МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

БОЖИГОРА Юрій Володимирович

Метод взаємодії користувача з ігровим штучним інтелектом / Method of user interaction with game artificial intelligence

спеціальність 122 - Комп'ютерні науки
освітньо-професійна програма – Комп'ютерні науки

Кваліфікаційна робота

Виконав студент групи КНм-21
Ю. В. Божигора
_____

Науковий керівник:
к.т.н., доцент І. В. Турченко
_____

Кваліфікаційну роботу
допущено до захисту:
«___» _____ 2025 р.
В. о. завідувача кафедри
_____ Н. В. Дзюбановська

Тернопіль – 2025

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «магістр»
спеціальність: 122 «Комп'ютерні науки»
освітньо-професійна програма – Комп'ютерні науки

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Н.М. Васильків
«____» _____20__року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
БОЖИГОРІ Юрію Володимировичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи
Метод взаємодії користувача з ігровим штучним інтелектом / Method of user interaction with game artificial intelligence
керівник роботи к.т.н., доцент І.В. Турченко
затверджені наказом по університету від 20 грудня 2024 року № 938

2. Строк подання студентом кваліфікаційної роботи 1 грудня 2025 р.

3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.

4. Основні питання, які потрібно розробити
- провести аналіз предметної області;
- провести аналіз літературних джерел з досліджуваної тематики і здійснити постановку задачі дослідження;
- розробити метод взаємодії користувача з ігровим штучним інтелектом;
- здійснити програмну реалізацію запропонованого методу.

5. Перелік графічного матеріалу у роботі

- схема алгоритму взаємодії користувача з ігровим штучним інтелектом
- концептуальна схема методу

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 20 грудня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Затвердження теми кваліфікаційної роботи, ознайомлення з літературними джерелами та складання плану роботи	до 01.01. 2025 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.03. 2025 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 20.05.2025 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 28.10. 2025 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 11.11.2025 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів	до 25.11.2025 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту	до 1.12.2025 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 04.12.2025 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії	до 14.12. 2025 р.	

Студент_____ Божигора Ю.В.
(підпис) (прізвище та ініціали)

Керівник кваліфікаційної роботи_____ Турченко І.В.
(підпис) (прізвище та ініціали)

РЕЗЮМЕ

Кваліфікаційна робота на тему «Метод взаємодії користувача з ігровим штучним інтелектом» на здобуття ступеня вищої освіти «Магістр» зі спеціальності 122 «Комп'ютерні науки» освітньо-професійної програми «Комп'ютерні науки» написана обсягом 73 сторінки і містить 5 рисунків, 2 таблиці, 5 додатків та 36 використаних джерел.

Метою кваліфікаційної роботи є розробка методу взаємодії користувача з ігровим штучним інтелектом, який забезпечує адаптацію поведінки ігрових агентів до дій, стилю гри та індивідуальних особливостей користувача.

Методи досліджень: системний аналіз (вивчення предметної області та архітектурних підходів), порівняльний аналіз, моделювання, методи машинного навчання, методи теоретичного узагальнення та структурування.

Результати дослідження: запропоновано метод взаємодії користувача з ігровим штучним інтелектом, який забезпечує підвищення адаптивності поведінки агентів на основі контекстної обробки дій користувача та персоналізацію ігрового процесу.

Результати роботи можуть бути використані при розробці сучасних ігрових середовищ для підвищення реалістичності, динамічності та індивідуалізації ігрового процесу.

Ключові слова: ШТУЧНИЙ ІНТЕРНЕТ, НЕІГРОВІ ПЕРСОНАЖІ, ІГРОВІ ПЕРСОНАЖІ, ІГРОВИЙ АГЕНТ, МЕТОДИ МАШИННОГО НАВЧАННЯ, КЛАСТЕРИЗАЦІЯ ПРОФІЛІВ КОРИСТУВАЧІВ, НАВЧАННЯ З ПІДКРІПЛЕННЯМ, ДИНАМІЧНА СКЛАДНІСТЬ ГРИ

ABSTRACT

Qualification work on the topic « Method of user interaction with game artificial intelligence» for the Master`s degree on the specialty 122 «Computer Science» of the educational and professional program «Computer Science» is written on 73 pages and contains 5 figures, 2 tables, 5 annexes and 36 used sources.

The purpose of the qualification work is to develop a method of user interaction with game artificial intelligence that ensures the adaptation of game agents' behavior to the user's actions, play style, and individual characteristics.

Research methods include system analysis (study of the subject domain and architectural approaches), comparative analysis, modeling, machine learning methods, and methods of theoretical generalization and structuring.

Research results: a method of user interaction with game artificial intelligence is proposed, which increases the adaptability of agent behavior based on contextual processing of user actions and enables personalization of the gameplay.

The results of the work can be applied in the development of modern game environments to enhance the realism, dynamism, and individualization of the gameplay.

Keywords: ARTIFICIAL INTELLIGENCE, NON-PLAYER CHARACTERS, PLAYER CHARACTERS, GAME AGENT, MACHINE LEARNING METHODS, USER PROFILE CLUSTERING, REINFORCEMENT LEARNING, DYNAMIC GAME DIFFICULTY.

ЗМІСТ

ВСТУП

Актуальність теми. Сучасна індустрія відеоігор є однією з найбільш динамічно зростаючих галузей цифрової економіки, що об’єднує мільярди користувачів по всьому світу та стимулює розвиток передових технологій штучного інтелекту (ШІ). Ефективна взаємодія користувача з ігровим штучним інтелектом відіграє ключову роль у формуванні ігрового досвіду, рівня залученості гравця, адаптивності ігрового процесу та персоналізації сценаріїв поведінки неігрових персонажів (англ. Non-Player Characters (NPC)). Сучасні вимоги до якості інтерактивних систем вимагають не лише високої продуктивності алгоритмів, але й інтуїтивності, передбачуваності та емоційного відгуку ігрового ШІ, що напряму впливає на задоволеність користувачів та комерційний успіх продукту.

Зі зростанням складності ігор і розширенням можливостей апаратного забезпечення виникає потреба у розробці нових методів взаємодії, які здатні забезпечити гнучку комунікацію між користувачем та інтелектуальними агентами, адаптуючись до індивідуальних стилів поведінки та стратегій гравців. Традиційні підходи до керування ігровим ШІ, які базуються на жорстко визначених правилах та шаблонах, дедалі більше виявляють свої обмеження щодо реалістичності, адаптивності та здатності до самонавчання.

Враховуючи швидкий розвиток технологій машинного навчання та глибоких нейронних мереж, з’являється потенціал для створення нових, більш ефективних методів взаємодії, що дозволяють інтегрувати людський досвід, поведінкові моделі та інтелектуальні адаптивні механізми у внутрішню логіку ігор. Це сприяє підвищенню ігрової складності, покращенню якості штучного інтелекту та створенню більш захоплюючого і персоналізованого ігрового середовища.

Таким чином, розробка та дослідження методів взаємодії користувача з ігровим штучним інтелектом є актуальним напрямом, що має як наукову цінність, так і практичне значення для ігрової індустрії, симуляційних систем, освітніх технологій та інших інтерактивних середовищ.

Мета і завдання дослідження. Метою кваліфікаційної роботи є розробка методу взаємодії користувача з ігровим штучним інтелектом, який забезпечує адаптацію поведінки ігрових агентів до дій, стилю гри та індивідуальних особливостей користувача.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

- провести аналіз предметної області та літературних джерел з досліджуваної тематики і здійснити постановку задачі дослідження;
- розробити метод взаємодії користувача з ігровим штучним інтелектом;
- здійснити програмну реалізацію запропонованого методу.

Об’єктом дослідження є процес взаємодії користувача з ігровим штучним інтелектом.

Предметом дослідження є методи та моделі адаптивної поведінки ігрових агентів штучного інтелекту в контексті взаємодії користувача з ігровим середовищем.

Методи дослідження: системний аналіз (вивчення предметної області та архітектурних підходів), порівняльний аналіз, моделювання (створення формалізованих моделей), методи машинного навчання, а також методи теоретичного узагальнення та структурування.

Результати дослідження: розроблено метод взаємодії користувача з ігровим штучним інтелектом, який забезпечує підвищення адаптивності поведінки агентів на основі контекстної обробки дій користувача.

Практичне значення отриманих результатів роботи полягає у можливості використання розробленого методу при створенні ігор різних жанрів – від симуляторів до екшен-проєктів – для підвищення реалістичності та варіативності поведінки штучного інтелекту, використання запропонованого підходу у сучасних ігрових середовищах для підвищення реалістичності, динамічності та індивідуалізації ігрового процесу.

Публікації та апробація кваліфікаційної роботи. Результати дослідження опубліковані у матеріалах матеріалах міжнародної науково-теоретичної конференції «Наукові інновації: теоретичні підходи та практичний вплив», 8-10

грудня 2025 року, Неаполь, Італія і міжнародної науково-практичної конференції «Дослідження у сфері науки, технологій та економіки», 10-12 грудня 2025 року, Люксембург, Люксембург (Додаток Д).

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1 Підходи, моделі та виклики взаємодії користувача з ігровим штучним інтелектом

Взаємодія користувача з ігровим штучним інтелектом (ІШІ) охоплює широкий спектр механізмів впливу, коли гравець і агент ШІ здійснюють взаємний вплив один на одного через інтерфейс гри, систему правил та поведінкові патерни самої системи. Наукові дослідження засвідчують, що відеоігри виступають оптимальною експериментальною платформою для дослідження взаємодії людини й штучного інтелекту, оскільки вони дають змогу апробувати різноманітні метафори взаємодії, моделі агентів із здатністю до навчання і сценарії із зворотним зв’язком гравця.

Під ігровим штучним інтелектом (ІШІ, англ. Game AI) розуміють сукупність алгоритмів, методів та архітектур, спрямованих на забезпечення поведінки агентів (неігрових персонажів, англ. NPC – Non-Player Characters) у віртуальних середовищах, які або імітують, або адаптивно реагують на дії користувача. Основні завдання ІШІ включають просторову навігацію, прийняття рішень, формування та реалізацію планів дій, адаптацію поведінки з урахуванням контексту, взаємодію з користувачем, а також динамічне реагування на ігрову активність гравця.

Ігровий агент – це елемент ігрової системи, що функціонує як самостійний суб’єкт взаємодії, здатний аналізувати стан середовища, реагувати на дії користувача та адаптувати свою поведінку відповідно до внутрішніх правил або моделей штучного інтелекту.

В ігровому контексті, особливо в комп’ютерних та настільних рольових іграх, ігровий персонаж (англ. player character (PC)) – це персонаж, дії, поведінка та рішення якого визначаються користувачем (гравцем) через ігровий інтерфейс.

На відміну від класичних прикладних систем, де ШІ виконує задачі оптимізації або автоматизації, у відеоіграх він реалізує:

– поведінкову адаптацію – реагування на зміни середовища та дії гравця;

– тактичне та стратегічне планування – визначення оптимальних дій у контексті цілей агента;

– генерацію природної взаємодії – діалог, комунікація, командна робота;

– підтримку наративу – створення динамічних подій відповідно до історії.

Особливість ігрового ШІ полягає в тому, що його цілі часто не збігаються з оптимальністю в класичному розумінні. Наприклад, у багатьох жанрах (RPG, Action, Adventure) ШІ не повинен «перемагати» гравця максимально ефективно, а навпаки – підтримувати певний баланс виклику, забезпечувати драматизм, створювати відчуття реалістичності та логічності світу [1].

Взаємодія користувача з ігровим ШІ може відбуватися на кількох рівнях:

– фізичний рівень – зіткнення, переслідування, динамічні бої;

– тактичний рівень – планування руху, командна координація;

– наративний рівень – діалоги з NPC, сюжетні взаємодії;

– соціальний рівень – співпраця з агентами в ролі союзників;

– психологічний рівень – відчуття емпатії, довіри, ворожості.

Останні дослідження підкреслюють, що користувацька взаємодія не може бути оцінена лише технічними показниками – необхідні психологічні, UX-орієнтовані та поведінкові метрики, що оцінюють емоційний досвід гравця [2].

Сучасні підходи до розробки ігрового штучного інтелекту вирізняються розмаїттям теоретичних і практичних основ:

– системи на основі правил (Rule-based systems) – передбачають, що агенти виконують дії відповідно до наперед визначених жорстких наборів правил;

– скінченні автомати станів (Finite State Machines, FSM) – функціонування агента базується на переході між різними станами залежно від конкретних умов (наприклад, такі стани як «патрулювати», «полювати», «уникати»);

– системи на основі корисності (Utility-based systems) – кожній можливій дії агента призначається числова вага (utility), а вибір дії здійснюється на основі максимізації значення цієї корисності;

– навчання з підкріпленням (Reinforcement Learning, RL) – агент формує свою поведінку на основі процесу отримання винагород чи покарань, поступово

оптимізуючи довгострокову функцію винагороди;

– нейромережеві підходи та гібридні моделі – агент використовує штучні нейронні мережі або комбінує різноманітні алгоритми, забезпечуючи адаптацію до динаміки та індивідуальних стилів гравця;

– моделі на основі великих мовних моделей (LLM-Game Agents) – агенти, які використовують можливості обробки природної мови, пам'яті, логічних операцій і генеративних стратегій, що дозволяє їм здійснювати взаємодію з гравцем на рівні діалогів, стратегічних підказок і планування дій. Наприклад, обсяг сучасних досліджень у цій галузі відображено в праці [3], де охарактеризовано архітектуру таких агентів та їхню адаптацію до різних ігрових жанрів.

Під час розгляду підходів до моделювання гравця (англ. player modeling) для забезпечення ефективної взаємодії між людиною та агентом штучного інтелекту виникає потреба у глибокому розумінні гравця – з урахуванням його ігрового стилю, поведінкових особливостей, уподобань і переваг. Саме ці завдання вирішує підхід моделювання гравця – напрям, що спрямований на розробку обчислювальних моделей, які відображають когнітивні, поведінкові та емоційні характеристики користувачів [4].

Для того, щоб система могла адаптувати гру до вподобань користувача, змінюючи наратив та інший контент, від нього необхідно отримати певну інформацію. Ця інформація отримується шляхом аналізу когнітивних, афективних та поведінкових моделей гравців, щоб створити модель, яка виражає їхню особистість, наміри та характеристики. Моделювання гравців (рисунок 1.1) [5], із загальної точки зору, є вивченням цих даних, яке дозволяє сприймати, записувати та аналізувати дії гравця, виявляючи певні закономірності в його взаємодії з віртуальним світом, для створення статистичної моделі. Ця модель потім використовується для створення досвіду [5].

Це концептуальна схема архітектури взаємодії гравця з ігровим штучним інтелектом, орієнтована на моделювання гравця та адаптивне керування сюжетом (drama management).

Коротко по елементах схеми:

- гравець – взаємодіє з грою через ігровий рушій;
- ігровий рушій – центральний компонент, який обробляє введення гравця та оновлює стан гри;
- стан гри – включає:
    1) фізичний стан (позиції, об’єкти, події);
    2) стан сюжету;
    3) історію (накопичені події та дії);
- слід гравця – журнал/лог дій і поведінки гравця, що передається на аналіз;
- моделювання гравця – аналізує слід гравця та формує модель гравця (стиль гри, уподобання, рівень навичок тощо);
- менеджер драми — використовує модель гравця та поточний стан гри для прийняття рішень щодо адаптації ігрового процесу;
- дії менеджера драми – впливи на гру (зміна складності, подій, поведінки NPC, сюжетних гілок).

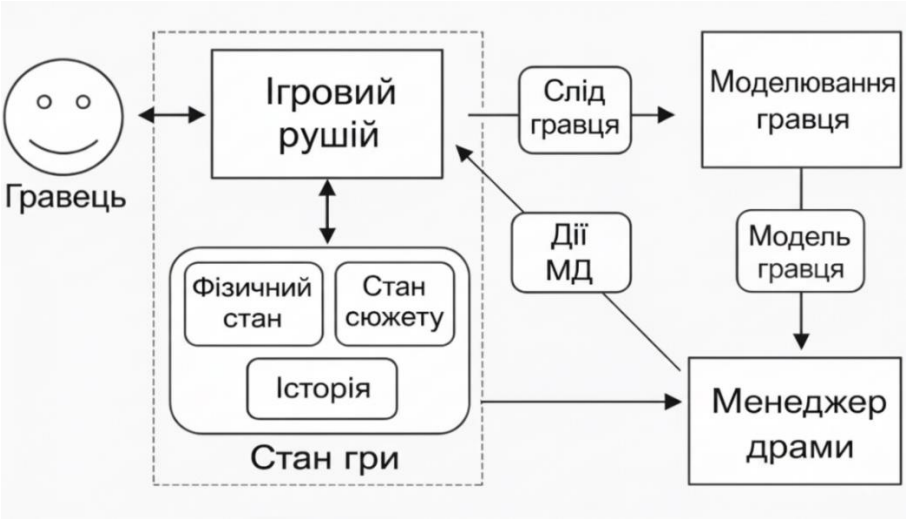


Рисунок 1.1 – Схема взаємодії користувача з ігровим ШІ

Моделювання гравця – це побудова моделі гравця на основі:

- стилю гри;
- поведінкових патернів;
- емоційного стану;
- рівня майстерності.

Основними компонентами моделювання гравця є такі:

– джерела даних: до аналізу залучаються дані про поведінку у грі (зокрема натискання певних кнопок, обрані маршрути, ухвалені рішення), метадані (наприклад, тривалість ігор, пройдені рівні), а також демографічна інформація;

– типи моделей:

4) статичні моделі (Profiling) – передбачають класифікацію гравця на підставі фіксованих характеристик (наприклад, тип гравця: дослідник, досягальник, соціальний тощо);

5) динамічні моделі – орієнтовані на відстеження змін поведінки протягом ігрової сесії (адаптація, навчання, зміна емоційних станів);

– цілі моделювання: моделі гравця застосовуються для персоналізації сценаріїв, динамічного регулювання складності (англ. DDA – dynamic difficulty adjustment), а також для прогнозування ймовірності відтоку гравця (churn prediction).

Зокрема, у дослідженні [6] було здійснено систематичний огляд методів, що ґрунтуються на аналізі статистичних даних, застосуванні алгоритмів машинного навчання та використанні гібридних моделей.

Після розгляду ігрового штучного інтелекту та технік моделювання гравця доцільно перейти до вивчення основних типів взаємодії. Згідно з оглядом, наведеним у праці у праці [7], ігри виступають експериментальним майданчиком для дослідження взаємодії людини з ШІ, і авторам вдалося виокремити кілька провідних метафор такої взаємодії.

Підходи, моделі чи типи сценаріїв, за якими може будуватися взаємодія між користувачем та ігровим агентом ШІ, відповідно до його ролі та функції, можна узагальнити таким чином [8]:

– ШІ як опонент (AI as Opponent) – агент виступає супротивником для користувача, адаптуючи власну поведінку відповідно до ігрового стилю гравця;

– ШІ як співучасник або напарник (AI as Teammate) – агент бере участь у спільному виконанні завдань разом із користувачем;

– ШІ як наставник чи коуч (AI as Mentor / Coach) – агент надає користувачу

поради, підказки, здійснює аналіз помилок і пропонує навчальні рекомендації;

– взаємодія зі спільною ініціативою (Mixed-Initiative Interaction) – агент і користувач спільно приймають рішення, при цьому агент може пропонувати певні варіанти дій, а користувач має можливість їх коригувати;

– адаптивна взаємодія (Adaptive Interaction) – агент змінює свою поведінку, рівень складності або стиль відповідно до дій гравця, що дає змогу підтримувати збалансований ігровий досвід.

Окрім цього, сучасні підходи, що ґрунтуються на використанні великих мовних моделей (англ. Large Language Models – LLM), розширюють можливості агентів у напрямку комунікації природною мовою, а також генерації стратегічних порад, діалогів та інтерактивних підказок. Зокрема, LLM-агенти здатні поєднувати механізми роботи з пам’яттю, логічного виведення (reasoning) та обробки вхідних і вихідних даних, що забезпечує їм підвищену гнучкість під час реагування на дії користувача [3].

При розробці методів взаємодії людини та агента ШІ слід враховувати такі ключові виклики:

- прозорість і інтерпретованість: агент повинен бути здатен обґрунтовувати свої дії та рішення, оскільки відсутність пояснень призводить до зниження довіри з боку гравця;
- баланс між автономністю та контролем: надмірна автономія агента може зменшувати залученість користувача, тоді як занадто велика залежність від дій гравця не забезпечує додаткових переваг;
- адаптивність до різних ігрових стилів: агенти мають враховувати різноманіття стилів гравців (наприклад, гравці-дослідники) і підлаштовувати свою поведінку до них;
- затримки й ефективність у веб-середовищах: для веб-додатків важливим є забезпечення високої швидкості реакції агента та наявність достатніх ресурсів для обробки даних;
- етичні аспекти та питання користувацького досвіду: необхідно уникати створення відчуття маніпуляції й дотримуватися приватності даних, що

відображають поведінку користувача;

– специфічні виклики підходів з навчанням з підкріпленням: серед основних проблем виділяють ефективність використання вибірки (sample efficiency), ризик потрапляння у локальні мінімуми функції винагороди й труднощі з генералізацією на нові сценарії;

– тенденція до використання LLM-агентів і мультижанрових моделей: зростає зацікавленість до агентів, які здатні функціонувати у різних жанрах і форматах ігрового процесу (зокрема, пригодницькі, симуляційні та командні ігри) [3].

Більшість сучасних моделей ігрового ШІ не враховують повною мірою індивідуальні особливості гравця. Проблемами залишаються:

– відсутність моделей, що безперервно аналізують стиль гри;
– складність прогнозування поведінки гравця в реальному часі;
– обмеженість систем адаптивної складності;
– недостатня інтеграція глибинних поведінкових моделей;
– проблема інтерпретованості рішень ШІ для гравця.

Актуальним завданням є розробка методу, який поєднує швидку реакцію, контекстну інтерпретацію та адаптацію до індивідуального стилю гри користувача.

1.2 Аналіз літературних джерел з досліджуваної тематики

Сучасні дослідження у сфері взаємодії людини з ігровим штучним інтелектом (ШІ) поєднують підходи з галузей взаємодії людини з комп’ютером, ігровий штучний інтелект та моделювання гравця, зосереджуючись на підвищенні рівня адаптивності, пояснюваності й залученості користувача.

У праці [7] розглянуто феномен «ШІ як гра», де нейромережеві агенти виступають не просто супротивниками, а учасниками спільного досвіду. Автори проаналізували, як ігри з нейронними агентами формують новий рівень взаємодії між гравцем і ШІ, зокрема через довіру, передбачуваність дій і комунікацію. Ця робота є важливою теоретичною базою для формування принципів інтерфейсної

взаємодії користувача з інтелектуальним агентом.

G. N. Yannakakis та співавт. у своїй оглядовій праці [4] систематизували підходи до моделювання гравця – створення обчислювальних моделей, що описують когнітивні, емоційні й поведінкові аспекти гравця. Дослідження окреслює методи побудови профілів гравців на основі машинного навчання, що дозволяє ШІ-персонажам адаптувати власну поведінку під індивідуальні особливості користувача.

Магістерська робота J. Lundström [9] емпірично вивчає сприйняття користувачами ШІ-компаньйонів у відеоіграх. Автор аналізує UX-фактори, що впливають на комфорт та довіру гравців до ШІ-партнерів, і показує, як рівень автономності агента впливає на досвід гравця. Ці результати мають безпосереднє значення для проєктування моделей співпраці гравця та агента.

Дослідження [10] зосереджено на концепції проєктування зі змішаною ініціативою, у межах якої людина й ШІ спільно приймають рішення в процесі гри. Автор пропонує принципи побудови ігрових систем, у яких агент не лише реагує, а й ініціює дії, що сприяє формуванню більш природної та продуктивної взаємодії.

У магістерській роботі Henrique M. de C. L. Fernandes [5] представлено підхід до моделювання гравця у рольових іграх на прикладі системи Radiant AI від Bethesda. Автор розробив метод покращення поведінкових моделей неігрових персонажів (NPC), що дозволяє підвищити рівень реалістичності взаємодії. Практична частина дослідження демонструє, як адаптаційні алгоритми можуть бути інтегровані у комерційні ігрові рушії.

Серед українських джерел слід відзначити магістерську роботу В. В. Сироти [12], присвячену впровадженню ШІ у мобільні ігри. У дослідженні висвітлено підходи до навчання агентів та аналізу впливу інтелектуальних компонентів на користувацький досвід у середовищі Unity. Це є цінним локальним прикладом прикладного впровадження ігрового ШІ.

Узагальнюючи, аналіз літератури показує, що більшість досліджень зосереджено на адаптивності поведінки ігрового агента, довірі користувача та моделюванні гравця. Однак менш розвиненими залишаються питання гібридних

методів взаємодії, де агент може як ініціювати, так і слідувати дії користувача, та інтеграції LLM-компонентів у реальний ігровий процес, що й визначає актуальність подальшої розробки нового методу взаємодії користувача з ігровим ШІ.

У людино–ШІ взаємодії ігрове середовище стає одним із ключових полігонів для експериментів, оскільки воно комбінує механіку, наратив і активну участь користувача. У праці [7] проведено систематичне дослідження 38 ігор з нейромережевими агентами, виділено домінантні метафори взаємодії («ШІ як суперник», «ШІ як напарник», «ШІ як ментор», «Спільна ініціатива») та застосовано керівні принципи взаємодії людини з ШІ для контексту ігор.

Робота [13] охоплює розвиток агентів глибокого навчання з підкріпленням (англ. Deep Reinforcement Learning, DRL) у відеоіграх – від аркадних жанрів до стратегій реального часу (RTS), із виокремленням ключових викликів, зокрема ефективності використання вибірки (sample efficiency), здатності до узагальнення (generalization) та багатоагентного навчання (multi-agent learning).

У статті [14] досліджено адаптивне моделювання користувача через поведінкові й фізіологічні дані, що дозволяє змінювати контент гри під стиль гравця.

Ці дослідження показують, що взаємодія користувача з ІШІ охоплює не лише технічні алгоритми, але й UX-аспекти, моделювання гравця, адаптацію та пояснюваність. Проте більшість робіт фокусується або на агентах (поведінка, навчання) чи на моделюванні гравця, але мало на концепції взаємодії (людина-агент) як методу. Це створює простір для розробки нового методу взаємодії користувача з ігровим ШІ.

Моделювання гравця – це область, що займається створенням моделей гравця для адаптації ігрових систем. У праці [14] використано комбінацію поведінкових і фізіологічних даних для побудови моделі гравця у гоночній грі: модель ідентифікує сильні й слабкі сторони гравця та підлаштовує контент.

У оглядах також виділяють типології гравців (наприклад, Bartle, BrainHex, HEXAD), що надає структурний підхід до класифікації. Ці моделі корисні для розуміння стилю користувача, що є важливим для методу взаємодії.

Моделювання гравця дає можливість персоналізувати взаємодію з агентом: наявність даних про гравця дозволяє агента адаптувати поведінку, пропонувати більш релевантні дії чи підказки. Однак багато моделей залишаються статичними або не враховують реального часу, що є обмеженням.

Методи формування поведінки агентів охоплюють широкий спектр підходів – від систем, заснованих на правилах (rule-based systems), та кінцевих автоматів (finite state machines, FSM) до сучасних методів глибокого навчання з підкріпленням (deep reinforcement learning, DRL) і гібридних моделей. У роботі Shao et al. [14] розглянуто основні класи методів DRL, зокрема алгоритми на основі градієнта політики (policy gradient), оцінювання функції цінності (value-based) та модельно-орієнтовані підходи (model-based). Автори також наголошують на ключових труднощах їх застосування в ігрових середовищах, серед яких проблеми балансу між дослідженням і використанням, узагальнення поведінки та багатоагентна взаємодія.

Окрім цього, у дослідженні [15] розглянуто методи self-play у навчанні з підкріпленням, які можуть бути застосовані до ігрових агентів, що навчаються взаємодіяти з гравцем або з іншими агентами.

Сучасні дослідження засвідчують активне впровадження методів навчання з підкріпленням в ігровій індустрії, водночас акцентуючи на проблемах їх масштабованості та практичної інтеграції [16]. У роботах з багатоагентних систем ігрові середовища розглядаються як платформа для аналізу кооперативної та конкурентної взаємодії агентів на основі поєднання навчання з підкріпленням і теорії ігор [17].

Дослідження взаємодії гравця з ігровим ШІ підкреслюють важливість формування коректних ментальних моделей штучних агентів для підвищення довіри та якості користувацького досвіду [18].

Водночас сучасні підходи відзначають зростаючу роль ШІ у створенні емоційного зв’язку між гравцем і ігровими персонажами [19]. [20] аналізує вплив шаблонів ШІ на ігровий процес та досвід гравця, що може бути корисним у розділі про UX-орієнтовані та поведінкові метрики.

Методи адаптації дозволяють агенту змінювати поведінку на основі даних користувача чи середовища. Проте виклики залишаються: пояснюваність, продуктивність у реальному часі, врахування стилю гравця, перенесення на веб-середовище.

Аналіз сильних і слабких сторін існуючих методів представлено нижче.

Переваги:

– роботи з моделювання гравця дають основу персоналізації;

– агентські методи DRL забезпечують високу автономність і складну поведінку;

– дослідження взаємодії людина-ШІ підкреслюють UX-аспекти, що часто ігноруються.

Недоліки / обмеження:

– більшість моделей гравців є статичними, не враховують змін протягом сесії;

– агентські моделі часто не враховують індивідуальних особливостей користувача або не надають пояснень своїх дій (проблема пояснюваності);

– веб-середовище або багатокористувацькі середовища мало описані в контексті взаємодії користувача з ігровим агентом;

– поєднання моделі гравця + адаптивного агента + UX-інтерфейсу – рідкісне; методологічний розрив між технічною реалізацією й UX-взаємодією.

Узагальнена таблиця 2.1 порівняння основних підходів до взаємодії користувача з ігровим штучним інтелектом (ШІ) представлена нижче [21].

Таблиця 2.1 – Порівняння підходів до взаємодії користувача з ігровим ШІ

№	Підхід / Метод	Основна ідея	Переваги	Обмеження / Недоліки
1	Rule-based (FSM, Behavior Trees)	Поведінка агента визначається наперед заданими правилами, сценаріями або станами	Простота реалізації; передбачуваність; контрольована логіка	Обмежена гнучкість; відсутність адаптації до користувача; складність масштабування

2	Reinforcement Learning (RL)	Агент навчається через винагороди й покарання; формує політику поведінки	Висока автономність; здатність навчатися у складних середовищах	Високі обчислювальні витрати; потреба у великій кількості епізодів; слабка пояснюваність
3	Imitation Learning (IL)	Агент навчається імітувати дії користувача або експерта	Ефективність при малих даних; природність поведінки	Залежність від якості демонстрацій; слабке узагальнення
4	Hybrid Approaches (RL + Rules / Neuro-symbolic)	Поєднання навчання з правилами або сценаріями	Компроміс між адаптивністю і контрольованістю	Підвищена складність реалізації; потреба в налаштуванні

Продовження таблиці 2.1

5	User Modeling (Behavioral / Physiological)	Аналіз стилю, дій і стану гравця для адаптації контенту	Персоналізація ігрового процесу; покращення UX	Потреба в даних користувача; обмеження в реальному часі
6	Player Typologies (Bartle, BrainHex, HEXAD)	Класифікація гравців за типами мотивації та стилем гри	Простота інтеграції; концептуальна зрозумілість	Не враховує динамічну зміну стилю гравця; суб'єктивність
7	Human-AI Mixed-Initiative Interaction	Людина й агент виступають як співтворці – обидві сторони ініціюють дії	Підвищена залученість користувача; інтерактивність	Висока складність реалізації; потреба у пояснюваному ШІ
8	Explainable AI (XAI) for Games	Агент пояснює свої рішення для підвищення довіри користувача	Прозорість, навчальний ефект, краща взаємодія	Складність формалізації пояснень; вплив на продуктивність
9	Self-Play RL	Агент навчається, змагаючись сам із собою або з іншими агентами	Стійке навчання без людського втручання; розвиток складних стратегій	Не враховує людську поведінку; низька інтерпретованість

Порівняння показує, що класичні підходи на основі правил (rule-based) та кінцевих автоматів (FSM) характеризуються низькою адаптивністю, тоді як методи навчання з підкріпленням (RL) та навчання з наслідуванням (IL) забезпечують динамічне навчання, проте потребують значних обчислювальних ресурсів. Моделі користувача та типології гравців дозволяють персоналізувати ігровий досвід, однак здебільшого залишаються статичними. Водночас відсутність механізмів

пояснюваності в більшості адаптивних систем ускладнює інтерпретацію рішень ігрового штучного інтелекту з боку користувача.

Крім того, ізольоване застосування окремих підходів не забезпечує комплексного врахування контексту взаємодії, поведінкових особливостей гравця та динаміки ігрового процесу.

Найбільш перспективним напрямом для нового методу є поєднання адаптивного агента (RL/IL) із моделлю користувача та пояснюваним UX-інтерфейсом – саме цей підхід становить основу розробленого в роботі методу взаємодії користувача з ігровим штучним інтелектом.

1.3 Постановка задачі дослідження

На підставі аналізу предметної області та літературних джерел можна виокремити базові вимоги, яким має відповідати метод взаємодії [8]:

1) адаптивність: агент повинен динамічно змінювати свою поведінку відповідно до актуальної моделі гравця;

2) прозорість: агент забезпечує зрозумілість і пояснюваність власних дій для користувача;

3) збалансованість ролей: важливим є збереження контролю з боку користувача, при цьому роль агента має залишатися допоміжною, а не домінуючою;

4) ефективність: у веб-реалізаціях необхідно досягати мінімальних затримок у роботі агента та високої продуктивності;

5) розширюваність: система має передбачати можливість інтеграції нових моделей поведінки, сценаріїв чи ігрових жанрів;

6) орієнтація на користувацький досвід (UX): інтерфейс взаємодії повинен бути інтуїтивно зрозумілим та надавати можливість дослідження й експериментування з агентом.

Рисунок 1.2 демонструє базові вимоги до методу взаємодії користувача з ігровим ШІ.

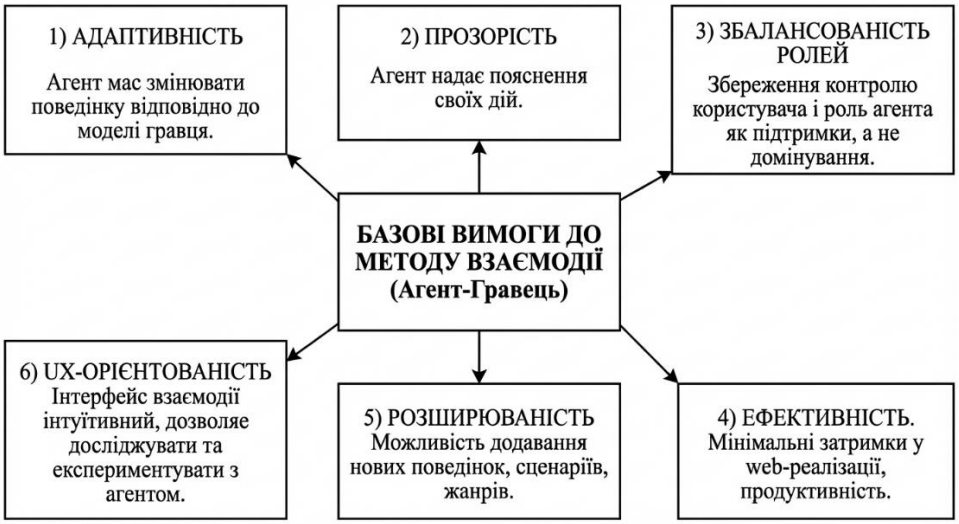


Рисунок 1.2 – Базові вимоги до методу взаємодії користувача з ігровим ШІ

Відповідно до цього метою кваліфікаційної роботи є розробка методу взаємодії користувача з ігровим штучним інтелектом, який забезпечує контекстно-залежну адаптацію поведінки ігрових агентів на основі аналізу дій користувача, стилю його гри та індивідуальних поведінкових характеристик.

Для досягнення поставленої мети необхідно виконати завдання:

- провести аналіз предметної області та літературних джерел з досліджуваної тематики і здійснити постановку задачі дослідження;
- розробити метод взаємодії користувача з ігровим штучним інтелектом;
- розробити концепцію методу адаптивної взаємодії гравця з NPC на основі аналізу дій, стилю гри та індивідуальних особливостей;
- запропонувати підхід до автоматизованого моделювання гравців для персоналізації складності та стратегії взаємодії з ШІ;
- обґрунтувати використання адаптивного ігрового агента з навчанням з підкріпленням для оптимізації досвіду користувача;
- реалізувати прототип ігрової сцени з механіками взаємодії та збором даних для подальшої адаптації середовища;
- створити інтегровану архітектуру, що забезпечує персоналізацію та безперервну адаптацію між користувачем та ігровим ШІ;

– реалізувати програмну систему, що демонструє роботу запропонованого методу.

Висновки до розділу 1

1. Розглянуто основні підходи до організації взаємодії користувача з ігровим ШІ, визначено переваги та обмеження сучасних моделей, а також окреслено ключові виклики щодо адаптивності, персоналізації та пояснюваності поведінки агентів.

2. Проведено аналіз літературних джерел з досліджуваної тематики і огляд сучасних досліджень, що засвідчує важливість розвитку адаптивних, пояснюваних і UX-орієнтованих методів, а також підкреслює актуальність поєднання моделей гравця, гібридних агентів і інтеграції великих мовних моделей для підвищення якості взаємодії.

3. Здійснено постановку задачі дослідження на основі аналізу предметної області. Сформульовано перелік базових вимог до методу взаємодії з ігровим ШІ, серед яких адаптивність, прозорість, баланс ролей, ефективність, розширюваність і орієнтація на користувацький досвід, що визначають напрям подальших розробок у цій сфері.

2 РОЗРОБКА МЕТОДУ ВЗАЄМОДІЇ КОРИСТУВАЧА З ІГРОВИМ ШІ

2.1 Загальна концепція методу

У цьому розділі представлено розроблений метод взаємодії користувача з ігровим штучним інтелектом, який базується на поєднанні трьох ключових компонентів:

- аналізу поведінки гравця в реальному часі,
- контекстної інтерпретації ситуації у грі,
- адаптивної поведінки ігрових агентів (NPC) відповідно до отриманих даних.

Метод орієнтований на використання в ігрових середовищах, де поведінка NPC має бути не лише технічно коректною, але й узгодженою з очікуваннями користувача, його стилем гри, цільовими установками та емоційним станом.

На відміну від класичних моделей (кінцевих автоматів, дерева поведінки (behavior trees), базовані на правилах), запропонований метод включає шар динамічної взаємодії, який дозволяє ігровому агенту реагувати на зміну стилю гри в режимі реального часу.

Загальна структура методу (представлено на рисунку 2.1) складається з таких рівнів:

- рівень збору та структуризації даних про гравця, який включає аналіз мікроповедінкових характеристик (частота дій, реакції, маршрути, патерни ухвалення рішень);
- рівень інтерпретації, який визначає стиль гри, поточний контекст та механіку взаємодії.
- рівень адаптації NPC, де відбувається генерація рішень відповідно до результатів аналізу.
- модуль інтеграції в ігровий рушій, що реалізує взаємодію між системою ШІ та внутрішньою логікою гри.

Рисунок 2.1 – Узагальнена структура методу

Ці рівні формують замкнений цикл адаптивності, в якому модель поступово коригує поведінку NPC відповідно до змін у динаміці гри та індивідуальних особливостей користувача. Концептуальна схема подана на рисунку 2.2 (розроблено автором).

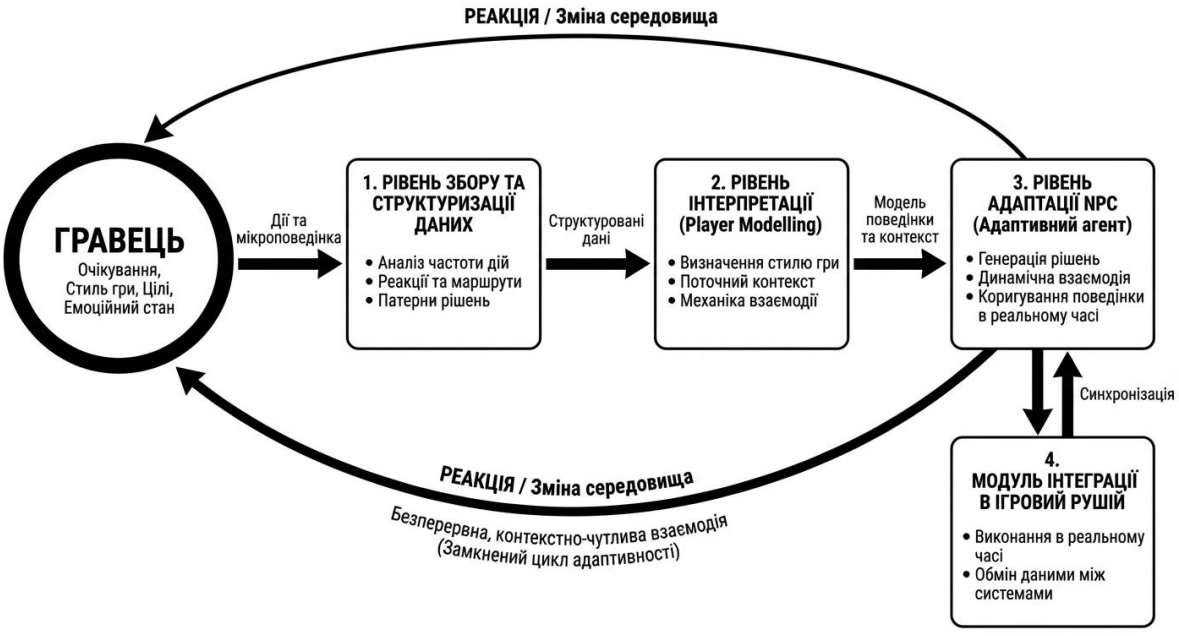


Рисунок 2.2 – Концептуальна схема методу

«Гравець → Дані → Інтерпретація → Адаптивний агент → Реакція → Нові дані → ...»

Таким чином, метод забезпечує безперервну взаємодію між гравцем і системою ШІ на основі принципів контекстної адаптації.

В основі запропонованого методу є створення механізму динамічної взаємодії між користувачем і ігровим агентом, що забезпечує адаптацію поведінки штучного інтелекту (ШІ) відповідно до стилю гри, емоційного стану та рівня майстерності гравця.

На відміну від відомих підходів, які реалізують статичні сценарії або узагальнені моделі гравців, запропонований метод базується на поєднанні трьох компонентів:

- моделі користувача, що описує поведінкові характеристики та мотиваційний тип гравця;
- адаптивного агента, який навчається на основі підкріплення (Reinforcement Learning) з корекцією через поведінкові евристики;
- модуля пояснюваної взаємодії (Explainable Interaction Layer), який формує зворотний зв’язок – пояснює гравцю дії агента та підвищує довіру до системи.

Таким чином, метод спрямований на підвищення персоналізованості, прозорості та ігрової залученості користувача.

Архітектура системи реалізує принципи двосторонньої адаптації – користувач впливає на поведінку агента через свої дії, а агент, у свою чергу, модифікує свою стратегію для підвищення відповідності стилю гри.

Основні компоненти системи:

1) вхідний рівень (захоплення поведінки) – збір даних про користувача: швидкість реакції, частота дій, тривалість ігрових сесій, частота помилок, обрані стратегії;

2) модуль моделювання користувача (User Modeling Module) – обробка зібраних даних, класифікація гравця за типом (наприклад, амбіційний гравець, дослідник, соціально орієнтований гравець, суперник за Бартлом) та визначення рівня навичок;

3) агент навчання з підкріпленням (Reinforcement Learning Agent – RLA) – основний ігровий агент, який виконує дії в середовищі та отримує винагороди залежно від реакції користувача;

4) евристичний контролер (Heuristic Controller) – корекція поведінки агента з урахуванням емоційного стану або контексту (наприклад, уникнення надмірної складності після серії поразок);

5) модуль пояснюваного інтерфейсу (Explainable Interface Module – XIM) – формування інтерпретованого зворотного зв'язку у вигляді повідомлень, підказок або динамічного аналізу дій;

6) цикл зворотного зв'язку – безперервний цикл «спостереження–адаптація–реакція», який дозволяє системі еволюціонувати разом із користувачем.

Схема алгоритму взаємодії користувача з ігровим ШІ представлена у додатку А.

2.2 Моделювання користувача та кластеризація профілів

Для персоналізації взаємодії користувача з ігровим агентом у запропонованому методі використовується модель користувача, що базується на поведінкових характеристиках і типології гравців. Кожному користувачу відповідає вектор ознак (характеристик):

$$U=\{s,r,e,t,p\}, \qquad (2.1)$$

Де

- s – середній час реакції користувача, який відображає швидкість прийняття рішень та моторну активність у процесі гри;
- r – рівень ризиковості дій, що характеризує схильність користувача до агресивних або небезпечних стратегій (наприклад, частота атак, ігнорування захисних механізмів);
- e – показник емоційної залученості, який оцінюється на основі

інтенсивності та частоти ігрових дій, різких змін поведінки, тривалості активних сесій;

- t – тип гравця, визначений відповідно до відомих типологій (зокрема, моделі Бартла: амбіційний гравець, дослідник, соціально орієнтований гравець, суперник), представлений у числовій формі;

- -p – рівень продуктивності користувача, що відображає ефективність ігрових дій (співвідношення успіхів і помилок, досягнення цілей, стабільність результатів).

Вектор ознак профілю користувача сформовано на основі узагальнення підходів, запропонованих у працях з моделювання гравців, поведінкової аналітики та адаптивних ігрових систем [22-27]. Обрані характеристики відображають когнітивні, поведінкові та мотиваційні аспекти взаємодії користувача з ігровим середовищем і широко застосовуються в сучасних дослідженнях у галузі ігрового штучного інтелекту.

Отже, у межах запропонованого методу для моделювання поведінки гравця використовується вектор характеристик, який узагальнює ключові поведінкові та ігрові показники користувача. Такий підхід відповідає сучасним концепціям моделювання гравців, згідно з якими профіль користувача формується на основі поєднання часових, поведінкових, емоційних та мотиваційних ознак.

Параметр s(середній час реакції) використовується для оцінювання швидкості прийняття рішень і моторної активності гравця, що широко застосовується у дослідженнях поведінкової динаміки користувачів у цифрових іграх [22, 25].

Показник r(рівень ризиковості дій) відображає схильність користувача до агресивних або експериментальних стратегій та використовується для ідентифікації стилю гри, що відповідає підходам до аналізу ігрових стратегій, запропонованих у працях з моделювання поведінки гравців [23, 26].

Компонента e(емоційна залученість) базується на концепціях потоку (flow) та імерсії, які розглядають інтенсивність взаємодії користувача з ігровим середовищем як ключовий чинник задоволеності та залученості гравця [24, 27].

Параметр t(тип гравця) визначається відповідно до типологій гравців, зокрема мотиваційних моделей, що застосовуються у сучасних системах персоналізації ігрового процесу [22].

Показник p(рівень продуктивності) характеризує ефективність виконання ігрових завдань та використовується для оцінювання рівня майстерності користувача, що є стандартним підходом у дослідженнях адаптивних ігрових систем [22, 25].

Таким чином, вектор Uпоєднує ключові аспекти поведінки, мотивації та емоційного стану гравця, що відповідає сучасним науковим підходам до моделювання користувачів у ігрових середовищах. Зазначений вектор використовується як вхідний параметр для алгоритму кластеризації k-means з метою формування узагальнених профілів користувачів і подальшої адаптації поведінки ігрового штучного інтелекту.

Сформований вектор U використовується як вхідний параметр для алгоритму кластеризації k-means з метою виявлення типових профілів користувачів та подальшої адаптації поведінки ігрового штучного інтелекту.

На основі цього вектора здійснюється кластеризація користувачів за допомогою алгоритму k-means, який дозволяє виокремити групи з подібними поведінковими характеристиками без попередньої розмітки даних. Під час кластеризації вектори ознак підлягають нормалізації, а категоріальні значення (наприклад, тип гравця) кодуються числовим способом для забезпечення коректного обчислення метрики подібності між користувачами. Результатом роботи алгоритму є віднесення кожного користувача до одного з кластерів, що інтерпретується як його поведінковий профіль.

З огляду на зазначене, для класифікації профілів користувачів доцільно використовувати саме алгоритм k-means, оскільки він забезпечує можливість виявлення типових моделей поведінки без необхідності попереднього маркування даних.

Обгрунтування використання k-means як основного алгоритму

- профіль користувача формується з поведінки;

- типи взаємодії не задані наперед;
- гра та ШІ мають адаптуватися динамічно.

На основі сформованого вектора характеристик (формула 2.1), що відображає індивідуальні особливості поведінки та ігрової активності користувача, будується профіль користувача, який використовується для подальшої адаптації поведінки ігрового штучного інтелекту. З метою автоматизованого виявлення типових моделей поведінки гравців застосовується алгоритм k-means, який належить до методів некерованого навчання.

Використання алгоритму k-means зумовлене відсутністю попередньо розмічених даних щодо типів поведінки користувачів та необхідністю динамічного формування профілів у процесі взаємодії з ігровою системою. Алгоритм виконує кластеризацію множини векторів U, об'єднуючи користувачів у групи на основі метрики подібності (евклідової відстані) між їхніми поведінковими характеристиками. Кожен кластер відповідає певному узагальненому профілю гравця, що відображає характерний стиль гри, рівень залученості та навичок.

На етапі попередньої обробки даних усі числові компоненти вектора Uнормалізуються, що забезпечує коректний вплив кожної ознаки на результат кластеризації. Категоріальна ознака типу гравця tподається у числовому вигляді з використанням відповідного кодування. Параметр k, який визначає кількість кластерів, обирається експериментально з урахуванням кількості типів гравців та показників якості кластеризації.

Результатом роботи алгоритму k-means є віднесення кожного користувача до одного з кластерів, що інтерпретується як поточний профіль користувача. Отриманий профіль використовується як вхідний параметр для модуля адаптивного ігрового агента, зокрема для коригування стратегії навчання з підкріпленням, рівня складності ігрових завдань та способів взаємодії з користувачем. Таким чином, кластеризація профілів дозволяє реалізувати контекстно-залежну адаптацію поведінки ігрового штучного інтелекту в режимі реального часу.

Застосування алгоритму k-means у межах запропонованого методу забезпечує гнучке та масштабоване моделювання користувачів, сприяє підвищенню

персоналізації ігрового процесу та створює основу для подальшого вдосконалення механізмів взаємодії між гравцем і ігровим агентом.

На основі результатів кластеризації профілів користувачів кожному кластеру ставиться у відповідність певний рівень динамічної складності гри (ДСГ), англ. Dynamic Difficulty Adjustment (DDA), який адаптується до індивідуального стилю гри та рівня майстерності користувача.

При цьому результати кластеризації профілю користувача використовуються для визначення початкового та коригувального рівня динамічної складності гри, що оновлюється в процесі взаємодії з ігровим агентом.

Нижче представлена таблиця 2.1 відповідності “кластер користувачів – параметри динамічної складності гри”.

Таблиця 2.1 – Відповідність кластерів профілів користувачів параметрам динамічної складності гри

№ кластера	Узагальнений профіль користувача	Характеристика поведінки	Рівень DDA	Основні параметри адаптації
C_1	Новачок / Casual Player	Повільна реакція, високий рівень помилок, низька ризиковість, середня емоційна залученість	Низький	Зменшення складності завдань; спрощення AI-стратегії; збільшення часу на реакцію; часті підказки
C_2	Обережний / Тактичний гравець	Помірна швидкість дій, низький ризик, стабільна продуктивність	Середньо-низький	Баланс між викликом і підтримкою; обмежена агресивність AI; адаптивні підказки
C_3	Дослідник (Explorer)	Висока активність, різноманітні дії, висока емоційна залученість	Середній	Варіативність ігрових сценаріїв; відкриття альтернативних шляхів; помірна складність
C_4	Змагальний / Aggressive Player	Висока швидкість реакції, ризикові дії, орієнтація на результат	Середньо-високий	Підвищення складності; агресивніша поведінка AI; скорочення допоміжних механізмів
C_5	Експерт / Hardcore Player	Висока продуктивність, низька кількість помилок, стабільна стратегія	Високий	Максимальна складність; складні тактики AI; мінімальна підтримка; підвищені вимоги до навичок

Кожному кластеру користувачів, отриманому в результаті застосування алгоритму k-means, ставиться у відповідність базовий рівень динамічної складності гри (DDA). Зазначені параметри використовуються як початкові умови для адаптації поведінки ігрового агента та можуть коригуватися в процесі навчання з підкріпленням залежно від поточної реакції користувача та контексту гри.

Таким чином, кластеризація профілів забезпечує контекстно-залежну персоналізацію ігрового процесу та формує основу для динамічної адаптації агента.

Алгоритм k-means використовується для формування узагальненого профілю користувача, тоді як RL-агент відповідає за адаптацію поведінки ігрового штучного інтелекту. Таким чином, ці алгоритми виконують різні, але взаємопов’язані функції в межах єдиного методу.

Псевдокод k-means для класифікації профілів користувачів – в додатку Б.

2.3 Адаптивний ігровий агент на основі навчання з підкріпленням

У запропонованому методі адаптивний ігровий агент реалізовано на основі навчання з підкріпленням (Reinforcement Learning, RL). Основна мета RL-агента полягає в тому, щоб динамічно коригувати свою поведінку відповідно до профілю користувача, отриманого через кластеризацію алгоритмом k-means, та рівня динамічної складності гри (DDA). RL-агент забезпечує динамічну адаптацію поведінки NPC відповідно до профілю користувача та параметрів DDA.

У [30] – огляд навчання з підкріпленням. Формальна модель описана нижче.

Адаптивний агент моделюється як марківський процес прийняття рішень (Markov Decision Process, MDP):

$$MDP = \langle S, A, P, R, \gamma \rangle, \qquad (2.2)$$

де:

– $S -$ множина станів що включає поточний профіль користувача $U = \{s, r, e, t, p\}$, параметри ігрового середовища та DDA;

– $A -$ множина можливих дій агента;

– $P(s' \mid s, a) -$ ймовірність переходу зі стану s у стан s'після виконання дії a;

– $R(s, a) -$ функція винагороди агента;

– $\gamma \in [0,1] -$ коефіцієнт дисконтування майбутньої винагороди.

Політика агента $\pi(a \mid s)$визначає ймовірність вибору дії $a \in A$у стані $s \in S$.

Мета навчання з підкріпленням описана нижче.

Метою адаптивного ігрового агента, що використовує навчання з підкріпленням, є максимізація очікуваної сумарної винагороди, яка визначається наступним виразом [28]:

$$G_t = \mathbb{E}[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}]. \qquad (2.3)$$

де:

– G_t – очікувана сумарна (накопичена) винагорода, яку агент прагне максимізувати, починаючи з моменту часу t;

– $\mathbb{E}[\cdot]$ – математичне сподівання, що враховує стохастичний характер середовища та ймовірнісну політику вибору дій агентом;

– k – індекс кроку взаємодії агента з ігровим середовищем;

– R_{t+k+1} – винагорода, отримана агентом на кроці $t + k + 1$після виконання дії;

– $\gamma \in [0,1]$ – коефіцієнт дисконтування, який визначає відносну важливість майбутніх винагород порівняно з поточною.

Інтерпретація коефіцієнта дисконтування: коефіцієнт дисконтування γ відіграє ключову роль у формуванні поведінки агента:

– при $\gamma \approx 0$агент орієнтується переважно на миттєву винагороду, що може призводити до короткострокових стратегій;

– при $\gamma \approx 1$агент враховує довгострокові наслідки своїх дій, формуючи більш стабільну та стратегічну поведінку.

У контексті ігрового штучного інтелекту вибір значення γдозволяє регулювати баланс між швидкою реакцією агента та його довгостроковою адаптацією до стилю гри користувача.

Зв’язок формули з ігровим процесом: у межах запропонованого методу очікувана сумарна винагорода формується з урахуванням таких факторів:

- успішність взаємодії користувача з ігровим середовищем;
- рівень емоційної залученості гравця;
- відповідність складності гри рівню майстерності користувача.

Максимізація значення G_tозначає, що агент навчається обирати такі дії, які забезпечують оптимальний баланс між викликом і комфортом гри, підвищуючи загальну якість взаємодії користувача з ігровим штучним інтелектом.

Вектор стану та інтеграція динамічної складності гри (DDA) представлена нижче.

Стан агента формується як вектор:

$$s_t = \{U, \text{ігрові параметри}, \text{DDA}\}. \quad (2.4)$$

де

- U – профіль користувача, отриманий через k-means;
- ігрові параметри – поточний стан гри, позиція гравця, результати останніх дій;
- DDA – рекомендований рівень складності відповідно до кластеру користувача.

Інтеграція DDA дозволяє модулювати функцію винагороди та коригувати дії агента залежно від профілю гравця.

Функція винагороди:

$$R(s_t, a_t) = \alpha \cdot \Delta\text{прогресу} + \beta \cdot \text{емоційне залучення} - \lambda \cdot \text{рівень помилок}, \quad (2.5)$$

де

- Δпрогресу – зміна досягнень користувача;
- емоційне залучення – показник реакції гравця на дії агента;
- рівень помилок – штраф за невдалу взаємодію;
- α, β, λ – вагові коефіцієнти.

Цикл взаємодії RL-агента описується як “стан – дія – винагорода – оновлення політики”:

1) спостереження стану s_t.

2) вибір дії a_t відповідно до політики $\pi(a_t \mid s_t)$;

3) виконання дії, отримання нового стану s_{t+1};

4) обчислення винагороди $R(s_t, a_t)$;

5) оновлення політики або Q-функції (2.6).

Оновлення Q-функції в алгоритмі Q-learning, іншими словами оновлення оцінки якості дії в поточному стані здійснюється відповідно до правила Q-learning [28]:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta \left[R(s_t, a_t) + \gamma \max_{a'} \quad Q(s_{t+1}, a') - Q(s_t, a_t) \right], \quad (2.6)$$

- $Q(s_t, a_t)$ — оцінка якості (цінності) виконання дії a_t у стані s_t, яка відображає очікувану сумарну винагороду;

- s_t — поточний стан середовища на момент часу t;

- a_t – дія, виконана агентом у стані s_t;

- s_{t+1} – наступний стан середовища, отриманий після виконання дії a_t;

- a' – можлива дія агента у наступному стані s_{t+1};

- $R(s_t, a_t)$ – миттєва винагорода, отримана агентом за виконання дії a_t у стані s_t;

- $\max_{a'} Q(s_{t+1}, a')$ – максимальне значення Q-функції серед усіх можливих дій у наступному стані;

- $\gamma \in [0,1]$ – коефіцієнт дисконтування, який визначає важливість майбутніх винагород;

- $\eta \in (0,1]$ – коефіцієнт навчання, що визначає швидкість оновлення Q-функції.

η – коефіцієнт навчання коефіцієнт навчання, який визначає швидкість адаптації агента до змін у поведінці користувача та ігрового середовища.

Інтерпретація формули в контексті ігрового ШІ: дане правило оновлення дозволяє агенту поступово коригувати оцінку ефективності своїх дій на основі досвіду взаємодії з ігровим середовищем:

- доданок $R(s_t, a_t)$відображає безпосередню реакцію середовища на дію агента;
- вираз $\gamma \max_{a'} Q(s_{t+1}, a')$враховує очікувану майбутню вигоду;
- різниця між новою оцінкою та попереднім значенням Q-функції визначає напрям і величину корекції.

У межах запропонованого методу ця формула використовується для адаптації поведінки NPC відповідно до стилю гри користувача та параметрів динамічної складності, забезпечуючи поступове формування оптимальної політики взаємодії.

RL-агент забезпечує двосторонню адаптацію, що дозволяє NPC підлаштовуватися під індивідуальний стиль гри та підтримувати оптимальний рівень складності завдань.

Псевдокод RL-агента для адаптивного ігрового ШІ представлений в додатку В.

Центроїд кластера – це вектор середніх значень ознак об’єктів, що належать до даного кластера, який використовується як його узагальнена характеристика та орієнтир при повторному розподілі об’єктів у процесі кластеризації.

Висновки до розділу 2

1. Представлено загальну концепцію методу взаємодії між користувачем та ігровим ШІ, який забезпечує контекстно-адаптивну поведінку NPC на основі реального аналізу дій гравця, інтерпретації стилю гри та динамічної адаптації під індивідуальні особливості користувача.

2. Розроблено підхід до автоматизованого моделювання гравця, що передбачає використання алгоритму k-means для кластеризації поведінкових

характеристик, що дозволяє персоналізувати рівень складності та стратегію взаємодії ШІ з користувачем.

3. Запропоновано реалізувати адаптивний ігровий агент на основі навчання з підкріпленням, який коригує свою поведінку згідно профілю користувача та динамічної складності, забезпечує оптимізацію взаємодії NPC та якісний баланс ігрового досвіду.

3 РЕАЛІЗАЦІЯ МЕТОДУ ВЗАЄМОДІЇ КОРИСТУВАЧА З ІГРОВИМ ШТУЧНИМ ІНТЕЛЕКТОМ

3.1 Реалізація програмної системи в Unreal Engine 4

Для демонстрації роботи запропонованого методу у середовищі Unreal Engine 4 [32-34] було створено експериментальний прототип ігрової сцени, що включає:

- ігрового персонажа (Player Character, PC), яким керує користувач;
- базову систему керування рухом (переміщення, повороти, взаємодія);
- механізми просторової навігації та взаємодії з ігровими об’єктами;
- компонент збору поведінкових даних;
- базову структуру класів, що буде надалі розширена модулем адаптивного штучного інтелекту.

У середовищі Unreal Engine 4 реалізовано ігрового персонажа з базовою системою керування, що включає можливість переміщення у просторі та виконання стрибків за допомогою вбудованого Character Movement Component. Додатково розроблено механіку взаємодії з об’єктами оточення: персонаж може підбирати предмети, використовуючи систему компонентів рушія.

Іншими словами прототип ігрової сцени містить:

- ігровий персонаж від третьої особи, що рухається рівнем;
- локальна сцена UE4 з базовою геометрією;
- механіка руху, стрибків та переміщення простором;
- взаємодія з об’єктами / тригерами (судячи з поведінки та руху камери);
- ігровий агент, який наближається до гравця та реагує на його поведінку (частково видно у русі та зміні позицій);
- поведінка переслідування.

Для реалізації прототипу системи взаємодії користувача з ігровим штучним інтелектом було обрано наступний стек технологій:

- ігровий рушій (англ. game engine): Unreal Engine 4, що забезпечує високу продуктивність, розширюваність та широкі можливості для створення

інтерактивного середовища.

– мову програмування: C++, що використовується для реалізації основної логіки гри та алгоритмів взаємодії користувача з ігровим агентом.

– технології Unreal Engine:

Даний стек технологій дозволяє ефективно реалізувати функціонал адаптивної поведінки ігрових агентів та інтегрувати алгоритми машинного навчання в ігровий процес.

Обрана архітектура проекту забезпечує сумісність із пропонованою в розділі 2 підходами й дозволяє інтегрувати модулі аналізу поведінки, класифікації стилю гри та адаптивної поведінки ігрового агента без суттєвої зміни базового коду.

Ігрові сцени представлені на рисунку 3.1.

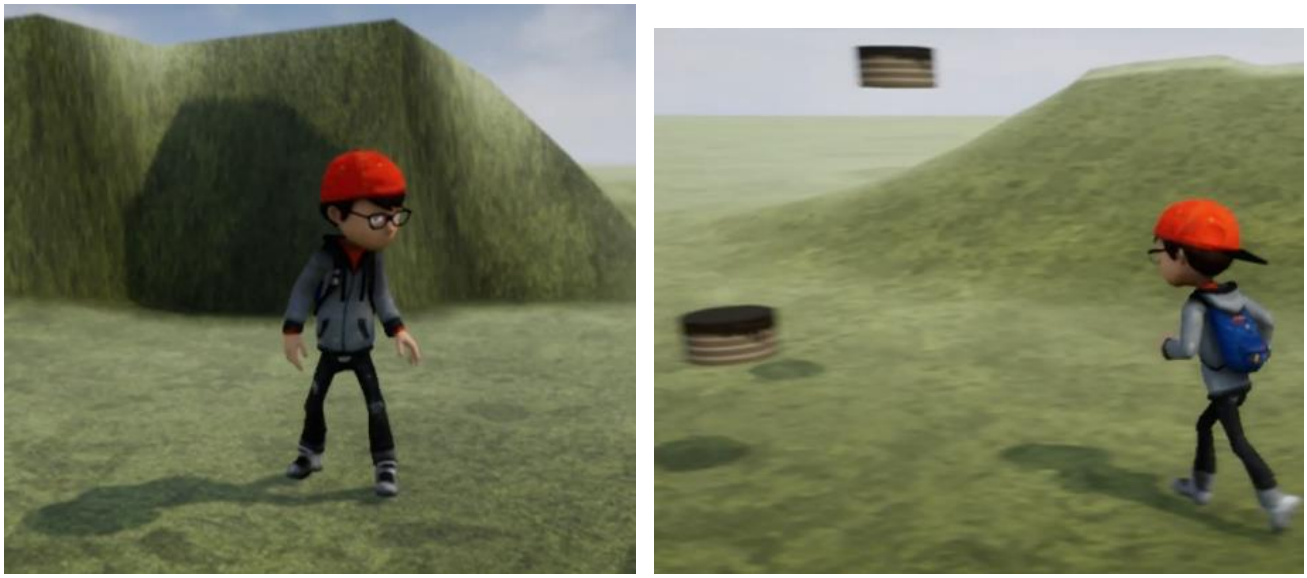


Рисунок 3.1 – Ігрові сцени

У прототипі застосовано такі базові засоби Unreal Engine 4 (UE4):

– Actor System – базова система класів UE4, що дозволяє визначати ігрові об’єкти, їх поведінку та взаємодію;

– Character Movement Component – вбудований компонент, який реалізує:

1) пересування персонажа у просторі;
2) обробку гравітації;

3) стрибки, біг, прискорення;
4) взаємодію з фізичними поверхнями;

- Input System – система прив’язки (binding) для обробки клавіш та осей:
  1) рух уперед/назад;
  2) рух ліворуч/праворуч;
  3) стрибок;
  4) взаємодія;
- Component & Collision System – використані для модульної побудови об’єктів та обробки зіткнень у середовищі гри, а саме:
  1) активації тригерів;
  2) взаємодії з предметами;
  3) визначення зон інтересу.

Ці технології утворюють основу, на якій надалі реалізується алгоритм збору поведінкових даних та контекстного аналізу.

Ігровий персонаж реалізований як клас APlayerCharacter, що успадковує ACharacter. У його структурі визначено:

- CapsuleComponent – фізичний колайдер;
- CharacterMovementComponent – система переміщення;
- CameraBoom та FollowCamera – компоненти камери третьої особи;
- UInteractionComponent – модуль взаємодії;
- UPlayerAnalyticsComponent – модуль збору поведінкових характеристик.

Фрагмент структури класу:

```
UCLASS()
class MYGAME_API APlayerCharacter : public ACharacter
{
   GENERATED_BODY()

public:
   APlayerCharacter();
```

```
protected:
    virtual void SetupPlayerInputComponent(UInputComponent*
PlayerInputComponent) override;

private:
    UPROPERTY()
    class UPlayerAnalyticsComponent* AnalyticsComponent;

    UPROPERTY()
    class UInteractionComponent* InteractionComponent;
};
```

Рух персонажа реалізовано шляхом прив’язки (англ. binding) дій до певних клавіш або елементів керування.

- MoveForward(float Value);
- MoveRight(float Value);
- Jump();
- LookUp та Turn.

На основі компонента руху персонажа (англ. Character Movement Component) було реалізовано такі функціональні можливості:

- швидкість переміщення;
- прискорення;
- обмеження повороту;
- плавну зміну напряму.

Система керування побудована відповідно до типової структури шаблону третьої особи (аннгл. third-person template), що забезпечує зрозумілість і відтворюваність експериментального прототипу.

Реалізація взаємодії з об’єктами, зокрема механіки підбору предметів, здійснюється за допомогою:

– використання колайдерів типу trigger на об’єктах;

– застосування компонента взаємодії UInteractionComponent;

– відправлення подій типу OnOverlapBegin.

Поведінка взаємодії включає:

– визначення відстані до об’єкта;

– активацію взаємодії (наприклад, підбір предмета);

– реєстрацію факту взаємодії у компоненті аналітики гравця (Player Analytics Component).

Модуль збору поведінкових даних (UplayerAnalyticsComponent) відповідає за:

- логування рухів (вектор швидкості, напрям);
- кількість стрибків;
- частоту взаємодій із предметами;
- час перебування у зонах;
- зміни стилю пересування.

Модуль збирає дані у вигляді часових вибірок кожні 0.1 секунди, формуючи початковий поведінковий профіль.

Структура логування:

```
struct FPlayerActionSample
{
   FVector Velocity;
   bool bIsJumping;
   bool bInteracted;
   float Timestamp;
};
```

На основі зазначених механік створено працездатний ігровий рівень, у якому:

- гравець може вільно пересуватися;
- виконувати стрибки;
- взаємодіяти з елементами оточення;
- генерувати повний набір поведінкових даних для подальшого аналізу.

Програмний код руху персонажа представлений у додатку Г.

Цей модуль утворює базу для реалізації методу адаптивної поведінки ігрового агента, описаного у розділі 2.

3.2 Реалізація модулів аналізу поведінки, класифікації стилю гри та адаптивної поведінки ігрового агента

Для реалізації прототипу використано такі технології та програмні засоби:

- ігровий рушій: Unreal Engine 4;
- мова програмування: C++;
- скриптова логіка та прототипування: Blueprints (UE4 Visual Scripting);
- архітектура ШІ: AI Controller, Behavior Components;
- навігація: NavMesh, AI MoveTo
- збір та аналіз даних: власні C++ модулі;
- методи машинного навчання:
    1) k-means (кластеризація профілів користувачів);
    2) навчання з підкріпленням (англ. Reinforcement Learning (Q-learning) для адаптації поведінки NPC.

Вибір Unreal Engine 4 зумовлений його широкими можливостями для реалізації ігрового ШІ, підтримкою C++ для високопродуктивної логіки та наявністю інструментів візуалізації й налагодження. C++ використано для реалізації критичних компонентів логіки, зокрема збору даних та навчання з підкріпленням, з метою підвищення продуктивності та гнучкості системи. Blueprint застосовувався для візуального прототипування та інтеграції з ігровою сценою.

Реалізація збору поведінкових даних користувача

Збір поведінкових даних реалізовано на рівні Player Character та Game Mode. У процесі гри фіксуються такі параметри:

- середній час реакції гравця;
- частота виконання дій;
- кількість помилок;
- тривалість сесії;

– стиль навігації та вибір стратегій.

Приклад фрагмента коду збору часу реакції:

```
float ReactionTime = GetWorld()->GetTimeSeconds() - LastActionTimestamp;
UserProfile.ReactionTime =
   (UserProfile.ReactionTime + ReactionTime) / 2.0f;
```

Зібрані дані формують вектор характеристик користувача $U = \{s, r, e, t, p\}$, який передається до модуля моделювання користувача.

Реалізація кластеризації профілів користувачів описано нижче.

Модуль кластеризації реалізовано у вигляді окремого C++ класу UserClusteringModule. Алгоритм k-means застосовується до накопичених векторів характеристик користувачів для визначення поточного поведінкового профілю.

Основні етапи реалізації:

– нормалізація числових ознак;

– кодування категоріальної змінної типу гравця;

– обчислення відстаней між векторами;

– віднесення користувача до відповідного кластера.

Фрагмент логіки визначення кластера:

```
int32 AssignedCluster = KMeansClassifier.AssignCluster(UserFeatureVector);
CurrentUserCluster = AssignedCluster;
```

Отриманий кластер використовується для ініціалізації параметрів Dynamic Difficulty Adjustment (DDA).

Реалізація механізму Dynamic Difficulty Adjustment (DDA)

Механізм DDA реалізовано як окремий контролер, що змінює параметри гри залежно від кластера користувача та поточного стану гри.

До параметрів DDA належать:

– складність завдань;

– агресивність NPC;

– швидкість реакції ворогів;

– частота підказок.

Приклад адаптації параметрів NPC:

```
void UDDAController::ApplyDifficulty(int32 ClusterID)
{
   if (ClusterID == 0)
      NPC_Aggressiveness = 0.3f;
   else if (ClusterID == 4)
      NPC_Aggressiveness = 0.9f;
}
```

Реалізація RL-агента ігрового ШІ

Адаптивний ігровий агент реалізований у вигляді AI Controller, який використовує принципи навчання з підкріпленням. Агент навчається оптимальної поведінки шляхом взаємодії з користувачем і середовищем.

Стан агента включає:

- поточний кластер користувача;
- рівень DDA;
- параметри ігрової ситуації.

Дії агента:

- вибір тактики пересування;
- зміна рівня агресивності;
- ініціація або уникнення конфлікту.

Фрагмент оновлення Q-функції:

```
Q[state][action] += LearningRate *
   (Reward + Gamma * MaxQ(nextState) - Q[state][action]);
```

Функція винагороди враховує:

- прогрес гравця;
- емоційну залученість;
- кількість помилок.

Таким чином, RL-агент забезпечує динамічну адаптацію поведінки NPC, що відповідає стилю гри користувача.

3.3 Інтеграція компонентів та цикл взаємодії

Усі реалізовані програмні компоненти експериментального прототипу інтегровані в єдиний адаптивний цикл взаємодії, який забезпечує безперервний обмін інформацією між користувачем та ігровим штучним інтелектом. Зазначений цикл відображає логіку функціонування запропонованого методу та реалізує принципи контекстної й персоналізованої адаптації поведінки ігрових агентів.

Загальний цикл взаємодії має такий вигляд (рисунок 3.2) :

Гравець → Збір даних → Кластеризація → DDA → RL-агент → Адаптація NPC → Нові дані

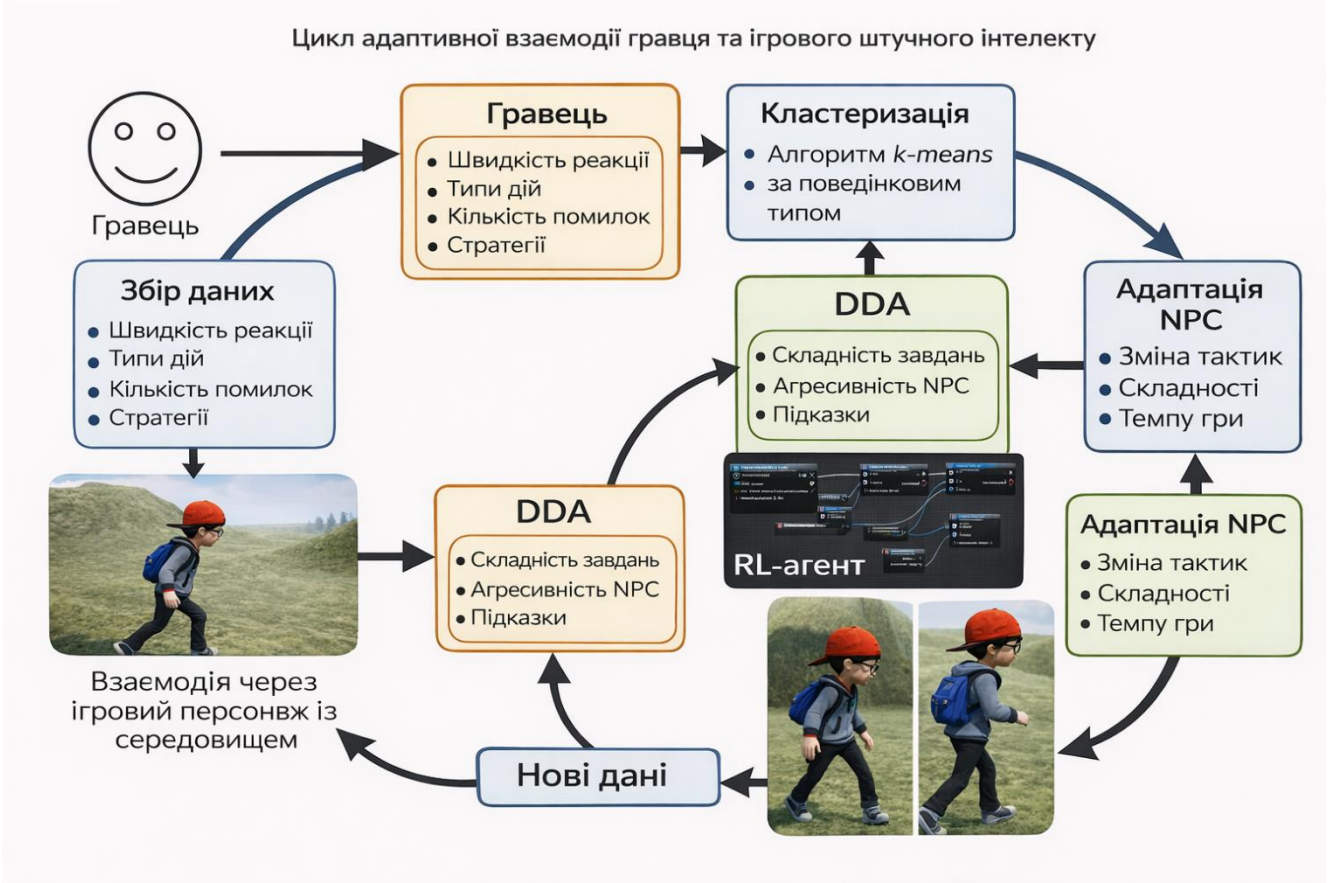


Рисунок 3.2 – Схема процесу адаптивної взаємодії з ігровим штучним інтелектом

На першому етапі користувач взаємодіє з ігровим середовищем через ігрового персонажа, виконуючи дії, що відображають його стиль гри, рівень майстерності та поведінкові особливості. У процесі цієї взаємодії система здійснює безперервний

збір поведінкових даних, зокрема інформації про швидкість реакції, частоту та тип дій, кількість помилок, тривалість ігрових сесій та вибір стратегій.

Отримані дані передаються до модуля моделювання користувача, де формується вектор характеристик користувача. На основі цього вектора здійснюється кластеризація профілю за допомогою алгоритму k-means, що дозволяє віднести користувача до одного з узагальнених поведінкових типів. Визначений кластер інтерпретується як поточний профіль користувача та використовується як контекстна інформація для подальшої адаптації ігрового процесу.

На наступному етапі результати кластеризації застосовуються в модулі динамічного коригування складності гри (Dynamic Difficulty Adjustment, DDA). Значення DDA визначає базові параметри складності ігрових завдань, рівень агресивності NPC, частоту підказок та інші характеристики ігрового середовища. При цьому параметри DDA можуть змінюватися як на початку ігрової сесії, так і в процесі гри залежно від поточних результатів взаємодії користувача з системою.

Ключову роль у циклі взаємодії відіграє адаптивний ігровий агент на основі навчання з підкріпленням, який отримує інформацію про стан гри, профіль користувача та рівень DDA. RL-агент обирає дії відповідно до поточної політики та коригує свою поведінку на основі отриманої винагороди, що відображає ефективність взаємодії з користувачем. У такий спосіб агент поступово навчається оптимальним стратегіям поведінки для різних типів гравців.

Результатом роботи RL-агента є адаптація поведінки NPC, яка проявляється у зміні тактик, складності, темпу гри та характеру взаємодії з користувачем. Змінена поведінка NPC, у свою чергу, впливає на подальші дії гравця, формуючи нові поведінкові дані, що знову надходять до системи аналізу.

Таким чином, запропонований цикл взаємодії реалізує принцип безперервної двосторонньої адаптації, відповідно до якого користувач впливає на поведінку ігрового штучного інтелекту через свої дії, а ігровий агент, у відповідь, динамічно коригує свою стратегію з метою підвищення відповідності стилю гри, підтримання оптимального рівня складності та зростання ігрової залученості користувача.

Реалізація такого інтегрованого циклу в середовищі Unreal Engine 4 підтверджує практичну придатність запропонованого методу та створює основу для подальшого розвитку систем адаптивного ігрового штучного інтелекту.

Висновки до розділу 3

1. Розроблено експериментальний прототип ігрової сцени в Unreal Engine 4, у якому реалізовано базову систему керування персонажем, механізми взаємодії з ігровими об’єктами та збір поведінкових даних, що закладає основу для подальшої адаптації ігрового середовища.

2. Розроблено програмну архітектуру, в якій реалізовано модулі аналізу користувацьких даних, кластеризації профілів гравців (алгоритм k-means), динамічної адаптації складності (DDA) та навчання адаптивного агенту на основі RL, що забезпечує персоналізацію ігрового процесу.

3. Інтегровано всі компоненти у єдиний адаптивний цикл взаємодії, що забезпечує безперервну двосторонню адаптацію між користувачем і ігровим ШІ, підтверджуючи ефективність запропонованого методу в умовах реального ігрового середовища. Прототип підтверджує практичну реалізовність запропонованого методу взаємодії користувача з ігровим штучним інтелектом.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи було:

1. Проведено аналіз предметної області взаємодії людини і штучного інтелекту у контексті комп'ютерних ігор. Проаналізовано підходи до організації взаємодії гравця з ігровим ШІ, визначено сильні та слабкі їх сторони, окреслено головні проблеми.

2. Здійснено огляд літературних джерел з досліджуваної тематики.

3. Сформульовано задачу дослідження й визначено основні вимоги до методу взаємодії з ігровим ШІ, включаючи адаптивність, прозорість, баланс ролей, ефективність, розширюваність і фокус на користувацькому досвіді.

4. Описано загальну концепцію методу взаємодії гравця з ігровим ШІ, що забезпечує контекстну адаптацію NPC на основі аналізу дій користувача, стилю гри та індивідуальних особливостей.

5. Запропоновано підхід до автоматизованого моделювання гравців з використанням k-means для кластеризації поведінкових даних, що дає змогу персоналізувати складність та стратегію взаємодії з ШІ.

6. Обґрунтовано використання адаптивного ігрового агента на основі навчання з підкріпленням, який змінює поведінку відповідно до профілю гравця та рівня складності для оптимізації досвіду користувача.

7. Розроблено прототип ігрової сцени у Unreal Engine 4 з базовою системою керування, механіками взаємодії та збором поведінкових даних для подальшої адаптації середовища.

8. Створено програмну архітектуру, яка включає модулі аналізу даних, кластеризації профілів, динамічного регулювання складності й RL-агентів, що забезпечує персоналізацію ігрового процесу.

9. Всі компоненти інтегровано в єдиний адаптивний цикл, який реалізує безперервну двосторонню адаптацію між користувачем і ігровим ШІ, підтверджуючи ефективність розробленого методу в реальній ігровій системі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Yannakakis G. N., Togelius J. Artificial Intelligence and Games. Springer Nature, 2018. 330 p. URL: https://gameaibook.org/book.pdf

2. Lankoski P., Björk S. Game Research Methods: An Overview. – Pittsburgh: ETC Press, 2015. 230 p.

3. Hu S., Huang T., Ilhan F., Tekin S.F., Kompella G. Liu, R., Liu L. A Survey on Large Language Model-Based Game Agents. arXiv preprint, arXiv:2404.02039, Apr. 2024. URL: https://arxiv.org/abs/2407.07663

4. Yannakakis G.N., Spronck P., Loiacono D., André E. Player Modeling. In: Artificial and Computational Intelligence in Games. Dagstuhl Follow-Ups, Vol. 6. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2013, pp. 45-59. DOI: 10.4230/DFU.Vol6.12191.45.

5. Fernandes H. M. de C. L. Player Modeling for Role-Playing Games – Improving Bethesda's Radiant AI. MSc Thesis. Instituto Superior Técnico, Universidade de Lisboa, 2018 URL: https://fenix.tecnico.ulisboa.pt/downloadFile/1126518582689776/Thesis.pdf

6. Data-Driven Approaches to Game Player Modeling: A Systematic Literature Review, ACM Computing Surveys, vol. 50, no. 6, Jan. 2018. doi:10.1145/3145814. Korea University Pure

7. Zhu J., Villareale J., Javvaji N., Risi S., Löwe M., Weigelt R., Harteveld C. Player-AI Interaction: What Neural Network Games Reveal About AI as Play. Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21), May 2021, Article No. 77, doi: 10.1145/3411764.3445307

8. Божигора Ю., Турченко І. Методи та моделі взаємодії користувача з ігровими агентами штучного інтелекту. Наукові інновації: теоретичні підходи та практич-ний вплив: матеріали наук.-практ. конф. (8-10.12.2025 р.. Неаполь. Італія). Неаполь. 2025. С. 160-163 URL: https://www.eoss-conf.com/wp-content/uploads/2025/12/Naples_Italy_08.12.25.pdf

9. Lundström J. Cooperation Between Player and AI: A study on player experience with AI companions. Master's Thesis. Umeå University, 2024. URL: https://www.diva-portal.org/smash/get/diva2:XXXXXXX/FULLTEXT01.pdf.

10. Yang D. Designing Mixed-Initiative Video Games. – Master's Thesis. – Northeastern University, 2023. – Available at: https://repository.library.northeastern.edu/files/neu:XXXXXXX.

11. Togelius, J. Playing Smart: On Games, Intelligence, and Artificial Intelligence. Cambridge, Massachusetts: The MIT Press, 2019. 192 p.

12. Сирота В. В. Інтелектуальні агенти в мобільних іграх : магістерська кваліфікаційна робота. Чернівці : ЧНУ ім. Ю. Федьковича, 2021. 85 с. URL: https://eprints.chnu.edu.ua/id/eprint/XXXX.

13. Shao, K., Tang, Z., Zhu, Y., Li, N., Zhao, D. A Survey of Deep Reinforcement Learning in Video Games. arXiv preprint arXiv:1912.10944, Dec 2019.

14. Hartmann, R., et al. Adaptive user modelling in car racing games using behavioural and physiological data. User Modeling and User-Adapted Interaction, 2017. Vol. 27. P. 267–311. DOI: 10.1007/s11257-017-9192-3.

15. Zhang R., Xu Z., Ma C., Yu C., Tu W.-W., Huang S. et al. A Survey on Self-play Methods in Reinforcement Learning, 2024. URL: https://arxiv.org/abs/2408.01072

16. Souchleris, K., Sidiropoulos, G. K., Papakostas, G. A. Reinforcement Learning in Game Industry – Review, Prospects and Challenges. Applied Sciences, 2023, 13(4), 2443. DOI: 10.3390/app13042443.

17. Lu, Y., Yan, K. Algorithms in Multi-Agent Systems: A Holistic Perspective from Reinforcement Learning and Game Theory. arXiv preprint arXiv:2001.06487, Jan 2020.

18. Zhu, J., Villareale, J. Understanding Mental Models of AI through Player-AI Interaction. Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems (CHI EA '21). 2021. DOI: 10.48550/arXiv.2103.16168.

19. Rashmi, C.P., Radhika H. Future of Play: AI Revolutionizing Player Interaction and Character Connection. Journal of Communication and Management, Vol.3, No.04 (2024): 276-286. DOI: 10.58966/JCM2024341.

20. Алєксєєв Д. Д. Дослідження впливу патернів штучного інтелекту на ігровий процес. Радіоелектроніка та молодь у XXI столітті : матеріали 28-го Міжнародного молодіжного форуму. Харків : ХНУРЕ, 2024. Т. 6. С. 466-467. URL: https://openarchive.nure.ua/handle/document/28258

21. Божигора Ю., Турченко І. Взаємодія користувача з ігровим штучним інтелектом: порівняння сучасних підходів та перспективи розвитку. Дослідження у сфері на-уки, технологій та економіки: матеріали наук.-практ. конф.(10-12.12.2025 р. Люксембург). Люксембург, 2025. С. 330-332. URL: https://isu-conference.com/wp-content/uploads/2025/12/Luxembourg_Luxembourg_10.12.25.pdf

22. Yannakakis G. N., Togelius J. Player Modeling. Cham : Springer International Publishing, 2018. 340 p.

23. Drachen A., Canossa A., Yannakakis G. N. Player modeling using self-organization in Tomb Raider: Underworld. Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG'2009). Milano, Italy, 2009.

24. Cowley B., Charles D., Black M., Hickey R. Toward an understanding of flow in video games. Computers in Entertainment, 2008. Vol. 6, No. 2. Article 20.

25. Mahlmann T., Togelius J., Yannakakis G. N. Modelling player behavior in racing games. Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG'2010). Copenhagen, Denmark, 2010. P. 1–8.

26. Yannakakis G. N., Hallam J. Towards optimizing entertainment preference. Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG'2007). Honolulu, USA, 2007. P. 321–328.

27. Nacke L. E., Lindley C. A. Flow and immersion in first-person shooters // Proceedings of the ACM Conference on Future Play (Future Play'2008). Toronto, Canada, 2008. P. 81–88. URL: https://www.researchgate.net/publication/221643959_Flow_and_immersion_in_first-person_shooters_measuring_the_player's_gameplay_experience

28. Sutton R. S., Barto A. G. Reinforcement Learning: An Introduction, 2nd ed., Cambridge, MA, USA: MIT Press, 2018.

29. Watkins C. J. C. H., Dayan P. Q-learning, Machine Learning, vol. 8, no. 3–4, pp. 279–292, 1992.

30. Kaelbling L. P., Littman M. L., Moore A. W. Reinforcement learning: A survey, Journal of Artificial Intelligence Research, vol. 4, pp. 237–285, 1996.

31. Risi S., Togelius J. Neuroevolution in Games: State of the Art and Open Challenges. IEEE Transactions on Computational Intelligence and AI in Games. Vol. 9. No. 1, 2015. pp. 25–41.

32. Epic Games. Unreal Engine. Unreal Engine official site. URL: https://www.unrealengine.com/en-US

33. Epic Games. Learning. Unreal Engine - Epic Developer Community. Unreal Engine community learning. URL: https://dev.epicgames.com/community/unreal-engine/learning

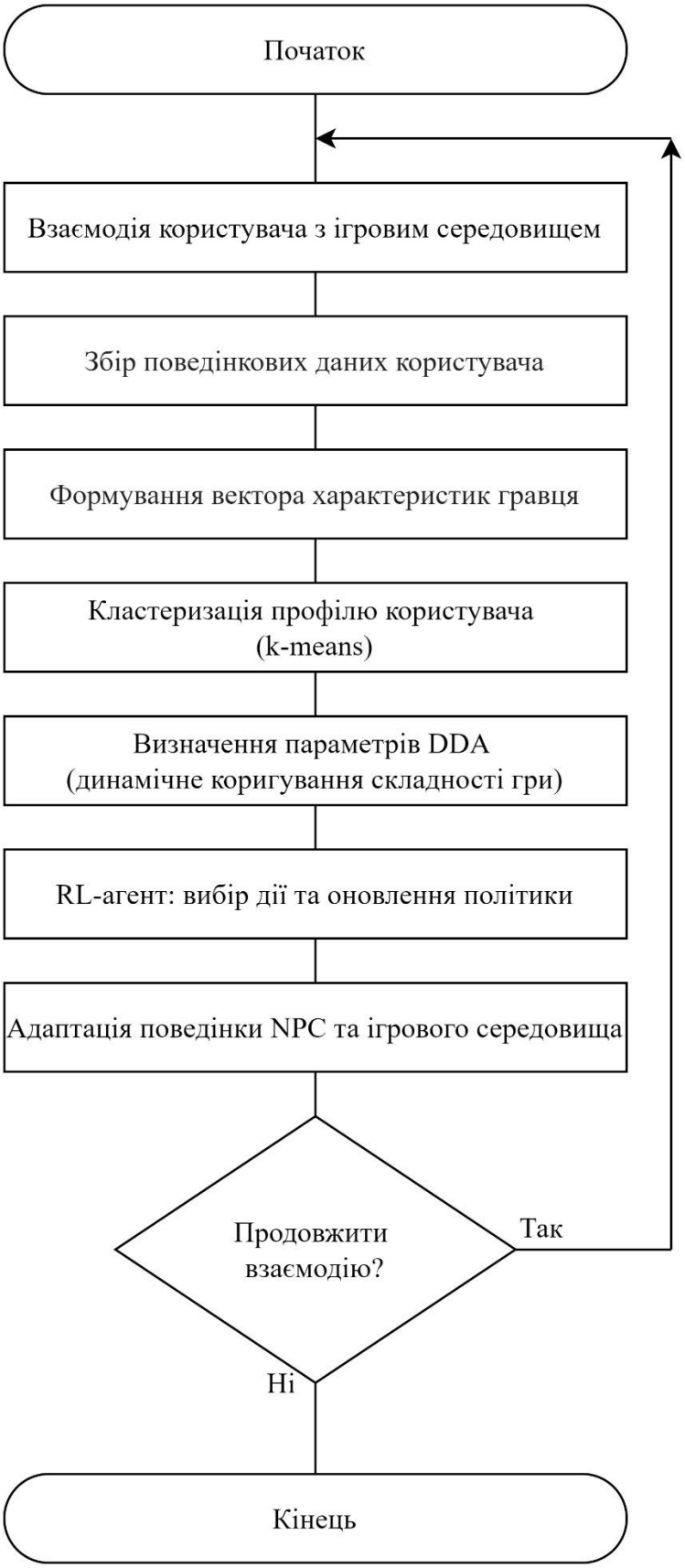
34. Epic Games. Unreal Engine Guides and White Papers. Unreal Engine official guides. URL:https://www.unrealengine.com/en-US/guides-and-white-papers

35. Millington I. AI for Games. Taylor & Francis, 2021. 84 p.

36. Комар М.П., Саченко А.О., Васильків Н.М., Гладій Г.М., Коваль В.С., Ліп'яніна-Гончаренко Х.В. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки» за першим (бакалаврським) рівнем вищої освіти. Тернопіль: ЗУНУ, 2024. 52 с.

Додаток А

Схема алгоритму взаємодії користувача з ігровим ШІ



Додаток Б

Псевдокод k-means для класифікації профілів користувачів

Алгоритм KMeans_UserProfiles

Вхід:

- D = {u1, u2, ..., un} // набір профілів користувачів

Кожен u_i містить характеристики/ознаки (features), напр. вік, активність, інтереси

- K // кількість кластерів
- max_iterations // максимальна кількість ітерацій
- tolerance // поріг для зупинки (опціонально)

Вихід:

- C = {C1, C2, ..., CK} // кластери користувачів
- centroids = {m1, m2, ..., mK} // центри кластерів

1. Ініціалізація:

- Випадково обрати K профілів з D як початкові центроїди m1, m2, ..., mK

2. Повторювати до досягнення max_iterations або поки центроїди не змінюються більше ніж tolerance:

a) Присвоєння кластерів:

Для кожного профілю u_i з D:

- Обчислити відстань(u_i, m_j) до кожного центроїда m_j
- Присвоїти u_i кластеру C_j з найменшою відстанню

b) Оновлення центроїдів:

Для кожного кластера C_j:

- Обчислити новий центроїд m_j як середнє значення всіх профілів у C_j

c) Перевірка на збіжність (опціонально):

- Якщо зміни центроїдів менші за tolerance → зупинити

3. Повернути кластери C та центроїди m

Додаток В

Псевдокод RL-агента для адаптивного ігрового ШІ

1. Ініціалізація:

- Встановити множину станів S (включно з профілем користувача, поточним DDA та параметрами гри)

- Встановити множину дій A агента

- Ініціалізувати Q-таблицю Q(s, a) або політику π(a|s)

- Встановити коефіцієнт навчання α та дисконтування γ

- Встановити ε для ε-greedy стратегії вибору дії

2. Цикл гри (для кожної ігрової сесії або кроку t):

a) Спостереження поточного стану s_t

b) Вибір дії a_t:

- З ймовірністю ε обрати випадкову дію (exploration)

- З ймовірністю 1-ε обрати дію з максимальним Q(s_t, a) (exploitation)

c) Виконати дію a_t у ігровому середовищі

d) Отримати:

- новий стан s_{t+1}

- винагороду R(s_t, a_t) на основі прогресу гравця, емоційного стану та DDA

e) Оновити Q-таблицю:

Q(s_t, a_t) ← Q(s_t, a_t) + α * [R(s_t, a_t) + γ * max_a' Q(s_{t+1}, a') - Q(s_t, a_t)]

f) s_t ← s_{t+1} (переходи до нового стану)

g) Оновити політику π(a|s) (якщо використовується метод, що не базується на Q-таблиці)

3. Повторювати кроки 2a–2g до завершення сесії

4. Після кожної сесії:

- Оновити профіль користувача U

- Кластеризувати новий профіль через k-means (за потреби)

- Скоригувати DDA для наступної сесії

Додаток Г

Програмний код руху персонажа

MyCharacter.h:

```
#pragma once

#include "CoreMinimal.h"
#include "GameFramework/Character.h"
#include "MyCharacter.generated.h"

UCLASS()
class YOURPROJECTNAME_API AMyCharacter : public ACharacter
{
   GENERATED_BODY()

public:

   AMyCharacter();

   UPROPERTY(VisibleAnywhere, BlueprintReadOnly, Category = Camera)
   class USpringArmComponent* CameraBoom;


   UPROPERTY(VisibleAnywhere, BlueprintReadOnly, Category = Camera)
   class UCameraComponent* FollowCamera;

protected:

   virtual void BeginPlay() override;


   void MoveForward(float Value);
   void MoveRight(float Value);

public:

   virtual void SetupPlayerInputComponent(class UInputComponent* PlayerInputComponent)
override;
};
```

MyCharacter.cpp:

```
#include "MyCharacter.h"
#include "GameFramework/SpringArmComponent.h"
#include "Camera/CameraComponent.h"
#include "GameFramework/Controller.h"
```

```
#include "Components/InputComponent.h"
#include "GameFramework/CharacterMovementComponent.h"


AMyCharacter::AMyCharacter()
{
	bUseControllerRotationPitch = false;
	bUseControllerRotationYaw = false;
	bUseControllerRotationRoll = false;


	GetCharacterMovement()->bOrientRotationToMovement = true;
	GetCharacterMovement()->RotationRate = FRotator(0.0f, 540.0f, 0.0f);
	GetCharacterMovement()->JumpZVelocity = 600.f;
	GetCharacterMovement()->AirControl = 0.2f;


	CameraBoom = CreateDefaultSubobject<USpringArmComponent>(TEXT("CameraBoom"));
	CameraBoom->SetupAttachment(RootComponent);
	CameraBoom->TargetArmLength = 300.0f;
	CameraBoom->bUsePawnControlRotation = true;


	FollowCamera = CreateDefaultSubobject<UCameraComponent>(TEXT("FollowCamera"));
	FollowCamera->SetupAttachment(CameraBoom, USpringArmComponent::SocketName);
	FollowCamera->bUsePawnControlRotation = false;
}

void AMyCharacter::BeginPlay()
{
	Super::BeginPlay();
}

void AMyCharacter::SetupPlayerInputComponent(UInputComponent* PlayerInputComponent)
{
	Super::SetupPlayerInputComponent(PlayerInputComponent);

	PlayerInputComponent->BindAxis("MoveForward", this, &AMyCharacter::MoveForward);
	PlayerInputComponent->BindAxis("MoveRight", this, &AMyCharacter::MoveRight);

	PlayerInputComponent->BindAxis("Turn", this, &APawn::AddControllerYawInput);
	PlayerInputComponent->BindAxis("LookUp", this, &APawn::AddControllerPitchInput);

	PlayerInputComponent->BindAction("Jump", IE_Pressed, this, &ACharacter::Jump);
	PlayerInputComponent->BindAction("Jump", IE_Released, this, &ACharacter::StopJumping);
}

void AMyCharacter::MoveForward(float Value)
{
```

```
	if ((Controller != nullptr) && (Value != 0.0f))
	{
		const FRotator Rotation = Controller->GetControlRotation();
		const FRotator YawRotation(0, Rotation.Yaw, 0);

		const FVector Direction = FRotationMatrix(YawRotation).GetUnitAxis(EAxis::X);

		AddMovementInput(Direction, Value);
	}
}

void AMyCharacter::MoveRight(float Value)
{
	if ((Controller != nullptr) && (Value != 0.0f))
	{
		const FRotator Rotation = Controller->GetControlRotation();
		const FRotator YawRotation(0, Rotation.Yaw, 0);

		const FVector Direction = FRotationMatrix(YawRotation).GetUnitAxis(EAxis::Y);

		AddMovementInput(Direction, Value);
	}
}
```

Додаток Д

Копії опублікованих результатів