

Thesis for Master's Degree

RPG Gym: RPG Game Simulator for Deep Learning

Donghyeok Park

School of Integrated Technology
Gwangju Institute of Science and Technology

2023

Thesis for Master's Degree

RPG Gym: RPG Game Simulator for Deep Learning

Donghyeok Park

School of Integrated Technology
Gwangju Institute of Science and Technology

2023

RPG Gym: RPG Game Simulator for
Deep Learning

RPG Gym: 딥러닝을 위한 롤 플레잉
게임 시뮬레이터

RPG Gym: RPG Game Simulator for Deep Learning

Advisor : Professor Kyung-Joong Kim

by

Donghyeok Park

School of Integrated Technology
Gwangju Institute of Science and Technology

A thesis submitted to the faculty of the Gwangju Institute of
Science and Technology in partial fulfillment of the
requirements for the degree of Master of Science in the
School of Integrated Technology

Gwangju, Republic of Korea
2022. 12. 21.

Approved by


(서명)
Professor Kyung-Joong Kim
Committee Chair

RPG Gym: RPG Game Simulator for Deep Learning

Donghyeok Park

Accepted in partial fulfillment of the requirements for the
degree Master of Science

12. 21. 2022

Committee Chair

(서명)

Prof. Kyung-Joong Kim

Committee Member

(서명)

Prof. Seungjun Kim

Committee Member

(서명)

Prof. Jin-Hyuk Hong

MS/ IN(CT) Donghyeok Park(박동혁). RPG Gym: RPG Game Simulator for Deep Learning(RPG Gym: 딥러닝을 위한 롤 플레이 게임 시뮬레이터). School of Integrated Technology. 2023. 31p. Prof. Kyung-Joong Kim
20201148

Abstract

Recently, artificial intelligence research using deep learning has been actively conducted. In particular, research related to artificial intelligence is being verified for performance in game environments, and being applied to improve the quality of content in games in the game industry. For example, game balance is adjusted using various data analysis and artificial intelligence methods. In addition, imitation learning or reinforcement learning is used to diversify game characters' behavior control and improve performance. However, there is no suitable simulator for conducting these studies and few simulators are similar to commercial games. OpenAI Gym or Mujoco that are mainly used to learn character behavior control, but these are very different from commercial games. Therefore, this paper proposes a new game simulator with an ARPG game structure that can be used in game balance adjusting and deep learning. The proposed simulator was implemented using Unreal Engine 5, a commercial game engine, and API for applying a deep learning framework is designed and provided. In addition, we conduct game balance experiments and character behavior control experiments using deep learning in the proposed simulator and measure performance.

Contents

Abstract	i
Contents	ii
List of Tables	iii
List of Figures	iv
I. Introduction	1
II. Background	3
2. 1. Action Role-Playing Game	3
2. 2. Game Engine	3
2. 3. Game Simulator for Deep Learning	4
2. 4. Simulator using Game Engine	5
2. 5. Deep Learning in Commercial Game	5
III. ARPG Game Simulator	8
3. 1. Game Features	9
3. 1. 1. Non-Targeting System	9
3. 1. 2. Characters	10
3. 1. 3. Enemy Monsters	13
3. 1. 4. Maps and Game Rule	15
3. 2. Simulator Features	16
3. 2. 1. Remote Procedure Call Component	16
3. 2. 2. Play Logging Component	18
3. 2. 3. Configuration Component	19
3. 2. 4. Extra Features	19
IV. Experiments and Result	21
4. 1. Game Balance Adjusting Test	22
4. 2. Imitation Learning Test	24
4. 3. Reinforcement Learning Test	25
V. Discussion	26
VI. Conclusion	28
References	29

LIST OF TABLES

Table 1. Character ‘Gideon’ attack and skills	11
Table 2. Character ‘Kwang’ attack and skills	11
Table 3. Character ‘Shinbi’ attack and skills	11
Table 4. Character ‘Sparrow’ attack and skills	12
Table 5. Character ‘Twinblast’ attack and skills	12
Table 6. Character ‘Wukong’ attack and skills	12
Table 7. Enemy monster ‘Countess’ attack and skills	14
Table 8. Enemy monster ‘Rampage’ attack and skills	14
Table 9. Enemy monster ‘Sevarog’ attack and skills	14
Table 10. Enemy monster ‘Terra’ attack and skills	15
Table 11. List of Python APIs	17
Table 12. List of player log data	18
Table 13. List of game result data	18
Table 14. Initial experimental parameters	22
Table 15. Balance adjusting experiment results	23
Table 16. Balance adjusting experiment results considering time	24
Table 16. Result of imitation learning test	25
Table 16. Result of reinforcement learning test	26

LIST OF FIGURES

Figure 1. A variety of ARPG genre games	3
Figure 2. General game engine structure	4
Figure 3. Commonly used commercial game engines	4
Figure 4. Self-driving car or drone simulator	5
Figure 5. Learning environments used for deep learning	6
Figure 6. Commercial games used for deep learning	7
Figure 7. The structure of the proposed RPG Gym	8
Figure 8. Various non-targeting systems in the RPG Gym	9
Figure 9. Behavior tree applied to each character	13
Figure 10. Behavior tree applied to each monster	15
Figure 11. Four maps included in RPG Gym	16
Figure 12. Structure of the RPC components	17
Figure 13. Imitation learning loss graph	24
Figure 14. Reinforcement learning reward graph	26

I. INTRODUCTION

Recently, various deep learning models have been introduced, and research that can be applied to the game industry is also being actively conducted. Examples of applications of artificial intelligence in the game industry include map creation, character balancing, abuse detection, and character behavior control. Map generation is being studied to create a map capable of game play by appropriately placing terrain or objects, which are the background elements of the game [1]. Character balancing is being studied to find the right ability so that the ability of the character, the subject of game play, is too high or too low to affect game play [2]. Abuse detection is the detection of behaviors that can adversely affect gameplay, and studies are being conducted to detect behaviors that use illegal programs or take unjustified gains [3]. Research on character control is to find ways to control character behavior that can be played like humans or are better than humans [4]. In order to solve problems, data or a simulator is needed. The game company owns its own game and can get enough data from the game, but researchers use limited environments or conduct research in environments that are different from real games. RPG is a game genre that includes various genres of game features and problems. When proceeding with game content with multiple players, there are several variables. These variables affect the difficulty or balance of game content. It is possible to attempt interactions such as combat or collaboration with a Non-Player Character (NPC), an important component of the game. In addition, behavioral control can be recorded and it is able to be replaced with better behavioral control algorithms to enhance the performance of each character or NPC.

Therefore, to simulate and solve these problems, this paper proposes RPG Gym, a game environment simulator with features similar to real games. RPG Gym is a

game simulator that imitates the features of the Action Role Playing game and provides a variety of environments and functions. The simulator includes six characters, four maps, and three enemy monsters. In addition, some components of the game were designed to be customized so that various studies such as character balancing, agent learning, and generalization could be conducted. RPG Gym uses a non-targeting system, an attack judgment method used in several games, to reproduce it at a level similar to that of commercial games. There are six types of characters provided, and each character uses basic attacks and eight skills. In addition, when conducting game content or performing artificial intelligence performance tests with human players, the behavior tree has been implemented so that it can operate on its own for comparative tests. There are four enemy monsters, each using a basic attack and five skills. The behavior tree was applied to deal with the character. RPG Gym can select the included characters, monsters, and maps to organize the raid content needed, and can set the character and monster's health points, attack power, and skills to use in detail. In this study, experiments conducted using RPG Gym simulate balance adjusting and agent performance problems addressed in the game industry based on raid content provided by RPG Gym and demonstrate various experimental possibilities in the proposed game simulator.

II. BACKGROUND

2. 1. Action Role Playing Game

Role Playing Game (RPG) is a game in which players control characters that express themselves and play a given role. Action Role Playing Game (ARPG) is a game genre that combines existing Role Playing games and action games. In ARPG games, players control characters directly using a keyboard and mouse or other controller equipment. ARPG games use the character's dynamic combat system. In order to implement ARPG games, Various additional elements such as character animation, Virtual Effects (VFX), and Sound Effects (SFX) are added. Most of the recently released games use elements of the ARPG genre, and various quest systems, resonance systems, and multi-player systems are additionally provided depending on the game plan. In addition, it can be classified into 2D or 3D games depending on whether it uses flat graphics or 3D geometry and can be classified into various genres depending on the components.



Figure 3. A variety of ARPG genre games. (a) Elden Ring [5], (b) Lost Ark [6], (c) Diablo II [7]

2. 2 Game Engine

Game engines are software for creating games and include several libraries necessary for game development. In addition, game engines are used not only in games but also in various fields such as architecture, medical practice, and aviation.

In particular, because game engines are mindful of reuse, they are developed so that they can be used in several types of games without being subordinated to one game. The game engine outputs visual elements using a 2D or 3D renderer or reflects user input. In addition, since it includes a physical engine, physical characteristics can be included in each object. In addition, it includes functions such as synchronization, artificial authentication, and scripting through a network.

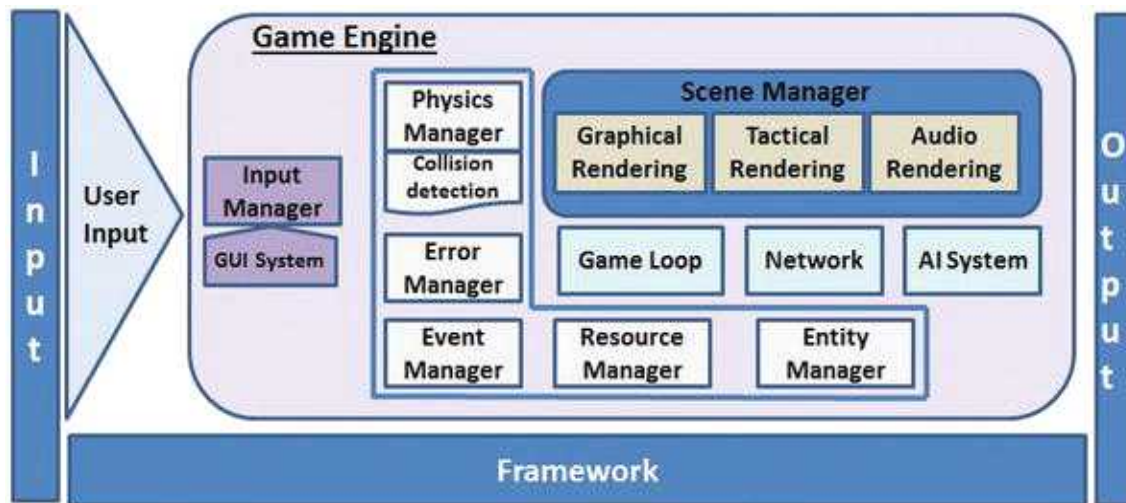


Figure 4. General game engine structure, The game engine includes various subsystems [8].



(a)



(b)



(c)

Figure 5. Commonly used commercial game engines. (a) Unreal Engine [9], (b) Unity [10], Cry Engine [11]

2. 3 Simulator using Game Engine

The game engine uses a rendering engine that produces realistic visual effects and a physical engine that imitates the physical laws of the real world or allows you to set up physical laws freely. In addition to game production, the functions of game engines that can reflect the real world are used in industries that require sophisticated simulations such as automobiles, aviation, and ships. With the recent increase in the performance of deep learning algorithms, deep learning-based

artificial intelligence is being applied in the field of autonomous vehicles, and sophisticated simulators that imitate the real world are being introduced to test them.

The AirSim [12] simulator is an open-source simulator implemented based on Unreal Engine 4 that provides an environment that includes a variety of environments that can simulate autonomous vehicles and drones. Since it is provided in the form of an Unreal plug-in so that users can easily modify it, it is possible to modify some of the environment or create a new type of environment. In addition, it provides a Python API that can obtain or control vehicle information for smooth deep learning model learning, so it can be easily used in simulators. The Carla [13] simulator is an open-source autonomous vehicle simulator using Unreal Engine 4 that provides a variety of urban environments and is designed to enable autonomous vehicles to drive. The Carla simulator provides several urban environments and can create the necessary environments through its map editor. In addition, by providing an API, it is possible to control non-player characters (NPC) such as vehicles, objects, and pedestrians present in the simulator, and provide various scenarios that may occur while driving in the simulator.

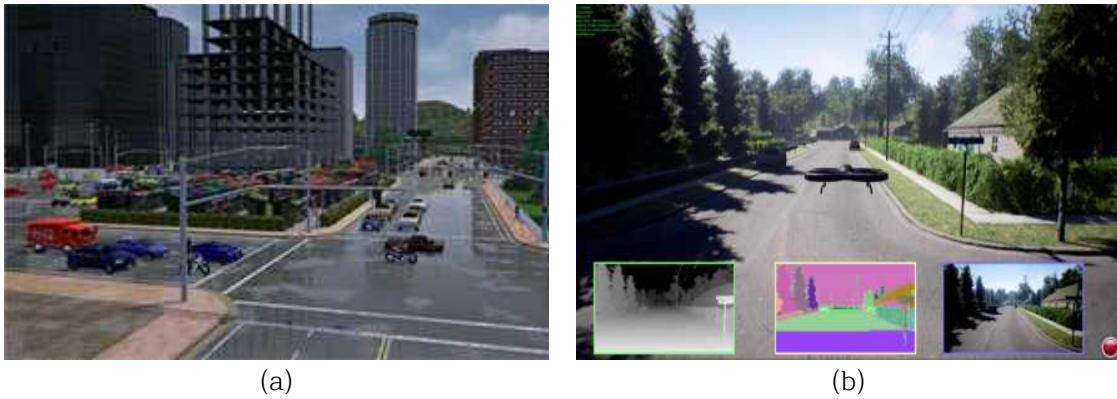


Figure 6. Self-driving car or drone simulator. (a) Carla simulator [12], (b) Airsim simulator [13]

2. 4 Game Simulator for Deep Learning

There are various platforms for applying deep learning and verifying performance, and it is mainly used by OpenAI Gym [14] environments and Mujoco [15] environments. The OpenAI Gym environment includes several Atari games and is designed to provide a Gym Environment API to facilitate the exchange of state and agent-determined actions in the environment. The Mujoco environment does not

exist in an environment with game characteristics, but the laws of physics are applied to the environment, and various environments such as robot arms and human walking are provided. Mujoco environment also provides APIs that make it easier for the environment and agents to exchange information with each other. Also, there is Neural MMO [16], an environment where large-scale multiple agents of OpenAI can be learned. Neural MMO provides an environment in which multiple agents interact with each other and researchers can learn multi-agent deep learning algorithms in this environment. Neural MMO acquire elements necessary for survival while exploring the provided environment or through combat with other agents.

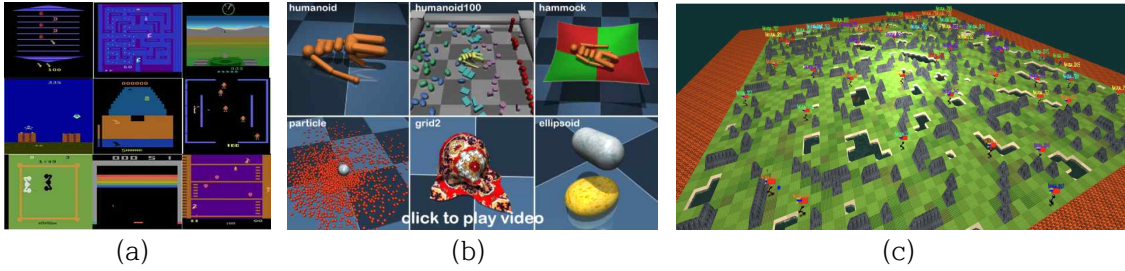


Figure 7. Learning environments often used for deep learning. (a) OpenAI Gym, (b) Mujoco, (c) Neural MMO

2. 5 Deep Learning in Commercial Game

The three game simulators described in Section 2.4 are relatively simple simulators that researchers use to prove deep learning models or other algorithms, and there are many differences from commercial games. However, there are cases where deep learning is applied to commercial games. The three game simulators described in Section 2.4 are relatively simple simulators that researchers use to prove deep learning models or other algorithms, and there are many differences from commercial games. However, there are cases where deep learning is applied to commercial games. First, StarCraft II, developed by Blizzard, is a real-time strategy (RTS) game in which players compete against their opponents by utilizing elements in a given game. Many players can use it together, but the game is usually played one-on-one. AlphaStar [17] is a StarCraft II agent developed by Google DeepMind that has won against humans. Alpha Start studied human play in the initial learning, reached a certain level, and based on this, reinforcement learning was applied to compete with each other to proceed with learning.

Second, DOTA 2, developed by Valve Corporation, is a multi-player online battle arena (MOBA) genre game in which ten players choose characters to manipulate and play 5 versus 5. DOTA2 requires interaction between team members, unlike StarCraft, in a way that a five-member team competes with the other team to win. In addition, there are more variables because the level, item, and skill level of the character to be manipulated must be efficiently managed. OpenAI has studied DOTA2 artificial intelligence, OpenAI Five [18], which performs well enough to win against professional gamers. Unlike StarCraft II study, OpenAI Five conducted learning using only reinforcement learning without using human data.

Finally, Blade & Soul, developed by NC Soft, is a massively multi-player online role-playing game (MMORPG) genre, in which players foster their characters in various ways. The game offers a variety of content, of which arena battle content called Bimu is fighting content in which players compete. NC Soft has studied deep learning-based artificial intelligence [19] applied to video content. This deep learning model uses human data to reach a certain level for initial learning and then uses reinforcement learning to improve the level. In addition, three types of models, offensive, defensive, and complex, were created by reflecting the fighting content, and it was shown the performance of winning against professional gamers



Figure 8. Commercial games used for deep learning. (a) StarCraft II, (b) DOTA 2, (c) Blade & Soul

III. ARPG GAME SIMULATOR

This section describes the process of developing RPG Gym, an ARPG gaming environment that can simulate problems addressed by the gaming industry. The ARPG gaming environment was developed using Unreal Engine 5. Characters used as components of the game used Paragon 3D modeling provided by Epic Games for free, while animation and VFX of characters were purchased and used in Epic Games' Marketplace to realize the character's movement and skill effects. Figure 7 is shown RPG Gym structure.

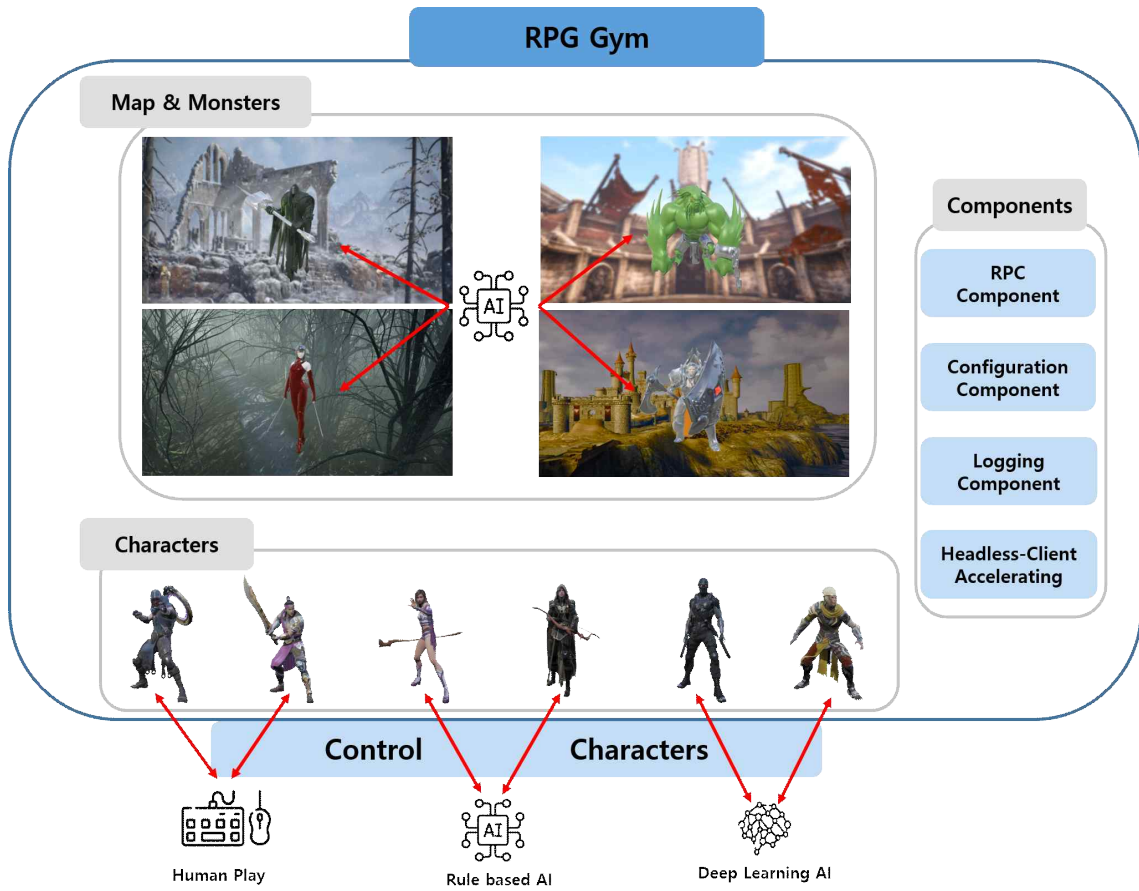


Figure 9. The structure of the proposed RPG Gym

3. 1. Game Features

3, 1. 1. Non-Targeting System

In ARPG games, the combat system is one of the components of gameplay. The method of dealing damage to the opponent in a combat system is classified into a targeting system and a non-targeting system. A targeting system is a system that can control characters, and automatically hold onto the other person, automatically look at the caught person. Even it can be dealing damage without looking. On the other hand, a non-targeting system is a damage transfer method mostly used in ARPG games, where a collision event occurs when two objects touch each other's collision areas and handles logic such as damage dealing when the event occurs.

RPG Gym uses a non-targeting system, and in the case of characters with melee attacks, it delivers damage when a collision event occurs based on the area of the weapon owned by the character and the collision area of the other character. RPG Gym uses a non-targeting system, and in the case of characters with melee attacks, it deals damage when a collision event occurs based on the area of the weapon owned by the character and the collision area of the other character. In the case of a range attack character, the damage is dealt when a collision event occurs based on the projectile collision area fired by the character and the collision area of the other character.

For skills used by characters, shape-based collision areas such as spheres, boxes, and capsules are set in the skill range and damage is dealt when the other party is in the range. However, when setting the shape-based collision area, it is difficult to detect other ranges because the Collision Shape provided by the Unreal Engine is a sphere, a box, and a capsule. Therefore, in to obtain a collision area other than a given collision shape, use dot product two character positions in a large collision shape. Both basic attacks and skills implemented in RPG Gym were implemented in this way.

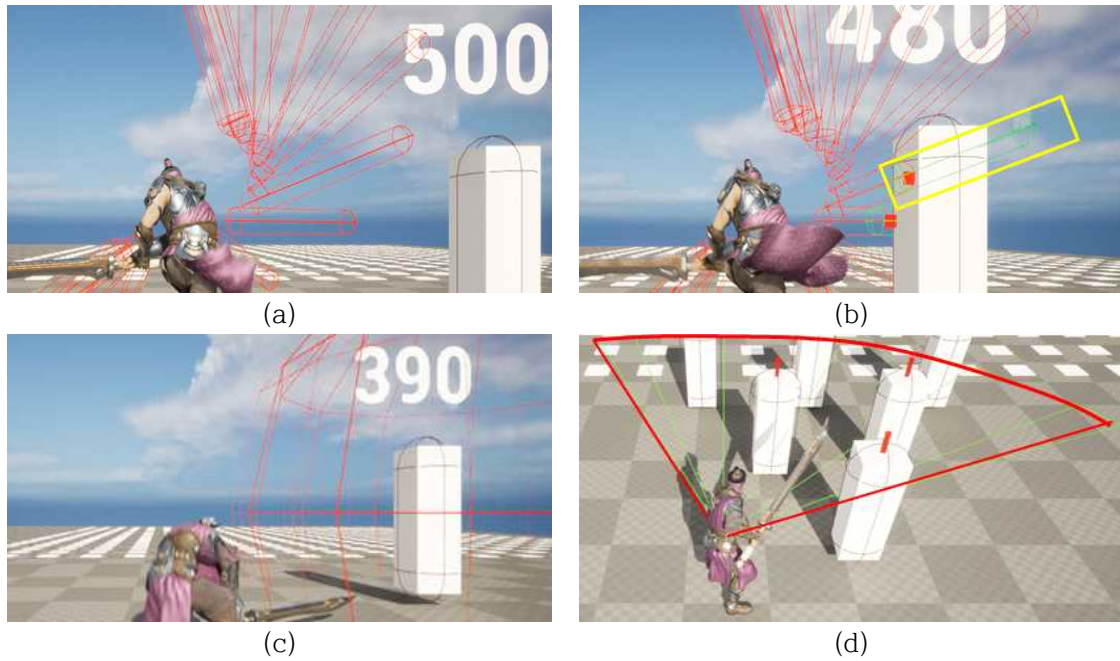


Figure 10. Various non-targeting systems in the RPG Gym.

3. 1. 2. Characters

RPG Gym included six types of characters, each character with different animations, skills, and VFX. Each character contains a basic attack, and the basic attack and attack skills damage the monster. RPG Gym included six types of characters, each character with different animations, skills, and VFX. The character's movement speed was set to 700cm/s, and jumping was set to be impossible. The movement of character uses a keyboard and a mouse. The character can move in the viewing direction by pressing the 'w' key, and the viewing direction can be controlled by moving the x-axis of the mouse.

Each character has a Health Point (HP) that represents life and decreases when attacked by the other person. It includes basic attacks and eight skills, and four of the eight skills are available. Each character's skills included different attack forms, damage, and reuse latency. The details of HP, attack power, skills, and skills applied to the character can be set by the user. From table 1 to table 6 describe attack and skill information used by each character, which is a component of the game environment. Delay refers to the animation playback time that ends after the action is started, and the skill can not perform other controls while the animation is being played.

Table 1. Character 'Gideon' attack and skills


	Skill Name	Delay(sec)	Description
	Attack	0.7	Melee attack
	Skill 0	1.78	Melee attack
	Skill 1	1.9	Melee attack
	Skill 2	0.97	Recover HP
	Skill 3	3.28	Melee attack
	Skill 4	2.0	A long-range attack
	Skill 5	2.78	A medium-range attack
	Skill 6	1.75	A medium-range attack
	Skill 7	2.83	A medium-range attack

Table 2. Character 'Kwang' attack and skills


	Skill Name	Delay(sec)	Description
	Attack	1.27	Melee attack
	Skill 0	3.92	Melee attack
	Skill 1	3.67	Melee attack
	Skill 2	0.97	Recover HP
	Skill 3	4.58	Melee attack
	Skill 4	4.07	A medium-range attack
	Skill 5	3.0	A medium-range attack
	Skill 6	3.67	A long-range attack
	Skill 7	3.67	A long-range attack

Table 3. Character 'Shinbi' attack and skills


	Skill Name	Delay(sec)	Description
	Attack	1.17	Range attack
	Skill 0	2.35	Range attack
	Skill 1	3.83	A medium-range attack
	Skill 2	1.83	Recover HP
	Skill 3	2.5	A long-range attack
	Skill 4	2.0	A medium-range attack
	Skill 5	3.17	A medium-range attack
	Skill 6	3.37	A long-range attack
	Skill 7	4.7	A long-range attack

Table 4. Character 'Sparrow' attack and skills


	Skill Name	Delay(sec)	Description
	Attack	0.67	Range attack
	Skill 0	2.0	Range attack
	Skill 1	3.08	A medium-range attack
	Skill 2	1.13	Recover HP
	Skill 3	2.0	A medium-range attack
	Skill 4	3.25	A medium-range attack
	Skill 5	2.08	A long-range attack
	Skill 6	2.42	A long-range attack
	Skill 7	2.5	A long-range attack

Table 5. Character 'Twinblast' attack and skills



	Skill Name	Delay(sec)	Description
	Attack	0.58	Range attack
	Skill 0	1.05	Range attack
	Skill 1	1.1	Range attack
	Skill 2	2.08	Recover HP
	Skill 3	4.25	A medium-range attack
	Skill 4	3.58	A medium-range attack
	Skill 5	4.83	A medium-range attack
	Skill 6	3.0	A long-range attack
	Skill 7	3.67	A long-range attack

Table 6. Character 'Wukong' attack and skills

	Skill Name	Delay(sec)	Description
	Attack	0.83	Melee attack
	Skill 0	3.83	Melee attack
	Skill 1	2.53	Melee attack
	Skill 2	1.13	Recover HP
	Skill 3	3.0	Melee attack
	Skill 4	2.83	A medium-range attack
	Skill 5	2.0	A long-range attack
	Skill 6	2.67	A long-range attack
	Skill 7	2.0	A medium-range attack

In addition, the character includes a Behavior tree that can replace a person's gameplay, assuming that several players play the game together. A behavior tree [20] is an artificial intelligence of a tree structure that branches and executes tasks

under various conditions. Behavior trees use selectors, sequences, and task nodes to operate to allow AI characters to perform appropriate tasks in a given situation. Unreal Engine provides visual-based visual tree production tools to implement NPCs such as collaborative characters and enemy monsters. RPG Gym implements a character behavior tree that can replace human gameplay using a production tool, thereby performing actions such as approach, attack, and avoidance to enemy monsters. Figure 9 is a behavior tree applied to each character.

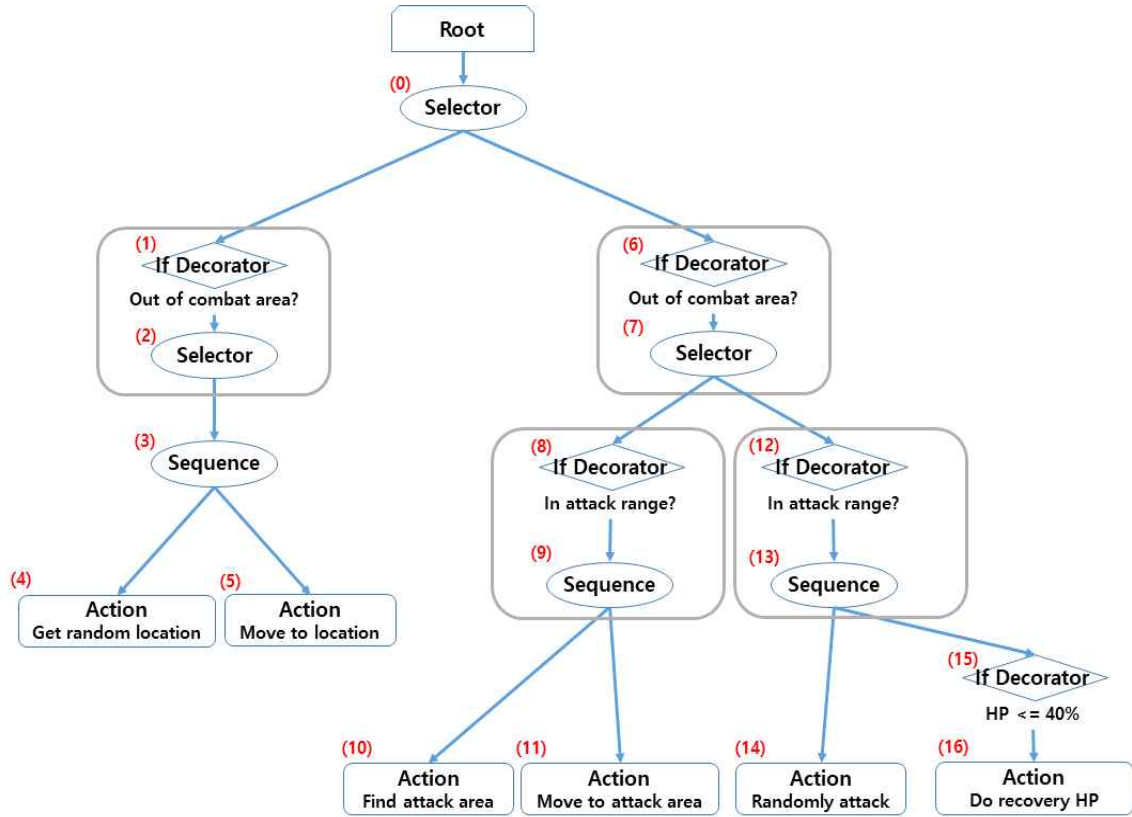


Figure 11. Behavior tree applied to each character. In the figure, the number refers to the execution order of the behavior tree.

3. 1. 3. Enemy Monsters

RPG Gym includes four types of enemy monsters, each with one basic attack and five skills. Enemy monsters use the Behavior tree versus characters. From table 7 to table 10 describe the details of the enemy monsters provided by the RPG Gym, and figure 10 is a behavior tree applied to the enemy monsters. The enemy monster's behavior tree has a tree structure similar to that of the player but makes the target attack the closest of several targets.

Table 7. Enemy monster 'Countess' attack and skills


	Skill Name	Delay (sec)	Description
	Attack	0.72	Melee attack
	Skill 0	2.83	A medium-range attack
	Skill 1	2.7	Melee attack
	Skill 2	3.12	A medium-range attack
	Skill 3	1.87	Melee attack
	Skill 4	3.31	A medium-range attack

Table 8. Enemy monster 'Rampage' attack and skills


	Skill Name	Delay (sec)	Description
	Attack	0.97	Melee attack
	Skill 0	2.1	A medium-range attack
	Skill 1	3.24	Melee attack
	Skill 2	2.57	Melee attack
	Skill 3	3.71	Melee attack
	Skill 4	4.0	A medium-range attack

Table 9. Enemy monster 'Sevarog' attack and skills



	Skill Name	Delay (sec)	Description
	Attack	0.8	Melee attack
	Skill 0	2.1	A medium-range attack
	Skill 1	2.82	A medium-range attack
	Skill 2	3.41	A medium-range attack
	Skill 3	2.43	A medium-range attack
	Skill 4	3.0	A medium-range attack

Table 10. Enemy monster ‘Terra’ attack and skills

	Skill Name	Delay (sec)	Description
	Attack	1.2	Melee attack
	Skill 0	2.1	Melee attack
	Skill 1	1.84	Melee attack
	Skill 2	3.3	Melee attack
	Skill 3	2.5	Melee attack
	Skill 4	4.2	A medium-range attack

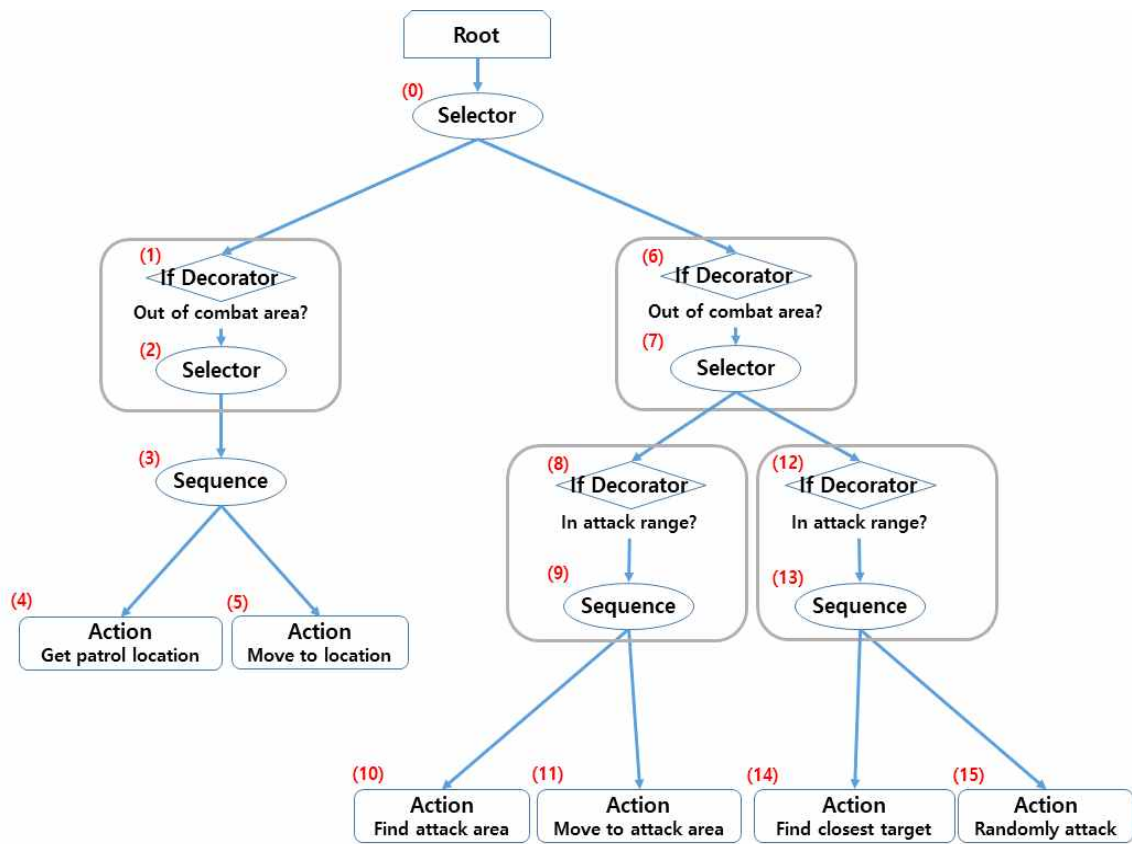


Figure 12. Behavior tree applied to each monster. In the figure, the number refers to the execution order of the behavior tree.

3. 1. 4. Maps and Game Rule

Game maps representing terrains and backgrounds are one of the important factors in the game. Content can be attempted by strategically utilizing terrains and object placement. RPG Gym provides four game maps and can be changed depending

on the settings. However, unlike the characters in a game map, the terrains and arrangement of objects in the map can not be customized. Figure 11 describes the map included in the RPG Gym.

RPG Gym reproduced raid content included in most ARPG games. Raid content is typically played in such a way that multiple players attack enemy monsters and win within the time limit. The proposed RPG Gym also follows the general way of proceeding with the raid content, but you can choose the number of characters participating, the type of characters, the skill of using the character, the map, the time limit, and the type of monster according to the researcher's intention.



Figure 13. Four maps included in RPG Gym

3. 2. Simulator Features

3. 2. 1. Remote Procedure Call Component

The remote procedure call (RPC) component is designed to interconnect game processes and deep learning model learning processes. When implementing deep learning models, they use a variety of programming languages and frameworks, but mostly Python. Thus, the RPG Gym provides an RPC component capable of

connecting with the Python process. The simulator corresponds to the server side of the RPC, and the Python process is a client. The Python process may manipulate or obtain information about a character in the game environment by calling a bound function in the game environment. Python API of RPG Gym is designed to make it easier for researchers to use and can access the game environment using the API. Figure 12 describes the RPC Component architecture, and table 11 lists Python APIs that have access to RPG Gym.

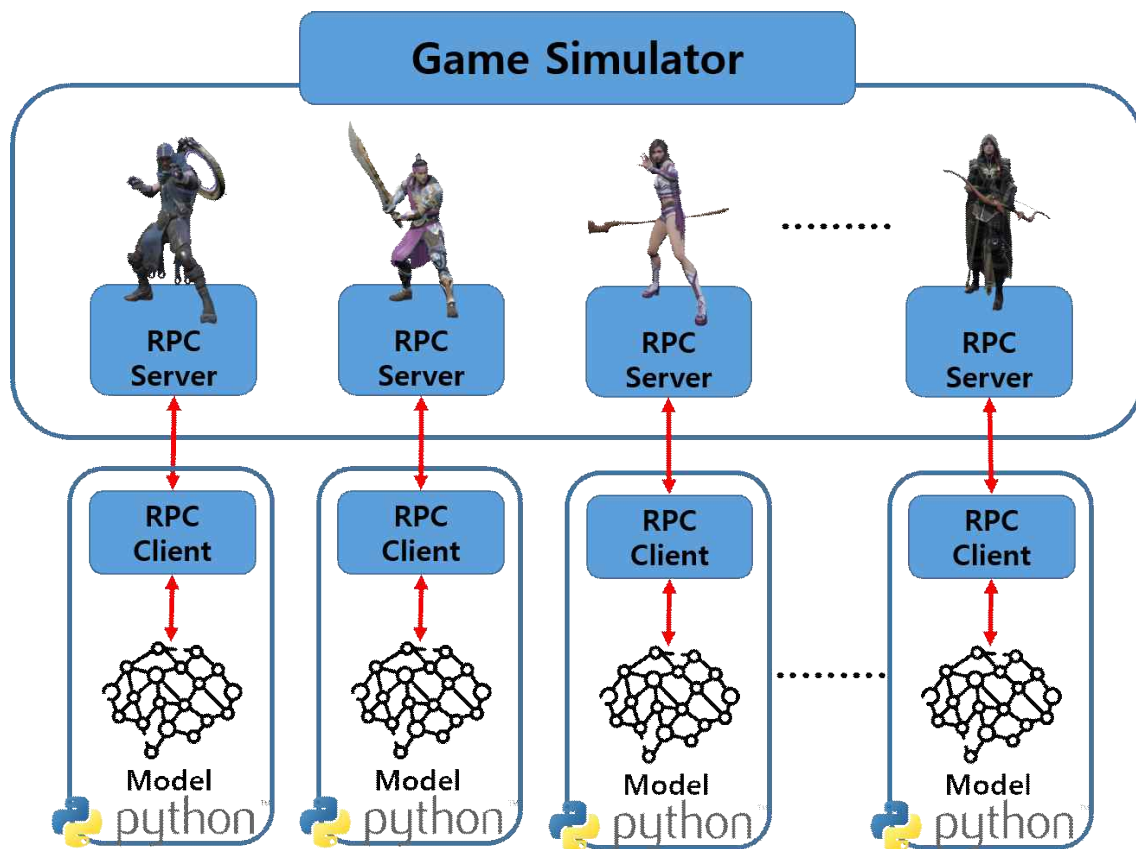


Figure 14. Structure of the RPC components

Table 11. List of Python APIs

API Name	Description
GetState	Obtain information about the character you manipulate. The information includes current location coordinates, HP, availability of skills, and distance from enemy monsters.
SetMoveAction	Move the character.
SetAbilityAction	Execute the character's basic attack or skills.
Restart	Restart the game.

3. 2. 2. Play Logging Component

Imitation learning and reinforcement learning are methods of controlling characters using deep learning models. Imitation learning is a method that training model after collecting expert human data and act the performance similar human level. Reinforcement learning is a model that can be learned without a dataset while interacting with the environment, but there are also reinforcement learning models that require data such as offline reinforcement learning. Therefore, it is necessary to have the function to collect data from players or experts, and the proposed game environment can collect human data that control characters by designing a player logging component. Data is collected since the start of the game, and by default, the log is recorded every 0.1 seconds, but the recording cycle can be adjusted by setting. Table 12 describes the data that can be collected. In addition, it records the achievement, elapsed time, and information of each character that occurred while performing content in the Raid content included in the RPG Gym.

Table 12. List of player log data

Name	Type	Description
CharcterLoc	3D vector	Current location of controlled character (x, y, z)
CharacterHP	float	Current HP of controlled character
Distance	float	Distance of controlled character to monster
EnemyLoc	3D vector	Current location of monster (x, y, z)
EnemyHP	float	Current HP of monster
MoveForward	float	The keyboard input value that moves the character
TrunRight	float	The mouse input value that moves the character
TurnYaw	float	Yaw value for the character to look at enemy monsters
CanAttack	bool	Verify that a default attack is enabled
CanSkillSlot0	bool	Verify that registered skills in skill slot 0 are available
CanSkillSlot1	bool	Verify that registered skills in skill slot 1 are available
CanSkillSlot2	bool	Verify that registered skills in skill slot 2 are available
CanSkillSlot3	bool	Verify that registered skills in skill slot 3 are available

Table 13. List of game result data

Name	Description
Success	Clear, Fail since exceed the time limit, Fail since die all characters
Elapsed Time	Time to proceed with the content regardless of success
Monster's remaining HP	When finishing a content, the monster's remaining HP
Character 's information	When finishing the content for each character participating, it represent the damage inflicted on the remaining HP and monsters.

3. 2. 3. Configuration Component

Configuration components provide flexibility when the proposed game environment learns a deep learning model. Each game released has different elements in raid content, and the characteristics of each character are different. Accordingly, a function capable of adjusting the features of each character, monster, and map by designing a configuration component is provided. The configuration component read a json file containing elements to be set at the time the game is initialized and reflects each element in the game element.

First, players that can be included in the raid content can be adjusted. Generally, raid content supports from 4 players to 30 players, but the proposed RPG Gym allows players to participate without restriction. However, there is only one character that can be controlled by a person, and other players must use a deep learning model or a Behavior tree. Second, you can adjust the map and time limit, which are the background of the content. The game map setting was designed in consideration of the generalization performance of the deep learning model, and one can be selected from four pre-made game maps. The time limit is an element designed to adjust the difficulty of game play and may be set in seconds. Third, the features of each character and enemy monster can be adjusted. For each character, HP can be adjusted to the desired amount, and four of the eight skills provided in advance are designed to be selected and used. In addition, it is possible to control the reuse wait time required to use the skill and the damage to be delivered to the monster. Monsters can be included in the content by selecting one of the four monsters and adjust HP and attack power.

3. 2. 4. Extra Features

- **Headless Client:** The RPG Gym environment includes several 3D modeling assets, character-driven animation, and VFX, which produces visual effects. When rendering multiple elements by running a game environment, a lot of computing resources are required. Furthermore, many computing resources are required even when learning deep learning models. Therefore, when learning a deep learning model, a function that can only execute game logic without rendering is required. The RPG Gym environment has a headless function, and when learning a deep learning model, a rendering process may be omitted.
- **Simulation Accelerating:** Simulator acceleration is a necessary function when many experiments are needed in a short period time. For example, if the task you are trying to solve takes five minutes, you have to spend five minutes entirely. However, RPG Gym is equipped with a simulator's acceleration function to help the simulation proceed quickly.

Unlike the existing deep learning environment, the proposed RPG Gym consists of an ARPG genre of games that can be simulated and implements some systems and contents used in general commercial games. OpenAI Gym and Mujoco are different from recent commercial games and are mainly used to prove the imitation learning or reinforcement learning algorithms. However, the RPG Gym is more universal than the existing learning environment because it can simulate the content of a typical commercial game.

IV. Experiment and Result

Adjusting game balance is one of the most important issues in the game industry. Balance control of the game can be approached from various aspects. For example, in the case of raid content, the difficulty of the game is adjusted based on the degree of success of the player. If the content difficulty of the game is easy and the success rate is very high, the player will lose interest, and if the difficulty is difficult, the player may not succeed in the content challenge and lose interest. In addition, in the case of RPG games, there are several types of characters, and if the performance difference between characters is severe, there is a possibility that the player will leave. There are unique skills that can be used for each character, and it can be judged that the game's balance is not appropriate due to the wide variation in skills.

NPCs, such as enemy monsters who collaborate with players in games or engage in battles, typically use rule-based artificial intelligence algorithms such as behavior trees. Rule-based artificial intelligence algorithms are difficult to generalize and have constant behavior patterns, so players can easily deal with behavior patterns when they reach a certain level. Therefore, attempts are being made to apply deep learning as a replacement for existing rule-based algorithms, and some studies show superior performance than humans.

To address the balance adjusting problem and intelligent NPC problem, which is importantly addressed in the game industry, we can simulate it in the RPG Gym proposed in this paper. The experiment proceeds with game difficulty adjusting and deep learning-based character behavior control in Raid content provided by RPG Gym.

4. 1. Game Balance Adjusting Test

In this experiment, we proceed with the experiment assuming the Raid content of the game against three players and one monster. After finish, the raid content, adjust the difficulty by using a heuristic method based on the success, elapsed time, and information of each character included in the log record. The initial difficulty set up using a heuristic. Which sets the HP, attack power, and skill of characters and enemy monsters, and sets the content's success goal at 50% to 60%. The difficulty is adjusted every 50 times, and the difficulty is adjusted by increasing or lowering the strength of the enemy monster according to the number of successes. In this experiment, the physical strength of the monster was set to 30,000, and table 14 shows parameters other than the physical strength setting of the monster.

Table 14. Initial experimental parameters

Parameter Values	Description	
Map	Island	
Limited Time	600 Seconds	
Accelerating Ratio	5.0: The speed of the game is five times faster.	
Play Count	50 Times: The standard for performance measurement is set to 50 times.	
Characters	Gideon	HP: 2,000 Attack power: 60.0 Use skills: Skill 0, Skill 2, Skill 5, Skill 7
	Kwang	HP: 2,000 Attack power: 60.0 Use skills: Skill 1, Skill 2, Skill 4, Skill 7
	Wukong	HP: 20,000 Attack power: 60.0 Use skills: Skill 0, Skill 2, Skill 5, Skill 7
Monster	Sevarog	HP: 30,000 Attack Power: 100.0 Use Skills: Skill 0, Skill 1, Skill 2, Skill 3, Skill 4

In the difficulty adjusting experiment, the first task failed 43 times, and the remaining HP average value of the monster in the case of failure is 9,554.37. In the after task, half of the remaining average HP of the monster subtracts from the monster HP of the previous task. As shown in table 15, as the HP of the monster decreases, the number of failures decrease and the elapsed time of content tends to decrease. In task 3 of table 15, when the HP of the monster was 22,250, the target success rate (60%) was recorded. Task 4 was an experiment conducted in the same

way, and the success rate was recorded at 66%. As a result of this experiment, we show that the difficulty of the game can be adjusted by setting the HP of the enemy monster to 22,250 to achieve the targeted success rate.

Table 15. Balance adjusting experiment results

Tasks	Monster's HP	Success Counts	Monster's remain HP (Failure case)	Elapsed Time (sec)
1	30,000	7	9554.37	172.23
2	25,250	20	6096.6	163.1
3	22,250	30	3548.73	150.57
4	20,550	33	3578.47	150.76

In table 15, the elapsed time specified in each task shows that the content terminates faster than the 600 seconds specified by the initial setting. This means that the overall difficulty of the content has been lowered. The time limit for Raid content provided in each commercial game is slightly different, but it is generally based on 10 minutes. Therefore, based on the initial balance adjusting experiment, we conduct an experiment to measure the success rate and elapsed time of content by increasing each character's and monster's abilities. The HP and attack power of characters and monsters were tripled to experiment, and the average value was obtained every 50 times to adjust the value. The target elapsed time in this experiment is 420 to 600 seconds, and the success rate is 50% to 60% as in the previous experiment.

In task 1 conducted with initial parameters, the number of successes was measured as 50 and the average elapsed time was measured as 216.04 seconds. Based on the results obtained from the previous experiment, a value three times higher was applied, but when all values were three times higher, the elapsed time increased by 66 seconds and the success rate was 100%. Therefore, the monster HP, monster attack power, character HP, and character attack power were adjusted using heuristic methods based on the initial parameters to control the success rate and elapsed time. As a result of the experiment, when the success rate and elapsed time were measured after adjusting the parameters as shown in task 7 in table 16, a 60% success rate and an average time of 433.52 seconds were measured.

Table 16. Balance adjusting experiment results considering time

Tasks	Monster's HP	Monster's Attack Power	Character's HP	Character's Attack Power	Success Counts	Elapsed Time (sec)
1	$22,250 \times 3$	100×2	$2,000 \times 3$	$\times 2$	50	216.04
2	$22,250 \times 3$	100×2	$2,000 \times 3$	$\times 1$	47	461.1
3	$22,250 \times 3$	100×3	$2,000 \times 3$	$\times 1$	0	187.98
4	$22,250 \times 3$	100×2.5	$2,000 \times 3$	$\times 1.5$	24	333.56
5	$22,250 \times 3.5$	100×2.25	$2,000 \times 3$	$\times 1.5$	43	381.1
6	$22,250 \times 4$	100×2.25	$2,000 \times 3$	$\times 1.5$	38	444.96
7	$22,250 \times 4$	100×2.35	$2,000 \times 3$	$\times 1.5$	30	433.52

4. 2. Imitation Learning for Character

Imitation learning [21] is a method of learning agents using human data. In the case of a game, data generated when a person controls a character can be used to implement a character's behavior control. In this experiment, 34,200 human data were collected from 30 times game plays with characters applied to the behavior tree. 80% of the collected data was used to learn the imaging learning model, and the remaining 20% was used as a test set. When applied to RPG Gym after completing model learning, the performance was measured compared to the behavior tree. The setting values of each character and monster applied in this experiment were the results of task 3 in table 15.

As a result of the experiment, the loss value graph shown in figure 13 tended to decrease as the epoch passed. However, when the learned model was tested on the RPG Gym, the character's control was unstable, and the performance was low compared to the behavior tree character.

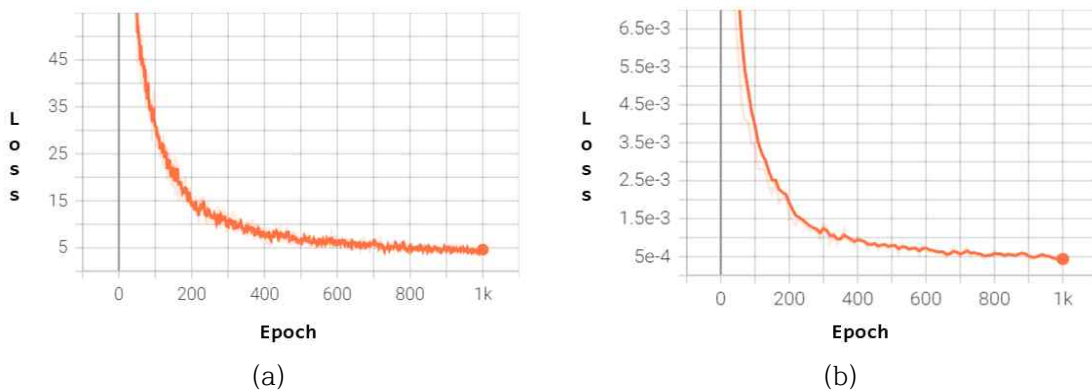


Figure 15. Imitation learning loss graph (a) training loss, (b) test loss

Table 17 shows the results of characters applied with Imitation Learning and two characters applied with Behavior Tree 10 times. Character 1 is an applied imitation model. As a result, the imitation learning character inflicted 35% lower damage compared to the behavior tree character. In addition, it was found that the count of successes decreased because imitation learning character has low performance.

Table 17. Result of imitation learning test

	With imitation learning character	All behavior characters
Success Counts	4	6
Average Clear Time	244 seconds	150.28 seconds
Character 1 (Kwang) Average Damage	4,925.71	7,558.56
Character 2 (Wukong) Average Damage	9875.99	8,258.54
Character 3 (Gideon) Average Damage	8134.28	6777.13

4. 3. Reinforcement Learning for Character

Imitation learning requires human data to learn agent's behavior. However, reinforcement learning [22] uses data generated by interacting with the environment in that the agent will learn without data. In addition, in the case of imitation learning, various and many data are required for generalization, but reinforcement learning has a relatively higher generalization performance than imitation learning because the agent learns while interacting with the environment without data set.

In this experiment, the agent was used using one of the reinforcement learning algorithms, proximal policy optimization (PPO) [23]. The experiment was conducted with a character using the PPO algorithm and two characters using the behavior tree. The setting values of each character and monster applied in the experiment are the same as in the imitation learning experiment. The reward function used in model learning is as follows. The learning model is rewarded with +40 or -40 depending on its success. In addition, the reward is adjusted according to the remaining HP of the character, the distance between the monsters, and the Yaw value, the angle at which the monster is viewed.

$$R = CLEAR \text{ or } FAIL + Character \text{ Remain } HP/100 - (Distance * 100) - Yaw$$

As a result of the experiment, the reward graph of the PPO model in figure 14, the character receives unstable rewards every episode. In addition, when the PPO model was applied to the character of the RPG Gym after learning, the performance was measured, which was lower than the behavior tree and the impedance learning model.

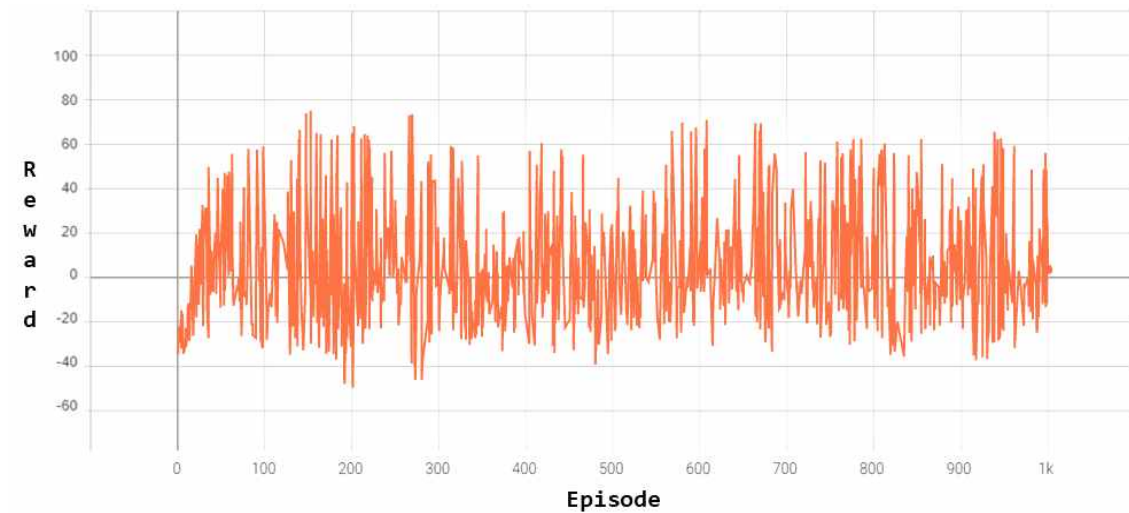


Figure 16. Reinforcement learning reward graph

Table 18. Result of reinforcement learning test

	With imitation learning character	All behavior characters
Success Counts	3	6
Average Clear Time	271.41 seconds	150.28 seconds
Character 1 (Kwang) Average Damage	3548.12	7,558.56
Character 2 (Wukong) Average Damage	10,378.62	8,258.54
Character 3 (Gideon) Average Damage	8,674.25	6777.13

V. DISCUSSION

RPG Gym, a new RPG game learning environment proposed in this paper, reconstructed the functions of existing commercial games and Raid content to be applied to the field of game artificial intelligence. It provides several games' content, characters, monsters, and game maps, and recently applied a non-targeting system that is mainly used in games. However, it has not fully reproduced the various systems and contents implemented by commercial games. For example, commercial games basically apply a multi-player system to allow multiple players to play together, and in addition to raid content, they provide special scenario plays through quest systems or form an economic system through in-game goods. In addition, artificial intelligence applied to non-player characters such as enemy monsters is designed to respond to sophisticated and diverse variables and has various types of artificial intelligence. Comparing these commercial games with the proposed RPG Gym, although game-side elements are lacking, various studies such as balance testing, agent learning, and generalization testing can be performed in the field of studying game artificial intelligence.

Several experiments were conducted to show that various game artificial intelligence studies could be conducted in the proposed RPG Gym. First, we conducted a balance control experiment on game play. However, the behavior tree used by the character is a lower performance than that of humans, so the setting value of the monster obtained through the experiment is not accurate. However, if NPC, whose performance has been verified by improving the performance of the included behavior tree to a level similar to that of humans, is used, the balancing control experiment can be sufficiently conducted. In addition, the implemented RPG Gym does not have a function to automatically adjust the balancing yet, but it will be possible to try various simulations by adding the auto-balancing adjust function

later. Second, when imitation learning and reinforcement learning methods that can control the character were applied, both methods showed lower performance than the behavior tree. This may be difficult to expect high performance because the deep learning model structure used in the experiment was unsuitable to solve the problem or provided only simple state information. Therefore, research is needed on what data is needed and what neural network structure is needed to solve the problem of controlling characters using deep learning methods.

VI. CONCLUSION

This paper proposes a new ARPG game simulator, RPG Gym, and explains the characteristics of the simulator and the experiments conducted in the simulator. The proposed RPG Gym includes a non-targeting system and raid content, which are combat systems used in recent commercial games. RPG Gym includes various types of characters, enemy monsters, and maps. In addition, to provide the function of the simulator, a Python API was created to easily access the simulator to obtain the necessary information or to manipulate components in the game. It also provides logging components, configuration components, headless and simulation accelerating functions implemented for convenience. In order to show that can simulate and solve problems dealt with in the game industry, this paper conducted game balance adjusting and deep learning based character behavior control.

In the case of the balance adjusting experiment, the experiment was conducted by adjusting the balance and elapsed time of the game using the behavior tree artificial intelligence character. As a result, found the success rate and the difficulty suitable for elapsed time. This balance simulation is expected to be able to find the right balance by directly considering and adjusting several variables, along with adjusting the difficulty by analyzing a person's play record in the real game industry. For character behavior control learning using deep learning, the experimental results did not prove good performance but showed that the deep learning model can use in the proposed simulator.

RPG Gym lacks functionality and game content compared to real games. However, I believe that if we continue to add content and features for researchers, it will be a good environment to solve various problems addressed in the game industry.

REFERENCES

- [1] S. Snodgrass and S. Ontanon, A hierarchical MdMC approach to 2D video game map generation, in Proc. Artif. Intell. Interactive Digit. Entertain. Conf., 2015, pp. 205-211.
- [2] Alexander Jaffe. Understanding Game Balance with Quantitative Methods. PhD Thesis. University of Washington, Seattle, WA, 2013.
- [3] H. Kim, S. Hong, and J. Kim, Detection of auto programs for MMORPGs, in Proc. Australas. Joint Conf. Artif. Intell., Cham, Switzerland: Springer, 2005, pp. 1281-1284.
- [4] A. Khalifa, A. Isaksen, J. Togelius, and A. Nealen, Modifying MCTS for human-like general video game playing, in Proc. 25th Int. Joint Conf. Artif. Intell., 2016, pp. 2514-2520.
- [5] <https://eldenring.bn-ent.net/>
- [6] <https://lostark.game.onstove.com/>
- [7] <https://diablo2.blizzard.com>
- [8] Zarrad, Anis. (2018). Game Engine Solutions. 10.5772/intechopen.71429.
- [9] <https://www.unrealengine.com/>
- [10] <https://unity.com/>
- [11] <https://www.cocos.com/>
- [12] S. Shah, D. Dey, C. Lovett, and A. Kapoor, AirSim: High-fidelity visual and physical simulation for autonomous vehicles, in Field and Service Robotics, M. Hutter and R. Siegwart, Eds. Cham, Switzerland: Springer, 2018, pp. 621-635.
- [13] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, CARLA: An open urban driving simulator, in Proc. 1st Annu. Conf. Robot Learn., 2017, pp. 1-16.
- [14] G. Brockman et al., OpenAI Gym, 2016, arXiv:1606.01540
- [15] E. Todorov, T. Erez and Y. Tassa, MuJoCo: A physics engine for model-based control, 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems,

2012, pp. 5026-5033, doi: 10.1109/IROS.2012.6386109.

[16] Suarez, J., Du, Y., Isola, P., Mordatch, I.: Neural MMO: A massively multiagent game environment for training and evaluating intelligent agents. arXiv preprint arXiv:1903.00784 (2019)

[17] O. Vinyals et al., Starcraft II: A new challenge for reinforcement learning, 2017, arXiv:1708.04782.

[18] C. Berner et al., Dota 2 with large scale deep reinforcement learning, 2019, arXiv:1912.06680.

[19] I. Oh, S. Rho, S. Moon, S. Son, H. Lee, and J. Chung, Creating pro-level AI for a real-time fighting game using deep reinforcement learning, IEEE Trans. Games, early access, Jan. 6, 2021, doi: 10.1109/TG.2021.3049539.

[20] Nicolau, M., Perez-Liebana, D., O'Neill, M., Brabazon, A.: Evolutionary behavior tree approaches for navigating platform games. IEEE Transactions on Computational Intelligence and AI in Games PP(99), 1-1 (2016)

[21] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, Imitation learning: A survey of learning methods, ACM Comput. Surv., vol. 50, no. 2, p. 21, 2017.

[22] Y. Li, Deep reinforcement learning: An overview, arXiv preprint arXiv:1701.07274, pp. 1-85, Nov. 2018.

[23] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, Proximal policy optimization algorithms, 2017, arXiv:1707.06347.