МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет

імені Петра Могили

Факультет комп'ютерних наук

Кафедра комп'ютерної інженерії

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри,
д-р техн. наук, проф.

_____ І. М. Журавська

« __ » _____ 2024 р.

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

Система збору інформації для водія з виведенням критичних параметрів

Спеціальність 123 Комп'ютерна інженерія

123 – КМР.ПЗ.00 – 605.21810521

Студент _____Д. В. Полівода
підпис
«__» _____ 202__ р.

Керівник
доц. каф. КІ, канд. техн. наук, доцент _____Я. М. Крайник
підпис
«__» _____ 202__ р.

Миколаїв – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук
Кафедра комп'ютерної інженерії

ЗАТВЕРДЖУЮ

Зав. кафедри _____ І. М. Журавська

«_____» _____ 2024 р.

ЗАВДАННЯ
на виконання кваліфікаційної магістерської роботи

Видано студенту групи 605м факультету комп'ютерних наук

Полівода Данило Володимирович
(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи
Система збору інформації для водія з виведенням критичних параметрів

Затверджена наказом по ЧНУ ім. Петра Могили №226 від 08.11.2023 №266.

2. Строк представлення кваліфікаційної роботи «____» _____ 20__ р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні
___ Очікуваним результатом роботи є: апаратно-програмний комплекс збору інформації для водія з виведенням критичних параметрів, який допоможе водію краще керувати транспортним засобом, а також допоможе уникненню дорожньо-транспортних пригод.

4. Перелік питань, що підлягають розробці
1) огляд сучасних підходів систем допомоги водію;
2) аналіз переваг та недоліків існуючих систем;
3) розробка апаратної частини системи збору критичної інформації;
4) розробка програмної частини системи збору критичної інформації;

5. Перелік графічних матеріалів
Слайди презентації, таблиці, рисунки

6. Завдання до спеціальної частини:

Проведення оцінки умов праці фахівців підприємства, а саме аналіз виробничого приміщення, оцінка природного освітлення. Визначення рівня електричної та пожежної безпеки підприємства._____

7. Консультанти:

Консультант	Кафедра (організація)	Частина роботи
Д-р біол. наук, професорка Григор'єва Л. І.	Кафедра екології Навчально-наукового медичного інституту ЧНУ ім. Петра Могили	Спеціальна частина з охорони праці та безпеки у надзвичайних ситуаціях
Крайник Я.М.	Кафедра комп'ютерної інженерії	Методична частина

Керівник роботи
Канд. техн. наук, доцент Крайник Ярослав Михайлович_____
(посада, прізвище, ім'я, по батькові)

_____
(підпис)

Завдання прийнято до виконання

_____Полівода Данило Володимирович_____
(прізвище, ім'я, по батькові студента)

_____
(підпис)

Дата видачі завдання «____» _____ 20_____ р.

КАЛЕНДАРНИЙ ПЛАН
виконання кваліфікаційної магістерської роботи

Тема: _Система збору інформації для водія з виведенням критичних параметрів

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Розробка та затвердження завдання на виконання КМР	01.09.2023	01.09.2023	Виконано
2.	Складання календарного плану КМР	15.09.2023	15.09.2023	Виконано
3.	Огляд літератури за темою роботи	19.09.2023	19.09.2023	Виконано
4.	Пошук та аналіз аналогічних рішень	06.10.2023	06.10.2023	Виконано
5.	Проєктування апаратної частини	08.10.2023	08.10.2023	Виконано
6.	Розробка алгоритмів відправки та збереження даних	09.10.2023	09.10.2023	Виконано
7.	Тестування отриманої системи	10.10.2023	10.10.2023	Виконано
8.	Оформлення звіту й презентації з КМР	25.10.2023	25.10.2023	Виконано
9.	Перший попередній захист КМР	26.10.2023	26.10.2023	Виконано
10.	Виправлення зауважень. Апробація результатів досліджень.	10.11.2023	10.11.2023	Виконано
11.	Рецензування	12.11.2023	12.11.2023	Виконано
12.	Другий попередній захист КМР	10.12.2023	10.12.2023	Виконано
13.	Підготовка до захисту	14.12.2023	14.12.2023	Виконано
14.	Захист	20.12.2023	20.12.2023	Виконано

Розробив здобувач _Полівода Данило Володимирович _____

(*прізвище, ім'я, по батькові)* *(підпис)*

«__» _____ 20__ р.

Керівник роботи доцент Крайник Ярослав Михайлович__ _____

(підпис)

«__» _____ 20__ р.

АНОТАЦІЯ

до кваліфікаційної магістерської роботи

«Система збору інформації для водія з виведенням критичних параметрів»

студент групи 605м Полівода Данило Володимирович

Керівник: канд. техн. наук, доцент Крайник Ярослав Михайлович

Актуальність теми кваліфікаційної роботи полягає у критичній потребі зменшення дорожньо-транспортних пригод та об’ємів трафіку на дорогах. Використання систем збору та обробки інформації для водіїв обіцяє значно підвищити якість та безпеку керування транспортними засобами.

В даний час кількість автомобілів збільшується з кожним роком. Керувати автомобілем у великому місті стає все складніше: машина 21 століття буквально просякнута розумними системами, які контролюють практично всі параметри, в тому числі і процес паркування в умовах обмеженого простору. Безумовно, така увага до якості мобільності, безперебійного руху та потенціалу контролю інтелектуальних технологій призведе до подальшого розвитку інформаційних технологій.

Актуальність теми кваліфікаційної роботи полягає у критичній потребі зменшення дорожньо-транспортних пригод та об’ємів трафіку на дорогах. Використання систем збору та обробки інформації для водіїв обіцяє значно підвищити якість та безпеку керування транспортними засобами.

Об’єкт дослідження (розробки): Методи та технології визначення критичних параметрів при керуванні транспортним засобом.

Предмет дослідження (розробки): Мобільний застосунок з можливістю комунікації з датчиками збору інформації для подальшого її виводу.

Мета роботи: Розробка системи збору інформації для водія з виведенням критичних параметрів на смартфон.

Робота складається з переліку скорочень, вступу, чотирьох розділів, висновку, переліку джерел посилання та трьох додатків.

У вступі наводяться ключові аргументи, що підкреслюють важливість обраної теми, визначаються об'єкт та предмет дослідження, ціль дослідження та завдання, які необхідно вирішити для досягнення мети роботи.

В першому розділі роботи здійснено огляд сучасних систем збору критичної інформації для водіїв, що забезпечує безпечне керування.

Другий розділ присвячений детальному опису алгоритмам та методам обробки критичної інформації.

У третьому розділі описано технології для налаштування комунікації програмного комплексу з датчиками, що автоматично передають критичну інформацію на смартфон.

У четвертому розділі описано архітектуру та процес розробки системи.

У висновку подано огляд результатів виконання кваліфікаційної роботи.

Додатки містять код програмного забезпечення та інформацію про апробацію роботи.

Спеціальна частина з охорони праці розглядає дотримання норм охорони праці на робочому місці.

Кваліфікаційна робота містить 64 сторінок (без додатків), 16 рисунків, 17 джерел посилання та додатки.

ABSTRACT

of the master's thesis

«Smart System for Breast Cancer Monitoring and Diagnosis»

Student: Serhii Ovchar

Supervisor: Cand. Sc. (Tech.), Assoc. Prof. Yaroslav Krainyk

The relevance of the topic of qualification work lies in the critical need to reduce traffic accidents and traffic volumes on the roads. The use of information collection and processing systems for drivers promises to significantly improve the quality and safety of driving vehicles.

Currently, the number of cars is increasing every year. Driving a car in a big city is becoming more and more difficult: a 21st century car is literally imbued with smart systems that control almost all parameters, including the parking process in conditions of limited space. Undoubtedly, such attention to the quality of mobility, smooth movement and control potential of intelligent technologies will lead to further development of information technologies.

The relevance of the topic of qualification work lies in the critical need to reduce traffic accidents and traffic volumes on the roads. The use of information collection and processing systems for drivers promises to significantly improve the quality and safety of driving vehicles.

Object of research (development) : Methods and technologies for determining critical parameters when driving a vehicle.

Subject of research (development) : Mobile application with the possibility of communication with sensors for collecting information for its further output.

The goal of the work: Development of a system for collecting information for the driver with the display of critical parameters on a smartphone.

The work consists of a list of abbreviations, an introduction, four chapters, a conclusion, a list of reference sources, and three appendices.

The introduction provides key arguments that emphasize the importance of the chosen topic, defines the object and subject of the research, the goal of the research and the tasks that must be solved to achieve the goal of the work.

In the first section of the work, an overview of modern systems for collecting critical information for drivers, which ensures safe driving, is carried out.

The second chapter is devoted to a detailed description of algorithms and methods of processing critical information.

The third section describes the technologies for setting up the software complex's communication with sensors that automatically transmit critical information to a smartphone.

The fourth chapter describes the system architecture and development process.

The conclusion provides an overview of the results of the qualification work.

Appendices contain software code and job approval information.

The special part on labor protection considers compliance with labor protection norms at the workplace.

The qualification work contains 64 pages (without appendices), 16 figures, 17 link sources and applications.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ADAS	–	Advanced driver-assistance systems
ABS	–	Antilock Braking System
AEB	–	Autonomous Emergency Braking systems
ENCAP	–	European New Car Assessment Programme
I2C	–	Inter-Integrated Circuit
BaaS	–	(англ. Backend as a Service) - модель надання послуг, що передбачає можливість легкої інтеграції хамрних сховищ та API
MRR	–	Middle Range Radar
UART	–	Universal Asynchronous Receiver/Transmitte
LLVM	–	(англ. Low Level Virtual Machine) - універсальна система аналізу, трансформації і оптимізації програм
ВООЗ	–	Всесвітня організація охорони здоров'я
МОН	–	Міністерство освіти і науки
ПЛІС	–	Програмована логічна інтегральна схема
ШІМ	–	Широтно-імпульсна модуляція

ВСТУП

В сучасному суспільстві більшість повсякденних завдань спрощуються або автоматизуються, і ця тенденція посилюється з кожним днем. Електронні пристрої та технології, що покращують життя, такі як смартфони та автомобілі, стали частиною повсякденного життя сучасних людей. Популярність цих систем зумовлена прагненням людей до комфорту та зручності.

В даний час кількість автомобілів збільшується з кожним роком. Керувати автомобілем у великому місті стає все складніше: машина 21 століття буквально просякнута розумними системами, які контролюють практично всі параметри. Безумовно, така увага до якості мобільності, безперебійного руху та потенціалу контролю інтелектуальних технологій призведе до подальшого розвитку інформаційних технологій. Довгострокове бачення автомобільної індустрії полягає в тому, щоб позбавити транспортний засіб від керма - іншими словами, досягти автономності водіння, коли автомобіль рухається без втручання людини.

Актуальність теми кваліфікаційної роботи полягає у критичній потребі зменшення дорожньо-транспортних пригод та об’ємів трафіку на дорогах. Використання систем збору та обробки критичної інформації для водіїв обіцяє значно підвищити якість та безпеку керування транспортними засобами.

Об’єкт дослідження (розробки): Методи та технології визначення критичних параметрів при керуванні транспортним засобом.

Предмет дослідження (розробки): Мобільний застосунок з можливістю комунікації з датчиками збору інформації для подальшого її виводу.

Мета роботи: Розробка системи збору критичної інформації для водія з виведенням критичних параметрів на смартфон.

Методи дослідження: аналіз і синтез, порівняння, комп’ютерне моделювання.

Практичне значення отриманих результатів полягає в результатах виконання цієї роботи, котру можна використовувати для покращення обставин при переміщенні по дорогах, особливо в умовах недостатнього освітлення, що забезпечує безпеку руху.

Робота пройшла **апробацію** під час XXVI Всеукраїнської науково-практичної конференції «Могилянські читання» (Миколаїв, 07–11 листопада 2023 р.).

Публікації. Основні положення та результати магістерської роботи опубліковані у збірнику матеріалів XXVI Всеукраїнської науково-практичної конференції «Могилянські читання–2023».

1 АНАЛІТИЧНИЙ ОГЛЯД СИСТЕМИ ТА ПОСТАНОВКА ЗАДАЧІ

За даними Всесвітньої організації охорони здоров'я, ситуація з дорожньо-транспортними пригодами погіршується. Щороку в дорожньо-транспортних пригодах гине близько 1,19 мільйона людей. Додатково від 20 до 50 мільйонів людей отримують несмертельні травми, які в багатьох випадках призводять до інвалідності.

Дорожньо-транспортний травматизм завдає значних економічних збитків окремим людям, їхнім родинам та країні в цілому. Ця шкода є наслідком вартості лікування та втрати продуктивності внаслідок смерті чи травми, а також часу, втраченого на роботу чи навчання для тих, хто має доглядати за травмованою особою. Збитки від дорожньо-транспортних пригод у більшості країн відбирають 3% ВВП країни.

Без сумніву, автомобілі є джерелом високого рівня небезпеки. Для підвищення безпеки водіння велика кількість автомобілів оснащується такими системами:

— Пасивний захист (ремені безпеки, безпечне скло, яке не утворює гострих осколків при розбитті, подушки безпеки і т.д.);

— Активний захист (система контролю швидкості, система контролю смуги руху, система екстреного гальмування, тощо).

Системи пасивного захисту спрацьовують одразу після аварії та призначені для зменшення тяжкості аварії, а системи активного захисту призначені для запобігання аварії та активуються під час руху автомобіля.

1.1 Огляд сучасних систем допомоги водію

Наявність систем допомоги водієві все частіше стає вирішальним фактором при покупці автомобіля. Згідно зі статистичними даними, при покупці нового автомобіля кожен п'ятий легковий автомобіль оснащується такою системою.

Такі виробники автомобілів як, Volvo, Mercedes, BMW, Tesla та інші, вже пропонують автомобілі з безліччю автономних функцій. Серед них повнофункціональний круїз-контроль з режимом stop&go та радаром, контроль смуги руху, допомога при паркуванні та інші. Фактично, якщо активувати всі системи допомоги водію у Mercedes-Benz EQE SUV AMG 53 4Matic, вони забезпечать кращий контроль ситуації, ніж будь-який водій. Однак, це не робить автомобіль на сто відсотків безпечним.

Усі ці системи мають різні найменування. Tesla називає свою систему "Autopilot", хоча це не зовсім вірно, і причиною цього є велика кількість аварій за участю цих машин, які вимикають свої системи допомоги водієві за декілька секунд до неминучої аварії. Через це, вже постраждали люди та багато автомобілів. Система Audi називається "Traffic Jam Assist", що надає допомогу при заторах. Volvo дала назву своєму розробленню "DrivePilot". Всі вони самостійно керують, принаймні на автостраді, сповільнюються, коли перед ними є щільний трафік, і прискорюються, коли ситуація покращується. Mercedes взагалі забирає у вас контроль під час критичних ситуацій, коли ви ризикуєте потрапити у ДТП, це робить система Evasive Steering Assist.

Насамперед система «автономного водіння» – це та система, яка дозволяє транспортному засобу безпечно їздити громадськими шляхами без участі людини. Такі терміни, як «автономне водіння», «без водія» та «самокерований автомобіль» описують ту саму технологію: комп'ютерний мозок і датчики, які можуть керувати транспортним засобом замість людини, принаймні, за деяких умов.

Просунуті системи "drive-assist" і "без водія" спираються на кілька типів датчиків, включаючи камери, радар і лідар (сенсор, який використовує невидимі лазерні промені для точного вимірювання відстаней), щоб допомогти комп'ютерному «мозку» зрозуміти, де знаходиться транспортний засіб щодо його оточення.

Кількість датчиків, кількість обчислювальної потужності, необхідне для «мозку», і загальна вартість системи, як правило, зростають у міру збільшення рівня автоматизації.

Система автопілота - це комплекс електронних пристроїв і програмного забезпечення, призначений для керування транспортним засобом, таким як літак, автомобіль, корабель або космічний апарат, без прямого втручання людини. Основна мета автопілота полягає в тому, щоб автоматично керувати рухом об'єкта, дотримуючись заданих параметрів шляхом використання сенсорів, навігаційних систем, алгоритмів обробки даних та системи зв'язку.

Рисунок 1.1 – Система автопілоту компанії Mercedes-Benz

Цілком безпілотні системи в процесі руху збирають величезну кількість даних і постійно стикаються з новими ситуаціями дорожнього руху. Система на кожному транспортному засобі зазвичай ділиться цими даними з віддаленим центром обробки, який, відповідно до викладених уроків, оновлює поведінку всіх інших транспортних засобів.

Наразі спільнотою автомобільних інженерів (SAE) визначено 5 типів машин-безпілотників. Вся автотехніка розсортована на групи відповідно до міжнародного стандарту, який був оновлений у 2015 році — саме тоді виробники розпочали активну діяльність із створення безпілотних машин.

Рівень 0: Без автоматизації

Стандарти SAE починаються з Рівня 0: відсутність автоматизації. Водій-людина несе 100% відповідальності за те, що SAE називає "динамічним завданням водіння", що означає роботу з фактичного водіння транспортного засобу на постійній основі.

Рівень 0 не так уже й важко зрозуміти, але навіть тут є деякі нюанси. Ймовірно, найважливішим моментом є те, що існує багато сучасних автомобілів із функціями допомоги водієві, які, як і раніше, відповідають Рівню 0.

Наприклад, антиблокувальна система не вважається автоматизацією, оскільки людині все ж доводиться натискати на педаль гальма. Навіть системи, які автоматизують миттєве завдання - наприклад, автоматичні системи екстреного гальмування, що поставляються на деякі нові автомобілі, не вважаються автоматизацією, тому що вони не автоматизують "динамічне завдання водіння" на постійній основі.

Як дуже старі автомобілі, так і багато сучасних автомобілів, навіть оснащених системами допомоги водієві, є транспортними засобами одного рівня - нульового.

Рівень 1: Деяка допомога для водія

SAE визначає систему Рівня 1 як систему керування або рулюванням, або прискоренням/гальмуванням на постійній основі, але тільки за обмежених конкретних обставин.

Крім дуже простих моделей початкового рівня, більшість автомобілів, зроблених в останні роки, мають систему круїз-контролю. Ви майже напевно використовували його, принцип простий: прискорюєтесь до бажаної швидкості, активуєте круїз-контроль, і система утримуватиме автомобіль на цій швидкості після того, як ви знімете ногу з педалі акселератора.

Це не вважається "автоматизацією" в рамках SAE, тому що динамічна частина завдання водіння не автоматизована: людина все ще має бути готовою

натиснути на гальмо і деактивувати систему, якщо попереду транспорт, що рухається повільніше.

Останніми роками автовиробники почали пропонувати більш просунуті системи круїз-контролю, так званий адаптивний круїз-контроль. Адаптивні системи круїз-контролю розумніші - вони використовують радар, щоб утримувати автомобіль на безпечній відстані до транспорту, що йде попереду. Якщо автомобіль попереду сповільнюється, система автоматично уповільнює і ваш автомобіль, щоб зберегти безпечну відстань.

Завдяки адаптивному управлінню круїз-контролем, одна частина динамічної задачі водіння - управління прискоренням і гальмуванням - автоматизована. Звісно, це автоматизація лише за певних обставин, а саме, коли ви вмикаєте систему під час руху шосе. Але цього вже достатньо, щоб присвоїти адаптивній системі круїз-контролю Рівень 1.

Рівень 2: Обмежена допомога з керуванням і гальмуванням

Рівень 2 - "часткова автоматизація". Це стосується систем асистентів водія, що забезпечують керування керуванням і прискоренням/гальмуванням, але знову ж таки, тільки за певних обставин. Якщо водій повинен регулярно втручатися - наприклад, коли автомобіль з'їжджає з шосе - тоді це, ймовірно, система Рівня 2. Важливо зазначити, що це не автопілот, навіть якщо іноді схоже на те, що машина їде сама.

Рівень 2 іноді може виглядати як автопілот, але це не так. Autopilot в Tesla і Super Cruise в General Motors, які можуть прискорюватися, гальмувати і кермувати за багатьох але не за всіх обставин, можуть, безумовно, дати відчуття як від автопілота. Але є причина, чому GM і зазвичай Tesla дуже обережні в тому, щоб не описувати їх як повністю автономні системи в їхніх нинішніх формах: під час використання обох, водій має бути готовим узяти керування на себе практично миттєво.

Це критична відмінність. Якщо системі навіть на короткий період потрібна участь людини, то система не повинна описуватися як автопілот.

Замість цього правильно називати її розширеною системою підтримки водія (ADAS, advanced driver-assist system).

Це дуже важливо, тому що очікування від цієї системи можуть вплинути на те, наскільки безпечним є її використання на практиці. Модель Tesla S, оснащена ранньою версією автопілота, брала участь в аварії зі смертельними наслідками в травні 2016 року, коли система автомобіля не змогла розпізнати трейлер, що перетинає дорогу, і водій, судячи з усього, не втрутився і не натиснув педаль гальма. Це була перша відома аварія зі смертельними наслідками, пов'язана з чимось близьким до системи автопілота. І вона наголосила, що для цих систем необхідно зробити більш ефективну роботу щодо забезпечення того, щоб водії були завжди готові взяти керування під контроль. В свою чергу ця ситуація привела до значних обговорень у галузі: як система могла забезпечити залученість водія без надмірного роздратування людини?

Роздратування може здатися дрібницею, коли ми говоримо про уникнення нещасних випадків зі смертельними наслідками. Але треба враховувати наступне: якщо система занадто дратує, її не використовуватимуть.

Поточна версія автопілота Tesla попереджає водіїв не прибирати руки з керма під час використання системи. Для забезпечення цього, система має спеціальні датчики, спрямовані на виявлення того, чи перебувають руки водія на кермі.

Якщо руки водія не перебувають на кермі, система дає візуальне нагадування через 30 секунд, за яким слідує звукове попередження через 45 секунд. Через хвилину без втручання з боку водія система автопілота вимикається і не може бути увімкнена, доки автомобіль не буде перезапущено.

Це непогана система, але огляди говорять про те, що її легко обдурити - водії можуть просто доторкнутися до рульового колеса, коли вмикається

індикатор, щоб скинути цикл попередження, і продовжувати їзду без рук на кермі.

Проте Super Cruise GM застосовує елегантне рішення. GM дозволяє керувати автомобілем без рук. Замість того, щоб спостерігати за рухами рульового колеса, використовується камера для відстеження положення голови водія. Якщо система виявляє, що погляд водія знаходиться не на дорозі, вона починає серію попереджень, щоб спробувати повернути увагу водія на дорогу.

Подібно до попереджень Tesla, Super Cruise проходить шлях від миготливої лампочки на ободі рульового колеса, звукового сигналу і вібрації сидіння до голосової команди - в цей момент система Super Cruise відключається. Але GM йде далі, ніж Тесла: якщо водій, як і раніше, не контролює ситуацію, система поступово повністю зупинить автомобіль, активує попереджувальні мигалки і викличе допомогу використовуючи систему OnStar від GM.

GM також побудували безліч додаткових заходів безпеки в Super Cruise, щоб він використовувався тільки в ситуаціях, в яких він може безпечно працювати. Якщо автомобіль не перебуває на шосе, маркування дорожніх смуг нечітко видно, або система думає, що водій не зосереджений на дорозі - він навіть не увімкнеться. Може звучати як рецепт постійного роздратування, але реалізовано все це елегантно. Більшість рецензентів погодилися з тим, що Super Cruise приємно використовувати.

Підіб'ємо підсумок, система Рівня 2 являє собою вдосконалену систему допомоги водіям, яка може допустити водіння в режимі без рук за певних обставин. Але водій повинен залишатися насторожі і бути готовим взяти на себе "динамічну задачу водіння" в найкоротші терміни, а системи намагаються забезпечити, щоб водій залишався в стані готовності протягом усього часу їх використання.

Рівень 3: Трохи ближче до автопілота

SAE визначає Рівень 3 як "умовний автопілот". Різниця між Рівнем 2 і Рівнем 3 - у ступені. На практиці це залежить від відповіді на запитання: наскільки людина на сидінні водія має бути готовою взяти керування машиною на себе?

Під час використання системи Рівня 2 водій має бути дуже уважним, готовим одразу ж узяти керування, якщо система зіткнеться з чимось, з чим вона не впорається. На Рівні 3 очікується, що система може керувати рухом, поки вона перебуває в межах своєї "робочої області розробки", що означає, що роль людини - бути в резерві.

Труднощі визначення і пояснення різниці між Рівнем 2 і Рівнем 3 проявляються проблемою Рівня 3 на практиці. Нам легко зрозуміти, що система Рівня 2 не є автопілотом, в той час як побачимо нижче Рівень 4 є автопілотом, і досить легко пояснити цю різницю користувачам. Але Рівень 3 просто існує між ними. Це автопілот, за винятком випадків, коли це не так.

Це виклик для інженерів, яким доручено розробку систем. У своєму виступі в 2016 році колишній глава з продуктів Ford Motor Company Радж Наїр пояснив, чому дорожня карта Ford для технологій автопілота повністю пропускає Рівень 3.

Виявлено, що не вдавалося безпечно відпрацьовувати деякі сценарії, без додавання таких технологій, як LiDAR і тривимірних карт високої чіткості. Як тільки ви доходите до цього моменту, ви насправді шукаєте рішення для Рівня 4. Таким чином, ми змінили наш напрямок, замість поліпшення drive-assist технологій на основі камери і радара, ми робимо стрибок до Рівня 4, і робимо все, що потрібно, щоб повністю позбутися водія, керма і педалей.

Простіше кажучи: після того, як інженери Ford додали всі технології, необхідні для того, щоб їхній прототип Рівня 3 відповідав їхнім стандартам безпеки, вони були, по суті, на Рівні 4. З огляду на це, вони подумали, а навіщо взагалі пропонувати Рівень 3? Багато інших автовиробників дійшли такого самого висновку про Рівень 3, але не всі. Є ще одна точка зору, а саме, що

Рівень 3 можна інтерпретувати як більш комфортну реалізацію концепцій систем Рівня 2, таких як Super Cruise.

Це погляд Audi. Audi запустить систему Рівня 3, яка називається "Traffic Jam Pilot", на своєму седані A8 2019 року. Опис системи відображає те, як вони бачать відмінності між Рівнем 2 і Рівнем 3:

При запуску Traffic Jam Pilot водіям більше не потрібно постійно стежити за автомобілем і дорогою. Вони повинні просто залишатися напоготові і брати керування на себе, коли система запропонує це зробити.

Але зверніть увагу: в той час як Audi запропонує цю систему в Німеччині, яка нещодавно ухвалила закон про легальність подібних систем, Audi не постачатиме її до США - принаймні, поки що - через побоювання щодо відповідальності та державного регулювання.

Неважко зрозуміти, чому. Навіть за власними заявами Audi, відмінність все ще не визначена. Ідея, схоже, полягає в тому, що система Audi дасть людині трохи більше часу на взяття керування, ніж система Рівня 2.

Рівень 4: Справжній автопілот, але з обмеженнями

SAE каже, що Рівень 4 - це "висока автоматизація водіння": системі взагалі не потрібен резерв у вигляді людини, якщо вона функціонує в межах свого "робочого простору розробки". Іншими словами, система Рівня 4 все ще має межі, але поки вона перебуває в цих межах, людське втручання не знадобиться - це справжній автопілот.

Більшість сучасних систем автопілота залежать від високодеталізованих тривимірних карт, які допомагають "мозку" транспортного засобу знати точно, де він перебуває, аж до кількох сантиметрів або менше. Ці системи зазвичай використовують кілька лідарів для фіксації оточення транспортного засобу від моменту до моменту. Потім зображення з лідарів порівнюють зі збереженою 3D-картою.

Деякі з розроблюваних систем використовують метод лідари-і-карти як основний метод визначення місцезнаходження транспортного засобу, в той

час як інші використовують його як запасний. Один із принципів повних систем автопілота, таких як системи автопілота в авіації, полягає в тому, що всі критичні підсистеми повинні дублюватися, на випадок, якщо щось не працює.

Система Рівня 4 є справжнім автопілотом, доки система працює у своїх межах. Неважливо, чи буде людина, що сидить на сидінні водія, відволікатися, спати або навіть бути відсутньою, система Рівня 4 безпечно доставить транспортний засіб до місця призначення, якщо вона працює всередині своїх передбачуваних меж.

Скоро з'являться автомобілі Рівня 4 - фактично, вони вже тут. Раніше цього року компанія Waymo, раніше відома як проєкт самокерованих автомобілів Google, почала розгортати мережу транспортних засобів Рівня 4 у пілотному ride-hailing сервісі сервіс неліцензованих таксі, таких як Uber і Lyft у Чандлері, штат Аризона. General Motors заявила, що її власна компанія з автопілотів, GM Cruise, планує розгорнути парк "тисяч" таксі Рівня 4 у щільному міському середовищі в США з 2019 року. Інші підуть протягом кількох років.

Waymo проводили місяці, "навчаючи" свої прототипи на дорогах і в умовах руху в районі Чандлера і навколо нього; GM Cruise робить те ж саме зі своїми автомобілями на базі Chevrolet Bolt у Сан-Франциско. Поки ці транспортні засоби обмежені областями, з якими їхні системи найбільш знайомі, всі вони були ретельно задокументовані.

Ці обмеження - те що робить автопілоти системами Рівня 4. Наприклад, незалежно від того, як використовуються карти, якщо транспортний засіб залежить від карти, це означає, що він обмежений, оскільки він не може їздити місцями, які ще не додані на карту. Звичайно, система Рівня 4 може мати інші обмеження, вона може не вмикатися, якщо, наприклад, виявляє сильний сніг.

Рівень 5: Повне безумовне автоматичне водіння в повному обсязіРівень 5 - це мрія: безумовне, тобто без обмежень автоматичне водіння, без очікування того, що водій коли-небудь буде змушений втручатися.

Іншими словами, система Рівня 5 має їздити будь-якими місцями, де може проїхати кваліфікований водій, за будь-яких умов, з якими може впоратися кваліфікований водій, повністю самостійно.

Зайве говорити, що зараз немає доступних систем рівня 5. Кілька автовиробників, зокрема Tesla і BMW, заявили, що вони матимуть системи Рівня 5 протягом декількох років, але багато експертів вважають, що справжня автономія п'ятого рівня забере роки, якщо це взагалі станеться.

Можливо, що системи Рівня 5 будуть кінцевим результатом після того, як безліч під'єднаних транспортних засобів Рівня 4 їздитимуть по дорогах і пройдуть навчання. Системи, які розгортаються в тисячах автомобілів, що працюють щодня, збиратимуть величезні обсяги даних і стикатимуться з безліччю нових ситуацій - вони швидко розвиватимуться самостійно. Між цим триваючим навчанням і розширенням областей, що добре відображаються, найпоширеніші системи фактично стануть Рівнем 5 - в кінцевому рахунку.

Також можливо, що справжня система Рівня 5 - єдина система, яка зможе безпечно їздити в завірюху Монтани, заторах у Шанхаї та інших ситуаціях, де може їздити досвідчений водій, не з'явиться ще багато років, незважаючи на сміливі обіцянки, зроблені деякими керівниками.

Справжній автопілот - це неймовірно складна проблема. Врахуйте, що навіть Waymo, яка розпочалася ще 2009 року як проект Google Self-Driving Car і має деяких із найкращих і найдосвідченіших інженерів у цій галузі, все ще зазнає труднощів із масовим впровадженням своїх автомобілів для безпечного руху повсякденними дорожніми ситуаціями в приміському Чандлері, де всі сценарії відомі, і галузь ретельно вивчена.

Здібна людина-водій, ймовірно, зможе добре їздити по Мумбаї за допомогою смартфона і невеликого знайомства з містом. Але, принаймні

зараз, у 2024 році, схоже, що мине ще багато років, перш ніж система автопілота зможе зробити те ж саме - і це означає, що справжнього автопілота найближчим часом не з'явиться.

1.2 Аналіз проблем систем збору критичної інформації

Було проведено дослідження систем, що збирають інформації для водіїв. Основними обмеженнями допоміжних систем є недосконале розпізнавання об'єктів, високий рівень аварій та смертельних випадків серед пішоходів. Вчені та інженери з Euro NCAP, прогнозують, що до 2030 року лише 15% нових автомобілів будуть обладнані системою автономного керування. Однак, якщо з'являться проблеми, то до 2040 року лише 5% автомобілів можна буде вважати автопілотованими.

Серед проблем, які частіше виникають у роботі автопілота, можна виділити наступні аспекти: проблему розпізнавання об'єктів за допомогою датчиків, відсутність їх універсальності та проблеми на ринку.

Проблемою, з якою стикаються насамперед, є датчики. Основна причина обмеженого розвитку автопілотів полягає у недостатньому розумінні автомобілем навколишнього простору. Інформація, яку надають сучасні датчики, є недостатньою, а також механізм прийняття рішень, за допомогою нейромережі, не має достатньої навченості. На більшості прототипів, таких як, Google Car, встановлені дорогі скануючі лідари, які не забезпечують необхідної повноти інформації для безпечного руху.

У будь-якому симуляторі, комп'ютер майже завжди виявляється кращим за людину. Це можна проілюструвати на прикладі перемоги штучного інтелекту над пілотом у симуляторі повітряного бою, коли він має повну інформацію про ситуацію. Однак, в реальному світі штучний інтелект не матиме такої переваги над людиною на дорозі.

За допомогою наявних датчиків, таких як камери, радари, лідари і т.д., неможливо повністю вирішити проблему самокерування.

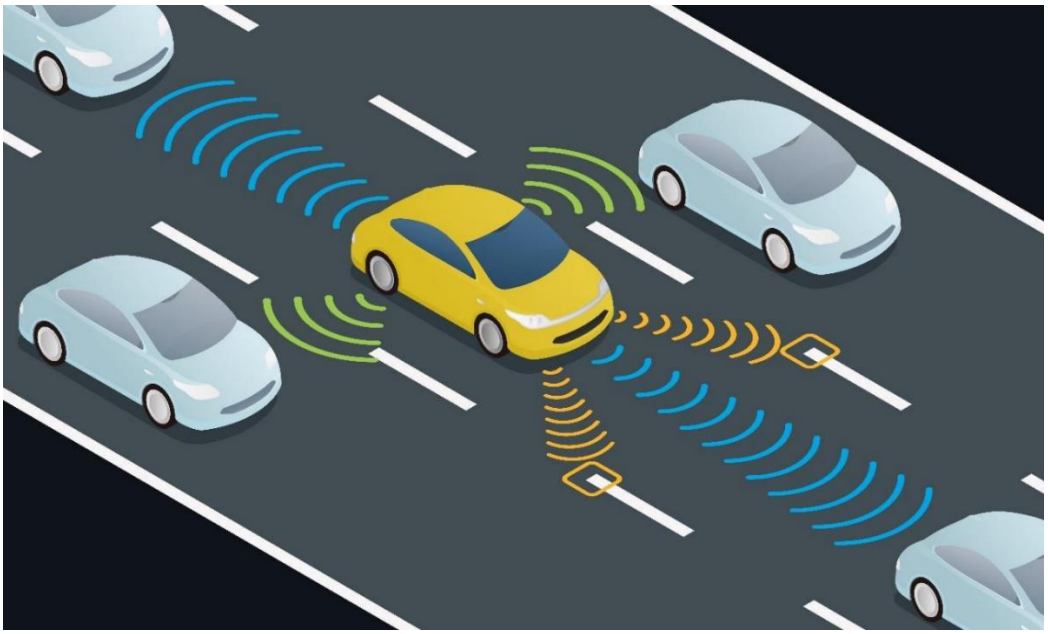


Рисунок 1.2 – Умовна робота датчиків

Інша проблема, яка виникає, полягає у відсутності універсальності. Прикладом даної проблеми є принцип Парето, який використовується в автопілотах. 80% дорожніх ситуацій машини розуміють, а 20%, що залишилися, призводять до аварійних ситуацій. З найпростіших завдань автопілот функціонує, але не без помилок. Серед таких найпростіших завдань є автобанний автопілот, автопарковка, заїзд у гараж. Розробникам залишилося реалізувати найважче: проїзд перехрестів, зміна смуги у жвавому потоці та ще кілька десятків схожих сценаріїв. З деякими завданнями взагалі не зрозуміло, що робити. Наприклад, автомобілю складно заздалегідь зрозуміти, чи проїде він між двома перешкодами, що близько розташовані. І поки що неможливо зрозуміти свій динамічний коридор (тобто межі простору, які автомобіль займає в динаміці), а без цього неможливо точно прорахувати дорожню ситуацію. Існуючі автопілоти настільки не універсальні, що вимагаю описувати окремо кожен існуючий сценарій керування автомобілем.

Проблема третя полягає у ринкових аспектах. Допоміжні системи мають бути не тільки функціональним, але й доступним з точки зору вартості. Навігаційні системи в автомобілях існують вже більше двадцяти років, але за ціною, яка становить кілька тисяч доларів, вони встановлюються лише на

менше ніж 20% нових автомобілів. Дослідження Euro NCAP показали, що лише 17% американських покупців готові витратити понад 5 тисяч доларів на автопілот. Щоб завоювати хоча б 15% світового ринку, система автопілотування повинна стати значно розумнішою і вдвічі дешевшою, ніж зараз. Проте, це не виглядає близькою перспективою.

Проте, у найближчі роки помічники з елементами автопілотування (наприклад, системи автопаркування, авторух у пробці, рух трасою з розміткою або навіть конвойні режими) будуть виготовлятись все більше. Автовиробники намагаються зробити їх доступнішими та масштабують ці функції, але наразі їх ще недостатньо. А у складних ситуаціях (ті самі 20 відсотків) управління на довгі роки залишиться на людині. Це тому, машина не може адекватно оцінити те, що відбувається, і найближчим часом не навчиться.

Завдання розпізнавання дорожніх знаків активно досліджується вже довгий час. Якщо буде створено метод, який працює з достатньою точністю, його можна використовувати в системах допомоги водієві, інвентаризації об'єктів придорожньої інфраструктури або для автоматизованого створення навігаційних карт. Дорожні знаки зроблені, щоб бути помітними і легко розпізнаватися і це робить їх хороших об'єктом для автоматичного розпізнавання. У той же час автоматичні алгоритми повинні розпізнавати знаки в умовах внутрішньокласовій мінливості, зміни освітленості, положення точки спостереження, розмиття і перекриттів.

Кращі на сьогоднішній день алгоритми спираються на машинне навчання і вимагають наявності великої і репрезентативною навчальної колекції, щоб обійти всі перераховані вище проблеми. Розробка призначена для збільшення безпеки на дорогах, а також полегшення процесу водіння. Інженери створюють рішення, які будуть автоматично розпізнавати дорожні знаки, фіксувати інформацію про допустимі швидкостях і обмеженнях, включаючи напрямок руху, наявність перехресть, поїзних перегонів та інших

даних. Чим більше попереджень отримує система від зовнішнього середовища, тим надійніше стає автомобіль і процес водіння.

Використання мобільного телефону як допоміжної системи є логічним рішенням, оскільки в сучасному світі майже у всіх людей є мобільні телефони, які завжди з ними. Тому відкриття програми для підвищення безпеки під час керування транспортним засобом не є складним завданням.

Підвищення безпеки, яке забезпечується системами допомоги водієві, є однією з причин їх популярності. Системи збору інформації для водіїв є необхідним кроком на шляху до глобальної мети зниження рівня смертності на дорогах до нуля. Ця мета також відображається в рейтингу безпеки нових автомобілів Euro NCAP, що сприяє розвитку допоміжних систем керування автомобілем. Все це впливає на обсяги виробництва датчиків, які необхідні для роботи подібних систем.

Основними проблемами створення допоміжних систем керування автомобілем є:

— Великі затрати на розробки;

— Високі вимоги до якості роботи допоміжних систем керування автомобілем;

— Конфіденційність компаній при висновках аналізу цих систем;

— Неточність та не універсальність подібних систем.

Для проведення досліджень в реальному середовищі необхідне дороге обладнання. Система повинна мати доступ до управління швидкістю автомобіля, поворотами та іншими функціями, а також мати додаткову систему охолодження для контроля обладнання. Також потрібен блок генератора для живлення системи та, за необхідності, обладнання для передачі управління від системи допомоги керування автомобілем до водія.

1.3 Формування вимог до системи

Для безпечної та ефективної роботи системи , необхідно встановити конкретні вимоги до системи збору критичної інформації для водіїв.

Невиконання цих вимог може призвести до порушення безпеки та досягнення низької продуктивності даного програмного забезпечення.

Оскільки ця кваліфікаційна робота призначена для надання допомоги водіям під час дорожнього руху, необхідно ретельно продумати систему. Вона повинна мати зручний інтерфейс та високу швидкодію.

Високоякісне програмне забезпечення має наступні ключові властивості:

− Ефективність. Програмний продукт повинен мати здатність забезпечувати необхідну працездатність при заданих умовах щодо виділених для цього ресурсів.

− Практичність. Здатність програмного продукту бути комфортним у навчанні та використанні, та бути привабливим для користувачів.

− Функціональність. Вимога до програмного продукту ставить акцент на його здатність вирішувати задачі, які є важливими та необхідними для користувачів.

− Надійність. Ця вимога до програмного продукту вказує на його здатність до безперервної роботи та коректного виконання задач протягом тривалого періоду використання.

− Доступність. Вибіркове використання окремих компонентів програмного продукту.

За статистикою, станом на 2023 рік 59% людей у світі (близько 4,7 мільярда) користуються смартфонами. У свою чергу, сучасні мобільні телефони мають у своїй конструкції вбудовані різноманітні датчики, зокрема акселерометри, гіроскопи, магнітометри тощо, за показниками яких можна створювати додатки на основі даних, котрі збираються з датчиків, що в свою чергу спрощує життя людей. Таким чином, дані можна використовувати на смартфоні під час подорожі та обробляти за допомогою мобільних датчиків, тим самим усуваючи ризики та покращуючи безпеку на дорозі.

Провівши огляд сфери допомоги водіям і систем збору інформації, проаналізована інформація наштовхує на створення системи на базі смартфона з використанням сенсора лідар. Система має аналізувати відстань між автомобілями, їх швидкість та обробляти дорожні знаки, що в сукупності принесе максимальну безпеку водію.

Висновки до розділу 1

Був виконаний аналіз літературних та наукових джерел й існуючих рішень, які вирішують питання безпеки водія під час поїздки. Відповідно до виконаного аналізу, були розроблені вимоги щодо реалізації системи. У якості технології збору інформації було обрано критичні данні, для забезпечення максимальної безпеки.

Враховуючи недоліки та особливості використання проаналізованих систем, можна запропонувати основні параметри, що необхідні для кращого досвіду керування транспортними засобами:

- Розпізнавання дорожніх знаків;
- Вимірювання швидкості та відстані;

2 ПРОГРАМНО-АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

Даний розділ містить відомості щодо створення методів аналізу критичних параметрів. Була розглянута принципова схема обробки критичних параметрів, базові алгоритми та принципи роботи пристроїв, способи взаємодії різних компонентів системи. На основі виконаних досліджень була представлена модель за якою може бути побудована система для збору критичної інформації з подальшим виведенням на смартфон.

2.1 Алгоритм розпізнавання дорожніх знаків

Око та камера смартфона - це дві різні засоби сприйняття зображень. Око людини - це природний орган зору, що сприймає світло та перетворює його на сигнали для мозку. Камера смартфона - це електронний пристрій, який захоплює зображення та відео за допомогою оптики та сенсорів, а потім перетворює ці зображення на цифрові дані.

Що спільного між оком людини та камерою смартфона, це їх здатність сприймати світло і зафіксовувати зображення. Обидва можуть бути використані для реєстрації візуальної інформації. Око та камера смартфона також можуть фокусуватися на об'єктах різної відстані для забезпечення ясного зображення. Крім того, обидва можуть реєструвати кольори та деталі, які є важливими для візуального сприйняття.

Головним завданням автоматичного розпізнавання камерою смартфона є аналіз навколишнього простору. Якщо буде розроблений метод, який працює з достатньою точністю, його можна використовувати в системах допомоги водієві, інвентаризації об'єктів придорожньої інфраструктури або для автоматизованого створення навігаційних карт. Однією з основних проблем на дорогах є, як не дивно, дорожні знаки.

Дорожні знаки створені таким чином, щоб бути помітними і легко розпізнаватися, що робить їх відмінним об'єктом для автоматичного розпізнавання. У той же час автоматичні алгоритми повинні бути здатні

розпізнавати знаки в умовах непогоди, змін освітлення, положення точки спостереження, розмиття та перекриттів. Найкращі на сьогоднішній день алгоритми базуються на машинному навчанні і потребують наявності великої та репрезентативної навчальної колекції, щоб подолати всі вищезазначені проблеми.

Розробка спрямована на підвищення безпеки на дорогах та полегшення процесу водіння. Інженери розробляють рішення, які автоматично розпізнають дорожні знаки, фіксують інформацію про допустимій швидкості та обмеження, включаючи напрямок руху, наявність перехресть, поїзних перегонів та інші дані.

Чим більше попереджень система отримує від зовнішнього середовища, тим надійніше стає автомобіль та процес водіння. Водієві фізично важко стежити за всіма параметрами дороги, особливо в тривалих поїздках. Програмне рішення може вирішити проблему з неуважністю та зменшити вплив людського фактору під час руху. Оскільки розпізнавання дорожніх знаків є однією з основних складових, необхідних для допоміжних систем керування та коректної роботи системи, необхідно розвивати транспортну інфраструктуру. Автомобіль повинен самостійно визначати розмітку, обмеження, знаки та умови руху. У разі відсутності дорожніх знаків, функціонування системи буде припинено.

Оскільки більшість аварій виникає через порушення швидкісного режиму, інженери автомобільних компаній було встановлено перед собою завдання вирішити цю проблему. Для досягнення цієї мети в автомобілі встановлюється система розпізнавання дорожніх знаків.

Основні функції цієї системи включають:

- визначення та підтвердження інформації про дорожні знаки;
- пошук інформації в базі даних та повідомлення водія;
- попередження за допомогою світлового або звукового сигналу, якщо швидкість руху не змінюється.

Отже, дивлячись на функції системи на базі камери смартфона, для кращого розпізнавання дорожніх знаків мною було розроблено наступну таблицю класифікацій дорожніх знаків.

Таблиця 2.1 – Класифікація дорожніх знаків

Форма	**Колір**	**Категорія**
Круглі	Жовті	Попереджувальні
Прямокутні	Червоні	Пріоритетні
Трикутні	Зелені	Заборонні
Шестикутні	Сині	Вказівні
Квадратні	Білі	Інформаційно-вказівні
	Інші кольори	Сервісу
		Допоміжні таблички

Виходячи з певних даних та аналізу систем та алгоритмів, для подальшого створення оптимального алгоритму розпізнавання дорожніх знаків мною було запропоновано наступний покроковий опис логіки виявлення об'єктів:

- камера аналізує навколишнє середовище і зчитує дані про дорожні знаки;

- система виявляє форму, схожу на знак;

- розпізнавання кольору і наявності додаткових символів;

- пошук відповідності в базі даних;

- інформування водія. Послідовність розпізнавання типу знаку:

- визначення форми: коло, прямокутник, квадрат;

- аналіз колірної гами;

- зчитування символів або написів на знаку.

2.2 Розрахунок гальмівного шляху

Кожен кваліфікований та досвідчений водій повинен володіти навичками гальмування та маневрування в умовах негоди, на слизькому дорожньому покритті та в екстремальних дорожніх ситуаціях.

Згідно з правилами дорожнього руху, водій повинен забезпечити відповідно до дорожньої обстановки, що склалася, необхідне для безпеки зниження швидкості автомобіля, аж до повної його зупинки. Тому вміння використовувати ефективні методи гальмування є важливим для водія, щоб мати впевненість у забезпеченні безпеки в екстрених ситуаціях на дорозі.

Гальмівний шлях - це відстань, яку проїде автомобіль після активації гальмівної системи до повної зупинки. Цей параметр є лише одним з технічних показників, який у поєднанні з іншими факторами визначає безпеку автомобіля. Важливо зауважити, що гальмівний шлях не включає в себе час реакції водія. Сукупність реакції водія на екстрену ситуацію та відстані від початку гальмування, коли водій натиснув на педаль, до повної зупинки автомобіля називається зупинним шляхом.

Тобто проаналізувавши дану інформацію можна вивести наступну закономірність:

$$L = T + S$$

де:

L – довжина зупинного шляху;

T – час реакції водія;

S – гальмовий шлях.

Зупинний шлях – це відстань, яку проходить транспортний засіб від моменту виявлення небезпеки на дорозі до повної зупинки автомобіля. Зупинний шлях складається з відстані, яку проходить автомобіль за час реакції водія, а також часу спрацьовування гальмівної системи. З урахуванням умов видимості в напрямку руху, водій повинен обрати таку швидкість, щоб зупинний шлях автомобіля не перевищував відстань видимості. В іншому випадку, необхідно знизити швидкість. Довжина зупинної доріжки залежить від реакції водія, стану транспортного засобу та дорожнього покриття.

Зупинний шлях автомобіля обчислюється за формулою:

$$S = (t1 + t2)V + kV2$$

де:

$t1$ - час реакції водія;

$t2$ - час спрацьовування гальм;

k - коефіцієнт ефективності гальмування;

V - швидкість руху автомобіля;

g - прискорення вільного падіння.

Час реакції водія - це проміжок часу від моменту виявлення небезпеки до виконання необхідних дій, таких як переміщення ноги на педаль гальма та натискання на неї. Цей час залежить від індивідуальних особливостей водія, його досвіду водіння, а також положення тіла, рук та ніг щодо керуючих органів автомобіля. Психоемоційний стан водія також впливає на час реакції. Втома, захворювання та особливо алкогольне або наркотичне сп'яніння значно збільшують час реакції.

Дистанція до повної зупинки автомобіля - це гальмівний шлях, який враховує відстань, пройдену машиною після натискання водієм на педаль гальма. Важливо пам'ятати, що зупинний шлях завжди більший за гальмівний, оскільки включає в себе ще й відстань, пройдену автомобілем після виявлення небезпеки та натискання на гальма. Довжина гальмівного шляху залежить від швидкості руху, стану дороги, шин та погодних умов.

У Всесвіті немає такої можливості миттєво зупинятися. Так само, як і будь-який автомобіль, коли водій натискає гальмо педалі, він не може негайно зупинитися. Причина полягає в тому, що для того, щоб автомобіль або будь-який об'єкт у нашому світі зупинився, необхідно, щоб він втратив енергію, яка його рухає. В результаті, кожен автомобіль має свій гальмівний шлях, який він проїжджає від моменту натискання педалі гальма до повної зупинки.

Проте, фактично гальмівний шлях будь-якого автомобіля залежить не лише від його характеристик та гальмівної системи, але й від реакції водія при натисканні педалі гальма. Оскільки для того, щоб прийняти рішення про необхідність гальмування і натиснути педаль гальма, потрібен час, який, хоч і

мінімальний, але достатній, щоб автомобіль встиг проїхати значну відстань. Особливо це важливо при великій швидкості руху, де за декілька мить секунди автомобіль проїжджає велику відстань. Тому, для розрахунку реальної довжини гальмівного шляху необхідно враховувати не лише час і відстань, пройдений автомобілем з моменту натискання водієм педалі гальма до моменту зупинки машини, але й час, необхідний для ухвалення рішення про гальмування. Справа в тому, що при ухваленні рішення про гальмування ми витрачаємо цінні секунди. Ось наприклад:

Час відгуку: Перш ніж водій натисне педаль гальма, він повинен оцінити дорожню ситуацію і визначити, чи гальмування необхідно. Також потрібно зрозуміти, яке необхідне гальмування – повна зупинка автомобіля чи просте зниження швидкості. Зазвичай, згідно з численними дослідженнями, для більшості водіїв для цього потрібно близько 0,1 секунди.

Час, необхідний для натискання педалі гальма: Після того, як водій зрозумів, що повинен гальмувати, необхідно ще приблизно 0,8 секунди, щоб перемістити ногу з педалі газу на педаль гальма і натиснути її. В таблиці 2.1 представлено залежності реакції водія та пройденого автомобілем шляху:

Таблиця 2.2 – Залежність довжини зупинного шляху

Швидкість автомобіля	Шлях до початку гальмування	Гальмівний шлях	Зупинний шлях
20 км/год	6м	6м	12м
40 км/год	12м	24м	36м
60 км/год	18м	55м	73м
80 км/год	24м	76м	100м
100 км/год	38м	105м	143м

2.3 Методи збору критичної інформації

Автоматизація передавання некритичних показань пацієнтів відіграє важливу роль у трансформації медичної індустрії, впливаючи на якість та ефективність надання медичних послуг.

2.3.1 Згорткові нейронні мережі

Згорткова нейронна мережа (англ. convolutional neural network, CNN) - спеціальна архітектура штучних нейронних мереж. спеціальна архітектура штучних нейронних мереж, запропонована Яном Лекуном і націлена на ефективне розпізнавання зображень, входить до складу технологій глибокого навчання (англ. deep leaning).

Ця технологія спирається на аналогії зорової ори головного мозку і його принципами роботи, в якій було відкрито прості клітини, які реагують на прямі лінії під різними кутами, і складні клітини, які реагують на активацію певного набору простих клітин.

Виходячи з цього, ідея згорткових нейронних мереж полягає в тому, що чергуються згорточні шари та шари підвибірок (англ. convolution layers) (англ. subsampling layers).

Архітектура цієї мережі представлена на рисунку 2.1:

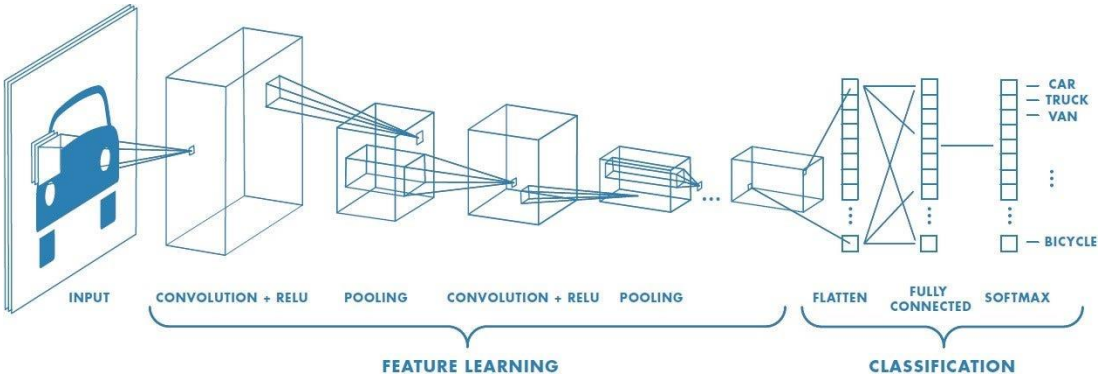


Рисунок 2.1 – CNN архітектура

Одним із важливих моментів у вивченні згорткових нейронних мереж є визначення "поділюваних ваг є визначення "поділюваних" ваг, тобто певна частина нейронів одного з досліджуваних шарів нейронної мережі може використовувати однакові вагові коефіцієнти. Такі нейрони використовують

однакові ваги, після знаходження, об'єднуються в карти ознак, а кожен з нейронів, пов'язаний з частиною нейронів попереднього шару через вхідні карти ознак.

Під час обчислення мережі кожен нейрон виконує згортку певної області попереднього шару, який визначається безліччю нейронів, пов'язаних із даним нейроном. Шари згорткової нейронної мережі, побудовані за цим принципом, називаються згортковими шарами.

У згортковій нейронній мережі, крім згорткових шарів, можуть бути шари підвибірки, які виконують функції зменшення розмірності простору карт ознак, а також повнозв'язні шари, у яких вихідний шар, як правило, завжди повнозв'язний.

2.3.2 Далекомір оптичного діапазону

Далекомір – це частний випадок лідара, у якого паралельно узький кут спостереження. Пристрій дивиться вперед в узькому сегменті і не отримує сторонніх даних, крім видалення об'єктів. Так працює оптичний Далекомір, заснований на принципі ToF. Робоча дистанція залежить від використовуваного джерела світла: для IK-светодиодів це десятки метрів, а лазерні лідари здатні стріляти світлом на кілометри уперед (рис. 2.3).

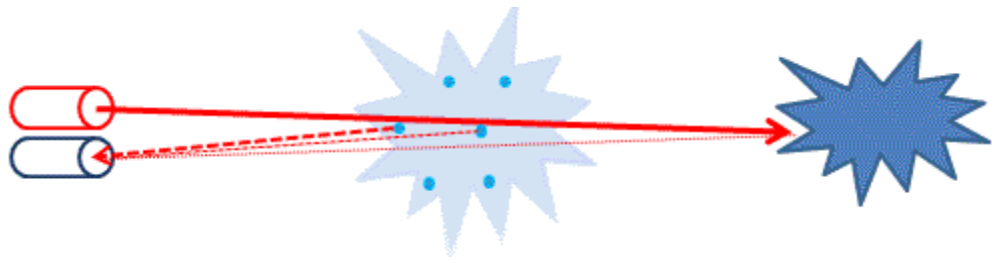


Рисунок 2.2 – Принцип роботи оптичного далекоміру

Одним із способів вимірювання часу польоту світлового променя є використання імпульсного лазера, а потім безпосередньо вимірювання часу,

що минув. У таких пристроях потрібна електроніка, здатна розрізняти пікосекунди, і тому вони дуже дорогі. Іншим методом є вимірювання фазового зсуву відбитого світла.

Для вимірювання фазового зсуву використовується колімований інфрачервоний лазер. Для поверхонь, шорсткість яких перевищує довжину хвилі падаючого світла, відбудеться дифузне відображення. Компонент інфрачервоного світла буде повертатися майже паралельно променю, що проходить для об’єктів.

Датчик вимірює зсув фаз між переданим і відбитим сигналами. На малюнку показано, як цю техніку можна використовувати для вимірювання відстані. Довжина хвилі модулюючого сигналу відповідає рівнянню:

$$c = f * \tau$$

де:

c – швидкість світла;

f – частота модуляції;

τ – відома довжина хвилі модуляції.

Загальна відстань D', яку проходить світло, що випромінюється, дорівнює:

$$D = B + 2A = B + (0 * \tau)/2\pi$$

де:

A - виміряна відстань; B – відстань від одиниці вимірювання фази.

Таким чином, необхідна відстань D між світлодільником і ціллю визначається як:

$$D = \tau * \theta /4\pi$$

де θ – електронно виміряна різниця фаз між пропущеним і відбитим світловими променями.

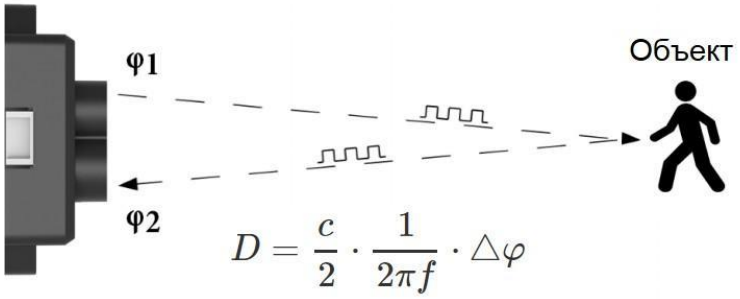


Рисунок 2.3 – Приклад роботи Lidar-датчику

Можна показати, що діапазон обернено пропорційний квадрату амплітуди прийнятого сигналу, що безпосередньо впливає на точність датчика.

2.3.3 Система навігації

На сьогоднішній день технології розпізнавання людської активності за допомогою датчиків телефону стрімко розвиваються. Смартфон містить у своїй конструкції різноманітні сенсори місцезнаходження та положення у просторі, а також мережеві технології: GPS, WLAN (також відомий як Wi-Fi), антени мобільного зв’язку, акселерометри, гіроскопи, магнітометри, барометри, камери, мікрофони тощо.

Для визначення місцезнаходження транспортного засобу, для подальшого визначення швидкості руху на відкритій місцевості, були досліджені три види рішень для позиціонування автомобіля (рисунок 2.4): — супутникові рішення; — рішення на основі сенсорів; — рішення на основі радіочастотних сигналів.

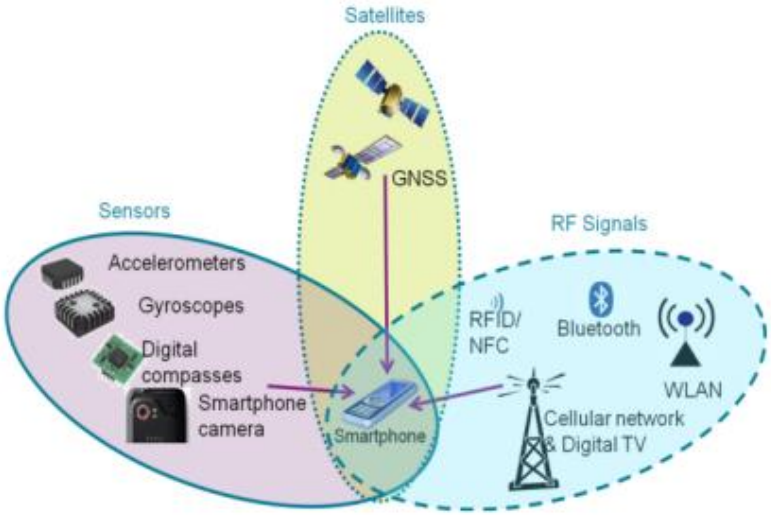


Рисунок 2.4 – Система навігації

На відкритій місцевості навігація переважно визначається за допомогою супутникових технологій. Маючи широкий обхват та високу точність, найбільш застосованою технологією є глобальна навігаційна супутникова система (GNSS), а саме глобальна система позиціонування (GPS). Існуюча технологія, базована на радіочастотних сигналах, представляє певні альтернативи позиціонування транспортного засобу. Методи з використанням мобільної мережі та WLAN зараз стандартні функції телефонів, зокрема iPhone та Android смартфонів. Крім того, компанія Nokia розробила Wi-Fi тріангуляційну систему, за допомогою якої користувач з більшою вірогідністю отримає точніше місцезнаходження.

Вбудовані сенсори мобільного телефону в парі з зовнішніми датчиками надають можливість неперервної навігації та аналізу стану автомобіля, що дозволяє притримуватись точності орієнтації та швидкості руху транспортного засобу.

2.4 Системи обробки критичних параметрів

2.4.1 Обробка даних GPS

Як вже було вище зазначено, швидкість транспортного засобу можна розрахувати з використанням даних, отриманих від датчиків GPS.

Для розрахунку швидкості на основі GPS координат можна використовувати наступну формулу:

$S = d/\Delta t$

Де:

S - швидкість аналізованого транспортного засобу;

d - відстань між двома точками виміряне та переведене у кілометри;

Δt- час, за який подолано відстань між двома точками, виміряне та переведене у кілометри.

В GPS дані часто надаються у вигляді послідовності координат (lat_1, lat_2) та (lon_1, lon_2) які представляють дві точки, між якими відбувається вимірювання швидкості.

Для розрахунку відстані d між двома координатами можна використовувати формулу гаверсинусів:

$$d = R \cdot arccos(sin(lat_1) \cdot sin(lat_2) + cos(lat_1) \cdot cos(lat_2) \cdot cos(lon_2 - lon_1))$$

Де:

R - радіус Землі, приблизно 6371 км;

lat_1, lat_2- широти першої і другої точок відповідно в радіанах;

lon_1, lon_2 - довготи першої і другої точок відповідно (в радіанах);

Для визначення часового інтервалу t між двома точками потрібно знати час, відповідний кожній точці, і обчислити різницю між цими часами.

Після того, як буде обчислено d і t, можна використовувати формулу швидкості, описану вище, для розрахунку швидкості на основі цих даних.

2.4.2 Виявлення об'єктів за кольором

Основна концепція алгоритму полягає в пошуку об'єкта за кольором. Оскільки знаки дорожнього руху мають тільки кілька кольорів, а більшість

тільки червоного і синього, то невелика кількість знаків дорожнього руху, що залишилася, мають інші кольори.

Далі розглянемо послідовність дій алгоритму, щоб виявити його плюси і мінуси. На самому початку беремо зображення, представлене у колірному просторі RGB.

Передбачається, що на ньому є зображені дорожні знаки. Отже, що для виявлення знака дорожнього руху на зображенні за кольором нам буде потрібно обробити 3 складові, такі як матриці M на N для кожного кольору - MxNx3, а це значить, що буде потрібно більше обчислювальних потужностей. Варто окремо зазначити, що зміна однієї зі складових у колірному просторі впливає на значення інших складових, тобто зміна G впливає на значення R і B.

HSB (Hue, Saturation, Brightness - тон, насиченість, яскравість) - колірна модель:

Hue - колірний тон, (наприклад, жовтий, червоний або блакитний). Варіюється в межах "0-360°", а так само іноді приводиться до діапазону значень "0-100" або "0-1".

Saturation - насиченість. Варіюється в межах "0-100" або "0-1". Що більший цей параметр, то "чистіший" колір, тому. А чим ближче цей параметр до нуля, тим ближче колір до нейтрального сірого.

Brightness (яскравість). Також задається в межах "0-100" і "0-1". Слід зазначити, що пристрій моделі HSV є найбільш близьким до людського сприйняття кольорів.

Після перекладу з одне колірної моделі в іншу знаємо діапазони значень HSB кольори знаку при різних видах освітлення (День, вечір, ніч).

Наприклад, якщо вивчити дані з Інтернету ми можемо побачити, що значення відтінків червоного кольори на знаку знаходиться в діапазонах, представлених у таблиці 2.

Таблиця 2.3 Діапазони червоного кольори

	Ясно	Дощ	Вечір	Ніч
HSV	300<H<360	300<H<360	300<H<360	22<H<360
	0.7<S<1	0.7<S<1	0.7<S<1	0.7<S<1
	0.5<V<1	0.5<V<1	0.5<V<1	0.5<V<1

Пошук знаків дорожнього руху відбувається шляхом здійснення перевіркою кожного пікселя на зображення, наступний крок є підстановкою значення пікселя в проміжки, які можна, можливо побачити таблиці 2. Після підстановки пікселя в проміжок порівнюються умови та якщо вони дотримуються, то піксель стає білого кольору, а якщо ні, то чорного. Після того, як завершиться завдання перевірки абсолютно всіх пікселів, зображення стане чорно білим і зокрема будуть виділені тільки контури знаків дорожнього руху.

Плюси даного підходу:

- простота реалізації;
- низьке навантаження на пристрій, за допомогою якого виробляють обчислення;
- стійкість до зміні погодних умов;
- адаптація алгоритму під всі існуючі класи дорожніх символів.
- Мінуси даного підходу:
- виявлення всіх дорожніх знаків на зображення, підходящих під певний колір, навіть якщо цього не потрібно;
- виявлення інших об'єктів, колір яких схожий з кольором дорожніх знаків;
- вигоряння або забруднення покриття дорожнього знаку можуть вплинути на результат роботи.

2.4.3 Виявлення об'єкта по формі

Дані методи будуть найбільш актуальні для знаків дорожнього руху, завдяки формі самих символів. Вони складається з простих примітивів такі як:

— коло;
— трикутник;
— квадрат;
— прямокутник.

На першому етапі отримуємо зображення і готуємо його. Підготовка полягає в тому, щоб перевести зображення з кольорового в зображення в градаціях сірого і виділення кордону об'єктів.

Надалі виготовляється пошук контурів, які є замкнутими, для того щоб серед них знайти контури, які нагадують фігуру.

На заключному етапі відбувається відбір задоволених контурів поставленими умовами. Крім приватних випадків важко знайти детектор кордонів, Котрий працював би суттєво краще, чим детектор Кенні. Завданням Кенні була розробка оптимального алгоритму детектування кордонів, Котрий б задовольняв наступним вимогам:

— Високий рівень виявлення кордонів;
— Локалізація, задовольняє всім умовам (точне перебування становище Межі);
— на кордон тільки один відгук.

З цих вимог обчислювалася цільова функція вартості помилок, яка мінімізувала для знаходження оптимального лінійного оператора для створення згортки зображення.

2.4.4 Детектор кордонів Кенні

Одним з найкращий детекторів кордонів в наш час є алгоритм Кенні.

Алгоритм детектора кордонів Кенні виконує не лише обчислення градієнта згладженого фільтром Гауса зображення. У контурі кордону видаляються локально не максимальні крапки, що лежать поряд з кордоном.

В алгоритмі Кенні використовується інформація про напрямок кордону для того, щоб не задіяти або прибирати точки. Які знаходяться безпосередньо поряд з кордоном, робиться це для того, що б безпосередньо не розірвати кордон поблизу градієнтів локальних максимумів. Далі з використанням пороговий фільтрації видаляються слабкі межі, а фрагмент кордону під час фільтрації обробляється як ціле.

У разі, якщо значення градієнта на якомусь знайденому фрагменті верхній поріг буде перевищено, то тоді цей фрагмент залишається також

«допустимим» кордоном і в тих місцях, де значення градієнта знижується нижче цього порога, до тих доки, поки вона не стане нижче нижнього порога.

Якщо вийде так, що якщо на всьому фрагменті немає значення більше крапки верхнього порога, то він забирається. Такі дії дозволяють суттєво знизити число розривів в вихідних кордонів.

Якщо увімкнути в алгоритм Кенні шумозаглушення - це підвищить стійкість результатів, але в теж час збільшить обчислювальні витрати, а також призведе до спотворення та можливої втрати інформації кордонів. З допомогою такого алгоритму здійснюється заокруглення кутів на об'єктах та руйнуються кордону в точках з'єднань.

Висновки до розділу 2

У рамках даного розділу були розглянуті різні методи збору критичних параметрі. Були розроблені алгоритми обробки критичної інформації, та визначено найкращі формули для розрахунків.

На основі виконаних досліджень були створені базові алгоритми, за якими передбачується побудувати систему. Для кращої роботи алгоритму також запропоновано власну класифікацію розпізнавання дорожніх знаків. Головною задачею системи я виявлення класу за формою, кольором, та інформацією, що несе певний об’єкт. Ця система повинна покращити розпізнавальні алгоритми та дати значний приріст для обробки даних.

У якості архітектури мережі було прийнято рішення використати згорткові нейронні мережі. Крім того було визначенні способи збору інформації, що буде об’єднана а одну систему. Дана система задовольняє поставленим вимогам щодо ефективності, простоти, дешевизни та швидкості обробки інформації.

3 ТЕХНІЧНО-АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ ЗБОРУ ІНФОРМАЦІЇ ДЛЯ ВОДІЯ

У даному розділі розглядається питання технічно-апаратного оснащення системи збору інформації для водіїв. Було проведено порівняння та пошук емпіричних даних щодо використання мікроконтролерів як основи для побудови системи. Досліджується питання вибору та підключення модулів та датчиків з смартфоном.

3.1 Мікроконтролер (Arduino Uno)

В якості програмованої платформи вирішено обрати один з мікроконтролерів родини Arduino. Arduino підходить, як початківцям так і професіоналам та являє собою електронний конструктор з зручною платформою швидкої розробки електронних пристроїв.

На рис. 3.1 зображено апаратну платформу Arduino Uno.

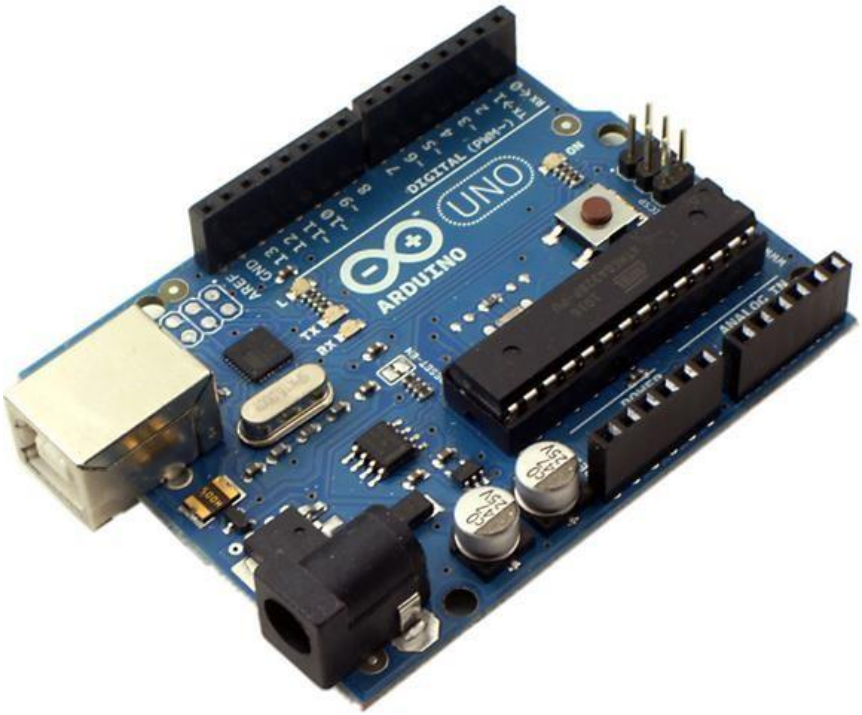


Рисунок 3.1 – Arduino Uno

Arduino Uno – це апаратна платформа, побудована на ATmega328. Цей контролер містить чотирнадцять цифрових входів та виходів (шість з них можуть бути використані як виходи ШІМ), генератор кварцовий 16 МГц, шість аналогових входів, роз'єм ICSP, силовий роз'єм, USB роз'єм та кнопку рестарту. Для початку роботи з цією платформою потрібно підключити її до комп'ютера за допомогою кабелю USB. Також за допомогою батареї або адаптера AC / DC можна подати живлення.

Arduino Uno використовує мікроконтролер ATmega8U2 що відрізняється від платформ, що використовували мікроконтролер FTDI USB для контакту з USB.У табл. 3.1 наведені технічні характеристики Arduino Uno.

Таблиця 3.1 – Технічні характеристики Arduino Uno

Мікроконтролер	**ATmega328**
Вхідна напруга (рекомендована, гранична)	5 В, 7-12 В
Цифрові Входи та Виходи	14 (6 з яких можуть використовуватися як виходи ШІМ)
Аналогові входи	6
Постійний струм через вхід та вихід	40 мА
Постійний струм для виведення 3.3 В	50 мА
Флеш пам`ять	32 Кб (ATmega328) з яких 0.5 КБ Використовуються для завантажувача

ОЗУ	2 КБ (ATmega328)
Тактова частота	16 МГц
Габарити	68 × 53 × 15 мм
Ціна	600 грн

Для цього проекту було вирішено обрати саме такий контролер, так як його характеристики найбільш повно відповідають даній розробці.

3.2 Trema GPS модуль ATGM336H

Trema GPS модуль ATGM336H - є навігаційним пристроєм, що дає змогу визначити свої координати за широтою, довготою та висотою. Додатково модуль здатний визначити поточну дату, час, швидкість і напрямок пересування.

Модуль отримує дані на основі інформації, що надходить із супутників навігаційних систем GPS (США), Глонасс (росія) і Beidou (Китай).

Модуль самостійно обробляє отриману інформацію і передає дані по шині UART у вигляді текстових повідомлень у форматі протоколу NMEA 0183, відрізняється низьким енергоспоживанням і високою чутливістю.

Рисунок 3.2 – Trema GPS модуль ATGM336H

Специфікація :

— Напруга живлення: 3,3 В або 5 В, підтримуються обидві напруги.

— Живлення резервної батареї: 3 В.

— Струм, споживаний модулем: до 25 мА.

— Інтерфейс: UART.

— Швидкість шини UART: 4800, 9600 (за замовчуванням), 19200, 38400, 57600, 115200 біт/с.

— Конфігурація шини UART: 8 біт даних, без перевірки парності, з одним стоповим бітом.

— Рівень логічної 1 на лініях шини UART: 3,3 В.

— Протокол передавання даних: NMEA 0183.

— Частота оновлення виведених даних: від 1 (за замовчуванням) до 10 Гц.

— Підтримувані навігаційні системи: GPS (США), Глонасс (Росія) і Beidou (Китай).

— Час холодного старту: ≤ 35 сек.

— Час гарячого старту: ≤ 1 сек.

— Точність позиціонування: <2 м.

— Точність вимірювання швидкості: < 0,1 м/с.

— Максимальна висота: 18000 м.

— Робоча температура: від -40 до +85 °C.

— Габарити: 30 х 30 мм.

— Вага: 15 г.

Підключення:

На платі модуля розташований роз'єм із 5 виводів.

TX - вихід даних шини UART від модуля. Підключається до виводу RX Arduino.

RX - вхід даних шини UART у модуль. Підключається до виводу TX Arduino.

Vcc - вхід живлення 3,3 або 5 В.

GND - загальний вивід живлення.

PPS - вихід меандра з частотою 1 Гц. Передній фронт імпульсів збігається з часом UTC.

Univsersal Asynchronos Reciever-Transmitter - це фізичний пристрій приймання та передавання даних по двох дротах. Він дає змогу двом пристроям обмінюватися даними на різних швидкостях.

У специфікацію UART не входять аналогові рівні, на яких ведеться спілкування між пристроями, UART - це протокол передавання одиниць і нулів, електричну специфікацію на себе беруть інші стандарти, як-от TTL (transistor-transistor logic - транзисторно-транзисторна логіка), RS-232, RS-422, RS-485 тощо (RS*[англ.*recommended standard] - рекомендований стандарт). Наразі в мікроконтролерах використовується здебільшого TTL (або точніше CMOS) UART для з'єднання не більше двох пристроїв. У наших прикладах ми часто називаємо його послідовним портом.

У разі під'єднання модуля не до апаратної, а до програмної шини UART, ви самі призначаєте виводи TX і RX Arduino, до яких під'єднується модуль.

Модуль дає змогу:

- Отримувати широту, довготу і висоту над рівнем моря.
- Отримувати швидкість і курс на істинний полюс.
- Отримувати дату і час UTC.
- Отримувати кількість спостережуваних супутників і супутників, які беруть участь у позиціонуванні.
- Отримувати дані про супутники: рівень прийому, положення і тип навігаційної системи.
- Отримувати геометричні фактори погіршення точності та точність позиціонування.
- Визначати помилку визначення координат, дати, часу, швидкості та курсу.

— Налаштовувати швидкість передачі даних по шині UART.
— Налаштовувати частоту оновлення виведених даних.
— Вибирати версію протоколу NMEA 0183 для формування повідомлень, що відправляються.
— Налаштовувати склад повідомлень NMEA 0183.
— Обирати супутникові навігаційні системи, дані яких потрібно отримувати.
— Обирати динамічну модель навігаційної платформи.
— Перезавантажувати модуль з вибором типу старту (холодний / гарячий).
— Перезавантажувати модуль зі скиданням налаштувань до заводських. Зберігати налаштування в енергонезалежну пам'ять модуля.

3.3 Lidar-Lite v3

У рамках даного розділу був виконаний огляд доступних технічних засоібв, на основі яких буде Лідар (LiDAR, Light Identification Detection and Ranging – світлове виявлення та визначення дальності) – технологія, яка за допомогою обробки сигналу відбитого світла отримує інформацію про об'єкти, що знаходяться на відстані. Іншими словами, пристрій випускає велику кількість лазерних імпульсів за певний відрізок часу, які виявляють об'єкти навколо себе та визначають відстань до них.

Завдяки лідару автомобіль бачить дорогу, інші авто, знаки, дерева, будинки та перешкоди. Більше того, він може визначати широкий спектр неметалічних матеріалів: скелі, дощ, хімічні сполуки, аерозолі, хмари та навіть окремі молекули.

Для проекту був обраний Lidar-Lite v3 (рис. 3.2)

Рисунок 3.3 – Lidar-Lite v3

Це невеликий одноточковий датчик дальності з унікальними оптичними та електричними алгоритмами, який може реалізувати точне та швидке вимірювання відстані, а також зберігати стабільну та високочутливу продуктивність.

TFmini Plus заснований на принципі TOF, а саме на принципі "Час польоту". Виріб періодично випромінює модуляційну хвилю ближнього інфрачервоного променя, яка буде відображена після контакту з об'єктом.

Пристрій отримує час польоту, вимірюючи різницю фаз у зворотному напрямку, а потім обчислює відносний діапазон між продуктом та об'єктом виявлення.

Датчик TFmini Plus значно покращив усі свої показники, збільшивши частоту кадрів з 100 Гц до 1000 Гц, зменшивши сліпу зону до 10 см, підвищивши точність та стабільність. Більше того, TFmini Plus знаходиться в корпусі з типом захисту IP65, що може ефективно запобігати потраплянню води та пилу, а також робить продукт більш пристосованим до зовнішнього світла або навколишнього середовища з різними температурами та відбиттям.

Датчик може забезпечити набагато менше енергоспоживання та більш гнучку частоту виявлення. Датчик сумісний із зв’язком по UART та IIC

(надіслати команду для перемикання) інтерфейсах та живиться від стандартних 5В. Середня потужність 0.55 Вт.

Цей датчик добре працює з усілякими контролерами типу Arduino. Його можна легко інтегрувати в систему при використанні з бібліотекою Arduino, розробленою від DFRobot.Технічні характеристики TFmini Plus наведені в таблиці 3.2.

Таблиця 3.2 – Технічні характеристики Lidar TF Luna

Інтерфейс	UART, I2C, I/O
Інше	допуск - 5мм
Відстань	0.1-40м
Діапазон температур	-20℃~60℃
Захист	IP65
Кут	3.6°
Логічні рівні	LVTTL (3.3В)
Матеріал	ABS+PC
Напруга	5В±0.5В
Потужність	450мВт (режим низької потужності: 85мВт)
Струм	≤200мА
Частота	кадрів: 1-10000 Гц (регулюється)
Точність	± 5 см @ (0,1-6м), ± 1% @ (6м-12м)
Довжина	950 нм; кабеля - 25см

3.4 HC-06 Bluetooth

HC-06 Bluetooth - модуль, який використовується для реалізації бездротового зв'язку з різними пристроями, як-от телефон або планшет, а також може використовуватися для обміну даними між двома модулями Arduino.

Характеристики:

- Живлення: 3,3В - 6 В;
- Максимальний струм: 45 мА;
- Швидкість передачі даних: 1200-1382400 бод;
- Робочі частоти: 2,40 ГГц - 2,48 ГГц;
- Підтримка специфікації bluetooth: версія 2.1;
- Дальність зв'язку: до 30 м;
- Для підключення до смартфона використовуються такі дані:
- Пароль: "1234" або "0000";
- Швидкість передачі даних: 9600 бод;
- Ім'я модуля: HC-06;

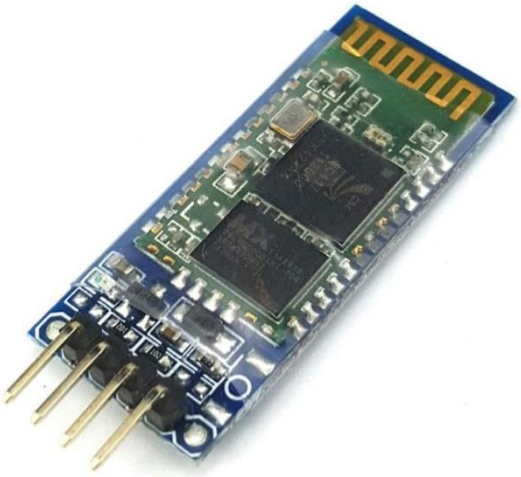


Рисунок 3.4 – HC-06 Bluetooth

Для зручності підключення до Arduino скористайтеся Trema Shield, Trema Power Shield, Motor Shield або Trema Set Shield. Для швидкого та зручного підключення рекомендується використовувати 4-провідний шлейф "мама-мама"

VCC - 5V;

GND - GND;

RX - TX;

TX - RX;

MCU-INT - світлодіод відображення статусу;

Clear (Reset) - скидання і перезавантаження модуля;

3.5 Моделювання схеми

У цій схемі мікроконтролер Arduino Uno використовується як центральний контролер, який отримує дані від Lidar, GPS-модуля та надсилає їх через Bluetooth для подальшої обробки або відображення на пристрої з Bluetooth підтримкою. Зв'язок з GPS модулем дає можливість отримання географічних координат, а Lidar-Lite допомагає виміряти відстань до об'єктів у визначеному напрямку.

1. **Мікроконтролер Arduino Uno**:
   - Порт USB для програмування та живлення.
   - GPIO-порти для підключення інших пристроїв.
2. **Lidar-Lite v3**:
   - Lidar-Lite v3 підключається через інтерфейс I2C або PWM. Для зручності використання рекомендується використовувати I2C, якщо це можливо. В цьому випадку потрібно підключити SDA і SCL датчика до відповідних GPIO-портів Arduino Uno (A4 та A5 відповідно).
3. **GPS-модуль Trema ATGM336H**:
   - GPS-модуль можна підключити до мікроконтролера через інтерфейс UART. Підключіть TX GPS-модуля до RX (пін 0) Arduino Uno, та RX GPS-модуля до TX (пін 1) Arduino Uno.
4. **Модуль Bluetooth HC-06**:
   - Модуль Bluetooth HC-06 також підключається через UART. Підключіть TX HC-06 до RX (пін 1) Arduino Uno, та RX HC-06 до TX (пін 0) Arduino Uno.

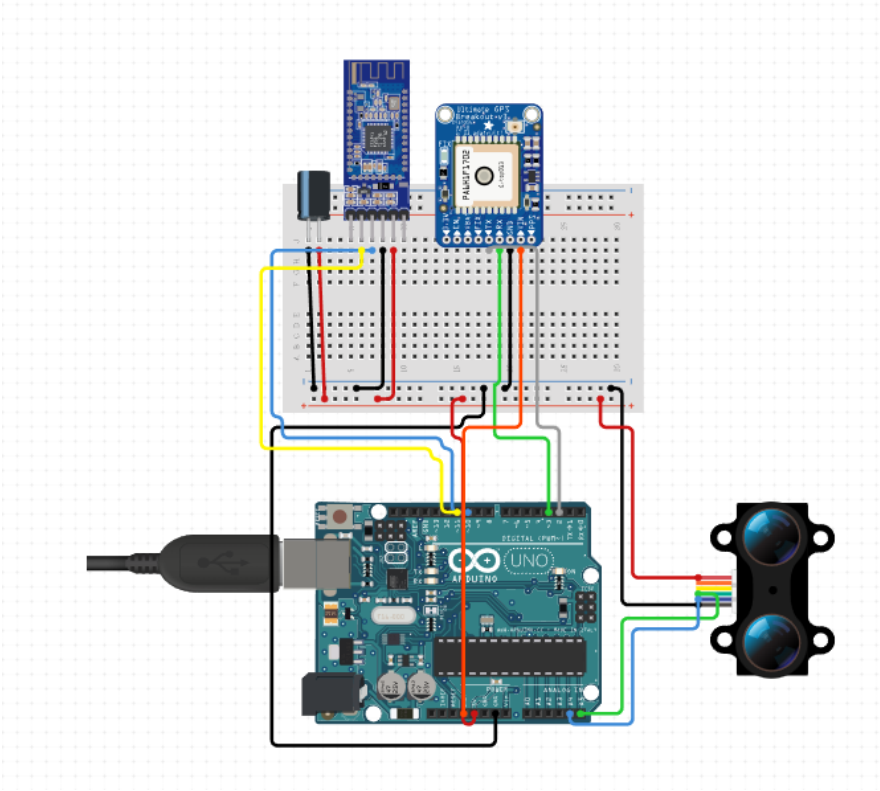


Рисунок 3.5 – Схема підключення пристрою

Висновки до розділу 3

У рамках даного розділу був виконаний огляд доступних технічних засоібв, на основі яких буде побудована система збору інформації з виводом критичних параметрів для водіїв. Були проаналізовані показники, які необхідні водіям для кращого керування транспортним засобом. Відповідно до результатів у якості датчиків та пристроїв були обрані:

– Arduino Uno;
– GPS Trema ATGM336H;
– Lidar-Lite v3;
– HC-06 Bluetooth.

В підсумку була розроблена модель схеми підключення пристроїв, що виникає собою систему збору та обробки інформації для водіїв.

4 РОЗРОБКА СИСТЕМИ ЗБОРУ ІНФОРМАЦІЇ ДЛЯ ВОДІЯ

4.1 Arduino IDE

Як програмне середовища розробки для Arduino, буде використовуватися Arduino IDE, оскільки воно відповідає всім основним вимогам:

кросплатформене;

безкоштовне;

відкритий вихідний код;

легка настройка, установка і простота у використанні;

різноманітність бібліотек, завдяки яким можна розширювати функціонал програми.

Arduino IDE - інтегроване середовище розроб лення для Windows, MacOS і Linux, розроблене на C та C++, призначене для створення і завантаження програм на Arduino-сумісні плати, а також на плати інших виробників.

Інтерфейс середовища програмування Arduino IDE показано на рис. 3.4.

В якості мови програмування для Arduino, буде використовуватися мова C/C++. Вона багатофункціональна і не вимагає додаткових налаштувань в середовищі розробки Arduino IDE.

C/C ++ – об'єктно-орієнтована мова програмування, що забезпечує модульність, роздільну компіляцію, обробку винятків, абстракцію даних. Так само є одним з найбільш широко використовуваних і поширених мов програмування. Використовується не тільки в розробці програмного забезпечення, але і при розробці драйверів різноманітних пристроїв. Добре підходить для програмного середовища Arduino IDE, так як спочатку мова програмування пристроїв Arduino заснована на C/C ++. На даний момент це найпоширеніший і доволі зручний спосіб запрограмувати мікроконтролер.

Є три основні частини, з яких складається мова програмування Arduino. Перш за все, функції, які дозволяють керувати дошкою. За допомогою функцій

можливо аналізувати символи, виконувати математичні операції та виконувати різні інші завдання - наприклад, digitalRead () та digitalWrite () дозволяють читати або писати значення.

Кожен скетч, написаний мовою Arduino, містить дві функції. Це setUp () та loop (). Скетч завжди починається з setUp (), який виконується один раз після увімкнення або скидання плати. Після його створення використовується loop (), щоб повторювати програму повторно, доки не вимкнеться живлення або не відбудеться скидання плати. Також є значення Arduino, які представляють константи та змінні. Більшість типів даних (array, bool, char, float тощо) подібні до тих, що існують у C ++. Можливо виконати перетворення типу. Остання частина мови ардуїно називається структурою. Вона містить невеликі елементи коду, такі як оператори.

У програмуванні для Arduino включення зовнішніх бібліотек є поширеною практикою для розширення функціональності плати Arduino.

Наступні бібліотеки:

#include <LIDARLite.h>

Ця бібліотека надає швидкий доступ до всіх основних функцій LIDAR-Lite через інтерфейс Arduino. Крім того, вона може надати користувачеві будь-якої платформи з шаблоном для написання власного програмного коду.

Файл LIDARLite.h ймовірно містить оголошення та визначення, необхідні для взаємодії з датчиком LIDAR таким як Garmin LIDAR-Lite.

Включаючи цей файл, ви отримуєте доступ до функцій, констант та типів даних, визначених у бібліотеці LIDARLite, що спрощує комунікацію з датчиком LIDAR у вашій Arduino-програмі.

#include <iarduino_GPS_ATGM336.h>

Рядок #include <iarduino_GPS_ATGM336.h> також є директивою препроцесора у мовах програмування C або C++. Ця директива вказує компілятору на включення вмісту файлу iarduino_GPS_ATGM336.h під час процесу компіляції.

У контексті Arduino, включення цього файлу відноситься до використання бібліотеки, яка надає можливість взаємодії з модулем GPS ATGM336. Ця бібліотека містить необхідні оголошення та функції для роботи з GPS-модулем даної моделі, що дозволяє отримувати дані про місцезнаходження з супутників за допомогою Arduino.

Включаючи цей файл, ви отримуєте доступ до функцій, констант і інших визначень, які дозволяють вам працювати з модулем GPS ATGM336 у вашій програмі на Arduino.

#include <SoftwareSerial.h>

Рядок #include <SoftwareSerial.h> також є директивою препроцесора у мовах програмування C або C++. Ця директива вказує компілятору на включення вмісту файлу SoftwareSerial.h під час процесу компіляції.

У середовищі Arduino, включення цього файлу вказує на використання бібліотеки SoftwareSerial, яка дозволяє створювати віртуальний UART (Universal Asynchronous Receiver/Transmitter) на будь-яких пінах мікроконтролера Arduino. Це корисно, коли всі вбудовані UART-порти вже використані для інших цілей, але вам потрібно встановити з'єднання з іншими пристроями через UART.

Інклудуючи файл SoftwareSerial.h, ви отримуєте доступ до функцій і класів, які дозволяють вам налаштовувати і взаємодіяти з віртуальними UART-портами на платах Arduino, що реалізовані програмно.

#include <opencv2/opencv.hpp>

Рядок #include <opencv2/opencv.hpp> є директивою препроцесора у мові програмування C++. Ця директива вказує компілятору на включення основного заголовкового файлу бібліотеки OpenCV, opencv.hpp, під час процесу компіляції.

OpenCV (Open Source Computer Vision Library) є популярною відкритою бібліотекою комп'ютерного зору та обробки зображень. Включення цього заголовкового файлу надає доступ до функціоналу, який надається

бібліотекою OpenCV, такого як обробка зображень, алгоритми комп'ютерного зору та різноманітні утиліти для роботи з зображеннями та відео.

За допомогою OpenCV ви можете виконувати завдання, такі як маніпулювання зображеннями, виявлення об'єктів, вилучення ознак, зшивання зображень та багато іншого, що робить її потужним інструментом для різноманітних застосувань у сфері комп'ютерного зору. Включення <opencv2/opencv.hpp> заголовкового файлу зазвичай є першим кроком у використанні функціоналу OpenCV у вашому програмному забезпеченні на мові програмування C++.

Як підсумок підключення бібліотек ми можемо реалізувати роботу нашої системи з датчиками, що наведено у додатку Б.

4.2 Open CV

Для реалізації алгоритму розпізнавання дорожніх знаків за допомогою камери смартфона краще всього підходить бібліотека з відкритим кодом OpenCV.

OpenCV - це open source бібліотека комп'ютерного зору, яка призначена для аналізу, класифікації та обробки зображень. Широко використовується в таких мовах як C, C++, Python і Java.

Перше, що нам необхідно зробити - це імпортувати бібліотеку. Є кілька шляхів імпорту, найпоширеніший - це використовувати вираз:

```
import cv2
```

Також можна зустріти таку конструкцію для імпорту цієї бібліотеки:

```
from cv2 import cv2
```

Для завантаження зображення ми використовуємо функцію cv2.imread(), де першим аргументом вказується шлях до зображення, а другим аргументом, що є необов'язковим, ми вказуємо, в якому колірному просторі ми хочемо зчитати наше зображення. Щоб зчитати зображення в RGB - cv2.IMREAD_COLOR, у відтінках сірого - cv2.IMREAD_GRAYSCALE. За замовчуванням цей аргумент приймає значення cv2.IMREAD_COLOR. Ця

функція повертає 2D (для зображення у відтінках сірого) або 3D (для кольорового зображення) масив NumPy. Форма масиву для кольорового зображення: висота x ширина x 3, де 3 - це байти, по одному байту на кожну з компонент. У зображеннях у відтінках сірого все трохи простіше: висота x ширина.

За допомогою функції cv2.imshow() ми відображаємо зображення на нашому екрані. Як перший аргумент ми передаємо функції назву нашого вікна, а другим аргументом - зображення, яке ми завантажили з диска, однак, якщо ми далі не вкажемо функцію cv2.waitKey(), то зображення моментально закриється. Ця функція зупиняє виконання програми до натискання клавіші, яку потрібно передати першим аргументом. Для того, щоб будь-яка клавіша була зарахована, передається 0.

```
def loading_displaying_saving():
    img = cv2.imread('girl.jpg', cv2.IMREAD_GRAYSCALE)
    cv2.imshow(sign, img)
    cv2.waitKey(0)
    cv2.imwrite(sign.jpg', img)
```

І, нарешті, за допомогою функції cv2.imwrite() записуємо зображення у файл у форматі jpg (дана бібліотека підтримує всі популярні формати зображень: png, tiff, jpeg, bmp і т.д., тож можна було зберегти наше зображення в будь-якому з цих форматів), де першим аргументом передається безпосередньо сама назва та розширення, а наступним параметром - зображення, яке ми хочемо зберегти.

4.3 Налаштування системи розпізнавання дорожніх знаків

Ми часто стикаємося з необхідністю детектування різних об'єктів на фотографіях або відео. Досить ефективно цю проблему можна вирішити, використовуючи бібліотеки OpenCV.

Принцип роботи програми полягає в такому: на вхід подається зображення, де попередньо ми очікуємо аналіз та класифікацію дорожніх

знаків. При цьому робота з відео така сама, бо в такому разі відео покадрово передається в програму і подальший алгоритм не відрізняється. Зазвичай усі зображення, звичні нашому оку, перебувають у форматі RGB. З ним незручно працювати, оскільки падіння тіні або світіння сонця будуть сильно спотворювати відтінки кольору. А робота програми заснована саме на визначенні кольору. Тому першим кроком буде перехід у колірний простір HSV. Тут ми вже будемо накладати колірний фільтр, який дозволятиме тільки ті кольори, які зустрічаються в дорожніх знаках. Щоб у контурах не було розривів, згладимо дозволені пікселі. Далі малюємо контури навколо дозволених пікселів. Щоб не було багато зайвих, можна поставити обмеження за площею контуру, наприклад, у цьому разі контур має перебувати в межі від 400 до 10000 пікселів. Поетапно всі дії із зображенням представлені на рисунку 4.1.

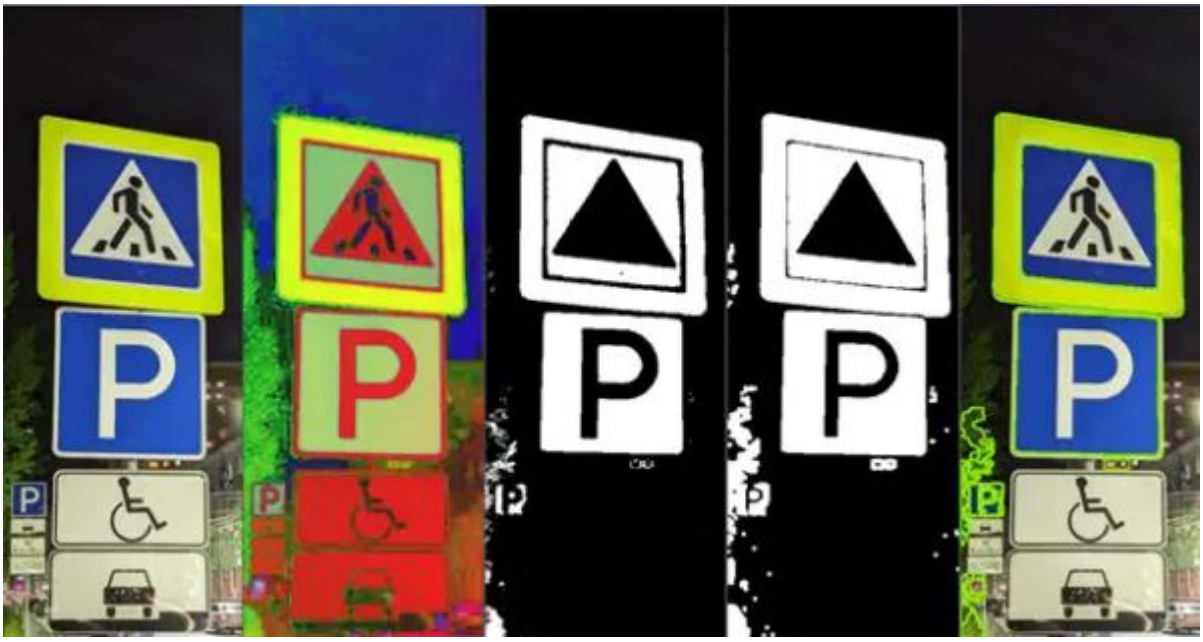


Рисунок 4.1 – Умовна робота датчиків

Як аргумент функція приймає кадр, переводить у колірний формат hsv, накладає маску тільки червоного кольору, згладжує області та знаходить контури. Цикл залишає контури, які підходять за розміром, і відмальовує їх. Функція повертає кадр, з намальованими контурами.

```
def detectionRED(frame, spisok):
    cv.imshow('img', frame)
    frame_hsv = cv.cvtColor(frame, cv.COLOR_BGR2HSV)
    cv.imshow('frame0', frame_hsv)
    frame_mask = cv.inRange(frame_hsv, (0, 162, 67), (255, 255, 255))
    cv.imshow('frame1', frame_mask)
    frame_dilate = cv.dilate(frame_mask, None, iterations=2)
    cv.imshow('frame2', frame_dilate)
    contours, hierarchy = cv.findContours(frame_dilate.copy(), cv.RETR_EXTERNAL, cv.CHAIN_APPROX_SIMPLE
        )
    sort = spisok
    for i in contours:
        area = cv.contourArea(i)

        if 400 < area < 10000:  # можно регулировать дальность
            sort.append(i)

    cv.drawContours(frame, sort, -1, (0, 23, 44), 2)
    return frame, sort
```

Рисунок 4.2 – Реалізація алгоритму збору

Далі, треба тільки вирізати дорожні знаки із зображення і зберегти окремим файлом. Функція приймає як аргумент назву зображення. Передає у функції відмальовування контурів і повертає зображення з намальованими контурами. Реалізація цього алгоритму зображена в коді наведеному на нижче.

```
def detectionIMAGE(image_name):
    start_time = time.time()
    image = cv.imread(image_name)
    print(type(image))
    cadr = cv.imread(image_name)
    spisok = []
    frame, sort = dtRED(image, sort)
    while True:
        cv.drawContours(frame, sort, -1, (0, 234, 135), 2)
        cv.imshow('frame', frame)

       (x, y, w, h) = cv.boundingRect(sort[0])
        detect = cadr[y:y + h, x:x + w]
       cv.imshow('q', detect)

        (x, y, w, h) = cv.boundingRect(sort[1])
        detect = cadr[y:y + h, x:x + w]
        cv.imshow('r', detect)

        (x, y, w, h) = cv.boundingRect(sort[2])
        detect = cadr[y:y + h, x:x + w]
        cv.imshow('u', detect)
        end_time = time.time()
        if cv.waitKey(1) == 27:
            break

    print(':', end_time - start_time)
    return frame
```

Рисунок 4.3 – Реалізація алгоритму коасифікації

Результат виконання алгоритму дуже хороший, тому що навіть у нічний час, програма відмінно відпрацювала і змогла детектувати знак, що знаходиться на задньому фоні. Час роботи програми склав менше секунди.

4.4 Реалізація системи

Для відображення параметрів мікроконтролера на смартфоні була обрана платформа Blynk.

Blynk — добре продуманий конструктор інтерфейсів.

Він доступний для платформ iOS і Android. Blynk розроблено для Інтернету речей. Він може дистанційно керувати обладнанням, може відображати дані датчиків, зберігати дані, візуалізувати дані та робити багато інших цікавих речей.

Платформа має три основні компоненти:

Додаток Blynk – дозволяє створювати приголомшливі інтерфейси для ваших проектів за допомогою різноманітних віджетів, які ми надаємо.

Blynk Server відповідає за весь зв’язок між смартфоном і обладнанням. Ви можете використовувати нашу хмару Blynk Cloud або запустити власний приватний сервер Blynk локально. Він має відкритий код, може легко працювати з тисячами пристроїв і навіть працює на Arduino UNO.

Бібліотека Blynk — доступна для всіх популярних апаратних платформ, забезпечує зв’язок із сервером і обробляє всі вхідні та вихідні команди.

Спочатку нам потрібно завантажити додаток Blynk для Android. Далі даємо назву нашому проекту та вибираємо плату (пристрій), Arduino Uno.

Ми обираємо Bluetooth як тип підключення, оскільки хочемо керувати ним через послідовний зв’язок. У типі підключення Bluetooth додаткове обладнання до смартфона не потрібно. У вікні віджетів багато контролерів. Тому, натиснувши цю кнопку, вікно системи буде розміщено на екрані проекту. Тепер нам потрібно завантажити бібліотеку Blynk для Arduino IDE, помістити бібліотеку в папку Arduino і відкрити Arduino IDE.

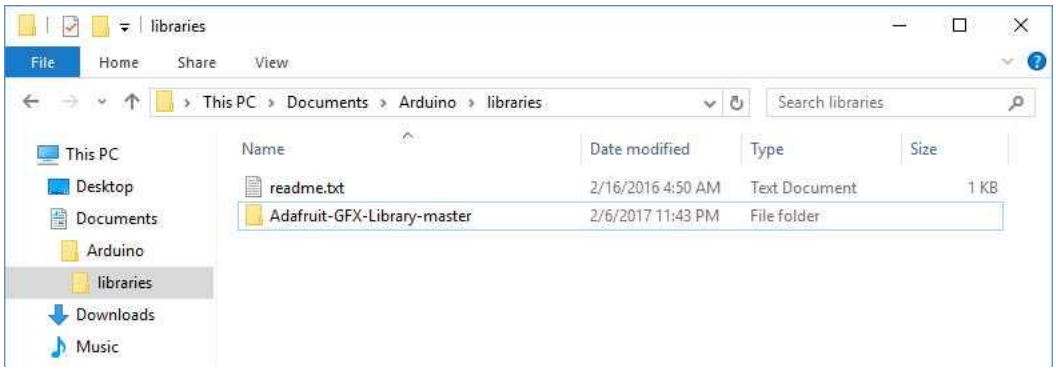


Рисунок 4.3 – Бібліотека Blynk

Бібліотеки часто поширюються у вигляді ZIP-файлу або теки. Назва теки - це назва бібліотеки. Усередині теки буде файл .cpp, файл .h і часто файл keywords.txt, тека з прикладами та інші файли, необхідні для роботи бібліотеки. Починаючи з версії 1.0.5, ви можете встановлювати в IDE бібліотеки сторонніх виробників. Не розпаковуйте завантажену бібліотеку, залиште її як є.

В Arduino IDE перейдіть до Sketch > Include Library > Add .ZIP Library. У верхній частині випадаючого списку виберіть опцію "Add .ZIP Library".

Вам буде запропоновано вибрати бібліотеку, яку ви хочете додати. Перейдіть до місця розташування .zip-файлу і відкрийте його.

Поверніться до меню Ескіз > Додати бібліотеку. Тепер ви маєте побачити бібліотеку внизу спадного меню. Вона готова до використання у вашому ескізі. ZIP-файл буде розпаковано у папку libraries у вашому каталозі скетчів Arduino.

Після того як було підключено всі датчики до плати Arduino UNO, підключено саму плату до смартфона, проведено налаштування та підключення всіх бібліотек реалізуємо код роботи системи, що наведений в додатках.

За допомогою програмного інтерфейсу застосунку додаємо спідометр, екран моніторингу швидкості та вікно попередження о дорожніх знаках . І в результаті отримаємо програмний інтерфейс з виводом критичних параметрів на смартфон. Реалізація програмного інтерфейсу наведена на рисунку 4..

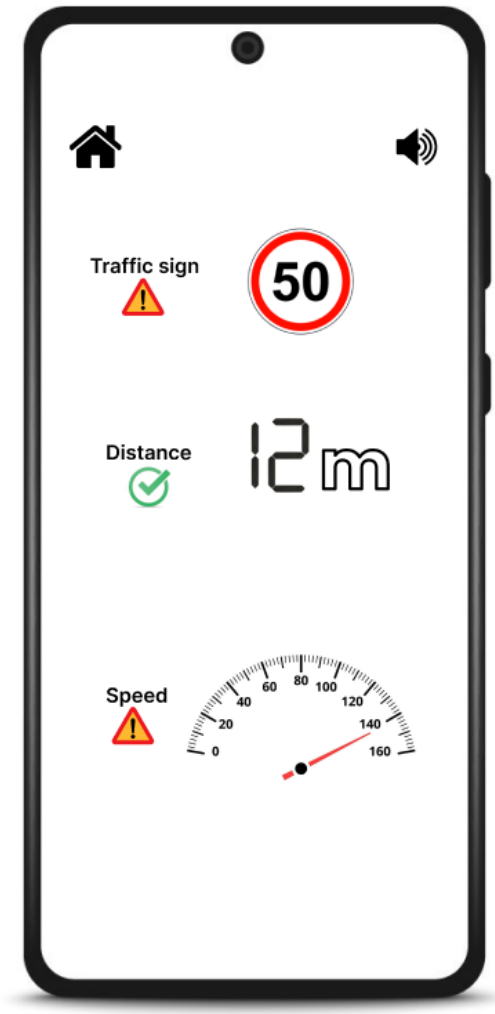


Рисунок 4.5 – Робота додатку

Висновки до розділу 4

У цьому розділі описано проектування програмної частини ситмеми виведення критичної інформаціх на смартфон. Для реалізації програмного коду АПК вибрано та обґрунтовано мову програмування C++ та середовище розробки Arduino IDE.

Розроблено алгоритм роботи додатку, який складається з наступних етапів: збирання даних, обробка, класифікація, розрахунки та виведення критичних параметрів.

Використано ряд бібліотек, які надають функції для роботи з Lidar датчиками, камерою смартфона, Bluetooth та GPS модулем. Описано роботу

та особливості системи допомоги водієві, яка дозволяє збирати, аналізувати та візуалізувати дані про поїздку.

Проведено калібрування датчиків GPS та Lidar, що дозволило підвищити точність вимірювань. Завантажено та налагоджено програмний код на додатку, використовуючи bluetooth та середовище розробки Arduino IDE.

Проведено тестування та відлагодження програмного коду системи, використовуючи смартфон та імітацію дорожньої обстановки. Таким чином, створено програмну частину системи, яка забезпечує безпечність поїздки водія та відповідає запланованим вимогам.

ВИСНОВКИ

В результаті роботи було створено систему збору даних водія з виводом критичних параметрів на смартфон, що дає водіями безпечніше керувати своїм транспортним засобом.

Було проведено аналіз та дослідження сучасних технологій та методів у сфері апаратно-програмних модулів для систем допомоги водієві.

Система, заснована та реалізована на алгоритмах аналізу та методах збору критичної інформації, надає глибше розуміння можливостей та обмежень таких технологій у вирішенні завдань систем контролю дорожнього руху.

Поставлені до роботи завдання було виконано – проведено огляд сучасних методів збору інформації в автомобільній сфері, проаналізовано переваги та недоліки існуючих систем. Було розроблено проєкт мобільного застосунку для водіїв, що надає можливість частково автоматизувати процес керування та продемонстровано практичну здійсненність та перспективність використання мобільних застосунків в сфері допомоги водіям.

Практичне значення роботи полягає в потенціалі значного зменшення дорожньо-транспортних пригод і надає розвиток інформаційним технологіям.

В подальшому слід очікувати, що подібний підхід до часткової автоматизації керування автомобілем буде використовуватись в більшості автомобілів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Azure AI Vision : вебсайт. URL: https://azure.microsoft.com/en-us/products/ai-services/ai-vision (дата звернення: 12.12.2023).
2. EasyWay : вебсайт. URL: https://www.eway.in.ua/ua/cities/mykolaiv (дата звернення: 12.12.2023).
3. Amazon Rekognitio : вебсайт. URL: https://aws.amazon.com/rekognition/?nc1=h_ls (дата звернення: 12.12.2023).
4. Алгоритм Віоли-Джонса : вебсайт. URL: https://towardsdatascience.com/understanding-face-detection-with-the-viola-jones-object-detection-framework-c55cc2a9da14 (дата звернення 27.01.2024).
5. Model Builder : вебсайт. URL: http://surl.li/pvrnu (дата звернення: 27.01.2024).
6. WebStorm: вебсайт. URL: https://www.jetbrains.com/webstorm/ (дата звернення 27.01.2025).
7. VueJS : вебсайт. URL: https://stfalcon.com/ru/blog/post/vue-js-guide-to-tech (дата звернення 27.01.2024).
8. Vue-Router : вебсайт. URL: https://rukovodstvo.net/posts/id_1178/ (дата звернення 27.01.2024).
9. Vuetify : вебсайт. URL: https://vuetifyjs.com/en/ (дата звернення 27.01.2024).
10. Firebase : вебсайт. URL: https://firebase.google.com/ (дата звернення 27.01.2024).
11. PyCharm : вебсайт. URL: https://www.jetbrains.com/pycharm/ (дата звернення 27.01.2024).
12. Python : вебсайт. URL: https://www.jetbrains.com/pycharm/ (дата звернення 27.01.2024).
13. Чому розумні міста прискорять перехід систем відеоспостереження у хмару? *Worldvision* : інтернет-магазин систем безпеки. Опубл. 28.01.2022. URL: https://worldvision.com.ua/chomu-rozumni-mista-

priskoryat-perekhid-sistem-videosposterezhennya-u-khmaru/ (дата звернення: 27.01.2024).

14. Мілякіна Т., Самойленко Г. «Безпечне місто». У Миколаєві впроваджують нові етапи проєкту. *Суспільне. Новини* : інтернет-видання. Опубл. 17.10.2023. URL: https://suspilne.media/596061-bezpecne-misto-u-mikolaevi-vprovadzuut-novi-etapi-proektu/ (дата звернення: 27.01.2024).
15. Standards by ISO/TC 268/SC 1. Smart community infrastructures. ISO: International Organization forStandardization. URL: https://www.iso.org/committee/656967/x/catalogue/ (Last accessed: 27.01.2024).
16. ITU-T Recommendations by series. Y series: Global information infrastructure, Internet protocol aspects,next-generation networks, Internet of Things and smart cities. ITU: Committed to connecting the world. URL: https://www.itu.int/itu-t/recommendations/index.aspx?ser=Y (Last accessed: 27.01.2024).

ДОДАТОК А
ДОВІДКА

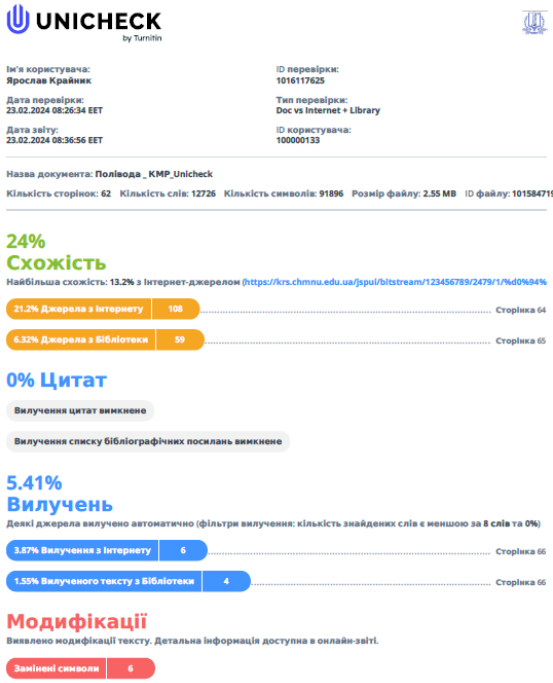
про перевірку на унікальність

пояснювальної записки кваліфікаційної магістерської роботи

на тему: « Система збору інформації для водія з виведенням критичних параметрів»

студента спеціальності 123 «Комп’ютерна інженерія», групи 605м

Поліводи Данила Володимировича



Перевірку тексту здійснено сервісом: онлайн-сервіс Unicheck

Результат перевірки тексту кваліфікаційної роботи: схожість складає 24%

Студент:	Керівник:
	канд. техн. наук, доцент
_____ Д. В. Поліводa	_____ Я. М. Крайник
підпис ініціали, прізвище	підпис ініціали, прізвище

Дата: «____» _____ 2024 р.

ДОДАТОК Б

```
#include <iarduino_GPS_ATGM336.h>
#include <SoftwareSerial.h>
#include <opencv2/opencv.hpp>


int del = 5; int cm1 = 30; int cm2 = 20; int cm3 = 10; int cm4 = 5;

void setup() { Serial.begin(9600); pinMode(7, OUTPUT); pinMode(2,
OUTPUT); pinMode(3, OUTPUT); pinMode(4, OUTPUT);
}


void loop() {
int distance, sum, total;
for (byte i = 0; i <= 10; i++) { distance = ultrasonic.Ranging(CM);
sum = sum + distance;
delay(del);
}
total = sum / 10;
Serial.println("Distance - " + String(total));


if (total >= cm1) {
digitalWrite(4, LOW); digitalWrite(3, LOW); digitalWrite(2, LOW);
}
if (total < cm1 && total >= cm2) {
digitalWrite(4,LOW); digitalWrite(3,LOW); digitalWrite(2,HIGH);
}
if (total < cm2 && total > cm3) {
digitalWrite(4,LOW); digitalWrite(3,HIGH); digitalWrite(2,LOW);
}

if (total <= cm3) {
digitalWrite(4,HIGH); digitalWrite(3,LOW); digitalWrite(2,LOW);
}


if (total > cm4) { noTone(7); } if (total <= cm4) { tone(7,100); }
}


const int stepsPerRevolution = 200;
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11); volatile byte
REV;
unsigned long int rpm,RPM; unsigned long st=0; unsigned long time;
int ledPin = 13;
int led = 0,RPMlen , prevRPM; int flag = 0;
int flag1=1;
#define bladesInFan 2
float radius=4.7;     // в дюймах int preSteps=0;
float stepAngle= 360.0/(float)stepsPerRevolution; float minSpeed=0;
float maxSpeed=280.0; float minSteps=0;
float maxSteps=maxSpeed/stepAngle; void setup()
{
myStepper.setSpeed(60); Serial.begin(9600); pinMode(ledPin, OUTPUT);
lcd.begin(16,2); lcd.print("Speedometer"); delay(2000);
attachInterrupt(0, RPMCount, RISING);
}
```

```
void loop()
{
readRPM();
radius=((radius * 2.54)/100.0); // преобразуем в метры
int Speed= ((float)RPM * 60.0 * (2.0 * 3.14 * radius)/1000.0); int
Steps=map(Speed, minSpeed,maxSpeed,minSteps,maxSteps);


if(flag1)
{
Serial.print(Speed); Serial.println("Kmh");
lcd.setCursor(0,0); lcd.print("RPM: "); lcd.print(RPM); lcd.print("
    "); lcd.setCursor(0,1); lcd.print("Speed: "); lcd.print(Speed);
lcd.print(" Km/h            ");
flag1=0;
}
int currSteps=Steps;
int steps= currSteps-preSteps; preSteps=currSteps;
myStepper.step(steps);
}
int readRPM()
{
if(REV >= 10 or millis()>=st+1000)
{
if(flag==0) flag=1;
rpm = (60/2)*(1000/(millis() - time))*REV/bladesInFan; time =
millis();
REV = 0;
int x= rpm; while(x!=0)
{
x = x/10;
RPMlen++;
}
Serial.println(rpm,DEC); RPM=rpm;
delay(500); st=millis(); flag1=1;
}
}


void RPMCount()
{
REV++;
if (led == LOW)
{
led = HIGH;
}
else
{
led = LOW;
}
digitalWrite(ledPin, led);
}


void adbEventHandler(Connection * connection, adb_eventType event,
uint16_t length, uint8_t * data)
{
if (event == ADB_CONNECTION_RECEIVE)
{
Serial.print("data:"); Serial.println(data[0],DEC);
if((data[0]) == COMMAND_SEND_TRUE) SendToAndroid = true;
else if ((data[0]) == COMMAND_SEND_FALSE) SendToAndroid = false; else
```

```
if ((data[0]) == COMMAND_PLAY_BEEP) playBeep();
}



open");

else if (event == ADB_CONNECTION_OPEN) Serial.println("ADB connection


else if (event == ADB_CONNECTION_CLOSE) Serial.println("ADB
connection


close");
else {
Serial.println(event);
}
}


void playBeep() {
for (int j = 0; j < 10; j++) { analogWrite(BeeperPin, 20); delay(50);
analogWrite(BeeperPin, 0); delay(150);
}
}
```