

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

«На правах рукопису»
УДК 004.42+004.932.72'1

«До захисту допущено»

Завідувач кафедри

_____ Едуард ЖАРИКОВ

« ____ » _____ 2025 р.

Магістерська дисертація

на здобуття ступеня магістра

**за освітньо-професійною програмою «Інженерія програмного забезпечення
інформаційних систем»**

зі спеціальності 121 «Інженерія програмного забезпечення»

**на тему: «Метод та програмні засоби відеомоніторингу характеристик
автомобільного транспортного потоку.»**

Виконав:

студент II курсу, групи ІІ-42мп

Проценко Роман Олександрович _____

Керівник:

д.т.н., проф.

Стеценко Інна Вячеславівна _____

Рецензент:

д.ф.-м.н., проф.

Дорошенко Анатолій Юхимович _____

Засвідчую, що у цій магістерській дисертації
немає запозичень з праць інших авторів без
відповідних посилань.

Студент (-ка) _____

Київ – 2025 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Рівень вищої освіти – другий (магістерський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма «Інженерія програмного забезпечення інформаційних систем»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Едуард ЖАРІКОВ

«__» _____ 2025р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Проценко Роман Олександрович

1. Тема дисертації «Метод та програмні засоби відеомоніторингу характеристик автомобільного транспортного потоку», науковий керівник дисертації Стеценко Інна Вячеславівна, доктор технічних наук, професор, затверджені наказом по університету від «06» листопада 2025 р. № 4841-с

2. Термін подання студентом дисертації «15» грудня 2025 р.

3. Об'єкт дослідження – програмне забезпечення відеомоніторингу автомобільного транспортного потоку.

4. Предмет дослідження – підходи, методи та засоби відеомоніторингу автомобільного транспортного потоку.

5. Перелік завдань, які потрібно розробити:

- аналіз проблеми та існуючих рішень;
- розробка підходу автоматичного визначення дорожнього полотна;
- вдосконалення показників TLCR та MTLCR з урахуванням різних типів транспортних засобів;
- розробка єдинної моделі для оцінки статичних та динамічних показників транспортного потоку;
- розробка програмного забезпечення для автоматизованого відеомоніторингу транспортного руху;
- дослідження ефективності розробленого програмного забезпечення.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу – 2 плакати
7. Орієнтовний перелік публікацій – дві публікація
8. Дата видачі завдання «1» вересня 2025 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання	Примітка
1	Ознайомлення з завданням	1.09.2025	
2	Аналіз предметної області	29.09.2025	
3	Аналіз існуючих рішень	13.10.2025	
4	Розробка методу автоматичного виявлення дорожнього полотна	20.10.2025	
5	Розробка та реалізація модуля обчислення мультикласового показнику TLCR	27.10.2025	
6	Розробка підходу з використанням єдинної моделі для оцінки статичних та динамічних характеристик	3.10.2025	
7	Виконання експериментальних досліджень	10.11.2025	
8	Оформлення пояснювальної записки	24.11.2025	
9	Подання дисертації на попередній захист	28.11.2025	
10	Подання дисертації на захист	15.12.2025	

Студент

Роман ПРОЦЕНКО

Науковий керівник

Інна СТЕЦЕНКО

РЕФЕРАТ

Розмір пояснювальної записки – 94 аркушів, містить 13 ілюстрацій, 28 таблиць, 2 додатки, 20 посилань на джерела.

Актуальність теми. У роботі розглянуто проблему аналізу характеристик автомобільного транспортного потоку, показано основні особливості існуючих методів оцінки статичних та динамічних показників транспортного трафіку, їх переваги та недоліки. Виявлено потребу у вдосконаленні методів та підходів, здатних комплексно у режимі, наближеному до реального часу, оцінювати стан транспортного потоку.

Мета дослідження. Основною метою є підвищення точності, швидкодії та масштабованості аналізу результатів відеоспостереження для подальшого використання в системах управління транспортною інфраструктурою.

Об'єкт дослідження: програмне забезпечення відеомоніторингу автомобільного транспортного потоку.

Предмет дослідження: підходи, методи та програмний засіб обробки та аналізу відеоданих для визначення характеристик транспортного потоку.

Для реалізації поставленої мети **сформульовані наступні завдання:**

- виконати аналіз сучасних методів та програмних рішень для відеомоніторингу дорожнього руху;
- визначити вимоги до системи відеомоніторингу транспортного потоку;
- розробити підхід автоматичного визначення маски дорожнього полотна;
- створити мультикласовий показник для оцінювання характеристик транспортного потоку;
- створити єдину модель для оцінки статичних та динамічних показників транспортного потоку;
- розробити програмне забезпечення для автоматизованого відеомоніторингу транспортного руху;

- оцінити ефективність та точність роботи запропонованих методів і програмних засобів.

Наукова новизна результатів магістерської дисертації полягає в тому, що:

- *вперше* розроблено метод автоматичного визначення дорожнього полотна в режимі, близькому до реального часу, який об'єднує виявлення ліній розмітки, геометричну фільтрацію, аналіз перспективної трансформації та формування полігональної маски дороги;
- *вдосконалено* показники TLCR та MTLCR для більш якісної оцінки завантаженості дорожнього полотна, за рахунок урахування різних типів транспортних засобів та їхнього впливу на пропускну здатність;
- *вперше* запропоновано єдиний підхід до визначення статичних та динамічних характеристик транспортного потоку на основі однієї моделі об'єктної сегментації, яка одночасно визначає маски транспортних засобів, їх класи, положення та обмежувальні рамки.

Практичне значення отриманих результатів полягає в розробці програмного засобу для визначення характеристик транспортного потоку в режимі близькому до реального часу та може використовуватися для оптимізації роботи світлофорів, прогнозування заторів, аналізу інфраструктури й статистичного моніторингу. Крім того, запропонований підхід може бути застосований і для інших типів руху.

Зв'язок з науковими програмами, планами, темами. Робота виконувалась на кафедрі інформатики та програмної інженерії Національного технічного університету України "Київський політехнічний інститут імені Ігоря Сікорського" в межах ініціативної НДР "Методи, моделі та середовище проектування та оцінювання ефективності програмних засобів штучного інтелекту для систем моніторингу та управління", державний реєстраційний номер 0125U001681.

Апробація. Наукові положення дисертації пройшли апробацію на V

Міжнародної науково-практичної конференції молодих вчених та студентів «Інженерія програмного забезпечення і передові інформаційні технології (Soft Tech-2025)» (Soft Tech-2025) – м. Київ.

Публікації. Наукові положення дисертації опубліковані в:

- 1) Проценко Р.О., Стеценко І.В. Метод та програмні засоби відеомоніторингу характеристик автомобільного транспортного потоку. Інженерія програмного забезпечення і передові інформаційні технології (Soft Tech-2025): матеріали ІХ Міжнародної науково-практичної конференції молодих вчених та студентів, 26-27 листопада 2025 року, м. Київ, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», ФІОТ.
- 2) Проценко Р.О., Стеценко І.В. (2026). Метод та програмні засоби відеомоніторингу характеристик автомобільного транспортного потоку. Міжвідомчий науково-технічний збірник «Адаптивні системи автоматичного управління» 1(48) [Прийнята до друку].

Ключові слова: ВІДЕОМОНІТОРИНГ, ТРАНСПОРТНИЙ ПОТІК, TLCS, ДОРОЖНЄ ПОЛОТНО, ВІДСТЕЖЕННЯ ТРАНСПОРТНИХ ЗАСОБІВ.

ABSTRACT

Explanatory note size – 94 pages, contains 13 illustrations, 28 tables, 2 applications, 20 references.

Topicality. Examines the problem of analyzing the characteristics of automobile traffic flow, shows the main features of existing methods for assessing static and dynamic indicators of transport traffic, their advantages and disadvantages. The need for improving methods and approaches capable of comprehensively assessing the state of the traffic flow in a mode close to real time has been identified.

The aim of the study. The main goal is to improve the accuracy, processing speed, and scalability of video surveillance analysis for its further use in transportation infrastructure management systems.

The object of research: software for video monitoring of automobile traffic flow.

The subject of research: approaches, methods and software for processing and analyzing video data to determine the characteristics of the traffic flow.

To achieve this goal, the **following tasks** were formulated:

- analyze modern methods and software solutions for video monitoring of road traffic;
- determine the requirements for the video monitoring system of traffic flow;
- develop an approach for automatic determination of the road surface mask;
- create a multi-class indicator for assessing the characteristics of traffic flow;
- create a single model for assessing static and dynamic indicators of traffic flow;
- develop software for automated video monitoring of traffic;
- assess the efficiency and accuracy of the proposed methods and software.

The scientific novelty of the results of the master's dissertation is:

- the approach to automatic determination of the road surface in a mode close to real time has been proposed, which combines the detection of marking lines, geometric filtering, analysis of perspective transformation and the formation of a polygonal road mask;
- the TLCR and MTLCR indicators have been improved for a better assessment of roadway congestion, taking into account different types of vehicles and their impact on throughput;
- the unified approach to determining static and dynamic characteristics of traffic flow has been proposed based on a single object segmentation model, which simultaneously determines vehicle masks, their classes, positions and bounding boxes.

The practical value of the obtained results is the developed method provides determination of traffic flow characteristics in a mode close to real time and can be used to optimize traffic lights, predict congestion, analyze infrastructure and statistical monitoring. In addition, the proposed approach can be applied to other types of traffic.

Relationship with working with scientific programs, plans, topics. Work was performed at the Department of Computer Science and Software Engineering of the National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute» within the initiative research «Methods, models and environment for designing and evaluating the effectiveness of artificial intelligence software tools for monitoring and control systems», state registration number 0125U001681.

Approbation. The scientific provisions of the dissertation were tested at the V International Scientific and Practical Conference of Young Scientists and Students "Software Engineering and Advanced Information Technologies (Soft Tech-2025)" - Kyiv.

Publications. The scientific provisions of the dissertation were published in:

- 1) Protsenko R.O., Stetsenko I.V. Method and software tools for video monitoring of characteristics of automobile traffic flow. Software engineering and advanced information technologies (Soft Tech-2025): materials of the IX International Scientific and Practical Conference of Young Scientists and Students, November

26-27, 2025, Kyiv, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, FIOT.

- 2) Protsenko R.O., Stetsenko I.V. (2026). Method and software tools for video monitoring of characteristics of automobile traffic flow. Interdepartmental scientific and technical collection "Adaptive automatic control systems" 1(48) [Accepted for publication].

Keywords: VIDEO MONITORING, TRAFFIC FLOW, TLCR, ROADWAY, VEHICLE TRACKING.

ЗМІСТ

ЗМІСТ	10
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	12
ВСТУП	13
1 АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ	16
1.1 Опис предметної області.....	16
1.2 Автоматичне визначення площини дорожнього покриття	18
1.3 Методи виявлення транспортних засобів	20
1.4 Аналіз характеристик дорожнього трафіку	23
1.4.1 Статичні характеристики дорожнього трафіку	23
1.4.2 Динамічні характеристики дорожнього трафіку.....	24
1.4.3 Методи аналізу дорожнього трафіку.....	25
1.5 Визначення статичних та динамічних характеристик автомобільного потоку	26
1.6 Постановка задачі	29
Висновки до розділу	29
2 РОЗРОБКА МЕТОДУ ОЦІНЮВАННЯ СТАНУ ДОРОЖНЬОГО ПОЛОТНА	31
2.1 Розробка підходу автоматичного визначення дорожнього полотна 31	
2.1.1 Детекція ліній на зображенні	31
2.1.2 Визначення ліній, обмежуючих дорожнє полотно	35
2.1.3 Детекція за допомогою сегментаційної нейромережі	39
2.2 Мультикласовий показник TLCR	41
2.2.1 Визначення класів для оцінки мультикласового TLCR	41
2.2.2 Визначення мультикласового TLCR	43
2.2.3 Модифікований мультикласовий TLCR (M2TLCR).....	45
2.3 Визначення статичних та динамічних характеристик	46

2.3.1	Визначення статичних характеристик потоку.....	48
2.3.2	Визначення динамічних характеристик	49
	Висновки до розділу	50
3	РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	53
3.1	Моделювання програмного засобу для оцінки характеристик транспортного потоку.....	53
3.1.1	Архітектура системи	53
3.1.2	Алгоритмічна модель	55
3.1.3	Взаємодія компонентів.....	56
3.2	Реалізація програмного забезпечення	57
3.2.1	Модуль визначення дорожнього полотна.....	57
3.2.2	Модуль виявлення та відстеження транспортних засобів.....	59
3.2.3	Модулі обчислення характеристик транспортного потоку.....	60
3.3	Оцінка ефективності розробленого програмного забезпечення.....	61
	Висновки до розділу	62
4	МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЄКТУ.....	64
4.1	Опис ідеї проєкту	64
4.2	Технологічний аудит ідеї проєкту	65
4.3	Аналіз ринкових можливостей запуску стартап-проєкту	66
4.4	Розроблення ринкової стратегії проєкту	72
4.5	Розроблення маркетингової програми стартап-проєкту.....	74
	Висновки до розділу	77
	ВИСНОВКИ	78
	СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	81
	ДОДАТОК А.....	83
	ДОДАТОК Б	105

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

TЗ – транспортний засіб;

ПЗ – програмне забезпечення;

LSD (Line Segment Detector) – алгоритм виявлення ліній на зображенні;

TLCR (Traffic Lane Congestion Ratio) – коефіцієнт завантаженості смуги транспортного руху;

MTLCR (Modified Traffic Lane Congestion Ratio) – модифікований коефіцієнт

M2TLCR (Multiclass Modified Traffic Lane Congestion Ratio) – модифікований мультикласовий коефіцієнт завантаженості смуги транспортного руху;

YOLO (You Only Look Once) – алгоритм виявлення об'єктів на зображеннях та відео, який виконує виявлення, використовуючи лише один прохід нейронної мережі;

BEV (Bird's Eye View) – проєкція «вигляд з висоти пташиного польоту», ортогональне перетворення зображення для усунення перспективних спотворень;

mAP (mean Average Precision) – середня точність, основна метрика для оцінки якості роботи алгоритмів детекції об'єктів (використовувалася в таблицях результатів навчання);

FPS (Frames Per Second) – кількість кадрів за секунду, одиниця вимірювання частоти оновлення відеопотоку та швидкодії системи;

OpenCV – бібліотека комп'ютерного зору та обробки зображень, яка надає інструменти для зчитування, обробки, аналізу та візуалізації зображень та відео.

ВСТУП

Актуальність теми. У сучасних умовах швидкої урбанізації та зростання кількості транспортних засобів ефективне управління дорожнім рухом є одним із ключових чинників стабільного функціонування міської інфраструктури. Автомобільні дороги виступають «артеріями» міста, від яких безпосередньо залежить мобільність населення, економічна активність і рівень безпеки. Помилки у плануванні транспортних потоків призводять до перевантаження окремих ділянок, виникнення заторів, збільшення часу поїздок і підвищення рівня аварійності.

У зв'язку з цим, актуальним є створення інтелектуальних систем відеомоніторингу, здатних у режимі близькому до реального часу аналізувати стан транспортного потоку, фіксувати ключові характеристики (інтенсивність, щільність, швидкість, напрямок руху, типи транспортних засобів) і забезпечувати прийняття обґрунтованих управлінських рішень.

Попри значні досягнення в цій сфері, існуючі методи моніторингу мають низку обмежень. Часто вони зосереджені лише на окремих параметрах руху, потребують попереднього калібрування камер, не забезпечують достатньої адаптивності до змін умов освітлення, погодних факторів або перспективи. Крім того, більшість підходів не дозволяють комплексно оцінювати динаміку дорожнього трафіку та його структуру в просторі. Це створює потребу у вдосконаленні методів і розробці програмних засобів, здатних об'єднати різні джерела інформації та забезпечити надійний аналіз відеоданих у реальному часі.

Мета дослідження. Основною метою є підвищення точності, швидкодії та масштабованості аналізу результатів відеоспостереження для подальшого використання в системах управління транспортною інфраструктурою.

Завдання дослідження. Для досягнення поставленої мети необхідно розв'язати такі завдання:

- виконати аналіз сучасних методів та програмних рішень для відеомоніторингу дорожнього руху;
- визначити вимоги до системи відеомоніторингу транспортного потоку;

- розробити підхід автоматичного визначення маски дорожнього полотна;
- створити мультикласовий показник для оцінювання характеристик транспортного потоку;
- створити єдину модель для оцінки статичних та динамічних показників транспортного потоку;
- розробити програмне забезпечення для автоматизованого відеомоніторингу транспортного руху;
- оцінити ефективність та точність роботи запропонованих методів і програмних засобів.

Об’єкт дослідження - програмне забезпечення відеомоніторингу автомобільного транспортного потоку.

Предмет дослідження - підходи, методи та програмний засіб обробки та аналізу відеоданих для визначення характеристик транспортного потоку.

Наукова новизна результатів магістерської дисертації полягає в тому, що:

- *вперше* розроблено метод автоматичного визначення дорожнього полотна в режимі, близькому до реального часу, який об’єднує виявлення ліній розмітки, геометричну фільтрацію, аналіз перспективної трансформації та формування полігональної маски дороги;
- *вдосконалено* показники TLCR та MTLCR для більш якісної оцінки завантаженості дорожнього полотна, за рахунок урахування різних типів транспортних засобів та їхнього впливу на пропускну здатність;
- *вперше* запропоновано єдиний підхід до визначення статичних та динамічних характеристик транспортного потоку на основі однієї моделі об’єктної сегментації, яка одночасно визначає маски транспортних засобів, їх класи, положення та обмежувальні рамки.

Практичне значення отриманих результатів полягає в розробці програмного засобу для визначення характеристик транспортного потоку в режимі близькому до реального часу та може використовуватися для оптимізації

роботи світлофорів, прогнозування заторів, аналізу інфраструктури й статистичного моніторингу. Крім того, запропонований підхід може бути застосований і для інших типів руху.

1 АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ

1.1 Опис предметної області

Сучасні транспортні системи великих міст характеризуються високою динамічністю та складністю структури транспортних потоків. Кількість автомобілів постійно зростає, що призводить до підвищеного навантаження на дорожню інфраструктуру, зниження швидкості руху, виникнення заторів та підвищення аварійності. Ефективне управління дорожнім рухом неможливе без оперативного та достовірного моніторингу стану транспортних потоків.

Відеомоніторинг дорожнього руху є одним із найперспективніших підходів до збору інформації про транспортні потоки. На відміну від традиційних методів збору даних (індукційних петель, радарів, ультразвукових сенсорів тощо), відеоспостереження дозволяє отримувати значно ширший спектр характеристик: не лише кількість транспортних засобів, а й їх типи, швидкість, напрям руху, щільність на різних ділянках дороги, поведінкові патерни водіїв тощо. Крім того, відеодані можуть бути оброблені повторно для вирішення нових аналітичних завдань, без необхідності додаткових вимірювань.

Основою сучасних систем відеомоніторингу є алгоритми комп'ютерного зору та штучного інтелекту, які забезпечують автоматичну обробку відеопотоку в реальному часі. Залежно від мети аналізу, виділяють декілька ключових напрямів обробки відеоданих:

- детекція транспортних засобів, що дозволяє визначити факт присутності об'єктів у кадрі;
- класифікація, яка розподіляє транспортні засоби за типами (легкові, вантажні, автобуси, мотоцикли тощо);
- сегментація дорожнього полотна, необхідна для відокремлення зони руху від фону;
- відстеження об'єктів (tracking), яке забезпечує визначення траєкторій і швидкостей;
- розрахунок характеристик транспортного потоку - інтенсивності, завантаженості, середньої швидкості, щільності тощо.

Залежно від використовуваних алгоритмів, системи відеомоніторингу умовно поділяють на три групи:

- класичні методи обробки зображень (rule-based, OpenCV-підходи);
- методи глибинного навчання (Deep Learning);
- гібридні підходи.

Класичні методи обробки зображень базуються на фільтрації, пороговій сегментації, аналізі контурів, руху та оптичного потоку. Такі методи мають низькі вимоги до обчислювальних ресурсів і забезпечують стабільність у простих умовах, але є чутливими до змін освітлення, тіней і погодних факторів.

Методи глибинного навчання (Deep Learning) — використовують нейронні мережі для виявлення, класифікації та сегментації. Найбільш відомими є архітектури YOLO [6], Faster R-CNN [7], Mask R-CNN [8] тощо. Вони демонструють високу точність, проте потребують потужного апаратного забезпечення та ретельного навчання.

Гібридні підходи поєднують переваги обох груп, а саме: використання OpenCV [4] для базової попередньої обробки (нормалізація, фільтрація шумів) і нейронних мереж для високорівневого аналізу (сегментація, детекція, визначення характеристик об'єктів).

У межах даного дослідження основна увага приділяється автоматичному визначенню дорожнього полотна за допомогою методів комп'ютерного зору (OpenCV), а також розробці мультикласового показника TLCR (Traffic Lane Congestion Ratio), який дозволяє враховувати не лише загальну кількість транспортних засобів на смузі, але й тип кожного з них, що підвищує точність оцінки завантаженості дороги.

Додатково, у роботі досліджується використання об'єктної сегментації для комплексного аналізу відеокадру. Такий підхід дає змогу за один прохід нейронної мережі визначати усі транспортні засоби на зображенні з окремими масками, з яких можна побудувати єдину сегментаційну маску та окремо розрахувати характеристики для кожного об'єкту (наприклад швидкість).

Застосування об'єктної сегментації дозволяє відмовитись від використання окремих моделей для семантичної сегментації та виявлення

об'єктів, що суттєво підвищує ефективність обчислень і спрощує інтеграцію алгоритмів у реальні системи відеомоніторингу.

Таким чином, предметна область даного дослідження охоплює розробку методів автоматизованої обробки відеоданих дорожнього руху, які базуються на поєднанні класичних і нейромережевих підходів комп'ютерного зору, з метою отримання повної картини стану транспортного потоку — його структури, інтенсивності та завантаженості.

1.2 Автоматичне визначення площини дорожнього покриття

Одним із ключових етапів побудови систем відеоаналізу транспортного потоку є визначення площини дорожнього покриття. Саме на основі правильно виділеної області дороги відбувається подальша детекція транспортних засобів, розрахунок їх швидкостей, інтенсивності руху та показників завантаженості. Помилки у визначенні площини дорожнього полотна призводять до спотворення отриманих характеристик, особливо у випадках, коли камера розташована під значним кутом або коли зображення має сильну перспективну деформацію. Тому автоматизація процесу виділення дорожнього полотна є необхідною умовою для підвищення точності аналізу трафіку та стабільності роботи системи в реальних умовах.

Перші підходи до визначення області дороги базувалися на використанні статичних шаблонів або ручного виділення області інтересу на зображенні. Такі методи вимагали попереднього калібрування кожної камери та не могли адаптуватися до змін умов зйомки, освітлення чи положення камери. У подальшому, з розвитком алгоритмів комп'ютерного зору, з'явилися методи, які дозволяють автоматизувати цей процес. Одним із базових підходів є використання аналізу кольорової гами зображення. Дорожнє покриття, як правило, має відносно однорідний колір і текстуру, тому за допомогою методів кластеризації, таких як K-means або GrabCut [9], можна виділити область, що відповідає дорозі. Проте такий підхід виявився недостатньо стабільним через сильну залежність від освітлення, погодних умов та відмінностей у типі дорожнього покриття.

Більш надійні результати забезпечують методи, засновані на

геометричному аналізі сцени. Вони використовують припущення про те, що лінії розмітки та межі дороги мають певні геометричні властивості, зокрема напрямки, які сходяться до точки горизонту. Одним із найпоширеніших підходів є використання перетворення Гафа [10] для виявлення ліній розмітки. На основі знайдених ліній можна відновити межі дорожнього полотна, а також оцінити положення горизонту. Якщо відома координата горизонту, зображення можна трансформувати до умовно ортогональної проекції за допомогою гомографічного перетворення. Це дозволяє перевести зображення з перспективного виду у вигляд, максимально наближений до вигляду зверху, що суттєво спрощує подальші обчислення площ, швидкостей і відстаней у піксельному просторі.

У бібліотеці OpenCV реалізовано широкий набір інструментів для виконання таких геометричних операцій. Зокрема, функції `cv2.findHomography()` та `cv2.warpPerspective()` дозволяють виконати проекційне перетворення з використанням матриці гомографії, яка обчислюється на основі відповідностей між точками у вихідному та цільовому зображеннях. Для автоматичного визначення цих точок можуть застосовуватись алгоритми виявлення особливостей, такі як SIFT [11], ORB [12] або Harris Corner Detector [13], які знаходять характерні орієнтири на зображенні (краї розмітки або бордюри) – ключові точки. Це дає можливість адаптувати процес навіть для різних типів доріг і різних умов освітлення.

У межах даної роботи метод автоматичного визначення дорожнього полотна ґрунтується на комбінації підходів, що поєднують геометричний аналіз сцени з класичними алгоритмами комп'ютерного зору. Попередня обробка кадру виконується засобами бібліотеки OpenCV — для згладжування шумів, підвищення контрастності та виділення границь. Далі застосовується алгоритм пошуку ліній дорожньої розмітки за допомогою детектора ліній, після чого отримані контури використовуються для орієнтовного визначення межі дорожнього полотна. Такий підхід забезпечує високу точність визначення дорожнього полотна навіть у складних умовах — при наявності тіней, зміні яскравості або частковому перекритті дороги транспортними засобами.

Визначення площини дорожнього покриття також є необхідним етапом для подальшого перетворення зображення до метрики, у якій піксельні відстані мають фізичний сенс. Застосування гомографічного перетворення дозволяє перетворити перспективне зображення у вигляд, наближений до ортогональної проекції. Це створює основу для точного розрахунку таких параметрів, як швидкість транспортних засобів, площа зайнятої дороги та мультикласовий показник TLCR. Таким чином, автоматичне визначення площини дорожнього покриття є критично важливою частиною процесу аналізу трафіку, оскільки воно забезпечує просторову узгодженість усіх подальших вимірювань.

Сучасні методи визначення дорожнього полотна еволюціонували від простих алгоритмів колірної фільтрації до гібридних рішень, які поєднують геометричний аналіз, класичні алгоритми OpenCV та гомографічні перетворення. Такий підхід дозволяє створювати системи, що здатні автоматично адаптуватися до умов середовища, забезпечуючи високу точність і стабільність визначення області дороги в реальному часі. Отримані результати є фундаментом для подальшого розрахунку характеристик транспортного потоку, таких як інтенсивність, завантаженість і швидкість руху транспортних засобів.

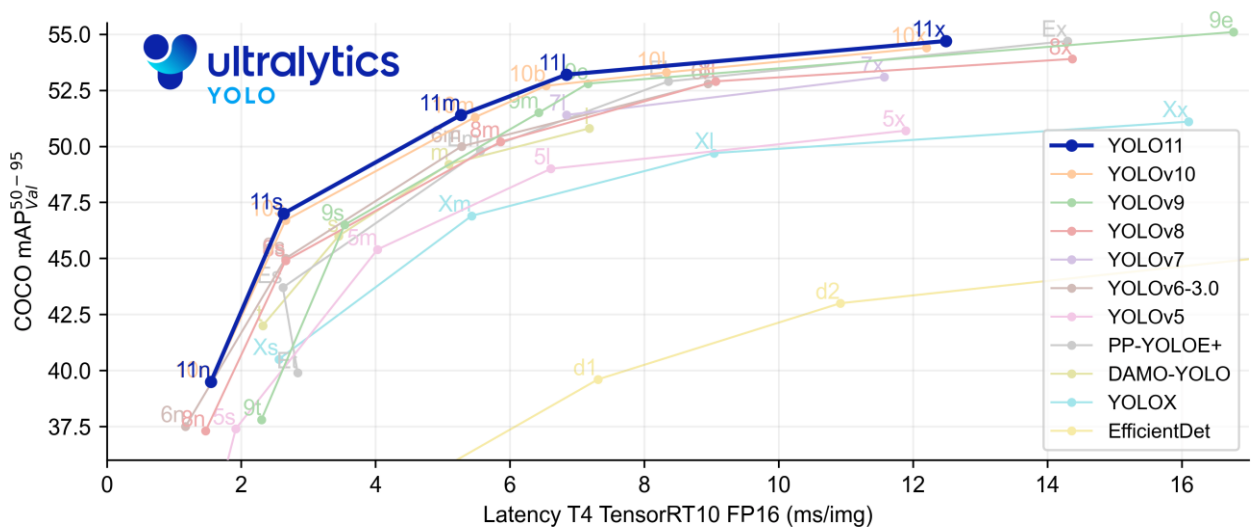
1.3 Методи виявлення транспортних засобів

Одним із ключових завдань у процесі відеоаналізу транспортного потоку є детекція транспортних засобів. Саме на цьому етапі формується базова інформація, що надалі використовується для відстеження руху, класифікації об'єктів, розрахунку швидкостей та визначення таких характеристик, як інтенсивність і завантаженість дороги. Від точності виявлення транспортних засобів безпосередньо залежить надійність усієї системи відеомоніторингу.

Перші підходи до виявлення транспортних засобів базувалися на класичних методах комп'ютерного зору, які використовували евристичні алгоритми обробки зображень. Основною ідеєю таких методів було порівняння послідовних кадрів відеопотоку для виявлення рухомих об'єктів. Найбільш поширеним підходом був метод фонові різниці, який дозволяв виділяти змінні області зображення шляхом порівняння поточного кадру з фоновим, що оновлюється в процесі спостереження. Для уточнення меж об'єктів

використовували методи виявлення контурів, такі як оператори Canny або Sobel, а також морфологічні перетворення для очищення результатів від шумів. У деяких випадках застосовувався аналіз оптичного потоку, який дозволяє оцінити напрям і швидкість руху пікселів. Реалізація таких методів можлива за допомогою бібліотеки OpenCV, що забезпечує широкий набір інструментів для обробки зображень і відео.

Класичні алгоритми мали суттєву перевагу у швидкодії та простоті реалізації. Вони не потребували навчання моделей, а тому могли застосовуватись на обмежених апаратних ресурсах. Однак вони виявилися надто чутливими до змін освітлення, погодних умов, тіней, а також не могли коректно відокремлювати об'єкти, що перекриваються. Через це їх точність значно знижувалася у складних реальних умовах, таких як багатосмуговий рух або перехрестя. З розвитком апаратних засобів та методів штучного інтелекту з'явилися більш точні рішення, засновані на глибинних нейронних мережах.



високу швидкодію при збереженні високої точності, що робить їх придатними для роботи в режимі близькому до реального часу. У свою чергу, Mask R-CNN і Detectron2 [13] дозволяють не лише виявляти транспортні засоби, а й виконувати сегментацію їх контурів, що є важливою властивістю для точного визначення площі зайнятої дороги. Сучасні універсальні моделі, такі як Segment Anything Model, надають можливість виконувати сегментацію без попереднього навчання під конкретні дані, що відкриває шлях до створення більш гнучких і адаптивних систем.

Попри високу точність, методи глибинного навчання мають і певні обмеження. Вони потребують значних обчислювальних ресурсів, що ускладнює їх використання в системах з обмеженими апаратними можливостями. Крім того, вони часто вимагають великих обсягів даних для навчання та ретельного налаштування гіперпараметрів. Тому у практичних рішеннях часто використовують комбіновані підходи, які поєднують класичні алгоритми обробки зображень з нейронними мережами.

Гібридні методи дозволяють досягати компромісу між точністю та швидкодією. Зазвичай класичні методи використовуються для попередньої обробки кадрів, наприклад для визначення області дорожнього полотна, зменшення шумів або нормалізації яскравості, тоді як нейронні мережі відповідають за точну детекцію транспортних засобів. Такий підхід забезпечує зменшення кількості хибних спрацьовувань і підвищує загальну стабільність системи, особливо в умовах змінної освітленості та складного фону.

Серед сучасних напрямів розвитку методів виявлення окреме місце займає об'єктна сегментація, яка поєднує переваги виявлення та семантичної сегментації. На відміну від звичайних моделей, що визначають лише прямокутні рамки навколо об'єктів, об'єктна сегментація дозволяє отримати точні контури кожного транспортного засобу окремо. Це особливо важливо для задач, де необхідно розраховувати площу, яку займає транспорт на смузі руху, або визначати рівень завантаженості дорожнього полотна. У межах даної роботи об'єктна сегментація використовується як універсальний інструмент, що дозволяє одночасно визначати транспортні засоби різних типів, а також

розраховувати їх швидкість і мультикласовий показник TLCR. Такий підхід усуває необхідність застосування окремих моделей для семантичної сегментації, виявлення та трекінгу, що знижує обчислювальні витрати та підвищує ефективність аналізу.

Об'єктна сегментація забезпечує комплексну оцінку транспортного потоку за один прохід нейронної мережі. Вона дозволяє поєднати визначення статичних і динамічних характеристик трафіку, що створює передумови для побудови єдиної системи відеомоніторингу, здатної працювати в реальному часі. Поєднання об'єктної сегментації з попередньою обробкою зображень засобами OpenCV дає можливість створити універсальну систему, яка є стійкою до змін перспективи, освітлення і погодних умов. Таким чином, сучасні методи виявлення транспортних засобів поступово переходять від роздільних спеціалізованих алгоритмів до інтегрованих рішень, які поєднують точність нейронних мереж із ефективністю класичних методів комп'ютерного зору.

1.4 Аналіз характеристик дорожнього трафіку

Аналіз дорожнього трафіку є ключовим елементом сучасних інтелектуальних транспортних систем. Його метою є визначення характеристик транспортного потоку — інтенсивності, щільності, швидкості, завантаженості смуг руху, типів транспортних засобів, напрямків руху та інших параметрів, які описують поточний стан дороги та дозволяють прогнозувати його зміни.

Серед існуючих параметрів, що описують стан дорожнього трафіку, виділяють два основних типи: статичні параметри, які характеризують ступінь заповненості дороги транспортними засобами та динамічні параметри, котрі описують інтенсивність руху, швидкість та зміни трафіку в часі.

1.4.1 Статичні характеристики дорожнього трафіку

Статичні параметри описують стан дороги у певний момент часу. Основним серед них є ступінь завантаженості смуги руху, тобто частка дорожнього полотна, зайнята транспортними засобами. Класичним способом вимірювання цього показника є визначення щільності руху — кількості автомобілів на одиницю довжини дороги або відсотка площі зображення, яку займають транспортні засоби.

Проте такі оцінки мають абсолютний характер, тобто залежать від геометрії дороги, кута огляду камери та реальних розмірів транспортних засобів. Це ускладнює порівняння результатів між різними ділянками. З цією метою запропоновано відносні метрики, зокрема показник TLCR, який оцінює відношення площі транспортних засобів до площі смуги руху, незалежно від ракурсу зйомки чи перспективи.

У межах даного дослідження запропоновано вдосконалений мультикласовий показник TLCR [2], який враховує типи транспортних засобів та їх реальний вплив на завантаженість. Це забезпечує більш точне відображення ситуації на ділянках дороги зі змішаним потоком (наприклад, де одночасно рухаються легкові автомобілі, вантажівки та автобуси).

1.4.2 Динамічні характеристики дорожнього трафіку

До динамічних характеристик належать показники, що відображають рух транспорту у часі. Найбільш поширеним серед них є показник AADT [14] (Average Annual Daily Traffic) — середньодобовий річний трафік на певній ділянці дороги. Він визначається як кількість автомобілів, що проїхали ділянку за рік, поділена на кількість днів у році. Показник AADT дозволяє оцінити ступінь завантаженості дороги у середньому за рік і є корисним для аналізу транспортної інфраструктури в стратегічному плануванні.

Однак цей підхід має обмеження — він не враховує часові коливання інтенсивності руху. У різні пори року, дні тижня або години доби транспортна ситуація може суттєво відрізнитись. Тому для оперативного управління дорожнім рухом використовують динамічні методи оцінювання інтенсивності, коли інтервал вимірювання скорочується до однієї години або навіть хвилини. Такий підхід дозволяє формувати актуальний профіль трафіку, що відображає зміни в реальному часі.

Іншим важливим динамічним параметром є швидкість руху транспортних засобів. Вона безпосередньо впливає на пропускну здатність дороги та є основним показником ефективності транспортного потоку. Використовуючи дані з відеокамер, можна визначати не лише середню швидкість руху, а й розподіл швидкостей між різними типами транспортних засобів, що дає змогу

виявляти потенційно небезпечні зони, де, наприклад, водії не знижують швидкість перед поворотами чи пішохідними переходами.

Завдяки сучасним методам комп'ютерного зору швидкість і напрям руху можна визначати автоматично, аналізуючи зсув транспортних засобів між послідовними кадрами відео. Методи трекінгу (наприклад, DeepSORT, ByteTrack) дозволяють прив'язувати об'єкти до унікальних ідентифікаторів і відстежувати їхні траєкторії з високою точністю.

Важливою динамічною характеристикою є також класифікація транспортних засобів за категоріями. За допомогою методів комп'ютерного зору можна розділити транспортні засоби на класи — легкові, вантажні, автобуси, мотоцикли тощо. Така класифікація дозволяє деталізувати аналіз транспортного потоку, оскільки кожен тип транспорту має різний вплив на пропускну здатність і структуру трафіку. На основі цього підходу розробляються мультикласові показники, зокрема TLCR (Traffic Lane Congestion Ratio), який враховує не лише кількість транспортних засобів, а й їх типи та вплив на смузи руху.

1.4.3 Методи аналізу дорожнього трафіку

Сучасні системи аналізу трафіку можна класифікувати за типом використовуваних технологій на три основні групи:

- а) сенсорні методи - використовують спеціалізовані пристрої — індукційні петлі, радары, інфрачервоні або ультразвукові сенсори. Вони забезпечують високу точність у точкових вимірах, але мають обмежену інформаційну ємність і вимагають значних фінансових та технічних витрат на встановлення.
- б) відеоаналітичні методи - базуються на аналізі відеоданих із камер спостереження. Забезпечують отримання комплексної інформації про транспортний потік — кількість, типи, швидкість і траєкторії руху транспортних засобів. Зокрема серед відеоаналітичних підходів виділяють:

- 1) класичні методи обробки зображень: фільтрація, порогова сегментація, фонове віднімання, аналіз оптичного потоку, перетворення Гафа;

- 2) методи глибинного навчання (YOLO, Mask R-CNN), що забезпечують високу точність виявлення об'єктів;
- 3) гібридні підходи, які поєднують класичні алгоритми попередньої обробки з нейронними моделями для високорівневої інтерпретації даних;
- 4) комбіновані (інтегровані) методи - поєднують переваги попередніх двох підходів і дозволяють проводити комплексний аналіз у реальному часі.

У межах даної роботи пропонується інтегрований метод, який поєднує алгоритми OpenCV для автоматичного визначення дорожнього полотна з об'єктною сегментацією, що дозволяє за один прохід визначати: межі дороги, усі транспортні засоби в кадрі, їх швидкість і напрям руху, значення мультикласового показника TLCR [2].

Такий підхід забезпечує одночасне визначення як статичних, так і динамічних характеристик трафіку, що робить систему універсальною та придатною для використання у задачах моніторингу в реальному часі.

1.5 Визначення статичних та динамічних характеристик автомобільного потоку

Визначення характеристик дорожнього трафіку є одним із ключових завдань систем відеомоніторингу, оскільки саме від правильності розрахунку таких параметрів залежить точність оцінки стану транспортної системи. Усі показники, що описують дорожній рух, умовно поділяють на статичні та динамічні. Статичні характеризують просторовий стан дорожнього полотна у конкретний момент часу, тоді як динамічні відображають зміну цього стану в часі. Для комплексного аналізу необхідно враховувати обидва типи характеристик, адже лише у поєднанні вони дозволяють сформувати повну картину поведінки транспортного потоку.

До основних динамічних характеристик трафіку належать інтенсивність руху та швидкість транспортних засобів. Інтенсивність описує кількість автомобілів, що перетнули певну ділянку дороги за фіксований проміжок часу. Цей параметр є фундаментальним при проектуванні та оцінюванні ефективності

транспортних розв'язок, оскільки дозволяє встановити залежність між кількістю транспортних засобів, пропускнуою здатністю дороги та рівнем заторів. Швидкість руху, у свою чергу, визначає рівень динамічності потоку та опосередковано впливає на показники транспортної безпеки. У класичних підходах ці параметри визначалися за допомогою спеціалізованих сенсорів, таких як індукційні петлі, радарні датчики або лазерні вимірювачі, що встановлювались безпосередньо на дорожньому полотні. Такі рішення забезпечували високу точність, однак мали істотні недоліки — високу вартість, складність обслуговування та низьку гнучкість при зміні конфігурації спостереження.

З розвитком комп'ютерного зору з'явилася можливість виконувати розрахунок динамічних параметрів на основі даних з відеокамер. Одним із найпростіших способів визначення інтенсивності є підрахунок кількості транспортних засобів, які перетнули контрольну лінію або область інтересу за певний інтервал часу. Для цього використовуються методи виявлення об'єктів і трекінгу, що дозволяють відстежувати траєкторію руху кожного автомобіля. На основі зміни координат центру об'єкта між послідовними кадрами визначається його швидкість. Такі методи добре працюють для камер, розташованих перпендикулярно або під невеликим кутом до дорожнього полотна, проте втрачають точність при зміні перспективи. Тому в сучасних підходах дедалі частіше застосовуються алгоритми геометричної корекції перспективи, що дозволяють перетворити зображення в умовно ортогональну площину, де вимірювання відстаней і швидкостей набуває більшої точності.

Паралельно з цим розвивається напрям використання об'єктної сегментації для оцінки параметрів трафіку. На відміну від традиційних методів підрахунку транспортних засобів, які оперують лише прямокутними рамками, об'єктна сегментація дозволяє отримати точну форму кожного автомобіля, що дає можливість визначати площу, зайняту транспортними засобами на смузі руху. Саме цей підхід став основою для розрахунку статичних характеристик дорожнього трафіку, зокрема показників завантаженості. Одним з методів такого типу є показник TLCR [2] (Traffic Lane Congestion Ratio), який характеризує

ступінь заповненості смуги руху транспортними засобами. Його значення обчислюється як відношення кількості пікселів, що належать транспортним засобам, до загальної кількості пікселів у межах виділеної смуги. Таким чином, показник TLCR є відносною величиною, яка дозволяє оцінити ступінь завантаженості незалежно від геометричних розмірів ділянки дороги.

Подальший розвиток методу TLCR привів до появи його модифікованих версій, що враховують вплив перспективи, різницю у ширині транспортних засобів і типи об'єктів. Використання об'єктної сегментації забезпечує можливість розрахунку цього показника автоматично, у межах одного процесу аналізу відеопотоку. Таким чином, стає можливим одночасно отримувати як статичні, так і динамічні характеристики трафіку — кількість, швидкість і площу зайнятого простору.

Окрім цього також існують дослідження з поєднання статичних і динамічних параметрів у єдину систему показників. Одним із таких підходів є визначення інтегрального показника TLIR (Traffic Line Intensity Ratio), який об'єднує інформацію про швидкість руху транспортних засобів і значення TLCR, що дозволяє формувати більш повну оцінку стану трафіку на певній ділянці дороги. Такий підхід є особливо корисним для аналізу змін інтенсивності руху в реальному часі, а також для прогнозування можливого утворення заторів.

Серед методів отримання статичних характеристик найбільш поширеними залишаються ті, що базуються на аналізі заповненості дорожнього полотна транспортними засобами. Застосування алгоритмів комп'ютерного зору дозволяє виконувати цю оцінку без використання додаткових сенсорів, а лише на основі даних з відеокамер. При цьому важливим етапом є коректне визначення області дороги, адже неточності у виділенні дорожнього полотна призводять до похибок у розрахунках. Для цього використовуються методи геометричної сегментації, фільтрації та попереднього аналізу зображення з використанням бібліотеки OpenCV.

Таким чином, сучасні методи визначення характеристик дорожнього трафіку базуються на інтеграції класичних алгоритмів комп'ютерного зору з нейронними моделями виявлення та сегментації. Це відкриває можливість

створення ефективних програмних систем відеомоніторингу, що здатні працювати у реальному часі та забезпечувати високий рівень автоматизації при аналізі транспортних потоків у сучасних міських умовах.

1.6 Постановка задачі

На основі проведеного аналізу існуючих методів і підходів до відеомоніторингу транспортного потоку встановлено, що більшість сучасних систем демонструють високу точність при виявленні транспортних засобів, проте мають низку обмежень, пов'язаних із недостатньою автоматизацією визначення дорожнього полотна, неврахуванням типів транспортних засобів при оцінці завантаженості та складністю інтеграції алгоритмів у системи реального часу.

У зв'язку з цим у даній роботі ставиться задача розроблення методу та програмного засобу відеомоніторингу транспортного потоку, який забезпечує:

- автоматичне визначення площини дорожнього покриття засобами OpenCV;
- детекцію транспортних засобів у відеопотоці з використанням алгоритмів глибинного навчання (YOLO);
- класифікацію транспортних засобів за типами;
- розрахунок мультикласового показника TLCR, що враховує відносну площу зайнятої смуги руху з урахуванням типів транспортних засобів;
- формування візуальної аналітики у реальному часі для подальшого використання у транспортних інформаційних системах.

Висновки до розділу

У першому розділі проведено аналіз існуючих методів відеомоніторингу транспортного потоку, розглянуто сучасні підходи до виявлення транспортних засобів, визначення характеристик дорожнього трафіку та автоматичного виділення дорожнього полотна.

Показано, що класичні алгоритми комп'ютерного зору забезпечують високу швидкодію, однак є чутливими до зовнішніх умов. Методи глибинного навчання демонструють високу точність, проте потребують значних обчислювальних ресурсів. Найбільш ефективними є гібридні підходи, які

поєднують переваги обох груп методів.

Проаналізовано параметри, що характеризують дорожній трафік — статичні (завантаженість, щільність) і динамічні (інтенсивність, швидкість). Особливу увагу приділено показникам TLCR та його модифікаціям, запропонованим у роботах [2] і [3], які заклали основу для подальшого розвитку підходів до оцінки завантаженості смуг руху.

У результаті сформульовано задачу створення ефективного програмного рішення для автоматизованого відеомоніторингу транспортного потоку, що поєднує класичні алгоритми OpenCV з методами глибинного навчання для отримання точних і стабільних оцінок характеристик дорожнього руху.

2 РОЗРОБКА МЕТОДУ ОЦІНЮВАННЯ СТАНУ ДОРОЖНЬОГО ПОЛОТНА

2.1 Розробка підходу автоматичного визначення дорожнього полотна

Одним із ключових етапів побудови системи відеомоніторингу транспортного потоку є автоматичне визначення області дорожнього полотна. Коректне виділення цієї області забезпечує основу для подальшого аналізу транспортних засобів, розрахунку швидкості, інтенсивності руху та показників завантаженості. У даному розділі описано процес розробки методів автоматичної виявлення дорожнього полотна, який включає попереднє оброблення відеозображень, виявлення ліній дорожньої розмітки та формування полігональної області дороги. Додатково проведено тестування можливостей об'єктної сегментації для автоматичного визначення дороги.

2.1.1 Детекція ліній на зображенні

На першому етапі для забезпечення стійкої роботи алгоритмів виявлення ліній було реалізовано попереднє оброблення зображень (pre-processing). Його метою є зменшення впливу шумів, тіней, перепадів освітленості та покращення контрастності, що є критично важливим для подальшої роботи детектора ліній.

Для виявлення ліній використано алгоритм LSD [15] (Line Segment Detector), реалізований у бібліотеці OpenCV. На відміну від класичного перетворення Гафа, LSD не потребує попередньої бінаризації зображення, а працює безпосередньо зі значеннями яскравості. Алгоритм аналізує локальні градієнти та об'єднує послідовності пікселів із подібним напрямом у прямолінійні сегменти. Кожна знайдена лінія представлена у вигляді двох кінцевих точок, що дозволяє легко оперувати з їх геометричними параметрами. LSD забезпечує високу точність при низьких обчислювальних витратах, що робить його придатним для використання в системах, які працюють у реальному часі.

У роботі було протестовано та порівняно кілька методів попереднього процесингу, кожен з яких буде описано нижче.

2.1.1.1 Метод вирівнювання гістограми.

Метод базується на застосуванні глобального вирівнювання гістограми

яскравості, яке забезпечує покращення контрасту зображення. Перед цим використовується гауссове розмивання з ядром 3×3 , щоб зменшити вплив локальних шумів. Суть методу полягає у перетворенні гістограми яскравості у рівномірний розподіл, завдяки чому темні ділянки стають світлішими, а занадто яскраві — темнішими.

Результатом є зображення з підвищеним контрастом і покращеною видимістю меж (рис. 2.1), що особливо важливо для виявлення розмітки у тіньових або слабо освітлених ділянках дороги.

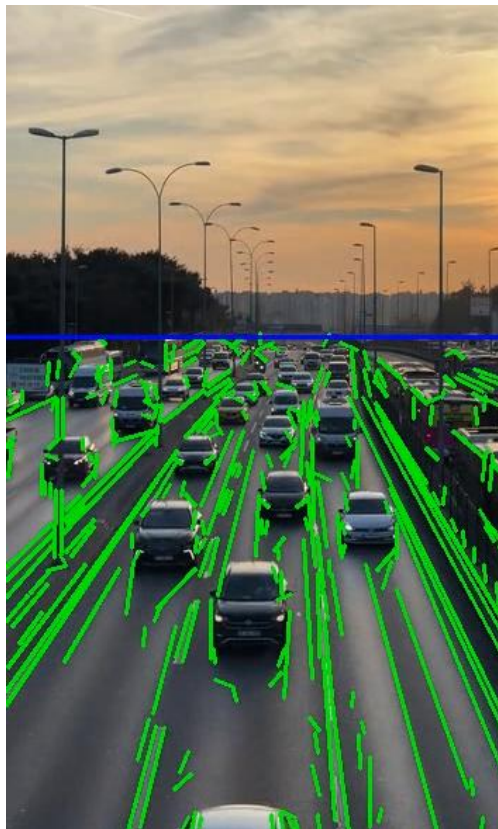


Рисунок 2.1 – Детектор ліній на зображенні з вирівнюванням гістограми.

2.1.1.2 Метод посилення різкості.

Цей метод реалізує підсилення контурів на основі оператора Лапласа [16], який виділяє ділянки з великим перепадом яскравості (градієнтні області). Лапласіан визначається як друга похідна зображення: Високі значення другої похідної відповідають границям об'єктів.

У коді обчислюється лапласіан зображення, після чого його додають до вихідного кадру, що призводить до збільшення контрасту локальних деталей. Отримане зображення має чіткіше виражені краї, що підвищує точність виявлення ліній навіть за слабкої розмітки (рис. 2.2).

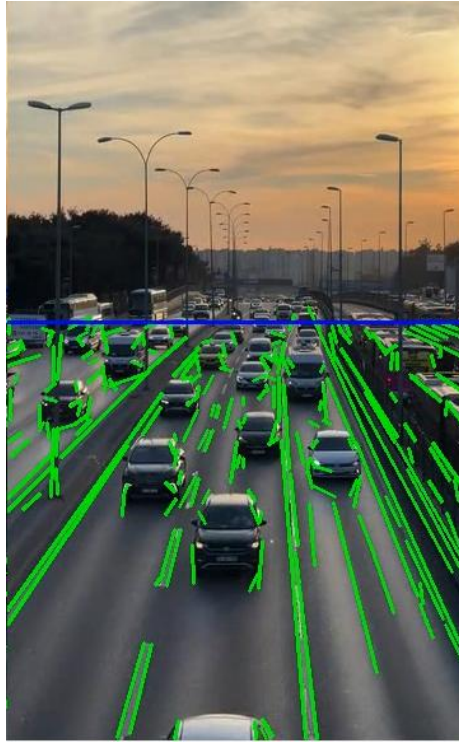


Рисунок 2. 2 – Детектор ліній на зображенні з посиленням різкості.

2.1.1.3 Метод видалення фону.

Цей метод спрямований на усунення неоднорідного фону та локальне вирівнювання яскравості. Основна ідея — відняти з оригінального зображення його розмиту версію, отриману за допомогою гауссового фільтру великого розміру. Цей прийом відомий як топ-хет (top-hat) або background subtraction.

У результаті залишаються лише високочастотні компоненти — тобто елементи, які відрізняються від середнього фону. Таке зображення є зручним для подальшого виявлення ліній, оскільки зникають повільні перепади яскравості, спричинені освітленням чи тінями (рис. 2.3).

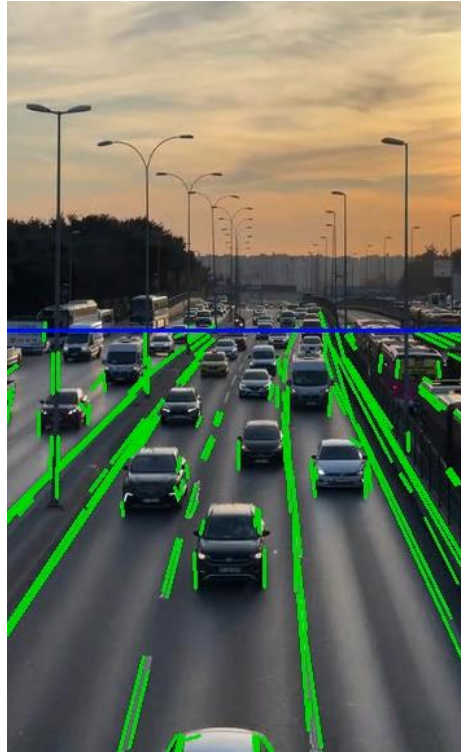


Рисунок 2.3 – Детектор ліній на зображенні з видаленням фону.

2.1.1.4 Порівняння ефективності різних методів препроцесінгу.

У процесі експериментів кожен метод використовувався як для порівняння їх ефективності у реальних відеокадрах. Після перевірки методів було отримано наступні висновки:

- метод вирівнювання гістограми забезпечив найкращу контрастність на зображеннях зі слабким освітленням, але був чутливим до шумів;
- метод посилення ліній дозволяв чітко виділяти розмітку, проте міг підсилювати шумові артефакти;
- метод видалення фону показав найкращу стабільність — він ефективно приглушував фон і зберігав локальні деталі розмітки навіть при різкому освітленні.

За результатами порівняння саме видалення фону із розміром фільтра 21×21 було обрано як основний метод попередньої обробки у подальшому алгоритмі визначення дорожнього полотна.

2.1.1.5 Виділення лінії горизонту.

Оскільки в реальних сценах можуть бути присутні численні лінії, які не належать до дорожньої розмітки (наприклад, тіні, межі бордюрів або артефакти зйомки), виникає потреба у визначенні лінії горизонту. Лінія горизонту є

важливим геометричним орієнтиром, який відділяє область дороги від верхньої частини зображення, що містить об'єкти фону. Її визначення дозволяє фільтрувати хибні лінії, масштабувати сцену та відновлювати геометрію дорожнього полотна у перспективній проекції.

У межах реалізованого методу лінія горизонту обчислюється на основі точок перетину детектованих ліній. Всі лінії, що належать до розмітки або меж смуг руху, у перспективному просторі сходяться до спільної точки зникнення, яка розташована на лінії горизонту. Відповідно, знаходження горизонту зводиться до визначення вертикальної координати

Отримане значення горизонту використовується для нормалізації координат, реконструкції ліній та подальшого визначення меж дорожнього полотна. Лінії, що перетинаються нижче горизонту, у наступних етапах можуть бути відфільтровані як такі, що не належать до основної площини дороги.

Таким чином, на основі комбінації попереднього процесингу, алгоритму LSD та геометричного аналізу перетинів реалізовано стійкий метод визначення ліній дорожньої розмітки та лінії горизонту. Це створює основу для наступного етапу — визначення ліній, які обмежують дорожнє полотно, та формування маски дороги як просторової області аналізу транспортного потоку.

2.1.2 Визначення ліній, обмежуючих дорожнє полотно

Після отримання набору ліній із попереднього етапу виконується геометричний аналіз, метою якого є вибір двох ліній, що утворюють межі проїзної частини — лівий та правий край дорожнього полотна. Для цього використовується послідовність процедур, описаних нижче.

2.1.2.1 Обчислення параметрів та аналіз нахилу ліній.

Для кожної знайденої лінії повторно обчислюються параметри k та b , а також координати точок перетину з горизонтом і нижньою межею кадру. Ці точки визначають просторову орієнтацію кожної лінії.

Наступним етапом видаляються лінії, які мають нахил, близький до лінії горизонту. А інші лінії, які лишилися, фільтруються та об'єднуються в потенційних кандидатів на обмежувальні лінії дорожнього полотна. На цьому етапі створюються лінії, які простягаються від нижньої частини зображення до

лінії горизонту.

Для того, щоб створити подібні лінії, виконується групування уже наявних ліній, якщо вони досить довгі, щоб вважатися лініями дорожнього полотна. Або ж декілька ліній мають дуже близький кут нахилу та зміщення (які були розраховані раніше), щоб їх можна було поєднати у єдину лінію (як приклад можна навести пунктирну дорожню розмітку). На зображенні це буде велика кількість ліній, не поєднаних між собою. Для того, щоб поєднати їх, ми згрупуємо всі лінії з подібним коефіцієнтом нахилу k та зміщенням b , усереднимо значення та отримаємо лінії (рис. 2.4), об'єднані лінії мають помаранчевий колір.



Рисунок 2.4 – Виділення ліній-кандидатів, що обмежують дорогу.

2.1.2.2 Формування полігону дорожнього полотна.

Надлишкова кількість ліній часто призводить до появи помилкових перетинів, які спотворюють форму дороги. Для усунення таких випадків застосовується ітеративна фільтрація:

- для кожної пари ліній обчислюється точка їхнього перетину;
- якщо перетин знаходиться нижче горизонту, то ці лінії вважаються проблемними;

- лінія, яка має найбільшу кількість таких перетинів, видаляється з набору;
- процес повторюється, доки не залишаться лише стабільні лінії без взаємних перетинів у робочій області (нижче горизонту).

Цей підхід дозволяє виключити шумові або хибні лінії, які не узгоджуються з геометрією дороги.

Після первинної фільтрації ліній виконується додатковий етап геометричного аналізу, що базується на перспективній трансформації й спрямований на вибір найбільш достовірних лівої та правої меж дорожнього полотна. У перспективному зображенні всі справжні дорожні лінії збігаються до точки зникнення, однак після переходу в ортогональний вигляд (вид зверху) ці лінії мають ставати паралельними. Ця властивість використовується як ключовий критерій для оцінки правильності меж.

На першому кроці зі сформованого набору стабільних ліній генеруються всі можливі пари кандидатів на межі. Для кожної пари обчислюються характерні точки — перетини з горизонтом та нижньою частиною кадру — на основі яких будується матриця гомографії, що переводить відповідний фрагмент сцени в ортогональну проєкцію. Після трансформації аналізуються не лише сама пара меж, але і внутрішні лінії розмітки, що знаходяться між ними. Для всіх ліній у ортогональному просторі визначаються їхні кути, і якщо вони відповідають реальній дорожній структурі, їхні напрямки мають бути майже однаковими. Для кожної пари меж підраховується кількість ліній, які стали майже паралельними після проєкції: чим більша ця кількість, тим узгодженіша геометрія ліній і тим більш імовірно, що саме ця пара є справжньою. Пара меж, яка демонструє максимальний ступінь паралельності внутрішніх ліній у ортогональному вигляді, вважається найбільш коректною й приймається як реальні межі дорожнього полотна. Такий підхід дозволяє усунути помилкові лінії, які можуть здаватися правильними в перспективі, але в ортогональному просторі виявляють суперечності, що значно підвищує точність і стабільність визначення дорожньої області навіть у складних умовах, на багатосмугових дорогах, при неповній або перерваній розмітці, тінях і перешкодах.

Після фільтрації лінії сортуються за координатою їх перетину з горизонтом. Це дозволяє впорядкувати їх у напрямку зліва направо. Далі вибираються дві лінії, що найбільше віддалені одна від одної по осі X — вони вважаються лівим та правим краєм дороги. Для підвищення стабільності результату вводиться поріг, який визначає мінімальну різницю між сусідніми лініями, щоб уникнути дублювання або надмірно близьких меж.

Використовуючи координати двох обраних ліній, будується чотирикутник (трапеція), що описує межі дороги. Цей полігон є маскою дорожнього полотна, що визначає область подальшого аналізу транспортного потоку (рис 2.5), червоним кольором зображенні відфільтровані лінії.



Рисунок 2.5 – Полігон дорожнього полотна.

Розроблений метод автоматичної виявлення дорожнього полотна має низку суттєвих переваг, що забезпечують його ефективність та практичну придатність у системах відеомоніторингу транспортного потоку. Насамперед, метод є повністю автоматичним, тобто не потребує ручного налаштування чи попереднього виділення області дороги. Це дозволяє застосовувати його до будь-яких відеопотоків незалежно від умов зйомки або розташування камери. Важливою властивістю є також стійкість до шумів та артефактів зображення, яка досягається завдяки багатокроковій фільтрації ліній і виключенню перетинів, що не відповідають геометрії дорожньої сцени.

Крім того, метод забезпечує геометричну коректність визначення дороги - сформований полігональний контур відповідає реальній перспективі зйомки, що дозволяє точно відобразити положення смуг руху та їх просторову орієнтацію. Значною перевагою є також висока швидкодія алгоритму, адже всі обчислення виконуються засобами бібліотеки OpenCV, що дозволяє працювати у режимі близькому до реального часу навіть на обладнанні з обмеженими ресурсами.

Отримана в результаті маска дороги має прикладне значення для кількох ключових задач. Вона використовується для обмеження зони пошуку транспортних засобів, що дозволяє уникати хибних детекцій за межами дорожнього полотна; для розрахунку мультикласового показника TLCR, який відображає рівень завантаженості смуги руху; а також для підготовки зображення до гомографічного перетворення, необхідного при переході до метричних координат та обчисленні швидкості транспортних засобів. Таким чином, розроблений метод є не лише ефективним з точки зору точності, але й функціонально важливим елементом у структурі комплексної системи аналізу дорожнього трафіку.

2.1.3 Детекція за допомогою сегментаційної нейромережі

Для порівняння з класичними алгоритмами комп'ютерного зору було проведено експериментальне тестування моделі об'єктної сегментації, спеціально адаптованої для визначення області дорожнього полотна. Метою цього експерименту було оцінити можливість використання глибоких нейронних мереж для автоматичного виділення проїзної частини у відеокадрах без застосування геометричних чи евристичних методів (таких як детекція ліній чи фільтрація контурів).

Для виконання сегментації використовувалася об'єктна нейронна мережа, навчена в режимі однооб'єктного виділення класу дороги. Мережа базувалася на архітектурі типу YOLO 11 seg, але з обмеженим набором класів, що дозволяло зменшити обчислювальні витрати й сфокусуватись лише на сегментації дорожнього полотна. Для навчання використовувалися зображення дорожніх сцен, відібрані з відкритих датасетів на платформі Roboflow [5].

Під час тестування було виявлено, що модель може коректно виділяти

дорожнє полотно у простих випадках — при стабільному освітленні, рівній дорозі та мінімальній кількості перешкод. Однак у реальних міських умовах результати виявилися нестабільними (рис. 2.6). Основні проблеми полягали у наступному - недостатня кількість розмічених прикладів саме з міських камер відеоспостереження (зміна перспективи, наявність перехресть, розмітки, бордюрів, тіней та транспортних засобів).



Рисунок 2.6 – Приклад роботи сегментаційної моделі виявлення дорожнього покриття.

Отримані результати свідчать, що за відсутності великого, якісно розміченого набору даних неймережевий підхід поступається класичним методам, заснованим на геометричному аналізі ліній.

Попри зазначені обмеження, отримані результати мають практичну цінність — вони показують потенціал сегментації у задачі визначення дорожнього полотна за умови наявності якісного датасету.

2.2 Мультикласовий показник TLCR

Мета мультикласового TLCR — врахувати різний вплив різних типів транспортних засобів на завантаженість смуги руху. Класична реалізація TLCR вимірює відношення площі, зайнятої транспортними засобами в кадрі, до площі смуги, і не розрізняє типи транспортних засобів. Це призводить до втрати інформації: великі й важкі авто (автобуси, вантажівки) мають більший негативний вплив на пропускну спроможність (через менші швидкості, довший гальмівний шлях, більший час прискорення), ніж легкові автомобілі чи мотоцикли. Тому логічно виділяти класи об'єктів і будувати агрегований показник із урахуванням їх ваги впливу.

2.2.1 Визначення класів для оцінки мультикласового TLCR

У традиційних підходах до аналізу транспортного потоку більшість показників оцінюють загалом, без розмежування типів транспортних засобів. Такий підхід є спрощеним, адже він розглядає весь транспорт як однорідну сукупність об'єктів, не враховуючи відмінності у габаритах, динаміці руху та впливі на пропуску здатність дороги. Насправді ж транспортний потік є гетерогенним середовищем, у якому різні типи транспортних засобів взаємодіють по-різному. Тому для більш об'єктивної оцінки стану дорожнього руху доцільно запровадити мультикласовий підхід, який дозволяє враховувати різні категорії транспортних засобів та їхній відносний вплив на транспортний потік.

У межах цієї роботи запропоновано розподіл транспортних засобів на три основні класи:

- малі транспортні засоби — мотоцикли, скутери, мопеди, електросамокати, а також надкомпактні автомобілі типу “мікро”. Ці об'єкти мають невеликі розміри, високу маневреність і можуть ефективно використовувати простір дороги, часто пересуваючись між смугами або по узбіччю. Їхній вплив на загальну заповненість смуги є мінімальним.
- звичайні транспортні засоби — це типові легкові автомобілі, які становлять основу більшості міських транспортних потоків. Саме

цей клас вважається базовим при оцінці пропускної здатності дороги та виступає «еталоном» для порівняння з іншими типами.

- великі транспортні засоби — до цієї категорії належать вантажівки, автобуси, фури, трактори та інша великогабаритна техніка. Такі транспортні засоби займають значну частину смуги руху, мають більший гальмівний шлях, повільніше прискорюються та, як наслідок, сильніше впливають на динаміку потоку, створюючи умови для локальних уповільнень або накопичень транспорту позаду.

Вибір саме цих трьох класів зумовлений не лише зручністю класифікації, а й практичними спостереженнями за міськими транспортними потоками. Аналіз відеозаписів дорожнього руху показав, що більшість транспортних засобів можна достовірно віднести до однієї з цих трьох категорій за їхньою формою, розміром та поведінковими характеристиками(рис 2.7).



Рисунок 2.7 – .

Основна ідея мультикласового підходу полягає в тому, що внесок кожного транспортного засобу у завантаження смуги є різним. Наприклад, два легкові автомобілі можуть займати ту ж площу проїзної частини, що й один автобус, але вплив на пропускну здатність від цих випадків буде відмінним. Великі транспортні засоби не лише займають більшу площу, а й обмежують швидкість інших учасників потоку, створюючи тимчасові затори або зони уповільнення.

Натомість малі транспортні засоби, хоча й можуть бути численними, рідко створюють значні перешкоди.

Щоб кількісно відобразити цей різний вплив, кожному класу транспортних засобів призначається ваговий коефіцієнт, який показує, наскільки суттєвим є внесок транспортного засобу даного типу у загальну заповненість дороги. Значення вагових коефіцієнтів можуть бути обрані як емпірично, так і на основі існуючих транспортних моделей, зокрема через використання показника Passenger Car Unit (PCU) — стандартної одиниці, що відображає еквівалентність різних типів транспорту відносно звичайного легкового автомобіля.

Для даного дослідження запропоновано такі орієнтовні вагові коефіцієнти: для малих транспортних засобів прийнято коефіцієнт - 0.75, для звичайних – 1.0, а для великих - 1.25.

Тобто один великий транспортний засіб еквівалентний двом легковим автомобілям за впливом на інтенсивність руху, тоді як малий транспортний засіб становить половину від базового значення. Ці коефіцієнти можуть бути адаптовані до конкретних міських умов на основі статистичних спостережень або регресійного аналізу.

Запропонований розподіл і система ваг створюють основу для розрахунку мультикласового показника TLCR (Traffic Lane Coverage Ratio), який дозволяє враховувати не лише просторову заповненість смуги транспортними засобами, але й їхню типологічну структуру. Таким чином, оцінка транспортного потоку стає більш реалістичною та інформативною, що відкриває можливості для глибшого аналізу транспортної динаміки, прогнозування заторів і прийняття рішень щодо оптимізації міського трафіку.

2.2.2 Визначення мультикласового TLCR

Для кількісної оцінки рівня завантаженості дорожньої смуги запропоновано показник мультикласового TLCR (Traffic Lane Coverage Ratio), який відображає частку площі смуги, зайняту транспортними засобами різних типів з урахуванням їхнього відносного впливу на рух. На відміну від класичних підходів, де заповненість оцінюється лише за площею об'єктів без урахування їхніх характеристик, мультикласовий TLCR враховує структуру транспортного

поток — тобто вагу кожного типу транспортного засобу в загальному навантаженні.

Нехай для однієї смуги руху відомі такі параметри:

- S_{lane} - площа смуги в пікселях на зображенні;
- S_i - сумарна площа в пікселях на зображенні, зайнята транспортними засобами i -го класу (малі, звичайні, великі);
- ω_i - вагові коефіцієнти для кожного класу.

Тоді мультикласовий TLCR визначається як зважене відношення сумарної зайнятої площі до площі смуги руху:

$$TLCR = \min \left(1, \frac{\sum_{i=1}^N \omega_i \cdot S_i}{S_{lane}} \right),$$

де N - кількість класів транспортних засобів.

У чисельнику враховується зважена площа — тобто площа кожного класу транспортних засобів із поправкою на ваговий коефіцієнт, який відображає відносний рівень їхнього впливу. Таким чином, великогабаритні транспортні засоби (автобуси, вантажівки) роблять більший внесок у загальний показник заповненості, ніж легкові автомобілі чи мотоцикли, навіть при однаковій площі контурів.

Якщо зважена сума площ перевищує площу смуги, метрика приймає граничне значення 1, що відповідає повному заповненню смуги руху. У цьому випадку незалежно від структури потоку (які саме класи заповнили смугу) система вважає, що її пропускна здатність вичерпана. Якщо ж смуга частково заповнена, то величина TLCR буде пропорційною до ступеня зайнятості, але водночас враховуватиме якісний склад потоку: поява навіть кількох великих транспортних засобів може істотно підвищити показник TLCR у порівнянні з ситуацією, коли ту саму площу займають лише легкові автомобілі.

Завдяки такому підходу показник TLCR набуває подвійного значення — геометричного (площа заповнення) та функціонального (вплив на динаміку руху). Це дозволяє отримувати більш реалістичну оцінку стану дорожнього потоку, особливо у складних міських умовах, де одночасно присутні різні типи транспортних засобів із різними характеристиками руху.

Важливо зазначити, що вибір вагових коефіцієнтів може бути адаптований до конкретних дорожніх умов або типів магістралей. Наприклад, для міських вулиць доцільно використовувати меншу різницю між коефіцієнтами, тоді як для автомагістралей – навпаки, підвищену вагу для великих транспортних засобів, які суттєво впливають на швидкість потоку.

Таким чином, мультикласовий показник TLCR є узагальненням класичної метрики заповненості, яке поєднує у собі просторові, вагові та структурні аспекти транспортного потоку, створюючи основу для подальшого аналізу його динаміки й оптимізації.

2.2.3 Модифікований мультикласовий TLCR (M2TLCR)

Класичний TLCR, що працює у піксельній площині зображення, страждає через перспективні спотворення та різну апаратну калібрування камер (кут огляду, відстань до сцени). Для усунення цих факторів пропонується модифікація, яка забезпечує відновлення площ в координатах bird-eye (вигляд зверху).

Процедура обчислення M2TLCR:

- калібрування / гомографія. За допомогою попередньо побудованої гомографії перетворити поточний кадр у bird-eye (ординарна проекція зверху).
- обчислення площ у міріці BE (bird-eye). Для кожного виявленого екземпляра обчислити площу його маски в BE-координатах:
- використати зважену формулу (TLCR).

Запропонований мультикласовий M2TLCR дозволяє отримати більш інформативну й практично значимішу оцінку завантаженості смуги руху, ніж класичний (однокласовий) TLCR. Механізм зважування класів дає змогу врахувати різний вплив різних видів транспорту (важкі вантажівки, автобуси, легкові автомобілі, мотоцикли) на пропускну здатність та формування заторів. Перехід до BE-метрики (модифікація MTLCR) усуває спотворення перспективи й робить показник порівнюваним між різними камерами і ділянками.

У таблиці 2.1 наведені порівняння значень різних показників завантаженості для зображення автомобільного транспортного потоку (рис. 2.8).

Таблиця 2.1 – Порівняння показників завантаженості *MTLCR* та *TLCR* для ділянки дорожнього руху.

	<i>M2TLCR</i>	<i>Мультикласовий TLCR</i>	<i>TLCR</i>
1	0,498	0,211	0,144

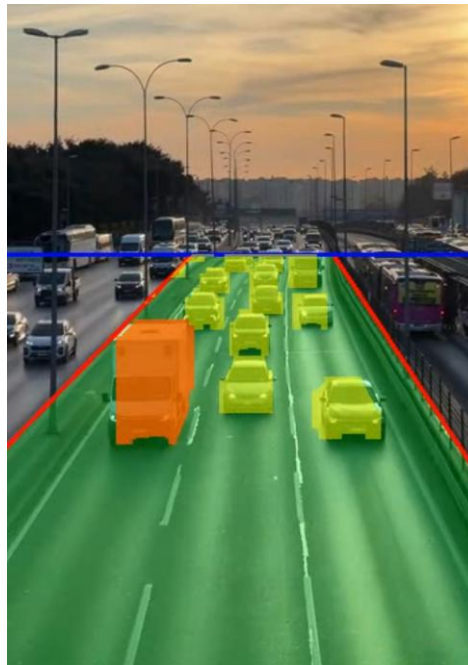


Рисунок 2.8 – Обчислення показників завантаженості дорожнього полотна.

2.3 Визначення статичних та динамічних характеристик

У сучасних системах відеоаналізу дорожнього руху визначення статичних і динамічних характеристик зазвичай реалізується через окремі моделі — семантичні сегментаційні мережі для виділення дорожнього полотна та об'єктів, і детекційні моделі (наприклад, YOLO [6]) для визначення транспортних засобів і їхніх траєкторій. Проте така архітектура є складною у практичній реалізації, вимагає попередньої синхронізації результатів між моделями, а також збільшує обчислювальні витрати.

У межах даного дослідження запропоновано єдиний підхід, який поєднує визначення статичних та динамічних характеристик дорожнього трафіку на основі моделі об'єктної сегментації. Ця модель дозволяє за один прохід нейронної мережі одночасно отримувати інформацію про контури кожного транспортного засобу (маски), їхні обмежувальні рамки (bounding boxes), типи об'єктів та просторові координати. Завдяки цьому досягається інтеграція двох підсистем аналізу – сегментації та виявлення – у єдиний обчислювальний цикл,

що значно підвищує швидкодію та узгодженість результатів.

Як основу використано об'єктну сегментаційну архітектуру YOLOv11-Seg, адаптовану під задачі дорожнього відеомоніторингу. Модель навчається на наборах даних, де кожен транспортний засіб має унікальну маску, що відокремлює його від інших об'єктів. На виході мережа генерує одночасно набір обмежувальних рамок для динамічного аналізу (відстеження руху, обчислення швидкості) та масок сегментації для оцінки статичних характеристик (ступеня заповненості, площі зайнятої дороги, TLCR).

У межах роботи була створена та натренована власна модель об'єктної сегментації на основі YOLOv11-Seg, використовуючи кастомні набори даних, сформовані та розмічені на платформі Roboflow [5]. Датасет складався зі зображень міських транспортних сцен, що містили різні типи транспортних засобів, дорожнє покриття та фонові об'єкти. У результаті модель продемонструвала високу здатність до узагальнення та коректно працює в різних умовах відеоспостереження — як для денних, так і для нічних сцен, забезпечуючи стабільну детекцію навіть при частковому перекритті об'єктів (рис. 2.9).

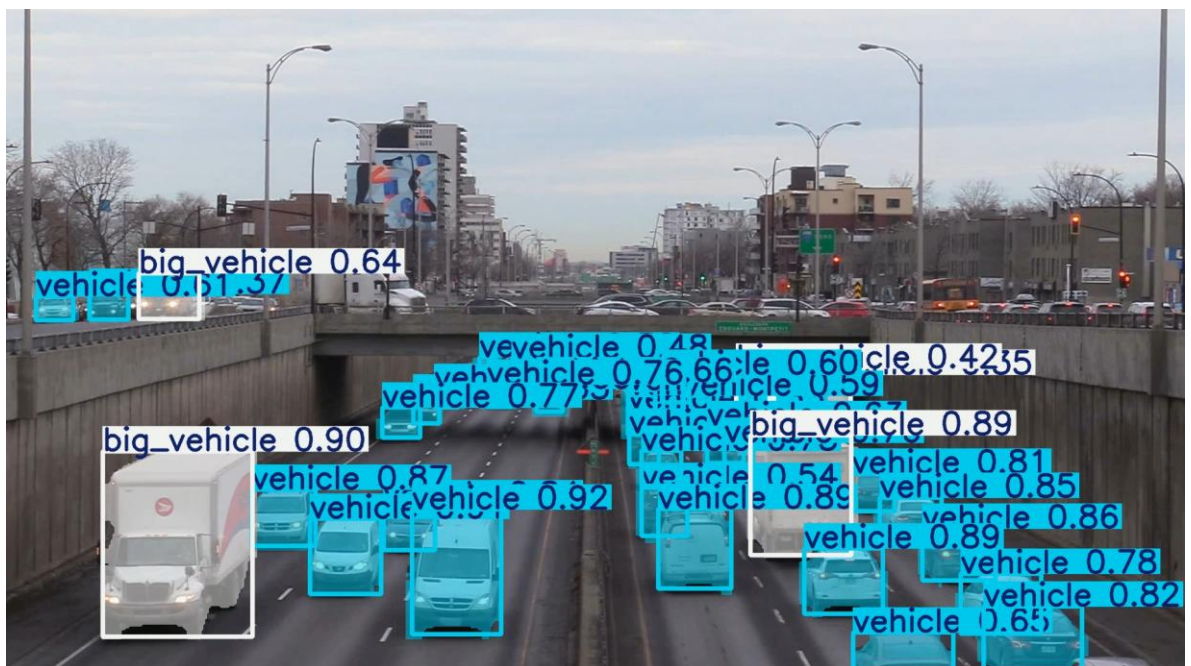


Рисунок 2.9 – Приклад роботи моделі об'єктної сегментації

Такий підхід дозволяє:

- уникнути необхідності використання двох окремих моделей (Unet для семантичної сегментації та YOLO для виявлення);
- усунути проблему асинхронності між результатами виявлення й сегментації;
- отримувати повну інформацію про транспортний потік за один обчислювальний прохід;
- підвищити узгодженість даних між статичними та динамічними метриками.

У результаті запропонована архітектура забезпечує єдність обробки, що є ключовим чинником для побудови ефективних систем аналізу транспортного потоку в режимі близькому до реального часу.

Навчена в дослідженні модель демонструє високу здатність до узагальнення та стабільну роботу в різних умовах. У таблиці 2.2 наведено значення метрик на тестовому датасеті для натренованої моделі.

Таблиця 2.2 – Метрики моделі на тестовому датасеті після тренування.

Клас	Кількість зображень	Кількість об'єктів	Box Precision	Box Recall	Box mAP50	Mask Precision	Mask Recall	Mask mAP50
0	65	142	0,781	0,599	0,708	0,767	0,585	0,702
1	568	2920	0,909	0,873	0,926	0,899	0,862	0,916
2	266	359	0,851	0,841	0,892	0,855	0,844	0,894
all	591	3421	0,847	0,771	0,842	0,840	0,764	0,840

2.3.1 Визначення статичних характеристик потоку

Статичні характеристики дорожнього потоку описують стан дороги у фіксований момент часу та відображають ступінь її заповненості транспортними засобами. У межах розробленого методу ці характеристики визначаються на основі масок об'єктів, отриманих за допомогою об'єктної сегментації. Кожен транспортний засіб у кадрі представлений у вигляді окремої двійкової маски, яка відображає точну форму об'єкта на площині дорожнього полотна.

Для оцінки заповненості дороги всі маски об'єктів, що належать до однієї смуги руху, об'єднуються в єдину сегментаційну маску заповненості. У межах цієї маски розраховується кількість пікселів, які належать транспортним засобам, після чого визначається відсоток заповнення відносно загальної площі дорожнього полотна.

На основі отриманої інформації обчислюється мультикласовий показник TLCR, який враховує типи транспортних засобів (малі, звичайні, великі) та їх вагові коефіцієнти впливу. Для підвищення точності оцінки площі застосовується гомографічне перетворення, що переводить перспективне зображення у bird-eye (вигляд зверху). Завдяки цьому піксельні площі набувають метричного змісту, а показники TLCR стають порівнюваними між різними камерами та сценами.

Таким чином, визначення статичних характеристик здійснюється шляхом:

- отримання масок усіх транспортних засобів із моделі об'єктної сегментації;
- об'єднання їх у загальну маску заповненості дороги;
- розрахунку мультикласового TLCR для кожної смуги руху;

Запропонований підхід дозволяє оцінювати рівень завантаженості дороги з урахуванням реальної структури транспортного потоку, без потреби в додаткових сенсорах або окремих сегментаційних моделях.

2.3.2 Визначення динамічних характеристик

Динамічні характеристики відображають зміну стану дорожнього потоку у часі — тобто швидкість, напрям руху та інтенсивність транспорту. У межах розробленого методу ці параметри визначаються на основі bounding box-ів, що генеруються тією ж об'єктною моделлю сегментації. Кожен об'єкт має унікальний ідентифікатор, що дозволяє відстежувати його між послідовними кадрами.

Швидкість руху транспортного засобу обчислюється на основі зміни координат центру bounding box-а між двома сусідніми кадрами. Враховуючи частоту кадрів відеопотоку (FPS) та параметри гомографії, зміщення у піксельному просторі переводиться у метричну швидкість. Для згладжування

флуктуацій у траєкторіях застосовується ковзне усереднення.

Інтенсивність руху визначається шляхом підрахунку кількості транспортних засобів, які перетнули певну контрольну лінію або область інтересу (ROI) за одиницю часу. На основі цих даних формується профіль руху, який може бути використаний для оцінки пропускної здатності дороги або прогнозування заторів.

Завдяки тому, що як статичні, так і динамічні характеристики отримуються з одного джерела — моделі об'єктної сегментації — забезпечується їхня синхронність і просторово-часова узгодженість. У результаті формується повна аналітична картина транспортного потоку, яка містить:

- кількість транспортних засобів у кадрі;
- середню швидкість та її розподіл за класами;
- напрямки руху;
- ступінь заповненості смуг і показник TLCR.

Таким чином, використання єдиної об'єктної моделі дозволяє виконувати одночасний розрахунок як статичних, так і динамічних характеристик, що забезпечує високу точність, узгодженість і ефективність системи відеомоніторингу.

Висновки до розділу

Об'єднання об'єктної сегментації (маски + класи) із трекінгом дає повний набір даних для обчислення як статичних (MTLCR, M2TLCR), так і динамічних характеристик транспортного потоку. Єдина об'єктна архітектура спрощує пайплайн: один прогін моделі дає маски для площ, обмежувальні рамки для трекінгу і embeddings для відновлення ідентичності об'єктів в часі. При правильній калібровці гомографії, корекції перекриттів і згладжуванні швидкостей отримана система дає надійну та узгоджену інформацію для прийняття рішень у транспортній інфраструктурі.

У цьому розділі було розроблено, обґрунтовано та реалізовано методичні та алгоритмічні засади системи автоматизованого відеомоніторингу характеристик автомобільного транспортного потоку.

На першому етапі дослідження розглянуто та реалізовано підхід до

виявлення дорожнього полотна на основі аналізу лінійних структур у відеопотоці. Запропоновані алгоритми поєднують методи попередньої обробки зображення із процедурами пошуку, групування та реконструкції ліній дорожньої розмітки. Розроблена функція виділення меж дорожнього полотна забезпечує стабільне формування області дороги навіть за наявності шумів, часткових затіненень або дефектів розмітки.

Додатково проведено тестування об'єктної сегментаційної нейромережі, навченої для виявлення дорожнього полотна. Незважаючи на обмеженість навчальної вибірки, результати підтвердили перспективність використання сегментаційних моделей у майбутніх версіях системи — за умови розширення та уточнення наборів розмічених даних.

Для комплексної оцінки стану транспортного потоку запропоновано мультикласовий показник TLCR (Traffic Lane Congestion Ratio), який враховує різні типи транспортних засобів — малі, звичайні та великі. Це дозволяє більш об'єктивно оцінювати рівень завантаженості смуги руху, оскільки різні типи транспортних засобів мають неоднаковий вплив на пропускну здатність дороги. Запропоновано узагальнену математичну модель TLCR, яка враховує площу об'єктів, їх вагові коефіцієнти (еквіваленти пасажирських авто, PCU) та швидкісні параметри. Такий підхід дозволяє відображати не лише факт заповненості дороги, а й динамічні зміни, пов'язані з поведінкою транспорту різних класів.

У подальшій частині розділу розроблено методи для визначення статичних та динамічних характеристик потоку. Статичні показники (щільність, TLCR) визначаються на основі аналізу об'єктних масок транспортних засобів у bird-eye (BE) координатах, а динамічні — за допомогою трекінгу у часовій послідовності кадрів. Для цього розроблено конвеєр обробки даних, який поєднує результати сегментації, виявлення та трекінгу.

Завершальним етапом стала розробка єдиної архітектури системи, яка базується на об'єктній моделі і забезпечує одночасне отримання всіх необхідних даних — масок, класів, баундінг-боксів та ознак для трекінгу. Це дозволило об'єднати обчислення статичних і динамічних параметрів у межах одного

обчислювального конвеєра, що підвищує узгодженість даних і зменшує часові затримки при аналізі відеопотоку.

Запропонований підхід забезпечує підвищену точність і стійкість до шумів відеопотоку, а також створює основу для подальшого вдосконалення системи відеомоніторингу з використанням сучасних моделей глибокого навчання та методів прогнозування транспортних процесів.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Моделювання програмного засобу для оцінки характеристик транспортного потоку

Розробка програмного засобу базується на модульному підході до проектування інформаційних систем обробки відеоданих. Головною метою моделювання є створення архітектури, яка забезпечує роботу в режимі близькому до реального часу з можливістю паралельної обробки геометричних та семантичних ознак сцени.

3.1.1 Архітектура системи

Загальна структура програмного засобу спроектована у вигляді конвеєра (рис. 3.1) послідовно-паралельної обробки даних. Концептуальна модель системи складається з трьох рівнів:

- *рівень отримання та попередньої обробки даних*: забезпечує захоплення відеопотоку, нормалізацію роздільної здатності та підготовку кадрів для аналізу.
- *рівень аналізу*: центральний елемент системи, що розділяє обчислення на два незалежних потоки – модуль визначення дорожнього полотна, що використовує методи класичного комп'ютерного зору та об'єктний детектор, котрий визначає та відслідковує динамічні об'єкти (транспортні засоби) за допомогою нейромережевих алгоритмів.
- *рівень обчислення метрик*: Об'єднує дані з попереднього рівня для трансформації координат у метричний простір та розрахунку кількісних показників трафіку.

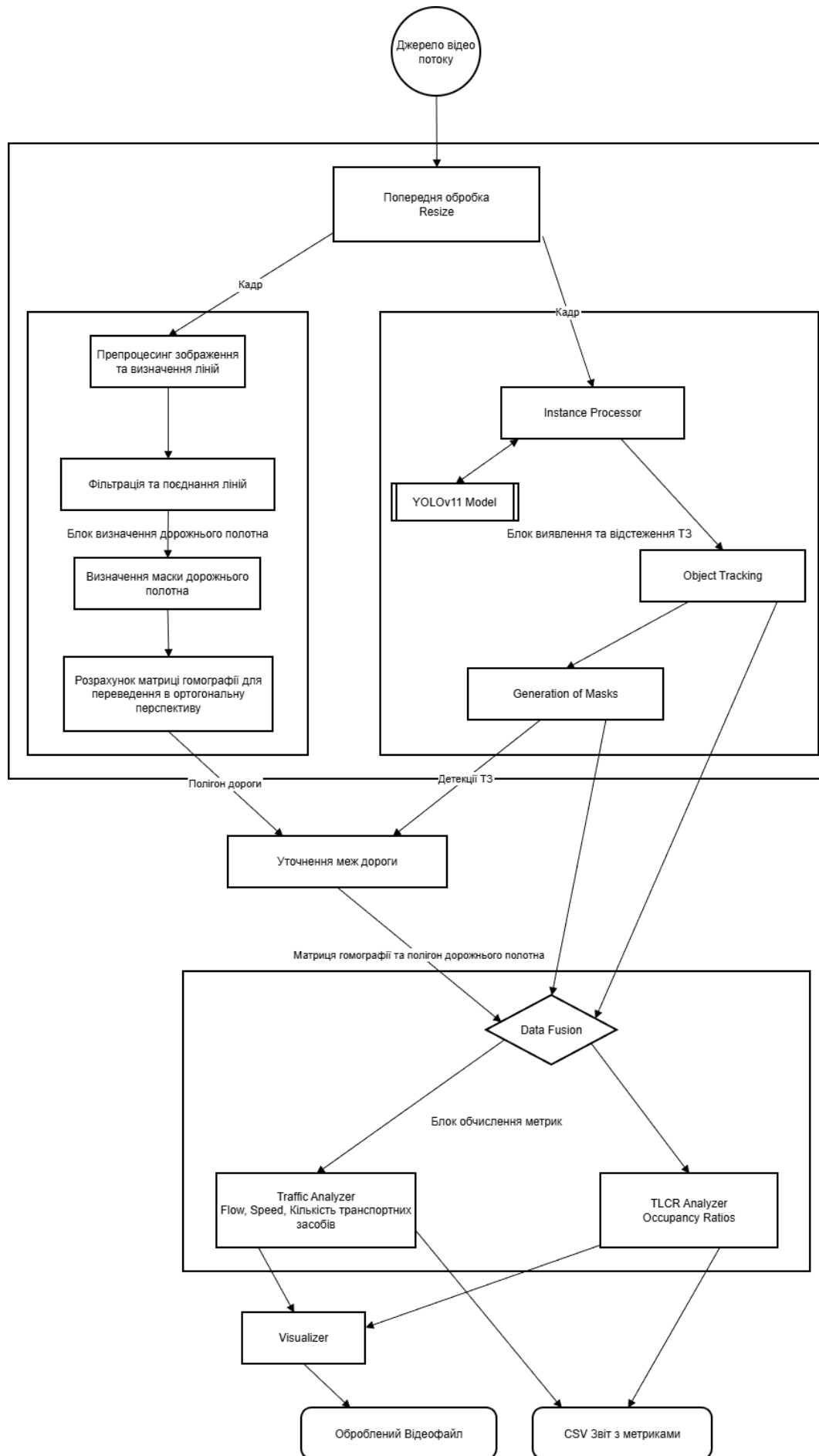


Рисунок 3.1 – Архітектура розроблюваного програмного засобу.

3.1.2 Алгоритмічна модель

Алгоритмічна модель функціонування програмного засобу (рис. 3.2) являє собою комплексний ітеративний процес, що інтегрує методи геометричного моделювання сцени та семантичного аналізу об'єктів. Процедура обробки розпочинається з роботи детектора дорожнього полотна, який, на відміну від систем з фіксованими параметрами, забезпечує динамічну локалізацію дорожнього полотна на кожному кадрі.

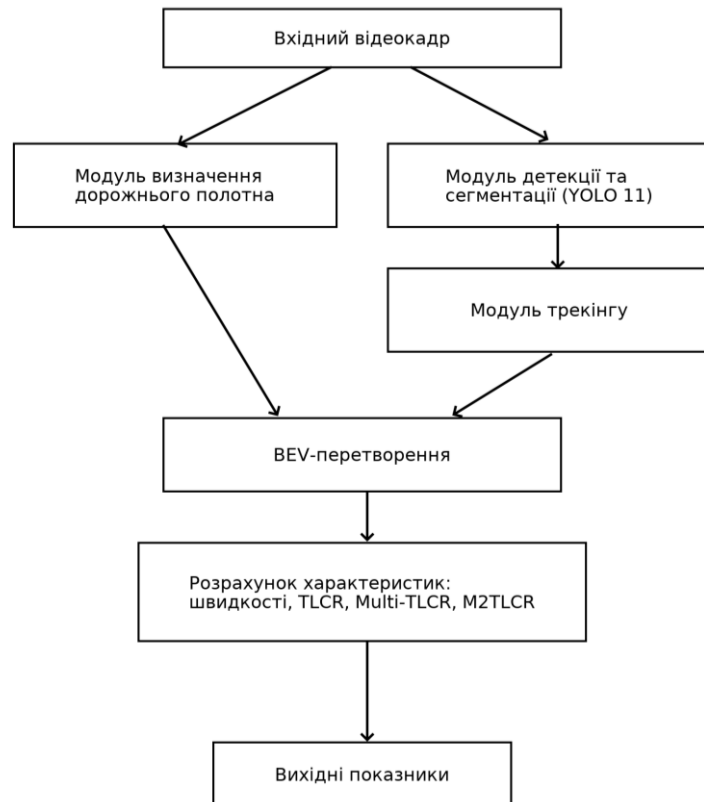


Рисунок 3.2 – Алгоритмічна модель функціонування ПЗ.

Паралельно з геометричним аналізом функціонує модель детекції та трекінгу, реалізована на базі нейромережі YOLO 11. Вона відповідає за виявлення транспортних засобів, їх масок, обмежувальних рамок та класифікацію. Для забезпечення часової послідовності даних застосовується алгоритм трекінгу, який пов'язує просторові координати об'єктів на поточному кадрі з їхніми станами на попередніх ітераціях, присвоюючи кожному транспортному засобу унікальний ідентифікатор для побудови траєкторії руху.

Завершальним етапом алгоритму є оцінка характеристик транспортного потоку шляхом поєднання та опрацювання даних, отриманих з модулів вище просторових та часових даних. На основі переміщення центрів об'єктів у

метричному просторі BEV розраховується усереднена швидкість потоку. Оцінка завантаженості інфраструктури виконується через розрахунок коефіцієнтів покриття смуги: базового TLCR, як відношення сумарної площі масок транспортних засобів до площі дорожнього полотна, та уточненого мультикласового показника (TLCR), і його варіації з урахуванням перспективи (M2TLCR). Останні дозволяють врахувати внесок різних категорій транспорту через вагові коефіцієнти.

3.1.3 Взаємодія компонентів

Логічна взаємодія компонентів системи передбачає механізм зворотного зв'язку(рис. 3.3). Дані про розташування транспортних засобів використовуються для уточнення меж дорожнього полотна (фільтрація хибних ліній), а уточнені межі дороги, в свою чергу, використовуються для фільтрації хибних детекцій об'єктів, що знаходяться поза межами проїжджої частини (наприклад, припарковані авто на тротуарі).

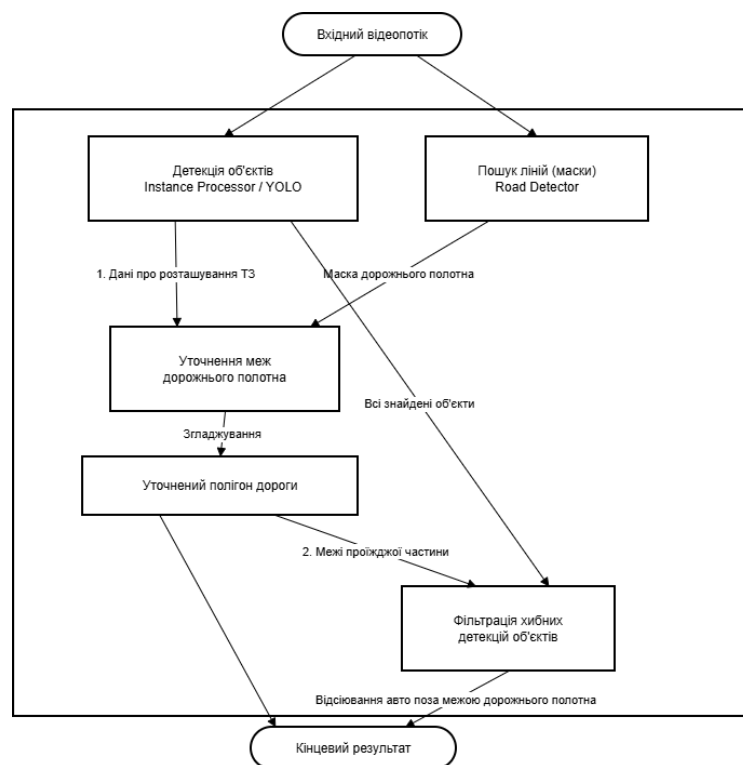


Рисунок 3.3 – Механізм зворотнього зв'язку.

Такий підхід до моделювання забезпечує високу адаптивність системи до зміни умов зйомки та ракурсу камери.

3.2 Реалізація програмного забезпечення

Програмний засіб реалізовано мовою Python [19], яку обрано за ефективність у задачах наукового моделювання та обробки зображень. Технологічний стек базується на бібліотеках OpenCV (для геометричних перетворень та фільтрації), NumPy [20] (для матричних обчислень) та фреймворку Ultralytics (для роботи з нейромережею YOLOv11).

Загальна архітектура системи побудована навколо головного керуючого модуля (`main.py`), який ініціалізує відеопотік та керує роботою спеціалізованих класів. Для забезпечення швидкодії в режимі реального часу реалізовано механізм асинхронного виконання: геометричний аналіз полігону дорожнього полотна та нейромережева детекція об'єктів виконуються у паралельних потоках за допомогою модуля `concurrent.futures`. Це дозволяє ефективно використовувати ресурси процесора та графічного прискорювача, за його наявності.

3.2.1 Модуль визначення дорожнього полотна

Модуль реалізовано у класі `RoadDetector`. Його основним завданням є динамічне визначення доступної для руху ділянки дороги без використання попередньо розмічених даних.

Алгоритм роботи модуля базується на послідовній фільтрації геометричних примітивів. Спочатку зображення перетворюється у відтінки сірого, після чого видаляється фоновий шум. Далі застосовується детектор ліній LSD для пошуку лінійних структур. Ключовим етапом є пошук пари ліній, які утворюють перспективну проєкцію дороги, що перевіряється шляхом їх трансформації у ортогональну проєкцію. Також модуль використовує дані про розташування транспортних засобів для уточнення меж дороги (метод зворотного зв'язку) та згладжує отриманий полігон у часі для уникнення різких змін розміру чи форми.

Основні методи класу `RoadDetector` та їх призначення наведено в таблиці 3.1.

Таблиця 3.1 — Функціональні елементи модуля RoadDetector.

<i>Назва методу</i>	<i>Опис функціональності</i>
detect_road	Головний метод, що керує процесом детекції. Виконує пошук ліній, оцінку горизонту та фільтрацію кандидатів на межі дороги.
_rem_bg	Виконує попередню обробку кадру: видаляє низькочастотний фоновий шум за допомогою різниці між оригіналом та розмиттям за Гаусом.
_detect_lines_lsd	Використовує алгоритм Line Segment Detector для знаходження всіх відрізків на зображенні.
_estimate_horizon	Розраховує координату лінії горизонту на основі точок перетину знайдених ліній.
_merge_and_reconstruct_lines	Групує розрізнені відрізки, що мають схожий кут нахилу та зсув, у суцільні лінії та знаходить їх координати відносно лінії горизонту.
_extract_road_mask	Визначає оптимальний полігон дороги, аналізуючи пари ліній на предмет їх паралельності у трансформованому просторі (BEV).
_refine_road_pts_with_detections	Коригує знайдені межі дороги, розширюючи або звужуючи їх так, щоб охопити максимальну кількість центрів виявлених транспортних засобів.

Закінчення таблиці 3.1

<i>Назва методу</i>	<i>Опис функціональності</i>
smooth_road	Згладжує координати полігону дороги, використовуючи історію попередніх кадрів та перевіряє валідність поточної детекції.
get_bev_transform_info	Розраховує матрицю перспективного перетворення та розміри зображення у проекції «вид зверху».
transform_class_masks_to_bev	Виконує перспективне перетворення масок окремих класів об'єктів у проекцію «вид зверху».
transform_mask_to_bev	Трансформує загальну об'єднану бінарну маску всіх транспортних засобів у проекцію «вид зверху».

3.2.2 Модуль виявлення та відстеження транспортних засобів

Модуль виявлення та відстеження ТЗ реалізовано у класі InstanceProcessor. Він відповідає за аналіз сцени та транспортного потоку, використовуючи модель глибокого навчання.

Модуль завантажує попередньо навчені ваги нейромережі об'єктної сегментації YOLOv11. Під час обробки кадру він виконує дві функції: детекцію об'єктів з генерацією масок сегментації та трекінг, що дозволяє присвоювати унікальні ідентифікатори об'єктам для відстеження їх переміщення між кадрами. Отримані результати діляться за класами (малі, звичайні та великі транспортні засоби) та об'єднуються у бінарні маски сегментації, які передаються на наступні етапи для аналізу. Опис методів модуля наведено в таблиці 3.2.

Таблиця 3.2 — Функціональні елементи модуля InstanceProcessor

<i>Назва методу</i>	<i>Опис функціональності</i>
process	Виконує інференс нейронної мережі YOLO у режимі трекінгу. Повертає результати детекції (бокси, маски, класи, ID трекінгу).
get_masks_by_class	Розбирає результати детекції та формує окремі бінарні маски для кожного цільового класу об'єктів.
combine_masks	Об'єднує маски всіх класів в одну загальну маску зайнятості дорожнього простору.

3.2.3 Модулі обчислення характеристик транспортного потоку

Розрахунок динамічних та статичних характеристик транспортного потоку реалізовано у двох наступних класах: TrafficAnalyzer та TLCRAnalyzer.

TrafficAnalyzer відповідає за динамічні характеристики транспортного потоку. Він працює з координатами центрів об'єктів у ортогональній проекції, розраховує пройдену відстань між кадрами та обчислює швидкість. На основі цього розраховує показник «Flow», що характеризує інтенсивність трафіку в поточний момент часу відносно максимальної зафіксованої швидкості за попередній період роботи програмного забезпечення.

TLCRAnalyzer виконує розрахунок метрик завантаженості дорожнього полотна. Він оперує площами масок, обчислюючи відношення зайнятої площі до вільної. Модуль підтримує як прості, так і зважені метрики, де різні типи транспорту мають різний вплив на кінцевий розрахунок.

Функції модулів та їх опис наведено у таблиці 3.3.

Таблиця 3.3 — Функціональні елементи модулів розрахунку характеристик транспортного потоку

<i>Клас</i>	<i>Назва методу</i>	<i>Опис функціональності</i>
TrafficAnalyzer	calculate_flow_speed	Основний метод розрахунку. Виконує проєкцію точок у BEV, обчислює швидкість кожного об'єкта, середню швидкість потоку та індекс інтенсивності руху.
	_get_bottom_center	Допоміжний метод для визначення точки прив'язки автомобіля (середина нижньої грані рамки) для точнішого позиціювання на дорозі.
TLCRAnalyzer	calculate_tlcr	Розраховує класичний коефіцієнт TLCR: відношення площі всіх об'єктів до площі дорожнього полотна.
	calculate_multy_tlcr	Розраховує зважений коефіцієнт (TLCR), застосовуючи вагові коефіцієнти до площ різних класів транспорту.
	calculate_m2tlcr	Розраховує модифікований показник на основі масок, трансформованих у проєкцію BEV, що усуває вплив перспективи.

3.3 Оцінка ефективності розробленого програмного забезпечення

Для перевірки придатності розробленого програмного засобу до використання в реальних умовах було проведено тестування його ефективності. Основним критерієм оцінки було обрано швидкодію системи (продуктивність), яка є критичною вимогою для задач моніторингу дорожнього руху в режимі близькому до реального часу.

Оцінка швидкодії здійснювалася шляхом вимірювання двох ключових показників під час обробки тестового відеопотоку:

- частота кадрів (FPS — Frames Per Second): кількість кадрів, яку система встигає повністю обробити (детекція, трекінг, розрахунок метрик, візуалізація) за одну секунду.
- середній час інференсу (Average Inference Time): середній час, що витрачається на повний цикл обробки одного кадру, виражений у мілісекундах (мс).

Тестування проводилося на різних апаратних конфігураціях для визначення залежності продуктивності від обчислювальної потужності обладнання. Результати експериментальних вимірювань наведено в таблиці 3.4.

Таблиця 3.4 — Результати тестування швидкодії програмного забезпечення

<i>Конфігурація обладнання</i>	<i>Частота кадрів, FPS</i>	<i>Середній час інференсу, мс</i>
Intel i5 12450h + NVIDIA GeForce RTX 3050 Laptop	1,38	726,8
Intel i5 12500h	0,72	1393,7
Intel Ultra 9 285k + NVIDIA GeForce RTX 5070 Ti	3,70	270,3

Висновки до розділу

У цьому розділі виконано програмну реалізацію інформаційної системи для оцінки характеристик транспортного потоку, що базується на гібридному поєднанні методів класичного комп’ютерного зору та глибокого навчання.

Розроблений програмний комплекс створено мовою Python з використанням бібліотек OpenCV, NumPy, Ultralytics та архітектури сегментаційної нейронної мережі YOLOv11, що забезпечило оптимальний баланс між точністю детекції об’єктів та обчислювальною швидкістю. Архітектура системи побудована за модульним принципом, де ключові компоненти — детектор дорожнього полотна, трекер транспортних засобів та

аналізатор метрик — функціонують у середовищі з підтримкою паралельних обчислень, що дозволяє обробляти відеопотік у режимі близькому до реального часу.

Результатом роботи є програмний засіб, який автоматично генерує комплексні показники завантаженості (TLCR, MTLCR) та розраховує динамічні характеристики потоку, що підтверджує ефективність обраних підходів для задач моніторингу дорожнього трафіку.

4 МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЄКТУ

4.1 Опис ідеї проєкту

Таблиця 4.1 — Опис ідеї стартап-проєкту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Платформа для аналізу характеристик дорожнього трафіку	Аналіз характеристик дорожнього трафіку у режимі близькому до реального часу	Отримання даних про трафік, готових до аналізу
	Візуалізація параметрів дорожнього трафіку	Представлення виявлених транспортних засобів та обчислених показників у вигляді, зрозумілому для користувача
	Збір даних для подальшого аналізу та застосування	Можливість збирати розраховані метрики та показники з відеокамер за потрібний проміжок часу для потрібної ділянки дороги, та подальший аналіз

Таблиця 4.2 — Опис ідеї стартап-проєкту

№	Техніко економічні характеристики ідеї	Продукція конкурентів			W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Мій проєкт	Data FromSky [17]	TomTom [18]			
1	Відстежування транспортних засобів	Наявне	Наявне	Відсутнє		N	
2	Оцінка швидкостей транспортних засобів	Наявна	Наявна	Наявна		N	
3	Мультикласова класифікація (типи ТЗ)	Наявна	Наявна	Відсутня			S

Закінчення таблиці 4.2

4	Оцінка завантаженості дорожнього полотна	Наявна	Відсутня	Наявна			S
5	Вартість впровадження та ліцензії	Низька	Висока	Висока			S
6	Масштабованість	Локальна	Локальна	Глобальна	W		
7	Обробка в режимі близькому до реального часу	Наявна	Наявна	Наявна		N	

4.2 Технологічний аудит ідеї проєкту

Таблиця 4.3 — Технологічна здійсненність ідеї проєкту

№	Ідея проєкту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Детекція транспортних засобів	Нейромережі (YOLOv11/v8), Ultralytics	Так	Доступна
2	Аналіз дорожнього полотна	OpenCV, LSD, Background Subtraction	Так	Доступна
3	Розрахунок метрик	Python, NumPy	Так	Доступна
4	Візуалізація роботи програми	Dashboard (Web/Desktop), Matplotlib, OpenCV	Так	Доступна
<i>Обрана технологія реалізації ідеї проєкту: Python3, Ultralytics, OpenCV, NumPy</i>				

Висновок: технологічна реалізація можлива, всі компоненти є доступними та мають широку підтримку спільноти.

4.3 Аналіз ринкових можливостей запуску стартап-проєкту

Таблиця 4.4 — Попередня характеристика потенційного ринку

№	Показники стану ринку	Характеристика
1	Кількість головних гравців, од	3
2	Загальний обсяг продаж, грн./ум.од	500 000 ум. од. за місяць
3	Динаміка ринку	Зростаюча
4	Наявність обмежень для входу	Середня (потрібні технічні компетенції)
5	Специфічні вимоги до стандартизації та сертифікації	ISO, вимоги щодо захисту персональних даних
6	Середня норма рентабельності в галузі або по ринку, %	40%

Висновок: ринок є привабливим через зростаючий попит на подібні рішення та відносно високу рентабельність програмних продуктів, попри наявність сильних глобальних гравців.

Таблиця 4.5 — Характеристика потенційних клієнтів стартап-проєкту

№	Потреба, що формує ринок	Цільова аудиторія	Відмінності у поведінці цільових груп клієнтів	Вимоги споживачів до товару
1	Моніторинг та оптимізація міського трафіку, керування світлофорами	Муніципалітети, комунальні підприємства, департаменти транспорту.	Довгий цикл прийняття рішень, тендерні закупівлі, висока бюрократія.	Надійність, наявність сертифікатів, інтеграція з існуючою інфраструктурою.
2	Проектування розв'язок, ЖК та ТРЦ (оцінка пропускної здатності)	Приватні забудовники, урбаністичні бюро, ТРЦ.	Орієнтація на результат, швидке прийняття рішень.	Точність даних, візуалізація звітів, вартість впровадження.

Закінчення таблиці 4.5

3	Аналіз ефективності зовнішньої реклами (кількість авто, що проїжджають повз білборд)	Рекламні агентства.	Потреба в конкретних метриках.	Детальна статистика по типах авто, низька ціна.
---	--	---------------------	--------------------------------	---

Таблиця 4.6 — Фактори загроз

№	Фактор	Зміст загрози	Можлива реакція компанії
1	Конкуренти	Глобальні гравці мають більший бюджет на маркетинг.	Зменшення ціни на поставлену послугу
2	Кошти на розробку	Нестача ресурсів для купівлі потужних серверів для обробки.	Оптимізація коду, використання хмарних рішень з оплатою за факт використання, залучення грантів.
3	Вихід аналогів	Поява на ринку ПЗ подібного функціоналу та знецінення існуючих рішень	Постійний розвиток функціоналу розробленого рішення, інвестиції в перспективні напрямки розвитку. Проведення рекламної компанії.

Таблиця 4.7 — Фактори можливостей

№	Фактор	Зміст можливості	Можлива реакція компанії
1	Зростання попиту на Smart City рішення	Зацікавленість у цифровізації міського простору та отриманні точних даних про трафік без встановлення дорогих апаратних датчиків.	Розробка універсального ПЗ, сумісного з більшістю сучасних IP-камер. Акцент у маркетингу на економічній вигоді.
2	Потреба в детальній класифікації транспорту	Існуючі системи часто рахують лише кількість авто, тоді як урбаністи потребують даних про типи (вантажівки, автобуси, легковіки) для розрахунку навантаження на інфраструктуру.	Впровадження та вдосконалення мультикласових метрик (TLCR, M2TLCR); Надання розширених аналітичних звітів для проєктувальників доріг.

Закінчення таблиці 4.7

3	Використання існуючої інфраструктури відеоспостереження	Велика кількість вже встановлених камер можуть генерувати аналітичні дані.	Пропозиція ліцензій за моделлю SaaS для обробки відеопотоків із камер замовника, створення API для легкої інтеграції.
4	Оптимізація транспортної інфраструктури	Установи, що забезпечують роботу транспортної системи, зацікавлені в динамічному моніторингу дорожнього руху для ухвалення рішень щодо розвитку інфраструктури.	Створення й впровадження потрібного функціоналу; Надання гнучких ліцензій під потреби клієнтів.

Таблиця 4.8 — Ступеневий аналіз конкуренції на ринку

№	Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1	Тип конкуренції: монополістична	Багато продавців, але товари диференційовані.	Чітке позиціонування унікальних переваг.
2	Рівень конкурентної боротьби: галузевий/міжгалузевий	Конкуренція як з IT-компаніями, так і з виробниками датчиків.	Акцент на універсальності програмного рішення.
3	Галузева ознака: ринок ПЗ та систем безпеки	Продукт на стику IT та фізичної інфраструктури.	Партнерство з інсталяторами систем відеоспостереження.
4	Конкуренція за видами товарів: товарно-родова	Вибір між відеоаналітикою, GPS-даними та фізичними датчиками.	Доведення переваг саме відеоаналітики (точність, класифікація, візуальний доказ).
5	Характер конкурентних переваг: цінова та якісна	Проект привабливий ціною та гнучкістю.	Впровадження гнучкої тарифної сітки, оптимізація вартості впровадження.
6	За інтенсивністю: сильна	Ринок динамічний, технології швидко старіють.	Постійне оновлення нейромереж (SOTA моделей) для утримання точності.

Таблиця 4.9 — Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
	DataFrom Sky, TomTom	Інтегратори з кастомними рішеннями	NVIDIA (GPU), Cloud (AWS/Azure)	Системи інформації та контроль якості	TomTom (GPS), індуктивні петлі, радары
Висновки	Конкуренція висока в преміум сегменті.	Низький бар'єр входу створює загрозу нових стартапів.	Вплив постачальників низький	Вплив клієнтів високий (можуть обрати дешевші аналоги).	Замінники надійні, але дорогі (радары) або менш детальні (GPS).

Проаналізувавши можливості роботи на ринку з огляду на конкурентну ситуацію, можна зробити висновок: оскільки ринок характеризується монополістичною конкуренцією, де існуючі продукти не впливають у великій мірі на поточну ситуацію в цілому, а мають свою специфічну сферу використання, позитивні та негативні сторони, то робота та вихід на даний ринок є можливою і реалізованою задачею. Відсутність тотального домінування одного гравця створює можливості для впровадження нового рішення, яке поєднує точність відеоаналітики з доступною вартістю.

Для успішного виходу на ринок продукт повинен мати функціонал, що відсутній у продуктів-аналогів, повністю задовольняти потреби користувачів шляхом використання існуючої інфраструктури, мати необхідний та достатній функціонал з конфігурування, забезпечувати технічну підтримку зі сторони розробників та надавати можливість розробки спеціального функціоналу за відповідною ліцензією.

Таблиця 4.10 — Обґрунтування факторів конкурентоспроможності

№	Фактор конкурентоспроможності	Обґрунтування
1	Ціна рішення	Використання відкритих рішень та існуючих камер дозволяє знизити ціну нижче конкурентів.
2	Точність класифікації	Використання сучасних нейромереж (YOLOv11) забезпечує кращу точність, ніж GPS-трекери.
3	Унікальні метрики	Врахування більшої кількості факторів дає кращу оцінку завантаженості дорожнього полотна.
4	Швидкість розгортання	Не потребує попереднього калібрування камер чи додаткової розмітки.
5	Візуалізація	Візуалізація дозволяє надати дані користувачеві у зрозумілому вигляді

Таблиця 4.11 — Порівняльний аналіз сильних та слабких сторін програмного засобу для аналізу характеристик автомобільного транспортного потоку

№	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з запропонованим						
			-3	-2	-1	0	+1	+2	+3
1	Вартість (доступність)	5		+					
2	Точність детекції	18				+			+
3	Масштабованість (покриття міста)	8					+		
4	Простота налаштування	19		+					
5	Функціональність	15					+		
6	Візуалізація	16			+				

Таблиця 4.12 — SWOT аналіз стартап-проєкту

Сильні сторони (S): – Низька собівартість впровадження. – Гнучкість (автоматичне визначення дорожнього полотна).	Слабкі сторони (W): – Локальний масштаб. – Обмежені ресурси для створення мінімально-працюючого продукту.
Можливості (O): – Впровадження нового функціоналу – Розширення сегменту користувачів	Загрози (T): – Поява конкурентів на ринку – Цінова політика конкурентів

Таблиця 4.13 — Альтернативи ринкового впровадження стартап-проєкту

№	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Безкоштовне надання певного функціоналу у користування споживачам на обмежений термін	Головний ресурс – люди, даний ресурс - наявний	2-3 місяці
2	Реклама	Залучення власних коштів для реклами товару	1-2 місяці
3	Написання статей та опис товару на відомих ресурсах	Головний ресурс – час, даний ресурс - наявний	2-3 тижні
4	Презентація товару на хакатонах й інших ІТ заходах	Ресурс – час та гроші для участі, наявні	1-3 місяці

4.4 Розроблення ринкової стратегії проєкту

Таблиця 4.14 — Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Державні установи	Висока. Зручність ПЗ для аналізу характеристик транспортного потоку	Середня. Наявна невелика кількість аналогів	Середня	Складна
2	Приватні установи	Висока. Зручність ПЗ для аналізу характеристик транспортного потоку	Середня. Наявна невелика кількість аналогів	Середня	Складна
Які цільові групи обрано: 1, 2					

Відповідно до проведеного аналізу, цільовими групами для просування програмного продукту визначено державні та приватні установи, що займаються обслуговуванням дорожньо-транспортних систем. Для обох сегментів характерні висока готовність до сприйняття продукту, середній рівень попиту, помірна конкуренція та складність входу, що підтверджує доцільність їх вибору.

З огляду на це обрано стратегію масового маркетингу, оскільки встановленням і експлуатацією транспортної інфраструктури займаються як державні структури, так і приватні компанії, а запропонований продукт є стандартизованим рішенням з можливістю розширення функціоналу залежно від умов ліцензування.).

Таблиця 4.15 — Визначення базової стратегії розвитку

Обрана альтернатива розвитку проєкту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Прямі продажі приватним клієнтам	Стратегія концентрованого маркетингу	Гнучкість, ціна, детальна аналітика	Стратегія диференціації

Таблиця 4.16 — Визначення базової стратегії конкурентної поведінки

Чи є проєкт «першопрохідцем» на ринку	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, які?	Стратегія конкурентної поведінки
Ні, ринкові лідери вже існують	Так, ціль компанії знайти нових споживачів та, частково, забрати існуючих у конкурентів	Частково (базові метрики), але додаємо унікальні (M2TLCR)	Стратегія комутанта: задоволення специфічних потреб локального бізнесу.

Таблиця 4.17 — Визначення стратегії позиціонування

№	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проєкту	Вибір асоціацій, які мають сформувати комплексну позицію власного проєкту
1	Точність та ціна	Диференціація	M2TLCR, Open Source база	Доступна аналітика
2	Простота	Нічева	Робота зі звичайними камерами	Розумна камера

Відповідно до проведеного аналізу можна зробити висновок, що стартап-компанія вибирає як базову стратегію розвитку - стратегію диференціації, як базову стратегію конкурентної поведінки - задоволення специфічних потреб локального бізнесу.

4.5 Розроблення маркетингової програми стартап-проекту

Таблиця 4.18 — Визначення ключових переваг концепції потенційного товару

№	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1	Зниження витрат на дослідження автомобільного трафіку	Автоматизація підрахунку	Низька вартість порівняно з конкурентами.
2	Отримання об'єктивних даних	Усунення людського фактору помилок	Вища точність класифікації ніж у GPS-сервісів.
3	Аналіз складних ситуацій	Розуміння причин заторів через використання ефективних метрик	Унікальні метрики, що враховують не лише завантаженість дорожнього полотна, а і вплив різних типів ТЗ на дорожню ситуацію.

Таблиця 4.19 — Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові
1. Товар за задумом	Програмне забезпечення універсальне, та дозволяє виконувати аналіз характеристик автомобільного транспортного потоку в режимі, близькому до реального часу

Закінчення таблиці 4.19

2. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх/Тл/Е/Ор
	Вартість обслуговування	М	Вр
	Знижка	Нм	Вр
	Безвідмовність	М	Тх
	Собівартість	М	Тх
	Швидкість	М	Е
	Інтегрованість	М	Тх
	Якість: РЕР 8		
	Спосіб взаємодії: додаток у мережі інтернет		
	Марка: -		
3. Товар із підкріпленням	До продажу: наявна повна документація, акції на придбання декількох ліцензій, знижки для певних сегментів на покупку товару		
	Після продажу: додаткова підтримка спеціалістів налаштування, підтримка з боку розробника		
За рахунок чого потенційний товар буде захищено від копіювання: захист інтелектуальної власності, патент			

Таблиця 4.20 — Визначення меж встановлення ціни

Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
100000 грн\місяць	Високий(2000 грн/год або підписки)	> 1 000 000 грн	Від 0 до 1 000 000 грн\місяць залежно плану.

Таблиця 4.21 — Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Задоволення потреб установ, що обслуговують інформаційно-транспортні системи	Аналіз характеристик транспортного потоку за даними з камер відеоспостереження	Розповсюдження образу з зібраним програмним засобом, готовим до встановлення до системи клієнта	Залучена система збуту

Таблиця 4.22 — Концепція маркетингових комунікацій

Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
Установи та підприємства, які можуть мати різні ситуації, де може знадобитися наше рішення	Соцмережі, пошта, конференції	Індивідуальний підхід до кожного клієнта	Показати експертність та переваги нашого продукту	Перетворення відео з камери на корисні дані

Як результат було створено ринкову (маркетингову) програму, що включає в себе визначення ключових переваг концепції потенційного товару, опис моделі товару, визначення меж встановлення ціни, формування системи збуту та концепцію маркетингових комунікацій.

Висновки до розділу

У четвертому розділі проведено маркетинговий аналіз стартап-проєкту системи відеоаналізу характеристик транспортного потоку. Визначено, що ринок знаходиться на стадії зростання, а конкуренція є майже монополістичною. Головними конкурентами є DataFromSky (технологічний лідер) та TomTom (масштабний лідер). Обрана стратегія диференціації базується на використанні унікальних метрик та ціновій доступності. Цільовим сегментом обрано приватних замовників та проєктувальників, для яких важливе співвідношення ціна/якість.

ВИСНОВКИ

У роботі виконано комплексне дослідження методів, алгоритмів і програмних засобів відеомоніторингу характеристик автомобільного транспортного потоку. Розроблено й обґрунтовано методичні, алгоритмічні та програмні рішення, спрямовані на підвищення точності, швидкодії та автономності аналізу дорожнього трафіку в режимі близькому до реального часу.

У роботі визначено основні недоліки існуючих систем (зокрема, необхідність кількох моделей та ручного калібрування) та сформульовано вимоги до єдиного автоматизованого засобу оцінки характеристик дорожнього потоку.

Розроблено метод автоматичного виявлення дорожнього полотна, який поєднує геометричний аналіз сцени з класичними алгоритмами комп'ютерного зору. Метод не потребує ручного калібрування, стійкий до змін освітлення та погодних умов і працює в реальному часі .

Вдосконалено показник TLCR завдяки введенню мультикласової складової, що дозволяє враховувати вагові коефіцієнти впливу для різних типів транспортних засобів, забезпечуючи об'єктивну оцінку завантаженості. Запропоновано модифіковану версію M2TLCR, що використовує гомографічне перетворення для усунення перспективних спотворень .

Створено та натреновано власну модель на кастомному наборі даних, що забезпечило її адаптованість до реальних міських умов і змін освітлення. Розроблена система об'єднує функції семантичної сегментації, виявлення та трекінгу, що усуває необхідність використання кількох окремих моделей, зменшує обчислювальні витрати та підвищує узгодженість результатів.

Проведені експерименти підтвердили ефективність запропонованого підходу. Система забезпечує стабільну роботу в режимі близькому до реального часу, коректно визначає межі дорожнього полотна, виконує детекцію транспортних засобів різних типів, обчислює їх швидкість і рівень завантаженості смуг. При цьому досягається висока узгодженість між статичними та динамічними показниками, що дозволяє отримувати комплексну аналітичну інформацію про стан дорожнього руху.

Наукова новизна даної роботи полягає у вдосконаленні методу відеомоніторингу транспортного потоку шляхом створення інтегрованого підходу до аналізу дорожньої сцени, який поєднує класичні алгоритми комп'ютерного зору та сучасні моделі глибинного навчання. Запропоновано новий метод автоматичного визначення дорожнього полотна на основі бібліотеки OpenCV із використанням геометричного аналізу сцени, що забезпечує коректне виділення проїзної частини без необхідності ручного калібрування та підвищує стійкість до змін освітлення, тіней і перспективи. У межах дослідження розроблено мультикласовий показник TLCR, який дозволяє кількісно оцінювати завантаженість дорожніх смуг із урахуванням типів транспортних засобів, а також його модифіковану версію M2TLCR, що усуває вплив перспективних спотворень за рахунок застосування гомографічного перетворення. Крім того, вперше поєднано визначення статичних і динамічних характеристик дорожнього руху в межах єдиної об'єктної архітектури, яка забезпечує одночасне виявлення транспортних засобів, визначення їхніх швидкостей та обчислення показника TLCR за один прохід нейронної мережі. Запропонована адаптивна модель об'єктної сегментації створює можливість комплексного аналізу транспортного потоку з високою точністю та узгодженістю між різними параметрами.

Практичне значення отриманих результатів полягає у створенні програмного засобу, який може бути інтегрований у сучасні системи інтелектуального керування транспортом. Розроблене рішення дозволяє здійснювати моніторинг завантаженості доріг у режимі близькому до реального часу, оптимізувати світлофорні цикли та регулювати транспортні потоки залежно від поточних умов руху. Крім того, система може бути використана для прогнозування заторів і перевантажень, збору статистичних даних про швидкість, інтенсивність і структуру транспортного потоку, а також для аналізу ефективності функціонування транспортної інфраструктури. Запропоновані методи та алгоритми можуть бути адаптовані для моніторингу не лише автомобільного, але й пішохідного або велосипедного руху, що розширює можливості їх практичного застосування. Результати роботи створюють

науково-прикладну основу для подальшого розвитку систем автоматизованого відеомоніторингу, здатних у реальному часі забезпечувати комплексну оцінку стану транспортної інфраструктури міст і підтримувати процес прийняття управлінських рішень у сфері міського планування та транспортної безпеки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Jocher G., Chaurasia A., Qiu J. YOLO by Ultralytics (Version 8.0.0). *GitHub*. 2023. URL: <https://github.com/ultralytics/ultralytics> (date of access: 07.12.2025).
2. Stetsenko, I.V., Stelmakh, O. (2020). Traffic Lane Congestion Ratio Evaluation by Video Data. *Advances in Intelligent Systems and Computing* 1019, 172-181. Springer, Cham. https://doi.org/10.1007/978-3-030-25741-5_18
3. Савастру С.В., Стеценко І.В. (2023). Методи обробки даних відеокамер спостереження транспортного руху в реальному часі. *Міжвідомчий науково-технічний журнал «Адаптивні системи автоматичного управління»* 2 (43), 164-173. <https://doi.org/10.20535/1560-8956.43.2023.292269>
4. OpenCV: Open Source Computer Vision Library. URL: <https://docs.opencv.org/4.x/> (date of access: 07.12.2025).
5. Roboflow: Computer Vision Datasets & APIs. URL: <https://universe.roboflow.com/> (date of access: 07.12.2025).
6. J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.
7. Ren S., He K., Girshick R., Sun J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Advances in Neural Information Processing Systems (NIPS)*. 2015. URL: <https://arxiv.org/abs/1506.01497>.
8. He K., Gkioxari G., Dollár P., Girshick R. Mask R-CNN. *2017 IEEE International Conference on Computer Vision (ICCV)*. Venice, Italy, 2017. P. 2961–2969. URL: <https://arxiv.org/abs/1703.06870>.
9. Rother C., Kolmogorov V., Blake A. "GrabCut": interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (SIGGRAPH)*. 2004. Vol. 23, No. 3. P. 309–314.
10. Mukhopadhyay P., Chaudhuri B. B. A Survey of Hough Transform. *Pattern Recognition*. 2015. Vol. 48, Issue 3. P. 993–1010.

11. Lowe D. G. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*. 2004. Vol. 60, No. 2. P. 91–110.
12. Rublee E., Rabaud V., Konolige K., Bradski G. ORB: An efficient alternative to SIFT or SURF. *2011 International Conference on Computer Vision*. Barcelona, Spain, 2011. P. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.
13. Harris C., Stephens M. A Combined Corner and Edge Detector. *Proceedings of the 4th Alvey Vision Conference*. 1988. Vol. 15. P. 147–151.
14. What is AADT (Annual Average Daily Traffic). *StreetLight Data*. URL: <https://www.streetlightdata.com/what-is-aadt/> (date of access: 07.12.2025).
15. Von Gioi R. G., Jakubowicz J., Morel J.-M., Randall G. LSD: A Line Segment Detector. *Image Processing On Line*. 2012. Vol. 2. P. 35–55. URL: <https://doi.org/10.5201/ipol.2012.gjmr-lsd>.
16. Weisstein E. W. Laplacian. *MathWorld — A Wolfram Web Resource*. URL: <https://mathworld.wolfram.com/Laplacian.html> (date of access: 07.12.2025).
17. Data From Sky: Traffic Analysis Software. URL: <https://datafromsky.com/> (date of access: 07.12.2025).
18. TomTom: Location Technology Specialist. URL: <https://www.tomtom.com/> (date of access: 07.12.2025).
19. Python 3.12 Documentation. URL: <https://docs.python.org/3/> (date of access: 07.12.2025).
20. NumPy Documentation. URL: <https://numpy.org/> (date of access: 07.12.2025)

ДОДАТОК А

GitHub репозиторій: <https://github.com/romanpro23/TrafficAnalyzer>.

Лістинг коду

Лістинг А.1. - Модуль `cfg.py`

КОНФІГУРАЦІЯ МОДЕЛІ

```
model_path = "src/models/segmentation_model.pt"
```

НАЛАШТУВАННЯ ДЖЕРЕЛА ДАНИХ

```
use_stream = False
```

```
input_video = "src/data/real_test_2.mp4"
```

```
show_metrics = True
```

ВИХІДНА ДИРЕКТОРІЯ ДЛЯ ЗБЕРЕЖЕННЯ РЕЗУЛЬТАТІВ

```
output_dir = "src/out/real_test_2"
```

ПАРАМЕТРИ ROAD DETECTOR

```
fixed_road_pts = None          # полігон дорожнього полотна, якщо не вказано,  
автоматична детекція
```

```
smoothing_steps = 10          # кількість кадрів для згладжування
```

```
deviation_threshold = 80.0     # поріг відхилення для згладжування
```

```
refine_road_with_detections = False # згладжування дороги з використанням  
детекцій
```

```
rem_bg_filter_size = 21        # розмір гаусового фільтра для видалення фону  
зображення
```

```
slope_thr = 0.2                # поріг відхилення нахилу ліній для групування
```

```
bias_thr = 20                  # поріг відхилення зміщення для групування
```

```
min_length = 120               # мінімальна довжина лінії(або сума декількох ліній)  
для включення
```

```
min_count = 3                  # мінімальна кількість ліній у групі для включення
```

ПАРАМЕТРИ INSTANCE PROCESSOR

```
target_classes = [0, 1, 2]     # класи, які враховувати під час роботи
```

ПАРАМЕТРИ VISUALIZER

```
class_colors = {                # кольори масок для різних класів об'єктів
```

```
0: (255, 0, 0),
```

```
1: (0, 255, 255),
```

```
2: (0, 0, 255)
```

```
}
```

ПАРАМЕТРИ TLCR ANALYZER

```
default_weight = 1.0           # значення за замовчуванням коефіцієнту ваги класу
```

```
class_weights = {              # коефіцієнти класів для розрахунку MTLCR
```

```

0: 0.75,
1: 1.0,
2: 1.25
}

#### ПАРАМЕТРИ TRAFFIC ANALYZER
max_flow_speed = 0          # максимальна швидкість потоку(якщо 0 -
                              визначається динамічно)
flow_smoothing_factor = 0.2  # коефіцієнт імпульсу для згладжування
                              швидкості потоку
initial_speed = 0.01         # початкове значення швидкості
initial_speed_settled_rate = 0.95  # відсоток від максимальної динамічної
                              швидкості, для оцінки швидкості потоку
max_speed_threshold_factor = 3.0  # максимальний поріг швидкості об'єкту
                              між двома кадрами

```

Лістинг А.2. - Модуль `road_detector.py`

```

import cv2
import numpy as np
from collections import deque

import cfg

class RoadDetector:
    def __init__(self, width, height):
        self.width = width
        self.height = height
        self.lsd = cv2.createLineSegmentDetector(0)

        self.fixed_pts = cfg.fixed_road_pts
        self.deviation_threshold = cfg.deviation_threshold
        self.smoothing_steps = cfg.smoothing_steps
        self.refine_with_detections = cfg.refine_road_with_detections
        self.rem_bg_filter_size = cfg.rem_bg_filter_size
        self.slope_thr = cfg.slope_thr
        self.bias_thr = cfg.bias_thr
        self.min_length = cfg.min_length
        self.min_count = cfg.min_count

        # Черга для згладжування полігону дороги
        self.road_mask_history = deque(maxlen=self.smoothing_steps)

    def _rem_bg(self, gray):
        """Видалення фону за допомогою фільтра Гауса."""

```

```

blured = cv2.GaussianBlur(gray, (self.rem_bg_filter_size, self.rem_bg_filter_size),
0)
substracted = gray.astype(np.float32) - blured.astype(np.float32)
normalized = cv2.normalize(substracted, None, 0, 255, cv2.NORM_MINMAX,
dtype=cv2.CV_8U)
return normalized

def _estimate_horizon(self, lines):
    """Оцінює лінію горизонту на основі медіани у-координат точок перетину
ліній."""
    if lines is None or len(lines) < 2: return int(self.height * 0.5)

    intersections = []
    lines_list = np.squeeze(lines)
    if lines_list.ndim == 1: lines_list = lines_list [np.newaxis, :]

    for i in range(len(lines_list)):
        for j in range(i + 1, len(lines_list)):
            x1, y1, x2, y2 = lines_list [i]
            x3, y3, x4, y4 = lines_list [j]
            denom = (x1 - x2) * (y3 - y4) - (y1 - y2) * (x3 - x4)
            if abs(denom) < 1e-6: continue

            px_num = (x1 * y2 - y1 * x2) * (x3 - x4) - (x1 - x2) * (x3 * y4 - y3 * x4)
            py_num = (x1 * y2 - y1 * x2) * (y3 - y4) - (y1 - y2) * (x3 * y4 - y3 * x4)

            px = px_num / denom
            py = py_num / denom

            if -self.width * 0.1 < px < self.width * 1.1 and -self.height * 0.1 < py <
self.height * 1.1:
                intersections.append(py)

    if not intersections:
        return int(self.height * 0.5)
    return int(np.median(intersections))

def _detect_lines_lsd(self, gray_img):
    """Виявлення відрізків ліній за допомогою LSD."""
    lines, _, _, _ = self.lsd.detect(gray_img)
    return lines

def _merge_and_reconstruct_lines(self, lines, horizon_y):
    """Об'єднує схожі лінії та реконструює їх у форматі (x_h, y_h, x_b, y_b, k,
b)."""
    if lines is None or len(lines) == 0:

```

```

return np.array( [], dtype=np.float32)

lines = np.squeeze(lines)
if lines.ndim == 1: lines = lines [np.newaxis, :]

info = []
for x1, y1, x2, y2 in lines:
    dx, dy = x2 - x1, y2 - y1
    k = np.inf if abs(dx) < 1e-6 else dy / dx
    b = x1 if np.isinf(k) else y1 - k * x1
    length = np.hypot(dx, dy)
    info.append({"k": k, "b": b, "length": length, "count": 1})

merged = []
used = np.zeros(len(info), dtype=bool)
for i, li in enumerate(info):
    if used [i]: continue
    group_k, group_b, total_len, count = [li ["k"]], [li ["b"]], li ["length"], 1
    used [i] = True

    for j, lj in enumerate(info [i + 1:], start=i + 1):
        if used [j]: continue
        dk = abs(li ["k"] - lj ["k"]) if np.isfinite(li ["k"]) and np.isfinite(lj ["k"]) else 0
        db = abs(li ["b"] - lj ["b"]) if np.isfinite(li ["b"]) and np.isfinite(lj ["b"]) else 0

        if dk < self.slope_thr and db < self.bias_thr:
            total_len += lj ["length"]
            count += 1
            group_k.append(lj ["k"])
            group_b.append(lj ["b"])
            used [j] = True

    merged.append({
        "k": np.mean( [k for k in group_k if np.isfinite(k)]) if
np.any(np.isfinite(group_k)) else np.inf,
        "b": np.mean( [b for b in group_b if np.isfinite(b)]) if
np.any(np.isfinite(group_b)) else group_b [0],
        "length": total_len,
        "count": count
    })

filtered = [g for g in merged if g ["length"] > self.min_length or g ["count"] >=
self.min_count]

new_lines = []
for g in filtered:

```

```

k, b = g ["k"], g ["b"]
if np.isinf(k):
    x_h, x_b = b, b
else:
    x_h = (horizon_y - b) / k
    x_b = (self.height - b) / k

if x_b < -0.5 * self.width or x_b > 1.5 * self.width: continue

new_lines.append( [x_h, horizon_y, x_b, self.height, k, b])

return np.array(new_lines, dtype=np.float32)

def _get_bev_transform_local(self, left_info, right_info, horizon_y):
    """Розрахунок BEV трансформу для пари ліній."""
    pts_f = np.array( [
        [left_info ["x_h"], horizon_y],
        [right_info ["x_h"], horizon_y],
        [right_info ["x_b"], self.height],
        [left_info ["x_b"], self.height]
    ], dtype=np.float32)

    width_bev = int(np.linalg.norm(pts_f [1] - pts_f [0]))
    height_bev = int(np.linalg.norm(pts_f [3] - pts_f [0]))
    if width_bev < 10 or height_bev < 10: return None, 0, 0

    dst_pts = np.array( [
        [0, 0], [width_bev - 1, 0], [width_bev - 1, height_bev - 1], [0, height_bev - 1]
    ], dtype=np.float32)

    try:
        M = cv2.getPerspectiveTransform(pts_f, dst_pts)
    except cv2.error:
        return None, 0, 0
    return M, width_bev, height_bev

def _extract_road_mask(self, lines, horizon_y):
    """Визначає полігон дороги, використовуючи фільтрацію за перетинами та
    BEV-паралельність."""
    if lines is None or len(lines) < 2: return None, None, None

    info = [{"k": line [4], "b": line [5], "x_h": line [0], "x_b": line [2], "line": line} for
    line in lines]

    def find_intersections_under_horizon(lines_info):
        counts = np.zeros(len(lines_info), dtype=int)

```

```

for i in range(len(lines_info)):
    for j in range(i + 1, len(lines_info)):
        li, lj = lines_info [i], lines_info [j]
        k_i, k_j = li ["k"], lj ["k"]
        if (np.isinf(k_i) and np.isinf(k_j)) or (
            np.isfinite(k_i) and np.isfinite(k_j) and abs(k_i - k_j) < 1e-6): continue

        if np.isinf(k_i):
            x_int = li ["b"]
        elif np.isinf(k_j):
            x_int = lj ["b"]
        else:
            denom = k_i - k_j
            if abs(denom) < 1e-6: continue
            x_int = (lj ["b"] - li ["b"]) / denom

        if np.isfinite(k_i):
            y_int = k_i * x_int + li ["b"]
        elif np.isfinite(k_j):
            y_int = k_j * x_int + lj ["b"]
        else:
            continue

        if y_int > horizon_y:
            counts [i] += 1
            counts [j] += 1
    return counts

```

```

filtered_info = info [:]
max_iterations = len(filtered_info)
for _ in range(max_iterations):
    counts = find_intersections_under_horizon(filtered_info)
    max_c = np.max(counts) if len(counts) > 0 else 0
    if max_c == 0: break
    worst_idx = np.argmax(counts)
    filtered_info.pop(worst_idx)
    if len(filtered_info) < 2: return None, None, None

```

```

filtered_info.sort(key=lambda l: l ["x_h"])
lines_raw = np.array( [l ["line"] for l in filtered_info], dtype=np.float32)
pts1 = lines_raw[:, 0:2].reshape(-1, 1, 2)
pts2 = lines_raw[:, 2:4].reshape(-1, 1, 2)

```

```

best_count = -1
best_edges = None
best_mask = None

```

```

n = len(filtered_info)

for i in range(n):
    left_edge = filtered_info [i]
    for j in range(n - 1, i, -1):
        right_edge = filtered_info [j]

        M, w_bev, h_bev = self._get_bev_transform_local(left_edge, right_edge,
horizon_y)
        if M is None: continue

        p1_bev = cv2.perspectiveTransform(pts1, M)
        p2_bev = cv2.perspectiveTransform(pts2, M)

        vec = p2_bev - p1_bev
        dx = vec[:, 0, 0]
        dy = vec[:, 0, 1]
        angles = np.degrees(np.arctan2(dy, dx))
        dev = np.abs(np.abs(angles) - 90)

        is_parallel = dev < 5
        num_parallel = np.sum(is_parallel)

        if num_parallel > best_count:
            best_count = num_parallel
            best_edges = (left_edge, right_edge)
            best_mask = is_parallel

if best_edges is None: return None, None, None

left_edge, right_edge = best_edges
pts = np.array( [
    [left_edge ["x_h"], horizon_y],
    [right_edge ["x_h"], horizon_y],
    [right_edge ["x_b"], self.height],
    [left_edge ["x_b"], self.height]
], dtype=np.int32)

return pts, best_edges, filtered_info

def _refine_road_pts_with_detections(self, raw_road_data, detection_results):
    """
    Фільтрує лінії дороги - вибирає дві крайні лінії, між якими
    знаходиться максимальна кількість горизонтальних центрів виявлених ТЗ.
    """
    filtered_lines = raw_road_data.get("filtred_info")

```

```

horizon_y = raw_road_data.get("horizon_y", int(self.height * 0.5))

if not filtered_lines or len(filtered_lines) < 2:
    return raw_road_data ["pts"]

vehicle_centers_x = []
for result in detection_results:
    if 'box' in result:
        x_min, y_min, x_max, y_max = result ['box']
        if (y_min + y_max) / 2 > horizon_y:
            vehicle_centers_x.append((x_min + x_max) / 2)

line_x_positions = []
for line_info in filtered_lines:
    x_b = line_info ["line"] [2]
    line_x_positions.append({'line_info': line_info, 'x_pos': x_b})

max_covered_centers = -1
best_left_line_info = filtered_lines [0]
best_right_line_info = filtered_lines [-1]

if vehicle_centers_x:
    for i in range(len(line_x_positions)):
        for j in range(i + 1, len(line_x_positions)):
            left_line = line_x_positions [i]
            right_line = line_x_positions [j]

            if left_line ['x_pos'] >= right_line ['x_pos']: continue

            covered_count = sum(
                1 for center_x in vehicle_centers_x if left_line ['x_pos'] <= center_x <=
right_line ['x_pos'])

            current_width = right_line ['x_pos'] - left_line ['x_pos']

            if covered_count > max_covered_centers:
                max_covered_centers = covered_count
                best_left_line_info = left_line ['line_info']
                best_right_line_info = right_line ['line_info']
            elif covered_count == max_covered_centers and max_covered_centers > 0:
                best_width = best_right_line_info ['line'] [2] - best_left_line_info ['line'] [2]
                if current_width < best_width:
                    best_left_line_info = left_line ['line_info']
                    best_right_line_info = right_line ['line_info']

xlh, ylh, xlb, ylb = best_left_line_info ['line'] [:4]

```

```

xrh, yrh, xrb, yrb = best_right_line_info ['line'] [:4]

final_pts = np.array( [
    [xlh, ylh],
    [xrh, yrh],
    [xrb, yrb],
    [xlb, ylb]
], dtype=np.int32)
return final_pts

def _smooth_road_mask(self, raw_pts, combined_vehicle_mask):
    """Згладжує полігон дорожньої маски, використовуючи історію."""
    if raw_pts is None:
        return np.mean(self.road_mask_history, axis=0).astype(np.int32) if
self.road_mask_history else None

    if not self.road_mask_history:
        self.road_mask_history.append(raw_pts)
        return raw_pts

    avg_pts = np.mean(self.road_mask_history, axis=0)
    current_threshold = self.deviation_threshold

    # Динамічний поріг на основі перекриття з ТЗ
    if combined_vehicle_mask is not None:
        avg_road_mask = np.zeros((self.height, self.width), dtype=np.uint8)
        cv2.fillPoly(avg_road_mask, [avg_pts.astype(np.int32)], 255)
        road_pixel_count = np.count_nonzero(avg_road_mask)
        overlap_ratio = 0.0

        if road_pixel_count > 0:
            intersection = cv2.bitwise_and(combined_vehicle_mask,
combined_vehicle_mask, mask=avg_road_mask)
            overlap_ratio = np.count_nonzero(intersection) / road_pixel_count

        if overlap_ratio > 0.5:
            current_threshold = self.deviation_threshold * 0.1
        elif overlap_ratio > 0.2:
            current_threshold = self.deviation_threshold * 0.4

    # Перевірка відхилення
    dist = np.mean(np.linalg.norm(raw_pts - avg_pts, axis=1))

    if dist > current_threshold:
        # Якщо відхилення завелике (помилка), повертаємо старе середнє
        return avg_pts.astype(np.int32)

```

```

else:
    # Якщо прийнятне, оновлюємо історію та повертаємо нове середнє
    self.road_mask_history.append(raw_pts)
    return np.mean(self.road_mask_history, axis=0).astype(np.int32)

def detect_road(self, frame):
    """Основний метод детекції дороги. Виконує LSD, оцінку горизонту та BEV-
    фільтрацію."""
    # 1. Фіксований полігон
    if self.fixed_pts is not None:
        pts = self.fixed_pts
        horizon_y = int((pts[0][1] + pts[1][1]) / 2)
        return {"pts": pts, "horizon_y": horizon_y, "edges": None, "filtred_info": None}

    # 2. Детекція ліній та оцінка горизонту
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    gray_proc = self._rem_bg(gray)
    lines = self._detect_lines_lsd(gray_proc)

    rough_horizon_y = self._estimate_horizon(lines)

    # Попередня фільтрація горизонтальних ліній
    pre_filtered = []
    if lines is not None:
        lines_list = np.squeeze(lines)
        if lines_list.ndim == 1: lines_list = lines_list[np.newaxis, :]
        for line in lines_list:
            x1, y1, x2, y2 = line
            if (y1 + y2) / 2 < rough_horizon_y - 20: continue
            angle = np.degrees(np.arctan2(y2 - y1, x2 - x1))
            if abs(angle) < 15 or abs(angle) > (180 - 15): continue
            pre_filtered.append(line)

    bev_filtered_lines = np.array(pre_filtered) if pre_filtered else None

    final_lines = self._merge_and_reconstruct_lines(bev_filtered_lines,
    rough_horizon_y)
    pts, edges, filtered_info = self._extract_road_mask(final_lines, rough_horizon_y)

    return {
        "pts": pts,
        "horizon_y": rough_horizon_y,
        "edges": edges,
        "filtred_info": filtered_info # Список об'єднаних та відфільтрованих ліній
    }

```

```

def smooth_road(self, road_data, detection_results=None,
combined_vehicle_mask=None):
    """Уточнює дорожню маску за допомогою детекцій транспортних засобів та
згладжує її."""
    # Якщо використовується фіксований полігон, згладжування не потрібне
    if self.fixed_pts is not None:
        return road_data

    raw_pts = road_data.get("pts")

    # 1. Уточнення за детекціями
    if self.refine_with_detections and road_data.get(
        "filtred_info") is not None and detection_results and raw_pts is not None:
        refined_pts = self._refine_road_pts_with_detections(road_data, detection_results)
        raw_pts = refined_pts
        road_data["pts"] = raw_pts

    # 2. Згладжування полігону дорожнього полотна
    final_pts = self._smooth_road_mask(raw_pts, combined_vehicle_mask)

    if final_pts is not None:
        road_data["pts"] = final_pts
        road_data["horizon_y"] = int((final_pts[0][1] + final_pts[1][1]) / 2)

    return road_data

def get_bev_transform_info(self, pts):
    """Розраховує та повертає матрицю мерспективного трансформу та BEV
розміри."""
    if pts is None: return None, 0, 0
    src_pts = pts.astype(np.float32)

    width_bev = int(np.linalg.norm(pts[1] - pts[0]))
    height_bev = int(np.linalg.norm(pts[3] - pts[0]))
    if width_bev < 1 or height_bev < 1: return None, 0, 0

    dst_pts = np.array([
        [0, 0], [width_bev - 1, 0], [width_bev - 1, height_bev - 1], [0, height_bev - 1]
    ], dtype=np.float32)

    try:
        M = cv2.getPerspectiveTransform(src_pts, dst_pts)
    except cv2.error:
        return None, 0, 0
    return M, width_bev, height_bev

```

```

def transform_class_masks_to_bev(self, class_masks, M, w_bev, h_bev):
    """Трансформує словник масок об'єктів у простір BEV."""
    if M is None:
        return {}

    class_masks_bev = {
        cid: cv2.warpPerspective(mask, M, (w_bev, h_bev),
flags=cv2.INTER_NEAREST)
        for cid, mask in class_masks.items()
    }
    return class_masks_bev

def transform_mask_to_bev(self, combined_mask, M, w_bev, h_bev):
    """Трансформує маску об'єктів у простір BEV."""
    if M is None or combined_mask is None:
        return None

    bev_mask = cv2.warpPerspective(combined_mask, M, (w_bev, h_bev),
flags=cv2.INTER_NEAREST)
    return bev_mask

```

Лістинг А.3. - Модуль `instance_processor.py`

```

import cv2
import numpy as np
from ultralytics import YOLO

import cfg

class InstanceProcessor:
    def __init__(self):
        self.model = YOLO(cfg.model_path)
        self.target_classes = cfg.target_classes

    def process(self, frame):
        """Опрацювання зображення, результат - детекція та трекінг об'єктів на зображенні."""
        results = self.model.track(frame, verbose=False, classes=self.target_classes)
        return results [0]

    def get_masks_by_class(self, frame, detections):
        """Поєднує маски різних об'єктів одного класу в єдинну для кожного з класів."""
        height, width = frame.shape [:2]
        class_masks = {cls_id: np.zeros((height, width), dtype=np.uint8) for cls_id in

```

```

self.target_classes}

if detections.masks is not None:
    masks_data = detections.masks.data.cpu().numpy()
    classes = detections.bboxes.cls.cpu().numpy()

    for mask, class_id in zip(masks_data, classes):
        class_id = int(class_id)
        if class_id in self.target_classes:
            resized_mask = cv2.resize(mask.astype(np.uint8), (width, height),
interpolation=cv2.INTER_LANCZOS4)
            class_masks[class_id] = cv2.bitwise_or(class_masks[class_id], resized_mask
* 255)

    return class_masks

def combine_masks(self, class_masks):
    """Об'єднує маски різних класів в єдину сегментаційну маску."""
    if not class_masks:
        return None

    first_mask = next(iter(class_masks.values()))
    height, width = first_mask.shape

    combined_mask = np.zeros((height, width), dtype=np.uint8)
    for mask in class_masks.values():
        combined_mask = cv2.bitwise_or(combined_mask, mask)

    return combined_mask

```

Лістинг А.4. - Модуль `traffic_analyzer.py`

```

import cv2
import numpy as np
import cfg

class TrafficAnalyzer:
    def __init__(self, fps):
        self.fps = fps

        self.smoothing = cfg.flow_smoothing_factor
        self.initial_v = cfg.initial_speed
        self.v_rate = cfg.initial_speed_settled_rate
        self.speed_threshold_factor = cfg.max_speed_threshold_factor
        self.max_flow_speed_cfg = cfg.max_flow_speed

```

```

self.use_dynamic_speed = (cfg.max_flow_speed == 0) # чи розраховувати
динамічну швидкість потоку
self.last_positions_bev = {} # позиція об'єктів на попередньому кадрі,
для розрахунку зміщення та швидкості
self.average_relative_speed = 0.0 # згладжена середня відносна швидкість
self.v_max_obs = self.initial_v # максимальна спостережувана швидкість
потоку

def _get_bottom_center(self, box):
    """Повертає нижню центральну точку баундінг бокса."""
    x_min, y_min, x_max, y_max = box
    return int((x_min + x_max) // 2), int(y_max)

def calculate_flow_speed(self, detection_results, M, width_bev, frame_id,
road_pts):
    """Розрахунок швидкості потоку."""
    # Якщо немає BEV-матриці або області дороги — повертаємо нулі
    if M is None or width_bev == 0 or road_pts is None:
        return 0.0, 0.0, 0

    # Якщо трекінг не повернув ID — швидкість не розраховується
    if not hasattr(detection_results.bboxes, 'id') or detection_results.bboxes.id is None:
        return 0.0, self.average_relative_speed, 0

    track_ids = detection_results.bboxes.id.cpu().numpy()
    boxes_xyxy = detection_results.bboxes.xyxy.cpu().numpy().astype(int)

    road_poly = road_pts.reshape((-1, 1, 2))

    speeds_pps = []
    new_positions = {}
    vehicles_in_road = 0

    # Розрахунок швидкості розпізнаних об'єктів
    for track_id, box in zip(track_ids, boxes_xyxy):
        track_id = int(track_id)
        cx, cy = self._get_bottom_center(box)

        # Перевірка, що об'єкт всередині полігону дороги
        if cv2.pointPolygonTest(road_poly, (cx, cy), False) < 0:
            continue

        vehicles_in_road += 1

    # Трансформація точки в BEV простір

```

```

pt = np.array( [(cx, cy)], dtype=np.float32).reshape(-1, 1, 2)
bev_pt = cv2.perspectiveTransform(pt, M)

if bev_pt is None or bev_pt.shape [2] != 2:
    continue

x_bev, y_bev = bev_pt [0, 0]

# Розрахунок швидкості в BEV
if track_id in self.last_positions_bev:
    last_x, last_y = self.last_positions_bev [track_id]
    dist = np.hypot(x_bev - last_x, y_bev - last_y)
    speed_pps = dist * self.fps

# Фільтрація потенційних аномалій
if speed_pps < width_bev * self.speed_threshold_factor:
    speeds_pps.append(speed_pps)

# Оновлення позиції об'єкта
new_positions [track_id] = (x_bev, y_bev)

self.last_positions_bev = new_positions

# Середня швидкість потоку у пікселях
if speeds_pps:
    avg_pps = np.median(speeds_pps)
    rel_speed = avg_pps / width_bev
else:
    avg_pps = 0.0
    rel_speed = 0.0

# Динамічне визначення швидкості потоку
if self.use_dynamic_speed:
    self.v_max_obs = max(self.v_max_obs, avg_pps)
    v_norm = self.v_max_obs * self.v_rate / width_bev
else:
    v_norm = self.max_flow_speed_cfg / width_bev

# Згладжування швидкості
self.average_relative_speed = (self.average_relative_speed * (1 - self.smoothing) +
rel_speed * self.smoothing)

# Розрахунок індексу потоку
if v_norm > 0.01:
    flow_index = min(1.0, self.average_relative_speed / v_norm)
else:

```

```
flow_index = 0.0
```

```
return flow_index, self.average_relative_speed, vehicles_in_road
```

Лістинг А.5. - Модуль `tlcr_analyzer.py`

```
import cv2
```

```
import numpy as np
```

```
import cfg
```

```
class TLCRAnalyzer:
```

```
    def __init__(self):
```

```
        self.class_weights = cfg.class_weights
```

```
        self.default_weight = cfg.default_weight
```

```
    def calculate_m2tlcr(self, class_masks_bev):
```

```
        """Розрахунок модифікованого мультикласового показника TLCR."""
```

```
        if not class_masks_bev:
```

```
            return 0.0
```

```
        first_mask = next(iter(class_masks_bev.values()))
```

```
        if first_mask is None or first_mask.size == 0:
```

```
            return 0.0
```

```
        h, w = first_mask.shape
```

```
        total_pixels = h * w
```

```
        if total_pixels == 0:
```

```
            return 0.0
```

```
        weighted_area = 0.0
```

```
        for class_id, bev_mask in class_masks_bev.items():
```

```
            coeff = self.class_weights.get(class_id, self.default_weight)
```

```
            area = np.count_nonzero(bev_mask > 0)
```

```
            weighted_area += area * coeff
```

```
        return weighted_area / total_pixels
```

```
    def calculate_modtlcr(self, bev_mask):
```

```
        """Розрахунок модифікованого показника TLCR."""
```

```
        if bev_mask is None or bev_mask.size == 0:
```

```
            return 0.0
```

```
        total_area = bev_mask.size
```

```
        vehicle_area = np.count_nonzero(bev_mask > 0)
```

```

return vehicle_area / total_area if total_area > 0 else 0.0

def calculate_multy_tlcr(self, class_masks, road_mask):
    """Розрахунок мультикласового показника TLCR (MTLCR)."""
    road_pixels_count = cv2.countNonZero(road_mask)
    if road_pixels_count == 0 or not class_masks:
        return 0.0

    weighted_tlcr = 0.0
    for class_id, mask in class_masks.items():
        intersection_area = cv2.countNonZero(cv2.bitwise_and(mask, mask,
mask=road_mask))
        coeff = self.class_weights.get(class_id, self.default_weight)
        weighted_tlcr += (intersection_area / road_pixels_count) * coeff

    return weighted_tlcr

def calculate_tlcr(self, combined_mask, road_mask):
    """Розрахунок показника TLCR."""
    road_pixels_count = cv2.countNonZero(road_mask)
    if road_pixels_count == 0 or combined_mask is None:
        return 0.0

    intersection_area = cv2.countNonZero(cv2.bitwise_and(combined_mask,
combined_mask, mask=road_mask))
    tlcr = intersection_area / road_pixels_count

    return tlcr

```

Лістинг А.6. - Модуль visualizer.py

```

import cv2
import numpy as np

import cfg

class Visualizer:
    def __init__(self, width, height):
        self.width = width
        self.height = height
        self.CLASS_COLORS = cfg.class_colors

    def visualize_instance_masks(self, output_frame, class_masks, road_mask):
        """Накладає кольорові маски виявлених об'єктів на дорожнє полотно."""

```

```

for cid, mask in class_masks.items():
    overlay_mask = cv2.bitwise_and(mask, mask, mask=road_mask)

    if np.any(overlay_mask > 0):
        color = self.CLASS_COLORS.get(cid, (255, 255, 255))

        full_color_overlay = np.full(output_frame.shape, color,
dtype=output_frame.dtype)

        mask_3ch = overlay_mask > 0
        if output_frame.ndim == 3:
            mask_3ch = np.stack( [mask_3ch] * output_frame.shape [2], axis=-1)

        alpha = 0.5

        # Обчислення змішування з використанням NumPy
        original_slice = output_frame [mask_3ch].reshape(-1, output_frame.shape [2])
        color_slice = full_color_overlay [mask_3ch].reshape(-1,
full_color_overlay.shape [2])

        blended_slice = (original_slice * (1 - alpha) + color_slice *
alpha).astype(output_frame.dtype)

        output_frame [mask_3ch] = blended_slice.flatten()

    return output_frame

def visualize_road_mask(self, frame, road_data, mask_color=(0, 255, 0),
alpha=0.3):
    """Накладає напівпрозору маску дороги та горизонт на кадр."""
    pts = road_data.get("pts")
    horizon_y = road_data.get("horizon_y", int(self.height * 0.5))
    output = frame.copy()

    if pts is not None:
        mask = np.zeros_like(frame, dtype=np.uint8)
        cv2.fillPoly(mask, [pts.astype(np.int32)], mask_color)
        output = cv2.addWeighted(output, 1.0, mask, alpha, 0)

    cv2.polylines(output, [pts], isClosed=True, color=(0, 0, 255), thickness=2)

    cv2.line(output, (0, horizon_y), (self.width, horizon_y), (255, 0, 0), 2)
    return output

```

Лістинг А.7. - Модуль main.py

```

import csv
import os
import cv2
import numpy as np
import time
import concurrent.futures

import cfg
from road_detector import RoadDetector
from instance_processor import InstanceProcessor
from tlc_r_analyzer import TLCRAnalyzer
from traffic_analyzer import TrafficAnalyzer
from visualizer import Visualizer

# --- Налаштування ---
input_source = cfg.input_video
output_dir = cfg.output_dir
use_stream = cfg.use_stream

# Ініціалізація джерела відео
cap = cv2.VideoCapture(0) if use_stream else cv2.VideoCapture(input_source)
if not cap.isOpened():
    source_name = "камери" if use_stream else input_source
    raise RuntimeError(f"Не вдалося відкрити джерело відео: {source_name}")

# Визначення розмірів та масштабування
fps = cap.get(cv2.CAP_PROP_FPS) or 30 if use_stream else
cap.get(cv2.CAP_PROP_FPS)

scale = 640. / max(cap.get(cv2.CAP_PROP_FRAME_WIDTH),
cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH) * scale)
height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT) * scale)

os.makedirs(output_dir, exist_ok=True)
output_video_path = os.path.join(output_dir, "demo.mp4")
csv_path = os.path.join(output_dir, "metrics.csv")

fourcc = cv2.VideoWriter_fourcc(*'mp4v')
out = cv2.VideoWriter(output_video_path, fourcc, fps, (width, height))

# Ініціалізація
road_detector = RoadDetector(width, height)
instance_processor = InstanceProcessor()
tlcr_analyzer = TLCRAnalyzer()
traffic_analyzer = TrafficAnalyzer(fps=fps)

```

```
visualizer = Visualizer(width, height)
```

```
# CSV
```

```
csv_header = ["frame_id", "MTLCR", "TLCR", "M2TLCR", "ModTLCR", "Flow",  
"Rel_Speed", "Vehicles_Count"]
```

```
csv_file = open(csv_path, 'w', newline="", encoding='utf-8')
```

```
csv_writer = csv.writer(csv_file)
```

```
csv_writer.writerow(csv_header)
```

```
# Головний цикл
```

```
frame_id = 0
```

```
total_processing_time = 0
```

```
start_time = time.time()
```

```
try:
```

```
    while True:
```

```
        frame_start_time = time.time()
```

```
        ret, frame = cap.read()
```

```
        if not ret:
```

```
            break
```

```
        frame = cv2.resize(frame, (width, height), interpolation=cv2.INTER_LINEAR)
```

```
# ПАРАЛЕЛЬНА ДЕТЕКЦІЯ (RoadDetector та InstanceProcessor)
```

```
with concurrent.futures.ThreadPoolExecutor(max_workers=2) as executor:
```

```
    future_detections = executor.submit(instance_processor.process, frame)
```

```
    future_road_data = executor.submit(road_detector.detect_road, frame)
```

```
    detection_results = future_detections.result()
```

```
    road_data_raw = future_road_data.result()
```

```
class_masks = instance_processor.get_masks_by_class(frame, detection_results)
```

```
combined_mask = instance_processor.combine_masks(class_masks)
```

```
# 1. Уточнення та згладжування дороги
```

```
road_data = road_detector.smooth_road(road_data_raw, detection_results,  
combined_mask)
```

```
output_frame = visualizer.visualize_road_mask(frame, road_data)
```

```
# 2. Перевірка наявності полігону
```

```
if road_data ["pts"] is None or road_data ["pts"].size == 0:
```

```
    metrics = [0, 0, 0, 0, 0, 0, 0]
```

```
else:
```

```
    pts = road_data ["pts"]
```

```
    M, w_bev, h_bev = road_detector.get_bev_transform_info(pts)
```

```

class_masks_bev = road_detector.transform_class_masks_to_bev(class_masks,
M, w_bev, h_bev)
bev_mask = road_detector.transform_mask_to_bev(combined_mask, M, w_bev,
h_bev)
road_mask = cv2.fillPoly(np.zeros((height, width), dtype=np.uint8),
[pts.astype(int)], (255,))

# Метрики
avg_m2tlcr = tlcr_analyzer.calculate_m2tlcr(class_masks_bev)
avg_mod_tlcr = tlcr_analyzer.calculate_modtlcr(bev_mask)
flow_index, rel_speed, vehicles_count =
traffic_analyzer.calculate_flow_speed(detection_results, M, w_bev, frame_id, pts)
tlcr = tlcr_analyzer.calculate_tlcr(combined_mask, road_mask)
multy_tlcr = tlcr_analyzer.calculate_multy_tlcr(class_masks, road_mask)

output_frame = visualizer.visualize_instance_masks(output_frame, class_masks,
road_mask)
metrics = [multy_tlcr, tlcr, avg_m2tlcr, avg_mod_tlcr, flow_index, rel_speed,
vehicles_count]
# Візуалізація
frame_end_time = time.time()
processing_time = frame_end_time - frame_start_time
total_processing_time += processing_time
avg_fps = frame_id / total_processing_time if total_processing_time > 0 else 0

if show_metrics:
    cv2.putText(output_frame, f"MTLCR: {min(1., metrics [1]):.3f} (TLCR:
{metrics [0]:.3f})", (10, 30),
    cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 2)
    cv2.putText(output_frame, f"M2TLCR: {min(1., metrics [2]):.3f} (Flow:
{metrics [4]:.3f})", (10, 50),
    cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 2)

csv_writer.writerow( [frame_id] + metrics)
if not use_stream:
    out.write(output_frame)
    print(f"Processed frame {frame_id}, {processing_time*1000:.1f} ms")
else:
    out.write(output_frame)
    cv2.imshow("Road Traffic Analysis (Press 'q' to exit)", output_frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

frame_id += 1
finally:

```

```
cap.release()
out.release()
csv_file.close()
if use_stream:
    cv2.destroyAllWindows()

end_time = time.time()
total_time = end_time - start_time
final_avg_fps = frame_id / total_time if total_time > 0 else 0

print("=" * 50)
print(f"Processing complete.")
print(f"Total Frames: {frame_id}")
print(f"Output saved to directory: {output_dir}")
print(f"Average FPS over run: {final_avg_fps:.2f}")
print(f"Average Inference Time: {1000. / final_avg_fps:.1f} ms")
```

ДОДАТОК Б

Результати перевірки роботи на співпадіння



Дата звіту 12/7/2025
Дата редагування 12/8/2025

Документ прийнятий

Звіт подібності

Метадані

Назва організації
National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute
Заголовок
ІП_42_мп_Проценко_ПЗ
Автор Науковий керівник / Експерт
ПроценкоСтеценко І.В.
підрозділ
ФІОТ, К-а інформатики та програмної інженерії

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.



10685
Кількість слів

84995
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		0
Інтервали		0
Мікропробіли		9
Білі знаки		43
Парафрази (SmartMarks)		3

Джерела

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://ela.kpi.ua/bitstream/123456789/19007/1/Dubinina_aref.pdf	10 0.09 %

2	http://dusfsu-nnip.com/wp-content/uploads/2020/05/%D0%97%D0%B1%D1%96%D1%80%D0%BD%D0%B8%D0%BA-%D1%82%D0%B5%D0%B7-V-%D0%9C%D1%96%D0%B6%D0%BD%D0%B0%D1%80%D0%BE%D0%B4%D0%BD%D0%BE%D1%97-%D0%BD%D0%B0%D1%83%D0%BA%D0%BE%D0%B2%D0%BE-%D0%BF%D1%80%D0%B0%D0%BA%D1%82%D0%B8%D1%87%D0%BD%D0%BE%D1%97-%D1%96%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D0%BA%D0%BE%D0%BD%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86%D1%96%D1%97-2019-%D1%80.pdf	7 0.07 %
3	http://ena.lp.edu.ua:8080/bitstream/ntb/45105/3/dysertaciya_kravchysbyn.pdf	6 0.06 %
4	http://megalib.com.ua/content/10591_Kriterii_priinyattiya_rishen_za_ymov_riziky.html	6 0.06 %
5	https://ela.kpi.ua/bitstream/123456789/57453/1/Matvieiev_bakalavr.pdf	5 0.05 %
6	http://dusfsu-nnip.com/wp-content/uploads/2020/05/%D0%97%D0%B1%D1%96%D1%80%D0%BD%D0%B8%D0%BA-%D1%82%D0%B5%D0%B7-V-%D0%9C%D1%96%D0%B6%D0%BD%D0%B0%D1%80%D0%BE%D0%B4%D0%BD%D0%BE%D1%97-%D0%BD%D0%B0%D1%83%D0%BA%D0%BE%D0%B2%D0%BE-%D0%BF%D1%80%D0%B0%D0%BA%D1%82%D0%B8%D1%87%D0%BD%D0%BE%D1%97-%D1%96%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D0%BA%D0%BE%D0%BD%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86%D1%96%D1%97-2019-%D1%80.pdf	5 0.05 %
7	http://dusfsu-nnip.com/wp-content/uploads/2020/05/%D0%97%D0%B1%D1%96%D1%80%D0%BD%D0%B8%D0%BA-%D1%82%D0%B5%D0%B7-V-%D0%9C%D1%96%D0%B6%D0%BD%D0%B0%D1%80%D0%BE%D0%B4%D0%BD%D0%BE%D1%97-%D0%BD%D0%B0%D1%83%D0%BA%D0%BE%D0%B2%D0%BE-%D0%BF%D1%80%D0%B0%D0%BA%D1%82%D0%B8%D1%87%D0%BD%D0%BE%D1%97-%D1%96%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D0%BA%D0%BE%D0%BD%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86%D1%96%D1%97-2019-%D1%80.pdf	5 0.05 %
з домашньої бази даних (0.00 %)		
ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
з програми обміну базами даних (0.00 %)		
ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
з Інтернету (0.41 %)		
ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	http://dusfsu-nnip.com/wp-content/uploads/2020/05/%D0%97%D0%B1%D1%96%D1%80%D0%BD%D0%B8%D0%BA-%D1%82%D0%B5%D0%B7-V-%D0%9C%D1%96%D0%B6%D0%BD%D0%B0%D1%80%D0%BE%D0%B4%D0%BD%D0%BE%D1%97-%D0%BD%D0%B0%D1%83%D0%BA%D0%BE%D0%B2%D0%BE-%D0%BF%D1%80%D0%B0%D0%BA%D1%82%D0%B8%D1%87%D0%BD%D0%BE%D1%97-%D1%96%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D0%BA%D0%BE%D0%BD%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86%D1%96%D1%97-2019-%D1%80.pdf	17 (3) 0.16 %
2	https://ela.kpi.ua/bitstream/123456789/19007/1/Dubinina_aref.pdf	10 (1) 0.09 %
3	http://ena.lp.edu.ua:8080/bitstream/ntb/45105/3/dysertaciya_kravchysbyn.pdf	6 (1) 0.06 %
4	http://megalib.com.ua/content/10591_Kriterii_priinyattiya_rishen_za_ymov_riziky.html	6 (1) 0.06 %
5	https://ela.kpi.ua/bitstream/123456789/57453/1/Matvieiev_bakalavr.pdf	5 (1) 0.05 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)
------------------	-------	---------------------------------------