[1팀]

알고리즘 최종보고서

알고리즘 분석 및 활용한 컨텐츠 제작

1팀 | 알고리즘 김 정 준 교수님 | 12.15

팀장 서 영 찬

팀원 김홍현 이우진

**목 차**

1. **본문 글꼴 및 양식 정의**
2. **대분류는 맑은 고딕. 16pt 굵은글씨**
3. 중분류는 맑은 고딕. 14pt
4. 소분류는 맑은 고딕. 12pt
5. 본문은 11pt 로 작성할 것.

**1. 개요**

1.1알고리즘

1.1.1 강화학습 기초

1.1.2 강화학습 예시

1.1.3 PPO 알고리즘

1.2 알고리즘 특징 및 장점

1.3 팀 구성. 역할 분담 및 일정

1.3.1 팀 구성 및 역할 분담

1.3.2 일정표

**2.팀 활동 내역**

2.1 주차별 활동 내역

**3. 설계**

3.1 설계 개요

3.1.1 중간 발표 개요

3.1.2 최종 발표 설계 개요

3.2 설계 내용

3.2.1 벽돌 깨기 게임 설계 내용

3.2.2 총알 피하기 게임 설계 내용

3.2.3 보물찾기 게임 설계 내용

**4. 소스 코드 및 주석**

4.1 벽돌 깨기 게임 주요 소스 코드 및 주석

4.2 총알 피하기 게임 주요 소스 코드 및 주석

4.3 보물찾기 게임 주요 소스 코드 및 주석

**5. 실행 결과 및 분석**

5.1 실행 결과 표

5.2 실행 결과 그래프

5.3 분석 및 개인 의견

**6. 결론**

6.1 평가

6.2 반성

6.3 향후 계획

**7. 참고 문헌**

**1. 개요**

1.1알고리즘

1.1.1 강화학습 기초

- 머신러닝이란 기계학습이라고 불리며 인공지능의 한 분야로. 컴퓨터가 학습할 수 있도록 하는 알고리즘이다.

종류로는 지도학습, 비지도학습, 강화학습이 있다.

가장 중요한 개념으로는 보상, Agent, Environment, State 등이 있다. 보상이란 스칼라, 즉 벡터가 아닌 피드백 신호로 한 스텝에서 얼마나 잘 하고 있는지에 따른 지표를 의미한다. 이 보상을 다 더하고 그 보상을 최대화 하는 것이 Agent의 역할이다. 그리고 강화학습의 가정이 모든 목적은 축적된 보상을 극대화 하는 것으로 묘사되기 때문에 가장 중요한 개념이다. Agent는 학습시키는 뇌라고 생각하면 되고 바깥에 있는 모든 것이 Agent이외의 모든 것이 환경이다. Agent의 Action을 통해 Observation과 reward가 발생하는데 이 과정을 각 스텝마다 반복한다. 여기서 Agent와 Environment의 State가 존재하는데 Agent의 State는 다음 액션을 하는데 필요한 정보를 의미하고 Environment State는 다음 observation과 reward를 계산하기 위한 것으로 Agent에게는 숨겨지는 부분이다. 추가적으로 Information State 는 Markov state라고도 불리며 내가 결정할 때 상태는 바로 이전의 상태에만 의존한다는 것이다. 또 다른 개념들로는 Policy, Value function, Model등이 있는데 Policy는 Agent의 행동을 규정하는 정책을 의미하고 Value function은 상황에 대한 판단 지표, Model은 Agent가 생각하기에 환경이 어떻게 될지 예측하는 것으로 완전하지 않을 수 있다.

강화학습이 다른 기계학습과 다른 점은 답을 알려주는 supervisor가 없고 보상만 존재하기 때문에 스스로 학습하고 보상이 즉각적이지 않다. 따라서 액션과 보상이 동시에 발생하지 않는다. 순서가 있는 데이터이기 때문에 시간(스텝)이 중요하다. Agent의 액션이 그 다음에 받는 데이터에 영향을 준다. 강화학습의 예시로는 알파고, 백가몬, unity에서 지원하는 ml-agent 등이 있다.

1.1.2강화학습 예시

게임으로 보다 알기쉽게 예시를 들어 설명한 강화학습으로 메탈슬러그 시리즈가 있다. 해당 게임 안의 오브젝트들이 강화학습에 대한 컴포넌트별로 매칭하게 되는데, 각 오브젝트들이 State, Agent, Environment, Action, Reward 5가지로 구성되어 있다고 가정했다. 전체 기능이 아닌 일부만 매칭하였기 때문에 구조만 보면 특정 상황에 대한 컴포넌트 동작의 방식을 구조화하였다. Agent는 사용자가 움직였던 플레이어 객체이고, Environment는 게임의 전반적 환경, Target은 보상을 받을 수 있는 목표물로 컴포넌트를 매칭하였다. 상태를 판별하기 위해 좌표값 데이터를 통해 매핑(Mapping)하고, 일정 강화학습을 거치고 나면, 특정 State에서 뉴럴네트워크를 거쳐서 나온 데이터를 통해 Action에서 2라는 값이 반환되며, 2라는 Action에 해당하는 행동은 총알을 발사하는 것으로 매핑했다고 가정하면, Agent는 해당 State에서 총알을 발사하게 되며, Target이 이를 맞고 사라짐과 동시에 Environment의 상태가 변화하고 보상을 받는 형식이다. 전반적인 State-Action-Reward절차는 Q함수와 큰 관련이 있는데, Q함수의 의미는 상태 s에서 액션 a를 수행했을 때 기대되는 미래가치라고 보면 된다. Q함수의 동작으로 보상이 결정되고 이 보상들을 극대화하는 것이 강화학습의 목적이다.

1.1.3 PPO 알고리즘

PPO 알고리즘은 Proximal Policy Optimization의 약자로 이름 그대로 Policy gradient Method를 최적화하는 알고리즘이다. 여기서 Policy gradient란 이름 그대로 정책의 방향으로 Policy에대한 수식을 편미분한 값이 향하는 방향을 의미한다. 여기서 Policy란 머신러닝 학습 방법 중 하나로 머신러닝 학습방법에는 Value Based랑 Policy Based가 있다. Value Based은 그 시점에서의 최적 하나의 값 만을 찾는 Greedy한 방법으로 Deterministic policy라고 할 수 있다. 또한 Policy Based는 다음 행동을 확률적으로 결정하는 방식으로 stochastic policy라고 할 수 있다. 이제 Policy gradient를 최적화해야 하므로 이에 대한 수식을 정해야 한다. 이 중 두가지 목적 함수 Start value (episode 기반 환경)와 Average reward per time-step을 정의한다. 각각 Start value 𝑱\_𝟏 (𝜽)= 𝑽^𝝅𝜽 (𝒔\_𝟏)=𝑬\_𝝅𝜽 [𝒓𝒆𝒘𝒂𝒓𝒅] 로,

Average reward per time-step **=** 로 정의한다. 이 두 함수를 연립하여 [수식 1]과 같이 편미분 한다.

**=**

**=   
 = log**

**=   
 =**

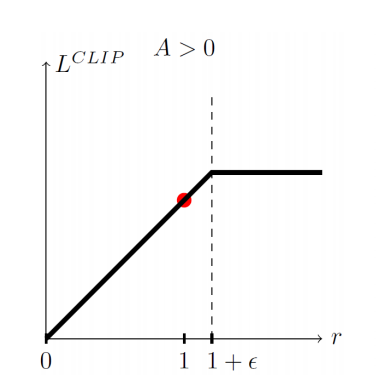
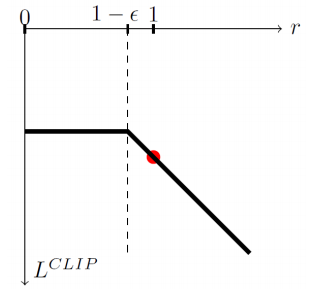
[수식 1]

여기서 를 로 바꾸면 Multi Step에도 적용이 가능하다.

즉 Multi Step MDP에서의 Policy gradient는 다음과같이 정의가 가능하다.  **=**

여기서 Clip을 걸어 제한을 준 것이 PPO 알고리즘이다.

**=** 를 적분하여 **L = ]** 라는 결과를 얻어내고 여기에 Clip을 걸어  **=** 일 때, [그림1] 과 같이 **L =** 의 결과를 얻을 수 있고 이를 PPO 알고리즘이라 한다.



[그림 1]

1.2 PPO알고리즘 특징 및 장점

1.2.1 Clipped 방법을 사용하는 이유

PPO의 철학이 여러 번 업데이트하여 경험을 쌓게 하여 학습하는 것이기 때문에 한번 업데이트 될 때 너무 큰 차이가 발생하면 안 된다. 따라서 범위를 벗어나는 샘플은 우리가 원하는 결과와 다르기 때문에 그 범위를 벗어나면 그 샘플(에피소드)은 버리는 효과가 발생한다.

1.2.2 PPO 장점

알고리즘으로 학습하기에 간단하고 프로그램이 실행되거나 할당된 작업을 실행하는데 걸리는 실제 시간이 적다. 각 상황에 맞추어 기계학습을 접목하는 것이 바람직하기 때문에 꼭 이 알고리즘이 성능이 뛰어나다고는 할 수 없다.

1.3 팀 구성. 역할 분담 및 일정

1.3.1 팀 구성 및 역할 분담

|  |  |  |
| --- | --- | --- |
| 직책 | 이름 | 역할 |
| 팀장 | 서영찬 | 강화학습 알고리즘 원리 설명 및  보물찾기 게임 설계, 분석 |
| 팀원 | 김홍현 | 강화학습 예시 수집 및  총알 피하기 게임 설계, 분석 |
| 팀원 | 이우진 | 강화학습 기초 설명 및  벽돌 깨기 게임 설계, 분석 |

1.3.2 일정표

|  |  |
| --- | --- |
| 주차 | 일정 |
| 1 | 알고리즘 기초 조사 및 팀원간 화합 도모 |
| 2 | 알고리즘 주제 선정. |
| 3 | 역할 분담 및 자료조사 |
| 4 | 조사한 자료 회의 및 발표자료 작성 |
| 5 | 1차 발표 |
| 6 | 알고리즘 활용한 게임 설계 |
| 7 | 알고리즘 활용한 게임 설계 및 분석 |
| 8 | 2차 발표. |

**2.팀 활동 내역**

2.1 주차별 활동 내역

|  |  |
| --- | --- |
| 주차 | 활동내역 |
| 1 | 팀원을 결정하고 주제를 결정하는 것까지 여러 알고리즘 후보가 있었는데 빅데이터나 영상처리 관련 알고리즘도 있었지만 기계학습 관련 알고리즘이 적합하다고 생각되어 PPO알고리즘으로 결정했다. |
| 2 | PPO 알고리즘으로 결정하고 각자 조금씩 자료조사를 하고 중간고사에 집중하기로 했다. |
| 3 | 역할을 나누지 않고 자료조사한 결과로 좋은 아이디어를 내지 못 했기 때문에 각자 역할을 정하여 자료조사를 다시 하기로 결정했다. |
| 4 | 역할별로 조사해 온 자료를 공유하고 서로 설명해주면서 이해가 되지 않는 부분과 설명이 부족한 부분을 피드백 해주었다. 발표자료를 만들면서 어려운 알고리즘을 쉽게 설명하는 방법이 제일 어려웠다. |
| 5 | 1차 발표 |
| 6 | 1차 발표 이후에 PPO알고리즘을 활용한 게임 제작을 결정했기 때문에 unity를 활용한 ml-agent를 사용하기로 했고 각자 게임을 하나씩 맡아서 설계하기로 했다. 벽돌 깨기, 총알 피하기, 보물찾기 게임으로 결정했다. |
| 7 | ml-agent 사용 방법과 unity 설계에서 어려운 부분이 많았기 때문에 기존에 존재하는 자료를 많이 참고했다. |
| 8 | 2차발표 |

**3. 설계**

3.1 설계 개요

3.1.1 중간 발표 개요

중간 발표는 팀원과 강의를 같이 듣는 사람들에게 알고리즘을 보다 쉽게 이해시키기 위한 점을 초점으로 맞추어 발표를 진행했다. 그래서 어려운 개념보다는 쉽고 중요한 개념 위주로 설명을 했고 어려운 수식이 있지만 이해하지 않으면 PPO알고리즘을 사용할 수 없기 때문에 설명을 진행했다.

3.1.2 구현 발표 개요

구현 발표는 중간 발표에서 소개했던 알고리즘을 이용한 응용 프로그램을 소개하는 것인데 각자 맡은 게임에 대해 소개했고 소스코드 설명보다는 작동 원리나 방식과 같은 부분에 대해 소개했다.

3.2 설계 내용

3.2.1 벽돌 깨기 게임 설계 내용

기본적인 벽돌 깨기 게임은 공과 공을 제어하는 막대로 구성되어 있는데 이 공을 이용하여 최대한 많은 벽돌을 깨는 것이 기본 개념이다. 이를 통해 얼마나 많은 벽돌을 깨는 것이 중점이 되었고 보상을 주는 기준도 벽돌을 깨는 개수와 상관이 있다. 벽돌 깨기는 2차원 게임으로 제어할 수 있는 막대는 가로축으로만 움직일 수 있기 때문에 x축만 제어한다. 모든 벽돌을 다 깨거나 공이 바닥에 닿을 경우에 Done함수를 호출해 Reset함수를 호출해서 다시 시작하게 했다. 수집하는 정보로는 3개의 안쪽 벽돌 상태, 공과 벽돌의 거리, 막대의 위치와 공의 위치 등이다.

3.2.2 총알 피하기 게임 설계 내용

총알 피하기 게임은 무작위로 날아오는 총알을 피해 최대한 오래 살아남는 게 게임의 주 목표다. 총알 피하기는 2D 평면 안에서 상하좌우로 움직이기 때문에 AI단에서 컨트롤하는 input은 x,y축 기준으로 구별하여서 움직이게 했다. 총알이 비행기 기체에 닿을 시 Done함수를 호출해 Reset함수를 호출하게 하여 게임을 다시 시작하게 했다. 수집하는 정보로는 총알이 기체 주변을 둘러싸고 있는 판단 패널에 대해 각 8개 평면이 공에 닿은 여부, 기체의 위치와 기체의 속도가 있다.

3.2.3 보물 찾기 게임 설계 내용

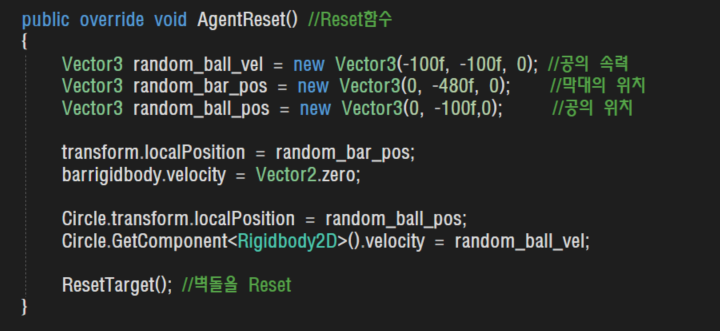
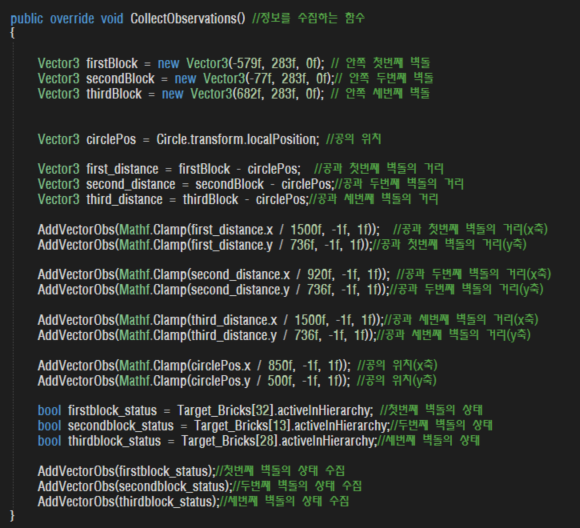
보물 찾기 게임은 랜덤하게 생성된 보물과 적 사이에서 보물을 골라 획득하며 최대한 빠르게 스테이지를 돌파하는 게임이다.

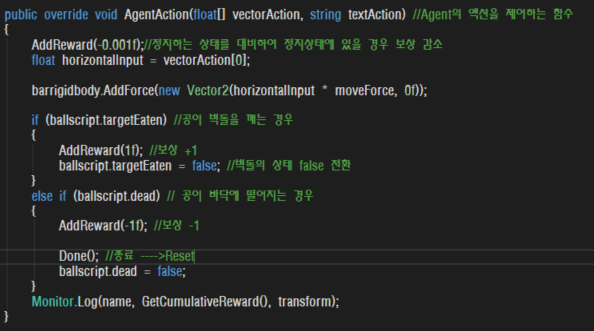
Player가 전진할 때마다 눈앞에 3가지 선택지에 대한 사항이

계속 Update 되며 왼쪽, 중앙, 오른쪽에 각각 몬스터가 있는지 없는지에 대한 내용과 Player의 현재 위치를 수집하여 AI가 다음 행동을 판단할 수 있도록 하였다. 단순한 학습이기에 보물을 먹으면 보상을, 몬스터에게 닿으면 처벌을 내리고 초기화 하도록 설계하였다.

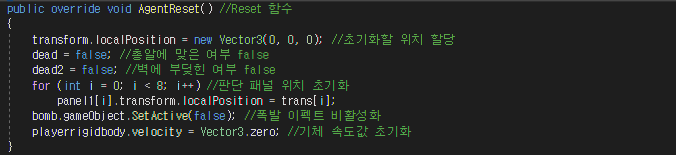
**4. 소스 코드 및 주석**

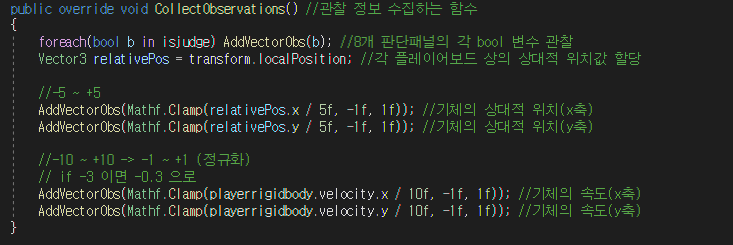
4.1 벽돌 깨기 게임 주석 및 소스코드

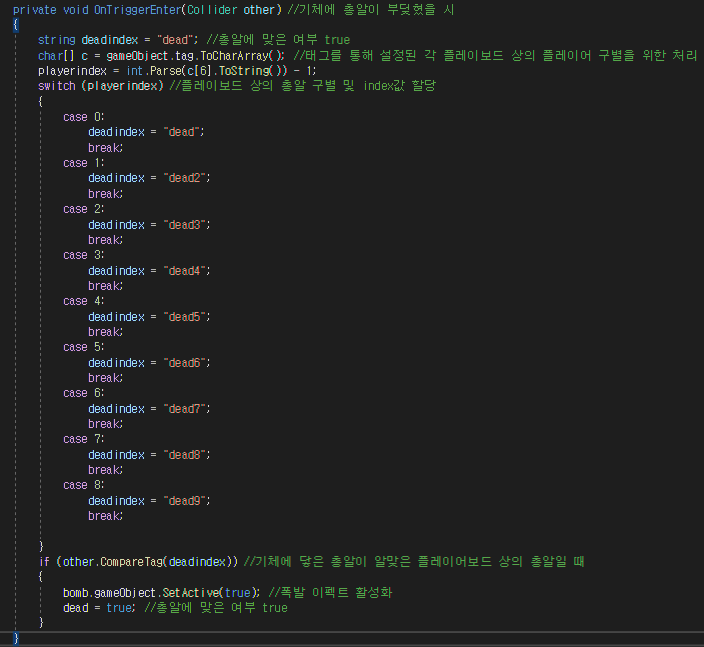
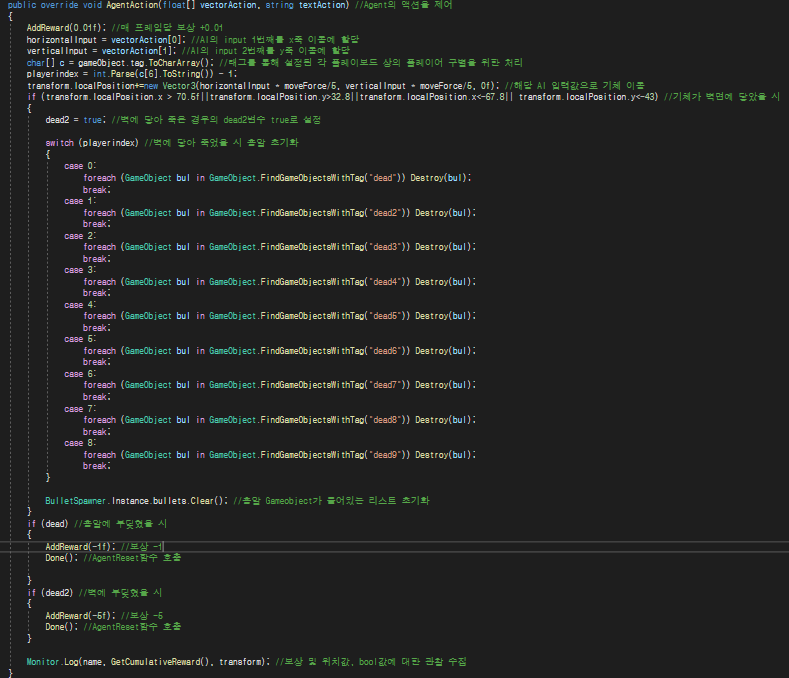
 



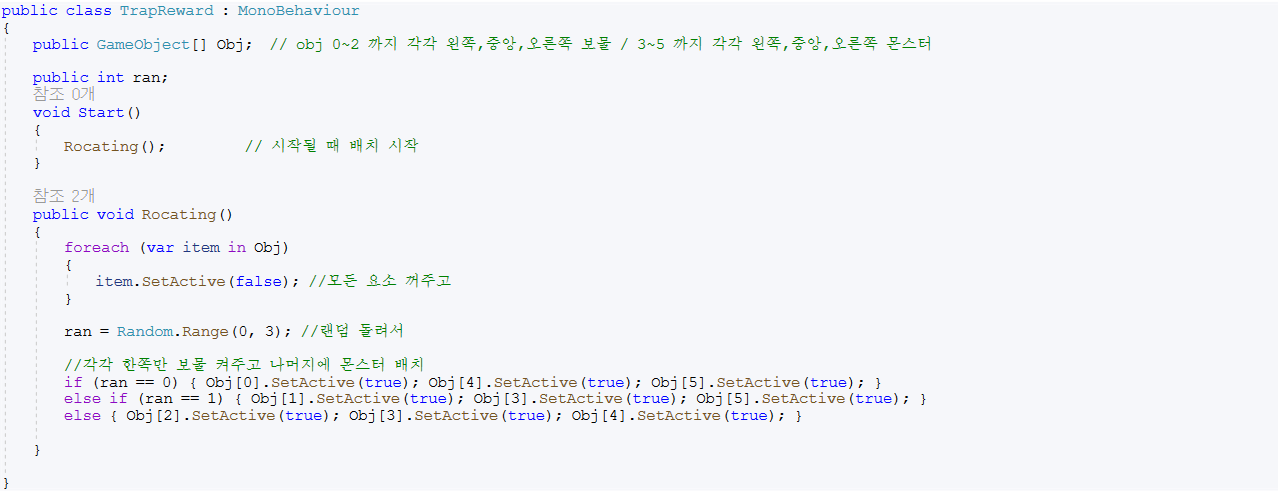
4.2 총알 피하기 게임 소스코드 및 주석

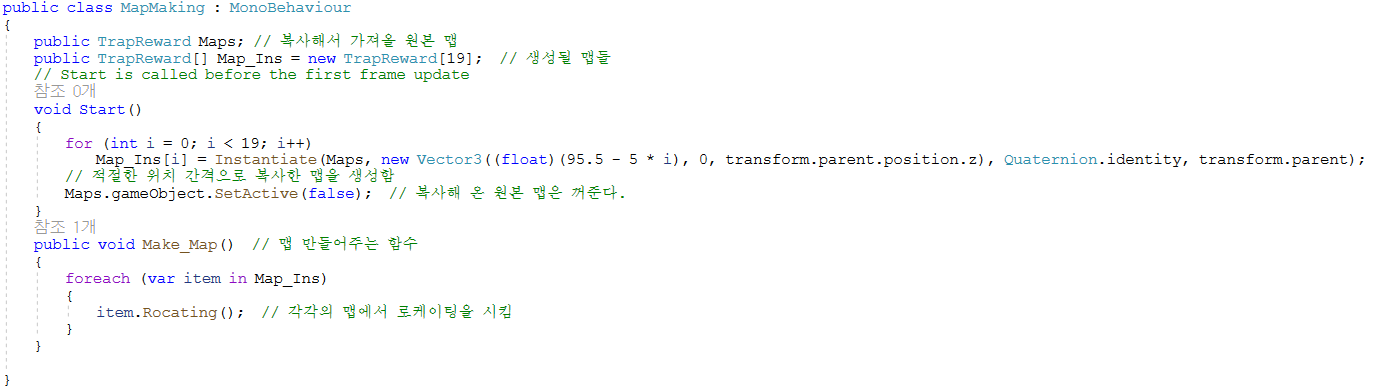


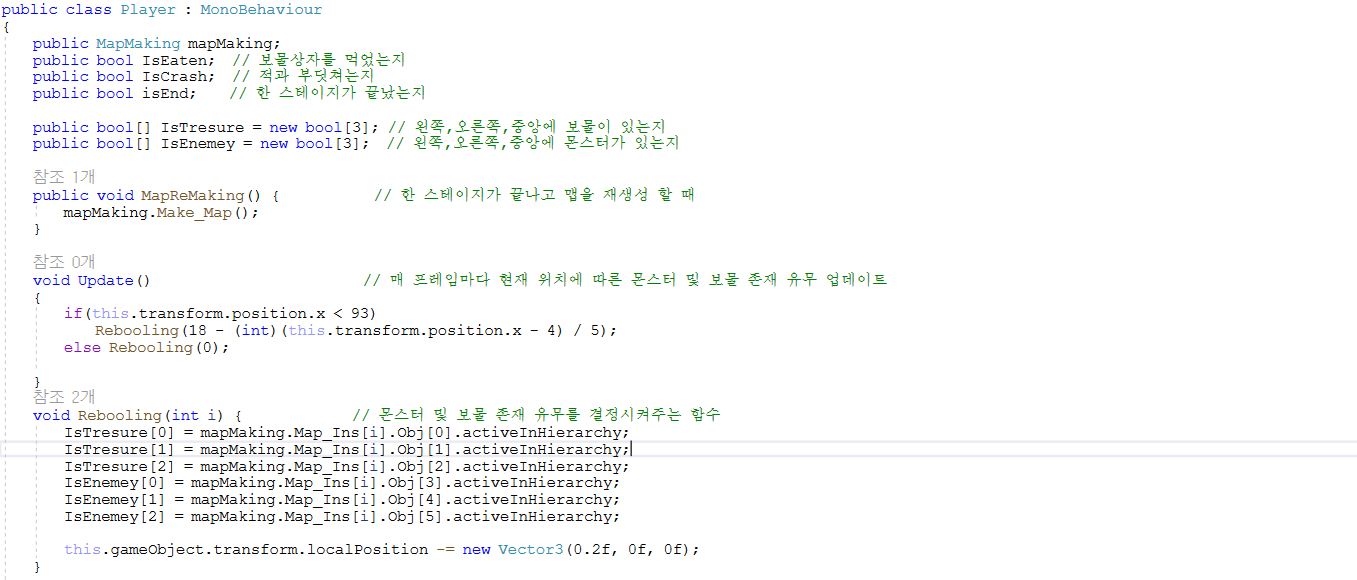


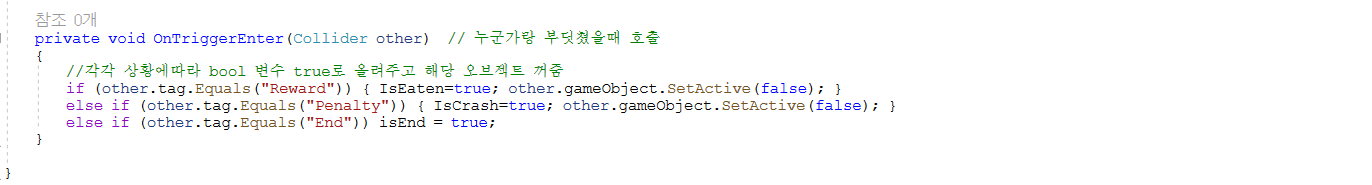


4.3 보물찾기 게임 소스코드 및 주석

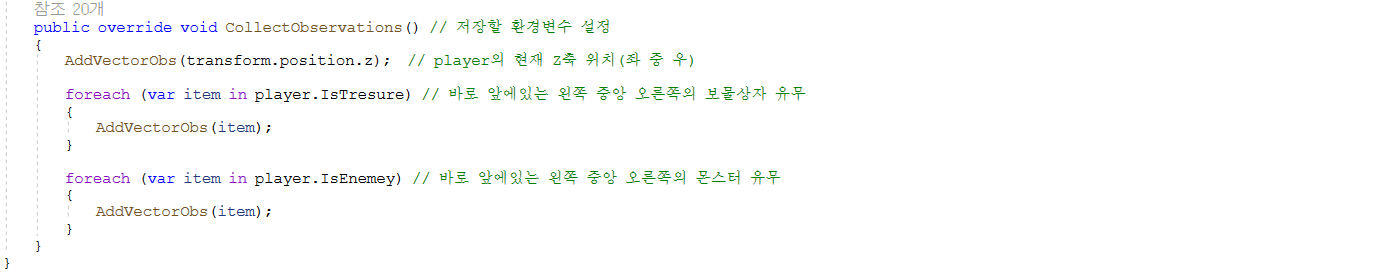








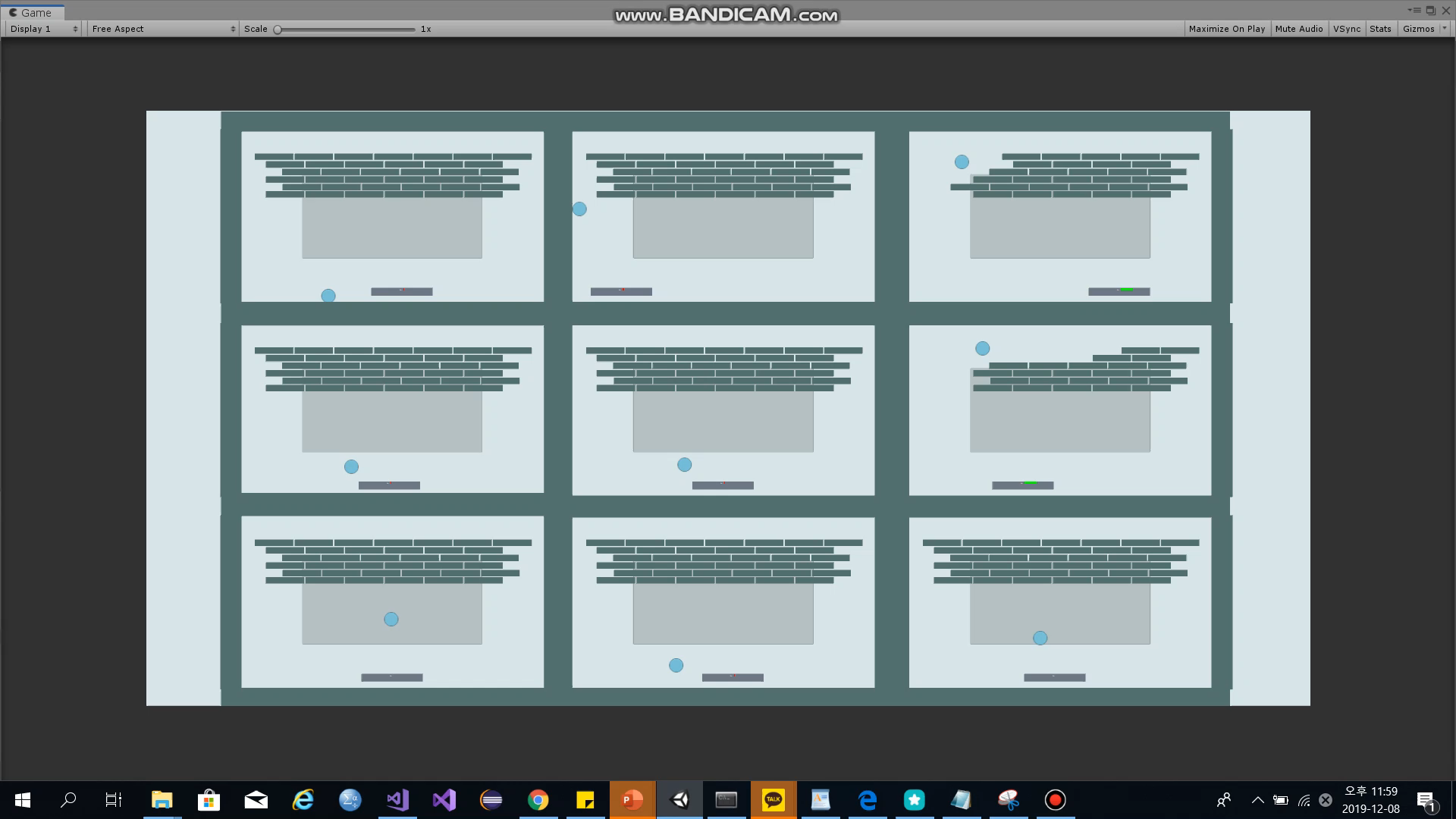




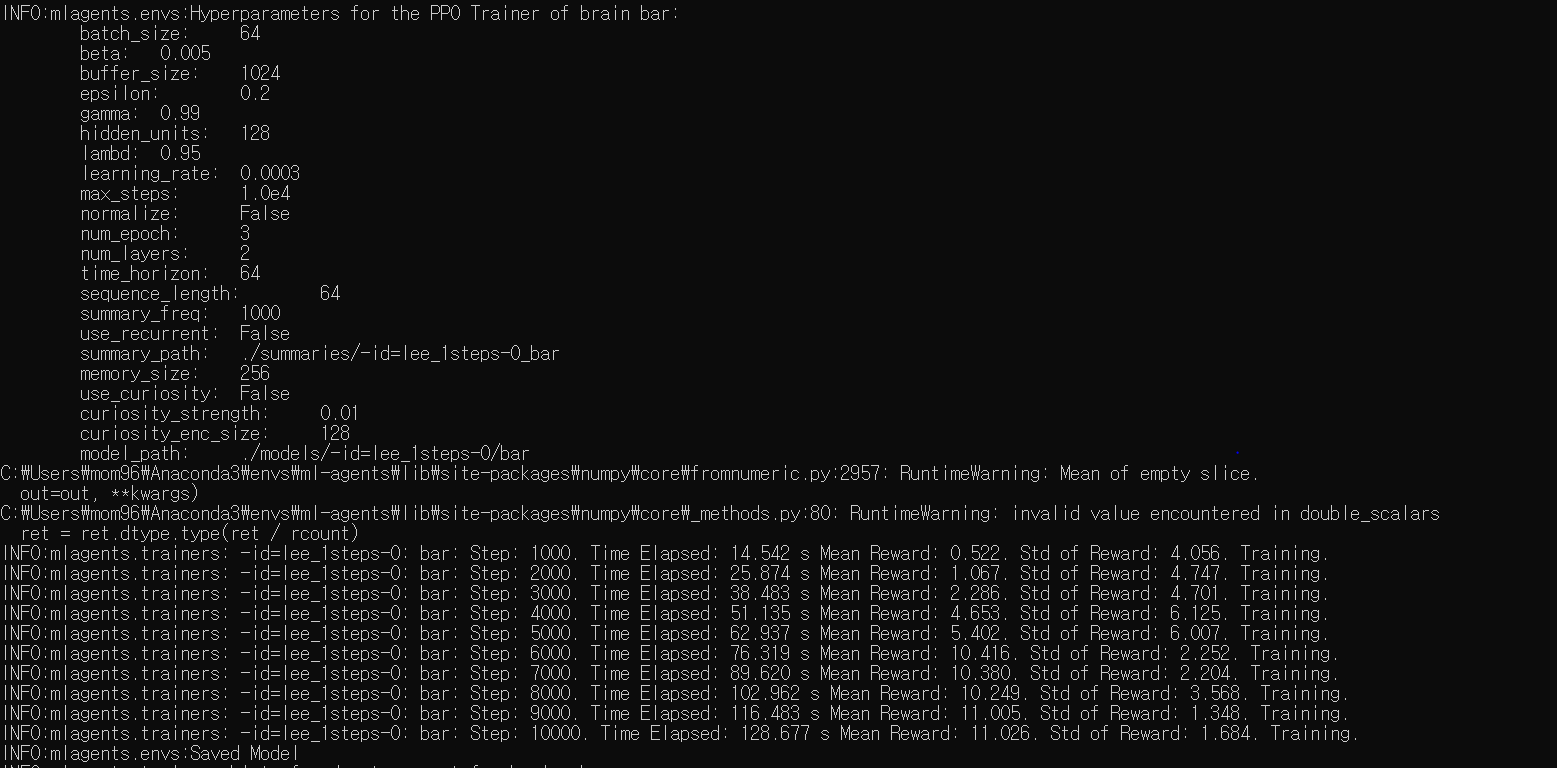
**5. 실행 결과 및 분석**

5.1 벽돌 깨기 실행 결과 화면

5.1.1 벽돌 깨기 실행 결과 화면 – 게임화면



5.1.2 벽돌 깨기 실행 결과 화면 – 콘솔화면



5.2 실행 결과 그래프

5.2.1 보상 그래프 – 벽돌 깨기 게임

Batch-size 64 / buffer-size 1024 / epsilon 0.3기준

Batch-size 16 / 64 / 256

Buffer-size 64 / 256 / 1024

Epsilon 0.1 / 0.2 / 0.3

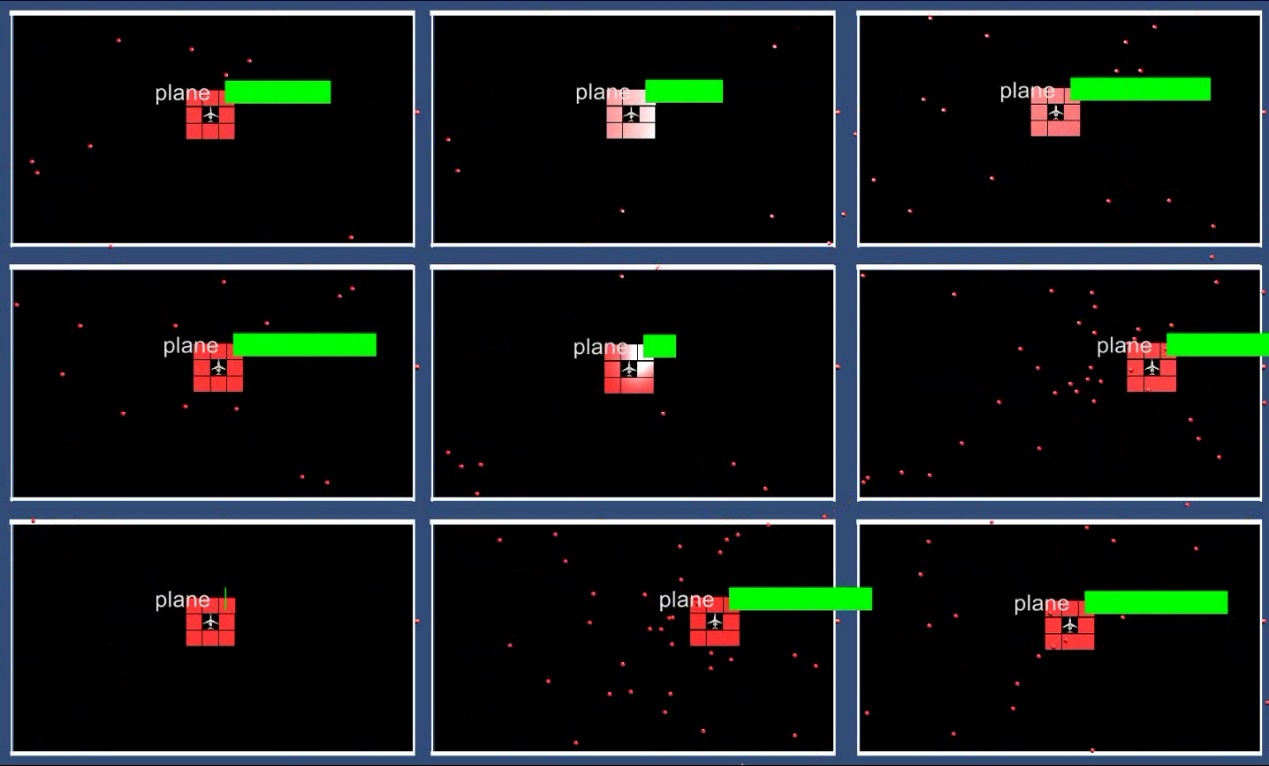
변화하면서 보상 비교

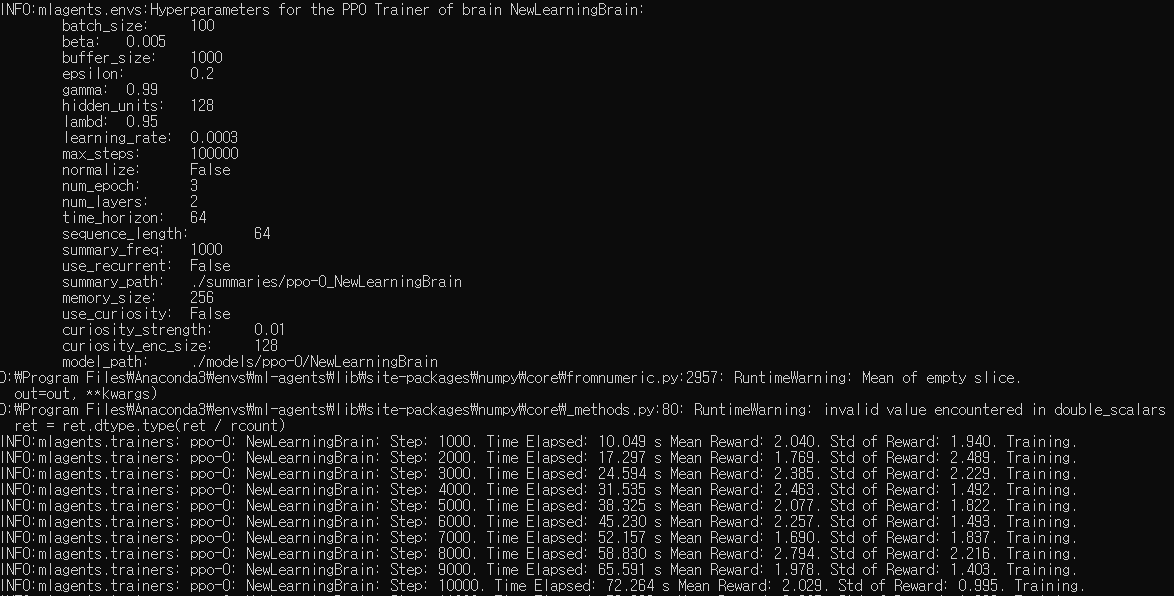
5.2.1.1 보상 그래프 해석 – 벽돌 깨기 게임

Batch-size는 buffer-size가 1024로 고정되어있을 때 16이 가장 이상적이었고 증가할수록 느슨하게 보상이 증가하는 것을 볼 수 있다. Buffer-size는 batch-size와 크기가 같을 때 보상이 증가하지 않고 -1에서 머무른 것을 보고 buffer-size는 batch-size의 배수여야 한다는 가정이 맞다는 것을 볼 수 있다.

Epsion은 0.1~0.3까지 증가시켜 보았지만 크게 차이는 보이지 않는다.

5.2 총알 피하기 실행 결과 화면

5.2.1 총알 피하기 실행 결과 화면 – 게임화면 

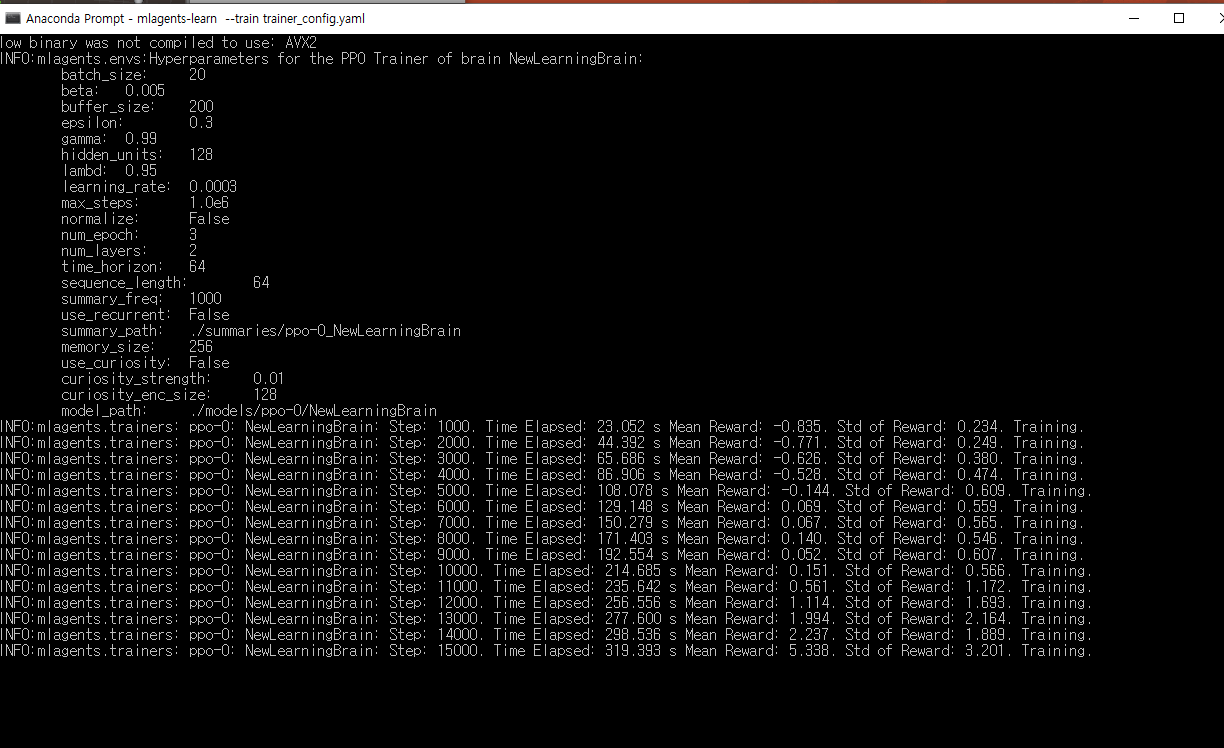
5.2.2 총알 피하기 실행 결과 화면 – 콘솔화면

5.3 보물찾기 실행 결과 화면

5.3.1 보물찾기 실행 결과 화면 – 게임화면



5.3.2 보물찾기 실행 결과 화면 – 콘솔화면



5.4 분석 및 개인 의견

김홍현: 총알 피하기 게임은 판단 패널의 bool 값을 통해 총알이 닿을 것인지 여부를 판단하는 게임이라 하이퍼 파라미터를 바꾸는 것에 대해 큰 영향을 주지 못하였다. 다음 기회가 있으면 위치값과 속도값 등을 동적으로 받아서 좀더 똑똑한 AI를 만들 수 있었으면 좋겠다.

이우진: 벽돌 깨기 게임은 간단하게 막대로 공을 이용하여 벽돌을 깨는 게임이기 때문에 epsilon값은 크게 영향을 주지 못했고 batch-size와 buffer-size는 4배 차이가 날 때 가장 이상적인 결과를 내는 것으로 보인다. 간단한 게임일수록 batch-size는 작은 것이 좋고 buffer-size도 batch-size에 맞추어 줄이는 것이 좋을 것 같다.

-서영찬 : 복잡한 게임이더라도 간단한 상황으로 변환해 최소한의 데이터를 수집하고 리워드를 줄 수 있게 설계하는 것이 중요하다고 생각이 들었다.

초기 설계 단계에서 거리 등의 복잡한 상황을 그대로 전달받아 저장했을 때는 50만번 넘게 학습해도 학습에 진전이 없었기에 간소화 단계를 걸쳤다. 결국 Player의 위치 및 보물 상자 유무로 간소화하고 이러한 세팅으로 훈련한 결과5만 번 안에 학습을 완료할 수 있었다.

각각의 콘텐츠에 따른 하이퍼 파라미터도 중요하지만, 게임에서 동적으로 전달해주는 데이터가 얼마나 간단하고 직관적인지가 AI 학습에 영향을 준다는 것을 알 수 있었다.

**6. 결론**

6.1 평가

팀원의 협조가 매우 매끄럽게 이루어진 팀 프로젝트였으며, 중간고사 발표에서 알고리즘 소개에 집중하여 쉽게 설명하지 못했던 점을 피드백 하여 기말때는 세부적인 설계보다는 전체적이고 대략적인 설계를 보여주었고 코드 설명보다는 작동원리나 예상되는 결과, 매개변수에 따른 보상 그래프를 통해 해석을 설명하였다.

**7. 참고 문헌**

UCL Course on RL David Silver의 강의자료

<http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/intro_RL.pdf>

<http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/pg.pdf>

ppo 논문

<https://arxiv.org/pdf/1707.06347.pdf>

A3C 논문

<https://arxiv.org/abs/1602.01783>

Unity 머신 러닝 에이전트 git 및 toolkit 관련

<https://github.com/Unity-Technologies/ml-agents>

<https://github.com/Unity-Technologies/ml-agents/blob/master/docs/Readme.md>

Unity 머신 러닝 에이전트 실행 및 분석 가이드

<https://hub.packtpub.com/implementing-proximal-policy-optimization-ppo-algorithm-in-unity-tutorial/>