Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Комп'ютерних інтелектуальних технологій та систем

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський) рівень

Розробка інтелектуальної системи безпеки
з використанням комп'ютерного зору
(тема)

Виконав:
студент 2 курсу, групи КІТм-20-1
Волков Д.П.
(прізвище, ініціали)

Спеціальність 123 Комп'ютерна інженерія

Тип програми освітньо-професійна

Освітня програма Комп'ютерні інтелектуальні технології

Керівник проф. Безсонов О.О.
(посада, прізвище, ініціали)

Допускається до захисту _____
(підпис)

Зав. кафедри КІТС _____ проф. Руденко О.Г.
(підпис)

2021 р.

Харківський національний університет радіоелектроніки

Факультет	Комп'ютерної інженерії та управління
Кафедра	Комп'ютерних інтелектуальних технологій та систем
Рівень вищої освіти	другий (магістерський)
Спеціальність (напрям)	123 – Комп'ютерна інженерія
Освітня програма	Комп'ютерні інтелектуальні технології

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
"____" _____ 20__ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Волкову Дмитру Павловичу
(прізвище, ім'я, по батькові)

1. Тема роботи (проекту) Розробка інтелектуальної системи безпеки з використанням комп'ютерного зору

затверджена наказом по університету від " 8 " листопада 2021р. № 1666Ст

2. Термін подання студентом роботи до екзаменаційної комісії 10 грудня 2021 р.

3. Вхідні дані до роботи Вимоги до інтелектуальної системи. Теоретичні відомості про способи роботи з компонентами системи. Огляд складових частин системи у вигляді апаратного та програмного забезпечення. Результати тестування компонентів системи.

4. Перелік питань, що потрібно опрацювати в роботі
Аналіз предметної області
Постановка задачі
Аналіз компонентів
Проектування системи
Реалізація схеми пристрою
Реалізація програмного забезпечення
Програмування пристрою
Тестування системи
Аналіз результатів роботи

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) 8 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Видача та узгодження теми проекту	08.11.2021	
2	Аналіз проблемної галузі, постановка задачі,		
	вибір інструментальних засобів	7.11.21-14.11.21	
3	Розробка моделі системи	15.11.21-18.11.21	
4	Розробка фізичної схеми пристрою	19.11.21-20.11.21	
5	Розробка програмного забезпечення	21.11.21-28.11.21	
6	Проведення випробування пристрою	29.11.21-7.12.21	
7	Оформлення пояснювальної записки	8.12.21-9.12.21	
8	Перевірка виконаного проекту керівником	10.12.2021	
9	Захист проекту	16.12.21 - 17.12.21	

Дата видачі завдання 8.11.2021 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Безсонов О.О.
(підпис) (посада, прізвище, ініціали)

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Комп'ютерних інтелектуальних технологій та систем

АНОТАЦІЯ
КВАЛІФІКАЦІЙНОЇ РОБОТИ

рівень вищої освіти другий (магістерський)

Розробка інтелектуальної системи безпеки з використанням комп'ютерного зору

(тема)

Виконав:
студент 2 курсу, групи КІТм-20-1
Волков Д.П.
(прізвище, ініціали)
Спеціальність 123 Комп'ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп'ютерні інтелектуальні технології
Керівник проф. Безсонов О.О.
(посада, прізвище, ініціали)

2021 р.

Вступ

Актуальність систем безпеки не пропадає з плином часу. За день створюється тисячі підприємств, будуються житлові будинки, банки, сховища, офіси, торгівельні зали, музеї, усе це потребує надійного захисту від зловмисників. Сучасні системи охорони модернізуються згідно з розвитком комп'ютерних технологій, на заміну старим системам відеоспостереження та бригадам охорони приходять інтелектуальні системи сигналізації, які у автономному режимі фіксують порушників та приймають усі можливі засоби для захисту території, для охорони якої вони встановлені.

Сучасний співробітник служби охорони – це літаючий робот. Дрони стали надзвичайно популярними в якості повітряних охоронців і вивели на новий рівень поняття безпеки. Безпілотні літальні апарати успішно використовуються для охорони об'єктів, територій і людей, а також для виявлення і своєчасного припинення незаконної діяльності. Сторожовий безпілотник весь час знаходиться на посаді і однаково ефективно працює в будь-який час доби.

Уже кілька років безпілотні літальні апарати успішно контролюють і охороняють стратегічно важливі об'єкти, родовища корисних копалин, промислові об'єкти, лінії передач, і т.д. Незабаром БПЛА серйозно потіснять інших гравців на ринку охоронних систем. Головний плюс БПЛА - в швидкості реагування і можливості повноцінного контролю навіть таких важкодоступних місць як даху висотних будівель. Сторожовий дрон може виконувати рутинну роботу з патрулювання периметра і регулярно здійснювати обліт територій, передаючи фото- і відеоінформацію операторам, або інтелектуальним системам, які використовують її для аналізу обстановки і прийняття рішень. Аерознімки і інша інформація, що отримується від квадрокоптера, допомагає службі охорони своєчасно припиняти протиправні дії. Заздалегідь запрограмувавши дрон слідувати певним маршрутом, можна домогтися практично повністю автономного режиму польоту. БПЛА однаково ефективні як днем, так і вночі.

Завдяки інфрачервоному зору, розумний коптер виявить порушника навіть у темряві і негайно передасть відео і фото оператору.

Дрони можуть підніматися в повітря і в разі несанкціонованого проникнення на територію, після спрацювання сигналізації, щоб провести аерофотозйомку над територією, що охороняється і передати інформацію на пульт служби охорони в режимі реального часу. Сучасні БПЛА отримують чітке зображення осіб і номерів машин навіть з великої висоти. Система автоматичного супроводу об’єкта дозволяє вести непомітне спостереження за порушником незалежно від того, чи знаходиться він в статичному положенні, пересувається пішки або на автомобілі.

Серед задач кваліфікаційної роботи є розробка інтелектуальної системи, що має функції для її використання у цілях охорони або патрулювання. Також розробка схеми взаємодії компонентів, їх варіації для використання у системі, порівняння розробленої схеми з аналогами.

Потрібно підібрати або розробити програмне забезпечення для пристроїв які включені до схеми, підібрати технології які дозволять вибраним пристроям виконувати задані функції охоронної системи.

Також потрібно розглянути вимоги до охоронних систем, їх типи та потрібний функціонал, для перенесення його до пристрою при розробці програмного забезпечення.

На сьогодні тезнології комп’ютерного зору являються найперспективнішими для будь якої галузі і насамперед для задач авіації. Завдяки цьому існує велика кількіст технології за допомогою яких є можливість створення охоронних дронів.

Основна частина

В даний час дуже багато охоронних систем передають повідомлення про проникнення або позаштатних ситуаціях по різних каналах зв'язку, які

передбачають передачу інформації через взаємопов’язану мережу зв’язку Росії та України. При цьому можуть використовуватися провідні канали зв’язку, оптичні, бездротовий зв’язок. Для цього можуть використовуватися такі зв'язкові протоколи, як TCPIP, PSTN, GSM і інші технології зв’язку.

Для таких модулів може знадобитися процедура обов’язкового підтвердження відповідності в зв'язку, або у формі декларування, або у формі обов’язкової сертифікації. Декларація або сертифікат в області Зв’язки підтверджують, що охоронне устаткування не порушить цілісність і стійкість функціонування мережі зв’язку загального користування.

Обладнання для охоронних систем, систем контролю доступу та інших технічних охоронних засобів, які розміщені в офісах або житлових приміщеннях, повинно пройти санітарно-епідеміологічну експертизу. На це охоронне устаткування повинні бути оформлені санітарно-епідеміологічні висновки.

На практиці охоронні системи використовуються дуже широко. Найчастіше варіації охоронних систем розроблюють під конкретну задачу, враховуючи умови території на якій вона буде працювати. Але використання дронів змінює ситуацію кардинально. Охоронна система становиться універсальною, стійкою та автоматизованою . А використання технологій комп’ютерного зору ще більш покращує систему, видаляючи людський фактор неуважності, тим самим збільшує її ефективність.

Комп’ютерний зір навчає машини виконувати ці функції, але він повинен робити це за набагато менше часу за допомогою камер, даних і алгоритмів, а не сітківки, зорових нервів і зорової кори. Оскільки система, навчена перевіряти продукти або спостерігати за виробничим активом, може аналізувати тисячі продуктів або процесів за хвилину, помічаючи непомітні дефекти або проблеми, вона може швидко перевершити людські можливості.

TensorFlow Lite для мікроконтролерів розроблено для запуску моделей машинного навчання на мікроконтролерах та інших пристроях з лише кількома кілобайтами пам’яті. Основне середовище виконання вміщається в 16 КБ на Arm Cortex M3 і може працювати з багатьма базовими моделями. Він не вимагає

підтримки операційної системи, будь-яких стандартних бібліотек C або C++ або динамічного виділення пам’яті.

Мікроконтролери, як правило, є невеликими, малопотужними обчислювальними пристроями, які вбудовані в апаратне забезпечення, яке вимагає базових обчислень. Додавши машинне навчання до крихітних мікроконтролерів, можна підвищити інтелект мільярдів пристроїв, які використовуються в нашому житті, включаючи побутову техніку та пристрої Інтернету речей, не покладаючись на дороге обладнання чи надійне інтернет-з’єднання, яке часто залежить від пропускної здатності та обмеження потужності і призводить до високої затримки. Це також може допомогти зберегти конфіденційність, оскільки дані не залишають пристрій. Уявіть собі розумні прилади, які можуть адаптуватися до вашої повсякденної роботи, інтелектуальні промислові датчики, які розуміють різницю між проблемами та нормальною роботою, а також чарівні іграшки, які можуть допомогти дітям вчитися веселим і захоплюючим способом.

TensorFlow Lite для мікроконтролерів написаний на C++ 11 і вимагає 32-розрядної платформи. Він був ретельно протестований з багатьма процесорами, заснованими на архітектурі серії Arm Cortex-M, і був портований на інші архітектуриA, включаючи ESP32. Фреймворк доступний як бібліотека Arduino. Він також може створювати проекти для середовищ розробки, таких як Mbed. Він є відкритим вихідним кодом і може бути включений до будь-якого проекту C++ 11.

Мікроконтролери мають обмежену оперативну пам’ять і сховище, що накладає обмеження на розміри моделей машинного навчання. Крім того, TensorFlow Lite для мікроконтролерів наразі підтримує обмежену підмножину операцій, тому не всі архітектури моделей можливі.

Щоб перетворити навчену модель TensorFlow для роботи на мікроконтролерах, вам слід використовувати API конвертера TensorFlow Lite Python. Це перетворить модель у FlatBuffer, зменшивши розмір моделі та змінить її для використання операцій TensorFlow Lite. Щоб отримати найменший

можливий розмір моделі, слід розглянути можливість використання квантування після навчання.

Для виконання поставлених задач кваліфікаційної роботи була розроблена схема функціонування охоронного пристрою , а також запрограмований мікроконтролерний модуль ESP32 Cam з використанням технології Tensorflow.

Висновки

При виконанні кваліфікаційної роботи була розроблена схема пристрою, яка виконує функції інтелектуальної системи охорони. Були розглянуті можливі компоненти для розробки, їх переваги та недоліки. Також були визначені технології за допомогою яких можливо реалізувати технологію комп'ютерного зору на базі мікроконтролера ESP32 Cam, а саме технологія TensorFlow від Google. Також була представлена модель пристрою у вигляді літаючого дрона, що буде виконувати функції переміщення основного мікроконтролера з комп'ютерним зором до точок сканування місцевості на можливі погрози. Було розглянуто аналогічні схожі за концептом проекти.

Список використаних джерел

Волков Д. П. Інтелектуальні системи у задачах навігації безпілотних літальних апаратів : тези VII всеукр. наук.-практ. конф. студентів, аспірантів та молодих вчених. м. Харків, 2020. С. 184–186.

Волков Д. П. Інтелектуальна система безпеки з використанням комп'ютерного зору : тези XXV міжнар. молод. форуму *Радіоелектроніка і молодь у XXI столітті*. конф. *Комп'ютерні системи і мережі управління та обробки даних*. м.Харків, 2021.

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 76 с., 27 рис., 1 дод., 15 джерел.

ВІДЕОСПОСТЕРЕЖЕННЯ, СИСТЕМА БЕЗПЕКИ, ОБРОБКА ЗОБРАЖЕНЬ, ESP32, TENSORFLOW.

Метою кваліфікаційної роботи є розробка інтелектуальної системи безпеки з використанням комп’ютерного зору. Огляд та порівняння подібних існуючих систем та розробка схеми власної системи.

У ході виконання кваліфікаційної роботи потрібно розробити схему функціонування сучасної системи безпеки для відстеження загроз на основі технологій комп’ютерного зору, підібрати компоненти системи, реалізувати схему пристрою, розробити програмне забезпечення та провести тестування програмного забезпечення.

ABSTRACT

Qualification project: 76 pages, 27 figures, 1 appendices, 15 sources.

VIDEO SURVEILLANCE, SECURITY SYSTEM, IMAGE PROCESSING, ESP32, TENSORFLOW.

The purpose of qualification project is to develop an intelligent security system using computer vision. Review and comparison of similar existing systems and development of the scheme of own system.

During the qualification project, it is necessary to develop a scheme of operation of a modern security system for tracking threats based on computer vision technologies, select system components, implement a device scheme, develop and test software.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БПЛА (БЛА) – Безпілотний літальний аппарат

CNN – Convolutional neural network

FTDI – Future Technology Devices International

FPC – Flexible Printed Circuit

GSM – Global System for Mobile

I2C – Inter-Integrated Circuit

ML – Machine learning

PWM – Pulse-width modulation

PSTN – Public switched telephone network

RSSI – Received Signal Strength Indicator

SMA – SubMiniature version A

SPI – Serial Peripheral Interface

TCP/IP – Transmission Control Protocol/Internet Protocol

TTL – Time to live

UART – Universal Asynchronous Receiver-Transmitter

ВСТУП

Актуальність систем безпеки не пропадає з плином часу. За день створюється тисячі підприємств, будуються житлові будинки, банки, сховища, офіси, торгівельні зали, музеї, усе це потребує надійного захисту від зловмисників. Сучасні системи охорони модернізуються згідно з розвитком комп'ютерних технологій, на заміну старим системам відеоспостереження та бригадам охорони приходять інтелектуальні системи сигналізації, які у автономному режимі фіксують порушників та приймають усі можливі засоби для захисту території, для охорони якої вони встановлені.

Охоронна сигналізація необхідна для запобігання або своєчасного реагування на надзвичайні події: пожежу, потоп, несанкціоноване проникнення на об'єкт приватної або комерційної власності. Функціональність охоронного обладнання безпосередньо залежить від комплексу придбаних датчиків. Існують пристрої, які реагують на тепло або інфрачервоне випромінювання (протипожежні), рух, відкриття дверей або вікон, розбивання скла, шум. Кожен датчик з'єднується з центральним пультом, розташованим на об'єкті. Після спрацьовування одного з пристроїв пульт автоматично подає сигнал в центр реагування вашого охоронного підприємства.

На даний час користується популярністю комплекс пристроїв, які з'єднуються між собою за допомогою GSM або Wi-Fi. Відсутність проводів не дозволяє вивести систему з ладу. Всі прилади оснащуються незалежними батареями, які здатні працювати тривалий час в автономному режимі.

Дрони вже ширше використовуються в індустрії безпеки, ніж багато хто міг уявити, якщо наше опитування є барометром. Шістнадцять відсотків респондентів, здебільшого професіонали з безпеки, керівники служби безпеки та інші топ-менеджери стверджують, що вже впроваджують цю технологію. Троє з п'яти (60%) або вже використовують дрони, або можуть передбачити, що з часом це зроблять, тому не здається гіперболічним описувати траєкторію зростання цього ринку як стрімке зростання.

Різні компанії активно займають ринок безпілотних апаратів, вони конкурують одна з одною за найкращу якість продукції, при цьому активно розробляючи різні види програмного забезпечення, модернізують бортову начинку свої апаратів, додають з кожною ітерацією версій продукту нові типи технологій, але забувають про головну річ, це – доступність. Не кожен зможе дозволити собі мати у озброєнні автоматичного дрона , який зможе замінити цілий комплекс, такого ж коштовного, охоронного устаткування. Тому виникає потреба у створенні більш доступних для суспільства автоматичних літальних апаратів.

Ринок мікроконтролерів дає можливість обрати будь яку плату, для будь яких задач, у тому числі для задач безпілотної авіації. Зараз створено купа якісних модульних датчиків, які за бажанням можна включати до існуючих систем, перепрограмувавши дані системи на підтримку нового функціоналу.

Усе це дозволяє створити пристрій який не буде уступати за функціоналом коштовним проектам, якими займаються великі гравці ринку безпілотної авіації.

1 АНАЛІЗ ЗАВДАННЯ

1.1 Огляд особливостей систем безпеки

Сучасний співробітник служби охорони – це літаючий робот. Дрони стали надзвичайно популярними в якості повітряних охоронців і вивели на новий рівень поняття безпеки. Безпілотні літальні апарати успішно використовуються для охорони об’єктів, територій і людей, а також для виявлення і своєчасного припинення незаконної діяльності. Сторожовий безпілотник весь час знаходиться на посаді і однаково ефективно працює в будь-який час доби.

Уже кілька років безпілотні літальні апарати успішно контролюють і охороняють стратегічно важливі об’єкти, родовища корисних копалин, промислові об’єкти, лінії передач, і т.д. Незабаром БПЛА серйозно потіснять інших гравців на ринку охоронних систем. Головний плюс БПЛА в швидкості реагування і можливості повноцінного контролю навіть таких важкодоступних місць як даху висотних будівель. Сторожовий дрон може виконувати рутинну роботу з патрулювання периметра і регулярно здійснювати обліт територій, передаючи фото- і відеоінформацію операторам, або інтелектуальним системам, які використовують її для аналізу обстановки і прийняття рішень. Аерознімки і інша інформація, що отримується від квадрокоптера, допомагає службі охорони своєчасно припиняти протиправні дії. Заздалегідь запрограмувавши дрон слідувати певним маршрутом, можна домогтися практично повністю автономного режиму польоту. БПЛА однаково ефективні як днем, так і вночі. Завдяки інфрачервоному зору, розумний коптер виявить порушника навіть у темряві і негайно передасть відео і фото оператору.

Дрони можуть підніматися в повітря і в разі несанкціонованого проникнення на територію, після спрацювання сигналізації, щоб провести аерофотозйомку над територією, що охороняється і передати інформацію на пульт служби охорони в режимі реального часу. Сучасні БПЛА отримують чітке зображення осіб і номерів машин навіть з великої висоти. Система

автоматичного супроводу об’єкта дозволяє вести непомітне спостереження за порушником незалежно від того, чи знаходиться він в статичному положенні, пересувається пішки або на автомобілі [1].

Економічно вигідно використання БПЛА для патрулювання протяжних територій. Для контролю великої охороняється площі потрібно щодня проїжджати десятки кілометрів на автомобілі, а маленькі дрони виконають цю роботу в лічені хвилини, що допоможе значно скоротити штат, заощадити час і гроші [1].

Переваги використання БПЛА:

а) роботизовані дрони не схильні до людських факторів (людські помилки), таких як неуважність, сонливість (потреба у сні), втома, почуття лінощів, почуття страху (страх отримати шкоду своєму організму при патрулюванні території), наявність шкідливих звичок (куріння, алкоголь, наркотики), хабарі (впускати на об’єкт сторонніх людей за певну плату), хворобу та нездужання, саме тому застосування охоронних роботизованих дронів дозволить виключити всі фактори та помилки людини, що дозволить досягти максимальної захищеності території;

б) роботизовані дрони - є разовим фінансуванням, персоналу ж необхідні щомісячні виплати заробітної плати, виділення соціального пакету, відлучки за сімейними обставинами чи хворобою;

в) роботизовані охоронні дрони - не закладено порушувати чи вчиняти протиправних дій, не вміють виробляти і шукати особисту вигоду, а персонал може і сам зробити крадіжку, і посприяти крадіжці з підприємства;

г) отримання персоналом при несенні служби та виконанні службових обов’язків ушкоджень, поранень, травм або найстрашніше – смерть персоналу внаслідок нападу злочинців (грабіжників), нести за це моральну, матеріальну, і навіть, можливо, кримінальну відповідальність буде роботодавець персоналу; вихід же з ладу роботизованого охоронного дрона в результаті дій злочинних елементів є не більше ніж матеріальною втратою, що не важко повернути в дію;

д) швидкість розпізнавання та швидкість оповіщення на небезпечні, а також тривожні ситуації у роботизованого охоронного дрона на порядок вища і непорівнянна зі швидкістю реакції середньостатистичної людини, якою є охоронець, нехай навіть ліцензований.

Основним недоліком роботизованих охоронних дронів є відсутність функції логічного судження (дана проблема вирішується за рахунок впровадження в логічний ланцюжок нейронних мереж), а також обмеження видимості датчиків. Крім того, ціна інтелектуальних роботів дуже висока, особливо для нашої країни, адже вся закордонна продукція приходить до нас або з обмеженим функціоналом або з високою вартістю (що відрізняється від зарубіжної ціни).

Однак, від року в рік можливості комп'ютерів, роботів, датчиків та іншої техніки, а ціна їх падає, але і це не є плюсом, так як у зв'язку з старінням старої техніки, за нею йде її потреба.

Мета створення та впровадження роботизованих охоронних дронів:

а) створення цілодобового контролю території;

б) створення роботизованої системи патрулювання;

в) створення алгоритмічного ланцюга взаємозв'язку дронів;

г) створення штучного інтелекту на основі нейронних систем для впровадження в охоронні дрони;

д) розробити та впровадити систему патрулювання території.

Розглянемо застосування роботизованого охоронного дрона, створеного з урахуванням мікропроцесора, у сфері охоронних систем. Важливими перевагами такого дрона є можливість програмування складної поведінки та простого підключення безлічі зовнішніх пристроїв.

Розглянемо, що конкретно слід удосконалити і модернізувати в дроні, щоб успішно використовувати його як робота-охоронця:

а) необхідно передбачити можливість підключення до дрону не тільки відеокамери, але також інших датчиків, що застосовуються в охоронній сфері: датчиків руху, тепла, світла , задимлення, вітру тощо;

б) розробники охоронних систем оцінили б можливість ведення статистики показань датчиків, щоб у разі відхилення від цієї статистики робот транслював сигнал тривоги (наприклад, якщо робот їздить повз шумні приміщення і тихі приміщення, то при виявленні шуму в тихому приміщенні відразу зрозуміло, що не все в порядку);

в) для забезпечення більшої міри надійності має бути можливим використання кількох роботів в тому самому будинку (поведінковий модуль повинен передбачати, щоб вони не стикалися, працювали в різних діапазонах частот і т. д.);

г) повинна бути можливість хаотичного руху робота між зазначеними йому ключовими точками, для того, щоб збити злочинця з пантелику, тому що регулярні дії дають злочинцеві стабільний час між двома проходами охоронцем одного приміщення, на проникнення в периметр, що охороняється. У поєднанні з попереднім пунктом, тобто використанням кількох роботів, надійність охорони буде дуже високою.

1.2 Мета та задачі кваліфікаційної роботи

Метою кваліфікаційної роботи є розробка сучасної системи безпеки з використанням технологій комп’ютерного зору та інтелектуальних систем розпізнавання об’єктів, розробка схеми функціонування інтелектуальної системи та порівняння з подібними існуючими системами.

Прикладом існуючої системи є дрон Sparrow від ізраільського стартапу Percepto (рис 1.1).

Інтелектуальну систему охорони можна представити у вигляді наземного або повітряного дрона який виконує функції патрулювання, за допомогою комп’ютерного зору відстежує ситуації, при яких виконує комплекс дій для запобігання небезпеки, як наприклад увімкнення сигналізації, сповіщення охорони, тощо.

Рисунок 1.1 – Безпілотний дрон Sparrow

Серед задач кваліфікаційної роботи є розробка інтелектуальної системи, що має функції для її використання у цілях охорони або патрулювання. Також розробка схеми взаємодії компонентів, їх варіації для використання у системі, порівняння розробленої схеми з аналогами.

Потрібно підібрати або розробити програмне забезпечення для пристроїв які включені до схеми, підібрати технології які дозволять вибраним пристроям виконувати задані функції охоронної системи.

Також потрібно розглянути вимоги до охоронних систем, їх типи та потрібний функціонал, для перенесення його до пристрою при розробці програмного забезпечення.

Потрібно провести порівняння з аналогічними створеними системами на ринку, зробити огляд компонентів та технологій які будуть використані при розробці. Виявити усі переваги та недоліки компонентів та системи у цілому.

Однією із задач кваліфікаційної роботи є зменшення розмірів подібної системи для її використання у житлових приміщеннях, офісах, складах або музеях.

2 ОПИС ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ БЕЗПЕКИ

2.1 Вимоги до системи безпеки

Сертифікація охоронних систем – обов’язкова процедура згідно з Переліком продукції, що підлягає обов’язковій сертифікації в Україні, затвердженого Наказом Держспоживстандарту України від 01.02.2005 №28 [2].

Електронні та радіоелектронні системи, що виконують функції охорони, сигналізації, контролю доступу, відеоспостереження підлягають обов’язковій сертифікації в системі сертифікації «УкрСЕПРО». При обов’язковій сертифікації підтверджуються обов’язкові вимоги щодо безпеки та електромагнітної сумісності виробів. На частину обладнання для систем відеоспостереження так само існують вимоги, що підтверджують вимоги вандалостійкості [2].

В даний час дуже багато охоронних систем передають повідомлення про проникнення або позаштатних ситуаціях по різних каналах зв’язку, які передбачають передачу інформації через взаємопов’язану мережу зв’язку Росії та України. При цьому можуть використовуватися провідні канали зв’язку, оптичні, бездротовий зв’язок. Для цього можуть використовуватися такі зв’язкові протоколи, як TCPIP, PSTN, GSM і інші технології зв’язку.

Для таких модулів може знадобитися процедура обов’язкового підтвердження відповідності в зв’язку - або у формі декларування, або у формі обов'язкової сертифікації. Декларація або сертифікат в області Зв’язки підтверджують, що охоронне устаткування не порушить цілісність і стійкість функціонування мережі зв’язку загального користування.

Обладнання для охоронних систем, систем контролю доступу та інших технічних охоронних засобів, які розміщені в офісах або житлових приміщеннях, повинно пройти санітарно-епідеміологічну експертизу. На це охоронне устаткування повинні бути оформлені санітарно-епідеміологічні висновки.

У грудні 2019 року Міжнародна організація зі стандартизації (ISO) оголосила про перші в світі затверджені ISO стандарти дронів після 12-місячного періоду консультацій з професіоналами, науковцями, підприємствами та широкою громадськістю. Очікується, що остаточна публікація матиме величезний вплив на майбутнє зростання світової індустрії дронів.

Цей важливий перший крок є частиною ширшого результату ISO, який, як очікується, спричинить швидке прискорення використання повітряних дронів організаціями, які прагнуть пожинати плоди цієї трансформаційної технології, на тлі запевнень щодо безпеки та безпеки в нових рамках. затвердженої відповідності нормативним вимогам [2].

Оголошення ISO являє собою величезний прогрес у стандартизації світової індустрії дронів і має особливе значення для задоволення експлуатаційних вимог більш відомих і поширених повітряних дронів, також відомих як UAS (системи безпілотних літаків). Нові стандарти включають протоколи щодо якості, безпеки, безпеки та загального «етикету» експлуатації комерційних повітряних дронів, які допоможуть сформувати майбутнє регулювання та законодавство.

Ключовою ознакою стандартів ISO є їх зосередженість на безпеці польотів, яка знаходиться в центрі уваги громадськості у зв’язку з аеропортами та іншими чутливими місцями. Нові стандарти пропагують «етикет» використання дронів, який посилює дотримання заборонених для польотів зон, місцевих правил, протоколів журналів польотів, документації з технічного обслуговування, навчання та планування польотів. Ефективність Стандартів у підвищенні безпеки повітряного транспорту буде ще більше посилена за рахунок постійного швидкого розвитку технології геоогородження та боротьби з безпілотниками, що забезпечують захист на передовій від «негідних» операторів дронів [2].

Стандарти також спрямовані на вирішення проблем громадськості, пов’язаних із конфіденційністю та захистом даних, вимагаючи, щоб оператори мали відповідні системи для обробки даних поряд із зв’язком та плануванням контролю під час польотів. Апаратне та програмне забезпечення всього відповідного операційного обладнання також має оновлюватися. Важливо, що

безвідмовне втручання людини необхідне для всіх польотів безпілотників, включаючи автономні операції, гарантуючи притягнення до відповідальності операторів дронів.

2.2 Схема програмно-апаратних засобів

Основним компонентом для розробки системи безпеки є мікроконтролерний модуль ESP32 Cam (Рис 2.1).

Рисунок 2.1 – Модуль ESP32 Cam

ESP32-CAM - плата для розробки ESP32 (рис. 2.3). Потужний двоядерний 32 бітний процесор Xtensa з частотою від 80 до 240 Мгц. Модуль від Espressif. Застосовується в якості камери спостереження. Систем розпізнавання осіб, жестів, QR кодів.

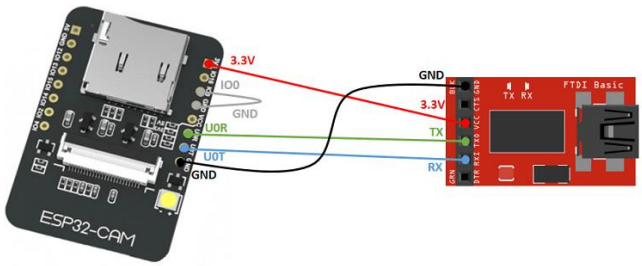

Рисунок 2.2 – Схема підключення програматора FTDI

Для програмування ESP32 Cam використовують FTDI програматор (Рис 2.2).

Характеристики:

а) WiFi 802.11b/g/n;

б) Bluetooth 4.2 LE з друкованою антеною, роз’єм u.FL.(IPX13);

в) 32 Мбіт SPI флеш-пам’ять;

г) 4 Мбіт PSRAM;

д) слот для мікро SD-карти до 4 Гб;

е) габаритні розміри - 40.5 х 27 х 4.5 мм;

ж) вага - 10 гр.

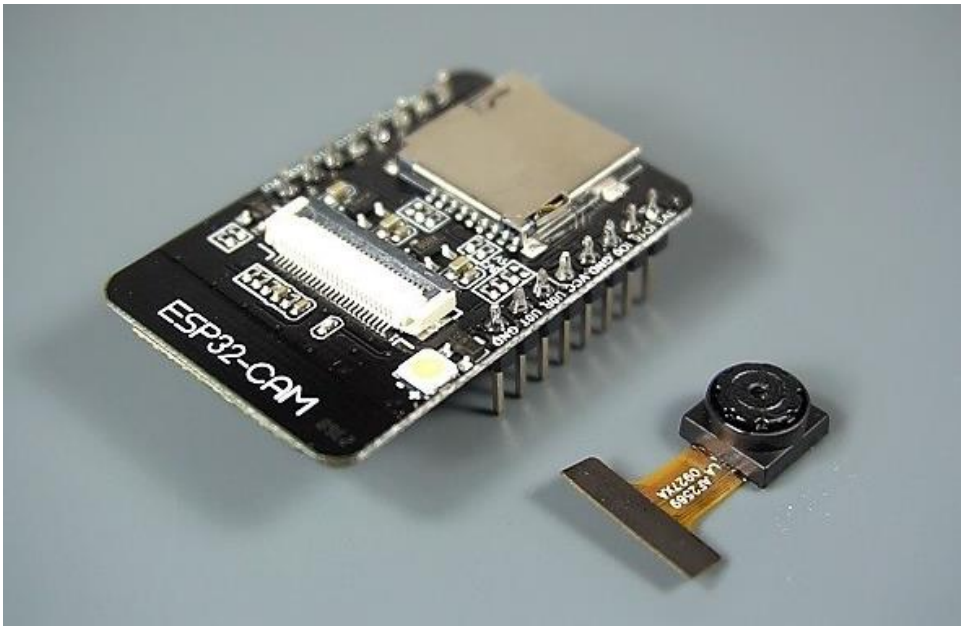

Рисунок 2.3 – Модуль ESP32 Cam та камера AI THINKER

Камера:

а) роз'єм FPC;

б) підтримка камер OV2640, OV7670;

в) формат зображення - JPEG (підтримує тільки OV2640), BMP, відтінки сірого;

г) світлодіодна підсвітка

д) розширення - UART, SPI, I2C, PWM;

е) кнопка скидання;

ж) живлення - 5 вольт через контактний роз’єм.

з) споживана потужність:

1) світлодіодне підсвічування вимкнена - 180 мА;
2) світлодіодне підсвічування включена (макс. Яскравість) - 310 мА;

и) Deep-sleep - 6 мА;

к) Modem-sleep - 20 мА;

л) Light-sleep - 6.7 мА.

CP2102 (рис. 2.4) - це USB-UART перетворювач (USB to UART Bridge), можна використовувати, щоб програмувати Arduino або інших Arduino-подібних контролерів, отримувати інформацію на комп’ютер з усього, що має послідовний інтерфейс з TTL логікою. Також можна використовувати для налагодження одноплатних комп’ютерів, особливо ті, що не мають відеовихід: NanoPi NEO, Orange Pi Zero, Orange Pi R1 і т.п ..

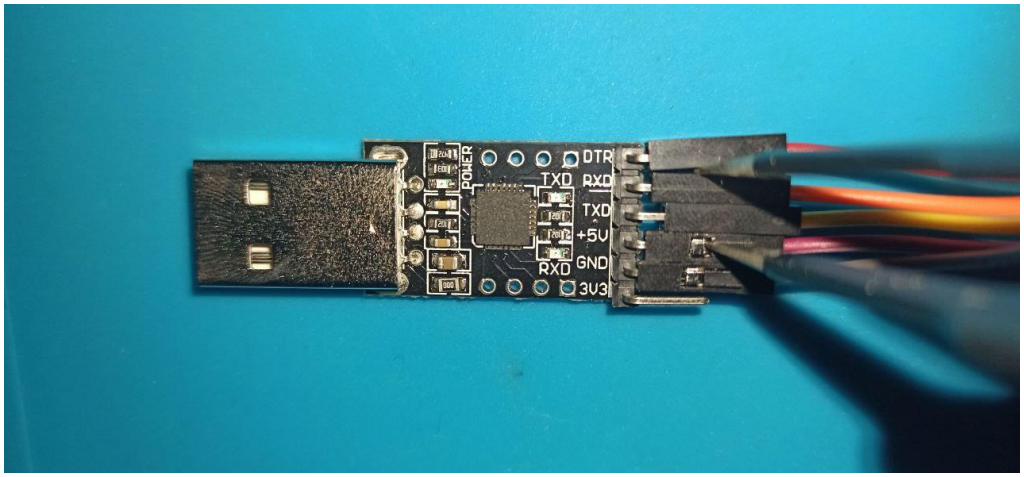

Рисунок 2.4 – CP2102 перетворювач USB-UART

CP2102 може похвалитися наявністю додатковим висновком DTR, який можна безпосередньо підключити до входу RESET на контролерах не мають USB на платі. Після цього при програмуванні тиснути кнопку RESET не

потрібно. Підтримкою виробника, сумісністю з оригінальними драйверами і ПО, на відміну від підроблених FTDI, у яких проблеми з рідними драйверами. Додатковими висновками (отворами під контакти) на платі, наприклад, дозволяють відводити USB в енергозберігаючий режим. Цікавою можливістю змінювати VID (Ідентифікатор виробника), PID (Ідентифікатор продукту) і текст, з яким розпізнається плата, збирати свій драйвер зі необхідними параметрами, що досить цікаво в комерційних проектах.

З одного боку перебувати роз'єм USB, з іншого 6 pin виводів: + 3.3v, GND, + 5v, TXD (TX), RXD (RX), DTR, на платі є монтажні отвори з функціями DCD, D3R, RTS, CTS, SUS , SUS, R1, RST. Крім цього на платі є 3 світлодіоди, червоний - POWER і два для RX і TX миготливих під час прийому-передачі даних.

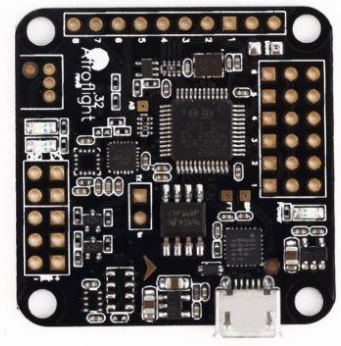

Рисунок 2.5 – Модуль контролю польоту Naze32 Acro V6 6DF

Характеристики:

а) чіп CP2102 від Silicon Labs;
б) швидкість обміну даними по UART 300Біт / сек - 1 Мбіт / сек;
в) буфер читання 576 байт, записи 640 байт;
г) підтримка USB 2.0 12Мбіт / сек;
д) підтримка режиму SUSPENDED USB;
е) вбудований стабілізатор живлення 3.3В 100мА;
ж) EEPROM з конфігураційними параметрами 1024 байт;

з) підтримувані ОС Windows 8/7 / Vista / Server 2003 / XP / 2000, Windows CE, Mac OS-X / OS-9, Linux, Android;

и) можливість налаштування параметрів плати та драйверів під свої проекти;

к) розміри плати 26.5 x 15.6 мм.

Невід'ємною частиною дрону є модуль контролю польоту, варіацій яких існує безліч. Приклад модулів контролю польоту:

Оновлена версія контролера польотів Naze32 6 DOF rev6 (рис. 2.5) - це один з найпопулярніших контролерів для гонок з акро управлінням і FPV польотами на квадрокоптера всіх міні класів починаючи від рам розміром 130мм і до 280мм.

Особливості версії REV 6:

а) USB порт перемістили в праву верхню сторону плати для більш зручного підключення;

б) новий чіп акселерометра MPU 6500 (в минулій версії Rev 5 використовується MPU 6050);

в) підтримка SBUS, тепер можна підключати безпосередньо ваш FRsky SBUS приймач без перехідників;

г) виділений порт Spectrum Satellite;

д) додатковий чіп пам'яті з 2MB для роботи blackbox і записи бортовий інформації.

Характеристики:

а) процесор: STM32F103CxT6 CPU (32bit ARM Cortex M3, 72MHz, 64K / 128K flash);

б) датчик гіроскоп + акселерометр Invensense MPU6500 3-axis gyro;

в) сучасний 32-бітний процесор STM32F103C8T6;

г) вбудований датчик напруги (до 6S);

д) підключення та налаштування контролера через Micro USB;

е) вихід на телеметрію;

ж) вбудований вихід на FrSky телеметрію з інвертором;

з) до 8-ми входів від приймача. Підтримка PPM з’єднання по одному дроту (FrSky, і ін.);

и) вибір різних рам квадрокоптера: Quad / Hexa / Tri / Bi / Y4 / Y6 / Octo;

к) дуже яскраві індикаційні світлодіоди;

л) підключення пищалки \ бузера прямо до плати;

м) піни 6 + 8 PWM I / O можна перепрограмувати як вхід або вихід для RC / CPPM / Motors / Servos;

н) другий UART, доступний для Spektrum Satellite RX або GPS;

о) порт супутникового приймача Spektrum (rev6);

п) бортовий самописець 2MB BlackBox (16Mbit) SPI flash memory;

р) SBUS інвертор;

с) CPPM (до 12 каналів) RC вхід;

т) 8 канальний стандартний PWM RC вхід;

у) вбудований microUSB роз’єм для телеметрії і оновлень через;

ф) програмовані світлодіоди;

х) накладки для підключення гідролокатора TRIG / ECHO з вбудованими резисторами (rev6).

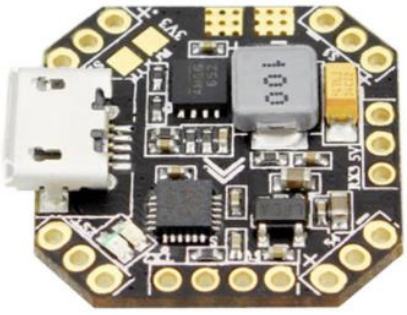

Рисунок 2.6 – Модуль контролю польоту EMAX F3 Femto

EMAX F3 Femto (рис. 2.6) - мініатюрний контролер польоту з процесором F3 на борту і прямим підключенням від батареї, який може працювати і прошиваться як SP Pro Racing EVO. Підходить для міні квадрокоптера розміром 130мм і менше.

Особливості:

а) малі габарити дають можливість встановити на різні рами і конфігурації мікро квадрокоптера;

б) хороший процесор STM32F303, здатний працювати з усіма портами одночасно (USB + SmartPort + SBus + LED Strip + Battery Monitoring + 4 motors), в зв'язці з акселерометром \ гіроскопом MPU9250 підключений через шину SPI;

в) вбудовані регулятори BEC 5V / 3A, 2-6S LiPo battery, пряме підключення від батарейки;

г) підключення приймачів по Sbus, PPM і Serial RX від 5V, приймачів Spektrum 1024/2048, SBUS, XBUS, SumD і SumH RX з вбудованим інвертором;

д) підтримує пряме з'єднання по 3.3V приймачів Spektrum Satellite;

е) підтримка телеметрії (frsky telemetry, smartport, і HoTT telemetry);

ж) пропускна здатність плати (PDB) 80A.

Порти UART:

а) UART3 (RX3) - використовує приймачі Spektrum Satellite RX, Spektrum 1024/2048, SBUS, XBUS, SumD і SumH RX і подібні;

б) UART2 / PPM - Serial RX or PPM RX + Telemetry. Працює RX2 для приймачів по PPM, використовує TX2 як вихід порту телеметрії по підключенню PPM.

Характеристики:

а) процесор: STM32F303;

б) акселерометр \ гіроскоп: MPU9250 (підключений через шину SPI);

в) вхідна напруга 2-6S LiPo battery;

г) живлення: 5v (3A);

д) розміри: 20 x 20 мм;

е) вага: 2.2 грам.

Контролер польоту LUX (рис. 2.7.) забезпечує високу якість управління польотом с допомогою потужного процесора F303 і гіроскопа з частотою оновлення до 8kHz по SPI шині, що дає можливість встановити ультра низьке число поновлення даних (looptimes) для ще більш точного управління дроном.

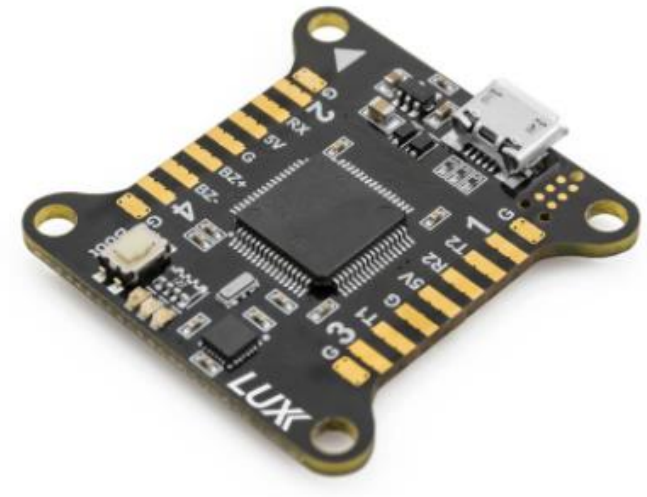

Рисунок 2.7 – Модуль контролю польоту LUX

Особливості:

а) компактна компонування плати з низьким профілем і якісними контактними майданчиками;

б) сучасний процесор F303 с високою частотою оновлення даних і Looptime (час виконання одного циклу мікроконтролера), це означає, що контролер ще швидше відстежує і реагує на зміну обстановки;

в) висока швидкість оновлення гіроскопа до 8kHz по SPI шині;

г) інтегрований FRSky RSSI фільтр;

д) підтримка SBUS приймачів через апаратний інвертор, а так само підтримка PPM;

е) рекомендована прошивка для ФАКРО режиму Betaflight.

Характеристики:

а) процесор: STM32F303RCT6, 32-бітний процесор;

б) мікросхема флеш-пам’яті на 256кб;

в) MPU6500 - комбінований гіроскорп-акселерометр;

г) розмір: 36 х 36 х 6 мм;

д) монтажні отвори: стандарт 30.5 мм М3;

е) вага: 6 грам.

Рисунок 2.8 – Модуль контролю польоту Skystars F722HD

Польотний контролер Skystars F722HD 20x20 (рис. 2.8) для квадрокоптера з FPV DJI системою на борту. Контролер оснащений процесором F7, 6 UARTS портів і регулятором напруги 10V 2A для Caddx Vista або DJI Air Unit. Плюс до всього, не дивлячись на дуже компактний дизайн, в контролер польоту включені барометр і Blackbox.

Характеристики:

а) процесор: STM32F722RGT6 216MHz;
б) гіроскоп: MPU6000 (SPI);
в) барометр;
г) екранне меню: Betaflight OSD і DJI HD OSD (UART6);
д) кількість UARTS: 6 апаратних (UART123456);
е) підтримка буззера;
ж) регулятор напруги: 5V 2A і 10V 2A;
з) вхідна напруга: 3-6S Lipo;
и) Blackbox: 16Mb;
к) прошивка: Betaflight;
л) таргет: SKYSTARS F7HD;
м) монтажні отвори: 20 х 20 мм / М3;

н) розміри: 35 x 28 x 7 мм;

о) вага: 4.2 г.

Також слід розглянути одну із готових реалізацій контролеру, який у своїй комплектації має усі потрібні датчики.

Lynxmotion Quadrino Nano був створений у результаті співпраці FlyingEinstein та Lynxmotion. Він був розроблений як один з найменших польотних контролерів, сумісних з MultiWii, на ринку. Плата включає в себе багато додаткових функцій, які зазвичай зарезервовані для набагато дорожчих польотних контролерів, таких як передові датчики, вбудований GPS, просте у використанні програмне забезпечення та чохол з гасінням вібрації.

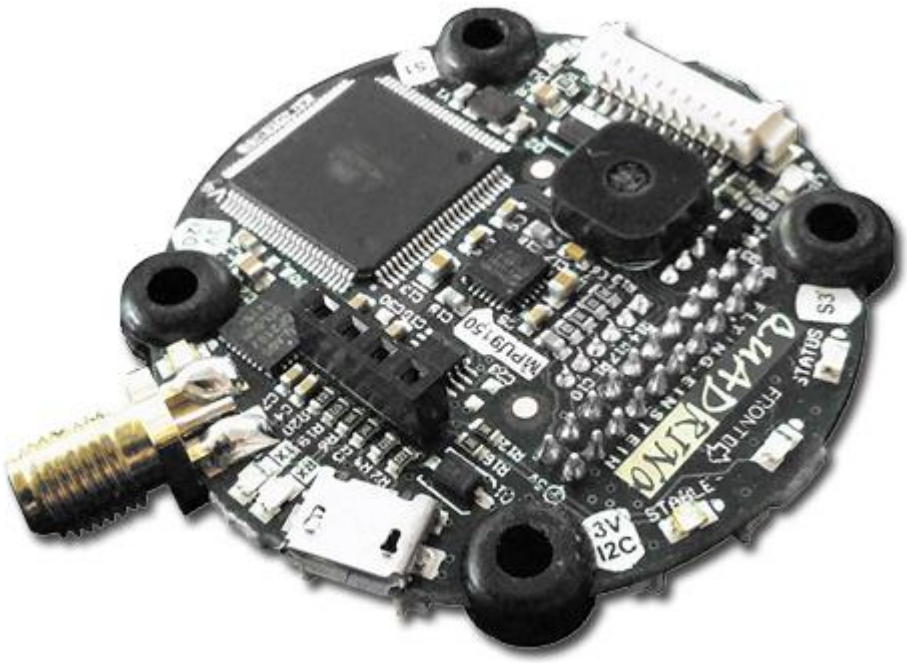

Рисунок 2.9 – Модуль контролю польоту Quadrino Nano

З додаванням Quadrino Nano до лінійки БПЛА Lynxmotion ми зможемо надати надійну платформу з точки зору апаратного та програмного забезпечення для тих, хто хоче перейти на БПЛА. Вбудовані датчики, а також GPS надають користувачеві доступ майже до всіх опцій, доступних через програмне

забезпечення MultiWii, а також дозволяють налаштовувати код і інтегрувати додаткові функції та продукти, які раніше були неможливими.

Quadrino простий у використанні з безкоштовним програмним забезпеченням «Firmware Configuration Tool (FCT)», яке позбавляє від необхідності змінювати код у середовищі розробки Arduino. Ця інтуїтивно зрозуміла програма проведе вас через весь процес налаштування польотного контролера, надавши багато документації щодо різних доступних опцій і параметрів. Коли ви завершите вибір параметрів, цей інструмент скомпілює та завантажить власне програмне забезпечення безпосередньо на вашу плату Quadrino Nano. Він також встановить для вас графічний інтерфейс конфігурації MultiWiiConf та/або WinGUI.

Рисунок 2.10 – Інтерфейс програмного забезпечення для Quadrino Nano

Технічні характеристики:

а) процесор ATMega 2560 (256Kb flash @ 16MHz) (рис. 2.12);

б) компактний шестишаровий дизайн друкованої плати;

в) варіанти кріплення: двостороння клейка піна 3M або шурупи;

г) вбудоване гасіння вібрації для друкованої плати;

д) завантажувач Arduino робить можливими проекти DIY;

е) роз’єм Micro-USB / порт програмування;

ж) сенсорний чіп Invensense MPU9150, який включає:

1) 3-осьовий гіроскоп;

2) 3-осьовий акселерометр;

3) 3-осьовий магнітометр;

з) MS5611 Барометр з пінопластовою кришкою, вбудованою в футляр;

и) Venus838FLPx 50Hz GPS чіпсет із зовнішньою антеною (роз’єм SMA):

1) частота оновлення GPS 50 Гц;

2) послідовний вихід NEMA для OSD (екранний дисплей);

к) порт двигуна з 8-кратними виходами регулятора швидкості;

л) радіопорт з 8 входами радіоканалів, дозволяє використовувати всі чотири допоміжні (AUX) входи;

м) два вільних послідовних порту (використовуйте для SBUS та/або радіозв’язків Bluetooth або 3DR);

н) три порти I2C, два з яких два 3,3 В і один 5 В;

о) вбудований OLED-порт 0,96" (OLED не входить в комплект);

п) порт приймача Spektrum R/C;

р) сервопорт з п’ятьма виходами;

с) роз’єм 2x10 I/0 порт розширення;

т) порт сигналізації LIPO та світлодіодний індикатор.

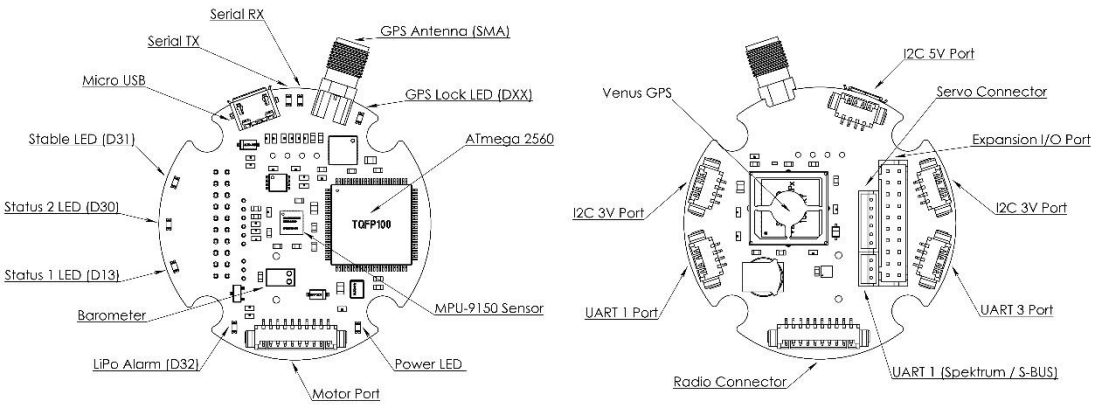

Рисунок 2.11 – Схема модуля Quadrino Nano

Високопродуктивний, low-power 8-розрядний мікроконтролер на базі AVR® RISC від Microchip (рис. 2.12) поєднує в собі 256 КБ флеш-пам’яті ISP, 8 КБ SRAM, 4 КБ EEPROM, 86 ліній вводу-виводу загального призначення, 32 робочих регістра загального призначення в режимі реального часу. лічильник, шість гнучких таймерів/лічильників з режимами порівняння, ШІМ, чотири USART, байто-орієнтований двопровідний послідовний інтерфейс, 16-канальний 10-розрядний АЦП і інтерфейс JTAG для налагодження на чіпі. Пристрій досягає пропускної здатності 16 MIPS на 16 МГц і працює в діапазоні 4,5-5,5 вольт.

Рисунок 2.12 – Процесор ATMega 2560

Виконуючи інструкції за один такт, пристрій досягає пропускної здатності, що наближається до однієї MIPS на МГц, балансуючи споживання енергії та швидкість обробки.

2.3 Приклад практичного застосування

На практиці охоронні системи використовуються дуже широко. Найчастіше варіації охоронних систем розроблюють під конкретну задачу,

враховуючи умови території на якій вона буде працювати. Але використання дронів змінює ситуацію кардинально.

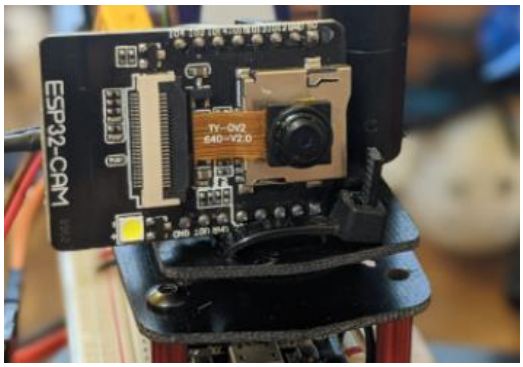

Рисунок 2.9 – ESP32 Cam, встановлена на корпус дрона

Охоронна система становиться універсальною, стійкою та автоматизованою . А використання технологій комп’ютерного зору ще більш покращує систему, видаляючи людський фактор неуважності, тим самим збільшує її ефективність. Приклад встановлення ESP32 Cam у якості камери зображено на Рисунку 2.9.

Таких охоронних систем потребують області різних спектрів, починаючи від власного користування за для захисту будинків, закінчуючи воєнною промисловістю та захисту кордону.

3 ОПИС ТЕХНОЛОГІЙ ВИКОРИСТАНИХ ДЛЯ РЕАЛІЗАЦІЇ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ БЕЗПЕКИ

3.1 Комп’ютерний зір

Комп’ютерний зір – це область штучного інтелекту (ШІ), яка дозволяє комп’ютерам і системам отримувати значущу інформацію з цифрових зображень, відео та інших візуальних вхідних даних і виконувати дії або давати рекомендації на основі цієї інформації. Якщо AI дозволяє комп’ютерам думати, то комп’ютерний зір дозволяє їм бачити, спостерігати і розуміти.

Комп’ютерний зір працює майже так само, як і людський, за винятком того, що у людей є фора. Людський зір має перевагу контексту протягом життя, щоб тренувати, як розрізняти об’єкти, наскільки вони віддалені, чи рухаються вони та чи є щось не так у зображенні.

Рисунок 3.1 – Результат роботи комп’ютерного зору

Комп’ютерний зір навчає машини виконувати ці функції, але він повинен робити це за набагато менше часу за допомогою камер, даних і алгоритмів, а не сітківки, зорових нервів і зорової кори. Оскільки система, навчена перевіряти

продукти або спостерігати за виробничим активом, може аналізувати тисячі продуктів або процесів за хвилину, помічаючи непомітні дефекти або проблеми, вона може швидко перевершити людські можливості .

Комп’ютерний зір використовується в різних галузях, починаючи від енергетики та комунальних послуг, закінчуючи виробництвом та автомобілем – і ринок продовжує зростати. Очікується, що до 2022 року він досягне 48,6 млрд доларів [3].

Комп’ютерному зору потрібно багато даних. Він проводить аналіз даних знову і знову, поки не розпізнає відмінності і в кінцевому підсумку не розпізнає зображення. Наприклад, щоб навчити комп’ютер розпізнавати автомобільні шини, йому потрібно отримати величезну кількість зображень шин і предметів, пов’язаних з шинами, щоб дізнатися відмінності та розпізнати шину, особливо без дефектів [3].

Для цього використовуються дві основні технології: тип машинного навчання, який називається глибоким навчанням, і згортка нейронна мережа (CNN).

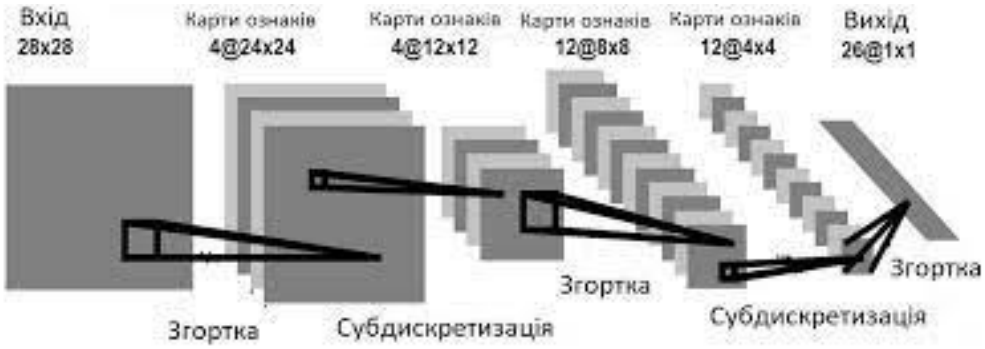

Рисунок 3.2 – Структура згорткової нейронної мережі

Машинне навчання використовує алгоритмічні моделі, які дозволяють комп’ютеру самостійно навчатися контексту візуальних даних. Якщо через модель подається достатня кількість даних, комп’ютер «подивиться» на дані і навчиться відрізняти одне зображення від іншого. Алгоритми дозволяють

машині вчитися самостійно, а не використовувати ситуацію коли хтось програмує її на розпізнавання зображення.

CNN допомагає моделі машинного навчання або глибокого навчання «виглядати», розбиваючи зображення на пікселі, яким надано теги або мітки. Він використовує мітки для виконання згорток (математична операція над двома функціями для створення третьої функції) і робить прогнози щодо того, що він «бачить». Нейронна мережа виконує згортки і перевіряє точність своїх прогнозів у серії ітерацій, поки передбачення не почнуть збуватися. Потім він розпізнає або бачить зображення так само, як і люди [3].

Глибоке навчання – це тип машинного навчання та штучного інтелекту (ШІ), який імітує спосіб отримання людьми певних знань. Глибоке навчання є важливим елементом науки про дані, що включає статистику та прогнозне моделювання. Це надзвичайно корисно для дослідників даних, яким доручено збирати, аналізувати та інтерпретувати великі обсяги даних; глибоке навчання робить цей процес швидшим і легшим.

У найпростішому випадку глибоке навчання можна розглядати як спосіб автоматизації прогнозної аналітики. У той час як традиційні алгоритми машинного навчання є лінійними, алгоритми глибокого навчання складаються в ієрархію зростаючої складності та абстракції.

Комп’ютерні програми, які використовують глибоке навчання, проходять майже такий же процес, як і малюк, який навчається ідентифікувати об’єкти. Кожен алгоритм в ієрархії застосовує нелінійне перетворення до свого входу і використовує те, що він дізнався, для створення статистичної моделі як вихідну інформацію. Ітерації продовжуються, поки вихід не досягне прийнятного рівня точності.

У традиційному машинному навчанні процес навчання контролюється, і програміст повинен бути надзвичайно конкретним, повідомляючи комп’ютеру, які типи речей він повинен шукати, щоб вирішити, містить зображення потрібний об’єкт чи не містить. Це трудомісткий процес, який називається вилученням функцій, і рівень успіху комп’ютера повністю залежить від

здатності програміста точно визначити набір функцій для визначення об'єкта на зображенні. Перевага глибокого навчання полягає в тому, що програма створює набір функцій сама без нагляду. Навчання без нагляду не тільки швидше, але і, як правило, точніше.

Спочатку комп’ютерна програма може бути забезпечена навчальними даними – набором зображень, для яких людина позначив кожне зображення собаки чи ні собаки метатегами. Програма використовує інформацію, яку вона отримує з даних дресирування, щоб створити набір функцій для об’єкту та побудувати прогнозну модель. У цьому випадку модель, яку спочатку створює комп’ютер при визначенні наприклад собаки, може передбачити, що все на зображенні, що має чотири ноги і хвіст, має бути позначено як собака. Звичайно, програма не знає про мітки чотири ноги або хвіст. Він просто шукатиме шаблони пікселів у цифрових даних. З кожною ітерацією прогнозна модель стає складнішою та точнішою.

Щоб досягти прийнятного рівня точності, програми глибокого навчання вимагають доступу до величезної кількості навчальних даних і потужності обробки, жодне з яких не було легко доступним для програмістів до ери BigData і хмарних обчислень. Оскільки програмування глибокого навчання може створювати складні статистичні моделі безпосередньо з власних ітераційних результатів, воно здатне створювати точні прогнозні моделі з великої кількості немаркованих, неструктурованих даних. Це важливо, оскільки Інтернет речей (IoT) продовжує набувати все більшого поширення, оскільки більшість даних, які створюють люди та машини, є неструктурованими та не позначеними.

Для створення потужних моделей глибокого навчання можна використовувати різні методи. Ці методи включають зниження швидкості навчання, трансферне навчання, навчання з нуля та відсівання [4].

Зниження швидкості навчання. Швидкість навчання є гіперпараметром, фактором, який визначає систему або встановлює умови для її роботи перед процесом навчання, який контролює, наскільки сильно змінюється модель у відповідь на оцінку помилки щоразу, коли змінюються вагові коефіцієнти

моделі. Занадто високі темпи навчання можуть призвести до нестабільних тренувальних процесів або вивчення неоптимального набору ваг. Занадто мала швидкість навчання може призвести до тривалого процесу навчання, який може застрягти [4].

Метод зниження швидкості навчання, також званий адаптивною швидкістю навчання, є процесом адаптації швидкості навчання для підвищення продуктивності та скорочення часу навчання. Найпростіші та найпоширеніші адаптації швидкості навчання під час навчання включають прийоми зниження швидкості навчання з часом [4].

Transfer learning передбачає вдосконалення попередньо навченої моделі; для цього потрібен інтерфейс до внутрішньої частини вже існуючої мережі. По-перше, користувачі передають існуючій мережі нові дані, що містять раніше невідомі класифікації. Після внесення змін до мережі нові завдання можна виконувати з більш конкретними можливостями категоризації. Цей метод має перевагу в тому, що вимагає набагато менше даних, ніж інші, що скорочує час обчислень до хвилин або годин [4].

Тренування з нуля. Цей метод вимагає від розробника зібрати великий набір мічених даних і налаштувати архітектуру мережі, яка може вивчати функції та модель. Ця техніка особливо корисна для нових програм, а також додатків з великою кількістю вихідних категорій. Однак загалом це менш поширений підхід, оскільки він вимагає надмірної кількості даних, через що навчання займає дні або тижні.

Dropout намагається вирішити проблему переобладнання в мережах з великою кількістю параметрів шляхом випадкового вилучення одиниць та їх з’єднань з нейронної мережі під час навчання. Було доведено, що метод Dropout може покращити продуктивність нейронних мереж у контрольованих навчальних завданнях у таких областях, як розпізнавання мовлення, класифікація документів та обчислювальна біологія [4].

Тип передового алгоритму машинного навчання, відомий як штучна нейронна мережа, лежить в основі більшості моделей глибокого навчання. У

результаті глибоке навчання іноді називають глибоким нейронним навчанням або глибокою нейронною мережею.

Нейронні мережі бувають кількох різних форм, включаючи повторювані нейронні мережі, згорткові нейронні мережі, штучні нейронні мережі та нейронні мережі з прямим зв'язком, і кожна з них має переваги для конкретних випадків використання. Проте всі вони функціонують дещо подібним чином – надаючи дані та дозволяючи моделі самостійно з'ясувати, чи правильно вона інтерпретувала або прийняла рішення щодо даного елемента даних.

Нейронні мережі використовують процес проб і помилок, тому їм потрібні величезні обсяги даних для навчання. Не випадково нейронні мережі стали популярними лише після того, як більшість підприємств прийняли аналітику великих даних і накопичили великі сховища даних. Оскільки перші кілька ітерацій моделі передбачають дещо обґрунтовані припущення щодо вмісту зображення або частин мови, дані, які використовуються на етапі навчання, мають бути позначені, щоб модель могла побачити, чи була її припущення точною. Це означає, що, хоча багато підприємств, які використовують великі дані, мають великі обсяги даних, неструктуровані дані менш корисні. Неструктуровані дані можуть бути проаналізовані моделлю глибокого навчання лише після того, як вона була навчена та досягла прийнятного рівня точності, але моделі глибокого навчання не можуть навчатися на неструктурованих даних [4].

Оскільки моделі глибокого навчання обробляють інформацію так само, як і людський мозок, їх можна застосувати до багатьох завдань, які виконують люди. Глибоке навчання в даний час використовується в більшості поширених інструментів розпізнавання зображень, обробки природної мови (NLP) і програмному забезпеченні розпізнавання мовлення. Ці інструменти починають з'являтися в таких різноманітних програмах, як самокеровані автомобілі та служби перекладу мов.

Сучасні приклади використання глибокого навчання включають усі типи програм для аналізу великих даних, особливо ті, які зосереджені на NLP, мовному перекладі, медичній діагностиці, торгових сигналах фондового ринку,

безпеці мережі та розпізнаванні зображень.

Конкретні галузі, в яких наразі використовується глибоке навчання, включають наступне:

Досвід клієнта (Customer experience). Моделі глибокого навчання вже використовуються для чат-ботів. І, оскільки воно продовжує розвиватися, очікується, що глибоке навчання буде впроваджено в різні підприємства, щоб покращити CX і підвищити рівень задоволеності клієнтів.

Генерація тексту. Машини навчають граматиці та стилю фрагмента тексту, а потім використовують цю модель для автоматичного створення абсолютно нового тексту, що відповідає правильному написанню, граматиці та стилю оригінального тексту.

Аерокосмічні та військові. Глибоке навчання використовується для виявлення об’єктів із супутників, які визначають цікаві зони, а також безпечні чи небезпечні зони для військ.

Промислова автоматизація. Глибоке навчання покращує безпеку працівників у таких середовищах, як фабрики та склади, надаючи послуги, які автоматично визначають, коли працівник або об’єкт наближаються занадто близько до машини.

Додавання кольору. До чорно-білих фотографій і відео можна додати колір за допомогою моделей глибокого навчання. У минулому це був надзвичайно трудомісткий ручний процес.

Медичне дослідження. Дослідники раку почали впроваджувати глибоке навчання у свою практику як спосіб автоматичного виявлення ракових клітин.

Комп’ютерний зір. Глибоке навчання значно покращило комп’ютерний зір, забезпечуючи комп’ютерам надзвичайну точність для виявлення об’єктів і класифікації зображень, відновлення та сегментації.

Згорткова нейронна мережа (ConvNet/CNN) – це алгоритм глибокого навчання, який може приймати вхідне зображення, призначати важливість (засвоювані ваги та упередження) різним аспектам/об’єктам зображення та мати можливість відрізнити один від іншого. Попередня обробка, необхідна в

ConvNet, набагато нижча в порівнянні з іншими алгоритмами класифікації. Хоча в примітивних методах фільтри розробляються вручну, при достатньому навчанні, ConvNets мають можливість вивчати ці фільтри/характеристики.

Архітектура ConvNet аналогічна моделі зв’язку нейронів у людському мозку і була натхненна організацією зорової кори. Окремі нейрони реагують на подразники лише в обмеженій області поля зору, відомому як Рецептивне поле. Набір таких полів перекривається, щоб охопити всю візуальну область.

Звичайні нейронні мережі отримують вхідні дані (один вектор) і перетворюють його через серію прихованих шарів. Кожен прихований шар складається з набору нейронів, де кожен нейрон повністю пов’язаний з усіма нейронами в попередньому шарі, і де нейрони в одному шарі функціонують повністю незалежно і не мають жодних зв’язків. Останній повністю підключений шар називається «вихідним шаром», і в налаштуваннях класифікації він представляє бали класу [5].

Звичайні нейронні мережі погано масштабуються до повних зображень. У CIFAR-10 зображення мають розмір лише 32*32*3 (32 ширини, 32 висоти, 3 колірних каналу), тому один повністю підключений нейрон у першому прихованому шарі звичайної нейронної мережі мав би 32*32*3 = 3072 ваги. . Ця сума все ще здається керованою, але очевидно, що ця повністю пов’язана структура не масштабується до більших зображень. Наприклад, зображення більш поважного розміру, напр. 200*200*3, призведе до нейронів, які мають 200*200*3 = 120 000 ваг. Більше того, не дуже гарно мати кілька таких нейронів, щоб параметри швидко складалися. Очевидно, що таке повне підключення є марнотратним, і величезна кількість параметрів швидко призведе до переобладнання [5].

Згорткові нейронні мережі використовують той факт, що вхідні дані складаються з зображень, і вони обмежують архітектуру більш розумним чином. Зокрема, на відміну від звичайної нейронної мережі, шари ConvNet мають нейрони, розташовані в 3 вимірах: ширина, висота, глибина. Потрібно звернути увагу, що слово глибина тут відноситься до третього виміру об’єму активації, а

не до глибини повної нейронної мережі, яка може посилатися на загальну кількість шарів у мережі. Наприклад, вхідні зображення в CIFAR- 10 – вхідний об’єм активацій, розмір якого становить 32*32*3 (ширина, висота, глибина відповідно). Нейрони в шарі будуть з’єднані лише з невеликою ділянкою шару перед ним, а не з усіма нейронами повністю з’єднаними способом. Більше того, кінцевий вихідний шар для CIFAR-10 мав би розміри 1*1*10, тому що у кінці архітектури ConvNet потрібно зменшити повне зображення в єдиний вектор оцінок класу, розташованих уздовж розміру глибини [5].

Існує кілька архітектур в області згорткових мереж, які мають назву. Найпоширенішими є:

LeNet. Перші успішні застосування згорткових мереж були розроблені Янном Лекуном у 1990-х роках. З них найвідомішою є архітектура LeNet, яка використовувалася для зчитування поштових індексів, цифр тощо.

AlexNet. Першою роботою, яка популяризувала згорткові мережі в комп’ютерному бачення, була AlexNet, розроблена Алексом Крижевським, Іллею Суцкевером та Джеффом Хінтоном. AlexNet був представлений на конкурс ImageNet ILSVRC у 2012 році і значно випередив друге місце (похибка 5 найпопулярніших 16% у порівнянні з 26% помилкою). Мережа мала дуже подібну архітектуру до LeNet, але була глибшою, більшою та містила згорткові шари, накладені один на одного (раніше було звичайним явищем, коли лише один шар CONV завжди слідував за шаром POOL) [5].

ZF Net. Переможцем ILSVRC 2013 стала Convolutional Network від Метью Зейлера та Роба Фергуса. Він став відомий як ZFNet (скорочення від Zeiler & Fergus Net). Це було покращення AlexNet шляхом налаштування гіперпараметрів архітектури, зокрема, шляхом розширення розміру середніх згорткових шарів та зменшення розміру кроку та фільтра на першому шарі [5].

GoogLeNet. Переможцем ILSVRC 2014 стала Convolutional Network від Szegedy et al. від Google. Його основним внеском стала розробка початкового модуля, який різко зменшив кількість параметрів у мережі (4M, порівняно з AlexNet з 60M). Крім того, у цій статті використовується середній пул замість

шарів повністю підключених у верхній частині мережі ConvNet, що усуває велику кількість параметрів, які, здається, не мають великого значення. Існує також кілька наступних версій GoogLeNet, остання версія Inception-v4 [5].

VGGNet. Друге місце в ILSVRC 2014 посіла мережа Карена Симоняна та Ендрю Зіссермана, яка стала відома як VGGNet. Його основний внесок полягав у тому, щоб показати, що глибина мережі є критичним компонентом для хорошої продуктивності. Їхня остання найкраща мережа містить 16 шарів CONV/FC і, що привабливо, має надзвичайно однорідну архітектуру, яка виконує лише згортки 3x3 і пул 2x2 від початку до кінця. Їх попередньо підготовлена модель доступна для використання в Caffe. Недоліком VGGNet є те, що він дорожчий для оцінки та використовує набагато більше пам'яті та параметрів (140 МБ). Більшість із цих параметрів знаходиться в першому повністю підключеному шарі, і з тих пір було виявлено, що ці рівні FC можна видалити без зниження продуктивності, що значно зменшує кількість необхідних параметрів [5].

ResNet. Залишкова мережа, розроблена Kaiming He et al. стала переможцем ILSVRC 2015. Вона має спеціальні з'єднання пропуску та інтенсивне використання пакетної нормалізації. В архітектурі також відсутні повністю підключені рівні в кінці мережі. Читач також посилається на презентацію Каймінга (відео, слайди) та деякі останні експерименти, які відтворюють ці мережі в Torch. ResNets наразі є найсучаснішими моделями згорткових нейронних мереж і є вибором за замовчуванням для використання ConvNets на практиці (станом на 10 травня 2016 року). Зокрема, дивіться також новіші розробки, які змінюють оригінальну архітектуру від Kaiming He et al. Відображення ідентифікації в глибоких залишкових мережах (опубліковано в березні 2016 року).

Подібно до людини, яка створює зображення на відстані, CNN спочатку розрізняє жорсткі краї та прості форми, а потім заповнює інформацію, виконуючи ітерації своїх передбачень. CNN використовується для розуміння окремих зображень. Рекурентна нейронна мережа (RNN) використовується подібним чином для відеопрограм, щоб допомогти комп'ютерам зрозуміти, як

зображення в серії кадрів пов’язані один з одним.

Рекурентна нейронна мережа (RNN) – це тип штучної нейронної мережі, яка використовує послідовні дані або дані часових рядів. Ці алгоритми глибокого навчання зазвичай використовуються для порядкових або тимчасових проблем, таких як мовний переклад, обробка природної мови (nlp), розпізнавання мовлення та підписання зображень; вони включені в популярні програми, такі як Siri, голосовий пошук і Google Translate. Подібно до прямих і згорткових нейронних мереж (CNN), рекуррентні нейронні мережі використовують навчальні дані для навчання. Вони відрізняються своєю «пам’яттю», оскільки вони беруть інформацію з попередніх входів, щоб впливати на поточний вхід і вихід. У той час як традиційні глибокі нейронні мережі припускають, що входи та виходи незалежні один від одного, вихід рекуррентних нейронних мереж залежить від попередніх елементів у послідовності. Хоча майбутні події також можуть бути корисними для визначення результатів даної послідовності, однонаправлені рекурентні нейронні мережі не можуть врахувати ці події у своїх прогнозах.

Вчені та інженери вже близько 60 років намагаються розробити способи, щоб машини могли бачити й розуміти візуальні дані. Експерименти почалися в 1959 році, коли нейрофізіологи показали кішці низку зображень, намагаючись співвіднести реакцію в її мозку. Вони виявили, що він спочатку реагує на жорсткі краї або лінії, і з наукової точки зору це означало, що обробка зображень починається з простих форм, таких як прямі краї [3].

Приблизно в той же час була розроблена перша технологія комп’ютерного сканування зображень, яка дозволила комп’ютерам оцифрувати та отримувати зображення. Ще одна віха була досягнута в 1963 році, коли комп'ютери змогли перетворити двовимірні зображення в тривимірні форми. У 1960-х роках ШІ з’явився як академічна область дослідження, і це також ознаменувало початок пошуку ШІ для вирішення проблеми людського зору.

У 1974 році була запроваджена технологія оптичного розпізнавання символів (OCR), яка могла розпізнавати текст, надрукований будь-яким

шрифтом або гарнітурою. Аналогічно, інтелектуальне розпізнавання символів (ICR) могло розшифрувати рукописний текст за допомогою нейронних мереж. Відтоді OCR і ICR знайшли свій шлях в обробці документів і рахунків-фактур, розпізнаванні номерних знаків транспортних засобів, мобільних платежах, машинному перекладі та інших поширених програмах.

У 1982 році нейробіолог Девід Марр встановив, що зір працює ієрархічно, і представив алгоритми для машин для виявлення країв, кутів, кривих і подібних основних форм. Одночасно вчений-комп'ютерник Куніхіко Фукусіма розробив мережу клітин, які могли розпізнавати закономірності. Мережа, яка називається Neocognitron, включала згорткові шари в нейронну мережу [3].

До 2000 року в центрі уваги дослідження було розпізнавання об'єктів, а до 2001 року з'явилися перші програми для розпізнавання облич у реальному часі. У 2000-х роках з'явилася стандартизація того, як набори візуальних даних позначаються та анотуються. У 2010 році став доступний набір даних ImageNet. Він містив мільйони позначених зображень у тисячах класів об'єктів і є основою для CNN і моделей глибокого навчання, які використовуються сьогодні. У 2012 році команда з Університету Торонто взяла участь CNN в конкурсі на розпізнавання зображень. Модель під назвою AlexNet значно знизила частоту помилок при розпізнаванні зображень. Після цього прориву рівень помилок впав лише до кількох відсотків [3].

3.2 TensorFlow

Google відкрив джерела своїх програм машинного навчання TensorFlow наприкінці 2015 року. Раніше вона була використана власноруч Google для розпізнавання мови, пошуку, фотографій та Gmail, тощо [6].

Колишня масштабована розподілена система навчання під назвою DistBelief зробила основний вплив на поточну реалізацію TensorFlow. TensorFlow не перша система Google із відкритим кодом на основі внутрішнього

проекту. Відома система Google MapReduce та файлова система Google (GFS) - основа сучасної Apache системи обробки даних, веб-сканування та Big Data, включаючи Hadoop, Nutch, та Spark. Крім того, система BigTable від Google є джерелом появи проекту Apache Hbase. Бібліотека реалізована на C ++ і має зручний API Python, а також API C ++. Тому що простіші залежності до бібліотек TensorFlow можна швидко розгорнути на різних архітектурах. Подібно до Theano (популярної бібліотеки чисельних обчислень для Python), обчислення на TensorFlow описуються як блок-схеми, що відокремлюють дизайн від впровадження. Такий підхід дозволяє практично не вимагати ніяких складних дій із сторони розробника. Така конструкція також може бути впроваджена і на мобільних пристроях. Єдина система охоплює широкий спектр платформ. TensorFlow також чудово працює з низкою нових, подібним чином розроблених бібліотек ML, включаючи Keras (TensorFlow 2.0 повністю інтегрований з Keras), поряд з такими бібліотеками, як PyTorch, який розроблений Facebook, та більш багатими інтерфейсами прикладного програмування для ML, такими як Fast.AI [6].

Однією з властивостей TensorFlow є його автоматичний режим можливості диференціації. Можна експериментувати з новим мережі без необхідності перевизначати багато ключових розрахунків [6].

Автоматичне розмежування значно полегшує впровадження зворотного розподілення, яке використовується у галузі машинного навчання названою нейронними мережами. TensorFlow приховує дрібні деталі зворотного розподілення, щоб можна було зосередитись на загальній картині. Використання TensorFlow схоже на використання WolframAlpha для задач обчислення. Ще однією особливістю цієї бібліотеки є її інтерактивна візуалізація середовища, що називається TensorBoard. Цей інструмент показує блок-схему, що відображає підсумкові журнали з часом, і відстежує продуктивність [6].

Приклад вихідного коду для використання TensorFlow на ESP32 Cam приведений за посиланням [7].

TensorFlow став екосистемою, що складається з багатьох цінних

інструментів. В основі його популярності та універсальності лежить повна бібліотека машинного навчання та шаблони моделей, які швидко розвиваються з новими функціями та можливостями. Ця популярність має певну ціну, і ця вартість виражається як складність, складні залежності, оновлення API або терміни припинення підтримки, які можуть легко зламати моделі та робочий процес, яки з'явився не так давно. Одна справа вивчати і використовувати останні вдосконалення свого коду, будуючи модель, щоб експериментувати зі своїми ідеями та гіпотезами, але зовсім інша, якщо ваша робота полягає в тому, щоб побудувати модель для довгострокового використання, обслуговування та підтримки [6].

Інша проблема, пов'язана з раннім TensorFlow, загалом стосувалась процесу налагодження коду. У TensorFlow 1 ліниве виконання робить його досить складним для тестування або налагодження коду, оскільки код не виконується, якщо він не загорнутий у сеанс, він же графік. Починаючи з TensorFlow 2, нетерпляче виконання нарешті стає першокласним громадянином. Крім того, ще одним бажаним доповненням до TensorFlow 2 є прийняття високорівневого API Keras. Це значно спрощує кодування, експериментування та підтримку вашої моделі. Це також покращує читабельність вашого коду та процес його навчання [6].

TensorFlow має тири основні проблеми:

Перший виклик – це масштаб. Модель виробничого рівня повинна навчатися з великими обсягами даних, і часто її непрактично або неможливо вмістити в одновузлову пам'ять комп'ютера. Це також можна розглядати як іншу проблему: як ви передаєте навчальні дані в модель? Здається, природним і інстинктивним способом є оголошення та залучення всього набору даних як структури Pythonic, наприклад масиву NumPy або pandas DataFrame, як ми бачили в багатьох прикладах із відкритим кодом. Але якщо дані занадто великі, то здається розумним використовувати інший спосіб передачі даних у екземпляр моделі, подібний до ітератора Python. Бібліотеки TensorFlow.io і TensorFlow спеціально створені для вирішення цієї проблеми.

Другою проблемою, яка зазвичай виникає при впровадженні TensorFlow, є керованість середовища розробки. Зворотна сумісність не є сильною стороною TensorFlow, оскільки історично існують дуже швидкі оновлення та нові випуски API, які замінюють або не підтримують старі. Це включає, але не обмежується версією бібліотеки, підписом API та застарілим стилем використання. Це перешкода для розробки, налагодження та обслуговування кодової бази, а також не допомагає в управлінні стабільністю та відтворюваністю виробничого середовища та його результатами. Це може легко стати проблемою для того, хто керує інфраструктурою розробки машинного навчання та стандартними методами корпоративного проекту.

Третя проблема - це зусилля щодо покращення API, випуску патчів та виправлення помилок. Щоб вирішити цю проблему, TensorFlow використовує ці зусилля для довгострокової підтримки. Як правило, для будь-якого випуску TensorFlow команда TensorFlow від Google прагне надавати ці виправлення лише протягом року. Однак для підприємства це занадто мало, щоб отримати належну віддачу від інвестицій від циклу розвитку. Таким чином, для критично важливих показників підприємств важливою є триваліша прихильність до випусків TensorFlow.

TensorFlow Enterprise було створено для вирішення цих проблем. TensorFlow Enterprise - це спеціальний дистрибутив TensorFlow, який доступний виключно через різні служби Google Cloud. TensorFlow Enterprise доступний через:

а) Google Cloud AI Notebooks ;

б) Google Cloud AI Deep Learning VMs ;

в) Google Cloud AI Deep Learning Containers ;

г) частково доступно на Google Cloud AI Training.

Такими залежностями, як драйвери та сумісність версій бібліотеки, керує Google Cloud. Він також забезпечує оптимізоване підключення до інших служб Google Cloud, таких як Cloud Storage і сховище даних (BigQuery). Наразі TensorFlow Enterprise підтримує версії 1.15, 2.1 і 2.3 Google Cloud, а команди GCP

і TensorFlow надаватимуть довгострокову підтримку протягом трьох років, включаючи виправлення помилок та оновлення. На додаток до цих ексклюзивних послуг і керованих функцій команда TensorFlow також виводить підтримку підприємства на інший рівень, пропонуючи послугу «білих рукавичок». Це окрема служба від служби підтримки Google Cloud. У цьому випадку інженери TensorFlow в Google працюватимуть з кваліфікованими корпоративними клієнтами, щоб вирішувати проблеми або виправляти помилки в передових програмах AI.

TensorFlow дозволяє розробникам створювати діаграми потоків даних, структури, які описують, як дані переміщуються по діаграмах або ряду вузлів обробки. Кожен вузол на діаграмі представляє математичну операцію, а кожне з'єднання або ребро між вузлами є багатовимірним або розтягуючим масивом даних. TensorFlow забезпечує весь цей процес за допомогою мови Python. Його легко вивчати та працювати з Python і надає зручні способи спільного вираження абстракцій високого рівня. Але справжня математика не виконується в Python. Бібліотеки, доступні для TensorFlow, записані у високопродуктивному двійковому файлі C. Python лише направляє трафік між компонентами і надає високорівневі моделі програмування для їх з'єднання [8].

3.3 TensorFlow Lite

TensorFlow Lite для мікроконтролерів розроблено для запуску моделей машинного навчання на мікроконтролерах та інших пристроях з лише кількома кілобайтами пам'яті. Основне середовище виконання вміщається в 16 КБ на Arm Cortex M3 і може працювати з багатьма базовими моделями. Він не вимагає підтримки операційної системи, будь-яких стандартних бібліотек C або C++ або динамічного виділення пам'яті.

Мікроконтролери, як правило, є невеликими, малопотужними обчислювальними пристроями, які вбудовані в апаратне забезпечення, яке

вимагає базових обчислень. Додавши машинне навчання до крихітних мікроконтролерів, можна підвищити інтелект мільярдів пристроїв, які використовуються в нашому житті, включаючи побутову техніку та пристрої Інтернету речей, не покладаючись на дороге обладнання чи надійне інтернет-з'єднання, яке часто залежить від пропускної здатності та обмеження потужності і призводить до високої затримки. Це також може допомогти зберегти конфіденційність, оскільки дані не залишають пристрій. Уявіть собі розумні прилади, які можуть адаптуватися до вашої повсякденної роботи, інтелектуальні промислові датчики, які розуміють різницю між проблемами та нормальною роботою, а також чарівні іграшки, які можуть допомогти дітям вчитися веселим і захоплюючим способом.

TensorFlow Lite для мікроконтролерів написаний на C++ 11 і вимагає 32-розрядної платформи. Він був ретельно протестований з багатьма процесорами, заснованими на архітектурі серії Arm Cortex-M, і був портований на інші архітектури, включаючи ESP32. Фреймворк доступний як бібліотека Arduino. Він також може створювати проекти для середовищ розробки, таких як Mbed. Він є відкритим вихідним кодом і може бути включений до будь-якого проекту C++ 11.

Підтримуються такі плати розробки:

а) Arduino Nano 33 BLE Sense;

б) SparkFun Edge;

в) STM32F746 Discovery kit;

г) Adafruit EdgeBadge;

д) Adafruit TensorFlow Lite for Microcontrollers Kit;

е) Adafruit Circuit Playground Bluefruit;

ж) Espressif ESP32-DevKitC;

з) Espressif ESP-EYE;

и) Wio Terminal: ATSAMD51;

к) Himax WE-I Plus EVB Endpoint AI Development Board;

л) Sony Spresense.

Мікроконтролери мають обмежену оперативну пам’ять і сховище, що накладає обмеження на розміри моделей машинного навчання. Крім того, TensorFlow Lite для мікроконтролерів наразі підтримує обмежену підмножину операцій, тому не всі архітектури моделей можливі.

Щоб перетворити навчену модель TensorFlow для роботи на мікроконтролерах, вам слід використовувати API конвертера TensorFlow Lite Python. Це перетворить модель у FlatBuffer, зменшивши розмір моделі та змінить її для використання операцій TensorFlow Lite. Щоб отримати найменший можливий розмір моделі, слід розглянути можливість використання квантування після навчання.

Багато платформ мікроконтролерів не мають підтримки вбудованої файлової системи. Найпростіший спосіб використовувати модель із вашої програми – включити її як масив C і скомпілювати у програму.

Наступна (прик. 3.1) команда unix згенерує вихідний файл C, який містить модель TensorFlow Lite як масив символів.

```
xxd -i converted_model.tflite > model_data.cc
```

Приклад 3.1 – Генерація вихідного файлу C

При розробці моделі для використання на мікроконтролерах важливо враховувати розмір моделі, робоче навантаження та операції, які використовуються.

Результат команди (прик. 3.1) буде виглядати як масив байтів(прик. 3.2).

```
unsigned char converted_model_tflite[] = {
  0x18, 0x00, 0x00, 0x00, 0x54, 0x46, 0x4c, 0x33, 0x00, 0x00,
0x0e, 0x00,
  // <Lines omitted>
};
unsigned int converted_model_tflite_len = 18200;
```

Приклад 3.2 – Вміст сгенерованого файлу C

Після створення файлу потрібно включити його в свою програму. Важливо змінити оголошення масиву на const для кращої ефективності пам’яті на вбудованих платформах.

Модель має бути достатньо маленькою, щоб уміститися в пам’яті цільового пристрою поряд з кодом програми, як у двійковому файлі, так і під час виконання.

Щоб створити меншу модель, можна використовувати все менше шарів у архітектурі. Однак маленькі моделі частіше страждають від недооборудування. Це означає, що для багатьох проблем має сенс спробувати використовувати найбільшу модель, яка поміститься в пам’яті. Однак використання більших моделей також призведе до збільшення навантаження на процесор.

Розмір і складність моделі впливають на навантаження. Великі складні моделі можуть призвести до більш високого робочого циклу, що означає, що процесор вашого пристрою витрачає більше часу на роботу і менше часу простою. Це збільшить споживання електроенергії та тепловіддачу, що може бути проблемою залежно від вашого застосування.

3.4 Мова програмування

За останні кілька десятиліть комп’ютерні технології розвивались з надзвичайною швидкістю, ноутбук може обчислювати швидше і зберігати більше інформації, ніж мейнфрейм комп’ютери 1960-х. Досить багато програмістів можуть згадати підношення колод перфокарт для подання до могутньої комп’ютерної системи, що заповнює кімнату, з величні 100 Кб пам’яті - значно менше пам’яті, ніж сьогодні використовує навіть смартфон. Комп’ютерні мови також еволюціонували. Зміни можуть бути не такими драматичними, але вони важливо. Більші, потужніші комп’ютери породжують більші, складніші програми, що, в свою чергу, породжує нові проблеми в управлінні та підтримці програм [9].

У 1970-ті роки такі мови, як C та Паскаль, допомогли відкрити епоху структурованого програмування, філософії, яка наводила певний порядок та дисципліну у галузі, яка дуже потребувала цих якостей [9].

Крім надання інструментів для структурованого програмування, C також випускав компактні, швидко працюючі програми, а також можливість вирішувати апаратні питання, такі як управління портами зв'язку та дисководами. Ці подарунки допомогли зробити C домінуючою мовою програмування у 1980-х. Тим часом у 1980-х роках спостерігається зростання нової парадигми програмування: об'єктно-орієнтоване програмування, або ООП, втілене в таких мовах, як SmallTalk та C ++[9].

C ++ – це мова програмування загального призначення, створена Б'ярном Страуструпом як розширення мови програмування C, або "C with Classes". З плином часу мова значно розширилася, і сучасний C ++ тепер має об'єктно-орієнтовані, загальні та функціональні функції на додаток до засобів для маніпуляцій з пам'яттю низького рівня. Він майже завжди реалізується як компільована мова, і багато постачальників надають компілятори C ++, включаючи Free Software Foundation, LLVM, Microsoft, Intel, Oracle та IBM, тому він доступний на багатьох платформах [10].

C ++ був розроблений з орієнтацією на системне програмування та вбудоване програмне забезпечення з обмеженими ресурсами та великі системи, що характеризує продуктивність, ефективність та гнучкість використання. C ++ також виявився корисним у багатьох інших контекстах, головними перевагами якого є програмна інфраструктура та обмежені ресурси, включаючи настільні програми, відеоігри, сервери та критично важливі програми [10].

C ++ стандартизований Міжнародною організацією зі стандартизації (ISO), остання стандартна версія ратифікована та опублікована ISO у грудні 2020 року як ISO / IEC 14882: 2020. Мова програмування C ++ була спочатку стандартизована в 1998 році як ISO / IEC 14882: 1998, яка потім була змінена стандартами C ++ 03, C ++ 11, C ++ 14 та C ++ 17. Поточний стандарт C ++ 20 замінює їх новими функціями та розширеною стандартною бібліотекою. До

початкової стандартизації в 1998 році C ++ був розроблений датським інформатиком Б'ярном Страуструпом у Bell Labs з 1979 року як продовження мови C; він хотів ефективною та гнучкою мовою, подібною до C, яка також надавала високорівневі функції для організації програм [10].

3.5 Середовище програмування

Інтегроване середовище розробки Arduino (IDE) (рис 3.3) – це кросплатформений додаток (для Windows, macOS, Linux), у якому використовується код з C та C ++. Він використовується для запису та завантаження програм на суміжні плати Arduino, але також до сторонніх мікроконтролерів.

Вихідний код для IDE випускається під загальною ліцензією GNU, версія 2. Arduino IDE підтримує мови C і C ++, використовуючи спеціальні правила структуризації коду. Ідентифікатор Arduino IDE постачає бібліотеку програмного забезпечення від проекту Wiring, яка забезпечує безліч загальних процедур введення та виведення даних.

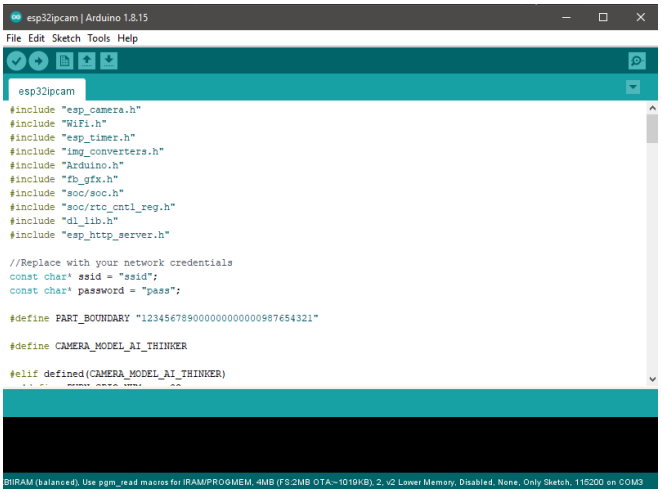

Рисунок 3.3 – інтерфейс Arduino IDE

Написаний користувачем код вимагає лише двох основних функцій для запуску ініціалізації та основного циклу програми, які компілюються та пов’язуються виконавчу програму з ланцюжком інструментів GNU, що також входить до пакету Arduino IDE. У Arduino IDE використовується програмна avrdude для перетворення виконуваного коду в текстовий файл у шістнадцятковому кодуванні, який завантажується на плату Arduino програмою-завантажувачем до пам’яті мікроконтролеру.

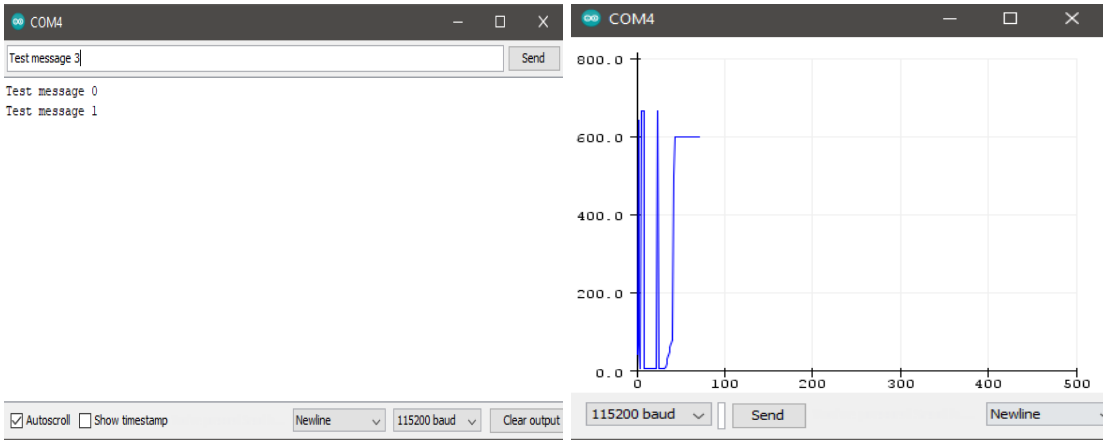

Рисунок 3.4 – Інструменти Arduino IDE

Також у Arduino IDE є інструменти за відслідковуванням COM портів (рис. 3.4), до яких відключений цільовий пристрій.

У жовтні 2019 року організація Arduino почала надавати ранній доступ до нового Arduino Pro IDE з налагодженням та іншими розширеними функціями.

4 РЕАЛІЗАЦІЯ СИСТЕМИ БЕЗПЕКИ

4.1 Апаратна частина

Реалізація інтелектуальної системи безпеки може базуватися на сумісному використанні із запрограмованим модулем контролю польотом Naze32 Flight Controller (рис 2.5).

Також можливі власні запрограмовані системи контролю польоту, як наприклад на базі мікроконтролеру ESP8266. У такому випадку компоненти підбираються власноруч. Такі варіації контролерів польоту є компактними та вмістяться у доволі невеликий корпус, що дозволить мінімізувати габарити пристрою, та дасть змогу такому пристрою зручно працювати у малих приміщеннях [11].

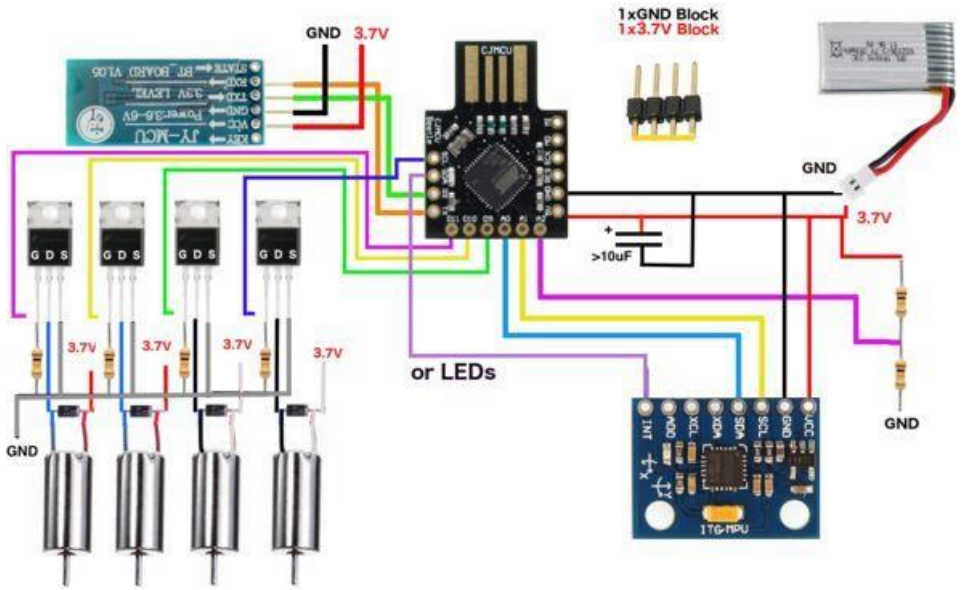

Рисунок 4.1 – Схема для контролю польотом на базі ESP8266

Приклад реалізації контролеру польоту на базі ESP8266 (рис. 4.1) має значно менші габарити за рахунок відокремлення датчика з гіроскопом, але

модуль Naze32 Acro (рис. 2.5) має влаштований набір датчиків безпосередньо на самій платі, що спрощує збирання схеми.

На рисунку 4.2 зображено макет схеми, де модуль ESP32 Cam виконує функції камери. За допомогою технології TensorFlow розпізнає завчано задані образи людей та подає сигнал тривоги до охоронного центру, або до локальної охоронної сигналізації, або відсилає повідомлення до смартфону власника. Контролер польоту завчасно запрограмований на обліт усієї території яка охороняється та робить деякі зупинки у зонах чекпоінтів для сканування навколишньої ситуації на предмет руху невідомих об’єктів , або зникнення об’єктів, що відслідковуються або охороняються.

Серед подібних проектів можна виділити анонсований охоронний пристрій від Ring, дочірньої компанії Amazon, що спеціалізується на розумних дверних замках та системах безпеки. Крихітна літаюча машина під назвою Always Home Cam (рис. 4.3) в автоматичному режимі веде патрулювання всередині будинку.

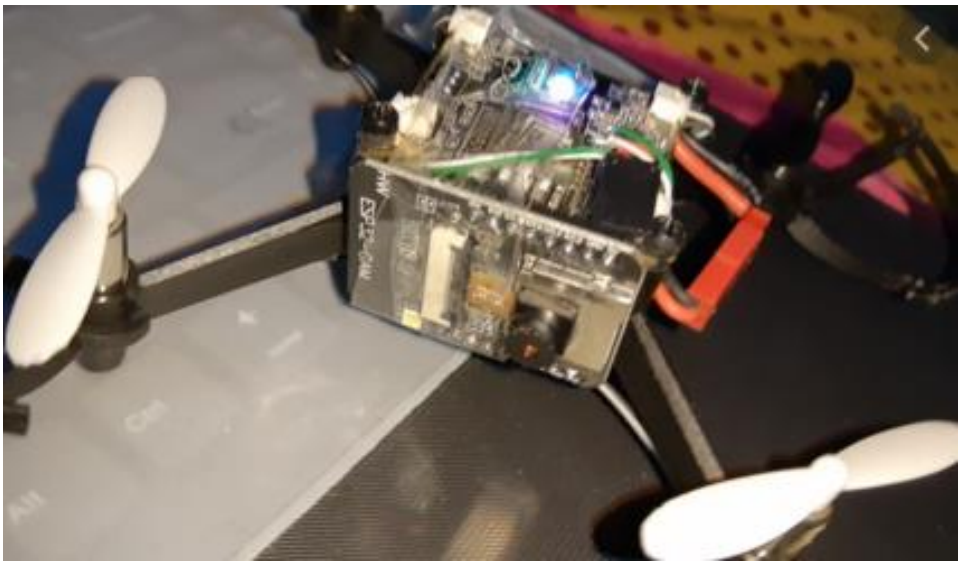

Рисунок 4.2 – Макет пристрою

Безпілотник, вартістю $ 250, сам злітає зі своєї зарядної станції, здійснює обліт приміщення, а потім сам повертається точно на базу. В цілому, він веде

себе як уже звичний всім робот-пилосос. З тією лише різницею, що дрон переміщується у повітрі і веде запис всієї підозрілої активності в приміщенні.

Зазвичай, дрон патрулює будинок в автономному режимі, але, при бажанні, господар може перехопити контроль над машиною.

Безпілотник розроблений з упором на максимальну безпеку. Його програмне забезпечення дозволяє машині автоматично уникати зіткнень і ухилятися від домашніх тварин. Незважаючи на це, пропелери дрона захищені легким каркасом. Це забезпечує захист як самої машині, так і того, з чим вона може зіткнутися [12].

Рисунок 4.3 – Always Home Cam

За словами засновника Ring Джеймі Сіміноффа, Always Home Cam є продуктом, «необхідність якого очевидна, але який дуже складно створити.» Дійсно, ідея літаючих охоронних дронів з’явилася задовго до того, як технології дозволили втілити її в життя. Але тепер ще одна розробка, яку раніше ми могли бачити тільки в фантастичних творах, стала реальною [12].

Незважаючи на те, Always Home Cam, є більш компактним та зі слів розробників надійним пристроєм, але є дуже коштовним. Створений пристрій на базі ESP32 Cam є більш доступним, і зможе виконувати ті ж самі функції що і його аналог від Ring.

4.2 Програмна частина

Щоб почати, бібліотека для модуля ESP32 Cam має бути встановлена в програмному забезпеченні Arduino IDE. Спочатку слід додати посилання на бібліотеки для програмування модулів ESP32 (прикл. 4.1) у розділі File > Preferences:

```
https://dl.espressif.com/dl/package_esp32_index.json,
http://arduino.esp8266.com/stable/package_esp8266com_index.json
```

Приклад 4.1 – Посилання на бібліотеки ESP32

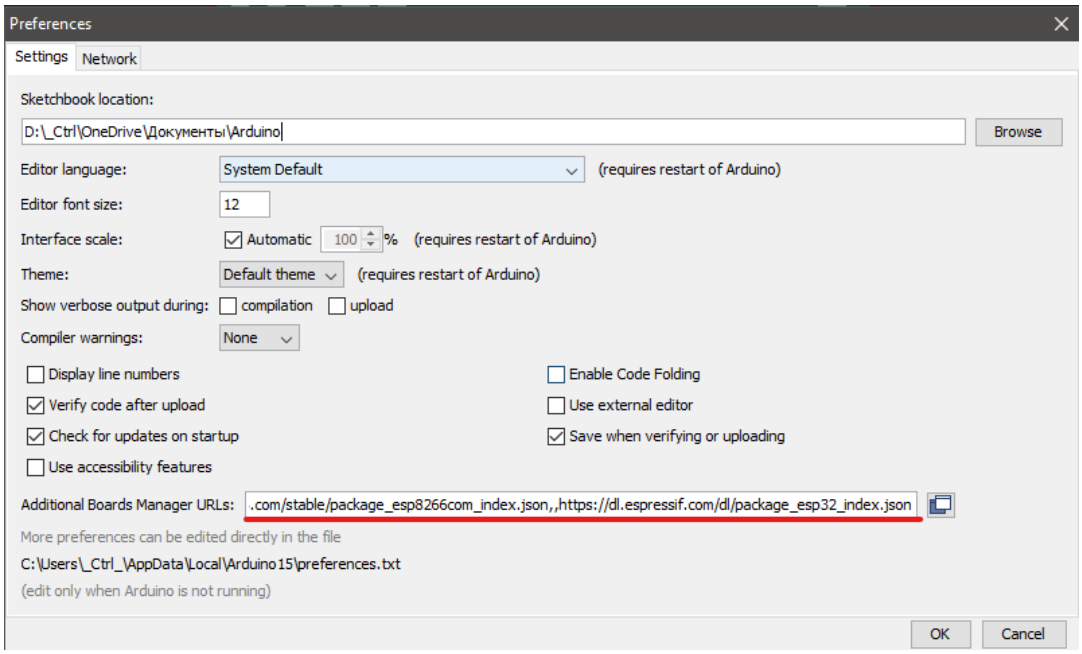

Рисунок 4.4 – Налаштування бібліотек для Arduino IDE

Тоді можна завантажити відповідну бібліотеку, перейшовши до розділу Tools> Board> Boards Manager (рис. 4.5).

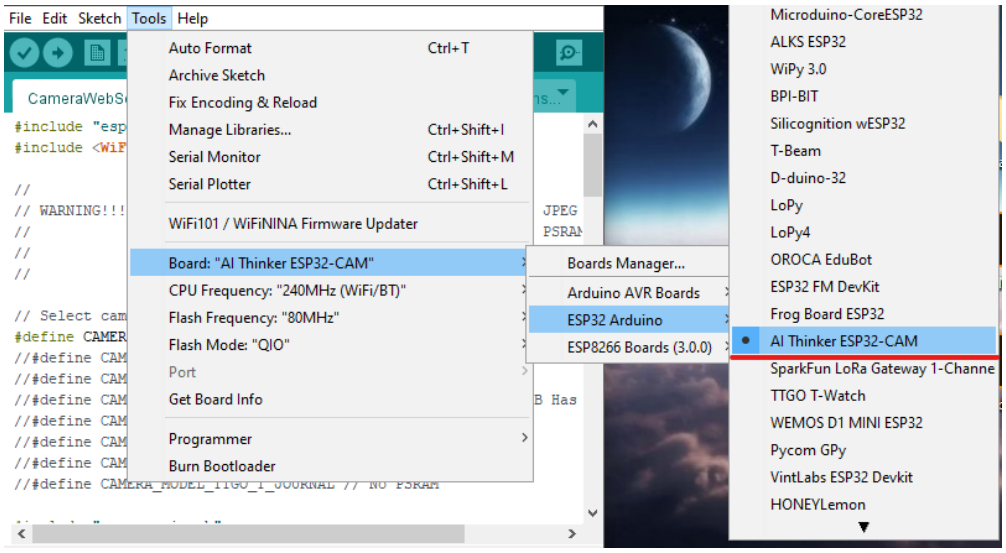

Рисунок 4.5 – Меню вибору типу плати

Для програмування плати ESP32 Cam знадобиться програмне забезпечення Arduino-IDE. Потрібно зробити підключення модулю згідно зі схемою відповідно до використовуваного USB TO TTL модуля. Під час програмування коду, тобто після компіляції коду, два піни GPIO0 і GND з'єднуються один з одним, це переводить плату у режим її програмування, і після успішної компіляції коду необхідно роз'єднати це з'єднання та зробити перезавантаження модуля для запуску проекту.

Першим кроком у використанні ESP32-CAM у поєднанні з Tensorflow.js є визначення об'єктів, які складають веб-сторінку, де відбувається висновок. Щоб використовувати бібліотеку Tensorflow Javascript, потрібно виконати наступні кроки: спочатку імпортувати бібліотеки Tensorflow JavaScript, потім завантажити модель, у цьому проекті буде використана модель ML, навчена COCO-SSD. Також потрібно створити мітки для оброблених об'єктів, які відображаються за допомогою моделі COCO-SSD на вхідному відео ідентифікованих елементів шляхом малювання прямокутників навколо об'єктів.

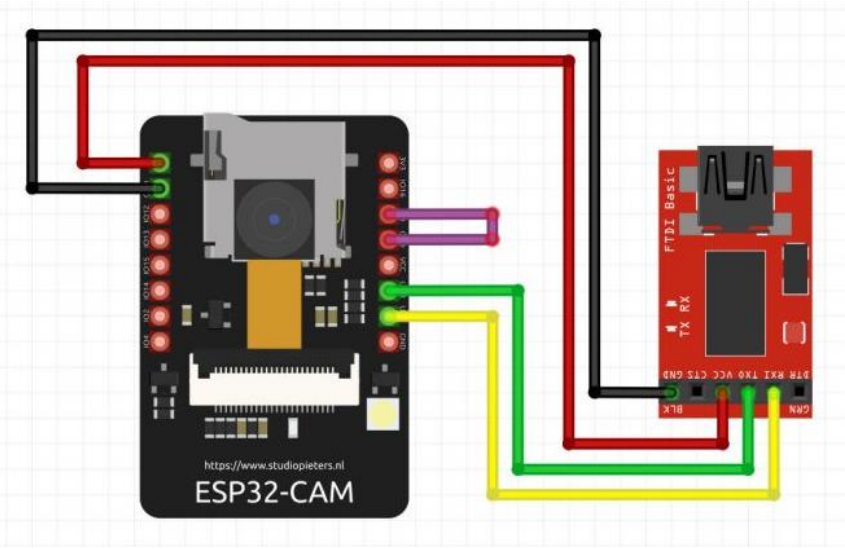

Рисунок 4.6 – Схема підключення контролеру до модуля USB TO TTL

У цій частині коду Tensorflow.js подано для аналізу отриманих зображень у браузері. У наступному рядку також імпортовано моделі COCO-SSD разом із Tensorflow.js.

```
    <script
src="https:\/\/ajax.googleapis.com/ajax/libs/jquery/1.8.0/
jquery.min.js"></script>
    <scriptsrc="https:\/\/cdn.jsdelivr.net/npm/@tensorflow/tfjs@1.
3.1/
dist/tf.min.js"></script>
    <script       src="https:\/\/cdn.jsdelivr.net/npm/@tensorflow-
models/
coco-ssd@2.1.0"></script>
```

Приклад 4.2 – Код для імпорту моделі COCO-SSD

У наступному розділі потрібно подати моделі так, щоб їх можна було розпізнати в результаті обробки отриманого зображення.

Після того, як пристрій виявить об’єкт на зображенні, тепер він має бути видимим для користувача у вигляді квадратів навколо виявленого зображення, для цього призначений наступний код із Прикладу 4.3.

```
function ObjectDetect() {
 result.innerHTML = "Please wait for loading model.";
  cocoSsd.load().then(cocoSsd_Model => {
    Model = cocoSsd_Model;
    result.innerHTML = "";
    getStill.style.display = "block";
  });
}
```

Приклад 4.3 – Код подання моделелей для обробки

Приклад 4.4 описує малювання рамок навколо визначених об’єктів. При роботі алгоритм зберігає дані про визначені об’єкти у контейнері Predictions класу Model, а саме клас визначеного об’єкту, його координати на зображенні, розмір. Відповідно кожному об’єкту у контейнері Predictions малюється рамка.

```
if (Predictions.length>0) {
          result.innerHTML = "";
          for (var i=0;i<Predictions.length;i++) {
            const x = Predictions[i].bbox[0];
            const y = Predictions[i].bbox[1];
            const width = Predictions[i].bbox[2];
            const height = Predictions[i].bbox[3];
            context.lineWidth = Math.round(s/200);
            context.strokeStyle = "#00FFFF";
            context.beginPath();
            context.rect(x, y, width, height);
            context.stroke();
            context.lineWidth = "2";
            context.fillStyle = "red";
            context.font = Math.round(s/30) + "px Arial";
            context.fillText(Predictions[i].class, x, y);
```

Приклад 4.4 – Код показу квадратів навколо виявленого зображення

Наштування у браузерному вікні (рис. 4.7) дозволяють змінювати деякі параметри камери, а саме налаштувати якість вихідного зображеня, його розмір, контрастність зображення та параметр ISO, що означає чутливість матриці модуля камери, відзеркатити зображення та вибрати об’єкт для підрахування

кількості елементів. Також є можливість регулювання яскравості світлодіоду на платі який виконує функції підсвічування навколишнього середовища при недостатній видимості.

У готовому пристрої такий функціонал є баластом для роботи системи , в подальшому такі параметри будуть змінюватися автоматично в залежності від інформації від датчиків світла.

Модель з TensorFlow Lite дозволяє визначати доволі багато типів об’єктів, що є дуже корисним для охоронної системи.

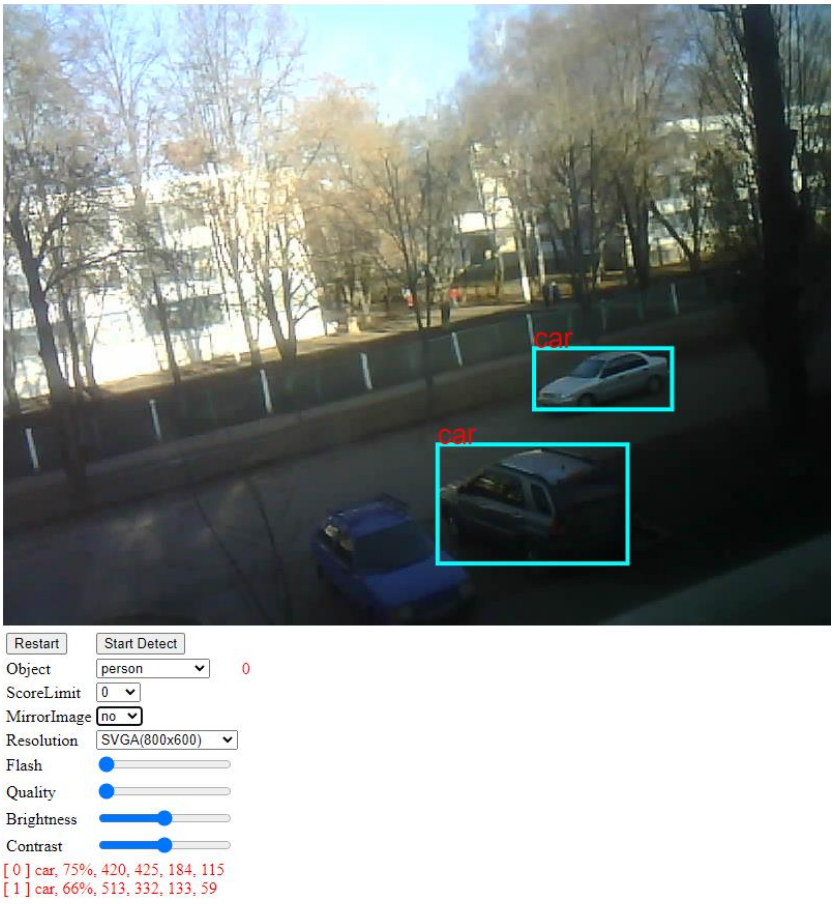

Рисунок 4.7 – Налаштування інтерфейсу для виявлення об’єктів

Дана модель COCO-SSD дозволяє виявляти такі типи об’єктів: людина, велосипед, автомобіль, мотоцикл, літак, поїзд, вантажівка, човен, світлофор, пожежний гідрант, знак зупинки, паркомат, лава, птах, кіт, собака, кінь, вівця, корова, слон, ведмідь, зебра, жираф, рюкзак, парасолька, сумочка, краватка,

валіза, фрісбі, лижі, сноуборд, спортивний м'яч, повітряний змій, бейсбольна бита, бейсбольна рукавичка, скейтборд, дошка для серфінгу, тенісна ракетка, пляшка, чарка, чашка, веделка, книга, годинник, ваза, ножиці , ведмедик, фен, зубна щітка.

Ця модель є портом TensorFlow.js моделі COCO-SSD.

Модель виявляє об’єкти, визначені в наборі даних COCO, який є великомасштабним набором даних виявлення, сегментації та субтитрів. Модель здатна виявляти 80 класів об’єктів. (SSD означає Single Shot MultiBox Detection).

Модель не вимагає знань про машинне навчання. Вона може приймати вхідні дані як будь-які елементи зображення (наприклад, елементи <img>, <video>, <canvas>) і повертає масив обмежуючих рамок з назвою класу та рівнем достовірності.

```
else if (cmd=="quality") {
    sensor_t * s = esp_camera_sensor_get();
    int val = P1.toInt();
    s->set_quality(s, val);
  }
  else if (cmd=="contrast") {
    sensor_t * s = esp_camera_sensor_get();
    int val = P1.toInt();
    s->set_contrast(s, val);
  }
  else if (cmd=="brightness") {
    sensor_t * s = esp_camera_sensor_get();
    int val = P1.toInt();
    s->set_brightness(s, val);
  }
```

Приклад 4.6 – Код обробки налаштувань камери

Так як розпізнавання об’єктів працює, потрібно додати реалізацію відправки повідомлень, як пирклад можна використати світлове сповіщення, щоб дати інформацію, що сторонній об’єкт був знайдений. Також можна реалізувати відправку POST повідомлення до серверу,що буде керувати ботом у Telegram, який у зручному форматі зробить повідомлення про стан охоронної системи. Також у даному блоці коду (прикл 4.6) можлива реалізація відправки

через послідовний порт інформації до контролеру польоту для подальшого відстеження об’єкту та зміни маршруту польоту

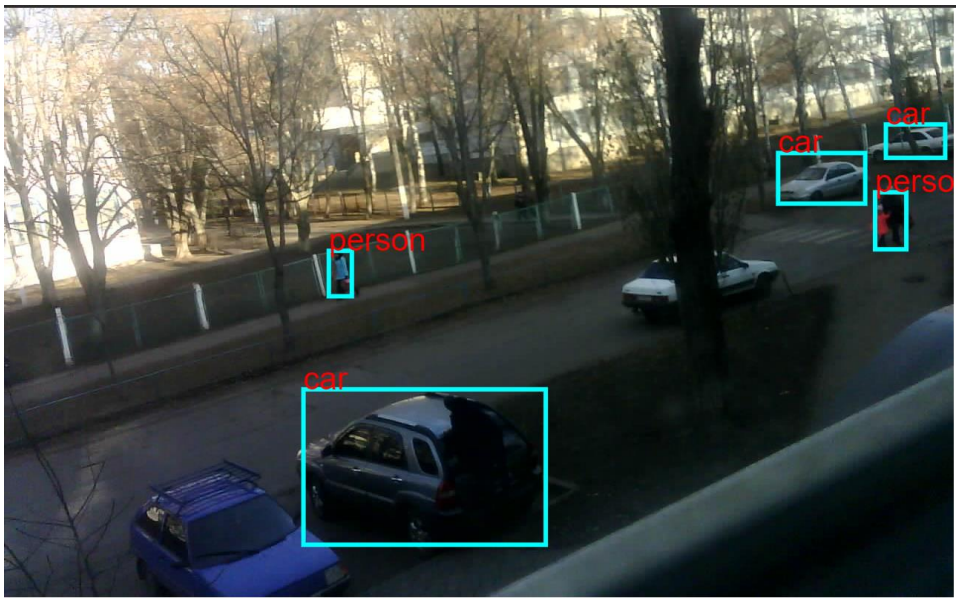

Рисунок 4.8 – Приклад виявлення об’єктів

```
void SendingAnAlert()
{
    Serial.println("Person detected");

    digitalWrite(33, HIGH);
    delay(100);
    digitalWrite(33, LOW);
    // other code for alert
}
```

Приклад 4.7 – Код для повідомлень

Для взаємодіії виводів з кодом TensorFlow можна використовувати інформаційні змінні, данні з них також виводяться до послідовного порту COM (рис. 4.9).

Цих даннх достатньо для написання функцій більш розширених ніж SendingAnAlert() для прийняття дій при виявленні об’єктів.

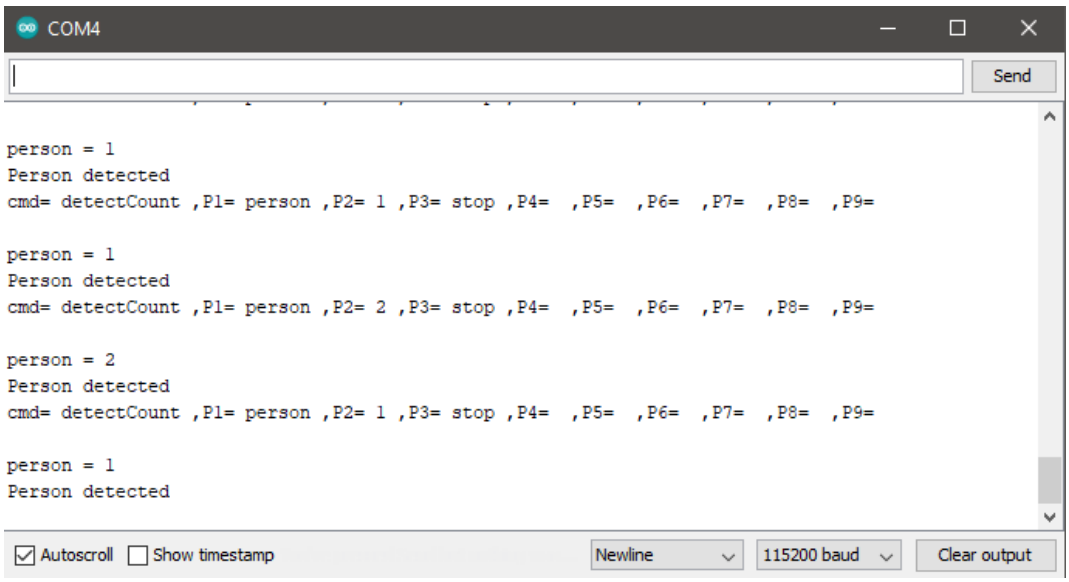

Рисунок 4.9 – Вивід інформації до послідовного порту

Інформація що передається до послідовного порту включає у себе стан зміни важелів керування налаштувань з веб інтерфейсу, а також кількість розпізнаних об’єктів у кадрі.

Повну реалізацію коду можна знайти за посиланням «https://github.com/cmdwww/ESP32Cam_Tensorflow». Репозиторій містить у собі проект для Arduino IDE, з відповідними налаштуваннями для ESP32 Cam,

ВИСНОВКИ

При виконанні кваліфікаційної роботи була розроблена схема пристрою, яка виконує функції інтелектуальної системи охорони. Були розглянуті можливі компоненти для розробки, їх переваги та недоліки. Також були визначені технології за допомогою яких можливо реалізувати технологію комп’ютерного зору на базі мікроконтролера ESP32 Cam, а саме технологія TensorFlow від Google. Також була представлена модель пристрою у вигляді літаючого дрона, що буде виконувати функції переміщення основного мікроконтролера з комп’ютерним зором до точок сканування місцевості на можливі погрози. Було розглянуто аналогічні схожі за концептом проекти.

Один такий літаючий дрон наприклад зможе виконувати усі функції охоронних систем, замінивши купу датчиків та камер, при цьому не потребуючи персоналу для обслуговування, та виконуючи завдання охорони автономно.

Також розроблено програмне забезпечення для мікроконтролерного модуля ESP32 Cam, за основу була взята бібліотека TensorFlow Lite яка спеціально створена для використання технологій комп’ютерного зору у пристроях з малою кількістю пам’яті.

За результатами тестування пристрою, було виявлено ряд переваг та недоліків, Модуль ESP32 Cam має доволі швидкий процесор для обробки зображень використовуючи TensorFlow, але при високій частоті роботи мікроконтролер нагрівається і це негативно впливає на стабільність роботи. Тому для стабільної роботи потрібно або зменшувати частоту та розмір кадрів які буде обробляти пристрій, або налаштовувати систему охолодження, що збільшить габарити. Серед переваг це простота пристрою, швидкість зміни програмного забезпечення та можливість встановлення його на будь який макет дрону.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Волков Д. П. Інтелектуальні системи у задачах навігації безпілотних літальних апаратів : тези VII всеукр. наук.-практ. конф. студентів, аспірантів та молодих вчених. м. Харків, 2020. С. 184–186.

2. Елементи і пристрої фізичної та електронної охорони об’єктів: конспект лекцій. Львів : Фенікс, 2000. 186 с.

3. What is computer vision? URL: https://www.ibm.com/topics/computer-vision (дата звернення: 14.11.2021).

4. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. California: O'Reilly Media Inc., 2019. 851 p.

5. CS231n Convolutional Neural Networks for Visual Recognition. URL: https://cs231n.github.io/convolutional-networks/ (дата звернення: 15.11.2021).

6. KC Tung. Learn TensorFlow Enterprise. Birmingham: Packt Publishing Ltd., 2020. 420 p.

7. TensorFlow-Lite and Platform.io URL: https://github.com/atomic14/tensorflow-lite-esp32 (дата звернення: 17.11.2021).

8. Object Detection System with ESP32-CAM and Tensorflow. URL: https://cifertech.net/object-detection-system-with-esp32-cam-and-tensorflow/ (дата звернення: 18.11.2021).

9. Prata S. C++ Primer Plus. Boston: Addison-Wesley, 2012. 1420 p.

10. Stroustrup B. The C++ programming language. Boston: Addison-Wesley 2013. 869 p.

11. Волков Д. П. Інтелектуальна система безпеки з використанням комп’ютерного зору: тези XXV міжнар. молод. форуму *Радіоелектроніка і молодь у XXI столітті*. конф. *Комп’ютерні системи і мережі управління та обробки даних*. м.Харків, 2021.

12. Ring’s latest security camera is a drone that flies around inside your house. URL: https://www.theverge.com/2020/9/24/21453709/ring-always-home-cam-

indoor-drone-security-camera-price-specs-features-amazon. (дата звернення: 18.11.2021).

13. Рендал Б., МакЛэйн Т. Малые беспилотные летательные аппараты. Теория и практика. Москва: Техносфера 2015. 312 с.

14. Mattmann C. Machine Learning with TensorFlow. Shelter Island: Manning, 2020. 773 p.

15. Trask A. Grokking Deep Learning. New York: Simon and Schuster, 2019. 336 p.