Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____
(повна назва)

Кафедра_____Програмної інженерії_____
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти - другий (магістерський)

Дослідження методів динамічних ігор для розробки програмного забезпечення математичних моделей конфліктно керованих процесів
(тема)

Виконав: студент 2 курсу, групи ПЗСм-19-1___
_____Юрченко М.Р._____
(прізвище, ініціали)

спеціальності 121- Інженерія програмного забезпечення
(код і повна назва спеціальності)

Освітньо-професійної програми
(тип програми)

Програмне забезпечення систем
(повна назва освітньої програми)

Керівник _____проф. Власенко Л.А._____
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри, проф. _____ З.В.Дудар

2020 р.

Харківський національний університет радіоелектроніки

Факультет_____Комп'ютерних наук_____

Кафедра_____Програмної інженерії_____

Рівень вищої освіти_____другий (магістерський)_____

Напрям _____121 Інженерія програмного забезпечення_____

Освітньо-професійна програма ___Програмне забезпечення систем_____

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«1»_____9_____ 2020 р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові_____Юрченку Микиті Романовичу_____
(прізвище, ім’я, по батькові)

1. Тема роботи Дослідження методів динамічних ігор для розробки програмного забезпечення математичних моделей конфліктно керованих процесів

затверджена наказом по університету від „30” ___10_____ 2020 р. № 1490

2. Термін здачі студентом закінченої роботи „ 15” _____12____ 2020 р.

3. Вихідні дані до роботи Використовувати мову програмування Python, середовище розробки Atom

4. Зміст пояснювальної записки (перелік питань, що потрібно розробити) вступ, постановка задачі, методи диференціальних ігор, проектування програмного забезпечення, опис програмної реалізації, висновки, перелік посилань

(Зворотній бік бланку завдання)

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, плакатів та слайдів) UML діаграми системи, комп'ютерні ілюстрації (слайди), екранні форми

6. Консультанти розділів роботи (проекту)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Спецчастина	Власенко Л.А.		07.12.2020

7. Дата видачі завдання «_1_»__9___2020 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів дипломного проекту	Термін виконання етапів проекту	Позначка про виконання
1	Аналіз предметної галузі	15.09.2020	Виконано
2	Аналіз поставленої задачі	01.10.2020	Виконано
3	Кодування програми	15.10.2020	Виконано
4	Тестування і налагодження програми	01.11.2020	Виконано
5	Підготовка пояснювальної записки	15.11.2020	Виконано
6	Підготовка презентації та доповіді	01.12.2020	Виконано
7	Нормоконтроль, рецензування	07.12.2020	Виконано
8	Попередній захист	09.12.2020	Виконано
9	Занесення диплома в електронний архів	10.12.2020	Виконано
10	Допуск до захисту у зав. кафедри	11.12.2020	Виконано

Студент _____
(підпис)

Керівник роботи _____ __проф. Власенко Л.А.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка до атестаційної роботи магістра: 56 с., 21 рис., 17 джерел.

ДИФЕРЕНЦІЙНІ ІГРИ, КОНФЛІКТНО-КЕРОВАНІ ПРОЦЕСИ, PYTHON, МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ, АПРОКСИМАЦІЯ ДАНИХ, ДИФЕРЕНЦІЙНІ РІВНЯННЯ, ГРА ПЕРЕСЛІДУВАННЯ, MAVLINK, SITL.

Об’єктом дослідження є застосування методів динамічних ігор для розробки програмного забезпечення математичних моделей конфліктно керованих процесів.

Метою роботи є рішення задачі переслідування втікача.

Метод розробки базується на мові програмування Python.

В результаті роботи здійснено постановку задачі, огляд літературних джерел, організація теоретичного апарату, проектування, реалізація та випробування програмного засобу моделювання та керування.

DIFFERENTIAL GAMES, CONFLICT-CONTROLLED SYSTEMS, PYTHON, MATHEMATICAL MODELLING, DATA APPROXIMATION, DIFFERENTIAL EQUATIONS, PURSUIT-EVASION GAME, MAVLINK, SITL.

Application of dynamic differential games methods for developing mathematical modelling software for conflict-controlled processes is researched.

The purpose of the work is to solve evader-pursuit task.

Development method is based on the Python programming language.

As a result, the domain in question is researched, a task to be developed is set, scientific sources are reviewed, theoretical apparatus is set up, UAV control and modelling software design, implementation and testing is done.

ЗМІСТ

СПИСОК УМОВНИХ СКОРОЧЕНЬ

БПЛА – Безпілотний літальний апарат

APM – Automated planning missions

EEPROM – Electrically Erasable Programmable Read-Only Memory

GCS – Ground Control Station

LIP – Lithium Polymer

MAVProxy – Micro Air Vehicle Proxy

MAVLINK – Micro Air Vehicle Link

NURBS – Non-uniform rational B-spline

SITL – Software in the Loop

UAV – Unmanned Aerial Vehicle

ВСТУП

Від часів описаної у Аристотеля [1] апорії Зенона про черепаху і Ахіллеса, людство переймається задачею переслідування.

Сьогодні стрімко зростає кількість систем з автоматизованим керуванням. Постійно зростає кількість задач, що виконуються з використанням безпілотних літальних апаратів. Виникає конфліктна ситуація, з якої випливає нагальна потреба захисту територій, комплексів, установ, та інших об’єктів від небажаного вторгнення з використанням автоматично керованих апаратів.

Поставлено задачу дослідження методів диференціальних ігор та екстраполяції траєкторії рухомих об’єктів, для побудування математичних моделей оптимального керування конфліктно керованого процесу взаємодії БПЛА. В роботі розглядається метод захисту від літальних апаратів за допомогою захисного комплексу зі станції контролю, сенсорів відстеження, та певної кількості однорідних дронів-винищувачів, розподілених на захищеній території. В основу методу покладено класичні моделі диференціальних динамічних ігор переслідування декількома переслідувачами. Одним з позитивних отриманих результатів є вдосконалення швидкості синтезу оптимальної стратегії втікача та переслідувачів за рахунок використання проміжних матриць меншої розмірності, ніж в оригінальному описаному методі. Також підібрано апроксимаційну поліноміальну модель для прогнозування маршруту БПЛА за опорними точками.

Отриманий засіб моделювання можна використовувати як для загального моделювання з додаванням шумів, так і безпосередньо для оптимального керування літальними, пересувними, водними, або підводними безпілотними апаратами. Його можна використовувати як для керування безпілотником з наземної станції контролю або з бортового комп’ютера, так й для глобального планування маршруту наближення.

1 ОГЛЯД ЛІТЕРАТУРИ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Огляд літератури

Теорія нелінійної апроксимації розробляється в математичній статистиці від XIX століття. Основні результати та практичні застосування описані у [2]. Задача апроксимації полягає в тому, щоби знайти для певного набору складних даних відповідний більш простий набір, який за певних припущень можна використовувати замість першого. Задача інтерполяції полягає у винайденні проміжних значень величини, виходячи з відомих значень на певному відрізку.

Загальні базові поняття теорії ігор зформульовано в класичній роботі фон Неймана і Моргенштерна [3]. Вона містить, зокрема, визначення:

– гри з нульовою або ненульовою сумою;

– гри з повною або неповною інформацією.

Наведені модельні результати відносяться насамперед до дискретних ігор з можливістю практичної адаптації до економічних, а не військових застосувань. Однак, загальні інтуїції теорії ігор в подальшому були розвинені в теорії динамічних диференціальних ігор.

Теорія динамічних ігор була запропонована у класичній роботі Айзекса [4]. Зокрема, в ній вперше сформульовано класичну “задачу про шофера-вбивцю”, яку схематично зображено на рисунку 1.2. Дослідження Айзекса було засновано на методі динамічного програмування з використанням рівняння Айзекса-Белмана для пошуку значення гри.

Теорія диференціальних ігор Айзекса представляє собою синтез загального контексту методів теорії ігор у застосуванні до фізико-кінематичних моделей. Хоча цей метод підходить лише для вузького класу ігор переслідування-втікання, різноманітні модельні приклади дали поштовх для подальшого розвитку багатьох теорій, заснованих на базовому рівнянні Айзекса-Белмана.

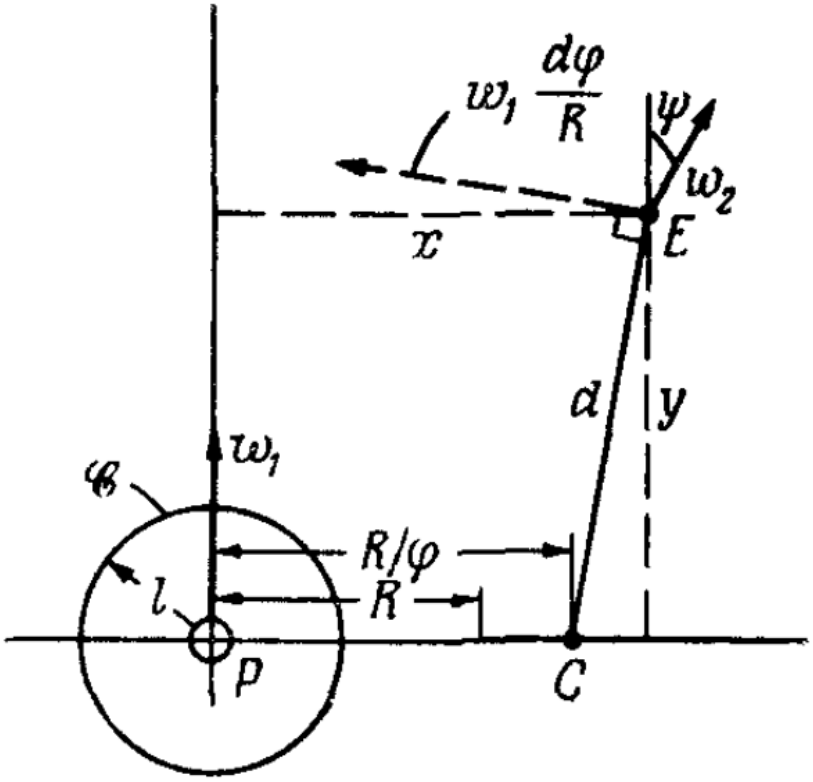

Рисунок 1.2 – Задача про шофера-вбивцю

На рисунку 1.2 мають місце наступні умовні позначення:

– вертикальну координату пішохода позначено як x;

– вертикальну координату пішохода позначено як y;

– радіус кола позначено як R;

– переслідувача позначено як P;

– втікача позначено як E;

– швидкість переслідувача позначено як ω_1;

– швидкість втікача позначено як ω_2;

– відстань між C та E позначено як d;

– кут повертання втікача позначено як $\psi \in [-1,\ 1]$;

– кут повертання переслідувача позначено як φ;

– модуль швидкості вектору x позначено як $\omega_1 \frac{d\varphi}{R}$.

Тож, динаміка системи має вигляд наступної системи диференціальних рівнянь

$$x' = -\frac{\omega_1}{R} y\,\varphi + \omega_2 \sin\psi,$$

$$y' = \frac{\omega_1}{R} x\,\varphi - \omega_1 + \omega_2 \cos\psi.$$

Теорія позиційних диференціальних ігор, запропонована в фундаментальній праці М.М. Красовського [5], дістала подальшого розвитку в рамках науково-дослідних програмах відомої Уральської, або Єкатеринбурзької школи, до якої входили

– А.І. Субботін;

– Ю.С. Осіпов;

– А.Б. Куржанський;

– А.Г. Ченцов.

Ключовою для розв’язання різноманітних задач у багатьох галузях стала система понять, до якої входять класичні визначення:

– позиційної диференціальної гри;

– сідлової точки малої гри;

– правила екстремального прицілювання;

– екстремального зсуву;

– альтернатив;

– програмних ітерацій;

– керованого контролю.

Завдяки раціональній комбінації чистих, мішаних, та контр-стратегій, став можливим опис диференціальної гри переслідування-втікання в рамках цілісної схеми. Також, було доведено фундаментальні теореми про альтернативи, що для будь-якого початкового стану гру можливо успішно завершити хоча б для одного гравця. Спеціальні множини початкових позицій, підібрані відносно термінальної множини, дістали назви стабільних мостів. Оптимальне керування тоді розкладається в послідовність екстремальних керувань, що дозволяють утримати

траєкторію конфліктного керованого процесу на стабільному мості. Нарешті, за наявності сідлової точки в т.з. малій грі, отримана оптимальна поведінка гравців не залежить від того, чи були для її будування використані чисті, мішані, або контр-стратегії.

Фундаментальний внесок в теорію оптимального керування з використанням моделей диференціальних динамічних ігор зробила також школа Л.С. Понтрягіна [6], до якої входили Р.В. Гамкрелідзе, В.Г. Болтянський та Є.Ф.Міщенко.

Визначним результатом цієї дослідницької програми стало формулювання Понтрягіним принципу максимуму. Гамкрелідзе знайшов доведення цього принципу в окремому випадку, а Болтянський – в загальному. Цей принцип залишається одним з фундаментальних засобів сучасної теорії оптимального керування [7].

Перший прямий метод Понтрягіна є одним з найпростіших та найуніверсальніших методів розв’язання широкого класу конкретних проблем переслідування. Також, Понтрягіним та Міщенко було отримано фундаментальний результат для вирішення глобальної проблеми ухилення від зіткнення в лінійному контексті, що дістав назву т.з. маневру ухилу. Відомий також результат розширення цього методу для нелінійних систем.

Подібний до прямого методу Понтрягіна метод опорних функціоналів описано у [8]. Цей метод за певних припущень відносно параметрів конфліктного процесу дає результати аналогічні тим, що описані у Понтрягіна.

Питання про необхідні та достатні умови завершення гри, а також метод пошуку часу завершення гри в лінійному випадку описано в [9]. Для процесів, що містять неявні диференціальні рівняння, диференційна гра описана у [10].

Актуальність використання диференціальних ігор для керування літальними апаратами продемонстровано у роботі [11]. В ній автори використовують підхід Айзекса в досить примітивній диференціальній грі в двомірному просторі, яку в

подальшому земульовано в двох фізичних симуляціях. Перша симуляція має високу подібність до реальних умов, а друга має ідеалізовану кінематику.

Для синтезу оптимальної стратегії переслідування доцільно використовувати класичні результати оптимізаційного підходу Понтрягіна, які викладено в досить лаконічному вигляді зокрема у [12].

Для практичних застосувань викладеної теорії використовуються чисельні методи, які детально описані в роботі [13]. Зокрема, йдеться про використання надзвичайно оптимізованих для якнайшвидшого виконання структур масивів та операцій лінійної алгебри над ними, а також засобів чисельного розв'язання систем звичайних диференціальних рівнянь.

В роботі [14] описані загальні підходи для використання мови програмування Python та бібліотеки Scikit в наукових обчисленнях та для безпосередньої реалізації та демонстрації роботи програмного забезпечення математичних моделей.

В роботі [15] лаконічно та оглядово викладено загальні принципи та архітектуру протоколу MAVLINK, що в останнє десятиліття став досить розповсюдженим, завдяки своїй універсальності, мінімалістичності, ефективності, швидкості та безпечності в якості формату обміну повідомленнями в середовищах автоматизованого керування:

– безпілотними літальними апаратами;

– безпілотними наземними роверами;

– безпілотними підводними човнами;

– безпілотними наводними човнами;

– станціями керування.

Протокол MAVLink має декілька сотень різних повідомлень. Деякі з них є універсальними та призначені для будь-яких безпілотних керованих апаратів, а деякі є специфічними для того чи іншого виду.

Кожне MAVLink-повідомлення має довжину від 11 до 279 байтів, і складається з наступних частин:

– символ початку пакету (повідомлення) довжиною 1 байт та зі значенням 0xFD;

– розмір корисного завантаження довжиною 1 байт;

– флаг несумісності довжиною 1 байт;

– флаг сумісності довжиною 1 байт;

– номер пакету в послідовності;

– ідентифікатор відправляючої сторони довжиною 1 байт;

– ідентифікатор компоненту довжиною 1 байт;

– ідентифікатор типу повідомлення у вигляді 3 байтів розташованих за принципом little endian;

– корисне навантаження пакету довжиною від 0 до 255 байт;

– CRC сума для перевірки цілісності пакету у вигляді 2 байтів розташованих за принципом little endian;

– сигнатура автентифікації довжиною від 0 до 13 байт.

В книзі[16] більш детально описані підходи для використання мови програмування Python та бібліотеки Scikit для обчислення різноманітних математичних моделей. Також написано, як використовувати бібліотеку Matplotlib для побудування та композиції:

– двовимірних графіків;

– тривимірних графіків;

– графіків в полярних координатах;

– діаграм;

– гістограм;

– статичних зображень.

Також в сьомому розділі описано побудування апроксимаційних поліномів для вирішення задачі екстраполяції даних.

1.2 Постановка задачі

Аналіз поставленої задачі сформульовано таким чином. Для дослідження застосування методів диференціальних ігор було обрано наступну загальну задачу. Стоїть завдання захистити певну територію від діяльності ворожих безпілотних літальних апаратів. Комплекс захисту складається з комплексу станцій стеження, що дозволяють відстежувати положення літальних апаратів, а також групи керованих перехопників-переслідувачів, що при наближенні до літального апарату можуть його знешкодити. Наочно цю задачу проілюстровано на рис. 1.1.

Основна задача побудування оптимального керування для перехоплювача буде складатися з наступних завдань:

– на основі даних зі станцій відстеження засобами теорії апроксимації необхідно передбачити траєкторію втікача;

– засобами теорії динамічних ігор знайти оптимальний шлях перехоплення, якщо такий він може існувати.

Для вирішення першої задачі обрано модель апроксимаційних поліномів, які найбільше підходять для екстраполяції відносно невеликих відрізків лінійних або викривлених ділянок шляху БПЛА, з огляду на отримане значення мінімальної помилки апроксимації за методом найменших квадратів.

Для вирішення другої задачі було обрано модель диференціальної динамічної гри переслідування одного втікача декількома однорідними переслідувачами з повною інформацією.

Алгоритм вирішення задачі пошуку оптимального керування зазначеним конфліктним процесом полягає в послідовному застосуванні апроксимаційного поліному для екстраполяції ділянки шляху БПЛА та підстановці його в модель диференціальної динамічної гри на місце втікача, і пошуку на підставі цих даних оптимального керування переслідування.

Для розробки програмної моделі необхідно перевести зазначені моделі у форму послідовності інструкцій та виразів прикладної мови програмування та зв’язати отриману математичну модель з безпосереднім керуванням тягою БПЛА або попереднім плануванням його маршруту.

Для тестування необхідно підставити в отриманий програмний засіб декілька нетривіальних наборів значень параметрів, а також спробувати під’єднати в якості керованого БПЛА достатньо розвинений емулятор.

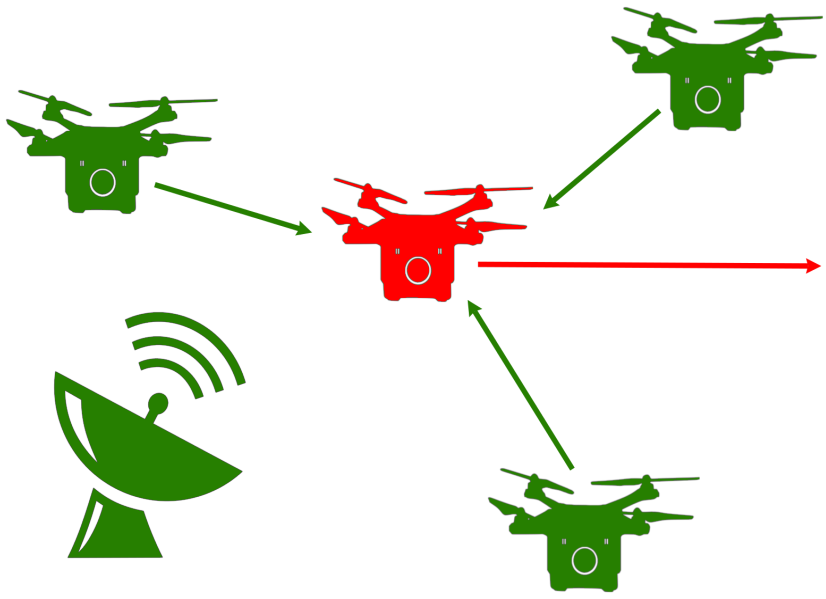

Рисунок 1.1 – Наочне зображення поставленої задачі

Таким чином, в роботі *поставлено задачу* дослідження, синтезу, та застосування методів динамічних диференційних ігор, зокрема, гри переслідування одного втікача декількома однорідними переслідувачами, а також методів екстраполяції траєкторії рухомих об’єктів, для побудування програмного забезпечення математичних моделей оптимального керування переслідувачів в контексті конфліктно керованого процесу взаємодії набору БПЛА-захисників та БПЛА-втікача у повітряному просторі.

2 ТЕОРІЯ АПРОКСИМАЦІЇ

2.1 Апроксимаційні поліноми

Теорія нелінійної апроксимації розробляється в математичній статистиці від XIX століття. Для апроксимації експериментальних даних використовуються сплайни – кусково задані функції, відрізки якої задані прямими та поліномами. Зокрема, для планування маршрутів в багатьох системах використовуються NURBS сплайни, або узагальнені сплайни Безьє. Втім, конформація кожного окремого відрізку такого сплайну послідовно залежить від параметрів попереднього. Таким чином, можливо апроксимувати лише цілий маршрут, а для цього на практиці скоріше за все не вистачить експериментальних даних.

В подальшому будемо брати до розгляду окрему ділянку функції апроксимації, тобто пряму або поліноміальну криву. Для апроксимації сигналів традиційно використовуються наступні ортогональні поліноми:

– Якобі;

– Гегенбауера;

– Лежандра;

– Чебишева першого роду;

– Чебишева другого роду;

– генералізовані Лагера;

– звичайні Лагера;

– Ерміта.

Щоб обрати, які поліноми найкраще підходять для поставленої задачі апроксимації траєкторії літального апарату, було проведено наступне попереднє дослідження. Було вилучено файл логування реального польоту квадрокоптеру, що відбувся 20.11.2018 о 12:59. Повний маршрут слідування наведено на рис. 2.1.

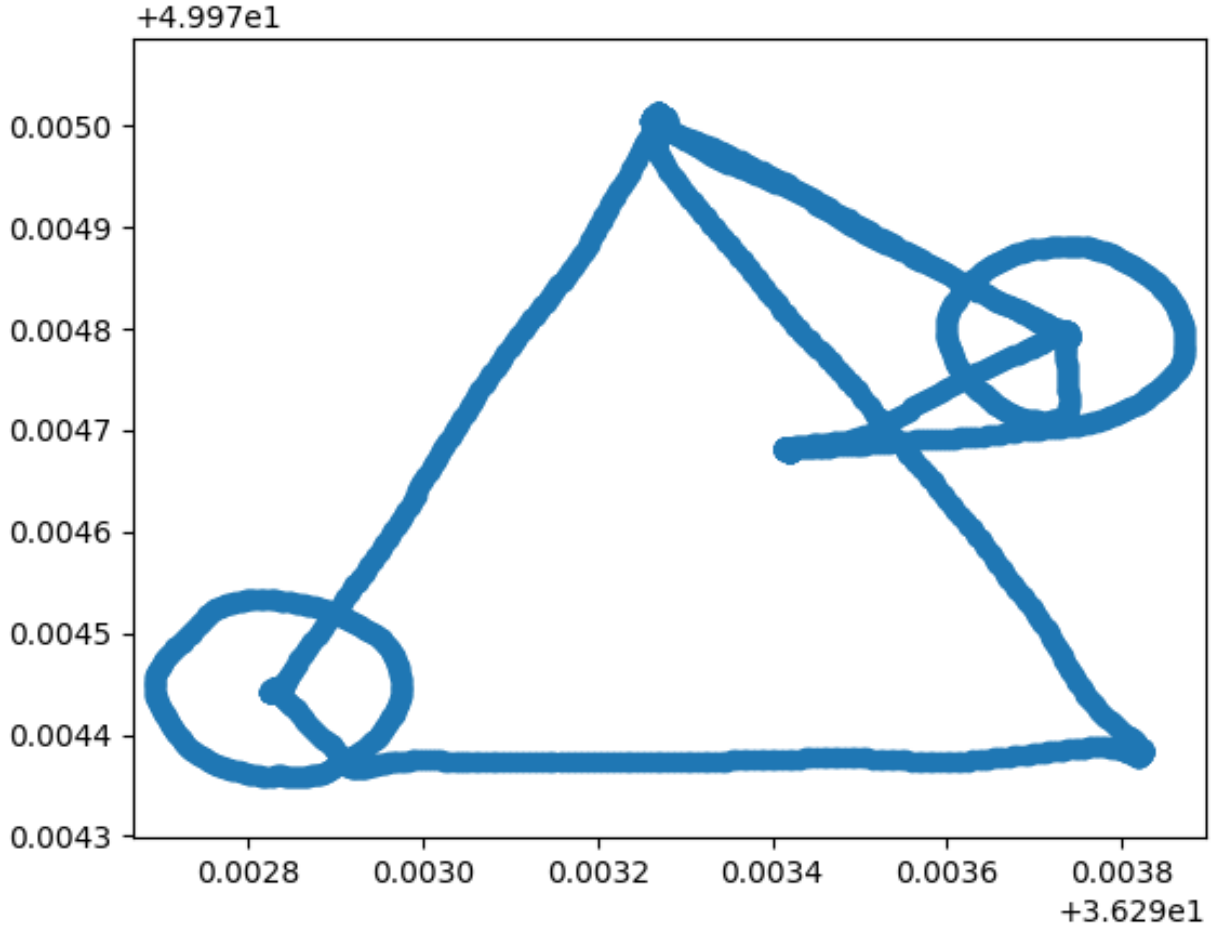

Рисунок 2.1 – Траєкторія польоту дрона

З файлу логування, що містився на картці пам’яті дрону за допомогою Unix засобів було вилучено Mavlink-повідомлення, що безпосередньо стосуються канонічної, тобто скоригованої вбудованими засобами контролеру ArduPilot, позиції. Ці повідомлення починаються з префіксу POS, тож команда має наступний вигляд:

```
$ cat flight.log | grep "POS, " > pos.txt
```

Далі, за допомогою програми на мові програмування Python було складено невелику программу для парсингу отриманого логу. З усього шляху було обрано відрізок в 50 точок, який зображено на рис. 2.2.

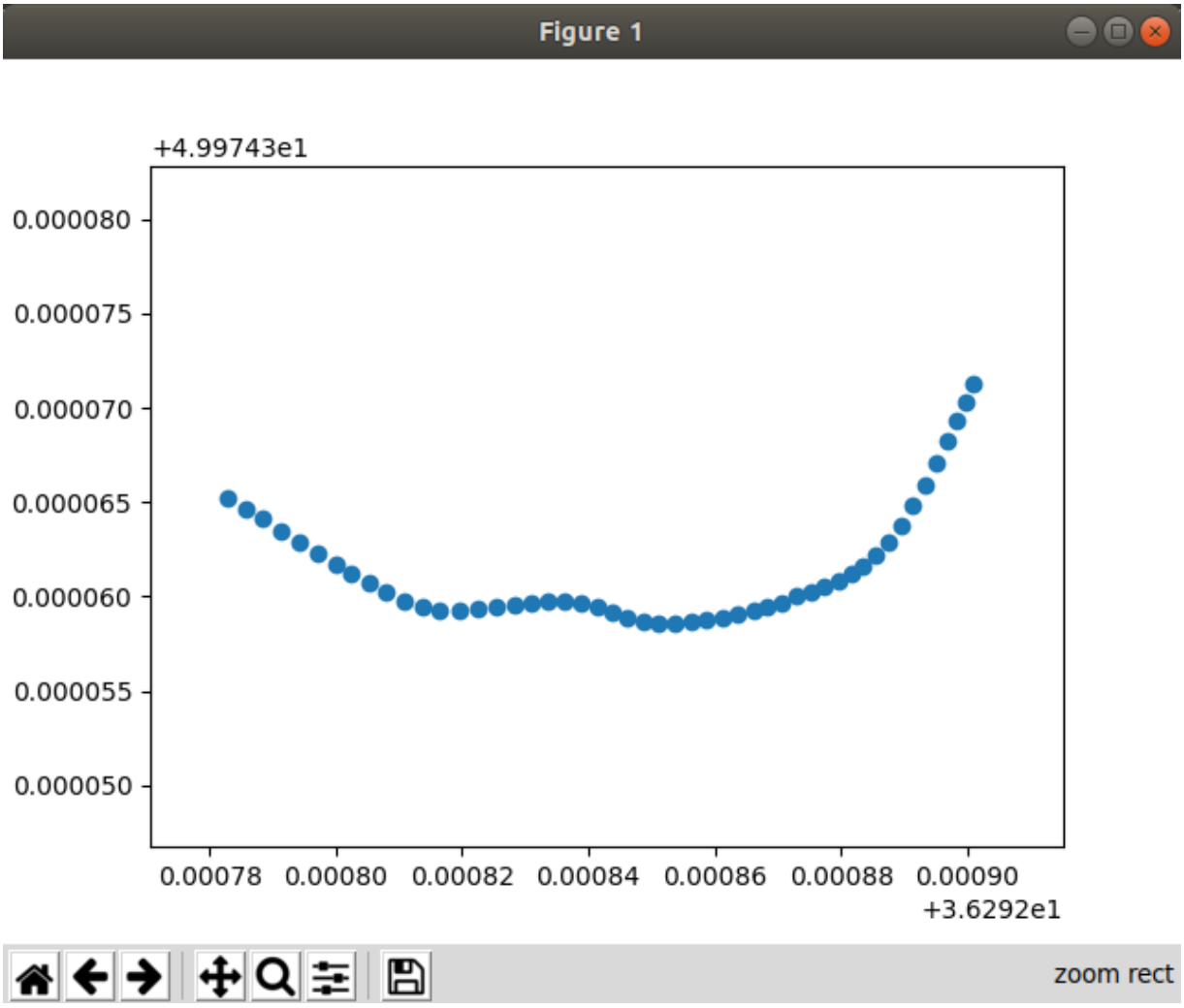

Рисунок 2.2 – Розглянутий відрізок

2.2 Екстраполяція за методом найменших квадратів

Масив точок, отриманий апроксимовано за методом найменших квадратів з використанням наступних функцій:

– звичайних степеневих поліномів вигляду $P_n = a_0 + a_1 x_1 + ... + a_n x_n$;

– поліномів Чебишева;

– звичайних поліномів Ерміта;

– генералізованих поліномів Ерміта;

– поліномів Лагерра;

– поліномів Лежандра.

Для лінійного випадку ступеню 1 отримані такі значення сумарної помилки (e):

– для степеневих поліномів e=8.218002189456714e-06;

– поліномів Чебишева e=8.218002189456714e-06;

– звичайних поліномів Ерміта e=8.218002189456714e-06;

– генералізованих поліномів Ерміта e=8.218002189456714e-06;

– поліномів Лагерра e=8.218002189475104e-06;

– поліномів Лежандра e=8.218002189456714e-06.

Результат апроксимації наведено на рис. 2.3.

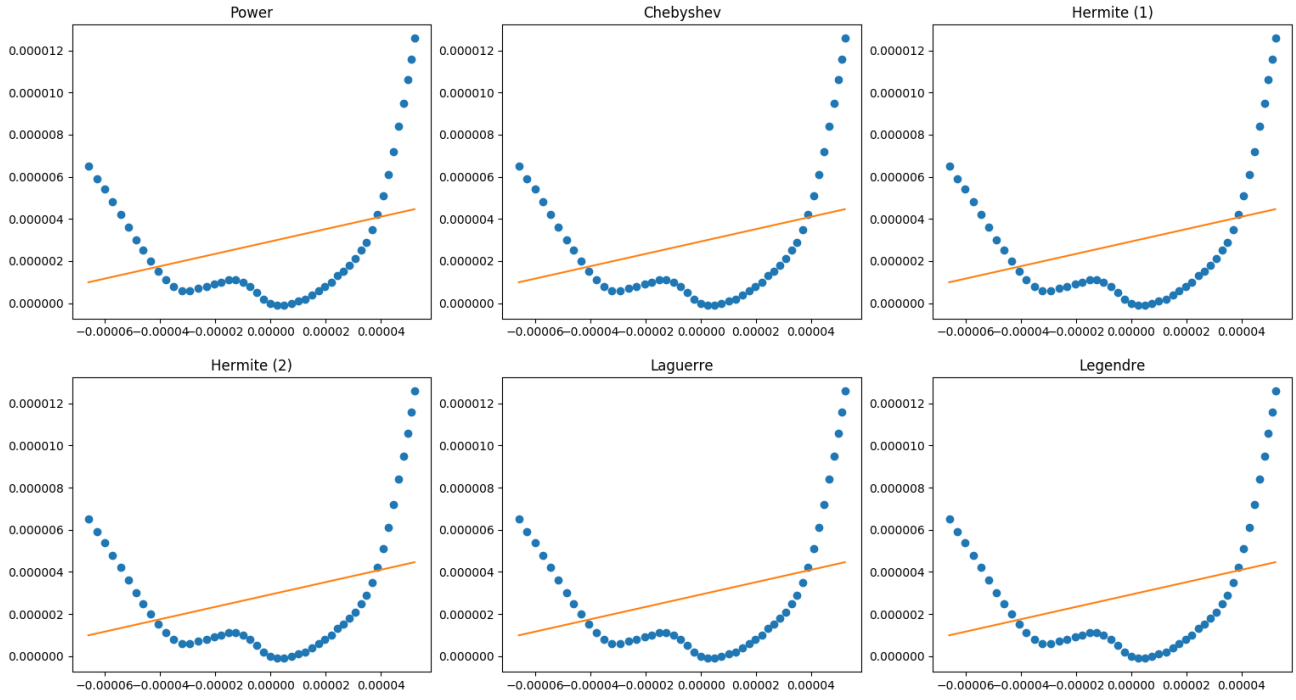

Рисунок 2.3 – Лінійна апроксимація

Таким чином, найнижчим є значення помилки при використанні поліномів ступеня, Чебишева, Ерміта, Лежандра. Найбільше значення помилки отримано

при апроксимації поліномом Лагерра. Для випадку ступеню 2 отримані такі значення сумарної помилки (e):

– для степеневих поліномів e=1.6702662750633454e-05;

– поліномів Чебишева e=1.670266074777227e-05;

– звичайних поліномів Ерміта e=1.670266074777227e-05;

– генералізованих поліномів Ерміта e=1.6702664762137437e-05;

– поліномів Лагерра e=1.670265874054166 5e-05;

– поліномів Лежандра e=1.6702660722174425e-05.

Таким чином, найнижчим є значення помилки при використанні поліномів Лагерра. Найбільше значення помилки отримано при апроксимації генералізованим поліномом Ерміта. Отримані результати наочно зображені на рис. 2.4.

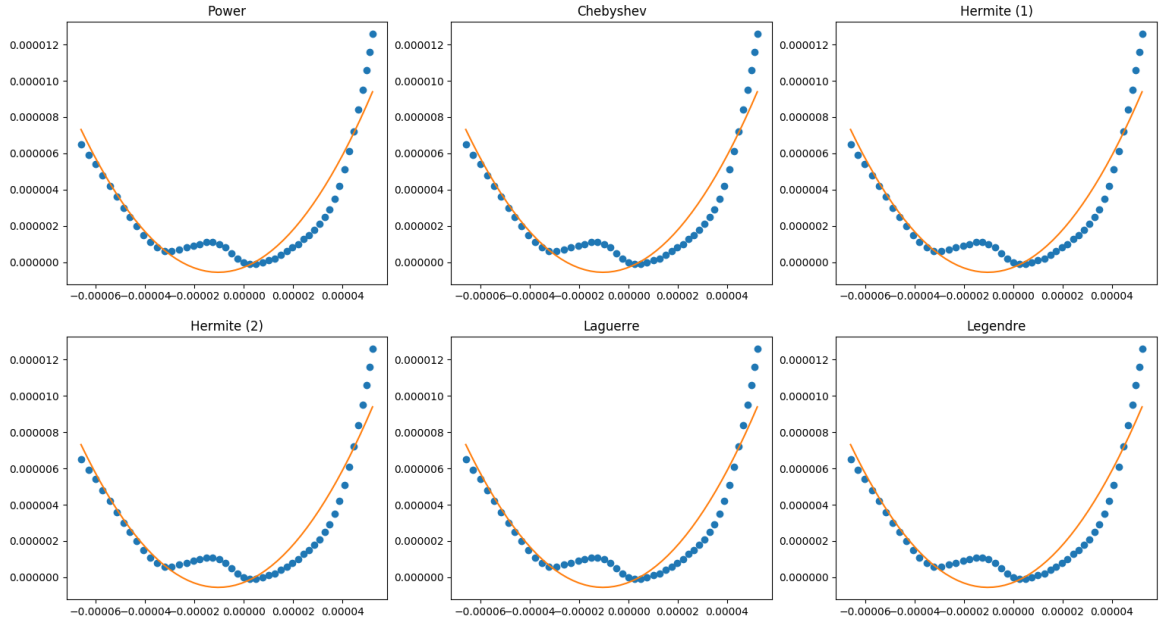

Рисунок 2.4 – Нелінійна апроксимація

З даного експерименту видно, що в подальшому використання поліномів Лагерра є оптимальним. Повний код обробки та апроксимації експериментальних даних наведено в додатку В.

3 ТЕОРІЯ ДИНАМІЧНИХ ІГОР

3.1 Диференціальні ігри

Диференційною грою називається така гра, що її результат визначається поведінкою певної керованої динамічної системи. Традиційним прикладом такої гри є *гра переслідування*, в якій беруть участь (в найпростішому випадку) один переслідувач та один переслідуваний об'єкт пов'язані в *динамічну систему*, що описується рівнянням

$$s'(t) = F(t,\ s(t),\ x(t),\ y(t)),\ s(t) \in E_n\,,$$

притому *початкова умова*

$$s_0 = s(t_0)$$

в момент початку гри $t_0 \in R$ вважається заданою. Переслідувач має обрати вимірюване за Лебегом керування $x(t) \in X$ (майже всюди), де X – замкнена обмежена множина в евклідовому просторі E_m. Керування, що додатково задовольняє цим двом умовам, називається припустимим. Аналогічно для множини Y вводиться припустиме керування втікача $y(t) \in Y$.

Для простоти розглянемо її як гру з повною інформацією, тобто таку, що всі учасники точно знають всю інформацію про інших участників.

Наприклад у випадку, коли права частина розглянутої динамічної системи є лінійною , диференційна гра також називається лінійною. Рівняння має вигляд

$$s`(t) = As(t) + Bx(t) + Cy(t),\ t \geq t_0,$$

де $s(t_0) = s_0$ – визначено, A, B, C – постійні матриці параметрів системи.

Обмеження на функції керування переслідувача та втікача можна визначити як

$$|u_e(t)| \leq \gamma \ \text{ та } \ \left|u_p(t)\right| \leq \sigma\,,$$

де $\gamma > 0,\ \sigma > 0$ – позитивні скаляри.

3.2 Необхідні та достатні умови завершення гри за заданий час

Будемо вважати, що гру можна завершити в момент T за час $T–t_0$, якщо переслідувач для будь–якого довільного керування втікача буде мати власне керування таке, що

$$||\pi s(T)|| \le d,$$

де $d \in R,\ d \ge 0$,

π – оператор проектування з евклідового простору E_m в E_k, $k \le m$.

Зрозуміло, що момент T буде залежати від початкового стану z_0. З того, що переслідувач та втікач є антагоністами можна ввести наступну нерівність

$$sup_{y(.)} inf_{x(.)} ||\pi s(T)|| \le d,$$

яку можна обгрунтувати наступним чином. Зафіксуємо керування втікача $y(t_0, T)$. Тоді можна знайти таке керування переслідувача $x(t_0,\ T) \in X$, що

$$||\pi s_x(T)|| = inf_{x \in X} ||\pi s(T)||.$$

Інакше кажучи, оптимальною стратегією є сідлова точка.

3.3 Диференціальна гра декількох переслідувачів

В більш узагальненому вигляді припустимо наявність $n+1$ гравців: одного переслідуваного та n однакових переслідувачів. Нехай ця диференційна гра переслідувача та втікача відбувається в евклідовому просторі E_m. Положення переслідувачів в момент t задаються як вектор векторів

$$x_i(t) = [x_{i_1}, x_{i_2},\ ...,\ x_{i_m}]^T,\ x = [x_1^T,\ x_2^T,\ ...,\ x_n^T],$$

а положення втікача в момент t задається як

$$y = [y_1, y_2, \ ..., \ y_m]^T \ .$$

Тоді має місце

$$u_{p_i}(t) = x_i^{'}, \ i = 1..n \ ,$$

$$u_p(t) = x' \ ; \ u_e(t) = y' \ .$$

Далі введемо вектор різниці між положеннями втікача та переслідувачів

$$z_i = [z_{i_1}^T, z_{i_2}^T, ..., \ z_{i_m}^T]^T = [(x_{i_1} - y_1)^T, (x_{i_2} - y_2)^T, \ ..., \ (x_{i_m} - y_m)^T]^T \ .$$

У компактному вигляді

$$z(t) = x(t) - 1_n \otimes y(t) \ ,$$

де 1_n – одиничний вектор розміру n ,

$\otimes$ – множення за Кронекером.

Відповідно має місце наступне співвідношення, що виражає динаміку системи

$$z'(t) = x'(t) - 1_n \otimes y'(t) \ .$$

Або підставивши $u_p(t) = x'$ та $u_e(t) = y'$ маємо

$$z'(t) = u_p(t) - 1_n \otimes u_e(t) \ .$$

При розгляданні гри з нульовою сумою, буде мати місце єдиний спільний функціонал якості. Задачею переслідувача буде його максимізувати, або мінімізувати. Якщо переслідувач буде намагатися максимізувати цей функціонал, то втікач буде навпаки намагатися його мінімізувати. Відповідно, якщо переслідувач буде намагатися мінімізувати цей функціонал, то втікач буде намагатися його максимізувати.

Введемо додатково обмеження на швидкості переслідувачів

$$|u_{p_{ij}}| \le \gamma_i, \ i = 1, \ 2, \ ..., \ m, \ j = 1, \ 2, \ ..., \ n \ ,$$

так що всі переслідувачі мають однакове обмеження. Також обмежимо швидкість втікача

$$|u_{e_i}| \le \sigma_i, i = 1, \ 2, \ ..., \ m \ .$$

3.4 Функціонали якості втікача та переслідувачів

При розгляданні гри з ненульовою сумою, буде мати місце два різних функціонала якості. Кожний гравець при цьому розглядається окремо.

Розглянемо спочатку функціонал якості для переслідувачів

$$J_p = \tfrac{1}{2} k_{pf} z^T(t_f) z(t_f) +$$

$$+ \tfrac{1}{2} \int_{t_0}^{t_f} [[z(t),\ u_p(t), 1_n \otimes u_e(t)]\ diag(q_p I,\ r_p I,\ 0)^T [z(t),\ u_p(t), 1_n \otimes u_e(t)]^T] dt\ ,$$

де k_{pf} – невід'ємний ваговий скаляр для переслідувача в момент закінчення,

q_p – невід'ємний ваговий скаляр для переслідувача в процесі керування,

r_p – додатній ваговий скаляр ресурсів переслідувача,

$diag(x_1,\ x_2,\ ...,\ x_k)$ – діагональна матриця розміру k, з елементами діагоналі $x_1,\ x_2,\ ...,\ x_k$,

I – одинична матриця,

t_f – момент закінчення гри,

t_0 – момент початку гри.

Функціонал втікача задамо в наступному вигляді

$$J_e = -\tfrac{1}{2} k_{ef} z^T(t_f) z(t_f) +$$

$$+ \tfrac{1}{2} \int_{t_0}^{t_f} [[z(t),\ u_p(t), 1_n \otimes u_e(t)]\ diag(-q_e I,\ 0,\ r_e I)^T [z(t),\ u_p(t), 1_n \otimes u_e(t)]^T] dt\ ,$$

де k_{ef} – невід'ємний ваговий скаляр для втікача в момент закінчення,

q_e – невід'ємний ваговий скаляр для втікача в процесі керування,

r_e – додатній ваговий скаляр ресурсів для переслідувача,

$diag(x_1,\ x_2,\ ...,\ x_k)$ – діагональна матриця розміру k, з елементами діагоналі $x_1,\ x_2,\ ...,\ x_k$,

I – одинична матриця,

t_f – момент закінчення гри,

t_0 – момент початку гри.

Вищезазначені функціонали можуть мати наступну інтерпретацію. Втікач намагається якомога збільшити зважені дистанції між ним та переслідувачами, при цьому витративши якомога менше свого ресурсу (пального, енергії, тощо). В свою чергу переслідувачі намагаються якомога зменшити зважені дистанції між ними та втікачем, при цьому витративши якомога менше свого ресурсу (пального, енергії, тощо).

3.5 Синтез керування

Для розглянутої гри з глобальною інформацією, класичний результат для поданої проблеми передбачає використання теорії оптимального керування. Зокрема, має місце наступна теорема.

Теорема 1. Якщо дана гра за участі n переслідувачів, динаміка якої задана рівнянням

$$z'(t) = u_p(t) - 1_n \otimes u_e(t),$$

функціонал якості переслідувачів задано як

$$J_p = \tfrac{1}{2} k_{pf} z^T(t_f) z(t_f) +$$

$$+ \tfrac{1}{2} \int_{t_0}^{t_f} [[z(t),\ u_p(t), 1_n \otimes u_e(t)]\ diag(q_p I,\ r_p I,\ 0)^T [z(t),\ u_p(t), 1_n \otimes u_e(t)]^T] dt,$$

функціонал якості втікача задано як

$$J_e = -\tfrac{1}{2} k_{ef} z^T(t_f) z(t_f) +$$

$$+ \tfrac{1}{2} \int_{t_0}^{t_f} [[z(t),\ u_p(t), 1_n \otimes u_e(t)]\ diag(-q_e I,\ 0,\ r_e I)^T [z(t),\ u_p(t), 1_n \otimes u_e(t)]^T] dt,$$

а також гра припускає рівновагу Неша зі зворотнім зв'язком, то оптимальною стратегією переслідувача є

$$u_p^*(t) = -\frac{1}{r_p} K_p z(t),$$

а оптимальною стратегією втікача є

$$u_e^*(t) = -\frac{1}{nr_e}(1_n^T \otimes I_{m\times m}) K_e z(t),$$

де

$$K_p' = -q_p I - \frac{1}{nr_e} K_e{}^T (1_n^T \otimes 1_n \otimes I_{m\times m}) K_p - \frac{1}{nr_e} K_p (1_n \otimes 1_n^T \otimes I_{m\times m}) K_e + \frac{1}{r_p} K_p K_p,$$

$$K_e' = -q_e I + \frac{1}{r_p} K_p^T K_e + \frac{1}{r_p} K_e^T K_p - \frac{1}{nr_e} K_e (1_n \otimes 1_n^T \otimes I_{m\times m}) K_e,$$

$$K_p(t_f) = k_{pf} I,$$

$$K_e(t_f) = k_{ef} I.$$

Доведення цієї теореми приведено в додатку до роботи [7]. Доказ засновано на підстановці означених функціоналів до функції Ляпунова, з подальшим інтегруванням і підстановкою в отриманий вираз оптимальних стратегій.

Для того, щоб ввести до розгляду також раніше зазначені обмеження на швидкість та ресурси, введемо співвідношення

$$r_{e_i} = \frac{1}{\sigma_i},\ i = 1..m,$$

відповідно до якого гравцю не вигідно збільшувати власну швидкість.

Наприклад, розглянемо наступну гру переслідування з одним переслідувачаем та одним втікачем, що їхнє початкове положення визначається як

$$x(t_0) = (0,\ 0),$$

$$y(t_0) = (1,\ 1),$$

а також мають місце наступні обмеження $r_e = 1;\ r_p = 2;\ q_e = 2;\ q_p = 2;\ k_{ef} = 1;$ $k_{pf} = 2$. Час початку гри $t_0 = 0$, час завершення $t_f = 4$. Відповідно до теореми 1 оптимальними керуваннями є:

$$u_{pi}^*(t) = -\frac{k_p}{r_p}(x(t) - y(t)),$$

$$u_e^*(t) = -\frac{k_e}{r_e}(x(t) - y(t)),$$

де K_p та K_e утворюють систему рівнянь Ріккаті:

$$k_p' = -q_p + \frac{1}{r_p}k_p^2 - \frac{2}{r_e}k_p k_e = -2 + \frac{1}{2}k_p^2 - 2k_p k_e ,$$

$$k_e' = -q_e + \frac{2}{r_p}k_p k_e - \frac{1}{r_e}k_e^2 = -2 + k_p k_e - k_e^2 ,$$

$$k_p(t_f) = k_{pf} = 2 ,$$

$$k_e(t_f) = k_{ef} = 1 ,$$

які можна вирішити за допомогою чисельних методів, наявних в пакеті Wolfram Mathematica, і отримати наступний графік функцій керування, який подано на рис. 3.1.

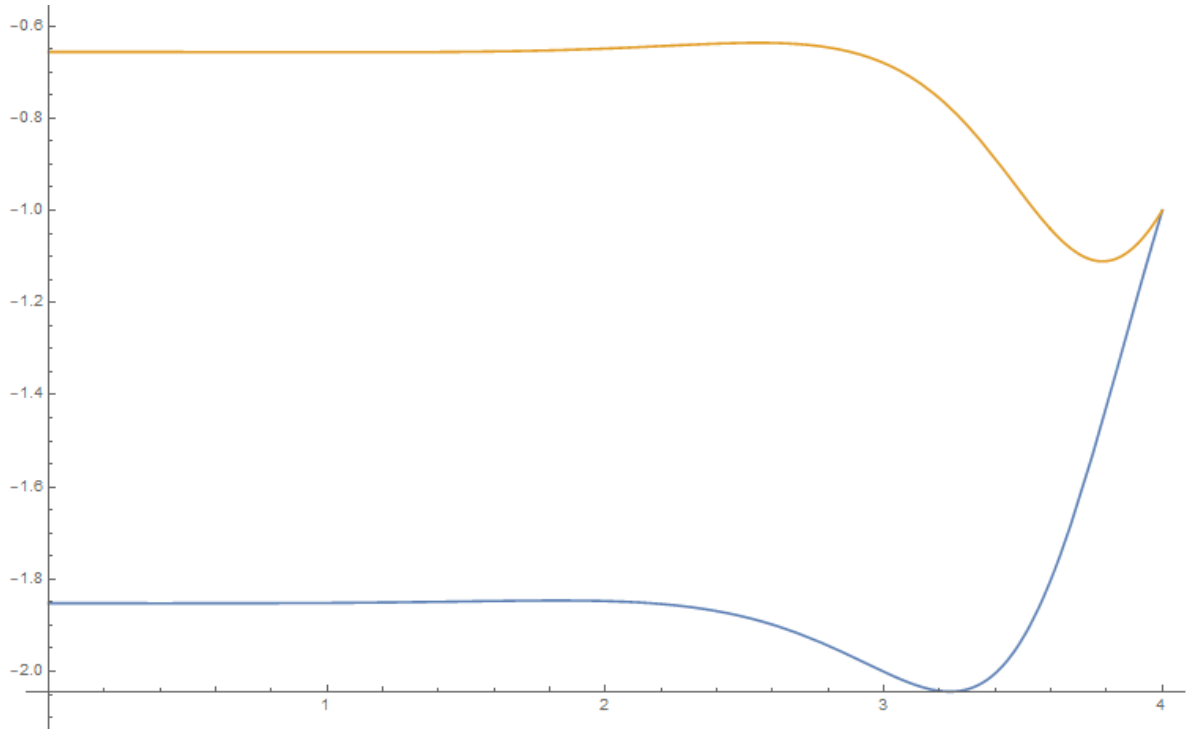

Рисунок 3.1 – Графік оптимальних керувань u_p^*, u_e^*

З графіку можна побачити, що синтезовані за зазначеним методом функції керування сходяться в одну точку саме в момент завершення гри. В подальшому, наклавши відповідні обмеження швидкості, їх можна використовувати безпосередньо в симуляції.

4 ПРОЕКТУВАННЯ ПРОГРАМНОЇ СИСТЕМИ

4.1 Аналіз вимог до програмної системи

Необхідно створити програмну систему для керування безпілотними літальними апаратами. Основні функціональні вимоги до цієї системи наступні:

– для побудови оптимальної програми керування мають використовуватися методи диференціальних ігор;

– можливість завдання початкових позицій та параметрів переслідувачів;

– можливість отримати результат у вигляді MAVLINK команд;

– можливість моделювання для двовимірної або тривимірної ситуації.

Також, до функціональних вимог належать:

– можливість під’єднати симулятор БПЛА;

– можливість за даними точками отримати апроксимоване керування віткача та за ним побудувати оптимальне керування переслідувача.

До основних нефункціональних вимог відносяться:

– можливість виконання на бортовому комп’ютері RaspberryPi;

– програма має містити набір прикладів власного використання.

Решту вимог буде подано у вигляді UML діаграм.

4.2 UML проектування програмної системи

Виходячи з природи поставленої задачі, найбільш доцільним для її вирішення буде використовувати об’єктно-орієнтований підхід. Діаграму класів наведено на рис. 4.1.

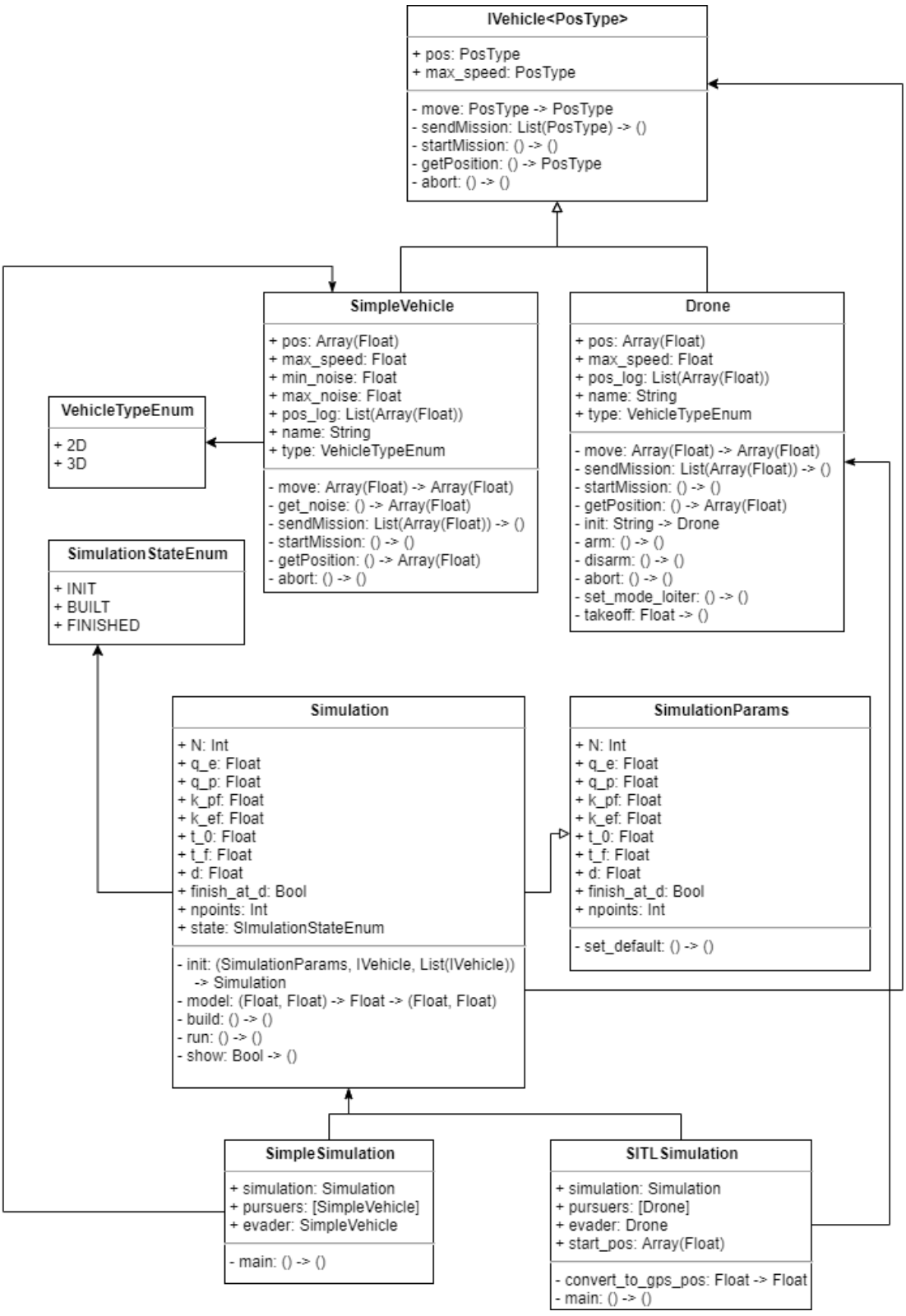

Рисунок 4.1 – Діаграма класів

Кількість наслідників інтерфейсу IVehicle можна й далі розширювати.

Для класу Simulation обов’язковим буде дотримання послідовності станів. Ці стани проілюстровано за допомогою діаграми станів на рис. 4.2.

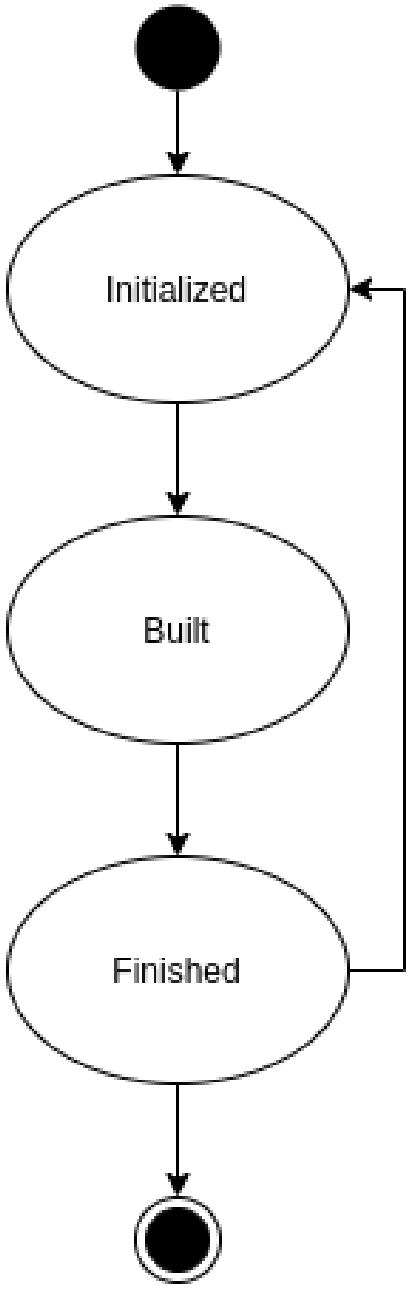

Рисунок 4.2 – Діаграма станів

При використанні в реальному переслідуванні, має місце активна взаємодія між GCS та бортовим контролером дрону. Ця взаємодія подана за допомогою діаграми взаємодії на рис. 4.3.

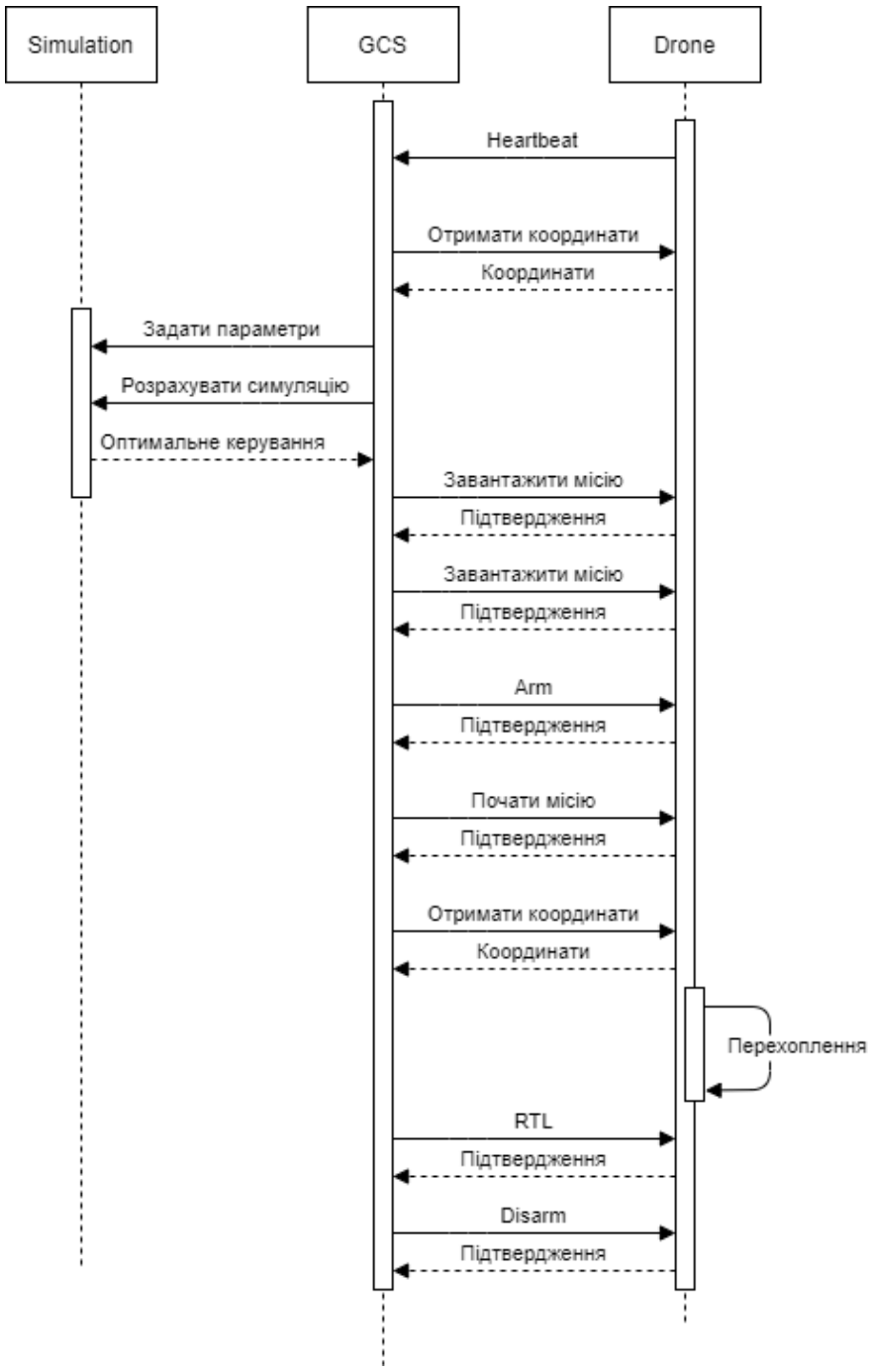

Рисунок 4.3 – Діаграма взаємодії

З поданої діаграми видно, що вся взаємодія станції керування з дроном має зворотній зв’язок у вигляді відповідного підтвердження. Однак, в цій схемі бракує зворотного зв’язку з симуляцією. Тому більш досконалою буде схема взаємодії, в якій симуляція коригується в залежності від реального, а не лише передбачуваного положення гравців. Таку взаємодію подано на рис. 4.4.

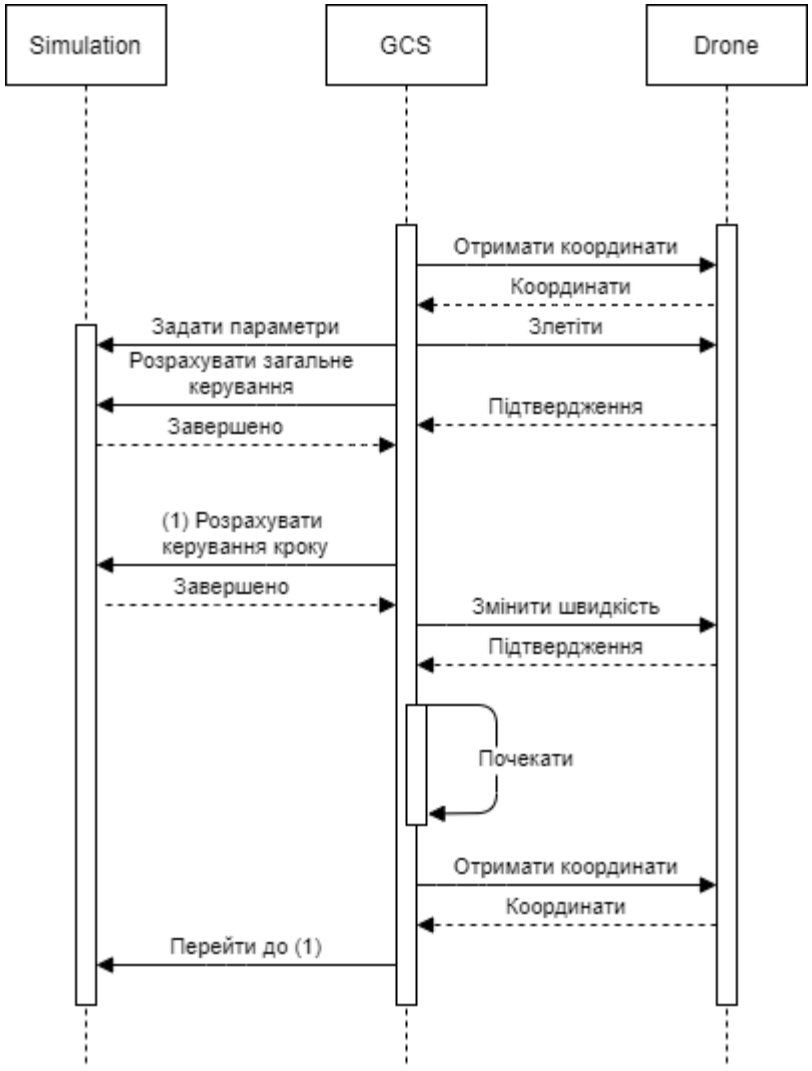

Рисунок 4.4 – Діаграма взаємодії зі зворотнім зв’язком

Перша схема взаємодії передбачає обрахунок цілого оптимального маршруту, то друга схема безпосередньо використовує оптимальне керування. Перша схема значно менше навантажує канал зв’язку та не вимагає real-time обмежень, але їй може забракувати точности. Друга схема натомість вимагає великої кількості повідомлень, але дозволяє точно влучити в тікача.

Отже, доцільним буде використовувати, наприклад, першу схему для початкового наближення та глобального планування маршруту, а другу на етапі активного переслідування в кінці. Використання другої схеми безпосередньо на борту БПЛА дозволить вирішити проблему навантаження комунікацій.

5 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

5.1 Опис програмної системи

Для реалізації було обрано мову програмування Python, з огляду на наступні переваги:

– висока швидкість роботи;

– велика кількість швидких алгоритмів для наукових обчислень;

– можливість запуску на бортовому комп’ютері RaspberryPi;

– наявність готових бібліотек для роботи з протоколом MAVLINK;

– наявність програмного симулятора БПЛА.

Було використано наступні бібліотеки:

– для рішення рівнянь Ріккаті використовується бібліотека наукових обчислень scipy;

– для матричних обчислень використовується бібліотека numpy;

– для графічного відображення використовується бібліотека matplotlib;

– для взаємодії з MAVLINK-керованим БПЛА (реальним, або симульованим) використовуєтсья бібліотека pymavlink;

– для симуляції БПЛА використовується бібліотека DroneKit-SITL.

Бібліотека наукових обчислень scipy має великий арсенал готових наукових обчислень. Крім того, вона є безпосередньо сумісною з бібліотекою швидких матричних обчислень numpy, швидкість якої досягається завдяки використанню FORTRAN складових.

Сьогодні існує декілька найбільш поширених протоколів керування БПЛА. Вони поділяються на пропрієтарні та відкриті. Пропрієтарні підходять лише для окремих моделей, або груп моделей контролерів БПЛА, такі як лінійки контролерів та дронів Naza. Притому лише для найдорожчих моделей цей протокол є доступними для зовнішніх розробників, а для більш дешевих моделей

можливо використання тільки спеціалізованих програм. Протокол автоматичного керування безпілотними апаратами MAVLINK, натомість, є повністю відкритим, доступним, простим, та досить широко розповсюдженим. Він може використовуватися для керування літальними, наземними, наводними, або підводними апаратами.

Використання MAVLINK-керованих контролерів порівняно з пропрієтарними дає набагато більше можливостей для гнучкого конфігурування комплектації дронів. Приклад власноруч зібраного автором MAVLINK-дрону наведено на рис. 5.1.

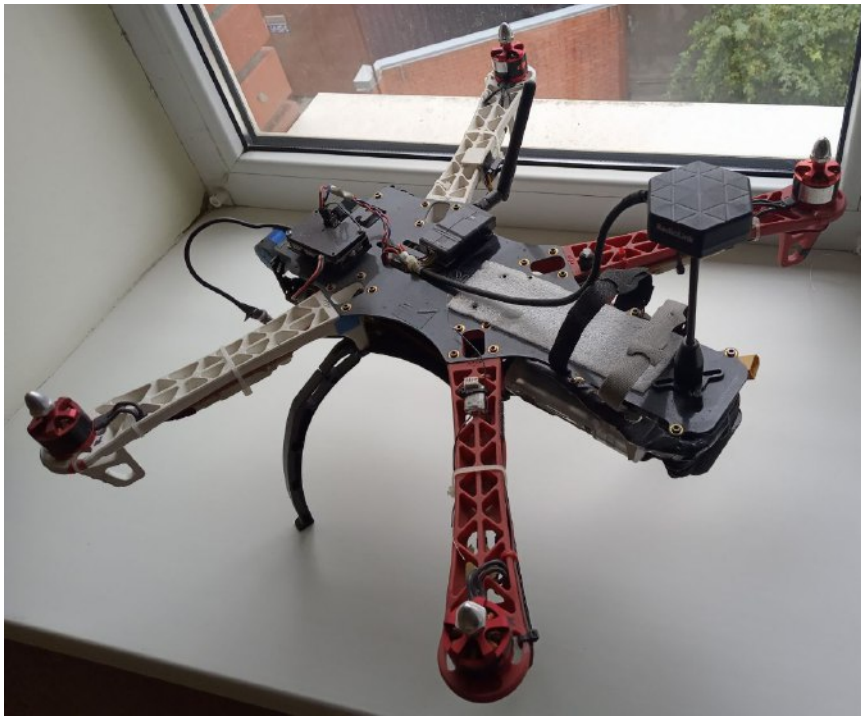

Рисунок 5.1 – Керований за протоколом MAVLINK дрон

Нарешті, MAVLINK має ту перевагу, що дозволяє гнучко налаштувати топологію мережі потоків керування та телеметрії. Зокрема, в цьому допомагає використання засобу MAVProxy.

5.2 Приклади використання та програмне моделювання

В якості найпростішого прикладу розглянемо переслідування в двовимірній площині. Кількість переслідувачів дорівнює 3. Перший переслідувач знаходиться на початку гри в позиції (-3, 0). Другий переслідувач знаходиться на початку гри в позиції (3, 0). Третій переслідувач знаходиться на початку гри в позиції (4, 1). Втікач знаходиться на початку гри в позиції (2, 2). Переслідувачі мають максимальне обмеження за швидкістю 1.0. Втікач має обмеження за швидкістю 0.8. Час початку гри дорівнює 0. Час завершення гри дорівнює 4. Кількість кроків дискретизації дорівнює 50. Коефіцієнт $k_{pf} = 10$. Коефіцієнт $k_{ef} = 5$. Коефіцієнт $q_e = 2$. Коефіцієнт $q_p = 1$. Результат симуляції подано на рис. 5.2. Переслідувачі та втікач не роблять максимальних можливих кроків, тому що згідно з запропонованою математичною моделлю їм обом невигідно збільшувати витрати ресурсів. Це також правильно з огляду на нелінійність наведеної на рисунку 5.3 вольт-ампер-годинної характеристики літієвих LIP батарей, що використовуються в реальних дронах, які можуть надати більшу швидкість саме на початку активного навантаження.

Розглянемо далі схожу диференційну гру з п’ятьмя переслідувачами. Перший переслідувач знаходиться на початку гри в позиції (20, 0). Другий переслідувач знаходиться на початку гри в позиції (0, 0). Третій переслідувач знаходиться на початку гри в позиції (0, 20). Четвертий переслідувач знаходиться на початку гри в позиції (20, 20). П’ятий переслідувач знаходиться на початку гри в позиції (0, 10). Втікач знаходиться на початку гри посеред переслідувачів в позиції (10, 10). Переслідувачі мають максимальне обмеження за швидкістю 1.0. Втікач має обмеження за швидкістю 0.62. Час початку гри дорівнює 0. Час

завершення гри дорівнює 10. Кількість кроків дискретизації дорівнює 50. Коефіцієнт $k_{pf} = 10$. Коефіцієнт $k_{ef} = 5$. Коефіцієнт $q_e = 2$. Коефіцієнт $q_p = 1$.

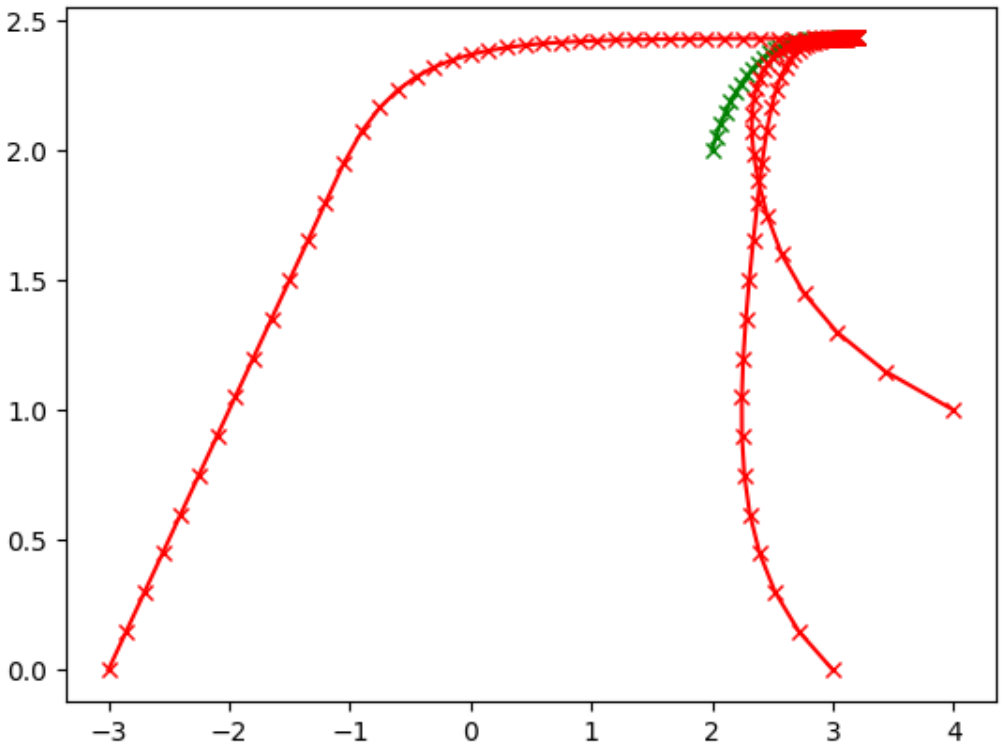

Рисунок 5.2 – Симуляція гри з трьома переслідувачами

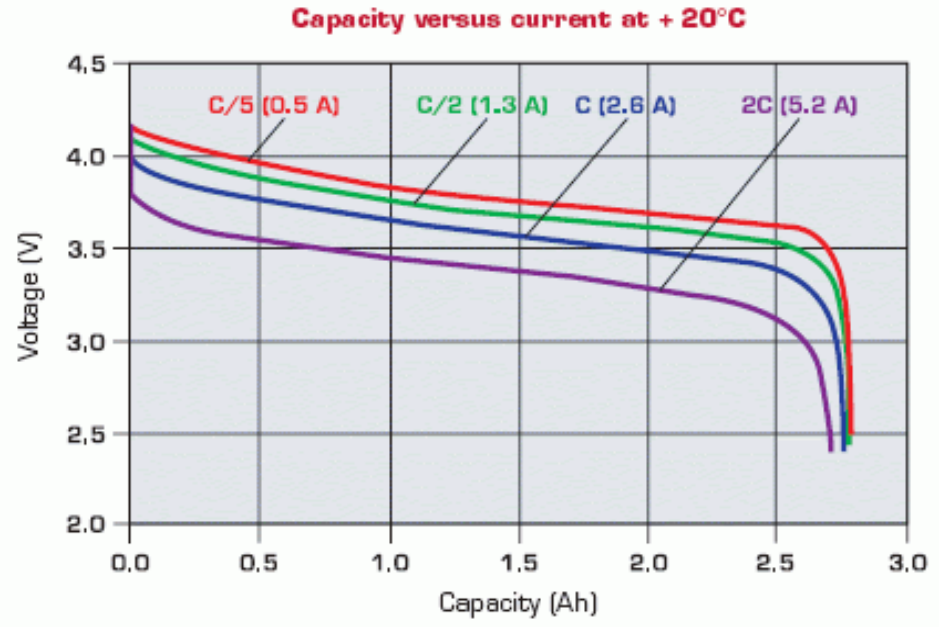

Рисунок 5.3 – Характеристика літієвих батарей

Результат визначеної вище симуляції подано на рис. 5.4.

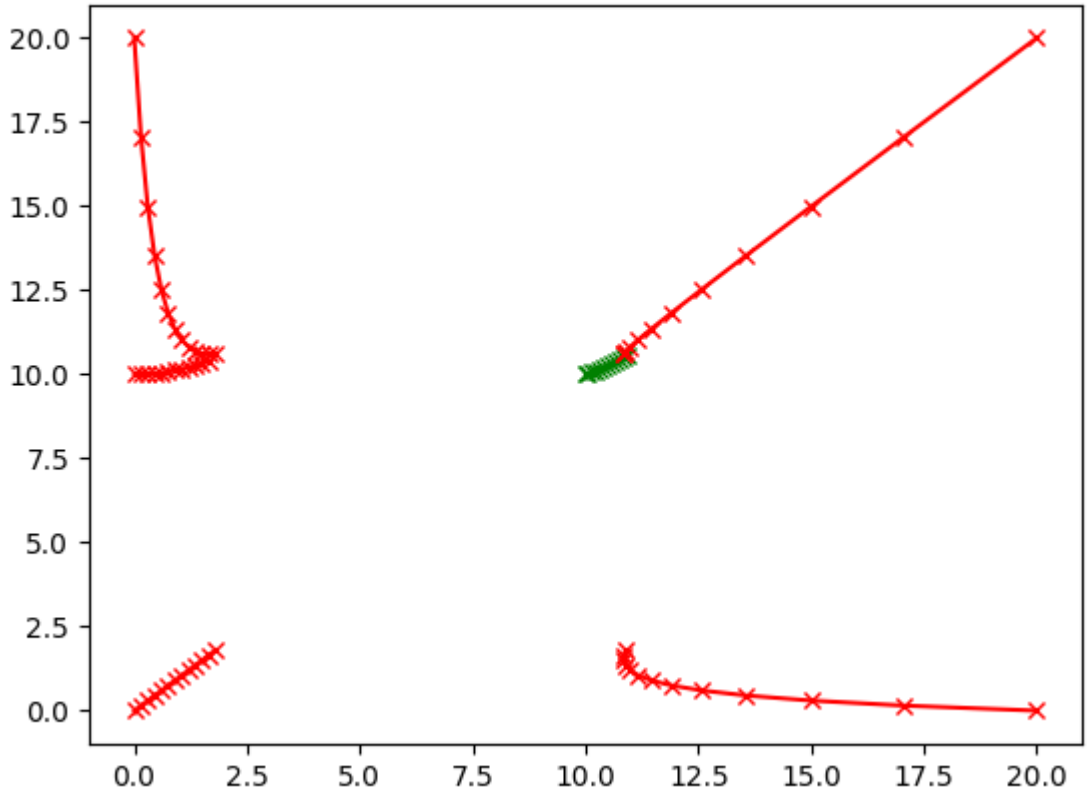

Рисунок 5.4 – Симуляція гри з п’ятьма переслідувачами

Далі розглянемо задачу переслідування двома переслідувачами в тривимірному просторі. Перший переслідувач знаходиться на початку гри в позиції (10, 0, 0). Другий переслідувач знаходиться на початку гри в позиції (0, 0). Втікач знаходиться на початку гри в пануючій позиції (5, 5, 3). Переслідувачі мають максимальне обмеження за швидкістю в горизонтальній площині 1.0, в вертикальній площині 0.5. Втікач має обмеження за швидкістю в горизонтальній площині 0.2, в вертикальній площині 0.1. Час початку гри дорівнює 0. Час завершення гри дорівнює 20. Кількість кроків дискретизації дорівнює 50. Коефіцієнт $k_{pf} = 10$. Коефіцієнт $k_{ef} = 5$. Коефіцієнт $q_e = 2$. Коефіцієнт $q_p = 1$. Результат подано на рис. 5.5.

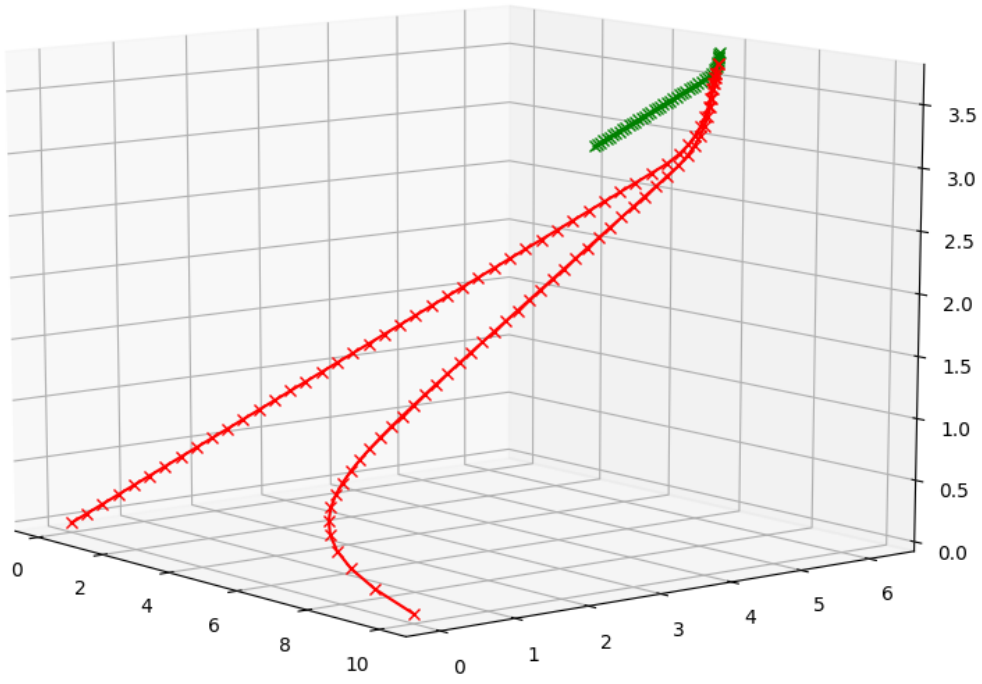

Рисунок 5.5 – Симуляція в тривимірному просторі

Розглянемо цей самий випадок з додаванням шуму величиною від 0.03 до 1.0. Результат подано на рис. 5.6.

Нарешті, розглянемо симуляцію з використанням програмного засобу ArduCopter SITL (Software in the loop), що симулює поведінку реального квадрокоптеру. Нехай гра розпочинається, коли всі гравці знаходяться в повітрі на висоті 20 метрів. Для цього вони спочатку мають вилетіти з точки початку координат та дістатися позицій зміщення, заданих в симуляції. Кількість переслідувачів дорівнює 2. Перший переслідувач знаходиться на початку гри в позиції (0, 0). Другий переслідувач знаходиться на початку гри в позиції (20, 0). Втікач знаходиться на початку гри в позиції (10, 10). Переслідувачі мають максимальне обмеження за швидкістю 10. Втікач має обмеження за швидкістю 3. Зауважимо, що ці обмеження ще не мають реальних одиниць виміру і будуть масштабовані до метрів на секунду, чи значення тяги двигунів (англ. throttle). Час початку гри дорівнює 0. Час завершення гри дорівнює 120.

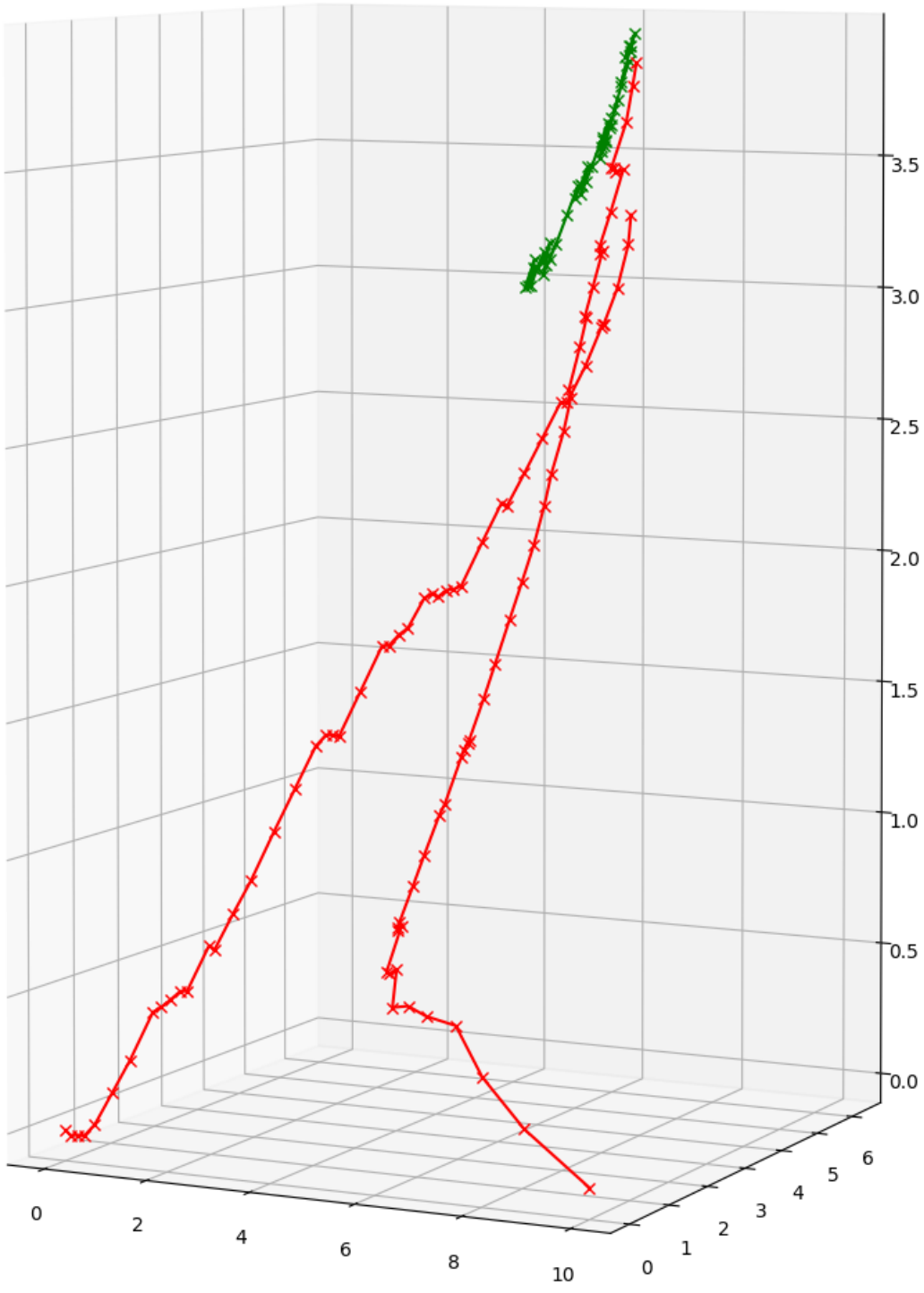

Рисунок 5.6 – Симуляція з додаванням шуму

Кількість кроків дискретизації дорівнює 10. Коефіцієнт $k_{pf} = 10$. Коефіцієнт $k_{ef} = 5$. Коефіцієнт $q_e = 2$. Коефіцієнт $q_p = 1$. Зауважимо також, що шляхи, що

будують контролери, сумісні з протоколом MAVLINK, можуть бути побудовані або як послідовність прямих, або як складний NURBS-сплайн (англ. Non-uniform rational B-spline). В нашій симуляції обрано другий варіант, тож кількість точок маршруту може бути обмежена десятьма. Результат попередньої симуляції засобами розробленого програмного засобу подано на рис. 5.7. Для спостереження за реальним, або симульованим дроном, що підтримує протокол MAVLINK, використовується программа APM Planner 2. На рисунках можна побачити місце ймовірного перехоплення між дорогами, нижче дерев. Це дванадцята точка шляху, перші дві відповідають за зайняття попередньої позиції, а останні десять отримані з симуляції за формулою

$$p = p_0 + p_s * 10^{-5},$$

де p – розрахована позиція в GPS координатах,

p_0 – початкова координата дрону,

p_s – координата дрону, отримана з математичної моделі.

Тож, в цілому цей сценарій використання можна узагальнити до наступних кроків:

– розрахунок параметрів математичної моделі;

– пристосування моделі до емуляції;

– безпосередня емуляція.

Передбачається, що наприкінці гри переслідувачем буде або використаний деякий прилад для перехоплення втікача,такий як сітка або підвісний таран, або просто застосована тактика “камікадзе”, тобто прямого зіткнення з втікачем.

Для підключення до моделі симуляції необхідно додати TCP, UDP, або Serial адресу у відповідне поле. Після цього позначка у верхній правій частині екрану стане зеленою з написом “Disconnect”. З огляду на безпечність використання дронів, для запобігання травмуванню оператора лопастями гвинтів, дрон може знаходитись в пасивному (disarmed) або активному (armed) стані.

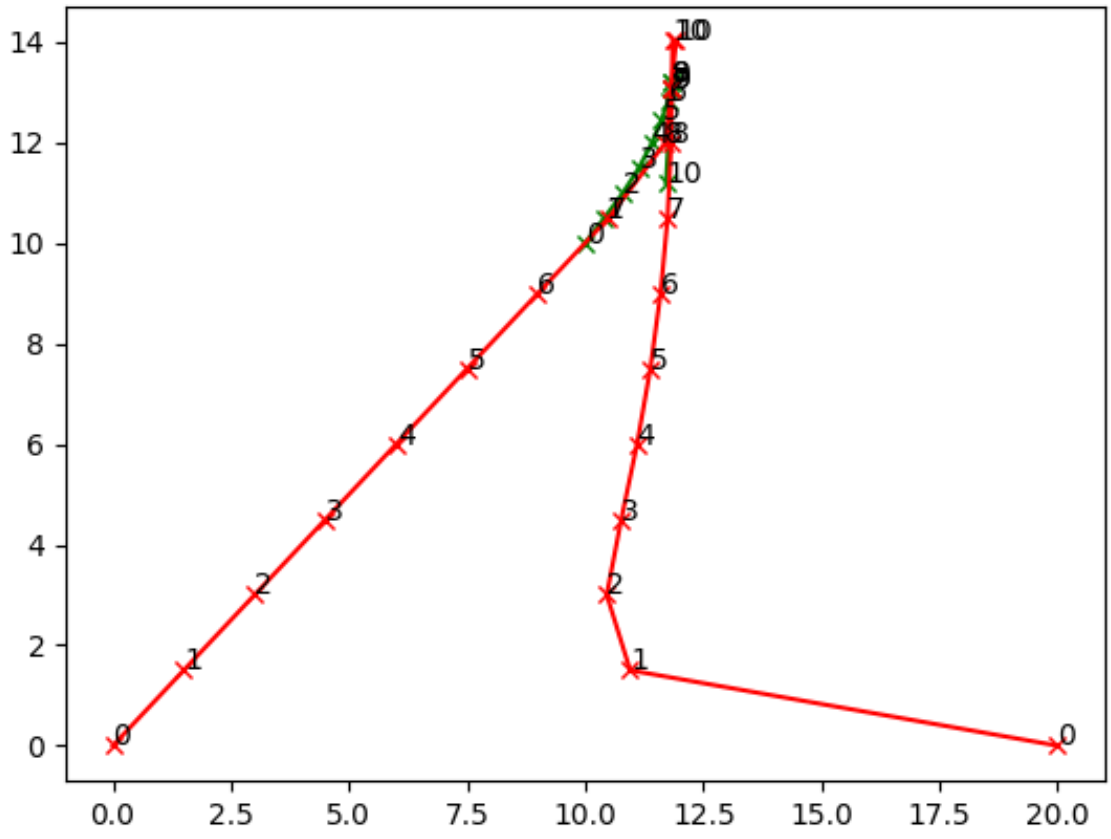

Рисунок 5.7 – Розрахунок математичної моделі гри

Скріншот роботи програми для першого переслідувача наведено на рисунку 5.8. Скріншот роботи програми для другого переслідувача наведено на рисунку 5.9.

Для ручного переведення дрону в той чи інший стан на вкладці Actions існує відповідна кнопка. Додатково існує паралельна система станів, основні зя яких наведено на восьми кнопках в нижній лівій частині екрану. В нашій симуляції використовується стан Loiter (активна автоматична стабілізація).

При завершенні симуляції, дрон можна посадити або перевівши його у стан RTL (англ. Return to Land), або у стан Land. Швидкість посадки залежить від налаштованого параметру в EEPROM (внутрішня енергонезалежна пам’ять дрону).

Рисунок 5.8 – Шлях першого переслідувача в APM Planner 2

Рисунок 5.9 – Шлях другого переслідувача в APM Planner 2

Скріншот роботи програми для втікача наведено на рисунку 5.10.

Рисунок 5.10 – Шлях втікача в APM Planner 2

Треба також зауважити, що в цьому випадку не враховується можливість такого розвитку ситуації гри, коли втікач помічає наявність переслідування і перемикає дрон на ручне керування. Втім, така ситуація може реально виникнути тільки за наявності:

– високотехнологічного дрону з охопленням камерами одночасно всіх 360 градусів горизонтальної площини;

– високотехнологічної станції керування втікачем, що її обладнано радаром достатньої дальності;

– використання втікача в межах зорового контакту оператора.

В простих випадках виконання будь-якої з цих умов не є реалістичним.

ВИСНОВКИ

В результаті роботи виконано постановку задачі, розглянуто стан її вирішення, підібрано математичну модель диференціальної гри переслідування декількома переслідувачами, зокрема функціонали якості, що залежать від параметрів гравців, та спосіб пошуку оптимального керування відносно цих функціоналів за допомогою розв’язання системи диференціальних рівнянь Ріккаті.

Зпроектовано та реалізовано програмний засіб, що використовує ці моделі для вирішення поставленої задачі. Продемонстровано, як отриманий засіб можна використовувати як самостійно для математичного моделювання, або в поєднанні з більш розвиненими середовищами емуляції БПЛА. Завдяки інтеграції з широко розповсюдженим стандартом керування безпілотними апаратами MAVLINK, цей засіб можна використовувати для керування не тільки літальними апаратами, а й наземними дронами чи підводними човнами. При цьому, його можна використовувати як з наземної станції керування, так й на бортових комп’ютерах.

В якості додаткового результату отримано також модель синтезу керування втікача, що дозволяє використовувати цю модель також іншою стороною конфліктного процесу для вирішення задачі про втікання.

В подальшому, цей засіб можна адаптувати з використанням більш складних сучасних моделей динамічних ігор, зокрема таких, що не використовують повну інформацію. Можна пристосувати програмний засіб до більш гнучкої адаптації до реального середовища. Модель можна розширити з метою динамічного обчислення оптимального часу завершення гри, а також для моделювання оптимального часу завершення гри у стохастичних системах простого руху з вінеровськими хвилюваннями[17].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Aristotelis Physica — Lipsiae: In aedibus B.G. Teubneri, 1879 — Liber Z, Capitulum 9, p.130

2. Ake B. Numerical methods for least squares problems — Philadelphia: Society for Industrial and Applied Mathematics, 1996 — 408 pp.

3. Фон Нейман, Дж., Моргенштерн О. Теория игр и экономическое поведение — М.: Наука, 1970 — 708 с.

4. Айзекс Р. Дифференциальные игры — М.: Мир, 1967 — 479 с.

5. Красовский Н.Н., Субботин А.И. — Позиционные дифференциальные игры — М.: Наука, 1974 — 456 с.

6. Понтрягин Л.С. — Избранные труды, Т. 2. М.:Наука, 1988 — 576 с.

7. Brogan, W. Modern Control Theory, 3rd edition — Upper Saddle River: Prentice Hall, 1991 — 653 pp.

8. Chikrii A. Conflict-controlled processes — Dordrecht:Springer Netherlands, 1997 — 404 pp.

9. Балакришнан А. Введение в теорию оптимизации в гильбертовом пространстве — М.: Мир, 1974 — 259 с.

10. Vlasenko, L.A., Rutkas, A.G. On a differential game in a system described by an implicit differential-operator equation // Diff. Equations 2015 №51, pp. 798–807.

11. Redulla A., Surya P., Simulating differential games with improved fidelity to better inform cooperative & adversarial two vehicle UAV flight // IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR) 2018, pp. 130-136.

12. Lin, Qu, Simaan. A Design of Entrapment Strategies for the Distributed Pursuit-Evasion Game // IFAC Proceedings Volumes 2011 №44/1, pp. 9334-9339.

13. Johnson P., Numerical solution methods for differential game problems — Cambridge:MIT, 2009 — 98 pp.

14. Wes McKinney, Python for Data Analysis — Cambridge:O'Reilly Media, 2012 — 470 pp.

15. Koubâa et al. Micro Air Vehicle Link (MAVlink) in a Nutshell: A Survey // IEEE Access, vol. 7, 2019, pp. 87658-87680

16. Robert Johansson, Numerical Python: Scientific Computing and Data Science Applications with Numpy, SciPy and Matplotlib — Apress, 2019 — 700 p.

17. Vlasenko, L.A., Rutkas, A.G., Chikrii, A.A. On a Differential Game in a Stochastic System // Proceedings of the Steklov Institute 2020 №309, pp. S185-S198