ХЕРСОНСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
(повне найменування вищого навчального закладу)
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ
(повне найменування інституту, назва факультету (відділення))
КАФЕДРА ПРОГРАМНИХ ЗАСОБІВ І ТЕХНОЛОГІЙ
(повна назва кафедри (предметної, циклової комісії))

Пояснювальна записка

до кваліфікаційної роботи бакалавра
(освітній рівень)

на тему: «Використання методів машинного навчання в розробці відеогри на мові програмування Python»

Виконала: студентка групи 4ПР1
спеціальності
121 - «Інженерія програмного забезпечення»
(шифр і назва спеціальності)

Вдовиченко В.С.
(прізвище та ініціали)

Керівник доц., к.т.н. Огнєва О.Є
(прізвище та ініціали)

Рецензент Корніловська Н.В.
(прізвище та ініціали)

Хмельницький – 2024

Херсонський національний технічний університет
(повне найменування закладу вищої освіти)

Факультет, відділення Інформаційних технологій та дизайну
Кафедра Програмних засобів і технологій
Освітній рівень бакалавр
Спеціальність 121 – Інженерія програмного забезпечення
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри
Програмних засобів і технологій
к.т.н. доц. Огнєва О. Є.
“___” _____ 2024 р.

З А В Д А Н Н Я
НА ВИПУСКНУ РОБОТУ СТУДЕНТУ

Вдовиченко Валерія Сергіївна
(прізвище, ім’я, по батькові)

Тема роботи «Використання методів машинного навчання в розробці відеогри на мові програмування Python»

керівник роботи доц., к.т.н. Огнєва О.Є,
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від .. . .. . 2024 р. № -

1. Строк подання студентом роботи _____
2. Вихідні дані до роботи літературні та періодичні джерела, матеріали преддипломної практики
3. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):
   a Дослідження та аналіз предметної області
   b Методи та засоби розробки для розробки гри
   c Дизайн та архітектура гри
   d Розробка та тестування гри
5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та	Підпис, дата

	посада консультанта	завдання видав	завдання прийняв

7. Дата видачі завдання _____18.02.2024_ _____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання роботи	Термін виконання етапів роботи	Примітки
1.	Отримання завдання	18.02.2024	Виконано
2.	Підбір літератури	28.02.2024	Виконано
3.	Аналіз предметної області	03.03.2024	Виконано
4.	Розробка та обґрунтування завдання	10.03.2024	Виконано
5.	Розробка концептуальної моделі	21.03.2024	Виконано
6.	Розробка алгоритму	25.03.2024	Виконано
7.	Проектування програми	01.04.2024	Виконано
8.	Розробка інтерфейсу програми	26.04.2024	Виконано
9.	Тестування програми	03.05.2024	Виконано
10.	Оформлення пояснювальної записки	30.05.2024	Виконано
11.	Захист кваліфікаційної роботи	24.06.2024	Виконано

Студент _____ Вдовиченко В.С.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Огнєва О.Є
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Кваліфікаційна робота бакалавра: 87 сторінок, 6 рисунків, 1 таблиця, 1 додаток та 18 джерел.

Мета роботи – розробка 2D відеогри у жанрі «платформер», який використовує методи процедурної генерації та машинного навчання для створення динамічного, захопливого та адаптивного ігрового процесу.

Об'єкт дослідження – 2D відеогра у жанрі «платформер», яка слугує експериментальною платформою для застосування та тестування інтеграції процедурної генерації та машинного навчання.

Предмет дослідження – застосування процедурної генерації та алгоритмів машинного навчання, зокрема багатошарового персептрона та генетичних алгоритмів, у створенні та управлінні механізмами 2D відеогри у жанрі «платформер».

Методи дослідження – аналіз, реалізація та тестування алгоритмів процедурної генерації для створення світу та використання методів машинного навчання для розробки та оптимізації поведінки персонажів, керованих штучним інтелектом.

Результат роботи:

- реалізовано алгоритми процедурної генерації для створення різноманітного та унікального ігрового середовища для кожного запуску гри;

- розроблено інтелектуальну та адаптивну поведінку головного персонажа завдяки використанню багатошарового персептрона та генетичних алгоритмів;

- автоматизована поведінка головного персонажа дозволить покращити процедури керування неігровими персонажами завдяки автоматизації їх тестування;

- покращено ігровий досвід завдяки поєднанню процедурної генерації світу та керування персонажем на основі машинного навчання, зробивши його більш захопливим та цікавим;

- продемонстровано практичну користь і масштабованість інтеграції методів машинного навчання в розробку ігор, створивши основу для майбутніх проектів.

Новизна роботи:

- Дипломна робота представляє новий підхід до розробки розробка 2D відеогри у жанрі «платформер» шляхом інтеграції процедурної генерації з методами машинного навчання, зокрема багатошаровим персептроном та генетичними алгоритмами. На відміну від традиційних методів розробки ігор, які покладаються на створені вручну рівні та скриптову поведінку персонажів, новий підхід використовує процедурні алгоритми для створення різноманітних і динамічних ігрових світів, гарантуючи, що кожне проходження гри пропонує унікальний досвід. Процедурна генерація не лише покращує досвід, але й значно скорочує час і ресурси, необхідні для розробки рівнів, що є більш ефективним способом створення захопливого контенту.

- Реалізація персонажів керованих багатошаровим персептроном та генетичними алгоритмами, може стати перспективним та цікавим підходом до ігрового ШІ. Така комбінація більше пасує до створення різноманітних супротивників, бо дозволяє розробляти інтелектуальну та адаптивну поведінку ШІ, яка може розвиватися і вдосконалюватися з часом. Тоді як багатошаровий персептрон здатен реагувати на оточення та дії гравця, генетичні алгоритми спрощують створення нових копій супротивників, які зберігають найкращі підходи вивчені з попереднього доступу та комбінують їх з новими спостереженнями. Інтеграція цих методів машинного навчання в ігровий ШІ все ще залишається відносно недослідженою сферою, що робить проект важливою спробою продемонструвати практичне застосування та переваги машинного навчання для покращення ігрової механіки та досвіду гравців.

Ключові слова: *розробка ігор, Python, PyGame, машинне навчання, TensorFlow, 2D платформер, процедурна генерація, генетичні алгоритми*

АНОТАЦІЯ

У дипломній роботі представлено розробку та реалізацію 2D відеогри у жанрі «платформер» за допомогою мови програмування Python та з використанням методів машинного навчання. Вона має наступні структурні частини: вступ, чотири розділи, висновки, список використаних джерел та додаток.

У грі реалізовано унікальне поєднання процедурної генерації для створення ігрового світу та двох режимів керування персонажем: керування гравцем та керування ШІ. Управління ШІ реалізовано за допомогою багатошарового персептрона, навченого з використанням генетичного алгоритму. Запропонований метод має на меті посилити інтеграцію алгоритмів машинного навчання в розробку відеоігор.

Практична реалізація розпочалася з налаштування середовища розробки та проектування архітектури і дизайну гри. Головним викликом було поєднання процедурної генерації світу та керування персонажами за допомогою ШІ. Процедурна генерація забезпечує динамічний і різноманітний ігровий світ, тоді як управління ШІ відповідає за адаптивний і складний ігровий процес. Для забезпечення надійності та коректності цих компонентів проводилося ручне та автоматизоване тестування з акцентом на продуктивність (враховуючи вади оптимізації мови програмування Python, через які вона відносно рідко застосовується для розробки відеоігор) та користувацький досвід.

Практичне значення цієї роботи полягає в її потенціалі для покращення взаємодії між процедурною генерацією та методами машинного навчання в розробці ігор. Вона пропонує масштабований та інноваційний підхід до створення цікавого та непередбачуваного ігрового середовища, покращуючи як процес розробки, так і досвід гравців. Успішна реалізація демонструє доцільність та потенціал для фінансової привабливості інтеграції передових методів машинного навчання в ігрову індустрію.

ANNOTATION

This thesis presents the development and implementation of a 2D platformer video game using the Python programming language and machine learning methods. It has the following structural parts: introduction, four chapters, conclusions, bibliography and appendix.

The game implements a unique combination of procedural generation to create the game world and two character control modes: player control and AI control. AI control is implemented using a multilayer perceptron trained using a genetic algorithm. The proposed method aims to enhance the integration of machine learning algorithms into video game development.

Practical implementation began with setting up the development environment and designing the game architecture and design. The main challenge was to combine procedural world generation and AI character control. Procedural generation provides a dynamic and diverse game world, while AI control is responsible for adaptive and complex gameplay. To ensure the reliability and correctness of these components, manual and automated testing was conducted with a focus on performance (given the optimisation flaws of the Python programming language, which is why it is relatively rarely used for video game development) and user experience.

The practical significance of this work lies in its potential to improve the interaction between procedural generation and machine learning techniques in game development. It offers a scalable and innovative approach to creating interesting and unpredictable game environments, improving both the development process and the player experience. Successful implementation demonstrates the feasibility and potential for financial profitability of integrating advanced machine learning techniques into the gaming industry.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

Генетичний алгоритм (ГА) — це еволюційний алгоритм пошуку, що використовується для вирішення задач оптимізації і моделювання шляхом послідовного підбору, комбінування і варіації шуканих параметрів з використанням механізмів, що нагадують біологічну еволюцію.

Багатошаровий персептрон (MLP) — це тип штучної нейронної мережі, що складається з декількох шарів нейронів, включаючи вхідний шар, один або кілька прихованих шарів і вихідний шар, який використовується для розпізнавання образів і навчання складних функцій.

Штучний інтелект (ШІ) — це система, здатна виконувати завдання, які потребують застосування людського інтелекту.

Платформер — жанр відеоігор, ігровий процес в якому складається зі стрибків персонажа по різноманітним платформам (звідси й назва) та через перешкоди, збирання предметів, зазвичай необхідних для проходження рівня.

2D (двомірний) — комп'ютерне покоління цифрових зображень — головним чином складається з двовимірних моделей (таких як геометричні 2D-моделі, текст і цифрові зображення) і методів, визначених для них.

Машинне навчання (ML) — це галузь досліджень штучного інтелекту, зосереджена на розробці та вивченні статистичних алгоритмів, здатних навчатися з даних і узагальнюватися на небачені дані, й відтак виконувати завдання без явних інструкцій.

Процедурна генерація — це метод алгоритмічного створення даних за допомогою комбінацій алгоритмів, поєднаних із випадковістю.

Ігровий світ — це віртуальне середовище у відеогрі, де відбувається ігровий процес, включаючи всі рівні, персонажів, об'єкти та правила, які визначають ігровий всесвіт.

Гравець — активний та безпосередній учасник гри.

Неігровий персонаж (NPC) — персонаж в комп'ютерних іграх, керований програмою.

Спрайт — окреме двовимірне зображення, що застосовується в комп'ютерній графіці як частина більшого зображення. Найчастіше — растрове зображення, що переміщується по екрану, незалежно від фону. Спрайт може бути статичним зображенням або анімованим. У останньому разі послідовність спрайтів змінюється, як кадри у відео.

Тайл — метод створення великих зображень, що складаються з невеликих фрагментів однакових габаритів, розташованих у сітці.

ВСТУП

Штучний інтелект (ШІ) був нерозривно пов'язаний з іграми з моменту їх появи. Перші приклади включають прості алгоритми прийняття рішень у таких іграх як "Pong" та "Space Invaders", де базовий ШІ контролював поведінку комп'ютерних супротивників. З розвитком технологій зростала і складність ШІ в іграх. У класичних іграх, наприклад "Pac-Man", використовувалися більш складні алгоритми для управління поведінкою привидів (супротивників у грі), що робило ігровий процес більш складним і захопливим. Ця тенденція продовжилася в настільних іграх, серед яких найбільш відомі шахи і Го, які розширили межі ШІ, вимагаючи від алгоритмів мислити на кілька ходів уперед і адаптуватися до людських стратегій.

Відносини між ШІ та іграми є симбіотичними. Ігри забезпечують контрольоване середовище, де алгоритми можна тестувати, оцінювати та вдосконалювати. І навпаки, досягнення в галузі штучного інтелекту постійно підвищують складність, реалістичність і захопливість відеоігор. Сьогодні штучний інтелект є популярним компонентом розробки ігор, забезпечуючи більш динамічний та захоплюючий досвід.

Актуальність проекту. На сучасному ігровому ринку, що стрімко розвивається, гравці вимагають більш динамічного та захопливого досвіду. Статичних, заздалегідь визначених ігрових світів і передбачуваної поведінки ШІ вже недостатньо, щоб захопити вимогливу аудиторію. Поява нових методів ШІ пропонує вирішення цієї проблеми шляхом створення різноманітного та непередбачуваного ігрового досвіду. Процедурна генерація дозволяє розробникам створювати величезні та різноманітні ігрові світи з мінімальним ручним втручанням, заощаджуючи час і ресурси та забезпечуючи унікальність кожного проходження гри.

Водночас керування персонажами за допомогою ШІ забезпечує більш складні та схожі на людські взаємодії. Традиційний ігровий ШІ часто слідує

заздалегідь визначеним сценаріям, які можуть швидко стати передбачуваними і нецікавими. Однак, використовуючи методи машинного навчання, ігрові персонажі можуть навчатися і адаптуватися до поведінки гравців, забезпечуючи більш захоплюючий і захоплюючий досвід.

Мета проекту. Основною метою цього проекту є розробка 2D відеогри у жанрі «платформер», який використовує методи процедурної генерації та машинного навчання для створення динамічного, захопливого та адаптивного ігрового процесу. Проект має на меті дослідити взаємодію між процедурною генерацією та штучним інтелектом, демонструючи, як ці технології можуть бути інтегровані для покращення ігрового дизайну. Гра матиме процедурно згенеровані рівні, що гарантуватиме унікальне середовище для кожного проходження, та використовуватиме персонажів, керованих ШІ за допомогою персептрону та навчених за допомогою ГА, забезпечуючи інтелектуальну та адаптивну поведінку, яка кидає виклик гравцям.

Об'єктом цього дослідження є 2D гра у жанрі «платформер», яка слугує експериментальною платформою для застосування та тестування інтеграції процедурної генерації та машинного навчання. Ця гра, розроблена за допомогою Python та PyGame, забезпечує контрольоване середовище для реалізації та оцінки запропонованих методологій.

Предметом дослідження є засоби та методи, що використовуються для поєднання процедурної генерації з машинним навчанням в контексті розробки ігор. Це включає в себе розробку та реалізацію алгоритму генерації процедур, архітектуру багатошарового персептрону та генетичний алгоритм, який використовується для навчання ШІ.

Наукова новизна отриманих результатів полягає в інноваційному підході до поєднання процедурної генерації та машинного навчання в єдиному ігровому середовищі. Це дослідження робить внесок у розвиток галузі, демонструючи, як ці технології можуть бути інтегровані для створення більш

динамічних та адаптивних ігрових світів, потенційно створюючи базу для майбутніх досліджень і розробок у галузі.

Практичне значення отриманих результатів полягає в потенційному застосуванні розроблених методологій в комерційній розробці та ігровій індустрії. Успішна реалізація цих методів може призвести до створення більш захопливих ігор з ШІ-персонажами, які демонструють природну та розумну поведінку. Це може значно покращити досвід гравців і надати нові інструменти для розробників ігор.

Теоретичне значення отриманих результатів це цінні ідеї та знання, які можуть бути застосовані в інших сферах досліджень та розробок. Реалізована гра сприяє розумінню того, як досліджені технології можуть працювати разом, відкриваючи перспективи для адаптації та розширення в майбутніх дослідженнях.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Машинне навчання в розробці ігор

Інтеграція машинного навчання у розробку ігор має багату історію, яка сягає корінням у перші дні розвитку штучного інтелекту (ШІ). Спочатку ШІ в іграх зосереджувався на простих системах, заснованих на правилах. Однак з розвитком обчислювальних потужностей і методів ШІ стали можливими більш складні моделі поведінки. Ця еволюція була позначена кількома ключовими етапами, які суттєво вплинули на ігрову індустрію. У 1950-х і 1960-х роках з'явилися перші приклади використання ШІ в іграх, коли були розроблені алгоритми гри в шашки та шахи. Шашкова програма Артура Семюеля, розроблена наприкінці 1950-х років, була однією з перших що використовували рудиментарну форму машинного навчання, а саме алгоритм навчання, який покращував свою продуктивність з часом завдяки накопиченню досвіду [1]. Цей період заклав основу для інтеграції алгоритмів навчання в ігровий ШІ. У 1970-х і 1980-х роках з'явилися відеоігри, в яких ШІ використовувався переважно для неігрових персонажів в аркадних іграх. Ці ранні ігрові ШІ були простими і детермінованими, покладаючись на заздалегідь визначені правила і шаблони. Однак дослідники та розробники почали досліджувати більш динамічні системи штучного інтелекту. Наприклад, у грі "Pac-Man" (1980) з'явилися привиди з різною поведінкою, що започаткувало концепцію різноманітних стратегій ШІ в межах одного ігрового середовища. Методи ШІ були застосовані до Pac-Man у [2].

У 1990-х роках, з розвитком персональних комп'ютерів та ігрових приставок, розширився потенціал для більш складного ШІ в іграх. "The Sims" (2000) використовувала ШІ для імітації реалістичної поведінки та взаємодії між віртуальними персонажами, створюючи більш захоплюючий і динамічний ігровий досвід.

21 століття стало свідком стрімкого прискорення застосування ML у розробці ігор. Розвиток глибокого навчання, зумовлений наявністю великих наборів даних і потужних графічних процесорів, уможливив створення надскладних систем ШІ. Складні ігри як "StarCraft II" почали застосовуватися як середа для глибокого навчання з підкріпленням і створення ШІ-супротивників, які можуть змагатися на професійному рівні [3]. Компанія DeepMind придбана Google розробила алгоритм AlphaGo, який використовував глибокі нейронні мережі та навчання з підкріпленням, щоб опанувати гру в го, перемігши чемпіонів світу серед людей і продемонструвавши потенціал машинного навчання у складних середовищах прийняття рішень [4].

1.2. Приклади застосування машинного навчання в розробці ігор

Машинне навчання має численні застосування в розробці ігор, покращуючи різні аспекти ігрового дизайну, геймплею та ігрового досвіду гравців. Нижче буде розглянуто кілька яскравих прикладів, що ілюструють різноманітні способи, якими машинне навчання змінює ігрову індустрію.

Машинне навчання все частіше використовується для автоматизації створення ігрового контенту - процесу, відомого як процедурна генерація контенту. Сюди входить генерація рівнів, карт і навіть цілих ігрових світів. Наприклад, "No Man's Sky" розроблена Hello Games і випущена у 2016 році, - це космічна гра с фокусом на дослідження світу, яка використовує процедурну генерацію для створення цілого всесвіту. У грі представлено 18 квінтильйонів планет, кожна з яких має унікальні екосистеми, ландшафти та форми життя. Процедурні алгоритми генерують рельєф, флору, фауну і навіть атмосферні умови кожної планети, забезпечуючи гравцям практично нескінченний і різноманітний ігровий світ для дослідження. Цікаво що саме ця гра надихнула на написання цієї бакалаврської роботи та довела що створювати цілі планети автоматизовано цілком можливо, хоча в початкових версіях розробники не спромоглися виконати усі обіцянки. Проблеми з розробкою підірвали довіру користувачів до моделі створення контенту та ледь не призвели до закриття

подібних проектів, але зрештою віртуальний світ наповнився життям як ніколи раніше. Впроваджуючи методи ML, розробники можуть створювати більш складний і різноманітний контент, який адаптується до дій та вподобань гравця, підвищуючи цікавість і занурення [5].

Адаптивний ШІ — це ігровий персонаж, який може навчатися та адаптуватися до поведінки гравця, забезпечуючи більш персоналізований та складний досвід. Алгоритми машинного навчання аналізують дані про гравця, щоб динамічно коригувати рівень складності та стратегії ШІ-супротивників. Яскравим прикладом використання цього інструменту є гра "Left 4 Dead", у якій було реалізовано ШІ-режисера, який відстежує поведінку гравця та адаптує складність і темп гри в режимі реального часу, забезпечуючи збалансований і захопливий досвід для гравців усіх рівнів. Моделювання гравців передбачає створення детальних профілів гравців на основі їхньої поведінки в грі. Ці профілі можна використовувати для адаптації ігрового досвіду до індивідуальних уподобань і стилів. Алгоритми машинного навчання аналізують такі дані, як час гри, вибір і рівень успішності, щоб побудувати ці моделі. Ця інформація може бути використана для рекомендації контенту або налаштування ігрової механіки відповідно до вподобань гравця [6].

Навчання з підкріпленням (reinforcement learning, RL) — це тип машинного навчання в якому агенти вчаться приймати рішення взаємодіючи з навколишнім середовищем і отримуючи зворотний зв'язок у вигляді винагород або покарань. RL використовується для створення висококваліфікованих ШІ-супротивників у різних іграх. Наприклад, Five від компанії OpenAI, це команда ШІ-агентів, навчених за допомогою RL, що досягла надлюдських показників у грі "Dota 2", продемонструвавши потенціал у складних стратегічних іграх у реальному часі [8].

Обробка природної мови (natural language processing, NLP) дозволяє іграм включати більш складні діалогові системи та інтерактивну розповідь. Використовуючи моделі машинного навчання, натреновані на великих

текстових масивах, ігри можуть генерувати реалістичні та контекстно-залежні діалоги для NPC. Застосування цього інструменту у комерційних проектах наразі викликає деякі складнощі, але вже є приклади коли незалежні розробники спромоглися модифікувати "The Elder Scrolls V: Skyrim", використавши NLP для створення більш цікавих і різноманітних діалогів з NPC. Крім того, чат-боти та віртуальні помічники в іграх можуть надавати підказки, навчальні посібники та підтримку, покращуючи загальний досвід гравця [7].

Методи машинного навчання також використовуються для покращення візуальних аспектів ігор. Нейронні мережі можуть генерувати високоякісні текстури, покращувати зображення та створювати реалістичну анімацію. Технологія Nvidia DLSS (Deep Learning Super Sampling) використовує нейронні мережі для масштабування зображень з низькою роздільною здатністю в реальному часі, забезпечуючи високоякісну графіку без зменшення продуктивності, яке зазвичай пов'язане з високою роздільною здатністю. Аналогічно, алгоритми ML можна використовувати для створення більш реалістичної анімації, навчаючись на основі даних захоплення руху, роблячи рухи персонажів більш плавними та реалістичними [9]. Автоматизоване тестування ігор - ще одна сфера, де ML робить значний внесок. Традиційне тестування ігор зазвичай вимагає значних зусиль для виявлення помилок і забезпечення функціонування гри. Алгоритми машинного навчання можуть автоматизувати цей процес, імітуючи проходження гри людиною і виявляючи проблеми більш ефективно. Ці алгоритми можуть навчитися досліджувати ігрове середовище, виконувати складні послідовності дій і виявляти аномалії, значно скорочуючи час і витрати, пов'язані з тестуванням ігор [10].

Інтеграція машинного навчання в розробку ігор революціонізує індустрію, пропонуючи нові можливості для творчості. Оскільки методи ML продовжують розвиватися, їх застосування в іграх, ймовірно, буде

розширюватися, що призведе до ще більш інноваційного та захоплюючого досвіду для гравців.

1.3. Процедурна генерація

Процедурна генерація — це метод створення даних алгоритмічно, який широко використовується в розробці ігор для динамічної генерації контенту. Ця методика має вирішальне значення в іграх з кількох причин, зокрема для збільшення якісного контенту у грі, економії часу на розробку та створення великих і складних ігрових світів, які неможливо розробити вручну. Процедурна генерація спирається на алгоритми, які використовують випадковість і заздалегідь визначені правила для створення контенту, що може змінюватися з кожним проходженням гри. Цей підхід дозволяє розробникам створювати величезні обсяги контенту без необхідності великих ручних зусиль, що дає можливість розробляти ігри з багатими та різноманітними середовищами, рівнями та сюжетними лініями.

Створюючи унікальний контент для кожного проходження, процедурна генерація гарантує, що гравці отримують різний досвід кожного разу, коли грають у гру. Це збільшує довговічність гри та утримує гравців у грі. Гра "Rogue" (1980), що є однією з найперших прикладів процедурної генерації, ілюструє цю перевагу, пропонуючи випадково згенеровані макети підземель при кожному запуску гри.

Процедурна генерація значно скорочує час і ресурси, необхідні для створення ігрового контенту. Розробники можуть створювати алгоритми, які генерують величезні ігрові світи, зменшуючи потребу в ручному створенні контенту та дозволяючи ефективно використовувати ресурси розробки. Ця ефективність особливо очевидна у великих іграх з відкритим світом, де створення кожного елемента вручну було б непрактичним. Наприклад, раніше згадана "No Man's Sky" використовує процедурну генерацію для створення практично нескінченного всесвіту з різноманітними планетами та

екосистемами, кожна з яких має унікальні характеристики та ландшафти. Процедурна генерація дозволяє створювати несподіваний та унікальний контент, який може посилити відчуття відкриття та дослідження у гравця.

1.4. Методи процедурної генерації для розробки відеоігор

Рандомізація — це найпростіша форма процедурної генерації, коли елементи розміщуються випадковим чином у ігровому світі. Ця техніка часто використовується для генерації предметів, ворогів або інших елементів в іграх. Хоча рандомізація може забезпечити варіативність, їй бракує структури та узгодженості більш просунутих процедурних методів. Щоб вирішити цю проблему, рандомізацію часто поєднують з іншими методами, щоб згенерований контент був придатним для гри та цікавим. Недоліком використання рандомізації в цьому проекті було те, що зрештою вона створює середовище, яке гравець не може подолати.

Генерація на основі граматики, яка була використана в проекті, використовує набір заздалегідь визначених правил для генерації контенту. Ця техніка особливо корисна для створення складних структур. Визначивши граматику, яка вказує, як можна комбінувати різні елементи, розробники можуть створювати зв'язний і різноманітний контент. Наприклад, гра "Spelunky" використовує граматичний підхід для створення своїх процедурно згенерованих рівнів.

Шум Перліна - це градієнтна шумова функція, яка використовується для створення плавних, природних текстур і рельєфу. Ця техніка широко використовується в розробці ігор здебільшого для створення графічних елементів, наприклад ландшафтів, хмар та інших природних явищ [11]. Шум Перліна дозволяє створювати послідовні та безперервні візерунки, що робить його ідеальним для генерації рельєфу в таких іграх, як "Minecraft".

Клітинні автомати — це обчислювальні моделі, які складаються з сітки комірок, кожна з яких може перебувати у скінченній кількості станів. Стан кожної клітини визначається набором правил, заснованих на станах сусідніх клітин. Популярна гра "Terraria" використовує клітинні автомати для створення своїх складних підземних середовищ [12].

Фрактали - це математичні множини, які демонструють самоподібність на різних масштабах. У процедурній генерації фрактали використовуються для створення детальних і складних патернів, які можуть бути застосовані для генерації рельєфу, створення текстур тощо. Використання фракталів дозволяє створювати деталізовані середовища зі складними візерунками та структурами.

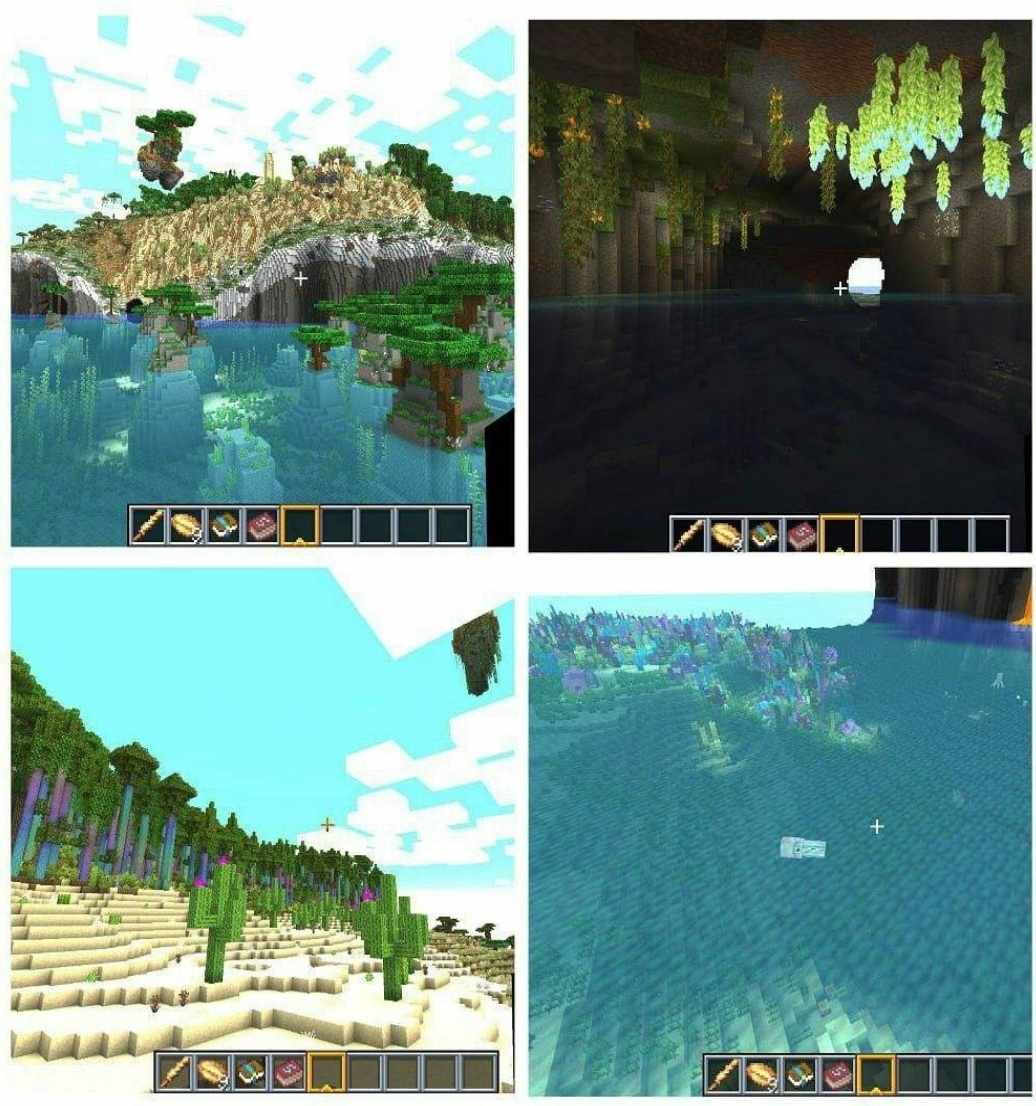


Рис. 1: Скріншоти з гри Minecraft. Версія станом на 10.06.24.

Системи Ліндемайєра — це тип формальної граматики, що використовується

для моделювання процесів росту рослин та інших організмів. У розробці ігор L-системи використовуються для генерації фракталоподібних структур, таких як дерева та інша рослинність. Ця техніка дозволяє створювати складні, органічні форми, які посилюють візуальне багатство ігрового середовища [13].

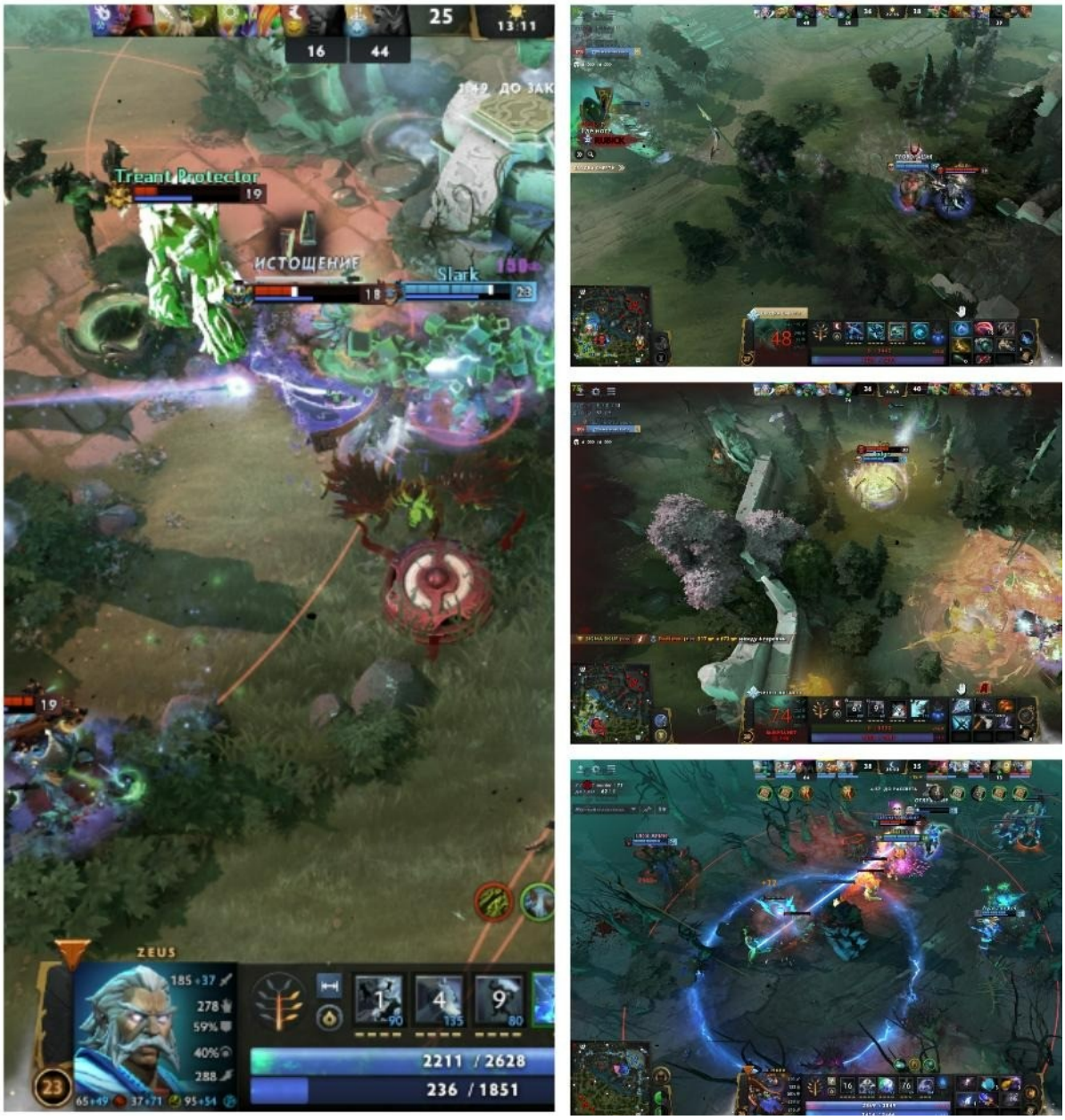
Методи машинного навчання все частіше використовуються в процедурній генерації для створення більш складного та адаптивного контенту. Наприклад, нейронні мережі можна навчити генерувати рівні, текстури та інші ігрові елементи на основі великих наборів даних. Такий підхід дозволяє створювати контент, який може адаптуватися до дій та вподобань гравця, забезпечуючи більш персоналізований та захопливий досвід.

1.5. Приклади застосування методів процедурної генерації у розробці відеоігор

Рис. 2: Скріншоти з гри Terraria. Версія Steam станом на 10.06.24.

"Rogue" (1980) — один з найперших і найвпливовіших прикладів процедурної генерації

в іграх. Гра має випадково згенеровані рівні підземелля, що робить кожне проходження унікальним. "Rogue" заклала основу для жанру roguelike, який характеризується процедурно згенерованими рівнями та цікавим повторним проходженням.



"Minecraft", розроблена студією Mojang Studios і випущена у 2011 році — це гра-пісочниця, яка використовує процедурну генерацію для створення свого величезного світу побудованого з блоків. Місцевість, печери та біоми гри генеруються за допомогою шуму Перліна та інших процедурних методів, що

дозволяє гравцям досліджувати нескінченний світ з різноманітними ландшафтами та ресурсами. "Minecraft" стала однією з найбільш продаваних ігор усіх часів, демонструючи привабливість процедурно згенерованого контенту.

Створена Дереком Ю у 2008 році "Spelunky" — це roguelike платформер, який використовує процедурну генерацію для створення своїх складних рівнів. Кожне проходження гри має новий набір рівнів з випадково розташованими ворогами, пастками та скарбами. Процедурна система генерації гри гарантує, що гравці стикаються з новими викликами та сюрпризами щоразу, коли вони грають, що сприяє цікавості повторного проходження та популярності гри.

"The Binding of Isaac" (2011) створена Едмундом МакМілленом — це ще один roguelike, який використовує процедурну генерацію для створення рівнів, ворогів і предметів. Кожне проходження гри є унікальним, з різним розташуванням кімнат, комбінаціями ворогів та предметами. Процедурна система генерації гри створює динамічний і непередбачуваний досвід, утримуючи гравців залученими і кидаючи їм виклик

"Terraria" (2011) від компанії Re-Logic, є пригодницькою грою-пісочницею, яка використовує процедурну генерацію для створення свого 2D-світу. Місцевість, біоми та підземні печерні системи гри генеруються за допомогою клітинних автоматів та інших процедурних методів. Це дозволяє гравцям досліджувати різноманітний і постійно мінливий світ, наповнений ресурсами, ворогами і прихованими скарбами.

"Dwarf Fortress", розроблена Bay 12 Games і випущена в 2006 році, є симулятором, відомим своєю складною і деталізованою генерацією світу. Гра використовує процедурну генерацію для створення всього світу, включаючи місцевість, цивілізації, історію та події. Кожна гра починається з унікального світу, що надає безмежні можливості для дослідження та розповіді.

Ці приклади ілюструють різноманітні застосування процедурної генерації у розробці ігор. Використовуючи процедурні методи, розробники можуть створювати багаті, динамічні та захопливі ігрові світи, які пропонують гравцям новий досвід з кожним проходженням.

1.6. Методи штучного інтелекту для розробки відеоігор

У розробці ігор для створення інтелектуальної поведінки та динамічного контенту використовують багато методів штучного інтелекту. Ці методи варіюються від простих систем на основі правил до складних алгоритмів машинного навчання.

Машини скінченних станів (МСС) — одна з найпростіших технік ШІ, що використовується в іграх. МСС складається зі скінченної кількості станів, переходів між цими станами та дій, пов'язаних з кожним станом. МСС особливо корисні для моделювання передбачуваної та дискретної поведінки, наприклад, патрулювання ворога або діалогових систем. Наприклад, у багатьох іграх жанру «стелс» персонажі охоронців керуються МСС для зміни станів таких як патрулювання, переслідування та повернення на початкову позицію [14].

Дерева поведінки — це ієрархічні моделі, які використовуються для управління поведінкою NPC. Дерева поведінки є більш гнучкими та масштабованими ніж МСС, що дозволяє складати складні моделі поведінки з простіших завдань. Кожен вузол у дереві це завдання, а структура дерева визначає порядок виконання цих завдань. Такий підхід широко використовується в сучасному ігровому ШІ для керування ворогами у іграх жанру «шутер від першої особи» та прийняття рішень у стратегічних іграх [15].

Пошук шляху має вирішальне значення для пересування NPC та навігації в ігровому світі. Алгоритм A* є найпоширенішим алгоритмом пошуку шляху в іграх. Він знаходить найкоротший шлях між двома точками, оцінюючи

потенційні шляхи на основі евристичної функції. Цей алгоритм є ефективним і гарантує найкоротший шлях, що робить його ідеальним для стратегій у реальному часі [16], наприклад серії ігор "StarCraft" або "Age of Empires", що використовують A* для переміщення та навігації ігрових одиниць (юнітів).

Методи машинного навчання, зокрема нейронні мережі, стають все більш поширеними у розробці ігор. Нейронні мережі можуть навчатися на основі даних і з часом покращувати свою продуктивність, що робить їх придатними для таких завдань, як керування персонажами, процедурне створення контенту та адаптивне регулювання складності. Навчання з підкріпленням, підвид машинного навчання, передбачає навчання агентів приймати рішення, заохочуючи бажану поведінку і караючи небажану. Цей підхід був використаний для створення ШІ-агентів, які можуть змагатися на рівні людини в таких складних іграх, як "Dota 2" і "StarCraft II".

1.7. Висновок до першого розділу

Інтеграція машинного навчання у розробку ігор продовжує революціонізувати індустрію, пропонуючи безпрецедентні можливості для творчості та інновацій. Від перших простих алгоритмів до сучасних складних моделей поведінки, ШІ та машинне навчання стали невід'ємною частиною створення захопливих і динамічних ігрових світів. Історичний розвиток технологій, таких як алгоритми Артура Семюеля та глибокі нейронні мережі, демонструє поступове зростання складності та адаптивності ігрових систем. Наприклад, успіх AlphaGo від DeepMind вказує на потенціал ШІ у вирішенні надскладних завдань у реальному часі, що знаходить відображення в сучасних стратегічних іграх. Застосування машинного навчання не обмежується лише супротивниками у іграх: воно також охоплює створення процедурно згенерованих світів, адаптивних NPC та реалістичних анімацій, що підвищують занурення гравців у гру.

Поглиблене використання машинного навчання у розробці ігор також відкриває нові горизонти для автоматизації та персоналізації ігрового контенту. Процедурна генерація, адаптивний ШІ та технології, такі як Nvidia DLSS, демонструють, як нейронні мережі можуть значно покращити як якість візуального контенту, так і ігровий досвід. Автоматизоване тестування з використанням машинного навчання дозволяє виявляти та виправляти помилки більш ефективно, що скорочує час та витрати на розробку. В цілому, інтеграція машинного навчання продовжує розширювати можливості гейм-дизайнерів, сприяючи створенню унікальних і захопливих ігрових продуктів, які адаптуються до потреб та вподобань гравців. Як наслідок, майбутнє ігрової індустрії виглядає надзвичайно перспективним, з безліччю нових можливостей для інновацій та творчого розвитку.