) – частка істинно позитивних виявлень серед усіх фактичних обмежувальних рамок;

metrics/mAP50(B) – середня точність моделі для різних категорій об'єктів з 50% порогом IoU;

metrics/mAP50-95(B) – середня точність моделі для різних категорій об'єктів з порогом IoU в межах 50-95%;

val/box_loss – метрика, яка використовується для оцінки точності моделі в прогнозуванні розташування та розмірів об'єктів на даних, які вона не бачила під час навчання;

val/cls_loss – метрика, яка використовується для оцінки точності моделі в прогнозуванні класів об'єктів на даних, які вона не бачила під час навчання;

val/dfl_loss – метрика, яка використовується для оцінки точності моделі в прогнозуванні карт ознак об'єктів на даних, які вона не бачила під час навчання;

lr/pg0 – learning rate для першої групи параметрів;

lr/pg1 – learning rate для другої групи параметрів;

lr/pg2 – learning rate для третьої групи параметрів.

Програмний код для запуску процесу тренування моделі:

```
import os
from ultralytics import YOLO
from config import dataset_file_path, train_image_size, model_file_path
model = YOLO('yolo11n.pt').train(data=dataset_file_path, epochs=100, imgsz=train_image_size, device='cpu')
os.replace(model.save_dir.joinpath('weights/best.pt'), model_file_path)
```

4.2 Результати роботи моделі

Для тестування використовувався БПЛА DJI Mini 4 Pro. Зроблено відеозапис польоту на містом з додаванням даних телеметрії.

Приклад даних телеметрії:

```
00:00:00,000 --> 00:00:00,033
[iso: 100] [shutter: 1/798.21] [fnum: 2.8] [ev: 0] [color_md : default] [ae_meter_md: 1] [focal_len: 24.00] [dzoom_ratio: 1.00], [latitude: 48.267013] [longitude: 25.914562] [rel_alt: 102.229 abs_alt: 426.185] [gb_yaw: -65.8 gb_pitch: -89.9 gb_roll: 0.0]
```

Конфігурація програми:

```
dataset_file_path = "./dataset/data.yaml"
model_file_path = "./model.pt"
metrics_file_path = "./metrics.srt"
video_file_path = "./video.mp4"
map_file_path = "./map.html"
device = "cpu"
original_image_width = 1920
original_image_height = 1080
train_image_size = 640
```

track_confidence = 0.5

sensor_width = 6.4

sensor_height = 4.8

focal_length = 24.00

Приклад знайдених об’єктів на відеозаписі наведено на рисунку 4.3.

Рисунок 4.3 – Приклад роботи трекінг моделі

Висновки за розділом 4

У даному розділі було зроблено аналіз результатів обчислювального експерименту. По результатам видно, що модель натренована загально добре, але є місця, де модель втрачає ідентифікатор об’єкта при його виході в поля зору камери БПЛА.

Описано набір даних, що використовувався для тренування моделі, параметри моделі в процесі тренування. Наведено приклад програмного коду для запуску процесу тренування моделі, приклад даних телеметрії і конфігурацію програми.

Відображення результатів роботи моделі на карті дає можливість зробити візуальну оцінку точності розрахунків. Для збільшення набору даних можна використати аугментацію.

ВИСНОВКИ

У ході проведення дослідження, описаного у кваліфікаційній роботі, було виконано огляд, аналіз сучасного стану, методів вирішення задачі трекінгу об’єктів і методів зменшення шуму в даних з сенсорів БПЛА.

Розроблено алгоритм географічної локалізації об’єктів за допомогою БПЛА.

Зроблено дослідження застосування фільтра Калмана до вимірів GPS.

Створено набір даних для тренування моделі з використанням аугментації.

Виконано тренування нейронної мережі для трекінгу об’єктів.

Розроблено програмне забезпечення для обробки відеозаписів і даних телеметрії.

Зроблено візуальну оцінку точності моделі за допомогою відображення маршрутів об’єктів на карті.

Результати отримані у ході проведення дослідження показують, що моделі нейронних мереж гарно справляються з задачею трекінгу об’єктів, але є місця, де модель втрачає ідентифікатор об’єкта при його виході в поля зору камери БПЛА.

Визначено оптимальним для практичного застосування комбінований підхід, де детектор (YOLO) визначає об’єкт, а трекер (BoT-SORT, ByteTrack) забезпечує його супровід між кадрами.

Вирішено проблему шуму в даних з сенсорів.

Результати можна застосовувати для створення карт місцевості у цивільних та військових завданнях.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Єсілевський В. С., Романенко Г. В. Трекінг об’єктів за допомогою безпілотного літального апарату. *29-й Міжнародний молодіжний форум «Радіоелектроніка і молодь у XXI столітті»* : зб. матеріалів форуму (м. Харків, 16-19 квітня 2025р.). Т. 7. Харків : ХНУРЕ, 2025. С. 309 – 310.

2. Romanenko Hryhorii. Object tracking using unmanned aerial vehicle. *IV Міжнародна науково-практична конференція «Європейські студії. Навчання і викладання: у світі технологій»* : збірник матеріалів (Харків, Україна – Клуж-Напока, Румунія, 12 листопада 2025р.). Харків : ХНУРЕ, 2025. С. 239.

3. Richard Szeliski. Computer Vision. Algorithms and Applications. Second Edition. Springer, 2022. С. 478. DOI: https://doi.org/10.1007/978-3-030-34372-9

4. Adrian Kaehler, Gary Bradski. Learning OpenCV 3. O’Reilly, 2016. С. 495.

5. Теорія і практика застосування безпілотних літальних апаратів. UA Dynamics, 2022. С. 52.

6. Maharani, D.A., Machbub, C., Yulianti, L. et al. Deep features fusion for KCF-based moving object tracking. *J. Big Data*. 2023. DOI: https://doi.org/10.1186/s40537-023-00813-5.

7. Yejin Yan, Wenxiao Huo, Jiayu Ou, Zhifeng Liu, Tianping Li. Improved SiamFC Target Tracking Algorithm Based on Anti-Interference Module. *Intelligent Sensing, Monitoring, and Optimization of Advanced Manufacturing Systems*. DOI: https://doi.org/10.1155/2022/2804114.

8. Jiao S., Ding H., Zhong Y., Yao X., Zheng J. A UAV Target Tracking and Control Algorithm Based on SiamRPN. *Journal of System Simulation*. 2023. DOI: http://dx.doi.org/10.16182/j.issn1004731x.joss.22-0142.

9. Hu W., Wang Q., Zhang L., Bertinetto L., Torr P. H. S. SiamMask: A Framework for Fast Online Object Tracking and Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2023. DOI: https://doi.org/10.48550/arXiv.2207.02088.

10. Chen X., Yan B., Zhu J., Wang D., Yang X., Lu H. Transformer Tracking.

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021. DOI: https://doi.org/10.48550/arXiv.2103.15436.

11. Meinhardt T., Kirillov A., Leal-Taixe L., Feichtenhofer Ch. TrackFormer: Multi-Object Tracking With Transformers. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (CVPR), 2022. DOI: https://doi.org/10.48550/arXiv.2101.02702.

12. Optical Flow. OpenCV Tutorials. URL: https://docs.opencv.org/3.4/d4/dee/tutorial_optical_flow.html (дата звернення: 10.12.2025).

13. Real-Time Hand Palm Detection and Tracking Augmented Reality Game Using Lucas Kanade Optical Flow Combined with Color Blob Detection. Hardjono B. et al. *2019 International Conference on Reliability, Infocom Technologies and Optimization* (ICORIS). 2019. P. 1–6. DOI: https://doi.org/10.1109/ICORIS.2019.8874892.

14. A multi-scale refinement corner detection algorithm based on Shi-Harris / Li J. et al. *Digital Signal Processing*. 2025. Vol. 158. Art. 105137. DOI: https://doi.org/10.1016/j.dsp.2025.105137.

15. Moving object Detection Based on Farneback Optical Flow / Zhang J. et al. *2023 42nd Chinese Control Conference* (CCC). 2023. P. 7701–7706. DOI: https://doi.org/10.23919/CCC58697.2023.10239983.

16. Aharon N., Orfaig R., Bobrovsky B.-Z. BoT-SORT: Robust associations multi-pedestrian tracking. *arXiv preprint*. 2022. arXiv:2206.14651. DOI: https://doi.org/10.48550/arXiv.2206.14651.

17. Zhang, Y. *et al.* (2022). ByteTrack: Multi-object Tracking by Associating Every Detection Box. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds) Computer Vision – ECCV 2022. ECCV 2022. Lecture Notes in Computer Science, vol 13682. Springer, Cham. DOI: https://doi.org/10.1007/978-3-031-20047-2_1.

18. Liu, H.; Pei, Y.; Bei, Q.; Deng, L. Improved DeepSORT Algorithm Based on Multi-Feature Fusion. *Applied System Innovation*. 2022. DOI: https://doi.org/10.3390/asi5030055.

19. Moving Average. Fundamentals of Statistics. URL: http://www.statistics4u.com/fundstat_eng/cc_moving_average.html (дата звернення: 10.12.2025).

20. UAV Drone: Object Tracking using Kalman Filter. Github.com. URL: https://github.com/yudhisteer/UAV-Drone-Object-Tracking-using-Kalman-Filter (дата звернення: 10.12.2025).

21. Intensive Review of Drones Detection and Tracking: Linear Kalman Filter Versus Nonlinear Regression, an Analysis Case / Taha B. et al. *Archives of Computational Methods in Engineering*. 2023. Vol. 30, Issue 5. P. 3131–3155. DOI: https://doi.org/10.1007/s11831-023-09894-0.

22. Enhanced Method of Object Tracing Using Extended Kalman Filter via Binary Search Algorithm / Mohsin J. et al. *Journal of Information Technology Management*. 2022. Vol. 14. P. 138–156. DOI: https://doi.org/10.22059/jitm.2022.86665.

23. YOLOv11: An overview of the key architectural enhancements / Terven J. et al. // *arXiv preprint*. 2024. arXiv:2410.17725. DOI: https://doi.org/10.48550/arXiv.2410.17725.

24. YOLOv8 to YOLO11: A comprehensive architecture in-depth comparative review / Hussain M. et al. // *arXiv preprint*. 2025. arXiv:2501.13400. DOI: https://doi.org/10.48550/arXiv.2501.13400.

25. Low-cost real-time aerial object detection and GPS location tracking pipeline / Ciaburro G. et al. *Open Photo*. 2024. Vol. 2. Art. 100069. DOI: https://doi.org/10.1016/j.ophoto.2024.100069.

26. Sambolek S., Ivasic-Kos M. Determining the geolocation of a person detected in an image taken with a drone. *Computer Vision and Image Understanding*. 2021. Vol. 210. Art. 103243. DOI: http://dx.doi.org/10.2139/ssrn.4373987.

27. Karney C. F. F. Algorithms for geodesics. *arXiv preprint*. 2011. arXiv:1109.4448. DOI: https://doi.org/10.1007/s00190-012-0578-z.

28. Car tracking. Github.com. URL: https://github.com/grigoriy-romanenko/car-tracking (дата звернення: 10.12.2025).

29. Car tracking dataset. Roboflow.com. URL: https://universe.roboflow.com/car-track/car-track-vpsso/dataset/3 (дата звернення: 10.12.2025).

30. Evaluating YOLO model performance. Kaggle.com. URL: https://www.kaggle.com/discussions/getting-started/523677 (дата звернення: 10.12.2025).