Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Методи розподіленого моделювання гетерогенних систем Інтернету речей

(тема)

здобувач 2 року навчання,
групи СПм-23-5
Олександр ЖУРАВЛЬОВ
(власне ім'я, прізвище)

Спеціальність
123 «Комп'ютерна інженерія»
(код і повна назва спеціальності)

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

Освітня програма
Системне програмування
(повна назва освітньої програми)

Керівник: проф. Максим ВОЛК
(посада, власне ім'я, прізвище)

Допускається до захисту

Зав. кафедри ЕОМ _____ Андрій КОВАЛЕНКО
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

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

Освітня програма Системне програмування
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“ ” _____ 2025 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Журавльову Олександру Юрійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Методи розподіленого моделювання гетерогенних систем Інтернету речей

затверджена наказом по університету від “ 21 ” квітня 2025 р. № 296ст

2. Термін подання студентом роботи до екзаменаційної комісії 16 червня 2025 р.

3. Вхідні дані до роботи

1. Технології моделювання хмарних систем
2. Технології організації гетерогенних комп'ютерних систем Інтернету речей
3. Існуючі системи моделювання
4. Методи та моделі моделювання Інтернету речей

4. Перелік питань, що потрібно опрацювати в роботі

1 Аналіз предметної області
2 Моделі та Методи розподіленого моделювання гетерогенних систем Інтернету речей
3 Експериментальні дослідження
4 Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) Слайд-презентація – 12 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної області	22.04.25-29.04.25	
2	Розробка моделей	30.04.25-05.05.25	
3	Реалізація алгоритмів	06.05.25-10.05.25	
4	Розробка структури програмних засобів	11.05.25-21.05.25	
5	Розробка програмних модулів	22.05.25-02.06.25	
6	Оформлення матеріалів кваліфікаційної роботи	03.06.25-05.06.25	
7	Подання кваліфікаційної роботи керівникові та її попередній захист	06.06.25-10.06.25	
8	Подання кваліфікаційної роботи на рецензування	11.06.25-12.06.25	

Дата видачі завдання 21 квітня 2025 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Максим ВОЛК
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 57 с., 12 рис., 2 табл, 1 дод., 44 джерела.

ХМАРНІ ОБЧИСЛЕННЯ, ТУМАННІ ОБЧИСЛЕННЯ, МОДЕЛЮВАННЯ, ІНТЕРНЕТ РЕЧЕЙ, БАЛАНСУВАННЯ НАВАНТАЖЕННЯ, ЯКІСТЬ ОБСЛУГОВУВАННЯ.

У роботі розглядається проблема управління перевантаженням у розподілених інформаційних системах на основі хмарних та туманних обчислень. Запропоновано динамічну систему управління перевантаженням (DCMB), яка забезпечує ефективний розподіл обчислювальних ресурсів та покращує якість обслуговування (QoS) у середовищі Інтернету речей (IoT). Основна ідея системи полягає у використанні механізмів пріоритетного обслуговування запитів та адаптивного балансування навантаження. Експерименти з використанням симуляторів iFogSim та CloudSim підтвердили скорочення часу обробки запитів та підвищення ефективності використання ресурсів. Отримані результати демонструють перспективність впровадження запропонованої системи для оптимізації роботи інфраструктури IoT.

ABSTRACT

Master’s thesis: 57 pages, 12 figures, 2 tables, 1 appendice, 44 sources.

CLOUD COMPUTING, FOG COMPUTING, MODELING, INTERNET OF THINGS, LOAD BALANCING, QUALITY OF SERVICE.

The paper considers the problem of congestion management in distributed information systems based on cloud and fog computing. A dynamic congestion management system (DCMB) is proposed, which provides effective allocation of computing resources and improves the quality of service (QoS) in the Internet of Things (IoT) environment. The main idea of the system is to use mechanisms for priority request service and adaptive load balancing. Experiments using the iFogSim and CloudSim simulators confirmed the reduction of request processing time and increased resource efficiency. The results obtained demonstrate the prospects of implementing the proposed system for optimizing the operation of the IoT infrastructure.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧКИ

БПЛА – безпілотний літальний апарат

ВМ – віртуальна машина

ОС – операційна система

ПЗ – програмне забезпечення

AKS – Azure Kubernetes Service

API – Application Programming Interfase

CBGA – Consensus-Based Grouping Algorithm

CoA – Change of Authorization

DDS – Data Distributed Service

DIS – Distributed Interactive Simulation

DEVS – Discrete Event System Specification

IaaS – Infrastructure as a Service

ISOA – Intelligent Self-Organized Algorithm

EKS – Elastic Kubernetes Service

ID – identity

FIFO – First In, First Out

IoT – Internet of Things

MQTT – Message Queue Telemetry Transport

NPC – nonpayer characters

RPC – Remote Procedure Call

RTOS – Real-Time Operating System

SaaS – Simulation as a Service

VM – Virtual Machine

UAV – Unmanned Aerial Vehicle

ВСТУП

Популярність Інтернету речей в цивільному житті, військовій практиці, розумних будинках, сільському господарстві, розумних містах змусив людей побачити перспективи його сумісного використання в інших галузях. Сукупність доступних датчиків та механізмів часто з’являються в житті як недороге, малоризикове та економічно ефективне спорядження. Дослідники зосередилися на забезпеченні децентралізованих, гетерогенних і недорогих пристроїв з можливостями автономної координації для виконання більш складних завдань, оскільки це важливий спосіб покращити гнучкість і надійність конкретного рішення Інтернету речей [1,2].

Оскільки різноманіття пристроїв та їх інтерфейсів є гетерогенним середовищем, архітектура розподіленої співпраці є важливим способом досягнення великомасштабної спільної взаємодії [3]. Централізована архітектура, яка була широко досліджена та застосована, має перевагу простішого дизайну алгоритму, але вона також має проблему високих вимог до мережі та центральних обчислювальних вузлів. Навпаки, кожен вузол в розподіленій архітектурі може спілкуватися та співпрацювати з іншими вузлом як незалежна сутність. Оскільки в мережі немає критичних вузлів, архітектура є високомасштабованою та надійною.

Відповідно до аналізу відповідних дослідників, широко використовувані методи розподіленого спільного призначення завдань можна розділити на евристичні алгоритми оптимізації [4,5], ринкові методи [6-9] та методи на основі альянсу [1,10] та інші [11]. Аналіз цих методів наведено у розділі 1.

Гетерогенні системи відносяться до архітектур, які відрізняються за своїми функціями, структурами, можливостями та розроблені для спільної роботи над досягненням комплексних цілей [12]. Це сукупність різноманітних платформ часто включає абсолютно різні пристрої, уможливлюючи більш багату взаємодію та ширшу сферу діяльності порівняно з однорідними

системи. Синергія в цих гетерогенних системах прагне використати відмінність сильних боків та можливостей кожного типу пристроїв, підвищення загальної ефективності, адаптивності, та функціональну стійкість у виконанні скоординованих завдань у різноманітних середовищ.

Однак перед ефективним розгортанням таких систем необхідно виконати складні завдання оптимізації або пошуку найліпшої конфігурації, тому дослідницьке співтовариство все ще стикається з численними фундаментальними технічними викликами. Ці виклики, як зазначено в [13], охоплюють такі сфери, як великі дані, Інтернет речей, складність завдань, автономне машинне навчання, масштабованість і неоднорідність, формування певної конфігурації та розподіл завдань, врахування людини, передача навчання, уніфіковані рамки, комунікаційні обмеження та невизначеність підключення. Крім того, існують складні проблеми застосування, описані в [14], включаючи адаптивні гетерогенні архітектури та методи моделювання роєвих систем; розподілене сприйняття і пізнання багатовимірних ситуацій [15]; інтелектуальне прийняття рішень і планування системи, якою можна керувати, довіряти та розвивати; автономне співробітництво управління робото-ройовими системами.

За останні роки було досягнуто значних успіхів у сфері керування системами Інтернету речей, планування завдань, управління обчислювальним процесом, мережевим зв'язком, та інтелектуальними алгоритмами. Перш за все, спільне визначення кластерів пристроїв охоплює технологію спільного позиціонування [16,17], спільне розпізнавання цілей, спільна оптимізація багатоджерельних датчиків та спільне супроводження обчислювального процесу та прийняття рішень [18]. Прийняття рішень і планування в кластерних системах включають традиційні методи, такі як угорський алгоритм, алгоритм аукціону, алгоритм рою частинок і алгоритм на основі коридору методів [19], а також мережі спільних завдань, побудовані за допомогою машинного навчання моделі [20].

Спільне керування кластерами пристроїв також було досліджене в теорії

дослідження фундаментальних режимів, таких як контроль узгодженості, контроль формування [21] і контроль стеження [22].

Коли виконують складні завдання незалежно від того, працюють прибори самостійно чи співпрацюють у кластерах їм потрібна архітектура моделювання для інтегрованого тестування та перевірки.

Завдяки своїм характеристикам технологія розподіленого моделювання має потенціал для обслуговування схем верифікації моделювання, що найбільше нагадує фактичне робоче середовище гетерогенних розподілених систем, задіяних у алгоритмах спільної роботи.

Минулі розробки в технології розподілених систем охоплювали дослідження синхронізації, засновані на повідомленні діалогу [23], реалізації гнучких технологій доступу, що засновані на активних послугах [24], і забезпечення надійної роботи з гарантією підтримки [25]. Ці роботи позиціонують розподілене моделювання як життєздатний підхід до точного моделювання та перевірки алгоритмів спільних завдань у кластерах [26-28], сприяючи ефективній оцінці та тестуванню в середовищах, ближчих до реальних умов. Завдяки розподіленому моделюванню стає можливим перевірити спільну поведінку та продуктивність систем у складних середовищах, а також оцінити міцність і надійність складних систем. Це сприяє значній підтримці гарантії швидкого розвитку та застосування технології Інтернету речей.

Крім того, існує багато систем моделювання, які також можуть забезпечити роботи гетерогенних систем при розгляді того, як люди керують подібними процесами [29–30].

Базуючись на теоретичній роботі попередників, ми пропонуємо систему моделювання з відкритою архітектурою. У цій роботі ми спочатку проведемо аналіз деяких відповідних технічних рішень, які вплинули на дану роботу (розділ 1). Далі представлена архітектура з трирівневою версифікаційною платформою моделювання та пропонується реалізація сумісної роботи алгоритмів гетерогенних систем Інтернету речей (розділ 2). В

експериментальному розділі (розділ 3) описані експериментальні процедури та розробка сценаріїв експериментів з обговоренням характеристик мережі. Також наведено довідкові пропозиції щодо створення подібних архітектури. У висновках пропонуються аналіз дослідження мереж взаємодії пристроїв Інтернету речей. У додатку наведена презентація до кваліфікаційної роботи.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Методи розподіленого спільного управління виконанням завдань в гетерогенних системах Інтернету речей

Алгоритми евристичної оптимізації широко використовуються, оскільки вони не вимагають інформації про градієнт і не покладаються на моделі задач з хорошими математичними властивостями.

Наприклад, у літературі [31] пропонується вдосконалений алгоритм оптимізації, заснований на аналізі поведінки голубів, для вирішення проблеми оптимізації кооперативних цільових пошуків. Він побудован по принципу централізованій архітектури керування. Для спільного виконання завдань з декількома пристроями в [4] запропонували метод планування за допомогою інтелектуального самоорганізованого алгоритму (ISOA). Пристрої обмінюються один з одним інформацією про стан і планування, а також локально оптимізують планування маршруту за допомогою покращеного алгоритму розподіленої колонії мурашок для оновлення планування маршруту (для мобільних пристроїв, роботів, БПЛА) та повторюють процес, доки завдання не буде виконано. Однак у статті припускається, що всі пристрої є однорідними.

Інша стаття [32] реалізує розподілений метод призначення завдань для ройових розвідувальних місій БПЛА на основі алгоритму вовчої зграї, включаючи алгоритм спільного пошуку на основі розвідувальної поведінки вовка та метод спільного призначення завдань для атаки після виявлення цілі. Алгоритм має хорошу масштабованість, але не враховує ризик пошуку невідомого середовища при оптимізації схеми.

Ринковий метод – це метод, за якого учасники торгів оцінюють переваги виконання різних завдань, транслюють пропозиції один одному та перемагають із найкращим рішенням, а учасники проводять повторну оцінку

після оновлення середовища або плану розподілу, доки немає конфлікту. З метою вирішення проблеми управління завдання гетерогенної системи, в статті [6] запропоновано алгоритм призначення завдань на основі покращеного CBGA (покращений алгоритм групування на основі консенсусу, похідний від CBBA [33]). Алгоритм має просту структуру, але менше уваги приділяється таким факторам, як взаємодія між пристроями.

Для забезпечення розподілу завдань у реальному часі в бездротових сенсорних мережах з обмеженими ресурсами, автори [34] запропонували схему на основі зворотного аукціону з використанням адаптивного алгоритму для кожного вузла (учасника торгів) для локального розрахунку його найкращої відповіді на ставку з негладкою та увігнутою функцією виплати.

Формування альянсу розділяє гетерогенну систему пристроїв на кілька невеликих альянсів за допомогою таких стратегій, як кооперативні ігри [10]. Ця архітектура спочатку розподіляє завдання між альянсами, а потім перерозподіляє отримані завдання всередині альянсу, щоб ефективно зменшити розмір задачі. Автори [1] використовують мережевий протокол розширеного контракту з багаторівневими зв’язками для реалізації спільного керування роями пристроїв, перевагою якого є швидкість розв’язання, коли масштаб рою великий. Однак ця література ігнорує вплив методу поділу підмножин пристроїв на вплив ройової поведінки. Наприклад, два пристрої, які повинні були співпрацювати, поділяються на різні альянси, що призводить до зниження якості рішення.

Дослідники також намагалися поєднати переваги різних архітектур. Коли проблема має складні обмеження, важко досягти гарного результату шляхом прямого застосування CBBA та інших методів, а повторні переговори спричиняють високі витрати на комунікацію. Тому деякі дослідники поєднують евристичні алгоритми з ринковими методами. Наприклад, можна врахувати обмеження часу завдання та обмеження перешкод, використовує інтелектуальні алгоритми оптимізації локально для о птимізації схеми, а потім погоджується з іншими пристроями.

Так само [36] розглядає мінімальну суму відстаней і мінімальний максимальний час завершення як цілі оптимізації та спочатку використовує генетичний алгоритм (GA) для локальної оптимізації, а потім використовує отриманий від CBAA алгоритм для досягнення консенсусу між вузлами.

З точки зору факторів, які розглядаються в дослідженні спільної місії гетерогенної системи пристроїв, розглядаються фактори, які в основному включають відстань пристроїв [7], площу покриття [4], планування маршруту [4], уникнення заборонених параметрів [2] тощо, тоді як загроза зовнішніх факторів співпраці в межах зовнішнього середовища розглядається рідка.

1.2 Існуючі системи моделювання Інтернету речей

За останні роки платформи моделювання Інтернету речей розвивалися майже одночасно з самою архітектурою розподілених гетерогенних систем (таблиця 1.1). Спочатку дослідницько-конструкторські підприємства разом з апаратними платформами надавали модулі моделювання, які можна було інтегрувати, наприклад в Simulink, дозволяючи користувачам сприяти вторинній розробці. Між тим, складність роботи росла зі збільшенням структур, що призвело до ширшого кола завдань. У міру розширення різноманітності приборів, складність моделювання також зросла. Отже, єдиний модуль (платформа моделювання) більше не міг задовільнити вимоги користувачів, і незалежне програмне забезпечення для моделювання почало з’являтися у співпраці між підприємствами та університетами [37].

Згодом, коли розробка функцій та інші технології мультисенсорного синтезу прогресували, платформи моделювання повинні були не тільки забезпечувати базове динамічне моделювання, але також імітувати цифрові сигнали від кількох датчиків, таких як відеокамери (навіть інфрачервоні) у [37] та SONAR для звукової навігації і визначення діапазону, що вимагало комплексного сприйняття ситуації у віртуальному середовищі.

Таблиця 1.1 – Програмні платформи для моделювання гетерогенних розподілених систем Інтернету речей

Програма	Розробник	Засоби, платформи	Моделюван-ня фізики	Сенсори, прилади
OpenRave	Carnegie Mellon University	OSG	IKFast	IMU, RGB, Robots
ARGoS	Free University of Brussels	Qt-OpenGL	ODE Chipmunk	IMU, RGB, Robots
Airsim	Microsoft	Unreal Engine	PhysX fastsim	IMU, RGBD, Segment, LiDAR, Drones, GV
IGibson	Stanford University	Unity	Unity	IMU, RGBD, Segment
Issac Gym	Nvidia	Issac Gym	PhysX	IMU, robots
OpenAI-Gym	OpenAI	Gym	Mujoco	IMU, Multijoint robot
MARSIM	University of Hong Kong	ROS OpenGL	ROS	IMU, LiDAR (HD)
Neurons Gym	Institute of Automation, Chinese Academy of Scien	Unity3D	Unity3D	IMU, LiDAR, Mecanum wheeled robot
RotorS	University of Zurich	Gazebo (ROS), OpenGL	ODE	IMU, RGBD, Drones

З прогресом технологій штучного інтелекту знайшли платформи моделювання з більшою корисністю у наданні інформації. Це використання

сприяє ефективним механізмам навчання, що призводить до появи нових алгоритмів навчання. Це дозволяє системам працювати краще для багатьох нелінійних задач.

Розробка технологій моделювання постійно прогресує. Вони еволюціонують від моделювання окремих об’єктів до багатьох різнорідних об’єктів, від простого статичного середовища до складних середовищ. Крім того, масштаби програмного забезпечення дедалі зростають.

Попереднього підходу до моделювання та доступу до кількох контролерів на одному обчислювального терміналу вже недостатньо для задоволення сучасних потреб. Архітектура розподіленої симуляція потрібна для вирішення проблем масштабованості. Тому завдання полягає в тому, щоб запропонувати масштабовану розподілену архітектуру спільного моделювання для задоволення зростаючих вимоги до функціональності та складності гетерогенних систем Інтернету речей.

Крім того, розподілений характер архітектури середовища моделювання має певні переваги: точніше відображає наявні комунікаційні характеристики в реальних робочих умовах, незалежно від того, чи використовується гетерогенна система з централізованою або розподіленою співпрацею. Ця перевага підкреслює значимість використання розподіленої мережевої структури для розгортання гетерогенних складних систем.

Хан та ін. [38] запропонували середовище моделювання, яке розроблене для застосувань IoT, який підтримує сервіс-орієнтовану об'єктну модель, що керована подіями. В [39] запропоновано програму SimIoT, яка розроблена як додаток до середовища моделювання SimIC. Віна надає декілька механізмів мережевого зв’язку для різноманітних датчиків IoT та хмарних датацентрів. Також пропонується механізм моделювання EdgeCloudSim, який зменьшує бар’єри у звичайних хмарних системах моделювання для сценаріїв кордонних обчислень. Фактично, це розширення середовища CloudSim [40].

Харшит та його колеги розробили симулятор IoT та туманних обчислень під назвою iFogSim, який є доповненням до CloudSim і реалізований на

платформі Java. У своїй роботі автори дослідили вплив різних стратегій управління ресурсами, оцінюючи їх за параметрами затримки, перевантаження мережі та вартості обчислень.

Проте iFogSim має певні обмеження: (а) оскільки він базується на Java, його можливості у моделюванні ключових параметрів мережевої інфраструктури є обмеженими; (б) він не забезпечує повну сумісність із різними версіями Java, а також має недостатньо детальну документацію, що ускладнює розробку та розширення функціональності.

Запропонований нами симулятор усуває ці недоліки, надаючи розширений мережевий інструментарій для моделювання таких параметрів, як затримка, перевантаження мережі та втрата пакетів. Це забезпечує значно вищу гнучкість у дослідженні та тестуванні різних мережевих конфігурацій із динамічними характеристиками.

Також була запропонована техніка для управління великою кількістю пристроїв IoT, зокрема мобільних. Середовище реалізовано на платформі CloudSim і використовує програми автоматизації процесу моделювання. Задрапована система IoTSim дозволяє обробляти BifData в середовищі IoT. Тут використовується модель MapReduce. Можна знайти практичні дослідження для демонстрації ефективності таких симуляторів. Основна мета подібних програмних систем полягає в наданні дослідникам інструментів для навчання взаємодії з пристроями Інтернету речей без додаткових пристроїв. Користувачі можуть ознайомитися з особливостями використання систем IoT для формування початкових знань систем на базі IoT.

Ще однин симулятор – SimpleIoTSimualtor [41], який використовується для симуляції середовищ IoT, які містять датчики та пристрої с підтримкою множині специфічних протоколів IoT, (наприклад, CoA та MQTT). Симулятор SimpleIoTSimulator виконується у 64-розрядній операційній системі RedHat Linux, тому має високу продуктивність.

Запропонована системи CrowdSenSim [42] розроблена для моделювання застосувань для розумного міста. В роботі моделюється вуличне освітлення.

Це приклад, а загалом система моделювання можна використовувати для застосувань, що реалізують збір даних із різних датчиків. CrowdSenSim має відкриту ліцензію на використання.

Ще однин симулятор – SimpleIoTSimualtor [42], який використовується для симуляції середовищ IoT, які містять датчики та пристрої с підтримкою множині специфічних протоколів IoT, (наприклад, CoA та MQTT). Симулятор SimpleIoTSimulator виконується у 64-розрядній операційній системі RedHat Linux, тому має високу продуктивність.

Також свій симулятор має фірма Google. Він має назву Google IoT та взаємодіє з різними сервісами Google [44]. Також він має високий рівень масштабованості і використовує багато пристроїв для збору даних. Має своє середовище візуалізації результатів моделювання. Важливим елементом є можливість використання Load Balancer для управління та балансування навантаження сумісно з AppEngine.

1.3 Постановка мети та завдань дослідження

Таким чином, недоліки сучасних рішень, на усунення яких спрямована дана робота це:

- обмежена масштабованість – запропонована архітектура дозволяє ефективніше керувати великою кількістю пристроїв;
- високе навантаження на центральні вузли – розподілене управління зменшує перевантаження серверів;
- недостатня точність синхронізації станів об’єктів – новий алгоритм синхронізації знижує затримки;
- висока складність інтеграції нових пристроїв – запропонована архітектура підтримує гнучку інтеграцію.

Метою роботи є розробка та впровадження методу розподіленого моделювання для гетерогенних систем Інтернету речей (IoT), який дозволяє

ефективно керувати ресурсами, синхронізувати дані та забезпечувати високу продуктивність при взаємодії множини IoT-пристроїв.

Задачі роботи:

- провести аналіз існуючих методів управління та моделювання гетерогенних IoT-систем;
- розробити трирівневу архітектуру розподіленого моделювання;
- реалізувати механізм керування синхронізацією стану об’єктів та обчисленнями в симуляторі;
- оцінити ефективність запропонованого методу за допомогою експериментальних досліджень;
- запропонувати шляхи оптимізації взаємодії між IoT-пристроями.

Об'єкт дослідження: процес моделювання гетерогенних розподілених системи Інтернету речей.

Предмет дослідження: методи та алгоритми управління ресурсами, синхронізації даних та комунікації в розподілених IoT-системах.

2 МОДЕЛІ ТА МЕТОДИ РОЗПОДІЛЕНОГО МОДЕЛЮВАННЯ ГЕТЕРОГЕННИХ СИСТЕМ ІНТЕРНЕТУ РЕЧЕЙ

2.1 Принципи побудови системи моделювання

Спираючись на досвід проектування модулів контролю та сприйняття від суміжних робіт, а також з урахуванням координації датчиків, пристроїв, мультироботних систем, а також взаємозв'язки в кінематиці, динаміці та інших аспектах, ми запропонували трирівневу розподілену архітектуру моделювання, призначену для гетерогенних систем Інтернету речей.

Беручи до уваги потенційну складність, пов’язану зі співпрацею роботи гетерогенних робототехнічних систем, ми прийняли три фундаментальні принципи для проектування:

- процес моделювання та результати повинні відповідати законам фізики, забезпечуючи їх узгодженість між розподіленими вузлами моделювання;

- можливості платформи руху кожного пристрою визначаються структурою, середовищем, атрибутами, аксесуарами і системою контролю;

- алгоритми спільної роботи, які використовуються в гетерогенних системах роботів, мають бути перевірені за допомогою автентичних розподілених методів перевірки.

Згідно з цими трьома принципами проектування ми пропонуємо організувати архітектуру моделі та алгоритми за трьома ключовими рівнями (рисунок 2.1).

Рівень 1. Мережевий рівень розподіленого віртуального моделювання. Його головним чином спрямовано на підтримку єдиного віртуального середовища для кожного розподіленого учасника.

Як більшість масових онлайн-рольових ігор використовували популярну техніку на основі мережі клієнт-сервер (C/S), де клієнти зберігають локальну

копію віртуальної мережі, а сервер зберігає параметри та результати моделювання відповідно чотирьом послугам, таким як фізична взаємодія, керування сценаріями, служба синхронізації та керування сеансами.

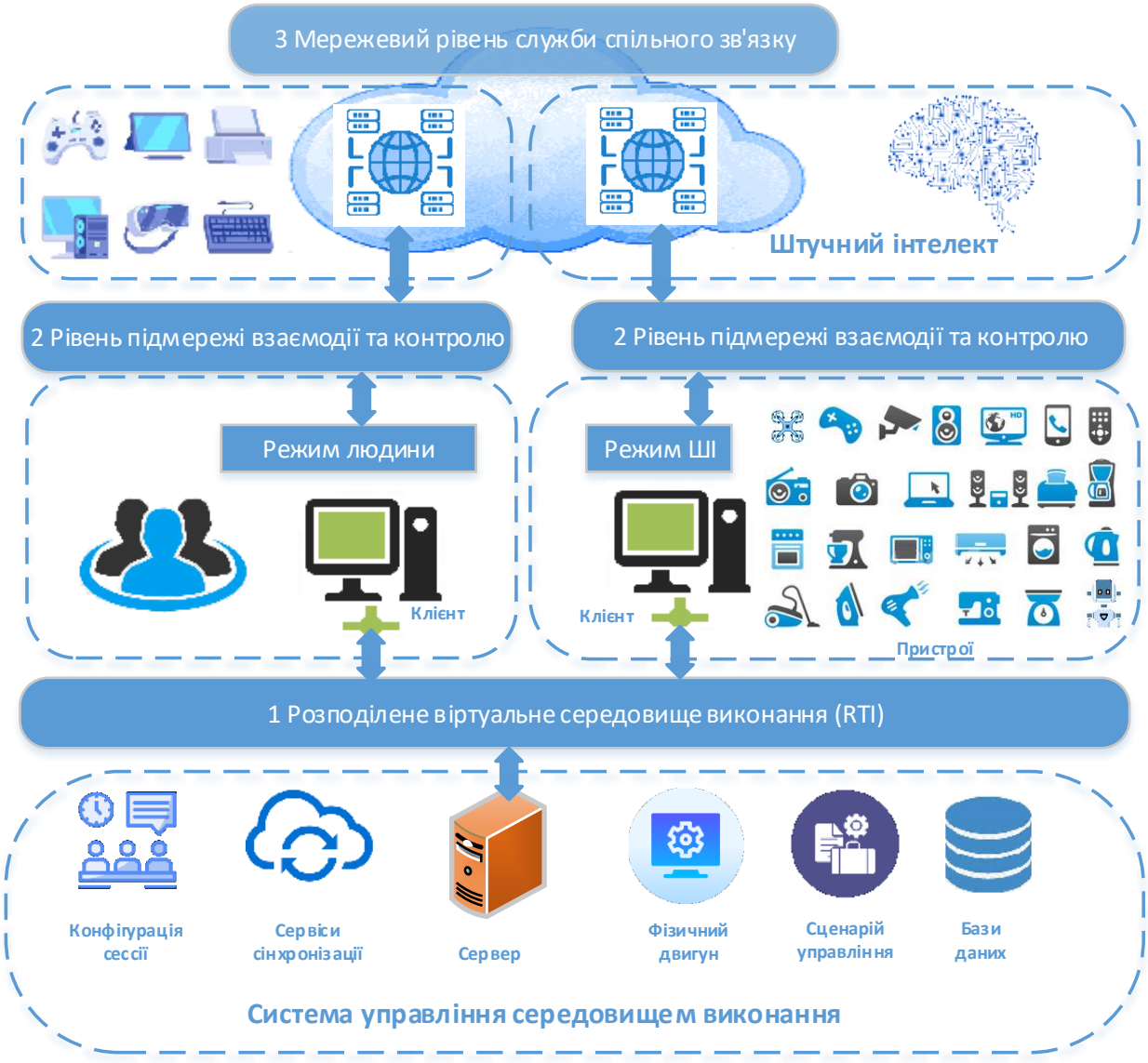

Рисунок 2.1 – Архітектура платформи розподіленого моделювання з трьома ключовими рівнями мережі

Рівень 2. Рівень підмережі взаємодії та контролю. Збережено багато переваг існуючих рішен. Розроблена підмережа з’єднує віртуального користувача із його зв’язаним «мозком» (тобто тим хто контролює поведінку користувача або пристороюю).

Віртуальний користувач може бути рухомою платформою з приводом і

сенсорними блоками, тоді як «мозок» може бути зрілим апаратним забезпеченням із набором RTOS (операційна система реального часу) або навіть людиною, тобто підмережа може працювати або в штучному інтелектуальному режимі або режимі людини.

Рівень 3. Мережевий рівень служби спільного зв'язку. Ця ідея, як було доведено, є успішною у розподілених вбудованих системах реального часу служби розподілених даних (DDS), також прийнято тут для надання базової інформації механізмам передачі. Вважається, що він підтримує більшість можливих співробітництв між верхнім обчислювальним вузлом гетерогенних систем Інтернету речей.

2.2 Розподілене віртуальне середовище виконання

Для організації рівних умов для усіх учасників одного віртуального середовища, розподілена мережа віртуального моделювання надає серію простих у використанні служб. Серед них служба керування сеансом, яка забезпечує механізми довгострокового з’єднання для віртуальних клієнтів та сервера (рисунок 2.2), у той час, коли керування сценаріями, фізичний двигун та служба синхронізації працюють разом, щоб координувати обробку, пов’язану з моделюванням контенту на розподілених платформах.

З точки зору розподіленого моделювання, керування сеансом є функціональною основою для зв’язку архітектури сервер–клієнт. Це універсальна архітектура як для централізованого так і децентралізованого керування. Надані основні функції в основному включають надання доступного порту під час створення служб моделювання на боці серверу бік. Клієнт виявляється та приєднується до серверної сторони в мережі, що забезпечує вибір ролі і публічність об'єктів віртуальної системи Інтернету речей для конкретних завдань моделювання.

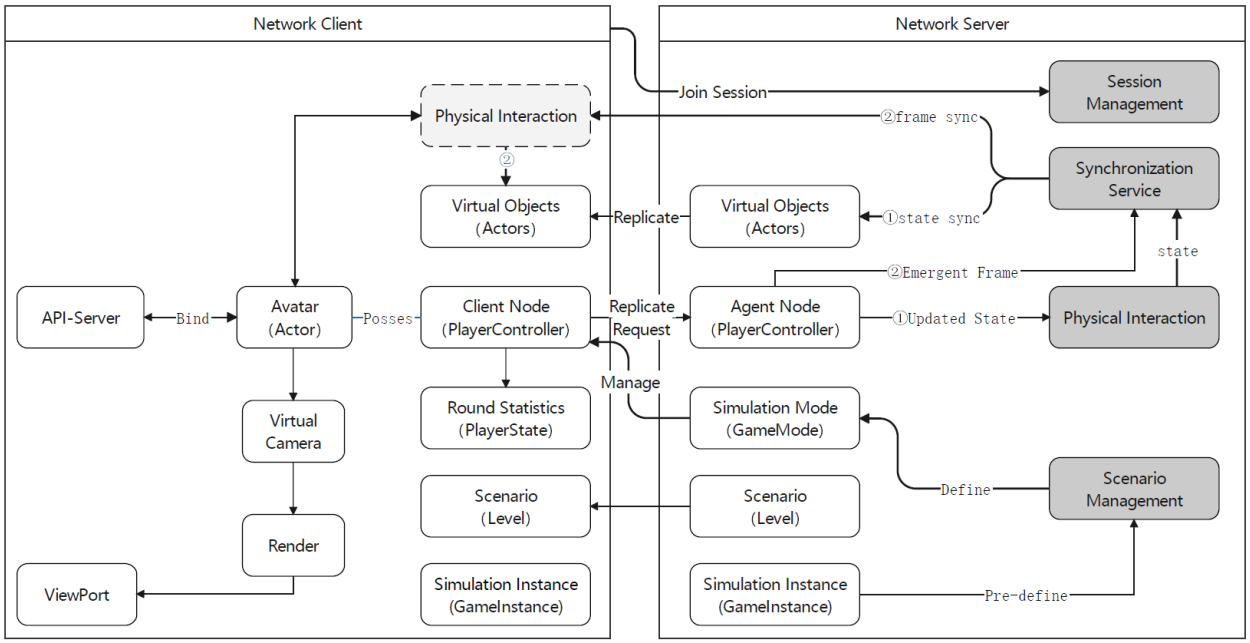

Рисунок 2.2 – Архітектура розподіленого середовища віртуального моделювання

Весь сеанс розподіленого моделювання ділиться на три етапи (рисунок 2.3).

Етап 1. Початок. Після створення та приєднання до сеансу симуляції сервер контролює для всіх учасників синхронне входження клієнтів у їх локальну карту моделювання, генерує відповідні віртуальні об'єкти на початковій позиції, встановленій у сценарії управління, і отримує дозволи на контроль над ними.

Етап 2. Процес моделювання. Під час моделювання, управління сеансом забезпечує безперервну підтримку з'єднання для синхронізації даних між сервером і клієнтом. Ці синхронізовані дані містять індивідуальну динаміку об’єкти на сцені, а також об’єкти роботизованої системи, якими керує клієнт.

Етап 3. Завершення. Після завершення завдання розподіленого моделювання, клієнти виходять із сеансу, керованого сервером і завершаються усі служби підтримки даних. Усі ці функції є налаштуваннями та оновленнями механіки багатокористувацьких сеансів ігрового движка.

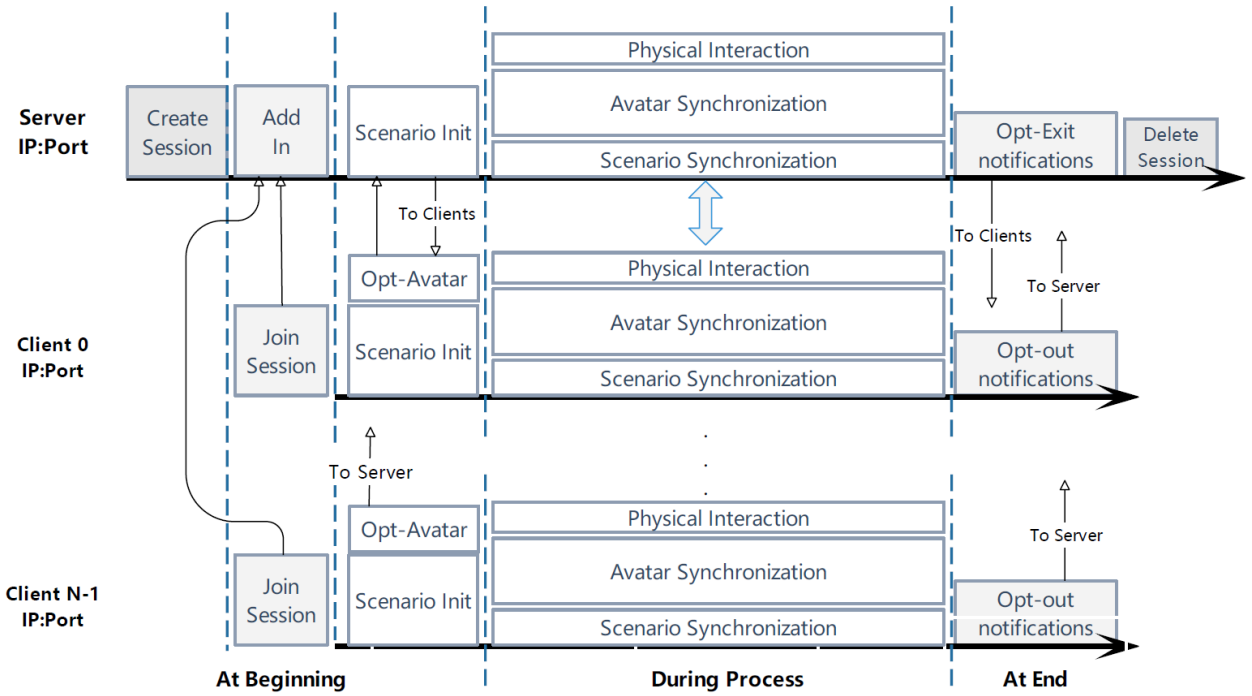

Рисунок 2.3 – Організація сеансу розподіленого моделювання

Сеанс в основному організовує сервер для клієнтів у довгострокових з'єднаннях і автоматично доставляє дані для синхронізації сценарію, моделей пристроїв, датчиків та реалізації фізичної взаємодії.

2.3 Управління сценаріями

Як і більшість систем моделювання та віртуальних ігор, сценарій зазвичай визначає глобальною системою координат у віртуальному просторі (або в локальній тривимірній декартовій координаті або за допомогою сферичних координат Землі). Проводиться ініціалізація різних об’єктів сутності в цій системі координат і система керує кожним об’єктом моделювання, наприклад ландшафтом, будівлями, дороги, неігровими персонажами (NPC), робототехнікою.

Як показано на малюнку 2.4, керування сценарієм – це в основному питання зберігання та обчислення.

Різниця полягає в тому, що сервер оновлює глобальний сценарій, а клієнти лише діють локальний простір. Розрізняємо бік серверу та клієнту.

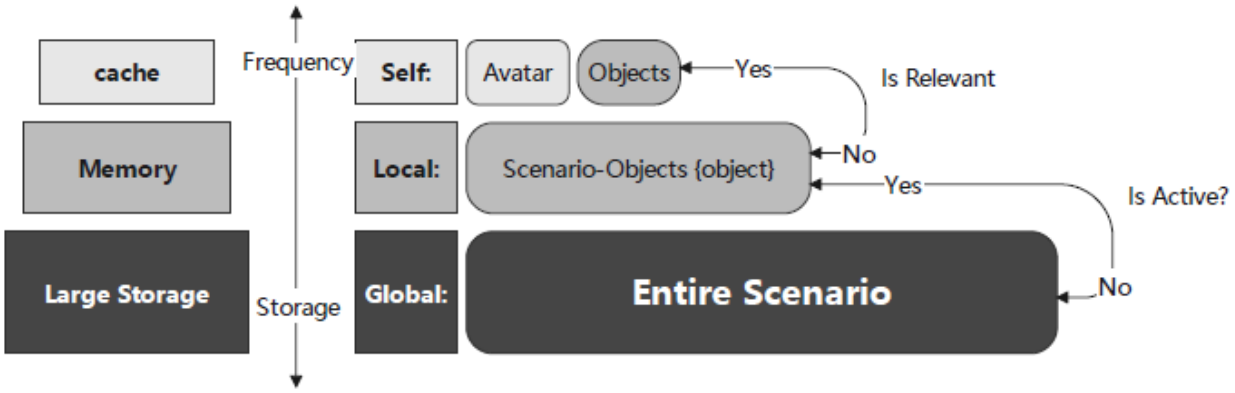

Рисунок 2.4 – Зберігання планів управління сценаріями на основі локальних залежностей та активностей

На сервері. На основі фундаментального розуміння того, що сервер має вищий рівень паралельних обчислень та можливості читання/запису пам’яті, він відповідає за підтримку глобальної інформацію про віртуальне середовище. При отриманні запиту від клієнта на оновлення властивостей об’єкта, сервер спочатку перевіряє потенційні конфлікти, а потім визначає вузол багатоадресної розсилки список на основі рівня активності кожного клієнта, задіяного в оновленні, і, нарешті, розповсюджує результати оновлення для відповідних клієнтів.

На боці клієнта. Керування сценою на стороні клієнта завантажує локальні дані об’єкта відповідно до специфіки вимоги кожного клієнта. Існує три основні категорії для обробки даних об’єктів:

- об’єкти, тісно пов’язані з контрольованими об’єктами, зберігаються в кеші для ефективного доступу і підтримки динамічних обчислень на більш високих швидкостях;
- об’єкти, які не є безпосередньо такими, на які впливає клієнт, але потребують усвідомлення, зберігаються в пам’яті та синхронізуються з командами реплікації серверу;
- непов’язані об’єкти встановлюються в неактивний стан і залишаються збереженими на жорсткому диску як ресурс.

2.4 Фізичний двигун

Для втілення фізичних законів у систему моделювання створено фізичний двигун (Physical Engine). Він забезпечує механізми взаємодії для підтримки обчислень на пропонуємой розподіленій платформі. Зазвичай обчислення взаємодії складається з трьох ключових етапів:

Етап 1. Оскільки певна взаємодія має бути викликана якоюсь конкретною умовою, ми можемо визначити умовну функцію F , як у рівнянні (2.1). Для довільно вибраних об'єктів z_i та z_j з узагальненими параметрами стану X_i та X_j, позитивність функції F може вказувати, чи виконуються умови взаємодії. Математично функція F є бінарнім відношенням:

$$f_{ij} = C\left(X_i, Y_j\right). \tag{2.1}$$

За допомогою графу $G = (V, R)$ може описувати відносини взаємодії всередині об’єктів. The множина вершин $V \triangleq \{ z_i \}$ і множина ребер $R = \{r_{i,j}\}$ будують матрицю суміжності, з кожним елемент $r_{i,j}$:

$$r_{ij} = \begin{cases} 1, f_{i,j} \geq 0 \\ 0, \text{в іншому випадку} \end{cases}. \tag{2.2}$$

Для тих дискретних взаємодій, керованих подіями, які можна лінійно розділити на пари, перехід безпосередньо до етапу 3.

Етап 2. Для відносно складних випадків, замість перерахування різних видів взаємодії моделі в деталях, ми використовуємо стандартні гамільтонові системи як теоретичний представник динаміки багатьох твердих тіл [50]. Припустимо, що існує зв’язний підграф $G' = (V', R')$. Цей підграф G' ілюструє набір об'єктів V' як ізольовані від інших об'єктів за допомогою відстані l. Ребра в R' і відповідні обмеження визначають дотичне розшарування (тобто простір

конфігурації), охоплений об’єктами в V', наприклад, разом із обмеженням якимось омеженням. Зв’язані об'єкти, підпорядковані принципу Даламбера-Лагранжа, визначають віртуальні переміщення та обмежені сили там, де були б також з припущенням, що є проблемою багатьох твердих тіл, які підлягають принципу найменшої дії Гамільтона. Рівняння руху є описується дифірінційними рівняннями. Для простоти викладення ми не будеми приводити їх у роботі.

Етап 3. Припустимо, що ми вже маємо явний шаблон виразу дифірінційного рівняння для кожного об’єкта, що бере участь у взаємодії підграф $G' = (V', R')$. Еволюцію ізольованих взаємодій системи можна розібрати та оновити для кожного розподіленого клієнта.

$$X_i(t+T)= f\left(X(t)_i, Y(T)_j\right). \tag{2.3}$$

2.5 Організація синхронізації, моніторингу та управління

Служба синхронізації є механізмом підтримки обмеженої взаємодії для кожного вузла у розподіленій віртуальній мережі. На рисунку 2.2 показано два шляхи синхронізації, а саме: оновлення фрейму та стану.

В об’єктах із зв’язками (наприклад, обмеженнями стану), якщо їх більше ніж два пов’язані клієнти, фрейм є керованим подією способом реконструкції руху моделювання пов'язаних клієнтів. Цей конкретний метод заснований на віддаленому виклику процедур (RPC), і сервер запускає відповідь відповідного клієнта для реконструкції стану моделі (тобто крок 2 у фізичній взаємодії).

Оновлення стану є багатоадресним методом у циклі моделювання. Клієнт визначиє, чи змінював локальний об’єкт для кожного інтервалу часу свій стан та надсилає запит на синхронізацію до серверу, якщо такі зміни відбулися. Сервер приймає запит і підтверджує його дійсність, а потім пакети об'єкта зміни атрибута програми в наступному стані синхронізації формують

фрейм і синхронізують його з відповідним клієнтом відповідно до списку багатоадресної передачі. Вибір часових інтервалів корелює з кількістю входжень в іншого клієнта локальних зонах і не перевищує верхню межу, дозволену мережею. Припускаючи це усі властивості з кількістю *D* визнаються такими, що потребують синхронізації з іншими *n – 1* клієнтів протягом допустимого часу *Ts* синхронізації. Таким чином мережевий трафік, який забезпечує відновлення стану можна розглядати як потік *Sv* може бути визначений як

$$S_v = \frac{(n-1)D(n)}{T_s}. \tag{2.4}$$

Мережа моніторингу та контролю є мережею один-до-одного, абстрагованою від Airsim (рисунок 2.5).

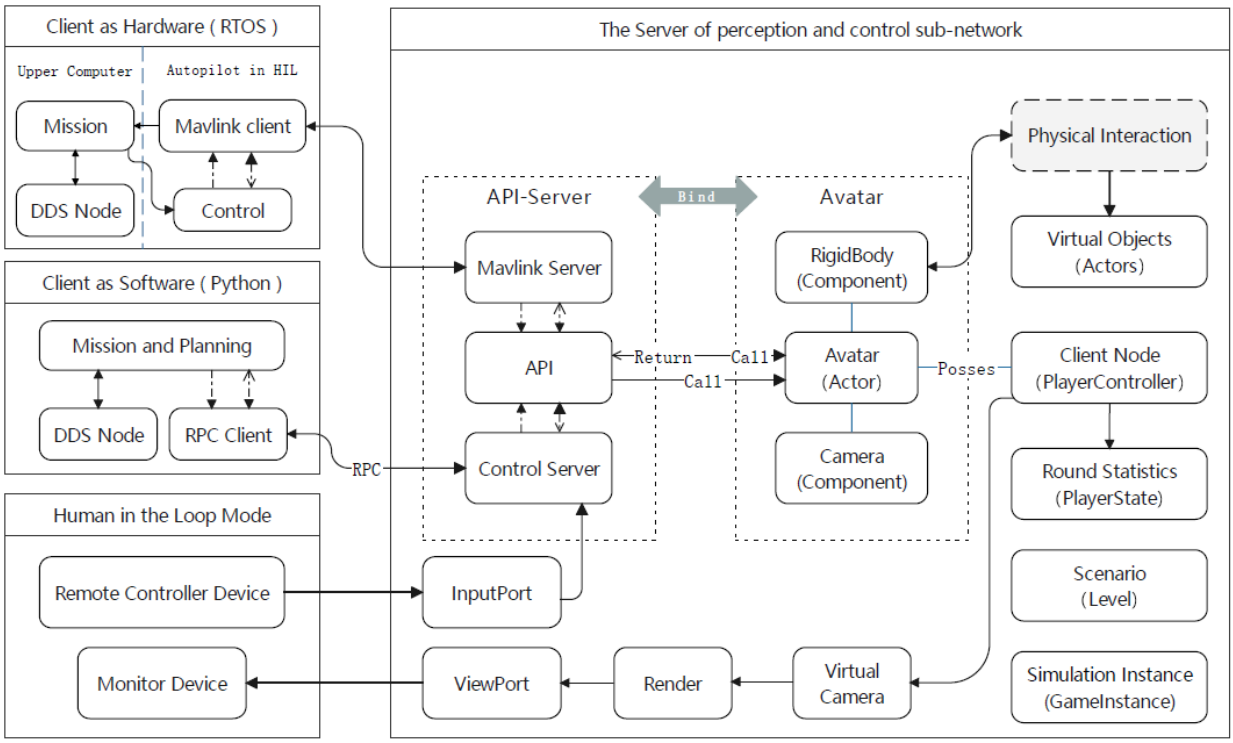

Рисунок 2.5 – Архітектура підмережі моніторингу та управління

Дані датчиків, сигнали приводу, дані приборів та бажаний стан можуть

використовувати Mavlink протокол для підключення апаратного забезпечення та віртуального клієнта (рисунок 2.5) через послідовний порт або використання віддаленого виклику процедур (RPC) локально для реалізації взаємодії між ними, програмного забезпечення та віртуального клієнта або доступу до користувача через периферійний пристрій взаємодії людина–комп’ютер.

На відміну від інших дворівневих мереж, підмережа моніторингу та контролю є незалежним механізмом підключення, для якого потрібна лише швидкість інтерфейсу передачі даних локального обладнання або швидкість читання/запису міжпроцесного зв’язку, щоб відповідати пропускній здатності, вимогам стану даних та контрольних інструкцій.

2.6 Режими штучного інтелекту та людини

Об’єктом верифікації моделювання для дизайну цієї платформи є робота алгоритма планування та контролю. Для клієнтів у режимі штучного інтелекту доступ за одним з двох алгоритмів. Методи в основному поділяються на апаратні в циклі та програмні в циклі.

Оскільки частота оновлення даних датчиків і даних керування фіксована, це також може бути встановлено, де m_i – розмір i-го дату, а f_i – відповідна частота оновлення. Таким чином, середній трафік на підмережі моніторингу і контролю становить

$$S_c = \sum_{i=1}^{n} f_i \cdot m_i \cdot \tag{2.5}$$

Апаратний цикл завдяки внеску спільноти з відкритим кодом можна безпосередньо інтегрувати за допомогою Mavlink. В циклі апаратного забезпечення виконується ввімкнення апаратного забезпечення автономного управління відповідною роботосистемою у віртуальному середовищі.

Обладнання в цикл зазвичай передається через послідовний інтерфейс зв'язку з узгодженою швидкістю передачі даних швидкість і фактична ефективна смуга пропускання відповідно до рівняння 2.5.

Моделювання програмного забезпечення в циклі, засноване на міжпроцесному режимі зв'язку, з'єднує вхід і вихід алгоритму штучного інтелекту із запитом і сервісної форми віддаленого виклику процесу, а алгоритм штучного інтелекту відповідає для сприйняття навколишнього середовища. Вирішення завдань, планування дій і конкретного контролю виконання функцій вузла та організаційної основи не обмежується архітектурою трирівневої розподіленої системи моделювання, якщо вона відповідає обмеженню швидкості зв'язку процесу локального контуру моніторингу-контролю.

Цикл у режимі людини надає дві можливості використання. Перший – безпосередньо надіслати введення ручного дистанційного керування людиною як командного сигналу до кінцевого API роботосистеми, яка є формою взаємодії людини з комп’ютером із безпосереднім керуванням. По-друге, клієнт-оператор існує в цифровому просторі як аватар, який спрямований на перевірку спільного режиму людино-машинного симбіозу.

Режим прямого керування в основному пов'язаний з керуванням роботою через послідовний порт симулятора контролера та реалізує режим симбіозу людини–комп’ютера через OpenXR доступ до пристроїв взаємодії людини з комп’ютером VR або AR.

2.7 Спільна комунікаційна мережа обслуговування

Мережа обслуговування зв'язку є децентралізованою і повністю рівноправною розподіленою структурою. Відрізняється від первинно-вторинної структури розподіленої мережі віртуального моделювання і сприйняття та структура підмережі управління «один-на-один».

Для організації зв'язку було використано FastDDS як проміжне

програмне забезпечення для цього рівня мережі. Однак, з точки зору побудови моделювання, варто зауважити, що спільну комунікаційну мережу та розподілене віртуальне моделювання мережі поділяють фізичну пропускну здатність мережі, але, загалом, пропускну здатність програми використання пропускної здатності тісно пов’язане з самою проблемою співпраці, і ми може дати лише оцінку запасу використання пропускної здатності.

Припускаючи, що фактична максимальна пропускна здатність, доступна для мережі, становить W і розподілена мережа віртуального моделювання використовує пропускну здатність S_v у рівнянні (2.4), таким чином, ми маємо,

$$S_{Service} = W - S_v . \tag{2.6}$$

Як на рисунку 2.6, DDS дає змогу використовувати всі інтелектуальні алгоритми робототехнічних систем у віртуальному режимі сцени для підключення до тієї самої мережі публікації/підписки. У спільному просторі даних мережі кожну тему можна розглядати як канал зв'язку, так і вузли пов'язані з темою в мережі підпишуться на тему, відповідно каналу і за потреби публікувати в ньому спільну інформацію. Процес публікації такий: спочатку кодується спільна інформація відповідно до стандартизованого формату JSON, потім серіалізується в рядок і, нарешті, публікується у темі. Це надає можливість перетворити мережу DDS, орієнтовану на дані, на мережу спільної роботи, орієнтовану на завдання з використанням вищевказаних механізмів. Упакований формат JSON дозволяє зменьшити потоки даних, що пересилаються між вузлами та збільшити швидкість синхронізації вузлів.

Тема організується за допомогою черги FIFO або черги з довільною вибіркою. У першому випадку вбір публікації виконується підсистемою управління, у другому – можливадовільна вибірка будь-яким вузлом за запитом.

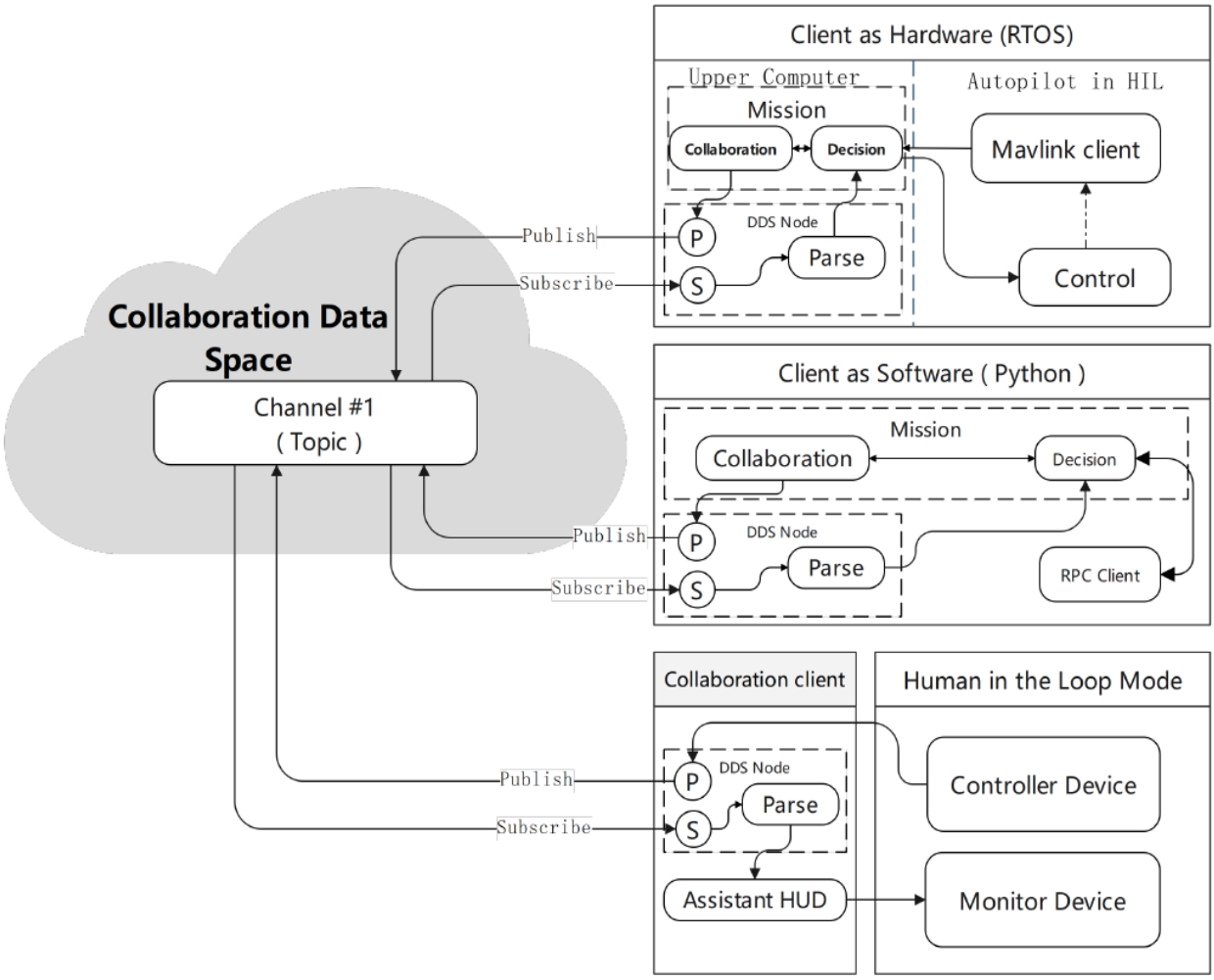

Рисунок 2.6 – Архітектура сервісної мережі спільного зв'язку

У роботі наведено процес протоколу взаємодії на основі наведеного вище механізму зв’язку. Спільне завдання представлено діаграмою діяльності (рисунок 2.7) між двух сутностей, які запитують і відповідають разом. Публічна мережа відноситься до каналу, на який будуть підписуватися всі інтелектуальні системи, головним чином для запитів і відповідей на спільні завдання. Коли запитувач і відповідач завдання збігаються, створюється спеціальний канал для безперервного спільного спілкування.

У деяких особливих випадках може не знадобитися формувати окремий тематичний канал. Коли вимоги до координації узгоджені та підтверджені, параметри завдання будуть достатньою інформацією, щоб керувати j-ю кооперативною системою для виконання відповідної операції взаємодії.

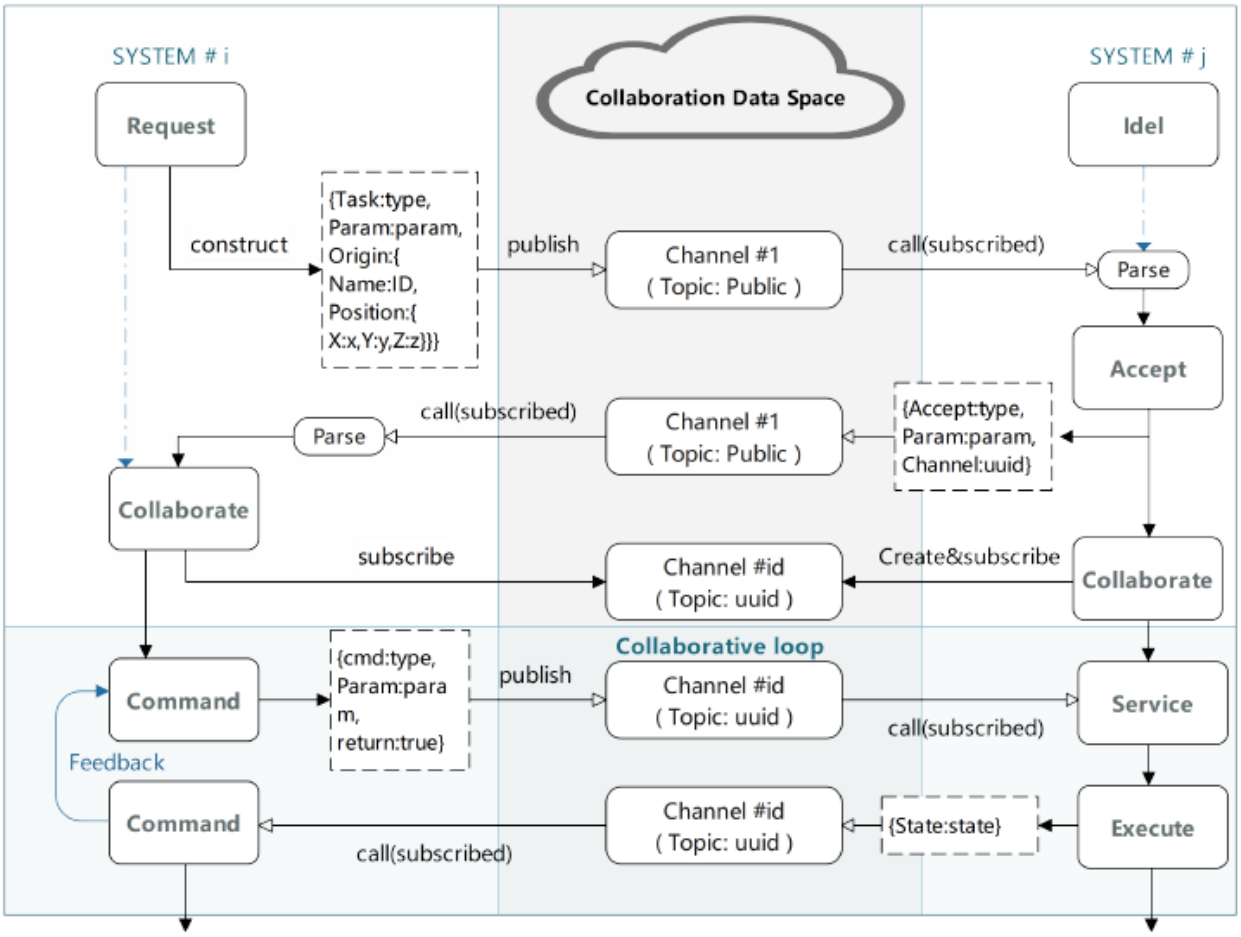

Рисунок 2.7 – Діаграма діяльності для процесу розподіленої співпраці

3 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ

3.1 Опис методу організації експериментів

Для вирішення вищезазначених проблем пропонується середовище моделювання, яке реалізує метод взаємодії розподілених завдань для різнорідних підмереж Інтернету речей (рисунок 3.1).

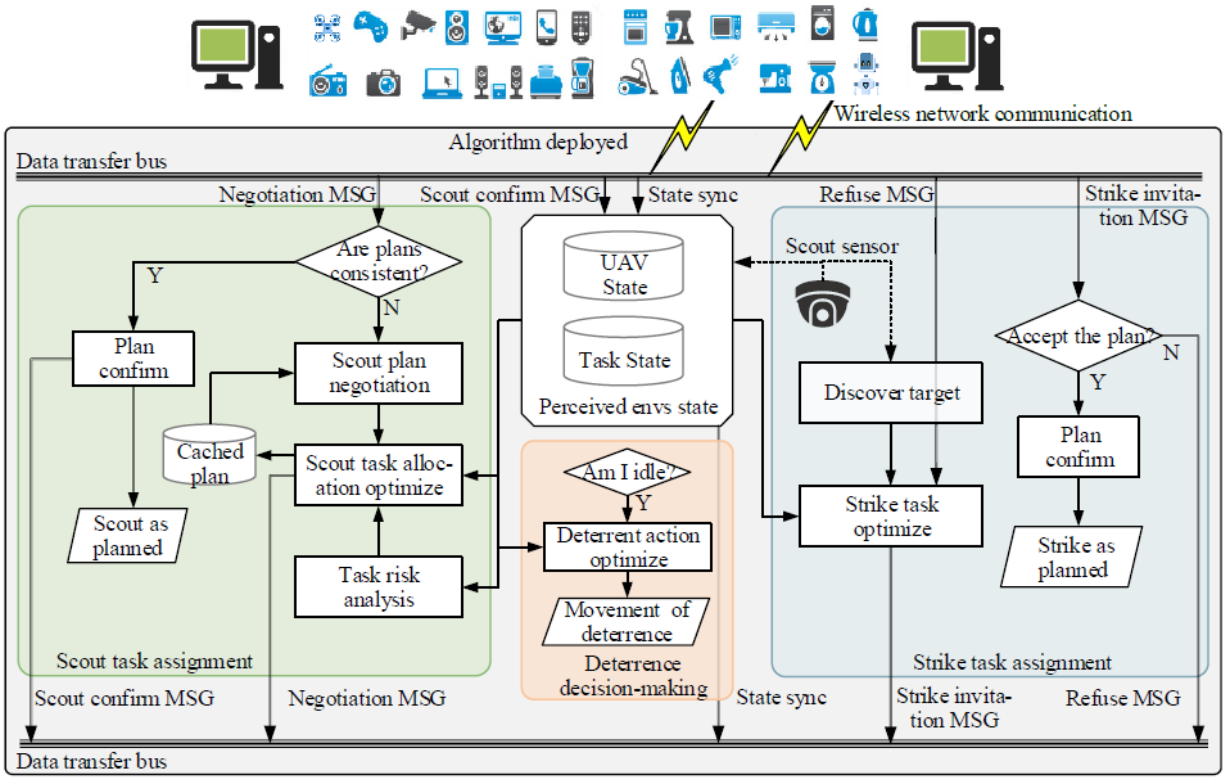

Рисунок 3.1 – Структура алгоритму спільного розподілу для завдань

Цей метод складається з трьох основних модулів: кроки щодо призначення завдання, виконання завдання та прийняття рішень щодо результатів виконання завдання. Під час призначення завдань, цей метод спочатку оцінює параметри завдань на наявних ресурсів і ставить завдання з метою мінімізації вартості і часу виконання завдання. На основі прийнятої інформації про стан середовища та історичну інформацію про статус ресурсів,

отриману шляхом спілкування з сусідніми вузлами, кожен пристрій використовує GA для створення локального плану розподілу завдань і координації часу, аналізує пріоритетність кожного завдання та веде переговори з сусідами для вирішення конфліктів.

Після цього завдання передається на відповідний вузол локально оптимізується виконання завдання та синхронізація його з іншими завданнями для координації виконання. Якщо запит буде відхилено, це приведе до повторної оптимізації плану розподілу завдань за ресурсами, доки завдання не будуть успішно призначені. При виникненні вузлів, які простоюють, виконується балансування навантажень з перерозподілом ресурсів.

3.2 Організація тестування продуктивності процесу моделювання

На основі запропонованої трирівневої розподіленої архітектури моделювання, процес тестування продуктивності можна умовно ділиться на чотири етапи.

Етап 1. Проектування віртуальної сцени, яка визначає систему координат розташування датчиків та приборів у сцені, розміщення їх у будівлі і встановлення початкових положень приборів, які можуть пересуватися.

Етап 2. Підключення обчислювальної платформи до мережі та виділення серверу і клієнтів, а також встановлення відповідних аватарів з їх спільних завдань.

Етап 3. Запуск служб моделювання та одночасно встановлення кадрів мережевого потоку з покращеним мережевим профайлером.

Етап 4. Аналіз служб розподілених даних, перевірка та порівняння з теоретичною оцінкою.

При реалізації експериментів будь-яка сутність або логічний об’єкт буде створено як підклас вузла. Тому вузли будуть використовуватися як загальний термін при аналізі та опису результатів.

3.3 Програмне забезпечення аналізу взаємодії вузлів

У цьому підрозділі наведено результати дослідження додаткового програмного забезпечення (рисунок 3.2): новий модуль функцій, який дозволяє користувачам виводити інформацію про мережевий трафік і продуктивність обміну пакетами у файли JSON. Хоча це вдосконалення не змінює використання системи моделювання, воно значно спрощує подальшу обробку даних, аналіз та робить оптимізацію більш ефективною.

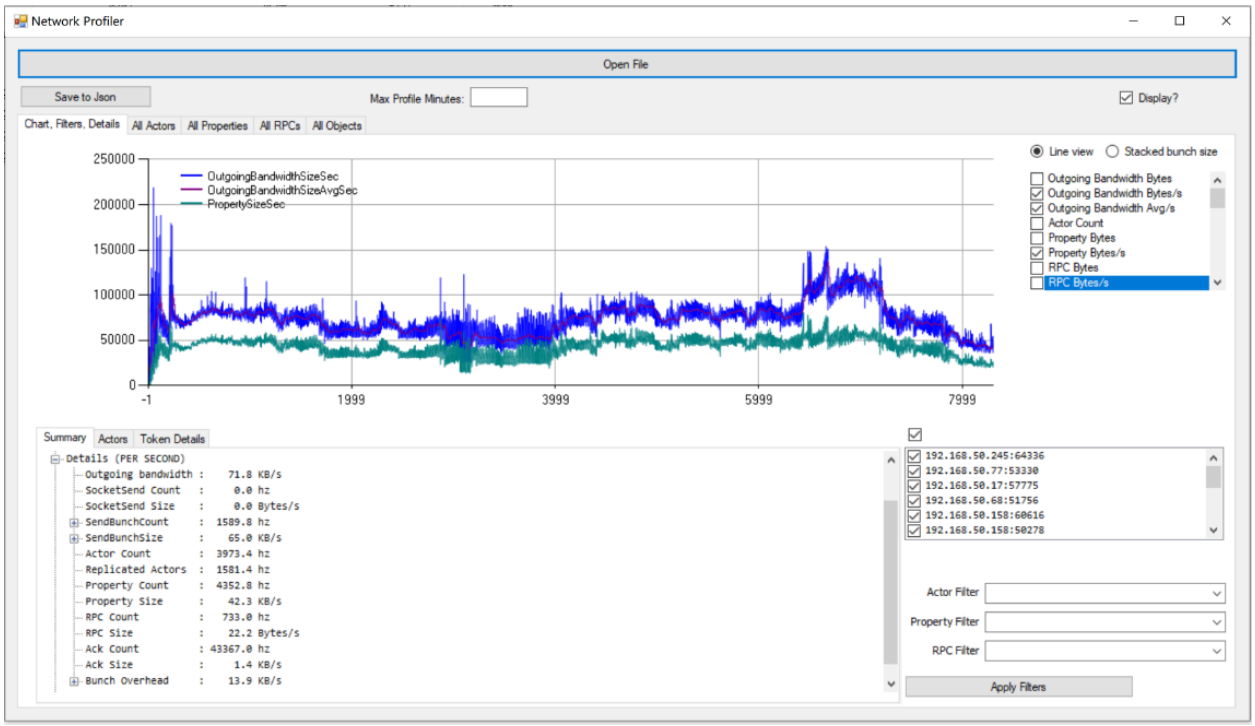

Рисунок 3.2 – Інтерфейс Network Profiler для аналізу даних

Network Profiler — це незалежний інструмент, призначений для відображення мережевого трафіку та інформація щодо продуктивності, якау отримує система управління движком симулятора під час виконання моделювання. Цей засіб високо ефективний у виявленні областей з високим споживанням пропускної здатності в багатокористувацьких мережах. Він дозволяє користувачам переглянути відсоток пропускної здатності, зайнятий

різними учасниками системи та знайти атрибути, що сприяють оптимізації продуктивності мережі.

Перед використанням Network Profiler користувачі повинні записати відповідні дані для аналізу. Записування можна досягти, як правило, увімкнувши функцію відстеження процесів двигуна та скомпілювавши механізм у збірку для налагодження або використовуючи збірку для неналагоджувальних конфігурацій.

Крім того, користувачі можуть керувати записом мережевих даних, додавши параметр командного рядка “networkprofiler=true” під час запуску двигуна або за допомогою іншої команди операційної системи.

Записані дані будуть збережені за вказаним шляхом, що дозволить користувачам відкрити файл в автономному інструменті для подальшого аналізу. Важливо зазначити, що тимчасові файли будуть бути перейменовано за певною схемою після завершення сеансу аналізу, полегшуючи точне відстеження та обробка даних.

3.4 Результати моделювання

З попереднього розділу, описана структура забезпечує сеанс віртуального моделювання управління в архітектурі C/S, базову логічну структуру сприйняття та управління, а також механізм співпраці з гнучкою конфігурацією. Порівняння основних особливостей з існуючим програмним забезпеченням приведено у таблиці 3.1. Було порівно програмне забезпечення для моделювання взаємодії кількох приборів. Основним моментом даної роботи є можливість керувати синхронізацією стану і обчисленнями для кількох клієнтів за допомогою сеансового механізму. Він враховує та порівнює характеристики синхронізації об'єктів, характеристики пропускної здатності в часовій області та зв'язок між пропускною здатністю та кількістю клієнтів у комбінації з даними експериментальних вимірювань.

Таблиця 3.1 – Конфігурація експериментальної системи

Програма	Рік оновлення	Архітектура C/S	Система управління	Механізм синхронізації	Управління фреймом
USARSim	2013	-	+	-	-
Stage	2020	+	+	+	-
Webots	2021	+	+	+	-
ARGoS	2022	+	+	+	-
CoppeliaSim	2024	+	+	+	-
Наше рішення	2025	+	+	+	+

У цій частині експерименту в основному вивчаються фактичні частоти реплікації використання акторів і час реплікації під час синхронізації. Було встановлено один або два прилади для кожного з трьох клієнтів, які можуть відстежувати параметри акторів, також можуть бути задіяними у фреймі. Вузли, які отримали інструкцію для виконання, починають її виконувати.

Результати візуалізації даних наведено на рисунку 3.3. Графік фіксує розподіл частоти виконання дій кожного вузла, який ініціює загальну зміну даних моделювання на початку, під час процесу та в кінці моделювання. Оскільки початковий стан відповідає режим сну кожного прибору, частота оновлення на початку є повільною (лише 10 Гц). У випадку виконання завдання, прибори починають активну роботу, що призводить до зміні параметрів, що контролюються. Тому частота оновлення збільшується (більше 75 Гц). Пасивні прибори залишалися на місці або не змінювали параметри середовища, тому медіана була дуже близькою до верхнього квартиля (близько 50 Гц). Крім того, сортування імен об'єктів було виконано від найменшого до найбільшого обсягу даних, синхронізованих у всьому процесі моделювання.

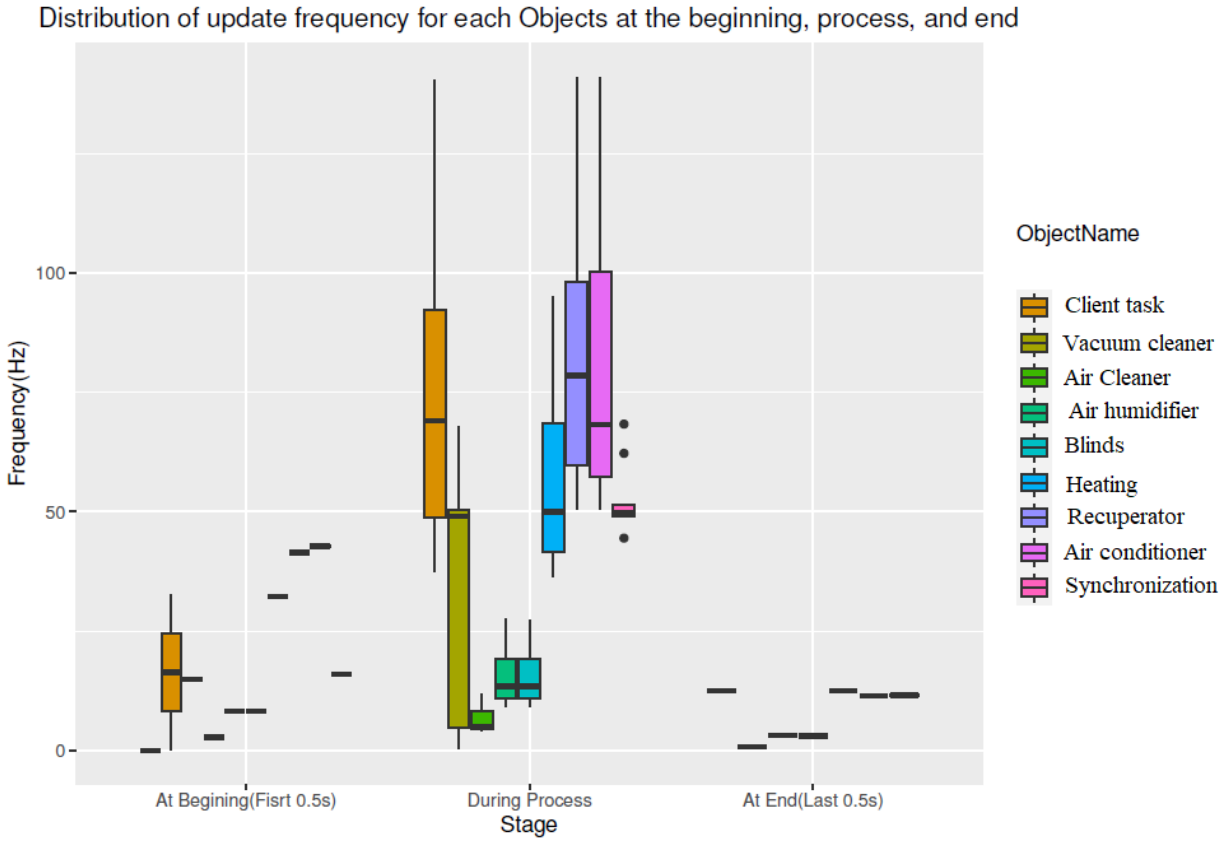

Рисунок 3.3 – Частота використання для кожного вузла протягом початка, процесу та в кінці

Для отриманих даних того самого експерименту також було проаналізовано час реплікації розподілу, коли різні стани акторів синхронно керувалися реплікацією на віддалених клієнтах, а результати візуалізації показані на графіку (рисунок 3.4). Актори що представляють об’єкти руху, мали менший час реплікації, оскільки їхній стан було описано відносно одиничними інтервалами роботи. Вони зазвичай завершали реплікацію станів протягом 0,03 мілісекунди, що може відповідати інтервалу часу, дозволеного для кадрової синхронізації. Піки в розподілі відповідають приблизної кількості незалежно оновлених станів або подій для цього актора.

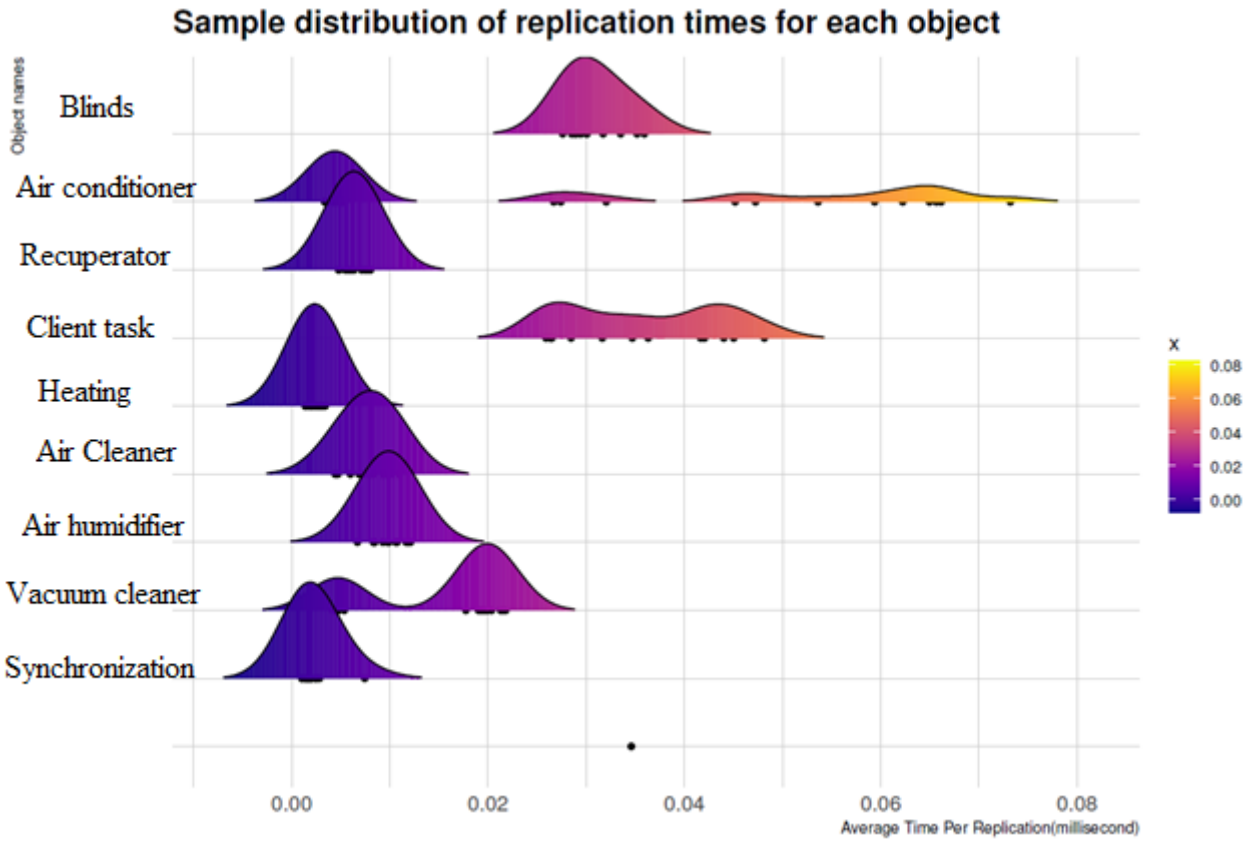

Рисунок 3.4 – Синхронізований графік часу реплікації різних станів вузлів

У наступному експерименті метою було перевірити пропускну здатність і загальну кількість даних. Пропускна здатність якомога більшої кількості клієнтів проводилася на основі наявних лабораторних ресурсів і перевірки кореляція між кадрами даних і даними стану. Було створено сервер (Intel i9 з 32 ГБ пам’яті) і запрошено до участі 12 людей. Вони надали свої ноутбуки у якості клієнтів (моделі та конфігурації комп’ютерів були різноманітними).

Було визначено спільні завдання, і нарешті сформовано 15 розподілених клієнтських вузлів для виконання експериментального розподіленого моделювання. Тривалість моделювання становила менше 240с. Рисунок 3.5 надає інтуїтивно зрозумілу картину смуги пропускання, яку займає кадр даних, де пакет спочатку використовується для завершення конфігураційної роботи, необхідної для ініціалізації на розподілену мережу. Моноліт кадру даних був значним, але частота синхронізації була відносно низькою. У проміжному процесі кадр даних в основному використовувався для синхронізації динамічних властивостей актора між різними клієнтами, тому

його моноліт був невеликим, але оновлювався відносно часто. Наприкінці моделювання, кількість об'єктів було зменшено, а споживання пропускної здатності, частота кадрів синхронізованих даних також були знижені. Наведена вище ситуація відповідає характеристикам етапу, показаних на рисунку 3.5.

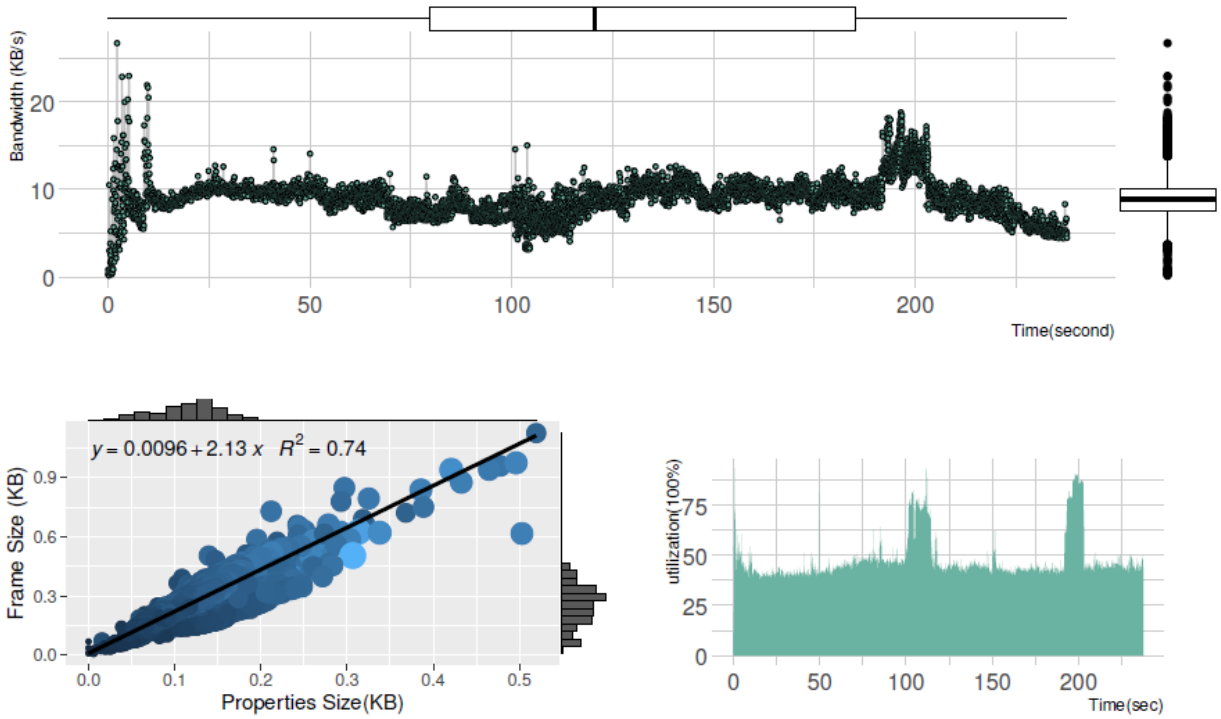

Рисунок 3.5 – Розподіл пропускної здатності в часовій області, кореляція розміру між кадру і даних, часовий розподіл даних атрибутів щодо пропускної здатності

Крім того, середня смуга пропускання, яку займають кадри даних становила 8 КБ/с, а максимальне використання пропускної здатності становило близько 25 КБ/с. Результати на рисунку 3.5 показують позитивну кореляцію між розміром кадру даних і кількістю атрибутивних даних у кадрі. Тому ми можемо використовувати приблизну оцінку кількості даних, які потрібно синхронізувати. Крім того, результати на рисунку 3.5 можуть бути перевірені двонаправлено за смугою пропускання. Синхронізація атрибутів коливалася нижче 50%, але іноді перевищувала 87,5% протягом усього часу

моделювання, що підтверджує коефіцієнт 2,13, отриманий шляхом лінійної підгонки.

На основі кількох експериментів проаналізовано зв’язок між фізичним масштабом об’єкта робототехнічної системи (еквівалентний масштабу клієнтського вузла розподіленого мережа віртуального моделювання в цій структурі) і пропускною здатність зв’язку в мережі.

У процесі аналізу даних, оскільки теоретичне значення максимальної пропускної здатності майже неможливо через існування стратегії синхронізації, розподіл пропускної здатності має конкретні характеристики середньої точки. Було обрано середнє значення пропускної здатності як точку вибірки для отримання співвідношення розподілу пропускної здатності зі шкалою клієнтів і загальною кількістю значень атрибутів.

У пропонованому методі оптимізації розподіл завдань і координація часу між вузлами, що пов'язані з високими обчислювальними ресурсами, визначає складність запропонованого нами методу. Тому, аналіз обчислювальної складності завдання сприятиме в подальшому вдосконаленню методу.

В алгоритмі розподілення ресурсів, самим трудомістким процесом є використання плану оптимізації за допомогою GA. Обчислювальну складність GA можна виразити через розмір популяції та кількість ітерацій. Для кожного плану, створеного GA, його придатність буде розраховано за допомогою запропонованого алгоритму. Найгірший випадок циклу алгоритму буде повторювати N разів, і для кожного циклу потрібно обчислити $N^* (N-1)$ прибутків винагороди за вирівнювання часу, тому складність алгоритму становить приблизно $O(N^3)$. Потім відповідно алгоритму розраховується придатність плану розподілу, в якому час початку між будь-якими подіями потрібно порівняти, щоб оцінити обчислювальну складність завдання. Отже, складність приблизно $O(N^2)$.

Таким чином, загальна обчислювальна складність алгоритму становить приблизно $O(N+N^3 + N^2)$. Серед них враховується налаштування, які можуть

вплинути на вартість розрахунку, а також вплинути на якість результатів оптимізації. У роботі ці два параметри просто лінійно відображаються з числа розподілу комбінацій, і перед розгортанням додатково вивчаються стратегію налаштування. Ці параметри важливі для досягнення компромісу між вартістю розрахунку та якістю оптимізації.

Через ймовірність високої обчислювальної складності при виконанні завдань за умови недостатньої інформації про стан фізичної системи, пропонується розподілений метод спільного розподілу завдань для гетерогенних роїв вузлів. Цей метод встановлює структуру розподілу завдань, що складається з метод розподілу завдань збору даних з датчиків, заснований на механізмі переговорів і розподілі завдань обчислень – методі, заснованого на механізмі запрошення.

Завдання-розподіл оцінює пріоритетність завдання відповідно до переваги пристроїв над задачею зменшення складності оптимізаційного алгоритму. Інформація з датчиків приймається стратегію координації часу для виконання аналізу стану системи та запуску штучного інтелекту для вирішення стратегії управління приборами. Цей метод дозволяє групі вузлів узгодити розподіл завдань у розподіленій структурі, і в той же час оцінює переваги кожної схеми розподілу, виконуючи координацію часу та механізмів. Розподілена структура не тільки покращує масштабованість рою, але також підвищує його надійність з більш складним електромагнітним середовищем.

ВИСНОВКИ

У кваліфікаційній роботі досліджено програмне забезпечення для моделювання взаємодії множини датчиків та пристроїв Інтернету речей в гетерогенному хмарному середовищі. Запропоновано механізм для керування синхронізацією стану, обчисленнями та зв’язку між множиною клієнтів та пристроїв [44]. Аналізуючи дані експериментальних досліджень, можна зробити наступні висновки.

Досліджено характеристики часу реплікації та характеристики частоти оновлення станів моделей під час синхронізації розподіленої симуляції. Було встановлено, що в запропонованій системі час реплікації та синхронізації об’єктів є відносно низьким, і частота оновлення може відповідати потребам співпраці кількох вузлів, що забезпечують використання в реальному часі та точність моделювання системі.

Проаналізована пропускна здатності кадрів даних протягом сеансу (фрейму) моделювання. Результати експериментів показують, що серверна сторона зайняла майже половину пропускної здатності даних за весь час сесії, що вказує на раціональний розподіл і використання передачі даних у запропонованій системі.

Запропоновано поверхневу модель оцінки пропускної здатності для оцінки вимог до пропускної здатності поточної моделі під час масштабування на стороні сервера та синхронізації. Ця модель дає важливу інформацію для кращого планування та оптимізації розподілу ресурсів і продуктивності системи.

У майбутніх дослідженнях можливо покращити наступні ключові технічні моменти щодо запропонованих рішень та структури системи розподіленого моделювання.

Хоча метод оновлення динамічної моделі пов’язаних об’єктів наведено в роботі, все ще потрібні зусилля для подальшого вдосконалення методу для підтримки моделювання на основі теорії системного зв'язку між об'єктами.

Синхронне управління, прийняте в цій архітектурі, вже частково покращило рішення проблеми затримки зв'язку, але все ще необхідно вивчити вплив просторово-часової проблеми узгодженості розподіленої системи, багаторівневого контролю та прийняття рішень;

Мережа спільної роботи за замовчуванням у цьому дослідженні – це проста комунікація на основі механізму подій. Він лише інтегрує служби даних, проектує та впроваджує базову структуру співпраці, і не враховує її вплив на загальну пропускну здатність розподіленої мережі зв'язку. Тому можливе поглиблене вивчення у майбутньому цього механізму у поєднанні з конкретними завданнями спільної роботи гетерогенної системи Інтернету речей.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Qin, B.; Zhang, D.; Tang, S.;Wang, M. Distributed Grouping Cooperative Dynamic Task Assignment Method of UAV Swarm. Appl. Sci. 2022, 12, 2865.

2. Zhang, J.; Xing, J. Cooperative task assignment of multi-UAV system. Chin. J. Aeronaut. 2020, 33, 2825–2827.

3. Jiang, X.; Zeng, X.; Sun, J.; Chen, J. Research status and prospect of distributed optimization for multiple aircraft. Acta Astronaut. 2021, 42, 524551. https://doi.org/10.7527/S1000-6893.2020.24551. (In Chinese)

4. Zhen, Z.; Xing, D.; Gao, C. Cooperative search-attack mission planning for multi-UAV based on intelligent self-organized algorithm. Aerosp. Sci. Technol. 2018, 76, 402–411.

5. Duan, H.; Zhao, J.; Deng, Y.; Shi, Y.; Ding, X. Dynamic Discrete Pigeon-Inspired Optimization for Multi-UAV Cooperative Search-Attack Mission Planning. IEEE Trans. Aerosp. Electron. Syst. 2021, 57, 706–720.

6. Ma, Y.; Zhao, Y.; Bai, S.; Yang, J.; Zhang, Y. Collaborative task allocation of heterogeneous multi-UAV based on improved CBGA algorithm. In Proceedings of the 16th International Conference on Control, Automation, Robotics and Vision, Shenzhen, China, 13–15 December 2020; pp. 795–800.

7. Dai, W.; Lu, H.; Xiao, J.; Zeng, Z.; Zheng, Z. Multi-Robot Dynamic Task Allocation for Exploration and Destruction. J. Intell. Robot. Syst. 2019, 98, 455–479.

8. Sheng, W.; Yang, Q.; Tan, J.; Xi, N. Distributed multi-robot coordination in area exploration. Robot. Auton. Syst. 2006, 54, 945–955.

9. Ye, F.; Chen, J.; Sun, Q.; Tian, Y.; Jiang, T. Decentralized task allocation for heterogeneous multi-UAV system with task coupling constraints. J. Supercomput. 2020, 77, 111–132.

10. Chen, J.;Wu, Q.; Xu, Y.; Qi, N.; Guan, X.; Zhang, Y.; Xue, Z. Joint Task Assignment and Spectrum Allocation in Heterogeneous UAV Communication

Networks: A Coalition Formation Game-Theoretic Approach. IEEE Trans. Wirel. Commun. 2021, 20, 440–452.

11. Jiang, Y. A Survey of Task Allocation and Load Balancing in Distributed Systems. IEEE Trans. Parallel Distrib. Syst. 2016, 27, 585–599.

12. Shakhatreh, H.; Sawalmeh, A.H.; Al-Fuqaha, A.; Dou, Z.; Almaita, E.; Khalil, I.; Othman, N.S.; Khreishah, A.; Guizani, M. Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges. IEEE Access 2019, 7, 48572–48634.

13. Rizk, Y.; Awad, M.; Tunstel, E.W. Cooperative Heterogeneous Multi-Robot Systems: A Survey. ACM Comput. Surv. 2019, 52, 29.

14. Xiang, J.; Dong, X.; Ding,W.; Suo, J.; Shen, L.; Xia, H. Key technologies for autonomous cooperation of unmanned swarm systems in complex environments. Acta Aeronaut. Astronaut. Sin. 2022, 43, 527570.

15. Tubis, A.A.; Poturaj, H.; Deren´ , K.; Z˙ urek, A. Risks of Drone Use in Light of Literature Studies. Sensors 2024, 24, 1205.

16. Ding, Y.; Xiong, Z.; Xiong, J.; Cui, Y.; Cao, Z. OGI-SLAM2: A hybrid map SLAM framework grounded in inertial-based SLAM. IEEE Trans. Instrum. Meas. 2022, 71, 2519014.

17. Xun, Z.; Huang, J.; Li, Z.; Xu, C.; Gao, F.; Cao, Y. CREPES: Cooperative RElative Pose EStimation towards Real-World Multi-Robot Systems. arXiv 2023, arXiv:2302.01036.

18. Wang, J.; Wu, Y.; Chen, Y.; Ju, S. Multi-UAVs collaborative tracking of moving target with maximized visibility in Urban Environment. J. Frankl. Inst. 2022, 359, 5512–5532.

19. Wang, Z.; Xu, C.; Gao, F. Robust Trajectory Planning for Spatial-Temporal Multi-Drone Coordination in Large Scenes. In Proceedings of the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 23–27 October 2021; pp. 12182–12188.

20. Raziei, Z.; Moghaddam, M. Adaptable automation with modular deep reinforcement learning and policy transfer. Eng. Appl. Artif. Intell. 2021, 103,

104296.

21. Badrno, H.; Baradarannia, M.; Bagheri, P.; Badamchizadeh, M.A. Distributed Predictive Consensus Control of Uncertain Linear Multi-agent Systems with Heterogeneous Dynamics. Iran. J. Sci. Technol. Trans. Electrical Eng. 2022, 47, 255–267.

22. Li,W.; Zhang, H.; Gao, Z.;Wang, Y.; Sun, J. Fully Distributed Event/Self-Triggered Bipartite Output Formation-Containment Tracking Control for Heterogeneous Multiagent Systems. IEEE Trans. Neural Netw. Learn. Syst. 2022, 34, 7851–7860.

23. Yang, S.M.; Kim, K. Implementation of the conversation scheme in message-based distributed computer systems. IEEE Trans. Parallel Distrib. Syst. 1992, 3, 555–572.

24. Xu, J.; Romanovsky, A.; Randell, B. Concurrent exception handling and resolution in distributed object systems. IEEE Trans. Parallel Distrib. Syst. 2000, 11, 1019–1032.

25. Chen, J.; Huang, L. Supporting Dynamic Service Updates in Pervasive Applications. In Proceedings of the 2011 IEEE 35th Annual Computer Software and Applications Conference, Munich, Germany, 18–22 July 2011; pp. 273–278.

26. Calderón-Arce, C.; Brenes-Torres, J.C.; Solis-Ortega, R. Swarm Robotics: Simulators, Platforms and Applications Review. Computation 2022, 10, 80.

27. Cho, W.J.; Kim, S.; Kim, Y.; Moon, Y.H. Advanced Co-Simulation Platform for UAV Simulations under Virtual Wireless Network Environments. IEEE Access 2022, 10, 95498–95508.

28. Phadke, A.; Medrano, F.A.; Sekharan, C.N.; Chu, T. Designing UAV Swarm Experiments: A Simulator Selection and Experiment Design Process. Sensors 2023, 23, 7359.

29. Koç, D.; Seçkin, A.Ç.; Satı, Z.E. Evaluation of Participant Success in Gamified Drone Training Simulator Using Brain Signals and Key Logs. Brain Sci. 2021, 11, 1024.

30. Covaciu, F.; Iordan, A.-E. Control of a Drone in Virtual Reality Using MEMS Sensor Technology and Machine Learning. Micromachines 2022, 13, 521.

31. Li, L.; Xu, S.; Nie, H.; Mao, Y.; Yu, S. Collaborative Target Search Algorithm for UAV Based on Chaotic Disturbance Pigeon-Inspired Optimization. Appl. Sci. 2021, 11, 7358.

32. Hu, J.;Wu, H.; Zhan, R.; Menassel, R.; Zhou, X. Self-organized search-attack mission planning for UAV swarm based on wolf pack hunting behavior. J. Syst. Eng. Electron. 2021, 32, 1463–1476.

33. Choi, H.; Brunet, L.; How, J.P. Consensus-Based Decentralized Auctions for Robust Task Allocation. IEEE Trans. Robot. 2009, 25, 912–926.

34. Edalat, N.; Tham, C.; Xiao, W. An auction-based strategy for distributed task allocation in wireless sensor networks. Comput. Commun. 2012, 35, 916–928.

35. Patel, R.; Rudnick-Cohen, E.; Azarm, S.; Otte, M.; Xu, H.; Herrmann, J.W. Decentralized Task Allocation in Multi-Agent Systems

36. Using a Decentralized Genetic Algorithm. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 3770–3776.

37. Makoviychuk, V.;Wawrzyniak, L.; Guo, Y.; Lu, M.; Storey, K.; Macklin, M.; Hoeller, D.; Rudin, N.; Allshire, A.; Handa, A.; et al. Isaac Gym: High Performance GPU-Based Physics Simulation for Robot Learning. arXiv 2021, arXiv:2108.10470.

38. S. N. Han, G. M. Lee, N. Crespi, N. Van Luong, K. Heo, M. Brut, and P. Gatellier, “Dpwsim: A simulation toolkit for iot applications using devices profile for web services,” in Internet of Things (WF-IoT), 2014 IEEE World Forum on. IEEE, 2014, pp. 544–547.

39. S. Sotiriadis, N. Bessis, E. Asimakopoulou, and N. Mustafee, “Towards simulating the internet of things,” in Advanced Information Networking and Applications Workshops (WAINA), 2014 28th International Conference on. IEEE, 2014, pp. 444–448.

40. C. Sonmez, A. Ozgovde, and C. Ersoy, “Edgecloudsim: An environment

for performance evaluation of edge computing systems," in Fog and Mobile Edge Computing (FMEC), 2017 Second International Conference on. IEEE, 2017, pp. 39–44.

41. T. Pflanzner, A. Kert´esz, B. Spinnewyn, and S. Latr´e, "Mobiotsim: towards a mobile iot device simulator," in 2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (Fi- CloudW). IEEE, 2016, pp. 21–27.

42. A. M. Khan, L. Navarro, L. Sharifi, and L. Veiga, "Clouds of small things: Provisioning infrastructure-as-a-service from within community networks," in Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on. IEEE, 2013, pp. 16–21.

43. "Google cloud platform," https://cloud.google.com/solutions/iot/, accessed: 2018-08-10.

44. Волк М.О., Бугрій А.М., Самойлов І.А., Фурманов А.А., Журавльов О.Ю., Волк Д.М. Симуляція та управління туманними та хмарними обчисленнями для інтернету речей. Вісник Херсонського національного технічного університету. No 3(90), 2024 с. 215-220. DOI: https://doi.org/10.35546/kntu2078-4481.2024.3.27