ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «БІОМЕТРИЧНА СИСТЕМА АВТОРИЗАЦІЇ ВОДІЯ З ВИКОРИСТАННЯМ RASPBERRY PI»

на здобуття освітнього ступеня магістра
зі спеціальності 126 Інформаційні системи і технології
освітньо-професійної програми Інформаційні системи та технології

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ *Олександр БОВКУН*

Виконав:
здобувач вищої освіти
група ІСДМ-61 — Олександр БОВКУН

Керівник:
науковий ступінь, вчене звання — Оксана ТКАЛЕНКО, к.т.н., доцент

Рецензент:
науковий ступінь, вчене звання — _____
Ім'я, ПРІЗВИЩЕ

Київ – 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра Інформаційних систем та технологій

Ступінь вищої освіти Магістр

Спеціальність 126 Інформаційні системи і технології

Освітньо-професійна програма Інформаційні системи та технології

ЗАТВЕРДЖУЮ
Завідувач кафедрою ІСТ

_____Каміла СТОРЧАК
«____» _____20___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ СТУДЕНТУ
Бовкуну Олександру Сергійовичу
(прізвище, ім'я, по батькові здобувача)

1.Тема кваліфікаційної роботи: «Біометрична система авторизації водія з використанням Raspberry Pi»

керівник кваліфікаційної роботи Оксана ТКАЛЕНКО, к.т.н., доцент
(ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від «30» жовтня 2025 року № 467.

2. Строк подання кваліфікаційної роботи: 26 грудня 2025 року.

3. Вихідні дані до кваліфікаційної роботи: Власна база еталонних зображень;
Значення порогів довіри (High=90%, Low=70%) та коефіцієнт згладжування (0.3);
Еталонні зображення для ініціалізації системи (навчання);
Потік відеоданих у реальному часі для тестування;
Науково-технічна література з питань, пов'язаних з технологіями комп'ютерного зору (Computer Vision), біометричною аутентифікацією.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Аналіз розвитку та сучасного стану систем захисту автомобілів (механічні, електронні, біометричні).
2. Обґрунтування вибору апаратної платформи (Raspberry Pi) та розробка архітектури системи.
3. Розробка програмного забезпечення для біометричної авторизації та керування периферією.
4. Тестування та аналіз результатів роботи системи в різних умовах експлуатації.

5. Перелік ілюстративного матеріалу: *презентація*
   1. Актуальність.
   2. Аналіз систем захисту.
   3. Архітектура системи біометричної авторизації.
   4. Апаратна реалізація.
   5. Принципова схема.
   6. Алгоритм роботи.
   7. Результати тестування.

6. Дата видачі завдання: 30 жовтня 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	30.10 – 05.11.25	
2	Вибір компонентів для пристрою	06.11 – 14.11.25	
3	Інжерний етап створення схеми підключення	15.11 – 20.11.25	
4	Написання коду для розпізнавання	21.11 – 28.11.25	
5	Моделювання роботи системи та програмна реалізація інтерфейсу користувача	29.11 – 05.12.25	
6	Тестування програмних алгоритмів та аналіз ефективності роботи системи	06.12 – 12.12.25	
7	Оформлення роботи: вступ, висновки, реферат	13.12 – 17.12.25	
8	Розробка демонстраційних матеріалів	18.12 – 23.12.25	

Здобувач вищої освіти _____ Олександр БОВКУН
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник роботи
кваліфікаційної роботи _____ Оксана ТКАЛЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 75 стор., 17 рис., 30 джерел.

Мета роботи - розробка та програмна реалізація автономної біометричної системи авторизації водія на базі платформи Raspberry Pi для підвищення рівня безпеки транспортних засобів.

Об'єкт дослідження – процеси захисту транспортних засобів від несанкціонованого доступу та викрадення із застосуванням технологій комп'ютерного зору.

Предмет дослідження – алгоритми, методи та програмно-апаратні засоби створення системи біометричної ідентифікації водія на основі згорткових нейронних мереж.

Короткий зміст роботи: Досліджені методи та технології захисту автомобілів, проведено порівняльний аналіз механічних, електронних та біометричних систем. Обґрунтовано вибір апаратної платформи Raspberry Pi та концепції Edge Computing для забезпечення автономності пристрою. Розроблено принципову електричну схему з використанням гальванічної розв'язки та релейної комутації системи запалювання. Створено програмне забезпечення мовою Python з використанням бібліотек OpenCV та face_recognition для детекції та розпізнавання облич у реальному часі. Здійснено вибір та налаштування порогів довіри, виконане тестування прототипу в умовах зміни освітлення та ракурсу для оцінки ефективності та точності алгоритмів розпізнавання.

КЛЮЧОВІ СЛОВА: БІОМЕТРИЧНА АВТОРИЗАЦІЯ, RASPBERRY PI, КОМП'ЮТЕРНИЙ ЗІР, PYTHON, OPENCV, FACE RECOGNITION, БЕЗПЕКА АВТОМОБІЛЯ, EDGE COMPUTING, НЕЙРОННА МЕРЕЖА.

ABSTRACT

Text part of the master`s qualification work: 75 pages, 17 pictures, 30 sources.

The purpose of the work is to develop and implement an autonomous biometric driver authorization system based on the Raspberry Pi platform to improve vehicle safety.

Object of research is the processes of protecting vehicles from unauthorized access and theft using computer vision technologies.

Subject of research is algorithms, methods, and software and hardware tools for creating a driver biometric identification system based on convolutional neural networks.

Summary of the work: We looked into ways to protect cars and compared mechanical, electronic, and biometric systems. We picked the Raspberry Pi hardware platform and the Edge Computing concept to make sure the device works on its own. We came up with a basic electrical diagram using galvanic isolation and relay switching for the ignition system. Software was created in Python using the OpenCV and face_recognition libraries for real-time face detection and recognition. Confidence thresholds were selected and configured, and the prototype was tested under changing lighting and angle conditions to evaluate the effectiveness and accuracy of the recognition algorithms.

KEYWORDS: BIOMETRIC AUTHORIZATION, RASPBERRY PI, COMPUTER VISION, PYTHON, OPENCV, FACE RECOGNITION, CAR SAFETY, EDGE COMPUTING, NEURAL NETWORK.

ЗМІСТ

ВСТУП

Найважливішим завданням у сучасній автомобільній індустрії є забезпечення надійного захисту транспортних засобів від викрадення. Зростання вартості автомобілів та стрімкий розвиток технологій злому призвели до того, що традиційні механічні замки та штатні електронні системи вже не гарантують повної безпеки, створюючи ризик втрати майна для власників[2, 26].

Актуальність теми визначається необхідністю переходу від застарілих методів захисту, що ідентифікують лише наявність ключа, до новітніх систем, здатних розпізнавати безпосередньо особистість водія. Використання концепції периферійних обчислень (Edge Computing) дозволяє створити автономну систему, яка не залежить від стабільності інтернету та хмарних сервісів, що є критичним для надійної експлуатації автомобіля в будь-яких умовах[24,26].

Метою цього дослідження є розробка інноваційної системи авторизації водія на базі одноплатного комп'ютера Raspberry Pi, що володітиме інтелектуальними функціями комп'ютерного зору для блокування несанкціонованого запуску двигуна. Цей пристрій можна уявити як «розумного охоронця», який, будучи інтегрованим у бортову мережу, знає власника в обличчя і дозволяє керування лише авторизованим особам.

Для досягнення мети дослідження були визначені завдання:

1. Аналіз існуючих систем захисту та методів їх обходу, що підтвердив вразливість механічних та електронних засобів і доцільність впровадження біометрії[27].
2. Обґрунтування вибору апаратної платформи Raspberry Pi та периферійних компонентів (камери, реле, стабілізаторів), які забезпечують достатню обчислювальну потужність для роботи нейромереж у реальному часі.
3. Розробка функціональної та принципової схеми пристрою, яка включає гальванічну розв'язку для захисту електроніки та силовий блок для керування запалюванням.

4. Створення програмного забезпечення мовою Python із використанням бібліотек комп’ютерного зору (OpenCV, face_recognition) для детекції обличчя, порівняння з еталоном та прийняття рішення про допуск.
5. Тестування прототипу для перевірки точності розпізнавання в різних умовах та налаштування адаптивних порогів довіри.

Об'єктом дослідження є процес захисту транспортних засобів від несанкціонованого доступу. Цей процес включає моніторинг спроб запуску двигуна та верифікацію прав користувача на керування автомобілем. Контроль доступу забезпечує збереження транспортного засобу, запобігаючи викраденню шляхом розриву електричних кіл запалювання. Оцінка надійності захисту базується на здатності системи протистояти сучасним методам електронного злому та безпомилково ідентифікувати власника.

Предметом дослідження є програмно-апаратний комплекс біометричної авторизації водія. Він обладнаний камерою, мікрокомп’ютером Raspberry Pi, системою релейної комутації та програмними алгоритмами на базі нейронних мереж. Пристрій призначений для збору візуальних даних, їх локальної обробки та порівняння з базою довірених осіб без передачі даних у зовнішні мережі. Зібрані дані використовуються для формування керуючого сигналу на дозвіл або заборону роботи двигуна, а також для ведення логів подій. Пристрій підвищує надійність захисту автомобіля, забезпечує зручність безключового доступу та унеможливлює використання авто сторонніми особами навіть за наявності викраденого ключа.

Методом дослідження є створення симуляції електричної схеми у середовищі Fritzing для візуалізації принципів роботи та написання коду для розуміння роботи пристрою для перевірки ефективності алгоритмів розпізнавання в умовах, наближених до експлуатаційних[20, 22].

Апробація результатів магістерської роботи:

1. Бовкун О. С. «Біометрична система авторизації водія з використанням Raspberry Pi». Тези доповіді на III Всеукраїнській науково-технічній конференції «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і світу». – Київ, 18 листопада 2025 року.

2. Бовкун О. С. «Розробка автономного модуля біометричної ідентифікації для системи запуску автомобіля». Тези доповіді на VIII Всеукраїнська науково-технічна конференція «Компьютерні технології: інновації, проблеми, рішення». Житомир – 02-03 грудня 2025р.

1 АНАЛІЗ РОЗВИТКУ ТА СУЧАСНОГО СТАНУ СИСТЕМ ЗАХИСТУ АВТОМОБІЛІВ

1.1 Загальні принципи та класифікація систем захисту

Питання безпеки автомобіля – це не просто окрема функція у великому списку технічних рішення. Воно завжди було однією з тих тем, що виникали одразу після того, як машина перестала бути дивиною і стала доступною широкому колу людей. З появою масового автотранспорту з'ясувалося, що турбота про справність двигуна чи гальм – лише половина справи[9]. Інша половина стосувалася того, як не дозволити сторонній людині керувати автомобілем або завдати йому шкоди. Через це поступово виросла ціла галузь технологій захисту, яка з роками все більше ускладнювалася. Автомобілі дорожчали, їх електронні системи ставали насиченішими, а разом із цим розширювалися методи викрадення. Тому виробники почали комбінувати різні засоби – від простих механічних пристроїв до повноцінних електронних комплексів, що працювали одночасно на кількох рівнях. У результаті з’явилася багатошарова система захисту, яка охоплює і фізичні бар’єри, і цифрові алгоритми, і мережеві компоненти[2, 28].

Автомобільні системи безпеки не однакові за принципом дії. Одні елементи блокують фізичний доступ або не дають провернути ключ запалювання, інші розпізнають спроби втручання або самостійно відключають важливі вузли. На зміну «чистій механіці» прийшли електронні модулі, де кожний датчик, мікроконтролер або радіоканал виконує власну роль. І поруч із цими двома великими групами з’явився третій напрям – біометрія. Технології, що раніше асоціювалися з охороною будівель або мобільними пристроями, потроху стали застосовуватися і в автоіндустрії[27].

Історія розвитку таких систем вийшла доволі послідовною. Спочатку намагалися просто не допустити сторонню людину до керма: блокатори, важкі замки, металеві конструкції на педалі чи коробку передач. Згодом увага змістилася

на електроніку, з’явилися іммобілайзери, які вже не потрібно було бачитись фізично; вони діяли «всередині» автомобіля. Далі інтеграція у CAN-шину та складні системи, що контролюють і аналізують набагато більше сигналів, ніж будь-які механічні засоби могли забезпечити. Попри різницю у технічній реалізації, загальна методика роботи таких систем тримається на кількох ключових ідеях. Насамперед це перевірка, хто намагається скористатися автомобілем – автентифікація. Після цього визначення, чи має ця людина право виконувати конкретну дію, наприклад, запустити двигун, а далі фіксація усіх подій, щоб власник мав змогу відстежити підозрілу активність. Ці три принципи поєднуються у різних пропорціях залежно від виробника та рівня захисту, але суть лишається спільною: багаторівневість.

Розвиток у цій сфері не зупинявся, тому що і методи злому розвиваються так само швидко. Те, що працювало надійно десять чи навіть п’ять років тому, сьогодні може виявитися недостатнім через появу електронних ретрансляторів, приладів для перехоплення сигналу, програмного інструментарію, здатного підбирати коди або емулювати ключі. Саме тому виробники змушені впроваджувати нові технології, зокрема біометрію, системи машинного навчання та хмарні сервіси, які дозволяють аналізувати поведінкові патерни та створювати ще складніший бар’єр для сторонніх. Такий контекст допомагає зрозуміти, чому має сенс детально розглянути кожну групу захисних рішень, починаючи з найдавніших механічних засобів і закінчуючи сучасними біометричними системами, що стали для автомобільної галузі не просто новим етапом, а фактично окремим напрямом розвитку.

Коли починаєш порівнювати різні засоби захисту автомобіля, досить швидко з’являється відчуття, що всі вони ніби належать до однієї сфери, але поводяться зовсім по-різному. Одні працюють виключно на фізичному рівні – важкі металеві конструкції, які просто не дають рухати кермом або натискати педалі. Інші, більш сучасні, працюють майже непомітно: вони не заважають людині фізично, але контролюють процес запуску, передають сигнали в електронні блоки, забороняють подачу палива або переривають запалювання. І між цими двома полюсами виник

окремий напрям – засоби, які аналізують саму людину, що намагається сісти за кермо.

Для зручності всі системи поділяють за способом, яким вони взагалі взаємодіють із автомобілем. Це може бути пряме механічне блокування або контроль через електроніку, а може бути система, що працює з тим, хто намагається отримати доступ до авто. Такий поділ не випадковий, адже кожен тип засобів має свої сильні й слабкі сторони. Механічні пристрої можна побачити неозброєним оком, і вони часто відлякують самим фактом своєї присутності. Електронні системи, навпаки, стають невидимими та працюють у глибині автомобіля, де зловмисник не завжди може швидко зрозуміти, що саме відключає запуск.

Окремо стоять біометричні рішення. Їхня поява стала можливою тільки після того, як комп’ютери та вбудовані процесори навчились обробляти зображення в режимі реального часу. Якщо раніше будь-яка система безпеки орієнтувалася на ключ або брелок, то тепер центр уваги – сама людина. Кожен водій має свої характерні риси, які не можна легко підробити, і саме на цьому ґрунтуються біометричні технології. Вони не обмежуються тільки обличчям. Це можуть бути відбитки пальців, особливості голосу, або специфічні рухові патерни. Автомобіль отримує можливість «пізнавати» свого власника так само природно, як це робить смартфон.

Попри очевидні відмінності, всі ці підходи поступово переплітаються в сучасних транспортних засобах. Механічні блокатори нікуди не зникли, але вони дедалі частіше стають лише додатковим рівнем. Електронні системи, навпаки, розширюються: з’явились іммобілайзери нового покоління, системи безключового доступу та різні варіанти шифрування сигналів. А біометрія вже не здається екзотикою, оскільки працює швидше і точніше, ніж багато традиційних методів.

Розвиток автомобільної безпеки можна уявити як довгу історію експериментів, спроб і відповідей на все нові виклики. Спершу в автомобілях не було нічого складнішого за залізні замки, які блокували кермо або коробку передач. Це були прості механізми: нічого не обчислювали, нічого не аналізували – просто

не дозволяли покрутити те, що треба крутити, аби поїхати. І такий підхід довго працював, адже тодішні методи крадіжок теж були доволі прямолінійними.

Згодом ситуація змінилась. Автомобілі стали дорожчими, в їх електриці з’явилось більше вузлів, і разом із цим зросла кількість способів, якими можна було їх обійти. Паралельно з розвитком транспорту розвивалися і способи злому, тому виробникам довелося переходити від механічних рішень до електронних. Так з’явилися іммобілайзери, які вже не блокували щось фізично, а розривали логіку запуску: двигун міг бути справним, пальне залите в бак але без дозволу від електронного блоку він просто не оживав.

Поступова цифровізація автомобіля привела до появи зовсім іншого класу систем, які працюють не з деталями, а з інформацією. Керування через CAN-шину, шифровані сигнали, перевірка автентичності ключа – усе це стало звичним. Тепер уже недостатньо просто мати відмичку або кусачки, бо електроніка сама вирішує, чи дозволяти запуск. І саме на цьому рівні почала з’являтись ідея, що можна розпізнавати не ключ, а людину, яка ним користується.

Логіка такої еволюції проста: що складніші методи викрадення, то більше шарів захисту потрібно. Автомобільні системи дедалі частіше комбінують механічні, електронні та цифрові елементи в єдиний комплекс. Навіть якщо один рівень і вдасться обійти, інші продовжать виконувати свою роботу. Саме так і формується багаторівнева структура безпеки, яка необхідна сучасному транспорту.

Ще одна причина постійних змін – поява пристроїв, що дозволяють перехоплювати або підміняти сигнали безконтактних ключів. За останні роки це стало однією з найактуальніших проблем. Сканери, ретранслятори, дублікатори – усе це підштовхнуло виробників до пошуку рішень, які не можна підробити простим радіоперехопленням. У цю нішу чудово лягли біометричні технології, бо вони не покладаються на сигнал чи код – вони працюють з рисами, що притаманні лише людині.

Сьогодні системи безпеки перетворилися на рухому, динамічну галузь, яка змінюється разом із методами, що спрямовані на її подолання. У цьому процесі немає остаточної точки, бо кожне нове покоління автомобілів приносить нові

можливості для захисту, але й нові ризики. Тому логічно розглядати всі сучасні засоби в нерозривному зв’язку між собою та в історичному контексті – від перших металевих блокаторів до інтелектуальних систем, які вміють аналізувати поведінку та зовнішність водія.

1.1.1 Механічні системи захисту

Механічні системи захисту є найдавнішим і водночас найочевиднішим способом запобігання несанкціонованому доступу до транспортних засобів[26]. Їхня історія бере свій початок ще з моменту появи перших автомобілів на початку XX століття, коли основним завданням було забезпечити фізичне блокування механізмів, що дозволяють запустити двигун або керувати транспортним засобом. Перші автомобілі практично не мали жодних засобів охорони – запалювання здійснювалося механічним способом, а будь-хто, хто мав доступ до автомобіля, міг його запустити. З розвитком автомобільної промисловості та збільшенням кількості транспортних засобів на дорогах постало питання безпеки та захисту від крадіжок.

Перші механічні системи охорони ґрунтувалися на простих замках запалювання. Водій мав спеціальний ключ, який механічно розмикав контакт у ланцюзі живлення системи запалювання. Ця проста, але ефективна конструкція стала основою для подальших поколінь систем безпеки. Уже в 1930-х роках більшість автомобілів оснащувалися стандартними замками запалювання, що унеможливлювали запуск двигуна без ключа. Згодом з’явилися більш складні механічні блокувальні пристрої, спрямовані не лише на запобігання запуску двигуна, а й на унеможливлення руху автомобіля. До таких систем належали механічні блокатори керма, коробки передач, педалей або коліс.

Ці пристрої стали популярними в середині XX століття, коли рівень автомобільної злочинності почав швидко зростати. Блокатори керма, наприклад, мали вигляд металевого стрижня або фіксатора, який з’єднував кермо з педаллю або фіксував його в нерухомому положенні, що робило неможливим поворот коліс.

Поступово механічні системи ставали більш складними та надійними. З’явилися пристрої, які вимагали не лише фізичного ключа, а й певних дій чи комбінацій для зняття блокування. Зокрема, були розроблені замки коробки передач, які фізично фіксували важіль перемикання в одному положенні (зазвичай у нейтральному), а також блокатори ручного гальма.

Такі системи використовувалися як додатковий захист у поєднанні із замком запалювання. У 1970–1990-х роках механічні засоби захисту отримали новий поштовх розвитку завдяки появі портативних знімних пристроїв, які власник міг встановити самостійно. Найпоширенішими стали металеві блокатори керма типу “клюка” або “бита”, які швидко фіксували руль у певному положенні. Ці пристрої отримали широку популярність через простоту використання та низьку вартість, хоча ефективність їх поступово знижувалася з розвитком інструментів злому.

Незважаючи на розвиток електронних систем охорони, механічні пристрої залишаються важливою складовою автомобільної безпеки. Їхня основна перевага полягає у простоті, незалежності від джерела живлення та високій стійкості до електронних методів злому. Навіть сучасні автомобілі з біометричними або безключовими системами запалювання часто мають механічний дублюючий замок запалювання або систему аварійного блокування рульового колеса, що гарантує безпеку у випадку відмови електроніки. Механічні системи захисту стали фундаментом для розвитку всіх подальших поколінь систем безпеки. Саме вони створили базову концепцію “авторизації” користувача через фізичний ключ, яку в майбутньому замінили електронні ключі, RFID-мітки та, зрештою, біометричні дані.

Технічні принципи, на яких ґрунтуються механічні системи захисту, визначаються простотою їхньої функції: створити фізичну перешкоду на шляху до виконання критичної дії – керування транспортним засобом або запуску двигуна. У центрі цієї ідеї лежить механічний переривник або фіксуючий елемент, що змінює геометрію або кінематику вузлів керування так, щоб без спеціального інструмента чи ключа виконати необхідну дію було неможливо. На рівні конструкції це може бути реалізовано як жорстке з’єднання, що блокує обертання або переміщення; як

шпіндель або штифт, що фіксує важіль у визначеному положенні; або як захисний корпус, що закриває доступ до органів керування. Усі ці рішення мають однакову мету – збільшити час і складність несанкціонованої операції настільки, щоб вона перестала бути вигідною для зловмисника. Матеріали та конструкційні рішення значною мірою визначають довготривалість та надійність механічних блокувань. Традиційно застосовують загартовану сталь та сплави з високою твердістю, іноді з додатковим використанням хромування чи інших покриттів для захисту від корозії та механічного зношення. У деяких пристроях застосовуються складні складні геометрії зубчастих зчеплень або багатошарові замкові механізми, що ускладнюють доступ, а також використання протиковзних поверхонь для ускладнення застосування інструментів. Конструкційні особливості, такі як приховане кріплення, особливі форми ключа чи нестандартні способи фіксації, збільшують рівень стійкості конструкції до типових методів впливу: висвердлювання, розрізання, вигинання або руйнування штифтів.

Аналіз типових способів злому дає чітке уявлення про межі ефективності механічних рішень. Теоретично найпростіші методи – фізичне зняття або зрізання замка, використання домкрата для від’єднання керма чи застосування важелів для примусового обертання – залишаються доступними навіть у випадку застосування високотвердої сталі. Однак на практиці саме трудомісткість операції, потреба у спеціальному обладнанні та час на виконання стають основними чинниками, що зменшують імовірність успішного злочину. Сучасні розробники застосовують концепцію багатоетапного захисту, коли для обходу потрібно подолати низку різнорідних перешкод, кожна з яких вимагає різних інструментів та навичок, що значно знижує оперативну привабливість викрадення. Історичні приклади конструкцій демонструють різні підходи до вирішення єдиної проблеми. Одні виробники надавали перевагу масивності – великі однорідні блокатори зводилися до мінімальної кількості рухомих частин, що зменшувало вразливість до механічного руйнування; інші шукали шляхів ускладнення замкового механізму, використовуючи складні конфігурації ключа та внутрішні штифти, які потребували точного відповідного інструменту для відкриття. Окремі рішення поєднували

блокування рульового вала з одночасним фіксуванням коробки передач або педалей, що унеможливлювало одночасну роботу декількох елементів керування. Такі системи відзначаються високою надійністю, але іноді мають недолік – значну масу та об’єм, що ускладнює їхнє зберігання та використання. У порівнянні з електронними рішеннями механічні блокатори мають як переваги, так і обмеження. Переваги випливають із принципової незалежності від джерела живлення, простоти обслуговування та доступності замінних частин. Механічні конструкції рідко виходять з ладу через електромагнітні перешкоди чи збої програмного забезпечення, їх можна ефективно експлуатаційно відновити при механічних пошкодженнях. Водночас, з розвитком електроніки та появою інструментів діагностики, що дозволяють відтворювати сигнали замків та реле, деякі механічні системи виявилися вразливими до непрямих методів обходу, наприклад, до демонтажу електронних блоків, які відповідають за управління механізмами, або застосування методів, що руйнують пластикові корпуса й отримують доступ до внутрішнього механізму[28].

Практична ефективність механічних систем також багато в чому залежить від їхньої інтеграції в загальну систему безпеки транспортного засобу. Сам по собі блокатор керма або педалей може затримати зловмисника на кілька хвилин, але якщо поруч існує легкодоступний електронний обхід, то час затримки може виявитися недостатнім. Тому в реальних проектах механічні пристрої часто розглядають як компонент багаторівневої стратегії захисту, де їхня роль – створити першу фізичну перешкоду і надати додатковий час на реакцію власника або системи сповіщення.

Крім того, з технічної точки зору важливим фактором є надійність у реальних експлуатаційних умовах. Механічні замки та блокатори зазнають впливу температурних коливань, вібрацій, пилу та корозії, що може з часом призводити до утрудненого використання або поломок. Саме тому сучасні конструкції передбачають застосування матеріалів та покриттів, стійких до агресивного середовища, а також передбачають можливість технічного обслуговування без спеціалізованих інструментів. У великій кількості комерційних та службових авто,

де інтенсивність використання висока, такі вимоги стають визначальними при виборі механічної системи захисту. Сучасний розвиток автомобільних механічних систем захисту демонструє поступовий перехід від простих замкових конструкцій до більш складних комбінованих механізмів, які інтегруються з електронними компонентами автомобіля. Хоча базовий принцип – створення фізичної перешкоди залишається незмінним, сучасні інженерні рішення спрямовані не лише на підвищення міцності конструкції, а й на забезпечення зручності використання, ергономіки та адаптації до нових технологій керування транспортним засобом. Відповідно, у нових моделях автомобілів механічний елемент усе частіше розглядається як допоміжний компонент багаторівневої системи безпеки, який працює у взаємодії з електронікою.

Одним із прикладів такої інтеграції є замки запалювання з електронним контролем, у яких механічна частина все ще присутня, але сам процес авторизації водія контролюється мікропроцесорним модулем. Така конструкція зменшує залежність від механічного ключа, але водночас забезпечує збереження базової фізичної перешкоди у разі спроби злому. Існують також гібридні блокатори рульової колонки, у яких механічна фіксація активується не вручну, а за допомогою електроприводу, що спрацьовує лише при автентифікації користувача або отриманні сигналу від системи доступу. Ці рішення ілюструють перехід до концепції “розумного” механічного захисту, де стале поєднання фізичних і цифрових елементів дає новий рівень безпеки.

У той самий час механічні рішення продовжують активно застосовуватися у сферах, де потрібна максимальна простота та автономність. Так, блокатори коробки передач і педалей залишаються популярними серед власників старіших автомобілів, у яких відсутні інтегровані системи безключового доступу. Їхня ефективність не залежить від стану акумулятора чи роботи електронних модулів, що є важливим чинником у складних кліматичних умовах або під час тривалого простою автомобіля. Це пояснює, чому навіть у нових моделях транспортних засобів виробники залишають певні механічні елементи у якості резервного засобу

безпеки – наприклад, можливість механічного відкриття дверей чи аварійного запуску через прихований замок запалювання.

Попри технологічний прогрес, ефективність механічних засобів безпеки поступово втрачає пріоритет у порівнянні з електронними, оскільки сучасні методи злому стали значно витонченішими. Наявність спеціалізованих інструментів для швидкого відмикання, знання конструкцій типових замків і доступ до креслень популярних моделей зробили можливим обхід багатьох механічних систем за лічені хвилини. Це призвело до того, що провідні виробники почали активно впроваджувати системи подвійного або прихованого блокування, у яких механічна дія дублюється або контролюється електронікою. Таким чином, механічний пристрій уже не розглядається як самодостатній засіб захисту, а стає елементом комбінованої структури, спрямованої на підвищення загальної стійкості системи безпеки до злому.

Іншим важливим напрямом розвитку стало застосування матеріалів нового покоління. Наприклад, використання високоміцних композитів або легких сплавів дозволяє зменшити масу пристрою без втрати жорсткості, що позитивно впливає на ергономіку. Водночас сучасні технології обробки металу, такі як лазерне загартування або багатошарове напилення, забезпечують підвищену стійкість до механічних пошкоджень і корозії. Це не лише продовжує термін служби механічних систем, а й створює додаткові перешкоди для використання грубої сили при спробі злому.

Однак навіть із урахуванням усіх цих удосконалень, ключова проблема механічних систем залишається незмінною – вони реагують лише на фізичну дію, а не на сам факт ідентифікації користувача. Іншими словами, механічна система не може “знати”, хто саме намагається запустити автомобіль – власник чи стороння особа. Ця концептуальна обмеженість і стала основною причиною переходу до електронних і, згодом, біометричних систем[25], які дозволяють поєднати фізичний захист із процесом авторизації. Саме тут і відкривається логічна перспектива для розробки нових підходів, у яких елементи традиційної механіки зберігаються як

структурна основа, але управління ними здійснюється інтелектуальними засобами розпізнавання.

1.1.2 Електронні системи захисту

Електронні системи захисту виникли як природне продовження еволюції, коли можливості механічних блокувань вже не задовольняли вимоги безпеки та зручності експлуатації сучасного автомобіля. Поява масових електронних компонентів у середині XX століття дала змогу не лише посилити контроль доступу, а й зробити цей контроль контекстно-залежним – враховувати часові, просторові та поведінкові ознаки, що супроводжують використання транспортного засобу. Іммобілайзери, датчики, брелоки та перші телематичні модулі надали можливість реалізувати сценарії, які раніше були неможливі: дистанційне відстеження, автоматичне блокування стартера, запис подій у чорний ящик автомобіля. Ці можливості призвели до якісного стрибка – захист перестав бути простим механічним бар'єром і перетворився на складну функціональну підсистему, що взаємодіє з іншими елементами автомобіля.

Ранні електронні рішення базувалися на простій логіці перевірки відповідності кодів: ключ або брелок передавав ідентифікатор, модуль порівнював його з вбудованою базою і давав дозвіл на запуск двигуна. Проте вже перші реалізації показали нові вектори атак – клонування сигналів, ретрансляція та компрометація кодів. Це стимулювало подальший розвиток протоколів захисту і впровадження криптографічних механізмів у автомобільні системи. Одночасно програмованість блоків дозволила вести журнал подій, що кардинально змінило підхід до розслідування спроб злому: тепер можна було відновити хронологію дій, виявити нетипові сценарії та адаптувати політику безпеки.

У міру ускладнення архітектури автомобіля електронні системи захисту перейшли від автономних модулів до інтегрованих рішень, взаємопов'язаних через шини даних. Така інтеграція дала суттєві переваги у вигляді централізованого

управління, синхронізації дій та можливості комплексної діагностики, але також породила нові проблеми: уразливість шини даних стала точкою входу для атак, які дозволяють впливати на кілька підсистем одночасно. Із цього моменту питання безпеки почало охоплювати не лише окремі електронні блоки, а й протоколи обміну, механізми аутентифікації повідомлень і захист від несанкціонованого доступу до внутрішньої мережі автомобіля. Важливо підкреслити, що електронні системи захисту принесли із собою не лише нові технічні можливості, а й нові вимоги до життєвого циклу системи. Програмне забезпечення стало критично важливим елементом, що потребує оновлення, моніторингу та управління вразливостями. У цьому контексті роль виробника перестала обмежуватися апаратною надійністю: з’явилася необхідність організувати процеси випуску патчів, криптографічних ключів та процедур реагування на інциденти. Для дослідника або інженера це означає, що проект системи захисту потрібно розглядати в ширшому часовому горизонті – не лише як апаратну реалізацію, а як сервіс, що потребує підтримки протягом усього життя автомобіля.

Сучасні електронні системи захисту автомобілів побудовані на комбінації сенсорних технологій, мікропроцесорного управління та мережевої взаємодії між блоками. Основою більшості таких систем є іммобілайзер (рис. 1.1) – пристрій, який блокує подачу палива або іскри на свічки запалювання, якщо код ідентифікації не відповідає очікуваному. Принцип його роботи полягає у взаємодії між транспондером, вмонтованим у ключ або брелок, і приймальним модулем, розташованим у блоці керування двигуном (ECU). При наближенні ключа система зчитує унікальний код, який шифрується й перевіряється на відповідність еталонному значенню. Якщо аутентифікація пройшла успішно – ECU розблоковує коло запалювання. У протилежному випадку автомобіль залишається заблокованим, навіть якщо зловмисник має фізичний доступ до замка.

Рис. 1.1 Іммобілайзер

У розвитку іммобілайзерів можна виділити кілька поколінь. Перше покоління використовувало статичні коди, що легко копіювалися, тому ефективність таких систем була відносно низькою. Друге покоління перейшло до динамічних (плаваючих) кодів, коли при кожній взаємодії генерувався новий ідентифікатор, який можна було перевірити лише один раз. Такий підхід суттєво ускладнив можливість перехоплення сигналу. Третє покоління, яке активно використовується й сьогодні, базується на криптографічних алгоритмах із симетричними та асиметричними ключами, що робить спроби злому практично неможливими без фізичного втручання у модуль. Паралельно з іммобілайзерами активно розвивалися системи сигналізації – як активного, так і пасивного типу. Пасивні рішення реагують на несанкціоноване відкриття дверей, удар чи рух усередині салону, активні ж можуть взаємодіяти з користувачем, надсилаючи сповіщення або навіть дозволяючи дистанційне блокування. Деякі сигналізації інтегровані з GPS-модулями, що забезпечує можливість відстеження транспортного засобу в реальному часі. Таким чином, електронна система перетворюється на інтелектуальний комплекс моніторингу, а не просто на джерело тривоги.

Важливою віхою розвитку стало впровадження безключового доступу (Keyless Entry/Start). У таких системах автовласник не потребує вставляти ключ у замок – достатньо мати при собі брелок, який взаємодіє з автомобілем через радіоканал. При наближенні користувача автомобіль розпізнає сигнал і розблоковує

двері, а при натисканні кнопки «Start» дозволяє запуск двигуна. Попри зручність, ці системи відкрили нові загрози – наприклад, атаки ретрансляції, коли сигнал брелока продовжується через підсилювач, що дозволяє зловмиснику обманути систему, ніби власник знаходиться поруч. Саме тому сучасні реалізації впроваджують захист на рівні часової затримки сигналу, криптографічний обмін та перевірку контексту (чи рухається брелок, чи ні).

Сьогодні електронні системи безпеки вже не працюють ізольовано. Вони тісно інтегровані з CAN-шиною (рис. 1.2) – головною комунікаційною мережею автомобіля, яка з'єднує всі основні модулі: двигун, коробку передач, гальма, дверні замки, освітлення тощо. Завдяки цьому сигнал блокування може миттєво поширюватися по всій системі, забезпечуючи комплексну реакцію на спробу злому. Проте така інтеграція породжує і вразливості: при несанкціонованому доступі до шини можливе відтворення команд, які імітують справжню поведінку системи. Це підштовхнуло виробників до розробки захищених протоколів CAN-FD, LIN, FlexRay, які підтримують автентифікацію повідомлень і контроль цілісності даних[2].

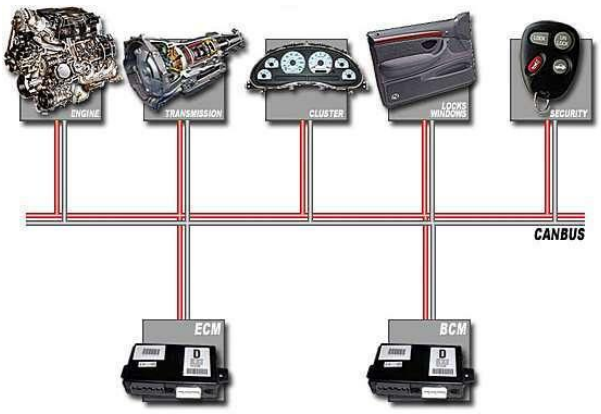


Рис. 1.2 Схема CAN-шини автомобіля

Подальша еволюція електронних систем захисту нерозривно пов'язана з цифровізацією автомобіля та переходом до концепції "connected car" – підключеного транспортного засобу, який постійно обмінюється даними із зовнішнім середовищем. Завдяки розвитку бездротових технологій (Wi-Fi, LTE, 5G,

Bluetooth Low Energy) автомобіль став не лише об’єктом механічного керування, а й учасником інформаційного простору. Це надало нові можливості для підвищення безпеки, проте водночас створило потенційні точки уразливості, які можуть бути використані для атак на електронні підсистеми. Тому сучасні системи захисту дедалі частіше інтегруються з телематичними модулями, що забезпечують централізований моніторинг, віддалену діагностику та динамічне оновлення захисних політик.

Телематичні рішення дозволяють не лише стежити за станом автомобіля, а й реагувати на підозрілу активність у режимі реального часу. Наприклад, при спробі несанкціонованого доступу система може автоматично повідомити власника, заблокувати запуск двигуна або передати координати поліції. Крім того, розвивається концепція Predictive Security – прогнозної безпеки, яка за допомогою алгоритмів машинного навчання аналізує поведінку користувача, маршрути та часові патерни, щоб виявити аномальні дії. Це означає, що автомобіль поступово перетворюється на самонавчальну систему, здатну розпізнавати не лише фізичну спробу вторгнення, а й нетипову поведінку під час експлуатації.

У межах цієї тенденції спостерігається злиття електронних та біометричних технологій. Раніше електронна система перевіряла лише наявність правильного сигналу, коду чи чипа. Однак сучасні підходи прагнуть до верифікації самої особистості користувача. Це стало можливим завдяки розвитку сенсорів і потужних одноплатних комп’ютерів, таких як Raspberry Pi, що здатні обробляти складні алгоритми розпізнавання в режимі реального часу. Біометричні модулі можуть бути інтегровані в систему запуску автомобіля, блок керування дверима чи навіть у мультимедійну панель, формуючи новий рівень персоналізованої безпеки, де кожен водій має унікальний «цифровий відбиток». Інтеграція таких систем вимагає не лише обчислювальної потужності, а й продуманих архітектурних рішень. Використання біометрії передбачає роботу з конфіденційними даними, що потребує надійного шифрування, зберігання та обмеження доступу. У цьому контексті активно розвивається напрям апаратних довірених модулів (TPM) та ізольованих обчислювальних середовищ, що гарантують неможливість зчитування

біометричних шаблонів навіть у разі фізичного втручання. Водночас поєднання традиційних електронних систем (іммобілайзерів, CAN-захисту, телематики) із біометричними рішеннями формує комплексний підхід, який забезпечує багаторівневу перевірку особи та стану транспортного засобу.

Сучасний етап розвитку електронних систем захисту можна охарактеризувати як перехідний – від класичних схем ідентифікації об’єкта до розумних систем автентифікації суб’єкта. Відбувається зсув фокусу з перевірки пристрою на перевірку особистості, що ним користується. Саме ця тенденція є логічним мостом до наступного етапу – біометричних систем безпеки, які поєднують технологічну точність, інтелектуальний аналіз і високий рівень персоналізації. Вони не лише підвищують ефективність захисту, а й роблять сам процес взаємодії між водієм і транспортним засобом природним, інтуїтивним і максимально зручним.

1.1.3 Біометричні системи захисту

Біометричні системи захисту стали результатом багаторічного розвитку ідеї, що найнадійніший ключ – це сама людина. Якщо механічні та електронні системи орієнтувалися переважно на об’єкти – замки, ключі, чипи, коди, то біометрія змінила сам підхід, перенісши фокус на суб’єкта, тобто на користувача транспортного засобу. Основна концепція полягає в тому, що кожна людина має унікальні фізіологічні або поведінкові ознаки, які можна виміряти, зберегти та використовувати для підтвердження особи[27]. Це може бути форма обличчя, відбиток пальця, структура райдужної оболонки ока, голос, навіть манера ходи чи спосіб натискання на педалі. Історія біометричних технологій розпочалася задовго до цифрової епохи. Уже наприкінці XIX століття французький криміналіст Альфонс Бертильйон запропонував систему ідентифікації за антропометричними ознаками – вимірами частин тіла. Хоча з часом ця система поступилася дактилоскопії, вона стала першим науковим підходом до ідеї використання

унікальних рис людини для підтвердження її особи. Вже у XX столітті, із розвитком комп'ютерних технологій, біометрія отримала потужний імпульс: з'явилися електронні сенсори, цифрова обробка зображень та алгоритми розпізнавання, що дозволили перейти від суто криміналістичного використання до комерційного. Спочатку це були системи контролю доступу до приміщень і баз даних, а згодом – персональні пристрої, банкомати, смартфони та транспортні засоби.

Перші спроби використати біометрію в автомобілях датуються початком 2000-х років, коли виробники преміум-класу, такі як Mercedes-Benz та BMW, експериментували з сенсорами відбитків пальців для запуску двигуна. Однак на той час технології були недостатньо стабільні: сенсори давали помилки при вологості або забрудненні, а вартість їх інтеграції залишалася високою. Проте поступовий розвиток електроніки, зменшення вартості сенсорів і зростання потужності мікропроцесорів зробили біометрію доступнішою. Особливо великий вплив справив розвиток одноплатних комп'ютерів, таких як Raspberry Pi, які дозволили обробляти складні алгоритми машинного зору без потреби у великих обчислювальних ресурсах.

На сучасному етапі біометричні системи охоплюють два головні напрями – фізіологічну та поведінкову біометрію. До першої належать такі методи, як розпізнавання обличчя, відбитків пальців, долоні чи райдужної оболонки ока. Вони базуються на незмінних характеристиках тіла людини, що робить їх високонадійними. Поведінкова біометрія, навпаки, орієнтується на індивідуальні звички користувача. Наприклад, як людина натискає педалі, тримає кермо, чи навіть її стиль водіння. Такі системи використовуються для додаткової ідентифікації, що дозволяє виявити спроби несанкціонованого керування транспортним засобом навіть після проходження основної авторизації.

Особливу роль у розвитку біометричних систем відіграють технології комп'ютерного зору. Алгоритми розпізнавання обличчя, що базуються на нейронних мережах, дозволяють не лише перевірити схожість із зразком, а й визначити живість об'єкта – переконатися, що перед камерою реальна людина, а не фотографія чи відео. Ця функція, відома як anti-spoofing (рис. 1.3), стала

обов’язковим компонентом сучасних систем, адже саме вона гарантує реальність авторизації[4, 7].

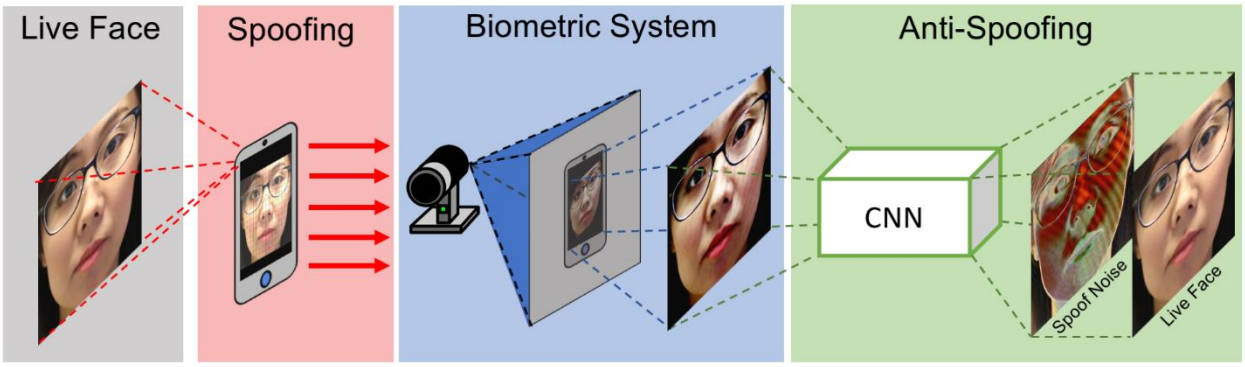


Рис. 1.3 Приклад роботи anti-spoofing

Завдяки цьому рівень надійності біометричних систем у деяких випадках уже перевищує електронні іммобілайзери. Варто підкреслити, що перехід до біометричних методів не означає повну відмову від попередніх технологій. Навпаки, у більшості сучасних рішень біометрія виступає додатковим рівнем захисту, який підвищує надійність існуючих електронних систем. Наприклад, автомобіль може вимагати не лише наявність електронного ключа, а й успішну перевірку обличчя чи відбитку водія. Такий підхід поєднує зручність із безпекою, адже лише уповноважена особа, навіть маючи фізичний ключ, зможе фактично запустити двигун.

Одним із напрямів розвитку є інтеграція охоронних систем у концепцію “Інтернету речей” (IoT). Це означає, що автомобіль може бути підключений до мережі Інтернет і обмінюватися даними з хмарними сервісами виробника або спеціальними мобільними додатками. Власник отримує можливість контролювати стан транспортного засобу у режимі реального часу, зокрема перевіряти, чи зачинені двері, чи активована сигналізація, а також віддалено запускати або блокувати двигун. Деякі преміум-виробники впроваджують біометричні системи ідентифікації водія. Вони можуть працювати на основі відбитків пальців, розпізнавання обличчя або навіть сканування райдужної оболонки ока. Такі рішення дозволяють не лише захистити автомобіль від викрадення, а й автоматично

підлаштовувати параметри керування – положення сидіння, дзеркал, клімат-контроль тощо під конкретного користувача.

Водночас у масовому сегменті активно розвиваються комплексні охоронні системи з двостороннім зв’язком. Вони поєднують у собі декілька функцій: охоронну сигналізацію, іммобілайзер, датчики руху, удару, нахилу та відчинення, а також супутникове відстеження. У разі спроби проникнення система не лише сповіщає власника, а й може автоматично передавати дані до служби безпеки або поліції. Окрему увагу слід приділити телематичним системам, що поєднують елементи навігації, комунікації та діагностики. Вони здатні збирати дані про маршрути, стиль керування, швидкість, різкі гальмування тощо, а також передавати інформацію на сервер для аналізу. На основі цих даних власник може отримати повідомлення про підозрілу активність або навіть попередження про спробу викрадення.

Попри технологічний прогрес, важливо розуміти, що навіть найрозвиненіші електронні системи мають свої вразливі місця. Основними загрозами залишаються:

1. використання ретрансляторів для подовження сигналу безконтактного ключа (так званий relay attack);
2. глушіння радіосигналів (jamming) для блокування роботи охоронної системи;
3. злам електронного блоку керування (ECU) через діагностичний роз’єм OBD-II;
4. програмна підміна ключа або викрадення криптографічних даних[9].

Саме тому сьогодні актуальним підходом є створення гібридних систем захисту, які поєднують електронні, механічні та програмні методи протидії. Наприклад, електронна система може блокувати запуск двигуна, поки механічний замок не буде розблокований, а система розпізнавання обличчя підтвердить особу водія. Такий підхід створює багаторівневий бар’єр, що суттєво ускладнює несанкціонований доступ.

Завдяки цим тенденціям електронні системи захисту поступово еволюціонують від простих сигналізацій до інтелектуальних охоронних платформ,

які не лише запобігають викраденню, а й забезпечують зручність, контроль та гнучкість для користувача.

1.2 Сучасні тенденції та перспективи розвитку систем захисту транспортних засобів

Сфера автомобільної безпеки переживає стрімку трансформацію, зумовлену розвитком цифрових технологій, зростанням рівня автоматизації та появою нових загроз. Якщо раніше головна мета систем захисту полягала у фізичному запобіганні викраденню, то сьогодні пріоритет зміщується у бік розумного моніторингу, персоналізації та кіберзахисту. Автомобіль перетворюється з механічного об’єкта на складну комп’ютерну платформу, що потребує постійного контролю, оновлення та захисту даних.

Одним із ключових напрямів сучасних тенденцій є інтеграція систем безпеки з хмарними сервісами. Це дозволяє не лише зберігати та аналізувати дані про використання автомобіля, а й оперативно оновлювати програмне забезпечення, усуваючи вразливості. Наприклад, виробники Tesla, BMW або Hyundai вже реалізували можливість дистанційного оновлення систем через мережу (технологія “over-the-air”), що підвищує рівень кіберзахисту та адаптивність охоронних рішень[15].

Водночас активно розвиваються інтелектуальні системи аутентифікації, що дозволяють визначати, хто саме сідає за кермо. Біометричні технології – розпізнавання обличчя, відбитків пальців, голосу чи навіть серцевого ритму починають відігравати ключову роль у забезпеченні персонального доступу до автомобіля. Такі рішення не лише унеможливлюють запуск двигуна сторонніми особами, а й надають додатковий рівень комфорту, адже транспорт автоматично підлаштовується під конкретного користувача.

Окрім цього, сучасні тенденції включають впровадження штучного інтелекту (ШІ) у системи безпеки. Завдяки машинному навчанню автомобіль може

самостійно розпізнавати небезпечні ситуації: наприклад, виявляти підозрілу активність навколо, спроби зчитування сигналу ключа або втручання у CAN-шину. Такі системи здатні діяти на випередження, запобігаючи загрозі ще до того, як вона призведе до реальних наслідків. Значний вплив на розвиток захисних систем має також електрифікація транспорту. Електромобілі, маючи іншу архітектуру керування, потребують нових способів блокування та моніторингу, зокрема через високовольтні системи живлення та цифрові контролери батарей. Відповідно, безпека стає не лише питанням фізичного захисту, а й енергетичної стабільності та цілісності даних, що передаються між електронними компонентами. Перспективи розвитку систем автомобільного захисту спрямовані у бік повної інтеграції біометричних технологій та створення єдиного цифрового профілю водія. У майбутньому саме цей профіль визначатиме доступ до автомобіля, його функцій та навіть параметрів руху. Система авторизації зможе розпізнавати користувача ще до того, як він наблизиться до транспортного засобу, завдяки комбінації камер, сенсорів та бездротових модулів зв’язку.

Окремий напрямок розвитку – це кіберзахист транспортних систем. Оскільки сучасні автомобілі оснащені численними бездротовими інтерфейсами (Wi-Fi, Bluetooth, LTE, GPS), вони стають потенційними об’єктами для хакерських атак. Саме тому все більше виробників упроваджують апаратні криптомодулі, які відповідають за шифрування сигналів, автентифікацію користувачів та безпечне оновлення ПЗ.

Важливою тенденцією є також перехід до модульних систем безпеки, які можна оновлювати або розширювати залежно від потреб користувача. Наприклад, до стандартного комплекту можна додати біометричний сканер, GPS-трекер чи бездротовий замок, утворюючи гнучку екосистему, що легко адаптується до нових вимог. У перспективі інтеграція таких систем із штучним інтелектом дозволить створити повністю автономні охоронні рішення, здатні самостійно приймати рішення, навчатися на основі поведінки користувача та передбачати потенційні загрози. Біометрична ідентифікація стане ключовим елементом цієї еволюції, адже

вона надає можливість поєднати зручність і безпеку без необхідності носити фізичні ключі чи картки.

Таким чином, розвиток систем безпеки рухається у напрямі інтелектуалізації, персоналізації та взаємодії між користувачем, автомобілем і цифровими мережами. Ці тенденції створюють передумови для впровадження нових рішень, серед яких: біометричні системи авторизації водія, що забезпечують високий рівень захисту та відповідають вимогам епохи “розумних” транспортних засобів.

1.3 Особливості та проблематика реалізації автономних біометричних систем на базі периферійних обчислень

Незважаючи на те, що сучасний ринок автомобільної безпеки демонструє стрімку динаміку розвитку, описану в попередніх підрозділах, на практиці існує глибокий технологічний розрив між рішеннями для преміального сегмента та масового ринку. Більшість інновацій, таких як розпізнавання обличчя, сканування райдужної оболонки ока чи голосова авторизація, залишаються прерогативою виключно нових транспортних засобів високого цінового класу. Власники автомобілів попередніх поколінь або бюджетних моделей фактично позбавлені доступу до таких технологій, оскільки пропрієтарна, тобто закрита архітектура заводських систем, не передбачає можливості їхньої доукомплектації чи модернізації сторонніми модулями. Це створює ситуацію, коли значна частина світового автопарку залишається захищеною лише морально застарілими механічними замками або примітивними іммобілайзерами, алгоритми злому яких давно відомі зловмисникам і доступні у вигляді готових електронних "відмичок".

Іншим критичним, але менш очевидним аспектом є сама архітектура сучасних «розумних» систем безпеки. У гонитві за нарощуванням функціонала та обчислювальною потужністю розробники дедалі частіше покладаються на хмарні технології (Cloud Computing). У такій моделі автомобіль виступає здебільшого терміналом для збору первинних даних, а основні операції, включно зі зберіганням

та звіркою біометричних шаблонів, виконуються на віддалених серверах. Хоча це дозволяє використовувати надпотужні нейромережі, для критично важливих систем безпеки такий підхід несе приховані ризики, які часто ігноруються маркетологами. Найперший і найбільш відчутний ризик – це латентність, або затримка передачі даних. Процес авторизації водія – це процедура, що за своєю природою вимагає миттєвої реакції, адже користувач очікує запуску двигуна в момент натискання кнопки.

Однак, якщо автомобіль знаходиться в зоні нестабільного покриття мобільної мережі – наприклад, на підземному паркінгу, у тунелі, в умовах щільної міської забудови або в гірській місцевості – час відповіді сервера може суттєво зростати, сягаючи кількох секунд, або ж з'єднання може взагалі зникнути. Для охоронної системи, яка має розблокувати запалювання "тут і зараз", така залежність від зовнішніх факторів зв'язку є неприпустимою вразливістю, що ставить під загрозу надійність експлуатації транспортного засобу.

Крім технічних обмежень, надмірна централізація даних породжує серйозну етичну проблему та проблему приватності[2]. Передача зображень обличчя, дактилоскопічних карт або голосових зліпків через загальнодоступні канали зв'язку, навіть за умови використання сучасних протоколів шифрування, створює потенційний вектор для кібератак, зокрема типу Man-in-the-Middle (перехоплення даних посередником). Більше того, накопичення величезних масивів біометричної інформації на серверах компаній-виробників робить ці бази даних надзвичайно привабливою ціллю для хакерських угруповань. Компрометація такої бази означатиме катастрофічні наслідки, адже, на відміну від скомпрометованого пароля, який можна змінити за хвилину, біометричні ідентифікатори людини є незмінними протягом усього життя. Втрата цифрового зліпка обличчя фактично означає неможливість безпечного використання біометричних систем у майбутньому.

Відповідно, виникає гостра науково-технічна потреба у зміні парадигми побудови захисних систем для масового використання. Необхідно відходити від повної залежності від "хмари" в бік архітектурних рішень, які здатні приймати

рішення про допуск автономно, без зовнішнього втручання. Це підводить нас до необхідності розгляду альтернативної моделі обчислень, яка могла б поєднати інтелектуальні можливості сучасних нейромереж із надійністю та ізольованістю локальних бортових систем.

У відповідь на системні недоліки централізованих архітектур, у сфері Інтернету речей (IoT) сформувався альтернативний підхід, відомий як периферійні обчислення (Edge Computing). Його фундаментальна суть полягає в перенесенні процесів обробки інформації максимально близько до джерела її виникнення[24]. У даному випадку, безпосередньо на бортовий комп'ютер автомобіля.

У контексті біометричної авторизації це означає зміну маршруту даних: камера не транслює "сирий" відеопотік у мережу, а передає кадри на локальний обчислювальний модуль, де розгорнута нейронна мережа. Остання виконує повний цикл операцій: від детекції обличчя до порівняння векторів ознак із збереженим еталоном не виходячи за межі замкненого цифрового контуру транспортного засобу. Така децентралізація вирішує комплекс критичних завдань:

1) По-перше, забезпечується абсолютна автономність: система функціонує однаково ефективно як у центрі мегаполіса з покриттям 5G, так і в місцях повної відсутності стільникового зв'язку.

2) По-друге, досягається найвищий рівень приватності, оскільки біометричні дескриптори зберігаються в локальній пам'яті пристрою, часто в зашифрованому вигляді, і фізично не можуть бути перехоплені під час транзакції через публічну мережу.

3) По-третє, швидкість реакції системи обмежується виключно тактовою частотою локального процесора, а не пропускною здатністю інтернет-каналу, що усуває проблему непередбачуваних затримок.

Технічна реалізація концепції Edge AI (штучного інтелекту на периферії) стала можливою завдяки стрімкій еволюції одноплатних комп'ютерів[15, 23]. Сучасні платформи, побудовані на енергоефективній ARM-архітектурі, досягли рівня продуктивності, який дозволяє виконувати складні математичні операції, необхідні для роботи згорткових нейронних мереж (CNN), у режимі реального часу.

На відміну від класичних мікроконтролерів, які обмежені простими логічними операціями, сучасні SBC мають достатній обсяг оперативної пам'яті та потужність графічних або тензорних ядер для обробки відеопотоку високої роздільної здатності. При цьому вони залишаються компактними, споживають мінімум енергії від бортової мережі та є доступними за ціною, що робить їх ідеальною апаратною основою для створення кастомних систем безпеки.

Використання таких універсальних платформ відкриває широкі можливості для так званого ретрофітингу (retrofitting) – процесу модернізації старих автомобілів новітніми технологіями. Розробка програмно-апаратного комплексу на базі доступних компонентів дозволяє створити гнучку систему, яку можна інтегрувати в електричну схему будь-якого автомобіля через стандартні інтерфейси керування реле та запалюванням. Це не лише підвищує рівень захисту конкретного транспортного засобу, а й демократизує доступ до передових технологій безпеки, перетворюючи їх з елітної опції на масовий інструмент захисту власності.

2 РОЗРОБЛЕННЯ БІОМЕТРИЧНОЇ СИСТЕМИ АВТОРИЗАЦІЇ ВОДІЯ

У цьому розділі буде розглянуто, як сучасні технології можуть бути використані для створення системи біометричної авторизації водія, що забезпечує додатковий рівень безпеки автомобіля. Уявімо пристрій, який здатен розпізнати водія за його обличчям ще до того, як буде дозволено запуск двигуна. Така система не лише підвищує рівень захисту транспортного засобу, а й створює новий рівень зручності у користуванні автомобілем.

Буде детально описано основні елементи, з яких складається система, принципи їхньої взаємодії, способи керування електричними колами та процес обробки біометричних даних. Також розглядається логіка роботи пристрою, від моменту подачі живлення до прийняття рішення про дозвіл або відмову в доступі.

Особлива увага приділяється вибору компонентів і програмних засобів, які забезпечують стабільність, точність і швидкодію системи. На прикладі створеного прототипу показано, як технології комп’ютерного зору та мікроконтролерного керування можна поєднати в одному пристрої, здатному ефективно виконувати завдання авторизації користувача в умовах реального середовища.

2.1 Формулювання вимог до пристрою

Біометрична система авторизації водія має на меті забезпечити додатковий рівень захисту автомобіля за рахунок перевірки особи перед подачею живлення на систему запалювання. Пристрій повинен виконувати функцію посередника між замком запалювання та колом живлення двигуна, дозволяючи його запуск лише після успішної ідентифікації користувача[25]. У процесі розробки було сформульовано низку функціональних і технічних вимог, які визначають загальні принципи побудови системи та її роботу в межах прототипу.

Основною вимогою до системи є можливість точної та швидкої ідентифікації водія за біометричними даними, зокрема, за зображенням обличчя. Пристрій має забезпечувати стабільну роботу алгоритму розпізнавання навіть за змінного освітлення та незначних відхилень у положенні обличчя відносно камери. Час обробки даних не повинен перевищувати кількох секунд, щоб не створювати затримки під час запуску автомобіля. Крім того, система повинна мати модульну структуру, що дозволяє відокремити апаратну частину (живлення, камера, реле, індикація) від програмної (розпізнавання, аналіз, прийняття рішення). Це спрощує налагодження, тестування та можливе оновлення окремих модулів без зміни всієї конструкції. Важливо, щоб пристрій працював автономно, тобто міг виконувати свої функції без підключення до зовнішнього сервера або мережі, використовуючи лише вбудовані обчислювальні ресурси[23]. Також потрібно щоб була наявність системи індикації, що сигналізує користувачу про стан пристрою. Після проходження авторизації повинен активуватись зелений індикатор, який повідомляє про дозвіл на запуск двигуна, а в разі відмови або помилки червоний. Це дозволяє зрозуміти результат перевірки без необхідності взаємодії з програмним інтерфейсом.

Окремо визначено вимоги до живлення та електричної безпеки. Усі елементи пристрою мають працювати при стабілізованій напрузі, відокремленій від основної бортової мережі автомобіля. Для цього передбачено блок живлення, який перетворює напругу 12 В у стабільні 5 В, необхідні для роботи електронних компонентів. Між низьковольтною логікою та силовим колом має бути гальванічна розв’язка, що запобігає потраплянню імпульсних перешкод у систему керування. Пристрій повинен мати можливість розширення функціоналу. Зокрема, передбачено інтеграцію датчика присутності або модуля бездротового зв’язку, який може передавати інформацію власнику у випадку несанкціонованої спроби доступу. Це відкриває перспективу подальшого розвитку системи у напрямку повноцінної телеметрії чи охоронного комплексу.

Загальний принцип взаємодії між елементами системи зображено на UML-схемі (рис. 2.1). У ній показано, як дані від користувача через камеру передаються до мікроконтролера, як відбувається їхня обробка, і як формується сигнал, що керує комутаційним елементом. Схема ілюструє логіку взаємодії між блоками. Від моменту активації живлення до завершення процесу авторизації. Такий підхід дає змогу чітко зрозуміти послідовність подій і визначити місця, де можливі затримки або помилки у роботі.

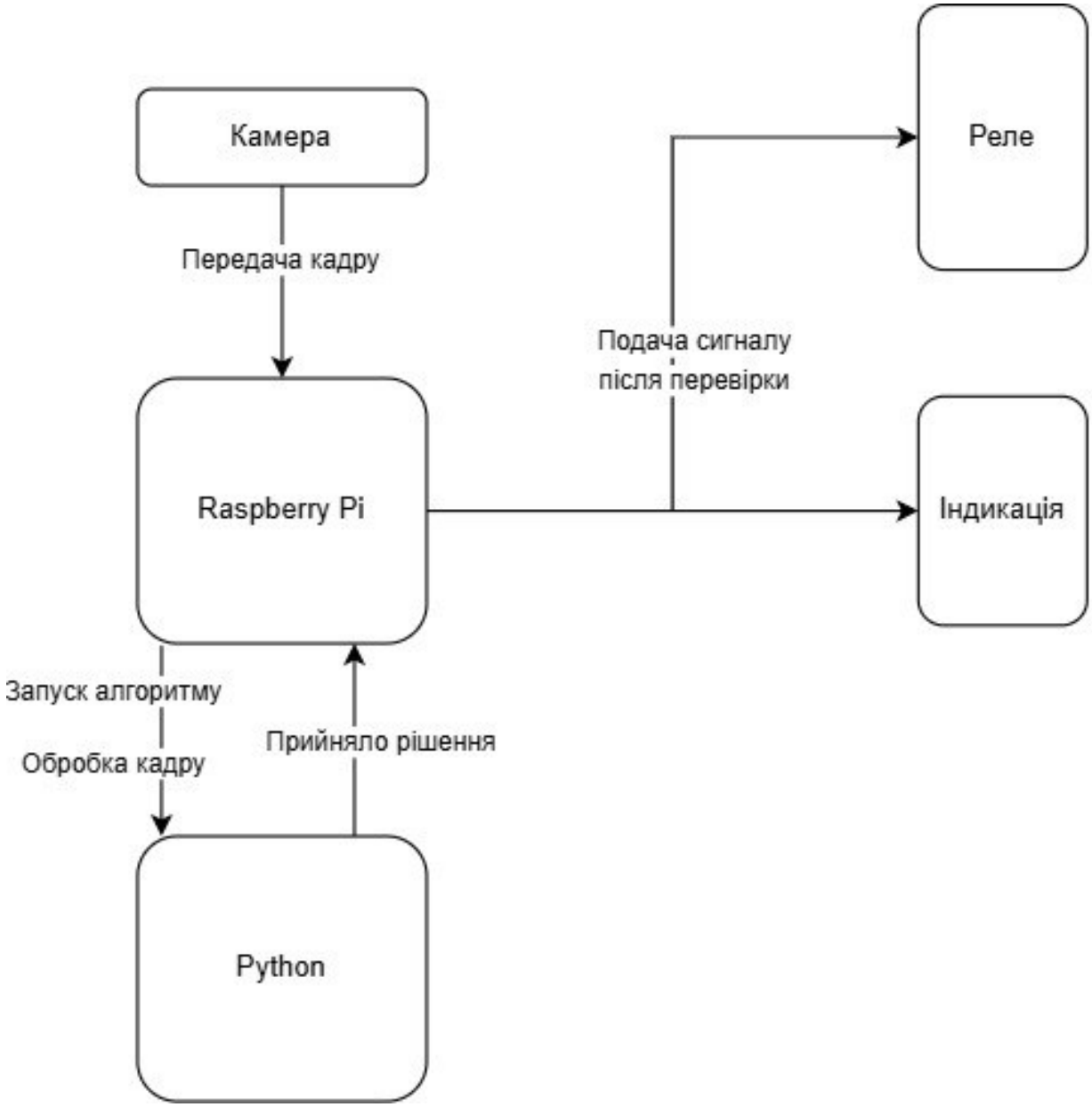


Рис. 2.1 Принцип взаємодії системи

2.2 Обґрунтування вибору компонентів для пристрою на Raspberry Pi

1) Мікроконтролер

Raspberry Pi 4 Model B (рис. 2.2) виступає центральним елементом усієї біометричної системи авторизації водія. Це одноплатний комп’ютер, який поєднує в собі мікропроцесор, оперативну пам’ять, інтерфейси введення-виведення та комунікаційні модулі[21].

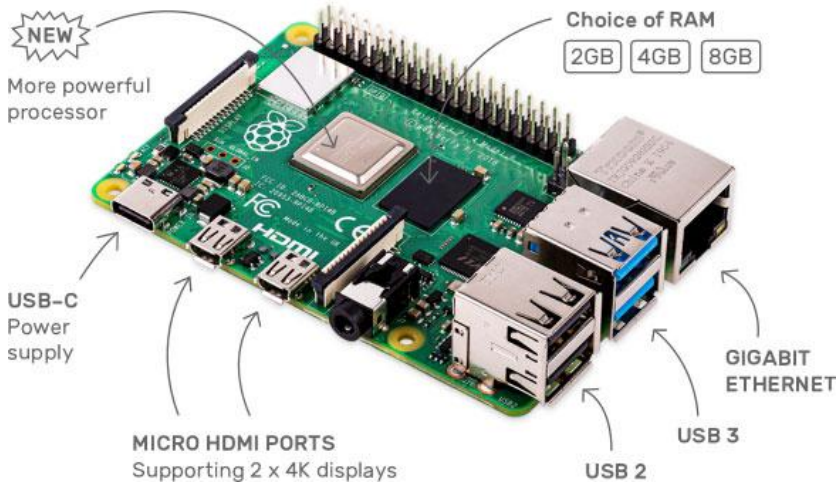


Рис. 2.2 Одноплатний комп’ютер Raspberry Pi 4 Model B

Основу складає чотириядерний процесор ARM Cortex-A72 із тактовою частотою 1,5 ГГц, що дозволяє виконувати одночасно кілька обчислювальних завдань, зокрема обробку зображень, порівняння біометричних даних і керування периферією. Обсяг оперативної пам’яті у різних версіях становить 2, 4 або 8 ГБ LPDDR4, що забезпечує достатній запас для запуску операційної системи Linux та бібліотек, необхідних для роботи із камерою та нейронними мережами.

Плата має розширені комунікаційні можливості: Gigabit Ethernet, Bluetooth 5.0 та Wi-Fi 802.11ac. Наявність Wi-Fi дозволяє передавати дані про стан системи або спроби доступу на зовнішній сервер чи смартфон, якщо це потрібно у подальших модифікаціях. Для підключення периферії використовується 40-контактний GPIO-роз’єм, який містить лінії живлення (3.3 V і 5 V), заземлення та

цифрові входи-виходи. Через ці контакти Raspberry Pi взаємодіє з модулем реле, індикаторами та іншими елементами.

Окремо варто зазначити Camera Serial Interface (CSI) – це роз'єм для підключення камери, який використовується у нашій системі для зчитування зображення обличчя водія. Через цей інтерфейс забезпечується передача відеопотоку з мінімальними затримками, що критично для алгоритмів розпізнавання. Інші порти, такі як HDMI та USB 3.0, використовуються лише під час налагодження або візуального контролю роботи програми.

Raspberry Pi 4 працює під керуванням операційної системи Raspberry Pi OS, на основі якої інсталюються бібліотеки для комп'ютерного зору (OpenCV, face_recognition, NumPy тощо)[17, 20]. Вони дозволяють реалізувати обробку зображень, виявлення обличчя та ідентифікацію користувача. Керування реле виконується через GPIO-піни за допомогою модулів RPI.GPIO або gpiozero мови Python. Вихід керуючого сигналу передається через оптрон та драйвер ULN2803A на модуль реле, який імітує розрив ланцюга запалювання.

У схемі живлення плата отримує стабілізовану напругу 5 вольт через USB-C або від модуля LM2596HV, який знижує напрягу автомобільного акумулятора з 12 V до безпечного рівня. Заземлення всіх пристроїв з'єднане спільно, що дозволяє узгодити логічні рівні та уникнути перешкод. Енергоспоживання моделі невелике до 3 Вт, що дає можливість живити пристрій навіть через допоміжну лінію автомобіля.

2) Камера

Камера TCM8240MD (рис. 2.3) виконує функцію основного сенсорного елемента системи, забезпечуючи зчитування зображення обличчя водія. Вона базується на CMOS-сенсорі, що відрізняється низьким енергоспоживанням, компактністю та високою швидкістю роботи[22]. Даний модуль має роздільну здатність 1,3 мегапікселя (1280×1024) і здатен формувати кадри з частотою до 30 кадрів за секунду. Конструктивно він розроблений для вбудованих систем, тому не потребує складного охолодження чи додаткових компонентів живлення.

Рис. 2.3 Камера TCM8240MD

Основна особливість TCM8240MD наявність кількох окремих ліній живлення: DVDD (1.6 V) для цифрової частини, PVDD (2.8 V) для аналогових елементів та IOVDD (до 2.8 V) для інтерфейсної логіки. Така архітектура дозволяє оптимізувати споживання енергії й знизити рівень шумів на зображенні. Передача даних відбувається через паралельний інтерфейс DOUT0–DOUT7, який забезпечує 8-бітне представлення кожного пікселя. Додатково підтримується тактовий сигнал DCLK і службові лінії HBLK та VBLK, що синхронізують передачу кожного рядка й кадру.

У складі прототипу камера відіграє роль сенсора, який подає дані у цифрову частину системи Raspberry Pi. З'єднання здійснюється через спеціальний адаптер або шлейф CSI. Інтерфейс SCL/SDA використовується для налаштування параметрів яскравості, експозиції, контрасту та кольорового балансу. Сам модуль підтримує автоматичне регулювання цих характеристик, тому може коректно працювати за змінного освітлення, характерного для автомобіля (наприклад, при переході з денного світла у тінь або підсвічуванні фар).

Завдяки своїй компактності камера легко інтегрується в корпус пристрою або навіть у панель автомобіля. Вона може бути розміщена в зоні огляду водія, де є прямий доступ до обличчя, наприклад, біля приладової панелі або дзеркала

заднього виду. Відстань зчитування може варіюватися від 30 до 60 см цього достатньо, щоб впевнено розпізнати риси обличчя в межах салону.

3) Стабілізація живлення для мікроконтролера

Модуль стабілізації живлення на основі мікросхеми LM2596HV використовується для перетворення напруги автомобільного акумулятора (12 В) у стабільну напругу 5 В, необхідну для роботи мікрокомп'ютера Raspberry Pi, камери та допоміжних компонентів. Оскільки в автомобільній електромережі напруга може коливатися в діапазоні 11–15 В, а під час запуску двигуна короткочасно зростати до 18 В, стабілізатор має бути достатньо надійним і стійким до перепадів. Саме модифікація HV (High Voltage) забезпечує підтримку вхідної напруги до 60 В, що робить цей перетворювач придатним для автомобільних застосувань.

Мікросхема LM2596HV (рис. 2.4) є імпульсним понижувальним стабілізатором типу buck converter. Принцип її роботи полягає в тому, що вхідна постійна напруга переривається високочастотними імпульсами (до 150 кГц), які потім згладжуються LC-фільтром дроселем і конденсатором. Така схема дозволяє досягати високого коефіцієнта корисної дії (до 90 %), при цьому втрати тепла мінімальні. На відміну від лінійних стабілізаторів, які розсіюють зайву енергію у вигляді тепла, LM2596HV перетворює її в імпульсну форму, що дає змогу значно зменшити нагрівання навіть при тривалому навантаженні. Модуль має чотири основні контакти: VIN+, VIN− для підключення до акумулятора, та VOUT+, VOUT− для живлення логічної частини системи. На платі розташовано підлаштувальний резистор, за допомогою якого можна точно виставити необхідну напругу на виході. Для безпеки перед входом підключається запобіжник номіналом 5–10 А, який захищає модуль від короткого замикання або перевантаження.

Рис. 2.4 Стабілізатор живлення LM2596HV

LM2596HV живить усі компоненти системи Raspberry Pi, оптрон, драйвер ULN2803A та індикатори. Кожна лінія підключення заземлюється спільно, щоб уникнути потенційних різниць між логічними рівнями. На виході блоку додатково встановлюється конденсатор великої ємності (від 470 мкФ), який компенсує короткочасні провали напруги під час пікових навантажень.

4) Оптрон

Оптрон TLP621 (Рис. 2.5) застосовується у системі для гальванічної розв'язки між мікрокомп'ютером Raspberry Pi та керуючими елементами силового ланцюга, зокрема драйвером реле. Його основна функція полягає у передачі логічного сигналу без прямого електричного контакту, що забезпечує захист контролера від високих струмів або стрибків напруги, які можуть виникати в автомобільній мережі. Усередині корпусу оптрона розміщено інфрачервоний світлодіод і фототранзистор, що реагує на випромінювання цього діода. Таким чином, сигнал передається у вигляді світлового імпульсу, ізолюючи вхідну і вихідну частину схеми.

Рис. 2.5 Оптрон TLP621

Типові характеристики TLP621 включають вхідний струм світлодіода близько 10 мА, допустиму напругу між входом і виходом до 5000 В, колекторний струм фототранзистора до 50 мА та граничну швидкість перемикання до 10 мікросекунд. Робоча напруга колектора може досягати 70 В, що робить цей компонент придатним для низьковольтних систем керування, таких як автомобільні реле або модулі сигналізації. Робочий діапазон температур від −55 до +110 °C, що важливо для експлуатації в умовах автомобільного середовища.

У прототипі оптрон використовується для моделювання захисту між Raspberry Pi та транзисторною матрицею ULN2803A. Схема з'єднання передбачає, що анод світлодіода оптрона під'єднується до GPIO-виходу Raspberry Pi через резистор (близько 330 Ом), а катод до загальної шини GND. Коли програмний модуль подає логічну «1» на цей вихід, через діод проходить струм, який активує фототранзистор на вихідній стороні. Колектор фототранзистора з'єднаний із входом ULN2803A, а емітер із землею.

Завдяки такій побудові сигнал передається виключно через світловий канал, тому навіть у разі короткого замикання на вихідній стороні (наприклад, у реле) напруга не повернеться до мікроконтролера. Оптрон також допомагає усунути електричні перешкоди, які можуть виникати через індуктивні навантаження. Цей

тип елементів часто використовують у промислових контролерах і системах з високими вимогами до безпеки.

5) Драйвер

Драйвер ULN2803A (рис. 2.6) є одним із ключових компонентів системи, що виконує роль підсилювального каскаду між логічною частиною (Raspberry Pi) та виконавчими елементами реле чи іншими навантаженнями. Ця мікросхема являє собою масив із восьми незалежних Дарлінгтон-пар транзисторів, кожна з яких може комутувати струм до 500 мА при напрузі до 50 В. Така конструкція дозволяє підключати одразу кілька пристроїв і керувати ними через стандартні GPIO-виходи, які самі по собі не можуть забезпечити достатню потужність для живлення котушок реле чи світлодіодів великої яскравості.

На вхідній стороні ULN2803A приймає сигнали логічного рівня (від 2,4 В до 5 В), що робить її сумісною з GPIO Raspberry Pi. Вихідні транзистори з відкритим колектором дозволяють комутувати навантаження, підключене до зовнішнього джерела живлення у нашому випадку до 5 В від стабілізатора LM2596HV. У корпусі мікросхеми передбачено загальний вивід COM, з'єднаний із внутрішніми захисними діодами, які відводять зворотну ЕРС, що виникає під час вимкнення індуктивного навантаження (наприклад, реле).

Типова схема підключення виглядає наступним чином: GPIO-вихід Raspberry Pi через оптрон TLP621 подає логічний сигнал на вхід драйвера ULN2803A. Вихід відповідного каналу з'єднується з керуючим входом реле-модуля, а загальний вивід COM підключається до позитивної шини живлення 5 В. Земля мікросхеми об'єднана із загальною шиною всієї системи. Така побудова забезпечує надійне розділення сигналів і стабільну комутацію навантаження без перевантаження GPIO.

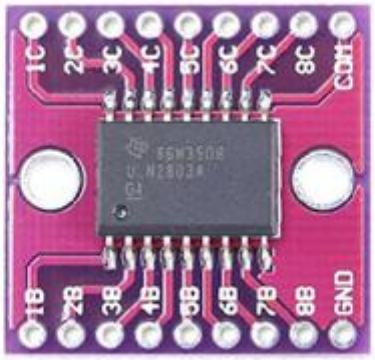


Рис. 2.6 Драйвер ULN2803A

Драйвер ULN2803A часто використовується в системах автоматики, оскільки поєднує в одному корпусі одразу вісім каналів комутації, що спрощує схему та зменшує кількість окремих елементів. Завдяки компактності та стандартному корпусу DIP-18, його зручно встановлювати на макетну плату або друковану схему. Наявність вбудованих діодів захисту знімає потребу в додаткових зовнішніх елементах, що підвищує надійність системи і спрощує збирання.

Модуль 1-Channel Relay 5V (рис. 2.7) використовується в системі як виконавчий елемент, що відповідає за комутацію живлення ланцюга запалювання автомобіля. Його головна функція створення електричного розриву між джерелом живлення та споживачем до моменту, поки водій не пройде авторизацію. Тільки після підтвердження особи через біометричну перевірку Raspberry Pi формує сигнал, який надходить на керуючий вхід реле, замикаючи контакти і дозволяючи подачу живлення до свічок запалювання.

Рис. 2.7 Модуль 1-Channel Relay 5V

Конструктивно модуль реле являє собою друковану плату з одноканальним електромеханічним реле, транзистором для керування котушкою, світлодіодним індикатором стану, а також набором клем для підключення зовнішніх проводів. Основні робочі контакти COM (спільний), NO (нормально розімкнутий) і NC (нормально замкнутий). У нашій схемі використовується пара COM–NO: у нормальному стані ланцюг розімкнутий, а після активації котушки реле замикається, дозволяючи протікання струму.

Реле живиться від постійної напруги 5 В, а для спрацьовування його котушці необхідно близько 70–80 мА струму. Такий рівень навантаження без проблем підтримується драйвером ULN2803A, який виступає проміжним підсилювачем сигналу. Для індикації спрацьовування на платі встановлено світлодіод, який загоряється при активації це дозволяє візуально відстежити стан системи. Також модуль має оптрон, який додатково ізолює вхід від вихідної частини, запобігаючи виникненню паразитних струмів.

Електромеханічне реле всередині модуля розраховане на комутацію навантажень до 10 А при 250 В змінного струму або 30 В постійного. Контакти виготовлені з матеріалу, стійкого до іскроутворення, що забезпечує довговічність навіть у середовищах з електромагнітними завадами. Котушка ізолюється від контактної групи, що виключає можливість взаємного впливу під час перемикання.

У схемі логіка роботи виглядає наступним чином: після подачі живлення від стабілізатора LM2596HV Raspberry Pi запускає процес розпізнавання. Якщо отриманий результат збігається з еталонним зразком, на вихід ULN2803A подається логічна «1». Той у свою чергу активує котушку реле, яке замикає контакти COM і NO, створюючи електричний шлях для живлення ланцюга запалювання. Якщо розпізнавання не відбулося, реле залишається у розімкнутому стані, і двигун неможливо запустити.

Також в системі є допоміжні компоненти. Вони відіграють важливу роль у забезпеченні стабільності, безпеки та коректності роботи всієї схеми. До цієї групи належать запобіжники, захисні діоди, резистори, індикаторні світлодіоди та інші

дрібні елементи, які не виконують обчислювальних функцій, але гарантують правильне функціонування пристрою в умовах експлуатації.

Одним із ключових елементів захисту є запобіжник. У схемі він встановлюється між акумулятором автомобіля та входом стабілізатора LM2596HV, запобігаючи пошкодженню пристрою у випадку короткого замикання або перенавантаження. Для таких систем зазвичай обирають запобіжники номіналом 5–10 А, залежно від сумарного споживання. У разі виникнення аварійної ситуації запобіжник перегорає першим, тим самим розриваючи коло й захищаючи дорожчі компоненти стабілізатор і мікроконтролер. У програмі Fritzing доцільно використовувати елемент Fuse 5×20 мм, який найкраще відповідає реальному вигляду автомобільного запобіжника у корпусі скляної трубки.

Ще один обов'язковий елемент захисний діод, який монтується паралельно котушці реле. Його основне завдання гасіння зворотних електричних імпульсів, що виникають під час вимикання індуктивного навантаження. Ці короткі, але потужні сплески напруги можуть пошкодити транзистори або мікросхему драйвера ULN2803A. Зазвичай використовується діод типу 1N4007 або будь-який інший випрямний діод із робочою напругою не менше 1000 В і струмом понад 1 А. Він встановлюється катодом до позитивного полюса живлення реле, анодом до вихідного контакту ULN2803A, створюючи шлях для розсіювання енергії котушки після вимикання.

Для візуального відображення станів системи застосовуються індикаторні світлодіоди. Зелений світлодіод сигналізує про дозвіл запуску двигуна, а червоний про відмову в авторизації. Вони підключаються до GPIO Raspberry Pi через резистори номіналом 330–470 Ом, які обмежують струм і захищають вихідні порти від перевантаження. Таке просте рішення дозволяє візуально контролювати логіку роботи програми навіть без підключення реального реле чи двигуна.

Крім цього, у схемі можуть бути використані конденсатори для згладжування коливань напруги. Один електролітичний конденсатор великої ємності (470-1000 мкФ) встановлюється після стабілізатора LM2596HV, щоб компенсувати короткочасні провали живлення при зміні навантаження, а кілька керамічних

конденсаторів (0.1-1 мкФ) розміщуються поруч із мікросхемами для придушення високочастотних перешкод.

2.3 Розроблення логічної архітектури системи авторизації водія

Система, що проєктується, є уособленням переходу від класичних автономних охоронних рішень до інтегрованих IoT-структур, здатних не лише виконувати окрему функцію, а й взаємодіяти з іншими цифровими сервісами. Основна ідея розробки полягає у створенні інтелектуального блоку, який контролює доступ до автомобіля шляхом розпізнавання особи водія. Пристрій, не виготовлений фізично але має повну концептуальну модель, що дозволяє розуміти його послідовність роботи, електричні з’єднання, логіку взаємодії компонентів та можливість розширення в майбутньому. Загальний принцип дії пристрою спирається на кілька ключових послідовних етапів:

Після подачі напруги від автомобільного акумулятора вмикається мікроконтролер, який керує всіма підключеними модулями. На етапі ініціалізації проводиться перевірка стану живлення, тестування камери та підготовка інтерфейсів обміну даними. Камера отримує дозвіл на зчитування кадру, після чого формується зображення обличчя водія. Зображення обробляється на рівні мікроконтролера або відправляється на обчислювальний модуль для порівняння з еталонними зразками, що зберігаються у внутрішній пам’яті системи. Якщо результати аналізу підтверджують збіг із даними авторизованого користувача, формується сигнал керування, який подається на модуль реле. Реле, у свою чергу, замикає силову лінію, що подає напругу на систему запалення.

Таким чином, автомобіль переходить у стан готовності до запуску. У випадку, якщо автентифікація не проходить успішно, силова лінія залишається розімкненою, а користувач отримує візуальний сигнал на світлодіоді, який вказує на відмову. В подальшій модифікації система може додатково передавати інформацію про невдалі

спроби запуску через бездротовий канал на мобільний пристрій або сервер користувача.

Логічна архітектура проєкту передбачає розмежування системи на кілька функціональних рівнів. Перший рівень це сенсорний, який включає пристрої зчитування (камеру та датчики стану). Другий рівень аналітичний, де відбувається обробка інформації, виконання алгоритмів розпізнавання та прийняття рішення. Третій рівень виконавчий, який відповідає за фізичну дію, тобто керування реле та передавання сигналів на підсистеми автомобіля. Останній рівень можна розглядати як інформаційно-комунікаційний, що забезпечує обмін даними між пристроєм та зовнішніми мережами, якщо система інтегрується в середовище Інтернету речей.

Пристрій не обмежується вузькою функцією блокування двигуна. Його структура дозволяє розглядати як основу для створення більш комплексної системи безпеки, здатної здійснювати моніторинг подій, вести журнал доступів, передавати сигнали про спроби несанкціонованого запуску, а також взаємодіяти з іншими вузлами транспортного засобу. Наприклад, через той самий мікроконтролер може здійснюватися контроль замикання дверей, сповіщення про стан батареї або навіть дистанційне оновлення бази користувачів. Окрему увагу при проєктуванні займає питання безпеки та стабільності роботи всієї системи. Хоча в межах цього дослідження розглядається лише теоретичний прототип, структура живлення і логіка ввімкнення були продумані так, щоб уникнути будь-яких ризиків короткого замикання чи перегріву компонентів. Для цього передбачено використання стабілізатора напруги, окремого каналу живлення для мікроконтролера і виконавчих елементів, а також діодного захисту ланцюга. Такий підхід імітує реальні інженерні рішення, які застосовуються у сучасних автомобільних системах, забезпечуючи сумісність і безпечну роботу пристрою при коливаннях напруги.

Була зроблена принципіальну схему (рис. 2.8) в програмі Fritzing де демонструється взаємозв’язок між мікроконтролером, камерою, стабілізатором, реле, драйвером і світлодіодною індикацією.

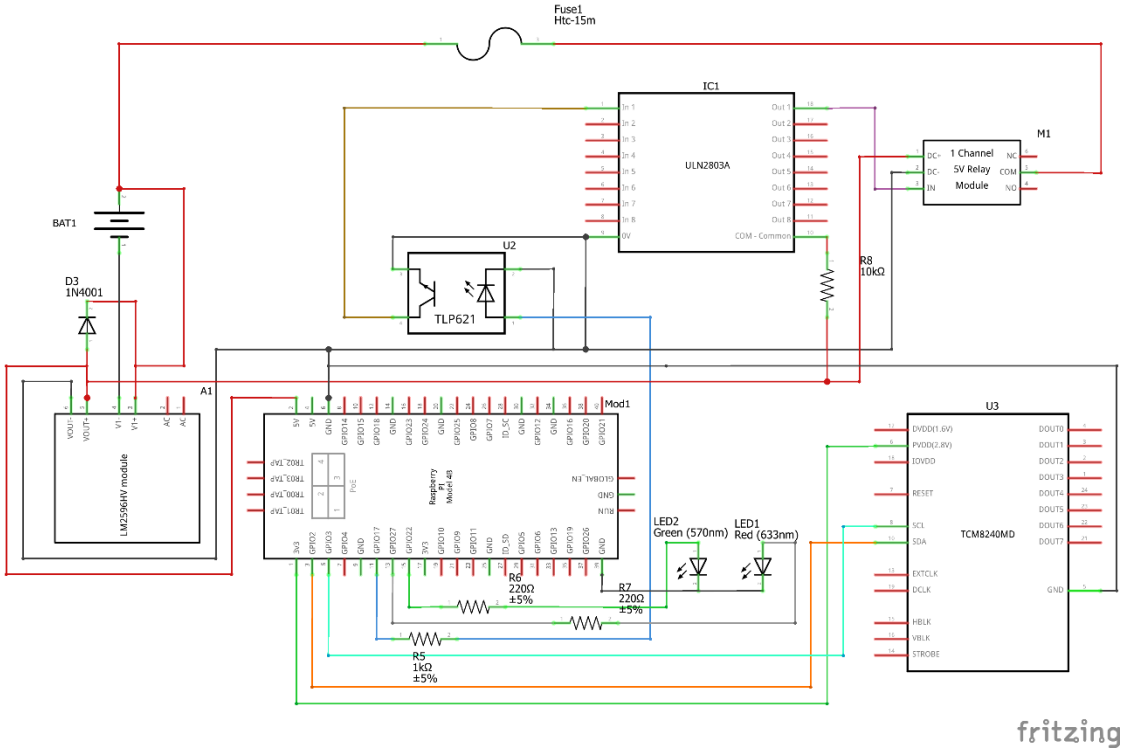


Рис. 2.8 Принципіальна схема пристрою

На схемі відображено послідовність з’єднань між логічними та силовими колами, а також позначено напрямки сигналів керування. Для ефективної роботи системи важливо, щоб усі її складові не існували окремо, а функціонували як цілісна структура. В основі лежить принцип послідовного обміну сигналами від моменту появи живлення до завершення процесу ідентифікації користувача. Після подачі живлення з автомобільного акумулятора енергія надходить на стабілізатор, де напруга перетворюється на рівень, безпечний для мікроконтролера. Цей етап критично важливий, адже коливання напруги в бортовій мережі можуть призвести до перезапуску або пошкодження електронних вузлів. Вихід стабілізатора живить Raspberry Pi, а далі ланцюг логічних елементів, які реагують на програмні сигнали. Саме з цього моменту починається активна взаємодія компонентів, що утворюють основу пристрою.

Мікроконтролер ініціює роботу камери, яка через інтерфейс передає дані у цифровому вигляді. Відбувається короткий період затримки, система формує кадр, проводить базову обробку зображення та готує його до подальшої перевірки. Усе це

відбувається автоматично, без втручання користувача. Якщо камера не підключена або не передає дані, пристрій переходить у режим очікування, не подаючи напругу на силові елементи. Після зчитування кадру запускається алгоритм перевірки. У пам’яті мікроконтролера зберігаються попередньо записані зразки обличчя осіб, яким дозволено керувати автомобілем. Алгоритм порівнює отримане зображення з базою, використовуючи ключові ознаки: пропорції, розташування очей, форму обличчя. Якщо збіг підтверджується, формується сигнал логічного рівня, який передається через драйвер ULN2803A до входу реле.

Драйвер виступає посередником між мікроконтролером і реле, захищаючи логічні виходи від перевантаження. У момент активації він пропускає імпульс на керуючий контакт реле. Це, своєю чергою, замикає силову лінію, яка пов’язана із системою запалення. Таким чином, автомобіль отримує дозвіл на запуск. Якщо результат авторизації негативний, драйвер не подає сигналу, і ланцюг залишається розімкненим. У такому випадку живлення на свічки запалення не надходить, що фактично унеможливлює запуск двигуна. Додатково до основного процесу система може виконувати кілька допоміжних дій. Наприклад, подавати індикацію про статус авторизації за допомогою світлодіодів: зелений означає дозвіл, червоний сигналізує відмову. Крім цього, передбачено можливість підключення бездротових модулів, які відправляють короткі повідомлення або записують лог подій на зовнішній сервер. Це розширює систему в напрямку Інтернету речей, дозволяючи реалізувати моніторинг у реальному часі. Структура пристрою побудована так, щоб мінімізувати ризик збоїв. Навіть якщо втрачено зв’язок із мережею чи відсутнє підключення до сервера, пристрій продовжує виконувати свої локальні функції: розпізнавання, перевірку, керування. Це забезпечує незалежність від зовнішніх факторів і дозволяє використовувати систему як автономну одиницю, що працює без постійного доступу до інтернету.

Окрім локального режиму, проєктована система має потенціал стати частиною ширшої екосистеми Інтернету речей. Це означає, що пристрій може не лише виконувати автономну перевірку особи водія, а й як приклад передавати дані про події на зовнішні пристрої або сервери. Таке рішення може відкрити новий

рівень безпеки, оскільки власник отримує змогу контролювати доступ до автомобіля навіть на відстані. Наприклад, в основі IoT-підходу лежить концепція обміну інформацією через мережу[24]. Для цього в системі можна передбачити модуль бездротового зв’язку, наприклад, Wi-Fi або GSM. Після проходження чи відмови в авторизації дані можуть бути передані у вигляді короткого повідомлення або запису в базу даних. Це дозволяє вести журнал спроб запуску автомобіля, фіксуючи час, дату, тип результату й навіть фото особи, що проходила перевірку. Такий підхід забезпечує глибший рівень контролю над безпекою транспортного засобу. Інтеграція системи з мобільним додатком або вебінтерфейсом відкриває додаткові можливості для власника. Через простий графічний інтерфейс користувач може переглядати журнал доступів, отримувати сповіщення про спроби несанкціонованого запуску чи навіть дистанційно блокувати роботу системи. Подібна функція може бути реалізована за допомогою MQTT-протоколу, який широко використовується в IoT-пристроях завдяки своїй простоті та енергоефективності. Через нього можна організувати канал обміну між мікроконтролером і хмарним сервером без суттєвих затримок і перевантаження мережі.

У перспективі така система може бути інтегрована у більші транспортні мережі наприклад, у рамках “розумного автопарку”, де кілька автомобілів підключені до єдиної панелі моніторингу. Адміністратор або власник компанії міг би бачити, хто саме використовував той чи інший автомобіль, коли була спроба запуску, і чи було це дозволено системою. Це забезпечує не лише індивідуальний рівень безпеки, а й створює умови для контролю та управління транспортними засобами на підприємстві.

3 РОЗРОБКА ПРОГРАМНОЇ ЧАСТИНИ ДЛЯ ПРИСТРОЮ НА RASPBERRY PI

3.1 Обґрунтування вибору інструментальних засобів та архітектури програмного забезпечення

Процес проектування програмної складової для вбудованих систем (embedded systems) є багатофакторною задачею, яка вимагає пошуку оптимального балансу між продуктивністю коду, швидкістю розробки та можливостями апаратної платформи. Оскільки в якості обчислювального ядра системи обрано одноплатний комп’ютер Raspberry Pi 4 Model B, програмне середовище повинно бути, з одного боку, достатньо легким, щоб не перевантажувати процесор ARM Cortex-A72, а з іншого мати потужний математичний апарат для реалізації алгоритмів комп’ютерного зору.

Враховуючи ці обмеження, в якості основної мови програмування було обрано Python (версія 3.11)[20]. Цей вибір зумовлений не лише популярністю мови, а й специфікою задачі. Хоча Python є інтерпретованою мовою і в «чистих» обчисленнях поступається компільованим мовам типу C++, у сфері машинного навчання та обробки зображень він виступає в ролі високорівневого інтерфейсу ("клею") для низькорівневих бібліотек, написаних на C/C++ і Fortran. Це дозволяє писати лаконічний та зрозумілий код, який при цьому виконує складні матричні перетворення з максимальною ефективністю, використовуючи векторні інструкції процесора. Крім того, екосистема Python має найкращу підтримку на платформі Raspberry Pi OS, що знімає більшість проблем із сумісністю драйверів камери та GPIO-інтерфейсів.

Ключовим елементом системи, що відповідає за візуальне сприйняття навколишнього середовища, стала бібліотека OpenCV (Open Source Computer Vision Library)[3, 17]. Вибір саме цього інструментарію продиктований його універсальністю. Замість написання власних алгоритмів захоплення кадрів, які б

вимагали глибокого знання протоколів роботи з відеосенсорами, OpenCV надає абстрагований шар доступу до відеопотоку. Це дозволяє програмі однаково стабільно працювати як із підключеною через шлейф CSI-камерою Raspberry Pi, так і зі звичайною USB веб-камерою на етапі налагодження коду на ПК.

Функціонал OpenCV у проєкті не обмежується лише захопленням зображення. Бібліотека використовується для попередньої обробки кадрів (preprocessing): зміни розміру (downscaling) для пришвидшення аналізу, конвертації колірних просторів (перехід від BGR до RGB або Grayscale) та нормалізації гістограми для покращення роботи в умовах слабкого освітлення. Такий підхід дозволяє підготувати "сирі" дані з сенсора до подальшої обробки нейромережею, відсіюючи шуми та зменшуючи обчислювальне навантаження на центральний процесор.

Не менш важливою складовою програмного стеку є бібліотека NumPy. Вона слугує фундаментом для всіх математичних операцій у системі. Справа в тому, що будь-яке цифрове зображення для комп’ютера це не картинка, а багатовимірна матриця чисел (тензор), де кожен елемент відповідає значенню пікселя. Стандартні списки Python не здатні ефективно обробляти масиви розміром у мільйони елементів (наприклад, кадр 1920x1080 має понад 6 мільйонів значень кольору). NumPy вирішує цю проблему, надаючи об’єкт ndarray (n-вимірний масив), який зберігається в пам’яті як неперервний блок байтів, аналогічно масивам у мові C. Це дозволяє виконувати векторизовані операції над зображеннями (наприклад, обрізку області обличчя, порівняння дескрипторів) практично миттєво, без використання повільних циклів. Тандем OpenCV та NumPy створює надійний базис, на якому будується вся подальша логіка біометричної авторизації.

Безпосередня реалізація та відлагодження програмного коду виконувалися в інтегрованому середовищі розробки (IDE) PyCharm. Вибір саме цього інструментарію зумовлений наявністю потужних засобів для статичного аналізу коду, що дозволяє виявляти потенційні помилки типізації та логічні невідповідності ще на етапі написання. Крім того, критично важливою функцією є вбудована підтримка віртуальних середовищ (Virtualenv), яка дозволила ізолювати бібліотеки

проєкту від системних пакетів операційної системи[20]. Це рішення є запорукою стабільності при перенесенні коду на одноплатний комп'ютер Raspberry Pi під управлінням ОС Linux (Raspbian), де конфлікти версій бібліотек можуть призвести до збоїв у роботі всієї системи.

З метою забезпечення гнучкості, масштабованості та спрощення процесу подальшої модернізації, архітектуру програмного комплексу було побудовано за модульним принципом. На етапі проєктування було відкинуто ідею створення монолітного скрипта, в якому логіка захоплення відео, алгоритми розпізнавання та інтерфейс користувача перемішані в одному файлі. Такий підхід, хоч і є простішим для початкової реалізації, унеможливлює ефективне налагодження та робить код жорстко залежним від конкретного апаратного забезпечення. Натомість було застосовано об'єктно-орієнтований підхід, що дозволило реалізувати принцип розділення відповідальності (Separation of Concerns). Загальна структура проєкту складається з п'яти взаємопов'язаних модулів, кожен з яких інкапсулює свою внутрішню логіку та взаємодіє з іншими через чітко визначені інтерфейси.

Фундаментом конфігурації всієї системи виступає модуль ***config.py***. Цей файл слугує єдиним центром керування параметрами, виключаючи практику використання "магічних чисел", розкиданих по всьому коду. У ньому зосереджено глобальні константи, що критично впливають на поведінку алгоритмів розпізнавання. Зокрема, тут визначено біометричні пороги високої та низької довіри, які дозволяють гнучко калібрувати баланс між безпекою та зручністю використання. Зміна цих значень дає можливість адаптувати систему під конкретні умови освітлення або вимоги безпеки, забезпечуючи розпізнавання власника навіть за наявності незначних змін у зовнішності. Крім того, у цьому модулі задаються часові інтервали для тайм-аутів сесії, що є важливим аспектом для енергозбереження бортової мережі автомобіля, та коефіцієнти пропуску кадрів, які дозволяють адаптувати навантаження на процесор залежно від продуктивності конкретної моделі мікрокомп'ютера.

Безпосередня взаємодія з апаратним забезпеченням відеозахвату винесена у спеціалізований модуль драйвера ***camera.py***. Робота з відеопотоком у реальному

часі вимагає стабільного та безпечного керування ресурсами, оскільки некоректне завершення роботи з камерою може призвести до блокування пристрою на рівні ядра операційної системи, що вимагатиме повного перезавантаження. Клас драйвера, реалізований у цьому модулі, інкапсулює низькорівневі виклики бібліотеки OpenCV, забезпечуючи безпечну ініціалізацію відеосенсора з попередньою перевіркою на помилки доступу. Також на цьому рівні відбувається примусове налаштування параметрів роздільної здатності та частоти кадрів, що дозволяє стандартизувати вхідні дані незалежно від типу підключеної камери. Критично важливою функцією модуля є механізм гарантованого звільнення ресурсів (деструктор), який забезпечує коректне закриття потоку даних при завершенні роботи програми, гарантуючи високу відмовостійкість системи.

Інтелектуальне ядро системи інкапсульоване у модулі ***biometrics.py***. Цей компонент відповідає за всі операції, пов'язані з комп'ютерним зором, нейронними мережами та математичними обчисленнями. Внутрішня логіка модуля побудована за конвеєрним принципом і включає етап попередньої обробки кадрів за допомогою спеціалізованої функції. Вона виконує конвертацію зображення у формат, сумісний з бібліотекою розпізнавання, та вирішує специфічну проблему фрагментації пам'яті, яка часто виникає при роботі бібліотеки dlib на архітектурі ARM. На етапі ініціалізації модуль автоматично сканує директорію з фотографіями власника, застосовуючи метод джитерингу (jittering). Цей алгоритм генерує серію випадково викривлених варіацій кожного вхідного зображення, що дозволяє створити узагальнений та стійкий до змін цифровий зліпок обличчя. Процес безпосередньої ідентифікації включає детекцію обличчя, визначення ключових біометричних точок та порівняння отриманого вектора ознак з еталонною базою за метрикою евклідової відстані[11, 12].

Для забезпечення якісного візуального зворотного зв'язку з водієм було розроблено модуль візуалізації ***interface.py***. Цей клас відповідає за графічне відображення стану системи безпосередньо на відеопотоці, реалізуючи елементи доповненої реальності. Функціонал модуля включає динамічну розмітку сцени: малювання рамок навколо знайдених облич та з'єднання біометричних точок

лініями. Колір графічних елементів динамічно змінюється залежно від рівня довіри системи: зелений сигналізує про успішну авторизацію, жовтий попереджає про зниження впевненості, а червоний вказує на невідому особу. Також модуль виводить текстову інформацію про хід сканування та час, що залишився до автоматичного блокування. Архітектурне рішення про відокремлення логіки візуалізації від алгоритмів розпізнавання дозволяє легко змінювати дизайн інтерфейсу або адаптувати його під різні типи дисплеїв без ризику порушити роботу критично важливих алгоритмів безпеки.

Координацію роботи всіх вищезгаданих компонентів здійснює головний контролер системи, реалізований у модулі ***main.py***. Цей модуль є точкою входу в програму, який послідовно ініціалізує класи камери, біометрії та інтерфейсу, після чого запускає нескінченний цикл обробки подій. Контролер керує лічильником кадрів для оптимізації обчислювального навантаження, пропускаючи аналіз проміжних зображень, та здійснює агрегацію результатів розпізнавання для уникнення хибних спрацювань на поодиноких кадрах. Також цей модуль відповідає за керування таймерами сесії та прийняття фінального рішення про успішність авторизації. У разі позитивного результату контролер ініціює процедуру допуску та відправку керуючого сигналу на виконавчі механізми автомобіля. Така модульна архітектура робить систему гнучкою, зрозумілою для підтримки та готовою до подальшого розширення функціоналу, наприклад, додавання нових методів аутентифікації.

3.2 Програмна реалізація алгоритмів розпізнавання та керування

1) config.py

Файл з налаштуваннями завжди відіграє роль тихого центру всієї системи, того місця, де зосереджені числові значення, параметри й константи, що формують характер роботи всього програмного комплексу. У даному випадку модуль config.py виступає не лише сховищем технічних параметрів, а й своєрідним інструментом,

який дозволяє відокремити логіку від налаштувань. Це означає, що зміна поведінки алгоритму не потребує переписування коду всередині основних файлів достатньо скорегувати кілька значень у конфігурації. Такий підхід добре підходить для системи, що працює з відеопотоком у реальному часі, оскільки він лишає можливість швидко адаптувати параметри під іншу камеру, інший рівень освітлення чи іншу жорсткість біометричної перевірки.

Початок файлу відкривається імпортом OpenCV: import cv2. Це робиться не заради виклику якихось складних функцій прямо в конфігурації, а щоб мати доступ до шрифтів, які OpenCV надає як частину власної графічної підсистеми. Звичайно, можна було б вбудувати цей параметр і в інший модуль, але з практичного погляду зручніше тримати і стилістичні налаштування такі як шрифти чи кольори в одному місці, поряд з усіма іншими змінними, що відповідають за оформлення інтерфейсу. Далі йде значення REFERENCE_DIR = "admin_faces". Воно задає директорію, де система шукатиме еталонні фотографії. Якщо директорію перенести або змінити, вся система переключиться на інший набір шаблонів без жодних втручань у основний код. Такий підхід дозволяє гнучко перебудовувати pipeline під потреби конкретного користувача або тестової середовища.

Лістинг коду:

```
import cv2

REFERENCE_DIR = "admin_faces"
```

Після цього розташовані два ключові пороги довіри: HIGH_CONFIDENCE_THRESHOLD = 90 та LOW_CONFIDENCE_THRESHOLD = 70. Обидва числові значення визначають, яке саме співпадіння між зразком і виявленим обличчям система вважатиме прийнятним[16]. Високий поріг тут грає роль суворого фільтра, що сигналізує про повне співпадіння, тоді як нижній поріг фіксує ситуації, коли зовнішність змінилася, але все ж залишаються характерні риси, достатні для дозволу. Завдяки цим параметрам можна настроювати систему так, щоб вона толерантно ставилася до зміни зачіски чи бороди, але водночас не пропускала випадкові подібності. Часові параметри працюють на формування загального ритму системи. Значення SCAN_TIMEOUT_SECONDS = 60 визначає,

скільки система чекатиме на появу відповідного обличчя, перш ніж визнати спробу невдалою. SUCCESS_EXIT_DELAY = 3 задає коротку паузу після успішної авторизації. Вона тут потрібна більше для естетики та зручності: користувач бачить підтвердження і має момент, щоб зрозуміти, що саме сталося, перш ніж інтерфейс зникне. FRAME_SKIP_RATE = 2 визначає, як часто система оброблятиме кадри. Оскільки розпізнавання операція ресурсомістка, особливо на слабших процесорах, тут використовується компроміс: кадри пропускаються таким чином, щоб загальна плавність картинки зберігалася, але процесор не був перевантажений непотрібними обчисленнями. Плавність роботи визначається змінною SMOOTH_FACTOR = 0.3. Коефіцієнт згладжування дозволяє пом’якшувати різкі перепади схожості між кадрами. Обчислення на основі біометрії іноді дають невеликі коливання значень через зміну освітлення або куту нахилу голови. Завдяки згладжуванню система працює спокійніше, без стрибків і випадкових "відмов", які могли б виникати при надто жорсткому аналізі кожного кадру окремо.

Лістинг коду:

```
HIGH_CONFIDENCE_THRESHOLD = 90
LOW_CONFIDENCE_THRESHOLD = 70
SCAN_TIMEOUT_SECONDS = 60
SUCCESS_EXIT_DELAY = 1
FRAME_SKIP_RATE = 2
#коефіцієнт згладжування (від 0.0 до 1.0)
#0.1 - дуже повільно/плавно
#0.3 - оптимальний баланс
#1.0 - миттєво/різко
SMOOTH_FACTOR = 0.3
```

Фінальні змінні задають кольори, якими інтерфейс позначає результати перевірки. Тут присутні три RGB-коди, що відповідають за зелений, жовтий і червоний кольори. Саме вони формують підказки на екрані, а також обрамлення навколо виявлених облич. Завершує блок шрифт, який визначає стиль тексту, що з’являється на екрані під час роботи системи. Обидві групи параметрів, і кольори, і шрифт дозволяють формувати візуальне сприйняття алгоритму, і саме тому знаходяться в конфігураційному модулі, а не змішані з логічним кодом.

Лістинг коду:

```
#візуалізація
COLOR_GREEN = (0, 255, 0)
COLOR_YELLOW = (0, 255, 255)
COLOR_RED = (0, 0, 255)
FONT = cv2.FONT_HERSHEY_SIMPLEX
```

2) camera.py

Файл, який відповідає за роботу з відеопотоком, завжди опиняється у центрі системи розпізнавання обличь, навіть якщо зовні здається звичайним “драйвером камер”. Саме тут вирішується, яким буде вихідний кадр: занадто темним, розмитим, перенасиченим або ж навпаки чітким та стабільним. У випадку біометричної системи камера не просто передає картинку, вона стає інструментом вимірювання. Тому цей модуль виступає своєрідним мостом між фізичної реальністю та алгоритмами розпізнавання, які надалі працюють з кожним кадром так, наче це математичний об’єкт. В коді передбачені як апаратні параметри, так і програмне коригування, яке підсилює деталізацію та робить кожен кадр придатнішим для подальшого аналізу.

Робота починається з імпорту необхідних бібліотек. Окрім стандартного модуля комп’ютерного зору OpenCV, тут підключається бібліотека NumPy. Це критично важливо, адже для комп’ютера зображення це велика матриця чисел, і саме NumPy дозволяє ефективно виконувати над нею математичні операції, необхідні для програмного покращення чіткості.

Лістинг коду:

```
import cv2
import numpy as np

class CameraDriver:
  def __init__(self, source=0, width=720, height=720, use_software_sharpen=True):
    self.source = source
    self.use_software_sharpen = use_software_sharpen
    self.cap = cv2.VideoCapture(source)
```

Ініціалізація класу CameraDriver одразу задає правила гри. Ми не просто відкриваємо камеру за індексом, а й визначаємо базові параметри майбутнього

потоку. Зверніть увагу на можливість увімкнення або вимкнення програмної різкості (use_software_sharpen) прямо в конструкторі. Це дає гнучкість у налаштуванні продуктивності. Далі йде блок налаштування апаратної частини. Тут примусово встановлюю роздільну здатність та частоту кадрів.

Лістинг коду:

```
#налаштування камери
    self.cap.set(cv2.CAP_PROP_FRAME_WIDTH, width)
    self.cap.set(cv2.CAP_PROP_FRAME_HEIGHT, height)
    self.cap.set(cv2.CAP_PROP_FPS, 30)

    #різкість щоб не було "мила"
    try:
      self.cap.set(cv2.CAP_PROP_SHARPNESS, 150)
    except Exception:
      pass
```

Далі йде власна математична модель для покращення зображення.

Змінна sharpen_kernel - це ядро згортки, матриця, яка діє як фільтр для виділення країв. Логіка цих чисел проста й елегантна: центральний піксель посилюється (множиться на 9), а сусідні пікселі приглушуються (множаться на -1). Це створює ефект високого контрасту на межах об'єктів, що критично важливо для алгоритмів розпізнавання, які орієнтуються на геометрію обличчя. Якщо камеру не вдалося ініціалізувати фізично, клас одразу повідомляє про критичну помилку, запобігаючи "тихому" збою.

Лістинг коду:

```
    self.sharpen_kernel = np.array([
      [-1, -1, -1],
      [-1, 9, -1],
      [-1, -1, -1]
    ])

    if not self.cap.isOpened():
      raise RuntimeError(f"Не вдалося відкрити камеру {source}")
```

Серце цього модуля метод get_frame. Саме він викликається в нескінченному циклі головної програми.

Лістинг коду:

```
def get_frame(self):
    ret, frame = self.cap.read()

    if ret and self.use_software_sharpen:
      #застосування фільтра
      try:
        frame = cv2.filter2D(frame, -1, self.sharpen_kernel)
      except Exception:
        pass  #помилка обробки - повертає оригінал

    return ret, frame
```

Після отримання "сирого" кадру (self.cap.read()), система перевіряє, чи увімкнено режим покращення. Якщо так, то до зображення застосовується фільтр filter2D з використанням раніше створеної матриці. Це перетворює розмиту картинку на чітке зображення з виразними деталями. Знову ж таки, блок обробки захищений від помилок: якщо математична операція дасть збій (наприклад, через пошкоджений кадр), система просто поверне оригінальне зображення, забезпечуючи безперервність відеопотоку.

Завершує клас метод очищення ресурсів.

Лістинг коду:

```
def release(self):
    if self.cap.isOpened():
      self.cap.release()
```

3) biometrics.py

Biometrics.py – це мозок системи. Саме тут відбувається перетворення звичайного набору пікселів у математичну модель людини. Цей файл відповідає за найскладнішу частину роботи: він має не просто знайти обличчя в кадрі, а зрозуміти, чи належить воно власнику, і зробити це за найменший проміжок часу[5, 29]. Код побудований навколо класу BiometricSystem, який інкапсулює всю логіку розпізнавання, роботи з пам'яттю та ведення журналів доступу.

На початку файлу підключаються бібліотеки для роботи з комп'ютерним зором (cv2, face_recognition) та операційною системою. Особливе місце займає

модуль pickle, який відповідає за серіалізацію об'єктів збереження навчених моделей у бінарний файл для швидкого доступу.

Лістинг коду:

```
import face_recognition
import cv2
import os
import numpy as np
import config
import datetime
import pickle  #збереження даних про навчання з обличь у файл

class BiometricSystem:
   def __init__(self):
      self.known_encodings = []
      self.known_names = []
      #файл в якому будуть зберігатись оброблені дані
      self.cache_file = "encodings.pickle"
```

Конструктор класу ініціалізує порожні списки для зберігання біометричних даних і визначає шлях до файлу кешу. Це підготовка фундаменту для подальшої роботи. Бібліотека OpenCV за замовчуванням працює в просторі BGR (Blue-Green-Red), тоді як бібліотеки аналізу облич (dlib) очікують RGB. Метод _get_safe_rgb виконує безпечну конвертацію, гарантуючи, що масив даних має правильний тип uint8 і безперервну структуру в пам'яті, що критично для уникнення помилок сегментації[17].

Лістинг коду:

```
def _get_safe_rgb(self, image_cv):
      #конвертація кадру для dlib
      rgb = cv2.cvtColor(image_cv, cv2.COLOR_BGR2RGB)
      if rgb.dtype != np.uint8:
         rgb = rgb.astype(np.uint8)
      return np.ascontiguousarray(rgb)
```

Завантаження бази найважча частина старту. Щоб уникнути перерахунку всіх фотографій при кожному запуску, система спершу перевіряє наявність файлу encodings.pickle.

Лістинг коду:

```
def load_database(self):
    #спроба завантаження з створеного pickle файлу
    if os.path.exists(self.cache_file):
      print(f"Знайдено кеш облич. Швидке завантаження...")
      try:
        with open(self.cache_file, "rb") as f:
          data = pickle.load(f)
        self.known_encodings = data["encodings"]
        self.known_names = data["names"]
        return True
      except Exception as e:
        print(f"Помилка читання кешу ({e}). Буде виконано пересканування.")
```

Якщо файл знайдено і успішно прочитано, система відновлює стан пам'яті за долі секунди, пропускаючи етап навчання. Якщо ж кеш відсутній як і відбувається коли вперше запускається пристрій то алгоритм переходить до "холодного старту". Він перевіряє наявність папки з еталонними фотографіями, визначеної у config.py, і збирає список усіх доступних графічних файлів.

Лістинг коду:

```
#якщо кешу немає то сканує папку з фото
    print(f"Аналіз фотографій з папки '{config.REFERENCE_DIR}'...")

    if not os.path.exists(config.REFERENCE_DIR):
      os.makedirs(config.REFERENCE_DIR)
      return False

    files = [f for f in os.listdir(config.REFERENCE_DIR)
        if f.lower().endswith(('.jpg', '.png', '.jpeg'))]
```

Кожне фото зчитується і перетворюється на 128-вимірний вектор[6, 24]. Використання параметра num_jitters=15 змушує нейромережу багаторазово деформувати зображення під час аналізу, що суттєво підвищує точність розпізнавання в майбутньому.

Лістинг коду:

```
for filename in files:
      path = os.path.join(config.REFERENCE_DIR, filename)
      img = cv2.imread(path)
      if img is None: continue

      rgb = self._get_safe_rgb(img)
      encs = face_recognition.face_encodings(rgb, num_jitters=15, model='large')
```

Продовження лістингу коду:

```
        if encs:
            self.known_encodings.append(encs[0])
            self.known_names.append("VODIY (ADMIN)")
```

Після завершення обробки всіх фотографій, отримані вектори "консервуються" у файл. Це завершує цикл самонавчання системи.

Лістинг коду:

```
#збереження результату у файл
    if self.known_encodings:
        print("Відбувається кешування...")
        with open(self.cache_file, "wb") as f:
            data = {
                "encodings": self.known_encodings,
                "names": self.known_names
            }
            pickle.dump(data, f)
        print("Кеш створено успішно.")

    return len(self.known_encodings) > 0
```

Функція log_result_to_file відповідає за фіксацію історії доступів. Кожна спроба, успішна чи ні, записується з міткою часу, що дозволяє проводити подальший аудит безпеки.

Лістинг коду:

```
def log_result_to_file(self, percent, status):
    timestamp = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    log_entry = f"[{timestamp}] Status: {status}, Similarity: {percent}%\n"
    try:
        with open("access_log.txt", "a", encoding="utf-8") as f:
            f.write(log_entry)
    except:
        pass
```

Метод process_frame починається з оптимізації. Обробка Full HD зображення надто повільна, тому кадр зменшується до 20% від оригіналу (fx=0.2). Це підвищує FPS без втрати якості розпізнавання, оскільки нейромережі важливі пропорції, а не деталізація пікселів. Тут також вираховуються координати обличчя (locs) та ключові точки (landmarks) для візуалізації.

Лістинг коду:

```
#обробка кадру
  def process_frame(self, frame):
    small_frame = cv2.resize(frame, (0, 0), fx=0.2, fy=0.2)
    rgb_small = self._get_safe_rgb(small_frame)

    locs = face_recognition.face_locations(rgb_small)
    landmarks = face_recognition.face_landmarks(rgb_small, locs)
    encs = face_recognition.face_encodings(rgb_small, locs)
```

Фінальна частина коду це порівняння знайденого вектора з базою через евклідову відстань[1, 24]. Результат інвертується у відсотки схожості. Система використовує трирівневу логіку довіри (SECURE, CHANGED, UNKNOWN), що базується на порогах з конфігураційного файлу. Цей блок завершує цикл обробки, повертаючи структуровані дані для прийняття рішення про допуск та відображення на екрані.

Лістинг коду:

```
results = []

    for encoding in encs:
      dists = face_recognition.face_distance(self.known_encodings, encoding)
      best_idx = np.argmin(dists)
      percent = round((1 - dists[best_idx]) * 100, 1)

      if percent >= config.HIGH_CONFIDENCE_THRESHOLD:
        status = "SECURE"
        trust = 2
      elif percent >= config.LOW_CONFIDENCE_THRESHOLD:
        status = "CHANGED"
        trust = 1
      else:
        status = "UNKNOWN"
        trust = 0
```

4) interface.py

Модуль interface.py відповідає за візуальну комунікацію системи з користувачем, виступаючи своєрідним перекладачем з мови математичних векторів на мову графічних образів. Його завдання не просто намалювати рамку, а забезпечити комфортне сприйняття процесу розпізнавання, приховуючи технічні нюанси роботи алгоритмів. Клас UIManager ізолює всю логіку малювання (rendering logic),

що дозволяє змінювати дизайн інтерфейсу без втручання в ядро біометричної системи. Однією з найпоширеніших проблем у системах комп'ютерного зору є "тремтіння" інтерфейсу: оскільки координати обличчя перераховуються щокадру, рамка може хаотично стрибати навіть якщо людина стоїть нерухомо. Щоб вирішити це, реалізовано механізм математичного згладжування (лінійної інтерполяції). Коефіцієнт factor визначає баланс між плавністю та швидкістю реакції.

Лістинг коду:

```
def smooth_value(self, current, target, factor):
    if current is None:
      return target
    return current * (1 - factor) + target * factor

  def smooth_box(self, current_box, target_box, factor):
    if not current_box:
      return target_box

    cur_t, cur_r, cur_b, cur_l = current_box
    tar_t, tar_r, tar_b, tar_l = target_box

    # Обчислення нових координат з урахуванням інерції
    new_t = int(self.smooth_value(cur_t, tar_t, factor))
    new_r = int(self.smooth_value(cur_r, tar_r, factor))
    # ... (аналогічно для інших сторін)
    return (new_t, new_r, new_b, new_l)
```

Головний метод візуалізації draw_ui виконує критично важливу операцію масштабування. Оскільки аналітичне ядро працює зі зменшеним зображенням (для економії ресурсів), інтерфейс повинен коректно відновити реальні розміри об'єктів на екрані. Тут жорстко заданий коефіцієнт scale = 5, який компенсує стиснення кадру в біометричному модулі.

Лістинг коду:

```
def draw_ui(self, frame, locations, landmarks, results, auth_status, time_left):
    #коефіцієнт масштабування (обернений до fx=0.2 з biometrics.py)
    scale = 5
    for (top, right, bottom, left), data in zip(locations, results):
      top *= scale;
      right *= scale;
      bottom *= scale;
      left *= scale
```

Далі система візуалізує рівень довіри через кольорову індикацію, перетворюючи сухі цифри ймовірності на інтуїтивно зрозумілий "світлофор": зелений для дозволу, жовтий для сумніву і червоний для заборони.

Лістинг коду:

```
color = config.COLOR_RED
if data['trust'] == 2:
   color = config.COLOR_GREEN
elif data['trust'] == 1:
   color = config.COLOR_YELLOW

cv2.rectangle(frame, (left, top), (right, bottom), color, 2)
```

Окрім рамок, система відображає ключові точки обличчя (landmarks). Це не лише додає інтерфейсу технологічного вигляду, а й дає користувачеві зворотний зв'язок: він бачить, що система "зачепилася" саме за його риси обличчя, а не за фон.

Лістинг коду:

```
for face_marks in landmarks:
        for _, points in face_marks.items():
        scaled_pts = [(int(x * scale), int(y * scale)) for x, y in points]
        for point in scaled_pts:
            cv2.circle(frame, point, 1, config.COLOR_YELLOW, -1)
```

Верхня частина екрану відведена під статус-бар інформаційну панель, яка повідомляє про загальний стан системи та час, що залишився. Це створює зрозумілий контекст для користувача (Heads-Up Display).

Лістинг коду:

```
        def _draw_status_bar(self, frame, auth_status, time_left):
            cv2.rectangle(frame, (0, 0), (1280, 60), (0, 0, 0), -1)
            msg = "ACCESS GRANTED" if auth_status else f"SCANNING...
({int(time_left)}s)"
            col = config.COLOR_GREEN if auth_status else config.COLOR_RED
            cv2.putText(frame, msg, (20, 40), config.FONT, 0.8, col, 2)
```

У випадку, якщо час на авторизацію вичерпано, спрацьовує метод блокування екрану. Він повністю перекриває відеопотік, однозначно сигналізуючи про відмову в доступі, що є фінальним акордом у сценарії невдалої ідентифікації.

Лістинг коду:

```
        def show_timeout_screen(self, frame):
            cv2.rectangle(frame, (0, 0), (1280, 720), (0, 0, 0), -1)
            cv2.putText(frame, "ACCESS DENIED", (400, 360), config.FONT, 1.5,
config.COLOR_RED, 3)
            cv2.imshow('Biometric System', frame)
            cv2.waitKey(3000)
```

5) main.py

Файл main.py відіграє роль центрального контролера (Orchestrator). Якщо попередні модулі були окремими інструментами камерою, алгоритмом чи інтерфейсом то цей скрипт визначає, як і коли вони взаємодіють. Його архітектура побудована навколо циклу обробки подій, який керує часом життя сесії, розподіляє ресурси процесора та приймає фінальне рішення про надання доступу. Для початку під’єдную усі бібліотеки які маю та з файлів які були написані до цього класи.

Лістинг коду:

```
import cv2
import time
import datetime
import config
from camera import CameraDriver
from biometrics import BiometricSystem
from interface import UIManager
```

Функція write_log відповідає за фіксацію історії роботи. Вона реалізує принцип "чорної скриньки", записуючи кожну важливу подію з точністю до секунди. Блок try-except тут є критичним елементом надійності: він гарантує, що помилка доступу до файлової системи (наприклад, переповнений диск) не призведе до аварійної зупинки всієї системи безпеки.

Лістинг коду:

```
def write_log(message):
  timestamp = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")
  log_entry = f"[{timestamp}] {message}\n"
  try:
    with open("access_log.txt", "a", encoding="utf-8") as f:
      f.write(log_entry)
  except Exception as e:
    print(f"Помилка запису логу: {e}")
```

Точка входу в програму це функція main(). Процес запуску огорнутий у блок обробки виключень для безпечного створення екземплярів драйверів. Ключовим моментом є виклик bio_system.load_database(). Це своєрідний запобіжник: якщо база еталонних облич відсутня або пошкоджена, система відмовиться запускатися, запобігаючи роботі в "сліпому" режимі.

Лістинг коду:

```
def main():
    try:
        write_log("=== ЗАПУСК СИСТЕМИ ===")
        camera = CameraDriver(width=1280, height=720, use_software_sharpen=True)
        bio_system = BiometricSystem()
        ui = UIManager()  # Створення інтерфейсу

        if not bio_system.load_database():
            return

    except Exception as e:
        print(f"Критична помилка. Екстренне закриття програми... {e}")
        return
```

Перед початком циклу ініціалізуються змінні стану. Окрім таймерів, тут створюються буфери smooth_locs та smooth_res. Вони необхідні для алгоритму стабілізації: система пам'ятатиме координати з попередніх кадрів, щоб інтерфейс не "тремтів". Змінна auth_counter слугує для захисту від випадкових спрацювань. Доступ надається лише після серії успішних розпізнавань.

Лістинг коду:

```
    frame_count = 0
    auth_counter = 0
    required_frames = 2
    is_authorized = False

    start_time = time.time()
    success_time = None
    max_session_percent = 0

    smooth_locs = []
    smooth_res = []
```

Основу програми складає нескінченний цикл while True. У кожній ітерації відбувається захоплення кадру з камери та перевірка тайм-ауту. Змінна time_left

постійно зменшується, і коли вона досягне нуля, система автоматично заблокує доступ.

Лістинг коду:

```
while True:
    ret, frame = camera.get_frame()
    if not ret: break

    frame_count += 1
    current_time = time.time()
    time_left = max(0, config.SCAN_TIMEOUT_SECONDS - (current_time - start_time))
```

Далі ресурсоємне розпізнавання виконується лише на кожному N-му кадрі (визначається через FRAME_SKIP_RATE), що розвантажує процесор. Отримані дані не виводяться напряму, а проходять через математичні фільтри ui.smooth_box[18]. Це перетворює дискретні вимірювання на плавний рух рамки. Якщо обличчя нове то воно додається відразу, якщо вже відстежується, то його координати інтерполюються.

Лістинг коду:

```
  if not is_authorized and frame_count % config.FRAME_SKIP_RATE == 0:
    raw_locs, raw_marks, raw_res = bio_system.process_frame(frame)

    if raw_locs:
      if not smooth_locs:
        smooth_locs = raw_locs
        smooth_res = raw_res
      else:
        temp_locs = []
        temp_res = []

        for i in range(len(raw_locs)):
          if i < len(smooth_locs):
            s_box = ui.smooth_box(smooth_locs[i], raw_locs[i], config.SMOOTH_FACTOR)
            temp_locs.append(s_box)
            temp_res.append({'name': raw_res[i]['name'], 'percent': s_pct, 'trust': s_trust})
          else:
            temp_locs.append(raw_locs[i])
            temp_res.append(raw_res[i])

        smooth_locs = temp_locs
        smooth_res = temp_res
      cached_marks = raw_marks
```

Продовження лістингу коду:

```
        else:
            smooth_locs = []
            smooth_res = []
            cached_marks = []
```

Система аналізує результати згладжених даних. Лічильник auth_counter працює як накопичувач впевненості: він збільшується, якщо в кадрі є довірена особа, і зменшується, якщо ні. Коли накопичено достатньо підтверджень (required_frames), активується прапор is_authorized.

Лістинг коду:

```
        found_valid_face = False
        for data in smooth_res:
            if data['percent'] > max_session_percent:
                max_session_percent = data['percent']
            if data['trust'] > 0:
                found_valid_face = True

        if found_valid_face:
            auth_counter += 1
        else:
            if auth_counter > 0: auth_counter -= 1

    # прийняття рішення
    if auth_counter >= required_frames:
        is_authorized = True
        if success_time is None:
            success_time = current_time
            result_msg = f"Авторизація успішна. Макс. відсоток: {max_session_percent}%"
            print(f"\n>>> {result_msg} <<<\n")
            write_log(result_msg)
```

Незалежно від розпізнавання, метод ui.draw_ui викликається у кожному кадрі, забезпечуючи відмальовку інтерфейсу. Програма передбачає три сценарії завершення: успішна авторизація (із затримкою для візуального підтвердження), вичерпання часу (показ екрану відмови) або ручне переривання клавішею 'q'. В кінці обов'язково викликається звільнення ресурсів камери.

Лістинг коду:

```
try:
        ui.draw_ui(frame, smooth_locs, cached_marks if 'cached_marks' in locals() else [],
smooth_res,
```

Продовження лістингу коду:

```
            is_authorized, time_left)
    except:
        pass

    cv2.imshow('Biometric System', frame)

    if is_authorized and (current_time - success_time) > config.SUCCESS_EXIT_DELAY:
        break

    if time_left <= 0 and not is_authorized:
        ui.show_timeout_screen(frame)
        break

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

camera.release()
cv2.destroyAllWindows()
```

4 ТЕСТУВАННЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ РОБОТИ СИСТЕМИ АУТЕНТИФІКАЦІЇ ВОДІЯ

Для забезпечення об’єктивності отриманих результатів та верифікації ефективності впроваджених алгоритмів оптимізації, зокрема механізмів пропуску кадрів та масштабування зображення, тестування програмного комплексу здійснювалося на апаратному забезпеченні з обмеженими обчислювальними ресурсами. Використання такої конфігурації дозволяє змоделювати реальні умови експлуатації на вбудованих системах або бюджетних комп'ютерах, які є типовими для систем контролю доступу.

В якості тестового стенду було використано портативний комп'ютер моделі HP 255 G5. Апаратна платформа пристрою базується на центральному процесорі AMD E2-7110 APU (4 ядра, базова тактова частота 1.8 ГГц), який відноситься до початкового сегмента та виступає ідеальним індикатором для перевірки швидкодії ресурсомістких алгоритмів комп'ютерного зору. Графічна підсистема представлена інтегрованим адаптером AMD Radeon R2 Graphics, роль якого в даній реалізації зведена до виведення інтерфейсу, оскільки основне обчислювальне навантаження покладено на CPU. Система оснащена 8 ГБ оперативної пам'яті стандарту DDR3L, чого цілком достатньо для зберігання завантажених бібліотек OpenCV та масиву біометричних даних без необхідності використання файлу підкачки. Дискова підсистема реалізована на базі твердотільного накопичувача (SSD) об'ємом 256 ГБ, що забезпечило мінімальну затримку при зчитуванні файлу кешу (encodings.pickle) під час ініціалізації системи та дозволило виконувати швидкий запис логів подій у реальному часі.

4.1 Методика проведення тестування та підготовка еталонних даних

Метою є перевірка працездатності розробленого програмного алгоритму біометричної ідентифікації, оцінка його швидкодії та точності розпізнавання

обличчя в різних умовах експлуатації. Основним критерієм оцінки справності коду є швидкість сканування та відсоток схожості (Similarity Percentage), який система визначає при порівнянні поточного зображення з камери з еталонними даними.

Для тестування роботи алгоритму було зроблено декілька фото мого обличчя з декількох ракурсів (центр, ліва та праві сторони). Основним критерієм для мене чи справно працює код це те на скільки швидко буде сканувати та скільки відсотків він буде бачити на моєму обличчі, порівнюючи його з еталонними даними які він зробив під час першого запуску.

Але спочатку мені треба переконатися що код працює справно. Для цього я завантажую еталонні фотографії в папку яку буде сканувати алгоритм на наявність їх (рис. 4.1).

Рис. 4.1 Еталонні фотографії для створення кешу

Після того як додав фото роблю перший запуск та бачу в консолі що алгоритм почав робити сканувати обличчя з наданих мною фотографій (рис. 4.2).

```
[СИСТЕМА] Аналіз фотографій з папки 'admin_faces'...
[СИСТЕМА] Навчання на 5 фото...
[ОК] Оброблено: 10.jpg
[ОК] Оброблено: 9.jpg
[ОК] Оброблено: Admin_11.jpg
[ОК] Оброблено: admin_2.jpg
[ОК] Оброблено: admin_8.jpg
[СИСТЕМА] Збереження бази у файл (кешування)...
[СИСТЕМА] Кеш створено успішно.
Початок відеоспостереження...
```

Рис. 4.2 Текстові повідомлення про роботу алгоритму

Після того як воно почало відеоспостреження то в консолі почало виводити відсотки які написані також і у вікні яке автоматично вікдкривається (рис. 4.3).

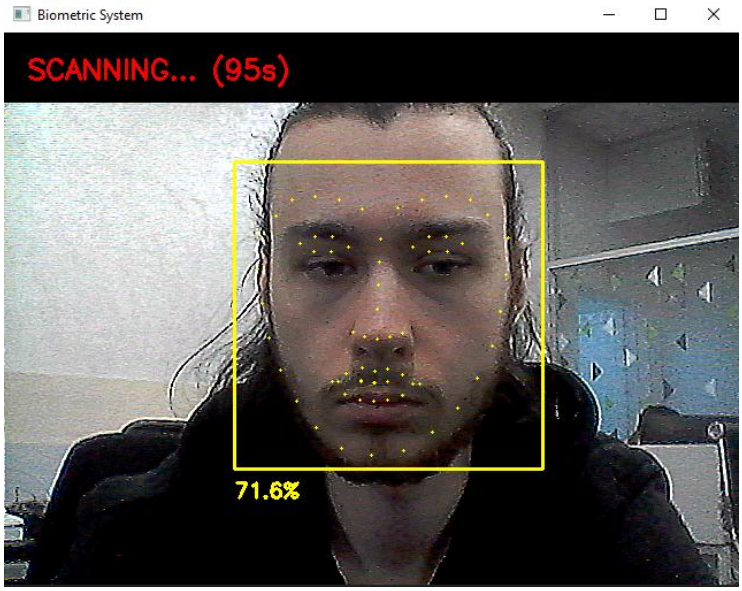


Рис. 4.3 Процес сканування обличчя за допомогою камери

Як пам’ятаємо після сканування може бути три відповіді:

1) менше 65% - схожості не має, не пройшов сканування;
2) від 65% до 90% - сканування пройде успішно але з затримкою по часу сканування;
3) від 90% - повна схожіть, програма автоматично закривається.

Після того як воно зробило сканування то видає повідомлення що сканування обличчя пройшло успішно але так як був діапазон від 65% до 90% то воно декілька разів перевіряє обличчя для рішення. Під час першого запуску системи алгоритм провів аналіз наданих фотографій. Результатом цього процесу стало створення двох системних файлів:

1. Файл кешу з розширенням .pickle для зберігання математичних моделей облич (енкодінгів), що дозволяє при наступних запусках уникати повторного сканування зображень та значно прискорює ініціалізацію системи.
2. Текстовий лог-файл .txt для фіксації історії подій, включаючи час, відсоток схожості та статус успішності сканування.

Для подальшої перевіркм будуть проводитися тести за різними сценаріями:

1) Зачіска: на скільки буде відхилення якщо на еталонному фото одна зачіска, а я захотів з другою відскануватися;
2) Освітленість: на скільки буде відхилення якщо в кімнаті де проводиться сканування більш або мент освітлене та як на це по відсоткам відреагує алгоритм;
3) Нахил голови до камери: на скільки буде відхилення по відсоткам якщо камера в теорії буде не по центру для сканування;

4.2 Аналіз результатів під час тестування алгоритмів розпізнавання обличчя

1) Зачіска;

Як було видно на фото, в мене вся зачіска була за кадром, а тепер можно зробити тест наскільки в мене буде схожість з еталонним фото якщо буде інша зачіска яка хоч якось буде змінювати вигляд. Починаю запуск програми але так як вже є кеш-файл то запуск буде швидшим (рис. 4.4). Спочатку воно буде показувати що подібність була нижче 65% але з часом воно почало бачити мене в жовтій зоні і згодом надало доступ.

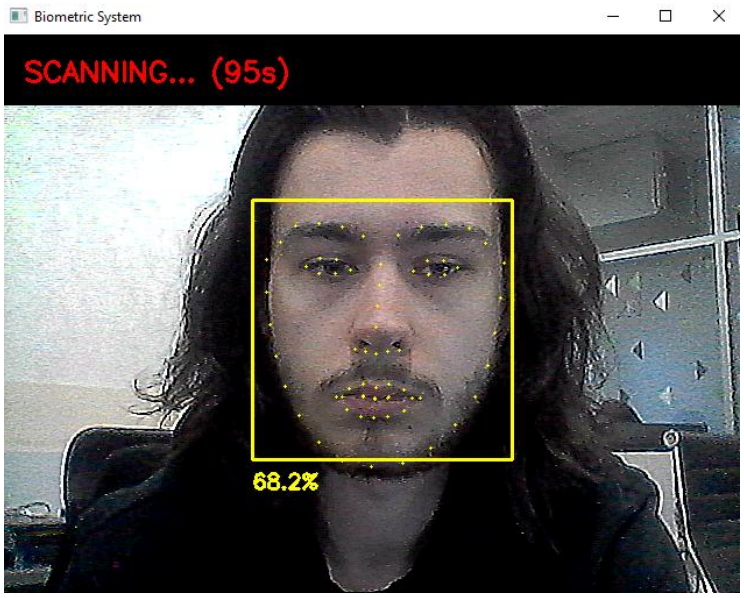


Рис. 4.4 Сканування обличчя з іншою зачіскою

Поводження алгоритму в цьому випадку демонструє правильну роботу адаптивного порогу довіри. Невеликі зміни у зовнішності не повинні призводити до блокування автомобіля, і система з цим справилась. Затримка перед авторизацією була очікуваною, оскільки частина біометричних маркерів змінилася, але основні риси обличчя залишились впізнаваними.

2) Освітленість

Наступна серія тестів була присвячена роботі системи при зміні освітлення. Перевірка проходила у двох протилежних умовах: майже повна темрява та надмірно яскраве світло. У ситуації з надлишком світла камера отримувала засвічені ділянки, що зменшувало кількість видимих дрібних рис обличчя. Незважаючи на це, алгоритм все ж знаходив характерні точки і обчислював подібність на прийнятному рівні. Система без помітних труднощів дозволила авторизацію (рис. 4.5).

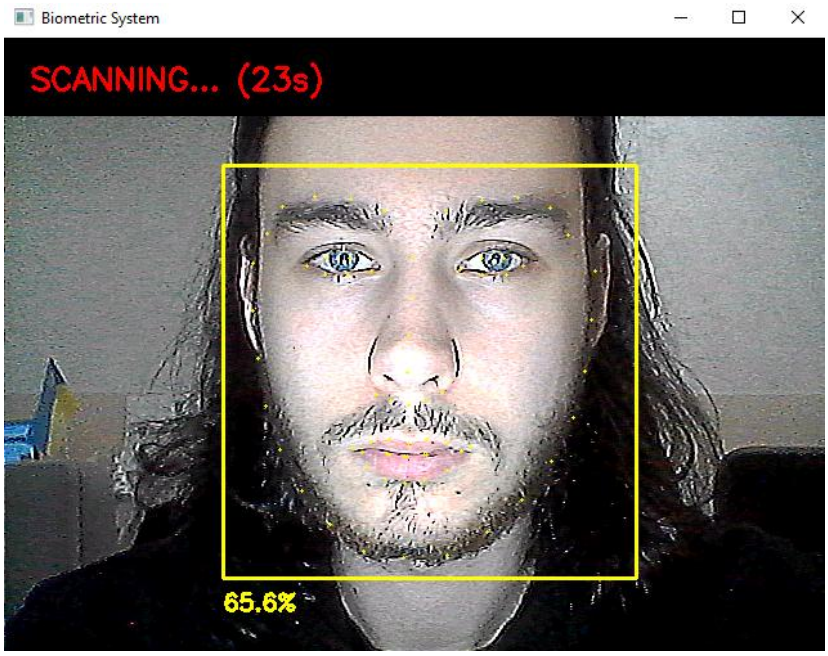


Рис. 4.5 Сканування з більшою кількістю світла

У темному приміщенні ситуація була складнішою: контур обличчя зливався з фоном, а очі й носогубна частина були менш контрастними. Значення схожості значно коливались і тривалий час утримувалися нижче порогових значень. У таких умовах алгоритм спрацював менш упевнено, що вказує на обмеження стандартних

моделей розпізнавання, які залежать від достатньої кількості світла. Це типовий результат для рішень, що працюють без інфрачервоної підсвітки або без глибинних камер.

3) Кут нахилу обличчя

Наступна перевірка полягає в тому щоб під час сканування повертати обличчя та подивись на результати які він надасть. Під час тестування було виявлено, що хоч в папці з зображеннями є фото обличчя з різних сторін але відсоток був в діапазоні від 58% до 63% незалежно від сторони чи кута під яким відбувалось сканування. Через це після проходження відліку в 90 секунд програма видає що «Доступ відхилено». Але помірний нахил голови, близько 20–25 градусів, навпаки, не створював критичних проблем. Алгоритм швидко адаптувався й успішно проходив кілька перевірок поспіль. Це показує, що система добре працює у звичайних умовах, але має природні обмеження при різких змінах пози (рис. 4.6).

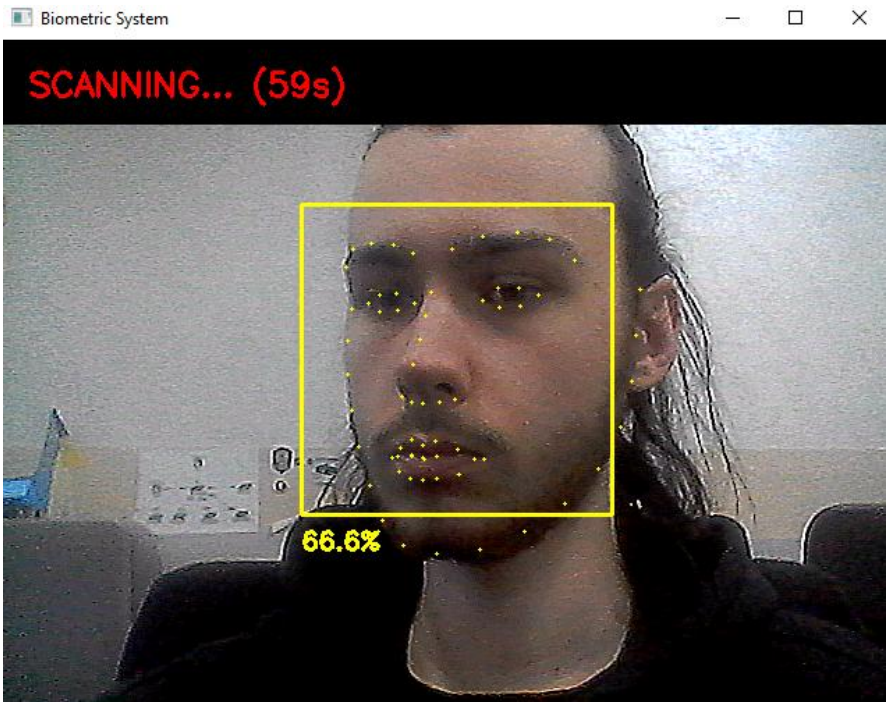


Рис. 4.6 Сканування під нахилом в сторону

Проведені експерименти підтверджують працездатність розробленого алгоритму та його придатність для використання в прототипі автомобільної системи авторизації. Вибраний підхід із порогами довіри дозволяє знаходити

компроміс між безпекою та зручністю, а також робить систему адаптивною до змін зовнішності користувача. Разом із цим тестування виявило низку зон для подальшого вдосконалення.

Зокрема, система демонструє чутливість до сильних змін освітлення та великих кутів повороту голови. Для вирішення цих проблем можливе розширення бібліотеки еталонних зображень, впровадження додаткової обробки кадрів (нормалізація яскравості, алгоритми вирівнювання обличчя) або перехід до більш складних нейромережевих моделей, які здатні працювати з менш чіткими зображеннями. Використання камер із глибинними датчиками також могло б знизити залежність від освітлення.

Таким чином, створена система є функціональною, логічно побудованою та демонструє поведінку, наближену до алгоритмів, які застосовуються у справжніх системах доступу. Вона вже придатна для практичного використання, але водночас має великий потенціал для подальшого розвитку, що робить її хорошою основою для майбутніх вдосконалень.

ВИСНОВКИ

У ході виконання дипломної роботи було проведено комплексне дослідження та розробку біометричної системи авторизації водія, яка покликана вирішити проблему недостатньої надійності традиційних засобів захисту автомобіля. Аналіз еволюції охоронних систем показав, що сучасні методи викрадення, такі як ретрансляція сигналу ключа або злам кодів іммобілайзера, роблять класичні електронні та механічні засоби вразливими. Було визначено, що найбільш ефективним шляхом підвищення безпеки є перехід від ідентифікації наявності ключа до безпосередньої аутентифікації особи водія за допомогою біометричних технологій.

В рамках практичної частини було спроєктовано та обґрунтовано архітектуру автономної системи на базі одноплатного комп'ютера Raspberry Pi 4 Model B. Використання концепції периферійних обчислень (Edge Computing) дозволило створити пристрій, який обробляє дані локально, без залежності від хмарних сервісів та інтернету, що є критичним для стабільності та приватності в автомобільній сфері . Розроблена принципова схема, що включає стабілізацію живлення, гальванічну розв'язку та релейне керування, забезпечує безпечну інтеграцію пристрою в електричне коло запалювання автомобіля.

Програмна реалізація, виконана мовою Python із використанням бібліотек комп'ютерного зору. Це забезпечує гнучкість системи та можливість її масштабування. Впроваджені алгоритми попередньої обробки зображення, адаптивні пороги довіри та механізми згладжування дозволили досягти стабільного розпізнавання обличчя в реальному часі, мінімізувавши навантаження на процесор.

Результати тестування підтвердили працездатність розробленого прототипу. Система успішно ідентифікує водія та надає дозвіл на запуск двигуна навіть за наявності незначних змін у зовнішності, таких як зміна зачіски . Були виявлені фактори які визначають напрямки для подальшого вдосконалення пристрою, зокрема через використання інфрачервоних камер.

ПЕРЕЛІК ПОСИЛАНЬ

1. Abadi M. et al. TensorFlow: A System for Large-Scale Machine Learning. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16). 2016. P. 265–283. URL: https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf (дата звернення: 04.12.2025).
2. Ahmad I. et al. Security and Privacy Challenges in Connected Vehicles: A Review. IEEE Access. 2021. Vol. 9. P. 150688–150711. URL: https://ieeexplore.ieee.org/document/9585375 (дата звернення: 04.12.2025).
3. Amos B., Ludwiczuk B., Satyanarayanan M. OpenFace: A general-purpose face recognition library with mobile applications. CMU School of Computer Science. 2016. URL: http://reports-archive.adm.cs.cmu.edu/anon/2016/CMU-CS-16-118.pdf (дата звернення: 04.12.2025).
4. Boulkenafet Z., Komulainen J., Hadid A. Face Spoofing Detection Using Colour Texture Analysis. IEEE Transactions on Information Forensics and Security. 2017. Vol. 11, no. 8. P. 1818–1830. URL: https://ieeexplore.ieee.org/document/7358784 (дата звернення: 04.12.2025).
5. Chen S. et al. MobileFaceNets: Efficient CNNs for Accurate Real-Time Face Verification on Mobile Devices. Chinese Conference on Biometric Recognition (CCBR). Springer, 2018. P. 428–438. URL: https://link.springer.com/chapter/10.1007/978-3-319-97909-0_46 (дата звернення: 04.12.2025).
6. Chollet F. Xception: Deep Learning with Depthwise Separable Convolutions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017. P. 1251–1258. URL: https://openaccess.thecvf.com/content_cvpr_2017/papers/Chollet_Xception_Deep_Learning_CVPR_2017_paper.pdf (дата звернення: 04.12.2025).
7. Deng J., Guo J., Xue N., Zafeiriou S. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2019. Vol. 41, no. 5. P. 2562–2575. URL: https://ieeexplore.ieee.org/document/8634938 (дата звернення: 04.12.2025).
8. Galbally J., Marcel S., Fierrez J. Biometric Antispoofing Methods: A Survey in Face Recognition. IEEE Access. 2014. Vol. 2. P. 1530–1552. URL: https://ieeexplore.ieee.org/document/6994829 (дата звернення: 04.12.2025).
9. Hussain R., Zeadally S. Autonomous Cars: Research Results, Issues, and Future Challenges. IEEE Communications Surveys & Tutorials. 2019. Vol. 21, no. 2. P.

1275–1313. URL: https://ieeexplore.ieee.org/document/8457076 (дата звернення: 04.12.2025).
10. ISO/IEC 18013-5:2021. Personal identification - ISO-compliant driving licence - Part 5: Mobile driving licence (mDL) application. International Organization for Standardization, 2021. URL: https://www.iso.org/standard/69084.html (дата звернення: 04.12.2025).
11. ISO/IEC 30107-3:2017. Information technology - Biometric presentation attack detection - Part 3: Testing and reporting. International Organization for Standardization, 2017. URL: https://www.iso.org/standard/67381.html (дата звернення: 04.12.2025).
12. Kazemi V., Sullivan J. One Millisecond Face Alignment with an Ensemble of Regression Trees. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2014. URL: https://www.cv-foundation.org/openaccess/content_cvpr_2014/papers/Kazemi_One_Millisecond_Face_2014_CVPR_paper.pdf (дата звернення: 04.12.2025).
13. King D. E. Dlib-ml: A Machine Learning Toolkit. Journal of Machine Learning Research. 2009. Vol. 10. P. 1755–1758. URL: https://jmlr.csail.mit.edu/papers/volume10/king09a/king09a.pdf (дата звернення: 04.12.2025).
14. Komulainen J. et al. Understanding the impact of different sources of variability on face anti-spoofing. 2016 23rd International Conference on Pattern Recognition (ICPR). IEEE, 2016. URL: https://www.researchgate.net/publication/307895926_Understanding_the_impact_of_different_sources_of_variability_on_IT_performance_during_target_search (дата звернення: 04.12.2025).
15. McKinsey & Company. The rise of edge AI in automotive. McKinsey Semiconductor Practice. 2023. URL: https://www.mckinsey.com/industries/semiconductors/our-insights/the-rise-of-edge-ai-in-automotive (дата звернення: 04.12.2025).
16. NIST. Face Recognition Vendor Test (FRVT) Part 1: Verification. National Institute of Standards and Technology. 2023. URL: https://pages.nist.gov/frvt/html/frvt1N.html (дата звернення: 04.12.2025).
17. OpenCV. OpenCV 4.x Documentation. Open Source Computer Vision Library. 2024. URL: https://docs.opencv.org/4.x/ (дата звернення: 04.12.2025).
18. Parkhi O. M., Vedaldi A., Zisserman A. Deep Face Recognition. British Machine Vision Conference. 2015. URL: https://www.robots.ox.ac.uk/~vgg/publications/2015/Parkhi15/parkhi15.pdf (дата звернення: 04.12.2025).
19. Patel K., Han H., Jain A. K. Secure Face Unlock: Spoof Detection on Smartphones. IEEE Transactions on Information Forensics and Security. 2016. Vol. 11, no. 10. P.

2268–2282. URL: https://ieeexplore.ieee.org/document/7492257 (дата звернення: 04.12.2025).
20. Python Software Foundation. Python 3 Documentation. 2024. URL: https://docs.python.org/3/ (дата звернення: 04.12.2025).
21. Raspberry Pi Foundation. Raspberry Pi 4 Model B – Product Brief. 2023. URL: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf (дата звернення: 04.12.2025).
22. Rosebrock A. Raspberry Pi Face Recognition. PyImageSearch. 2018. URL: https://pyimagesearch.com/2018/06/25/raspberry-pi-face-recognition/ (дата звернення: 04.12.2025).
23. Shao J., Zhang H. Performance Analysis of Deep Learning Inference on Raspberry Pi. Journal of Systems Architecture. 2020. Vol. 108. P. 101783. URL: https://www.google.com/search?q=https://www.sciencedirect.com/science/article/pii/S138376211930472X (дата звернення: 04.12.2025).
24. Shi W. et al. Edge Computing: Vision and Challenges. IEEE Internet of Things Journal. 2016. Vol. 3, no. 5. P. 637–646. URL: https://ieeexplore.ieee.org/document/7488250 (дата звернення: 04.12.2025).
25. Singh A. K. Machine Learning Trained Face Recognition based Automotive Ignition System. International Journal of Engineering Research & Technology (IJERT). 2020. Vol. 9, no. 5. URL: https://www.ijert.org/research/machine-learning-trained-face-recognition-based-automotive-ignition-system-IJERTV9IS040488.pdf (дата звернення: 04.12.2025).
26. Tey S. S., Pay C. L. Vehicle Anti-Theft System Using Face Recognition and Fingerprint. Journal of Physics: Conference Series. 2023. Vol. 2465. URL: https://iopscience.iop.org/article/10.1088/1742-6596/2465/1/012015 (дата звернення: 04.12.2025).
27. Uddin M. A., Islam S. Biometric Authentication for Automobiles: A Review. International Journal of Advanced Computer Science and Applications (IJACSA). 2019. Vol. 10, no. 9. URL: https://thesai.org/Publications/ViewPaper?Volume=10&Issue=9&Code=IJACSA&SerialNo=18 (дата звернення: 04.12.2025).
28. United States Patent and Trademark Office. Biometric vehicle access system patent documents. 2020. URL: https://ptacts.uspto.gov/ptacts/public-informations/petitions/1558290/download-documents?artifactId=b_uZ-7jeg6GYvFIEztqmau9KdBDTS19dS_6866pqeX9_w2szQVSXyAU (дата звернення: 04.12.2025).
29. Viola P., Jones M. Rapid Object Detection using a Boosted Cascade of Simple Features. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). 2001. Vol. 1. P. I-511. URL:

https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf (дата звернення: 04.12.2025).

30. Zhang K. et al. Joint Face Detection and Alignment Using Multi-task Cascaded Convolutional Networks. IEEE Signal Processing Letters. 2016. Vol. 23, no. 10. URL: https://www.researchgate.net/publication/352267175_Face_recognition_using_multitasking_cascading_convolutional_networks (дата звернення: 04.12.2025).

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)

Державний університет інформаційно-комунікаційних технологій

Кафедра Інформаційних систем та технологій

МАГІСТЕРСЬКА РОБОТА
на тему:
“Біометрична система авторизації водія з використанням Raspberry Pi”

Виконав: студент групи ІСДМ-61
Олександр БОВКУН

Керівник: к.т.н., доцент кафедри ІСТ
Оксана ТКАЛЕНКО

Київ - 2026

Актуальність роботи зумовлена тим, що традиційні засоби захисту — механічні замки та електронні іммобілайзери — вже не гарантують безпеки. З розвитком технологій зловмисники навчилися перехоплювати та ретранслювати сигнал ключа, що дозволяє викрасти авто за лічені хвилини.

Ідея дослідження полягає у зміні самого підходу до безпеки: система має перевіряти не наявність ключа, який можна вкрасти або скопіювати, а особистість водія.

Метою роботи є створення автономної системи на базі мікрокомп’ютера **Raspberry Pi.** Використання технології **Edge Computing** дозволяє обробляти дані локально, без передачі в інтернет. Це критично важливо, адже "розумний охоронець" має працювати миттєво і надійно, незалежно від наявності мобільного зв'язку чи серверів.

Для досягнення поставленої мети було визначено та виконано завдання:

1) Проведення аналізу сучасних методів захисту автомобілів та існуючі системи захисту.

2) Вибір компонентів для пристрою.

3) Інженерна розробка схеми пристрою, яка передбачає безпечне підключення до бортової мережі авто.

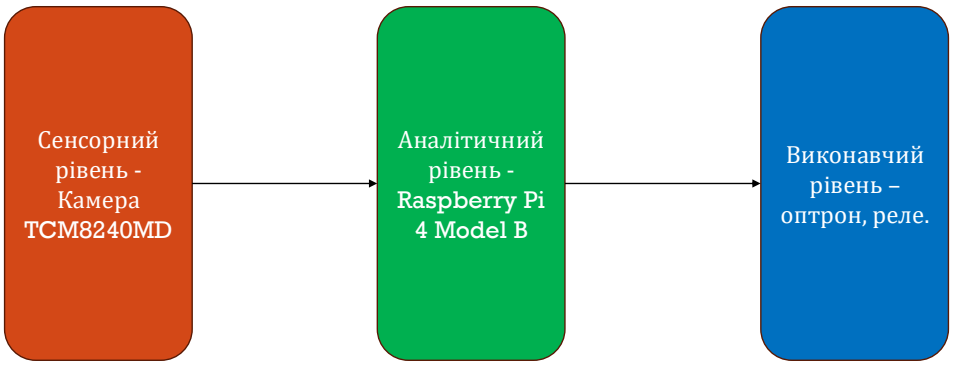
4) Написання програмного коду мовою Python, який реалізує алгоритми розпізнавання обличчя.

5) Проведення тестування системи для перевірки точності розпізнавання водія в різних умовах.

АНАЛІЗ СИСТЕМ ЗАХИСТУ

	Іммобілайзер	Радіо-брелок	Розроблена система
Захист від ретрансляції	Вразливий	Вразливий	Особистість
Захист при крадіжці ключів	Немає	Немає	Захищений
Автономність	Висока	Висока	Висока
Вартість	~ 12 500 грн	~20 500 грн	~ 5000 грн

АРХІТЕКТУРА СИСТЕМИ БІОМЕТРИЧНОЇ АВТОРИЗАЦІЇ



КОМПОНЕНТИ ПРИСТРОЮ

Raspberry Pi

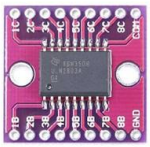
Стабілізатор живлення

Камера

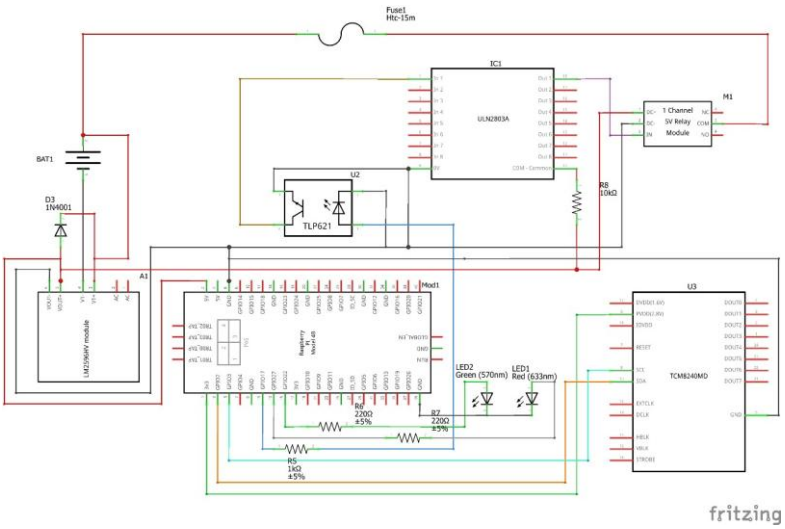
Одноканальний реле 5В

Пристрій

Драйвер ULN2803A

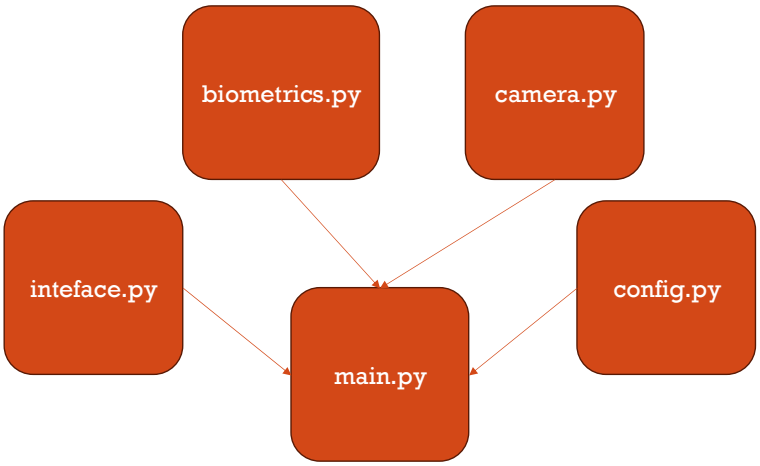


ПРИНЦИПІАЛЬНА СХЕМА ПРИСТРОЮ

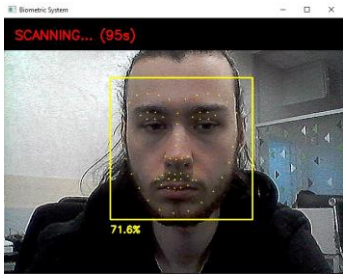


СТРУКТУРА АЛГОРИТМІВ ДЛЯ РОЗПІЗНАВАННЯ

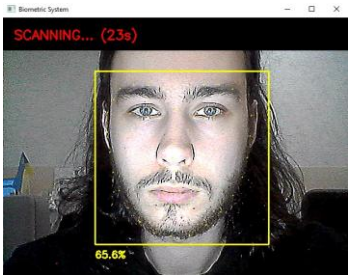
Всього було створено п’ять **python** файлів:
1) **config.py** – це налаштування параметрів для майбутнього сканування;
2) **camera.py** – ініціалізація камери та захоплення відео.
3) **biometrics.py** – розпізнавання обличчя;
4) **interface.py** – інтерфейс для камери який виводиться на екран;
5) **main.py** – відповідає за запуск розпізнавання використовуючи попередні файли.



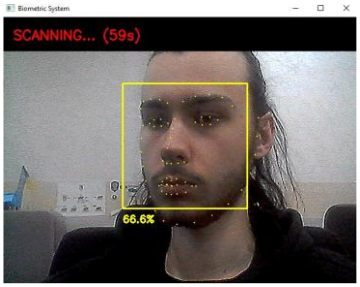
ТЕСТУВАННЯ АЛГОРИТМІВ РОЗПІЗНАВАННЯ



Звичайне сканування



Сканування з іншим освітленням



Сканування під нахилом

ВИСНОВКИ

Впровадження біометричних технологій може значно підвищити рівень захисту транспортних засобів, унеможливлюючи використання ретрансляторів сигналу ключа. У ході роботи було проведено аналіз сучасних вразливостей та розроблено автономну систему на основі мікрокомп'ютера **Raspberry Pi 4 Model B** з використанням камери **TCM8240MD**, релейного модуля, драйвера **ULN2803A** та оптрона **TLP621**. Пристрій забезпечує високу надійність та приватність даних завдяки концепції **Edge Computing**, що дозволяє обробляти інформацію локально без залежності від хмарних сервісів. Використання бібліотек **Computer Vision** та мови **Python** дозволяє виконувати точну ідентифікацію водія в реальному часі, забезпечуючи гнучкість налаштування порогів довіри.

Представлено приклад роботи готового прототипу, який успішно розпізнає власника навіть за змін зовнішності, з можливістю подальшого вдосконалення шляхом інтеграції інфрачервоних камер для роботи в нічний час або **GSM**-модуля для сповіщень.

Апробація результатів магістерської роботи

*1) Бовкун О. С. «Біометрична система авторизації водія з використанням **Raspberry Pi». Тези доповіді на III Всеукраїнській науково-технічній конференції «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і світу». – Київ, 18 листопада 2025 року.*

2) Бовкун О. С. «Розробка автономного модуля біометричної ідентифікації для системи запуску автомобіля». Тези доповіді на VIII Всеукраїнська науково- технічна конференція «Компьютерні технології: інновації, проблеми, рішення». Житомир – 02-03 грудня 2025р.

ДЯКУЮ ЗА УВАГУ!