НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Радіотехнічний факультет

Кафедра прикладної радіоелектроніки

«На правах рукопису»
УДК _____

До захисту допущено:
В.о. зав. кафедри
_____ Михайло СТЕПАНОВ
«___»_____20__ р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою «Інтелектуальні технології радіоелектронної техніки»

за спеціальністю 172 «Телекомунікації та радіотехніка»

на тему: «Автоматизоване проектування повітряних дронів на основі графової граматики»

Виконав:
студент 2 курсу, групи РЕ-11мп
Струков Демід Денісович _____

Керівник:
Ст. викладач, PhD
Мирончук Олександр Юрійович
_____

Рецензент:
Ст. викладач
Захарченко Оксана Степанівна _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2022 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Радіотехнічний факультет
Кафедра прикладної радіоелектроніки

Рівень вищої освіти – другий (магістерський)

Спеціальність – 172 «Телекомунікації та радіотехніка»

Освітньо-професійна програма «Інтелектуальні технології радіоелектронної техніки»

ЗАТВЕРДЖУЮ
В.о.зав. кафедри
_____ Михайло СТЕПАНОВ
«___»_____2022 р.

ЗАВДАННЯ
на магістерську дисертацію студента
Струков Демід Денісович

1. Тема дисертації «Автоматизоване проектування повітряних дронів на основі графової граматики»
   науковий керівник дисертації Мирончук Олександр Юрійович
   затверджені наказом по університету від «09» листопада 2022 р. №4137-С
2. Об’єкт дослідження: Система автоматичного створення конструкцій дронів
3. Термін подання студентом дисертації 11 грудня 2022 року
4. Вихідні дані: Система автоматизованого проектування повітряних дронів на основі графової граматики.
5. Перелік завдань, які потрібно розробити: Розгляд існуючих методів автоматизованого проектування різних конструкцій. Розробка та реалізація системи автоматизованого проектування повітряних дронів на основі графової граматики.
6. Орієнтовний перелік графічного (ілюстративного) матеріалу: Презентація по роботі в обсязі не менше 10 слайдів.
7. Орієнтовний перелік публікацій: доповідь на конференції.
8. Дата видачі завдання 05 вересня 2022 року

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Актальність проблеми та огляд існуючих рішень.	08.09 – 16.09	Розділ 1
2	Розбір роботи яка стала основою тематики.	17.09 – 24.09	Розділ 1
3	Планування стартап проекту	25.09 – 02.10	Розділ 2
4	Аналіз оточення для симуляції «Unity» та аналіз системи «ml-agents»	03.05 – 09.10	Розділ 3
5	Навчання примітивного дрону у симуляції	10.10 – 16.10	Розділ 3
6	Створення алгоритму опису конструкції	17.10 – 22.10	Розділ 3
7	Створення конструкцій на основі опису	23.10 – 29.10	Розділ 3
8	Створення проекту на python	30.10 – 09.11	Розділ 3
9	Робота над алгоритмом передбачення ефективності та результати	10.11 – 17.11	Розділ 4 /Додатки
10	Оформлення магістерської дисертації	09.12.2022р.	

Студент Демід СТРУКОВ

Керівник Олександр МИРОНЧУК

РЕФЕРАТ

Звіт про дипломну роботу: 58 сторінки, 34 рисунків, 1 додаток, 21 джерел.

Автоматизоване проектування вже давно є досить важливою частиною розвитку технологій, а віднедавна алгоритми машинного навчання дістались змоги генерувати різноманітний контент, від чогось простого як створення ілюстрацій за запитом користувача, до згортання клітин білка, що допомагає у створенні ліків.

Якщо припустити, що більш інноваційний дизайн, який буде отриманий в результаті випадкового генерування може покращити функціональність конструкції, та виявити більш підходящі рішення для конкретних задач. Тому э шанс в якому побудована комп'ютерна модель спроектована для завдання – завдяки системі, яка не зазнала надмірного впливу попередніх умовностей.

Метою роботи є створення різноманітних конструкцій дронів, які теоретично є більш оптимальним рішенням для конкретних завдань, та порівняння результатів з вже існуючими рішеннями.

Об'єкт дослідження – конструкції дронів.

Предметом дослідження – система автоматичного створення конструкцій дронів, по запиту вирішення необхідної задачі.

Область застосування – автоматизація процесів. У дипломній роботі проходить проектування системи автоматизованого створення повітряних дронів для формування конструкцій під необхідні задачі.

Методи дослідження – алгоритми побудови графів для репрезентації конструкцій. Симуляції для її відтворення у віртуальному світі. Навчання з підкріпленням для тестування конструкції у симуляції. Алгоритми пошуку по графу, для оптимізації пошуку конструкції.

Будування системи проводиться завдяки симуляції в прикладній програмі Unity на пакеті для створення агентів навчання з підкріпленням «ml-agents»

Концепція створення процедурних конструкцій не є новою. Ця технологія вже давно використовується у комп’ютерних іграх, для генерації будь чого: створінь, будівель, місцевості. Ця робота була виконана від натхнення від дослідження [3], яке зорієнтоване на створення процедурних створінь для вирішення завдання пересування по різноманітному оточенню. Використовуючи цю технологію, можна створити систему створення конструкцій дронів, за якою можна перевірити, чи є конструкція існуючих дронів оптимальним рішенням, та знайти подібні або кращі рішення до базових конструкцій.

Публікації:

Струков Д.Д. Автоматизоване проектування легких літальних апаратів на основі графової граматики / Д.Д. Струков, О.Ю. Мирончук // Міжнародна науково-технічна конференція «Радіотехнічні проблеми, сигнали, апарати та системи»: матеріали конференції, 22 - 24 листопада 2022 р., м. Київ, Україна / КПІ ім. Ігоря Сікорського, РТФ. – Київ : КПІ ім. Ігоря Сікорського, 2022. – С. 100–102.

НАВЧАННЯ З ПІДКРІПЛЕННЯМ, АВТОМАТИЗОВАНЕ ПРОЕКТУВАННЯ, ГРАФ, ДРОН, МАШИНЕ НАВЧАННЯ

ABSTRACT

Thesis report: 58 pages, 34 figures, 1 appendix, 21 references.

Computer-aided design has long been an important part of technology development, and more recently, machine-learned algorithms have been able to generate a variety of content, from something as simple as creating illustrations at the user's request to folding protein cells to aid in drug development.

Assuming that the more innovative design that results from random generation can improve the functionality of the design, and identify more suitable solutions for specific problems. Therefore, there is a chance in which the constructed computer model is designed for the task – thanks to a system that has not been unduly influenced by previous conventions.

The purpose of the work is to create a variety of drone designs that are theoretically more optimal solutions for specific tasks and compare the results with existing solutions.

The object of study is the design of drones.

Subject of research – the system of automatic creation of drone spacecraft, on request to solve the necessary problem.

Scope of application – Process automation. In the diploma work is the design of the system of automated creation of aerial drones for the formation of structures for the necessary tasks.

Research methods – Algorithms for constructing graphs for representing structures. Simulations for its reproduction in the windy world. Reinforcement learning for testing the design in simulation. Graph search algorithms to optimize the design search.

The system is built through simulation in the Unity application program on the package for creating reinforcement learning agents "ml-agents"

The concept of creating procedural constructs is not new. This technology has long been used in computer games to generate anything: creatures, buildings, terrain. This work was inspired by a scientific study [3], which focuses on the

creation of procedural creatures to solve the problem of movement on a variety of terrain. Using this technology, it is possible to create a system for creating drone designs, which can be used to check whether the design of existing drones is the optimal solution and to find similar or better solutions to the basic designs.

Publications:

Струков Д.Д. Автоматизоване проектування легких літальних апаратів на основі графової граматики / Д.Д. Струков, О.Ю. Мирончук // Міжнародна науково-технічна конференція «Радіотехнічні проблеми, сигнали, апарати та системи»: матеріали конференції, 22 - 24 листопада 2022 р., м. Київ, Україна / КПІ ім. Ігоря Сікорського, РТФ. – Київ : КПІ ім. Ігоря Сікорського, 2022. – С. 100–102.

REINFORCEMENT LEARNING, COMPUTER-AIDED DESIGN, GRAPH, DRONE, MACHINE LEARNING

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

RL – Reinforcement learning – Навчання з підкріпленням;

MCTS – Monte Carlo tree search – Дерево пошуку Монте-Карло;

БпЛА – Безпілотний літальний апарат;

САПР – Система автоматизованого проєктування;

DOT – Graph description language – Мова опису графів

ML – Machine learning – Машинне навчання

ГА – Генетичний алгоритм

ГГ – Графова-граматика

GNN – Graph neural network – Графові нейронні мережі

PPO – Proximal Policy Optimization – Оптимізація проксимальної політики

GHS – Graph Heuristic Search – Графовий евристичний пошук

NN – Neural network – Нейронні мережі

ВСТУП

Топологічні конфігурації конструкцій зазвичай розробляються вручну людськими дизайнерами. Однак, можливо, що серед великої кількості можливих конфігурацій існують деякі рішення, які потенційно перевершують існуючі рішення і, водночас, відрізняються від конфігурацій, створених на основі інших інтуїтивних ідей, і, отже, не можуть бути виявлені без методу автоматизованого проектування. У роботі розглядається проблема автоматизованого проектування систем з застосування методу, заснованого на графовій граматиці, який поєднує динамічне моделювання з метою оцінки та еволюції змін. Зокрема, показано, що метод дозволяє знаходити різні нові рішення проблеми проектування двовимірних конструкцій. Для синтезу системи пропонується розподіл дій параметричної оптимізації та топологічного синтезу.

1 АВТОМАТИЗОВАНЕ ПРОЕКТУВАННЯ

1.1 Основні положення про автоматизоване проектування та штучний інтелект

Автоматизація та розуміння конструкції робота, а також взаємодія між конструкцією та контролером робота вже давно є ключовим питанням досліджень [1]. Це особливо складна дослідницька проблема, оскільки простір проектування є широким і важкорозв'язним, а інструменти для автоматизованого та ефективного дослідження дуже обмежені. Щоб забезпечити можливість великомасштабного пошуку роботів потрібні підходи для структурування простору проектування, інструменти для ефективного пошуку, а також симулятори для вивчення та оцінки багатьох тисяч конструкцій [2].

Автоматизоване проектування – це технологія яка являється по своїй суті використанням комп’ютерних систем для полегшення створення інженерних рішень, які пов’язані з аналізом чи оптимізацією існуючих систем. Завдяки САПР інженери можуть створювати свої проекти, розраховувати та оптимізувати їх.

Основна проблема роботи з більшістю САПР полягає у тому, що треба навчатись роботі з програмою, інструментом (роботі з її інтерфейсом), що може займати немало часу та людських ресурсів. Функціонал, який дозволяє створювати готове рішення одним «кліком» на кнопку, завжди був бажаною потребою для користувачів. Насамперед це важливо тому, що у сучасному світі більшість можливостей впирається у стінку бюджету. Проектування проекту представляє собою комплексний процес, який займає багато часу, грошей та сил людей. Тому будь-які рішення, які надають можливість пришвидшувати проектування, є необхідними.

У 2020 році команда DeepMind змогла вирішити одну з найскладніших проблем біології, пов'язаною з передбаченням як білки складаються з ланцюжка амінокислот у тривимірні форми. Рішення реалізоване за

допомогою штучного інтелекту. Структури білка, передбачені штучним інтелектом, та експериментально визначені структури, які збігаються майже повністю, представлені на рис. 1.1.

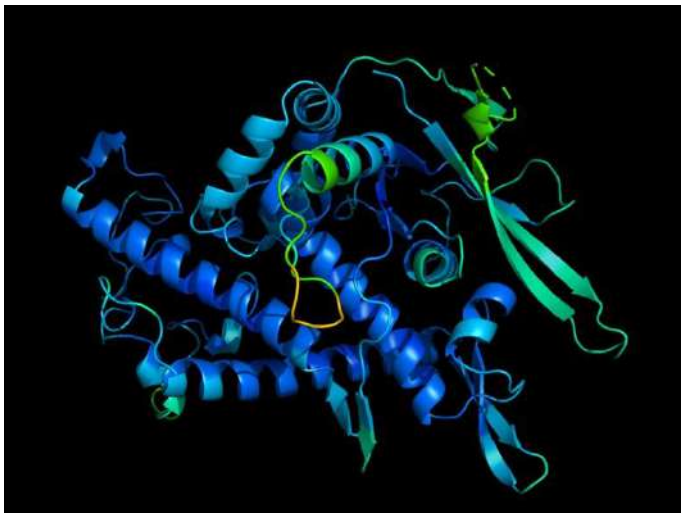


Рис. 1.1 – Структури білка передбачені штучним інтелектом (синій колір) та експериментально визначені (зелений колір)

Організм використовує десятки тисяч різних білків, кожен з яких складається з десятків і сотень амінокислот. Порядок амінокислот диктує, як міріади поштовхів і притягань між ними породжують складні тривимірні форми білків, які, в свою чергу, визначають, як вони функціонують. Знання цих форм допомагає дослідникам розробляти ліки, які можуть затримуватися в білкових щілинах. А можливість синтезувати білки з потрібною структурою може прискорити розробку ферментів для виробництва біопалива та розкладання відходів пластику.

Здатність точно передбачати структуру білка за його амінокислотною послідовністю є величезним благом для медико-біологічних наук і медицини. Це значно прискорило б зусилля, спрямовані на розуміння будування клітин, і дозволило б швидше і досконаліше відкривати ліки.

Ще одним прикладом систем автоматичного проектування можна назвати штучний інтелект DALL-E, який створює зображення по запиту, що надходить від користувача.

DALL-E вивчив взаємозв'язок між зображеннями і текстом, який використовується для їх опису. Він використовує процес під назвою «дифузія», який починається з шаблону випадкових точок і поступово змінює цей шаблон у бік зображення, коли розпізнає певні аспекти цього зображення.

Це дуже корисна система для копірайтерів, веб-розробників, тому що штучний інтелект може створити абсолютно точні зображення, які відповідатимуть вашому контенту. Окрім того, зображення, згенеровані за допомогою DALL-E, будуть вільні для використання і не матимуть авторських прав.

Також слід відмітити актуальність можливостей DALL-E у сфері дизайну персонажів. За допомогою нейромережі можуть бути створені будь-які мультяшні персонажі або реалістичні герої. Це суттєво спростить роботу дизайнерам, бо вони можуть генерувати кілька сотень прикладів персонажів з повною свободою дій по будь якому запиту. Приклад ілюстрацій «DALL-E» по запиту «Лисиця у космосі намальована Моне» представлений на рис. 1.2.

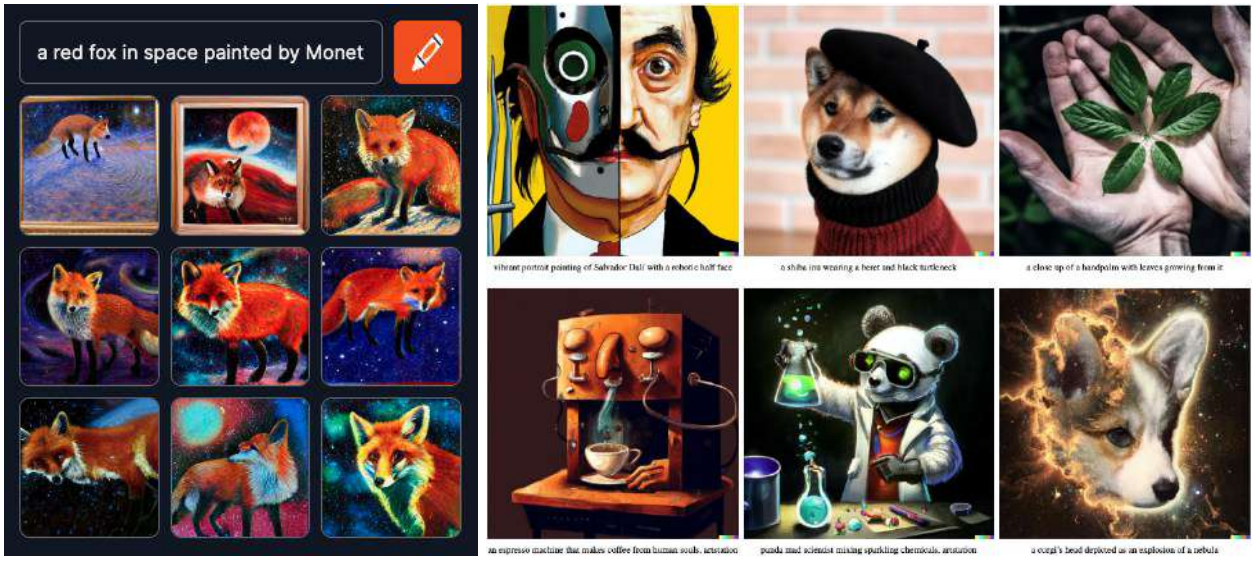


Рис. 1.2 – Ілюстрації «DALL-E» по запиту «Лисиця у космосі намальована Моне» (зліва) та випадкові ілюстрації, згенеровані системою (справа)

1.2 Огляд існуючих конструкції повіряних дронів

Безпілотні літальні апарати (БпЛА) все частіше використовуються у різноманітних сферах через своє швидке та економічно ефективне розгортання. БпЛА можна використовувати не тільки для розвідки, ударних операцій, але і як комунікаційну платформу. Зі зростанням вимог до автономності, інтелекту та багатозадачності БпЛА, ефективність та рівень інтелекту одномашинної роботи БпЛА поступово не відповідають вимогам їх застосування. При одиночному польоті обмежений запас енергії обмежує дальність польоту БпЛА і діапазон можливостей.

БпЛА можуть використовуватися в багатьох сценаріях і мають різноманітні варіанти виконання. Відповідно існує багато способів класифікації БпЛА.

За типом крила БпЛА поділяються на наступні види:

- з фіксованим крилом;
- з роторним крилом;
- з маховим крилом.

За вагою БпЛА можна розділити на наступні види:

- мікро (< 1 кг);
- міні (1-25 кг);
- важкі (> 25 кг).

За тривалістю польоту/далекобійністю БпЛА можна розділити на:

- короткі (< 5 год, < 100 км);
- середні (5-24 год, 100-400 км);
- довгі/далекобійні (> 24 год, > 1500 км).

За максимальною висотою польоту БпЛА можна класифікувати як:

- маловисотні (< 1 км);
- середньовисотні (1-10 км);
- висотні (> 10 км).

Крім того, безпілотні апарати можуть не тільки літати в повітрі, але й пересуватися по водній поверхні або під водою. Такі безпілотні апарати розділяються на гідролітаки, що запускаються з підводного човна, та підводні, що ще більше розширює класифікацію.

На рис. 1.3 представлено приклад різноманітних конструкцій БпЛА, які використовуються на теперішній час.

Рис. 1.3 – Приклади разноманітних конструкцій дронів: одноротрні, багатороторні, з нерухомим крилом, гібридні з нерухомим крилом

З рис. 1.3 то можна побачити, що більшість різновидів конструкцій БпЛА використовується для дронів воєнного призначення. Але звичайно, це не увесь список, і є безліч різноманітних конструкцій БпЛА, різних призначень, приклад яких представлено на Рис. 1.4.

Для більшості людей уявлення про конструкцію дрона сформувалось у вигляді звичайного квадрокоптера. Квадрокоптер являє собою найпростішу конструкцію, яка є ефективною з практичної точки зору. Квадрокоптер являється досить простим механізмом, у якого є чотири кінцівки і у кожній з них знаходиться гвинт. Мультикоптери з трьома, шістьма або вісьмома маніпуляторами також можливі, але працюють вони за тим же принципом, що і квадрокоптер. Два з роторів обертаються за годинниковою стрілкою, а два інших – проти годинникової стрілки. Квадрокоптери аеродинамічно нестабільні і вимагають бортового комп'ютера для перетворення вхідних

команд в команди, які змінюють швидкість обертання гвинтів для отримання бажаного руху.

Концепція квадрокоптера не є новою. Проекти пілотованих квадрокоптерів з'явилися ще в 1920-х і 1930-х роках, але ці ранні концепції мали погані характеристики, високий рівень нестабільності і вимагали багато зусиль від пілота. Розвиток електронних технологій в комп'ютерах управління польотом, безколекторних або безщіткових двигунах, менших мікропроцесорах, акумуляторах, акселерометрах, камерах і навіть GPS-системах зробив можливим проектування і польоти квадрокоптерів. Простота квадрокоптера зробила його дуже ефективною платформою для аерофотозйомки та відеозйомки.

Однак не слід обмежуватися використанням узагальненої конструкції. Якщо взяти до уваги те, що конструкція квадрокоптера являє собою загальне рішення для більшості проблем з необхідності використання повітряних дронів, то мають бути більш підходящі конструкції для кожної з задач окремо.

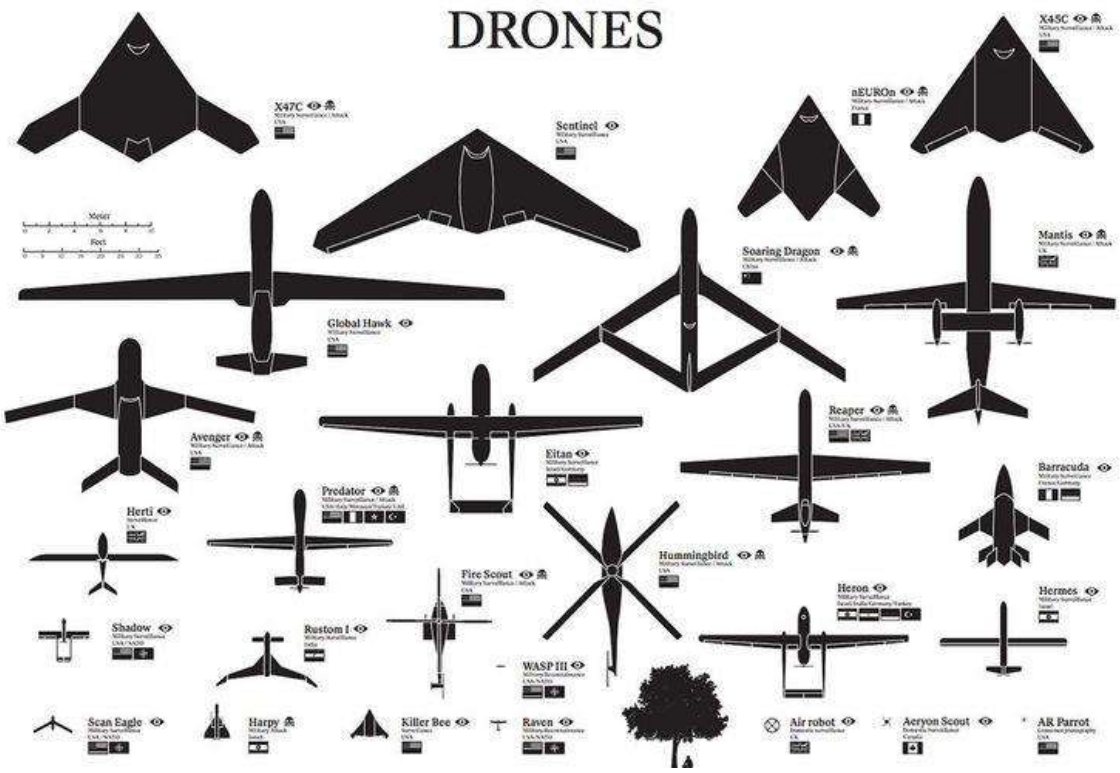


Рис. 1.4 – Розширений перелік можливих конструкцій повітряних дронів

1.3 Запропонована система автоматизованого проектування конструкцій дронів

Для вирішення завдання підбору конструкції дрона відповідно до поставленого завдання ця робота пропонує систему засновану на моделюванні для одночасної оптимізації фізичних структур та контролерів дронів. Мета системи – взяти набір примітивних компонентів, заданих користувачем, та згенерувати оптимальну структуру дрона та контролер для вирішення поставленої задачі. Примітивні компоненти включають різні типи суглобів, ланок та пропеллерів, кожен з яких із запропонованими користувачем атрибутами, такими як кути повороту, осі, розміри та вага. Користувач може задати склад примітивів відповідно до наявних у нього фізичних компонентів. У роботі кожна конструкція представлена у вигляді графа. Для ефективного пошуку у просторі графів конструкцій вводиться рекурсивна граматика графів, яка підкреслює мобільність та можливість виготовлення конструкції.

У цілому, у цій роботі представлені такі основні моменти:

- Рекурсивна граматика графів, що дозволяє генерувати широкий спектр внутрішньо реалізованих форм конструкцій.
- метод евристичного пошуку за графом для ефективного пошуку простору проектування, що описується граматикою. Цей метод порівнюється з альтернативними варіантами, включаючи пошук по дереву Монте-Карло та випадковий пошук.
- Демонстрація оптимізації з урахуванням задачі та оцінка кожної запропонованої конструкції. У роботі продемонстровані різноманітність інноваційних конструкцій дронів, для різних задач. Крім того, робота виявляє низку дронів для однієї задачі, або декількох одразу.

1.4 Споріднена робота

Натхнення на виконання даної роботи отримано від дослідження RoboGrammar: Graph Grammar for Terrain-Optimized Robot Design [3], у якому команда змогла створити систему, яка генерує оптимізовані конструкції роботів для пересування по заданій місцевості.

Формальні методи проектування є альтернативним підходом, який використовує шлях на основі моделей для проектування роботів та їх контролерів. В [3] демонструють формальний підхід, використовуючи велику бібліотеку проектування та спеціальні інструменти для конфігурування модульних роботів. Такі методи можуть вимагати значної ручної праці, не масштабуються та не узагальнюються добре.

Для пошуку в параметризованому просторі проектування, у нашому випадку в графовій граматиці, потрібні ефективні алгоритми пошуку. Генетичні алгоритми (ГА) ітеративно покращують дизайн за допомогою процесу, натхненого природною еволюцією. Застосовуються у реальних системах, ГА передають позитивні "фенотипи" між поколіннями роботів та дозволяють створювати великі популяції роботів. Робота [5] присвячена розвитку істот оптимізованих для пересування у тривимірному середовищі за допомогою ГА. Істоти представлені у вигляді спрямованих графів, які можуть включати багатогранники. Кінематичне дерево кожної істоти визначається шляхом простеження унікальних шляхів через спрямований граф. ГА широко використовується у співтоваристві робототехніків з очевидним успіхом [6], однак у деяких випадках вони не дуже добре масштабуються і можуть мати високу чутливістю до вхідних параметрів. Вони також можуть бути чутливими до таких параметрів, як розмір популяції, і швидкість мутацій, і існує мало свідчень того, що вони демонструють збіжність до глобальних чи навіть локальних мінімумів.

2 ПЛАНУВАННЯ СТАРТАП-ПРОЕКТУ

2.1 Зміст ідеї та аналіз ринку

Так як раніше у роботі було вказано про використання систем автоматизованого генерування контенту (генерування зображень, голосів, моделей) вони є прототипами майбутніх систем, і наразі більшість цих систем не монетизуется. Вони є або опен-соурс (тобто з відкритим доступом до проекту), або закриті, і це дає змогу використовувати їх технологію безкоштовно. І лише невелика частина усіх продукцій саме виставляє свою технологію на маркет, та продають надання доступу до користування системами на деякий час.

Таблиця 2.1 – Опис ідеї стартап-проекту

Зміст ідеї	*Напрямки застосування*	*Вигоди для користувача*
Автоматизоване створення конструкцій повітряних дронів	1. Автоматизоване процессів	Підвишення різноманітності результатів, ігноруючи сталі принципи конструювання
	2. Генерація рішення	Дозволяє швидше отримати результат, і у більшій кількості

Таблиця 2.2 – Фактори загроз при проектуванні проекту

№ п/п	*Фактор*	*Зміст загрози*
1.	Проблема конструкції	За цього фактору, сгенерована конструкція може мати неможливе до створення у реальному світи модель.
2.	Складність завдання правил для генерації конструкцій	Для складання правил користувачу треба розуміти базові знання з завдання правил до агента який використовує навчання з підкріпленням
3.	Складність впровадження саме нових сегментів	Наразі у системи, немає плану з легкого вбудування нових сегментів, для цього користувачу необхідно розуміти принципи роботи з симуляцією Unity

Так як ця система планується до створення на основі вже існуючого рішення, перетворивщи її завдання на конструкцію дронів з покращенням

алгоритму навчання, яке надасть можливість використовувати систему одразу з конструкцією, та більш денамічно адаптуватись до завдань.

Цей проект не розглядається для стартап-проекту, так як є експериментом задля перевірки можливих більш підходящіх конструкцій дронів, порівнянно з існуючими.

2.2 Заходи з комерціалізації проекту

Для цієї роботи присутне монетизація у вигляді надані доступу до використання системи. Таким чином будь-яка компанія, або юридичне лице може придбати доступ до використання системи, та створити конструкцію під свое необхіне завдання.

Наприклад є можливість кооперацій з іншими фірмами, такими як [11]. При кооперації з цією організацією, можна отримати генероване рішення, для покращення лісів завдяки створенню різноманітних повітряних істот, як один з їх прикладних «Ріко». Це дозволить розшилити пошук можливих рішень для роботів, та не орієнтуватись лище на існуючі рішення, та знаходити різноманітні рішення завдань.

Також є можливість кооперації з такими компаніями як Booston Dynamics яка найбільш відома за розробку серії динамічних високомобільних роботів, які спроможні на значну моторику руху. Таким чином в кооперації можливо можна знайти варіації конструкцій які не є основаними на живих істотах, та були спроектовані випадково.

Таблиця 2.3 – Опис ідеї стартап-проекту

№ п/п	*Потреба, що формує ринок*	*Цільова аудиторія (цільові сегменти ринку)*	*Відмінності у поведінці різних потенційних цільових груп клієнтів*	*Вимоги споживачів до товару*
	Базова потреба, яку задовольняє товар (згідно концепції потенційного товару)	Визначити потенційні цільові групи клієнтів, що можуть бути зацікавлені у задоволені	Вписати фактори, що формують поведінку клієнта (стандарти, технічні регламенти, інші фактори цінового	- до продукції - до компанії-постачальника

		означеної потреби	та нецінового характеру) та особливості купівлі та експлуатації товару	
1.	Проект дозволяє концептуалізувати можливі конструкції дронів	Будь-які групи які займаются проектуванням дронів, та мають необхідність у вирішенні нестандартних проблем при їх використанні	Регламентом використання товару є необхідність мати, підписку за яким він може використовувати цю технологію	Необхідними факторами для користувача є, лише заначення сегментів для створення конструкції, та завдання необхідне для вирішення

Загальне використання системи повинне бути безкоштовним, і лише монетизуватися завдяки підтримки. Тому, що воно є дослідженням задля знаходження можливих варіацій у конструкції дронів.

Але завжди існує можливість продавати продукт на використання, у вигляді системи підписки.

Таблиця 2.4 – Визначення ключових переваг концепції потенційного товару

№ п/п	*Потреба*	*Вигода, яку пропонує товар*	*Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити*
1.	Швидкість проектування конструкції	Завдяки автоматизації цей процес пришвидшується	Алгоритм може створювати конструкції, з багатомірного простору, не зважаючи на існуючі рішення. За цього може випливати дуже випадкові конструкції, до яких людина би не дійшла
2.	Автоматична перевірка конструкції	Перевірка на можливість створення	Дозволяє одразу перевіряти створені конструкції, без необхідності користувачу перевіряти її
3.	Оптимізація під різні завдання	Зменшення кількості необхідних деталей	Користувач задає необхідні параметри, і система знаходить конструкції які відповідають цим параметрам

2.3 Висновки по стартапу

Як було вказано раніше, ринкова комерціалізація проекту є можливою, і може бути введений у вигляді монетизації за якою клієнт отримає продукт на

користування онлайн, та не буде необхідності мати спец. обладнення. Але так зазвичай продукт є одноразовим (тобто, кліенту необхідно один раз створити конструкцію, і потім він буде її використовувати без додаткових інвестицій у товар), може виникнути проблема з пошуком нових клієнтів, тому більшу частину бюджету треба вкласти в рекламу продукту.

Тому впроваджувати будь яку монетизації немає необхідності, так як він являється більш експериментом задля вирішення проблеми можливого існування інших оптимальних конструкцій, задля деяких завдань з використанням повітряних дронів.

3 РОЗРОБКА РІШЕННЯ

3.1 Формулювання проблем при створенні системи

Розробка ефективних інструментів автоматизованого проектування для робототехніки, включаючи алгоритми пошуку та еволюційного проектування, є ключовим дослідним завданням. Ми розглядаємо кілька підходів, докладно описуючи різні переваги, які вони пропонують, та додатки, для яких вони найкраще підходять. Багато останніх підходів використовують глибоке навчання, яке надає потужні інструменти для автоматизованого проектування. Розроблено модульну стратегію коеволюції, в якій набір примітивних агентів навчається динамічно самозбиратися в складове тіло, одночасно навчаючись координувати свою поведінку для керування тілом. Однак їх метод призводить тільки до простих робіт з кількома компонентами і передбачає фізично неправдоподібну реконфігурованість. Навчання з підкріпленням також застосовується для постійного вдосконалення агентів. У їх системі проектування довкілля і конструкції змінюються, щоб агент міг навчатися ефективніше.

Методи глибокого навчання особливо підходять для спільної оптимізації, тіла робота та контролера, та мають потенціал для покращення продуктивності. У сфері роботів, також є приклад наскрізного проектування контролера та проектування структури з використанням глибоких латентних уявлень. Формальні методи проектування є альтернативним підходом, який використовує підхід на основі моделей для проектування роботів та їх контролерів. [7] демонструють формальний підхід, використовуючи велику бібліотеку конструкторів та спеціальні інструменти для конфігурування модульних роботів. На сайті конфігурації модульних роботів особливо добре підходять для цих методів [8]. Такі методи можуть вимагати значної ручної праці, не масштабуються та не узагальнюються добре.

3.2 Графова граматика

У роботі ми розглядаємо рекурсивні граматики графів (ГГ) як метод для генерування конструкцій, які обмежені здатністю використовувати конструкцію із заданих користувачем компонентами. Формальні граматики, набори правил створення допустимих рядків з алфавіту мови, широко використовуються в лінгвістиці та обробці природної мови. Це поняття може бути поширене на графові граматики, які визначають множини допустимих графів, а не лінійні послідовності. Для цілей автоматизованного проектування графів зазвичай описують просторові конфігурації механічних компонентів. Ранні приклади включають граматику графів для епіциклічних систем зубчастих передач та машин на основі Meccano. У робототехніці графові граматики були застосовані для моделювання самозбирання робототехнічних систем, а також структури взаємодій роботів та законів управління. Граматика графів може також використовуватись для опису фізичної структури, а також процесу виготовлення меблів, або годинників (Рис. 3.1).

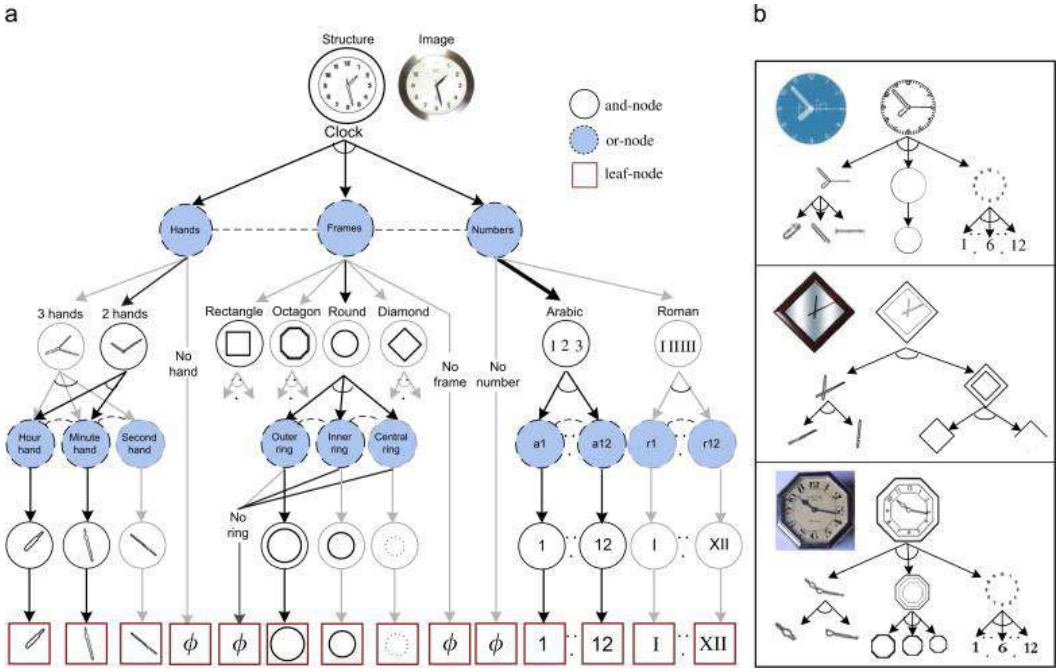


Рис. 3.1 – Приклад використання графової граматики для створення/розпізнавання стрілочних годинників

Ця граматика дозволяє створювати складні 3D-моделі меблів, які можуть бути виражені у вигляді деталей і з'єднувачів, які можна виготовити. Масштабованість та застосовність до фізичних систем є ключовими перевагами графових граматик. Вони забезпечують спосіб параметризації простору пошуку для запобігання нерозв'язності, в той же час дозволяючи при цьому з'являтися багатьом різним структурам.

Найбільш подібною до нашої роботи є робота [9], в якій описує пасивних динамічних брашируючих роботів за допомогою граматики графів та оцінює їх за допомогою динамічного моделювання. Їх робота показала потужність графових граматик в автоматизованому проектуванні роботів, з широкий спектрів роботів, що виникають з відносно обмеженої граматики. В їх роботі не розглядався дорогий аспект синтезу управління (а також пов'язану з ним проблему кооптимізації) і зосереджувалась виключно на 2D системах з маятниковим рухом. У нашому сценарії моделювання та оцінка конструкцій є набагато складнішою і вимагає більш масштабованого алгоритму пошуку.

3.2.1 Визначення базових сегментів задля генерації конструкцій

Для виконання створення конструкцій дронів, необхідно сформолювати основні сегменти які будуть використовуватись при створенні конструкції.

Таблиця 3.1 – Список сетментів вискористанних для конструювання дронів

	Тіло	Це початковий сегмент, який використовується як основа дрона, до нього буде приєднуватись інші компоненти
	Кінцівка	Це з’єднання яке може об’єднувати декілька сегментів разом, для продовження ланцюга конструкції

	Коліно (фіксоване з’єднання)	Буде виконувати роль з’єднання з декількома точками з’єднань. Це додасть до системи більше простору можливих конструкцій.
	Обертач (з’єднання двигуном)	Дозволяє сегменту обертатись по осі, при цьому динамічно змінювати конструкцію.
	Пропелер	Найважливіша частина дрона, яка дозволяє штовхати дрон у напрямку заданому пропеллером.

Усі сегменти можуть мати (>= 1) точок з'єднання (гнізда), за якими інші сегменти можуть буди приєднані до нього. Кожна з цих точок має свій унікальний індекс на сегменті.

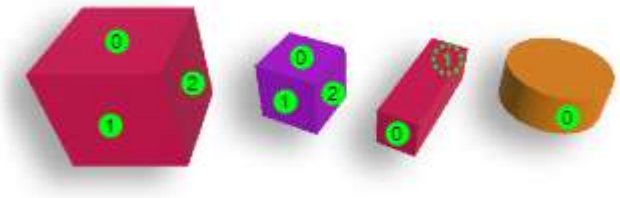


Рис. 3.2 – Точки з'єднань деяких сегментів, та їх індеки

3.2.2 Створення конструкцій дронів на основі графової граматики

При створенні конструкцій дронів, сегментам які будуть використовуватись для побудови дрона, треба надати розуміння яким чином кожен з них може бути з'єднаний один з одним. Для цього пропонується система правил з'єднань сегментів основаної на мові графів DOT. За допомогою DOT можна побудувати та описати будь який граф, або систему графів. Крім того, одним з великих переваг цьої мови полягає у тому, що вона є досить простою для читання, та написання людиною, та використанню для комп'ютера. Це нам стане до руки, бо як опису сегментів конструкції, так

і написанням правил з'єднань цих сегментів, буде займатись користувач, він зможе редагувати їх так, як йому заманеться без значних зусиль.

Приклад графа з двома листками написаний на мові DOT має наступний вигляд

```
digraph G
{
     size="2";
     a [label="Foo"];
     b [shape=box];
     a -> b -> c [color=blue]
     b -> d [style=dotted];
}
```

При відображенні коду ми отримаємо такий граф, представлений на рис. 3.3.

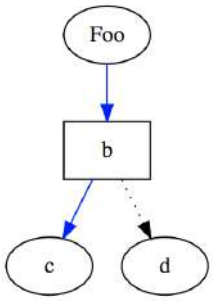


Рис. 3.3 – Приклад графу на мові DOT, з приведеного коду

Як бачимо з рис. 3.3 граф має корінь, однин вузол, та два листка. До з’єднання між вузлами як і до самих вузлів можливо додавання додаткових параметрів, які у виділяються у квадратних дужках. У приведеному приклади задані параметри змінюють колір з’єднання, та форму вузла. Але будь які параметри можуть буди задані користувачем. При виконанні роботи, використання цих параметрів необхідне для надання можливості керувати яким чином кожен з сегментів може буде з'єднаний з іншим. Для цього треба задати загальні правила з’єднань сегментів.

Таким чином для створення правил необхідно задати усі можливі сегменти, та усі можливі правила з'єднання цих сегментів, до кожного з можливих індексів. Також на додаток до цього кожен з сегментів може мати додаткові параметри, які необхідні нам при створенні конструкції.

У роботі виділено основний параметр для усіх сегментів - це маса цієї деталі.

Враховуючі усі приведені вище вимоги до правил, ми отримуємо такий список:

- Перелік усіх сегментів;
- Маса усіх сегментів;
- Перелік усіх можливих з’єднань між сегментами.

Таким чином ми отримуємо файл у усіма правилами, яка дозволяє досягти комплексних конструкцій, та мати великий простір для створення різноманітних дронів. Діаграма цього файлу приведена на Рис. 3.4, де зліва приведена спрощена діаграма того як кожен з сегментів може з'єднуватись з іншим, а справа - повна діаграма усіх сегментів, які присутні у роботі (діаграма не відображає індекси цих з'єднань, а лише повторює його декілька разів).

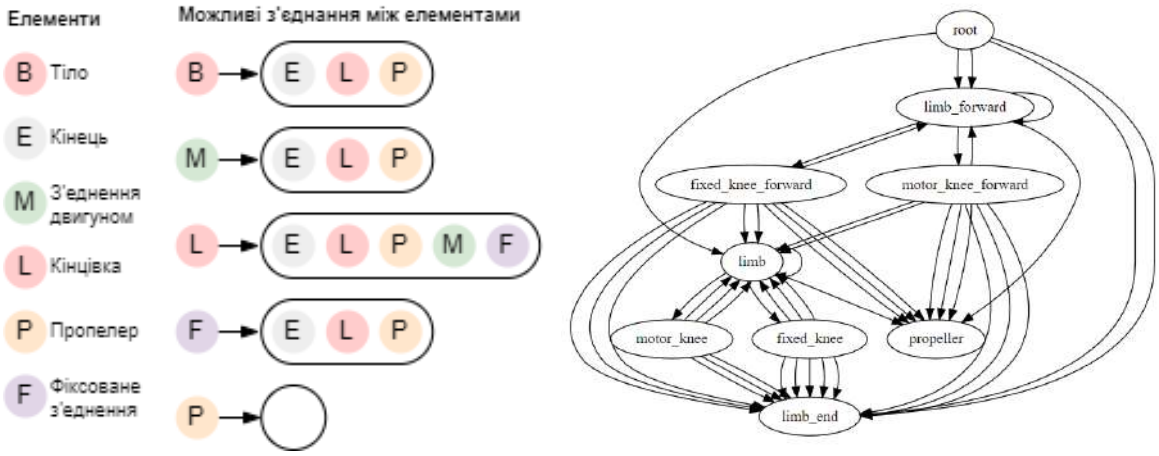


Рис. 3.4 – Діаграма правил конструкції дрона

Алгоритм створення конструкції базується на рекурсивному підході. Тобто спочатку створюється основа або тіло дрона. Воно являє собою корінь при генерації. З кожного його можливого з’єднання обирається випадковий сегмент, та додається до конструкції. Потім з кожного із створених сегментів знову оирається випадковий сегмент (якщо такий є), та додається до дрону. Цей процес продовжується до того, поки усі сегменти дрона не стануть листками (тобто без можливих з’єднань до них). Але так, як рекурсивні

алгоритми, можуть створити нескінченний граф і постійно обирати випадкові сегменти з декількома з’єднаннями, була введена максимальна глибина графу, яка керує максимальною довжиною ланцюга з’єднань. Якщо довжина ланцюга з’єднань перевищує задане значення, випадковий сегмент заміняється на сегмент пропеллера.

Псевдокод алгоритму генератора конструкції:

```
Вхід: Поточний сегмент k, кількість можливих індексів з’єднань поточного
сегменту I, глибина графу d, усі можливі сегменти M
Вихід: Сгенерована конструкція дрона
Программа: GenerateDrone()
        b = сегмент тіла з списку M
        GeneratePart(b, 0)

Программа: GeneratePart(k, d)
        For i ← 0 to I do
                E = усі сегменти які можуть бути з′єднані з k, по індексу i
                If E порожній:
                        Пропуск усього далі
                p = елемент пропеллеру, якщо такий є у E
                If d ≥ 5 та p існує:
                        Додати до графу конструкції елемент p
                Else:
                        Додати до графу конструкції випадковий елемент з E
                GeneratePart(k, d + 1)
```

Таким чином з’єднуючі сегменти за правилами можемо отримати будь яку комплексну конструкцію дрона. На рис. 3.5 показано приклад згенерованого дрона (зліва), та його графової граматики (справа). Для забезпечення симетрії у дрона та оптимізації обрахування після додавання вузлів для з'єднувачів з обох боків тіла обидва кінцівки однієї пари визначаються в одній гілці графа. Це можна побачити на відгалуженні правого графа рис. 3.5Рис. 3.5, яке йде від тіла і до кінцівки.

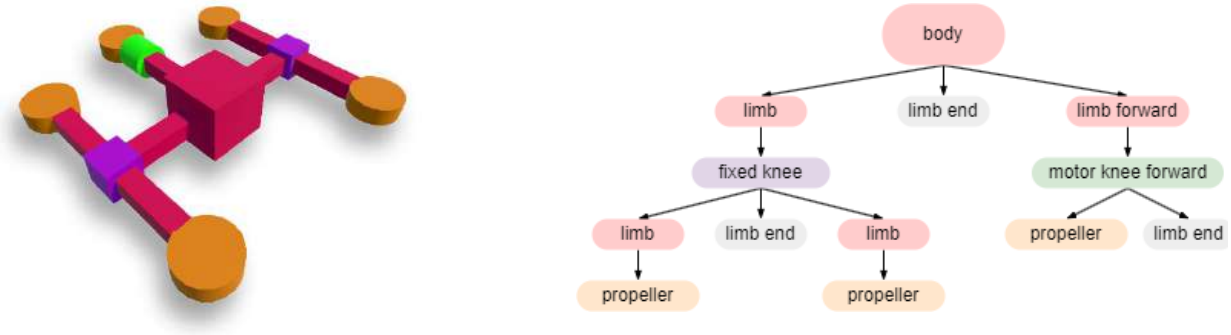


Рис. 3.5 – Конструкція дрона і репрезентуюча його графова граматика

3.3 Симуляція конструкції дронів у віртуальному оточенні

Створення алгоритму пошуку конструкції вимагає метод, за яким можна було б перевіряти якість, з якою дрон виконує поставлене йому завдання. Для досягнення цієї цілі необхідно створити фізичну симуляцію, де конструкція буде сформована за допомогою динаміки жорсткого тіла, де сегменти будуть зчленованими жорсткими тілами.

Для цього буде використовуватись симуляція на основі навчання з підкріпленням, де агент (тобто та сутність, яка навчається виконувати якусь задачу) може навчатись виконувати поставлену йому задачу, завдяки правилам нагороди. Загалом агент, який виконує навчання з підкріпленням, здатний сприймати та інтерпретувати навколишнє середовище, виконувати дії та навчатися шляхом спроб і помилок.

У навчанні з підкріпленням розробники розробляють метод заохочення бажаної поведінки та покарання за негативну поведінку (тобто reward policy). Цей метод присвоює позитивні значення бажаним діям, щоб заохотити агента, і негативні значення небажаній поведінці. Це програмує агента шукати довгострокову і максимальну загальну винагороду для досягнення оптимального рішення. Ці довгострокові цілі допомагають агенту не зациклюватися на менших цілях. З часом агент вчиться уникати негативу і прагнути до позитиву.

Наведені вище результати були відтворенні у середовищі Unity, але незважаючи на те, що у бібліотеці «ml-agents» є свої інструменти для

навчання агентів, вони не зможуть бути використанні для алгоритму пошуку конструкції, бо навчанні агента без якогось фідбеку, та ще увесь процес навчання будуть відбуватися завдяки інструментам на основі Python. Тому, для цього були використанні інші інструменти з тієї ж бібліотеки «ml-agents», але для інтеграції керування оточенням Unity, вже через Python.

Однією з проблем цього методу полягає у тому, що ми не маємо можливості передавати між оточенням будь які данні, а тільки обмежений список типів (рядки, цілі значення, значення з плавучою крапкою). Для вирішення питання передачі графу конструкції між оточеннями, було введений алгоритм перетворення графа у список значень. Де кожне значення, репрезентує номер за списком сегмента у файлі правил. Таким чином отримаємо послідовність значень, кожна з яких є індексом сегмента, так як ми завжди можемо отримати з значення сам сегмент взявши його з файлу правил (при умові, що файл не був змінений), ми завжди будемо отримувати один, та той самий граф при реконструкції.

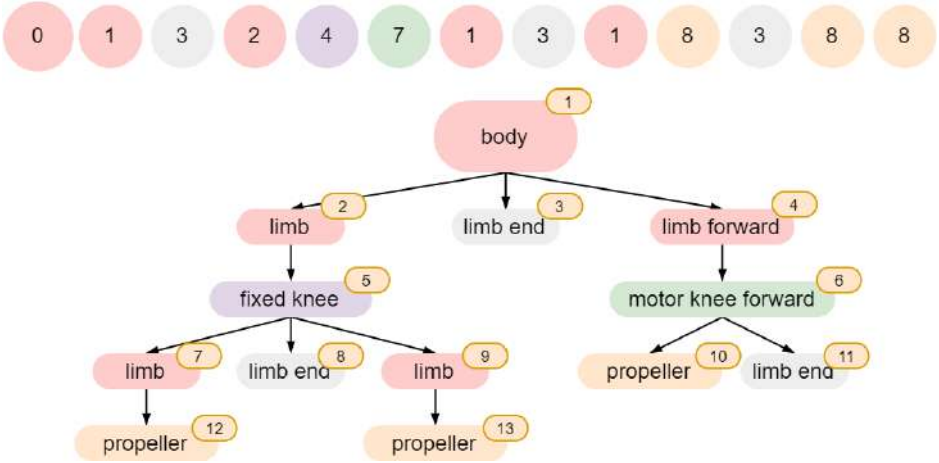


Рис. 3.6 – Алгоритм конвертації (розгортки) списку сегментів у граф

При виконанні симуляції задля навчання агентів можуть бути використанні різноманітні інструменти. Наприклад, робота яка була взята за основу використовувала самостійно побудовану симуляцію, на основі фізичного бібліотеки Bullet, яка дозволяє створювати оточення з жорсткими тілами та симулювати їх роботу. Хоча він являє собою потужний інструмент для тестування агентів з жорстким тілом, але будучи саме бібліотекою, для

його роботи буде потрібно багато додаткових власних налаштувань зі сторони C++ коду, для повноцінної роботи. В даний момент за вимог часу відведеного на виконання цієї роботи, було взято альтернативний варіант у вигляді рушія Unity, який є оточенням для створення ірог, але також з недавнього часу отримав підтримку навчання агентів «ML-Agents». Це дозволяє створювати та редагувати агентів набагато швидше, ніж з використанням Bullet.

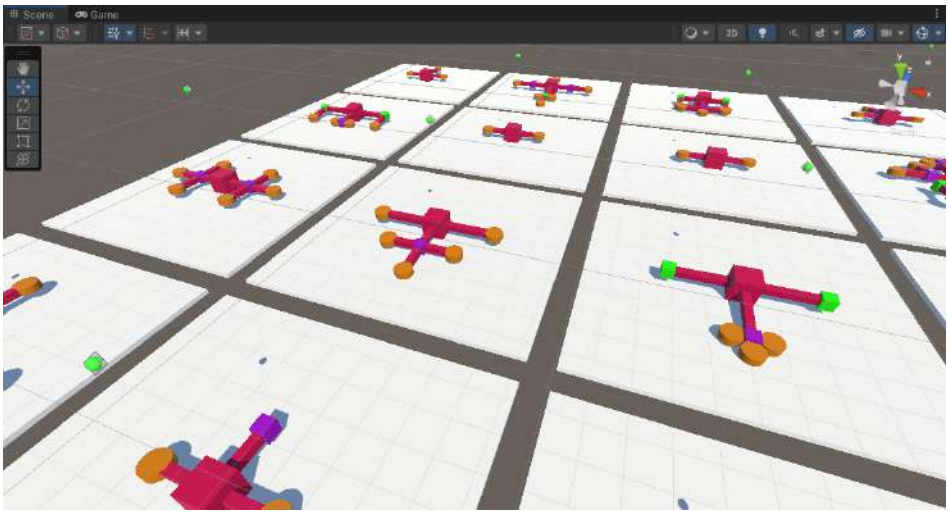


Рис. 3.7 – Виконання алгоритму випадкової генерації конструкцій дронів

Проблема графового методу генерації конструкцій у тому, що він може утворювати некоректні конструкції, які не можуть буди створенні у реальному світі, навіть з усіма нюансами. Однією з таких проблем є перетин сегментів один з одним (самозіштовхування). Це трапляється тому, що граф конструкції який подається на тестування у симуляції, немає розуміння коли один з його створених вузлів може перетинатись з іншим, поки не буде виконана симуляція, бо, ще поки немає репрезентуючої його конструкції, а є лише план побудування у вигляді графа. Одна з таких конструкцій зображена на Рис. 3.8, у цього дрона сегмент пропеллера перетинається з кінцівкою, тому це не є дійсною конструкцією.

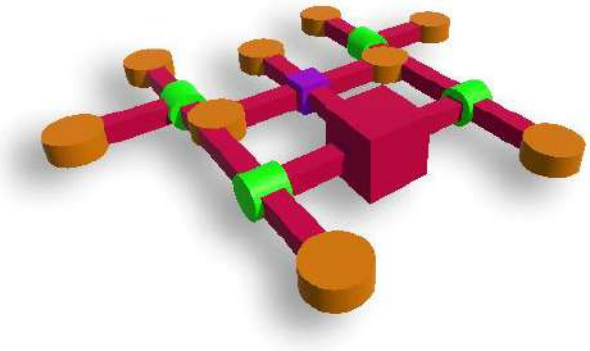


Рис. 3.8 – Приклад некоректної конструкції (з перетином самого себе).

Задля вирішення цієї проблеми, конструкції необхідно проходити додатковий етап валідації конструкції. Він виконується у симуляції, де створюється конструкція, та сегменти перевіряються на перетин один з одним. І якщо, ця перевірка була хибною, то процес далі не продовжується, та генерується наступна конструкція.

3.4 Тестування конструкції дрона на оточенні, та політика винагороди

Для перевірки наскільки згенерована конструкція може виконати поставлене завдання, необхідно отримати результат її роботи у оточенні. Для цього виконується навчання з підкріпленням, яке випадковими кроками шукає оптимальні параметри для виконання агентом завдання. Результатом її роботи буде значення у виді відношення кількості кроків до винагороди. Навчання з підкріпленням буде підбирати параметри для максимізації винагороди.

За допомогою навчання з підкріпленням ми можемо легко реалізувати функцію вартості, запустити на ній градієнтний спуск і бути впевненими, що отримаємо відмінні результати з відносно невеликим налаштуванням гіперпараметрів. Шлях до успіху в навчанні з підкріпленням не такий очевидний – алгоритми мають багато рухомих частин, які важко налагоджувати, і вони вимагають значних зусиль в налаштуванні для отримання хороших результатів.

Тому для вирішення цієї задачі було обрано Proximal Policy Optimization (PPO), який досягає балансу між простотою реалізації, складністю вибірки та простотою налаштування, намагаючись обчислити оновлення на кожному кроці, яке мінімізує функцію витрат, забезпечуючи при цьому відносно невелике відхилення від попередньої політики.

$$L^{CLIP}(\theta) = \widehat{\mathrm{E}}_t\left[\min(r_t(\theta)\widehat{\mathrm{A}}_t, clip(r_t(\theta), 1 - \varepsilon, 1 + \varepsilon)\widehat{\mathrm{A}}_t)\right], \quad (3.1)$$

де θ – параметр політики; $\widehat{\mathrm{E}}_t$ – позначає емпіричне очікування на часових інтервалах; r_t – відношення ймовірностей за нової та старої політики відповідно; $\widehat{\mathrm{A}}_t$ – оціночна перевага в момент часу t; ε – гіперпараметр, зазвичай 0,1 або 0,2.

Ця задача реалізує спосіб оновлення Trust Region, сумісний зі стохастичним градієнтним спуском, і спрощує алгоритм за рахунок усунення штрафу KL та необхідності робити адаптивні оновлення. У тестах цей алгоритм показав найкращу продуктивність на задачах безперервного керування та майже відповідає продуктивності ACER на Atari, незважаючи на те, що є набагато простішим у реалізації.

Загальне проектування правил для дрона буде залежить від необхідної задачі. Нехай, першим завданням дрона буде просте паріння на місці, у відповідній точці у симуляції. Для цього нам потрібні наступні правила винагороди:

$$\delta_v = min(dist(v_c, v_t), s_{match}), \quad (3.2)$$

$$r_v = \left(1 - \left(\frac{\delta_v}{s_{match}}\right)^2\right)^2, \quad (3.3)$$

$$R = r_v - r_a * 0.2 - v_c * 0.05 - v_a * 0.05, \quad (3.4)$$

де v_c – поточна швидкість; v_t – цільова швидкість у напрямі точки; s_{match} – максимальна швидкість з якою дрон може рухатись; r_a – правило балансу (кут нахилу дрона відносно горизонту); v_a – кутова швидкість дрона.

Основним правило для виконання цієї задачі є необхідність агента рухатись у напрямі цільової точки. Для цього достатньо щоб вектор поточного напряму агента, дорівнювала вектору напряму від агента до

цільової точки. Таким чином вектор швидкості агента буде намагатися відповідати напряму у якому агенту необхідно рухатись, та дозволить досягти цілі. Також нам потрібні дочірні правила, для пришвидшення процесу навчання, а саме, правило балансу за яким агент буде отримувати негативну винагороду, за той час коли він не знаходиться горизонтально рівно. А також надання невеликої негативної винагороди, коли агент рухається, у будь якому напрямі, що дозволить стабілізувати його рух.

Останнє, що залишилось, це проекспериментувати з різними параметрами алгоритму PPO, та політики правил винагороди для зменшення часу навчання агента.

У експерименті буде використовуватись одна і та сама, конструкція дрона.

Рис. 3.9 – Тестування різних політик у агента та параметрів у PPO

На графіку «ep_len_mean» з Рис. 3.9 можна побичити різні параметри протестовані на завданні паріння на місці, та кількість іттерацій до вирішення завдання. На початку єкспериментів, агент вирішував завдання у 1 мільйон шагів, але після підбору параметрів, це значення зменшилось до 200 тисяч шагів. Усі змінені параметри можна побачити списком зліва на Рис. 3.9

3.5 Алгоритм пошуку конструкції для вирішення задачі

На Рис. 3.11 представлено умовний приклад графу конструкцій дронів та їхня оцінка ефективності виконання поставленої задачі. Кожен сегмент цього Використовуючи цей граф, алгоритм може пришвидшити пошук більш відповідної конструкції, а не генерувати випадкові. Цей процес відбувається завдяки алгоритму евристичного пошуку по графу Graph Heuristic Search (GHS) [10], який узагальнює знання з досліджених зразків на неперевірені конструкції, підвищуючи ефективність пошуку. Аналізуючи оцінку ефективності, евристика може спрямовувати пошук, допомагаючи швидше знаходити оптимальні рішення. Для цього застосовується глибоке навчання, а саме графові нейронні мережі (GNN). GNN особливо добре підходять для навчання на топологічних або структурованих вхідних даних, таких як конструкції роботів [10].

Основною увагою у алгоритмі MCTS полягає у пошуку найкращих дій при розширенні дерева дій, для максимізації результату. Застосовуючи MCTS для ігор, алгоритм виконує безліч розгортань, при яких відбувається відтворення багатьох ігор. У кожному з цих відтворень, алгоритм проходить гру до кінця випадковими вузлами, де кожен вузол - це якась дія у грі, поки не отримує фінальний результат. Цей результат використовується алгоритмом для задання ваги усім пройденим вузлам, які допоможуть обирати кращі результати у наступних розгортуваннях. Кожен прохід алгоритму Монте-Карло відбувається у декілька етапів:

Вибір (Selection): Спочатку обирається вузол кореня a_1 (поточний стан гри), та обходяться його дочірні вузли, поки не буде досягнуто листового вузла (дія яка ще не була просимульована).

Розширення (Expansion): Якщо вузол не є кінцем гри, то алгоритм обирає один з дочірніх вузлів, від вузла листка.

Симуляція (Simulation): Відтворювати випадкові дії від листового вузла, поки гра не закінчиться Q.

Зворотне поширення (Backpropagation): Отримані результати гри після випадкових дій, використати для оновлення вузлів від кінця гри, до листового вузла.

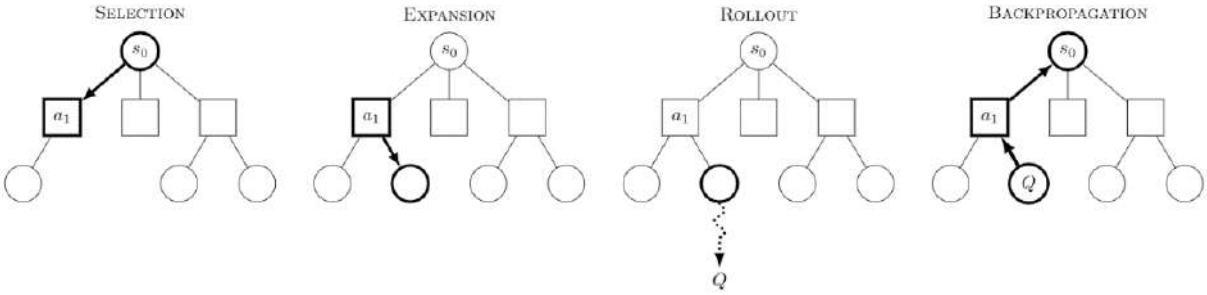


Рис. 3.10 – Дерево пошуку Монте-Карло

При обері рекомендуется брати вузол для якого:

$$\frac{w_i}{n_i} + c\sqrt{\frac{\ln N_i}{n_i}}, \tag{3.5}$$

w_i – кількість виграшів для вузла після і-го ходу; n_i – кількість симуляцій для вузла після і-го переміщення; N_i – загальна кількість симуляцій батьківського вузла після і-го переміщення; c – параметр розвідки, теоретично рівний $\sqrt{2}$; на практиці зазвичай вибирається емпірично.

У нашому випадку ми не маємо дій для симуляції, але ми можемо піти іншим шляхом. Використовуючи сегменти дрона, як вузли для алгоритму, можна проводити симуляцію кожної конструкції та залежно від результатів проводити зворотне поширення.

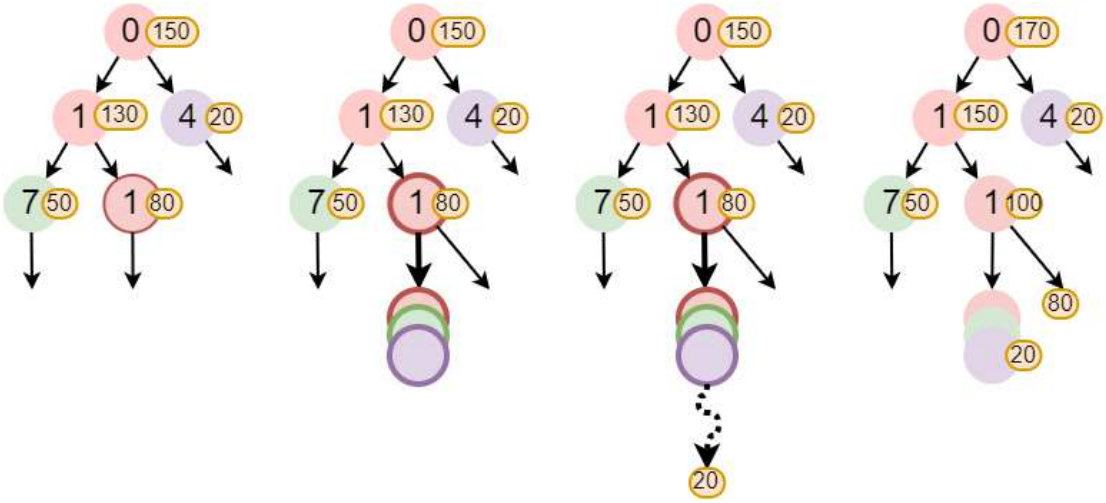


Рис. 3.11 – Граф конструкцій дронів з оцінками ефективності виконання поставленої задачі

Для уникнення пошуку лише по одному напряму, де він може впасти у «локальний мінімум», при пошуку більш відповідної конструкції у додаток до GHS використовується метод E-greedy search. Це простий метод збалансування розвідки та експлуатації, шляхом випадкового вибору між розвідкою та експлуатацією, який описується виразом

$$A_t = \begin{cases} argmax_{\ a}\ Q_t(a), \text{ з ймовірністю } 1-\varepsilon \\ a, \text{ з ймовірністю } \varepsilon \end{cases}, \quad (3.6)$$

де A_t – дія в момент часу t; $Q_t(a)$ – таблиця пари станів та дій у момент часу t; ε – ймовірність обрати випадкову дію.

4 СИМУЛЯЦІЯ ТА РЕЗУЛЬТАТИ РОБОТИ

4.1 Результати симуляції

Отриманими результатами є система для створення дронів, у якій вхідні данні у систему – це набір, сегментів, з'єднань та необхідного до вирішення завдання. Результатом роботи системи є згенеровані через графічну граматику безліч різноманітних конструкцій дронів з компонентів на вході, які підходять для вирішення поставленого завдання (рис. 4.1).

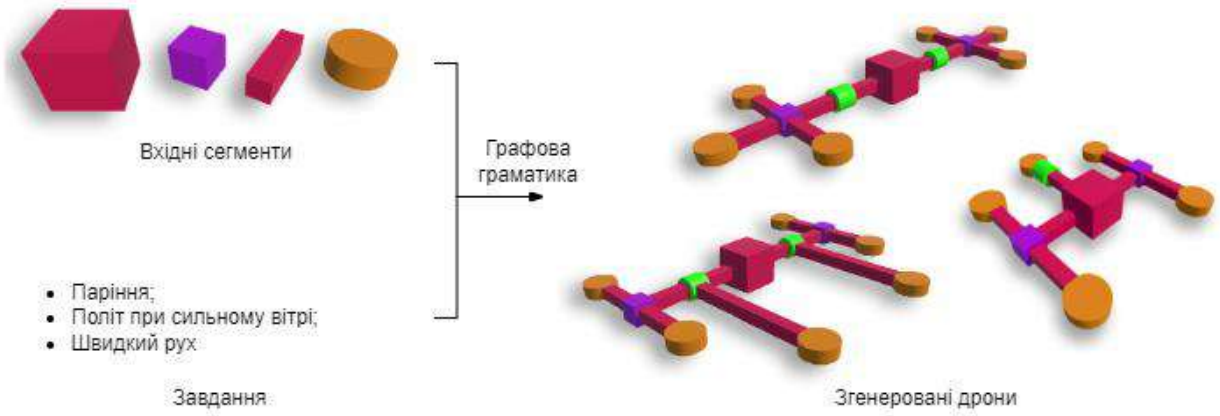


Рис. 4.1 - Згенеровані дрони за допомогою графової граматики

Перевиремо роботу системи на завданні паріння на місці. Так як воно є досить простим, ми повинні отримати багато рішень у короткий час.

Для пришвидшення пошуку конструкцій, були додані правила, фільтри та параметри за якими можна вважати конструкцію працездатною, або навпаки не робочою.

Першим і необхідним параметром який використовуется при навчанні будь якого ML-агента, це максимальна кількість кроків для симуляції. Вона необхідна для обмеження часу навчання однієї спроби у симуляції. Після досягнення цього значення, спроба симуляції завершується і починається наступна. Наразі обмеження задано як 200 кроків.

Також один з необхідних параметрів - це максимальна кількість кроків навчання однієї конструкції. Цей параметр необхідний для обмеження часу навчання, по усім спробам симуляціям для однієї конструкції.

Для самого агенту були введені деякі правила по передчасному завершенню спроби симуляції, такі як:

- Торкання до землі. Вважається, що якщо дрон доторкнувся до землі, він не має розуміння як парити над землею, тому краще одразу переключитись на наступну спробу;
- Переворот більше ніж на 90°. Якщо дрон перевернувся більше за це значення, то для стабілізації, він використає забагато кроків, або просто зіткнеться з землею раніше;
- Виліт за межі зони симуляції. Дрон, який вилетів далі за межі симуляції, має некоректну поведінку для поточної задачі і такі принципи повинні ігноруватись при навчанні.

На рис. 4.2 представлено графіки результатів навчання різноманітних конструкцій дронів.

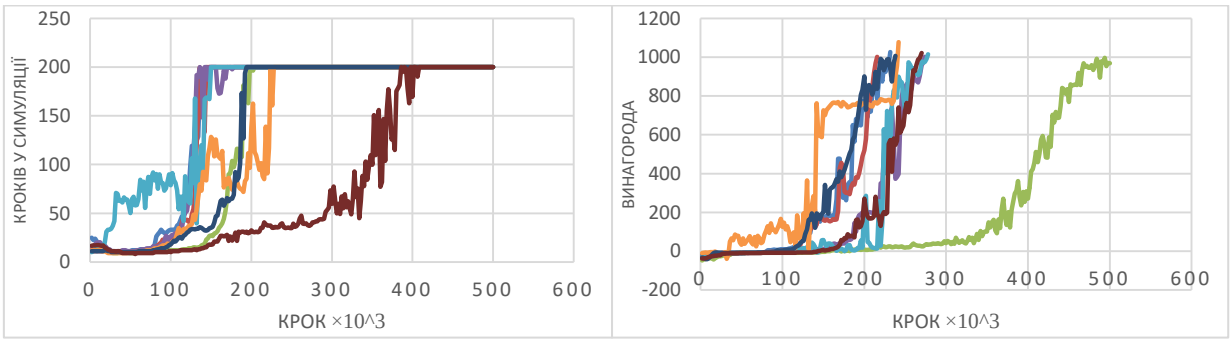


Рис. 4.2 – Представленні результати навчання деякої кількості дронів. Кожен графік описує свою конструкцію.

Кроків у симуляції – це кількість кроків для однієї спроби симуляції. Винагорода – це середня винагорода за однієї спроби симуляції.

Таким чином можна побачити, що більша кількість агентів досягає межі на $100 - 150 \times 10^3$ по кількості кроків на одну спробу симуляції, але це не є показником навчаності агента, а лише відображає, що дрон навчився знаходитись у повітрі достатньо часу, та не торкатись землі. Візуальні експерименти показали, що агент навчається у відповідності завданню саме після значення винагороди у межах 800-1000. Таким чином значення

максимальної кількості кроків на усі симуляції повинно буди у межах з $200 - 250 \times 10^3$.

Також з графіків на рис. 4.2 можна зробити обмеження на винагороду. Коли винагорода досягне =>1000, то можна вважати, що конструкція може бути використана для рішення завдання.

Якщо у 50 поколінь тестування конструкції у симуляції не буде збільшення винагороди, то ця конструкція відкидається.

На рис. 4.3 і рис. 4.4 представлені згенеровані конструкції дронів, які змогли вирішити завдання, та результати їх навчання.

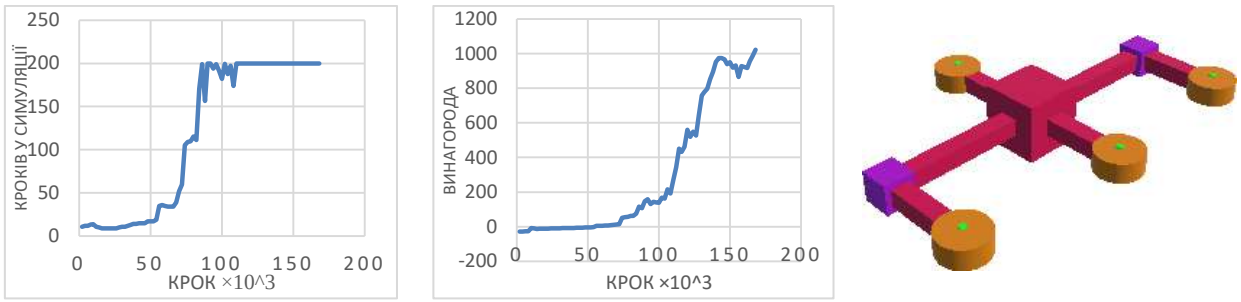


Рис. 4.3 – Успішна конструкція (час навчання – 2.5 хвилини)

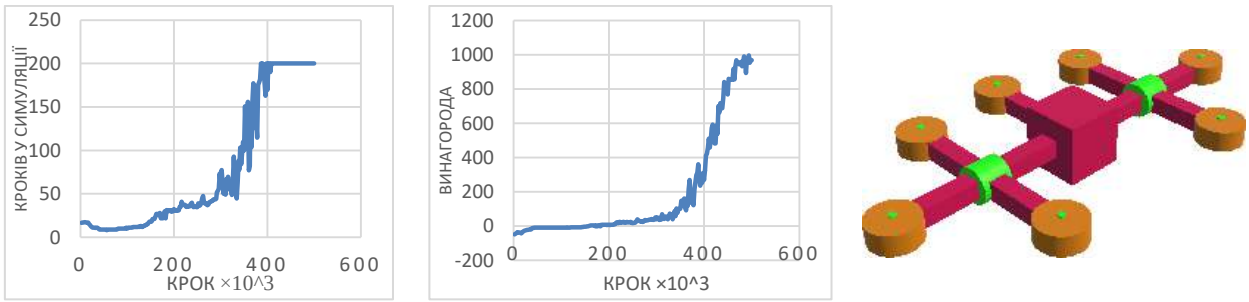


Рис. 4.4 - Успішна конструкція (час навчання – 8.6 хвилини)

На рис. 4.5 і рис. 4.6 наведені приклади конструкцій, які не змогли вирішити завдання, та симуляція яких завершилась значно раніше.

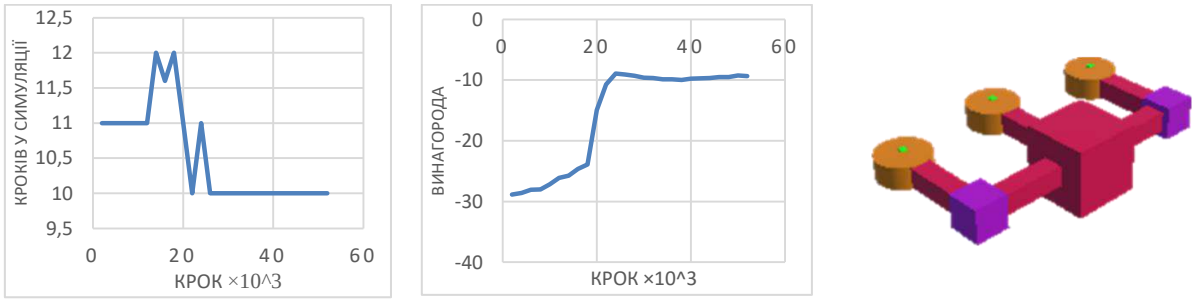


Рис. 4.5 - Неуспішна конструкція (час навчання – 0.5 хвилини)

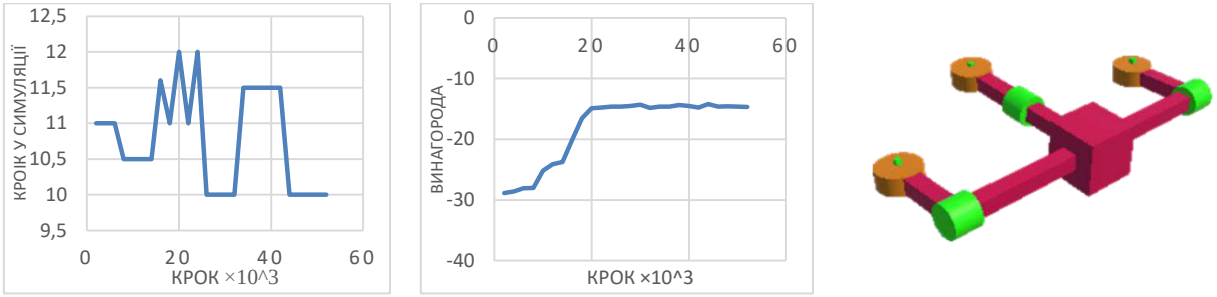


Рис. 4.6 - Неуспішна конструкція (час навчання – 0.3 хвилини)

Для визначення поточного часу навчання при тестуванні симуляції конструкцій усі вирахування були проведені на обчислювальній техніці з наступними характеристиками:

- Процесор: Intel core i9 – 20 ядер, 3.7 ГГц;
- Відеокарта: Nvidia GeForce 3070Ti 8-ГБ пям'яті;
- Пам'ять: 32 GB 3600 МГц DDR4;
- Диск: SSD Samsung 970 Evo Plus. Швидкість читання – 3500 МБ/с. Швидкість запису –2300 МБ/с.

Як видно з результатів, обробка кожної конструкції займає в середньому від 2 до 8 хвилин, в залежності від комплексності конструкції, та її працездатності. Таким чином для повного пошуку оптимальних конструкцій необхідно перевірити щонайменш 1000 варіантів конструкцій. Час, який необхідний для цього, буде вираховуватись як $1000 * \sim 5 = 5000$ хвилин, а це 83 години.

На рис. 4.7 представлено вигляд згенерованих конструкцій і значення винагороди, отримане для них. Як бачимо, система дуже швидко знайшла конструкції які впораються з завданням, але також ще вона спроектувала декілька неробочих конструкцій, та ті які не зможуть вирішити це завдання.

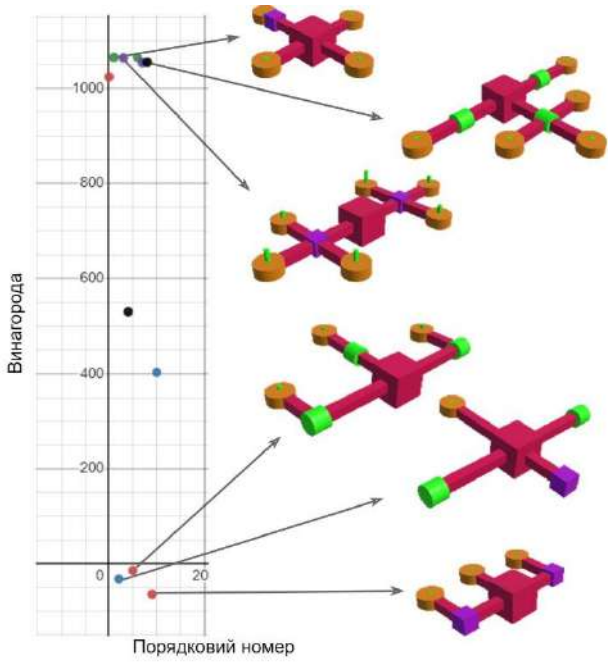


Рис. 4.7 – Результат роботи системи на завданні з парінням на місці

4.2 Порівняння результатів з стандартними конструкціями дронів

Проведені експерименти показали, що зазвичай система приходить до конструкції з декількома пропелерами, які знаходяться з кожної сторони від дрона (тобто звичайним квадрокоптерам). Це означає, що у більшості поставлених завдань дійсно достатньо використання звичайної конструкції дрона. Але для проведення більш детального аналізу, треба більше часу, та складніші завдання.

4.3 Обмеження та майбутні покращення

Приклад конструкцій, представлений в цій роботі, розглядає тільки двосторонню симетрію і типи з'єднань, які зазвичай використовують при побудові дронів. Але, це не означає що система підтримує тільки ці сегменти, тому відкривається великий потенціал для додавання нових сегментів у систему, та підвищення різноманітності конструкцій.

Ця система гарантує, що імітовані конструкції принаймні можуть бути виготовлені або можуть бути побудовані в їх імітованих конфігураціях. За допомогою відповідними правилами, граматики дронів забезпечують використання обмеженого набір сегментів і гарантують, що сегменти можуть

бути з'єднані між собою фізично. Забезпечення точної відповідності між змодельованими та фізичними конструкціями не розглядається в цій роботі в явному вигляді, однак, це залишається цікавим напрямком для майбутніх досліджень.

Але так, як у цій роботі не були проведені інші тести з іншими завданнями. Більш складні експерименти не можуть бути виконані у поточний час, за проблем з електроенергією, бо на їх виконання необхідно декілька днів роботи алгоритму. Тому ця можливість може бути винесена на доопрацювання та покращення системи.

Також мала кількість варіантів сегментів є значним обмеженням для створення більш великого простору можливих конструкцій. Додавання більш різноманітних сегментів може створити значне підвищення різноманітності результатів. Одними з таких покращень може бути:

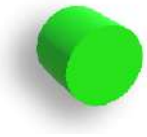
- Додавання пропеллерів направлених не тільки вгору, а ще і у інші напрямки;
- Можливості з’єднуватись коліну також зверху та знизу, а не лише під кутом у 90°;
- Зробити генерацію несиметричних конструкцій.

4.4 Вибір елементної бази на основі обраних сегментів

В таблиці 4.1 приведено опис сегментів з симуляції у відповідності до реальних елементів.

Таблиця 4.1 – Опис сегментів з симуляції до реальних елементів

	Тіло буде містити в собі усі схемотехнічні елементи, які необхідні для роботи з іншими елементами системи.
	З’єднанням буде виступати кронштейн який буде з’єднувати декілька сегментів разом.

	Сегмент коліна нам не потрібен, так як ми можемо використати декілька кронштейнів, які об’єднані у один.
	У ролі обертача буде виступати сервопривід, який буде дозволяти обертати сегменти, що приєднані до цього сегменту.
	Безколекторний двигун з пропелером, які обиратимуться на основі результатів симуляції.

Хоча згенеровані конструкції являються сегментними, тобто тіло, та кожне з’єднання є окремим сегментом, при створенні реальної моделі у випадках де не використовується обертач можна зробити усю конструкцію суцільним блоком.

Для побудови конструкції згенерованої алгоритмом, як приклад, можна взяти схему звичайного квадрокоптера (рис. 4.8), та використати елементу базу для побудови будь яких конструкцій.

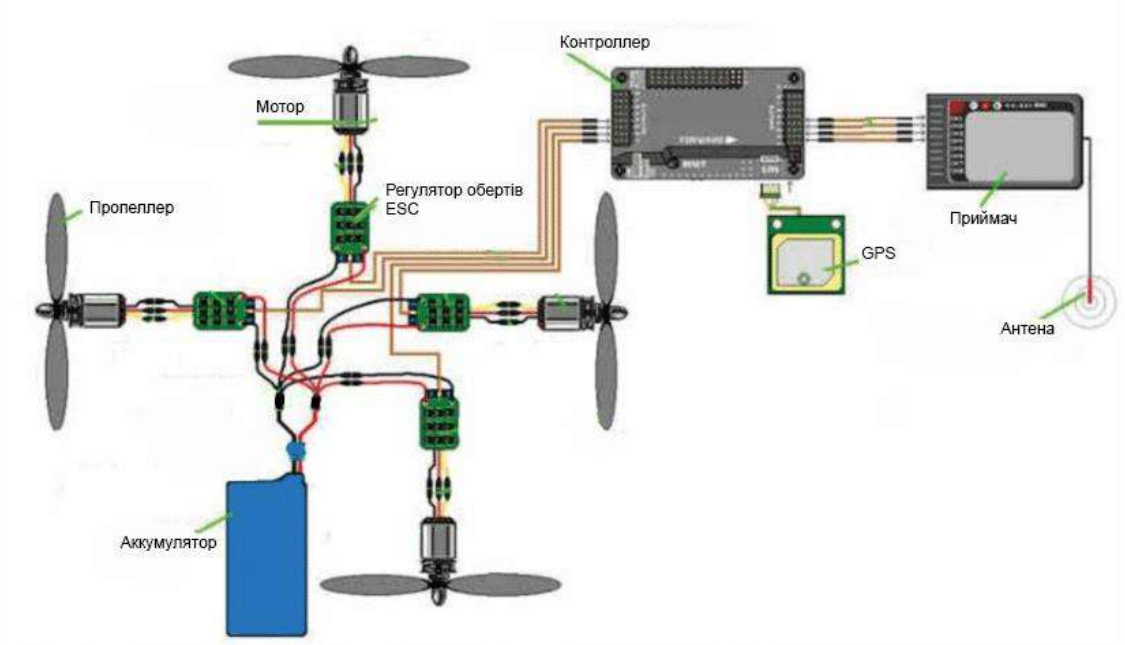


Рис. 4.8 – Схема квадрокоптера

Мотор для даної конструкції дрона може бути моделі BR2204 2300KV CW/CCW (рис. 4.9). BR - Brushless (з англ. безщітковий, тобто безколекторний); 2204 – діаметр і висота статора, відповідно 22 і 4 мм;

2300KV – це кількість обертів у хвилину; CW та CCW – обертається за годинниковою, та проти годинникової стрілки.

Рис. 4.9 – Двигуни для пропелерів BR2204 CW/CCW

Для вибору пропелера, який буде використовуватись при створенні реальної моделі, треба спочатку розрахувати його параметри у симуляції. Було створено конструкцію Рис. 4.10 за якою були виставленні такі параметри, щоб два пропелери змогли підняти цю вагу. Підібравши масу ми отримаємо, що максимальну вагу яку зможуть підняти ця конструкція становить 410г, це 205г ваги на один пропелер. Виходячи з цього нам необхідно підібрати схожий за параметром пропелер.

Рис. 4.10 – Конструкція для вираховування сили пропелера у симуляції

Один із важливих параметрів це KV. Мотори з високим KV ідеально підійдуть для швидкого обертання маленьких пропелерів, у той час, як мотори з низьким KV будуть легко обертати великі пропелери на великих дронах з високою вантажопідйомністю.

У світі існує безліч різних видів пропелерів для мультикоптерів, різняться вони як матеріалами, розмірами, видом кріплення та іншими характеристиками, розглянемо основні характеристики пропелерів для мультикоптера.

Діаметр і крок: як правило, вказуються безпосередньо на пропелері, наприклад 5030 (іноді вказується 5x3, що рівносильно), що означає діаметр

пропелера рівносильно), що означає діаметр пропелера 5 дюймів, і крок дорівнює 3.

Матеріал: для невеликих апаратів в основному використовують пластикові пропелери, вони недорогі, міцні і при цьому мають відмінні тягові характеристики.

Рис. 4.11 – Пропелери для дрона

Знайдені онлайн тести на стенді Рис. 4.12 з використанням даного двигуна, та під’єднаного до нього пропелера, показали що сила тяжіння у одного пропелера буде дорівнювати 1400 грам. Це означає, що обрані вище БК мотор з пропелером, може витягнути необхідну для нас вагу у 205 грам, на один пропелер.

Рис. 4.12 – Стенд для тесту сили тяжіння пропелера

Одним із мінусів БК двигуна є обов'язкове застосування електронних регуляторів обертів (далі ESC, з англ. electronic speed controller –електронний контролер швидкості) для запуску двигуна.

ESC можуть бути з вбудованим понижувальним регулятором напруги на 5 вольт, такі регулятори позначаються написом plush і можуть живити,

наприклад, радіоприймач, але на мультикоптерах вони практично не використовуються, створювалися вони, як правило, для авіа та автомоделей.

30A – Потужність ESC в амперах.

2-4S LIPO – рекомендована напруга літій полімерного акумулятора. Знаючи, що один елемент акумулятора (1S) має вольтаж 4.2 вольта, в повністю зарядженому стані, не складно порахувати діапазон напруг $2 * 4.2 - 4 * 4.2$, тобто від 8.4 до 16.8 вольт.

Рис. 4.13 – ESC для контролю двигунів.

Контролер уже містить у собі вібророзв'язку, тому його не потрібно ставити на демпферний майданчик, а потрібно кріпити прямо на корпус. Також у нього на борту є гіроскопічний, барометричний датчики, магнітометр, компас і додатково під'єднують модуль GPS, щоб апарат міг повернутися на точку зльоту сам, якщо втратить зв'язок з апаратурою управління, але це не єдина функція. До цього польотного контролера в комплекті йде модуль PMU і LED, а також може підключатися модуль Bluetooth і OSD телеметрія.

Рис. 4.14 – Польотний контролер DJI Naza M v2 DJI Naza M v2

Основні параметри які необхідно виділити з документації до контролера:

- Рекомендований передавач: 4 канали та більше з частотою 2,4 ГГц
- Рекомендований акумулятор: LiPo 2S-6S
- Підтримка регуляторів ходу: до 400 Гц

Рис. 4.15 – Futaba R3008SB приймач сигналу керування

Восьмиканальний приймач R3008SB 2.4GHz FHSS від Futaba з гарантованою якістю для правильного технічного обслуговування радіокерованих моделей. Основною особливістю ресивер є двонаправлений зв'язок з передавачем T-FHSS Air-2GHz, який використовує порт S.BUS2. Також на борту приймача встановлено два порти PWM і S.BUS.

Для забезпечення енергією сучасні мультироторні системи використовують переважно два види аккумуляторів.

- LiPo – Найпоширеніші, мають безліч форм і різних характеристик.
- Li-ion – Менш поширені, як правило використовуються для апаратів, розрахованих на довгий політ. Спаюються вручну з окремих елементів під конкретне завдання.

Виходячи з елементів обраних раніше, можемо використати АКБ ONBO 5200mAh 4S 45C Lipo якого буде достатньо для нашого дрона. Зовнішній вигляд обраного АКБ представлено на рис. 4.16.

Рис. 4.16 – АКБ ONBO 5200mAh 4S 45C Lipo

4.5 Канал радіозв'язку з дроном

Як вказано в описі, обраний приймач R3008SB працює на частоті 2.4GHz і використовує технологію FHSS для обміну даними з пультом керування. Технологія FHSS (з англ. frequency-hopping spread spectrum) широко використовується для побудови захищених каналів зв’язку [12, 13]. Її суть полягає у псевдовипадковій зміні робочої частоти по закону, який відомий і приймачеві і передавачеві. На рис. 4.17 представлено схему двоканального FHSS, на якому відображено залежність того, в який момент часу на якій частоті буде вестися передача даних.

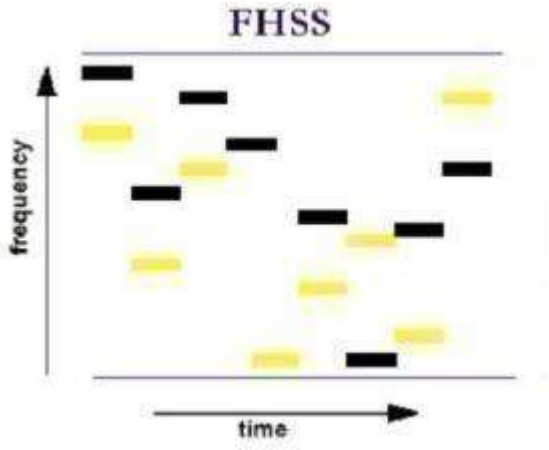


Рис. 4.17 – Схема двоканального FHSS

Перевагами технології FHSS є те, що передача може вестися в досить широкому діапазоні частот. Інформація передається послідовно порціями на різних піддіапазонах. Якщо якийсь із частотних піддіапазонів буде заглушено, передача продовжуватиметься на решті. Алгоритми обробки

дозволять виявити заглушені піддіапазони і виключити їх з закону зміни робочої частоти. Також слід відмітити, що для перехоплення інформації необхідно знати закон зміни робочої частоти.

Для передачі цифрових даних може бути використана частотна маніпуляція FSK. Приклад FSK сигналу і його спектр представлено на рис. 4.18.

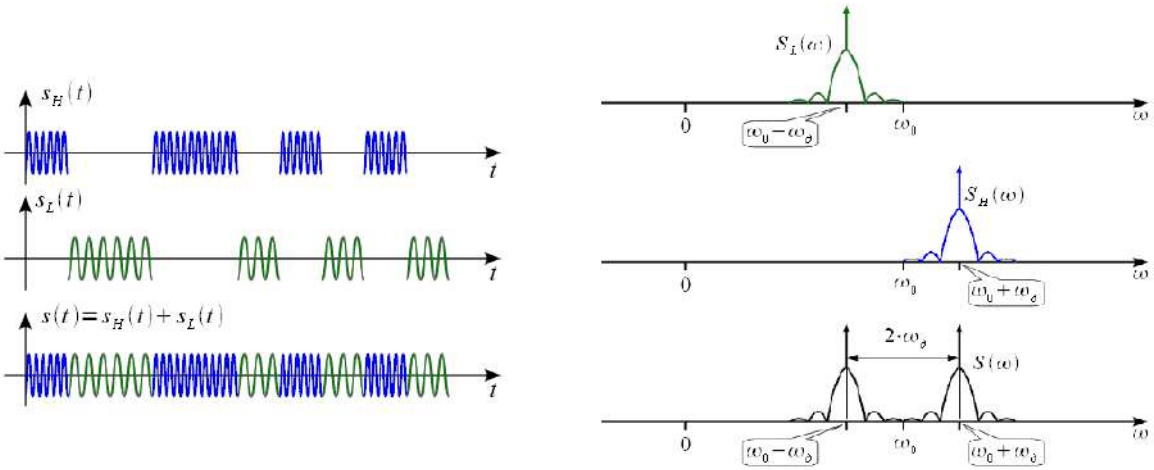


Рис. 4.18 – Сигнал FSK і його спектр

Приклад структурної схеми передавача сигналів з FSK модуляцією із застосуванням технології FHSS представлено на рис. 4.19.

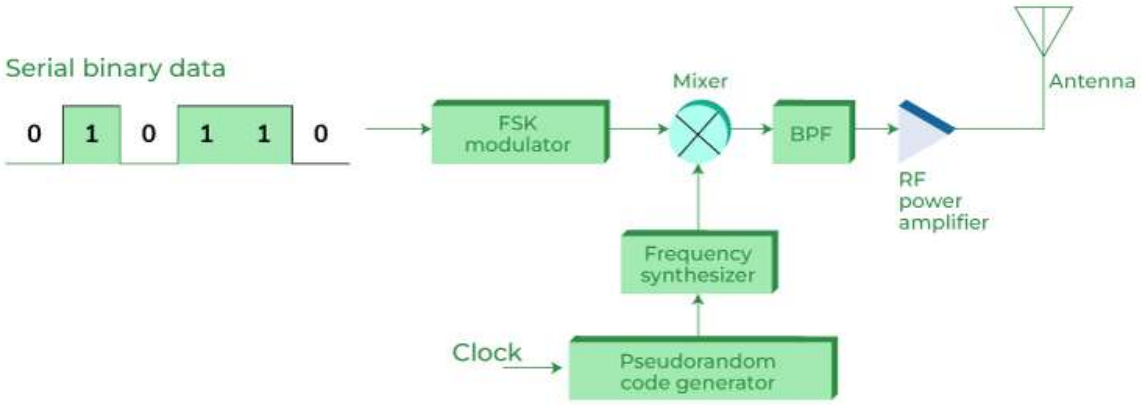


Рис. 4.19 – Структурна схема передавача FSK FHSS

До недоліків FHSS можна віднести низьку швидкість передачі даних, що зумовлюється затримкою при кожній зміні робочої частоти. Однак це вирішується шляхом застосування широкосмугових сигналів OFDM [14-20].

ВИСНОВКИ

Інтелектуальні та ефективні методи генеративного проектування роботів визначатимуть майбутнє процесів проектування у робототехніці. У роботі представлений метод, який використовує декілька різних методів для ефективного та масштабного дослідження конструкцій роботів. Розробляючи граматику графів, яка дозволяє генерувати широкий спектр конструкцій, робота показала, що творчі рішення можуть з'являтися у відповідь на різні завдання. Важливо те, що створенні конструкції можуть бути фізично виготовленні, і існує значний потенціал для того, щоб їх можна було втілити в реальні сценарії та завдання.

Як показали поточні експерименти, на завданні паріння на місці більшість згенерованих конструкцій являються дуже схожими на базові конструкції типу «квадрокоптер». Таким чином можна вважати, що для цього завдання конструкції типу «квадрокоптер» є оптимальним рішенням.

За проблеми часу виконання роботи та поточної проблеми з електроенергією, робота обрахована на примітивних завданнях, які не є досить значним показником повноцінної масштабності системи, але доводить реалії того, що система працює.

Перевагами технології FHSS, яка використовується для радіозв’язку з дроном є те, що передача може вестися в досить широкому діапазоні частот. Інформація передається послідовно порціями на різних піддіапазонах. Якщо якийсь із частотних піддіапазонів буде заглушено, передача продовжуватиметься на решті. Алгоритми обробки дозволять виявити заглушені піддіапазони і виключити їх з закону зміни робочої частоти. Також слід відмітити, що для перехоплення інформації необхідно знати закон зміни робочої частоти.

До недоліків FHSS можна віднести низьку швидкість передачі даних, що зумовлюється затримкою при кожній зміні робочої частоти. Однак це вирішується шляхом застосування широкосмугових сигналів OFDM.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Jonathan Hiller and Hod Lipson. 2011. Automatic design and manufacture of soft robots. IEEE Transactions on Robotics 28, 2 (2011), 457–466.

2. Jordan B Pollack, Gregory S Hornby, Hod Lipson, and Pablo Funes. 2003. Computer creativity in the automatic design of robots. Leonardo 36, 2 (2003), 115–121.

3. Allan Zhao, Jie Xu. RoboGrammar: Graph Grammar for Terrain-Optimized Robot Design. [Електронний ресурс]. – Рижим доступу до ресурсу: https://cdfg.mit.edu/assets/files/robogrammar.pdf Дата доступу: 08.10.2022

4. Josh C Bongard. 2013. Evolutionary robotics. Commun. ACM 56, 8 (2013), 74–83.

5. ZM Bi and Wen-Jun Zhang. 2001. Concurrent optimal design of modular robotic configuration. Journal of Robotic systems 18, 2 (2001), 77–87.

6. Gil Lederman, Markus N Rabe, Edward A Lee, and Sanjit A Seshia. 2018. Learning heuristics for automated reasoning through deep reinforcement learning. arXiv preprint arXiv:1807.08058 (2018).

7. Gangyuan Jing, Tarik Tosun, Mark Yim, and Hadas Kress-Gazit. 2018. Accomplishing high-level tasks with modular robots. Autonomous Robots 42, 7 (2018), 1337–1354.

8. Karl Sims. Evolving Virtual Creatures [Електронний ресурс]. – Рижим доступу до ресурсу: https://www.karlsims.com/papers/siggraph94.pdf Дата доступу: 08.10.2022

9. Fritz R Stöckli and Kristina Shea. 2015. A simulation-driven graph grammar method for the automated synthesis of passive dynamic brachiating robots. In ASME 2015 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. American Society of Mechanical Engineers Digital Collection.

10. Gil Lederman, Markus N Rabe, Edward A Lee, and Sanjit A Seshia. 2018. Learning heuristics for automated reasoning through deep reinforcement learning. arXiv preprint arXiv:1807.08058 (2018).

11. Gabrielle Huston. Horizon Zero Dawn Is Real Now Thanks To Forest Ranger Druids [Електронний ресурс]. – Рижим доступу до ресурсу: https://www.thegamer.com/horizon-zero-dawn-real-life-forest-rehabilitation-ranger-druids / Дата доступу: 12.11.2022

12. B. Kaplan, İ. Kahraman, A. Görçin, H. A. Çırpan and A. R. Ekti, "Measurement based FHSS–type Drone Controller Detection at 2.4GHz: An STFT Approach," 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring), 2020, pp. 1-6, doi: 10.1109/VTC2020-Spring48590.2020.9129525.

13. S. Zoican, "Frequency hopping spread spectrum technique for wireless communication systems," 1998 IEEE 5th International Symposium on Spread Spectrum Techniques and Applications - Proceedings. Spread Technology to Africa (Cat. No.98TH8333), 1998, pp. 338-341 vol.1, doi: 10.1109/ISSSTA.1998.726253.

14. В.В. Котляров, О.Ю. Мирончук, О.О. Шпилька, «Математичний опис та формалізація типів спотворень у цифровому каналі зв’язку з OFDM-сигналами», Вісник НТУУ "КПІ". Серія Радіотехніка, Радіоапаратобудування, №66, с. 10-18, 2016. doi: 10.20535/RADAP.2016.66.10-18.

15. А. Ю. Мирончук, А. А. Шпилька, С. Я. Жук, “Метод двухэтапного совместного оценивания информационных символов и частотной характеристики канала в системах связи с OFDM, ” Известия вузов. Радиоэлектроника, vol. 63, no. 8, pp. 497–508, 2020, doi: https://doi.org/10.20535/S002134702008004X.

16. О. Ю. Мирончук, О. О. Шпилька, Д. Д. Струков, А. А. Петровський, і А. О. Герасименко, «ЗАСТОСУВАННЯ НЕЙРОННОЇ МЕРЕЖІ ДЛЯ ОЦІНЮВАННЯ ЧАСТОТНОЇ ХАРАКТЕРИСТИКИ БАГАТОПРОМЕНЕВОГО КАНАЛУ В СИСТЕМАХ ЗВ’ЯЗКУ З ТЕХНОЛОГІЄЮ OFDM», Вісник ВПІ, вип. 4, с. 99–104, Серп. 2021.

17. А. Ю. Мирончук, О. О. Шпилька, и С. Я. Жук, «Метод оценивания частотной характеристики канала а OFDM системах на основе фильтрации и экстраполяции пилот-сигналов,» Вісник НТУУ«КПІ». Серія Радіотехніка, Радіоапаратобудування, № 78, с. 36-42, 2019. https://doi.org/10.20535/RADAP.2019.78.36-42.

18. O. Myronchuk, O. Shpylka, S. Zhuk and Y. Myronchuk, "Algorithm Of Two-Stage Channel Frequency Response Estimation In OFDM Systems Based On Kalman Filter," 2021 IEEE 3rd Ukraine Conference on Electrical and Computer Engineering (UKRCON), 2021, pp. 241-246, doi: 10.1109/UKRCON53503.2021.9575234.

19. T. Jakubík and J. Jeníček, "Feasibility of OFDM and FHSS in a Single Home Automation and Security Network," 2020 43rd International Conference on Telecommunications and Signal Processing (TSP), 2020, pp. 19-22, doi: 10.1109/TSP49548.2020.9163493.

20. A. S. Bora, T. H. Singh and P. -T. Huang, "An All-Digital Wideband OFDM-based Frequency-hopping System using RF Sampling Data Converters," *2021 National Conference on Communications (NCC)*, 2021, pp. 1-5, doi: 10.1109/NCC52529.2021.9530022.

21. Струков Д.Д. Автоматизоване проектування легких літальних апаратів на основі графової граматики / Д.Д. Струков, О.Ю. Мирончук // Міжнародна науково-технічна конференція «Радіотехнічні проблеми, сигнали, апарати та системи»: матеріали конференції, 22 - 24 листопада 2022 р., м. Київ, Україна / КПІ ім. Ігоря Сікорського, РТФ. – Київ, 2022. – С. 100–102

ДОДАТОК А

Вміст файла правил конструкції формату DOT (.dot), який використовуеться при побудові дронів

```
digraph drone_graph {
     root[socket = 4];
     limb[socket = 2];
     limb_forward[socket = 2];
     limb_end[socket = 1];
     fixed_knee[socket = 4];
     fixed_knee_forward[socket = 4];
     motor_knee[socket = 4];
     motor_knee_forward[socket = 4];
     propeller[socket = 1];
     root->limb_forward[socket = 0];
     root->limb_forward[socket = 1];
     root->limb[socket = 2, socket_mirror = 3, mirror = 1];
     root->limb_end[socket = 0];
     root->limb_end[socket = 1];
     limb->fixed_knee[socket = 1];
     limb->motor_knee[socket = 1];
     limb->limb[socket = 1];
     limb->propeller[socket = 1];
     limb_forward->fixed_knee_forward[socket = 1];
     limb_forward->motor_knee_forward[socket = 1];
     limb_forward->limb_forward[socket = 1];
     limb_forward->propeller[socket = 1];
     fixed_knee->limb[socket = 1];
     fixed_knee->limb[socket = 2, socket_mirror = 3];
     fixed_knee->limb[socket = 3, socket_mirror = 2];
     fixed_knee->limb_end[socket = 1];
     fixed_knee->limb_end[socket = 2, socket_mirror = 3];
     fixed_knee->limb_end[socket = 3, socket_mirror = 2];
     motor_knee->limb[socket = 1];
     motor_knee->limb[socket = 2, socket_mirror = 3];
     motor_knee->limb[socket = 3, socket_mirror = 2];
     motor_knee->limb_end[socket = 1];
     motor_knee->limb_end[socket = 2, socket_mirror = 3];
     motor_knee->limb_end[socket = 3, socket_mirror = 2];
     fixed_knee_forward->limb_forward[socket = 1];
     fixed_knee_forward->limb[socket = 2, socket_mirror = 3, mirror = 1];
     fixed_knee_forward->propeller[socket = 1];
     fixed_knee_forward->propeller[socket = 2, socket_mirror = 3, mirror = 1];
     fixed_knee_forward->limb_end[socket = 1];
     fixed_knee_forward->limb_end[socket = 2, socket_mirror = 3, mirror = 1];
     motor_knee_forward->limb_forward[socket = 1];
     motor_knee_forward->limb[socket = 2, socket_mirror = 3, mirror = 1];
     motor_knee_forward->propeller[socket = 1];
     motor_knee_forward->propeller[socket = 2, socket_mirror = 3, mirror = 1];
     motor_knee_forward->limb_end[socket = 1];
     motor_knee_forward->limb_end[socket = 2, socket_mirror = 3, mirror = 1];
}
```

ДОДАТОК Б

Приклади створених коснтрукцій дронів, системою для завдання «паріння на місці»