Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Програмної інженерії
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: «Розробка програмної частини системи запобігання зіткненню для великогабаритної техніки»

Виконав: студент IV курсу, групи СПз-41
спеціальності 121 – Інженерія програмного забезпечення
(шифр і назва спеціальності)

(підпис) Стойко Д.І. (прізвище та ініціали)

Керівник (підпис) Пастух О.А. (прізвище та ініціали)

Нормоконтроль (підпис) Стоянов Ю.М. (прізвище та ініціали)

Завідувач кафедри (підпис) Петрик М.Р. (прізвище та ініціали)

Рецензент (підпис) Осухіська Г.М. (прізвище та ініціали)

Тернопіль
2026

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра програмної інженерії
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ проф. Петрик М.Р.
(підпис) (прізвище та ініціали)
« » 20___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавр
(назва освітнього ступеня)

за спеціальністю 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

студенту Стойку Денису Ігоровичу

1. Тема роботи Розробка програмної частини системи запобігання зіткненню для великогабаритної техніки

Керівник роботи Пастух Олег Анатолійвич, д.т.н., проф.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом по університету від « 06 » квітня 2026 року № 4/9-173

2. Термін подання студентом роботи 11.06.2026

3. Вихідні дані до роботи наукові літературні джерела

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1 Огляд предметної області.
2. Проєктування та реалізація.
3. Тестування.
4. Безпека життєдіяльності, основи охорони праці

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Тема роботи. 2. Актуальність, мета, задачі дослідження
3. Існуючі технології реалізації подібних систем.
4. Функціональні та нефункціональні вимоги .5. Загальна архітектура системи.
6. Варіанти використання. 7. Компоненти програми для налаштування параметрів системи.
8. Програмні засоби та технології. 9. Інтерфейси реалізації застосунку..
10. Фрагмент коду модуля під'єднання до апаратної частини системи.
11. Алгоритм надсилання повідомлень. 12. Результати навчання нейронної мережі.
13. Тестування. 14. Висновки по роботі.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці			

7. Дата видачі завдання _____ 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1.	Аналіз предметної галузі дослідження	06.04–17.04.26	Виконано
2.	Обґрунтування актуальності дослідження	18.04–26.04.26	Виконано
3.	Огляд існуючих механізмів та засобів	27.04–04.05.26	Виконано
4.	Проведення дослідження механізмів та засобів		
	опрацювання даних	05.05–10.05.26	Виконано
5.	Оформлення розділу «Огляд предметної області»	11.05–14.05.26	Виконано
6.	Оформлення розділу «Проєктування та реалізація »	15.05–18.05.26	Виконано
7.	Оформлення розділу «Тестування»	19.05–22.05.26	Виконано
8.	Оформлення розділу «Безпека життєдіяльності,		
	основи охорони праці»	23.05–26.05.26	Виконано
9.	Нормоконтроль	25.05–04.06.26	Виконано
10.	Перевірка на плагіат	05.06–12.06.26	Виконано
11.	Попередній захист роботи	11.06.26	Виконано
12.	Захист кваліфікаційної роботи	18.06.26	

Студент _____ Стойко Д.І.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Пастух О.А.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка програмної частини системи запобігання зіткненню для великогабаритної техніки // Кваліфікаційна робота освітнього рівня «бакалавр» // Стойко Денис Ігорович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем та програмної інженерії, кафедра програмної інженерії, група СПз-41 // Тернопіль, 2026 // С. - 51, табл. - 3, рис. - 17, слайд. – 14, додат. - , бібліогр. – 31.

Ключові слова: бібліотека, давач, інтерфейс, обмін даними, CAN-шина, Python, YOLO

У першому розділі досліджено предметну область, розглянуто наявні на даний момент аналогічні системи запобігання зіткнень, виконано огляд доступних технологій їх втілення.

У другому розділі були визначені вимоги, котрі необхідні для розробки кожного із програмних компонентів системи (графічного інтерфейсу, обміну даними через CAN-шину, нейромережі YOLO). Створені діаграми варіантів використання. Реалізований графічний інтерфейс користувача, котрий забезпечує під'єднання до апаратної частини системи по мережі. Втілено опрацювання помилок при допомозі спливаючих вікон, реалізовано методи для спрощення процесу застосування застосунку. Забезпечено ефективний обмін даними із найменшими часовими затримками та обробкою помилок. Навчена нейронна мережа, котра забезпечує високу точність і швидкість виявлення ймовірного зіткнення.

Третій розділ присвячено тестуванню компонентів системи за допомогою розроблених функціональних тестів. Усі створені тести пройшли успішне виконання.

Четвертий розділ висвітлює важливі питання безпеки життєдіяльності та основ охорони праці.

ABSTRACT

Development of the software part of the collision avoidance system for large-sized equipment // Bachelor Thesis // Stoiko Denys // Ternopil Ivan Puluj National Technical University, Computer and Information Systems and Software Engineering Faculty, Department of Software Engineering, group SPs-41 // Ternopil, 2026 // P. - 51, tab. - 3, fig. - 17, slide -14, annexes - , references - 31.

Keywords: library, sensor, interface, data exchange, CAN-bus, Python, YOLO

The first section explores the subject area, considers the currently available similar collision avoidance systems, and reviews available technologies for their implementation.

The second section identifies the requirements necessary for the development of each of the system's software components (graphical interface, data exchange via CAN bus, YOLO neural network). Use case diagrams are created. A graphical user interface is implemented that provides connection to the system hardware over the network. Error handling using pop-up windows is implemented, methods are implemented to simplify the application application process. Effective data exchange with the lowest time delays and error handling is provided. A neural network is trained that provides high accuracy and speed of detecting a potential collision.

The third deals with the testing system components using the developed functional tests. All created tests were successfully completed.

The fourth section highlights important issues of life safety and the basics of labor protection.

ПЕРЕЛІК СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ – штучний інтелект.

ARP (англ. Address Resolution Protocol) – протокол визначення адрес, фундаментальний мережевий протокол, який пов'язує логічні IP-адреси (мережевий рівень) з фізичними MAC-адресами (канальний рівень) у локальних мережах.

CAN-шина (англ. Controller Area Network) — надійна цифрова мережа для обміну даними між електронними блоками керування транспортного засобу, що працює як "нервова система".

OpenCV (англ. Open Source Computer Vision Library) – бібліотека алгоритмів комп'ютерного зору, обробки зображень та чисельних алгоритмів загального призначення з відкритим кодом.

YOLO (агл. You Only Look Once) – нейромережа, яка здатна аналізувати відео або фото в реальному часі та визначати об'єкти на зображенні – від облич і автомобілів до упаковок товарів і медичних знімків.

ЗМІСТ

ВСТУП

Актуальність теми дослідження. Сучасний транспорт переживає суттєві зміни, спричинені активним розвитком автономних технологій. Впровадження безпілотних транспортних засобів та автономних систем керування стає все більш поширеним явищем. Постає потреба в ефективних системах запобігання зіткнень. У зв'язку з цим, розробка програмних компонентів, здатних взаємодіяти з автономними системами, стає ключовою задачею, при забезпеченні узгодженості в роботі всього транспортного парку.

При цьому, великогабаритна техніка, що виконує роботу в обмежених просторах, наприклад, на сільськогосподарських угіддях або на будівництві, частіше стикається з високим ризиком зіткнень, і як наслідок, пошкоджень техніки, що зі свого боку тягне за собою збільшення термінів виконання завдань та збільшення фінансових витрат. Також робота в таких умовах не тільки загрожує безпеці обладнання, а й може становити небезпеку як для операторів даної техніки, так і навколишнього середовища. Як наслідок, розробка ефективних систем запобігання зіткнень стає невід'ємною для запобігання пошкодженням, що врешті-решт призведе до скорочення операційних витрат і підвищення продуктивності у сфері використання крупногабаритної техніки.

Завдання розробки системи запобігання зіткнень поділяється на розробку програмної та апаратної частини. Задачею ж кваліфікаційної роботи є саме розробка програмних компонентів, котрі забезпечують коректну роботу системи.

Метою роботи є розробка програмних компонентів для системи автономного запобігання зіткнення для великогабаритної техніки.

Були сформульовані завдання, виконання яких необхідне для досягнення поставленої мети:

- виконати аналіз предметної галузі та провести огляд аналогів;
- спроектувати компоненти системи;
- реалізувати компоненти;
- провести тестування.

1 ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

Ця робота заснована на сучасних технологічних тенденціях у транспортній галузі, де на даний момент відбувається активне впровадження автономних транспортних засобів та систем управління. У контексті великогабаритної техніки, такої як будівельна та сільськогосподарська, розробка програмних компонентів для систем запобігання зіткненням стає одним із ключових завдань, через високий ризик зіткнень [1].

Ця тема охоплює кілька важливих аспектів. Основний з них - зменшення аварій і пошкоджень великогабаритної техніки є важливим моментом, який вирішується розробкою ефективних систем запобігання зіткнень, заснованих на поєднанні сучасних давачів та ШІ.

Одним з основних напрямків є також оптимізація операцій великогабаритної техніки. Розробка програмних компонентів спрямована на зниження ризиків простоїв, підвищення ефективності використання обладнання та, зрештою, створення систем, що сприяють економічному ефекту від впровадження автономних рішень у даній предметній галузі.

1.1 Огляд аналогів

У ході проведеного огляду аналогів було виявлено кілька систем зі схожим функціоналом.

1.1.1 SICK Safety Collision Prevention System

Система [2] є комплексом давачів, що включають лазерні і радарні пристрої, здійснює активний моніторинг навколишнього середовища та детектує перешкоди в реальному часі. Інтеграція штучного інтелекту в алгоритми обробки даних забезпечує адаптивність до умов, котрі змінюються, таким чином підвищуючи ефективність запобігання зіткнень Ілюстрація принципу роботи системи представлена на рисунку 1.1.

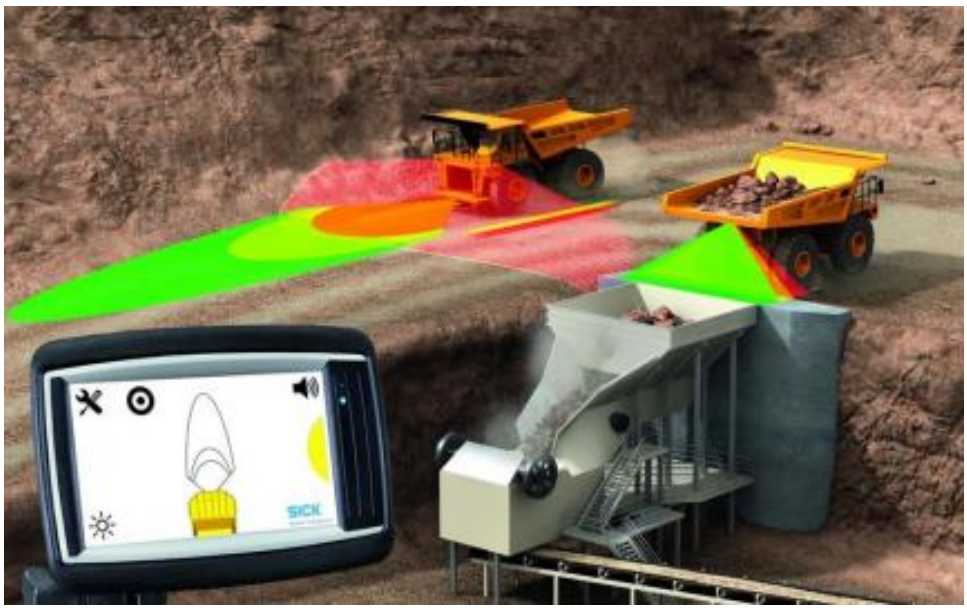

Рисунок 1.1 – Ілюстрація принципу роботи "SICK Safety Collision Prevention"

1.1.2 Caterpillar Cat Detect Safety System

Система [3] є комплексним рішенням для автоматизації контролю ситуації на будівельних майданчиках, як давачі використовуються камери і радари для виявлення перешкод при роботі спеціалізованої техніки. Інтелектуальні алгоритми обробки зображень і даних із радарів дозволяють класифікувати об'єкти, визначати їх наявність та траєкторію їх руху та видавати попередження про присутність об'єктів у полі руху машини. Автоматичне гальмування у разі небезпеки також підтримується ШІ.

Інтерфейс програмної частини системи, що використовується оператором для контролю ситуації, що відбувається, представлений на рисунку 1.2.

Рисунок 1.2 – Інтерфейс системи «Caterpillar Cat Detect Safety»

1.1.3 Komatsu Intelligent Machine Control

Система [4] містить GPS, камери і пристрій, що сканує навколишній простір за допомогою лазерних променів для створення тривимірної мапи зовнішнього середовища. Застосовуючи ті дані, котрі зібрані з цих пристроїв, алгоритми ШІ визначають перешкоди та приймають рішення про необхідні корекції маршруту та швидкості, що також забезпечує автоматичне запобігання зіткнень техніки при роботі і, як наслідок, безпечне пересування спеціалізованої техніки по будівельному майданчику. Інтерфейсом програмної частини системи є застосунок, який дозволяє оператору найбільш точно відстежувати ситуацію в режимі реального часу за рахунок огляду камер на 360 градусів.

На рисунку 1.3 представлений графічний інтерфейс для спрощення роботи оператора системи.

Рисунок 1.3 – Інтерфейс системи «Komatsu Intelligent Machine Control»

1.1.4 John Deere ActiveCollision Avoidance System

Система [5] використовує передові давачі та алгоритми машинного навчання. ШІ забезпечує більш точне розпізнавання об'єктів, а також адаптацію системи до різних умов та типів перешкод. У разі можливого зіткнення система може автоматично втручатися з керуванням або гальмуванням.

Інтерфейс програмної частини системи представлений на рисунку 1.4.

Рисунок 1.4 – Інтерфейс системи «John Deere ActiveCollision Avoidance»

За підсумками огляду аналогів було проведено порівняльний аналіз

описаних систем. Як загальні висновки можна відзначити, що кожна з систем використовує поєднання певних давачів та ШІ.

Таким чином, система «SICK Safety Collision Prevention» використовує лазерні та радарні давачі, що дає їй можливість швидко реагувати та виявляти перешкоди з високою точністю у широкому діапазоні погодних умов.

Система «Caterpillar Cat Detect Safety» як давачі використовує камери та радари, що допомагає ефективно реалізувати попередження користувача про наявність перешкоди та автоматичне гальмування.

Система Komatsu Intelligent Machine Control використовує давачі GPS, камери та пристрій, що сканує простір за допомогою лазерних променів, що дає можливість визначення точного місцезнаходження миттєво, автоматичного керування та автоматичного регулювання траєкторії та швидкості руху.

Також варто відзначити, що система John Deere Active Collision Avoidance відрізняється автоматичним втручанням завдяки використанню ШІ

В результаті аналізу різних систем автономного запобігання зіткнень для великогабаритної техніки видно, що кожна з систем збільшує ефективність використання техніки завдяки використанню спеціалізованих давачів та ШІ.

1.2 Огляд існуючих технологій реалізації

Як основні існуючі технології реалізації подібних систем, можна виділити такі:

– сенсори і давачі для контролю ситуації;

– комп'ютерний зір для обробки інформації, що надходить з давачів;

– технології зв'язку, які забезпечують обмін даними;

– операційні системи для вбудованих пристроїв, необхідних для забезпечення автономності;

– багатозадачність та паралельні обчислення для забезпечення високої швидкості роботи.

Найчастіше, у ролі мінімально необхідних давачів виступають камери.

Оскільки за використання алгоритмів комп'ютерного зору вони відіграють найважливішу роль при розпізнаванні об'єктів. Однак найбільш якісний результат визначення та детектування об'єктів досягається при використанні камер спільно з лідаром. Сам собою лідар є радаром, заснованим у світлі, з якого система дізнається яскравість і дальність цілі

Для виявлення перешкод, аналізу оточуючого світу і прийняття рішень з огляду на візуальну інформацію потрібне використання комп'ютерного зору. На даний момент найбільш ефективним засобом реалізації цієї мети є бібліотека OpenCV, яка є найбільш розвиненою бібліотекою для роботи із зображеннями та відеоматеріалами.

Машинне навчання та ШІ застосовуються для обробки даних, прийняття рішень та адаптації системи до умов, що змінюються. Ці технології покращують точність виявлення та класифікації об'єктів, а також сприяють ефективній адаптації до різних сценаріїв. В даному контексті, як найбільш перевірені технології можна виділити передбачену нейронну мережу YOLO [6], основним принципом якої є найбільш ефективне за поєднанням швидкості та якості визначення об'єктів.

Бездротові технології зв'язку забезпечують ефективну взаємодію між компонентами системи, включаючи обмін даними із зовнішніми джерелами. Операційні системи для вбудованих пристроїв гарантують стабільну роботу системи при обмежених обчислювальних ресурсах, забезпечуючи оптимальне функціонування.

Використання мобільних платформ, таких як Jetson Nano [7] та Raspberry Pi, додає високу продуктивність і компактність у систему, що стає ключовим фактором у контексті автономних технічних пристроїв.

Отже можна виділити три ключові технології:

– мова програмування Python;
– CAN -шина для обміну даними
– YOLO як нейромережа.

Використання мови програмування Python у проекті надає зручність та гнучкість у розробці програмних компонентів. Python дозволяє легко інтегрувати

різні технології, такі як комп'ютерний зір, машинне навчання та взаємодія з CAN - ш ін.

CAN -шина, зі свого боку, є важливим компонентом для забезпечення обміну інформацією між камерами та вузлом управління машини. Тому вона є основною для втілення узгодженого функціонування системи.

Нейромережа YOLO забезпечує точне детектування та класифікацію детектованих об'єктів негайно. Її висока ефективність істотно покращує якість роботи системи автономного запобігання зіткненню.

Таким чином, спільний вплив даних засобів реалізації створює сучасне та ефективне рішення та забезпечує високу точність, швидку передачу даних та адаптивність до різних сценаріїв руху.

1.3 Висновки до першого розділу

В цьому розділі описується предметна область, розглядаються існуючі на даний момент аналоги систем запобігання зіткнень і доступні технології їх реалізації.

2 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ

В цьому розділі описано процес проєктування розроблюваних компонентів.

Також наведено програмні засоби та особливості реалізації компонент системи.

2.1 Вимоги до системи

Основні функціональні та нефункціональні вимоги були виявлені на основі аналізу предметної галузі.

2.1.1 Вимоги до застосунку

В ході огляду існуючих рішень була виявлена необхідність у розробці програми з інтерфейсом для налаштування параметрів системи, що забезпечує можливість їх вільного редагування користувачем. Виходячи з цього, було виявлено такі функціональні вимоги.

– програма повинна забезпечувати під’єднання до віддаленого пристрою через мережу;

– програма повинна відображати відеопотік з віддаленого пристрою в режимі реального часу;

– програма повинна забезпечувати можливість налаштування таких параметрів як розмір області визначення об'єкта, вибір об'єктів, що визначаються;

– застосунок повинен забезпечувати обробку помилок, що виникають в результаті під’єднання, передачі даних та обробки відеопотоку.

Також було сформульовано такі нефункціональні вимоги:

– застосунок повинен володіти зручним та інтуїтивно зрозумілим інтерфейсом;

– програмна складова повинна бути реалізована за допомогою мови програмування Python.

2.1.2 Вимоги до компоненту обміну даними

Також у ході огляду було виявлено необхідність реалізації ефективного обміну даними між системними компонентами. З цією метою було прийнято рішення використати CAN- шину. Під час розробки програмного компонента зазначені такі вимоги:

– втілення миттєвого ефективного обміну даними між камерами та вузлом управління;

– забезпечення обробки даних із мінімальними затримками;

– повинна бути передбачена обробка помилок, пов'язаних із некоректними даними.

2.1.3 Вимоги до нейронної мережі

Аналогічно, в ході аналізу вимог була виявлена необхідність використання алгоритмів машинного зору для розпізнавання об'єктів, що знаходяться в полі зору камери.

Під час розробки компонента нейронної мережі зазначені такі вимоги до її функціонування:

– повинна забезпечувати визначення об'єктів із мінімальною затримкою;

– має забезпечувати ефективне визначення об'єктів;

– повинна забезпечувати детектування всіх об'єктів, що попадають у зону видимості;

– повинна зберігати показники ефективності у разі зміни погодних умов.

2.2 Архітектура системи

Архітектура всієї системи представлена на рисунку 2.1, розроблювані програмні компоненти позначені пунктиром. Сполучним компонентом є програма для налаштування параметрів. Користувач спочатку налаштовує необхідні йому параметри під’єднання та детектування. Після чого натискає кнопку «Під’єднатися», тим самим активуючи процес передачі даних. Далі, на вхід

системи надходить неперервний потік відео з IP камери. Кожен кадр цього потоку надходить у нейронну мережу для детектування та класифікації об'єктів. Після цього, зроблений кадр відображається на екрані користувача.

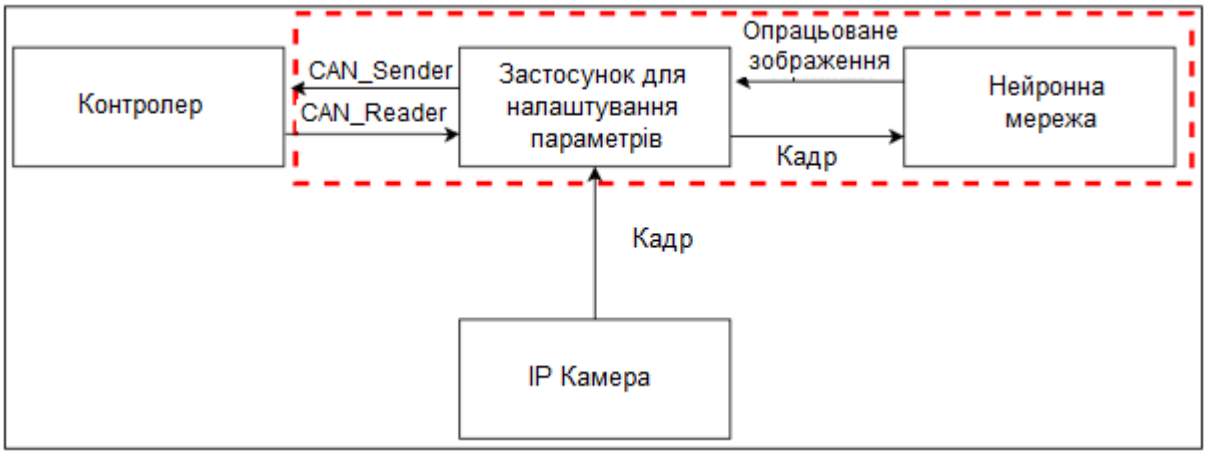

Рисунок 2.1 – Загальна архітектура системи

Паралельно з цим відбувається обмін даними з контролером за допомогою CAN- шини, ці дані містять інформацію про наявність об'єкта в області руху. Якщо в повідомленні, що надходить на контролер, зазначено наявність об'єктів у зоні руху, то відбувається зупинка автомобіля.

2.3 Проектування програми для налаштування параметрів

Згідно з описаними у п. 2.1.1 вимогами, була розроблена діаграма варіантів використання графічного інтерфейсу, представлена на рисунку 2.2 Актор «Користувач» має доступ до всього функціоналу, він може під’єднатися до системи, використовуючи спочатку задані параметри, або після їх зміни. Як зміна параметрів маються на увазі дії щодо зміни ширини і висоти зони, при попаданні об'єктів, що детектуються, в яку відбувається гальмування, а також щодо зміни об'єктів детектування. Також користувач може отримати довідку, в якій буде відображено зміст параметрів та необхідність їх зміни залежно від ситуації.

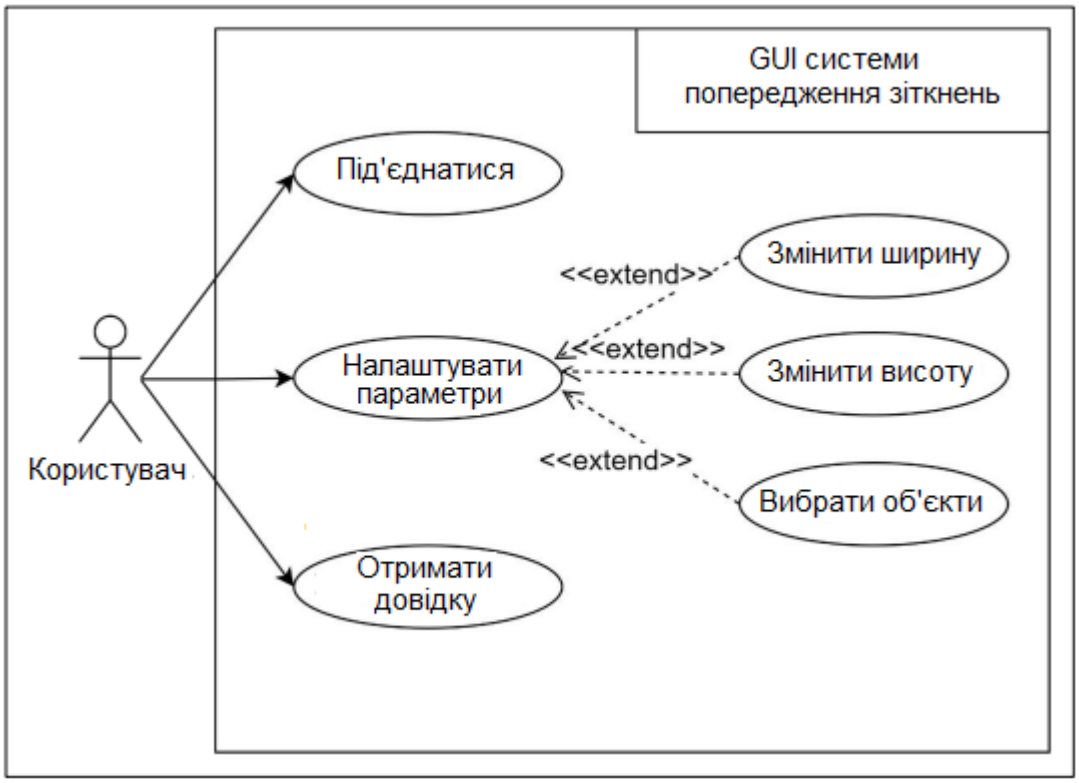

Рисунок 2.2 – Варіанти використання застосунку

На основі діаграми варіантів використання, а також функціональних вимог до компонента застосунку налаштування параметрів системи, була складена діаграма необхідних для ефективної роботи компонентів, вона представлена на рисунку 2.3.

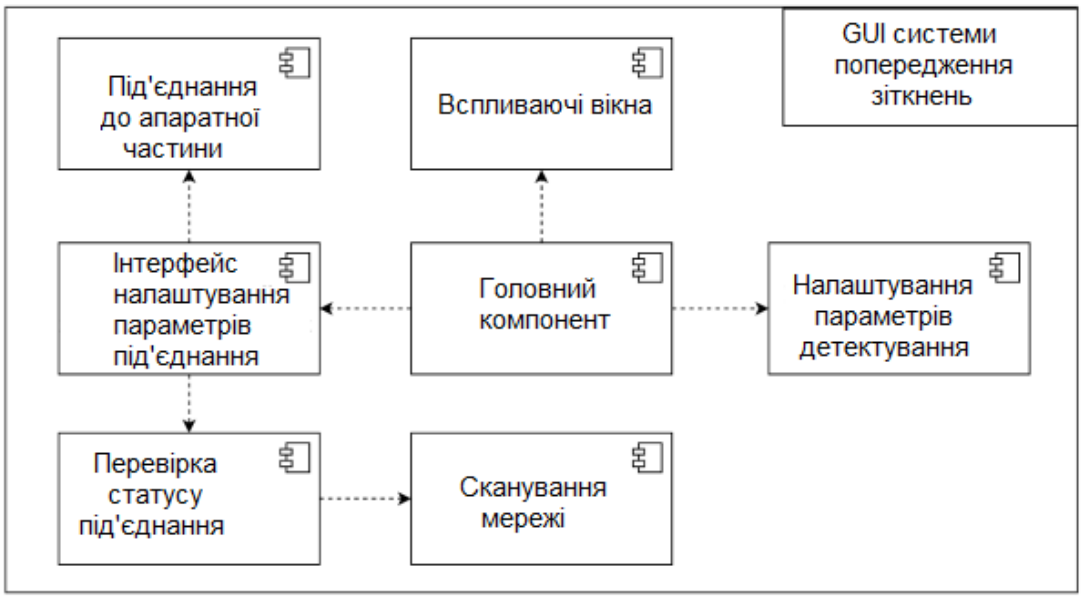

Рисунок 2.3 – Компоненти програми для налаштування параметрів системи

Застосунок для налаштування параметрів, який є інтерфейсом системи запобігання зіткнень, складається з наступних компонентів :

– «головний компонент» - основний модуль, в якому реалізована логіка програмного компонента;

– «інтерфейс налаштування параметрів під’єднання» - модуль, що повністю відповідає за налаштування параметрів під’єднання;

– «під’єднання до апаратної частини системи» - модуль, що відповідає за перевірку коректності введених параметрів під’єднання, у разі успішності перевірки, компонент зв'язується з апаратною частиною і починається відтворення відеопотоку;

– «перевірка статусу під’єднання» - модуль, який відповідає за відображення прапора під’єднання;

– "сканування мережі" - модуль, що відповідає за перевірку наявності пристроїв у мережі;

– «інтерфейс налаштування параметрів детектування» - модуль, що відповідає за зміну розміру області руху та об'єктів детектування;

– "випливаючі вікна" - модуль, що відповідає за обробку помилок.

На основі вимог до компонента було спроектовано макет інтерфейсу, який представлений на рисунку 2.4.

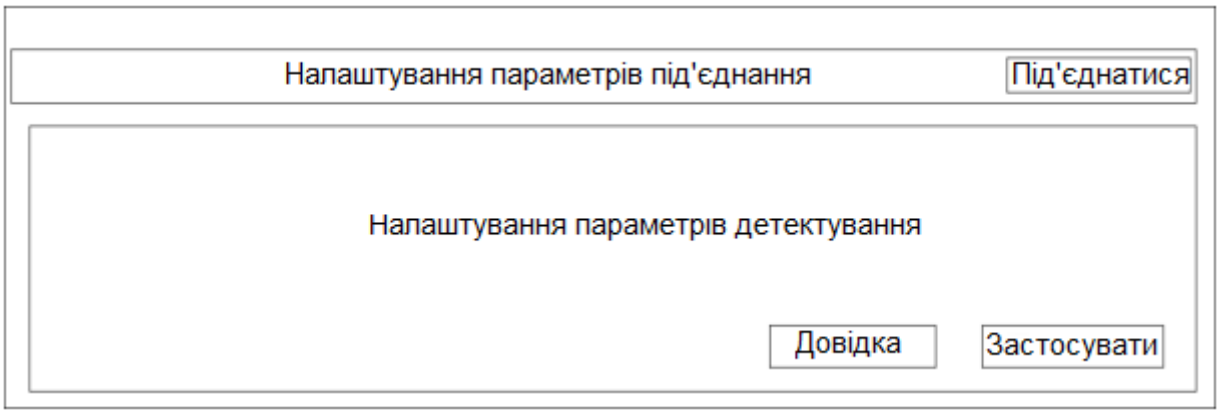

Рисунок 2.4 – Макет застосунку

У даному макеті передбачено вікно налаштування параметрів під’єднання з

кнопкою «Під’єднатися», після натискання на яку відкривається вікно з відеопотоком для відображення роботи програми.

Також у вікні «Налаштування параметрів детектування» буде реалізовано можливість редагування розміру області реагування, об'єктів детектування та необхідної точності. При натисканні на кнопку «Застосувати», відбуватиметься оновлення налаштувань, при натисканні на кнопку «Довідка» відкриватиметься вікно з інструкцією з коректного налаштування параметрів.

Також для відображення помилок при роботі програми передбачені спливаючі вікна з текстом помилки. Їхній інтерфейс представлений на рисунку 2.5.

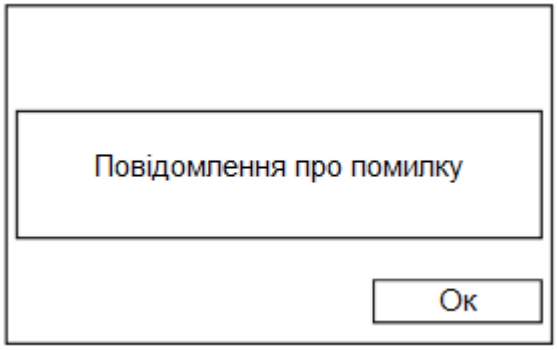

Рисунок 2.5 – Повідомлення про помилку

2.4 Проектування компоненту обміну даних

У процесі розробки програмних компонентів для системи запобігання зіткнень великогабаритної техніки було виявлено необхідність реалізації ефективного обміну даними між компонентами системи [8]. Отже, були розглянуті різні варіанти реалізації такого обміну.

Використання CAN- шини як основи компонента обміну даними дає можливість найбільш ефективної роботи системи. Так як у протоколі CAN реалізовано механізм виявлення помилок, який у разі необхідності передає у мережу зв'язаних пристроїв прапор помилки, який аварійно завершує передачу поточного повідомлення, після цього дане повідомлення скидається [9]. Таким чином у системі досягається не суперечливість даних.

Так як даний компонент забезпечує обмін інформацією між камерами та вузлом управління автотранспортом, що відіграє одну з ключових ролей узгодженої роботи системи, було прийнято рішення використовувати CAN шину з причин найбільш надійного та стійкого методу обміну даними [10]. Окрім того, дана технологія призначена для використання в автомобільних умовах, що означає можливість функціонування за екстремальних температурних умов і при впливі вібрацій [11].

2.5 Проектування компонента нейронної мережі

Згідно з описаними у п. 2.1.3 вимогами до наявності та реалізації нейронної мережі, необхідно підібрати архітектуру для найбільш ефективного та швидкого детектування всіх об'єктів, що потрапляють у поле зору камери. Для цього кожен кадр відеопотоку повинен проходити через мережу якнайменше разів.

Алгоритм YOLO є інструментом для детектування об'єктів у реальному часі завдяки своїй архітектурі , яка дозволяє здійснювати детектування та класифікацію об'єктів на зображенні за один прохід через мережу [12], тобто паралельно. Такий ефект досягається завдяки розбиттю зображення або кожного з кадрів відеопотоку на квадратну сітку. У кожному осередку сітки YOLO робить кілька передбачень наявності центру об'єкта та його меж. Крім того, вона оцінює відсоток достовірності цих передбачень.

Паралельно з цим нейромережа передбачає можливість приналежності об'єкта до кількох різних класів для кожної з осередків сітки, складаючи карту ймовірностей класів.

Далі передбачення рамок та класів поєднуються для отримання остаточного результату класифікації та детектування об'єктів. Для видалення надлишкових рамок використовується метод не максимального придушення, який залишає лише рамки з найбільшою ймовірністю достовірності [13]. Ілюстрація вищеописаної функціонування алгоритму продемонстрована на рисунку 2.6.

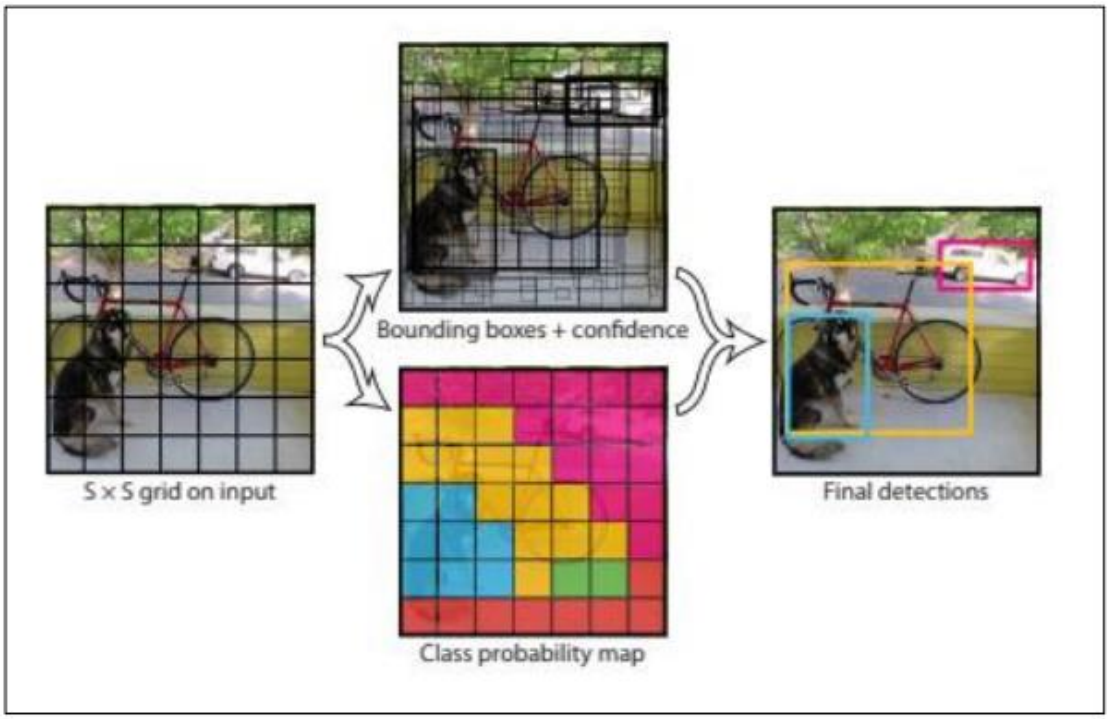

Рисунок 2.6 – Ідея функціонування моделі YOLO

Також варто відзначити, що YOLO є моделлю, яка попередньо навчена на вкрай великих обсягах даних, що дозволяє не навчати модель з нуля, а донавчити її для специфіки конкретного завдання за допомогою додавання користувача датасету з необхідними зображеннями. Таким чином використання YOLO є найбільш ефективним способом паралельного розв'язання задач детектування та класифікації в режимі реального часу.

2.6 Реалізація

Далі у підрозділі описано основні програмні засоби та процес програмного втілення компонентів системи.

2.6.1 Програмні засоби реалізації

Для реалізації програмних компонентів системи запобігання зіткнень великогабаритної техніки було обрано мову програмування Python 3.7 [14]. Для написання коду та налагодження використовувалося середовище розробки PyCharm 2023.3.3 (Community Edition) [15], навчання нейромережі проводилося в

середовищі розробки Google Collab [16], дані для навчання були взяті з сайту Roboflow [17].

У процесі розробки було застосовані наступні програмні засоби і бібліотеки:

– Customtkinter 5.2.2 [18] – бібліотека Python, яка надає кастомні віджети для Tkinter, роблячи інтерфейси більш сучасними та зручними;

– CTkMessagebox 2.5 [19] – розширення для бібліотеки Customtkinter, яке надає зручну реалізацію стилізованих повідомлень, відповідно було використано при виведенні спливаючих повідомлень помилки на екран користувача;

– Scapy 2.5.0 [20] – бібліотека Python для створення, відправлення, захоплення та аналізу мережевих пакетів;

– Clearml 1.16.1 [21] – платформа управління життєвим циклом машинного навчання, що включає відстеження експериментів;

– Ultralytics 8.2.28 [22] – бібліотека Python, розроблена авторами моделі YOLO, надає інструменти для побудови, навчання та розгортання моделей глибокого навчання.

2.6.2 Реалізація застосунку

Реалізація користувацького інтерфейсу. При реалізації використовувалася бібліотека «Custom TKinter». З її допомогою було реалізовано інтерфейс застосунку на основі розробленого раніше макета. Інтерфейс представлений на рисунку 2.7.

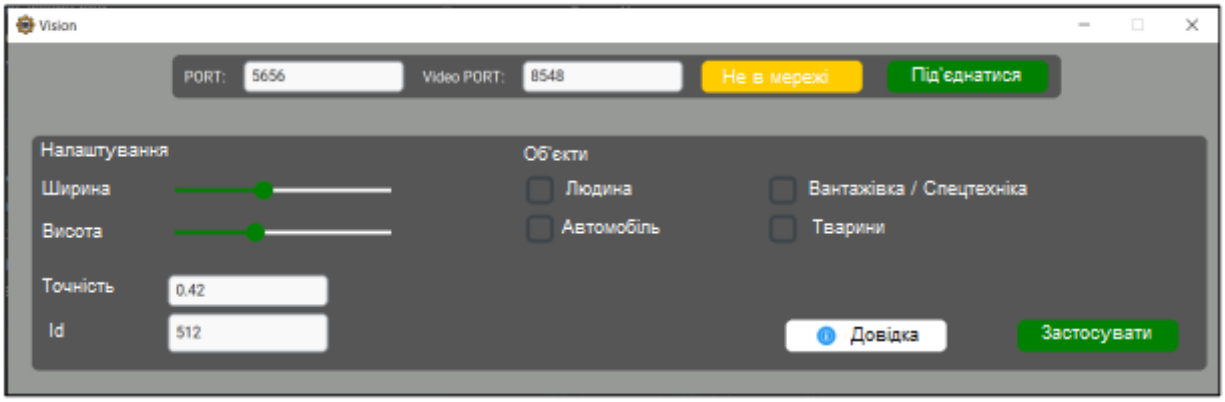

Рисунок 2.7 – Інтерфейс розробленого застосунку

Як параметри налаштування під’єднання були додані можливості налаштування порту для прийняття повідомлень і фактичний номер порту комп'ютера користувача, дані налаштування задані за замовчуванням, але підлягають зміні у разі потреби. Також у даному блоці реалізована кнопка для під’єднання та перевірка статусу під’єднання, яка змінює колір та статус «Не в мережі» на статус «В мережі» після натискання кнопки «Під’єднатися».

Як параметри для налаштування детектування передбачено вибір об'єктів детектування, що здійснюється за допомогою позначки користувачем необхідних позицій. Також передбачено налаштування ширини та висоти області, при попаданні в яку транспорт зупиняється. Це налаштування реалізовано за допомогою переміщення повзунка. При його переміщенні вліво параметр зменшується, а при переміщенні вправо збільшується. Також додано налаштування точності визначення об'єктів при збільшенні якої якість визначення об'єктів збільшується, та налаштування id повідомлення, що надходить у CAN шину.

Реалізація обробки помилок. Для коректної роботи програми було передбачено обробку помилок за допомогою спливаючих вікон, що містять повідомлення про помилку або попередження про неможливість коректної роботи системи. Відповідно повідомлення розділені на дві категорії: попередження та помилки. Інтерфейс вікна попередження наведений на рисунку 2.8.

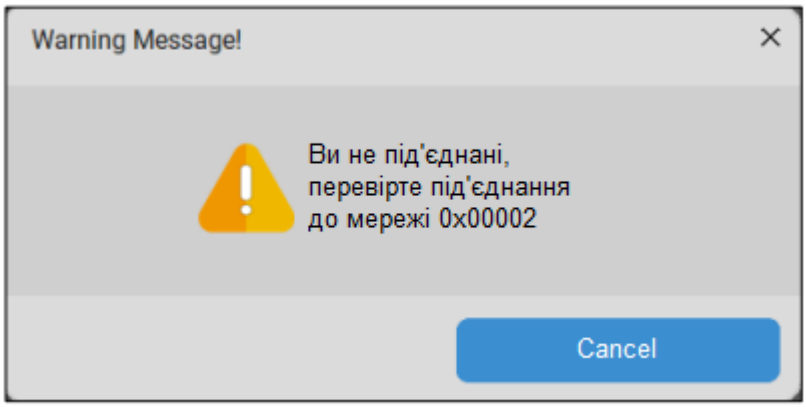

Рисунок 2.8 – Інтерфейс спливаючого вікна попередження

Наразі передбачено чотири ситуації для появи вікна попередження:

– "Ви не під’єднані, перевірте під’єднання до мережі" - попередження, що з'являється за відсутності доступу до мережі;

– "У мережі немає пристроїв" - попередження, що виникає за відсутності пристроїв у мережі;

– "Введений неприпустимий порт користувача" - попередження, що виникає при вказанні номера порту поза діапазоном від 0 до 65535.

– "Введений неприпустимий порт відео" - попередження, що виникає при вказанні номера порту, що не входить у діапазон від 0 до 65535.

Також були передбачені ситуації, за яких з'являється спливаюче вікно помилки

– "Повідомлення не надіслано" - помилка, яка виникає при спробі застосувати налаштування детектування без під’єднання до апаратного модуля системи;

– "При під’єднанні відбулася помилка, зачекайте і спробуйте знову" - помилка, що виникає якщо апаратний модуль не встиг розпочати свою роботу.

Інтерфейс вікна помилки представлений на рисунку 2.9.

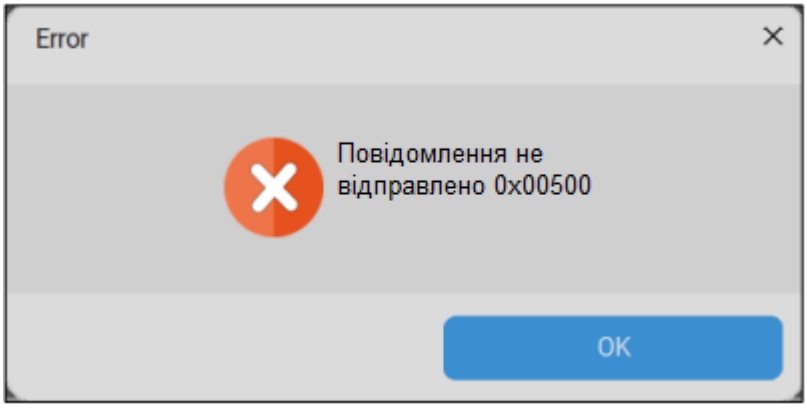

Рисунок 2.9 – Інтерфейс вікна помилки

Реалізація модуля під’єднання до апаратної частини системи. При реалізації модуля на початку відбувається перевірка стану прапора під’єднання, за умови під’єднання пристрою викликається метод kiii_jetson для відключення програмної частини системи. Інакше починається процес під’єднання, перш за все з інтерфейсу

користувача витягуються значення портів. Після чого відбувається перевірка коректності, при неправильності даних виводяться спливаючі вікна помилки.

Якщо дані коректні, то відбувається налаштування адреси для під’єднання. Потім оновлюється файл конфігурації, прапор під’єднання встановлюється у стан True.

Після чого викликається метод перевірки з'єднання self.call якщо з'єднання встановлено, створюється об'єкт Run_Videoplayer_iMG, надсилається повідомлення з даними для коректного запуску відеопотоку. І відбувається безпосередній запуск. Також оновлюється статус кнопки під’єднання і запускається моніторинг з'єднання за допомогою методу self.thread_cheker . Інакше викликається метод kill_jetson . Фрагмент коду модуля наведений у лістингу 2.1.

Лістинг 2.1 – Код модуля під’єднання до апаратної частини системи

```
def connect_to_jetson(self):
    Global flag_connect,base_ip,Target_Host,main_gateway
    if flag connect:
         self.kill jetson()
    else:
         try:
          if base ip != None:
           user ip = base ip
           user port = int(self.port our entry.get())
           user_video_port = int(self.port video entry.get())
           if user_port>65535 or user_port==user_video_port or
           user_port <0:
                  messageboxes.show_warning ("Введений неприпустимий
порт користувача 0x00200")
           elif user_video_port > 65535 or user_video_port <0:
                  messageboxes.show_warning ("Введений неприпустимий
порт відео 0x00201")
                else:
          target_ip = Target_Host target_port = 7474
          self.a = Connect Module()
          self.a.set_iPort_address(user ip, user port)
          self.a.set_adrress((target ip, target port))
          self.a.create_socket data()
           update_file({"ip": main gateway, "port data": user port,
          "port video": user video port})
     flag_connect = True
     self.connect button txt = 'Від’єднатися'
     self.back conncet = 'red'
     check_ 2 = self.call() if check 2:
           self.b = Run Videoplayer IMG()
```

Реалізація модуля сканування мережі. Як модуль сканування був реалізований метод для пошуку всіх пристроїв, що знаходяться в локальній мережі та встановлення з'єднання з доступними пристроями. Цей метод було реалізовано для спрощення використання програми оператором. Код методу scan наведений у лістингу 2.2.

Лістинг 2.2 - Код методу сканування мережі

```
def scan(self,target_ip):
    socket_scan = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    socket_scan.bind((HOST,PORT))
    arp = ARP(pdst=target_ip+"/24")
    ether = Ether(dst="ff:ff:ff:ff:ff:ff")
    packet = ether/arp
    result = srp(packet, timeout=3, verbose=0)[0]
    ip_pull = []
    for sent, received in result:
        ip_pull.append(received.psrc)
    for i in range(len(ip_pull)):
        if ip_pull[i]!=target_ip:
            give = 'give'
            socket_scan.sendto(give.encode(),(ip_pull[i],7474))
            try:
                wait_connect = select.select([socket_scan],[],[],5)
                data, addr = wait_connect[0][0].recvfrom(2048)
                if data != None:
                    ty ='ty'
                    socket_scan.sendto(ty.encode(),(ip_pull[i],7474))
                    self.ADDR_CONNNET = (ip_pull[i],7474)
                    self.TARGET_HOST = target_ip
                    break
```

Даний метод приймає параметр target_ip , який вказує IP -адресу цільового пристрою. Всередині методу scan створюється UDP сокет socket_scan, який зв'язується з локальним хостом та портом. Потім за допомогою бібліотек scapy створюється і відправляється ARP -запит для сканування мережі. ARP -запит створюється за допомогою класів ARP та Ether .

Отримані відповіді на ARP -запити обробляються, і IP -адреси доступних для під’єднання пристроїв зберігаються до списку ip_pull. Далі виводиться список всіх доступних пристроїв, після чого виконується перевірка кожної знайденої IP -адреси. Якщо IP- адреса не збігається з цільовим target_ip, сокет надсилає

повідомлення give на виявлений пристрій.

Сокет чекає відповіді пристрою упродовж п'яти секунд за допомогою функції select. При отриманні відповіді за цей час, виводяться дані пристрою, і сокет відправляє повідомлення ty. Потім атрибути ADDR_CONNNET та TARGET_HOST оновлюються, а цикл сканування переривається.

Реалізація модуля перевірки статусу під’єднання. Також було реалізовано модуль перевірки статусу під’єднання для зручності користувачів. Даний модуль використовує описану вище функцію сканування, за допомогою якої налаштовуються параметри та відбувається оновлення файлу конфігурації. Після цього метод оновлює колір та текст кнопки залежно від статусу під’єднання. Код оновлення представлений у лістингу 2.3.

Лістинг 2.3 – Код оновлення статусу під’єднання

```
if cheker_status:
    self.check_connect_button_txt = 'В Мережі'
    self.back_conncet_check = 'green'
else:
    self.check_connect_button_txt = 'Не в Мережі'
    self.back_conncet_check = yellow_back
    self.button_connect_status = CTkButton(
    master=self.up bar,
    text=self.check_connect_button_txt,
    fg_color=self.back_conncet_check,
    command=self.check_network).grid(
        row=0,
        column=10,
        padx=10,
        pady=5,
        sticky="w")
```

2.6.3 Реалізація компоненту обміну даними

Для забезпечення максимального відклику системи, що дає можливість найбільш якісного контролю ситуації в режимі реального часу та мінімізації затримок було прийнято рішення реалізувати кожний із класів в окремому потоці. На додачу усередині кожного з класів реалізована функція перевірки контрольної суми [23]. Метод crc8 застосовується з метою надання цілісності передавання

даних. Він дозволяє перевірити чи не відбулося спотворення даних у процесі передачі. Ця перевірка є стандартною при використанні CAN протоколу.

Клас Can_Reader призначений для безперервного моніторингу can порту на предмет даних. Функція start_write_msg в даному класі відповідає за обробку повідомлень, що надходять, і запускає процес аналізу цих повідомлень. У цьому методі реалізовано перевірку на наявність даних у повідомленні. Якщо повідомлення є порожнім, то викликається метод класу re, який відповідає за скидання всіх параметрів об'єкта класу для підготовки до прийому нового повідомлення. Якщо повідомлення, що надходить, не порожнє і на початку присутній знак «$», що означає початок повідомлення, прапор writer встановлюється в стан True , що вказує на початок прийому даних. Після початку прийому даних вони додаються до списку у форматі шістнадцяткових рядків. Далі відбувається аналіз першого елемента у списку даних для визначення максимальної довжини повідомлення. Якщо довжина списку даних досягає максимальної довжини, то проводиться обчислення crc та порівняння його зі значенням, вказаним у повідомленні. При збігу цих значень викликається метод check_can_list, що відповідає за аналіз типу повідомлення. Після цього викликається метод re для підготовки до прийому нового повідомлення.

Також в даному класі реалізований метод run, який відповідає за нескінченний цикл читання повідомлень, котрі надходять. Цей цикл може завершитися тільки при встановленні прапора stop_ALL у стан True .

У класі Can_Sender відбувається формування та надсилання повідомлень . За формування повідомлень відповідає метод pack_format , в який передаються дані для відправлення. У ньому спочатку ініціалізується порожній список, який міститиме порожнє повідомлення. Далі відбувається перевірка довжини даних, якщо вона перевищує 8 байт або дорівнює нулю, виводиться повідомлення про помилку. Після проходження перевірки відбувається збір даних, підрахунок контрольної суми, яка також додається до повідомлення. Далі на початок повідомлення додається вихідний байт, після чого повідомлення вважається сформованим.

Також у класі Can_Sender описаний метод run , який реалізує основний алгоритм відправлення даних по CAN -шині. У ньому реалізований безкінечний цикл, який перевіряє наявність об'єкта в зоні руху транспорту, за наявності такої інформації метод формує повідомлення з кодом 0xFD та відправляє його. Після чого прапор наявності об'єкта встановлюється в стан False. Якщо об'єкт не виявлено, то при формуванні повідомлення додається код 0xFC, після чого відбувається відправлення повідомлення. Цикл завершується лише за умови досягнення стану True прапором stop_ALL. Ця перевірка також включена у реалізацію циклу.

Реалізація методу run класу Can_sender представлена у лістингу 2.4.

Лістинг 2.4 - Реалізація методу run класу Can_sender

```
def run(self):
    global sending_special_info, id_msg_can, serial_port
    try:
        print('Start CAN')
        while True:
            if sending_special_info:
                msg_usb_to_can = self.pack_format(
                    id_msg_can,False,
                    [0xFD,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF])
                serial_port.write(msg_usb_to_can)
                sending_special_info = False
                time.sleep(0.5)
            else:
                msg_usb_to_can = self.pack_format(
                    id_msg_can,False,
                    [0xFC,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF])
                serial_port.write(msg_usb_to_can)
                time.sleep(0.5)
            if stop_ALL == True:
                print('video error')
                break
    except:
         print('drop')
```

Алгоритм формування та надсилання повідомлень продемонстрований на рисунку 2.10 У цьому алгоритмі відображено формування нового повідомлення, перевірка виявлення об'єкта, на основі якої в підсумкове повідомлення додаються

дані про наявність або відсутність об'єкта, що детектується, в зоні руху великогабаритної техніки, обчислення контрольної суми повідомлення, що є повідомлення та перевірка отримання прапора зупинки, при позитивному результаті якої алгоритм завершує роботу, а інакше зациклюється і починає формування нового повідомлення з такими ж кроками.

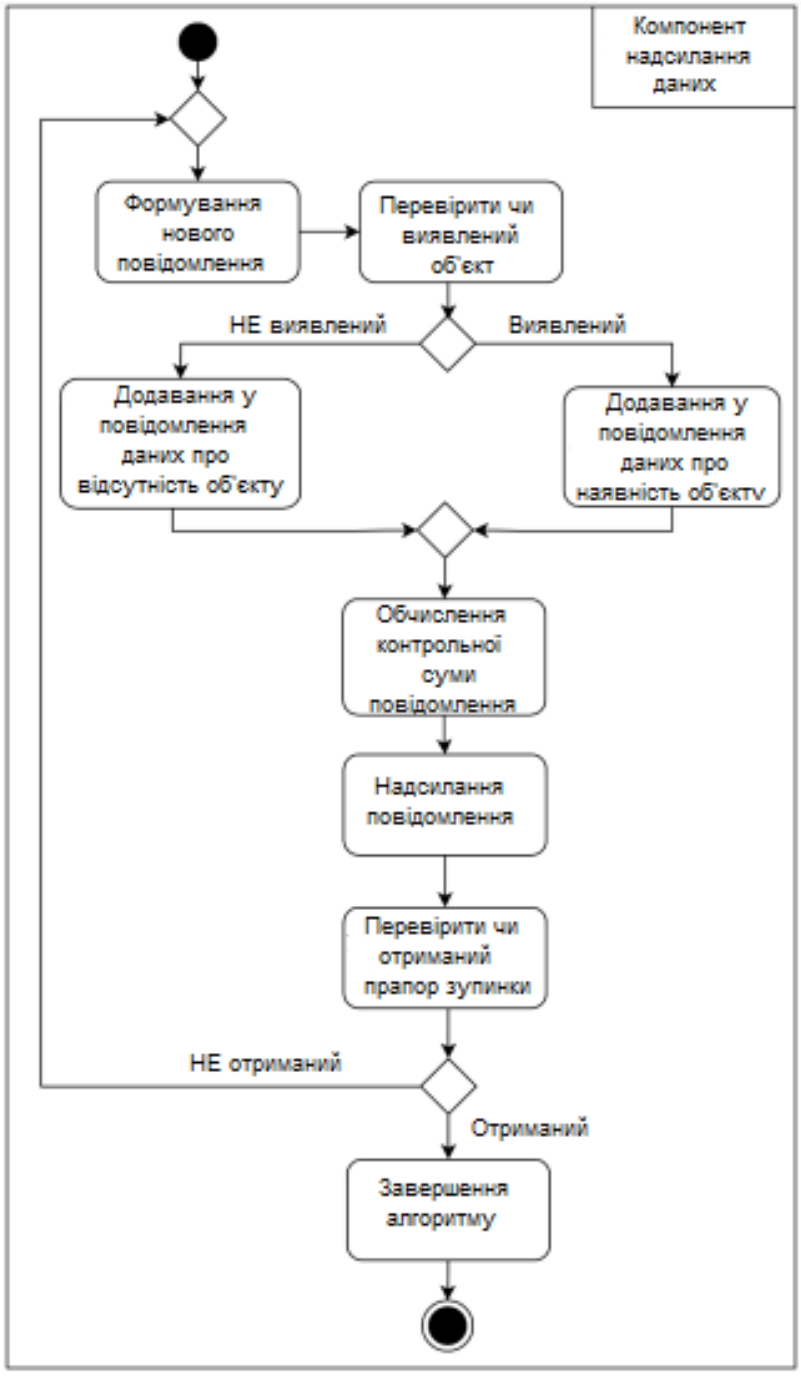

Рисунок 2.10 – Алгоритм відправки повідомлень

2.6.4 Реалізація нейронної мережі

На даний момент компанія Ultralistics випустила нову модель YOLO 8 [24], яка в порівнянні з іншими версіями забезпечує найкращі результати. Порівняння моделей представлене на рисунку 2.11.

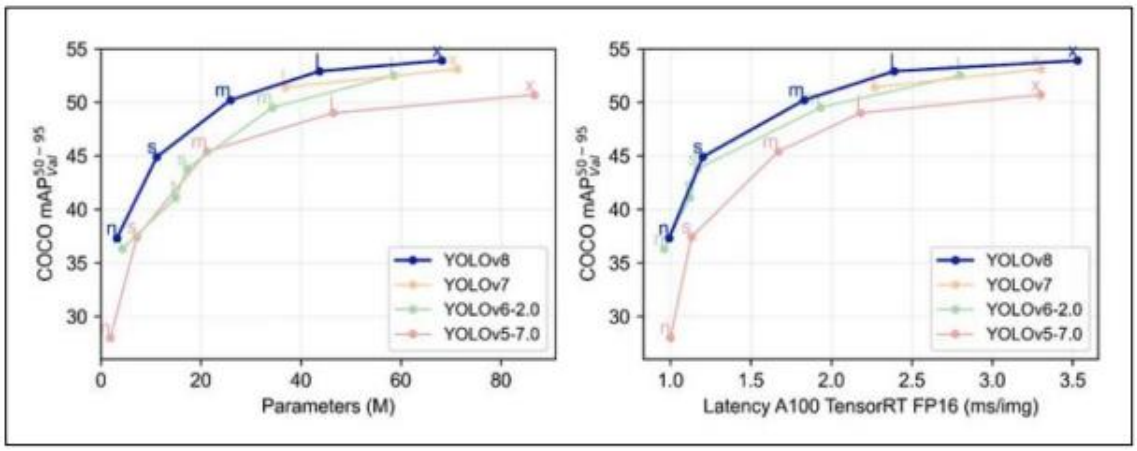

Рисунок 2.11 – Порівняння моделей YOLO

При порівнянні продуктивності різних моделей YOLO на однаковому наборі даних COCO порівнювалися різні показники. Як можна бачити за графіками, точність восьмої версії є найвищою.

Виходячи з цього, було прийнято рішення використати саме восьму версію для нейромережі. Датасет був узятий із сайту roboflow [25]. Оскільки було прийнято рішення донавчити нейромережу розпізнавати стоп-знаки, датасет складається з 3048 зображень стоп-знаку у різних ракурсах. Він спочатку розділений на тренувальну, тестову та валідаційну вибірки . Тренувальна вибірка містить 2552 зображення, тестова 213, а валідаційна 283. Також варто відзначити, що кожна з картинок доповнена рамкою, що позначає межі об'єкта - анотацією. Приклад зображень, що використовуються, представлений на рисунку 2.12.

Рисунок 2.12 – Приклад зображень датасету

Навчання відбувалося у середовищі Google Collab. Код навчання представлений у лістингу 2.5.

Лістинг 2.5 – Навчання нейронної мережі

```
model = YOLO('yolov8m.pt')
results = model.train(
    data='/content/data.yaml',
    imgsz=1280,
    epochs=50,
    batch=8,
    name='yolov8m_v8_50e')
```

За результатами навчання було побудовано графіки, котрі наведені на рисунку 2.13.

На рис. 2.13 можна помітити стрибки на тренувальній вибірці, але не на валідаційній. З часом втрати зменшуються, а метрики збільшуються, що дає змогу зрозуміти, що нейромережа дійсно навчається. Варто зазначити, що підсумкова точність згідно з метрикою precision – 0,993. Середня швидкість розпізнавання 12,751 ms.

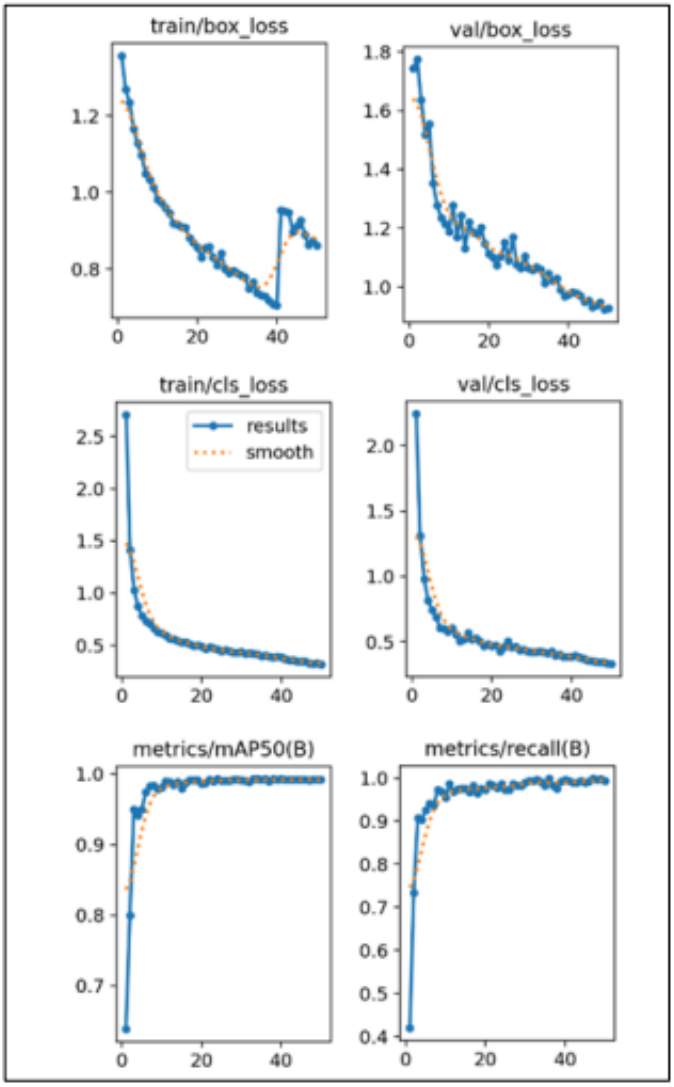

Рисунок 2.13 – Результати навчання нейронної мережі

2.7 Висновки до другого розділу

Таким чином були визначені вимоги, необхідні для розробки кожного з оголошених компонентів системи.

Для графічного користувальницького інтерфейсу були розроблені та описані діаграми варіантів використання та компонентів, а також спроектовані макети як самої програми налаштування параметрів системи, так і спливаючого повідомлення про помилку. При проектуванні враховувалися вимоги, визначені на початку етапу. При проектуванні компонента обміну даних були описані переваги та можливості при використанні протоколу CAN, серед яких встановлення найбільш стійкого і надійного з'єднання між компонентами системи, яке

досягається за рахунок вбудованого протоколу виявлення помилок, що аварійно закінчує передачу поточного повідомлення при їх виникненні. Варто відзначити, що завдяки цьому протоколу підвищується швидкість передачі даних. При проектуванні нейронної мережі було описано принцип роботи моделі YOLO, а також виявлено явну перевагу її використання у вигляді швидкого та ефективного розв'язання задач детектування та класифікації об'єктів.

В ході реалізації були розроблені необхідні компоненти системи з урахуванням виявлених раніше вимог.

Так, був реалізований інтуїтивно зрозумілий графічний інтерфейс користувача, що забезпечує під’єднання до апаратної частини системи по мережі. У цьому інтерфейсі було реалізовано блоки налаштування параметрів під’єднання та детектування. Також було реалізовано обробку помилок за допомогою спливаючих вікон, а також реалізовано методи для спрощення процесу використання програми. Компонент обміну інформацією, реалізований на основі протоколу CAN, забезпечує ефективний обмін даними, з мінімальними затримками та обробкою помилок, при якій виключається обробка пошкоджених повідомлень. Також була навчена нейронна мережа, яка забезпечує високу точність та швидкість детектування.

3 ТЕСТУВАННЯ

Для якісного проведення процесу тестування були розроблені та проведені функціональні тести [26] для кожного з програмних компонентів системи.

3.1 Тестування застосунку

Для тестування програми було проведено функціональне тестування основних функцій. Результати тестування представлені у таблиці 3.1.

Таблиця 3.1 - Функціональне тестування застосунку

№	Назва тесту	Кроки	Очікуваний результат	Тест пройдено?
1	**2**	**3**	**4**	**5**
1	Перевірка роботи кнопки «Під’єднатися»	1. Перевірити заповненість полів «Port» та «video port» 2. Натиснути на кнопку «Під’єднатися»	Відкривається вікно з відеопотоком .	Так
2	Перевірка коректності даних поля «Port»	1. Заповнити поле Port числом, що не входить у діапазон від 0 до 65535. 2 Натиснути на кнопку «Під’єднатися»	З'являється спливаюче вікно з помилкою «Введений недопустимий порт користувача»	Так
3	Перевірка коректності даних поля «Video port»	1. Заповнити поле Port числом, що не входить у діапазон від 0 до 65535. 2 Натиснути на кнопку «Під’єднатися»	З'являється спливаюче вікно з помилкою «Введений недопустимий порт відео »	Так
4	Перевірка коректної роботи статусу під’єднання «Не в мережі»	1. Перевірити статус без під’днання до апаратної частини	Плашка має жовтий колір та статус «Не в мережі»	Так
5	Перевірка коректної роботи плашки статус підключення «У мережі»	1. Під’єднатися до апаратної частини системи 2. Перевірити статус під’днання	Плашка має зелений колір і статус «У мережі»	Так

Продовження таблиці 3.1

1	2	3	4	5
6	Перевірка відображення відеопотоку	1. Під’єднатися ключитися до апаратної частини системи	Відкривається вікно з відеопотоком	Так
7	Перевірка збільшення ширини області зупинки при зрушенні повзунка налаштування вправо	1. Під’єднатися до апаратної частини системи 2. Збільшити ширину в налаштуваннях 3. Натиснути кнопку «застосувати»	Ширина області зупинки збільшується	Так
8	Перевірка зменшення ширини області зупинки при зсуві повзунка налаштування вліво	1. Під’єднатися до апаратної частини системи 2. Зменшити ширину в налаштуваннях 3. Натиснути кнопку «Застосувати»	Ширина області зупинки зменшується	Так
9	Перевірка детектування об'єкта «Людина»	1. Під’єднатися до апаратної частини системи 2. Відзначити в полі об'єкти «Людина» Натиснути кнопку «застосувати»	Об'єкт «Людина» розпізнається на відеопотоці	Так
10	Перевірка детектування об'єкту «Машина»	1. Під’єднатися до апаратної частини системи 2. Відзначити у полі об'єкти «Машина» Натиснути кнопку «застосувати»	Об'єкт «Машина» розпізнається на відеопотоці	Так
11	Перевірка детектування об'єкту «Вантажівка»	1. Під’єднатися до апаратної частини системи 2. Відзначити в полі об'єкти «Вантажівка» Натиснути кнопку «застосувати»	Об'єкт «Вантажівка» розпізнається на відеопотоці	Так
12	Перевірка детектування об'єкта «Тварини»	1. Під’єднатися до апаратної частини системи 2. Відзначити у полі об'єкти «Тварини» Натиснути кнопку «застосувати»	Об'єкт «Тварини» розпізнається на відеопотоці	Так
13	Обробка помилки «Повідомлення не надіслано»	3. 1. Натиснути кнопку «Застосувати» до натискання кнопки «Підключитися»	З'являється спливаюче вікно з помилкою «Повідомлення не надіслано»	Так

3.2 Тестування компонента обміну даними

Для проведення тестування було розроблено набір ручних тестів, що покриває реалізований функціонал. Тестування проводилося за допомогою імітації прийому та відправлення повідомлень. Протокол тестування представлений у таблиці 3.2.

Таблиця 3.2 – Функціональне тестування компонента обміну даними.

№	Назва тесту	Кроки	Очікуваний результат	Тест пройдено?
1	Обробка коректних даних	Надсилання коректних даних у форматі JSON	Дані записалися у відповідні змінні	Так
2	Надсилання повідомлення за наявності об'єкта перед машиною	Встановити прапор sending_special_info на стан True	Відправка коректного повідомлення по CAN шині	Так
3	Надсилання повідомлення за відсутності об'єкта перед машиною	Встановити прапор sending_special_info на стан False	Відправка коректного повідомлення по CAN шині	Так
4	Коректне завершення алгоритму	Встановити прапор stop ALL у стан True	Алгоритм завершує роботу	Так

3.3 Тестування нейронної мережі

Для тестування коректної роботи нейронної мережі були розроблені ручні тести, які покривають необхідний функціонал у вигляді коректного визначення об'єктів за різних навколишніх умов.

Тестування проводилося за допомогою завантаження відеопотоків у нейронну мережу. Протокол тестування представлений у таблиці 3.3.

Таблиця 3.3 - Функціональне тестування нейронної мережі

№	Назва тесту	Кроки	Очікуваний результат	Тест пройдено?
1	Перевірка коректності роботи за результатами, наближеними до ідеальних	Завантажити в нейронну мережу відео з умовами, наближеними до ідеальних	Нейронна мережа розпізнає стоп знак	Так
2	Перевірка коректності роботи в умовах сутінків	Завантажити в нейронну мережу відео, зняте в сутінки	Нейронна мережа розпізнає стоп знак	Так
3	Перевірка коректності роботи при знаходженні в тіні	Завантажити в нейронну мережу відео зі знаком, що перебуває в тіні	Нейронна мережа розпізнає стоп знак	Так
4	Перевірка коректності роботи в нічний час	Завантажити в нейронну мережу відео, зняте в нічний час	Нейронна мережа розпізнає стоп знак	Так

3.4 Висновки до третього розділу

Таким чином, в ході тестування були розроблені та втілені функціональні тести для кожного із компонентів. Для компонента застосунку були проведені 18 тестів, що покривають основний функціонал. Для компонента обміну даними проводилося 4 тести. Також, для нейронної мережі було проведено 4 тести. Усі розроблені тести пройшли успішно.

4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

В цьому розділі необхідно проаналізувати важливі питання, котрі стосуються безпеки життєдіяльності та основ охорони праці.

4.1 Класифікація шкідливих та небезпечних виробничих факторів

Шкідливий виробничий фактор – небажане явище, яке супроводжує виробничий процес і вплив якого на працюючого може призвести до погіршення самопочуття, зниження працездатності, захворювання, виробничо зумовленого чи професійного, і навіть смерті, як результату захворювання. Небезпечний виробничий фактор – небажане явище, яке супроводжує виробничий процес і дія якого за певних умов може призвести до травми або іншого раптового погіршення здоров’я працівника (гострого отруєння, гострого захворювання) і навіть до раптової смерті [32].

Поділ несприятливих чинників виробничого середовища на шкідливі та небезпечні зумовлене різним характером їх дії на людський організм, тим, що вони потребують різних заходів та засобів для боротьби з ними та профілактики викликаних ними ушкоджень, а також рядом причин організаційного характеру. В той же час між шкідливими та небезпечними виробничими факторами інколи важко провести чітку межу. Один і той же чинник може викликати травму і профзахворювання (наприклад, високий рівень іонізуючого або теплового випромінювання може викликати опік або навіть призвести до миттєвої смерті, а довготривала дія порівняно невисокого рівня цих же факторів – до хвороби; пилинка, що потрапила в око, спричиняє травму, а пил, що осідає в легенях, – захворювання, що зветься пневмоконіоз). Через це всі несприятливі виробничі чинники часто розглядаються як єдине поняття – небезпечний та шкідливий виробничий фактор (НШВФ) [33]. За своїм походженням та природою дії всі НШВФ можна поділити на 5 груп: фізичні, хімічні, біологічні, психофізіологічні та соціальні До фізичних НШВФ відносяться машини та механізми або їх елементи, а

також вироби, матеріали, заготовки тощо, які рухаються або обертаються; конструкції, які руйнуються; системи, устаткування або елементи обладнання, які знаходяться під підвищеним тиском; підвищена запиленість та загазованість повітря; підвищена або понижена температура повітря, поверхонь приміщення, обладнання, матеріалів; підвищені рівні шуму, вібрації, ультразвуку, інфразвуку; підвищений або понижений барометричний тиск та його різкі коливання; підвищена та понижена вологість; підвищена швидкість руху та підвищена іонізація повітря; підвищений рівень іонізуючих випромінювань; підвищене значення напруги в електричній мережі; підвищені рівні статичної електрики, електромагнітних випромінювань; підвищена напруженість електричного, магнітного полів; відсутність або нестача світла; недостатня освітленість робочої зони; підвищена яскравість світла; понижена контрастність; прямий та віддзеркалений блиск; підвищена пульсація світлового потоку; підвищені рівні ультрафіолетової та інфрачервоної радіації; гострі крайки, зачипки, шершавість на поверхні заготовок, інструментів та обладнання; розташування робочого місця на значній висоті відносно землі (підлоги); слизька підлога; невагомість.

Хімічні НШВФ:

- за характером дії на організм людини поділяються на токсичні, задушливі, наркотичні, подразнюючі, сенсибілізуючі, канцерогенні, мутагенні та такі, що впливають на репродуктивну функцію;

- за шляхами проникнення в організм людини поділяються на такі, що потрапляють через: 1) органи дихання; 2) шлунково-кишковий тракт; 3) шкіряні покриви та слизова оболонка;

- які перебувають у різному агрегатному стані: 1)твердому 2)газоподібному 3)рідкому.

Біологічні НШВФ – це: - патогенні мікроорганізми (бактерії, віруси, рикетсії, спірохети, грибки, найпростіші) та продукти їхньої життєдіяльності; - макроорганізми (тварини та рослини) та продукти їхньої життєдіяльності. До психофізіологічних НШВФ відносяться фізичні (статичні та динамічні) перевантаження і нервово-психічні перевантаження (розумове перенапруження,

перенапруження аналізаторів, монотонність праці, емоційні перевантаження). 6 Соціальні НШВФ – це неякісна організація роботи, понаднормова робота, змушеність праці в колективі з поганими відносинами між його членами, соціальна ізольованість з відривом від сім_ї, зміна біоритмів, незадоволеність роботою, фізична та/або словесна образа та її ризик, насильство та його ризик. Один і той же НШВФ за природою своєї дії може належати водночас до різних груп.

4.2 Вплив вібрації на людину

Вібрація - це механічні коливання пружних тіл або коливальні рухи механічних систем. Для людини вібрація є видом механічного впливу, який має негативні наслідки для організму [33].

Причиною появи вібрації є неврівноважені сили та ударні процеси в діючих механізмах. Створення високопродуктивних потужних машин і швидкісних транспортних засобів при одночасному зниженні їх матеріалоємності неминуче призводить до збільшення інтенсивності і розширення спектру вібраційних та віброакустичних полів. Цьому сприяє також широке використання в промисловості і будівництві високоефективних механізмів вібраційної та віброударної дії.

Дія вібрації може приводити до трансформування внутрішньої структури і поверхневих шарів матеріалів, зміни умов тертя і зносу на контактних поверхнях деталей машин, нагрівання конструкцій. Через вібрацію збільшуються динамічні навантаження в елементах конструкцій, стиках і сполученнях, знижується несуча здатність деталей, ініціюються тріщини, виникає руйнування обладнання. Усе це приводить до зниження строку служби устаткування, зростання імовірності аварійних ситуацій і зростання економічних витрат. Вважають, що 80% аварій в машинах і механізмах здійснюється внаслідок вібрації. Крім того, коливання конструкцій часто є джерелом небажаного шуму. Захист від вібрації є складною і багатоплановою в науково-технічному та важливою у соціальноекономічному відношеннях проблемою нашого суспільства [33].

Вплив вібрації на людину залежить від її спектрального складу, напрямку дії, прикладення, тривалості впливу, а також від індивідуальних особливостей людини. При оцінці вібраційного впливу потрібно враховувати, що коливальні процеси притаманні живому організму. В основі серцевої діяльності і кровообігу та біострумів мозку лежать ритмічні коливання. Внутрішні органи людини можна розглядати як коливальні системи з пружними зв'язками. Частоти їх власних коливань лежать у діапазоні 3..6 Гц. Частоти власних коливань плечового пояса, стегон і голови щодо опорної поверхні (положення стоячи) складають 4...6 Гц, голови щодо плечей (положення сидячи) 25...30 Гц.

При впливі на людину зовнішніх коливань (хитавиці, струсів, вібрації) відбувається їхня взаємодія з внутрішніми хвильовими процесами, виникнення резонансних явищ. Так, зовнішні коливання частотою менш 0,7 Гц утворюють хитавицю і порушують у людини нормальну діяльність вестибулярного апарата. Інфразвукові коливання (менш 16 Гц), впливаючи на людину, пригнічують центральну нервову систему, викликаючи почуття тривоги, страху. При певній інтенсивності на частоті 6..7 Гц інфразвукові коливання, втягуючи у резонанс внутрішні органи і систему кровообігу, здатні викликати травми, розриви артерій, тощо [33].

Вібрація, що діє на людину, має широкий діапазон – від десятих часток одного до декількох тисяч Гц. Характерними ознаками шкідливого впливу вібрації на людину є можливі зміни у функціональному стані: підвищена втома, збільшення часу моторної реакції, порушення вестибулярної реакції. Медичними дослідженнями встановлено, що вібрація є подразником периферичних нервових закінчень, розташованих на ділянках тіла людини, що сприймають зовнішні коливання. Адекватним фізичним критерієм оцінки її впливу на організм людини є коливальна енергія, що виникає на поверхні контакту, а також енергія, поглинена тканинами і передана опорно-руховому апарату та іншим органам. У результаті впливу вібрації виникають нервовосудинні розлади, ураження кістково-суглобної та інших систем організму. Відзначаються, наприклад, зміни функції щитовидної залози, сечостатевої системи, шлунково-кишкового тракту. Так, медичні

дослідження показали , що у працюючих в умовах вібрації відбуваються значні зміни кістковосуглобної системи, які виражаються у функціональній перебудові кісткової тканини, регіональному остеопорозі, кістовидних утвореннях у кістках, асептичному некрозі кісток, хронічних переломах. Відзначається, що терміни виникнення змін у кістках у працівників вібраційних професій коливається в межах від 6-8 місяців до 2-5 років.

Шкідливість вібрації збільшується при одночасному впливі на людину таких факторів, як знижена температура, підвищений шум, запиленість повітря, тривала статична напруга тощо. Сучасна медицина розглядає виробничу вібрацію як могутній стрес-фактор, що має негативний вплив на психомоторну працездатність, емоційну сферу і розумову діяльність, підвищує ймовірність виникнення різних захворювань і нещасних випадків. Особливо небезпечний тривалий вплив вібрації для жіночого організму. Широкий комплекс патологічних відхилень, викликаний впливом вібрації на організм людини, кваліфікується як віброзахворювання [33].

Вібрація як фізичний чинник виробничого середовища спостерігається в металообробній, гірничодобувній, металобудівній, машинобудівній, авіаційній та інших галузях народного господарства. Джерелом вібрації можуть бути різні механізми, вібраційне устаткування, віброінструменти, акустичні системи, транспортні та сільськогосподарські машини.

Загальна вібрація поділяється на транспортну вібрацію, яка діє на людину на робочих місцях в транспортних засобах (трактори сільськогосподарські та промислові, самохідні сільськогосподарські машини (комбайни), тягачі, грейдери ті інші); транспортно-технологічну вібрацію, яка діє на людину на робочих місцях машин з обмеженою рухливістю (екскаватори, крани промислові та будівельні, гірничі комбайни, транспорт виробничих приміщень та інші) та технологічну вібрацію, яка діє на людину на робочих місцях стаціонарних машин чи передається на робочі, де немає джерел вібрації (верстати та метало-деревообробне, пресувально-ковальське обладнання, ливарні машини, електричні машини, насосні агрегати та вентилятори, обладнання для буріння свердловин, бурові верстати, машини для тваринництва, очищення та сортування зерна (у тому числі сушарні),

обладнання промисловості будматеріалів (крім бетоноукладачів), установки хімічної та нафтохімічної промисловості та інші.

Оператори машин, які зазнають у процесі трудової діяльності впливу вібрації, підлягають попереднім та періодичним медичним оглядам відповідно до Порядку проведення медичних оглядів працівників певних категорій, затвердженого Наказом МОЗ України від 21.05.2007 р. №246. Обов'язкові попередні (під час прийняття на роботу) та періодичні (протягом трудової діяльності) медичні огляди дозволять визначити стан здоров'я працівника та можливість виконання без погіршення стану здоров'я професійних обов'язків, своєчасно виявити ранні ознаки хронічного професійного захворювання, забезпечує динамічне спостереження за станом здоров`я в умовах дії шкідливих та небезпечних факторів і трудового процесу, вирішує питання щодо можливості продовжувати роботу в умовах дії шкідливих та небезпечних факторів і трудового процесу [33].

За результатами періодичних медичних оглядів роботодавець забезпечує проведення відповідних оздоровчих заходів Заключного акта у повному обсязі та усуває причини, що призводять до професійних захворювань. Організовує проведення лабораторних досліджень умов праці на робочих місцях та вживає заходів до усунення небезпечних і шкідливих для здоров'я виробничих факторів.

4.3. Висновки до четвертого розділу

В цьому розділі проаналізовано важливі питання охорони праці та безпеки в надзвичайних ситуаціях, висвітлено питання класифікації шкідливих та небезпечних виробничих факторів та впливу вібрації на людину.

ВИСНОВКИ

При виконання роботи були створені програмні елементи частини системи запобігання зіткненню для великогабаритної техніки, при цьому були вирішені перелічені нижче завдання:

– проаналізовано предметну галузь та здійснено огляд аналогів;
– виконано проєктування програмних компонентів системи;
– програмно втілені ці компоненти;
– здійснено функціональне тестування компонентів.

Для подальшого поліпшення та доопрацювання системи варто впровадити нейромережу, здатну класифікувати та детектувати більшу кількість об'єктів, ніж та, яка використовується в даний момент. При цьому і класифікація, і детектування повинні проходити без втрати швидкості та ефективності.

Також як покращення можна використати додавання огляду в 360 градусів для оператора даного програмно-апаратного комплексу, що дозволить забезпечити більший контроль над ситуацією, що відбувається не тільки попереду, а й навколо великогабаритної техніки. На даний момент виявлено кілька можливих стратегій реалізації даного завдання, проводиться їх порівняльний аналіз.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Amato F., Nardone R., Santone A. CAN-Bus AttackDetection With Deep Learning. // Transactions on Intelligent Transportation Systems, IEEE 2021. 5081–5090 pp.

2. SICK Safety Collision Prevention. URL: https://www.sick.com/ag/en/catalog/archive/s3000-anti-collision/c/g329351 (дата звернення: 25.04.2026).

3. Caterpillar Cat Detect Safety System. URL: https://www.cat.com/en_US/support/operations/technology/product-link-owners-manuals/cat-object-detection-manuals.html (дата звернення: 25.04.2026).

4. Komatsu Intelligent Machine Control. URL: https://www.komatsu.eu/en/komatsu-intelligent-machine-control (дата звернення: 25.04.2026).

5. John Deere ActiveCollision Avoidance System. URL: https://www.deere.com/en/technology-products/precision-ag-technology/guidance/auto-trac/ (дата звернення: 25.04.2026).

6. Kumar P., Narasimha Swamy S., Purohit G., Raju K. S., Real-time, yolo-based intelligent surveillance and monitoring system using jetson tx2. // Data Analytics and Management Proceedings of ICDAM, Springer 2021. 461– 471 pp.

7. Jetson Software Documentation. URL: https://docs.nvidia.com/jetson/tation (дата звернення: 29.04.2026).

8. Donmez T. C. M. Anomaly Detection in Vehicular CAN Bus UsingMessage Identifier Sequences. // IEEE Access, 2021. 105–108 pp.

9. Marchetti M., Stabili D. Anomaly detection of can bus messagesthrough analysis of id sequences. // IEEE Intelligent VehiclesSymposium (IV), 2017. 1577–1583 pp.

10. Taylor A., Japkowicz N., Leblanc S. Frequency-based anomaly detection for the automotive can bus. // 2015 World Congress on IndustrialControl Systems Security (WCICSS), 2015. 45–49 pp.

11. Müter M., Asaj N. Entropy-based anomaly detection for in-vehiclenetworks. // 2011 IEEE Intelligent Vehicles Symposium (IV), 2011. 1110– 1115 pp.

12. Diwan, T., Anirudh, G., & Tembhurne, J. V. (2023). Object detection using YOLO: Challenges, architectural successors, datasets and applications. Multimedia Tools and Applications, 82(6), 9243–9275. https://doi.org/10.1007/s11042- 022-13644-y

13. Redmon J., Divvala S., Girshik R. You Only Look Once: Unified, Real-Time Object Detection.//IEEE Conference on Computer Vision and Pattern Recognition Conference, 2016. 45–49 pp.

14. Python 3.7. URL: https://www.python.org/downloads/release/python-373/ (дата звернення: 30.04.2026).

15. Pycharm. URL: https://www.jetbrains.com/pycharm/ (дата звернення: 30.04.2026).

16. Google Collab Documentation. URL: https://cloud.google.com/colab/docs (дата звернення: 30.04.2026).

17. Roboflow Documentation. URL: https://help.roboflow.com/ (дата звернення: 02.05.2026).

18. Customtkinter Documentation. URL: https://customtkinter.tomschimansky.com/ (дата звернення: 02.05.2026).

19. CTkMessagebox Documentation. URL: https://libraries.io/pypi/CTkMessagebox (дата звернення: 02.05.2026).

20. Scapy Documentation. URL: https://www.packetlevel.ch/html/scapy/docs/scapy.pdf (дата звернення: 03.05.2026).

21. Clearml Documentation. URL: https://clear.ml/docs/latest/docs/ (дата звернення: 03.05.2026).

22. Ultralytics Documentation. URL: https://docs.ultralytics.com/hub/models/ (дата звернення: 05.05.20262026).

23. Gao X., Huang D., Chen Y., Jin W., Luo Y. The design of a distributed control system based on CAN bus. // IEEE International Conference on Mechatronics and Automation, 2017. 1118-1122 pp.

24. YOLOv8 Documentation. URL: https://yolov8.org/yolov8-documentation/ (дата звернення: 19.05.2026).

25. Dataset Stop_sign. URL: https://universe.roboflow.com/wewilldoit/stopsign-hx5qv/dataset/2# (дата звернення: 20.05.2026).

26. Види функціонального та нефункціонального тестування. URL: https://dan-it.com.ua/uk/blog/vidy-funkcionalnogo-i-nefunkcionalnogo-testirovanija/ (дата звернення: 29.05.2026).

27. Методичні вказівки до виконання кваліфікаційної роботи бакалавра для здобувачів спеціальності 121 – Інженерія програмного забезпечення, всіх форм навчання / укладачі Михалик Д.М., Цуприк Г.Б., Бревус В.М. Тернопіль: Тернопільський національний технічний університет імені Івана Пулюя, 2024. 45 с.

28. Stefanyshyn, I., Pastukh, O., Stefanyshyn, V., Baran, I. , Boyko, I. Robustness of AI algorithms for neurocomputer interfaces based on software and hardware technologies CEUR Workshop Proceedings, 2024, 3742, pp. 137–149.

29. Boyko, I. , Petryk, M. , Mudryk, I. , Stoianov, Y., Tsupryk, H. Mathematical Model of the Capacitor Based on Zeolite Material. Proceedings - International Conference on Advanced Computer Information Technologies, ACIT, 2021, pp. 45–48.

30. Олянін, Д., Цуприк, Г. (2025) Transformer Neural Networks in Industry 4.0 / Д. Олянін, Г. Цуприк, Т. Говорущенко, О. Багрій-Заяць, І. Андрущак // Computer Information Technologies in Industry 4.0: proceedings of the 3rd International Workshop (CITI-2025), Ternopil, Ukraine, 11–12 June 2025. – Ternopil : Ternopil Ivan Puluj National Technical University, 2025 (Scopus) https://ceur-ws.org/Vol-4057/

31. Boyko, I. , Petryk, M. , Mykhailyshyn, R . Excitons in resonant tunnelling structures based on AlN/GaN/AlN/AlGaN/AlN nitride: spectral dependences and intensities of interband optical transitions. Ukrainian Journal of Physical Optics, 2022, 23(3), pp. 180–191

32. Заікіна Д., Глива В. Основи охорони праці та безпека життєдіяльності. 2019. URL: https://doi.org/10.31435/rsglobal/001 (дата звертання: 10.12.2025).

33. Зеркалов Д.В. Безпека життєдіяльності та основи охорони праці. Навчальний посібник. К.: «Основа». 2016. 267 с.