DOI 10.31891/2307-5732-2024-337-3-33
УДК 004.42:004.32:004.35

ПОСВІСТАК ВАЛЕРІЙ
Київський національний університет технологій та дизайну
https://orcid.org/0009-0001-7785-1378
e-mail: valposv@gmail.com
МІРОШНИЧЕНКО ДМИТРО
Київський національний університет технологій та дизайну
https://orcid.org/0009-0002-0904-1645
e-mail: dmiroshnycheko@gmail.com

АРХІТЕКТУРА СИСТЕМИ АВТОНОМНОГО КЕРУВАННЯ ДЛЯ FPV-ДРОНІВ

Автономне керування дроном пропонує практичне рішення проблем, присутніх при ручному управлінні дроном оператором. Існують обмеження, які виключають можливість наявності оператора, а деякі рішення мають значно нижчу ефективність за відсутності автономного алгоритму. Велика дистанція, радіоперешкоди або великі обсяги даних, що треба аналізувати, не такі критичні, якщо у дрона є можливість приймати рішення автономно без оператора. Через особливості базового аматорського FPV-дрона його польотний контролер не має достатніх обчислювальних спроможностей для виконання складних алгоритмів таких як визначення та відстеження об'єктів, розрахунку правильної траєкторії тощо. У таких випадках необхідне використання допоміжного комп'ютера. Дана робота досліджує важливі нюанси та специфіку інтеграції допоміжних комп'ютерів із польотними контролерами, а також описує апаратну та програмну частини реалізації автономного управління дроном.

Ключові слова: дрон, квадрокоптер, безпілотний літальний апарат (БПЛА), роботизовані системи, автономне управління, польотний контролер, Raspberry Pi, MAVLink, Robot Operating System.

POSVISTAK VALERII, MIROSHNYCHENKO DMYTRO
Kyiv National University of Technologies and Design

ARCHITECTURE OF AUTONOMOUS CONTROL SYSTEM FOR FPV-DRONES

Drone technologies and robotic systems, that can be used to optimize or replace human labor in different areas, are continuing to develop and expand their area of application. Computational hardware, that can be used as an extension to basic drone hardware, is becoming more powerful for a smaller price and size. Autonomous drone control provides a practical solution for issues that are present when a drone is controlled manually by an operator. There are limitations that exclude the possibility of operator control and some solutions are much less effective if there is no autonomous algorithm. A large distance, radio interference or a huge amount of data to be analyzed are not as critical if a drone is capable to make decisions autonomously without an operator. Because of the features of a basic amateur FPV-drone its flight controller does not operate enough computing power to execute complex algorithms such as object detection, object tracking, calculation of a proper trajectory etc. In cases like this a companion computer should be used. Depending on an area of application, drone control algorithms may vary. Flight controllers' firmware has some embedded functionality with relatively simple algorithms like pre-defined flight routes that use GCS-coordinates. In the case when more complex behavior is required (image processing, AI integration or autonomous decision making with manual controls), such firmware provides an ability to accept various drone commands that correspond to the MAVLink communication protocol. This work investigates important details of companion computers integration with flight controllers with examples and describes hardware and software aspects of implementation of autonomous drone controls. The research also provides solutions to problems that can be possible when implementing such a system.

Keywords: drone, quadcopter, unmanned aerial vehicle (UAV), robotic systems, autonomous controls, flight controller, Raspberry Pi, MAVLink, Robot Operating System.

Постановка проблеми у загальному вигляді та її зв'язок із важливими науковими чи практичними завданнями

Із розвитком дронових технологій також розширюються сфери їх застосування. Мікроконтролери та комп'ютери стають більш доступними та продуктивними, що відкриває можливості розробки пристроїв для специфічних задач. Дрони (також відомі як безпілотні літальні апарати) можуть використовуватись у агропромисловому комплексі [1], для виявлення лісових пожеж [2], для охоронних [3] або військових задач, спостереження або розвідки, доставки їжі або медикаментів [4] тощо. У випадку повністю ручного керування дроном присутній ряд недоліків. Потреба в операторах значно здорожує процес з точки зору потенційної ефективності та передбачає певний відсоток некоректної роботи через помилки, зумовлені людським фактором. Одним із поширених способів керування БПЛА є аналоговий або цифровий радіозв'язок, що обмежений певною дистанцією та вразливий до перешкод. Залежно від сфери застосування, критичність цих факторів відрізняється, від незначного впливу до унеможливлення проведення тієї самої роботи операторами через відсутність зв'язку або масштаб роботи (наприклад збір великого обсягу даних). Реалізація автономного алгоритму управління дроном вирішує вищезазначені проблеми, підвищуючи ефективність та мінімізуючи потребу в операторах.

Аналіз досліджень та публікацій

Зріст кількості досліджень про дронові технології та можливість впровадження алгоритмів автономного управління дроном за останні роки підтверджує актуальність даної теми. Дослідники фокусуються на різних підходах до реалізації. Використання дронів є можливим у різноманітних сферах

заради оптимізації або заміни людської роботи [1-7]. Незважаючи на різні сфери застосування дронів, більшість із них містять корисну інформацію, що може бути застосована для інших цілей.

В деяких дослідженнях розглядається комбінація обробки зображень з камери, даних навігаційних модулів та різних датчиків для коректної орієнтації у просторі [3, 4]. Присутні дослідження про апаратну інтеграцію мікрокомп'ютерів Raspberry Pi та обробку зображень із використанням комп'ютерного зору та машинного навчання [6, 7]. Додатковою областю дослідження є використання функціоналу прошивок польотних контролерів, таких як ArduPilot чи BetaFlight [8], а також розгляд різних режимів польоту, корисних для реалізації алгоритму автономного польоту [7]. Окремі роботи фокусуються на дослідженні принципів роботи, а також вразливостей протоколу MAVLink [9, 10]. Концепція програмного управління дроном з допомогою кватерніонів є перспективною при реалізації алгоритмів автономного управління [11, 12].

Окремо варто підкреслити вклад Ніколаса Рема [13] та Калеба Бергквіста [14] у дослідження розробки роботизованих систем із використанням різних алгоритмів та компонентів. Також у вітчизняній та міжнародній бібліотечній галузі наявні корисні публікації О. Мясіщева та інших [5, 8].

Узагальнюючи, тенденція розвитку досліджень у даній сфері зберігає темпи зростання. Присутня велика кількість документації про окремі апаратні та програмні компоненти (прошивки ArduPilot та Betaflight, SITL тестування, протокол MAVLink, UART та I2C протоколи, польотний контролер SpeedyBee F405 V3, мікрокомп'ютер Raspberry Pi 5 8GB) [15-19], а також публікації про концепти реалізації систем автономного управління дроном для різних цілей. Розгляд деталей та проблем при інтеграції комп'ютера Raspberry Pi 5 8GB із бюджетним польотним контролером, а також реалізації програмного керування дроном був би корисним для української бібліотечної галузі.

Формулювання цілей статті

Метою даної роботи є дослідження шляхів реалізації автономної системи для квадрокоптерного FPV-дрона, а також розгляд деяких проблем, із якими ми зіштовхнулись при розробці прототипу подібної системи на базі польотного контролера SpeedyBee F405 V3 та мікрокомп'ютера Raspberry Pi 5 Board 8GB. У роботі будуть розглянуті особливості апаратної та програмної частин автономної системи, переваги та недоліки прошивок польотних контролерів, протоколи апаратної та програмної комунікації, а також деякі MAVLink команди для автономного керування дроном.

Виклад основного матеріалу

Базовий FPV-дрон складається із рами, польотного стеку (польотний контролер та регулятор обертів), радіоприймача для керування, камери, відеопередавача, двигунів із пропелерами та акумуляторної батареї. Окрім стандартних компонентів можуть бути додаткові — звуковий сигнал (buzzer), різноманітні сенсори, такі як гіроскоп, акселерометр, LIDAR, GPS-модуль, компас (у випадку якщо вони не вбудовані у польотний контролер). Серед усіх базових компонентів нас цікавить польотний контролер. Він обробляє отримані команди маніпуляції із дроном, стабілізує їх та надсилає на регулятор обертів, який у свою чергу безпосередньо керує моторами для досягнення бажаного результату, наприклад нахилу вперед на n градусів. Класичне ручне управління FPV-дроном відбувається з допомогою радіоконтролера із двома стіками, кожен із яких має горизонтальну та вертикальні вісі. Відносну позицію 4 вісей польотний контролер отримує завдяки радіоприймачу відповідної частоти, та корегує поведінку моторів, надсилаючи сигнал на регулятор обертів. Окрім стіків на контролері може бути довільний набір тумблерів або кнопок, що можуть бути налаштовані для виконання певного функціоналу — запуску двигунів, зміни режиму польоту тощо. Загалом можна виділити 4 можливі маніпуляції із літальним апаратом: газ (throttle), рискання (yaw), крен (roll) і тангаж (pitch) — рис. 1.

Рис. 3. Можливі маніпуляції при управлінні літальним апаратом

Для виключення компоненти із ручним управлінням необхідно було б реалізувати певний алгоритм поведінки, що міг би виконуватись на дроні автономно. Польотні контролери зазвичай мають доволі обмежені обчислювальні спроможності та малий обсяг оперативної та флеш-пам'яті. На момент написання даної роботи оптимальним варіантом для придбання є польотні контролери із мікроконтролерами типів F4 чи F7. Контролер типу F4 (STM32F405) може бути наділений процесором до 180 МГц, флеш-пам'яттю до 1536 кБ та оперативною пам'яттю до 320 кБ [20]. Ці обмеження унеможливлюють реалізацію складних алгоритмів, що вимагають інтенсивних розрахунків чи більшого обсягу пам'яті. Варто зазначити, що прошивки (firmware) польотних контролерів, такі як ArduPilot, Betaflight, PX4, iNAV, можуть мати функціонал автономного керування, наприклад політ по заданих GCS-координатах, при наявності навігаційного модуля на дроні [8]. Для реалізації більш складних алгоритмів із обробкою зображень, штучним інтелектом тощо необхідна наявність потужнішого комп'ютера на дроні. Деякі виробники дронів створюють потужні польотні контролери у вигляді єдиного компонента, проте у цій роботі буде розглянута можливість інтеграції окремих компонентів при самостійній збірці дрона. Незважаючи на обмежені обчислювальні характеристики, польотні контролери є чудовою базою для розширення функціоналу — їх можна використовувати як основу для під'єднання датчиків, приймачів та передавачів, навігаційних модулів та інших периферійних пристроїв, у тому числі допоміжних мікроконтролерів чи комп'ютерів (companion computers), таких як Raspberry Pi, Arduino Giga, Nvidia Jetson TX2 та інших, що підтримують апаратні та програмні протоколи комунікації, сумісні із конкретними польотними контролерами.

Вибір прошивки (firmware) польотного контролера залежить від підтримки конкретного обладнання та від поставлених цілей. Кожна технологія має свої переваги та недоліки. Betaflight широко використовується ентузіастами, які беруть участь у перегонах FPV-дронів або літають у якості хоббі. При цьому Betaflight у першу чергу фокусується саме на ручному керуванні дроном, тому містить великий набір відповідних налаштувань та функцій. ArduPilot теж може використовуватись для ручного керування, проте також містить широкий набір просунутих алгоритмів для автономного керування дроном. Для взаємодії із дронами, що мають прошивку ArduPilot, можливе використання програмного забезпечення наземних станцій Mission Planner та QGroundControl. Воно дозволяє конфігурувати польотні місії, налаштовувати поведінку автопілота тощо. Варто зазначити, що через малий обсяг флеш-пам'яті на деяких польотних контролерах, прошивка ArduPilot може містити лише частковий функціонал заради оптимізації розміру кінцевих бінарних файлів прошивки [15]. Як Betaflight так і ArduPilot мають відкриту кодову базу та велику спільноту користувачів, але у кінцевому підсумку для розробки системи автономного керування дроном варто віддати перевагу ArduPilot через наявність корисного функціоналу.

Режими польоту. На момент написання даної роботи ArduPilot містить 25 вбудованих режимів польоту для коптерів, 10 з яких використовуються найчастіше [16]. Залежно від режиму, змінюється поведінка дрона, деякі з них активують стабілізацію чи повністю переходять у режим автопілота тощо. Наприклад, режим ACRO (від слова акробатика) представляє собою повністю ручне керування без стабілізації, цей режим найчастіше використовується операторами, коли необхідний тонкий контроль поведінки дрона. GUIDED (керований) добре підходить для інтеграції із допоміжним комп'ютером або наземною станцією, оскільки у цьому режимі підтримуються MAVLink команди для руху дрона (крен, тангаж, швидкість у певному напрямку, цільові GCS-координати тощо) — деякі будуть розглянуті у даній роботі. LAND ініціює приземлення. STABILIZE автоматично стабілізує крен і тангаж при ручному керуванні. Повний опис конкретних режимів польоту доступний у документації ArduPilot [16]. В інших роботах виділені режими польоту, що можуть бути корисними для автономного керування [7].

Протоколи апаратної комунікації. Комунікація допоміжного комп'ютера із польотним контролером може відбуватись із допомогою інтерфейсів UART, I2C, SPI. I2C представляє собою master-slave (controller-target) комунікацію, у якій до master-пристроя можна послідовно під'єднати декілька slave пристроїв. I2C протокол передбачає 2 шини: SCL (serial clock) для ініціації синхросигналів та SDA (serial data) для передачі даних. Згідно з документацією, ArduPilot підтримує комунікацію one master – many slaves, де slaves — набір датчиків, компас тощо [17]. На момент написання даної роботи можливість використання польотного контролеру у якості slave-вузла потребує дослідження та потенційної модифікації кодової бази ArduPilot. UART представляє собою one master – one slave протокол, що передбачає канал передачі даних із двома лініями — Tx-Rx та Rx-Tx, де Tx — джерело, а Rx — отримувач. Польотні контролери зазвичай мають декілька портів UART, що можуть використовуватись для радіоприймачів, відеопередавачів, камери та інших пристроїв, включаючи допоміжний мікрокомп'ютер. На Рис. 4 зображено коректне підключення UART-портів мікрокомп'ютера Raspberry Pi 5 та SpeedyBee F405 V3.

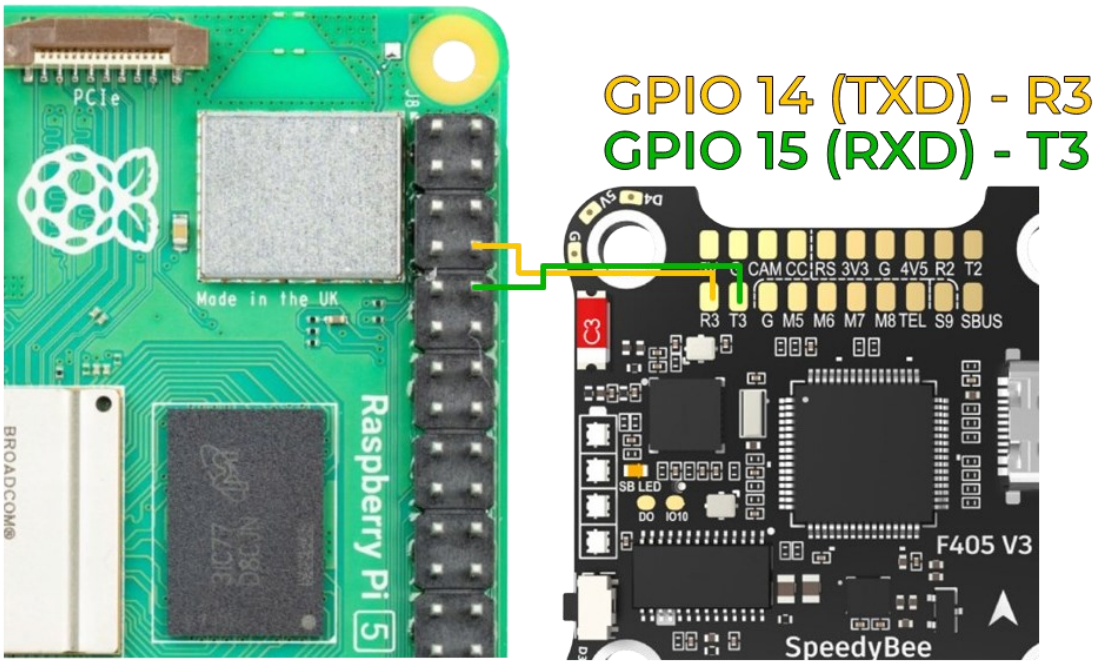

Рис. 4. Коректне підключення Raspberry Pi 5 до польотного контролера SpeedyBee F405 V3

Для живлення Raspberry Pi можливо використовувати відповідні піни на платі, проте лінія 5В на польотному контролері SpeedyBee F405 V3 має силу струму до 2А, в той час як для живлення Raspberry Pi 5 необхідно мати 5А. Для вирішення проблеми можливе підключення джерела живлення (акумуляторної батареї тощо) до USB Type-C порту Raspberry Pi. На FPV-дронах зазвичай використовується акумуляторна батарея типу LiPo або Li-ion, що підключається до силових кабелів польотного контролера через роз'єм XT60. Для живлення Raspberry Pi необхідно використовути XT-60 розгалужувач та відповідний конвертер напруги та сили струму — 5В та 5А. За замовчуванням UART-порти на Raspberry Pi 5 вимкнені. Для їх активації необхідно використовувати команду raspi-config, що запускає графічний інтерфейс, де можна змінити необхідні налаштування. Необхідно обрати опцію 5 — Interfacing Options, а потім опцію P6 — Serial. Після цього на спливаючому повідомленні буде можливість активувати серійний порт UART — рис. 3.

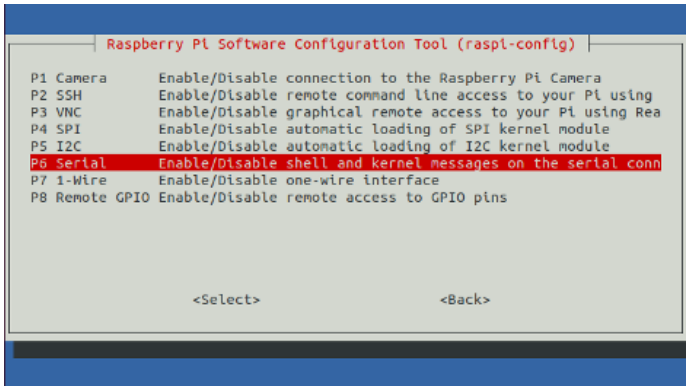

Рис. 5. Налаштування UART у інтерфейсі команди raspi-config

MAVLink повідомлення. ArduPilot підтримує MAVLink (Micro Air Vehicle Link) — відкритий протокол зв'язку, що використовується як для обміну даними між дроном та наземною станцією, так і між компонентами дрона. Наземна станція чи мікрокомп'ютер надсилає команди та керує дроном, в той час як польотний контролер може надавати дані телеметрії. Бібліотека MAVLink реалізує кодування та декодування пакетів згідно із протоколом, проте не регламентує якими засобами вони мають бути передані. У випадку обміну повідомленнями між компонентами дрона він може відбуватись через серійні порти UART, проте при наявності відповідних засобів можлива комунікація через бездротоий зв'язок Bluetooth, інтернет тощо. Структура повідомлення MAVLink складається із символа початку повідомлення (STX), довжини повідомлення (LEN), лічильника повідомлення, що використовується для виявлення втрати повідомлень (SEQ), ідентифікатора системи (SYS) та компонента (COMP) відправника, типу повідомлення MSG, тіла повідомлення PAYLOAD та контрольної суми пакету (CKA та CKB) [9]. На Рис. 6 зображена повна структура повідомлення MAVLink v1.

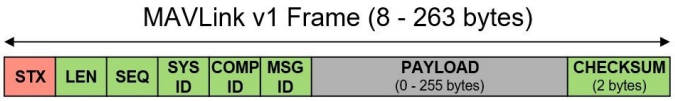

Рис. 6. Структура повідомлення MAVLink v1

MAVLink v1 мав ряд безпекових вразливостей, серед яких надсилання пакетів стороннім суб'єктом [10], тому у MAVLink v2, що має зворотню сумісність із першою версією, було додано можливість підпису (SIGNATURE) повідомлень — див. Рис. 7, а також інші покращення.

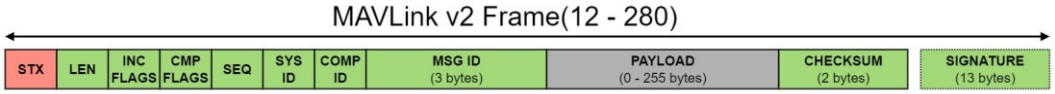

Рис. 7. Структура повідомлення MAVLink v2

Існує ряд консольних інструментів або бібліотек для різних мов програмування, що реалізують протокол MAVLink і дозволяють полегшити з ним роботу: mavlink-router, MAVProxy, mavp2p, DroneKit, pymavlink та інші. Як зазначено вище, частиною структури повідомлення є ідентифікатор його типу у числовому вигляді. MAVLink v2 повідомлення мають ID > 255, тому для них зарезервовано 3 байти замість 1. Залежно від типу повідомлення, набір параметрів у його тілі (payload) може відрізнятися. Повна інформація наявна в документації бібліотеки протоколу MAVLink [21]. Розглянемо декілька корисних типів MAVLink команд.

SET_ATITUDE_TARGET (MSG_ID = 82) дозволяє задавати рівень газу (від 0 до 1) та кватерніон цільової точки повороту [11, 12]. Протокол MAVLink також підтримує параметри значень body_roll_rate, body_pitch_rate та body_yaw_rate у радіанах на секунду, для крену, тангажу та рискання відповідно, проте вони не підтримуються на прошивці ArduPilot. На Рис. 8 зображено надсилання даної команди з допомогою бібліотеки DroneKit на мові Python.

```python
def set_attitude_target(roll_angle=0.0, pitch_angle=0.0,
                        yaw_angle=None,
                        thrust=0.5):
    if yaw_angle is None:
        yaw_angle = vehicle.attitude.yaw

    msg = vehicle.message_factory.set_attitude_target_encode(
        0,  # time_boot_ms
        1,  # Target system id
        1,  # Target component id
        0b00000111,
        to_quaternion(roll_angle, pitch_angle, yaw_angle),  # Attitude quaternion
        0,  # Body roll rate, not supported by ArduPilot
        0,  # Body pitch rate, not supported by ArduPilot
        0,  # Body yaw rate, not supported by ArduPilot
        thrust  # Thrust, from 0 to 1, where 0.5 is hovering
    )
    vehicle.send_mavlink(msg)
```

Рис. 8. Використання команди SET_ATTITUDE_TARGET з допомогою бібліотеки DroneKit

На Рис. 9 зображена функція to_quaternion, що відповідає за конвертацію значень кутів вісей крену, тангажу та рискання у кватерніон [11, 12].

```python
def to_quaternion(roll=0.0, pitch=0.0, yaw=0.0):
    t0 = math.cos(math.radians(yaw * 0.5))
    t1 = math.sin(math.radians(yaw * 0.5))
    t2 = math.cos(math.radians(roll * 0.5))
    t3 = math.sin(math.radians(roll * 0.5))
    t4 = math.cos(math.radians(pitch * 0.5))
    t5 = math.sin(math.radians(pitch * 0.5))

    w = t0 * t2 * t4 + t1 * t3 * t5
    x = t0 * t3 * t4 - t1 * t2 * t5
    y = t0 * t2 * t5 + t1 * t3 * t4
    z = t1 * t2 * t4 - t0 * t3 * t5

    return [w, x, y, z]
```

Рис. 9. Конвертація значення кутів крену, тангажу та рискання у кватерніон

ArduPilot підтримує команду SET_ATTITUDE_TARGET у режимах польоту Guided та Guided_NoGPS. Вона дозволяє доволі гнучко управляти дроном, якщо необхідна максимальна точність, проте вся складність полягає у реалізації такого алгоритму, який би розраховував коректні кути вісей залежно від задачі. Також для руху дрона Guided режим підтримує команди SET_POSITION_TARGET_LOCAL_NED та SET_POSITION_TARGET_GLOBAL_INT. Обидві команди дозволяють конфігурувати швидкість та прискорення для вісей x (північ-південь), y (схід-захід), z (вниз-вгору) та деякі інші параметри. У якості цільової позиції SET_POSITION_TARGET_GLOBAL_INT дозволяє встановити WGS84-координати (широту та довготу), в той час як SET_ATTITUDE_TARGET підтримує зміщення згідно із розширеним фільтром Калмана (EKF).

MAV_CMD_DO_SET_MODE (MSG_ID = 11) дозволяє змінити режим польоту дрона. З допомогою бібліотеки DroneKit можна змінити значення mode на об'єкті vehicle. На Рис. 10 відображено зміну режиму польоту дрона, а також програмний запуск його моторів.

```
vehicle.mode = VehicleMode("GUIDED_NOGPS")
vehicle.armed = True

while not vehicle.armed:
    vehicle.armed = True
    time.sleep(1)
```

Рис. 10. Зміна режиму польоту та запуск моторів

Тестування та відлагодження програмного коду. Перед тестуванням алгоритму автономної системи на реальному дроні необхідно впевнитись у його коректній поведінці. Для цього можливе використання ArduPilot SITL (Software In The Loop), що дозволяє програмно симулювати поведінку дрона. Таким чином з допомогою MAVProxy або інших інструментів можна тестувати управління дроном без наявності апаратної частини [19]. Для DroneKit проектів є можливість застосування dronekit-sitl. Для наочної трьохвимірної візуалізації польоту дрона існують симулятори Gazebo [22], AirSim та інші.

Robot Operating System (ROS) — фреймворк, що пропонує широкий спектр інструментів для розробки роботизованих систем. Він сумісний із деякими Linux-дистрибутивами, окремі версії також можна використовувати на ОС Windows. На Raspberry Pi OS, що є рекомендованою ОС для встановлення на Raspberry Pi 5, Robot OS встановлюється з допомогою декількох консольних команд. Основною перевагою ROS є реалізація клієнт-серверної архітектури. Функціонал роботизованої системи розділяється на окремі модулі (nodes), відповідальні за певну поведінку — див. Рис. 11. Для комунікації між компонентами ROS реалізує 3 концепції:

1. **Topic** — дозволяє компонентам обмінюватись даними в реальному часі за принципом відправник-підписник (наприклад потік кадрів із модуля камери на модуль визначення об'єктів, де останній є підписником на відповідний Topic);
2. **Service** — у цьому випадку відправник надсилає запит на сервіс і очікує відповіді. Використовується для операцій, що швидко виконуються або для блокуючого запиту;
3. **Action** — вид сервісу, де операція може займати багато часу та під час її виконання сервіс може надсилати відправнику її статус (наприклад дистанція, яку залишилось пролетіти).

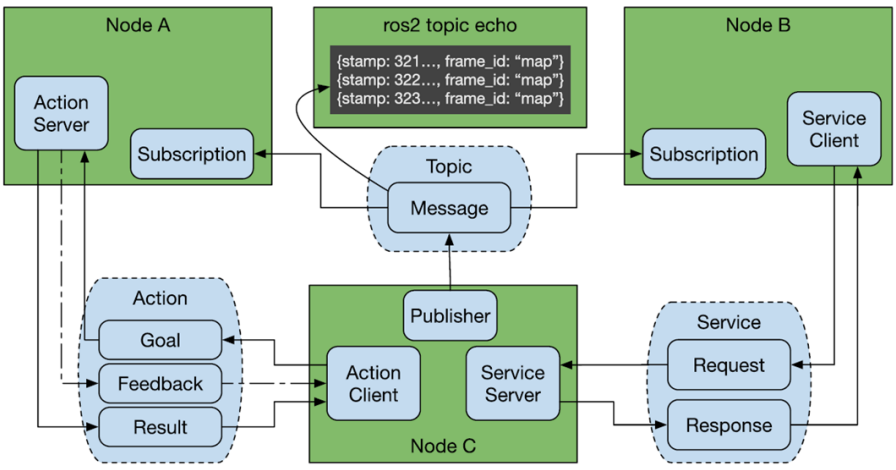

Рис. 11. Архітектура роботизованої системи на Robot OS

Обробка зображень із камери. Зазвичай при збірці FPV-дронів використовуються аналогові камери із відповідними відеопередавачами. У нашому випадку це камера FPV Caddx Ratel 2 Micro 1200TVL 1/1.8" та передавач RushFPV Tank Solo 1.6W. На момент написання даної роботи цифрові камери значно дорожчі та виробляються меншим числом компаній. Більше того, при використанні цифрових камер у класичному аматорському FPV-дроні присутній ряд інших недоліків: більша затримка зображення у деяких випадках, менша максимальна дистанція через обмеження виробників, що є важливою при ручному керуванні дрона. Якщо стоїть задача обробляти зображення на Raspberry Pi, це є проблемою, оскільки на даному комп'ютері немає входу для аналогового відеосигналу формату NTSC або PAL. Також прошивка ArduPilot для SpeedyBee F405 V3 через малий обсяг флеш-пам'яті не містить функцію MAVLink Camera, що дозволяє керувати потоком зображень із камери [7]. Є декілька рішень даної проблеми (табл. 1).

Таблиця 1

Рішення проблеми

№	Рішення	Переваги	Недоліки
1	Встановлення додаткового модуля-камери, сумісного із Raspberry Pi 5	Найменша затримка, найменше втрат у якості (а можливо навіть і краще зображення порівняно з аналоговим)	Вища вартість за опцію №2 (але важливо зазначити, що вона є приблизно однаковою із аналоговою FPV-камерою)
2	Встановлення USB карти відеозахоплення, що конвертує аналоговий сигнал у цифровий, на Raspberry Pi 5	Низька вартість	Можлива затримка, спотворення зображення та вихід пристрою із ладу
3	Встановлення USB приймача відеосигналу 5.8 GHz на Raspberry Pi 5	Залежно від моделі, потенційно більша надійність порівняно із опцією №2	Вразливість до радіоперешкод, можливе спотворення зображення та вихід пристрою із ладу. Вартість схожа із опцією №1

Опція №1 із використанням додаткової камери є доступним і надійним рішенням даної проблеми. Для покращення результатів обробки зображень можливо використовувати офіційний Raspberry Pi 5 модуль-камеру із ширшим кутом огляду.

Висновки

При розробці системи автономного управління дроном на базі Raspberry Pi 5 та SpeedyBee F405 V3 були досліджені та описані аспекти апаратних характеристик аматорських FPV-дронів та основні принципи ручного управління дроном. Розглянуто обмеження стандартних польотних контролерів, можливість їх розширення допоміжними комп'ютерами та коректне під'єднання із використанням інтерфейсу UART. Був проведений аналіз наявних прошивок польотних контролерів, який демонструє переваги використання ArduPilot при розробці систем автономного управління, а також ПЗ наземних станцій Mission Planner і QGroundControl. Описано принцип роботи та структуру повідомлень протоколу програмної комунікації із дроном MAVLink, а також інструменти, створені для спрощення його використання. Наведено приклади програмного керування дроном з допомогою бібліотеки DroneKit, що є обгорткою над MAVLink повідомленнями. Акцентовано увагу на важливості тестування реалізованих алгоритмів використовуючи інструменти SITL у комбінації із графічними симуляторами. Розглянуто переваги фреймворку Robot Operating System при побудові роботизованих систем завдяки клієнт-серверній архітектурі та реалізації міжвузлової комунікації. Досліджено проблему обробки зображення із аналогової FPV-камери через відсутність підтримки аналогового відеосигналу комп'ютером Raspberry Pi 5 та рекомендовано використання додаткової камери.

Розширення функціоналу польотних контролерів шляхом використання мікрокомп'ютерів є перспективною концепцією у сфері дронових технологій. Можливість виконання спеціалізованого програмного коду відкриває шлях для реалізації різноманітних алгоритмів автономного управління дроном. На момент написання даної роботи мікрокомп'ютери вже мають вражаючу обчислювальну спроможність, враховуючи їх розмір та відносно невелику вартість, наприклад Raspberry Pi 5 наділений чотирьохядерним процесором із частотою 2.4 GHz, що у 2-3 рази потужніше ніж його попередник Raspberry Pi 4. Можна очікувати, що тенденція збільшення потужності комп'ютерів буде продовжуватись. Це робить можливим виконання складніших алгоритмів із відстеженням об'єктів, впровадження штучного інтелекту тощо. Також наявні розширювальні плати, що дозволяють використовувати більше видів зв'язку, наприклад LTE, що може бути корисним для певних розробок.

Узагальнюючи, автоматизація управління дроном шляхом використання автономних алгоритмів буде продовжувати впроваджуватись ще у багатьох сферах, дозволяючи підвищувати ефективність промислових процесів, зберігати довкілля, а також використовуватись при надзвичайних ситуаціях.

Література

1. Garre P., Harish A. Autonomous Agricultural Pesticide Spraying UAV. IOP Conference Series: Materials Science and Engineering. 2018. V. 455. P. 012030. DOI: 10.1088/1757-899x/455/1/012030.

2. Grari M., Yandouzi M., Mohammed B., Boukabous M., Idrissi I. Comparative study of teachable machine for forest fire and smoke detection by drone. Bulletin of Electrical Engineering and Informatics. 2024. V. 13, № 3. P. 1970–1979. DOI: 10.11591/eei.v13i3.6578.

3. Hell P. M., Varga P. J. Drone Systems for Factory Security and Surveillance. Interdisciplinary Description of Complex Systems. 2019. V. 17, № 3. P. 458–467. DOI: 10.7906/indecs.17.3.4.

4. Li X., Tupayachi J., Sharmin A., Martinez M. Ferguson. Drone-Aided Delivery Methods, Challenge, and the Future: A Methodological Review. Drones. 2023. V. 7, № 3. P. 191. DOI: 10.3390/drones7030191.

5. Lienkov S., Myasischev A., Ovcharuk V., Lenkov E., Lytvynenko N. Development of Multifunctional Rotary UAV Based on Pixhawk Family Flight Controllers. Advances in Military Technology. 2023. V. 18, № 1. P. 21–34. DOI: 10.3849/aimt.01752.

6. Liu X. et al. Combination of UAV and Raspberry Pi 4B: Airspace detection of red imported fire ant nests using an improved YOLOv4 model. Mathematical Biosciences and Engineering. 2022. V. 19, № 12. P. 13582–13606. DOI: 10.3934/mbe.2022634.

7. Ortega L. D., Loyaga E. S., Cruz P. J., Lema H. P., Abad J., Valencia E. A. Low-Cost Computer-Vision-Based Embedded Systems for UAVs. Robotics. 2023. V. 12, № 6. P. 145. DOI: 10.3390/robotics12060145.

8. Lienkov S., Myasischev A., Banzak O., Husak Y., Starynski I. Use of rescue mode for UAV on the basis of STM32 microcontrollers. International Journal of Advanced Trends in Computer Science and Engineering. 2020. V. 9, № 3. P. 3506–3513. DOI: 10.30534/ijatcse/2020/156932020.

9. Atoev S., Kwon K.-R., Lee S.-H., Moon K.-S. Data analysis of the MAVLink communication protocol. 2017 International Conference on Information Science and Communications Technologies (ICISCT). IEEE, 2017. DOI: 10.1109/icisct.2017.8188563.

10. Kwon Y.-M., Yu J., Cho B.-M., Eun Y., Park K.-J. Empirical Analysis of MAVLink Protocol Vulnerability for Attacking Unmanned Aerial Vehicles. IEEE Access. 2018. V. 6. P. 43203–43212. DOI: 10.1109/access.2018.2863237.

11. Fresk E., Nikolakopoulos G. Full quaternion based attitude control for a quadrotor. 2013 European Control Conference (ECC). IEEE, 2013. DOI: 10.23919/ecc.2013.6669617.

12. Carino J., Abaunza H., Castillo P. Quadrotor quaternion control. 2015 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, 2015. DOI: 10.1109/icuas.2015.7152367.

13. Nicholas Rehm, LinkedIn page. URL: https://www.linkedin.com/in/nicholasrehm/ (accessed May 14, 2024).

14. Caleb Bergquist, LinkedIn page. URL: https://www.linkedin.com/in/caleb-bergquist/ (accessed May 14, 2024).

15. Firmware Limitations on AutoPilot Hardware — Copter documentation. URL: https://ardupilot.org/copter/docs/common-limited-firmware.html (accessed May 14, 2024).

16. Flight Modes — Copter documentation. URL: https://ardupilot.org/copter/docs/flight-modes.html (accessed May 14, 2024).

17. Sensor Drivers — Dev documentation. URL: https://ardupilot.org/dev/docs/code-overview-sensor-drivers.html (accessed May 14, 2024).

18. Communicating with Raspberry Pi via MAVLink — Dev documentation. URL: https://ardupilot.org/dev/docs/raspberry-pi-via-mavlink.html (accessed May 14, 2024).

19. SITL Simulator (Software in the Loop) — Dev documentation. URL: https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html (accessed May 14, 2024).

20. STM32F4 - ARM Cortex-M4 High-Performance MCUs - STMicroelectronics. URL: https://www.st.com/en/microcontrollers-microprocessors/stm32f4-series.html (accessed 14.05.2024).

21. Messages (common) MAVLink Developer Guide. URL: https://mavlink.io/en/messages/common.html (accessed May 14, 2024).

22. Koenig N., Howard A. Design and use paradigms for gazebo, an open-source multi-robot simulator. 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566). IEEE. DOI: 10.1109/iros.2004.1389727.