КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інформаційних систем та технологій

Спеціальність 126 – Інформаційні системи та технології
Освітня програма «Програмні технології інтернет речей»

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему:

«Інформаційно-аналітична система управління логістичною складовою транспортної компанії»

Студента 2-го курсу групи ІРма-21

Якова КІРЄЄВА
(прізвище, ім'я, по батькові)

(підпис студента)

Науковий керівник:

Д.т.н., доцент
(науковий ступінь, вчене звання)

Андрій ОНИЩЕНКО
(прізвище, ім'я, по батькові)

(дата) (підпис)

Попередній захист:

(Висновок: "До захисту в Екзаменаційній комісії")

Завідувач кафедри інформаційних систем та технологій _____ Олександр КУЧАНСЬКИЙ _____

(підпис) (прізвище, ініціали) (дата)

Київ 2022

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра Інформаційні системи та технології
Освітній рівень Магістр
Спеціальність 126 Інформаційні системи та технології
Освітня програма Програмні технології інтернет речей

ЗАТВЕРДЖУЮ
Завідувач кафедри
Д.т.н., доцент Олександр КУЧАНСЬКИЙ

_____
«__» _____ 2022 року

ЗАВДАННЯ
НА ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Студент: Кірєєв Яків Сергійович
Група: **ІРма-21**

1. **Тема дипломної роботи**: «Інформаційно-аналітична система управління логістичною складовою транспортної компанії».

Затверджена протоколом засідання кафедри ІСТ №4 від «28» листопада 2021 р.

2. **Строк подання студентом готової роботи** – «11» травня 2022 р.

3. **Цільова установка та вихідні дані до роботи**: дослідження особливостей використання методів та інструментів для програмування та проектування логістичної системи з можливістю керуванням безпілотним літальним апаратом.

4. **Зміст роботи**: маркетинговий аналіз та аналіз середовища проекту, аналіз існуючих рішень у сфері IoT, аналіз існуючих методів проектування мережних систем, аналіз користувацьких рішень та пристроїв.

5. **Перелік графічного матеріалу**: приклади моделювання проектних рішень, графіки аналізу станів проекту, спроектований додаток користувача, функціонал керування дроном, графічне відображення контролю логістичної складової та польоту дрона.

6. **Календарний план виконання роботи:**

Етапи виконання дипломних робіт	**Термін виконання**
1. Вибір теми дипломної роботи	28.10.21

2. Наказ про затвердження тем дипломних робіт та призначення наукових керівників	28.11.21
3. Формування переліку нормативних матеріалів, літератури з проблематики дипломної роботи	17.01.22
4. Розробка плану дипломної роботи і його погодження з науковим керівником	07.02.22
5. Написання І розділу дипломної роботи	11.03.22
6. Написання ІІ розділу дипломної роботи	27.03.22
7. Написання ІІІ розділу дипломної роботи	20.04.22
8. Написання IV розділу дипломної роботи	30.04.22
8. Підготовка висновків і пропозицій	06.05.22
9. Попередній захист дипломної роботи	11.05.22

Дата видачі завдання «____» _____2021 р.

Керівник роботи: д.т.н., доцент Андрій ОНИЩЕНКО _____ (підпис)

Завдання прийняв до виконання:

студент групи **ІРма-21** Яків КІРЄЄВ _____(підпис)

АНОТАЦІЯ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра Інформаційних систем та технологій
Освітня програма «Програмні технології інтернет речей»

Дипломна робота магістра Кірєєва Якова Сергійовича

Тема роботи: «Інформаційно-аналітична система управління логістичною складовою транспортної компанії».

Мета дипломної роботи магістра – дослідження тенденцій розвитку логістики з використанням технологій IoT та написання функціоналу для запуску дрона в межах логістичної складової транспортної компанії.

Об'єкт дослідження – процес управління логістичною діяльністю підприємства що займається реалізацією IoT пристроїв, позитивні тенденції впливу впровадження перспективних мережних рішень в транспортних компаніях

Предмет дослідження – теоретичні та методичні аспекти, практичний інструментарій управління логістичною діяльністю підприємства.

Методи дослідження – аналіз сучасних рішень на базі проведення практики, аналіз перспективності мережних пристроїв, сучасних потреб в логістичній складовій підприємства, аналіз бібліотеки мови програмування Python, завдяки чому був проаналізований функціональний додаток з керування безпілотним літаючим апаратом.

Теоретичне підґрунтя – проходження практики на базі ТОВ «ДІСТІ ПРО», аналіз сучасних проектних рішень з IoT та їх впровадження на підприємстві.

Методологічною базою роботи є загальнонаукові принципи проведення досліджень, теоретичні й методичні основи системного підходу, методи експертних оцінок та ринкового аналізу.

Дипломна робота складається зі змісту, вступу, основної частини, яка включає чотири розділи, висновків та списку використаних джерел. Всього 130 сторінок.

Ключові слова: IOT, Python, ООР, логістика, логістична діяльність, удосконалення логістичної діяльності, оптимізація.

ABSTRACT

TARAS SHEVCHENKO KYIV NATIONAL UNIVERSITY

Faculty of Information Technology

Department of Information Systems and Technologies

Educational program "Software technologies of the Internet of Things"

Master's degree paper of Kirieiev Yakiv Serhiyovych

R&D: "Information-analytical management system of the logistics component of the transport company".

The purpose of the master's thesis is to study the trends in logistics with the use of IoT technologies and write a functional for launching a drone within the logistics component of the transport company.

The object of research - the process of managing the logistics activities of the enterprise engaged in the implementation of IoT devices, positive trends in the impact of the implementation of promising network solutions in transport companies

The subject of research - theoretical and methodological aspects, practical tools for managing the logistics activities of the enterprise.

Research methods - analysis of modern solutions based on practice, analysis of prospects of network devices, current needs in the logistics component of the enterprise, analysis of the Python programming language library, thanks to which the functional application for unmanned aerial vehicle control was analyzed.

Theoretical basis - internship on the basis of LLC "DISTI PRO", analysis of modern design solutions for IoT and their implementation at the enterprise.

The methodological basis of the work is general scientific principles of research, theoretical and methodological foundations of a systematic approach, methods of expert evaluation and market analysis.

Thesis consists of content, introduction, main part, which includes four sections, conclusions and a list of sources used. Only 110 pages.

Key words: IOT, Python, OOP, logistics, logistics activity, improvement of logistics activity, optimization.

ЗМІСТ

ВСТУП

У даній роботі було спроектовано, досліджено та задокументовано систему критично важливої мережевої інфраструктури для дистанційного керування логістичною складовою в транспортній компанії з використанням технологій IoT. Запропоновано актуальні рішення щодо проектування даних систем з використанням сучасної техніки від закордонних постачальників комунікаційного обладнання. Написано та протестовано інтерфейс користувача для програмування маршрутів дрона. Таким чином, був створений комплекс системи контролю логістики на дорозі та в повітрі, що показали себе як успішне рішення в різних сферах застосування транспортних засобів й логістики. Пристрої Teltonika пропонують розширені функції програмного забезпечення, такі як функції автоматизації та можливості віддаленого керування, які допомагають заощадити час і, в даному випадку, підвищують безпеку рішення.

Ця робота спрямована на вивчення впливу інформаційних технологій на конкурентоспроможність компаній/ланцюгів постачання. Тобто вона має намір проаналізувати, чи сприяє впровадження інформаційних технологій покращенню витрат, часу та обслуговування клієнтів. Для досягнення цієї мети використовується кейс та методологія аналізу контенту. Результати показують, що існує позитивний зв'язок між впровадженням технологій, а саме RFID, A.R. і конкурентоспроможність компаній/ланцюгів постачання. За допомогою цих технологій можна покращити декілька процесів у ланцюжку поставок, а також зменшити витрати, пов'язані з оплатою праці, покращити запаси та управління транспортними каналами, а також покращити обслуговування клієнтів. Поліпшення цих показників має, як наслідок, підвищення конкурентоспроможності, що дозволяє компаніям диференційовано реагувати на потреби ринку, підвищуючи тим самим задоволеність клієнтів. Оскільки інформаційні технології вважалися дуже важливими для бізнесу, дуже важливо покращити наше розуміння основних переваг, пов'язаних із ними, та їхнього внеску в підвищення конкурентоспроможності компаній та ланцюгів поставок, тому ця стаття є важливим внеском для науковців та професіоналів у цій галузі. поле.

Управління логістикою — це компонент управління ланцюгом поставок, який використовується для задоволення запитів клієнтів шляхом планування, контролю та реалізації ефективного переміщення та зберігання відповідної інформації, товарів і послуг від місця відправлення до пункту призначення. Без системи логістичного управління постачальники логістичних послуг стикаються

з багатьма проблемами. Деякі з них: обслуговування клієнтів стає неефективним, вартість транспортування товарів може зрости, запуск нових продуктів може бути під загрозою. Отже, програмне забезпечення системи управління логістикою вирішує всю цю проблему і надає багато функцій, таких як управління складом, управління парком, обробка замовлень, контроль запасів.

Кваліфікаційна робота магістра складається з вступу, чотирьох розділів, висновків, списку використаних джерел, додатків. Загальний обсяг роботи складає 128 сторінок, у тому числі 114 сторінок основного тексту. Робота містить 30 рисунків, 1 додаток, список використаних джерел складається з 40 найменувань.

Актуальність обраної теми. Більшість промислових IoT-проектів знаходиться за межами міської інфраструктури. Ще один значний сектор — це тимчасові або спливаючі рішення, не прив'язані до одного місця. Перші логістичні компанії, які зможуть використовувати дрони, різко скоротять транспортні витрати, пов’язані з капітальними витратами вантажівок, такими як витрати на технічне обслуговування, паливо, страхування тощо.

Наукова новизна. Було запропоновано нові способи імплементації дронів для доставки в логістичну складову транспортної компанії за допомогою інтернет речей з використанням безпілотних літальних апаратів (БЛА), що дозволило підвищити безпеку персоналу транспортних компаній, ефективність критичної мережевої інфраструктури, а також зменшити витрати на персонал під час доставки товарів методом "останньої милі".

Використання в роботі сучасних досягнень науки і техніки. В роботі побудована архітектура проєкту оптимізації доставки дронами на базі IoT, а саме бази мікропроцесорних контролерів Raspberry Pi, платформи ArduPilot, дронів, LTE/Bluetooth модулів стільникових роутерів, використовувалось середовище об’єктного-орієнтованого програмування Python IDE, застосовувались алгоритми бібліотеки управління дронами DroneKit, польоту, побудови маршрутів та доставки, бібліотека веб додатку Cherry Py. Зазначені технології відносяться до сучасних трендів при розробці систем інтернет речей.

Метою роботи є розгляд теоретичних положень та розроблення практичних рекомендацій з удосконалення надання транспортних послуг, сільського господарства та харчової промисловості для підвищення ефективності управління логістичною діяльністю підприємства.

Об'єктом дослідження є процес управління логістичною діяльністю аграрного підприємства. Предметом дослідження є теоретичні та методичні аспекти, практичний інструментарій управління логістичною діяльністю підприємства.

База дослідження – практика на підприємстві ТОВ «ДІСТІ ПРО».

Методи дослідження. Виконання дипломної роботи другого (магістерського) рівня вищої освіти здійснено на застосуванні загальних та спеціальних методів дослідження: статистичний аналіз, систематизації та узагальнення (для діагностики логістичної діяльності підприємства).

Практична значущість магістерської роботи. Результати цієї роботи показують, що є потенційні переваги у застосування дронів в логістичній складовою транспортної компанії: підтримка гуманітарної логістики, скорочення часу доставки, зниження вартості, покращена гнучкість і підвищена стійкість. Проблеми, пов'язані з безпілотниками в логістиці, згруповані в технічні, організаційні, пов'язані з безпекою питання. Також запропонована система нівелює всі проблеми виявлені в існуючих системах і дозволяє виконувати легке і швидке масштабування за рахунок IoT. Результати рекомендується включити на виробництво. За результатами проведеного дослідження розроблено проект з модернізації програмного забезпечення логістичної діяльності, який передбачає покращення процесу доставки товару. Економічний ефект від впровадження запропонованих заходів полягає в економії витрат на доставку товару та організацію ланцюгів постачання. Результатом для логістичної системи є модернізація існуючого програмного забезпечення.

Рекомендації щодо використання результатів роботи. Результати дослідження можуть бути використані вітчизняними підприємствами з надання транспортних послуг, сільського господарства та харчової промисловості для підвищення ефективності управління логістичною діяльністю підприємства, зокрема ТОВ «ДІСТІ ПРО» для покращення процесу доставки товару, економії часу та зниження витрат на доплату за понаднормову працю.

Результати впровадження досліджень. Розроблені в дипломній роботі пропозиції були представлені на розгляд керівному складу ТОВ «ДІСТІ ПРО» , де було визнано можливість їх практичного застосування.

Ця робота забезпечує чітке розуміння та глибоке уявлення про вимоги, цілі, робочий процес, дизайн та реалізацію цільової програми. Вона вичерпно описує рішення, запропоноване в контексті IOT, розроблені послуги та алгоритми з точки зору, орієнтованої на застосування. У цьому відношенні він зосереджується на якості даних і аспектах конфіденційності, а також на описовій аналітиці та деталях реалізації графічного інтерфейсу користувача GUI.

Отриманий результат можливість модернізації існуючого на підприємстві виробничого обладнання та системи управління логістичною складовою до вимог концепції IoT без їх повної заміни, що дозволить значно спросити процес переходу та здешевити його.

Положення, викладені у роботі пройшли апробацію на 88-й Міжнародній науковій конференції молодих учених, аспірантів і студентів «Наукові здобутки молоді – вирішенню проблем харчування людства у XXI столітті», що проводилась у м. Київ, 20 квітня 2022 р.

Публікація, пов’язана з роботою - Кірєєв Я.С., Онищенко А.М. «Розвиток критичної інфраструктури в закладах охорони здоров'я з використанням IoT-технологій»: зб. тез. 88-ї Міжнародної наукової конференції молодих учених, аспірантів і студентів «Наукові здобутки молоді – вирішенню проблем харчування людства у XXI столітті», м. Київ, 20 квітня 2022р.

РОЗДІЛ 1. АНАЛІЗ ІСНУЮЧИХ ПРИСТРОЇВ, ДАТЧИКІВ, ТЕХНОЛОГІЙ І РІШЕНЬ

1.1 Аналіз інформації

Технічний компонент, який лежав в основі організації та пошуку інформації, слід поставити під сумнів у світлі нової наукової парадигми, інтегруючи такі операції за допомогою цілісного підходу, що полегшує легкий доступ до інформації. Ця робота має на меті пролити світло на концепцію інформаційної системи для управління організаціями в обох гносеологічних онтологічних термінах, незалежно від допоміжних технологій, тим самим сприяти створенню міцної основи для обговорення цієї концепції. У глобальній економіці інформація та знання є найбільшими конкурентними перевагами, які можуть мати організації. За останні роки ми стали свідками надзвичайної трансформації суспільства, тобто ми перейшли від суспільства, заснованого на промисловості та транспорті, до суспільства, заснованого на знаннях.

Одним з основних завдань для керівництва є розуміння того, що означає інформація: як нею керувати та інтерпретувати, і які рішення вона дозволяє приймати в епоху всесвітніх комунікацій, оскільки інформація є ланкою, яка нас об'єднує. Завдяки можливості швидко передавати велику кількість даних через континенти, ми перетворюємо світ на глобальний мегаполіс.

Ще одна проблема, з якою стикаються менеджери, — це багатство інформації в сучасному суспільстві, чиїм найочевиднішим сигналом лиха в такому суспільстві є поєднання виробництва великої кількості інформації, інтенсивного використання інформаційних технологій та комунікації та безперервного процесу навчання. Артикуляція цих трьох аспектів свідчить про те, що інформаційне суспільство швидко перейшло в суспільство знання. Символічна культура цього суспільства вимагає нових видів навчання, організації та управління, а отже, і для управління інформацією.

В суспільстві інформації та знань існує кілька ієрархічних рівнів або прогресивних етапів, безпосередньо пов'язаних із процесом навчання цих знань. Таким чином, можна виділити три стадії: дані, інформація та знання. Посильний пішки чи на коні поступався дорогою інформації. Отже, про що ж тоді вся інформація? Будь-який ресурс може стати цінним для збирання, збереження, копіювання, продажу, крадіжки і навіть іноді мотивом вбивства.

Багато людей в організаціях проводять свій робочий день, збираючи, вивчаючи та обробляючи інформацію. Деякі галузі розвиваються на основі

інформаційного ресурсу для виробництва технологій (технологія процесу - комп'ютер, технологія продукції - програмне забезпечення та комунікаційні технології - комунікаційне обладнання + програмне забезпечення) з метою збору, зберігання, обробки, передачі та легкого доступу до інформації.

Менеджери не можуть відкрити газету, не зіткнувшись із терміном «інформація». У незліченній кількості книг є слово «інформація». Багато людей в організаціях виконують діяльність, пов'язану зі словом «інформація». Здається, легко описати, з чого він складається. Однак, коли ми починаємо думати про термін «інформація», ми відчуваємо певні труднощі з пошуком відповідного визначення. Частково через те, що менеджери не розуміють інформацію, тому що вони настільки звикли працювати з нею щодня, що не знають про складнощі. Менеджери усвідомлюють труднощі лише тоді, коли стикаються з новою мовою.

Можливість неправильного тлумачення завжди є. Враховуючи важливу роль, яку відіграє комунікація в організації, ті, хто бере участь у прийнятті рішень, повинні знайти способи мінімізувати ймовірність помилки. Для цього необхідно зрозуміти, як розвивається комунікація – як інформація передається від людини до людини, від комп'ютера до комп'ютера, а також між людьми та комп'ютерами. Потреба в розумінні інформації – що це таке і як вона надходить – не обмежується лише великими організаціями. Щоразу, коли людина спілкується з іншим, ми отримуємо потік інформації, оскільки спілкування є засобом надання інформації іншим. Незважаючи на те, що терміни «інформація» і «знання» використовуються з багато змінною частотою, вони не означають одне й те саме. Оскільки «інформація» – це не те саме, що «дані», незважаючи на те, що ці два слова часто плутають. Тому важливо розрізняти тонкі відмінності між цими двома поняттями. Дані зазвичай надають газети, звіти та комп'ютерне програмне забезпечення. Наприклад, список фондових бірж на сторінці фінансових новин становить дані. Коли дані отримуються і вводяться в глобальну інформаційну структуру, отриману раніше, ці дані перетворюються в інформацію. Тож коли хтось читає прайс-лист газети, то отримує інформацію про різні компанії. Що дозволяє менеджерам отримувати інформацію з даних журналу, так це їх попередні знання про те, що означають ці значення та як працює фондовий ринок.

Дані не несуть значення чи значення фактів, образів чи звуків, оскільки в них відсутні елементи відношення.

Інформація = дані + значення (1.1)

Інформація є результатом додавання до даних певної моделі відносин, що встановлюють їх формат. Діяти на основі інформації – це не тільки діяти на

основі даних, які її складають, а й діяти відповідно до встановлених відносин, тобто відповідно до колективних або індивідуальних стандартів форматування, а через них – на те, як сприймається реальність і подальші дії.

Інформація – це величина, яка вимірює та/або зменшує невизначеність: тобто все, що зменшує невизначеність, яку відчуває спостерігач щодо настання певної події. Інформація є продуктом ходу перетворення та зв'язку даних або інших інформаційних операцій, за допомогою яких підсумовуються значущі концептуальні відносини між елементами з урахуванням певної мети комунікації. Інформація відноситься до сукупності фактів та/або подій у зручному форматі прийняття рішень у контексті, який визначає відносини між даними.

Поняття інформації можна зрозуміти з багатьох різних точок зору. Інформація – це об'єкт, створений людьми, щоб представляти для них подію в реальному світі, що об'єднує та пов'язує набір записів або даних.

Коли інформація засвоюється в тій мірі, в якій вона може бути використана, її називають знанням. Це плавне поєднання досвіду, цінностей, контекстної інформації та експертного розуміння, і забезпечує основу для оцінки та включення нового досвіду та інформації. В організаціях він зустрічається не тільки в документах і звітах, але і в організаційних процедурах, процесах, практиках і нормах. Знання зароджуються і застосовуються в умах авторитетів у цій галузі.

Знання – це інформація, яка сприймається як достовірна і приймається шляхом інтеграції даних, дій, інформації, а іноді і припущень. Потреба знати вимагає, щоб хтось отримував, комбінував та інтерпретував інформацію. Інформацію можна розглядати як «субстанцію», яка може бути придбана, збережена і володіння особою або групою та передана від людини до людини або від групи до групи. Інформація має певну стабільність і, мабуть, найкраще вважати її існуючою на рівні суспільства. Тому можна сказати, що інформація живить знання.

Знання = інтерналізована інформація + здатність використовувати її в нових ситуаціях (1.2).

Таким чином, знання є сумішшю інформації, досвіду та знань, які забезпечують структуру, яку можна застосувати для оцінки нової інформації або нових ситуацій. Знання по суті та внутрішньо закладено в людей, які на індивідуальному рівні набагато складніші та непередбачуваніші, ніж суспільство в цілому; отже, не дивно, що отримати знання набагато важче, ніж інформацію.

Знання існують переважно всередині людей, воно є невід'ємною частиною людської складності та непередбачуваності. Знання має фундаментальну подвійність: це щось, що зберігається (принаймні іноді ми маємо намір це зробити) і те, що тече (щось, що передається від людини до людини). Цілком можливо, саме ця подвійність знань (щось, що потоки та процес зберігання) ускладнює їх обробку та керування.

1.2 Види інформації

Усередині організацій менеджери використовують різного роду інформацію для досягнення цілей. Що стосується застосовності до різних рівнів управління, інформацію можна згрупувати наступним чином:

- Інституційна інформація (глобальна): дозволяє топ-менеджерам спостерігати та оцінювати всі змінні, пов'язані з еволюцією середовища та внутрішньою продуктивністю, метою яких є керування та оцінка внутрішньої ефективності організації, а також визначення, реалізація та контроль стратегії.
- Інформація середнього рівня: дозволяє керівникам середньої ланки або менеджерам з координації розподіляти та керувати ресурсами у своїй зоні відповідальності або бізнес-підрозділі; тобто стежити за виконанням своєї сфери відповідальності чи бізнес-підрозділу та виправляти будь-які відхилення від цілей, які необхідно досягти.
- Інформація про оперативний рівень: дає змогу оперативному менеджеру контролювати та контролювати щоденну діяльність і завдання, а також контролювати географічний район, за який він відповідає.

Джерела інформації (походження) можуть належати до таких категорій:

- Офіційне джерело: напр. преса, база даних, наукова (статті) або технічна інформація (патент), документація організацій, замовників, постачальників, держави тощо.
- Неофіційне джерело: напр. семінари, конференції, відвідування клієнтів, виставки, реклама, інформація або навіть чутки про продукти, клієнтів, постачальників тощо.

З точки зору організації інформацію можна згрупувати в:

- Структурована інформація: ті, які слідують заздалегідь визначеному шаблону (наприклад, заповнена форма, бізнес-одиниця тощо).
- Неструктурована інформація: інформація, яка не відповідає попередньо визначеному шаблону (наприклад, стаття в журналі).

Пропонують подальший поділ інформації:

- Інформація про діяльність організацій: та, яка дозволяє організації забезпечити її нормальну роботу (наприклад, замовлення клієнтів, замовлення на покупку, рахунки-фактури, вхідні матеріали складів тощо).
- Соціальна інформація: інформація, яка дозволяє налагоджувати стосунки між людьми і може впливати на їхню поведінку (наприклад, внутрішні інформаційні бюлетені, ділові зустрічі, рекламні та політичні кампанії тощо). Соціальна інформація забезпечує взаємодію між людьми в організації та за її межами, вона найчастіше неструктурована і зазвичай присутня на всіх рівнях організації.
- Стратегічна інформація: відповідна інформація, яка дозволяє організаціям працювати краще за рахунок активного вивчення недосконалості інформації та факторів продукту та ринку.

1.3 Інформаційно-комунікаційні технології

Поняття технології легко зрозуміло тим, хто нею користується і постійно звертається до неї. Існує одностайність щодо неявної концепції, хоча її важливо пояснити, тобто технологія — це складний набір знань, засобів і ноу-хау, організованих з метою виробництва. Таким чином, можна говорити про технології виробництва ІМС високої щільності та вимірювань, які підтримуються глобальною мережею проектно-виробничих центрів, з'єднаних між собою супутниками.

Охоплюючи апаратне забезпечення, програмне забезпечення та все, що стосується комунікацій та архітектур, пов’язаних з усіма цими компонентами, – коротко кажучи, вони є технологічною інфраструктурою інформаційних систем.

Будь-яка технологія складається з трьох компонентів:

1. Знання: самі по собі не є технологією.
2. Медіа: Додайте технологію, але не обмежуйте її; в невмілих руках будь-яка технологія є марною тратою інвестицій.
3. Ноу-хау: це спеціалізація без належних засобів, але ніяких результатів не можна отримати, і вона швидко виходить з ужитку через відсутність застосування.

Щоб покращити свою конкурентну позицію на ринку, організації діють одним із двох способів: з одного боку, необхідно спостерігати та аналізувати споживача, що може призвести до технологічних інновацій, або, з іншого боку,

аналізувати переваги. заміни однієї технології іншою, що дозволяє підвищити їх продуктивність.

Будь-яка технологія завжди звертається до різних наукових дисциплін, наприклад, лазерна технологія, яка поєднує оптичні знання, електроніку, механіку рідин і термодинаміку. Наукові дослідження

має на меті здобуття або вдосконалення знань (тимчасова визначеність), а створення технологій — на виробництво в промислових умовах. Технологія має сенс лише з точки зору гарантованого результату: технологія існує лише тоді, коли її перевіряють і коли вона забезпечує виробництво за певних умов, тобто технологія вирішує проблему.

Інформаційно-комунікаційні технології можна визначити як сукупність знань, матеріальних ресурсів (інфраструктури) та ноу-хау, необхідних для виробництва, маркетингу та/або використання товарів і послуг, пов'язаних з тимчасовим або постійним зберіганням даних, а також їх обробка та передача.

Поява та розвиток технологій є вирішальним поштовхом до появи нових форм і стратегій для вирішення проблем, пов'язаних із тим, як працює конкуренція. Використання інформаційно-комунікаційних технологій поступово розширюється настільки, що англійський вираз «Information Systems» не є систематичним, повним і організованим способом збору, сортування, обробки, аналізу, поширення інформації організаціями та підтримки. - прийняття рішень керівниками.

Інформаційно-комунікаційні технології дозволяють зберігати, обробляти, отримувати доступ і передавати інформаційні потоки; тому процес (апаратне забезпечення) і технологію продукту (програмне забезпечення) не слід плутати з самим продуктом (інформацією).

Розуміння різниці між тим, що є інформацією для управління бізнесом, та інформаційно-комунікаційними технологіями є життєво важливим для менеджерів з тієї простої причини, що така інформація допомагає їм приймати рішення, незалежно від допоміжної технології. Проте керівники також не повинні забувати, що інформаційно-комунікаційні технології є засобом підтримки інформаційної системи. Не вся інформація, перетворена в дані, збирається, обробляється, зберігається і поширюється з використанням

інформаційно-комунікаційних технологій (комп’ютерів і комунікаційних мереж). Значна інформація про діяльність організацій збирається, обробляється, зберігається і поширюється за допомогою олівцевої та паперової технології (ручна технологія). Таким чином, різниця між інформаційними технологіями та інформацією може бути втілена в таких аспектах:

• Комп'ютер: технологія процесу, тобто здатність виконувати інструкції програм;

• Програмне забезпечення: Технологія продукту (група програм, які налаштовують на роботу всі компоненти комп'ютера, тобто це форма людського вираження поведінки машини: комп'ютер);

• Комунікаційні мережі: відповідає технології, яка дозволяє передавати дані від однієї технології до іншої.

• Організація: як люди збираються разом для виконання процедур збору, сортування, обробки, аналізу та отримання результатів (інформації).

• Інформація: продукт (нематеріальний актив, який допомагає приймати рішення).

• Люди: менеджери, які приймають рішення.

1.4 Індустрія 4.0

Сьогодні багато сфер, включаючи пов’язані суспільства, наукові дослідження, інформаційні технології та бізнес, стикаються з серйозними проблемами, пов’язаними з великими даними, пов’язаними з неоднорідністю даних, високою швидкістю виробництва даних, а також величезним обсягом даних, створених на різних рівнях і різними організаціями. Потік даних повсюдно впливає на соціальні контексти , будучи безцінним джерелом інформації для вирішення проблем добробуту жителів міст. У контексті розумного міста управління розумним міським транспортом можна розглядати як дуже складний і багатогранний виклик великих даних. Він сильно покладається на інформацію, зібрану в множинних, широко поширених і різноманітних джерелах даних, а також на здатність витягувати з них корисні ідеї та цілісне розуміння, часто використовуючи методи аналізу даних і машинного навчання . Крім даних, для вирішення всіх технічних проблем і задоволення вимог користувачів, необхідно спеціально прийняти комплексне рішення програмного стека (від платформи до сервісів і додатків). В даний час зростаюча доступність платформ великих даних і постачальників хмарних обчислень, часто

з власними версіями готових до використання рішень для великих даних, представляє чудову можливість для отримання цінності з даних і досягнення стратегічного прогресу в багатьох секторах.

Процес цифровізації – це масштабний проект, метою якого є створення нового мережево-інформаційного суспільства, керованого інформаційно-комунікаційними технологіями, які знаходять, збирають, обробляють та поширюють інформацію через глобальні телекомунікаційні мережі. Сьогодні ми спостерігаємо так звану «Четверту промислову революцію» (Індустрія 4.0). Термін був введений у 2011 році як частина стратегії Hi-Tech в Німеччині. Індустрія 4.0 характеризується розвитком обміну даними та автоматизацією та включає впровадження кіберфізичних систем, Інтернету речей, хмарних обчислень, використання безпілотних транспортних засобів та 3D-друку.

Розглядаючи цифрову економіку крізь призму виробництва, як основні напрямки розвитку можна відзначити впровадження нових матеріалів, перехід на нові технології виробництва товарів, автоматизацію виробничих процесів та застосування інновацій у сфері логістики. . Основною відмінністю сучасної економіки від цифрової є ступінь використання прогнозування в економічних процесах. До впровадження технологій підприємства оцінювали ефективність виробничого процесу шляхом розрахунку відповідних результатів фактично, тобто після виробництва.

Технологія Інтернету речей існує вже більше десяти років, але її широке поширення починає проявлятися зараз. Майже кожна галузь впроваджує інновації, пов’язані з цією технологією та пов’язаними з ними мобільними додатками. Однак IoT в логістиці забезпечує найбільш помітні випадки використання.

Зараз зростає тенденція до грамотного планування виробництва з метою зниження витрат і найбільш ефективного розподілу наявних ресурсів. Поряд із традиційною логістикою (відділення поштою або кур'єрською доставкою) розвивається доставка дронів, роботів, безпілотних транспортних засобів. Все це стало можливим з появою технології Інтернету речей. (IoT).

1.5 Методи в IoT

Саму концепцію та термін IoT представив Кевін Ештон - засновник дослідницької групи Auto-ID в Массачусетському технологічному інституті в 1999 році. IoT інтегрує навколишні об'єкти в єдину мережу. Вони обмінюються інформацією один з одним і працюють без людини в реальному часі (рис. 1.1).

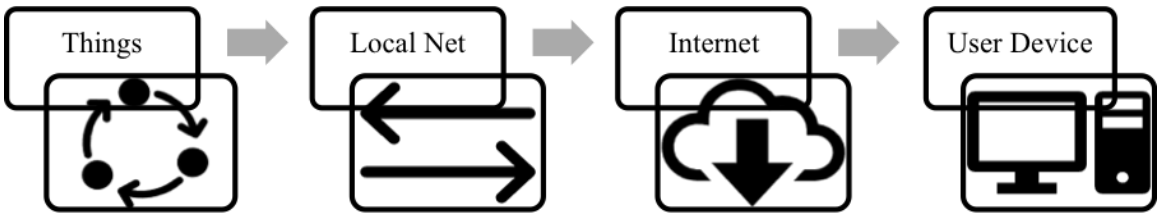


Рисунок 1.1 - IoT (Речі – локальна мережа – Інтернет – Пристрій користувача)

Інтернет речей став реальністю завдяки інтеграції кількох технологій, зокрема бездротового зв'язку, мікроелектронних систем і, звісно, Інтернету. Це переносить світ на новий рівень. Його вплив позначається на всіх сферах життя людини, від процесу їзди автомобіля до способу здійснення покупок і навіть отримання та обліку електроенергії вдома. Розумні датчики та чіпи, вбудовані у фізичні речі навколо нас, щодня постійно обмінюються між собою тисячами гігабайт даних.

Вже давно існує добре розвинена інфраструктура мобільних і фіксованих мереж, а це означає, що широке впровадження IoT набуває реального втілення. Система здатна працювати як на масового споживача, так і на обслуговування бізнесу в цілому. Безсумнівно, тенденція використання Інтернету речей із використанням сенсорів, додатків і платформ з часом буде тільки посилюватися, оскільки фінансові вливання в цю технологію відносно незначні.

Однією з головних труднощів на шляху до розвитку Інтернету речей є відсутність єдиних стандартів. Ця обставина ускладнює інтеграцію бездротових мереж та об'єктів в єдину мережу. Ідеальна технологія, розроблена для поєднання трьох основних функцій:

• енергоефективність;

• стабільність;

• безпека.

Все ще знаходиться в стадії розробки. Крім того, існує ризик кібератак на дані системи IoT, що порушує довіру до інновацій. Тому вдосконалення системи безпеки для всіх пристроїв, які беруть участь у мережі, є одним із головних завдань ринку IoT. Технологія Інтернету речей знайшла своє застосування не тільки в домашньому середовищі, наприклад, розумна побутова техніка та персональні цифрові пристрої, а й у комерційних секторах, сільському

господарстві, охороні здоров'я, нерухомості та безпеці, а також швидко набирає популярності в таких галузях, як логістика.

Нові технології, що впроваджуються, поширюються по всьому ланцюжку створення вартості в логістиці, а саме: складські операції, перевезення вантажів та кінцеві поставки. Також завдяки інноваціям покращується ефективність виробництва, обслуговування клієнтів та безпека. Інтернет речей допомагає вирішувати оперативні проблеми, використовуючи найкращий можливий варіант.

«Фізичний» Інтернет можна використовувати у вигляді безпосередньо підключених пристроїв (датчики, роботи), а також у вигляді Інтернет-провідника між пристроями. Це з'єднання забезпечується бездротовими технологіями для передачі даних, такими як Bluetooth, RFID і Wi-Fi, а також мобільними мережами 3G (4G) і LTE, що об'єднують всі численні пристрої в одну мережу.

Впровадження IoT в логістику дає швидкий та ефективний результат. За допомогою цієї технології можна відстежувати стан активів, пакетів і людей у режимі реального часу по всьому ланцюжку створення вартості. Є можливість автоматизувати бізнес-процеси, щоб виключити ручну працю, підвищити якість і передбачуваність, а також знизити експлуатаційні витрати. Більше того, Інтернет речей дозволяє оптимізувати спільну роботу людей і пристроїв, підключених до комп'ютерної мережі, а також забезпечує моніторинг і рухає процес у правильному напрямку. Нарешті, аналіз можна застосувати до всього ланцюга створення вартості, щоб визначити ширші можливості для вдосконалення та застосування найкращих практик.

Існує думка, що IoT несе потенційний ризик для працівників, оскільки інновації зменшують трудові ресурси. Тим не менш, його слід розглядати як інструмент забезпечення безперебійного виконання операцій та максимізації прибутку. Ця інноваційна технологія гарантує покращення в наступних областях:

• оптимізація застосовуваних засобів;

• зменшення проблем безпеки, таких як підробка та крадіжка;

• точний моніторинг ресурсів і робочого процесу;

• чітка видимість у реальному часі та своєчасне реагування на події;

• аналіз потоку реальних даних для адекватного та швидкого прийняття рішень;

• скорочення ручної обробки даних для підвищення точності та скорочення витрат часу;

• виявлення нових можливостей на основі вивчення моделей поведінки споживачів;

• підвищення якості роботи з клієнтами.

1.6 Аналіз ринкових рішень

Процес глобалізації призводить до того, що ланцюги поставок стають дедалі складнішими і постійно зростають. Відповідно, ця тенденція також впливає на керівництво подібними мережами та індустрію зберігання. Зростає тиск на логістику і все більш важливою складовою у вирішенні проблем транспортних компаній стає Інтернет речей. Сьогодні його метою є задоволення потреб глобальної економіки, що швидко розвивається.

Управління запасами та складуванням є однією з найважливіших частин пов'язаної логістичної екосистеми. Розміщення невеликих недорогих датчиків дозволить компаніям легко відстежувати запаси, контролювати їх стан і розташування, а також створювати інтелектуальну складську систему. Датчики IoT можна використовувати для відстеження запасів і надання даних, які допоможуть визначити тенденції для прогнозування майбутніх потреб у запасах. Це допоможе уникнути ситуацій з недостатнім запасом і надлишковим запасом. Таким чином, впровадження технології IoT дозволить успішно запобігти будь-які втрати, забезпечити безпечне зберігання товарів, а також швидко знайти потрібний товар. Тому людські помилки зведені до мінімуму.

Наприклад, всесвітньо відомий гігант Amazon досяг значних успіхів у автоматизації управління складом. У 2012 Amazon придбала роботів, щоб транспортувати товари з одного пункту складу на інший. Ключовим моментом є те, що роботи створені для роботи в тандемі з працівниками складу. Очевидно, що зусилля компанії спрямовані на автоматизацію складу для створення дешевшого та ефективнішого ланцюга поставок.

Впровадження безпілотників у логістичний ланцюжок стало одним із головних завдань сьогодні для таких гігантських компаній, як Google, Amazon та DHL, які розробили власні БПЛА та вже проводять випробування. Інша велика

торгова компанія Walmart має намір використовувати БПЛА всередині логістичних центрів: дрони будуть переміщатися по складському простору, роблячи до 30 фотографій в секунду, і ця інформація буде використовуватися, наприклад, у разі інвентаризації. Таким чином, впроваджена система Інтернету речей дозволить у кілька разів скоротити час працівників складу на процес інвентаризації, а також знизить ризик людського фактора (наприклад, нестачі товару) під час процедури.

IoT має широке застосування в галузі доставки та логістики. Можливо, логістичні компанії отримують найбільшу вигоду. Ця галузь, включаючи її субдомени «на вимогу» та ланцюг поставок, є першою, яка використовує технологію IoT. Багато аналітиків вважають, що термін «IoT» на багато років молодший за початкові випадки використання IoT в логістиці.

Нижче наведено деякі випадки, коли це нововведення залишає свій значний слід.

З пристроями IoT, підключеними до програми для смартфона, вам не потрібно сканувати одиниці доставки вручну. Замість цього ви можете поставити мітки на блоки, і спеціально створений пристрій буде сканувати їх, коли об'єкт спостережень рухається перед пристроєм.

Ви отримаєте результати цього сканування на свій смартфон. Немає необхідності бути присутнім на місці. Смартфон покаже результати незалежно від вашого поточного місцезнаходження.

Окрім відстеження запасів, ви також можете забезпечити ефективне управління складом. IoT може допомогти вам поповнити магазини, скласти сукупне планування та знайти будь-який товар за мить.

Десятиліття тому контейнери, які перевозили товари за кордоном і всередині країни, вимагали б перевірку платежу кілька разів під час однієї подорожі. Це залишається проблемою навіть зараз для більшості компаній-доставників, які не бажають включати блокчейн.

Спочатку ця технологія була призначена для криптовалют. Однак деякі з лідерів логістичної галузі застосували його у своєму бізнесі, оцінивши його значимість.

Блокчейн різко скорочує час доставки товарів, усуваючи більшість блокувальників перевірки. Він забезпечує прозору систему, де кожна зацікавлена сторона може переглядати ту саму інформацію. Крім того, він забороняє вносити будь-які зміни, щоб запобігти спробам маніпулювання записом.

IoT в ланцюжку поставок може бути не новим, але застосування Blockchain з'явилося зовсім недавно. Тим не менш, результат вражаючий. Крім часу, компанії також можуть суттєво мінімізувати операційні витрати.

Мобілізація вантажів за допомогою вантажівок і кораблів може зайняти дні і тижні, щоб дістатися до місця призначення. Менеджерам важливо забезпечити, щоб температура контейнера залишалася розумною. Крім того, менеджерам необхідно вести спостереження за транспортними засобами в режимі реального часу та підтримувати зв'язок з водіями.

Пристрої Інтернету речей не лише передають ці дані менеджерам у режимі реального часу за допомогою додатків для смартфонів, але й надають прогнозну аналітику. Сюди входять, наприклад, кількість миль, пройдених вантажівкою, і відповідна витрата палива (рисунок 1.2)

Рисунок 1.2 – типові сценарії застосування системи контролю технологій Інтернету Речей в логістиці

1.7 Міні-дрони для доставки

Міні-дрони стають, мабуть, найкращим застосуванням IoT у логістиці. У надто перевантажених міських районах і широко поширених приміських районах наземні транспортні засоби можуть бути дорогими. Більше того, виклики останньої милі роблять його ще складнішим.

Дрони можуть стати дуже ефективною заміною. Все, що вони коштують після покупки, — це акумулятор і додаток на основі IoT, який може забезпечити підключення.

Однак ми ще не побачили нормативної бази, яка б забезпечувала послуги доставки на основі дронів у більшому масштабі.

Датчики на основі IoT продовжують живити автономні транспортні засоби інформацією про навколишнє середовище. Ця здатність забезпечує безперебійний досвід. Транспортні засоби залишаються в русі навіть на інтенсивних розв'язках, оскільки пристрої розраховують точну відстань самокерованого автомобіля від кожного довколишнього об'єкта.

Для цих можливостей аналітики прогнозують, що безпілотні транспортні засоби стануть основним вибором логістичних операторів через кілька років.

Вплив IoT на транспортну та логістичну галузь:

- Прозорість і однозначність.
- Значне зниження витрат.
- Різке покращення доставки на останню милю.
- Сприяння клієнтам, водіям та управлінському персоналу.
- Моніторинг і спостереження, що створює безпечну логістичну роботу.
- Розділення турбот.
- Виконання соціальних обов'язків без додаткових витрат.
- Планування майбутнього за історичними даними.

Підприємства з логістики, доставки та ланцюга поставок можуть заробити значну суму, включивши IoT. Існують робочі тематичні дослідження компаній, які отримують величезну віддачу від інвестицій. Зокрема, вищезгадані варіанти використання визначають майбутнє IoT в логістиці.

Перераховано найбільш перспективні програми технологій Big Data, Machine Learning та IoT у логістиці:

- складська роботизація – від «розумних» навантажувачів до дронів. Наприклад, у Amazon маленькі роботи KIVA самостійно переміщують предмети всередині складу, скорочуючи витрати на 20%. У цій же компанії дроти, що літають, успішно доставляють замовлення віддаленістю до 30 хвилин. Тут же відзначимо ще один приклад складської цифровізації з використанням квадрокоптерів, коли вони наприкінці робочого дня сканують штрих-коди на товарах, автоматично передаючи дані до системи обліку. Це підвищує ефективність процесів інвентаризації на 20%. Зокрема, компанія DroneScan заявляє, що їх дрони лише за 2 дні проведуть інвентаризацію якісніше, ніж 80 осіб за 3 дні.

- оптимізація фінального етапу доставки товару до споживача, так званої «останньої милі». Вартість цього завдання може становити до 28% від загальної ціни доставки. Це відбувається через особливості міської інфраструктури, наприклад, відсутність під'їзних шляхів, ремонт доріг, пробок та інших зовнішніх факторів. Постійний збір та аналітика таких даних дозволяє оперативно перебудувати маршрут і підібрати техніку, що підходить для конкретного замовлення [3] . Вищезгаданий приклад Amazon із дронами показує, що їх використання знижує вартість останньої милі до $1 при доставці малогабаритних вантажів (менше 2,25 кг).
- трекінг вантажів за допомогою RFID- міток, які дозволяють стежити за переміщенням товару протягом усього ланцюжка поставок. Такий безперервний моніторинг скорочує збитки через порушення умов зберігання та транспортування продукції, що швидко псується, або товарів з особливостями перевезення. Економія може становити до 30%. Наприклад, транспортна компанія DHL встановлює на вантажі IoT-датчики Smart Sensor. А інший великий перевізник, Cerasis впроваджує Big Data рішення для оптимізації маршрутів, скорочення витрати пального та зниження негативного впливу на навколишнє середовище. Також компанія планує використовувати IoT-датчики для безперервного моніторингу стану своїх машин, щоб знижувати витрати на ремонт та зменшувати час простою.
- обов'язкове маркування продукції за допомогою вже згаданих RFID-міток або DataMatrix-кодів. Нагадаємо, в Росії з січня 2019 року введено обов'язкове цифрове маркування низки товарів: ліки, тютюнова продукція, парфуми та туалетна вода, шини та покришки, взуття, деякі види одягу та текстилю, фотографічне обладнання та молочна продукція. Таке маркування робить унікальною кожну одиницю продукції, тому ритейлерам потрібно змінювати процедури оптового відвантаження та приймання товарів, що раніше устоялися. Зокрема, агрегувати коди в палетних етикетках та вбудовувати ці рішення у вже існуючу IT-інфраструктуру, а також організуючи їхню інтеграцію з наглядовими органами.
- скорочення операцій, що не додають цінності за методологією Lean . Наприклад, трудомісткості комплектувальника замовлень на читання паперових листів підбору або інструкцій з екрана планшета. Щоб усунути це, розподільний центр мережі супермаркетів "Вірний" впровадив систему голосового відбору Vocollect. Вона

дозволила збільшити продуктивність на 35% і довести до 99,97% точність операцій із збору замовлень, радикально скоротивши помилки у комплектації через людський фактор. Тепер вся взаємодія із системою керування складом йде через голос: комплектувальник просто слухає команди та виконує завдання, ставлячи запитання або повідомляючи про готовність. Це відмінний приклад практичного використання алгоритмів Machine Learning для розпізнавання мови.

1.8 Датчики в проектному рішенні

Далі наведено датчики IoT, що було використано в процесі проектування дипломного рішення мережної інфраструктури для транспортної компанії, виходячи з аналізу ринку та бази проведення практики в компанії ТОВ «ДІСТІ ПРО».

Teltonika FMT100 – спеціальний маленький та водонепроникний GNSS трекер з підтримкою Bluetooth та внутрішньою резервною батареєю. Найбільша у своєму класі внутрішня антена GNSS з високим коефіцієнтом посилення дозволяє встановлювати трекер прямо на автомобільному акумуляторі під капотом. FMT100 оснащений спеціальним затискним роз'ємом для швидкого підключення кабелю живлення до автомобільного акумулятора. Міцний водонепроникний корпус, вбудований акселерометр і датчики гіроскопа з надзвичайно точною функцією відстеження зіткнень роблять цей пристрій ідеальним для рішень страхового ринку. (рис. 1.3)

Рисунок 1.3 - FMT100

Також нижче наведена принципова схема в якій описано загальний зміст датчика(рис. 1.4)

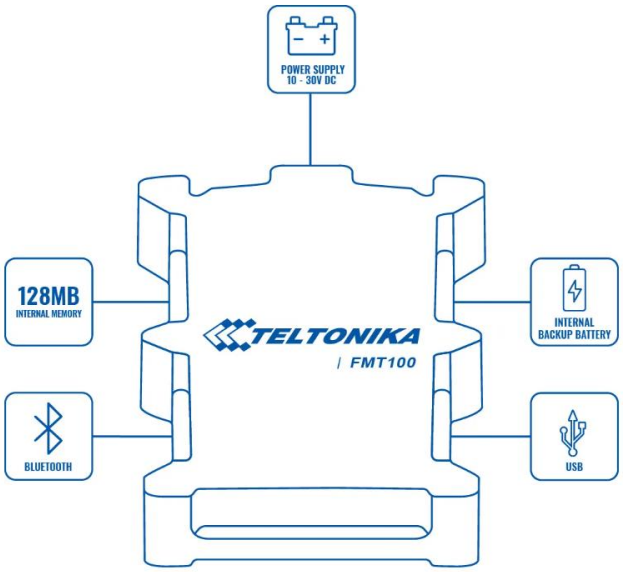


Рисунок 1.4 – схематичний склад сенсора

Teltonika FMM130 - ПЕРЕДОВИЙ трекер з внутрішніми антенами GNSS та LTECAT-M1/NB-IoT/GSM та вбудованою батареєю. Пристрій оснащений GNSS / Bluetooth і LTE CAT-M1/NB-IoT модулями з резервним підключенням до мережі 2G, внутрішніми антенами GNSS і LTE, цифровими, аналоговими входами і цифровими виходами, що налаштовуються, негативним входом, імпульсними входами. Це ідеально підходить для додатків, де необхідне отримання даних про віддалені об'єкти: управління автопарком, компанії з прокату автомобілів, компанії таксі, громадський транспорт, логістичні компанії, особисті автомобілі та інше.(рис. 1.5)

Рисунок 1.5 – Teltonika FMM130

Потужний маршрутизатор Dual LTE Cat6 для важливих додатків. Оснащений двома модемами LTE для двох одночасних з'єднань, що забезпечує миттєве плавне перемикання сервісів LTE та балансування навантаження. 5 портів Gigabit Ethernet, DualBand 802.11ac WiFi, Bluetooth LE та USB-інтерфейси з програмним забезпеченням RutOS та функціями безпеки роблять цей пристрій незамінним в умовах, коли втрата з'єднання неприпустима (рис. 1.6)

Рисунок 1.6 – загальний вигляд роутера

Висновки за розділом 1.

За результатами розділу 1 було проаналізовано, що є поняття «інформація», які види інформації існують, як вона впливає на ланцюги постачання на логістичну складову транспортних компаній, як можна покращити логістику за допомогою дронів та які технології щоденно допомагають компаніям аналізувати дорожній трафік й планувати свою безпосередню діяльність. Проведено аналіз рішень компаній-лідерів в транспортному сегменті. Виявлено ефективність та перспективність застосування новітніх технологій в логістичній складовій, оцінено потенцільну ефективність імплементації дронів для повітряної доставки в транспортній компанії. В наступному розділі наведено топологічні й проектні рішення для побудови мережної інфраструктури з використанням технологій IoT, основні переваги та недоліки вибору постачальника рішень, та саме які моделі є найвігиднішими з аналізу ринку, та за результатами проходження науково-дослідної практики.

РОЗДІЛ 2. РОЗРОБКА ПРОЕКТУ ІОТ-РІШЕННЯ ТА МЕТОДІВ ОБРОБКИ ДАНИХ В ІОТ

2.1 Початок проектування дипломного рішення

Зміни бажання клієнтів, брак часу, нестабільні ринкові ціни, управління очікуваннями клієнтів, неправильна обробка вантажів, оптимізація логістичних маршрутів - ось найбільші завдання, з якими щодня стикаються компанії, що займаються доставкою та кур'єрськими послугами. Більше того, згідно з звітом BSI та TT Club Cargo , викрадення вантажів у дорозі було найвищим серед усіх крадіжок вантажів у 2020 році – 71%. Втрати зі складів та інших складських приміщень збільшились до 25%.

Щоб задовольняти ринковий попит, залишатися на плаву, своєчасно надавати клієнтам актуальну інформацію та оптимізувати повсякденні витрати, сучасний логістичний бізнес вимагає не тільки відстеження та управління автопарком, а й товарами, що доставляють: посилками, пакетами, коробками, палетами, продуктами тваринництва, харчування, і т.д.

Саме тому відстеження наближення для визначення випадків втрати та знаходження чи моніторингу навантаження та розвантаження товарів стає важливою процедурою у сфері доставки вантажів. Звичайно, дуже важливою залишається контроль маршруту, відстеження часу доставки та фіксування останнього відомого місцезнаходження вантажу. Це дозволяє оптимізувати процес доставки та значно знизити операційні витрати.

Крім цього, менеджери автопарків повинні ефективно управляти умовами транспортування вантажу у процесі відправлення, щоб забезпечити своєчасну доставку до потрібного пункту призначення. Особливо це стосується таких критично важливих параметрів як температура вантажу, вологість і навіть виявлення ударів. Всі ці дані та події (якщо умови транспортування порушені, посилка втрачена або доставлена не за адресою) повинні відстежуватися, реєструватися та одразу ж відправлятися у програмний додаток для відстеження автопарку.

Тут є серйозне технічне завдання – сигнали GPS часто недостатньо точні, щоб бути практичними для використання у приміщеннях або на вузьких вулицях, оскільки вони зменшуються та розсіюються навколишніми структурами – дахами та стінами. Більше того, діапазон помилок визначення місця

розташування деяких GPS-чіпів може бути більшим, ніж сам простір усередині приміщення. Хорошою новиною є те, що ці проблеми можуть бути ефективно вирішені за допомогою бездротової технології Bluetooth, нових маячків Teltonika EYE Beacon та/або датчиків EYE Sensor , а також автомобільних GPS-трекерів .

Всі GPS-трекери марки Teltonika на базі платформи FMB підтримують бездротове з'єднання Bluetooth 4.X LE, тому вони можуть ефективно взаємодіяти з Bluetooth-пристроями, такими як маячки та датчики. Для цього всього лише необхідно, щоб трекер був встановлений в автомобілі і налаштований стандартно в тому місці, де антена Bluetooth не закрита великими металевими деталями поблизу.

Якщо потрібно відстежувати лише місце розташування посилки , слід використовувати зв'язку: автомобільний трекер і EYE Beacon. У випадках, коли необхідно отримувати дані про місцезнаходження та додаткові дані про вантаж , такі як: температура, вологість, виявлення магнітів, переміщення, випадкове падіння, удар, падіння на асфальт або підлогу, рекомендується використовувати комплекти що складаються з: трекера транспортних засобів і датчиків EYE Sensor.

Як це працює: наприклад, можна використати модель FMB140 автомобільного трекера марки Teltonika. Маленький та легкий EYE Beacon або EYE Sensor повинен бути прикріплений до того предмета доставки, який необхідно відстежувати та контролювати: посилку, пакет, коробку, палет тощо. Оскільки кожен EYE Beacon і EYE Sensor має унікальний ідентифікаційний номер (ID), що передається по ефіру з інтервалами, що налаштовуються, GPS-трекери зчитують, ідентифікують і відправляють ці дані у вигляді профілю iBeacon або Eddystone в поєднанні з даними про місцезнаходження за GNSS на сервер .

Спеціальне програмне забезпечення, розроблене постачальником телематичних послуг, визначає та показує розташування всіх Beacons та/або Sensors (тобто маркованих товарів) на основі близькості до найближчого трекера та допомагає керувати процедурою призначення BLE-датчиків. За допомогою зручного доступу через ПК, ноутбук або смартфон можна отримати доступ до моніторингу товарів, що значно підвищує ефективність бізнесу.

Для забезпечення максимальної вигоди ці пристрої мають дві зручні функції, на які слід звернути увагу - “Proximity Event” (Подія наближення) та “Виявлення за фільтрами” (Виявлення за фільтрами). Давайте розглянемо спеціальні функції вбудованого програмного забезпечення, які у поєднанні з новими продуктами Teltonika EYE Sensors та EYE Beacon роблять їх ідеальним вибором для двох важливих випадків використання у сфері доставки товарів.

"Proximity Event" (Подія наближення). Зручна програма для налаштування

GPS-трекера від Teltonika дозволяє вибирати різні параметри та сценарії для задоволення потреб компанії. Завдяки цьому автомобільний трекер може генерувати події, пов'язані з місцезнаходженням, події про втрачені та знайдені BLE датчики в залежності від рівня сигналу Bluetooth.

FMB140 може генерувати події, пов'язані з близькістю до об'єкта, залежно від рівня сигналу Bluetooth, який отримується від маячків. Це дозволяє своєчасно і точно визначити місце розташування кожного рухомого маяка або датчика (тобто об'єкта відстеження, що цікавить нас), згрупованих у зони "Близько", "Далеко" та "Дуже Далеко".

В результаті можуть бути сформовані списки предметів доставки, що відстежуються, в кожній зоні або ініційовані повідомлення про конкретні події. Наприклад, якщо товар залишив всі відстежувані зони (подія "Втрата"), GPS-пристрій зареєструє цей факт, визначить та запише координати останнього відомого місця та точний час події. Усе це робиться задоволення потреб бізнесу настільки ефективно, наскільки це технічно можливо.

"Detection By Filter" (Виявлення фільтру). Ця функція користувальницької фільтрації дозволяє групувати та називати відстежувані об'єкти за певною ознакою або властивістю, важливою для бізнес-операцій (наприклад, товари для внутрішньої доставки або експорту; товари вищого чи низького пріоритету; небезпечні товари, пошкоджені товари тощо). Ця опція допомагає контролювати та керувати предметами або опціями сортування, складування, процедур навантаження/розвантаження та більш ефективно оптимізувати використання корпоративного автопарку, уникаючи таким чином дорогих помилок, заощаджуючи час та ресурси компанії, що призводить до максимальної ефективності.

Підсумовуючи, можна звернути увагу на той факт, що зв'язок Bluetooth має низьку вартість, високу енергоефективність і точність, працює незалежно від мережі та має менше перешкод, а так само проста в установці та розгортанні. Маяки та датчики Teltonika з рівнем сигналу та інтервалами передачі даних, що настроюються відповідно до вимог замовника, можуть бути легко інтегровані в середу з легкою можливістю масштабування. Все це в сукупності призведе до помітної різноманітності та прибутковості проектів, поліпшення ділової репутації, конкурентоспроможності та повернення інвестицій.

2.2 Система контролю логістичної складової в поточному рішенні IoT

Ринок відстеження транспортних засобів впевнено та швидко зростає по всьому світу, так само, як і суперництво серед постачальників телематичних послуг та інтеграторів. Переважаючий базовий сценарій відстеження транспортних засобів може бути недостатньо хорошим для ефективної конкуренції та підтримки бізнесу в довгостроковій перспективі. Щоб вирішити цю проблему та допомогти таким компаніям, Teltonika Telematics готова запропонувати економічно ефективне рішення.

Це пояснює (і підтверджує) популярність простих автомобільних GPS-трекерів із базовим сценарієм відстеження. Ці пристрої мають необхідний набір функцій, прості в установці та налаштуванні, компактні, доступні за ціною і повинні ідеально підходити для задоволення потреб сегмента ринку легкових автомобілів (корпоративних або приватних), а також невеликих комерційних автомобілів. Звучить добре, але є проблема, з якою потрібно впоратися – справа не тільки в зростаючій конкуренції серед постачальників телематичних послуг та інтеграторів, дефіциті електронних компонентів, але й у очікуваннях водіїв та/або менеджерів автопарків, що зростають, у швидко мінливих умовах. Вони очікують на більшу вигоду без переплати.

Тому базового відстеження GPS-координат транспортних засобів (особливо корпоративних) та використання сценарію Поїздки вже недостатньо. В даний час вирішальними факторами при виборі моделі GPS-трекер є додаткові корисні функції, які інтегратори зможуть запропонувати кінцевим користувачам і клієнтам, не витративши при цьому цілий стан. В ідеалі необхідно також удосконалити вже встановлені автомобільні трекери, а також мати можливість швидкого та простого монтажу та оновлення. Загалом це привабливе поєднання допоможе успішно конкурувати на ринку, отримати більше проектів і розширити бізнес або, принаймні, забезпечити його стійкість.

Для демонстрації цього рішення було обрано модель GPS-трекера Teltonika FMT100 категорії СПЕЦІАЛЬНІ, розроблену для страхової телематичної галузі. Докладніше про його застосування та переваги Ви можете дізнатися тут . Крім того, ми використовуємо донгл для бортової діагностики - пристрій, який підключається безпосередньо до порту OBD-II (він же OBD2) автомобіля і з'єднується з бездротовим трекером Bluetooth.

Зверніть увагу, що FMT100 – це пристрій, який не відноситься до типу OBD. Однак доступ до даних OBD-II і діагностичних кодів несправностей (DTC) у поєднанні з власним набором функцій дає керівництву автопарку значні додаткові переваги - діагностичні параметри автомобіля, що цікавлять, і моніторинг їх роботи в режимі реального часу, можливість проведення

своєчасного технічного обслуговування, що дозволяє уникнути дорогого капітального. ремонту та простоїв. Крім того, можуть бути отримані індивідуальні звіти на запит, що також тягне за собою поліпшення поведінки та дисципліни водіїв, зниження витрат на страхування та ведення бізнесу тощо.

Принцип роботи - OBD-II донгл зчитує відповідні параметри та/або коди автомобіля та відправляє їх на трекер FMT100 через Bluetooth, використовуючи спеціальний протокол передачі даних. Після цього FM-пристрій надсилає ці дані разом з даними про місцезнаходження GNSS і, в даному випадку, інформацією про відстеження, пов'язаною зі страховою телематикою, по мережі GSM на сервер для подальшого аналізу та складання звітів.

Таким чином, менеджери автопарку можуть контролювати відразу два потоки даних – один від автомобільного трекера, а інший від порту OBD-II. Зручно, практично та просто. Зверніть увагу, що модулі Teltonika FMB працюють лише з донглами OBD-II на базі мікроконтролерів ELM327 або STN1110.

Що відстежувати - найпоширенішими та найцікавішими параметрами для моніторингу є швидкість автомобіля, оберти двигуна, температура моторного масла, рівень палива, витрата палива, температура охолоджуючої рідини, помилки EGR, коди несправностей тощо. Загалом усе залежить від специфіки конкретного автопарку. Список налаштувань для відстеження може бути змінений у будь-який час за Вашим бажанням.

Як настроїти – спочатку необхідно підключити Bluetooth OBD-II донгл до OBD-порту автомобіля. Це займе всього хвилину, не вимагатиме жодних спеціальних інструментів чи приміщень. По-друге, необхідно налаштувати Bluetooth моделі FMT100 за допомогою конфігуратора Teltonika , як показано нижче. Для цього перейдіть до розділу меню “Bluetooth” та увімкніть відповідні функції. (рис. 2.1)

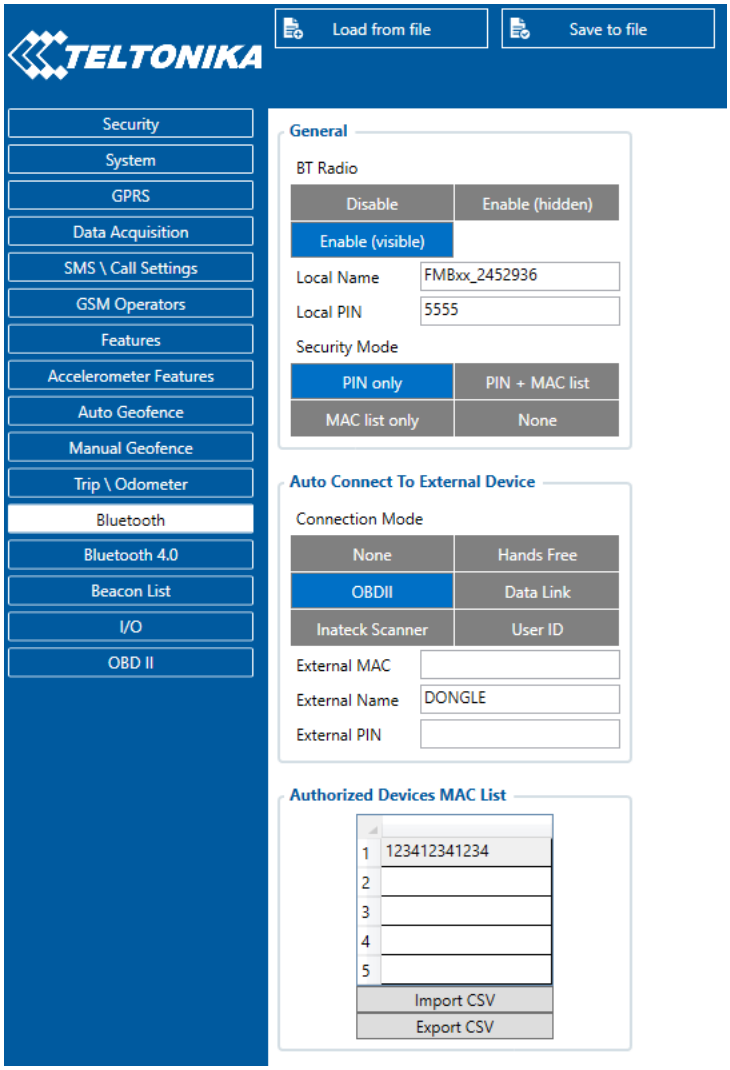


Рисунок 2.1 – інтерфейс логістичної системи IoT Bluetooth від Teltonika

Нарешті, в розділі меню 'OBD II' можна обрати з представленого списку параметри, що цікавлять Вас, вказати пріоритет для кожного з них і зберегти конфігурацію, натиснувши кнопку 'Зберегти в пристрій'. Точний список стандартних параметрів та кодів, доступних для відстеження, залежить від виробника, моделі та року випуску автомобіля. (рис. 2.2)

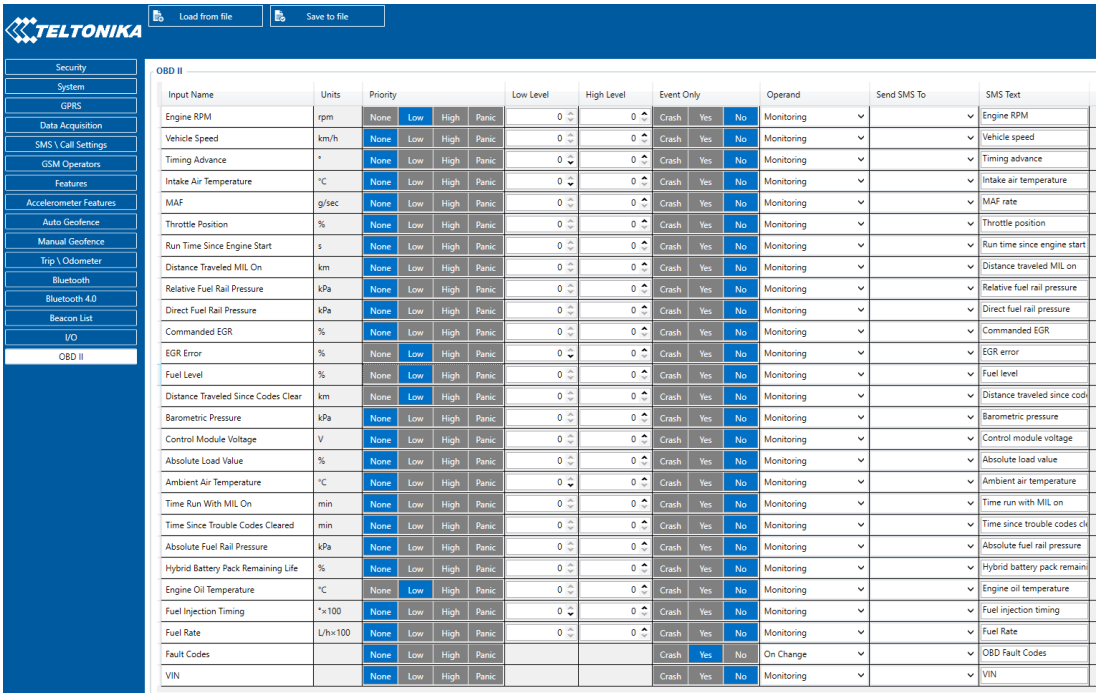


Рисунок 2.2 – Таблиця контролю за датчиками в межах автопарку

Процес встановлення донгла OBD-II виконується швидко, без проблем та займає буквально кілька секунд. Найприємніше – ці пристрої широко доступні по всьому світу та доступні для будь-якого бюджету автопарку.

Підсумовуючи, можна сказати, що це рішення дозволяє підприємствам отримати краще від обох пристроїв - вже готовий до роботи набір функцій GPS-трекера Teltonika, а також численні переваги даних OBD-II і DTC, що одночасно допомагають значно покращити відстеження, моніторинг та керування автопарком. Оновлення прошивки та зміна конфігурації автомобільних трекерів Teltonika можна проводити за допомогою нещодавно оновленого інструменту FOTA WEB. Це потужне програмне рішення, яке допомагає швидко та ефективно керувати GPS-пристроями.

2.3 Датчик FMT100 з функцією контролю основних показників в кабіні водія

У цій роботі буде запропоновано кілька оптимальних мережевих топологій для логістичної складової, одна з них наведена нижче (рис. 2.3)

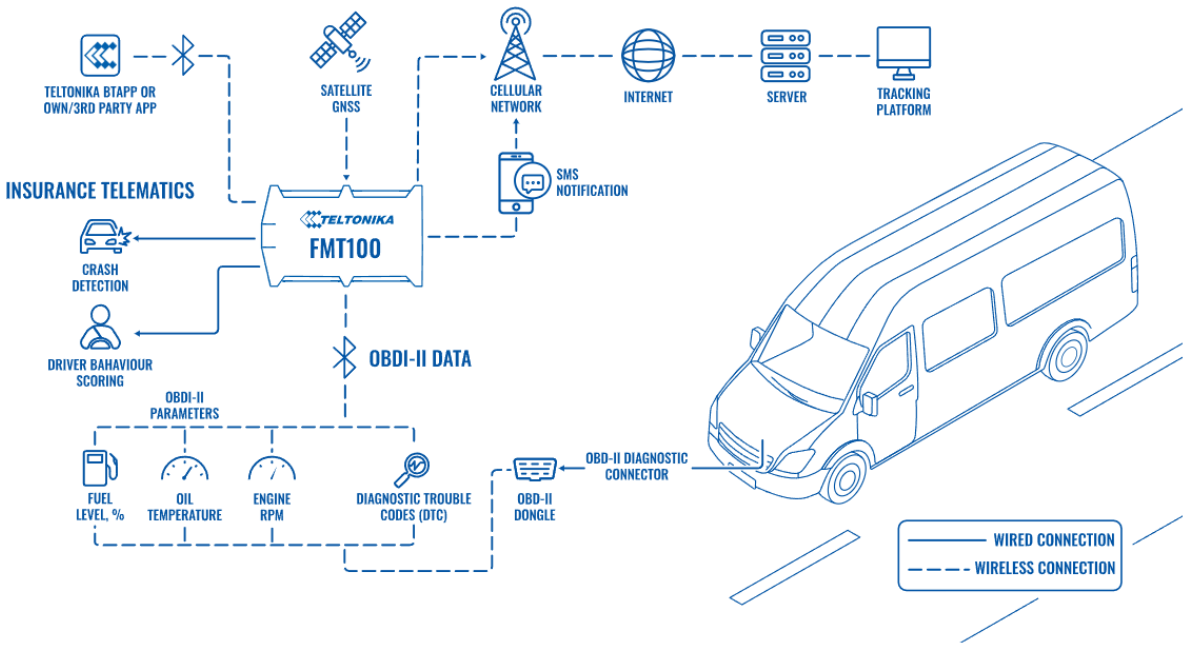


Рисунок 2.3 – актуальна топологія з використанням датчика Teltonika FMT100

Переваги системи:

1. Отримуйте більше, платячи менше – тепер менеджери автопарків можуть відслідковувати та контролювати два потоки даних – від GPS-трекера автомобіля та порту OBD-II – одночасно, не витрачаючи на цей цілий стан.
2. Швидке і просте встановлення - Bluetooth OBD-II донгл можна підключити буквально за кілька секунд. Жодних дротів, ніякого паяння, ніяких спеціальних інструментів, ніякого безладдя. Рішення застосовується до будь-якого автомобільного трекера Teltonika не OBD-типу і підтримує з'єднання Bluetooth.
3. Настроюваний набір параметрів OBD-II для кожного проекту - щоб отримати максимальну вигоду, виберіть у конфігураторі Teltonika тільки ті дані, які відносяться до Вашого проекту або бізнесу, а решту пропустіть. Платіть телекомунікаційним компаніям тільки за дані, які Вам дійсно потрібні і які Ви дійсно використовуєте.
4. Підвищення рентабельності та конкурентоспроможності - використовуючи додаткові дані OBD-II і DTC, підприємства зможуть домогтися помітної економії коштів, покращити звички та дисципліну

водіїв, знизити ризиковану поведінку водіїв, аварії, ремонт, технічне обслуговування, страхування та експлуатаційні витрати, що призведе до підвищення інвестицій. , Збільшення грошового потоку, прибутку та конкурентоспроможності.

2.4 Система відеоконтролю та контролю безпеки водія авто на базі датчику RUT950

Наразі постачальники інтернет-послуг перевантажені трафіком, оскільки внаслідок карантину значний обсяг населення працює віддалено та споживає значну частину інтернет трафіку. Як показує практика, раціональним рішенням для закладів охорони здоров'я або IT-компаній, які надають їм послуги, є оновлення існуючої мережної інфраструктури за допомогою стільникових маршрутизаторів Перевезення цінних речей, таких як готівка, коштовності, документи та подібні предмети, є важливою місією, яка вимагає спеціалізованого обладнання, навченого персоналу та запобіжних заходів безпеки. На щастя, нові технології забезпечують набагато вищий рівень захисту як для співробітників, так і для активів, які доставляють. Рішення IoT дозволяють контролювати парк броньованих вантажівок в режимі реального часу, щоб штаб міг знати, що відбувається з кожним з них у будь-який момент і в будь-який момент.

Дистанційне керування зображенням значно підвищує безпеку бронеавтомобілів з критичною місією доставити всі цінні речі до встановленого пункту призначення. Однак рішення вимагає певного рівня автоматизації, оскільки камери та відео реєстратор мають почати працювати до того, як автомобіль залишить паркувальний майданчик. Таким чином, пристрій підключення повинен спочатку мати можливість інтегрувати апаратне забезпечення та запропонувати достатню гнучкість мікропрограми для налаштування правил автоматизації. Крім того, наявність стабільного надійного з’єднання є вирішальним для віддаленого моніторингу в реальному часі в таких місіях. Навіть хвилини втрати доступу можуть змінити гру у сенсі безпеки персоналу та вантажу. І безпека мережі настільки ж важлива, оскільки порушене з’єднання також може призвести до реальної небезпеки.

У цьому рішенні кілька камер розміщені зовні та всередині автомобіля. За допомогою дротового з’єднання вони підключаються до DVR (цифрового відеореєстратора), який, у свою чергу, підключається до стільникового маршрутизатора RUT950, забезпечуючи підключення для всіх цих пристроїв. Маршрутизатор з подвійною SIM-картою 4G забезпечує стабільний та надійний доступ до Інтернету в транспортному засобі, що рухається, з автоматичним перемиканням збоїв, забезпечуючи безперервність мережі навіть

у разі втрати основного з’єднання. Наявність резервного підключення до іншого оператора є важливою для безпеки цього рішення, оскільки будь-яка хвилина втраченого зв’язку створює підвищений рівень загрози.

Однією з основних функцій, доступних в RUT950 для цього рішення, була функція планування, яка автоматично активує джерело живлення DVR за допомогою цифрового виходу. Це дозволяє вмикати камери до того, як персонал прибуде в зону, і забезпечує активність системи камер без будь-якої взаємодії з людьми. В результаті центр управління в штаб-квартирі може мати видимість транспортного засобу та його оточення.

Штаб-квартира підключається до маршрутизатора за допомогою безпечного VPN-з'єднання. RutOS пропонує різноманітні послуги VPN на вибір, щоб відповідати вимогам різних рішень. Зашифроване з’єднання захищає дані від перегляду або зміни третіми сторонами зі зловмисними цілями, тому відеозаписи з камери в прямому ефірі та записи можуть безпечно дістатися до віддаленого центру моніторингу та серверів(рис. 2.4).

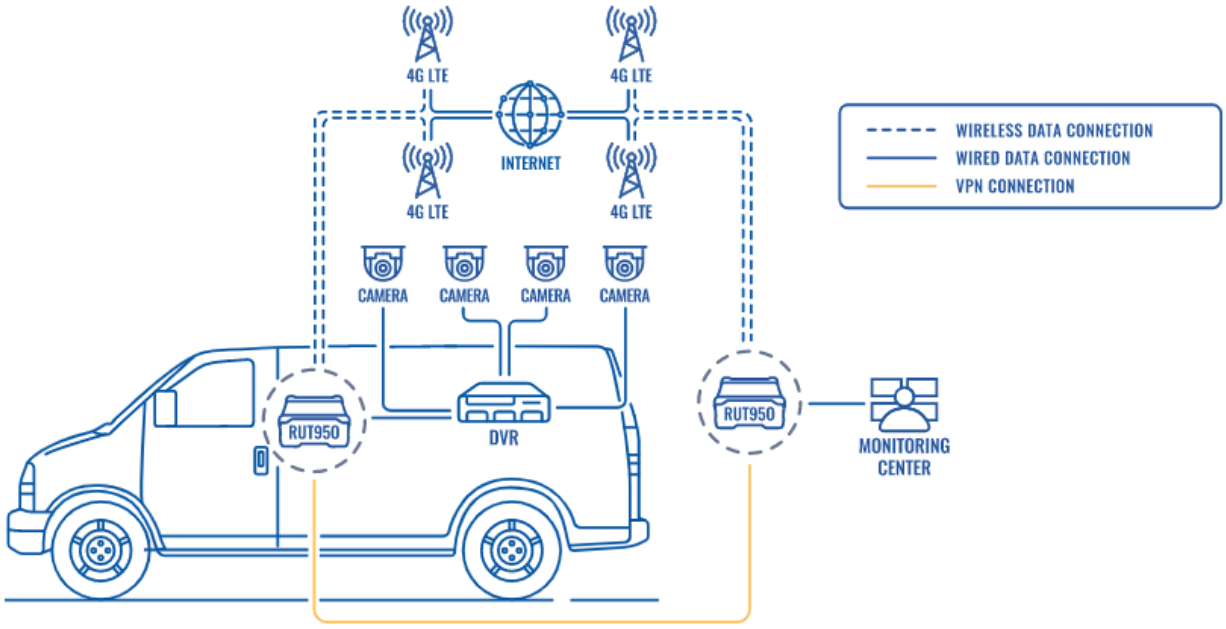


Рисунок 2.4 – топологія актуального рішення

Переваги:

1. RUT950 забезпечує 4G підключення з автоматичним перемиканням між двома SIM-картами, що ідеально підходить для забезпечення зв'язку з резервування каналу зв'язку в транспортному засобі, що рухається.
2. Міцна та довговічна конструкція RUT950 витримує вібрацію вантажівки та широкий діапазон температур.
3. Функція планування, доступна RUT950, дозволяє включити рішення без будь-якої взаємодії з людиною до прибуття персоналу на

паркування.

4. Використання одного з множини доступних VPN-сервісів у RutOS забезпечує зашифрований, безпечний віддалений доступ зі штаб-квартири до відеореєстратора.

2.5 Мережна система для розгортання дрона в умовах міської локації

Більшість промислових IoT-проектів знаходиться за межами міської інфраструктури. Ще один значний сектор — це тимчасові або спливаючі рішення, не прив'язані до одного місця. Деякі приклади таких проектів можуть бути будівельні майданчики, тимчасові місця проведення заходів, тимчасова роздрібна торгівля або служби екстреного реагування та служби екстреної допомоги.

Звичайно, такі проекти не можуть покладатися на провідне з'єднання та вимагають альтернативних варіантів підключення. Крім того, більшість з них навіть не мають доступу до джерела живлення. Таким чином, їм потрібне рішення, яке вирішує обидві ці проблеми.

Для розгортання дрону також було обрано рішення компанії Teltonika Networks:

Стаціонарна база – не найкращий вибір при розгортанні дрона в певних ситуаціях. Наприклад, рятувальники повинні обстежити територію після стихійного лиха, або військова частина повинна перевірити активність, коли вони входять у ворожу зону. Для швидкого реагування в польових операціях їм потрібно мобілізувати застарілі та застарілі системи, які все ще використовують центри управління. Нові технології мають бути мобільними та компактними.

Не кажучи вже про те, що має бути спосіб забезпечити стабільне підключення до Інтернету 4G у віддалених місцях. У складних умовах якість з’єднання може погіршитися або перерватися. Тому рішення потребує безпечних механізмів. Налаштування також потрібно ретельно контролювати, щоб усі системи функціонували правильно, щоб запобігти можливим збоям під час розгортання.

TALOS — це мобільне рішення IoT для роботи з дронами в польових умовах, розроблене PROBOTEK. Він містить повне налаштування контролю за допомогою вбудованої клавіатури, миші та екрана. Установка працює і як мобільна база, і як GCS (наземна станція управління) і може точно керувати одним дроном або цілим роєм. TALOS оснащений новітніми технологіями і може використовуватися в комерційних, військових сценаріях і рішеннях швидкого реагування(рисунок 2.5).

Промисловий стільниковий маршрутизатор Teltonika Networks RUTX12 забезпечує підключення за допомогою технології Dual LTE CAT6 4G в цьому рішенні IoD. Маршрутизатор має чудове резервне з'єднання з двома SIM-картами 4G в режимі реального часу, якщо основне з'єднання вимкнеться. RUTX12 має функцію балансування навантаження, яка гарантує, що один інтерфейс не буде перевантажуватися, а трафік буде спрямований відповідно.

Оскільки дронам потрібні точні координати землі в повітрі, RUTX12 забезпечує відстеження через GNSS (глобальна навігаційна супутникова система) для точного контролю. Проста конструкція RUTX12 дозволяє легко інтегруватися в рішення, яке також є досить міцним, коли справа доходить до роботи в польових умовах і транспортування.

Крім того, віддалений доступ RMS дозволяє постійно контролювати все рішення. Інтерфейс RMS дає змогу збирати дані в режимі реального часу про всі підключені пристрої з будь-якої відстані, що є важливою функцією в дуже мобільній установці з пристроями, що швидко рухаються.
Переваги:

- Готове рішення, просте у використанні та не потребує спеціальних технічних знань.
- Забезпечує як підключення, так і варіанти живлення, тому все, що потрібно зробити користувачеві, - це підключити кінцеві пристрої і рішення готове до роботи.
- Функція балансування навантаження, доступна в RUTX12, дозволяє використовувати кілька джерел WAN для збільшення пропускної спроможності даних та мінімізації часу простою.
- Безліч інтерфейсів для підключення різного обладнання: 5 портів Gigabit Ethernet, дводіапазонний Wi-Fi Wave-2 802.11ac, Bluetooth LE та USB.
- Може витримувати суворі умови, включаючи спеку, холод, вологість та пил.

Нижче наведено топологію рішення з підключення дрона(рис. 2.5)

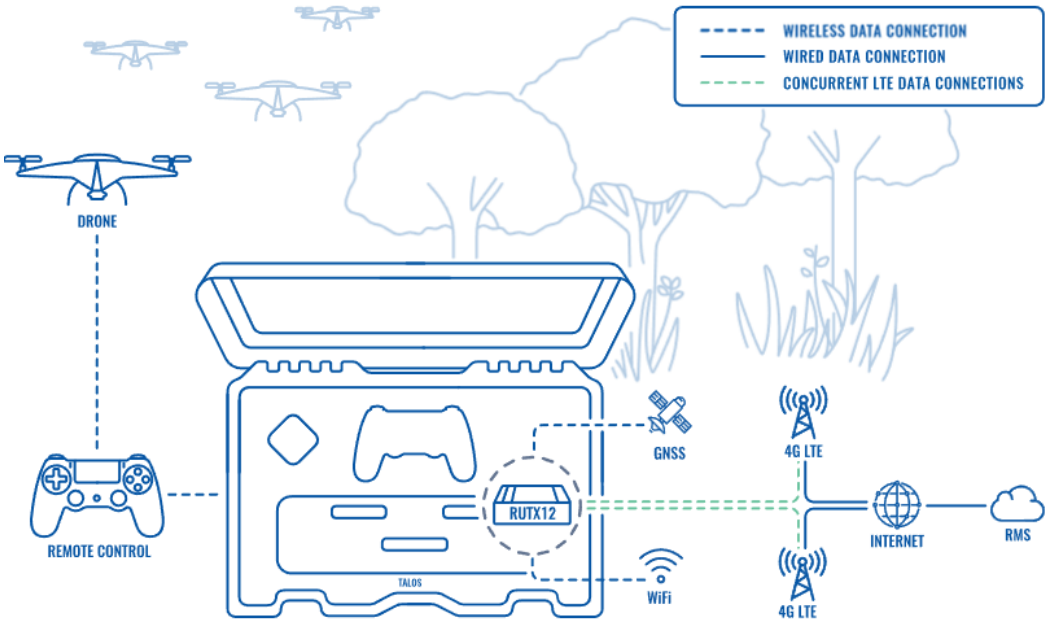


Рисунок 2.5 – топологія системи розгортання дрона за використанням системи TALOS від Teltonika

Teltonika Networks RUTX12 має балансування навантаження, оберігаючи ефективний розподіл даних.

RUTX12 пропонує чудове підключення 4G завдяки технології Dual SIM і LTE Cat 6, що дозволяє мати стабільний інтернет в будь-якому місці.
RUTX12 може бути інтегрований у будь-якій рішучості, щоб його прямий дизайн, практичний для військової та індустрії використання.
TX12 пропонує глобальне відстеження позиціонування за допомогою GNSS, дозволяючи одному легкого переміщення контролю керуванням та підключенням.

Висновки за розділом 2.

На базі практики було розглянуто рішення компанії Teltonika Networks, що на протязі науково-дослідної практики проявили себе как ефективні та надійні рішення для побудови критичної мережної інфраструктури особливо під час воєнного стану що може замінити технологію СтарЛінк від Ілона Маска використовуючи не супутниковий звя'зок а технологію стільникового зв'язку з вищою швидкістю аніж

РОЗДІЛ 3. ОГЛЯД ТЕХНОЛОГІЙ НАПИСАННЯ ТА ЗАПУСКУ ДРОНІВ НА МОВІ ПРОГРАМУВАННЯ PYTHON

3.1 Огляд технологій

Оскільки індустрія дронів еволюціонує від простих RC-роботів у небі до реальних галузей, одне можна сказати напевно: програмування дронів матиме вирішальне значення для реалізації справді автономних дронів.

Вивчити основи можна з програмного забезпечення для дронів, а потім перейдемо до фактичного програмування дронів на Python.

В розділі 3 наведено такі технології:

- Програмне забезпечення з відкритим вихідним кодом, яке структурує безпілотник (політ ardupilot).
- Програмування дрона на Python.
- Керування імітованим дроном за допомогою dronekit python.

Дрони бувають різні. По конструкції розрізняють кілька типів дронів:

- мультироторний - коптери;
- літаковий - fixed wing;
- гібридний – він злітає вертикально, потім використовує крила.

Також дрони поділяються на споживчі (consumer) та комерційні (commercial).

Споживчі, це ті, які можна купити в магазині і використовувати як літаючу камеру. DJI Phantom та Mavic – хороші моделі. Є й менш дорогі, але вони безперечно поступаються за якістю. Такі квадрокоптери використовують для зйомки різноманітних заходів, будівель, історичних об'єктів. Наприклад, з безпілотника можна зробити серію фотографій будівлі або пам'ятника, а потім створити з них 3D модель об'єкта методом фотограмметрії.

Як правило, такі дрони літають на ручному управлінні, рідше за місією в автономному режимі GPS-координатів. Ринок споживчих квадрокоптерів більше ніж наполовину належить одній компанії - DJI. З нею дуже важко конкурувати, оскільки там роблять реально якісний продукт: доступний, функціональний, зручний. Хоча в області квадрокоптерів для селфі DJI починає тіснити компанія Skydio зі своїм дроном R2. Особливість цього дрону в тому,

що він може літати автономно, наприклад, за мотоциклістом у лісі. При цьому безпілотник бачить усі перешкоди та прокладає автономний безпечний маршрут у реальному часі так, щоб людина завжди залишалася у кадрі.

Комерційні дрони використовуються в компаніях для вирішення певного завдання. Дрони стежать за станом сільгоспполів, літаючи над ними регулярно та роблячи фотографії, інші дрони вміють розпорошувати добрива точково. Їх використовують на будовах, у кар'єрах. Щодня вони облітають будівельний об'єкт, роблять фотографії, за якими створюється 3D-модель у хмарі, і вона допомагає відслідковувати щоденні зміни.

Приклад російської компанії, яка активно працює із цією технологією на ринку США, — Traceair.

Ну і звичайно, у всіх на слуху доставка товарів дроном. Невідомо, чи запрацює колись сервіс Amazon Prime Air, але вже зараз компанія Matternet доставляє товари в Цюріху та деяких містах США, а компанія Zipline давно відправляє медикаменти в політ над просторами Африки. У Росії успіхів тут поки що набагато менше, нещодавно була новина про дрон Пошти Росії, який розбився на першому тесті, а Ощадбанк тестує доставку грошей дронами.

Компанії Volocopter і Ehang вже мають літаючі прототипи таксі, а компанія з російським корінням Hoversurf розробляє літаючий байк.

У приміщеннях теж є завдання для комерційних дронів, але поки вони не дуже поширені, у цій галузі йдуть інтенсивні R&D-дослідження. Можливі застосування для такого виду дронів:

- інвентаризація складських приміщень ;
- інспекції будівництв усередині будівель ;
- контроль за безпекою у підземних шахтах ;
- інспекції промислового обладнання у цехах .

Стек програмного забезпечення Drone з відкритим вихідним кодом

Корисною аналогією для розуміння стека польотів є комп’ютери. Існують різні рівні комп’ютера:

1. **Апаратне забезпечення** :
   - Процесори.

- Зберігання.
- ОЗП.

2. **Прошивка** (код низького рівня, який керує апаратним забезпеченням):
   - Windows.
   - Mac.
   - Linux.
3. **Додатки** :
   - Пасьянс.
   - Microsoft Word.
   - Microsoft Paint.

Скажімо, ви розробник, який хоче написати нову програму для онлайн-покеру. Щоб працювати, програмне забезпечення має взаємодіяти з фізичним обладнанням. На щастя, розробник може просто писати високорівневий код і залежати від операційної системи (програмного забезпечення) для зв'язку з апаратним забезпеченням від імені коду високого рівня.

Подібні відносини існують і з програмним забезпеченням для дронів. Апаратний рівень на дроні складається з двигунів, escs, батарей тощо, але як щодо рівня мікропрограми? Введіть ArduPilot, Linux дронів.

Прошивка ArduPilot

Так само, як мікропрограмне забезпечення Windows відповідає за зв'язок з апаратним забезпеченням комп'ютера, ArduPilot відповідає за керування обладнанням дрона. Насправді без ArduPilot або якоїсь мікропрограми керування польотом було б неможливо літати на багатороторних БПЛА. Це тому, що ArduPilot надсилає близько 400 команд в секунду до двигунів дрона, що забезпечує плавний і стійкий політ.

Ardupilot має вирішальне значення для програмістів дронів, оскільки дозволяє їм зосередитися на місіях або додатках високого рівня. Наприклад, припустимо, ви намагаєтеся розробити місію доставки дрона. Останнє, про що ви хочете турбуватися, це те, які значення ШІМ записуються у ваші двигуни 400 разів на секунду! ArduPilot абстрагує низькорівневі обов'язки дрона від програміста.

Dronekit python — це бібліотека python з відкритим вихідним кодом, яка надає функції високого рівня для керування рухом дронів, перевірки стану транспортного засобу та багатьох інших речей.

3.2 Протокол MAVLink

Коли ви намагаєтеся зателефонувати на чийсь номер телефону, це має структуру. Ви повинні набрати код країни, потім код місцевості, а потім номер особи. Це стандартизована система. Якщо я хочу зателефонувати своєму дядькові, я повинен використовувати цю стандартну систему І знати цифри, які його представляють.

MAVLink є протоколом для спілкування з дроном. Протокол MAVLink складається з двох речей(рис. 3.1):

Рисунок 3.1 - MAVLink

1. **Структура пакетів** : так само, як і всі телефонні номери мають однакову (країну)-(локальну)-(особисту) структуру, MAVLink пропонує стандартну структуру пакетів, щоб легко отримувати та передавати дані. Дані MAVLink 1 складаються лише з 8 байтів, а дані MAVLink 2 складаються з 14 байтів.
2. **Стандартні повідомлення** : якщо я хочу зателефонувати своєму дядькові Рону, я просто набираю його номер 1-999-9999, і він бере трубку (сподіваюся). Аналогічно, протокол MAVLink пропонує конкретні повідомлення з певними функціями. Наприклад, повідомлення MAVLink 78 і команда 22 є командою зльоту, яка запускає безпілотник у повітря, з цільовою висотою, що міститься в повідомленні. На додаток до команд, повідомлення MAVLink також можуть бути суто інформаційними. Наприклад, повідомлення MAVLink 25 містить поточний статус GPS транспортного засобу.

Як реалізується протокол MAVLink?

MAVLink пропонує стандартні повідомлення, які можуть бути прийняті та зрозумілі мікропрограмою ArduPilot.

Чудова особливість MAVLink полягає в тому, що він працює з будь-яким програмним забезпеченням дрона, яке підтримує MAVLink, а не тільки з Ardupilot. Існує багато різних типів прошивки дрона з підтримкою MAVLink, ще одним є PX4.

Оскільки MAVLink є лише стандартним протоколом пакетів/повідомлень, його можна структурувати в бібліотеках практично для будь-якої мови програмування. Це включає:

- Java.
- C++.
- Go.
- Python.
- Багато інших.

Після того, як бібліотека представлена на бажаній мові програмування, її можна використовувати для отримання інформації від дрона або керування дроном.

У python бібліотека pymavlink визначає повідомлення MAVLink у формі python. Бібліотека python dronekit використовує pymavlink і встановлює з'єднання з дроном. Це дозволяє безпосередньо керувати дроном прямо зі скрипту на Python, тому будь-який дрон MAVLink є програмованим дроном.

Програмування дронів на Python

Тепер, коли маємо базове розуміння стека програмного забезпечення для дронів з відкритим вихідним кодом, фактично можна розпочати кодування дрона за допомогою python dronekit! Як зазначалося раніше, ми навіть можемо розпочати програмування дрона без реального дрона! Ми зробимо це за допомогою симулятора ArduPilot SITL.

3.3 ArduPilot SITL

SITL означає 'Software-In-The-Loop'. Є багато цікавих аспектів SITL, деякі з них:

- Той самий вихідний код можна зібрати для справжньої дошки автопілота або імітованого дрона
- Ми можемо протестувати справжню прошивку прямо з нашого комп'ютера. Наприклад, що станеться з дроном, якщо він раптово втратить сигнал GPS? Це можна перевірити на SITL.
- Також можна протестувати високорівневі скрипти dronekit python проти змодельованого ardupilot, перш ніж випробувати код у польових умовах.
- Ми можемо використовувати власний симулятор ArduPilot SITL (на фото нижче) або використовувати SITL з більш просунутим симулятором, таким як Gazebo. (рис. 3.2)

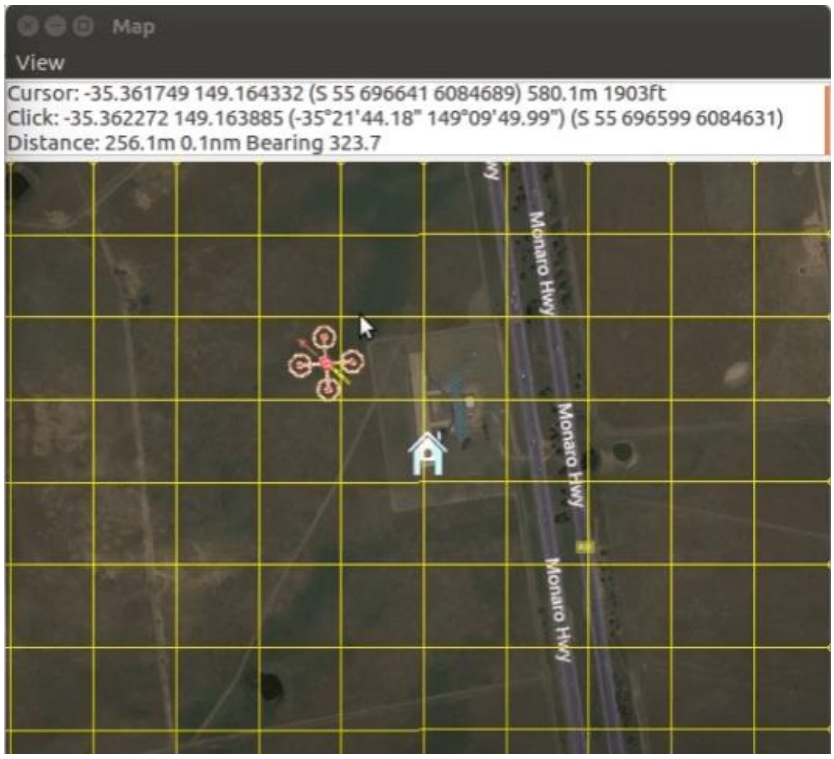


Рисунок 3.2 – карта в реальному часі з координатами

Щоб запустити змодельований дрон ArduPilot, нам потрібно завантажити деякі залежності.

Завантажимо вихідний код ArduPilot

4. git https://github.com/ardupilot/ardupilot.
5. Ardupilot.
6. git submodule update – init – рекурсивно.

7. ArduCopter.
8. ../Tools/autotest/sim_vehicle.py – консоль – карта.

Тепер дрон ArduPilot SITL запущений. Дроном SITL можна керувати за допомогою MAVProxy, dronekit python або іншої наземної станції керування.

Тепер, коли у нас є спосіб запустити змодельований квадрокоптер MAVLink, нам потрібно завантажити dronekit python.

Завантажте Dronekit Python

- pip install dronekit==2.9.2.

3.4 Написання першого сценарію Dronekit Python

Напишемо базовий скрипт для dronekit python, який буде керувати дроном на певній цільовій висоті, а потім просто приземлитися. Знову ж таки, все це буде зі скрипту на Python. Можна застосувати це на своєму транспортному засобі SITL або на реальному програмованому дроні.

Простий код програми для зльоту та посадки:

```
#############Встановлення бібліотек#####################


from dronekit import connect, VehicleMode,LocationGlobalRelative,APIException
import time
import socket
import exceptions
import math

##############Функції##########################


##Function to arm the drone props and takeoff at targetHeight (m)


def arm_and_takeoff(targetHeight):
```

```
    while vehicle.is_armable!=True:
        print("Waiting for vehicle to become armable.")
        time.sleep(1)
    print("Vehicle is now armable")

    vehicle.mode = VehicleMode("GUIDED")

    while vehicle.mode!='GUIDED':
        print("Waiting for drone to enter GUIDED flight mode")
        time.sleep(1)
    print("Vehicle now in GUIDED MODE. Have fun!!")

    vehicle.armed = True
    while vehicle.armed==False:
        print("Waiting for vehicle to become armed.")
        time.sleep(1)
    print("Look out! Virtual props are spinning!!")

    vehicle.simple_takeoff(targetHeight)

    while True:
        print("Current Altitude: %d"%vehicle.location.global_relative_frame.alt)
        if vehicle.location.global_relative_frame.alt>=.95*targetHeight:
            break
        time.sleep(1)
    print("Target altitude reached!!")

    return None
```

############Основний файл#############

####sim_vehicle.py opens up port on localhost:14550

```
vehicle = connect('127.0.0.1:14550',wait_ready=True)
```

```
####Arm the drone and takeoff into the air at 5 meters


arm_and_takeoff(5)
print("Vehicle reached target altitude")


####Once drone reaches target altitude, change mode to LAND



vehicle.mode=VehicleMode('LAND')
while vehicle.mode!='LAND':
        print("Waiting for drone to enter LAND mode")
        time.sleep(1)
print("Vehicle now in LAND mode. Will touch ground shortly.")
```

Результуючий код дозволив нам робити відносно простий зльот та посадку дрона за підключенням до IP адреси та токену персоналізованого доступу самого дрону.

Висновки за розділом 3.

В цьому розділі було проведено огляд основних технологій та бібліотек для написання коду додатку, що дозволяє керувати дроном з використанням мови програмування Python, використовуючи протокол MAVLink та SITL/Gazebo. Наведено базовий приклад коду для квадрокоптеру на базі контролера Raspberry Pi.

РОЗДІЛ 4. РЕАЛІЗАЦІЯ ПРОГРАМНОГО ДОДАТКУ КЕРУВАННЯ ДРОНОМ

4.1 Написання базового функціоналу

За поставленою задачею написання програми для керування безпілотним літальним апаратом було обрано мову програмування Python, що дозволяє швидко та ефективно впровадити систему безпосереднього керування IoT контролером Raspberry Pi на самому дроні.

Нижче наведено код функціоналу першої програми що виконує функції контролю дрона, для базового запуску дрона та перевірки його систем(Додаток Б):

Результат підключення до дрона наведено нижче(рис. 4.1)

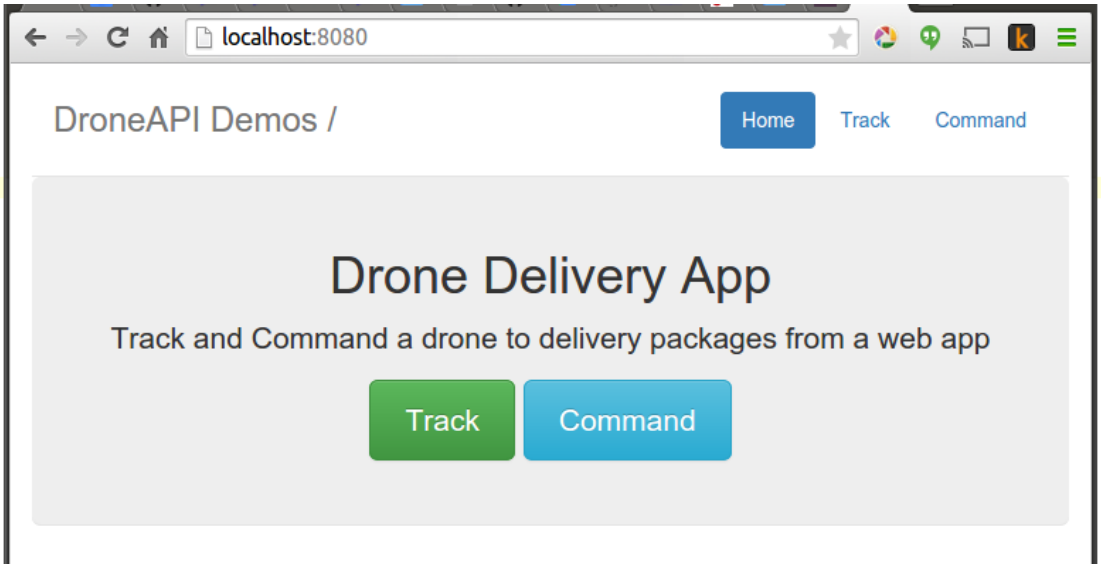


Рисунок 4.1 – стартова сторінка додатку

Можемо відстежити положення дрона(рис. 4.2):

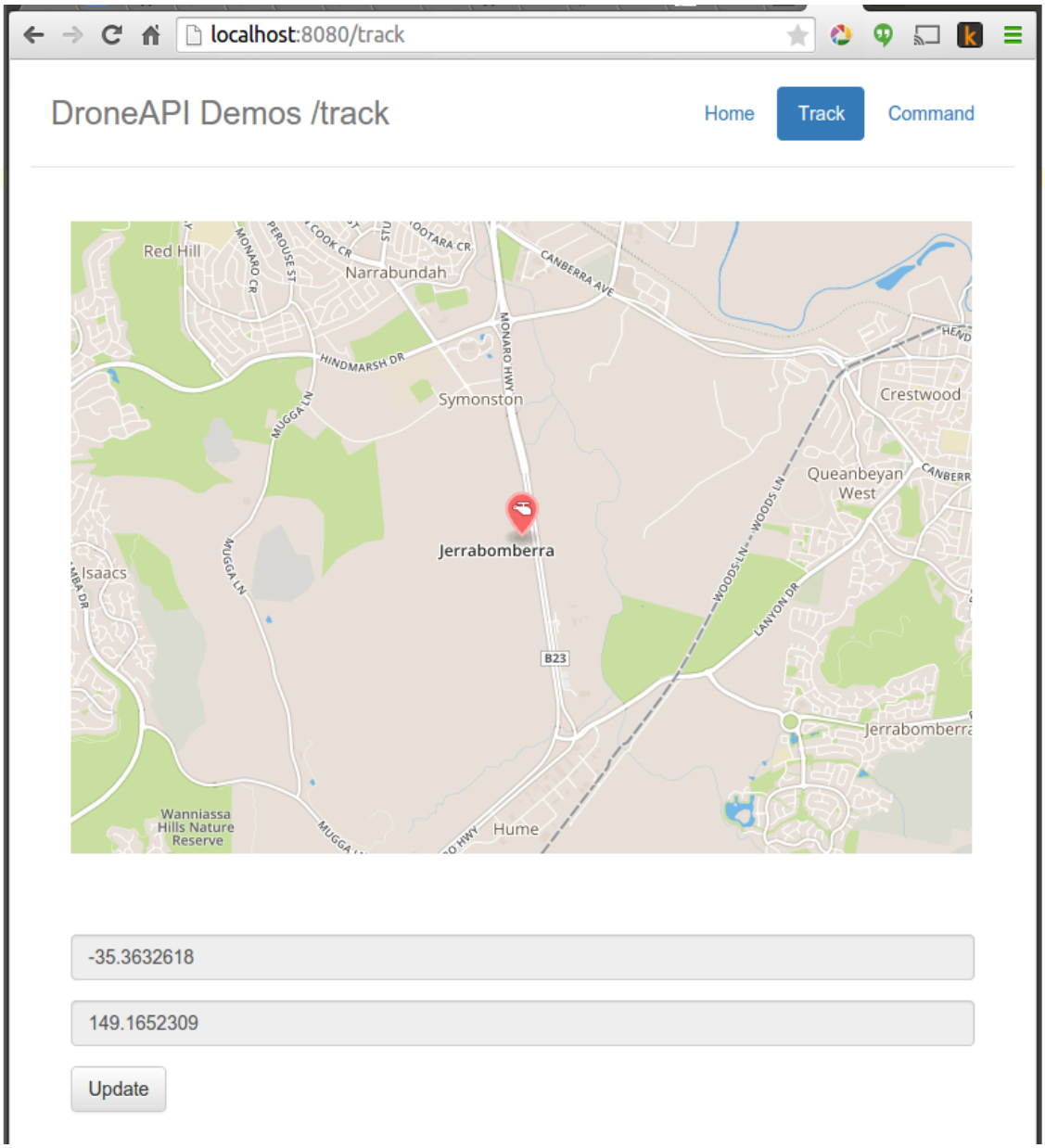


Рисунок 4.2 – відстеження координат

Після цього матимемо змогу задати координати для пересування дрону(рис.4.3):

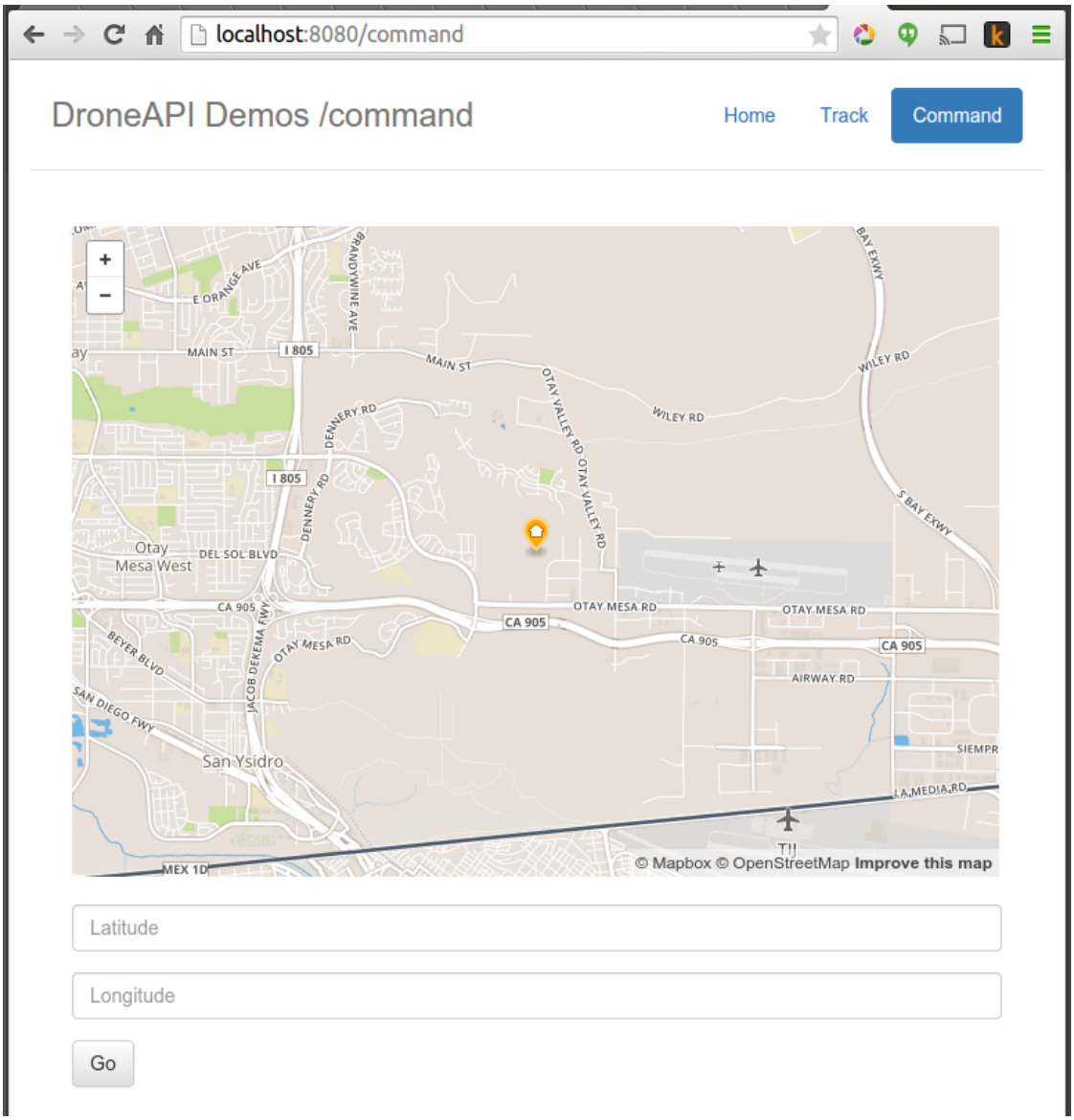


Рисунок 4.3 – завдання для польоту

4.2 Розширення функціонування дрону

Наступним етапом було написання маршрутів доставки з фіксованою швидкістю. Він демонструє три методи для явного визначення цільової позиції та дві команди для керування рухом шляхом встановлення векторів швидкості транспортного засобу. Він також показує, як надсилати команди для керування рисканням (напрямом, на який вказує передня частина автомобіля), регіоном інтересу, швидкістю та місцем розташування, а також деякі корисні функції для перетворення між системами відліку.

Подальший функціонал можна описати як задання задач до дрона, маршрути, виконання дій за замовчуванням, тощо(рис. 4.4):

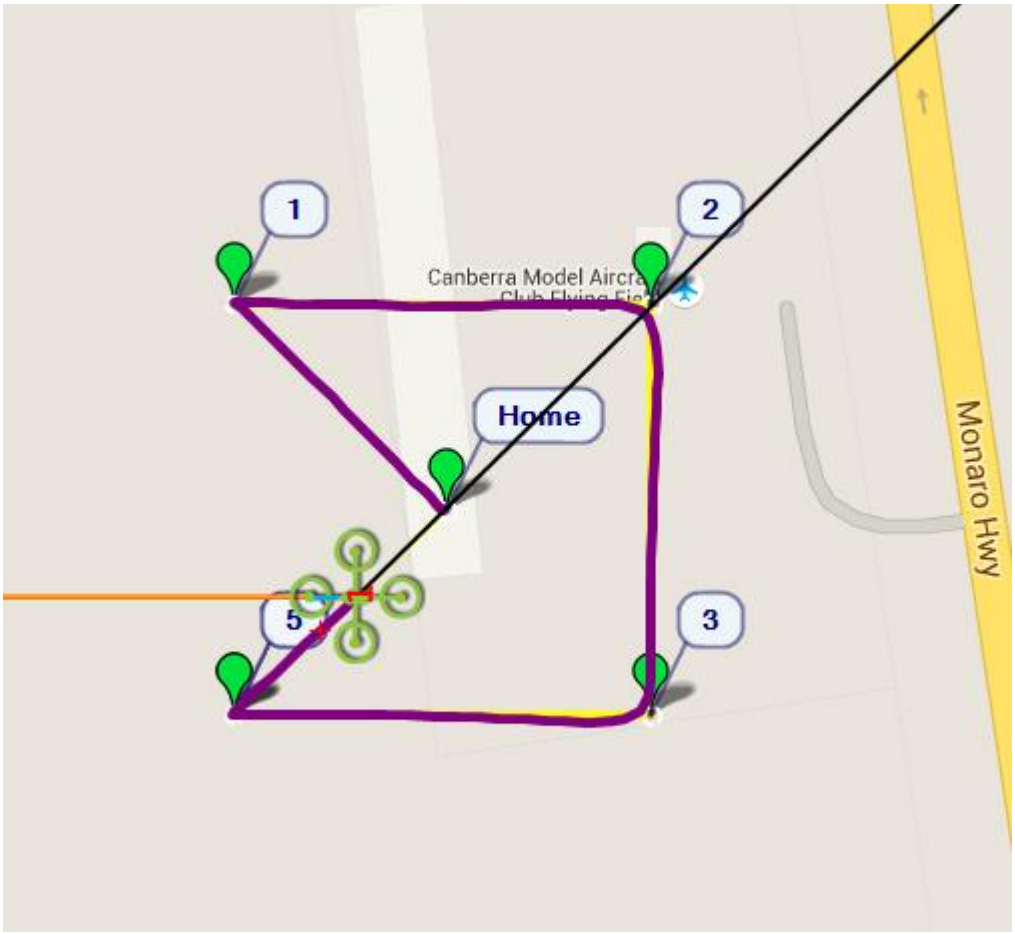


Рисунок 4.4 – приклад постановок задач до квадрокоптера

Подальший код наведено в додатку В.

4.3 Проектування маршрутів та місій

Далі показано, як керувати рухом квадрокоптера і надсилати негайні команди в режимі Керування . Він демонструє три методи для явного визначення цільової позиції та дві команди для керування рухом шляхом встановлення векторів швидкості транспортного засобу. Він також показує, як надсилати команди для керування рисканням (напрямом, на який вказує передня частина дрона), регіоном інтересу, швидкістю та місцем розташування, а також деякі корисні функції для перетворення між системами відліку(рис. 4.5).

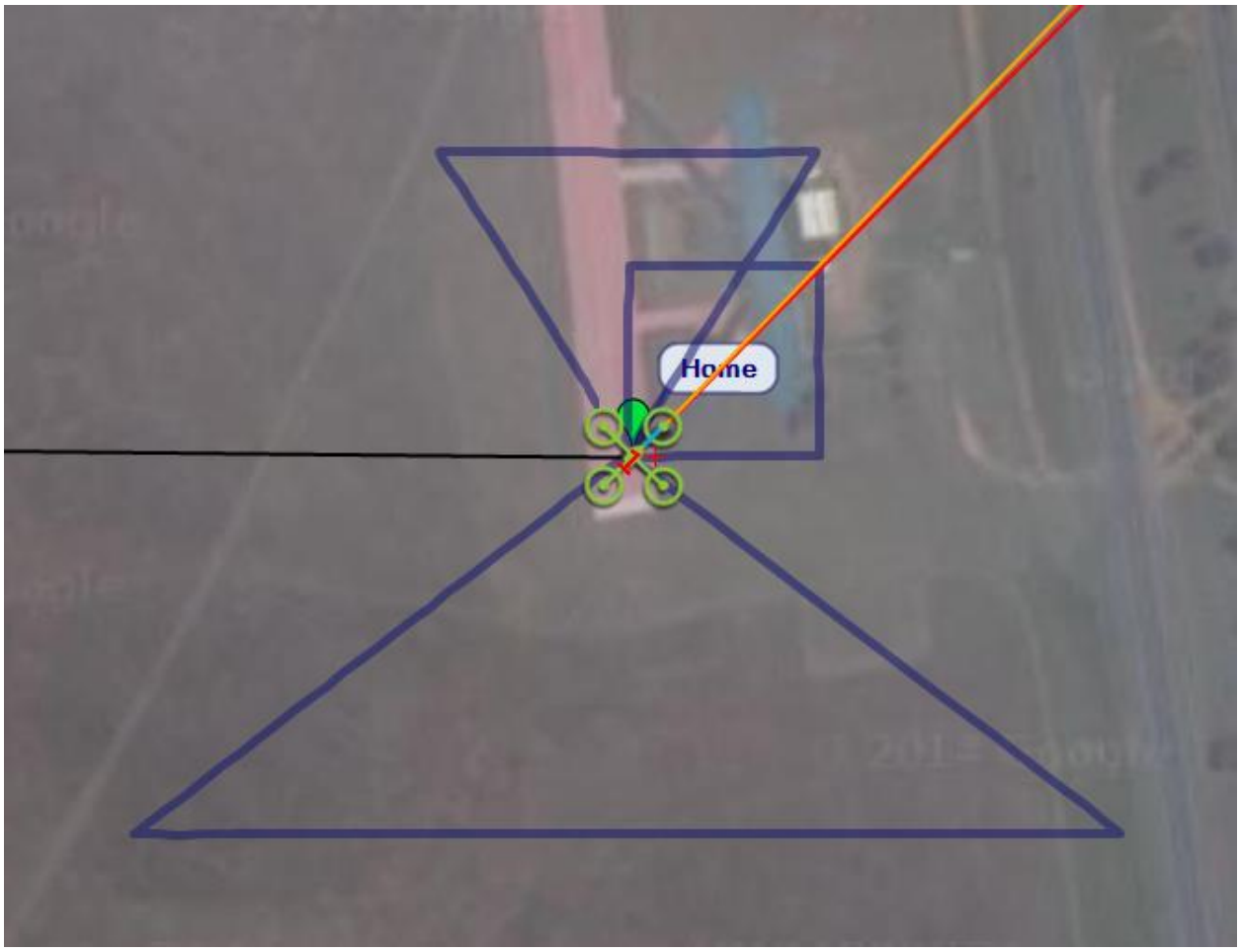


Рисунок 4.5 – Встановлення місця призначення за допомогою положення та зміни швидкості

Нижче показано механізм поривань та зміни швидкості(рис. 4.6)

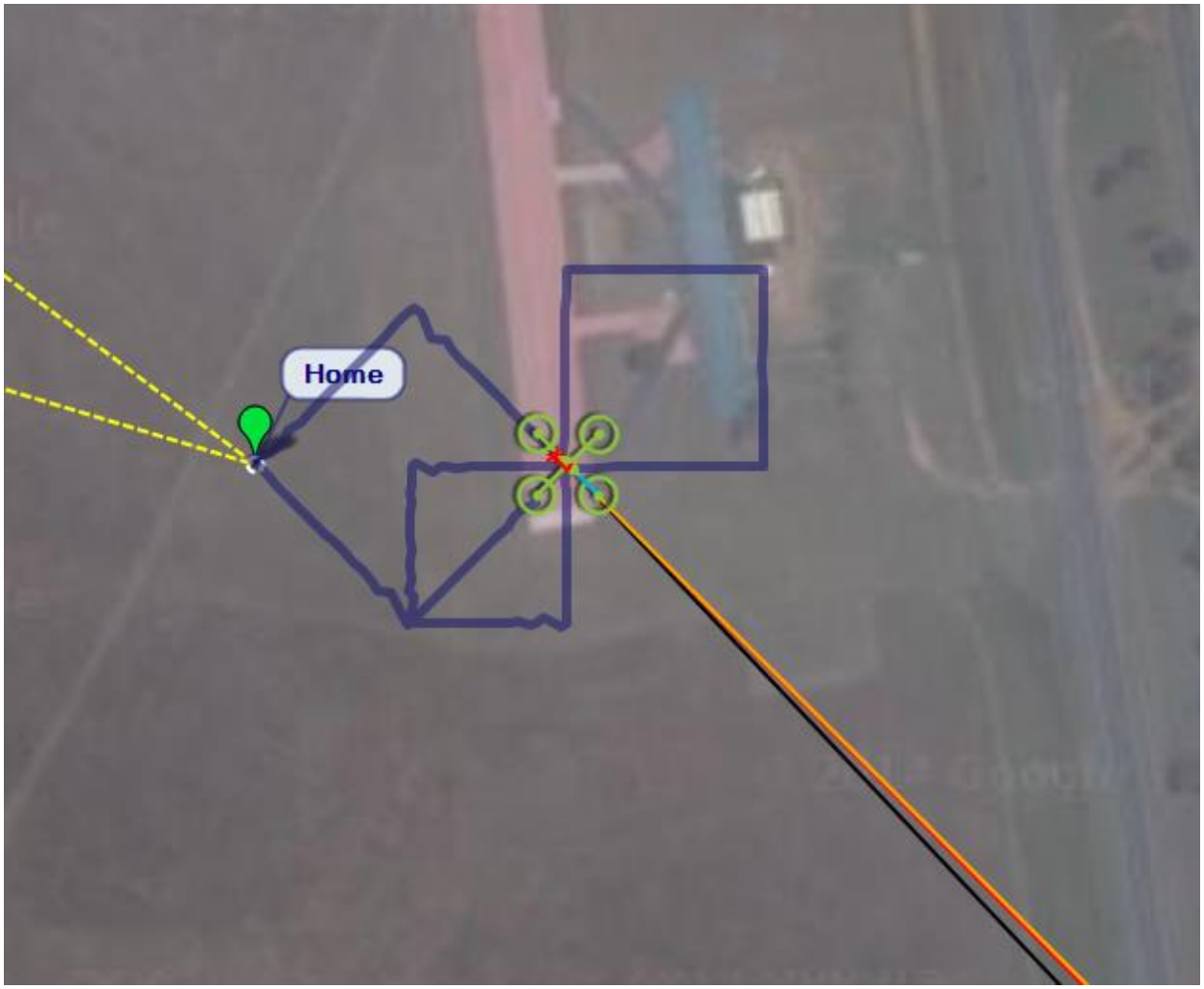


Рисунок 4.6 - Встановлення пункту призначення за допомогою швидкості та зміна повороту та місцезнаходження під час виконання маршруту

Результуючий код функціоналу наведено в Додатку Г:

Висновки до розділу 4.

Написано основні фрагменти комплексного додатку для керування дроном, в вигляді мобільного додатку розглянуто основні алгоритми реалізації планування місій, маршрутів, корегування координат на відстеження поточної висоти та координат квадрокоптера.

ВИСНОВОК

У цій роботі розглядаються потенціали та проблеми, які виникають при взаємодії з дронами або безпілотні автономні транспортні засоби в управлінні ланцюгом поставок та логістиці. Запропоновано для використання рішення компанії Teltonika Networks для побудови мережної інфраструктури IoT для контролю та забезпечення зв'язку в умовах підвищенного попиту на логістику та запровадження нових технологій в систему Дрон + Логістика. Було проведено систематичний огляд літератури, щоб охопити динаміку, що оточує безпілотники, і надати своєчасний і вичерпний огляд того, що було вивчено на даний момент і що необхідно дослідити в майбутньому. Було відібрано та ретельно проаналізовано 40 публікацій. Результатами цього дослідження щодо потенційних переваг застосування дронів у логістиці є:

За результатами розділу 1 було проаналізовано, що є поняття «інформація», які види інформації існують, як вона впливає на ланцюги постачання на логістичну складову транспортних компаній, як можна покращити логістику за допомогою дронів та які технології щоденно допомагають компаніям аналізувати дорожній трафік й планувати свою безпосередню діяльність. Проведено аналіз рішень компаній-лідерів в транспортному сегменті. Виявлено ефективність та перспективність застосування новітніх технологій в логістичній складовій, оцінено потенцільну ефективність імплементації дронів для повітряної доставки в транспортній компанії. В розділі 2 наведено топологічні й проектні рішення для побудови мережної інфраструктури з використанням технологій IoT, основні переваги та недоліки вибору постачальника рішень, та саме які моделі є найвігіднішими з аналізу ринку, та за результатами проходження науково-дослідної практики

На базі практики було розглянуто рішення компанії Teltonika Networks, що на протязі науково-дослідної практики проявили себе как ефективні та надійні рішення для побудови критичної мережної інфраструктури особливо під час воєнного стану що може замінити технологію СтарЛінк від Ілона Маска використовуючи не супутниковий звя'зок а технологію стільникового зв'язку з вищою швидкістю аніж

В розділі 3 було проведено огляд основних технологій та бібліотек для написання коду додатку, що дозволяє керувати дроном з використанням мови програмування Python, використовуючи протокол MAVLink та SITL/Gazebo.

Наведено базовий приклад коду для квадрокоптеру на базі контролера Raspberry Pi.

В рамках розділу 4 написано основні фрагменти комплексного додатку для керування дроном, в вигляді мобільного додатку розглянуто основні алгоритми реалізації планування місій, маршрутів, корегування координат на відстеження поточної висоти та координат квадрокоптера.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. PWC, Побудова довіри до дронів. Великобританія [Електронний ресурс] // Режим доступу до ресурсу: https://www.pwc.co. uk/issues/intelligent-digital/drones-and-trust.html, 2019. (дата звернення 10 травня 2022 р.).
2. Ф. Клаузер, С. Педрозо, Великі дані з неба: популярні уявлення про приватні дрони в Швейцарії, Geograph. Helv. 72 (4) (2017) 285–293. дата звернення 10 травня 2022 р.).
3. Агентство з авіаційної безпеки Європейського Союзу (EASA), звіт про оцінку дослідження міської повітряної мобільності, [Електронний ресурс] // Режим доступу до ресурсу: https://www.easa.europa.eu/sites/default/files/dfu/uam_detailed_survey_evaluation.pdf, 2021. (дата звернення 10 травня 2022 р.)).
4. J. West, P. Klofstad, J. Uscinski та ін., Громадянська підтримка використання та регулювання внутрішніх дронів, Am. політ. Res. 47 (1) (2019) 119–151. (дата звернення 10 травня 2022 р.)
5. Цвікл, Х. Фарбер, Дж. Хамм, Порівняння занепокоєння громадськості та підтримки регулювання дронів з поточною правовою базою, Behav. наук. Закон 37 (1) (2019) 109–124 . (дата звернення 10 травня 2022 р.)
6. Департамент транспорту, транспорту та транспортних технологій: набір даних хвилі 5 трекера ставлення громадськості, словник і таблиці. [Електронний ресурс] // Режим доступу до ресурсу:https:// www.gov.uk/government/publications/transport-and-transport-technology-publi c-attitudes-tracker, 2021. (дата звернення 10 травня 2022 р.)
7. H. Eißfeldt, V. Vogelpohl, M. Stolz, A. Papenfuß, M. Biella, J. Belz, D. Kügler, The acceptance of civil drones in Germany, CEAS Aeronaut. J. 11 (3) (2020) 665–676, [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/10.1007/s13272-020-00447-w. (дата звернення 10 травня 2022 р.)
8. Дж. Нельсон, Т. Грубешич, Д. Воллес, А. Чемберлен, Погляд згори: огляд сприйняття громадськістю безпілотних літальних апаратів та приватності, J. Urban Technol. 26 (1) (2019) 83–105. (дата звернення 10 травня 2022 р.)
9. C. Lidynia, R. Philipsen, M. Ziefl, Droning on about drones—accepting of drones of drones in percepted barriers in civil use

contexts, in: P. Savage-Knepshield, (дата звернення 10 травня 2022 р.)
10. Дж. Чен (ред.), Досягнення людського фактора в роботах і безпілотних системах. Advances in Intelligent Systems and Computing, vol. 499, Springer, Cham, 2017, [Електронний ресурс] // Режим доступу до ресурсу: https://doi.org/10.1007/978-3-319-41959-6_26. (дата звернення 10 травня 2022 р.)
11. P. Boucher, You wouldn't have your granny use them': малювання кордонів між прийнятним і неприйнятним Застосування цивільних дронів, наук. інж. Етика 22 (5) (2016) 1391–1418. (дата звернення 10 травня 2022 р.)
12. Департамент транспорту, Громадський діалог щодо використання дронів у Великобританії, [Електронний ресурс] // Режим доступу до ресурсу: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/579550/drones-uk-public-dialogue.pdf, 2016 р. (дата звернення 10 травня 2022 р.).
13. Б. Редді, Д. ДеЛаурентіс, Опитування щодо зменшення невизначеності сприйняття безпілотних літаків громадськістю та зацікавленими сторонами, Транспорт. Res. Рек. 2600 (2016) 80–93 (дата звернення 10 травня 2022 р.).
14. Р. Клотьєр, Д. Грір, Д. Грір, А. Мехта, Сприйняття ризику та сприйняття дронів громадськістю, Risk Anal. 35 (6) (2015) 1167–1183. (дата звернення 10 травня 2022 р.)
15. Департамент транспорту, транспорту та технологій: трекер суспільного ставлення, підсумковий звіт хвиль 1 і 2, Лондон. [Електронний ресурс] // Режим доступу до ресурсу:https://www.gov.uk/go vernment/publications/transport-and-transport-technology-public-attitudes-tracke r, 2018. (Доступ 24 серпня 2021). (дата звернення 10 травня 2022 р.)
16. L. Tan, B. Lim, G. Park, K. Low, V. Yeo, Public acceptance of drone applications in a high urbanized environment, Technol. соц. 64 (2021), [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/10.1016/ j.techsoc.2020.101462. (дата звернення 10 травня 2022 р.)
17. К. Седіг, М.Б. Сітон, І.Р. Дреннан, С. Ческес, К.Н. Дейнті: «Дрони — чудова ідея! що таке AED?» нові ідеї якісного дослідження сприйняття громадськістю використання дронів для доставки автоматичних зовнішніх дефібриляторів,

Resuscitation 4 (травень 2020 р.) (2020 р.) 100033, [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/10.1016/j.resplu.2020.100033. (дата звернення 10 травня 2022 р.)
18. Поштова служба США, сприйняття громадськістю доставки дронів у Сполучених Штатах, [Електронний ресурс] // Режим доступу до ресурсу:https://www.uspsoig.gov/document/public-perception-drone-delivery-united-states, 2016. (дата звернення 10 травня 2022 р.)
19. К. Р. Теодор, Резюме проектного середовища NASA для проекту нових вертикальних ліфтів (DELIVER), в: Матеріали Міжнародної технічної конференції AHS з проектування аеромеханіки для трансформаційного вертикального польоту, Сан-Франциско, Каліфорнія, США, 2018. (дата звернення 10 травня 2022 р.)
20. С. Комасова, Й. Тесар, П. Сукуп, Сприйняття ризиків, пов'язаних з дроном, у чеському суспільстві, Технол. соц. 61 (грудень 2019 р.) (2020 р.), [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/10.1016/j. techsoc.2020.101252. (дата звернення 10 травня 2022 р.)
21. Y. Wang, H. Xia, Y. Yao, et al., Flying eyes and hidden controllers: aqualitary study of People's a konfidencial perceptions of civilian drones in the US, Proc. Прив. Enhanc. Технологія. (3) (2016) 172–190. (дата звернення 10 травня 2022 р.)
22. L.M. PytlikZillig, B. Duncan, S. Elbaum, C. Detweiler, Безпілотник під будь-яким іншим ім'ям: цілі, надійність кінцевого користувача та кадрування, але не термінологія, впливають на суспільну підтримку дронів, IEEE Technol. соц. маг. 37 (1) (2018) 80–91, [Електронний ресурс] // Режим доступу до ресурсу: https://doi.org/ 10.1109/MTS.2018.2795121. (дата звернення 10 травня 2022 р.)
23. Дж. Кумза, К. Добсон, Гендерне різноманіття в індустрії БПЛА (дрони), міжнародний. Дж. Гендер, наук. Технологія. 10 (3) (2019) 366–377. [Електронний ресурс] // Режим доступу до ресурсу:http://genderandset.open.ac.uk/index.php/genderandset/article/view/564. (Accessed 2 July 2021). (дата звернення 10 травня 2022 р.)

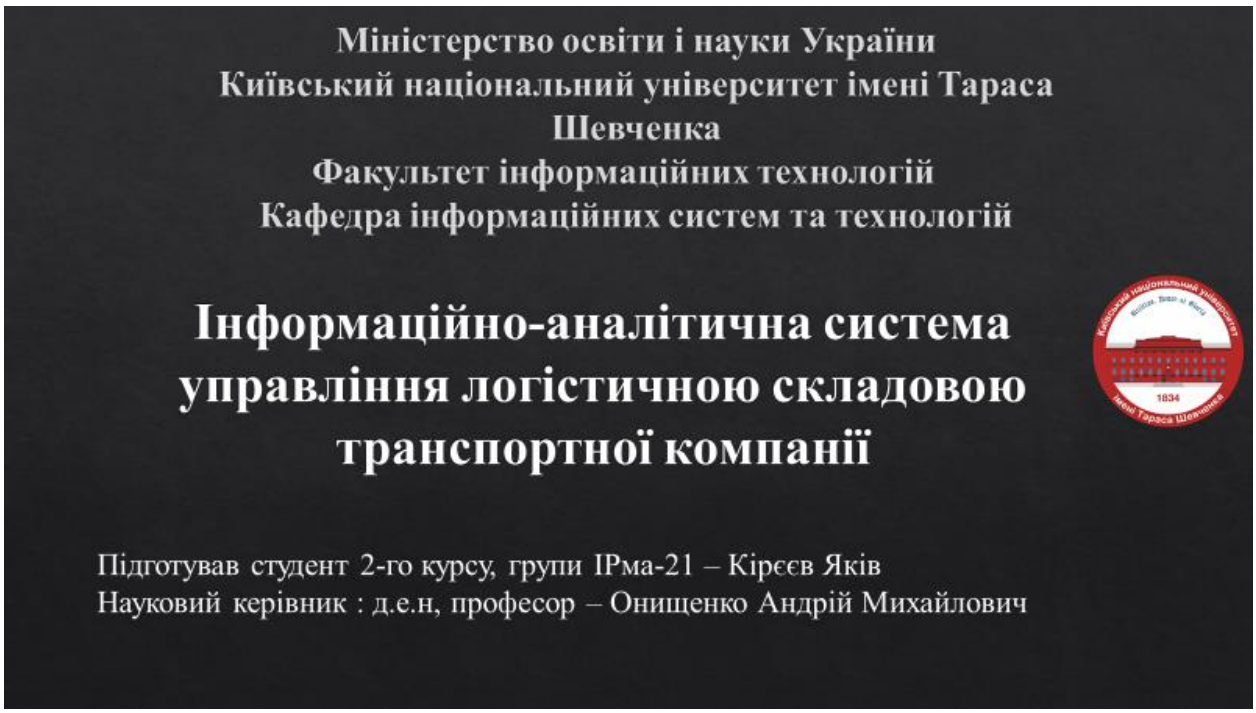
24. S. Ogilvie, A. McCarthy, W. Allen, et al., Unmanned aerial vehicles and biosecurity: enabling participatory-design to help address social licence to operate issues, Forests 10 (8) (2019) 1–19. (дата звернення 10 травня 2022 р.)
25. S. Truog, et al., Insights before flights: how community perceptions can make or break medical drone deliveries, Drones 4 (3) (2020) 1–14. (дата звернення 10 травня 2022 р.)
26. M. Bauer, G. Gaskell, Towards a paradigm for research on social representations, J. Theor. Soc. Behav. 29 (2) (1999) 163–186. (дата звернення 10 травня 2022 р.)
27. S.L. Macsween-George, Will the public accept UAVs for cargo and passenger transportation? IEEE Aerospace Conf. Proc. 1 (2003) 357–367, [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/ 10.1109/AERO.2003.1235066. (дата звернення 10 травня 2022 р.)
28. X. Zhu, T.J. Pasch, A. Bergstrom, Understanding the structure of risk belief systems concerning drone delivery: a network analysis, Technol. Soc. 62 (April) (2020) 101262, [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/10.1016/j.techsoc.2020.101262. (дата звернення 10 травня 2022 р.)
29. E.M. Markowitz, M.C. Nisbet, A.J. Danylchuk, S.I. Engelbourg, What's that buzzing noise? Public opinion on the use of drones for conservation science, Bioscience 67 (4) (2017) 382–385, [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/10.1093/biosci/bix003. (дата звернення 10 травня 2022 р.)
30. C. Del-Real, A.M. Díaz-Fern´andez, Lifeguards in the sky: examining the public acceptance of beach-rescue drones, Technol. Soc. 64 (December 2020) (2021), [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/10.1016/j.techsoc.2020.101502. (дата звернення 10 травня 2022 р.)
31. D. Paddeu, I. Shergold, G. Parkhurst, The Social Perspective on Policy towards Local Shared Autonomous Vehicle Services (LSAVS), vol. 98, 2020, pp. 116–126 . (дата звернення 10 травня 2022 р.)
32. O. Kunze, F. Frommer, The matrix vs. The Fifth element - assessing future scenarios of urban transport from a sustainability

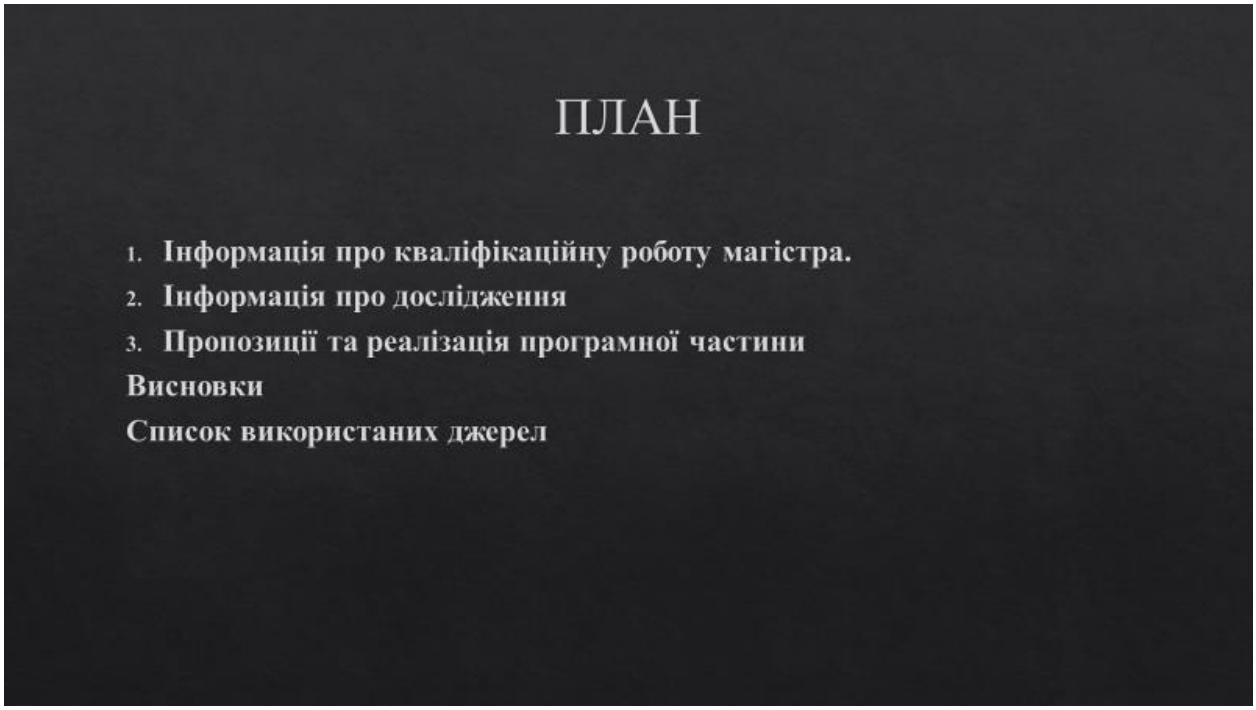
perspective, Sustainability 13 (6) (2021) . (дата звернення 10 травня 2022 р.)
33. Amazon, Amazon Prime air, [Електронний ресурс] // Режим доступу до ресурсу:https://www.amazon.com/Am azon-Prime-Air/b?ie UTF8&node 8037720011, 2021. (дата звернення 10 травня 2022 р.)
34. Financial Times, UK regulator gives green light to delivery drone trials, Financial Times [Online] 20 April 2021. [Електронний ресурс] // Режим доступу до ресурсу:https://www.ft.com/content/6648 7d88-a6b3-4e46-9b8a-00e38e93d3af, 2021(дата звернення 10 травня 2022 р.).
35. J. Stilgoe, T. Cohen, Rejecting acceptance: learning from public dialogue on self- driving vehicles, Sci. Publ. Pol. (2021) 1–11, [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/10.1093/scipol/ scab060, 00. (дата звернення 10 травня 2022 р.)
36. B. Sah, R. Gupta, D. Bani-Hani, Analysis of barriers to implement drone logistics, Int. J. Log. Res. Appl. (2020) 1–20, [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/10.1080/ 13675567.2020.1782862, 0(0). (дата звернення 10 травня 2022 р.)
37. M. MacIas, C. Barrado, E. Pastor, P. Royo, The future of drones and their public acceptance, in: AIAA/IEEE Digital Avionics Systems Conference - Proceedings, 2019-Septe(Vll), 2019, [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/10.1109/DASC43569.2019.9081623. (дата звернення 10 травня 2022 р.)
38. M.P. Wadey, S.N. Cope, R.J. Nicholls, K. Mchugh, G. Grewcock, T. Mason, Ocean & Coastal Management Coastal fl ood analysis and visualisation for a small town, Ocean Coast Manag. 116 (2015) 237–247, [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/10.1016/j. ocecoaman.2015.07.028. (дата звернення 10 травня 2022 р.)
39. Xuejun Zhang, Yang Liu, Yu Zhang, Xiangmin Guan, Daniel Delahaye, Li Tang, Safety assessment and risk estimation for unmanned aerial vehicles operating in national airspace system, J. Adv. Transport. October (2018) 1–11, [Електронний ресурс] // Режим доступу до ресурсу:https://doi.org/ 10.1155/2018/4731585. (дата звернення 10 травня 2022 р.)

40. G. Marsden, et al., All change? The future of travel demand and the implications for policy and planning, first report of the commission on travel demand, London. [Електронний ресурс] // Режим доступу до ресурсу:http://www.demand.ac.uk/wp-content/uploads/2018/04/Fut ureTravel_report_final.pdf, 2018 (дата звернення 10 травня 2022 р.)

Додатки

Додаток А





ЗАВДАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ МАГІСТРА

- оволодіння сучасною методологією наукового дослідження;
- закріплення знань, умінь і навичок, здобутих у процесі вивчення дисциплін за магістерською програмою;
- оволодіння сучасними методами збирання, аналізу та оброблення наукової інформації;
- Побудова поточного ІОТ рішення за результатами проходження всієї магістерьскої програми;
- формування уявлень про сучасні інформаційні технології та IoT-системи;
- формування пропозицій для реалізації ефективних стратегій на підприємствах в наслідок опанування навичок і знань з логістики та стеку технологій Інтернету Речей.

ВСТУП

- **Метою кваліфікаційної роботи магістра** є оволодіння студентом необхідних компетенцій, які дадуть йому можливість впливати на об'єкти управлінської діяльності і домогтися високих техніко-економічних показників з розвитку на перспективу обраної теми дипломної роботи. За обраною темою дипломної роботи повинен бути проведений аналіз актуальних рішень, має бути побудована система з урахуванням наукової новизни в галузі Інтернету Речей, повинна бути сформована база актуальних пропозицій в сфері логістики для забезпечення високої стратегічної ефективності під час використання цієї системи.

Інформаційно-аналітична система управління логістичною складовою транспортної компанії

DHL, датчики для збору додаткової інформації.

- Цей логістичний гравець має вбудовані датчики у всі свої транспортні засоби доставки, а смартфони з підтримкою GPS покривають будь-які прогалини. Третя сторона перевіряє ці датчики на точність, а потім дані про надійність та своєчасність цих датчиків використовуються, коли DHL бере участь у торгах за новими контрактами. DHL розробила Smart Truck для покращення планування маршрутів на основі даних, отриманих з GPS-трекерів і доріг. Телематичні бази даних дозволяють водіям швидко отримувати оновлення маршрутів і уникати заторів, спричинених аваріями, погодними умовами тощо. Наразі Smart Truck зменшив загальну кількість миль на 15%, допомагаючи зменшити споживання палива та викиди CO2.

UPS, оптимізація процесів «останньої милі».

- Остання миля ланцюга поставок, як відомо, неефективна, коштує до 28% загальної вартості доставки упаковки. Під час процесу доставки, навіть після паркування поблизу, GPS-телефон куратора передає дані в центр UPS, постійно враховуючи, скільки часу займає доставка. Це дозволяє логістичним компаніям бачити моделі, які можна використовувати для оптимізації їх стратегій доставки.

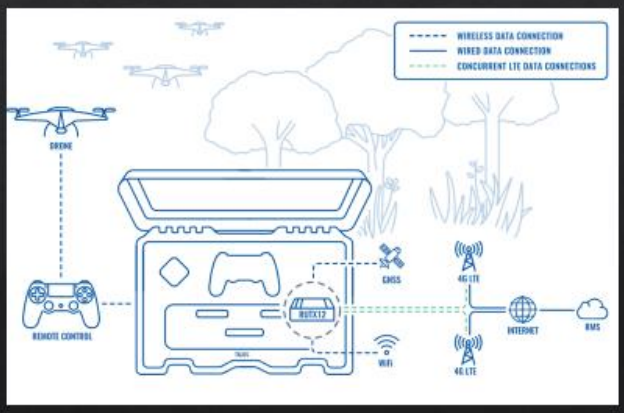
TIBA Іспанія, дані третьої сторони.

- Зберігання швидкопсувних продуктів у свіжому вигляді було постійною проблемою для логістичних компаній. Однак великі дані та Інтернет речей можуть дати водіям і менеджерам доставки набагато краще уявлення про те, як вони можуть запобігти витратам через загибель товарів. Датчик температури встановлений усередині вантажівки, щоб контролювати стан продуктів всередині і передавати ці дані разом з даними про дорожній рух і роботу на дорозі на центральний комп'ютер маршрутизації. Цей комп'ютер може попередити водія, якщо спочатку вибраний маршрут приведе до танення швидкопсувних продуктів, і замість цього запропонувати альтернативні маршрути.

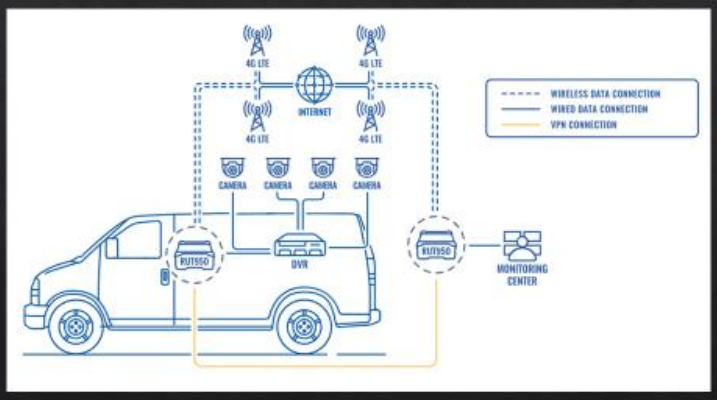
NAEKO Logistics, оптимізація маршрутів.

- Наглядність ланцюга поставок є пріоритетом цієї складської компанії. Він надає особам, які приймають рішення, потужний і динамічний інструмент нагляду, який вимірює інформацію про затримки, демередж і випадки затримання, а також роботу постачальників логістичних послуг і автоматично створює звіти про всі ці дані. Крім того, оптимізація маршруту відіграє вирішальну роль у визначенні шляху транспортного засобу над можливими маршрутами всередині складу, щоб оптимізувати транспортний потік з точки зору витрат і часу. Ця розвідка маршрутизації дозволяє компанії заощадити час і витрати на ручне встановлення послідовності для персоналу, скоротити пробіг і звести до мінімуму невдалі поставки.

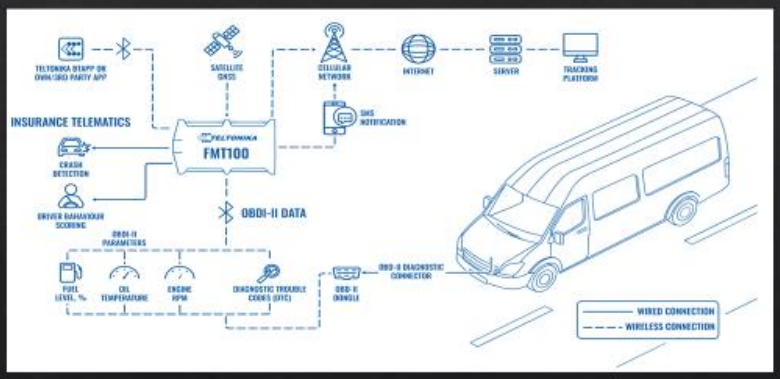
Інформаційно-аналітична система управління логістичною складовою транспортної компанії



Побудова критичної мережної інфрастурктури для логістики

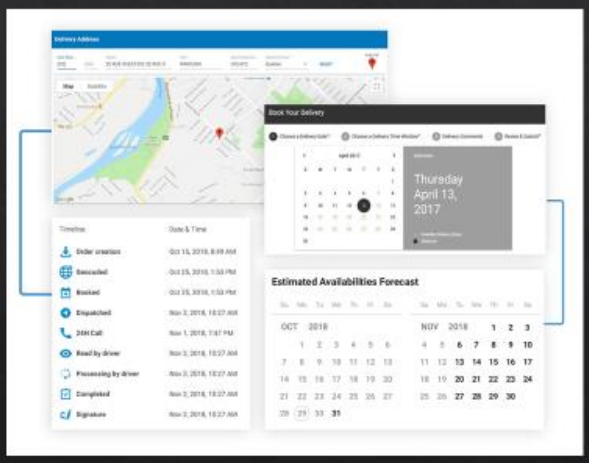


FMT100 – датчик для основних показників авто та водія



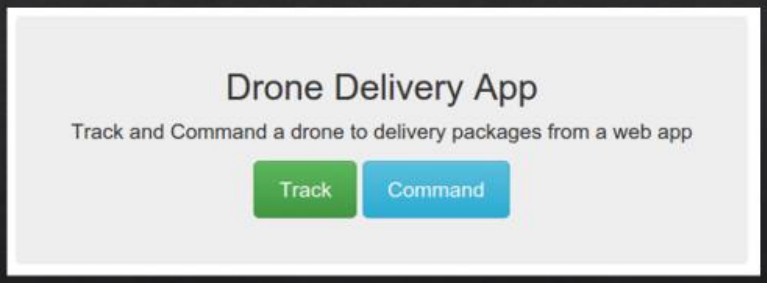
Система контролю логістичною складовою за допомогою Datapine

Система контролю логістичною складовою

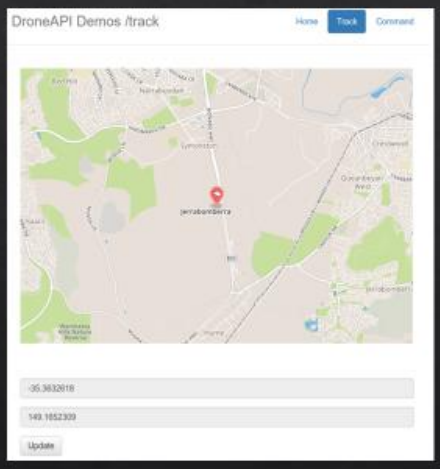


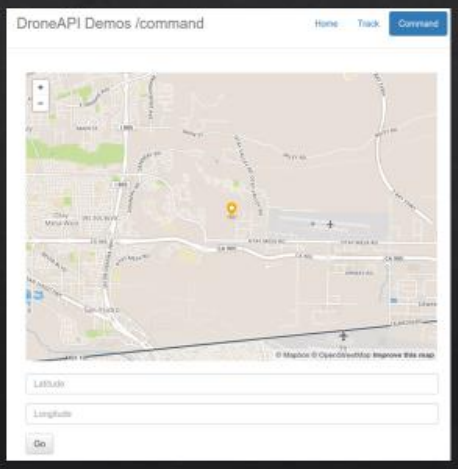
Система контролю та автоматизації роботи транспортного дрона на прикладі DJI M600 PRO

Інтерфейс користувача для контролю роботи дрона



Інтерфейс користувача для контролю роботи дрона





Застосовані технології

Основний стек технологій, що були застосовані в роботі:

- ООП на базі CherryPy Python, реалізовано програмний інтерфейс користувача та ядро зв'язку мережної інфраструктури
- Big Data
- Фреймворк Django для побудови системи аналізу та моніторингу логістичної складової
- Основні системні стратегії з організації IOT рішень
- Пропрієтарні рішення компанії Teltonika на базі проходження практики.

Висновок

У даній роботі було спроектовано, досліджено та задокументовано систему критично важливої мережевої інфраструктури для дистанційного керування логістичною складовою в транспортній компанії з використанням технологій IoT. Запропоновано актуальні рішення щодо проектування даних систем з використанням сучасної техніки від закордонних постачальників комунікаційного обладнання. Написано та протестовано інтерфейс користувача для програмування маршрутів дрона. Таким чином, був створений комплекс системи контролю логістики на дорозі та в повітрі, що показали себе як успішне рішення в різних сферах застосування транспортних засобів й логістики. Пристрої Teltonika пропонують розширені функції програмного забезпечення, такі як функції автоматизації та можливості віддаленого керування, які допомагають заощадити час і, в даному випадку, підвищують безпеку рішення.

Додаток Б

Код програми для 4 Розділу 1 підрозділу:

```
from dronekit import connect, VehicleMode,LocationGlobalRelative,APIException
import time
import socket
import exceptions
import math

## Функція запуску пропелерів та виходу на цільову висоту (в метрах)


def arm_and_takeoff(targetHeight):

    while vehicle.is_armable!=True:
        print("Очікування можливості зльоту.")
        time.sleep(1)
    print("Апарат може злетіти ")

    vehicle.mode = VehicleMode("Керування ")

    while vehicle.mode!=' Керування':
        print("Очікування переходу дрону в режим Керування ")
        time.sleep(1)
    print("Дрон у статусі Керування! Насолоджуйтесь!!")

    vehicle.armed = True
    while vehicle.armed==False:
        print("Очікування можливості зльоту.")
        time.sleep(1)
    print("Обережно! Пропелери у русі!!")

    vehicle.simple_takeoff(targetHeight)

    while True:
        print("Current Altitude: %d"%vehicle.location.global_relative_frame.alt)
        if vehicle.location.global_relative_frame.alt>=.95*targetHeight:
            break
```

```
            time.sleep(1)
        print("Цільової висоти досягнуто!!")

        return None

####sim_vehicle.py opens up port on localhost:14550


vehicle = connect('127.0.0.1:14550',wait_ready=True)


#### Поставте дрон на зліт  і злітайте в повітря на 5 метрів


arm_and_takeoff(5)
print("Дрон зайняв цільову висоту ")


#### Як тільки дрон досягне цільової висоти, зміниться режим на Посадка

vehicle.mode=VehicleMode('Посадка')
while vehicle.mode!='Посадка':
        print("Очікування режиму Посадка ")
        time.sleep(1)
print("Дрон перейшов у статус Посадка. Скоро приземлиться.")
```

Далі був написаний код для базового функціоналу користувача з функціями пошуку, завдання координат та висоти польоту дрону:

```
from __future__ import print_function

import os

import simplejson

import time


from dronekit import connect, VehicleMode, LocationGlobal, LocationGlobalRelative
```

```
import cherrypy

from jinja2 import Environment, FileSystemLoader


# Налаштуйте розбір параметрів, щоб отримати рядок підключення

import argparse

parser = argparse.ArgumentParser(description='Створює веб-програму на базі CherryPy, яка відображає карту карти, щоб ви могли переглядати поточне положення автомобіля та надсилати команди транспортного засобу для польоту до певної широти та довготи. Запуститься та підключитися до SITL, якщо не вказано рядок підключення.')

parser.add_argument('--connect',

                    help="цільовий рядок з'єднання транспортного засобу. Якщо не вказано, SITL запускається і використовується автоматично.")

args = parser.parse_args()


connection_string = args.connect


# Запустіть SITL, якщо не вказано рядок підключення

if not connection_string:

    import dronekit_sitl

    sitl = dronekit_sitl.start_default()

    connection_string = sitl.connection_string()


local_path = os.path.dirname(os.path.abspath(__file__))
```

```
print("local path: %s" % local_path)



cherrypy_conf = {
    '/': {
        'tools.sessions.on': True,
        'tools.staticdir.root': local_path
    },
    '/static': {
        'tools.staticdir.on': True,
        'tools.staticdir.dir': './html/assets'
    }
}



class Drone(object):
    def __init__(self, server_enabled=True):
        self.gps_lock = False
        self.altitude = 30.0

        # Підключений до дрона
        self._log('Підключений до дрона')
```

```
        self.vehicle = vehicle
        self.commands = self.vehicle.commands
        self.current_coords = []
        self.webserver_enabled = server_enabled
        self._log("Початок доставки дрона")

        # Зареєструвати спостерігачів
        self.vehicle.add_attribute_listener('location', self.location_callback)

    def launch(self):
        self._log("Очікування розташування...")
        while self.vehicle.location.global_frame.lat == 0:
            time.sleep(0.1)
        self.home_coords = [self.vehicle.location.global_frame.lat,
                            self.vehicle.location.global_frame.lon]

        self._log("Очікується можливість озброїтися...")
        while not self.vehicle.is_armable:
            time.sleep(.1)

        self._log('Запуск початкової послідовності завантаження')
        self.change_mode('КЕРУВАННЯ')
```

```
    self.arm()
    self.takeoff()

    if self.webserver_enabled is True:
        self._run_server()

def takeoff(self):
    self._log("Зліт")
    self.vehicle.simple_takeoff(30.0)

def arm(self, value=True):
    if value:
        self._log('дрон чекає на політ...')
        self.vehicle.armed = True
        while not self.vehicle.armed:
            time.sleep(.1)
    else:
        self._log("Посадка!")
        self.vehicle.armed = False

def _run_server(self):
    # Запустити веб-сервер, якщо ввімкнено
```

```
        cherrypy.tree.mount(DroneDelivery(self), '/', config=cherrypy_conf)


        cherrypy.config.update({'server.socket_port': 8080,
                                'server.socket_host': '0.0.0.0',
                                'log.screen': None})


        print("'Сервер прив'язаний до всіх адрес, порт 8080
```

Ви можете підключитися до нього за допомогою веб-браузера, використовуючи URL-адресу, яка виглядає так:

```
http://localhost:8080/

''')
        cherrypy.engine.start()


    def change_mode(self, mode):
        self._log("Changing to mode: {0}".format(mode))
module 'collections' has no attribute 'MutableMapping'
        self.vehicle.mode = VehicleMode(mode)
        while self.vehicle.mode.name != mode:
            self._log('  ... polled mode: {0}'.format(mode))
            time.sleep(1)


    def goto(self, location, relative=None):
        self._log("Goto: {0}, {1}".format(location, self.altitude))
```

```
    if relative:
        self.vehicle.simple_goto(
            LocationGlobalRelative(
                float(location[0]), float(location[1]),
                float(self.altitude)
            )
        )
    else:
        self.vehicle.simple_goto(
            LocationGlobal(
                float(location[0]), float(location[1]),
                float(self.altitude)
            )
        )
    self.vehicle.flush()

def get_location(self):
    return [self.current_location.lat, self.current_location.lon]

def location_callback(self, vehicle, name, location):
    if location.global_relative_frame.alt is not None:
```

```
            self.altitude = location.global_relative_frame.alt


        self.current_location = location.global_relative_frame


    def _log(self, message):
        print("[DEBUG]: {0}".format(message))



class Templates:
    def __init__(self, home_coords):
        self.home_coords = home_coords
        self.options = self.get_options()
        self.environment = Environment(loader=FileSystemLoader(local_path + '/html'))


    def get_options(self):
        return {'width': 670,
                'height': 470,
                'zoom': 13,
                'format': 'png',
                'access_token':
'pk.eyJ1Ijoia2V2aW4zZHIiLCJhIjoiY2lrOGoxN2s2MDJzYnR6a3drbTYwdGxmMiJ
9.bv5u7QgmcJd6dZfLDGoykw',
                'mapid': 'kevin3dr.n56ffjoo',
```

```
            'home_coords': self.home_coords,
            'menu': [{'name': 'Home', 'location': '/'},
                    {'name': 'Track', 'location': '/track'},
                    {'name': 'Command', 'location': '/command'}],
            'current_url': '/',
            'json': ''
            }

    def index(self):
        self.options = self.get_options()
        self.options['current_url'] = '/'
        return self.get_template('index')

    def track(self, current_coords):
        self.options = self.get_options()
        self.options['current_url'] = '/track'
        self.options['current_coords'] = current_coords
        self.options['json'] = simplejson.dumps(self.options)
        return self.get_template('track')

    def command(self, current_coords):
        self.options = self.get_options()
```

```
        self.options['current_url'] = '/command'
        self.options['current_coords'] = current_coords
        return self.get_template('command')

    def get_template(self, file_name):
        template = self.environment.get_template(file_name + '.html')
        return template.render(options=self.options)


class DroneDelivery(object):
    def __init__(self, drone):
        self.drone = drone
        self.templates = Templates(self.drone.home_coords)

    @cherrypy.expose
    def index(self):
        return self.templates.index()

    @cherrypy.expose
    def command(self):
        return self.templates.command(self.drone.get_location())
```

```
    @cherrypy.expose
    @cherrypy.tools.json_out()
    def vehicle(self):
        return dict(position=self.drone.get_location())

    @cherrypy.expose
    def track(self, lat=None, lon=None):
        # Обробити запит POST від Command
        # Надсилання пакета MAVLink з інструкціями переходу
        if(lat is not None and lon is not None):
            self.drone.goto([lat, lon], True)

        return self.templates.track(self.drone.get_location())

# Підключення до дрона
print('Підключення до дрона встановлено: %s' % connection_string)
vehicle = connect(connection_string, wait_ready=True)

print('Запуск дрона...')
Drone().launch()

print('Чекаємо додаток cherrypy ...')
```

```
cherrypy.engine.block()


if not args.connect:
    # Вимкнути симулятор, якщо він був запущений
    sitl.stop()
```

Ми починаємо запускати веб-сервер, викликаючи cherrypy.engine.start().

CherryPy - це дуже маленький і простий веб-сервер що був використаний.

Додаток В

Код програми для 4 Розділу 2 підрозділу

```
from __future__ import print_function


from dronekit import connect, VehicleMode, LocationGlobalRelative, LocationGlobal, Command

import time

import math

from pymavlink import mavutil



# Налаштуйте розбір параметрів, щоб отримати рядок підключення

import argparse

parser = argparse.ArgumentParser(description=Демонструє основні операції.')

parser.add_argument('--connect',

                    help=" цільовий рядок з'єднання транспортного засобу. Якщо не вказано, SITL автоматично запускається та використовується.")

args = parser.parse_args()


connection_string = args.connect

sitl = None
```

```
# Запустіть SITL, якщо не вказано рядок підключення
if not connection_string:
    import dronekit_sitl
    sitl = dronekit_sitl.start_default()
    connection_string = sitl.connection_string()


# Підключитися до транспортного засобу
print(Підключилися до транспортного засобу: %s' % connection_string)
vehicle = connect(connection_string, wait_ready=True)


def get_location_metres(original_location, dNorth, dEast):
    """
    Повертає об’єкт LocationGlobal, що містить метри широти/довготи `dNorth` та `dEast` від
    вказано "початкове_розташування". Повернуте місце розташування має те саме значення `alt`
    як "початкове_розташування".

    Ця функція корисна, коли ви хочете переміщати транспортний засіб у певних місцях щодо
    поточне положення автомобіля.
```

Алгоритм відносно точний на невеликих відстанях (10 м у межах 1 км), за винятком близькості до полюсів.

Для отримання додаткової інформації див.:

http://gis.stackexchange.com/questions/2951/algorithm-for-offsetting-a-latitude-longitude-by-some-amount-of-meters

"""

earth_radius=6378137.0 #Radius of "spherical" earth

Зміщення координат у радіанах

dLat = dNorth/earth_radius

dLon = dEast/(earth_radius*math.cos(math.pi*original_location.lat/180))

#New position in decimal degrees

newlat = original_location.lat + (dLat * 180/math.pi)

newlon = original_location.lon + (dLon * 180/math.pi)

return LocationGlobal(newlat, newlon,original_location.alt)

def get_distance_metres(aLocation1, aLocation2):

"""

Повертає відстань від землі в метрах між двома об'єктами LocationGlobal.

Цей метод є наближеним і не буде точним на великих відстанях і близько до

полюси землі. Він походить із тестового коду ArduPilot: https://github.com/diydrones/ardupilot/blob/master/Tools/autotest/common.py

```
    """
    dlat = aLocation2.lat - aLocation1.lat
    dlong = aLocation2.lon - aLocation1.lon
    return math.sqrt((dlat*dlat) + (dlong*dlong)) * 1.113195e5


def distance_to_current_waypoint():
    """
    Отримує відстань у метрах до поточної маршрутної точки.
     Він повертає None для першої точки маршруту (Home location).
    """
    nextwaypoint = vehicle.commands.next
    if nextwaypoint==0:
        return None
    missionitem=vehicle.commands[nextwaypoint-1] # команди мають нульовий індекс
    lat = missionitem.x
    lon = missionitem.y
    alt = missionitem.z
    targetWaypointLocation = LocationGlobalRelative(lat,lon,alt)
    distancetopoint = get_distance_metres(vehicle.location.global_frame, targetWaypointLocation)
```

```
    return distancetopoint


def download_mission():
    """
Завантажте поточну місію з дрона.
    """
    cmds = vehicle.commands
    cmds.download()
    cmds.wait_ready() # wait until download is complete.


def adds_square_mission(aLocation, aSize):
    """
    Додає команду зльоту та чотири команди точки маршруту до поточної місії.
    Маршрутні точки розташовані так, щоб утворити квадрат із стороною довжиною 2*aSize навколо вказаного LocationGlobal (aLocation).

    Функція передбачає, що vehicle.commands відповідає стану місії дрона
    (ви повинні завантажити принаймні один раз під час сесії та після завершення місії)    """
```

```
cmds = vehicle.commands


print(" Очистіть усі існуючі команди ")

cmds.clear()


print(" Визначте/додайте нові команди.")

# Додайте нові команди. Значення/порядок параметрів задокументовано в класі Command.


#Додати команду MAV_CMD_NAV_TAKEOFF. Це ігнорується, якщо транспортний засіб уже в повітрі.

cmds.add(Command( 0, 0, 0, mavutil.mavlink.MAV_FRAME_GLOBAL_RELATIVE_ALT, mavutil.mavlink.MAV_CMD_NAV_TAKEOFF, 0, 0, 0, 0, 0, 0, 0, 0, 10))


# Визначте чотири розташування MAV_CMD_NAV_WAYPOINT та додайте команди

point1 = get_location_metres(aLocation, aSize, -aSize)

point2 = get_location_metres(aLocation, aSize, aSize)

point3 = get_location_metres(aLocation, -aSize, aSize)

point4 = get_location_metres(aLocation, -aSize, -aSize)

cmds.add(Command( 0, 0, 0, mavutil.mavlink.MAV_FRAME_GLOBAL_RELATIVE_ALT, mavutil.mavlink.MAV_CMD_NAV_WAYPOINT, 0, 0, 0, 0, 0, 0, point1.lat, point1.lon, 11))
```

```
    cmds.add(Command( 0, 0, 0, mavutil.mavlink.MAV_FRAME_GLOBAL_RELATIVE_ALT, mavutil.mavlink.MAV_CMD_NAV_WAYPOINT, 0, 0, 0, 0, 0, 0, point2.lat, point2.lon, 12))

    cmds.add(Command( 0, 0, 0, mavutil.mavlink.MAV_FRAME_GLOBAL_RELATIVE_ALT, mavutil.mavlink.MAV_CMD_NAV_WAYPOINT, 0, 0, 0, 0, 0, 0, point3.lat, point3.lon, 13))

    cmds.add(Command( 0, 0, 0, mavutil.mavlink.MAV_FRAME_GLOBAL_RELATIVE_ALT, mavutil.mavlink.MAV_CMD_NAV_WAYPOINT, 0, 0, 0, 0, 0, 0, point4.lat, point4.lon, 14))

    # додати фіктивну маршрутну точку "5" в точку 4 (повідомляє нам, коли ми досягли пункту призначення)

    cmds.add(Command( 0, 0, 0, mavutil.mavlink.MAV_FRAME_GLOBAL_RELATIVE_ALT, mavutil.mavlink.MAV_CMD_NAV_WAYPOINT, 0, 0, 0, 0, 0, 0, point4.lat, point4.lon, 14))



    print(" Завантажте нові команди в дрон")

    cmds.upload()




def arm_and_takeoff(aTargetAltitude):

    """

    Встановити політ дрону до aTargetAltitude.

    """
```

```
print("Basic pre-arm checks")

# Не дозволяйте користувачеві спробувати поставити на взльот, доки автопілот не буде готовий

while not vehicle.is_armable:

    print(" Waiting for vehicle to initialise...")

    time.sleep(1)



print("Arming motors")

# Copter should arm in GUIDED mode

vehicle.mode = VehicleMode("GUIDED")

vehicle.armed = True


while not vehicle.armed:

    print(" Waiting for arming...")

    time.sleep(1)


print("Taking off!")

vehicle.simple_takeoff(aTargetAltitude) # Take off to target altitude


# Зачекайте, поки транспортний засіб досягне безпечної висоти, перш ніж обробляти goto (інакше команда

 # після Vehicle.simple_takeoff буде виконано негайно).
```

```
    while True:
        print(" Висота: ", vehicle.location.global_relative_frame.alt)
        if  vehicle.location.global_relative_frame.alt>=aTargetAltitude*0.95:  #Trigger just below target alt.
            print("Досягнув цільової висоти ")
            break
        time.sleep(1)



print('Створити нову місію (для поточного місцезнаходження)')
adds_square_mission(vehicle.location.global_frame,50)



# З Copter 3.3 ви зможете злетіти за допомогою предмета місії. Літак повинен злетіти, використовуючи предмет місії (наразі).
arm_and_takeoff(10)

print("Початкова місія ")
# Скиньте місію на першу (0) точку шляху
vehicle.commands.next=0

# Встановіть режим AUTO, щоб почати місію
vehicle.mode = VehicleMode("AUTO")
```

```
# Місія моніторингу.

# Демонструє отримання та встановлення номера команди

# Використовує distance_to_current_waypoint(), зручну функцію для пошуку

# відстань до наступної маршрутної точки.

while True:

    nextwaypoint=vehicle.commands.next

    print('Відстань до маршрутної точки (%s): %s' % (nextwaypoint, distance_to_current_waypoint()))



    if nextwaypoint==3: # Перейти до наступної точки маршруту

        print('Перейти до маршрутної точки 5, коли досягнете маршрутної точки 3')

        vehicle.commands.next = 5

    if nextwaypoint==5: # Фіктивна точка маршруту - як тільки ми досягаємо точки 4, це правда, і ми виходимо.

        print("Вийдіть із «стандартної» місії, коли вирушаєте до кінцевої точки шляху (5)")

        break;

    time.sleep(1)



print("Повернутися до запуску")

vehicle.mode = VehicleMode("RTL")
```

```
# Закрийте дрон перед виходом із сценарію
print("Закрити дрон ")
vehicle.close()

# Вимкніть симулятор, якщо він був запущений.
if sitl is not None:
    sitl.stop()
```

Додаток Г

Код програми для 4 Розділу 3 підрозділу:

```
from __future__ import print_function


from dronekit import connect, VehicleMode, LocationGlobal, LocationGlobalRelative

from pymavlink import mavutil # Необхідний для визначення командних повідомлень

import time

import math



# Налаштуйте розбір параметрів, щоб отримати рядок підключення

import argparse

parser = argparse.ArgumentParser(description='Керуйте вертольотом і надсилайте команди в режимі Керування')

parser.add_argument('--connect',

                    help=" Цільовий рядок з’єднання транспортного засобу. Якщо не вказано, SITL автоматично запускається та використовується.")

args = parser.parse_args()

connection_string = args.connect

sitl = None



# Запустіть SITL, якщо не вказано рядок підключення
```

```
if not connection_string:
    import dronekit_sitl
    sitl = dronekit_sitl.start_default()
    connection_string = sitl.connection_string()


# Підключення
print(' Підключення до дрона ввімкнено: %s' % connection_string)
vehicle = connect(connection_string, wait_ready=True)


def arm_and_takeoff(aTargetAltitude):
    """
    Злітає дроном і летить до цільової висоти.
    """

    print("Основні перевірки перед польотом ")
    # Не дозволяйте користувачеві літати, поки автопілот не буде готовий
    while not vehicle.is_armable:
        print(" Очікування ініціалізації дрона...")
        time.sleep(1)
```

```
print("Запуск двигунів ")
# Copter should arm in GUIDED mode
vehicle.mode = VehicleMode("Керування")
vehicle.armed = True

while not vehicle.armed:
    print(" Waiting for arming...")
    time.sleep(1)

print("Taking off!")
vehicle.simple_takeoff(aTargetAltitude) # Take off to target altitude

    # Wait until the vehicle reaches a safe height before processing the goto (otherwise the command
    #  after Vehicle.simple_takeoff will execute immediately).
    while True:
        print(" Altitude: ", vehicle.location.global_relative_frame.alt)
        if  vehicle.location.global_relative_frame.alt>=aTargetAltitude*0.95:  #Trigger just below target alt.
            print("Reached target altitude")
            break
        time.sleep(1)
```

#Arm and take of to altitude of 5 meters

arm_and_takeoff(5)

"""

Convenience functions for sending immediate/guided mode commands to control the Copter.

The set of commands demonstrated here include:

* MAV_CMD_CONDITION_YAW - set direction of the front of the Copter (latitude, longitude)

* MAV_CMD_DO_SET_ROI - set direction where the camera gimbal is aimed (latitude, longitude, altitude)

* MAV_CMD_DO_CHANGE_SPEED - set target speed in metres/second.

The full set of available commands are listed here:

http://dev.ardupilot.com/wiki/copter-commands-in-guided-mode/

"""

def condition_yaw(heading, relative=False):

"""

Надішліть повідомлення MAV_CMD_CONDITION_YAW, щоб вказати транспортний засіб за вказаним курсом (у градусах).

Цей метод встановлює абсолютний заголовок за замовчуванням, але ви можете встановити параметр "відносний".

на "True", щоб встановити відхилення відносно поточного заголовка.

За замовчуванням поворот автомобіля буде відповідати напрямку руху. Після встановлення

рискання, використовуючи цю функцію, не має способу повернутися до рискання за замовчуванням у напрямку слідування

подорожей». (https://github.com/diydrones/ardupilot/issues/2427)

Для отримання додаткової інформації див.:

http://copter.ardupilot.com/wiki/common-mavlink-mission-command-messages-mav_cmd/#mav_cmd_condition_yaw

"""

```
if relative:
    is_relative = 1 #yaw relative to direction of travel
else:
    is_relative = 0 #yaw is an absolute angle
# create the CONDITION_YAW command using command_long_encode()
msg = vehicle.message_factory.command_long_encode(
```

```
        0, 0,    # target system, target component
        mavutil.mavlink.MAV_CMD_CONDITION_YAW, #command
        0, #confirmation
        heading,    # param 1, yaw in degrees
        0,          # param 2, yaw speed deg/s
        1,          # param 3, direction -1 ccw, 1 cw
        is_relative, # param 4, relative offset 1, absolute angle 0
        0, 0, 0)    # param 5 ~ 7 not used
    # відправити команду транспортному засобу
    vehicle.send_mavlink(msg)


def set_roi(location):
    """
Надішліть повідомлення MAV_CMD_DO_SET_ROI, щоб спрямувати кардан камери
    зазначений регіон інтересу (LocationGlobal).
    Автомобіль також може повернутись обличчям до ROI.

    Для отримання додаткової інформації див.: http://copter.ardupilot.com/common-mavlink-mission-command-messages-mav_cmd/#mav_cmd_do_set_roi
    """
    # створіть команду MAV_CMD_DO_SET_ROI
```

```
    msg = vehicle.message_factory.command_long_encode(
        0, 0,    # цільова система, цільовий компонент
        mavutil.mavlink.MAV_CMD_DO_SET_ROI, #command
        0, # підтвердження
        0, 0, 0, 0, #params 1-4
        location.lat,
        location.lon,
        location.alt
        )
    # команда
    vehicle.send_mavlink(msg)
```

"""

Функції для полегшення перетворення між різними системами відліку. Зокрема ці

полегшує навігацію в термінах «метрів від поточної позиції» під час використання команд, які беруть

абсолютні позиції в десяткових градусах.

Ці методи є лише наближеними і можуть бути менш точними на більших відстанях і на близькій відстані

до полюсів Землі.

Зокрема, він забезпечує:

* get_location_metres – отримати LocationGlobal (десяткові градуси) на відстані (м) на північ та схід від даного LocationGlobal.

* get_distance_metres - отримати відстань між двома об'єктами LocationGlobal в метрах

* get_bearing - отримати пеленг у градусах до LocationGlobal"""

def get_location_metres(original_location, dNorth, dEast):

"""

Повертає об’єкт LocationGlobal, що містить метри широти/довготи `dNorth` та `dEast` від

вказано "початкове_розташування". Повернене LocationGlobal має те саме значення `alt`

як "початкове_розташування".

Ця функція корисна, коли ви хочете переміщати транспортний засіб у певних місцях щодо

поточне положення автомобіля.

Алгоритм є відносно точним на невеликих відстанях (10 м у межах 1 км), за винятком близькості до полюсів.

Для отримання додаткової інформації див.: http://gis.stackexchange.com/questions/2951/algorithm-for-offsetting-a-latitude-longitude-by-some-amount-of-meters

"""

```
    earth_radius = 6378137.0 # Радіус «кулястої» землі
    # Зміщення координат у радіанах
    dLat = dNorth/earth_radius
    dLon = dEast/(earth_radius*math.cos(math.pi*original_location.lat/180))

    # Нова позиція в десяткових градусах
    newlat = original_location.lat + (dLat * 180/math.pi)
    newlon = original_location.lon + (dLon * 180/math.pi)
    if type(original_location) is LocationGlobal:
        targetlocation=LocationGlobal(newlat, newlon,original_location.alt)
    elif type(original_location) is LocationGlobalRelative:
        targetlocation=LocationGlobalRelative(newlat, newlon,original_location.alt)
    else:
        raise Exception("Invalid Location object passed")

    return targetlocation;


def get_distance_metres(aLocation1, aLocation2):
    """
```

Цей метод є наближеним і не буде точним на великих відстанях і поблизу полюсів Землі. Він походить із тестового коду ArduPilot:

https://github.com/diydrones/ardupilot/blob/master/Tools/autotest/common.py

```
    """
    dlat = aLocation2.lat - aLocation1.lat
    dlong = aLocation2.lon - aLocation1.lon
    return math.sqrt((dlat*dlat) + (dlong*dlong)) * 1.113195e5


def get_bearing(aLocation1, aLocation2):
    """
```

Повертає пеленг між двома об’єктами LocationGlobal, переданими як параметри.

Цей метод є наближеним і може бути неточним на великих відстанях і поблизу полюсів Землі. Він походить із тестового коду ArduPilot: https://github.com/diydrones/ardupilot/blob/master/Tools/autotest/common.py

```
    """
    off_x = aLocation2.lon - aLocation1.lon
    off_y = aLocation2.lat - aLocation1.lat
    bearing = 90.00 + math.atan2(-off_y, off_x) * 57.2957795
    if bearing < 0:
        bearing += 360.00
    return bearing;
```

"""

Функції для переміщення транспортного засобу в задане положення (на відміну від керування рухом шляхом встановлення компонентів швидкості).

Методи включають:

* goto_position_target_global_int - встановлює позицію за допомогою команди SET_POSITION_TARGET_GLOBAL_INT у

Кадр MAV_FRAME_GLOBAL_RELATIVE_ALT_INT

* goto_position_target_local_ned - встановлює позицію за допомогою команди SET_POSITION_TARGET_LOCAL_NED у

Кадр MAV_FRAME_BODY_NED

* goto - зручна функція, яка може використовувати Vehicle.simple_goto (за замовчуванням) або

goto_position_target_global_int для переходу до певної позиції в метрах

На північ та схід від поточного розташування.

Цей метод повідомляє про відстань до пункту призначення."""

def goto_position_target_global_int(aLocation):

"""

Надішліть команду SET_POSITION_TARGET_GLOBAL_INT, щоб надіслати запит на переліт транспортного засобу до вказаного LocationGlobal.

Для отримання додаткової інформації див.:

https://pixhawk.ethz.ch/mavlink/#SET_POSITION_TARGET_GLOBAL_INT

Перегляньте наведене вище посилання для отримання інформації про type_mask (0=ввімкнути, 1=ігнорувати).

Під час написання прискорення та рискання ігноруються. """

```
    msg = vehicle.message_factory.set_position_target_global_int_encode(
        0,       # time_boot_ms (not used)
        0, 0,    # target system, target component
        mavutil.mavlink.MAV_FRAME_GLOBAL_RELATIVE_ALT_INT, # frame
        0b0000111111111000, # type_mask (only speeds enabled)
        aLocation.lat*1e7, # lat_int - X Position in WGS84 frame in 1e7 * meters
        aLocation.lon*1e7, # lon_int - Y Position in WGS84 frame in 1e7 * meters
        aLocation.alt, # alt - Altitude in meters in AMSL altitude, not WGS84 if absolute or relative, above terrain if GLOBAL_TERRAIN_ALT_INT
        0, # X velocity in NED frame in m/s
        0, # Y velocity in NED frame in m/s
        0, # Z velocity in NED frame in m/s
        0, 0, 0, # afx, afy, afz acceleration (not supported yet, ignored in GCS_Mavlink)
        0, 0)    # yaw, yaw_rate (not supported yet, ignored in GCS_Mavlink)
    # send command to vehicle
    vehicle.send_mavlink(msg)
```

```
def goto_position_target_local_ned(north, east, down):
```

"""

Надіслати команду SET_POSITION_TARGET_LOCAL_NED для запиту перельоту транспортного засобу до вказаного

розташування на Півночі, Сході, Внизу кадру.

Важливо пам’ятати, що в цьому кадрі додатні висоти вводяться як від’ємні значення «Вниз». Отже, якщо вниз дорівнює "10", це буде на 10 метрів нижче домашньої висоти.

Починаючи з AC3.3, метод поважає налаштування кадру. До цього кадр ігнорували. Для отримання додаткової інформації див.:

http://dev.ardupilot.com/wiki/copter-commands-in-guided-mode/#set_position_target_local_ned

Перегляньте наведене вище посилання для отримання інформації про type_mask (0=ввімкнути, 1=ігнорувати). Під час написання прискорення та рискання ігноруються.

"""

msg = vehicle.message_factory.set_position_target_local_ned_encode(

0, # time_boot_ms (not used)

0, 0, # target system, target component

mavutil.mavlink.MAV_FRAME_LOCAL_NED, # frame

0b0000111111111000, # type_mask (only positions enabled)

north, east, down, # x, y, z positions (or North, East, Down in the MAV_FRAME_BODY_NED frame

0, 0, 0, # x, y, z velocity in m/s (not used)

```
        0, 0, 0, # x, y, z acceleration (not supported yet, ignored in GCS_Mavlink)
        0, 0)    # yaw, yaw_rate (not supported yet, ignored in GCS_Mavlink)
    # send command to vehicle
    vehicle.send_mavlink(msg)
```

```
def goto(dNorth, dEast, gotoFunction=vehicle.simple_goto):
    """
```

Переміщує транспортний засіб у положення d на північ від поточного положення.

Метод приймає аргумент вказівника функції з одним параметром `dronekit.lib.LocationGlobal` для

цільове положення. Це дозволяє викликати його за допомогою різних команд налаштування позиції.

За замовчуванням він використовує стандартний метод: dronekit.lib.Vehicle.simple_goto().

Метод повідомляє про відстань до цілі кожні дві секунди. """

```
    currentLocation = vehicle.location.global_relative_frame
    targetLocation = get_location_metres(currentLocation, dNorth, dEast)
    targetDistance = get_distance_metres(currentLocation, targetLocation)
```

```
    gotoFunction(targetLocation)


    #print "DEBUG: targetLocation: %s" % targetLocation

    #print "DEBUG: targetLocation: %s" % targetDistance


    while vehicle.mode.name=="GUIDED": #Stop action if we are no longer in guided mode.

        #print "DEBUG: mode: %s" % vehicle.mode.name

        remainingDistance=get_distance_metres(vehicle.location.global_relative_frame, targetLocation)

        print("Distance to target: ", remainingDistance)

        if remainingDistance<=targetDistance*0.01: #Just below target, in case of undershoot.

            print("Reached target")

            break;

        time.sleep(2)
```

"""

Функції, які переміщують транспортний засіб, вказуючи компоненти швидкості в кожному напрямку.

Дві функції використовують різні команди MAVLink. Основна відмінність полягає в тому

що в залежності від використовуваної рами швидкість NED може бути відносно транспортного засобу

орієнтація.

Методи включають:

* send_ned_velocity - Встановлює компоненти швидкості за допомогою команди SET_POSITION_TARGET_LOCAL_NED

* send_global_velocity - Встановлює компоненти швидкості за допомогою команди SET_POSITION_TARGET_GLOBAL_INT

"""

def send_ned_velocity(velocity_x, velocity_y, velocity_z, тривалість): """

Переміщення транспортного засобу в напрямку на основі заданих векторів швидкості та

на зазначену тривалість.

Для цього використовується команда SET_POSITION_TARGET_LOCAL_NED з маскою типу, що дозволяє лише

компоненти швидкості

(http://dev.ardupilot.com/wiki/copter-commands-in-guided-mode/#set_position_target_local_ned).

Зауважте, що з AC3.3 повідомлення слід повторно надсилати кожну секунду (приблизно через 3 секунди

без повідомлення швидкість впаде до нуля). У AC3.2.1 і раніше зазначено

швидкість зберігається до тих пір, поки вона не буде скасована. Наведений нижче код повинен працювати на будь-якій версії

(відправлення повідомлення кілька разів не викликає проблем).

Перегляньте наведене вище посилання для отримання інформації про type_mask (0=ввімкнути, 1=ігнорувати).

Під час написання прискорення та рискання ігноруються. """

```
msg = vehicle.message_factory.set_position_target_local_ned_encode(
    0,       # time_boot_ms (not used)
    0, 0,    # target system, target component
    mavutil.mavlink.MAV_FRAME_LOCAL_NED, # frame
    0b0000111111000111, # type_mask (only speeds enabled)
    0, 0, 0, # x, y, z positions (not used)
    velocity_x, velocity_y, velocity_z, # x, y, z velocity in m/s
    0, 0, 0, # x, y, z acceleration (not supported yet, ignored in GCS_Mavlink)
    0, 0)    # yaw, yaw_rate (not supported yet, ignored in GCS_Mavlink)

# send command to vehicle on 1 Hz cycle
for x in range(0,duration):
    vehicle.send_mavlink(msg)
    time.sleep(1)
```

def send_global_velocity(velocity_x, velocity_y, velocity_z, duration):

"""

Переміщення транспортного засобу в напрямку на основі заданих векторів швидкості.

Для цього використовується команда SET_POSITION_TARGET_GLOBAL_INT лише з увімкненою маскою типу

компоненти швидкості

(http://dev.ardupilot.com/wiki/copter-commands-in-guided-mode/#set_position_target_global_int).

Зауважте, що з AC3.3 повідомлення слід повторно надсилати кожну секунду (приблизно через 3 секунди

без повідомлення швидкість впаде до нуля). У AC3.2.1 і раніше зазначено

швидкість зберігається до тих пір, поки вона не буде скасована. Наведений нижче код повинен працювати на будь-якій версії

(відправлення повідомлення кілька разів не викликає проблем).

Перегляньте наведене вище посилання для отримання інформації про type_mask (0=ввімкнути, 1=ігнорувати).

Під час написання прискорення та рискання ігноруються. """

msg = vehicle.message_factory.set_position_target_global_int_encode(

0, # time_boot_ms (not used)

```
        0, 0,    # target system, target component
        mavutil.mavlink.MAV_FRAME_GLOBAL_RELATIVE_ALT_INT, # frame
        0b0000111111000111, # type_mask (only speeds enabled)
        0, # lat_int - X Position in WGS84 frame in 1e7 * meters
        0, # lon_int - Y Position in WGS84 frame in 1e7 * meters
        0, # alt - Altitude in meters in AMSL altitude(not WGS84 if absolute or relative)
        # altitude above terrain if GLOBAL_TERRAIN_ALT_INT
        velocity_x, # X velocity in NED frame in m/s
        velocity_y, # Y velocity in NED frame in m/s
        velocity_z, # Z velocity in NED frame in m/s
        0, 0, 0, # afx, afy, afz acceleration (not supported yet, ignored in GCS_Mavlink)
        0, 0)    # yaw, yaw_rate (not supported yet, ignored in GCS_Mavlink)

    # send command to vehicle on 1 Hz cycle
    for x in range(0,duration):
        vehicle.send_mavlink(msg)
        time.sleep(1)

"""
Fly a triangular path using the standard Vehicle.simple_goto() method.
```

```
The method is called indirectly via a custom "goto" that allows the target position to
be

specified as a distance in metres (North/East) from the current position, and which
reports

the distance-to-target.

"""

print("TRIANGLE path using standard Vehicle.simple_goto()")


print("Set groundspeed to 5m/s.")

vehicle.groundspeed=5


print("Position North 80 West 50")

goto(80, -50)


print("Position North 0 East 100")

goto(0, 100)


print("Position North -80 West 50")

goto(-80, -50)
```

"""

Проведіть трикутний шлях за допомогою команди SET_POSITION_TARGET_GLOBAL_INT і вказавши

цільове положення (а не керувати рухом за допомогою векторів швидкості). Команда є

викликається з goto_position_target_global_int() (через `goto`).

Метод goto_position_target_global_int викликається опосередковано зі спеціального "goto", який дозволяє

цільове положення, яке вказується як відстань у метрах (північ/схід) від поточної позиції,

і який повідомляє про відстань до цілі.

Код також встановлює швидкість (MAV_CMD_DO_CHANGE_SPEED). У AC3.2.1 Copter розганятиметься до цієї швидкості

поблизу центру своєї подорожі, а потім сповільнюється, досягаючи цілі.

У AC3.3 швидкість змінюється відразу.

"""

print("TRIANGLE шлях із використанням стандартного повідомлення SET_POSITION_TARGET_GLOBAL_INT і з різною швидкістю.")

print("Position South 100 West 130")

print("Set groundspeed to 5m/s.")

vehicle.groundspeed = 5

goto(-100, -130, goto_position_target_global_int)

```
print("Set groundspeed to 15m/s (max).")

vehicle.groundspeed = 15

print("Position South 0 East 200")

goto(0, 260, goto_position_target_global_int)


print("Set airspeed to 10m/s (max).")

vehicle.airspeed = 10


print("Position North 100 West 130")

goto(100, -130, goto_position_target_global_int)



"""

Проведіть транспортним засобом по квадратному шляху 50 м, використовуючи команду SET_POSITION_TARGET_LOCAL_NED

і визначення цільового положення (замість керування рухом за допомогою векторів швидкості).

Команда викликається з goto_position_target_local_ned() (через `goto`).


Позиція вказується в термінах NED (північний схід вниз) щодо домашнього розташування.
```

ПОПЕРЕДЖЕННЯ: «D» у NED означає «Вниз». Використання додатного значення D призведе до вбивання автомобіля в землю!

Код перебуває в режимі сну на певний час (DURATION), щоб дати транспортному засобу час досягти кожної позиції (а не

надсилання команд на основі близькості).

Код також встановлює область інтересу (MAV_CMD_DO_SET_ROI) за допомогою методу `set_roi()`. Це вказує на

підвіс камери у вибраному місці (у цьому випадку він вирівнює весь автомобіль, щоб вказати на ROI)."""

print("Шлях SQUARE із використанням SET_POSITION_TARGET_LOCAL_NED і параметрів положення")

DURATION = 20 #Set duration for each segment.

print("Північ 50 м, схід 0 м, висота 10 м протягом %s секунд " % DURATION)

goto_position_target_local_ned(50,0,-10)

print("ROI у поточному місці (початкове положення)")

Зверніть увагу, що це потрібно викликати після команди goto як першу команду «переміщення» певного типу

команди "скидання" ROI/YAW

set_roi(vehicle.location.global_relative_frame)

time.sleep(DURATION)

```
print("Північ 50 м, схід 50 м, висота 10 м ")

goto_position_target_local_ned(50,50,-10)

time.sleep(DURATION)


print("ROI у поточному місці ")

set_roi(vehicle.location.global_relative_frame)


print("Північ 0 м, схід 50 м, висота 10 м ")

goto_position_target_local_ned(0,50,-10)

time.sleep(DURATION)


print("North 0m, East 0m, 10m altitude")

goto_position_target_local_ned(0,0,-10)

time.sleep(DURATION)
```

"""

Політ транспортного засобу по КВАДРАТНОЇ траєкторії, використовуючи вектори швидкості (основний код викликає

Команда SET_POSITION_TARGET_LOCAL_NED з увімкненими параметрами швидкості).

Нитка перебуває в режимі сну на час (DURATION), який визначає відстань, яку буде пройдено.

Код також встановлює відхилення (MAV_CMD_CONDITION_YAW) за допомогою методу set_yaw() у кожному сегменті

так, щоб передня частина автомобіля була спрямована в напрямку руху"""

#Set up velocity vector to map to each direction.

vx > 0 => fly North

vx < 0 => fly South

NORTH = 2

SOUTH = -2

Note for vy:

vy > 0 => fly East

vy < 0 => fly West

EAST = 2

WEST = -2

Note for vz:

vz < 0 => ascend

vz > 0 => descend

```
UP = -0.5

DOWN = 0.5



# Квадратний шлях із використанням швидкості

print("SQUARE path using SET_POSITION_TARGET_LOCAL_NED and velocity parameters")


print("Поривання 180 абсолютний (південь)")

condition_yaw(180)


print("Швидкість на південь і вгору ")

send_ned_velocity(SOUTH,0,UP,DURATION)

send_ned_velocity(0,0,0,1)



print("Поривання 270 абсолютний (захід)")

condition_yaw(270)


print("Velocity West & down")

send_ned_velocity(0,WEST,DOWN,DURATION)

send_ned_velocity(0,0,0,1)
```

```
print("Поворот 0 абсолютний (північ))")

condition_yaw(0)


print("Швидкість на північ ")

send_ned_velocity(NORTH,0,0,DURATION)

send_ned_velocity(0,0,0,1)



print("Поривання 90 абсолютний (схід)")

condition_yaw(90)


print("Швидкість Схід ")

send_ned_velocity(0,EAST,0,DURATION)

send_ned_velocity(0,0,0,1)



"""
```

Політ транспортним засобом по трасі DIAMOND, використовуючи вектори швидкості (основний код викликає the

Команда SET_POSITION_TARGET_GLOBAL_INT з увімкненими параметрами швидкості).

Нитка перебуває в режимі сну на час (DURATION), який визначає відстань, яку буде пройдено.

Код встановлює відхилення (MAV_CMD_CONDITION_YAW) за допомогою методу `set_yaw()`, використовуючи відносні заголовки

так, щоб передня частина автомобіля була спрямована в напрямку руху.

В кінці другого сегмента код встановлює нове місце розташування до поточної точки.

"""

print("Шлях DIAMOND із використанням SET_POSITION_TARGET_GLOBAL_INT і параметрів швидкості ")

vx, vy паралельні півночі та сходу (незалежно від орієнтації автомобіля)

print("Yaw 225 absolute")

condition_yaw(225)

print("Швидкість на південь, захід і вгору ")

send_global_velocity(SOUTH,WEST,UP,DURATION)

send_global_velocity(0,0,0,1)

print("Рискання 90 відносно (до попереднього заголовку рискання)")

condition_yaw(90,relative=True)

print("Швидкість на північ, захід і вниз ")

send_global_velocity(NORTH,WEST,DOWN,DURATION)

```
send_global_velocity(0,0,0,1)

print("Встановити поточне місце розташування нового дома ")
vehicle.home_location=vehicle.location.global_frame
print("Отримайте нове домашнє місце розташування ")
# Це перезавантажує домашнє розташування в DroneKit і GCS
cmds = vehicle.commands
cmds.download()
cmds.wait_ready()
print(" Домашнє розташування: %s" % vehicle.home_location)

print("Поривання 90 відносно (до попереднього заголовку рискання)")
condition_yaw(90,relative=True)

print("Швидкість на північ і схід ")
send_global_velocity(NORTH,EAST,0,DURATION)
send_global_velocity(0,0,0,1)


print("Рискання 90 відносно (до попереднього заголовку рискання)")
condition_yaw(90,relative=True)
```

```
print("Швидкість Південь і Схід ")
send_global_velocity(SOUTH,EAST,0,DURATION)
send_global_velocity(0,0,0,1)

"""
Приклад завершується. Посадка на поточному місці."""

print("Встановлення режиму Посадка...")
vehicle.mode = VehicleMode("Посадка")
# Закрийте об’єкт дрона перед виходом зі сценарію
print("Закрийте об’єкт дрона перед виходом зі сценарію ")
vehicle.close()

# Вимкніть симулятор, якщо він був запущений.
if sitl is not None:
    sitl.stop()

print("Завершено ")
```