Journal of Korea Game Society 2020 Feb; 20(1): 57-66
http://dx.doi.org/10.7583/JKGS.2020.20.1.57

게임 프로그래밍

사용자와의 협력 플레이를 위한 강화학습 인공지능 프로세스 구축

정원조
굿게임 스튜디오
joedano13@gmail.com

Build reinforcement learning AI process for cooperative play with users

Won-Joe Jung
Dept. of R&D, GoodgameStudio Corp

요 약

연구는 MOBA 게임에서 선호도가 낮은 Supporter를 대체하는 인공지능을 강화학습을 이용한 구현을 목표하였다. ML_Agent를 이용해 게임의 규칙, 환경, 관측 정보, 보상 처벌을 구성하였다. DPS 에이전트로 구성된 그룹과, Support 에이전트가 있는 그룹으로 나누어 강화학습을 진행하였다. 결과 데이터인 누적 보상 값, 사망 횟수 바탕으로 결론을 도출하였다. 협력 플레이 그룹이 비교 그룹보다 평균 누적 보상 값이 3.3 더 높게 측정되었으며 사망 횟수 총합 평균은 3.15 낮게 되었다. 이를 바탕으로 죽음을 최소화하고 보상을 최대화하는 협력 플레이를 수행하는 강화학습을 확인할 수 있었다.

ABSTRACT

The goal is to implement AI using reinforcement learning, which replaces the less favored Supporter in MOBA games. ML_Agent implements game rules, environment, observation information, rewards, and punishment. The experiment was divided into P and C group. Experiments were conducted to compare the cumulative compensation values and the number of deaths to draw conclusions. In group C, the mean cumulative compensation value was 3.3 higher than that in group P, and the total mean number of deaths was 3.15 lower. performed cooperative play to minimize death and maximize rewards was confirmed.

Keywords : AI(인공지능), Reinforcement learning(강화학습), Machine learning(기계학습), Unity3D(유니티3D)

Received: Nov. 11. 2019 Revised: Dec. 04. 2019
Accepted: Dec. 17. 2019
Corresponding Author: Won-Joe Jung(Goodgame Studio)
E-mail: joedano13@gmail.com

ISSN: 1598-4540 / eISSN: 2287-8211

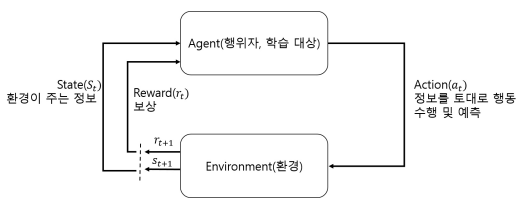
1. 서 론

2019년 현재 머신 러닝 기술은 4차 산업시대의 핵심으로 국방, 의료, 교육, 보안, 생활, 게임과 같이 여러 분야에 접목하여 국가와 기업들이 연구하고 있다[1]. 머신 러닝을 게임에 접목 시킨 대표적인 사례로는 구글 소속 DeepMind사의 알파고와 알파스타 비영리 범용 인공지능 연구 단체인 OpenAI의 DOTA 2가 존재한다. 기존 연구 사례들은 인공지능이 사람과 경쟁 하여 기존의 명시적인 알고리즘보다 비교 우위의 게임 인공지능을 구현한 것이다. 본 연구는 기존의 사람과 인공지능간의 경쟁이 아닌 협력을 목표로 행동하는 인공지능을 머신러닝 기술을 적용하는 것을 목표 하였다.

인공지능 협력 학습 연구를 위해 선정된 게임의 장르는 MOBA(Multiplayer Online Battle Arena)이다. 라이엇 사의 '리그오브레전드'에서는 다른 유저들을 보좌하는 서포터 포지션의 선호도가 가장 낮았다[2]. 이로 인하여 팀 구성 매칭에 불균형이 발생하고 사용자의 콘텐츠 몰입에 방해 요소로 인지하였다. 이에 대안으로 플레이어 포지션 선호도가 낮은 플레이어 서포터를 대신할 인공지능 학습이 가능한가를 실험환경으로 구축하였다. 연구는 협력 역할수행 게임에서 사용자와 협력 플레이를 통하여 강화학습을 진행하는 AI 구성 프로세스 구축하고자 한다.

2. 사전연구

2.1 강화학습(Reinforcement Learning)



[Fig. 1] Mechanism of Reinforcement Learning

강화 학습(Reinforcement Learning)은 머신 러닝의 한 영역으로 에이전트(Agent)가 환경과의 상호작용을 통해 보상(Reward)을 최대화하는 행동(Action)을 선택하여 반복 학습하는 방법이다[3]. [Fig.1]은 강화학습의 작동 방식을 표현한 이미지로 강화 학습에서 에이전트는 행위자(Actor)의 역할을 갖는다[4]. 에이전트들은 자신들의 상태와 환경에서 발생하는 정보들을 관측하여 관측된 정보를 토대로 보상을 받을 수 있는 행동을 수행하는 주체(主體)다. 에이전트는 상태(State), 환경(Environment)등을 인식하여 행동을 수행하고 행동의 영향으로 환경의 상태는 변화한다. 에이전트가 올바른 행동을 할 경우 에이전트에게는 보상이 제공된다. 에이전트는 반복 학습을 통해 수많은 시행착오를 거치며 처벌과 보상을 받고 보상을 최대화하는 방향으로 행동을 수행하게 된다[5]. 에이전트가 행동을 수행한 이후의 환경의 상태와 보상은 일정치 않으며 특정 상태에서 수행할 행동을 선택하는 규칙을 정책(Policy)이라고 한다[6].

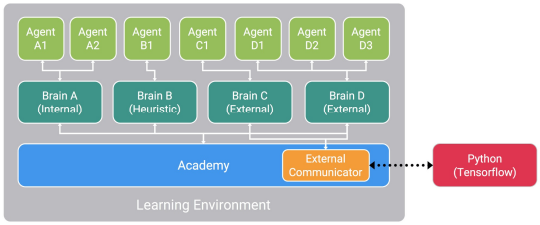
2.2 강화학습 디지털 게임 적용 사례

강화학습을 디지털 게임에 적용한 사례로는 OpenAI의 도타2와 딥마인드의 알파스타가 있다. OpenAI는 도타2에 강화학습을 적용하여 기존의 명시적인 알고리즘으로 구현된 인공지능들보다 월등한 실력을 보여주며 사람과의 대전에서 7257전 7215승 42패(승률 99.4%)의 승률을 보여주었다. 딥 마인드의 알파스타는 스타크래프2에 강화학습을 적용하는 프로젝트로 기존 플레이어들의 정보들을 학습하여 모방하고 약 1억 2000만 번의 자체 대결을 통해 스타크래프트2를 학습하였다. 사람의 기준으로 191년 동안 스타크래프트2를 플레이한 시간이다. 알파스타는 상위 0.2%인 그랜드 마스터 등급을 달성하였으며 일반 유저들과의 대결에서의 승률은 99.8% 1대 3의 불리한 조건속의 승부에서도 99.76~99.93%라는 높은 승률을 기록하였다. 하지만 이와 같은 연구 사례들은 사람과의 경쟁위주의 연구로 강화학습을 게임에서의 협력, 보좌를

위한 연구는 아직 부족한 실상이다.

2.3 Unity3D ML_Agnet

ML_Agent는 Unity 3D엔진 환경에서 강화 학습 연구를 할 수 있는 머신러닝 플랫폼이다[7].



[Fig. 2] ML_Agent Structure

[Fig.2]는 ML_Agent의 구조를 도식화한 것이다. ML_Agent에서 각 브레인은 환경으로부터 발생하는 상태 정보와 행동 공간을 정의하고 연결되어 있는 에이전트의 행동을 결정하는 정책 결정권자다. ML_Agent에서는 하나의 단일 브레인에 복수의 에이전트를 연결하여 학습 시킬 수 있으며 에이전트의 관측 정보를 어떻게 제공 하느냐에 따라 에이전트들의 정보교환, 공유가 결정되는데 기본 설정 값으로 에이전트들은 서로 독립되어 정보를 공유하지 않는다. 브레인의 종류는 External, Internal, Player, Heuristic 4가지가 존재한다. External은 관측된 정보, 결과를 아카데미의 External Communicator를 통해 외부 머신 러닝 라이브러리인 텐서플로우와 통신하여 학습과 정책을 결정한다. Internal은 External로 학습이 이루어진 모델 을 Unity 3D에서 직접 시연하거나 다른 플랫폼으로 빌드 하기 위해 사용하는 브레인이다. Player 브레인은 모방(Imitation)학습을 위해 사용하거나 에이전트의 작동에 문제가 없는지 확인하기 위한 브레인으로 에이전트의 직접 플레이 테스트용으로 사용한다. Heuristic은 명시적 알고리즘으로 작성된 코드들을 기반으로 행동을 결정하는 브레인이다.

2.4 PPO(Proximal Policy Optimization)

PPO(Proximal Policy Optiomization)는 2015년 12월에 설립된 비영리 AGI(Artificial General Intelligence)의 연구단체인 OpenAI에서 만든 알고리즘으로 기존의 강화학습 알고리즘과 비교하여 높은 성능과 개선된 사용 편의성을 제공하여 OpenAI와 ML_Agent의 기본 강화학습 알고리즘으로 채택되었다. PPO는 머신러닝에서 학습을 시킬 때 사람이 직접 설정하는 Hyperparameter 튜닝이 상대적으로 적으면서 우수한 결과를 얻을 수 있다. PPO는 구현의 편의성, 샘플의 복잡성 및 튜닝 용이성 사이의 균형을 유지하고 각 단계에서 업데이트를 계산하여 비용을 최소화하여 이전 정책과의 편차를 비교적 적도록 하는 강화학습 알고리즘이다.

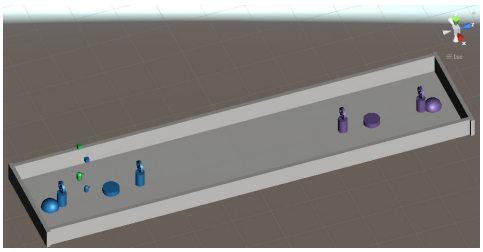
3. 실험 설계

본 연구는 협력 역할수행 게임에서 사용자와 협력 플레이를 통하여 강화학습을 진행하는 AI 구성 프로세스 구축을 목표로 한다. 이를 위해 첫째 사용자 행위를 수행하는 인공지능 DPS(Damage Per Second) 에이전트를 구성하였다. 둘째 목표를 수행할 수 있도록 협력하는 Support 에이전트의 강화학습 환경을 설계하였다. 셋째 실험을 위한 MOBA 게임 구성, 에이전트의 정보 전달을 위한 관측 환경 구성, 에이전트의 보상 처벌 프로세스를 구축하였다. 이를 바탕으로 협력 AI 강화학습 프로세스 실험을 진행하였다.

강화 학습 실험은 콘텐츠 제작 소프트웨어인 유니티 엔진(Unity Engine 2019.2.6)의 환경을 기반으로 구현하였으며 유니티 엔진의 머신 러닝 플랫폼인 ML_Agent 0.10.0 버전을 이용해 강화 학습을 진행하였다.

3.1 게임 규칙 및 진행 방식

실험을 진행한 게임의 규칙은 MOBA 게임들과 동일한 규칙을 가지고 있다. [Fig.3]은 실험을 진행하기 위해 구현한 환경의 이미지다. 건물 오브젝트 포탑, 억제기, 포탑, 넥서스 순으로 배치되어있다. 에이전트가 상대방의 모든 건물을 파괴하면 승리하고 자신의 모든 건물이 파괴당하면 패배한다.



[Fig. 3] Environment Image

3.2 학습 Agent 설계

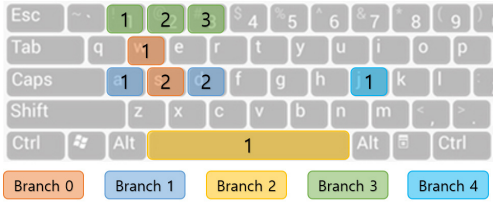
[Fig. 4] DPS Agent (left), Support Agent (right)

[Fig.4]는 에이전트 오브젝트 이미지로 공격만 가능한 DPS 에이전트와 공격, 치료를 할 수 있는 Support 에이전트가 존재한다.

[Table 1] Branch of Agents

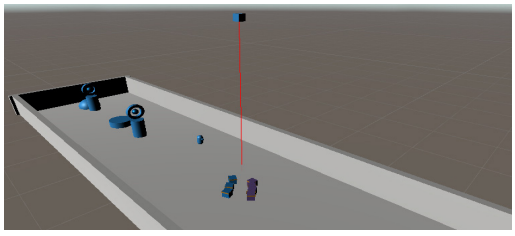
Branch	function	DPS Inpute	Support Input
Branch 0	Vertical movement	3	3
Branch 1	Horizontal movement	3	3
Branch 2	Specify Goal	2	2
Branch 3	Stats rise	2	4
Branch 4	Use healing skills	1	2

DPS 에이전트는 공격 능력치만 상승 시킬 수가 있으며 Support 에이전트는 공격, 수비, 치료 3가지 능력중 하나를 에이전트 스스로 선택해 상승시킬 수 있으며 에이전트는 행동을 수행하기 위한 5가지의 Branch[1]를 가진다. [Table 1]은 에이전트들의 Branch이다. 0, 1단계 Branch는 가상 커서의 상하 좌우 이동에 사용하며 2단계 Branch는 에이전트의 이동 목표 지점을 위해 사용한다. 3단계 Branch는 에이전트가 레벨 업을 할 경우 에이전트의 능력치를 상승 시킬 때 사용한다. 4단계 Branch는 스킬을 사용할 때 쓰는 것으로 Support 에이전트만 입력을 발생 시킬 수 있다.



[Fig. 5] Branch Input of Agent

[Fig.5]는 에이전트의 입력을 키보드로 표현했을 때의 이미지로 0은 아무것도 입력하지 않음을 뜻한다. 에이전트는 0, 1, 2단계 Branch의 Input을 통해 가상의 커서를 조작하여 에이전트의 이동 지점을 정할 수 있다.



[Fig. 6] Agent's Move Processing

[Fig.6]은 에이전트의 이동을 보여주는 이미지로 정육면체의 가상 커서를 기준으로 Y축 아래방향으로 Ray[2]를 발사하면 Ray가 접촉한 지점으로 에

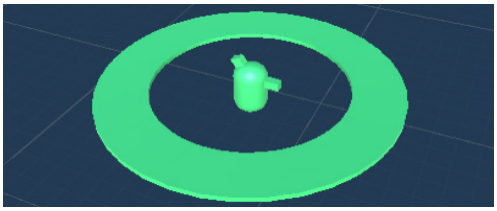
1) Branch는 행동을 수행하기 위한 에이전트의 명령 신호로 임의로 지정된 범위의 값을 매개변수로 행동을 수행한다.

이전트가 이동 한다. 에이전트가 공격을 하는 방법은 에이전트를 중심으로 6.5m 범위에서 가장 가까운 적을 자동으로 1초에 1회씩 발사체를 발사해 공격한다. 공격을 통해 미니언을 에이전트가 파괴시키거나 아니면 에이전트를 중심으로 15 범위에서 미니언이 파괴될 경우 경험치를 획득한다. 일정 경험치를 획득하게 되면 레벨 업을 하게 되고 레벨은 최대 6까지 올릴 수 있다. 레벨 업을 할 때마다 3번 Branch를 통해 능력치를 상승 시킬 수 있다. 각 에이전트는 1레벨 때 300의 체력과 30의 공격력을 가지고 시작하며 Support에이전트의 치료스킬은 1레벨에 20의 체력을 회복시켜준다. 상승시킬 수 있는 능력치는 DPS 스탯, Tank 스탯, Support 스탯 총 3가지가 존재한다.

[Table 2] Detailed Stats

Item	DPS	Tanking	Support
offense power	+30	+5	+3
health	+20	+120	+70
skill	0	0	+40

[Table 2]는 각 스탯의 능력치 상승을 산출한 표로 DPS 스탯은 공격력을 제일 많이 올려주지만 체력 상승이 낮고 Tank 스탯은 공격력 상승이 낮지만 체력이 매우 높게 상승된다. Support 스탯은 공격력과 체력 상승폭이 낮지만 치료 스킬의 치료양이 증가하게 된다. [Fig.7]은 Support 에이전트가 4번 Branch의 매개변수 발생시켜 치료 스킬을 사용하는 이미지다.

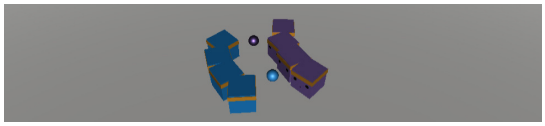


[Fig. 7] Scope of Care Skills for Support Agents

Support 에이전트 중심으로 5.5m 범위에 DPS 에이전트가 있을 경우 치료 스킬을 Support 에이전트가 사용할 수 있다. 치료 스킬을 사용할 경우 범위표시가 나타나며 DPS, Support 에이전트의 체력을 회복한다. 에이전트의 체력이 0이하가 되어 죽을 경우 에이전트레벨×5초 후에 넥서스 건물 앞에서 다시 부활하게 된다.

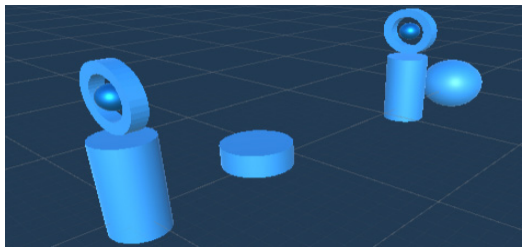
3.3 정보 전달 환경 설계

환경을 구성하기 위한 오브젝트는 미니언, 슈퍼 미니언, 포탑, 억제기, 넥서스가 있다. [Fig.8]의 미니언들은 35초마다 블루, 퍼플 미니언 몬스터 오브젝트가 6개씩 호출된다. 미니언은 상대방의 건물을 공격하기 위해 움직이고 미니언 본인을 중심으로 5m 범위에서 가장 가까운 적을 1초에 1회씩 공격한다.



[Fig. 8] Minion Object Finite State Processing

상대방의 억제기를 파괴하면 아군 슈퍼 미니언이 소환된다. 슈퍼 미니언은 공격력과 체력이 높아 상대방의 건물을 공격하기가 더 수월해진다.



[Fig. 9] Turret, Suppressor, Turret, Nexus Object

[Fig.9]의 포탑은 포탑 오브젝트를 중심으로 11m 범위에서 가장 가까운 적을 공격하며 1.3초마

2) 가상의 광선을 발사해 지정된 방향과 거리 이내에 부딪히는 오브젝트가 있는지를 참, 거짓 값으로 출력한다.

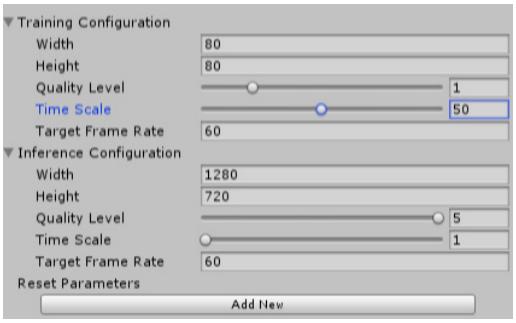
다 70의 데미지를 주는 발사체를 발사해 공격을 한다. 상대방의 넥서스를 파괴하면 파괴한쪽이 게임을 승리하게 되고 환경이 초기화된다.

[Table 3] Detailed Stats of Objects

Item	Minion	Super minion	Turret	Suppressor	Nexus
offense power	12	75	70	0	0
Range	5	5	11	0	0
health	210	1000	650	650	650

[Table 3]은 각 오브젝트의 세부 능력치를 산출한 표다.

3.4 Academy 설정



[Fig. 10] Academy settings

[Fig. 10]은 본 실험을 진행하면서 사용한 아카데미의 설정 값이다. 아카데미는 환경에 존재하는 모든 Brain과 에이전트의 데이터를 취합해 외부 커뮤니케이터로 텐서플로우 API와 통신하여 학습을 진행하는 관리자 역할의 오브젝트다. Quality Level은 1로 맞췄으며 Time Scale은 50 Target Frame Rate는 60으로 맞추어 강화학습을 진행하였다.

3.5 관측 정보 처리

DPS 에이전트는 총 124개의 관측정보를 가지고 있으며 Support 에이전트는 총 105개의 관측정보를 가지고 학습을 진행하였다. DPS 에이전트는 20프레임 주기로 관측 후 행동하였으며 Support 에이전트는 10프레임 주기로 관측 후 행동하였다. [Table 4]는 에이전트의 관측 정보 표다.

[Table 4] Observation list

Item	DPS	Support
level	1	1
Experience	1	1
Stat Point	1	3
offense power	0	1
Stern (Can not move)	1	0
health	1	2
Stats	1	3
It was killed?	1	1
Deaths	1	1
Allied Agent Death Check	0	1
Available skill check	1	1
Distance check with allies	0	1
Skill range check	0	1
Treatment amount check	0	1
Turret Cross Range Check	1	1
Target check	1	1
Target health	1	1
Target coordinate value	2	2
Virtual cursor coordinate values	2	2
Character coordinate values	2	4
The number of energies in the range	1	1
Number of allies in range	1	1
Allies survive	4	4
Survival of enemy objects	0	4
Allied object location value	4	0
Enemy Object Location Value	4	8
Raycast for tag identification	88	56
Allied Minion Coordinate Values	2	0
Allied minions street	1	1
Enemy minion street	1	1
total	124	105

3.6 보상 처벌 설계

DPS, Support 에이전트는 승리하면 1의 보상을 받고 패배하면 -1의 처벌을 받으며 에이전트는 지속적으로 -0.000001만큼의 처벌을 주어 에이전트가 아무것도 안하고 가만히 있는 걸 방지하였다. 게임을 승리하면 각 에이전트들은 죽은 횟수×0.05만큼의 처벌을 받게 되고 Support 에이전트는 치료 스킬 사용횟수×0.01만큼의 추가 보상을 받는다. 에이전트가 공격할 수 있는 오브젝트를 발견하여 공격하면 0.01의 보상을 지속적으로 받게 되며 DPS 에이전트는 아군 미니언과의 거리를 7m이하로 유지하면 0.01의 보상을 받게 된다.

3.7 PPO 파라미터 설정

[Fig.11]은 학습을 진행하면서 사용한 PPO 알고리즘의 매개변수 값이다.

```
batch_size: 256
beta: 0.002
buffer_size: 4096
epsilon: 0.2
hidden_units: 128
lambd: 0.95
learning_rate: 1e-3
learning_rate_schedule: linear
max_steps: 100000
memory_size: 512
normalize: false
num_epoch: 8
num_layers: 4
time_horizon: 64
sequence_length: 64
summary_freq: 1000
use_recurrent: true
vis_encode_type: simple
reward_signals:
    extrinsic:
        strength: 1.0
        gamma: 0.99
    curiosity:
        strength: 0.02
        gamma: 0.99
        encoding_size: 256
```

[Fig. 11] PPO Hyperparameter

batch_size, buffer_size, beta, max_steps, memory_size, num_layers, num_epoch의 설정 값들을 변경하였다. batch_size는 한 번의 경사하강(Gradient Descent) 업데이트 할 때마다 사용할 경험(Step)들의 수를 의미한다. buffer_size는 학습을 시작하기 전에 얼마나 많은 경험(정보)을 저장할지 결정한다. beta는 엔트로피의 정규화의 정도를 결정하며 이 값의 설정을 통해 에이전트의 행동을 일관되게 할지 랜덤하게 할지 조절 할 수 있다. max_steps은 에이전트의 학습 진행 횟수를 나타내는 요소로 max_steps을 1,000,000으로 설정하여 에이전트가 관측 후 1번 행동하여 1,000,000번 행동하면 학습이 종료된다. memory_size는 순환 신경망의 은닉 상태(hidden state)를 저장하는데 사용되는 배열의 크기를 의미한다. num_layers은 신경망의 은닉을 몇 개나 사용할지 결정한다. num_epoch는 경사 하강 (Gradient Descent) 학습 업데이트 동안에 버퍼(Buffer) 데이터에 대해 학습을 몇 번 수행할 지 결정한다.

4. 실험 및 결과

본 연구 실험에선 특정 목표를 수행하는 플레이어 에이전트를 보좌하여 더 나은 결과를 내도록 도와주는 에이전트를 강화학습을 통해 구현하는 것이다. 실험에 사용한 에이전트는 DPS 에이전트와 Support 에이전트가 존재하며 DPS 에이전트는 목표 달성(게임의 승리)을 위해 공격 위주로 설계된 에이전트, Support에이전트는 DPS에이전트를 보좌하는 목적으로 설계된 에이전트다. 실험은 DPS에이전트 2개로 구성된 P(Player Agent) 그룹과 DPS 에이전트와 Support 에이전트로 구성된 C(Cooperation) 그룹으로 나뉘어 실험을 진행하였다. 에이전트가 Hyperparameter에서 지정된 학습량을 달성하고 유니티 엔진에서 학습이 끝낸 모델을 사용하여 10번의 시연을 하고 난후의 결과 값을 비교 하였다. 비교 검증의 항목은 에이전트의 평균 사망 횟수, 에이전트가 획득한 최대 보상 값과 평균값을 비교하였다. 실험 결과 C 그

룹이 P 그룹 보다 죽은 횟수의 총합이 63 더 낮았으며 평균 사망 횟수가 3.15 더 낮았다. C 그룹의 DPS 에이전트의 누적 보상 값이 P 그룹의 DPS 에이전트 보다 3.3 더 높았으며 최대로 획득한 보상 값은 4.4 더 높았다.

4.1 사망 횟수 비교

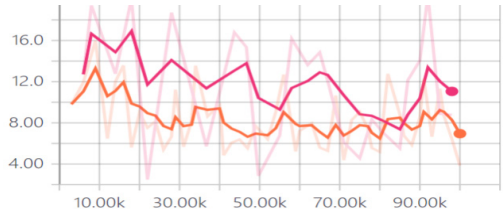
첫 번째 비교 검증 항목인 사망 횟수 비교는 에이전트가 특정 목표를 수행하는데 얼마나 많이 죽었는지를 체크하는 것이다. [Table 5]는 학습이 끝난 에이전트를 유니티 엔진 내에서 10번 동안 시연하여 사망 횟수 비교 실험을 진행한 결과표다. 실험 결과 P 그룹의 총 사망 횟수는 214이며 그룹 평균 사망 횟수는 10.7 C 그룹의 총 사망 횟수는 151이며 그룹 평균 사망 횟수는 7.55를 기록했다. 이와 같은 결과는 Support 에이전트가 협력을 통하여 Player 에이전트의 죽음을 최소화하는 방향으로 학습이 진행되었음을 확인할 수 있었다.

[Table 5] Death count comparison table

Count	Group P		Group C	
	DPS 1	DPS 2	DPS 1	Support
1	11	11	8	10
2	12	12	7	9
3	9	9	6	9
4	9	9	5	5
5	9	8	8	8
6	11	12	8	8
7	10	12	6	7
8	15	15	8	6
9	9	9	7	7
10	10	12	9	10
Death Total	105	109	72	79
Death Average	10.5	10.9	7.2	7.9
Group Total	214		151	
Group Average	10.7		7.55	

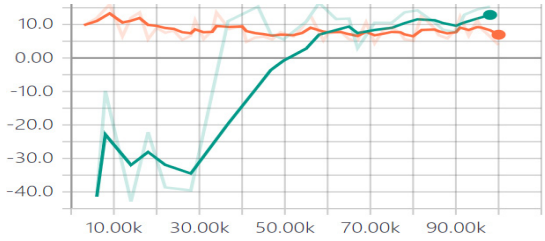
4.2 누적 보상 값 비교

구글 텐서 보드를 사용해 에이전트의 누적 보상 값을 비교 확인하였다. 텐서보드를 이용해 P 그룹과 C 그룹의 누적 보상 값을 그래프로 확인 할 수 있다. 그래프의 X축은 에이전트의 Step의 진행 횟수를 나타내며 Y축은 에이전트의 누적 보상 값을 나타낸다.



[Fig. 12] DPS Agent Cumulative Compensation Graph

[Fig.12]는 텐서보드를 통해 확인할 수 있는 누적 보상 값으로 주황색이 P 그룹의 DPS 에이전트 분홍색이 C 그룹의 DPS 에이전트의 평균 보상 획득 값이다. P 그룹의 DPS 에이전트의 평균 누적 보상 값은 8.1이며 C 그룹의 DPS 에이전트의 평균 누적 보상은 11.4로 C 그룹의 에이전트가 3.3 더 높으며 최대로 획득한 누적 보상 값은 P 그룹 DPS 에이전트는 15.9 C 그룹의 DPS 에이전트는 20.3으로 C 그룹의 에이전트가 4.4 더 높게 나왔다.



[Fig. 13] DPS, Support Agent Cumulative Compensation Graph

[Fig.13]은 Support 에이전트의 누정 보상 그래프로 연두색이 C 그룹의 Support 에이전트의

누적 보상 값이다. 학습 초기에는 -누적 보상 값을 기록 했지만 학습이 진행됨에 따라 점점 상승하였으며 50000 Step이후에는 P 그룹의 DPS 에이전트보다 더 높은 누적 보상을 기록하게 된다. Support 에이전트의 평균 누적 보상 값은 초반의 보상을 기록한 것 때문에 0.6으로 낮지만 보상이 음수가 아닌 양수를 발생했을 때를 기점으로 계산하면 평균 누적 보상 값은 11.1이다. 최대 누적 보상 값은 16.4로 P 그룹의 DPS 에이전트보다 0.5 더 높은 수치를 기록하였다. Support 에이전트의 누적보상 값은 협력을 통한 결과에 대한 보상(Reward)으로 협력을 바탕으로 강화학습을 진행하였음을 확인할 수 있었다.

5. 결 론

본 연구는 MOBA 장르 게임의 목표 달성 협력을 위한 사용자와 함께 참여하는 강화학습 인공지능 프로세스 구축을 목표하였다. 본 연구의 구현을 위한 AI는 PPO 강화학습 알고리즘 기반의 ML_Agent를 사용하여 실험을 진행하였다. 결과 분석을 바탕으로 사용자와 AI간의 협력을 통한 목표 달성이 가능한지 비교검증 하였다. [Table 6]은 실험 결과 값을 정리한 표다.

[Table 6] Experiment result table

Item	Group P		Group C	
	DPS 1	DPS 2	DPS 1	Support
Total Deaths	214		151	
Deaths Total average	10.7		7.55	
Maximum Compensation Value	15.9		20.3	16.4
Average Compensation Value	8.1		11.4	0.6, 11.1

C 그룹이 P그룹과 비교하였을 때 사망 횟수의 총합이 63 낮았으며 사망 횟수 총합 평균은 3.15 낮게 측정되어 C그룹은 협력을 통해 죽음을 최소화하는 방향으로 플레이하는 것을 확인할 수 있었다. 텐서보드의 학습결과 데이터에서 학습초기 C 그룹의 Support 에이전트의 누적 보상 값이 -41.41을 기록하였지만 학습이 진행되면서 개선되어 누적 보상 값은 최대 16.4로 증가하였다. 이는 협력 Support 에이전트가 DPS 에이전트에 맞추어 행동을 결정함으로써 목표 달성 협력을 위한 학습이 진행되었음을 확인할 수 있다. DPS 에이전트로만 이루어진 P 그룹보다 DPS 에이전트와 DPS를 보좌해주는 Support 에이전트로 이루어진 C 그룹은 P 그룹보다 목표를 수행하는데 있어 평균 사망 횟수가 3.2 낮았으며 누적 보상 값 또한 C 그룹의 DPS 에이전트가 P 그룹의 DPS 에이전트보다 평균 누적 보상 값이 3.3 최대 누적 보상 값은 4.4 더 높게 측정 되었다. 또한 Support 에이전트는 게임을 플레이하면서 Tanking 스탯 능력치를 가장 먼저 올리고 이후 Support 스탯과 DPS 스탯을 올리면서 Support 에이전트는 균형잡힌 능력치 상승을 선택하여 게임을 플레이하였다. 이러한 결과 값과 Support 에이전트의 행동을 통해 에이전트가 강화학습을 통해 목표를 수행하더라도 이를 보좌해주는 에이전트에 따라 더 개선된 결과를 보여줄 수 있음을 확인할 수 있었다.

본 연구는 다음과 같은 한계점을 가지고 있다. 첫 번째 한계점은 에이전트 끼리 경쟁하는 환경이 아닌 협력만 가능한 환경에서만 학습실험을 진행하였다. 두 번째 한계점은 실험을 진행하는데 있어 발생할 수 있는 변수 창출의 요소가 기존 상용화된 게임과 비교했을 때 현저히 적다는 점이다. 기존 상용화 게임에서는 아이템, 스킬, 필살기, 특성 등 다양한 요소가 존재하지만 본 실험에서는 구현하지 못했다. 세 번째 한계점은 실험을 진행한 워크스테이션의 연산처리 능력의 한계 때문에 에이전트에게 제한된 학습을 진행할 수밖에 없었다. 네 번째 한계점은 보상을 많이 획득하는 방향으로 행동하는 강화학습의 작동 메커니즘 때문에 특정 목

표를 수행하는데 있어서 발생하는 인과관계를 에이전트에게 전달하는 어려움이 존재하기 때문에 규칙이 더 복잡한 게임에 한해서는 한계점이 존재할 수 있다. 일례로 실험을 진행 하던 중 목표를 수행하지 않고 가만히 있는 게 보상을 최대화 하는 결과가 발생하기도 하여 정지 상태를 유지하거나 목표를 인지하지 못해 특정 지역으로만 이동하는 에이전트도 존재하였다. 이러한 현상의 원인이 학습 부족으로 인한 결과인지 관측, 보상 처벌 설계의 오류로의 결과로 발생하는 것인지 명확하게 파악할 수 없었다. 인공지능의 학습 과정에 관한 논리 구조를 명확히 파악 불가능이 이번 학습 실험에서 판단되었다.

향후 강화학습 알고리즘이 더 발전되고 컴퓨터의 연산 처리 능력이 개선된다면 본 연구의 실험 환경보다 발전된 환경에서의 실험 또한 진행하여 본 연구의 한계점들을 보완하고자 한다. 2019년 현재 알파고를 기점으로 머신 러닝 기반의 인공지능 기술이 폭발적으로 성장하고 발전함에 따라 게임 업계에서도 머신 러닝 기술을 게임에 활용하려는 움직임을 보이고 있다[8]. 본 연구 또한 게임 콘텐츠 개발에 있어서 협력을 위한 머신 러닝 적용 실험 사례 연구로서 본 연구를 통해 향후 머신 러닝을 활용한 게임 개발에 도움 되길 기대한다.

REFERENCES

[1] Gyeong-wan Guk “AI technology and application cases by industry”, BioIN, 2019 March

[2] Jung-Hyun Kim, “Analysis of preference of the AOS playing character based on personality types”, Journal of Next-generation Convergence Information Services Technology, Vol.3, No.1, pp. 61-70, 2014 May

[3] S.Y.Jang, H.J.Yoon, N.S.Park, J.K.Yun, Y.S.Son, “Research Trends on Deep Reinforcement Learning”, Electronics and Telecommunications Trends, Vol.34, No.4, pp. 1-14, 2019 August

[4] Seong-Chan Hong “Artificial Intelligence Design and Implementation of Fighting Game Using Reinforcement Learning”, KwangWoon university graduate school, 2019

[5] Mun-seok Gang "Online Reinforcement Learning That Learns Fast in a Maze Environment", MyongJi university graduate school, 2000 June

[6] Dong-eun Yang “A Study of Tennis game AI through DQN", Korea university graduate school of Computer & Information Technology , 2017 December

[7] Unity, "Introducing: Unity Machine Learning Agents Toolkit", 2017 September 19

[8] B.H. Cho, C.J. Park, “Research Trends in Game AI", Electronics and Telecommunications Research Institut, 2008 August

정 원 조 (Jung, Won Joe)

약 력 : 2012-2015 공주대학교 게임학 박사
2015-2018 우송대학교 게임멀티미디어 초빙교수
2018-현재 굿게임스튜디오 개발이사

관심분야 : 게임엔진, 가상현실, 인공지능, 인터페이스