МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет
імені Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ
Завідувач кафедри інтелектуальних
інформаційних систем, д-р техн. наук, проф.
_____Ю. П. Кондратенко
«____»_____2024 р.

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

БАГАТОКРИТЕРІАЛЬНА СИСТЕМА ФОРМУВАННЯ ЗБАЛАНСОВАНИХ КОМАНД В ІГРОВОМУ СИМУЛЯТОРІ ARMA 3

Спеціальність 122 «Комп'ютерні науки»

122 – КРБ – 401.22010303

Виконав студент 4-го курсу, групи 401
_____ І. В. Горшколєпов
«17» червня 2024 р.

Керівник: канд. техн. наук, доцент
_____Є. В. Сіденко
«17» червня 2024 р.

Миколаїв – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет ім. Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

Рівень вищої освіти **бакалавр**
Спеціальність **122 «Комп'ютерні науки»**
(шифр і назва)
Галузь знань **12 «Інформаційні технології»**
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри інтелектуальних інформаційних систем, д-р техн. наук, проф.
_____ Ю. П. Кондратенко
«____» _____2024 р.

З А В Д А Н Н Я
на виконання кваліфікаційної роботи

Видано студенту групи 401 факультету комп'ютерних наук Горшколєпову Іллі Віталійовичу.

1. Тема кваліфікаційної роботи «Багатокритеріальна система формування збалансованих команд в ігровому симуляторі Arma 3».

Керівник роботи Сіденко Євген Вікторович, канд. техн. наук, доцент.

Затв. наказом Ректора ЧНУ ім. Петра Могили від «28» грудня 2023 р. № 271

2. Строк представлення кваліфікаційної роботи студентом «19» червня 2024 р.

3. Вхідні (початкові) дані до роботи: багатокритеріальні методи прийняття рішень; характеристика бійців; найменування підрозділів; платформа для тестування системи.

Очікуваний результат: ефективна система формування збалансованих команд, що інтрегрована у Discord бота.

4. Перелік питань, що підлягають розробці (зміст пояснювальної записки):

– аналіз сучасного стану військових симуляторів та методів формування команд;

- підбір алгоритмів та методів аналізу навичок гравців;
- методологія та інструменти розробки;
- програмування та тестування системи.

5. Перелік графічного матеріалу: презентація.

6. Консультанти розділів роботи:

Розділ	Прізвище, ініціали та посада консультанта	Підпис
Спеціальна частина з охорони праці	Алексєєва А. О., доцент кафедри екології	

Керівник роботи __канд. тех. наук, доцент Сіденко Є. В._____
(наук. ступінь, вчене звання, прізвище та ініціали)

_____
(підпис)

Завдання прийнято до виконання ___Горшколєпов І. В._____
(прізвище та ініціали)

_____
(підпис)

Дата видачі завдання «_14_» _____січня____ 2024 р.

КАЛЕНДАРНИЙ ПЛАН
виконання кваліфікаційної роботи

Тема: «Багатокритеріальна система формування збалансованих команд в ігровому симуляторі Arma 3»

№	Найменування роботи	Початок	Закінчення	Примітки
1	Подання заяви на затвердження теми та керівників КРБ	10.11.2023	15.11.2023	
2	Отримання завдання на виконання КРБ	10.01.2024	15.01.2024	
3	Складання календарного плану роботи на весь період виконання КРБ	16.01.2024	30.01.2024	
4	Отримання завдання на переддипломну практику	15.04.2024	29.04.2024	
5	Проходження переддипломної практики, збір та аналіз матеріалів до КРБ	29.04.2024	11.05.2024	
6	Розробка звіту з переддипломної практики	12.05.2024	15.05.2024	
7	Виконання КРБ: аналіз сучасного стану військових симуляторів та методів формування команд, огляд існуючих технологій, розробка системи, розробка ПЗ	13.05.2024	22.06.2024	
8	Перший попередній захист КРБ на засіданні комісії кафедри	27.05.2024	27.05.2024	
9	Доробка та остаточне оформлення КРБ	28.05.2024	09.06.2024	
10	Другий попередній захист КРБ на засіданні комісії кафедри	10.06.2024	10.06.2024	
11	Подання КРБ рецензенту	13.06.2024	13.06.2024	
12	Подання КРБ, її електронної копії та інших документів (відгуку, рецензії) до захисту	17.06.2024	21.06.2024	
13	Захист КРБ перед екзаменаційною комісією (ЕК)	24.06.2024	28.06.2024	

Розробив студент _____Горшколєпов І. В._____ _____
(прізвище та ініціали) (*підпис*)

Керівник роботи канд. тех. наук, доцент Сіденко Є. В. _____
(наук. ступінь, вчене звання, прізвище та ініціали) *(підпис)*

«_29_» _____01_____ 2024 р.

АНОТАЦІЯ

кваліфікаційної роботи студента групи 401 ЧНУ ім. Петра Могили

Горшколєпова Іллі Віталійовича

Тема: «Багатокритеріальна система формування збалансованих команд в ігровому симуляторі Arma 3»

Кваліфікаційна робота бакалавра присвячена створенню багатокритеріальної системи формування збалансованих команд в ігровому симуляторі Arma 3.

Об’єкт роботи – ігрові процеси у військових симуляторах.

Предмет роботи – методи та підходи до планування і оптимізації командних структур у військових симуляторах.

Мета – формування збалансованих команд в ігровому симуляторі Arma 3 на основі аналізу навичок гравців та вимог підрозділів.

У першому розділі розглянуто сучасний стан військових симуляторів, та метдоди за якими формуються команди. У другому розділі аналізуються способи оцінювання навичок гравців, та існуючі аналітичні методи прийняття рішень . У третьому розділі описуються методології та інструменти, що використовувалися при побудові системи. В четвертому розділі описуються середовище тестування системи та її програмна реалізація, додатково аналізується ефективність підрозділів після використання системи.

В результаті виконаної роботи реалізовано багатокритеріальну систему формування збалансованих команд в ігровому симуляторі Arma 3 та були зроблені висновки щодо можливості використання подібних систем у військових установах.

КРБ викладена на 67 сторінки, вона містить 4 розділи, 43 ілюстрацій, 2 таблиці, 26 джерел в переліку посилань.

Ключові слова: військові симулятори, TOPSIS, характеристика, Discord бот, радар-діаграмма, багатокритеріальна система.

ABSTRACT

qualification work of a student of group 401 of BSNU named after Petro Mohyla Horshkoliepov Illia

"A multi-criteria system for forming balanced teams in the game simulator Arma 3"

Bachelor's thesis is dedicated to creating a multi-criteria system for forming balanced teams in the Arma 3 game simulator.

The object of the study is game processes in military simulators.

The subject of the study is methods and approaches to planning and optimizing team structures in military simulators.

The aim is to form balanced teams in the Arma 3 game simulator based on player skills analysis and unit requirements.

Chapter 1 examines the current state of military simulators and the methods used to form teams. Chapter 2 analyzes methods for evaluating player skills and existing decision-making analytical methods. Chapter 3 describes the methodologies and tools used in building the system. Chapter 4 describes the testing environment for the system and its software implementation, and additionally analyzes the efficiency of the units after using the system.

As a result of the study, a multi-criteria system for forming balanced teams in the Arma 3 game simulator was implemented, and conclusions were drawn regarding the potential use of similar systems in military institutions.

The thesis spans 67 pages, comprising 4 chapters, 43 illustrations, 2 tables, and 26 sources in the references list.

Keywords: military simulators, TOPSIS, characteristics, Discord bot, radar chart, multi-criteria system.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

TOPSIS – Technique for Order Preference by Similarity to Ideal Solution

SAW – Simple Additive Weighting

AHP – Analytic Hierarchy Process

MOORA – Multi-Objective Optimization by Ratio Analysis

IDE – Integrated development environments

ВСТУП

У сучасному світі зростає популярність глобальних військових симуляторів, таких як ARMA 3, які не лише розважають гравців, а й стають важливим інструментом для навчання та підготовки військових. Ці ігри відтворюють реалістичні віртуальні битви та дають можливість гравцям випробувати свої навички та стратегії у різних воєнних сценаріях.

У зв'язку з цим, ефективне формування та планування команд стає критично важливим завданням як для гравців, що бажають максимізувати свої шанси на успіх у віртуальних битвах, так і для самих військових, що використовують ці симулятори для навчання та тренувань.

Об’єктом роботи є ігрові процеси у військових симуляторах.

Предметом роботи є методи та підходи до планування і оптимізації командних структур у військових симуляторах.

Метою роботи є формування збалансованих команд в ігровому симуляторі Arma 3 на основі аналізу навичок гравців та вимог підрозділів. Основні сили будуть задіяні в розробці алгоритмів, які забезпечать балансування складу команд з урахуванням різноманітних критеріїв, таких як навички стрільби, тактика, комунікабельність, лідерство тощо.

В рамках дослідження будуть розглянуті та порівняні різні методи прийняття рішень, такі як TOPSIS, SAW, AHP тощо, з метою вибору найбільш ефективного методу для реалізації цієї системи.

Основними завданнями дослідження будуть:

– аналіз сучасного стану військових симуляторів та методів формування команд;

– підбір алгоритмів та методів аналізу навичок гравців;

– реалізація системи для формування збалансованих команд у військовому симуляторі Arma 3;

– тестування та валідація розробленої системи на реальних гравцях та віртуальних битвах.

Результати дослідження сприятимуть підвищенню ефективності та реалістичності віртуальних битв у військовому симуляторі Arma 3 та інших подібних іграх. Також це дозволить гравцям отримувати більше задоволення від участі у віртуальних бойових операціях та сприятиме подальшому розвитку військових симуляторів, а військовим дозволить більш ефективніше проводити тренування та покращувати професійні навички.

1 АНАЛІЗ СУЧАСНОГО СТАНУ ВІЙСЬКОВИХ СИМУЛЯТОРІВ ТА МЕТОДІВ ФОРМУВАННЯ КОМАНД

1.1 Військові симулятори. Їх призначення та приклади

Військові симулятори – це переважно комп’ютерні ігри, які модулюють військові сценарії та операції для навчання та розваг (див. рис. 1.1). Основна ідея таких ігор – наблизити ігровий процес максимально до реалістичності. Саме такий підхід приваблює не тільки звичайних гравців, а ще й військових.

Рисунок 1.1 – Військовий симулятор

Військові симулятори, через свою універсальність, реалістичність та свободу дій, дозволяють випробовувати реальні стратегії та тактики, покращувати професійні військові навички, а також робити це в розважливій формі.

1.1.1 ARMA 3

ARMA 3 це один з найреалістичніших військових симуляторів, розроблений чеською компанією Bohemia Interactive (див. рис. 1.2). Вона відзначається своєю надзвичайною деталізацією та можливостями, які вона пропонує для тактичного планування та ведення бойових дій.

ARMA 3 має вбудований редактор місій, який дозволяє гравцям створювати власні сценарії. Тобто кожен гравець може стати розробником, створюючи унікальні бойові ситуації, що відповідають його баченню та стратегіям. Від

простих патрулювань до складних багатоступеневих операцій – можливості обмежені лише уявою користувача.

Додатковою особливістю є широкий спектр зброї та техніки, взятої з реального світу. У гравців є можливість використовувати сучасне озброєння та спорядження, яке відображає актуальні військові стандарти, що забезпечує високий рівень занурення та реалістичності.

Що стосується фізики, то ARMA 3 пропонує реалістичні моделі поведінки об'єктів та персонажів. Від стрільби до руху техніки, все в грі відображає реальні фізичні закони, що робить геймплей не лише захопливим, але й навчальним, оскільки гравці можуть дізнатися про реальні військові стратегії, тактики, особливості.

Рисунок 1.2 – Військовий симулятор ARMA 3

Необхідно також зазначити, що розробку ARMA 3 спонсорувала армія США. Це надало змогу наблизити гру до рівня справжнього військового симулятора. Завдяки цьому ARMA 3 часто використовується для навчання та тренувань військових підрозділів у реальних умовах.

Проте, саме через свою складність та реалістичність, ARMA 3 може бути викликом для нових гравців. Гра вимагає серйозного підходу та знань у військовій справі, що робить її більш привабливою для дорослих користувачів, знайомих з військовою справою.

1.1.2 Squad

Squad – це багатокористувацький тактичний шутер, розроблений студією Offworld Industries, який ставить акцент на командній взаємодії та реалістичному зображенні сучасних бойових дій (див. рис. 1.3). На відміну від ARMA 3, Squad більше орієнтований на піхотні бої на менших за розміром мапах.

Особливості гри в тому, що вона більш динамічна та менш вимоглива до знання військових аспектів та механік самої гри, тому легше підходить для нових гравців. А графіка в Squad, хоча й менш деталізована порівняно з ARMA 3, все ж залишається реалістичною та привабливою. Вона пропонує гравцям реалістичні ландшафти та архітектуру, які створюють правдиве відчуття перебування на полі бою.

Рисунок 1.3 – Військовий симулятор Squad

Якщо ARMA 3 позиціонує себе, як більш серйозний симулятор для військових, де необхідно самому створювати для себе сценарій, то Squad можна вважати більш казуальною грою, що створена для ознайомлення звичайних гравців з реалістичністю бойових дій та військовою справою. Проте слід зазначити, що вона також чудово підходить для навчання тактиці та покращення професійних навичок.

1.1.3 DCS World

DCS World – це військовий симулятор повітряних боїв, розроблений компанією Eagle Dynamics (див. рис. 1.4). Гра має вражаючу деталізацію моделей літаків і вертольотів, а також точну реалізацію фізики боїв.

Однією з особливостей DCS World є можливість віртуального пілотування різноманітних літаків, від винищувачів до бомбардувальників, що реалістично моделюються з урахуванням їхньої поведінки та можливостей.

DCS World пропонує гравцям надзвичайно деталізовані моделі літаків, що включають в себе не лише зовнішній вигляд, але й внутрішню конструкцію кабін. Кожен перемикач, кнопка та індикатор мають своє реальне призначення та функціонують так само, як у справжніх літальних апаратах. Це робить гру унікальною для тих, хто бажає глибше зануритися у світ авіації.

Важливо зазначити, що кожен літак веде себе відповідно до реальних аеродинамічних законів, що створює відчуття справжнього пілотування. Це дозволяє навчитися керувати літаком у різних умовах, включаючи бойові ситуації та складні маневри при різних погодних умовах.

Рисунок 1.4 – Військовий симулятор DCS World

Що стосується доступності, то DCS World може виглядати складним для новачків через свою велику кількість деталей та вимог до вмінь пілотування. Однак

саме завдяки цим факторам, гра пропонує глибокий та насичений геймплей, який привертає до себе велику кількість любителів авіаційних симуляторів.

1.1.4 VBS3

VBS3 - це військовий симулятор, розроблений компанією Bohemia Interactive Simulations, який відомий своєю високою реалістичністю та використанням для тренування та симуляцій військових операцій (див. рис. 1.5).

Високодеталізовані моделі та картографія VBS3 дозволяють створювати реалістичні сценарії бойових дій, що точно відображають реальні умови. Це включає в себе точне відображення ландшафтів, будівель, транспортних засобів та інших об'єктів, що можуть бути присутніми на полі бою.

Симуляція в VBS3 має реалістичну фізику та поведінку об'єктів, що дозволяє створювати точні моделі бойових дій. Це дає змогу військовим використовувати VBS3 для відпрацювання тактичних прийомів, планування операцій та оцінки ефективності своїх дій у різних ситуаціях.

Крім того, VBS3 має вбудований редактор сценаріїв, що дозволяє користувачам створювати власні симуляції та адаптувати їх під конкретні потреби, що робить VBS3 гнучким інструментом, який може бути використаний для вирішення широкого спектру задач, від навчання військових до планування складних операцій.

Рисунок 1.5 – Військовий симулятор VBS3

VBS3 відрізняється від ARMA 3 тим, що ARMA 3 став комерційним проектом для великої аудиторії в тому числі і для звичайних гравців. Водночас,

VBS3 спрямований на використання військовими організаціями для навчання та тренувань. Це робить VBS3 більш спеціалізованим інструментом, який задовольняє потреби професійних військових. Додатково, в неї є можливість підключення функції віртуальної реальності, що покращує процес тренування.

Таким чином, VBS3 є потужним та багатофункціональним інструментом, який забезпечує реалістичну симуляцію бойових дій та підтримує військові тренування на високому рівні.

1.1.5 Ready or Not

Ready or Not - це тактичний шутер, що розроблений студією VOID Interactive. Гра надає гравцям можливість зануритися в реалістичні умови виконання місій, що передбачають втручання в надзвичайні ситуації, звільнення заручників та нейтралізацію загроз (див. рис. 1.6).

Однією з ключових особливостей Ready or Not є деталізована модель фізики, яка відображає реальні умови бойових дій. Додатково гравці мають доступ до великого вибору спеціального обладнання та зброї, що використовується поліцейськими силами. Це включає в себе різні види вогнепальної зброї, гранати, бронежилети, щити та інші засоби, необхідні для виконання тактичних операцій.

Ready or Not відрізняється від інших військових симуляторів тим, що зосереджена саме на тактиці проведення операцій поліцейським підрозділом. Гравці не зустрічають тут ворожої техніки або гравців-ворогів, як в ARMA 3 або Squad. Натомість гра пропонує інтерактивні місії, де головними опонентами є боти, які діють як злочинці.

Взаємодія з ботами в грі відбувається через тактичні прийоми, зокрема координовані дії команди, проникнення в приміщення та використання спеціального обладнання для нейтралізації загроз.

Рисунок 1.6 – Військовий симулятор Ready or Not

Ready or Not приваблює саме своїми механіками гри та реалістичним підходом до виконання поліцейських операцій. Гра вимагає від гравців ретельного планування, вміння працювати в команді та швидкого реагування на змінювані обставини. Це робить її ідеальною для тих, хто хоче випробувати себе у ролі поліцейського спецназу, виконуючи складні та небезпечні завдання в реалістичних умовах.

1.2 Методи формування команд

Формування збалансованих та ефективних команд є ключовим фактором успіху в будь-якому військовому симуляторі, адже саме правильно сформована команда зможе ефективно виконати поставлені задачі.

Тому для формування команд гравці використовують поширені, однак мало дієві методи, що призводить до негативних наслідків у майбутньому.

1.2.1 Випадковий вибір

Метод випадкового вибору полягає в тому, що система або алгоритм генерують склад команди на випадкових принципах без залучення конкретних критеріїв або аналізу навичок гравців. Цей підхід може бути використаний у випадках, коли немає специфічних вимог щодо складу команди або коли гравці віддають перевагу випадковості у формуванні груп.

Перевагами цього методу можна вважати його простоту та швидкість, тобто метод випадкового вибору не вимагає складних розрахунків чи збору даних про гравців, що робить його дуже швидким у реалізації. Додатково його можна назвати несподіваним та цікавим, адже випадковий підбір може привнести елемент несподіваності та зробити гру цікавішою для гравців, які люблять непередбачуваність та «hardcore».

Щодо недоліків, то одними з таких є нерівномірний розподіл навичок та, як наслідок, зниження ефективності. Цей метод не забезпечує баланс між гравцями з різними навичками, що може призводити до створення команд, де одні гравці значно сильніші за інших, що в свою чергу порушує баланс гри.

Використовується такий метод під час швидких матчів, де гравці бажають скоріше перейти до ігрового процесу, не витрачаючи часу на детальний підбір команди. Яскравим прикладом використання такого методу, з вище наведених, є військові симулятори Squad та Arma 3.

1.2.2 За навичками

Метод формування команд за навичками ґрунтується на аналізі компетенцій та навичок гравців. Цей підхід передбачає оцінку кожного гравця за різними параметрами, такими як стрільба, тактика, медична допомога тощо, та підбір команди таким чином, щоб забезпечити баланс та ефективність.

Великими перевагами цього методу вважаються баланс та ефективність. Врахування навичок кожного гравця дозволяє створити більш збалансовані команди, що підвищує їхню ефективність у виконанні завдань. Як наслідок, відбувається оптимізація ресурсів, тобто кожен гравець виконує ті ролі, в яких він найкращий, що дозволяє максимально ефективно використовувати усі наявні ресурси.

Однак метод є часвитратним та складним у реалізації. Оцінка та аналіз навичок гравців потребують збору та обробки великої кількості даних, що може бути технічно складно та довго.

Найчастіше такий метод використовується у важливій та відповідальній грі. Або у рейтингових іграх, де гравцям необхідно підвищувати свій ранг майстерності, або у кіберспортивних турнірах.

1.2.3 За рангом

Метод формування команд за рангом базується на ієрархічній структурі, де гравці розподіляються відповідно до їхнього рангу чи досвіду. Цей підхід передбачає вибір лідерів на основі їхнього статусу та вправності, а інші гравці розподіляються в команду в залежності від їхнього рівня.

Такий метод дозволяє створити організовану, ієрархічна структуру команд в якій існує чіткий розподіл обов’язків між рангами, що в свою чергу дозволить гравці з вищим рангом навчати та допомагати менш досвідченим.

Проте недоліком можна вважати і сам метод, а конкретніше його вплив на людські якості. Можливе виникнення нерівності в команді, а також залежності. Є вірогідність того, що команди можуть стати занадто залежними від кількох лідерів, що може бути ризиковано у випадку їхньої відсутності.

Найчастіше такий метод використовується у військових симуляторах, де офіцери керують групами солдатів.

1.2.4 На основі машинного навчання

Метод формування команд на основі машинного навчання використовує алгоритми та моделі, щоб автоматично аналізувати навички, стиль гри та стратегічні вміння гравців. Цей підхід дозволяє створювати збалансовані команди, враховуючи індивідуальні сильні та слабкі сторони кожного гравця.

Переваги підходу на основі машинного навчання це точність та об’єктивність, адже алгоритми машинного навчання обробляють великі обсяги даних та використовують математичні алгоритми та моделі. Можливість алгоритмів оновлюватися та адаптуватися робить метод універсальним.

Проте при реалізації виникнуть деякі складнощі – потреба в значних ресурсах та інвестиціях, а налаштування та тестування моделей машинного навчання може бути технічно складним процесом для звичайного гравця.

Такий метод використовується в більш популяризованих та рейтингових іграх, де необхідно мати більш збалансований підбір гравців.

Висновки до розділу 1

Отже, можемо прийти до висновку, що велика кількість військових симуляторів, які використовують різноманітні алгоритми для формування команд гравців. Однак, проаналізувавши їх можна прийти до висновку, що у таких симуляторах використовуються доволі примітивні методи, що негативно впливають на результат усієї гри. Тому, для підвищення ефективності і якості ігрового процесу, необхідно впроваджувати більш складні та продумані системи формування команд. А активно вдосконалювати системи аналізу навичок гравців, дозволить підвищити рівень гри та досягти балансу між командами. Інтеграція більш складних алгоритмів дозволить не лише підвищити точність оцінок, але й забезпечити гнучкість у визначенні ваг критеріїв, що є ключовим аспектом для створення збалансованих та конкурентоздатних команд.

2 ПІДБІР АЛГОРИТМІВ ТА МЕТОДІВ АНАЛІЗУ НАВИЧОК ГРАВЦІВ

2.1 Оцінювання навичок гравців

В організації ефективної гри ключове значення має адекватне оцінювання навичок гравців. Це допомагає не лише у формуванні збалансованих команд, а й в удосконаленні геймплею.

Для цього, як і для формування команд, існують деякі підходи, які вже використовуються у багатьох військових симуляторах.

2.1.1 Аналіз ігрової статистики

Один з перших та найбільш очевидних методів оцінювання навичок гравців - це аналіз їхньої ігрової статистики. Цей підхід передбачає збір та аналіз даних про продуктивність гравців у різних аспектах гри. Наприклад, це може бути кількість знищених ворогів, кількість виконаних завдань, кількість годин, проведених у грі, та різні відзнаки, отримані гравцем. Збір таких даних дозволяє створити детальну картину навичок та компетенцій гравця.

Аналіз ігрової статистики є корисним для формування команд, спрямованих на досягнення найкращих результатів у різних сценаріях гри. Метод може використовуватися для надання гравцям індивідуальної рецензії та рекомендацій щодо покращення їхньої гри, а збір даних і їхній аналіз допоможе виявити сильні та слабкі сторони кожного гравця, що дозволить створювати більш збалансовані команди.

Проте цей аналіз має свої обмеження. Все залежить від самої гри, а якщо ми говоримо про військові симулятори, які акцентують на реалістичності, то дані зі статистики можуть бути не завжди точними, через саму специфіку гри. Причина цього в тому, що такі ігри часто мають велику кількість змінних, які впливають на результат, однак складні в оцінюванні. Наприклад, тактичні рішення або співпраця в команді можуть мати вирішальний вплив, але їх складно виміряти лише за допомогою статистики.

Існує також можливість того, що гравці, усвідомлюючи, які параметри аналізуються, можуть змінювати свою гру, щоб досягти кращих результатів у цих аспектах. Це може призвести до викривлення результатів та погіршенню взаємовідносин.

2.1.2 Тестування

Тестування є ще одним методом оцінювання навичок гравців, який полягає в проведенні спланованих ігрових сеансів. Під час цих сеансів спостерігачі або системи аналізу відстежують реакції гравців на різні ситуації та завдання. Це дозволяє отримати точну інформацію про реакції гравців на різні геймплейні ситуації, їхні стратегії та тактики.

Цей метод може бути особливо корисним для оцінювання навичок, які складно виміряти за допомогою статистики. Наприклад, це може бути реакція на стрес, прийняття рішень в умовах невизначеності, здатність працювати в команді та інші навички, які є важливими для успішної гри у військові симулятори.

Однак цей метод має деякі складнощі. Проведення тестувань може бути витратним за часом та ресурсами, особливо якщо тестування проводиться у реальних ігрових ситуаціях. Крім того, результати тестування можуть бути спотворені різними факторами, такими як емоції гравців під час тестування або умови, в яких проводиться тестування. Наприклад, гравець може бути втомленим або знаходитися під впливом стресу, і як результат – неактуальна статистика гравця.

2.1.3 Самооцінка

Самооцінка є ще одним методом оцінювання навичок гравців, який ґрунтується на тому, що гравець самостійно оцінює свої власні навички та здібності у грі. Цей метод дозволяє гравцям аналізувати свої сильні та слабкі сторони, визначати цілі та напрямки для подальшого розвитку.

Самооцінка може бути виконана шляхом заповнення анкет або опитувальників, які спрямовані на оцінку різних аспектів навичок у грі. Також це дозволяє аналізувати свою власну гру під час процесу гри, роблячи висновки щодо своєї ефективності, стратегій та тактики. Це дозволяє гравцям більш усвідомлено підходити до процесу гри та працювати над покращенням своїх навичок.

Однак, як і будь-який метод оцінки, самооцінка має свої недоліки. Оцінка може бути суб'єктивною, оскільки гравці можуть переоцінювати або недооцінювати свої навички через різні психологічні фактори. Наприклад, гравець може бути занадто критичним до себе або, навпаки, занадто впевненим у своїх здібностях, що може призвести до неточних результатів оцінки.

2.1.4 Комбінований підхід

Комбінований підхід до оцінювання навичок гравців поєднує в собі різні методи та інструменти з метою отримання більш об'єктивної та повної картини їхніх здібностей.

Цей підхід може включати в себе аналіз ігрової статистики для об'єктивної оцінки результатів гравця, використання тестування для визначення конкретних навичок та здібностей, самооцінку для врахування особистих вражень та емоцій гравця під час гри, а також врахування рангу або досвіду гравця у військових симуляторах.

Комбінований підхід дозволяє знизити вплив можливих переваг або обмежень кожного окремого методу шляхом компенсації їхніх недоліків один одним. Наприклад, об'єднання статистичних даних з результатами тестувань може надати більш повну картину навичок гравця, враховуючи як об'єктивні показники, так і суб'єктивні враження. Крім того, цей підхід дозволяє враховувати різні аспекти гри, які можуть бути важливими для оцінки загальної ефективності гравця.

Таким чином, комбінований підхід є більш гнучким та адаптивним, тому ідеально підходить для створення характеристики гравця, яка враховує всі аспекти його гри. Використання цього підходу може допомогти у формуванні

збалансованих команд, де кожен гравець зможе максимально ефективно використовувати свої навички та здібності для досягнення спільної мети.

2.2 Алгоритми та методи аналізу навичок

Маючи на руках всі необхідні дані про гравців, необхідно визначитися яким алгоритмом краще буде їх розподіляти серед команд. Для цього ідеально підходять методи прийняття рішень, що працюють з багатокритеріальними даними. Деякі більш підходячи методи для поставленого завдання, а саме системи сбалансованих команд, будуть наведені нижче.

2.2.1 Метод TOPSIS

Метод TOPSIS є одним з найбільш ефективних та розповсюджених аналітичних методів для прийняття рішень в умовах багатокритеріальності. Цей метод дозволяє визначити найкращу альтернативу серед множини можливих, використовуючи критерії порівняння з ідеальними та антиідеальними рішеннями (див. рис. 2.1).

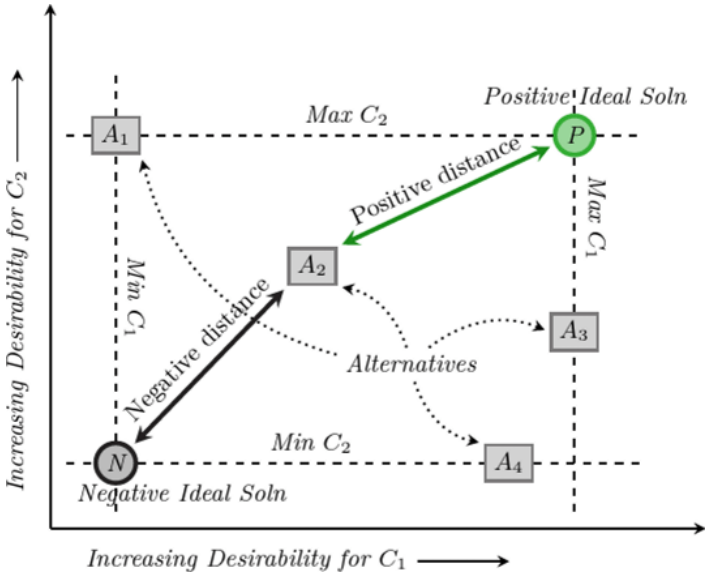

Рисунок 2.1 – Метод прийняття рішень TOPSIS

Головна ідея методу TOPSIS полягає в тому, щоб знайти альтернативу, яка є максимально наближеною до позитивно-ідеального рішення і максимально

віддаленою від негативно-ідеального рішення. Позитивно-ідеальне рішення визначається як гіпотетичне рішення, що забезпечує найкращі можливі значення для всіх критеріїв, тоді як негативно-ідеальне рішення представляє найгірші можливі значення для всіх критеріїв.

Існує кілька основних етапів, які дозволяють послідовно оцінювати та ранжувати альтернативи. До них відносяться: формування матриці рішень, нормалізація критеріїв, зважування критеріїв, визначення ідеальних та антиідеальних рішень, обчислення відстаней до ідеальних рішень та обчислення відносної близькості до ідеального рішення.

Однією з основних переваг методу TOPSIS є його простота та інтуїтивність. Він легко зрозумілий і може бути використаний навіть без глибоких знань у математиці або статистиці, що робить його доступним для широкого кола користувачів. Це, в свою чергу, сприяє його популярності серед практиків у різних сферах. Крім того, цей метод забезпечує комплексний аналіз, дозволяючи враховувати різні критерії та їх ваги. Це важливо, оскільки в реальних ситуаціях прийняття рішень часто доводиться розглядати множинні аспекти одночасно. Завдяки цьому метод забезпечує більш повне і багатогранне бачення ситуації, що сприяє більш обґрунтованим і збалансованим рішенням.

Однак, не зважаючи на свої численні переваги, він має і певні недоліки. Один з них полягає у чутливості до вибору ваг критеріїв. Результати методу можуть значно змінюватись в залежності від того, як розподілені ваги між критеріями. Це може впливати на об'єктивність оцінок і призводити до спотворення результатів, якщо ваги вибрані неправильно або суб'єктивно.

Іншим недоліком є необхідність нормалізації критеріїв, що може додавати додаткової складності в процес обчислень. Це особливо проблематично у випадках, коли критерії мають різні одиниці вимірювання або значно відрізняються за масштабом. Також метод ігнорує можливі взаємозв'язки між критеріями. Він розглядає кожен критерій окремо, не враховуючи, що деякі критерії можуть бути пов'язані між собою або взаємно впливати один на одного. Це може призводити до

спрощення реальної ситуації і недостатньо точного відображення комплексності проблеми.

Проте, незважаючи на певні обмеження, цей метод залишається одним з найпоширеніших у практиці багатокритеріальної оптимізації завдяки своїй простоті, інтуїтивності та широким можливостям застосування. А у контексті військових симуляторів метод TOPSIS може бути використаний для планування та підбору команд шляхом оцінки навичок гравців за різними критеріями, такими як стратегічне мислення, точність стрільби, комунікаційні навички тощо.

2.2.2 Метод SAW

Метод SAW, як і метод TOPSIS, є одним з найпростіших і найбільш розповсюджених методів багатокритеріальної оптимізації. Його сутність полягає в сумарній оцінці кожної альтернативи шляхом додавання зважених значень всіх критеріїв (див. рис. 2.2).

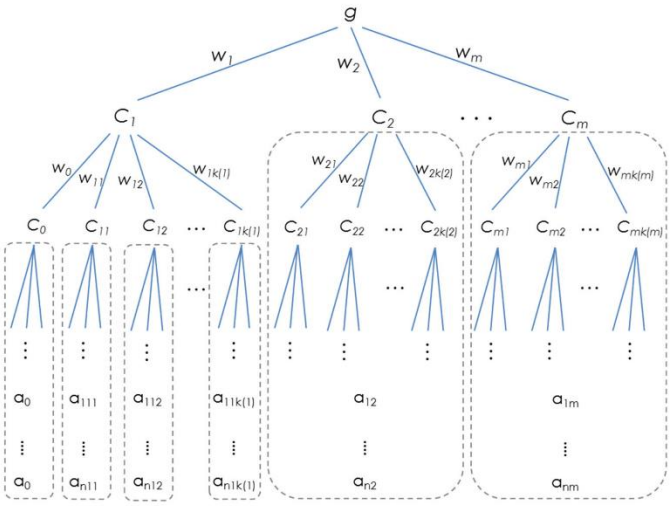

Рисунок 2.2 – Метод прийняття рішень SAW

Метод ґрунтується на припущенні, що загальна оцінка альтернативи може бути представлена як сума її зважених оцінок за кожним критерієм. Для цього необхідно визначити ваги критеріїв, нормалізувати їх значення та обчислити зважену суму для кожної альтернативи. А альтернатива з найвищим значенням суми вважається найкращою.

Основною перевагою методу SAW є його простота та легкість використання. Так як він є одним з найпростіших і найбільш зрозумілих методів для багатокритеріальної оптимізації, робить його доступним для широкого кола користувачів, незалежно від їхньої спеціалізації. А вже це сприяє його популярності серед практиків у різних галузях, таких як економіка, управління та інженерія.

Крім того, метод характеризується швидкістю розрахунків. Обчислення зважених сум для різних альтернатив є швидким та легким процесом, що робить метод ефективним інструментом для прийняття рішень у ситуаціях, де важлива оперативність. Також він відзначається своєю гнучкістю. Метод може бути застосований до різних типів задач, що робить його універсальним інструментом для вирішення багатокритеріальних проблем. Ця універсальність дозволяє використовувати метод SAW у різних галузях, адаптуючи його до специфічних вимог кожної задачі.

Проте, SAW має і певні недоліки. Один з них полягає у чутливості до нормалізації критеріїв. Результати методу можуть значно залежати від способу нормалізації, що може впливати на об'єктивність оцінок і, як наслідок, може спотворити результати і призвести до некоректних висновків. Іншим недоліком є ігнорування можливих взаємозв'язків між критеріями. Метод розглядає кожен критерій окремо, не враховуючи можливих взаємозв'язків між ними, а вже це може призводити до спрощення реальної ситуації і недостатньо точного відображення комплексності проблеми.

Крім того, метод SAW має обмежену можливість для вирішення складних багатокритеріальних задач. У випадках, коли завдання є дуже складним і вимагає врахування численних факторів і їхніх взаємозв'язків, метод SAW може бути недостатньо точним і гнучким. Тож для більш складних задач можуть знадобитися більш просунуті методи багатокритеріальної оптимізації.

Загалом, метод SAW є ефективним та надійним, а завдяки простоті та швидкості аналізу має місце в використанні його для вибору оптимальної команди

в контексті військових симуляторів, де командний склад впливає на результати бойових операцій.

2.2.3 Метод AHP

Метод AHP є потужним інструментом для багатокритеріального прийняття рішень, розроблений Томасом Сааті. Метод дозволяє структурувати складні проблеми прийняття рішень у вигляді ієрархії і здійснювати систематичний аналіз на кожному рівні ієрархії (див. рис. 2.3).

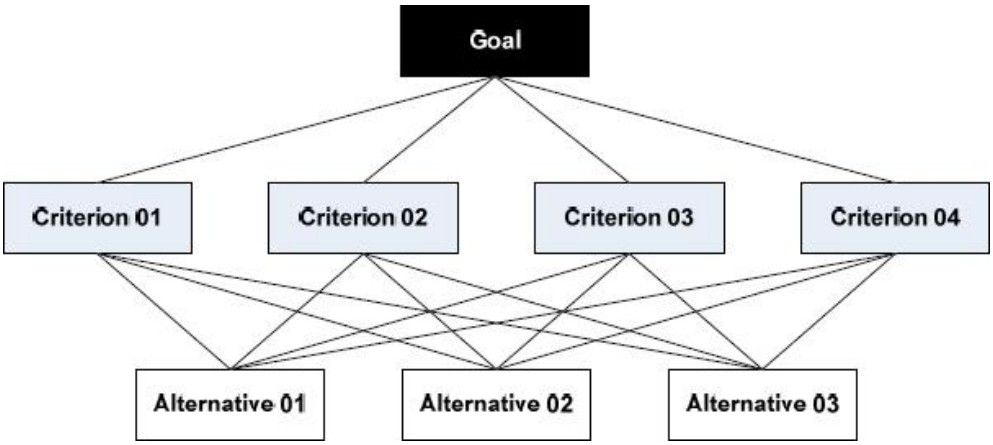

Рисунок 2.3 – Метод прийняття рішень AHP

Він ґрунтується на декомпозиції складної проблеми на ієрархічну структуру, що включає мету, критерії, підкритерії (якщо необхідно) та альтернативи. Потім експерти здійснюють парні порівняння елементів на кожному рівні ієрархії, визначаючи їх відносну важливість за допомогою шкали Сааті. На основі цих порівнянь обчислюються ваги критеріїв і альтернатив, які дозволяють прийняти обґрунтоване рішення.

Однією з головних переваг методу AHP є його структурованість. Він забезпечує чітку ієрархічну структуру для аналізу комплексних проблем, дозволяючи розбити складні задачі на більш прості та зрозумілі компоненти. Це спрощує процес прийняття рішень, роблячи його більш систематичним і логічним. Ще однією важливою перевагою є його здатність інтегрувати кількісні та якісні дані, що дозволяє враховувати різні аспекти рішень, що робить його універсальним

інструментом для аналізу складних проблем. Завдяки цьому можна враховувати не лише кількісні показники, а й суб'єктивні думки та експертні оцінки.

Також метод відзначається прозорістю процесу. Кожен етап методики є зрозумілим і прозорим для учасників процесу, що сприяє кращому розумінню та обґрунтованості прийнятих рішень. Це робить його зручним інструментом для роботи в команді та для прийняття колективних рішень.

Однак метод AHP має і свої недоліки. Один з них полягає у суб'єктивності оцінок. Результати сильно залежать від суб'єктивних оцінок експертів, що може впливати на об'єктивність та точність кінцевих рішень. Це означає, що якість результатів значною мірою залежить від компетентності та об'єктивності експертів, які проводять оцінювання. Також метод є чутливим до неконсистентності в парних порівняннях. Неконсистентність може призводити до спотворення результатів та зниження їхньої надійності. Тому важливо забезпечити високу ступінь узгодженості парних порівнянь для отримання достовірних результатів.

Крім того, метод AHP може бути складним для застосування у випадках з великою кількістю критеріїв та альтернатив. Процес парних порівнянь стає трудомістким та складним, що може вимагати значних зусиль та часу для проведення аналізу.

У висновку, метод AHP є потужним інструментом для прийняття рішень в умовах багатокритеріальності. Його ієрархічний підхід дозволяє структурувати складні проблеми та здійснювати систематичний аналіз на кожному рівні. Завдяки своїй здатності інтегрувати кількісні та якісні дані, він знаходить широке застосування у різних сферах. Незважаючи на деякі обмеження, пов'язані з суб'єктивністю оцінок та чутливістю до неконсистентності, цей метод залишається одним з найефективніших засобів багатокритеріального аналізу.

2.2.4 Метод MOORA

Метод MOORA є ефективним інструментом для багатокритеріального прийняття рішень, розробленим виключно для оцінки та ранжування альтернатив

за допомогою вагових коефіцієнтів. Цей метод використовується для вирішення складних проблем прийняття рішень, де важливо оцінювати альтернативи з різних кутків зору та враховувати їх взаємозалежність (див. рис. 2.4).

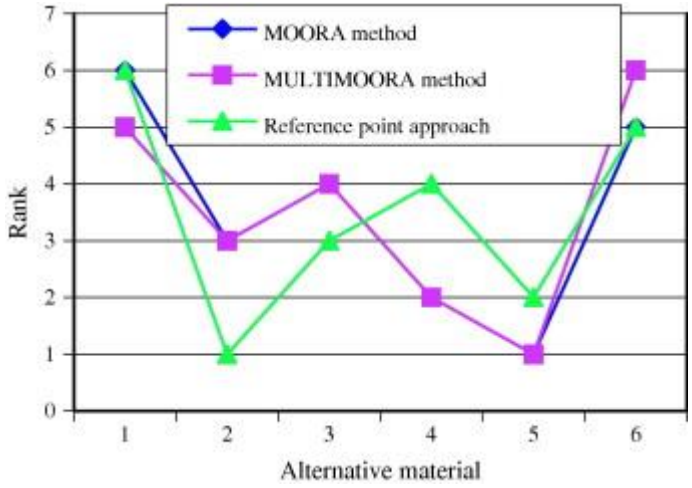

Рисунок 2.4 – Метод прийняття рішень MOORA

MOORA базується на порівнянні альтернатив за допомогою взважених коефіцієнтів, які відображають важливість кожного критерію. Основна мета - визначити та ранжувати альтернативи відповідно до їх відносного значення за всіма критеріями. Метод дозволяє обирати ту альтернативу, яка досягає найкращого балансу між всіма критеріями, і таким чином є найбільш оптимальною для прийняття рішення.

Головні переваги методу – його простота та ефективність. Він відносно простий у реалізації, що дозволяє користувачам швидко та ефективно ранжувати альтернативи за багатокритеріальними оцінками, що робить його доступним для широкого кола практиків та зручним для використання у різних галузях. Ще однією важливою перевагою є його гнучкість у виборі ваг критеріїв. Ваги критеріїв можна встановлювати залежно від конкретної ситуації або експертних оцінок, що дозволяє адаптувати метод до специфічних вимог кожного завдання, що забезпечує більшу точність та обґрунтованість прийнятих рішень.

Також метод відзначається здатністю до інтеграції різних типів даних. Він дозволяє враховувати як кількісні, так і якісні аспекти альтернатив, що робить його

універсальним інструментом для аналізу складних проблем. Це особливо важливо в умовах, де необхідно враховувати різноманітні фактори та показники.

Однак MOORA має і свої недоліки. Один з них полягає у чутливості до вибору мінімальних та максимальних значень для нормалізації. Результати можуть значно змінюватись в залежності від вибору цих значень, що може впливати на об'єктивність оцінок. Тому важливо обережно підходити до вибору нормалізаційних параметрів для забезпечення достовірності результатів. Іншим недоліком є суб'єктивність визначення ваг критеріїв. Визначення ваг може бути суб'єктивним і вимагати узгодження між експертами, що може ускладнювати процес прийняття рішень. Це означає, що якість результатів залежить від компетентності та об'єктивності експертів, які проводять оцінювання.

Тож метод MOORA потужним інструментом, що дозволяє ефективно враховувати різні аспекти альтернатив, а його простота та ефективність роблять його популярним в різних галузях, в тому числі й у військових симуляторах. Попри певні обмеження, такі як чутливість до вибору нормалізаційних параметрів та суб'єктивність визначення ваг, метод залишається одним з найефективніших засобів багатокритеріального аналізу. Крім того, його можна поєднувати з іншими методами багатокритеріального аналізу для підвищення точності та надійності результатів, що дозволяє компенсувати окремі недоліки та отримати більш комплексний підхід до вирішення складних проблем.

Висновки до розділу 2

Існує багато різних алгоритмів та методів, які можна використовувати для оцінки навичок гравців у відеоіграх. Найкращий метод буде залежати від конкретної гри, цілей аналізу та доступних даних. У даному контексті для оцінки навичок гравців передбачається використання комбінованої методики. Цей підхід включає три основні компоненти: тестування, аналіз статистики, самооцінка.

На основі цих даних для формування збалансованих команд буде використовуватися метод TOPSIS. Цей метод дозволяє ефективно оцінювати та

ранжувати гравців за різними критеріями, порівнюючи їх з ідеальним та антиідеальним рішеннями. TOPSIS забезпечує об'єктивний підхід до визначення найкращих кандидатів для команд, враховуючи комплексний набір критеріїв, таких як стратегічне мислення, точність стрільби, комунікаційні навички тощо.

У перспективі, для покращення точності та гнучкості системи оцінки навичок гравців, планується поєднання методу TOPSIS з методом MOORA. Метод MOORA додає можливість врахування взаємозалежностей між критеріями та оцінювання альтернатив на основі взважених коефіцієнтів. Це дозволить підвищити точність оцінок, забезпечити гнучкість у визначенні ваг критеріїв та провести більш комплексний аналіз навичок гравців.

3 МЕТОДОЛОГІЯ ТА ІНСТРУМЕНТИ РОЗРОБКИ

3.1 Багатокритеріальний метод аналізу рішень TOPSIS

Метод TOPSIS є одним з методів багатокритеріального аналізу рішень, що допомагає оцінити і вибрати найкращі варіанти за кількома критеріями. Основна ідея полягає в тому, що найкращий варіант повинен бути найближчим до ідеального рішення і найдальшим від найгіршого (див. рис. 3.1).

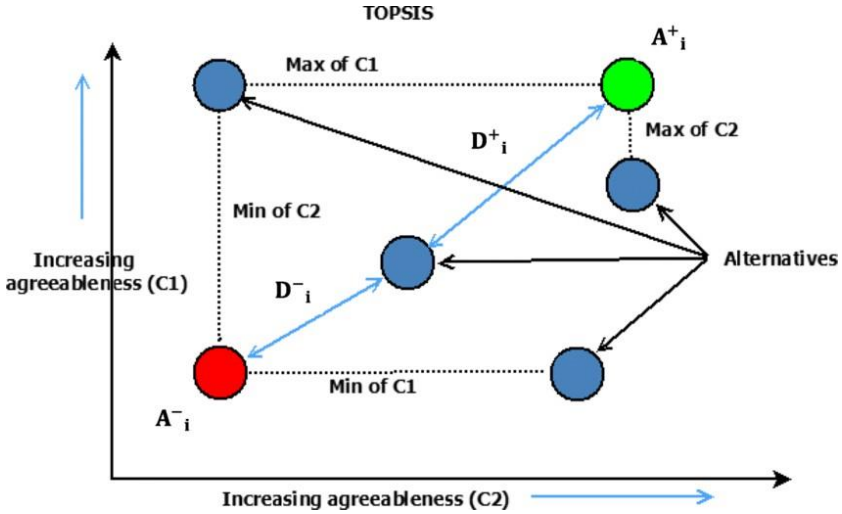

Рисунок 3.1 – Метод прийняття рішень TOPSIS

Спочатку формується матриця рішень, де кожен рядок представляє варіант, а кожен стовпчик – критерій. Далі відбувається нормалізація цієї матриці. Виконується це для того, щоб різні одиниці виміру не впливали на результат. Формула нормалізації:

$$r_{ij} = \frac{x_{ij}}{\sqrt{\sum_{k=1}^{m} x_{kj}^2}}, \tag{3.1}$$

де r_{ij} – нормалізоване значення. Після визначаються вагові коефіцієнти для кожного критерію, щоб показати їх важливість. У подальшому формується зважена нормалізована матриця з якої після визначаються ідеальне та антиідеальне рішення. Відбувається обчислення відстаней до ідеального та антиідеального рішень.

Відстань до ідеального рішення:

$$S_i^+ = \sqrt{\Sigma(v_{ij} - v_j^+)} \quad (3.2)$$

Відстань до антиідеального рішення:

$$S_i^- = \sqrt{\Sigma(v_{ij} - v_j^-)} \quad (3.3)$$

Далі відбувається розрахунок відносної близькості до ідеального рішення:

$$C_i^* = \frac{S_i^-}{S_i^+ + S_i^-} \quad (3.4)$$

І під самий кінець відбувається ранжування варіантів, де вони розташовуються за значенням відносної близькості до ідеального рішення у порядку зменшення. Варіант з найбільшим таким значенням є найкращим.

Переваги такого методу в тому, що він простий та зрозумілий у використанні, можливо враховувати як кількісні, так і якісні критерії, а також наявна чітка процедура ранжування альтернатив.

3.2 Мова програмування Python

Python – це високорівнева мова програмування загального призначення, створена в 1991 році (див. рис. 3.2). Її основні особливості в тому, що вона є простою у використанні, має читабельний синтаксис і багаті можливості.

Рисунок 3.2 – Логотип мови програмування Python

Однією з найбільш визначальних характеристик Python є його високорівнева природа. Це означає, що мова дозволяє писати код, який близький до людської мови, а не до машинної. Завдяки цьому, програмування на Python стає легким і зрозумілим навіть для початківців. Крім того, Python є інтерпретованою мовою, що означає, що код виконується рядок за рядком без необхідності попередньої компіляції. Це значно полегшує тестування та налагодження програм.

Також Python підтримує динамічну типізацію, що означає, що типи змінних визначаються автоматично під час виконання програми. Це спрощує процес написання коду, але вимагає певної уважності під час налагодження.

Дуже важлива особливість Python в тому, що він має велику стандартну бібліотеку, яка включає модулі для роботи з файлами, системними викликами, інтернет-протоколами, базами даних, веб-серверами тощо. Крім того, Python підтримує різні парадигми програмування, такі як об'єктно-орієнтоване, процедурне та функціональне програмування. Python є кросплатформеною мовою, тобто він працює на різних операційних системах, таких як Windows, macOS і Linux.

Це дозволяє розробляти програми, які можуть бути виконані на будь-якій з цих платформ без додаткових змін у коді, що ставить Python вище рівнем над іншим мовами програмування.

3.3 Інтегроване середовище розробки PyCharm

PyCharm є одним із найпопулярніших інтегрованих середовищ розробки для мови програмування Python, створеним компанією JetBrains (див. рис. 3.3). Це IDE забезпечує розробникам набір інструментів, які спрощують написання коду, його налагодження та тестування. PyCharm має декілька особливих характеристик, що так приваблюють розробників.

Рисунок 3.3 – Логотип інтегрованого середовища розробки PyCharm

Це IDE аналізує написаний код і надає рекомендації для його автоматичного доповнення, що дозволяє значно прискорити процес розробки. Інтегрований дебагер допомагає легко знаходити і виправляти помилки у програмах. Це дуже зручно, адже не потрібно використовувати зовнішні програми для дебагінгу. PyCharm підтримує багато популярних фреймворків, таких як Django, Flask, і React, що дозволяє розробляти як веб-додатки, так і інші типи програм, що робить його ну дуже універсальним. IDE має вбудовані інструменти для роботи з Git, SVN та іншими системами контролю версій, що є важливим аспектом сучасної розробки програмного забезпечення, а підтримка віртуальних середовищ дозволяє ізолювати проекти один від одного та управляти їхніми залежностями без конфліктів.

У результаті цього багато програмістів й обирають саме це інтегроване середовище розробки.

3.4 Бібліотеки IDE PyCharm

При розробці системи на мові програмування Python у середовищі PyCharm було використано декілька бібліотек для зручного використання системи.

3.4.1 Бібліотека NumPy

NumPy – це одна з ключових бібліотек у мові програмування Python, яка забезпечує підтримку великих, багатовимірних масивів та матриць, разом із

великою колекцією високорівневих математичних функцій для операцій з цими масивами (див. рис. 3.4).

Рисунок 3.4 – Логотип бібліотеки NumPy

Дана бібліотека активно використовується для обробки даних у різних областях, від фізики та інженерії до машинного навчання та штучного інтелекту. Вона особливо корисна у ситуаціях, коли потрібно ефективно обробляти великі обсяги даних, виконувати трансформації даних або застосовувати складні математичні формули.

3.4.2 Бібліотека discord.py для створення ботів у Discord

Discord.py – бібліотека на мові програмування Python, призначена для створення ботів, які можуть взаємодіяти з користувачами на платформі Discord (див. рис. 3.5). Discord є популярним місцем спілкування, особливо серед геймерів та розробників програмного забезпечення, і боти можуть здійснювати багато корисних та розважальних функцій на серверах.

Рисунок 3.5 – Логотип бібліотеки Discord.py

Бібліотека надає зручний інтерфейс для взаємодії з Discord API, дозволяючи легко створювати, модифікувати та управляти ботами. Важливо зазначити, що вона має багатий набір функцій, які дозволяють створювати складні боти з можливістю відправки повідомлень, управління каналами, керування ролями користувачів і багато іншого, що в свою чергу дуже необхідно у розробці систем, що беруть конкретні дані. Discord.py дозволяє легко реагувати на різні події в Discord, такі як входження нових користувачів на сервер, відправлення повідомлень. Це дозволяє визначати команди, які будуть виконувати певні дії, коли користувачі використовують їх у чаті.

У результаті ми маємо зручну бібліотеку, що дозволить інтегрувати систему у платформу Discord для зрозумілості та покращення інтерфейсу.

3.4.3 Бібліотека matplotlib.pyplot для візуалізації даних

Бібліотека matplotlib.pyplot є інструментом для створення графіків та візуалізації даних у мові програмування Python (див. рис. 3.6). Вона надає широкі можливості для побудови різних типів графіків і діаграм, що дозволяє аналізувати та відображати дані в зручному вигляді.

Рисунок 3.6 – Логотип бібліотеки matplotlib.pyplot

Ця бібліотека використовується у багатьох галузях, включаючи наукові дослідження, фінансовий аналіз, веб-розробку та інші області, де потрібно візуалізувати дані. Конкретно у випадку системи формування збалансованих команд, ця бібліотека використовується для візуалізації результатів обчислень методом TOPSIS у вигляді радар-діаграми.

Висновки до розділу 3

Отже, розділ включає детальний опис методів та технологій, які застосовувалися під час розробки багатокритеріальної системи формування збалансованих команд. Для вирішення поставленої задачі був обраний метод TOPSIS, який дозволив ефективно оптимізувати вибір команд на основі набору критеріїв. Програмування велося мовою Python з використанням інтегрованого середовища розробки PyCharm. Важливим елементом процесу стала візуалізація системи та аналіз отриманих результатів за допомогою бібліотек NumPy, discord.py та matplotlib.pyplot.

4 ПРОГРАМУВАННЯ ТА ТЕСТУВАННЯ СИСТЕМИ

4.1 Середовище тестування

Для тестування розробленої системи було обрано діскорд канал «Ukraine home» (див. рис. 4.1). Даний канал є українською спільнотою гравців, які активно використовують ігровий симулятор Arma 3 (див. рис. 4.2). Саме через використання військового симулятора Arma 3 цією спільнотою, а також наявність в мене прав модератора в ній, дозволяє інтегрувати систему без зайвих проблем.

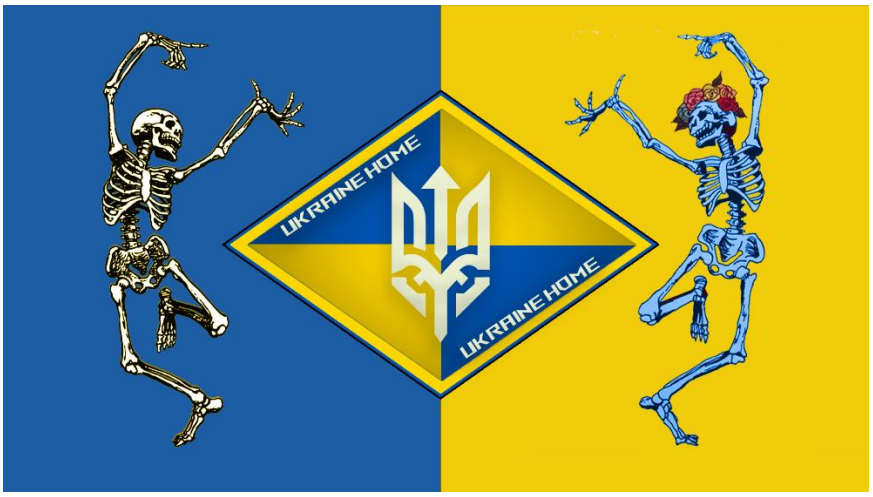

Рисунок 4.1 – Логотип спільноти

Також важливим аспектом є велика та активна спільнота гравців, що дозволяє збирати цінні дані та відгуки для подальшого покращення системи.

Рисунок 4.2 – Адміністрація спільноти безпосередньо у Arma 3

4.2 Визначення необхідних навичок бійців для підрозділів та їх ваги

Для побудови ефективних команд в симуляторі Arma 3, були визначені ключові навички та характеристики бійців, які враховуються при формуванні команд.

Таблиця 4.1 – Кількістна оцінка навичок гравців

	Адекватність	Кількість годин у грі	Стрільба	Тактика	Комунікабельність
Enot	10	1000	10	10	10
Kotran	8	600	6	10	10
Bogdas	6	831	8	10	10

Заповнення даних відбувалося за допомогою комбінації трьох підходів: самооцінка, тестування інструкторами, готова оцінка самою грою. У таблиці було продемонстровано лише 5 видів навичок, проте загальна їх кількість становить 21: адекватність, кількість годин у грі, стрільба, тактика, комунікабельність, адаптивність, лідерство, навченість, спостережливість, володіння специфічною зброєю, знання медицини, навички водіння броньованою технікою, навички володіння льотною технікою, навички управління дронами, навігація, вік, рівень відігрування, навички бомбардування, навички десантування, знання двигуна Arma 3.

Додатково були створені кілька підрозділів, кожен з яких має свої вимоги до навичок і ваги для кожної навички. Ці ваги визначають пріоритетність навичок для конкретного підрозділу.

Таблиця 4.2 – Підібрані ваги до кожного підрозділу

Підрозділ	Адекватність	Кількість годин у грі	Стрільба	Тактика	Комунікабельність

Special Force Group [FOXHOUND]	0,7	0,7	0,9	0,8	0,8
Перше механізоване відділення [ВОЛЯ]	0,5	0,2	0,7	0,6	0,5
1 Полк Армійської Авіації [OWL]	0,8	0,8	0,1	0,8	0,9
MedCom "Omega" [OMG]	0,7	0,7	0,6	0,8	0,8
Assault [ASL]	0,7	0,5	0,8	0,8	0,9
Адміністрація [UAH]	1	0,9	0,1	0,1	0,7

Для більшого розуміння специфік та завдань підрозділів, буде наведена інформація про деякі з них від їх засновників.

Перше механізоване відділення загального призначення "ВОЛЯ" – це підрозділ, основною функцією якого є ведення безпосередніх бойових сутичок з ворогом з підтримкою важкої техніки (див. рис. 4.3). Відділення спеціалізується на таких завданнях, як штурм, зачистка, оборона, розвідка боєм та цільове знищення ворожих підрозділів. Через використання механізованої підтримки, ми здатні ефективно діяти в різних умовах та на різних типах місцевості, швидко переміщатись з позиції на позицію, ефективно підтримувати інші підрозділи.

Рисунок 4.3 – Логотип підрозділу «ВОЛЯ»

MedCom "Omega" [OMG] – медичне командування "Омега" – високопрофесійна тактична медична група, головна мета якої надання швидкої та ефективної допомоги пораненим у зоні бойових дій, надання гуманітарної медичної допомоги постраждалим в умовах військових конфліктів чи кризових ситуацій, евакуації поранених та забезпечення медичної підтримки в процесі проведення тактичних операцій (див. рис. 4.4).

Рисунок 4.4 – Логотип підрозділу «Omega»

Assault [ASL] – це механізований загін який виконує завдання на броньованій техніці.Основні завдання групи це штурм або прикриття піхоти на лінії фронту (див. рис. 4.5).

Рисунок 4.5 – Логотип підрозділу «ASL»

Special Force Group [FOXHOUND] – це елітний підрозділ, що спеціалізується на виконанні надсекретних і надскладних місій по всьому світу (див. рис. 4.6). Наші бійці – це воїни майбутнього, що поєднують у собі силу, розум і високі технології.

Рисунок 4.6 – Логотип підрозділу «FOXHOUND»

Адміністрація [UAH] – це підрозділ до якого входять гравці, що мають досвід роботи з двигуном гри Арма 3 (див. рис. 4.7). Основна місія підрозділу полягає в підтримці та покращенню якості гри, слідкування за дотриманням правил спільноти, технічне обслуговування серверу гри та Discord сервера.

Рисунок 4.7 – Логотип підрозділу «UAH»

1 Полк Армійської Авіації [OWL] – це елітний авіаційний підрозділ, що спеціалізується на проведенні високоточних і високоефективних операцій з повітря. Наші пілоти є майстрами своєї справи, здатними виконувати місії будь-якої складності, включаючи розвідку, підтримку з повітря та евакуацію. Використовуючи найсучасніші вертольоти і безпілотні літальні апарати, ми забезпечуємо оперативну мобільність і перевагу в будь-якому конфлікті.

На основі цих таблиць, використовуючи метод TOPSIS, система буде рекомендувати бійцям у який підрозділ їм краще йти.

4.3 Програмна реалізація

Для інтеграції системи з платформою Discord було створено бота через Discord Developer Portal. Бот отримав відповідний токен доступу, який використовується для автентифікації та зв'язку з Discord API.

Для реалізації функціоналу бота було використано бібліотеку discord.py (див. рис. 4.8). Ця бібліотека дозволяє легко взаємодіяти з Discord API та реалізовувати різноманітні функції, такі як обробка команд, отримання та відправка повідомлень.

```
import discord
import numpy as np
import matplotlib.pyplot as plt
from discord.ext import commands

intents = discord.Intents.default()
intents.message_content = True

bot = commands.Bot(command_prefix='!', intents=intents)
```

Рисунок 4.8 – Імпорт бібліотеки та налаштування бота

Цей код імпортує необхідні бібліотеки та налаштовує бота. Зокрема, discord.Intents.default() активує базові наміри (intents) для бота, а commands.Bot створює об'єкт бота з префіксом команд '!'.

Далі було визначено ваги для кожного підрозділу, які будуть використовуватись при обчисленні результатів для бійців (див. рис. 4.9).

```
weights = {
    'FOXHOUND': [0.7, 0.7, 0.9, 0.8, 0.8, 0.8, 0.6, 0.7, 0.7, 0.9, 0.6, 0.8, 0.5, 0.7, 0.7, 0.9, 0.5, 0.7, 0, 0.7, 0],
    'ВОЛЯ': [0.5, 0.2, 0.7, 0.6, 0.5, 0.6, 0, 0, 0.5, 0, 0.6, 0.8, 0, 0, 0, 0.6, 0.4, 0.7, 0, 0.5, 0],
    'OWL': [0.8, 0.8, 0, 0.8, 0.9, 0.4, 0.5, 0.7, 0.7, 0.7, 0.5, 0, 0.8, 0, 0.7, 0.9, 0.6, 0.5, 0.6, 0.7, 0],
    'OMG': [0.7, 0.7, 0.6, 0.8, 0.9, 0.8, 0.8, 0.8, 0.7, 0.3, 0.9, 0.5, 0.5, 0, 0.5, 0.8, 0.6, 0.7, 0, 0.6, 0],
    'ASL': [0.7, 0.5, 0.8, 0.8, 0.9, 0.8, 0.5, 0.6, 0.7, 0.7, 0.5, 0.7, 0.1, 0, 0.7, 0.8, 0.5, 0.5, 0, 0.3, 0],
    'UAH': [1.0, 0.9, 0, 0, 0.7, 0.8, 0.8, 0.8, 0.3, 0, 0, 0, 0, 0, 0, 0.7, 0.8, 0, 0, 0.8]
}
```

Рисунок 4.9 – Ваги для підрозділів

Також було зібрано дані про характеристики бійців, які включають різні параметри, що впливають на їх ефективність (див. рис. 4.10).

```
data = {
    'Enot': [10, 1000, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 9, 10, 10, 10, 24, 10, 10, 10],
    'Kotran': [8, 600, 6, 10, 10, 9, 10, 10, 7, 6, 10, 6, 10, 4, 5, 10, 20, 10, 8, 10],
    'Bogdas': [6, 831, 8, 10, 10, 8, 9, 10, 5, 6, 10, 9, 7, 6, 5, 10, 19, 8, 5, 10],
    'Skelet': [7, 231, 4, 5, 6, 7, 9, 10, 3, 7, 10, 9, 5, 6, 6, 10, 18, 7, 3, 10],
    'Fish': [10, 1000, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 9, 10, 10, 10, 24, 10, 10, 10],
}
```

Рисунок 4.10 – Дані бійців

Нижче наведено код реалізує функцію нормалізації, яка приводить характеристики бійців до спільної шкали, дозволяючи коректно порівнювати їх між собою (див. рис. 4.11).

```
def normalize(data):
    keys = list(data.keys())
    matrix = [data[key] for key in keys]
    transposed_matrix = list(map(list, zip(*matrix)))
    normalized_transposed_matrix = []
    for col in transposed_matrix:
        min_val = min(col)
        max_val = max(col)
        normalized_col = [(x - min_val) / (max_val - min_val) if max_val != min_val else 0 for x in col]
        normalized_transposed_matrix.append(normalized_col)
    normalized_matrix = list(map(list, zip(*normalized_transposed_matrix)))
    normalized_data = {keys[i]: normalized_matrix[i] for i in range(len(keys))}
    return normalized_data
norm_data = normalize(data)
```

Рисунок 4.11 – Нормалізація даних

Ця функція застосовує ваги до нормалізованих даних, враховуючи специфічні вимоги кожного підрозділу (див. рис. 4.12).

```
def weighted_normalize(norm_data, weight):
    weighted_data = {}
    for key, values in norm_data.items():
        weighted_data[key] = np.multiply(values, weight)
    return weighted_data
```

Рисунок 4.12 – Зважування нормалізованих даних

Далі визначаються ідеальні та антиідеальні значення для кожного підрозділу (див. рис. 4.13).

```
def ideal_anti_ideal(weighted_data):
    ideal = np.max(list(weighted_data.values()), axis=0)
    anti_ideal = np.min(list(weighted_data.values()), axis=0)
    return ideal, anti_ideal
```

Рисунок 4.13 – Визначення ідеальних та антиідеальних значень

Після цього необхідно розрахувати відстані для кожного бійця до ідеального та антиідеального рішень (див. рис. 4.14).

```
def distances_to_ideal(weighted_data, ideal, anti_ideal):
    distances = {}
    for key, values in weighted_data.items():
        d_plus = np.sqrt(np.sum(np.square(values - ideal)))
        d_minus = np.sqrt(np.sum(np.square(values - anti_ideal)))
        distances[key] = (d_plus, d_minus)
    return distances
```

Рисунок 4.14 – Розрахунок відстаней до ідеального та антиідеального рішень

Останій крок для реалізації метода TOPSIS створюється функція, яка обчислює коефіцієнти близькості для кожного бійця, що дозволяє оцінити їх відповідність ідеальному рішенню (див. рис. 4.15).

```
def closeness_coefficients(distances):
    coefficients = {}
    for key, (d_plus, d_minus) in distances.items():
        coefficients[key] = d_minus / (d_plus + d_minus)
    return coefficients
```

Рисунок 4.15 – Коефіцієнти Близькості

Подальші дії були з створення команд для взаємодії з ботом. Команда «!calculate» виконує розрахунки та виводить результати у вигляді коефіцієнтів близькості для кожного бійця та підрозділу (див. рис. 4.16).

```python
@bot.command()
async def calculate(ctx):
    global results
    results.clear()
    for unit, weight in weights.items():
        min_length = min(len(next(iter(data.values()))), len(weight))
        norm_data_clipped = {key: values[:min_length] for key, values in norm_data.items()}
        weight_clipped = weight[:min_length]
        weighted_data = weighted_normalize(norm_data_clipped, weight_clipped)
        ideal, anti_ideal = ideal_anti_ideal(weighted_data)
        distances = distances_to_ideal(weighted_data, ideal, anti_ideal)
        coefficients = closeness_coefficients(distances)
        results[unit] = coefficients

    response = ""
    for unit, coefficients in results.items():
        response += f"Підрозділ: {unit}\n"
        for name, coeff in coefficients.items():
            response += f"  Боєць: {name}, Коефіцієнт близькості: {coeff:.4f}\n"

    await ctx.send(response)
```

Рисунок 4.16 – Команда «!calculate»

Команда «!plot» створює радарні діаграми для обраних бійців та відправляє їх у канал (див. рис. 4.17 та 4.18).

```python
@bot.command()
async def plot(ctx):
    global results
    if not results:
        await ctx.send("Спочатку введіть команду `!calculate`.")
        return

    def plot_radar_charts_for_players(results, players):
        num_players = len(players)
        fig, axes = plt.subplots( nrows: 1, num_players, figsize=(6 * num_players, 6), subplot_kw=dict(polar=True))

        if num_players == 1:
            axes = [axes]

        for ax, fighter in zip(axes, players):
            labels = list(results.keys())
            scores = [results[unit][fighter] for unit in labels]

            num_vars = len(labels)
            angles = np.linspace( start: 0, 2 * np.pi, num_vars, endpoint=False).tolist()
            angles += angles[:1]
            scores += scores[:1]
```

Рисунок 4.17 – Команда «!plot»

```
        ax.plot(angles, scores, 'o-', linewidth=2, label=fighter)
        ax.fill(angles, scores, alpha=0.25)
        ax.set_thetagrids(np.degrees(angles[:-1]), labels)
        ax.set_title(f"Радарна діаграма для {fighter}")

    plt.tight_layout()
    filename = 'radar_chart.png'
    plt.savefig(filename)
    plt.close()

    return filename

players = ['Kotran','Enot','Skelet']
filename = plot_radar_charts_for_players(results, players)

await ctx.send(file=discord.File(filename))
```

Рисунок 4.18 – Команда «!plot»

Ну і заключну частину виконує рядок «bot.run('TOKEN')», який запускає бота, використовуючи токен, отриманий під час реєстрації бота у Discord Developer Portal.

4.4 Результат виконання програми

При запуску бота на Discord сервері та введення команди «!calculate» у текстовому каналі з’явиться (див. рис. 4.19):

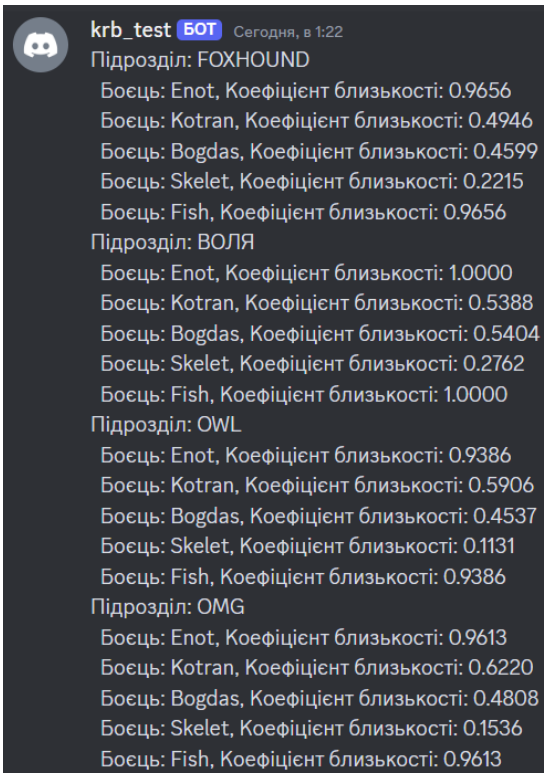

Рисунок 4.19 – Результат команди «!calculate»

І після введення команди «!plot» у текстовому каналі виведи радар-діаграму, що наглядно продемонструє результат роботи системи (див. рис. 4.20-4.22):

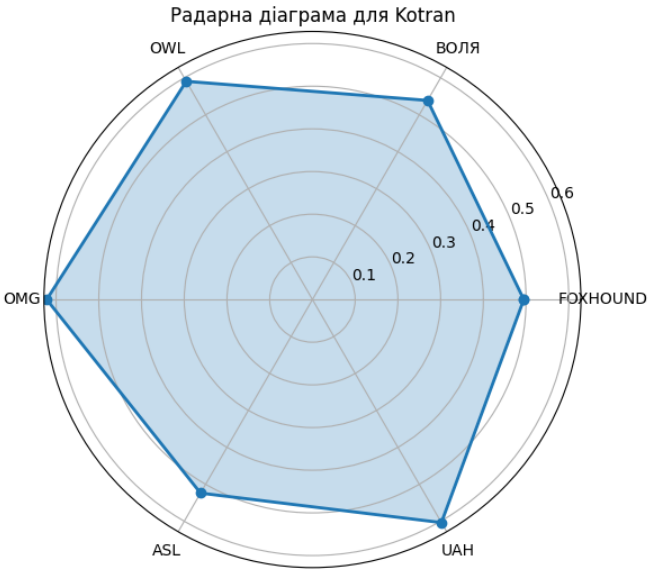

Рисунок 4.20 – Результат команди «!plot» для бійця Kotran

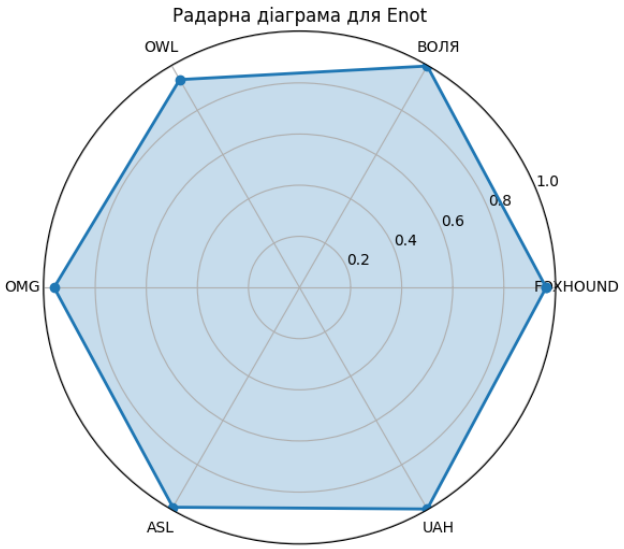

Рисунок 4.21 – Результат команди «!plot» для бійця Enot

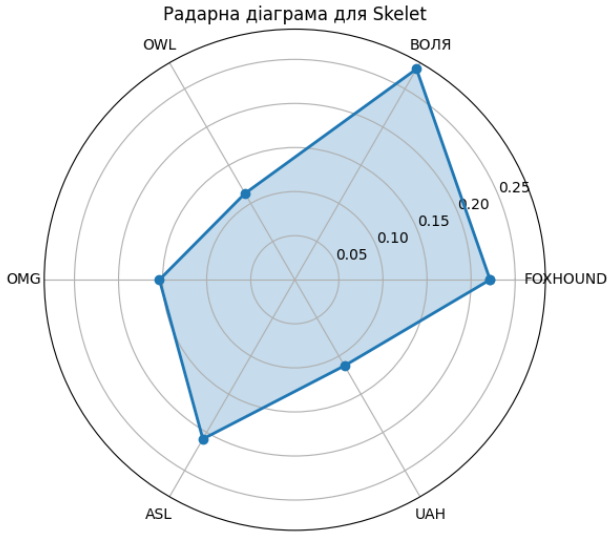

Рисунок 4.22 – Результат команди «!plot» для бійця Skelet

4.5 Ефективність системи

Інтеграція системи автоматизованого формування збалансованих команд на основі багатокритеріального аналізу показала значне зростання ефективності роботи різних підрозділів. Розробка та впровадження цієї системи дозволили

оптимізувати процес відбору бійців, враховуючи їхні індивідуальні навички та компетенції, що сприяло покращенню загальної координації та продуктивності команд.

4.5.1 Ефективність підрозділу «ВОЛЯ»

Завдяки покращенню координації та оптимізації складу команд, час виконання штурмових операцій зменшився на 20%, що дозволило знижувати втрати та підвищувати успішність операцій. Використання багатокритеріальної системи дозволило краще планувати та виконувати операції з зачистки територій, що призвело до підвищення успішності на 25%.

Покращена комунікація та розподіл ресурсів дозволили підвищити ефективність оборонних операцій на 18%. Оптимізація складу підрозділів та краща координація дозволили зменшити час на проведення розвідки боєм на 15%, що сприяло більш ефективному виявленню ворожих сил (див. рис. 4.23).

Результат інтеграції багатокритеріальної системи – збільшення ефективності підрозділу на 22.5%.

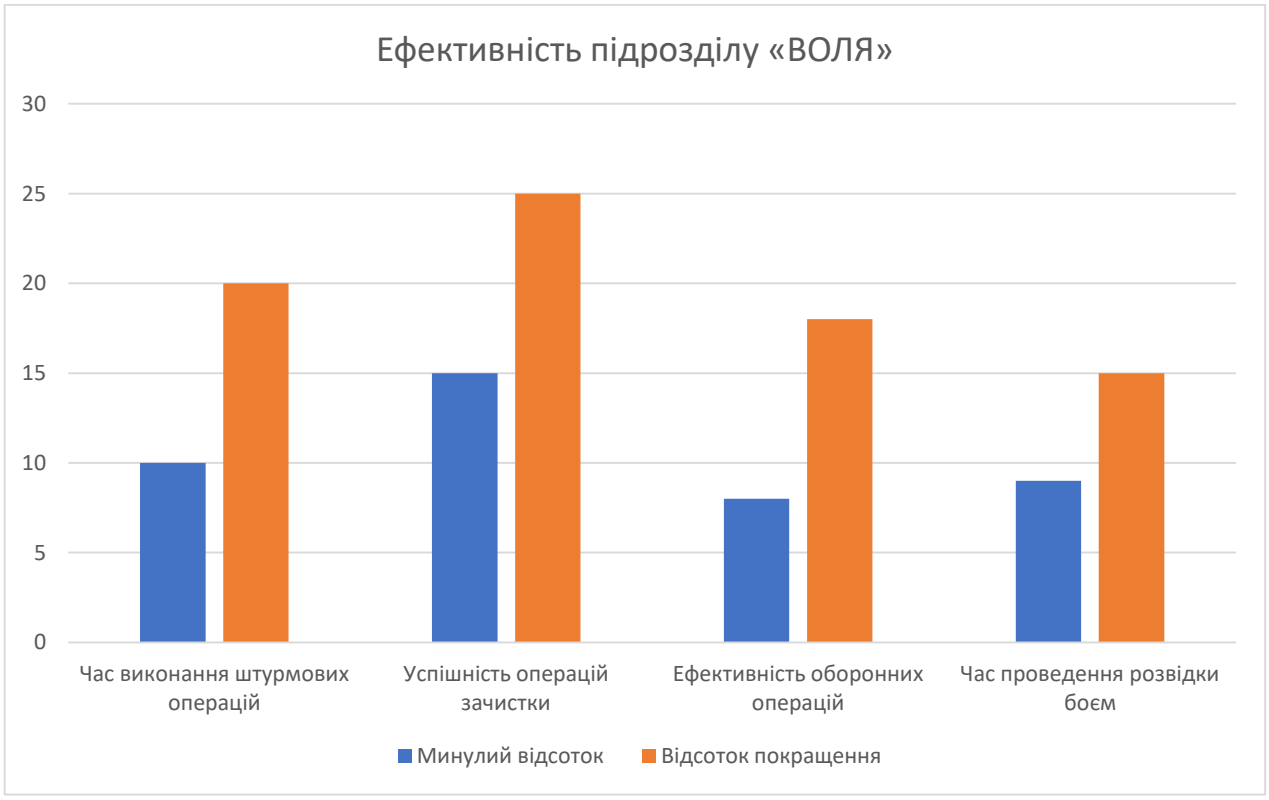

Рисунок 4.23 – Ефективність підрозділу «ВОЛЯ»

4.5.2 Ефективність підрозділу «Omega»

Підрозділ «Omega», високопрофесійна тактична медична група, також зазнав значного покращення ефективності після інтеграції системи TOPSIS для оцінки та відбору персоналу. Завдяки більш точному відбору медиків та координаторів, підрозділ зміг покращити свою роботу за кількома ключовими параметрами. Ефективність надання швидкої допомоги пораненим у зоні бойових дій збільшилася на 22% завдяки залученню медиків з високими медичними знаннями та навичками. Завдяки кращій адаптивності та лідерським якостям координаторів час на евакуацію зменшився на 18% (див. рис. 4.24).

Результат інтеграції багатокритеріальної системи – збільшення ефективності підрозділу на 21.25%.

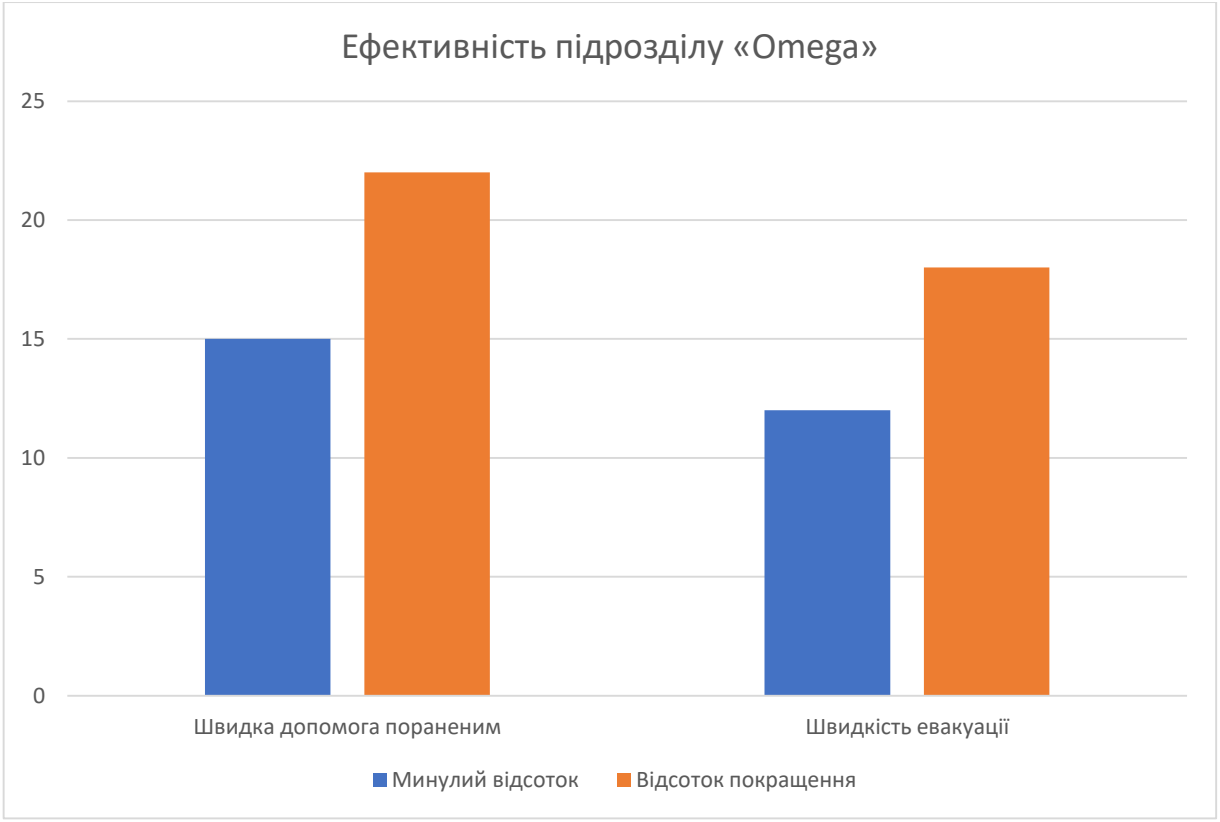

Рисунок 4.24 – Ефективність підрозділу «Omega»

4.5.3 Ефективність підрозділу «Assault»

Підрозділ «Assault» спеціалізується на виконанні завдань на броньованій техніці, включаючи штурм та прикриття піхоти на лінії фронту. Після впровадження системи, підрозділ зазнав значного підвищення ефективності у різних аспектах своєї діяльності.

Ефективність штурмових операцій підвищилася на 30%. Це було досягнуто завдяки більш точному відбору бійців з високими навичками тактики та стрільби, що дозволило краще планувати та виконувати штурмові дії. Впровадження системи підвищило ефективність прикриття піхоти на 28%, завдяки кращій комунікації та координації дій між різними підрозділами, а також оптимізації використання броньованої техніки.

Також зросла маневреності підрозділу на 25%,що за допомогою точнішому відбору водіїв з високими навичками водіння броньованою технікою, що дозволило швидше і безпечніше пересуватися на полі бою (див. рис. 4.25).

Результат інтеграції багатокритеріальної системи – збільшення ефективності підрозділу на 27.67%.

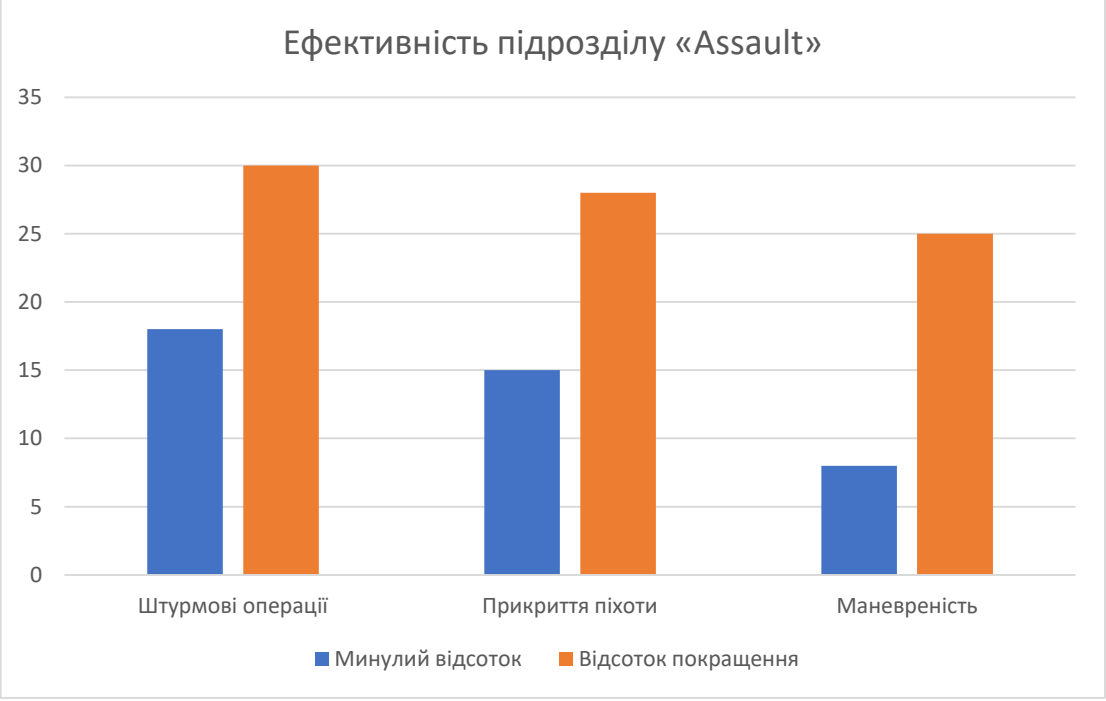

Рисунок 4.25 – Ефективність підрозділу «Assault»

4.5.4 Ефективність підрозділу «Special Force Group»

Підрозділ «FOXHOUND» є елітним військовим підрозділом, спеціалізованим на виконанні надсекретних і складних місій по всьому світу. Після впровадження системи, підрозділ значно підвищив ефективність в різних аспектах своєї діяльності. Впровадження нових технологій і технік комунікації підвищило точність цільового знищення ворожих об'єктів на 30%. Це дозволяє здійснювати більш точні та ефективні удари без зайвих жертв та ризиків для власних сил.

А підбір елітних бійців, що володіють безпілотними апаратами і високоточною зброєю забезпечило підрозділу перевагу в технологічному плануванні і виконанні операцій.

Результат інтеграції багатокритеріальної системи – збільшення ефективності підрозділу на 30%.

4.5.5 Ефективність підрозділу «UAH»

За рахунок відбору членів підрозділу з високим рівнем технічної експертизи та знань двигуна гри Arma 3, час запуску сервера зменшився на 15%. А завдяки поліпшенню комунікацій між членами підрозділу та гравцями спільноти, збільшено рівень взаємодії та підтримки, що призвело до зростання задоволеності гравців на 20%. Також впровадження системи дозволило покращити моніторинг та дотримання правил гри, що зменшило кількість порушень і підвищило загальний рівень дисципліни на сервері на 18% (див. рис. 4.26).

Загальне покращення ефективності підрозділу після інтеграції багатокритеріальної системи становить 17.67%**.**

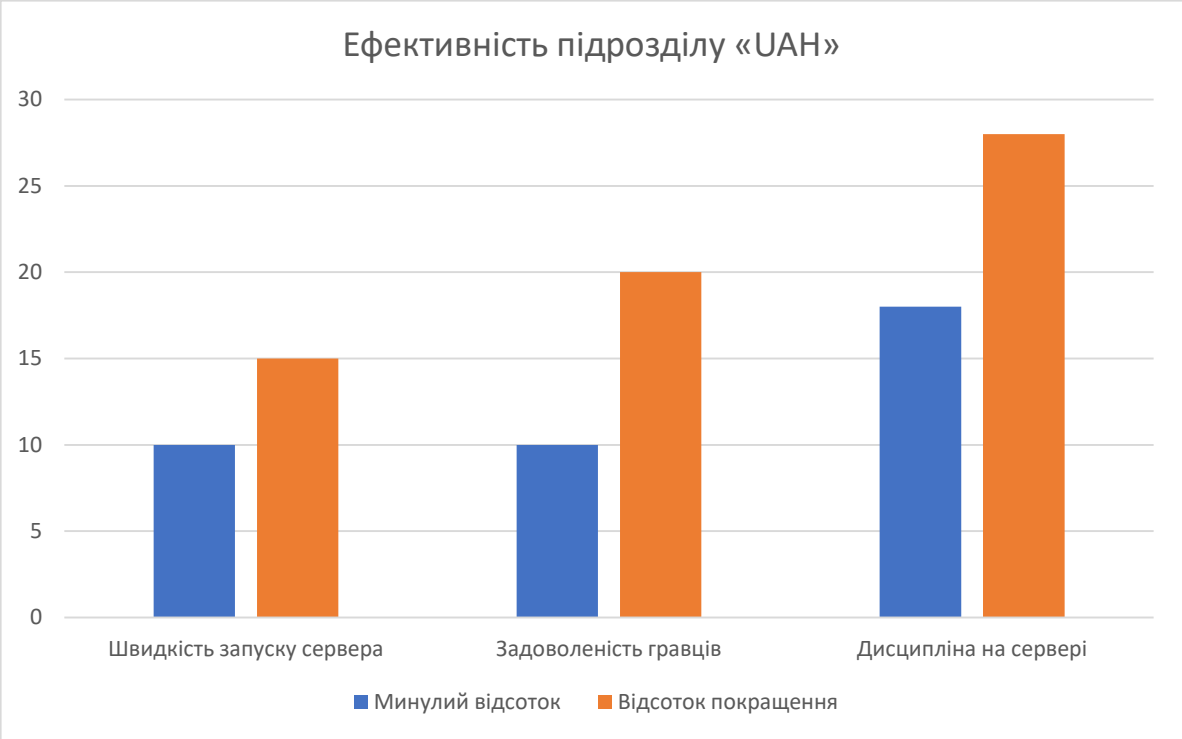

Рисунок 4.26 – Ефективність підрозділу «UAH»

4.5.6 Ефективність підрозділу «OWL»

За рахунок відбору та навчання пілотів з високим рівнем володіння літальною технікою та високими навичками польотів, точність виконання місій зросла на 25%. Використання найсучасніших вертольотів і безпілотних літальних апаратів дозволило підрозділу забезпечити оперативну мобільність та перевагу в будь-якому конфлікті, що призвело до збільшення маневреності на 30%.

Підбір пілотів з гарними комунікативними навичками дозволив зменшити випадки «дружній вогонь» на 40%, що збільшило задоволеність гри та зменшило кількість конфліктних ситуацій (див. рис. 4.27).

Загальне покращення ефективності підрозділу "OWL" після інтеграції багатокритеріальної системи становить 25.67%.

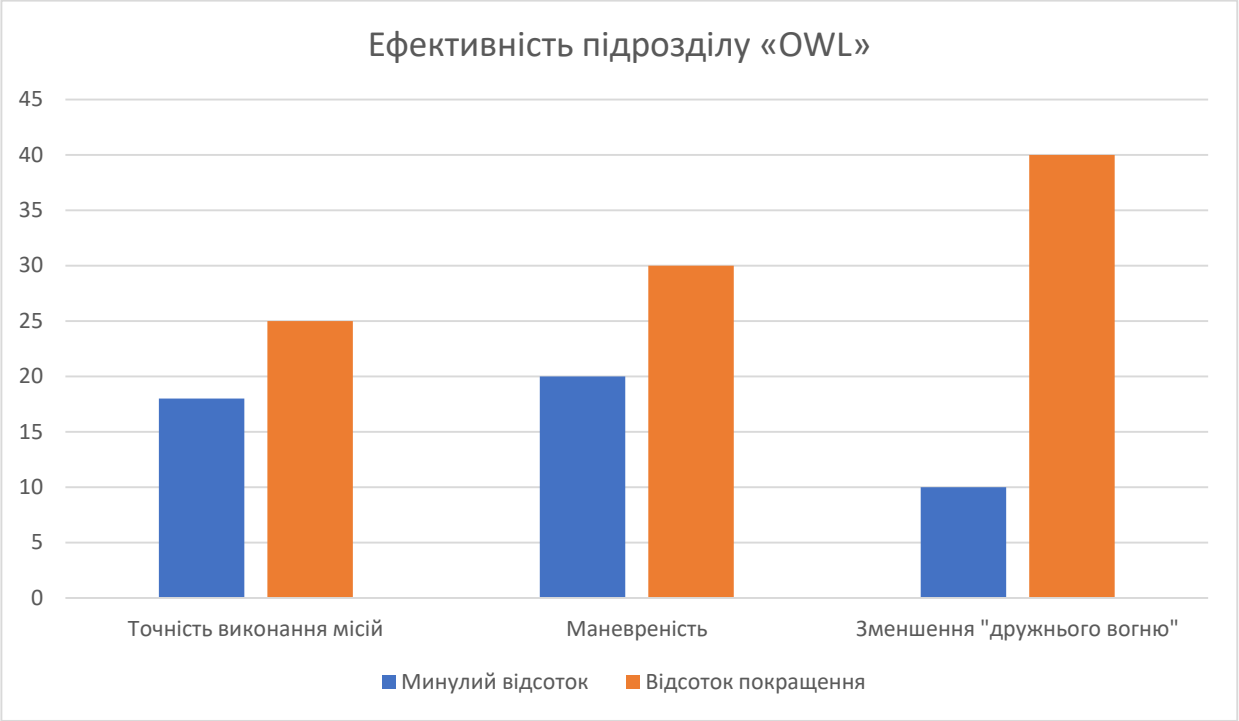

Рисунок 4.27 – Ефективність підрозділу «OWL»

Висновки до розділу 4

Розроблена система була успішно інтегрована та протестована в Discord каналі «Ukraine home», що дозволило отримати цінні дані та відгуки від великої спільноти гравців Arma 3. Для формування ефективних команд було визначено та оцінено ключові навички бійців, та підібрані ваги до вимог підрозділів. А інтеграція системи показала значне підвищення ефективності роботи підрозділів.

ВИСНОВКИ

У даній кваліфікаційній роботі бакалавра було проведено детальний аналіз сучасного стану військових симуляторів, та методів формування команд в цих іграх. Основна увага була приділена розробці системи для планування і оптимізації командних структур на основі аналізу навичок гравця. Для досягнення цієї мети було аналізовано декілька алгоритмів та обрано найбільш підходящий, який забезпечить балансування складу команд з урахуванням таких критеріїв, як навички стрільби, тактика, комунікабельність, лідерство й інші.

В процесі дослідження було проведено порівняльний аналіз різних методів прийняття рішень, зокрема TOPSIS, SAW, AHP, для вибору найбільш ефективного підходу. Розроблена система була протестована на реальних гравцях та віртуальних битвах, що підтвердило її ефективність та можливість застосування в практичних умовах.

Дана кваліфікаційна робота має сприяти підвищенню якості та реалістичності віртуальних битв у військовому симуляторі ARMA 3 та інших подібних іграх. Подібні системи також мають потенціал покращити навчальний процес та тренування військових за допомогою військових симуляторів, забезпечуючи більш ефективне формування команд та розвиток професійних навичок учасників.

Таким чином, дослідження є важливим кроком у напрямку подальшого розвитку військових симуляторів і сприяє їхньому інтегруванню в навчальні центри збройних сил різних країн світу.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ARMA 3 official site URL: https://www.bohemia.net/games/arma3. (дата звернення: 07.05.2024).

2. Squad official site URL: https://www.joinsquad.com/. (дата звернення: 07.07.2024).

3. DCS World official site URL: https://www.digitalcombatsimulator.com/en/. (дата звернення: 10.05.2024).

4. VBS3 official site URL: https://bisimulations.com/products/vbs-builder-edition. (дата звернення: 10.05.2024).

5. U.S. Army Training and Doctrine Command URL: https://www.tradoc.army.mil/. (дата звернення: 10.05.2024).

6. Discord server «Ukraine Home» URL: https://discord.com/invite/HzbSkNaamS. (дата звернення: 10.05.2024).

7. Beckett, S., & Cosh, E. (2005). Team effectiveness in the military: A review of the literature. Human Resource Management Review, 15(3), 307-327. (дата звернення: 10.05.2024).

8. Hwang, C.L.; Yoon, K. (1981). Multiple Attribute Decision Making: Methods and Applications. New York: Springer-Verlag. (дата звернення: 10.05.2024).

9. M. Walczuk and P. Karczmarek, "Fuzzy Analytic Hierarchy Process Based on Graphical Components," *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, Luxembourg, Luxembourg, 2021, pp. 1-7, doi: 10.1109/FUZZ45933.2021.9494576. (дата звернення: 10.05.2024).

10. Forman, Ernest H.; Saul I. Gass (липень 2001). "The analytical hierarchy process—an exposition". Operations Research. 49 (4): 469–487. (дата звернення: 15.05.2024).

11. Hassanzadeh, Hamidreza; Rouhani, Modjtaba (2010). "A multi-objective gravitational search algorithm". In Computational Intelligence, Communication Systems and Networks (CICSyN): 7–12. (дата звернення: 15.05.2024).

12. I. Irvanizam, "Multiple attribute decision making with simple additive weighting approach for selecting the scholarship recipients at Syiah Kuala university," *2017 International Conference on Electrical Engineering and Informatics (ICELTICs)*, Banda Aceh, Indonesia, 2017, pp. 245-250, doi: 10.1109/ICELTICS.2017.8253272. (дата звернення: 15.05.2024).

13. Discord (2024). Discord Developers Documentation URL: https://discord.com/developers/docs/intro. (дата звернення: 06.06.2024).

14. A. K. Gupta, "Framework for the Selection of Sustainable Suppliers using Integrated Compensatory Fuzzy AHP-TOPSIS Multi-criteria Approach," *2022 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, Kuala Lumpur, Malaysia, 2022, pp. 0772-0775, doi: 10.1109/IEEM55944.2022.9989663. (дата звернення: 06.06.2024).

15. Python Software Foundation (2024). Python 3.x документація URL: https://docs.python.org/3/index.html. (дата звернення: 06.06.2024).

16. G. Tianmin, Z. Jiemin and M. Jian, "Research on Webshell Detection Method Based on Machine Learning," *2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE)*, Xiamen, China, 2019, pp. 1391-1394, doi: 10.1109/EITCE47263.2019.9094767. (дата звернення: 06.06.2024).

17. Matplotlib development team. Matplotlib Quick Start Guide URL: https://matplotlib.org/stable/users/explain/quick_start.html. (дата звернення: 06.06.2024).

18. M. Afshari and T. J. Gandomani, "A Typical Practical Team Structure and Setup in Agile Software Development," *2021 7th International Conference on Electrical, Electronics and Information Engineering (ICEEIE)*, Malang, Indonesia, 2021, pp. 483-487, doi: 10.1109/ICEEIE52663.2021.9616743. (дата звернення: 10.06.2024).

19. Nawinda and A. Sofwan, "Fuzzy Simple Additive Weighting for Polytechnic Major Selection Based on High school student Interest and Ability," 2022 International Conference on Information Technology Systems and Innovation (ICITSI),

Bandung, Indonesia, 2022, pp. 353-357, doi: 10.1109/ICITSI56531.2022.9970876. (дата звернення: 10.06.2024).

20. A. Syahputra, Safrizal, K. Puspita, R. Maulida, J. Elnovreny and W. Fahrozi, "Analytic Hierarchy Process (AHP) Modelling For ATM Machine Placement," *2020 8th International Conference on Cyber and IT Service Management (CITSM)*, Pangkal, Indonesia, 2020, pp. 1-4, doi: 10.1109/CITSM50537.2020.9268873. (дата звернення: 10.06.2024).

21. A. V. Mangalan, S. Kuriakose, H. Mohamed and A. Ray, "Optimal location of warehouse using weighted MOORA approach," *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, Chennai, India, 2016, pp. 662-665, doi: 10.1109/ICEEOT.2016.7754764. (дата звернення: 10.06.2024).

22. H. Causa and W. K. M. Brauers, "Location of a seaport by MOORA optimization," *2014 International Conference on Advanced Logistics and Transport (ICALT)*, Hammamet, Tunisia, 2014, pp. 275-280, doi: 10.1109/ICAdLT.2014.6866324. (дата звернення: 10.06.2024).

23. Z. Zhu, J. Pan and X. Liu, "Optimal Ordering and Transportation Model Design Based on BP Neural Network and Entropy Weight TOPSIS," *2021 IEEE Conference on Telecommunications, Optics and Computer Science (TOCS)*, Shenyang, China, 2021, pp. 500-503, doi: 10.1109/TOCS53301.2021.9689006. (дата звернення: 10.06.2024).

24. N. J. McNeese, B. G. Schelble, L. B. Canonico and M. Demir, "Who/What Is My Teammate? Team Composition Considerations in Human–AI Teaming," in *IEEE Transactions on Human-Machine Systems*, vol. 51, no. 4, pp. 288-299, Aug. 2021, doi: 10.1109/THMS.2021.3086018. (дата звернення: 10.06.2024).

25. C. -L. Yang, M. S. Irfana and F. Samopa, "Team building by data clustering with constraints," *Proceedings of the 2014 IEEE 18th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, Hsinchu, Taiwan, 2014, pp. 390-395, doi: 10.1109/CSCWD.2014.6846876. (дата звернення: 12.06.2024).

26. L. C. Silva, T. Poleto, V. D. H. de Carvalho and A. P. C. S. Costa, "Selection of a Business Process Management system: An analysis based on a Multicriteria problem," *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, San Diego, CA, USA, 2014, pp. 295-299, doi: 10.1109/SMC.2014.6973923. (дата звернення: 12.06.2024).

ДОДАТОК А Реалізація багатокритеріальної системи формування збалансованих команд в ігровому симуляторі Arma 3 на мові Python

```
import discord
import numpy as np
import matplotlib.pyplot as plt
from discord.ext import commands

intents = discord.Intents.default()
intents.message_content = True

bot = commands.Bot(command_prefix='!', intents=intents)

weights = {
    'FOXHOUND': [0.7, 0.7, 0.9, 0.8, 0.8, 0.8, 0.6, 0.7, 0.7, 0.9, 0.6, 0.8, 0.5, 0.7, 0.7,
0.9, 0.5, 0.7, 0, 0.7, 0],
    'ВОЛЯ': [0.5, 0.2, 0.7, 0.6, 0.5, 0.6, 0, 0, 0.5, 0, 0.6, 0.8, 0, 0, 0, 0.6, 0.4, 0.7,
0, 0.5, 0],
    'OWL': [0.8, 0.8, 0, 0.8, 0.9, 0.4, 0.5, 0.7, 0.7, 0.7, 0.5, 0, 0.8, 0, 0.7, 0.9, 0.6,
0.5, 0.6, 0.7, 0],
    'OMG': [0.7, 0.7, 0.6, 0.8, 0.9, 0.8, 0.8, 0.8, 0.7, 0.3, 0.9, 0.5, 0.5, 0, 0.5, 0.8,
0.6, 0.7, 0, 0.6, 0],
    'ASL': [0.7, 0.5, 0.8, 0.8, 0.9, 0.8, 0.5, 0.6, 0.7, 0.7, 0.5, 0.7, 0.1, 0, 0.7, 0.8,
0.5, 0.5, 0, 0.3, 0],
    'UAH': [1.0, 0.9, 0, 0, 0.7, 0.8, 0.8, 0.8, 0.3, 0, 0, 0, 0, 0, 0, 0.7, 0.8, 0, 0, 0.8]
}

data = {
    'Enot': [10, 1000, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 9, 10, 10, 10, 24, 10, 10,
10],
    'Kotran': [8, 600, 6, 10, 10, 9, 10, 10, 7, 6, 10, 6, 10, 4, 5, 10, 20, 10, 8, 10],
    'Bogdas': [6, 831, 8, 10, 10, 8, 9, 10, 5, 6, 10, 9, 7, 6, 5, 10, 19, 8, 5, 10],
    'Skelet': [7, 231, 4, 5, 6, 7, 9, 10, 3, 7, 10, 9, 5, 6, 6, 10, 18, 7, 3, 10],
    'Fish': [10, 1000, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 9, 10, 10, 10, 24, 10, 10,
10],
}

def normalize(data):
    keys = list(data.keys())
    matrix = [data[key] for key in keys]
    transposed_matrix = list(map(list, zip(*matrix)))
    normalized_transposed_matrix = []
    for col in transposed_matrix:
        min_val = min(col)
        max_val = max(col)
        normalized_col = [(x - min_val) / (max_val - min_val) if max_val != min_val else 0
for x in col]
        normalized_transposed_matrix.append(normalized_col)
    normalized_matrix = list(map(list, zip(*normalized_transposed_matrix)))
    normalized_data = {keys[i]: normalized_matrix[i] for i in range(len(keys))}
    return normalized_data
norm_data = normalize(data)

def weighted_normalize(norm_data, weight):
    weighted_data = {}
    for key, values in norm_data.items():
        weighted_data[key] = np.multiply(values, weight)
    return weighted_data
```

```
def ideal_anti_ideal(weighted_data):
    ideal = np.max(list(weighted_data.values()), axis=0)
    anti_ideal = np.min(list(weighted_data.values()), axis=0)
    return ideal, anti_ideal

def distances_to_ideal(weighted_data, ideal, anti_ideal):
    distances = {}
    for key, values in weighted_data.items():
        d_plus = np.sqrt(np.sum(np.square(values - ideal)))
        d_minus = np.sqrt(np.sum(np.square(values - anti_ideal)))
        distances[key] = (d_plus, d_minus)
    return distances

def closeness_coefficients(distances):
    coefficients = {}
    for key, (d_plus, d_minus) in distances.items():
        coefficients[key] = d_minus / (d_plus + d_minus)
    return coefficients

results = {}


@bot.command()
async def calculate(ctx):
    global results
    results.clear()
    for unit, weight in weights.items():
        min_length = min(len(next(iter(data.values()))), len(weight))
        norm_data_clipped = {key: values[:min_length] for key, values in norm_data.items()}
        weight_clipped = weight[:min_length]
        weighted_data = weighted_normalize(norm_data_clipped, weight_clipped)
        ideal, anti_ideal = ideal_anti_ideal(weighted_data)
        distances = distances_to_ideal(weighted_data, ideal, anti_ideal)
        coefficients = closeness_coefficients(distances)
        results[unit] = coefficients

    response = ""
    for unit, coefficients in results.items():
        response += f"Підрозділ: {unit}\n"
        for name, coeff in coefficients.items():
            response += f"  Боєць: {name}, Коефіцієнт близькості: {coeff:.4f}\n"

    await ctx.send(response)

@bot.command()
async def plot(ctx):
    global results
    if not results:
        await ctx.send("Спочатку введіть команду `!calculate`.")
        return

    def plot_radar_charts_for_players(results, players):
        num_players = len(players)
        fig, axes = plt.subplots(1, num_players, figsize=(6 * num_players, 6),
subplot_kw=dict(polar=True))

        if num_players == 1:
            axes = [axes]

        for ax, fighter in zip(axes, players):
```

```
            labels = list(results.keys())
            scores = [results[unit][fighter] for unit in labels]

            num_vars = len(labels)
            angles = np.linspace(0, 2 * np.pi, num_vars, endpoint=False).tolist()
            angles += angles[:1]
            scores += scores[:1]

            ax.plot(angles, scores, 'o-', linewidth=2, label=fighter)
            ax.fill(angles, scores, alpha=0.25)
            ax.set_thetagrids(np.degrees(angles[:-1]), labels)
            ax.set_title(f"Радарна діаграма для {fighter}")

        plt.tight_layout()
        filename = 'radar_chart.png'
        plt.savefig(filename)
        plt.close()

        return filename

    players = ['Kotran','Enot','Skelet']
    filename = plot_radar_charts_for_players(results, players)

    await ctx.send(file=discord.File(filename))

bot.run('RARARARAR')
```