Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Інформаційних управляючих систем
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)
(рівень вищої освіти)

Дослідження ройових методів в задачах децентралізованого управління групою безпілотних літальних апаратів

(тема)

Виконав: студент 2 курсу, групи ІУСТм-19-1
Рогач В.Д.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Інформаційні управляючі системи та технології
(повна назва освітньої програми)

Керівник доц. Кудрявцева М.С.
(посада, прізвище, ініціали)

Допускається до захисту
зав. кафедри _____ Петров К.Е.
(підпис) (прізвище, ініціали)

2020 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Інформаційних управляючих систем
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Інформаційні управляючі системи та технології
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

(підпис)

« » _____ 2020 р.

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ АТЕСТАЦІЙНУ РОБОТУ

студентові Рогачу Володимиру Дмитровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження ройових методів в задачах децентралізованого управління групою безпілотних літальних апаратів
затверджена наказом по університету від "27" жовтня 2020 р. № 1455 Ст
2. Термін здачі студентом закінченої роботи 9 грудня 2020 р.
3. Вихідні дані до роботи Науково-технічні публікації, статті, результати досліджень щодо управління безпілотними літальними апаратами
4. Перелік питань, що потрібно опрацювати в роботі
1 Огляд і аналіз існуючих методів, моделей і алгоритмів управління безпілотними літальними апаратами
2. Дослідження методів розробки децентралізованих систем
3. Реалізація ройового методу в задачах децентралізованого управління
4. Практична реалізація
5. Перелік графічного матеріалу із зазначенням обов'язкових рисунків
Рисунок 1 – Ройові алгоритми. Рисунок 2 – Структурна схема децентралізованого управління групою БПЛА. Рисунок 3 Точки знаходження БПЛА та їх околиці. Рисунок 4 - Робота алгоритму рою бджіл при різних

наборах параметрів. Рисунок 5-7 Адаптивні алгоритми управління групою БПЛА. Рисунок 8-12 Екранні форми

КАЛЕНДАРНИЙ ПЛАН

Номер	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	2.11.2020	
2	Аналіз предметної галузі	03.11.2020 – 04.11.2019	
3	Постановка задачі та узгодження з керівником	05.11.2020 – 06.11.2020	
4	Огляд і аналіз існуючих методів, моделей і алгоритмів управління безпілотними літальними апаратами	07.11.2020 – 10.11.2020	
5	Дослідження методів розробки децентралізованих систем	11.11.2020 – 20.11.2020	
6	Р7ізація ройового методу в задачах децентралізованого управління групою безпілотних літальних апаратів	21.11.2020 – 25.11.2020	
7	Практична реалізація	26.11.2020 – 30.11.2020	
8	Підготовка пояснювальної записки та графічного матеріалу	01.12.2020 – 09.12.2020	
9	Надання пояснювальної записки на перевірку керівнику	09.12.2019	
10	Захист перед ЕК	16.12.2019	

Дата видачі завдання ____02.11.2020 р.____

Студент _____
(підпис)

Керівник роботи _____ доц. Кудрявцева М.С.
(підпис) (посада, прізвище, ім'я, по батькові)

РЕФЕРАТ

Пояснювальна записка містить 90 сторінки, 12 рисунків, 36 джерел.

ДЕЦЕНТРАЛІЗОВАНЕ УПРАВЛІННЯ, БЕЗПІЛОТНИЙ ЛІТАЛЬНИЙ АПАРАТ, ДРОН, ГРУПА БПЛА, АДАПТИВНИЙ АЛГОРИТМ, РОЙОВИЙ ІНТЕЛЕКТ, БДЖОЛИНИЙ АЛГОРИТМ, РОЗВІДНИК, НАГЛЯДАЧ, РОБІТНИК

Метою атестаційної роботи є дослідження ройових методів, методів децентралізованного управління групою безпілотних літальних апаратів, адаптивних алгоритмів для ефективного вирішення завдань в умовах неконтрольованих ситуацій.

Об’єктом дослідження є група безпілотних літальних апаратів (БПЛА).

Предметом дослідження є методи колективного розуму, ройові методи.

Методи дослідження – методи системного та об’єктно-орієнтованого аналізу, методи колективного розуму, ройові методи.

У рамках наукових результатів дослідження представлено структурну схему децентралізованого управління групою БПЛА, метод розробки адаптивного алгоритму управління групою БПЛА, адаптивні алгоритми групи БПЛА.

У рамках практичних результатів наведено аналітичну реалізація бджолиного алгоритму управління групою БПЛА, реалізація алгоритму групового польоту на мові Python, програмно-апаратна реалізація алгоритмів за допомогою програмного забезпечення AeroSIM Radio Control Training Simulator, яка моделює поведінку дронів у групі.

ABSTRACT

The certification work contains 90 pages, 12 figures, 36 references.

DECENTRALIZED CONTROL, UNMANNED AIRCRAFT, DRON, UAV GROUP, ADAPTIVE ALGORITHMS, SWEEP INTELLIGENCE, BEE ALGORITHM, SCOUT, OBSERVER, WORKER

The purpose of the certification work is the study of swarm methods, methods of decentralized control of a group of unmanned aerial vehicles, adaptive algorithms for effective problem solving in uncontrolled situations.

The research object is group of unmanned aerial vehicles (UAVs).

The subject of research is the methods of collective intelligence, swarm methods.

Research methods - methods of system and object-oriented analysis, methods of collective intelligence, swarm methods.

Within the framework of the research results, block diagram of decentralized control of a UAV group, a method for developing an adaptive control algorithm for a UAV group, adaptive algorithms for a UAV group.

Within the framework of practical results, an analytical implementation of the bee control algorithm for a group of UAVs, an implementation of a group flight algorithm in Python, software and hardware implementation algorithms using softwareAeroSIM Radio Control Training Simulatorwhich simulates the behavior of drones in a group.

ЗМІСТ

ВСТУП

На даний момент технології, які використовуються в безпілотних літальних апаратах, такі як бортові комп'ютери, електродвигуни, акумулятори та в першу чергу програмне забезпечення, дуже швидко розвиваються. Сучасними квадрокоптерами, що знаходяться до однієї години в повітрі, легко маневрувати між перешкодами і піднімати важке обладнання, може керувати навіть користувач з малим досвідом.

БПЛА під управлінням оператора, який стежить за чітким виконанням роботи, може виконувати завдання різного роду складності: зйомка матеріалів для фільму, процес доставки вантажів, сканування місцевості, а також пошуково-рятувальної операції, взяття проб води, повітря, грунту.

Для вирішення небезпечних для людини задач група БПЛА має переваги щодо поодинокого квадрокоптеру, такі як: взаємозамінність у випадках позаштатних ситуацій, розширена інформованість про навколишнє середовище за рахунок комунікації всередині групи, підвищена надійність і швидкість виконання завдань і здатність колективно виконувати більш складні завдання.

Однак найбільшу цінність представляє можливість децентралізованого управління групою БПЛА, коли єдиний центр управління пристроями або оператор відсутні. Таке рішення дозволяє вирішувати стратегічно важливі завдання.

В даній роботі розглядається задача децентралізованного управління групою безпілотних літальних апаратів для ефективного вирішення завдань в умовах неконтрольованих ситуацій з використанням ройових методів.

Вже давно люди почали цікавитися так званою «ройовою поведінкою», тобто яким чином птахи летять на південь величезними косяками, не збиваючись з курсу. Як величезні колонії мурах працюють так злагоджено і зводять структури, що за складністю не поступаються сучасним мегаполісам?

Як бджоли можуть так точно визначати і добувати в необхідному для всієї колонії їжу? Виявилося, що штучний інтелект за своїми властивостями не обов'язково може бути схожий на людину, більш того, вже існують і використовуються моделі, які взагалі не мають ніякого відношення до людей і імітують абсолютно іншу форму розуму.

1 ОГЛЯД І АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ, МОДЕЛЕЙ І АЛГОРИТМІВ УПРАВЛІННЯ БЕЗПІЛОТНИМИ ЛІТАЛЬНИМИ АПАРАТАМИ

1.1 Загальний анализ існуючих видів алгоритмів і методів їх реалізації

Алгоритм −це система точних і зрозумілих розпоряджень про зміст і послідовності виконання кінцевого числа дій, необхідних для вирішення будь-якої задачі даного типу.

Алгоритми мають наступні властивості [1]:

− універсальність (масовість) −застосовність алгоритму до різних наборів вихідних даних;

− дискретність − процес вирішення завдання за алгоритмом розбитий на окремі дії;

− кінцівку − кожна з дій і весь алгоритм в цілому обов'язково завершуються;

− результативність − по завершенні виконання алгоритму обов'язково виходить кінцевий результат;

− здійсненність (ефективність) − результату алгоритму досягається за кінцеве число кроків;

− детермінованість (визначеність) − алгоритм не повинен містити приписів, зміст яких може сприйматися неоднозначно. Тобто один і той же припис після виконання повинен давати один і той же результат;

− послідовність − порядок виконання команд повинен бути зрозумілий виконавцю і не повинен допускати неоднозначності.

Існують різні класи алгоритмів [2]:

− обчислювальні алгоритми, що працюють з порівняно простими видами даних, такими як числа і матриці, хоча сам процес обчислення може бути довгим і складним;

− інформаційні алгоритми, що представляють собою набір порівняно простих процедур, які працюють з великими обсягами інформації (алгоритми баз даних);

− керуючі алгоритми, що генерують різні керуючі впливи на основі даних, отриманих від зовнішніх процесів, якими алгоритми керують.

З моменту своєї появи обчислювальні машини вміли працювати з алгоритмами, тобто виконували набори інструкцій, що описують порядок дій комп'ютера для досягнення якогось результату. Проаналізуємо існуючі види алгоритмів.

У лінійному алгоритмі операції виконуються послідовно-лінійно, в порядку їх запису. Кожна операція є самостійною, незалежною від будь-яких умов. На схемі блоки, що відображають ці операції, розташовуються в лінійній послідовності. Реалізаціями лінійних видів алгоритмів є всілякі цикли, умови-розгалуження, виклики підпрограм і деяких інших сутностей, які в цілому не міняють послідовності виконання коду.

Наприклад, число місць на автопарковці *N*. Якщо число автомобілів, що в'їхали на парковку, *K* = *N*, то слід включити червоне табло «МІСЦЬ НЕМАЄ», інакше повинна горіти вивіска «ВІЛЬНІ МІСЦЯ».

Лінійні алгоритми чудово справляються як з простими, так і з досить складними завданнями.

Але є ускладнені завдання −виявлення взаємозв'язку в процесах, які на перший погляд ніяк між собою не пов'язані. Наприклад, чи є кореляція між сезонними змінами числа продажів смартфонів і кількістю аварій на дорогах. Для вирішення таких задач використовується лінійний регресійний аналіз, який дозволяє знайти залежності однієї величини від іншої, займається пошуком моделі цього зв'язку, вираженої у функції регресії [3].

Але, якщо випадкових факторів, між якими необхідно виявити відповідність, стає занадто багато, регресійний аналіз призводить до великої кількості рівнянь і неоднозначності отриманих результатів. Крім того, вимоги до вихідних даних в регресійному аналізі доволі складні.

Для вирішення задачі прогнозування значень показників від часу використовуються тимчасові ряди (наприклад, прогнозування продажів на підставі аналізу даних за минулі періоди), але такі методи досить складні для дослідника, потребують теоретичних знань і припускають наявність репрезентативної вибірки великого обсягу за однакові проміжки часу [4].

Складнощі з лінійними алгоритмами виникають ще й в тому випадку, коли людина не знає точно, як повинен виглядати кінцевий результат обчислень. В цьому випадку для вирішення завдань використовуються нейронні мережі. Модель на базі нейромереж максимально близько моделює роботу людського мозку, де між нейронами встановлюються зв'язки, а в процесі навчання визначається важливість (вага) цих зв'язків. Така модель дозволяє обробляти величезні обсяги даних і отримувати набагато більш точні результати, ніж регресійний аналіз.

Можливість навчання − одна з головних переваг нейронних мереж перед традиційними алгоритмами. В процесі навчання нейронна мережа здатна виявляти складні залежності між вхідними даними і вихідними, а також виконувати узагальнення. Це означає, що в разі успішного навчання мережа зможе повернути вірний результат на підставі неповних, частково спотворених або відсутніх в навчальній вибірці даних [4].

Моделі на базі нейромереж індивідуальні, як людина. Потужний комп'ютер і заздалегідь підготовлене середовище для вирішення завдання є обов'язковими для його функціонування. Об'єднувати обчислювальні зусилля декількох нейромереж необхідно вручну, причому якщо поставлена задача змінилася, то всю «мережу мереж» доведеться перебудовувати і перенавчати.

Однак в природі зустрічається явище, коли живі або не зовсім живі створіння, які володіють порівняно низьким індивідуальним інтелектом (або взагалі його відсутністю), об'єднуються в спільноти, які ведуть себе набагато розумнішими, ніж кожен з учасників групи окремо. Йдеться про різновиди штучного інтелекту − ройовий інтелект (Swarm Intelligence).

Ройовий інтелект являє собою колективну поведінку окремих осіб (агентів) в системі, що самоорганізується, без вираженого центру управління, тобто в децентралізованої групі [5].

Діючих агентів в сучасній теорії РІ часто називають «боїди». Вперше цей термін з'явився в 1986 році. Boids - назва програми, розробленої дослідником Крейгом Рейнольдсом, яка імітує поведінку птахів, що збилися в зграю. Назва «boid» утворено від «birds id», або «пташині об'єкти».

Одиночний боїд слідує простим правилам, які закладені в нього природою (фізичними законами або генетикою). Незважаючи на відсутність командного центру, який вказував би, що саме робити кожному з боїдів, їх випадкові взаємодії призводять до того, що загальна поведінка групи стає виражено інтелектуальною. При цьому кожен конкретний боїд не обов'язково веде себе розумно. Таку поведінку демонструють, наприклад, мурахи, бджоли, терміти, вовки, краплі води, імунна система людини, кажани і т.ін.

Кожен конкретний мураха або бджола швидше за все навіть не підозрює, що колонія, учасником якої він є, поводиться розумно. З іншого боку, вовки і летючі миші досить інтелектуальні індивіди. Однак, говорити про наявність чи відсутність інтелекту у краплі води взагалі не представляється можливим, оскільки це предмет неживої природи. Проте всі ці агенти: мураха, бджола, вовк, крапля води, об'єднавшись в однорідні групи (мурахи з мурахами, бджоли з бджолами і т.ін.), демонструють більш розумну поведінку, ніж кожен з них окремо.

До переваг РІ можна віднести відразу кілька аспектів:

- надійність виконання проектів. Проекти, над якими працює колонія, не залежать від ефективності участі кожного конкретного агента. Більш того, навіть якщо деякі особини роблять щось неправильно або взагалі гинуть, це ніяк не відбивається на кінцевому результаті;
- гнучкість в прийнятті рішень. Група агентів, об'єднана в рамках ройового інтелекту, здатна надзвичайно швидко реагувати як на зовнішні загрози, так і на внутрішні проблеми в колонії;

– самоорганізація – відсутність центру контролю і командування дозволяє колонії приймати швидкі, вірні і безпрецедентно гнучкі рішення і бути повністю незалежною від якості управління;

– масштабованість – управління невеликою групою людей вимагає певних зусиль, а керівництво великою організацією – величезних зусиль. Чим більше людей задіяно в організації, тим нижче ефективність всього цього об'єднання. Колонія боїдів, що діє за принципами ройових методів, ефективна завжди. Вона може бути утворена як декількома агентами, так і десятками - сотнями тисяч. Рій будь-якого масштабу не вимагає ніякого керівництва і максимально раціонально сам вирішує всі поставлені перед ним завдання;

– прогнозна здатність. Рій вміє «передбачати» дії подразника. Оскільки всі рішення приймаються в колонії децентралізовано, вони можуть випереджати поведінку середовища або подразника. Можливі випадки, коли рій починає правильно реагувати на якусь подію з майбутнього до того, як ця подія починає відбуватися;

– багатозадачність - самоорганізація рою і незалежність дій агентів роблять можливим виконання декількох завдань одночасно.

Недоліки ройових методів є наслідком переваг. Логічно передбачити реакцію системи, чия робота базується на ройових методах, надзвичайно складно, оскільки функціонування кожного окремого агента стохастичне і логічно не пов'язане з поведінкою всієї системи в цілому. Колонія чутлива до будь-яких зовнішніх і внутрішніх змін умов і правил взаємодії. Навіть невеликі відмінності в умовах поставлених завдань призводять до помітної зміни поведінки рою на різних рівнях, при цьому внести зміни саме в логіку поведінки всього рою, а не окремого агента майже неможливо.

Дослідження ройових методів активно ведуться по всьому світу. Прикладами побудови поведінкових моделей за допомогою РІ можуть бути, наприклад, алгоритм штучної бджолиної колонії (англ. Artificial bee colony algorithm, ABC). Вперше він описаний турецьким дослідником Дервіс Карабога в 2005 році [5]. За допомогою трьох жорстко закріплених ролей - «працівник»,

«спостерігач» та «розвідник» алгоритм дозволяє реалізувати складні механізми вирішення завдань щодо оптимізації.

Ще одна цікава різновид запропонована іранськими вченими Сейдалі і Сейєдом Мохаммадом Мірджалілі і австралійцем Андрю Льюїсом в 2014 році [6]. Вони досліджували розподіл ролей і поведінку вовків у зграї і на основі цього запропонували нову модель ройового інтелекту.

Окремо стоїть інтелектуальний алгоритм крапель води (англ. IWD). Це модель, згідно з якою водяні краплі знаходять оптимальний шлях до місця призначення, змінюючи русло річки. Цьому сприяють три важливі параметри. За рахунок власної швидкості краплі здатні захоплювати грунт з дна річки, і чим вище швидкість, тим більша кількість грунту кожна крапля здатна забрати з собою, відповідно, тим вільніше стане шлях для подальших агентів. Швидкість потоку зростає там, де немає грунту, яку необхідно розчищати. Оптимальним можна назвати такий шлях, на якому зустрічається найменше грунту і де можна розвинути найбільшу швидкість. За допомогою IWD можна реалізувати стратегію оптимізації, коли випадкові агенти інтелектуально взаємодіють один з одним таким чином, що спільними зусиллями змінюють русло річки і створюють оптимальний шлях.

Ройові алгоритми вже застосовується в безлічі областей: управління світлофорами і оптимізація трафіку на автошляхах часто побудовані з застосуванням даних принципів.

Великий потенціал в обробці Big Data, особливо якщо даних багато, зв'язок між ними багатовимірний, самі ці дані динамічно змінюються, а кінцева мета обробки постійно варіюється.

В інформаційних технологіях ройові методи використовуються для побудови систем складної маршрутизації, наприклад, AntNet − протокол маршрутизації мережевого трафіку, побудований на базі аналізу поведінки мурах. «Мурахи-розвідники», що йдуть першими, за допомогою феромонів залишають інформацію наступним за ними «робочим» мурашкам. Таким чином, розвідники розмічають шлях, відрізаючи непотрібні ділянки, де для їх

побратимів немає ніякого інтересу. У протоколі маршрутизації використовуються «пакети-розвідники», єдина мета яких − визначити затримки в мережі, тобто виявити неоптимальні, «погані» маршрути, з найбільшим часом проходження сигналу. Інформацію «розвідників» успадковують «робочі пакети», на підставі яких динамічно перебудовуються таблиці маршрутизації.

У медицині методи РІ застосовуються для ранньої діагностики онкологічних захворювань, а в енергетиці ці підходи дозволяють оптимізувати розподіл електроенергії та знизити пікові витрати на 25-40%.

Найбільш зрозумілим «побутовим» застосуванням ройових методів, швидше за все, буде його використання в Internet of Things (IoT). Завдяки таким принципам, невеликі обчислювальні системи, вбудовані в побутову техніку (Embedded Systems), зможуть самоорганізовуватися в рій в рамках концепції «розумний будинок» для виконання більш складних завдань. В кінцевому підсумку це повинно буде привести до синергетичного зростання інтелектуальної потужності інтернету речей і, відповідно, створення більш комфортних умов для людини в межах його житла. Правда, залишається відкритим питання: наскільки такою системою можна буде управляти?

Найбільша дослідницька активність методів ройового інтелекту припадає на останні чотири-п'ять років. На сьогоднішній день важко навіть уявити всі області, де можуть бути задіяні ройові алгоритми. На сьогоднішній день поняття ройового інтелекту все тісніше переплітається з алгоритмами обробки і оптимізації великих кількостей і потоків інформації.

1.2 Аналіз ройових методів і алгоритмів реалізації

За аналогією з тим, як великі групи тварин/комах можна об'єднати одним загальним поняттям − рій, розвиваючись, інженери стали моделювати «ройовий інтелект» (РІ) − спроби зробити роботизовані, автоматичні і автоматизовані рої.

Одним з фундаментальних правил є той факт, що для ройового інтелекту необхідна (досить) велика кількість агентів, здатних взаємодіяти між собою і навколишнім середовищем локально.

Спостерігаючи за різними природними прикладами роїв, людство придумало різні моделі ройових методів, чия поведінка грунтувалося на різних шляхах взаємодії з навколишнім середовищем і між собою. Так само модель передбачає наявність так званої «мультиагентної системи», яка визначається як система, що складається з безлічі інтелектуальних агентів −програм, здатних самостійно протягом деякого, досить тривалого проміжку часу, виконувати поставлене завдання [7].

Проаналізуємо різноманітні алгоритми реалізації ройових методів, представлені на рисунку 1.1.

Рисунок 1.1− Ройові алгоритми

Проаналізуємо методи рою частинок, мурашиний і бджолині алгоритми, методи штучної імунної системи і менш розповсюджені методи: крапель води, алгоритм зозулі, метод альтруїзму, метод гравітаційного пошуку.

Метод рою часток (МРЧ) є методом чисельної оптимізації, що підтримує загальну кількість можливих рішень, які називаються частками або агентами, і переміщуючи їх в просторі до кращого знайденого в цьому просторі рішення, весь час знаходиться в зміні через знаходження агентами більш вигідних рішень [8].

Найперша комп'ютерна модель МРЧ була придумана в 1986 році Крейгом Рейнольдсом, який займаючись створенням графічної моделі, придумав досить прості правила поведінки для часток рою, діючи за якими, рій виглядав вкрай схожим на реальний аналог пташиного рою. Але класична модель МРЧ була створена в 1995 році Расселом Еберхарт і Джеймсом Кеннеді.

х модель відрізняється тим, що частки-агенти рою, крім підпорядкування певним правилам обмінюються інформацією один з одним, а поточний стан кожної частки характеризується місцем розташування частки в просторі рішень і швидкістю переміщення.

Якщо проводити аналогію зі зграєю, то можна сказати, що всі агенти алгоритму частки (в зграї вони можуть бути птахами або рибами), ставлять три досить простих завдання [8]:

− всі агенти повинні уникати перетину з оточуючими їх агентам;

− кожна частка повинна коригувати свою швидкість відповідно зі швидкостями оточуючих її частинок;

− кожен агент повинен намагатися зберігати досить малу відстань між собою і оточуючими його агентами.

Мурашиний алгоритм − оптимізаційний алгоритм з наслідуванням колонії мурах − один з найбільш ефективних алгоритмів для вирішення задач з пошуку маршрутів в графах і по знаходженню приблизних рішень для задачі комівояжера. Суть алгоритму полягає в застосуванні моделі функціонування колонії мурах до вирішення різних завдань [9].

Першим, хто зумів застосувати поведінку мурах для вирішення завдання про найкоротший шлях, став Марко Дориго на початку 90-х років XX століття. Пізніше також були вирішені багато оптимізаційні задачі за допомогою мурашиних алгоритмів. В даний час ці алгоритми показують кращі результати в деяких завданнях.

Алгоритм штучної бджолиної колонії −алгоритм ройового інтелекту, заснований на імітації поведінки колонії бджіл, може використовуватися в задачах оптимізації. Необхідною умовою для його застосування є наявність деякої топологічної відстані або її аналогу на області рішень [10].

Для збору нектару в бджолиній колонії застосовується два види бджіл: бджоли-розвідники і бджоли-робітники. Перші проводять дослідження території, навколишній вулику, на предмет наявності нектару. Після повернення у вулик, бджоли-розвідники повідомляють інформацію про

кількість нектару, напрямки його розташування і відстані до нього. Далі, в найбільш підходящі області вилітають робочі, причому, чим більше нектару в даній області, тим більше бджіл вилітає в неї. Крім збору меду, в їх завдання входить оновлення інформації про дану та прилеглі області.

Штучна імунна система – обчислювальна система, яка здатна адаптуватися і використовувати схожі з реальною імунною системою принципи і механіки [11].

Даний метод має три основні теорії, які описують його функціонування і взаємодію між елементами:

- теорія негативного відбору;
- теорія імунної мережі;
- теорія клональной селекції.

Інші алгоритми найменше поширені і рідко використовуються [12]:

- метод крапель води, що знаходять або мають найтісніший контакт, або найбільш оптимальні «шляхи для води», подібно річках;
- алгоритм зозулі - заснований на паразитуванні, подібно до того, як деякі види зозуль відкладали яйця в чужі гнізда, з часом навчившись імітувати кольори чужих яєць;
- метод альтруїзму, заснований на тому, що кожен агент "піклується" про оточуючих, не звертаючи уваги на себе;
- метод гравітаційного пошуку – укладено в дотриманні закону всесвітнього тяжіння (всі тіла притягуються одне до одного), а саме в пошуку найбільш якісних, «важких», агентів.

Таким чином, в даному пункті проаналізовано основні використовувані алгоритми ройового інтелекту: рою часток, мурашиний і бджолині алгоритми, методи штучної імунної системи і менш розповсюджені методи: крапель води, алгоритм зозулі, метод альтруїзму, метод гравітаційного пошуку. Всі провнвлізовані алгоритми актуальні на сьогоднішній день і несуть в собі безліч перспектив у розвитку.

1.3 Аналіз сучасних безпілотних літальних апаратів

Дрон і безпілотний літальний апарат (БПЛА) −це сучасні пристрої, як правило, програмно-апаратні, які стали невід'ємнім аспектом сучасної суспільності та поширюються у різних галузях, що дозволяють вирішувати в тому числі складні або небезпечні для людини задачі. Наприклад, у побутовій сфері − це задачі «Розумного дому», у комерційній сфері − це задачі постачання, у науковій або прикладній сфері – дослідження хіміко-біологічного складу зразків, у поліцейській та рятувальній сферах. Дуже корисним є використання дронів у містах для вирішення задач моніторингу місцевості.

Розглянемо основні визначення [13].

Дрон −це будь-який мобільний безпілотний транспортний засіб, тобто транспортний засіб без пілота на борту, заздалегідь запрограмований для виконання конкретного завдання в повітрі, на суші або під водою. Таким чином, термін «дрон» позначає не тільки сучасний рекреаційний квадрокоптер, але і морські (наприклад, підводний човен) або наземні автономні машини, а також хобі-машини з дистанційним управлінням. При застосуванні безпілотників для повітряних місій, цей термін в основному використовується засобами масової інформації для позначення простого літального апарату. Вони в основному доступні як готові рішення.

Безпілотний літальний апарат −це апарат, який літає без пілота, віддалено і повністю контролюється з іншого місця (земля, інший літак, космос) або запрограмований і повністю автономний. Це стосується великих безпілотних літальних апаратів з автопілотами, які знайшли широке застосування в цивільному і оборонному секторах.

Використання обох термінів залежить від особливостей літального апарату, а також сфери застосування.

Безпілотний літальний комплекс (БпЛК) – це сукупність декількох пристроїв, що включають сам безпілотний літальний апарат, а також обладнання, необхідне для його експлуатації: наземна станція керування, антенна система і катапульти.

У даній работе розглядаються пристрої БПЛА.

Переваги використання БПЛА [14]:

- швидкість - необхідна інформація по БПЛА може бути надана клієнту швидше за допомогою спеціальних камер і каналу даних, а не за допомогою традиційних методів зйомки, які іноді можуть бути повільними;
- економічність - дані виходять швидше, ніж за допомогою звичайних методів збору, тому різні місії можна виконувати за більш короткий проміжок часу, таким чином, вартість нижче, ніж при використанні інших апаратів;
- безпека - операторам БПЛА не потрібно перебувати на території, яка може бути небезпечною з різних причин;
- високий рівень точності - БпЛК може отримати високоточні дані. Це пов'язано з дублюванням інформації, отриманої під час польоту – чим більше перекриттів, тим докладніше записана інформація;
- для проведення місії або завдання, немає необхідності в залученні кваліфікованого бортового пілота, оскільки вони є безпілотними, а всі системи розроблені таким чином, що втручання людини в роботу мінімізується;
- БПЛА здатний літати і збирати інформацію під час дощу, хмарності, туману і темряви.

1.4 Аналіз існуючих систем централізованого та децентралізованого управління

За принципом управління для групового управління пристроями можно розглядати централізовані та децентралізовані системи.

Централізоване управління всіх елементів групи здійснюється з єдиного кординаційного центру, політикой чітко розмежуваються види діяльності або операції.

Децентралізоване управління передбачає відсутність єдиного керуючого центру формування координаційних команд для кожного з елементів групи.

Розглянемо існуючі системи управління групами пристроїв централізованого і децентралізованого управління.

В роботі [15] розглядається задача автоматизації роботи групи квадрокоптерів на місцевості, в якій дрони здійснюють зліт, посадку і політ по заданій траєкторії. Квадрокоптери під управлінням оператора можуть виконувати завдання різного роду складності, зйомка матеріалів для фільму або пошукова операція, проте найбільшу цінність представляє можливість автоматизації виконуваного завдання, коли потрібен тільки один оператор, який стежить за чітким виконанням роботи, спостерігач, тобто розглядається система централізованого управління. Можна автоматизувати процес доставки вантажів, сканування місцевості, а також пошуково-рятувальні операції.

В роботі проведено огляд групового управління, зокрема методи групового управління і стратегії, що застосовуються на практиці. На основі аналізу групового управління обраний метод стройового руху квадрокоптера.

Робота [16] присвячена актуальній проблемі математичного моделювання і теорії управління: завдання децентралізованого управління мультиагентною системою, що складається з агентів, що моделюють автономних роботів, з метою забезпечення руху групи агентів ладом, що має задану геометричну форму.

Для рівномірного розподілу агентів в зоні виконання місії, підтримки стійкого зв'язку всередині групи і уникнення зіткнень роботи повинна під час руху дотримуватися деяка геометрична структура ладу (певне розташування

відносно один одного або відносно центру мас групи, що утворить певну геометричну фігуру).

Для вирішення даного завдання використовуються такі підходи:

– задати заздалегідь бажану відстань між парами агентів і застосувати теорію жорсткості графів [17,18];

– задати бажане положення агента щодо його сусідів набором векторів і скористатися правилами консенсусу (усереднення) [19-21];

– в кожен момент часу передавати агентам інформацію про становище і напрямки руху віртуальної формації, на підставі чого кожний агент може сконструювати віртуального лідера і слідувати за ним [22-23].

При розробці алгоритмів на основі одного з трьох перерахованих вище підходів, як правило, виникають суттєві труднощі в обробці наступних позаштатних ситуацій:

– вихід з ладу одного або декількох агентів з подальшою втратою можливості передачі інформації, особливо агента-лідера;

– придбання зв'язку з агентом, який у міру руху до цільової точки виявився в зоні чутності інших агентів;

– неполадки зв'язку з координаційним центром (в підходах, в яких передбачається безперервно або періодично передавати з даного центру агентам значиму для управління інформацію).

В роботі [23] представлена математична модель, що описує рух групи роботів, і розроблені повністю децентралізоване правило управління і алгоритм, які дозволяють здійснювати ефективне управління рухом агентів зі збереженням геометричної форми ладу (певних взаємних відстаней відносно один одного) за умов повної автономності агента і можливості отримання інформації тільки від своїх найближчих сусідів. При цьому правило управління працездатне в умовах виникнення описаних позаштатних ситуацій.

Пропонується оригінальне децентралізоване правило управління, що поєднує в собі деякі принципи консенсусу (усереднення), а також елементи підходу з використанням віртуальних лідерів.

Використовуються нові для області елементи: сформульовані критерії побудови ладу, відповідного заданій геометричній структурі з необхідною точністю, з опорою на введені в даній роботі поняття про геометричну структуру ладу і віртуальних лідерів, враховується сукупність різних обмежень. Відмінною особливістю розроблених правил управління є те, що набір віртуальних лідерів індивідуальний для кожного агента, агент розраховує координати віртуальних лідерів і їх пріоритети за оригінальними розробленим правилам, що дозволяє вибирати місце в геометричній структурі ладу динамічно без фіксованої прив'язки агента до того чи іншого положення в геометричній структурі ладу.

1.5 Постановка задачі

Для досягнення конкретної мети, що стоїть перед групою безпілотних літальних апаратів, в випадку централізованого управління кожен дрон може виконувати заздалегідь відому певну послідовність дій.

У разі ж децентралізованого управління, яке є більш гнучким, ця послідовність повинна бути знайдена системою управління групою дронів в процесі досягнення мети. Причому дії БПЛА групи повинні бути певним чином узгоджені.

Таким чином, виникає задача створення адаптивного алгоритму управління групою пристроїв. Це завдання полягає в реалізації системою управління агентами (БПЛА) послідовності дій всіх учасників групи під час реалізації процесу досягнення поставленої мети.

Для реалізації даної мети в роботі:

− наведено обгрунтування децентралізованого управління групою БПЛА;

- проведено дослідження ройових методів та алгоритмів, обрано бджолиний метод для реалізації системи управління групою БПЛА;
- наведено структурну схему децентралізованого управління групою БПЛА;
- представлено метод розробки адаптивного алгоритму управління групою БПЛА;
- розроблені адаптивні алгоритми управління групою БПЛА (алгоритм патрулювання БПЛА-«розвідників», БПЛА-«спостерігачів» та алгоритм БПЛА-«робітників», які безпосередньо виконують поставлене завдання);
- представлена програмно-апаратна реалізація.

2. ДОСЛІДЖЕННЯ МЕТОДІВ РОЗРОБКИ ДЕЦЕНТРАЛІЗОВАНИХ СИСТЕМ

2.1 Обгрунтування децентралізованого управління групою БПЛА

Найбільшою перевагою беспілотних літальних апаратів є можливість виконання задач, що можуть бути дуже небезпечними або неможливими для виконання людиною. У цьому сенсі використання БПЛА більш зручний для виконання задач з фото- або відео-моніторингу.

До недоліків БПЛА можна віднести лімітовану вантажопід'ємність та обмеження на час польоту. Що стосується умов використання БПЛА, то проблемою може бути наявність різних перешкод. Але ці недоліки можуть бути вирішені шляхом використання групи літальних апаратів. У цьому підході існують певні проблеми, а саме – координація та взаємодія певних одиниць БПЛА у групі та дистанція зв'язку з пультом керування.

Актуальність даної роботи полягає в забезпеченні контролю над групою БПЛА, а саме – група БПЛА має певні переваги над поодиноким БПЛА:

- підвищена інформативність про навколишнє середовище за рахунок комунікації у самій групі;
- взаємозамінність БПЛА у зв'зку з позаштатною ситуацією;
- підвищена надійність;
- швидкість виконання задач;
- можливість колективно виконувати більш складні задач.

При виконанні поставлених задач група БПЛА може зіткнутися з певними перешкодами, які потрібно уникнути усією групою, або зменшити кількість виведених з ладу БПЛА. Після виконання завдання потрібно (якщо це поставлено задачею) повернутися до початкового пункту відправлення [24].

Оскільки поставлену задачу виконує група БПЛА, то потрібно використовувати механізм передачі даних про загрозу між певними одиницями

БПЛА. Це потрібно задля того щоб зменшити шанс зіткнення з загрозою кожного БПЛА у групі.

Більше того, якщо один з БПЛА у групі втратить зв’язок з оператором, який керує ним (або усією групою), потрібно, щоб дана одиниця могла продовжувати виконання поставленої задачі завдяки іншому БПЛА у групі, який має зв’язок з оператором. Тобто, якщо БПЛА втратив зв’язок з оператором, за замовчуванням він почне повертатися у бік оператора, поки не відновить зв’язок.

У контексті виконання поставленої задачі це некоректна поведінка, оскільки йому потрібно знаходитися у групі БПЛА, та уникати загроз і досягати виконання поставленої задачі. Задля цього, сусідній БПЛА, який не втратив зв’язок з оператором, буде передавати йому інформацію щодо нових команд або загроз, яку «втрачений» БПЛА буде виконувати згідно своєї початкової задачі [24].

Необхідність групового управління обумовлена тим, що багато задач виконуються швидше, точніше та з меншими ресурсними витратами.

Зростанню актуальності використання децентралізованого управління для вирішення практичних завдань в значній мірі сприяє здешевлення елементної бази з одночасним зменшенням її розмірних характеристик, що робить можливим використання великих груп БПЛА.

Також підвищенню ефективності використання децентралізованих методів сприяє зростання складності завдань, покладених на БПЛА, збільшення частки невизначеності в умовах виконання місії, а також зростання рівня довіри до систем управління групами пристроїв.

Зазначені фактори сприяють необхідності оперативного прийняття рішення і максимізації самостійності дій БПЛА.

2.2 Дослідження ройових методів і алгоритмів

Проведемо детальне дослідження найбільш пристосовних методів для формалізації групових польотів БПЛА. Це метод рою часток, мурашиний та бджолиний алгоритми. На основі дослідження оберемо найкращий метод.

Алгоритм метод рою часток представлений на рисунку 2.1.

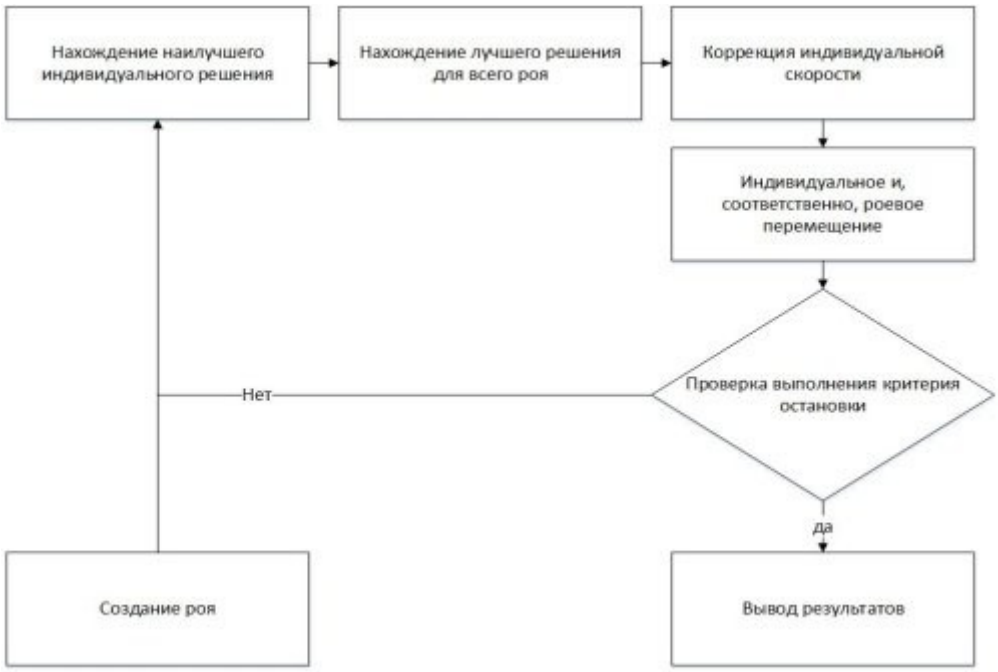


Рисунок 2.1 − Метод роботи МРЧ

Як видно з алгоритму рою частинок, − це ітеративний процес, який постійно перебуває в зміні. Для того, щоб зрозуміти, як функціонує алгоритм МРЧ, можна розглянути область пошуку у вигляді багатовимірного простору з агентами алгоритму. Спочатку все агенти знаходяться в випадкових місцях простору і з випадковим вектором швидкості. У кожній з точок, яку частка відвідує, вона розраховує задану функцію і фіксує оптимальне значення шуканої функції. Так само всі частинки знають місце розташування найкращого результату пошуку в усьому рої і з кожної ітерацією агенти коригують вектори

своїх швидкостей та їх напрямки, намагаючись наблизитися до найкращої точці рою і при цьому бути ближче до свого індивідуального максимуму [25].

На рисунку 2.2 наведено приклад роботи МРЧ.

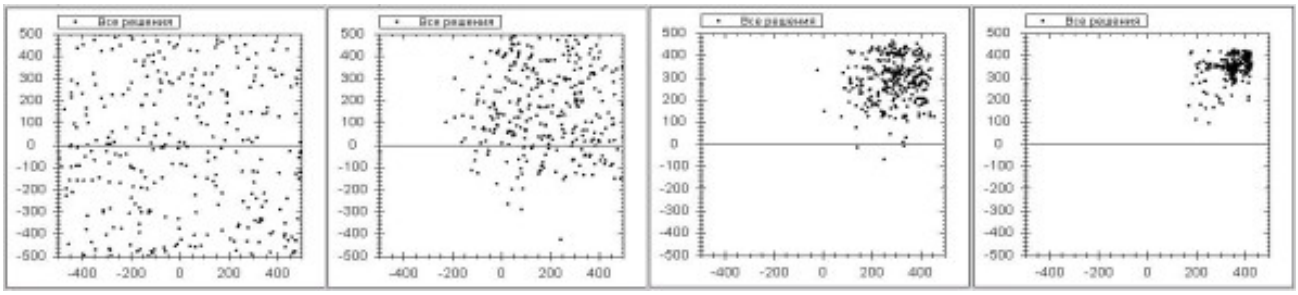


Рисунок 2.2 −Приклад роботи рою методом МРЧ

Концепцію даного алгоритму описує формула, згідно з якою коригується модуль і напрямок швидкості агентів:

$$v\omega v + rnd\ ()\ (Pbest\text{-}x)\ c1 + rnd\ ()\ (gbest\text{-}x)\ c2,$$

де ω - коефіцієнт інерції, що визначає баланс між тим, наскільки широко буде "заходити" в дослідженні агент і тим, наскільки сильно агент буде бажати залишитися поруч зі знайденими раніше оптимальними рішеннями;

Pbest - коесординати найкращою знайденої агентом точкою;

gbest - координати найкращої ройовиї точки;

x - поточні координати точки;

rnd () - випадковий коефіцієнт, який приймає значення від 0 до 1;

c1, c2 - постійні прискорення.

Спочатку цей алгоритм застосовувався для досліджень соціального психолога Кеннеді, але найбільше поширення цей алгоритм зміг отримати при рішеннях завдань оптимізації різних нелінійно-багатовимірних рівняннях. Цей алгоритм в сучасному світі застосовується в машинному навчанні, для рішень завдань оптимізації і в різних точних і експериментальних науках, таких як біоінженерія і т.і.

Мурашиний алгоритм. У цьому алгоритмі мурашина колонія розглядається як мультиагентна система, в якій всі агенти діють самостійно по дуже простим алгоритмам, але вся система в цілому веде себе вкрай розумно. Поведінка колонії мурах грунтується на самоорганізації, що досягається за рахунок взаємодії агентів на низькому рівні заради спільної мети. Особи можуть взаємодіяти як з допомогою прямого обміну інформацією (хімічний, візуальний контакт), так і за допомогою непрямого обміну (стігмержі). Він полягає в тому, що якийсь агент може змінювати область простору з допомогою деякого речовини (феромона), після чого інші агенти можуть використовувати цю інформацію для визначення власного маршруту. В результаті концентрація феромонів на маршруті визначає пріоритет його вибору. Крім того, «феромон» може випаровуватися.

Концепція алгоритму полягає в здатності мурах знаходити найкоротший шлях вкрай швидко і адаптуватися до різних зовнішніх умов. При русі кожен мураха позначає свій шлях феромонами, що в подальшому використовується іншими мурахами. Це і є простий алгоритм одного агента, який в сумі всіх агентів колонії дозволяє знаходити найкоротший шлях або змінювати його при виявленні перешкоди [25]. Даний приклад зображено на рисунку 2.3.

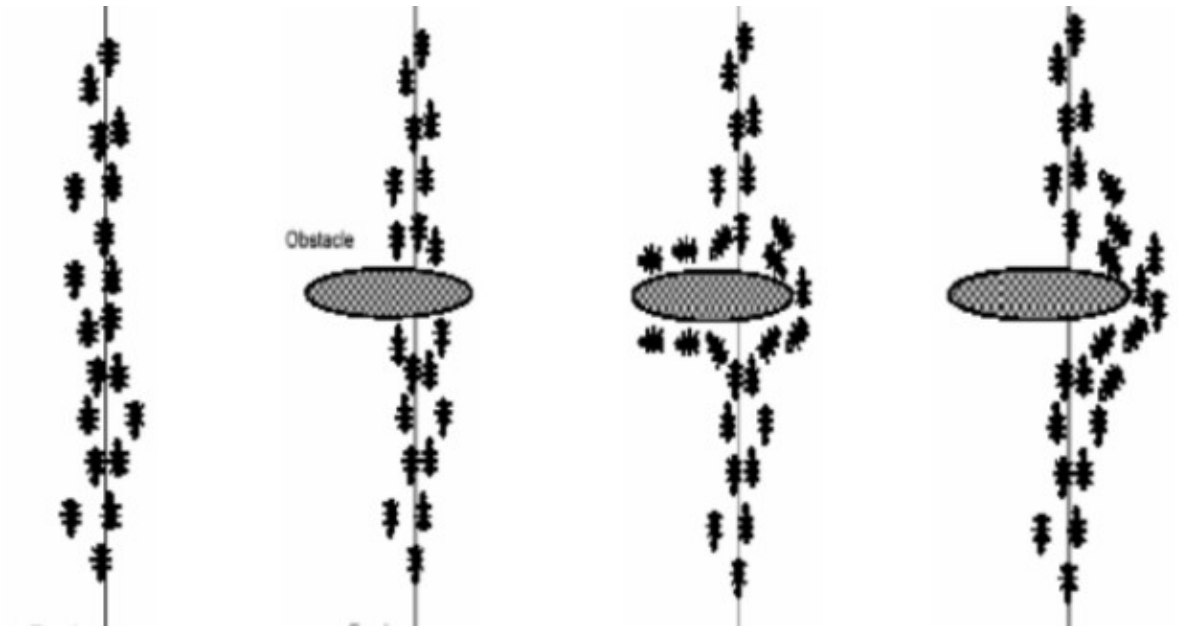


Рисунок 2.3 −Приклад знаходження мурахами нового шляху при появі перешкоди

Мурашиний алгоритм представимо у вигляді наступного набору команд [25]:

Поки (не виконані умови виходу):

1. Створення агентів;
2. Пошук відповідного рішення;
3. Зміна феромону;
4. Допоміжні дії (не обов'язково).

Далі розглянемо кожен крок докладніше.

1. Створення агентів:

- початкове розташування, де розміщується агент, залежить від початкових умов і обмежень задачі. Агенти можуть або бути в одній точці, або в різних з повтореннями, або в різних без повторень;

- також вказується початкове значення феромона, щоб значення не були нульовими.

2. Пошук відповідного рішення:

- ймовірність того, що відбудеться перехід з вершини і в j, можна визначити за формулою

$$P_{ij}(t) = \frac{\tau_{ij}(t)^{\alpha}\left(\frac{1}{d_{ij}}\right)^{\beta}}{\sum_{j \in allowed\ nodes} \tau_{ij}(t)^{\alpha}\left(\frac{1}{d_{ij}}\right)^{\beta}}$$

де *τij (t)* - рівень феромону,

dij- евристична відстань,

α, β - константні параметри.

якщо *α = 0*, з більшою ймовірністю вибереться найближче місто;

якщо *β = 0*, вибір буде грунтуватися лише на феромони.

Необхідний знайдений експериментальним способом компроміс між цими двома величинами.

1. Зміна феромону

- рівень феромону змінюється за формулою

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{\substack{k \in Colony\ that \\ used\ edge\ (i,j)}} \frac{Q}{L_k}$$

Де *ρ* - інтенсивність випаровування,

Lk (t) - ціна поточного рішення *k*-го мурашки,

Q - параметр, який має значення порядку ціни оптимального рішення,

$\frac{Q}{Lk(t)}$ - феромон, який відкладається k-м мурахою, який використовує ребро.

1. Допоміжні дії: в основному використовують алгоритми локального пошуку.

Незважаючи на достатнью ефективність, алгоритми рою частинок і мурашиний мають деякі недоліки. В силу великої кількості параметрів, що можуть настроюватися, розглянуті методи дуже чутливі до їх вибору і вимагають ретельної настройки. Вплив таких налаштувань неоднозначно впливає на ефективність роботи алгоритму і вимагає досліджень при вирішенні кожної конкретної задачі.

Процес вибору налаштувань має мало теоретичних обгрунтувань і в більшості випадків зводиться до «підгонки» значень. Також необхідно досить велику кількість апріорної інформації про розв'язувану задачу з метою вибору значень параметрів, що відповідають за локальну і глобальну збіжність. Сильною стороною даних методів є висока ефективність в разі успішно обраних параметрів, що настроюються, а також простота програмної реалізації.

У зв'язку з зазначеними недоліками алгоритмів рою частинок і мурашиного розглянемо метод рою бджіл, який володіє меншою чутливістю до параметрам, але більш складною логікою роботи.

Штучна бджолина колонія використовує алгоритм схожий з видобутком нектару медоносними бджолами. Замість поля з квітами розглянемо область рішень. Замість нектару використовуємо критерії задачі оптимізації, цільову функцію.

На кожній ітерації алгоритму вибираються області з кращим значенням цільової функції, вони називаються «кращі», з решти вибирається ще області, які є «перспективними». Можна задати певну мінімальну відстань між двома сусідніми областями. В цьому випадку, при виникненні накладення, область з гіршим значенням цільової функції відсікається. Замість неї вибирається інша область. Дані області запам'ятовуються і при наступній ітерації в них посилається певна кількість бджіл.

Схема описаного алгоритму представлена на рисунку 2.4.

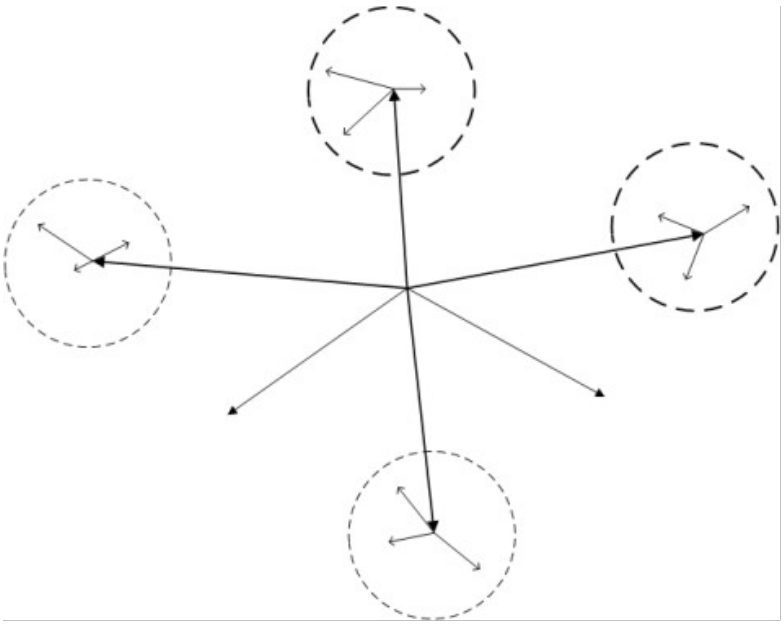


Рисунок 2.4 – Схематичне зображення стратегії алгоритму

Роботу алгоритму можна розбити на два етапи [25]:

1. Ініціалізація:

- при ініціалізації для n розвідників генеруються початкові положення;
- в найпростішому випадку використовується метод випадкового перебору.

2. Локальний пошук:

- після формування списків кращих і перспективних областей, в їх околиці відправляються «робочі бджоли»;
- в деяких варіантах алгоритму число відправляємих бджіл залежить від якості області з точки зору цільової функції. Ця залежність може бути лінійною або визначатися по більш складним правилам;
- в даному випадку, в кожну область висилається фіксована кількість бджіл, в залежності від класу, до якого належить дана область;
- за кожну ітерацію розвідники відправляються на нові області.

В даному алгоритмі використовується кілька параметрів:

- кількість розвідників,
- кількість кращих і перспективних,
- радіус локальної розвідки,
- кількість бджіл для кожного класу області,

- мінімально можлива відстань між сусідніми областями.

Якість одержуваних рішень значно залежить від вибору даних параметрів. Крім того, від цього вибору залежить і швидкість роботи алгоритму.

Існує безліч модифікацій даного алгоритму. Вони покращують якість результату і швидкість його роботи. В основному це відбувається завдяки зменшенню залежності від параметрівб що підбираються.

В результаті даного дослідження виявлено основні сильні сторони методів, області застосування, а також перспективи розвитку кожного з розглянутих методів.

На основі проведеного дослідження обрано бджолиний алгоритм, тому що при запуску групи БПЛА у зони з можливими дестабілізуючими факторами та загрозами, технічні пристрої можуть бути умисно або випадково пошкоджені. Бджолиний алгоритм дозволить розбити простій на участки; в можливо небезпечні зони відправити «бджіл дослідників», які дозволять зпочатку дослідити область, а якщо вона безпечна (якщо рішення найдено в околицях даної точки), туди буде відправлено інших агентів для повного і точного пошуку ефективних рішень.

2.3 Вимоги до розробки децентралізованої системи для управління групою БПЛА

Основною вимогою децентралізованих систем є відсутність єдиного центру. Це є водночас і основною перевагою таких систем, тому що центр управління може бути знищений.

При розробці системи необхідно передбачити надійне та безумовне виконання завдання та досягнення поставленої цілі. При виконанні завдання можуть виникнути різні перешкоди, як випадкові, так і навмисні. Тому

необхідно сформувати захисні умови для безпілотних літальних літальних апаратів. Однією такою умовою підвищення надійності виконання завдання є обєднання БПЛА у групи. Іншою умовою є використання спеціальних БПЛА-розвідників та БПЛА-наглядачів, які будуть постійно збирати дані про навколишнє середовище і передавати їх БПЛА-робітникам для успішного виконання завдання, збереження цілісності пристроїв та повернення до початкового пункту відправлення.

При такому сценарії безпілотні літальні апарати часто використовуються в якості мобільних датчиків, що використовуються для збору даних, які потім обробляються в хмарній службі.

3 РЕАЛІЗАЦІЯ РОЙОВОГО МЕТОДУ В ЗАДАЧАХ ДЕЦЕНТРАЛІЗОВАНОГО УПРАВЛІННЯ ГРУПОЮ БЕЗПІЛОТНИХ ЛІТАЛЬНИХ АПАРАТІВ

3.1 Структурна схема децентралізованого управління групою БПЛА

Структурна схема децентралізованого управління групою БПЛА на основі бджолиного ройового методу наведена на рисунку 3.1. Група БПЛА представлена у вигляді рою бджіл.

На схемі представлені 3 групи БПЛА-бджіл:

− БПЛА-розвідники (першими виконують дослідження простору, збирають дані про навколишнє середовище);

− БПЛА-спостерігачі (досліджують більш великі області в просторі пошуку, ніж розвідники);

− БПЛА-робітники (безпосередньо виконують поставлене завдання).

БПЛА-розвідники збирають дані про навколишнє середовище та передають їх БПЛА-спостерігачам та БПЛА-робітникам для безпечного виконання завдання.

БПЛА-спостерігачі також збирають дані про навколишнє середовище, але вони досліджують великі області в просторі пошуку (квадрати), у порівнянні з розвідниками, та передають дані БПЛА-робітникам.

БПЛА-робітники вважаються найбільш цінними та стратегічно важливими пристроями, що безпосередньо виконують поставлене завдання для досягнення мети. Вони використовують отримані дані про навколишнє середовище та отримують на вхід завдання, яке потрапояє децентралізовано, наприклад, зі хмари. Аналогічно відправляється результат виконання завдання.

Рисунок 3.1−Структурна схема децентралізованого управління групою БПЛА

3.2 Метод розробки адаптивного алгоритму управління групою БПЛА

Розглянемо основні етапи реалізації адаптивного бджолиного алгоритму.

1. Ініціалізація алгоритму. На етапі ініціалізації потрібно знайти або задати кілька початкових точок і обчислити значення цільової функції в них.

Групою БПЛА, або роєм бджіл є вектор випадкових рівномірно розподілених величин X, .

Задача полягає в знаходженні точки $x_{пот}$ в середині області працездатності D_x в n-вимірному просторі

Розглянемо три види пристроїв БПЛА:

s_j − БПЛА-розвідники;

w_k − БПЛА-робітники;

q_l −БПЛА-спостерігачі (досліджують більш великі області в просторі пошуку, ніж розвідники).

Рій БПЛА представляє множину

Всі частинки рою діють індивідуально, підкоряючись загальному принципу руху в бік найкращої персональної і найкращою спільної позиції. Суть алгоритму полягає в багаторазовому повторенні методу ненаправленного випадкового пошуку (процес відправки БПЛА) в обмежених областях простору пошуку, званих ділянками [26].

Для кожного типу БПЛА задається параметр r(0;1], який відповідає за розмір досліджуваної ділянки. Для БПЛА цей параметр приймає такі значення:

r_{Sj}=1 −БПЛА-розвідники досліджують увесь простір;

r_{Wk}=0,1 або 10% −від вихідного розміру області пошуку для БПЛА-спостерігачів;

r_{ql}=0,05, або 5% для БПЛА-робітників.

На етапі ініціалізації потрібно знайти або задати кілька початкових точок знаходження БПЛА в області , таких, що , наприклад: $X_1(x_1,y_1,\ldots,z_n)$, $X_2(x_2,y_2,\ldots,z_n)$, $X_3(x_3,y_3,\ldots,z_n)$ (рисунок 3.1) і обчислити значення цільової функції в них.

Рисунок 3.2−Точки знаходження БПЛА та їх околиці

2. Локальний пошук.

В залежності від цільової функції в області обираються ділянки:

b −найкращі, що відповідають найбільшим значенням цільової функції;

g −добрі, що відповідають найбільш близьким до найкращих значени цільовох функції.

Далі на ділянки b відправляються БПЛА-робітники

На ділянки g відправляються БПЛА-спостерігачі

Загальна рекомендація по вибору кількості ділянок і бджіл така. На кожен з кращих ділянок має припадати більше бджіл, ніж на добру ділянку, а розмір околиці повинен бути менше [27].

Необхідно відзначити, що БПЛА-робітники надсилаються не в точності на те місце, де БПЛА-розвідники та спостерігачі знайшли перспективні і кращі місця, а в їх околиці. Крім того, околицю, яка визначає область, в яку може бути послано БПЛА, можна зменшувати в міру збільшення номера ітерації, щоб поступово рішення сходилося до самого «кінця» екстремуму. Але якщо область

зменшувати занадто швидко, то рішення може «застрягти» в локальному екстремуму [27].

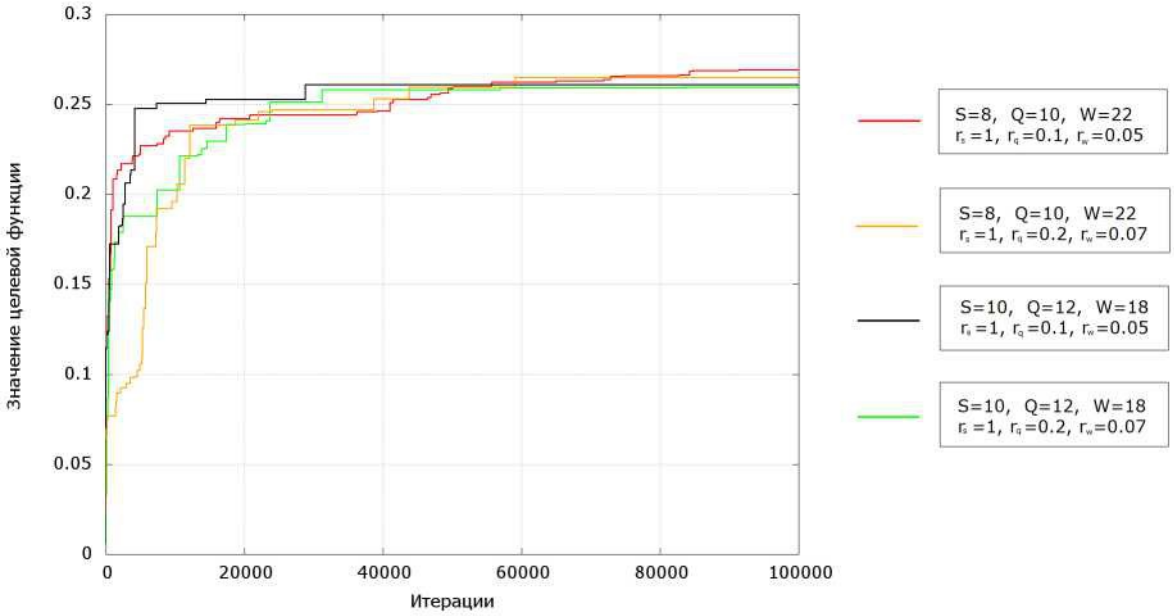
Розмір досліджуваних ділянок може бути заданий як статично, так і динамічно змінюватися в залежності від кількості ітерацій. У разі реалізації динамічно змінюємої області пошуку збільшується швидкість збіжності алгоритму, але з'являється можливість зависання алгоритму в локальному екстремуму. В цьому випадку глобальний оптимум може бути не знайдений. Виходячи з цього при реалізації даного алгоритму для цих параметрів були обрані фіксовані значення [28].

Після того, як група БПЛА досліджує знайдені області, знову відбувається упорядкування ділянок по спадаючому значенню цільової функції серед усіх БПЛА, що беруть участь в процесі пошуку, потім здійснюється подальший вибір кращих і хороших ділянок і відбувається нова відправка БПЛА. Ці кроки повторюються до тих пір, поки не буде задоволений критерій зупинки – досягнення поставленої мети групи БПЛА.

Критеріїв зупинки може бути кілька. Наприклад, якщо ми знаємо значення цільової функції в глобальному екстремумі, то можемо повторювати алгоритм до тих пір, поки функція не досягне деякого значення, близького до бажаного. Якщо значення функції в екстремумі невідомо, то можемо повторювати кроки алгоритму до тих пір, поки протягом досить великої кількості ітерацій знайдене рішення не буде поліпшуватися.

На рисунку 3.3 наведені графіки збіжності методу рою бджіл, в залежності від різних наборів настроювальних параметрів. З наведених графіків видно, що вибір параметрів алгоритму не є критичним фактором при вирішенні задачі оптимізації.

Рисунок 3.3–Робота алгоритму рою бджіл при різних наборах параметрів



Таким чином, розглянуто задачу забезпечення надійності від відмов технічних пристроїв групи БПЛА, яку потрібно вирішувати на етапі проектування системи. При цьому враховується неповнота апріорної інформації про випадкові зовнішні процеси, які можуть виникнути (поява перешкод для БПЛА). Розглянутий в роботі метод рою бджіл, незважаючи на доволі складну логіку роботи і процес організації обчислення, в порівнянні з іншими методами, показує свою ефективність.

Основні труднощі у використанні інших ройових методів даної групи − вибір параметрів, що настроюються. Не існує універсального набору параметрів, а перебирати різні комбінації налаштувань з метою вибору найкращої для кожної нової задачі не завжди представляється можливим [29].

В роботі запропонована модифікація методу рою бджіл, що спрощує логіку його роботи і програмну реалізацію. Дана модифікація показала свою ефективність і універсальність. Розглянутий метод в меншій мірі залежить від якості настройки параметрів [30].

Таким чином, якщо розглядати всі критерії в комплексі, з точки зору найкращого рішення задачі, обмежених часових і обчислювальних ресурсів, відсутності можливості ретельного налаштування параметрів алгоритмів, відсутності відомостей про розв'язувану задачу, то використання методу рою

бджіл є кращим при вирішенні задачі оптимального параметричного синтезу за критерієм надійності.

3.3 Адаптивні алгоритми групи безпілотних літальних апаратів при вирішенні завдання

Алгоритми зльоту, польоту, посадки виконуються автоматично завдяки вбудованим (запрограмованим) алгоритмам в ці пристрої, що спрощує рішення задачі і дозволяє сконцентруватися на більш важливих аспектах роботи.

Розглянемо алгоритм патрулювання БПЛА-«розвідників», БПЛА-«наглядачів» та БПЛА-«робітників», які безпосередньо виконують поставлене завдання.

На рисунку 3.4 наведено алгоритм патрулювання БПЛА-«розвідників».

На рисунку 3.5 наведено алгоритм патрулювання БПЛА-«спостерігачів».

На рисунку 3.6 наведено алгоритм БПЛА-«робітників», які безпосередньо виконують поставлене завдання.

Рисунок 3.4 − Алгоритм патрулювання БПЛА-«розвідників»

Рисунок 3.5− Алгоритм патрулювання БПЛА-«спостерігачів»

Рисунок 3.6 − Алгоритм БПЛА-«робітників» для виконання поставленого завдання

Адаптивність наведених алгоритмів полягає у наявності зворотніх зв’язків і передачі інформації про виникнення перешкоди іншим групам БПЛА:

− в алгоритмі патрулювання БПЛА-«розвідників» передача інформації всім БПЛА-розвідникам та БПЛА-спостерігачам;

− в алгоритмі патрулювання БПЛА-«спостерігачів» передача інформації БПЛА-«розвідникам» і БПЛА-«робітникам».

4. ПРАКТИЧНА РЕАЛІЗАЦІЯ

4.1 Практичная реалізація бджолиного алгоритму управління групою БПЛА

Практична реалізація бджолиного алгоритму управління групою БПЛА розглянемо на прикладі. Нехай в якості цільової функції виступає функція). Це модель квадратичної поверхні, яка гарно описує простір, в якому можуть пристрої БПЛА виконувати поставлене завдання.

Знак «−» в даному випадку означає глобальний максимум функції, знак «+» локальний мінімум. Глобальний (і єдиний) максимум цієї функції знаходиться в точці (0; 0; 0), причому f (0, 0, 0) = 0.

Визначимо необхідний набір параметрів:

- кількість БПЛА-розвідників: 10;
- кількість БПЛА, що відправляються на кращі ділянки: 5;
- кількість БПЛА, що відправляються на інші вибрані ділянки: 2;
- кількість кращих ділянок: 2;
- кількість обраних ділянок: 3;
- розмір області для кожної ділянки: 10.

Нехай БПЛА-розвідники потрапили на наступні ділянки (список впорядкований по спадаючій цільової функції):

1) *f (15, 18, 2) = -553;*
2) *f (-30, -15, 3) = -1134;*
3) *f (22, -31, -5) = -1470;*
4) *f (18, 40, 7) = -1973;*
5) *f (-25, 47, 9) = -2915;*
6) *f (60, 86, -11) = -11117;*
7) *f (-91, -99, 13) = -18351;*
8) *f (17, -136, -15) = -19010;*
9) *f (-152, -1, 1) = -23106;*

10) *f (-222, 157, 1) = -73934.*

Спочатку будуть обрані 2 кращі точки:

1) *f (15, 18, 2) = -553;*

2) *f (-30, -15, 3) = -1134.*

Потім будуть обрані інші 3 перспективні ділянки:

1) *f (22, -31, -5) = -1470;*

2) *f (18, 40, 7) = -1973;*

3) *f (-25, 47, 9) = -2915.*

В околиці кращих точок будуть відправлені по 5 БПЛА:

Для першої кращої точки значення координат, якими обмежується ділянка:

[15 - 10 = 5; 15 + 10 = 25] для першої координати;

[18 - 10 = 8; 18 + 10 = 28] для другої координати;

[2 - 10 = -8; 2 + 10 = 12] для третьої координати;

Тобто для першої кращої точки інтервал, якими обмежується ділянка за трьома координатами:

[5,25]; [8,28]; [-8,12].

Аналогічно розраховуються інтервали для обраних ділянок за трьома координатами:

[12, 32]; [-41, -21]; [-15, 5];

[8, 28]; [30, 50]; [-3, 17];

[- 35, 15]; [37, 57]; [-1, 19].

По кожній з координат розмір області однаковий і дорівнює 20, але це не обов'язково може бути так.

У кожен з кращих інтервалів відправляємо по 5 БПЛА, а на вибрані ділянки по 2 БПЛА. Причому, не будемо змінювати положення БПЛА, які знайшли кращі і вибрані ділянки, інакше є ймовірність того, що на наступній ітерації максимальне значення цільової функції буде гірше, ніж на попередньому кроці.

Нехай тепер на першій кращий ділянці маємо наступні БПЛА:

1) *f (15, 18, 2) = -553;*
2) *f (7, 12, 4) = -209;*
3) *f (10, 10, -1) = -201;*
4) *f (7, 5, -3) = -83;*
5) *f (-2, -5, -3) = -381.*

Як видно, вже серед цих нових точок є такі, які кращі, ніж попереднє рішення.

Такі ж дії виконуємо і з другою кращою ділянкою, а потім аналогічно і з вибраними ділянками. Після чого серед усіх нових точок знову відзначаються кращі і вибрані, а процес повторюється знов.

4.2 Реалізація алгоритму групового польоту на мові Python

Розглянемо приклад реалізації алгоритму рою бджіл на мові Python.

Мова програмування Python має ряд переваг [31]:

− популярна високорівнева мова програмування з динамічною семантикою;

− досить проста для роботи, саме тому одна з найпоширеніших: використання Python знижує вартість розробки та обслуговування програм;

− Python досить легко взаємодіє з іншими мовами, особливо з мовами групи C;

− Python активно використовується для вирішення завдань інтелектуального аналізу даних і машинного навчання, тобто для розробки додатків, які можуть навчатися і видавати результати автоматично, без людського втручання;

– Python має відмінну продуктивність при обробці даних.

Розроблена програмна система містить наступні файли:

− pybee.py −основні класи, що реалізують алгоритм рою бджіл;

– beetest.py – основний модуль прикладу, в якому містяться всі параметри алгоритму, це скрипт для запуску;

– beeexamples.py – ці модулі містять різні класи бджіл для різних цільових функцій;

– beetestfunc.py – допоміжні функції для візуалізації процесу розрахунку.

Розглянемо основні класи.

У модулі pybee містяться наступні класи:

– floatbee – базовий клас для бджіл, положення яких описується числами з плаваючою точкою;

– hive – клас вулика, всередині якого і відбуваються основні дії алгоритму по виділенню кращих і обраних ділянок, а також відправка бджіл на потрібні позиції;

– statistic – допоміжний клас для збору статистики по збіжності алгоритму рою бджіл. Цей клас накопичує кращі результати для кожного кроку ітерації.

Клас floatbee – це базовий клас, від якого необхідно успадковуватися і перевизначити один метод, крім того в конструкторі потрібно визначити члени:

```
self.position
self.minval
self.maxval
```

Наприклад, положення у визначається трьома координатами, причому перша координата може змінюватися в межах [-1; 1], а друга в межах [-11; 12], третя [-1.0,1.0]. Тоді заповнення цих членів може бути таким:

```
self.minval=[-1.0,-11.0, -1.0]
self.maxval=[1.0,12.0, 1.0]
self.position=[random.uniform   (self.minval   [n],self.maxval
[n])for n inrange(2)]
```

Нехай цільова функція у нас буде *f (x, y) = -x ^ 2 - y ^ 2- z ^ 2*, тоді calcfitness () може виглядати наступним чином:

```
Class mybee (pybee.floatbee): ...
def calcfitness (self):"" "Розрахунок цільової функції. Цей метод необхідно перевантажити в похідному класі. Функція не повертає значення цільової функції, а тільки встановлює член self.fitness Цю функцію необхідно викликати після кожної зміни координат бджоли" ""
self.fitness=0.0
for val in self.position:self.fitness -=val * val
```

Тоді з урахуванням вищесказаного конструктор для бджоли буде виглядати наступним чином:

```
classmybee (pybee.floatbee): ...
def__init__(self):self.minval=[-1.0,-11.0,
-1.0]self.maxval=[1.0,12.0, 1.0]
self.position=[random.uniform (self.minval [n],self.maxval
[n])for n inrange(2)]self.calcfitness ()
```

Основні кроки алгоритму рою бджіл реалізовані всередині класу hive (вулик), поговоримо про нього докладніше.

Розглянемо параметри конструктора:

```
classhive: ...
def__init__(self,scoutbeecount,selectedbeecount,bestbeecount,
\ selsitescount,bestsitescount, \ range_list,beetype):
```

У конструктор передається 7 параметрів:

– scoutbeecount- кількість бджіл-розвідників;

– selectedbeecount- кількість бджіл, що посилаються на кожну з обраних ділянок;

– bestbeecount - кількість бджіл, що посилаються на кожну з кращих ділянок;

– selsitescount - кількість обраних (перспективних, але не кращих) ділянок;

– bestsitescount - кількість кращих ділянок;

– range_list - список, елементи якого визначають розміри обраних і кращих ділянок по кожній координаті;

– beetype - клас бджіл, похідний від floatbee, який використовується в алгоритмі.

Для відправки бджоли в околиці точки в класі floatbee використовується метод

```
defgoto (self,otherpos,range_list)
```

в якому otherpos- точка, в околицю якої надсилається бджола, range_list - той же самий список розмірів околиць.

Розглянемо код конструктору класу hive.

```
classhive:                                                    ...
def__init__(self,scoutbeecount,selectedbeecount,bestbeecount,\
selsitescount,bestsitescount,\            range_list,beetype):
self.scoutbeecount=scoutbeecountself.selectedbeecount=selected
beecountself.bestbeecount=bestbeecount
self.selsitescount=selsitescountself.bestsitescount=bestsitesc
ount
self.beetype=beetype
self.range=range_list
#       Краща       на       даний       момент       позиція
self.bestposition=None
# Краще на даний момент здоров'я бджоли (чим більше, тим
краще)
self.bestfitness=-1.0e9
#   Початкове   заповнення   рою   бджолами   з   випадковими
координатами
 beecount =scoutbeecount + selectedbeecount * selsitescount +
bestbeecount       *       bestsitescountself.swarm=[Beetype
()fornin xrange(Beecount)]
#           Кращі           і           вибрані           місця
self.bestsites=[]self.selsites=[]
self.swarm.sort
(floatbee.sort,reverse=True)self.bestposition=self.swarm
[0] .Getposition ()self.bestfitness=self.swarm [0] .fitness
```

По-перше, в ньому зберігаються всі передані параметри. По-друге, створюється член self.bestposition, який зберігає в собі краще на даний момент

рішення. Значення цільової функції для кращого на даний момент рішення зберігається в члені self.bestfitness.

Потім створюється рій бджіл. Загальна кількість бджіл в ньому дорівнює сумі кількості розвідників, кількості бджіл посилаються на один обрану ділянку, помножене на кількість обраних ділянок, і кількості бджіл, що посилаються на один з кращих ділянок, помножене на кількість кращих ділянок [32].

```
beecount = scoutbeecount + selectedbeecount * selsitescount +
bestbeecount * bestsitescount
```

Потім заповнюється список члена self.swarm (рій):

```
self.swarm=[Beetype ()forninxrange(Beecount)]
```

В конструкторі всі параметри бджоли повинні ініціюватися випадковим чином, тому виходить, що при створенні рою бджіл, всі вони виступають в ролі бджіл-розвідників, яких відправляють на випадкові точки. З точки зору алгоритму це не обов'язково має бути так. Можна в перший раз відправити тільки задану кількість бджіл-розвідників, а бджіл, призначених для кращих і обраних ділянок, відправляти на наступних ітераціях. Але в даному випадку на самому початку на випадкові місця відправляються всі бджоли, які беруть участь в алгоритмі [33, 34].

Після того як рій створений, він сортується по спадаючій цільової функції у кожної бджоли. В якості опції порівняння передається метод sort () з класу floatbee.

Після сортування кращим буде рішення у бджоли, що знаходиться в самому початку списку. Значення цільової функції для даної бджоли зберігається в член self.bestfitness, а в член self.bestposition зберігається копія координат кращої бджоли. Для отримання копії координат використовується метод getposition () з класу floatbee:

```
classfloatbee:                                                    ...
defgetposition (self):"" "Повернути копію (!) Своїх координат"
""
returnvalforvalinself.position]
```

Для виконання однієї ітерації алгоритму рою бджіл необхідно з екземпляра класу hive викликати метод

```
Def nextstep (self)
```

Цей метод складається з двох логічних етапів. Спочатку вибираються кращі і вибрані ділянки. Вибір кращих ділянок відбувається наступним чином:

```
classhive:                                                        ...
def next step (self):# Вибираємо найкращі місця і зберігаємо
посилання на тих, хто їх знайшов
self.bestsites=[self.swarm                                        [0]]
                        curr_index                                 =1
for currbee in self.swarm [curr_index: -1]:# Якщо бджола
перебуває в межах вже зазначеного кращого ділянки, то її
положення не вважаємо
if                                          currbee.otherpatch
(self.bestsites,self.range):self.bestsites.append    (currbee)
if              len(self.bestsites)==self.bestsitescount:break
 curr_index +=1
```

Бджоли, що знайшли кращі ділянки, поміщаються в список self.bestsites. По-перше, туди поміщається бджола, яка перебуває на початку списку, а потім відбувається перебір інших бджіл до тих пір, поки не буде відібрано потрібну кількість.

Причому пропускаються бджоли, які знаходяться в одній ділянці з бджолою, яка вже знаходиться в списку self.bestsites. curr_index зберігає номер бджоли в списку, яку будемо перевіряти наступною.

Перевірка того, чи ділянка, знайдена бджолою нова або якась бджола вже знаходиться поблизу, відбувається всередині методу otherpatch () класу floatbee.

```
Class floatbee: ...
def other patch (self,bee_list,range_list):"" "Перевірити чи знаходиться бджола на тій же ділянці, що і одна з бджіл в bee_list. Range_list - інтервал зміни кожної з координат" ""
if len(Bee_list)==0:return True
for curr_bee in bee_list: position=curr_bee.getposition ()
for n in xrange(len(self.position)):if abs(self.position [n] - position [n])>range_list [n]:return True
return False
```

В даному випадку в якості bee_list передається список кращих бджіл self.bestsites, а в якості range_list все той же список розмірів околиць. Ділянка, знайдена бджолою, вважається новою, якщо в списку bee_list немає бджоли, відстань до якої по кожній з координат не перевищує відповідного розміру з range_list.

Після знаходження таким чином кращих ділянок аналогічно відзначаємо вибрані ділянки.

```
classhive: ...
def nextstep (self): ...
self.selsites=[]
for currbeeinself.swarm [curr_index: -1]:ifcurrbee.otherpatch (self.bestsites,self.range)and currbee.otherpatch (self.selsites,self.range):self.selsites.append (currbee)
if len(self.selsites)==self.selsites count:break
```

Бджоли з обраних ділянок зберігаються в список self.selsites. Після того як відзначили кращі і вибрані ділянки, відправимо бджіл на відповідні позиції.

```
Class hive: ...
Def nextstep (self): ...
# Номер чергової відправляємої бджоли. 0-ую бджолу нікуди не відправляємо
 bee_index =1
for best_bee in self.bestsites: bee_index=self.sendbees (best_bee.getposition (),bee_index,self.bestbeecount)

or sel_bee in self.selsites: bee_index=self.sendbees (sel_bee.getposition (),bee_index,self.selectedbeecount)
for curr_bee in self.swarm [bee_index: -1]: Curr_bee.gotorandom()
self.swarm.sort (floatbee.sort,reverse=True)self.bestposition=self.swarm [0] .Getposition ()self.bestfitness=self.swarm [0] .fitness
```

За відправку бджіл відповідає метод sendbees () класу hive. Він полягає в тому, що відправляється задана кількість бджіл, пропускаючи тих, які вже знаходяться серед кращих і обраних бджіл. Всередині sendbees () викликається метод goto () класу floatbee.

```
classfloatbee: ...
defgoto (self,otherpos,range_list):"" "Переліт в околицю місця, яке знайшла інша бджола. Не в те саме місце!" ""
# До кожної з координат додаємо випадкове значення
self.position=[Otherpos [n] +random.uniform (-range_list [n],range_list [n]) \for n in xrange(len(Otherpos))]
# Перевіримо, щоб не вийти за задані межі
self.checkposition ()
# Расчитаем і збережемо цільову функцію
self.calcfitness ()
```

Спочатку заповнюється список нових координат, які складаються з точки, в яку відправляють бджолу і випадкової величини в інтервалі від -range_list [n] до range_list [n]. Потім викликається метод checkposition (), який коригує координати, якщо вони виходять за задані межі (списки minval і maxval).

```
Class floatbee: ...
def checkposition (self):"" "Скорегувати координати бджоли,
якщо вони виходять за встановлені межі" ""
for n in range(len(self.position)):
If     self.position     [n]<self.minval     [n]:self.position
[n]=self.minval [n]
else   if   self.position   [n]>self.maxval   [n]:self.position
[n]=self.maxval [n]
```

Після того як були відправлені бджоли на кращі і вибрані місця, бджіл, що залишилися, з вулика відправляємо на випадкові координати. Потім все бджоли знову упорядковуються відповідно до зменшенням цільової функції і зберігаються кращі позиції і краще значення цільової функції [34].

Метод nextstep () викликається користувачем до тих пір поки не буде виконана одна з умов зупинки.

Клас Statistic у модулі pybee не використовується в алгоритмі безпосередньо, він призначений допомогти в налагодженні і візуалізації результатів.

Цей клас зберігає знайдене рішення для кожного кроку ітерації. Причому, один екземпляр класу statistic можна використовувати для декількох незалежних запусків алгоритму. Це допомагає подивитися, наприклад, як сходяться параметри при усередненні за кількома запусками.

При запуску програмної системи спочатку створюється клас для збору статистики:

```
stat=pybee.statistic ()
```

Далі задаються параметри алгоритму рою бджіл. В першу чергу задається клас використовуваних бджіл.

```
beetype=beeexamples.spherebee
#beetype = beeexamples.dejongbee
#beetype = beeexamples.goldsteinbee
#beetype = beeexamples.rosenbrockbee
#beetype = beeexamples.testbee
#beetype = beeexamples.funcbee
```

Потім йде завдання інших параметрів алгоритму:

```
# Кількість бджіл-розвідників
scoutbeecount=300
# Кількість бджіл, що відправляються на вибрані, але не кращі ділянки
selected beecount=10
# Кількість бджіл, що відправляються на кращі ділянки
bestbeecount=30
# Кількість обраних, але не кращих, ділянок
selsitescount=15
# Кількість кращих ділянок
bestsitescount=5
# Кількість запусків алгоритму
runcount=1
# Максимальна кількість ітерацій
maxiteration=1000
# Через таку кількість ітерацій без знаходження кращого рішення зменшимо область пошуку
max_func_counter=10
```

На рисунку 4.1 (а, б, в, г, д, ж) можна побачити як БПЛА-бджоли поступово накопичуються навколо одного кращого рішення.

На цьому рисунку червоні точки – БПЛА-бджоли, що знайшли кращі рішення, жовті точки –вибрані рішення, а чорні точки – інші БПЛА- бджоли, включаючи БПЛА-розвідників, які розташовуються випадковим чином.

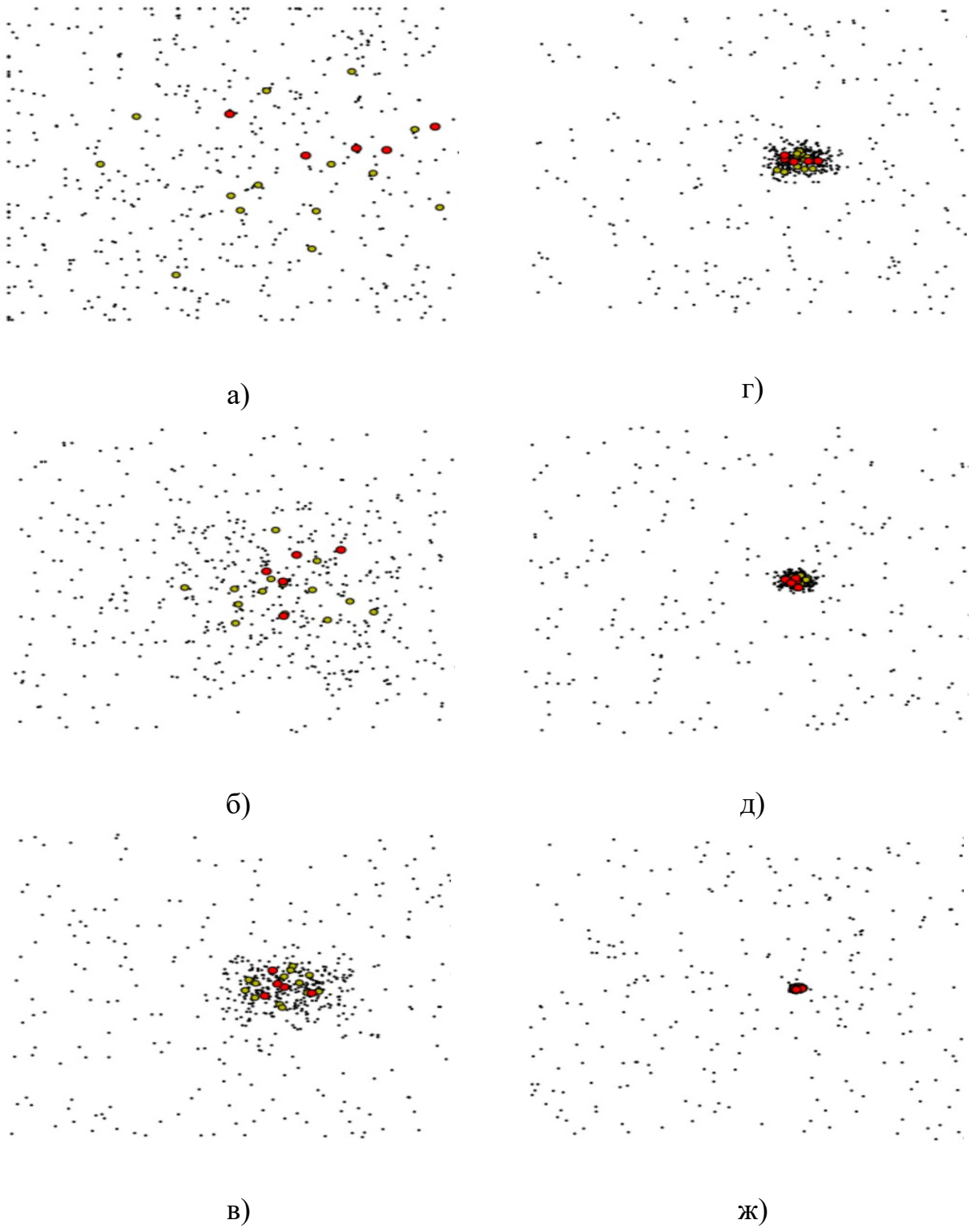


Рисунок 4.1 − Поступове накопичення БПЛА-бджіл навколо одного кращого рішення

Зростання цільової функції при одному запуску алгоритму зображене ни рисунку 4.2.

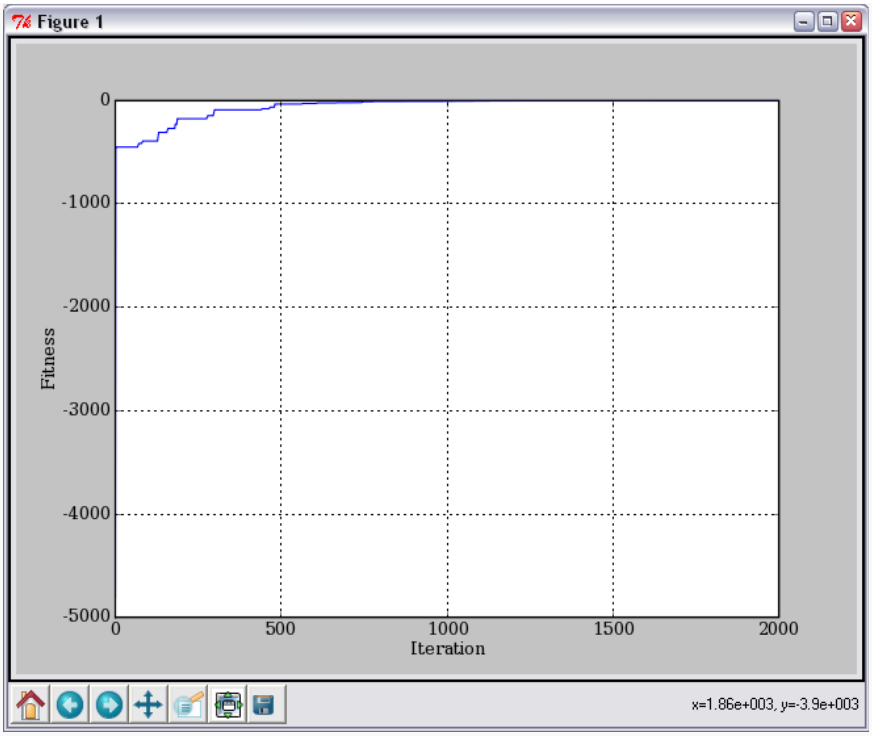


Рисунок 4.2 – Зростання цільової функції при одному запуску алгоритму

Зростання цільової функції, усереднене по 10 запускам алгоритму зображено на рисунку 4.3.

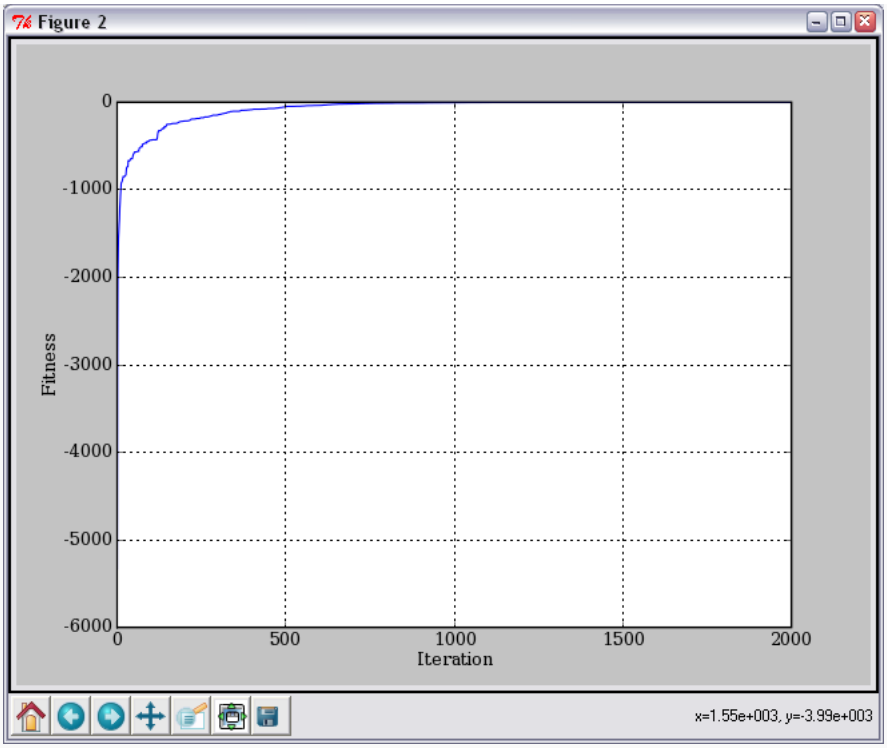


Рисунок 4.3 – Зростання цільової функції, усереднене по 10 запускам алгоритму

На рисунку 4.4 наведене зменшення розмірів областей для кожної з координат.

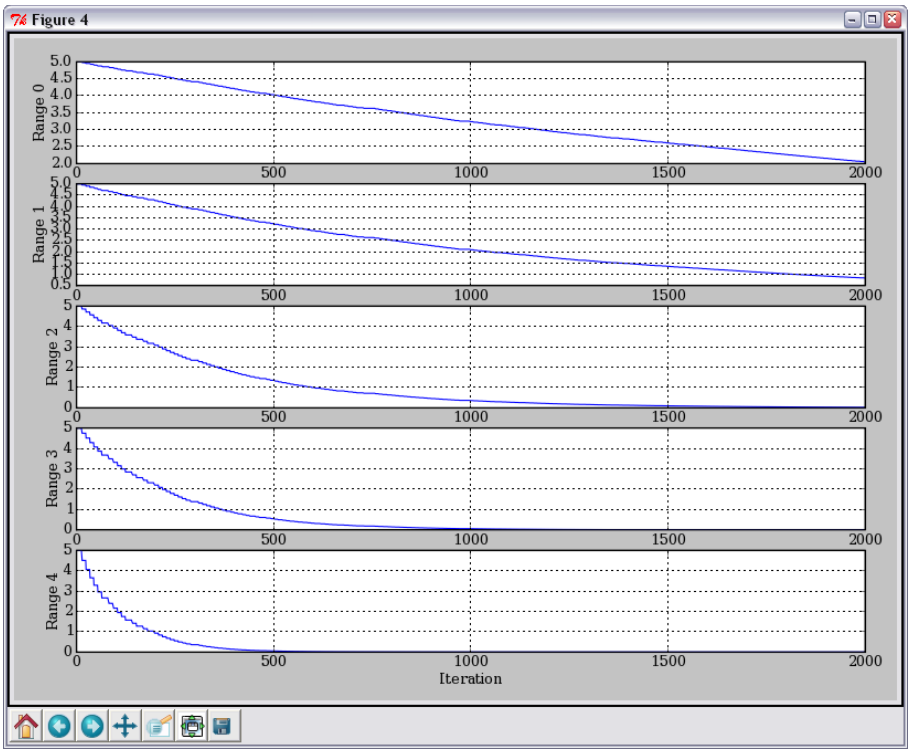


Рисунок 4.4 – Зменшення розмірів областей для кожної з координат

Таким чином, в даному пункті розглянуто алгоритм БПЛА- бджіл для знаходжень глобального екстремуму функції і наведено можливу реалізацію на мові Python.

4.3 Програмно-апаратна реалізація

На основі розглянутих в пункті 3.3 алгоритмів за допомогою програмного забезпечення AeroSIM Radio Control Training Simulator представлена програмна реалізація, яка моделює поведінку дронів у групі.

Це один з перших симуляторів, призначений саме для моделювання польотів коптерів. Даний симулятор містить цікаві вправи, які дозволяють звикнути до пульта управління, встановлювати свою карту місцевості, вибирати не тільки коптер, підключати майже будь-який пульт управління.

ENTERPRISE VERSION, CUSTOMIZED VERSION дозволяють підтримку коптерів:

- Mavic 2 Enterprise
- Mavic 2 Zoom
- Mavic Air
- Mavic Pro
- SparkPhantom 4 Pro
- Inspire 1 Pro
- Inspire 2Matrice 210 RTK
- Xiaomi FIMI X8

У даній роботі розглядається політ групи БПЛА.

В якості БПЛА-розвідників і БПЛА-спостерігачів пропонується використовувати бюджетний варіант дрона Xiaomi FIMi X8 SE 2020, тому що є вірогідність його втрати через специфіку виконуємих завдань. Не дивлячись на

вартість, даний БПЛА має безпосередні переваги для виконання завдань розвідки: запис відео з роздільною здатністю 4K Ultra HD, безліч розумних режимів зйомки, швидкість горизонтального польоту до 64 км / ч, зручний пульт з вражаючим радіусом дистанційного керування до 8 кілометрів, невелика вага, велика автономність роботи.

В якості БПЛА-працівника використовується дрон DJI Mavic 2 Pro. Його перевагами для вирішення завдань в групі є:

– дальності польоту на відкритому просторі і при відсутності перешкод до 7 км;

– оснащення більш ефективної силовою установкою і літієвими акумуляторами високої щільності, що дозволяє Mavic Pro триматися в повітрі до 27 хвилин при оптимальних умовах;

– висока швидкість польоту. Ефективна конструкція корпусу дозволяє знизити опір повітря, аеродинамічна конфігурація згідно вектору і куту польоту вперед, дозволяє досягати швидкості до 65 км / год (в спортивному режимі);

– HD-відео в 4К 12 МП камера Mavic Pro гарантує ідеальні кадри. Мініатюрні високоточні осі підвісу камери забезпечують прекрасну стабілізацію навіть при високій швидкості польоту;

– інтелектуальні режими зйомки. Функція ActiveTrack дозволяє автоматично тримати в кадрі вказаний об'єкт: людина, тварина або навіть автомобіль. Trace − режим польоту за зазначеним об'єктом, навколо нього або попереду. Profile −політ паралельно об'єкту зйомки. Spotlight −автоматичне утримання зазначеного об'єкта у фокусі;

– безпеку польотів. Технологія раннього розпізнавання перешкод SkyAutonomy дозволяє Mavic Pro облетіти їх або зупинитися, уникаючи зіткнення навіть поза зоною видимості пілота. GPS-позиціонування дозволяє дрону зависати на відкритому майданчику, вільному від перешкод. У разі втрати зв'язку із супутником датчики візуального позиціонування дозволяють Mavic Pro зависати навіть в приміщеннях.

Приклад роботи групи БПЛА запропонованим ройовим методом, що містить БПЛА-розвідників, БПЛА-спостерігачів, БПЛА-розвідників представлений на рисунку 4.5.

Рисунок 4.5 – Приклад роботи групи БПЛА запропонованим ройовим методом

ВИСНОВКИ

Атестаційна робота виконано згідно з методичними вказівками і містить основні розділи [35]:

- огляд і аналіз існуючих методів, моделей і алгоритмів управління безпілотними літальними апаратами;
- дослідження методів розробки децентралізованих систем;
- реалізація ройового методу в задачах децентралізованого управління;
- практична реалізація.

На сьогоднішній день застосування малогабаритних дронів для ефективного вирішення завдань в умовах неконтрольованих ситуацій є перспективним напрямком. Головними причинами впровадження дронів в сферу безпеки життєдіяльності є: висока оперативність, що особливо важливо в надзвичайних ситуаціях; економічна ефективність завдяки відносній дешевизні дрона; надійність, оскільки відсутній людський фактор; відсутність або суттєве зниження загрози для життя та здоров’я персоналу тощо.

Метою атестаційної роботи є дослідження ройових методів, методів децентралізованного управління групою безпілотних літальних апаратів, адаптивних алгоритмів для ефективного вирішення завдань в умовах неконтрольованих ситуацій.

Для досягнення поставленої мети в роботі:

- наведено обгрунтування децентралізованого управління групою БПЛА; проведено дослідження ройових методів та алгоритмів, обрано бджолиний метод для реалізації системи управління групою БПЛА;
- в рамках наукових результатів наведено структурну схему децентралізованого управління групою БПЛА; представлено метод розробки адаптивного алгоритму управління групою БПЛА; розроблені адаптивні алгоритми управління групою БПЛА (алгоритм патрулювання

БПЛА-«розвідників», БПЛА-«спостерігачів» та алгоритм БПЛА-«робітників», які безпосередньо виконують поставлене завдання);

– у рамках практичних результатів представлена програмно-апаратна реалізація.

Надалі планується покращення алгоритму для реалізації більш гнучкого управління групою об'єктів, руху по складних траєкторіях, можливістю зміни траєкторії під час роботи та обробці більшої кількості позаштатних ситуацій [36].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Томас Х. Кормен. Алгоритмы: вводный курс. −М.: «Вильямс», 2014. − 208 с..

2. Игошин В. И. Математическая логика и теория алгоритмов. 2-е изд., М.: ИЦ «Академия», 2008.−448 с.

3. Б.В. Шамша, А.М. Гуржій, З.В. Дудар, В.М. Левікін. Математичне забезпечення інформаційно-управляючих систем. − Харків .: Компанія Сміт, 2005.-445 с.

4. Б.В. Шамша, А.Н. Гуржій, В.М. Левикін, Т.Б. Шатовська Математичне забезпечення інформаційно-управляючих систем. Прогнозування-Харків .: Компанія Сміт, 2013.−371 с.

5. Dervis Karaboga. An idea based on honey bee swarm for numeral optimization, 2005.

6. Seyedali Mirjalili, Seyed Mohammad Mirjalili, Andrew Lewis. Grey Wolf Optimizer, 2014.

7. Sabu m Thampi SWARM INTELLIGENCE // arxiv.org - Алгоритм рою часток (дата звернення 18.10.2020)

8. Григор'єв І. В., Мустафіна С. А. реалізація методу роя часток на nvidia cuda // Науч.Форум, 2014. Алгоритм рою частинок. Опис і реалізації на мовах Python і C # // jenyay.net. 2011.

9. Анопов М. Ф., Катуева Е. В., Міхалічук В. І. Алгоритми рою бджіл і частинок в завданню забезпечення надійності по поступовим відмов // Наука та Освіта. - 2015. - № 1.

10. Лебедєв Б. К., Лебедєв В. Б. Розміщення на основі методу бджолиної колонії // Известия ПФУ. Технічні науки.

11. Чураков М., Якушев А. Мурашині алгоритми // 2006.

12. Матренин П.В. Методи стохастичною оптимізації: навчальний посібник / П.В. Матренин, М.Г. Гриф, В.Г. Сека // Новосибірськ: Изд-во НГТУ, 2016. - 67 с.

13. Класифікація безпілотних літальних апаратів / О. І.Тимочко, Д. Ю. Голубничий, В. Ф. Третяк, І. В. Рубан. // Системи озброєння і військова техніка. –2007. – №1. – С. 61–66.

14. Заблотский А. БПЛА: первое знакомство / А. Заблотский, Р. Ларинцев. // Авиация и время. – 2008. – №2. – С. 15.

15. Каляєв І.А. Моделі і алгоритми колективного управління в групах роботів / Каляєв І.А., Гайдук А.Р., Капустян С.Г - 2009. – 280 с

16. Eren T., Belhumeur P., Anderson B. et al. A framework for maintaining formation based on rigidity // Proc. of the 15th IFAC World Congress. Vol. 15. Barcelona: International Federation of Automatic Control, 2002. P. 1306–1306.

17. Rodrigues J., Figueira D., Neves C., et al. Leader-following graph-based distributed formation control // Robotica. 2009. N 75. P. 8–14.

18. Olfati-Saber R., Fax J. A., Murray R. M. Consensus and Cooperation in Networked Multi-Agent Systems // Proceedings of the IEEE. 2007. Vol. 95, N 1. P. 215–233.

19. Wang J., Nian X., Wang H. Consensus and formation control of discrete-time multi-agent systems // Journal of Central South University of Technology. 2011. Vol. 18, N 4. P. 1161–1168.

20. Zhengping W., Zhihong G., Xianyong W., et al. Consensus Based Formation Control and Trajectory Tracing of Multi-Agent Robot Systems // Journal of Intelligent Robotic Systems. 2007. Vol. 48, N 3. P. 397–410.

21. Lalish E., Morgansen K., Tsukamaki T. Formation tracking control using virtual structures and deconfliction // Proceedings of the 2006 IEEE Conference on Decision and Control. San Diego: IEEE, 2006. P. 5699–5705.

22. Zhonghai Z., Jian Y., Wenxia Z., et al. Formation control based on a virtual-leader-follower hierarchical structure for autonomous underwater vehicles //

International journal of advancements in computing technology. 2012. Vol. 4, N 2. P. 111–121.

23. Lewis M. A., Tan K. High precision formation control of mobile robots using virtual structures // Autonomous Robots. 1997. Vol. 4, N 4. P. 387–403.

24. Рогач В.Д., наук.керівник Кудрявцева М.С. Дослідження алгоритмів управління групою безпілотних літальних апаратів // XXIV Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті». – Харків: ХНУРЕ, 7-9 квітня 2020. – С. 32-33

25. Водолазький, І. А. Ройовий інтелект і його найбільш поширені методи реалізації / І. А. Водолазький, А. С. Єгоров, А. В. Краснов. - Текст: безпосередній // Молодий вчений. - 2017. - № 4 (138). - С. 147-153. - URL: https://moluch.ru/archive/138/38900/ (дата звернення: 09.10.2020).

26. Катуева Я.В. Аналіз складних систем в умовах неповноти інформації в задачі оптимізації надійності по поступовим відмов // Інформатика та системи управління. 2010. №4 (26). С. 61-68.

27. Абрамов О.В., ДігоГ.Б., ДігоН.Б., Катуева Я.В., Назаров Д.А. Параметричний синтез технічних систем в невизначених середовищах // Інформатика та системи управління. 2009. № 1 (19). С. 55-65.

28. Kennedy J., Eberhart RC Particle swarm optimization // Proceedings of IEEE International Conference on Neural Networks. 1995. Vol.4. IEEE Publ., 1995. P. 1942-1948.

29. Zolotukhin O., Kudryavtseva M. Authentication Method in Contactless Payment Systems – International Scientific and Practical Conference «Problems of Infocommunications. Science and Technology», October 9 - 12, 2018

30. Filatov, V., Yerokhin, A., Zolotukhin, O. Kudryavtseva, M Personalized Adaptation of Learning Environments 8TH INTERNATIONAL CONFERENCE ON ADVANCED OPTOELECTRONICS AND LASERS CAOL*2019 September 06-08, 2019, Sozopol, Bulgaria

31. Васильєв О. «Python на прикладах. Практичний курс».- Наука і техніка, 432 с.

32. Pham DT, Ghanbarzadeh A, Koc E, Otri S, Rahim S and Zaidi M. The Bees Algorithm. Technical Note, Manufacturing Engineering Centre, Cardiff University, UK, 2005

33. D. Karaboga, AN IDEA BASED ON HONEY BEE SWARM FOR NUMERICAL OPTIMIZATION,TECHNICAL REPORT-TR06,Erciyes University, Engineering Faculty, Computer Engineering Department 2005.

34. The Bees Algorithm – A Novel Tool for Complex Optimisation Problems D.T. Pham, A. Ghanbarzadeh, E. Koc, S. Otri , S. Rahim , M. Zaidi Manufacturing Engineering Centre, Cardiff University, Cardiff CF24 3AA, UK.

35. Методичні вказівки до організації виконання та захисту атестаційної роботи другого (магістерського) рівня вищої освіти за спеціальністю 122 Комп'ютерні науки, освітньої програми «Комп'ютерні науки» для студентів усіх форм навчання [Текст] / Упоряд.: М.В. Євланов, В.Г. Іванов, Л.М. Ребезюк, Н.В. Рябова. – Харків: ХНУРЕ, 2020. 58 с.

36. Filatov V., Yerokhin A., Zolotukhin O., Kudryavtseva M. The Information Space Model in the Tasks of Distributed Mobile Objects Managing //Information extraction and processing. – №47 (123), 2019.- p. 80-86