**УДК 004**
**DOI https://doi.org/10.36994/2788-5518-2025-02-10-13**

# ОГЛЯД СУЧАСНИХ МЕТОДІВ ОБРОБКИ СЕНСОРНОЇ ІНФОРМАЦІЇ ТА ПРОГРАМНИХ АРХІТЕКТУР БОРТОВИХ СИСТЕМ УПРАВЛІННЯ БЕЗПІЛОТНИХ ПОВІТРЯНИХ СУДЕН

***Петросян А. Р.,*** *асистент, Державний університет «Житомирська політехніка», Житомир, Україна. http://orcid.org/0000-0003-0960-8461, xckaytz@gmail.com*

***Анотація.*** *Стаття присвячена огляду сучасних методів обробки сенсорної інформації та програмних архітектур бортових систем управління безпілотних повітряних суден (БПС). Показано, що підвищення автономності БПС залежить від узгодженого застосування алгоритмів обробки даних і принципів інженерії програмного забезпечення за умов обмежених обчислювальних ресурсів, жорстких вимог до детермінізму та надійності. Розглянуто багатошарову структурну організацію програмного забезпечення БПС, включно з драйверним рівнем, операційними системами реального часу, проміжним шаром та прикладними компонентами. Проаналізовано поширені архітектурні патерни – модульність, видавець-передплатник, подійно-орієнтований та потік даних – які забезпечують масштабованість, гнучкість і стійкість роботи автопілотів. Особливу увагу приділено методологіям проєктування V-моделі та Model-Based Design, що дають змогу скоротити цикл розробки і підвищити якість систем за рахунок ранньої верифікації моделей і автоматизації формування коду.*

*У роботі виконано порівняльний аналіз сучасних методів тестування вбудованих систем: модульного, інтеграційного, SIL- та HIL-тестування, що забезпечують комплексну перевірку програмно-апаратних рішень у сценаріях, наближених до реальних умов експлуатації. Розглянуто класичні та інтелектуальні підходи до злиття даних, зокрема фільтри Калмана та їх модифікації, фільтр Mahony, генетично оптимізовані фільтри частинок, а також роль цифрових фільтрів у попередній обробці сигналів. Показано значущість ПІД-регуляторів, лінійно-квадратичних методів та навігаційних алгоритмів.*

*Огляд демонструє, що подальший розвиток БПС пов'язаний із розв'язанням суперечності між зростаючою складністю алгоритмів і ресурсними обмеженнями бортових платформ. Перспективними напрямами визначено гібридні методи фільтрації, оптимізацію обчислювальних процесів, інтерпретовані моделі машинного навчання та вдосконалення інструментів апаратно-програмного тестування.*

***Ключові слова:*** *безпілотні повітряні судна; обробка сенсорних даних; злиття даних; програмна архітектура; SIL/HIL-тестування; алгоритми управління.*

# A REVIEW OF MODERN METHODS FOR SENSOR DATA PROCESSING AND SOFTWARE ARCHITECTURES FOR ONBOARD CONTROL SYSTEMS OF UNMANNED AERIAL VEHICLES

***Arsen Petrosian,*** *assistant, Zhytomyr Polytechnic State University, Zhytomyr, Ukraine. http://orcid.org/0000-0003-0960-8461, xckaytz@gmail.com*

***Abstract.*** *The article is devoted to reviewing modern methods of sensor information processing and software architectures of onboard control systems for unmanned aerial vehicles (UAVs). It is shown that increasing the autonomy of UAVs depends on the coordinated application of data processing algorithms and software engineering principles under conditions of limited computing resources and strict requirements for determinism and reliability. The multi-layered structural organization of UAV software is considered, including the driver level, real-time operating systems, the intermediate layer, and application components. Common architectural patterns – modularity, publisher-subscriber, event-driven, and data flow – are analyzed, which ensure the scalability, flexibility, and stability of autopilots. Particular attention is paid to V-model and Model-Based Design methodologies, which reduce the development cycle and improve system quality through early model verification and code generation automation.*

*The paper provides a comparative analysis of modern methods for testing embedded systems: modular, integration, SIL, and HIL testing, which ensure comprehensive verification of software and hardware solutions in scenarios close to real operating conditions. It examines classical and intelligent approaches to data fusion, in particular Kalman filters and their modifications, the Mahony filter, genetically optimized particle filters, and the role of digital filters in signal preprocessing. The importance of PID controllers,*

*linear-quadratic methods, and navigation algorithms, including computationally complex computer vision methods, is demonstrated.*

*The review demonstrates that the further development of BPS is associated with resolving the contradiction between the growing complexity of algorithms and the resource limitations of onboard platforms. Promising areas include hybrid filtering methods, optimization of computational processes, interpreted machine learning models, and improvement of hardware and software testing tools.*

***Key words:*** *unmanned aerial vehicles; sensor data processing; data fusion; software architecture; SIL/HIL testing; control algorithms.*

**Вступ**

БПС стали критично важливим компонентом для вирішення широкого кола завдань, від іграшкі до моніторингу територій, автономної доставки товарів та інтернету речей [1; 2]. Основним фактором їх ефективності є рівень автономності, який безпосередньо залежить від здатності бортового обчислювального комплексу в реальному часі вирішувати такі завдання, як навігація в незнайомому середовищі, побудова карти місцевості, планування траєкторії, розпізнавання об'єктів тощо. Вирішення подібних завдань базується на інтегрованій обробці даних з різних сенсорів: інерційного вимірювального блока; супутникового приймача; оптичного і лідарного камер тощо.

Основна науково-технічна проблема полягає у суперечності між зростаючою обчислювальною складністю алгоритмів автономності, таких як нелінійна фільтрація, глибоке навчання, оптимізаційне планування, і жорсткими обмеженнями БПС за масою, габаритам, енергоспоживанню, вартості тощо. Ця суперечність посилюється вимогами до високої надійності та роботи в детермінованому реальному часі. В результаті, завдання створення бортового програмного забезпечення (ПЗ) виходить за рамки чистої алгоритміки і вимагає застосування відповідних методів програмної інженерії для забезпечення необхідної якості: продуктивності, надійності, супроводу тощо.

**Мета статті** – провести аналіз сучасних методів і алгоритмів обробки сенсорної інформації в бортових обчислювальних системах БПС. У фокусі аналізу знаходяться програмно-архітектурні аспекти: обчислювальна ефективність, архітектурні паттерни, забезпечення детермінізму та методи верифікації. На основі виявлених обмежень визначаються перспективні напрямки досліджень на стику теорії алгоритмів, апаратного забезпечення та інженерії програмного забезпечення.

**Виконання досліджень**

За класифікацією асоціації UVS International (Unmanned Vehicle Systems), БПС за льотними характеристиками можуть бути тактичні, стратегічні та спеціального призначення [3]. Існують і інші класифікації БПС [1].

Незалежно від того, до якого типу належить БПС, структуру існуючого ПЗ БПС можна умовно розділити на такі шари [4–7]:

1. Шар драйверів пристроїв (Device Driver Layer).
2. Шар операційної системи (OS Layer).
3. Проміжний шар (Middleware Layer).
4. Шар додатка (Application Layer).

Даний підхід відповідає багатошаровому архітектурному патерну. Перший шар використовується для взаємодії з апаратним забезпеченням і дає змогу абстрагування від конкретних апаратних компонентів (датчики, виконавчі механізми тощо), що дозволяє верхнім шарам працювати з ними через єдиний програмний інтерфейс.

Другий шар забезпечує виконання ключових вимог до ПЗ: детермінізм, передбачуваність і надійність. Завдяки такій структурі ПЗ може працювати на різних операційних системах реального часу (ОСРЧ): NuttX, ChibiOS, FreeRTOS або навіть RT-Linux.

Третій шар пов'язує високорівневий шар з низькорівневими шарами ОСРЧ. Кожне ПЗ БПС використовує свій механізм обміну повідомленнями в проміжному шарі.

Верхній, четвертий шар містять модулі, які забезпечують безпосередньо логіку роботи автопілота (зчитує стан датчиків, виконує стабілізацію орієнтації, формування керуючого впливу тощо).

Як бачимо на сучасному етапі розвитку ПЗ для БПС переважає модульна архітектура. Це обумовлено необхідністю підвищення масштабованості, надійності та зручності супроводу ПЗ. Модульний підхід дозволяє розділяти систему на незалежні компоненти з чітко визначеними інтерфейсами, що спрощує інтеграцію нових алгоритмів, наприклад, на основі машинного навчання. Одним з найбільш наочних прикладів такої архітектури є автопілот на основі програмного стека PX4 [8] з міжмодульним обміном через шину uORB. Програмний стек PX4 показано на рис.1.

На противагу модульній архітектурі існує монолітна, яка характерна для більш ранніх поколінь автопілотів. Однак основна перевага такої архітектури – забезпечення високої продуктивності та простоти реалізації, але обмежує можливості масштабування, модифікації та повторного використання коду. Прикладом використання даної архітектури є добре відомий автопілот BetaFlight [9], вся логіка якого об'єднана в єдиний цикл управління без чітких інтерфейсів між компонентами. Основний пріоритет – мінімальні затримки і висока чуйність при FPV-польотах.

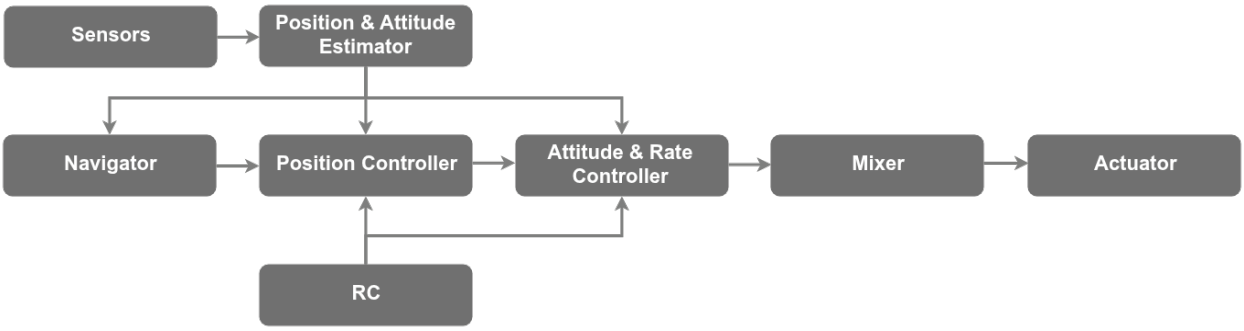


Рис. 1. Структурні елементи програмного стеку PX4 [8]

Поряд із багатошаровим архітектурним патерном, у системах автопілотування БПЛА також широко застосовуються патерни: видавець-передплатник (publisher-subscriber) [8; 10], подійно-орієнтований (event-driven) [11] та поток даних (dataflow) [12]. Їхнє використання є ключовим для забезпечення ефективності, надійності та гнучкості систем автопілотування БПС, що працюють у реальному часі в умовах змінюваного середовища.

Патерн видавець-передплатник ефективно організовує комунікацію між модулями в розподілених архітектурах. Датчики виступають видавцями, які публікують дані в загальну шину, а модулі автопілота підписуються лише на потрібні їм дані. Це зменшує зв'язність між модулями, спрощує інтеграцію нових компонентів та підвищує ефективність шляхом передачі даних лише за їхньої зміни.

Подійно-орієнтований патерн лягає в основу логіки управління польотом, зокрема при перемиканні між режимами – зльоту, посадки, навігації за маршрутними точками чи утримання позиції. Він забезпечує миттєву реакцію системи на зовнішні події: команди оператора, різкі зміни погодних умов чи виникнення аварійних ситуацій.

Патерн потоку даних є фундаментальним для обробки інформації. Він структурує обчислення у вигляді послідовності спеціалізованих блоків, кожен з яких виконує певне завдання, наприклад, фільтрацію сигналів із датчиків або розрахунок керуючих впливів.

Таким чином, сукупність цих патернів дозволяє створювати функціонально повні, гнучкі, масштабовані та надійні системи автопілотування. Вони забезпечують чітке розмежування відповідальностей, спрощують тестування та інтеграцію та дозволяють швидко адаптуватися до нових вимог. Це має вирішальне значення для БПЛА, де будь-яка помилка може призвести до критичних наслідків, а швидкість реакції безпосередньо впливає на успіх місії.

Розвиток БПС висуває підвищені вимоги до якості, надійності та часу розробки їхніх систем, зокрема автопілота. Для відповідності цим викликам все частіше впроваджуються сучасні методології проєктування, серед яких важливу роль відіграють V-модель [13] та Model-Based Design [14; 15].

V-модель – методологія, яка інтегрує процеси проектування та тестування в єдиний цикл. Вона передбачає послідовне виконання етапів проектування з подальшим переходом до відповідних їм етапів тестування. Такий підхід забезпечує виявлення та усунення дефектів на найраніших стадіях.

Model-Based Design (MBD) – методологія, в якій основою розробки є модель системи. Вона переносить фокус з ручного написання коду на створення та ітеративне вдосконалення виконавчої моделі системи. Вона дозволяє: візуалізувати алгоритми управління та логіку роботи системи; автоматично генерувати код з моделей, наприклад, за допомогою MATLAB/Simulink; проводити тестування на ранніх етапах розробки.

Поєднання V-моделі та Model-Based Design дозволяє не лише скоротити час проектування, але і принципово підвищити надійність системи управління БПС.

Тестування вбудованих систем є ключовим етапом життєвого циклу розробки, оскільки помилки в логіці управління можуть призвести до критичних відмов та аварійних ситуацій. Як уже згадувалося, це обумовлено унікальним поєднанням програмних і апаратних компонентів, жорсткими обмеженнями реального часу, ресурсами пам'яті та енергії, а також вимогами до надійності. На відміну від тестування ПЗ для комп'ютерів, тестування вбудованих систем вимагає комплексного підходу, який враховує взаємодію з фізичним світом. Застосовуються такі методи: модульне та інтеграційне тестування [16]; Software-in-the-Loop (SIL) [17; 18] та Hardware-in-the-Loop (HIL) [13; 17; 19].

Модульне тестування застосовується для перевірки функціональної коректності окремих компонентів системи управління БПС, таких як функції, класи або модулі. Основне завдання даного виду тестування полягає в забезпеченні правильності алгоритмічної реалізації вихідного коду. Проведення модульного тестування дозволяє локалізувати і усунути дефекти до інтеграції компонентів в систему, що істотно знижує трудомісткість наступних етапів налагодження і підвищує загальну надійність системи.

Інтеграційне тестування, в свою чергу, являє собою наступний рівень контролю якості, спрямований на перевірку коректності взаємодії між компонентами та оцінку їх спільного функціонування в складі цілісної системи. Основна увага приділяється аналізу інтерфейсних взаємодій та узгодженості обміну даними між ними. На даному етапі виявляються дефекти, що виникають внаслідок несу-

місності інтерфейсів, порушень протоколів взаємодії, некоректної обробки виняткових ситуацій або конфліктів при зверненні до спільних ресурсів.

Модульне та інтеграційне тестування є необхідною, але недостатньою умовою для забезпечення надійності вбудованих систем. Ці методи працюють у відриві від цільової апаратної платформи і не враховують таких критичних факторів: обробка переривань, продуктивність у реальному часі, детермінованість реакції на зовнішні події тощо. SIL- та HIL-тестування якраз і використовуються для подолання обмежень класичних методів, забезпечуючи перевірку системи в умовах, максимально наближених до реальних.

SIL-тестування призначене для виконання розробленого програмного коду не на цільовому мікроконтролері, а на хостовому комп'ютері. Такий підхід дозволяє проводити масштабне і високошвидкісне тестування алгоритмів в умовах, наближених до реального часу, без необхідності доступу до фізичного обладнання. Мета SIL-тестування полягає в підтвердженні функціональної еквівалентності розробленого коду його модельному прототипу, а також у виявленні помилок, внесених на даному етапі розробки або що проявляються при тривалому виконанні програми. Слід зазначити, що SIL-тестування абстрагується від особливостей взаємодії з периферійними пристроями цільового мікроконтролера.

У статті [18] демонструється концепція "Forerunner UAV" (БПС, який оснащений камерою та летить попереду підрозділів екстреної допомоги, що рухаються вперед, щоб підвищити обізнаність водіїв про ситуацію на дорозі за допомогою повітряного огляду дорожньої ситуації та повідомлень про небезпеку, що насувається). Автори розглядають програмно-апаратний комплекс для SIL-тестування. Її завдання – тестування алгоритмів стеження за наземним транспортом (з урахуванням раптових змін маршруту) на основі нейромереж YOLO і генерація даних для навчання нейромереж. Це дозволяє проаналізувати роботу алгоритмів без використання реальної відеокамери, яка повинна знаходиться на борту БПС.

HIL-тестування спрямовано на виявлення та усунення проблем, які не виявляються на попередніх етапах моделювання та SIL-тестування. В рамках HIL-стенду реальний апаратний пристрій інтегрується в контрольоване віртуальне середовище, що моделює роботу інших компонентів системи. Такий підхід дозволяє піддавати тестований об'єкт впливу широкого діапазону сценаріїв, включаючи штатні, граничні та аварійні режими, відтворення яких в реальних умовах ускладнено або пов'язане з ризиком. HIL-тестування забезпечує перевірку не тільки програмної логіки, але і коректність функціонування системи в цілому, включаючи взаємодію програмних і апаратних компонентів.

Стаття [19] присвячена розробці та впровадженню платформи для апаратного HIL-тестування в реальному часі для поліпшення моделювання польоту та стабілізації БПС. Автори представляють процес проектування та виготовлення спеціалізованого 3-вимірного тестового стенду з гіроскопічною підвіскою (рис.2). Цей стенд дозволяє тестувати і налаштовувати параметри ПІД-контролерів для управління кутовими рухами (крен, тангаж, рискання) в реальному часі, використовуючи вбудовані датчики і апаратне забезпечення.

Рис. 2. 3-вимірна тестова платформа для HIL-тестування [19]

Обробка інформації в БПС включає кілька послідовних етапів: збір даних, фільтрацію шумів, інтеграцію різнорідних джерел, прогнозування станів і прийняття рішень.

Наукові дослідження в цій галузі розвиваються переважно в двох напрямках, пов'язаних із застосуванням класичних та інтелектуальних методів обробки даних.

Класичні підходи, засновані на теоріях оптимальної фільтрації, байєсівського оцінювання та оптимального управління, характеризуються математичною строгістю, передбачуваністю та стійкістю. Однак їх принциповим обмеженням залишається недостатня здатність до адаптації в умовах швидко мінливого зовнішнього середовища.

Сучасні інтелектуальні методи включають алгоритми машинного навчання, нейронні мережі та еволюційні алгоритми, які дозволяють підвищити рівень автономності систем і якість обробки даних у складних динамічних сценаріях. Разом з тим, їх практична реалізація пов'язана з необхідністю значних обчислювальних ресурсів, а також проблемами верифікації та інтерпретованості рішень, що стримує їх впровадження в критично важливі системи.

Важливим напрямком досліджень залишається розробка гібридних рішень, що поєднують переваги класичних та інтелектуальних підходів при оптимізації обчислювального навантаження. Особлива увага приділяється створенню ефективних алгоритмів, здатних функціонувати на ресурсообмежених бортових контролерах, гарантуючи необхідні показники точності, відмовостійкості та надійності системи.

Особлива увага в рамках даного напрямку приділяється розробці та вдосконаленню алгоритмів злиття даних (data fusion). Зазначені алгоритми забезпечують інтеграцію інформації, що надходить від різних датчиків: інерційних вимірювальних блоків, супутникових навігаційних приймачів, лідарів і камер – з метою формування єдиної і достовірної моделі навколишнього середовища.

Ефективне злиття даних сприяє підвищенню точності, надійності та стійкості навігації в умовах невизначеності та впливу зовнішніх перешкод. Прикладом класичного алгоритму злиття даних є фільтр Калмана [20] і його численні модифікації, що застосовуються для оцінки стану динамічних систем у реальному часі. Даний алгоритм є одним з найбільш поширених і ефективних методів оптимальної фільтрації за наявності гаусових шумів і лінійних моделей динаміки об'єкта.

Однак, він має ряд обмежень, які знижують його застосовність в реальних умовах експлуатації БПС. До основних недоліків даного підходу відносяться: припущення лінійності динамічної моделі і моделі вимірювань; нестійкість при значних відхиленнях від гауссового розподілу помилок. У зв'язку з цим активно розвиваються різні модифікації класичного фільтра Калмана, такі як розширений фільтр Калмана (EKF), фільтр Калмана без запаху (UKF), а також фільтр частинок. Вони широко використовуються в популярних автопілотах, наприклад, PX4 і Ardupilot [21; 22]. Порівняльний аналіз зазначених фільтрів та особливості їх застосування наведені в роботі [23].

Популярною альтернативою фільтрам Калмана є фільтр Mahony [24], що належить до класу алгоритмів злиття даних.

Даний алгоритм являє собою обчислювально ефективний метод оцінки орієнтації, заснований на використанні зворотного зв'язку по похибці. На відміну від калманівських підходів, фільтр Mahony не вимагає лінеаризації математичної моделі системи, що істотно знижує обчислювальні витрати і спрощує реалізацію на бортових контролерах з обмеженими ресурсами.

До інтелектуальних алгоритмів злиття даних можна віднести фільтр частинок на основі генетичного алгоритму [25]. Даний підхід поєднує принципи фільтра частинок і генетичного алгоритму, що дозволяє підвищити ефективність оцінки стану системи в умовах сильної нелінійності і невизначеності вимірювань.

У класичному фільтрі частинок розподіл ймовірності стану апроксимується множиною частинок, ваги яких оновлюються на основі функції правдоподібності. Однак при обмеженій кількості частинок виникає проблема виродження частинок. Використання генетичних операторів селекція, кросовер і мутація, дозволяє підтримувати різноманітність частинок і уникати передчасної збіжності фільтра.

У системах управління БПС точність оцінки параметрів стану істотно залежить від якості даних, що надходять з бортових датчиків. У процесі експлуатації такі дані часто містять шуми, викликані зовнішніми перешкодами, вібраціями, нестабільністю живлення або обмеженою точністю вимірювальних елементів. Наявність цих шумів призводить до спотворення вихідної інформації, що знижує достовірність подальших обчислень. Для забезпечення коректної роботи алгоритмів злиття даних необхідно попередньо виконати фільтрацію сигналів. Вона може бути виконана з використанням цифрових фільтрів [26; 27].

Не менш важливим аспектом функціонування БПС є методи управління при виконанні польотних завдань. Найбільш широко застосовуються ПІД-регулятори [28; 29], завдяки своїй простоті, надійності та здатності ефективно компенсувати відхилення при зовнішніх збуреннях. Вони використовуються для стабілізації кутів орієнтації, підтримки висоти, швидкості та положення апарату.

У сучасних системах управління БПС також знаходять застосування методи на основі лінійно-квадратичного управління [30]. Їх використання дозволяє підвищити стійкість системи і забезпечити оптимальну якість управління навіть при наявності шумів і невизначеності в даних датчиків.

Для виконання різних завдань БПС важливу роль відіграють алгоритми навігації. Сучасні методи навігації досить різноманітні (наприклад, навігація з використанням комп'ютерного зору) і в більшості

випадків є обчислювально складними, тому істотно залежать від конкретного сценарію застосування. У зв'язку з цим їх зазвичай реалізують на додатковому обчислювальному модулі, наприклад на міні-ПК Raspberry Pi, NVIDIA Jetson або аналогічних одноплатних комп'ютерах, що працюють паралельно з основним польотним контролером.

**Висновок**

Проведений огляд сучасних методів обробки сенсорної інформації та програмних архітектур бортових систем управління БПС показав, що підвищення автономності апаратів безпосередньо залежить від ефективної інтеграції алгоритмічних рішень з інженерією програмного забезпечення. Модульні архітектури, патерни обміну даними, а також методології V-моделі та Model-Based Design забезпечують необхідну масштабованість, надійність та детермінізм роботи систем у реальному часі. Класичні та інтелектуальні методи злиття даних, поєднані з сучасними підходами до тестування (SIL, HIL), формують основу для створення високоточної та стійкої навігації в умовах невизначеності та динамічних зовнішніх впливів.

Разом із тим, подальший розвиток систем управління БПС вимагає комплексного вирішення суперечності між зростаючими обчислювальними потребами алгоритмів та обмеженими ресурсами бортових платформ. Перспективними напрямами є оптимізація обчислень, розробка гібридних моделей фільтрації, дослідження інтерпретованих методів машинного навчання та вдосконалення інструментів апаратно-програмного тестування. Синергія алгоритмічних, архітектурних та інженерних підходів є ключем до створення більш автономних, безпечних і надійних безпілотних систем наступного покоління.

***Література***


1. Singhal G., Bansod B., Mathew L. Unmanned aerial vehicle classification, applications and challenges: A review. 2018. DOI: https://doi.org/10.20944/preprints201811.0601.v1.
2. Петросян А. Р., Петросян Р. В., Колос К. Р. (2021). Розробка платформи віддаленого управління інфраструктурою Інтернет речей. *Технічна інженерія*. (1 (87)). 73–80. DOI: https://doi.org/10.26642/ten-2021-1(87)-73-80.
3. Govorcin M., Pribicevic B., Dapo A. Comparison and analysis of software solutions for creation of a digital terrain model using unmanned aerial vehicles. *14th International Multidisciplinary Scientific GeoConference SGEM 2014*. 2014. P. 99–108. DOI: https://doi.org/10.13140/2.1.2352.4803.
4. Chong M., Chuntao L. Design of flight control software for small unmanned aerial vehicle based on VxWorks. *Proceedings of 2014 IEEE Chinese Guidance, Navigation and Control Conference*. 2014. P. 1831–1834. DOI: https://doi.org/10.1109/CGNCC.2014.7007459.
5. PX4 System Architecture | PX4 Guide (main). *PX4 Autopilot Documentation*. URL: https://docs.px4.io/main/en/concept/px4_systems_architecture (дата звернення: 21.10.2025).
6. Architectural Process for Flight Control Software of Unmanned Aerial Vehicle with Module-Level Portability / T. Jargalsaikhan et al. *Aerospace*. 2022. No. 2. Vol. 9. P. 62. DOI: https://doi.org/10.3390/aerospace9020062.
7. Hegde M., Raveendra M. Development of flight control software for Unmanned Aerial Vehicle. *Int. J. Emerg. Technol. Comput. Sci. Electron*. 2015. Vol. 14. P. 661–664.
8. PX4 Architectural Overview | PX4 Guide (main). *PX4 Autopilot Documentation*. URL: https://docs.px4.io/main/en/concept/architecture (дата звернення: 21.10.2025).
9. Welcome to the Development Section | Betaflight. *Pushing the limits of UAV performance | Betaflight*. URL: https://www.betaflight.com/docs/development (дата звернення: 21.10.2025).
10. Meier L., Honegger D., Pollefeys M. PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms. *IEEE international conference on robotics and automation (ICRA)*. 2015. P. 6235–6240. DOI: https://doi.org/10.1109/ICRA.2015.7140074.
11. Bossard F. Event driven architecture for hard real-time embedded systems. *2nd Embedded Real Time Software Congress (ERTS'04)*. 2004. URL: https://hal.science/hal-02275439v1/document
12. On Embedding a Dataflow Architecture in a Multi-Robot System / M. J. Bagchi et al. *2022 Sixth IEEE International Conference on Robotic Computing (IRC) (Italy, 5–7 December 2022)*. 2022. DOI: https://doi.org/10.1109/irc55401.2022.00052.
13. Mihalic F., Truntic M., Hren A. Hardware-in-the-Loop Simulations: A Historical Overview of Engineering Challenges. *Electronics*. 2022. Vol. 11. No. 15. P. 2462. DOI: https://doi.org/10.3390/electronics11152462.
14. Model-Based Design of UAV Autopilot Software / M. Zuo et al. *2nd International Conference on Computer and Information Applications (ICCIA 2012)* (China, 8–9 December 2012). 2012. DOI: https://doi.org/10.2991/iccia.2012.290.
15. Badra A. A., Aïello O., Chaudemar J.-C. Applying a Model-Based Systems Engineering Approach to Model an Unmanned Aerial Vehicle Mission. *IEEE International Systems Conference (SysCon)* (Vancouver, BC, Canada, 17–20 April 2023). 2023. DOI: https://doi.org/10.1109/syscon53073.2023.10131224.
16. Aniche M. Effective Software Testing: A Developer's Guide. Manning Publications Co., 2022. 328 p.
17. Testing automotive embedded systems under X-in-the-loop setups / G. Tibba et al. *ICCAD '16: IEEE/ACM INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN* (Austin, Texas. New York, NY, USA, 2016). DOI: https://doi.org/10.1145/2966986.2980076.
18. Software-in-the-loop simulation of the forerunner UAV system / A. Hiba et al. *IFAC-PapersOnLine*. 2022. Vol. 55. No. 14. P. 139–144. DOI: https://doi.org/10.1016/j.ifacol.2022.07.596.
19. Hancer M., Bitirgen R., Bayezit I. Designing 3-DOF Hardware-In-The-Loop Test Platform Controlling Multirotor Vehicles. *IFAC-PapersOnLine*. 2018. № 4. Vol. 51. P. 119–124. DOI: https://doi.org/10.1016/j.ifacol.2018.06.058.
20. Industrial Applications of the Kalman Filter: A Review / F. Auger et al. *IEEE Transactions on Industrial Electronics*. 2013. Vol. 60. No. 12. P. 5458–5471. DOI: https://doi.org/10.1109/tie.2012.2236994.

21. Using PX4's Navigation Filter (EKF2). *PX4 Guide (main)*. URL: https://docs.px4.io/main/uk/advanced_config/tuning_the_ecl_ekf.html (дата звернення: 21.10.2025).

22. Extended Kalman Filter (EKF) – Copter documentation. *ArduPilot – Versatile, Trusted, Open*. URL: https://ardupilot.org/copter/docs/common-apm-navigation-extended-kalman-filter-overview.html (дата звернення: 21.10.2025).

23. Evaluation of localization by extended Kalman filter, unscented Kalman filter, and particle filter-based techniques / I. Ullah et al. *Wireless Communications and Mobile Computing*. 2020. Vol. 2020. P. 1–15. DOI: https://doi.org/10.1155/2020/8898672.

24. Mahony R., Hamel T., Pflimlin J.-M. Nonlinear Complementary Filters on the Special Orthogonal Group. *IEEE Transactions on Automatic Control*. 2008. Vol. 53. No. 5. P. 1203–1218. DOI: https://doi.org/10.1109/tac.2008.923738.

25. Zhao B., Hu J., Ji B. An improved particle filter based on genetic resampling. *International Conference on Automation, Mechanical Control and Computational Engineering*. Atlantis Press, 2015. P. 1353–1358. DOI: https://doi.org/10.2991/amcce-15.2015.125.

26. Managing Gyro Noise with the Dynamic Harmonic Notch Filters – Copter documentation. ArduPilot – Versatile, Trusted, Open. URL: https://ardupilot.org/copter/docs/common-imu-notch-filtering.html (дата звернення: 21.10.2025).

27. Petrosian R., Chukhov V., Petrosian A. Development of a method for synthesis the FIR filters with a cascade structure based on genetic algorithm. *Technology audit and production reserves*. 2021. Vol. 4. No. 2 (60). P. 6–11. DOI: https://doi.org/10.15587/2706-5448.2021.237271.

28. Lopez-Sanchez I., Moreno-Valenzuela J. PID control of quadrotor UAVs: A survey. *Annual Reviews in Control*. 2023. Vol. 56. P. 100900. DOI: https://doi.org/10.1016/j.arcontrol.2023.100900.

29. Petrosian R., Pilkevych I., Petrosian A. Algorithm for optimizing a PID controller model based on a digital filter using a genetic algorithm. *Proceedings of the 3rd Edge Computing Workshop*. 2023. Vol. 3374. Pp. 97–111. URL: https://ceur-ws.org/Vol-3374/paper07.pdf

30. Attitude UAV Stability Control Using Linear Quadratic Regulator-Neural Network (LQR-NN) / Oktaf Agni Dhewa et al. *IIUM Engineering Journal*. 2024. Vol. 25. No. 2. P. 246–265. DOI: https://doi.org/10.31436/iiumej.v25i2.3119.

### *References*

1. Singhal G., Bansod B., Mathew L. (2018). Unmanned Aerial Vehicle Classification, Applications and Challenges: A Review. *Preprints*. DOI: https://doi.org/10.20944/preprints201811.0601.v1.

2. Petrosian A. R., Petrosian R. V., Kolos K. R. (2021). Rozrobka platformy viddalenoho upravlinnia infrastrukturoiu Internet rechei [Development of remote control platform for the internet of things infrastructure]. Tekhnichna inzheneriia [Technical engineering]. (1 (87)). 73–80. DOI: https://doi.org/10.26642/ten-2021-1(87)-73-80 [in Ukrainian].

3. Govorcin M., Pribicevic B., Dapo, A. (2014). Comparison and analysis of software solutions for creation of a digital terrain model using unmanned aerial vehicles. *In 14th International Multidisciplinary Scientific GeoConference SGEM 2014* (pp. 99–108). DOI: https://doi.org/10.13140/2.1.2352.4803.

4. Chong M., Chuntao L. (2014). Design of flight control software for small unmanned aerial vehicle based on VxWorks. *In Proceedings of 2014 IEEE Chinese Guidance, Navigation and Control Conference* (pp. 1831–1834). DOI: https://doi.org/10.1109/CGNCC.2014.7007459.

5. PX4 System Architecture | PX4 Guide (main). *PX4 Autopilot Documentation*. URL: https://docs.px4.io/main/en/concept/px4_systems_architecture (accessed: 21.10.2025).

6. Jargalsaikhan T., Lee K., Jun Y.-K., Lee S. (2022). Architectural Process for Flight Control Software of Unmanned Aerial Vehicle with Module-Level Portability. *Aerospace*. 9 (2). 62. DOI: https://doi.org/10.3390/aerospace9020062.

7. Hegde M., Raveendra M. (2015). Development of flight control software for Unmanned Aerial Vehicle. *Int. J. Emerg. Technol. Comput. Sci. Electron*. 14. 661–664.

8. PX4 Architectural Overview | PX4 Guide (main). *PX4 Autopilot Documentation*. URL: https://docs.px4.io/main/en/concept/architecture (accessed: 21.10.2025).

9. Welcome to the Development Section | Betaflight. Pushing the limits of UAV performance | Betaflight. URL: https://www.betaflight.com/docs/development (accessed: 21.10.2025).

10. Meier L., Honegger D., Pollefeys M. (2015). PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms. *IEEE international conference on robotics and automation (ICRA)* (pp. 6235-6240). DOI: https://doi.org/10.1109/ICRA.2015.7140074.

11. Bossard F. (2004). Event driven architecture for hard real-time embedded systems. In 2nd Embedded Real Time Software Congress (ERTS'04). URL: https://hal.science/hal-02275439v1/document

12. Bagchi M. J., Kulkarni D. D., Nair S. B., Das P. K. (2022). On Embedding a Dataflow Architecture in a Multi-Robot System. *In 2022 Sixth IEEE International Conference on Robotic Computing (IRC). IEEE.* DOI: https://doi.org/10.1109/irc55401.2022.00052.

13. Mihalic F., Truntic M., Hren A. (2022). Hardware-in-the-Loop Simulations: A Historical Overview of Engineering Challenges. *Electronics*. 11 (15). 2462. DOI: https://doi.org/10.3390/electronics11152462.

14. Zuo M., Liu Y., Qian Y., Hu X., Zhao X. (2012). Model-Based Design of UAV Autopilot Software. *In 2nd International Conference on Computer and Information Applications (ICCIA 2012)*. Atlantis Press. DOI: https://doi.org/10.2991/iccia.2012.290.

15. Badra A., Aïello O., Chaudemar J.-C. (2023). Applying a Model-Based Systems Engineering Approach to Model an Unmanned Aerial Vehicle Mission. *In 2023 IEEE International Systems Conference (SysCon)*. IEEE. DOI: https://doi.org/10.1109/syscon53073.2023.10131224.

16. Aniche M. (2022). *Effective Software Testing: A Developer's Guide*. Manning Publications Co. 328 p.

17. Tibba G., Malz C., Stoermer C., Nagarajan N., Zhang L., Chakraborty S. (2016). Testing automotive embedded systems under X-in-the-loop setups. *In ICCAD '16: IEEE/ACM INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN*. ACM. DOI: https://doi.org/10.1145/2966986.2980076.

18. Hiba A., Bauer P., Nagy M., Simonyi E., Kisari A., Kuna G. I., Drotar I. (2022). Software-in-the-loop simulation of the forerunner UAV system. *IFAC-PapersOnLine*. 55 (14). 139–144. DOI: https://doi.org/10.1016/j.ifacol.2022.07.596.

19. Hancer M., Bitirgen R., Bayezit I. (2018). Designing 3-DOF hardware-in-the-loop test platform controlling multirotor vehicles. *IFAC-PapersOnLine*. 51 (4). 119–124. DOI: https://doi.org/10.1016/j.ifacol.2018.06.058.

20. Auger F., Hilairet M., Guerrero J. M., Monmasson E., Orlowska-Kowalska T., Katsura S. (2013). Industrial Applications of the Kalman Filter: A Review. *IEEE Transactions on Industrial Electronics*. 60 (12). 5458–5471. DOI: https://doi.org/10.1109/tie.2012.2236994.

21. *Using PX4's Navigation Filter (EKF2) | PX4 Guide (main)*. PX4 Autopilot Documentation. URL: https://docs.px4.io/main/uk/advanced_config/tuning_the_ecl_ekf.html

22. *Extended Kalman Filter (EKF) – Copter documentation. ArduPilot – Versatile, Trusted, Open.* URL: https://ardupilot.org/copter/docs/common-apm-navigation-extended-kalman-filter-overview.html

23. Ullah I., Su X., Zhu J., Zhang X., Choi D., Hou Z. (2020). Evaluation of localization by extended Kalman filter, unscented Kalman filter, and particle filter-based techniques. *Wireless Communications and Mobile Computing*. 2020. 1–15. DOI: https://doi.org/10.1155/2020/8898672.

24. Mahony R., Hamel T., Pflimlin J.-M. (2008). Nonlinear Complementary Filters on the Special Orthogonal Group. *IEEE Transactions on Automatic Control*. 53 (5). 1203–1218. DOI: https://doi.org/10.1109/tac.2008.923738.

25. Zhao B., Hu J.-W., Ji B. (2015). An improved particle filter based on genetic resampling. *In 2015 International Conference on Automation, Mechanical Control and Computational Engineering*. Atlantis Press. DOI: https://doi.org/10.2991/amcce-15.2015.125.

26. *Managing Gyro Noise with the Dynamic Harmonic Notch Filters* – Copter documentation. ArduPilot – Versatile, Trusted, Open. URL: https://ardupilot.org/copter/docs/common-imu-notch-filtering.html

27. Petrosian R., Chukhov V., Petrosian A. (2021). Development of a method for synthesis the FIR filters with a cascade structure based on genetic algorithm. *Technology audit and production reserves*. 4 (2 (60)). 6–11. DOI: https://doi.org/10.15587/2706-5448.2021.237271.

28. Lopez-Sanchez I., Moreno-Valenzuela J. (2023). PID control of quadrotor UAVs: A survey. *Annual Reviews in Control*, 56, 100900. DOI: https://doi.org/10.1016/j.arcontrol.2023.100900.

29. Petrosian R., Pilkevych I., Petrosian A. (2023). Algorithm for optimizing a PID controller model based on a digital filter using a genetic algorithm. *Proceedings of the 3rd Edge Computing Workshop*. 2023. 3374. 97–111. URL: https://ceur-ws.org/Vol-3374/paper07.pdf

30. Oktaf Agni Dhewa, Fatchul Arifin, Ardy Seto Priyambodo, Anggun Winursito, Yasir Mohd. Mustafa. (2024). Attitude UAV Stability Control Using Linear Quadratic Regulator-Neural Network (LQR-NN). *IIUM Engineering Journal*. 25 (2). 246–265. DOI: https://doi.org/10.31436/iiumej.v25i2.3119.