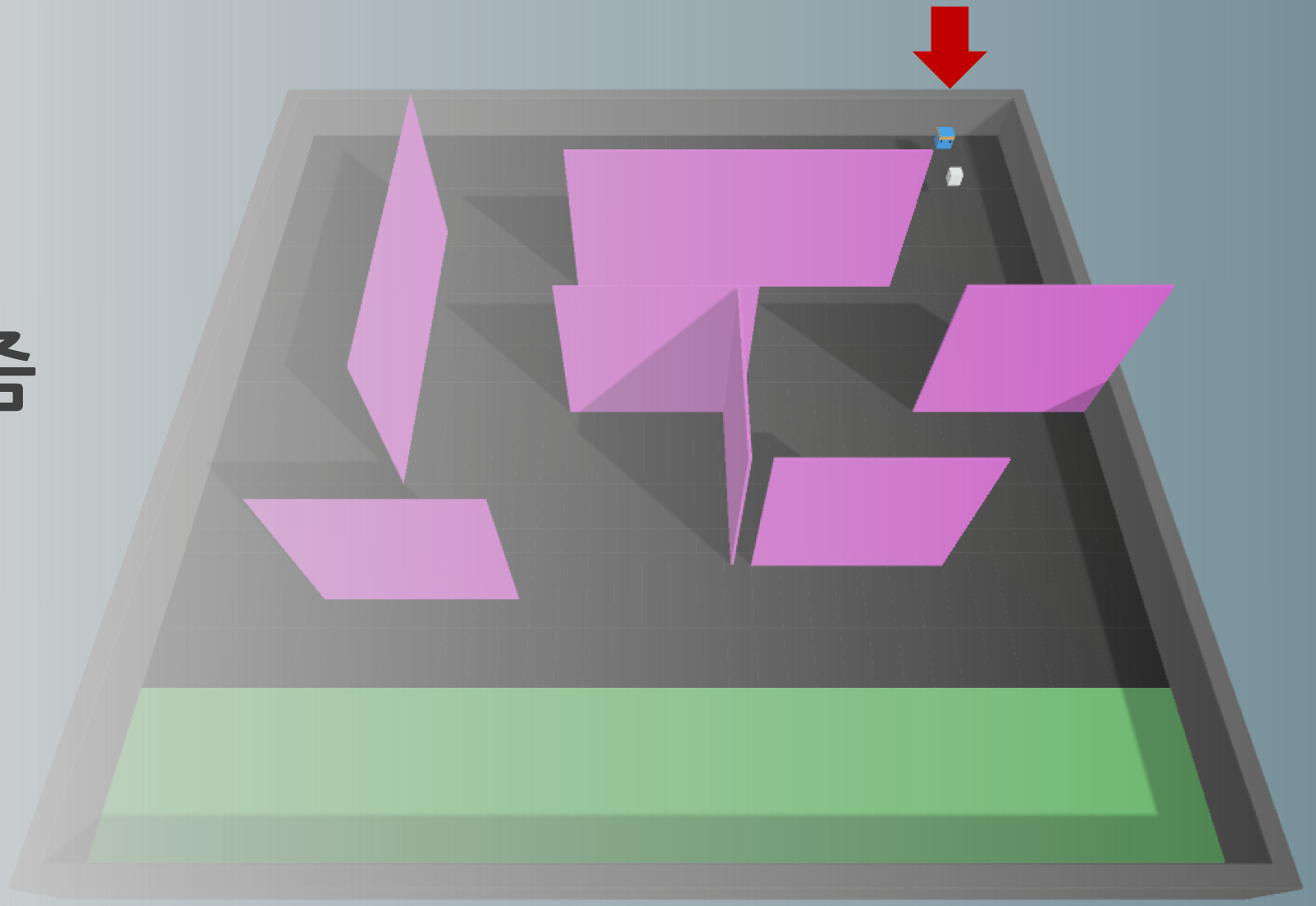
인공지능과 알고리즘

IC-PBL Project2:
Unity ML-Agent 기반의 강화학습 구현

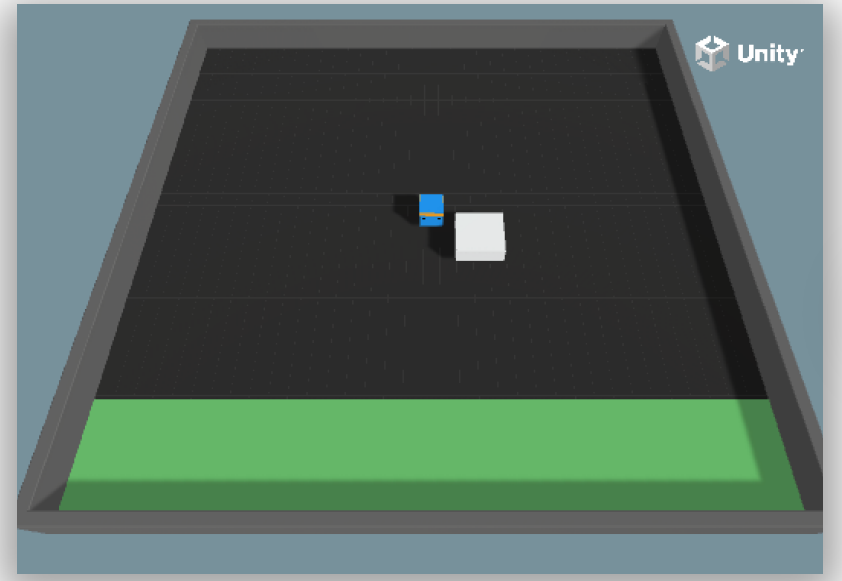


김준우 2019066917
최태훈 2020011994
김지훈 2020045123

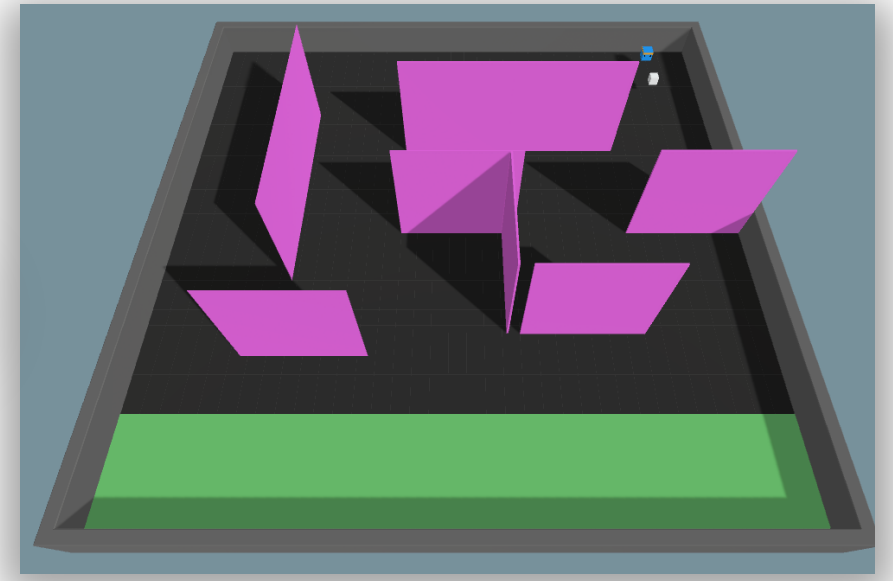
목차

예제 설명

기본 예제

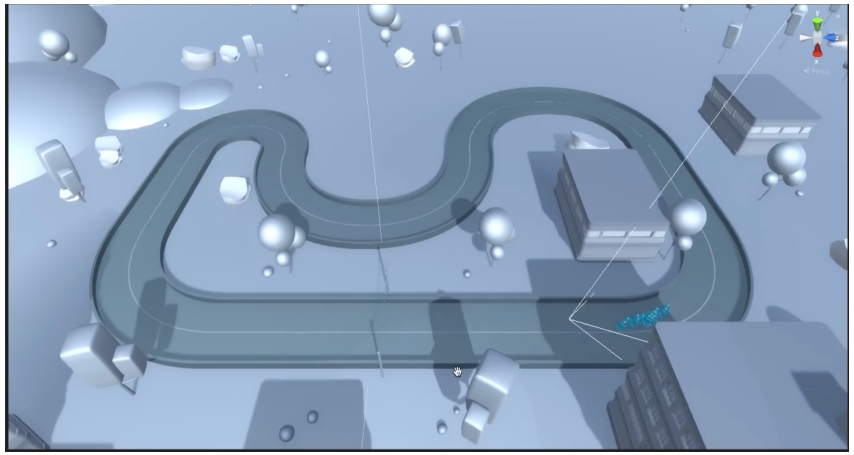


수정 예제



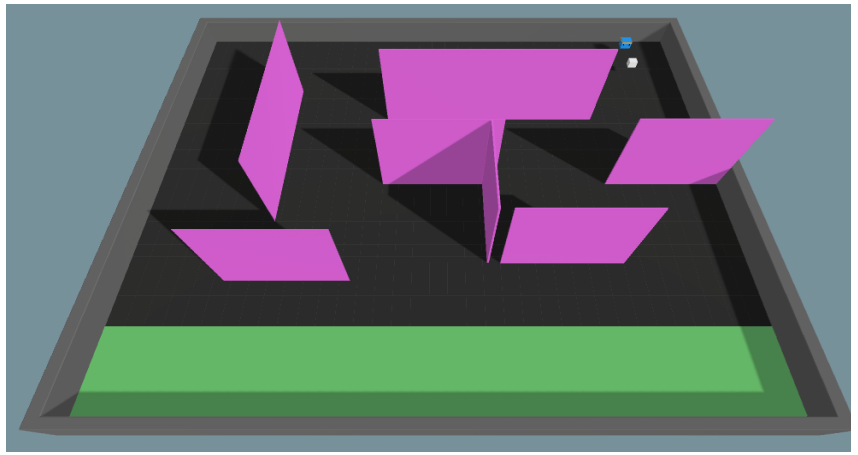
장애물을 넣어 에이전트가 장애물에 닿으면 **페널티**를 주는 부분을 추가

예제 설명



Kart Racing Game with Machine Learning

Idea



보상 Checkpoint -> 페널티 Checkpoint

https://www.youtube.com/watch?v=i0Vt7l3XrIU

Unity Technologies

Verified

10k followers · Copenhagen, Denmark · http://unity.com

Follow

You can now follow organizations

Organization activity like new discussions, sponsorships, and repositories will appear in your dashboard feed.

OK, got it!

Popular repositories

ml-agents — Public

The Unity Machine Learning Agents Toolkit (ML-Agents) is an open-source project that enables games and simulations to serve as environments for training intelligent agents using deep reinforcement ...

C# 17.3k 4.2k

UnityCsReference — Public

Unity C# reference source code.

C# 11.9k 2.5k

EntityComponentSystemSamples — Public

C# 7.2k 1.6k

FPSSample — Public

A first person multiplayer shooter example project in Unity

C# 4.9k 1.8k

PostProcessing — Public archive

Post Processing Stack

C# 3.7k 749

arfoundation-samples — Public

Example content for Unity projects based on AR Foundation

C# 3.1k 1.2k

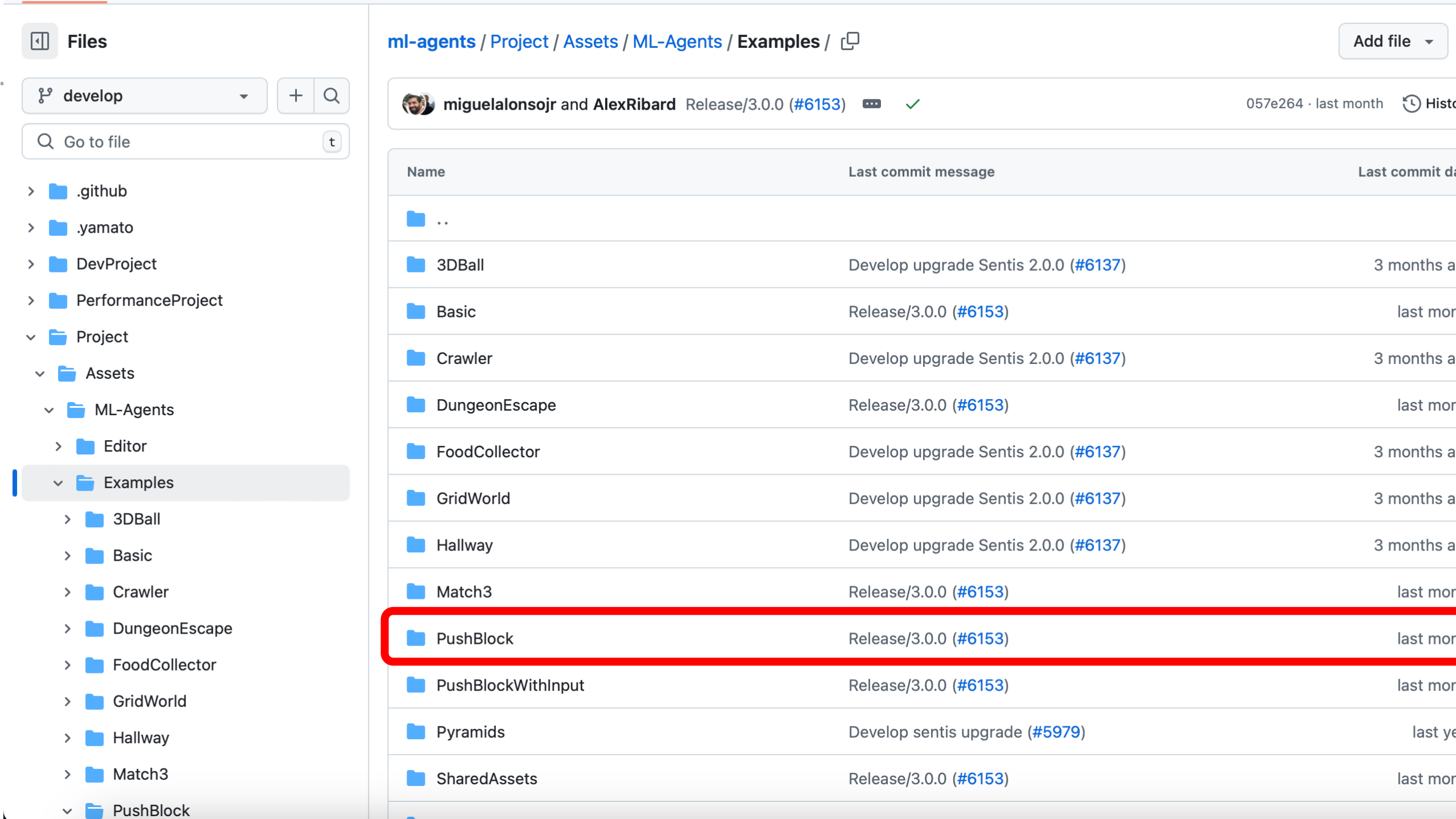
View all

Top languages

C# Go Python C++ C

Most used topics

unity unity3d billing-7054 machine-learning robotics-simulation


Files
develop
Go to file
t
.github
.yamato
DevProject
PerformanceProject
Project
Assets
ML-Agents
Editor
Examples
3DBall
Basic
Crawler
DungeonEscape
FoodCollector
GridWorld
Hallway
Match3
PushBlock
ml-agents / Project / Assets / ML-Agents / Examples /
Add file
miguelalonsojr and AlexRibard Release/3.0.0 (#6153)
057e264 · last month
Name
Last commit message
..
3DBall
Develop upgrade Sentis 2.0.0 (#6137)
3 months a
Basic
Release/3.0.0 (#6153)
last mor
Crawler
Develop upgrade Sentis 2.0.0 (#6137)
3 months a
DungeonEscape
Release/3.0.0 (#6153)
last mor
FoodCollector
Develop upgrade Sentis 2.0.0 (#6137)
3 months a
GridWorld
Develop upgrade Sentis 2.0.0 (#6137)
3 months a
Hallway
Develop upgrade Sentis 2.0.0 (#6137)
3 months a
Match3
Release/3.0.0 (#6153)
last mor
PushBlock
Release/3.0.0 (#6153)
last mor
PushBlockWithInput
Release/3.0.0 (#6153)
last mor
Pyramids
Develop sentis upgrade (#5979)
last ye
SharedAssets
Release/3.0.0 (#6153)
last mor

코드 설명

코드 실행 흐름 요약

1.에피소드 시작 시 OnEpisodeBegin()에서 환경 초기화.

2.매 스텝마다 OnActionReceived()에서 에이전트의 행동을 수행하고 보상/페널티 계산.

3.블록이 목표 지점에 도달하면 ScoredAGoal()로 보상 부여 및 에피소드 종료.

4.충돌 이벤트는 OnCollisionEnter()에서 처리.

코드 설명

클래스 개요

PushAgentBasic 클래스는 ML-Agents의 Agent 클래스를 상속받아 강화학습 에이전트를 정의한다.

1. State (상태)
   - 상태는 에이전트가 현재 환경에서 관찰할 수 있는 정보이다.
   - 이 정보를 바탕으로 에이전트는 최적의 행동을 결정한다.
   - PushAgentBasic 코드에서 상태는 에이전트의 위치, 블록의 위치, 목표의 위치 등으로 구성된다.

   에이전트의 위치: transform.position

   블록의 위치: block.transform.position

   목표 위치: goal.transform.position

   에이전트와 블록의 속도: m_AgentRb.velocity, m_BlockRb.velocity

코드 설명

2. Action (행동)

- 행동은 MoveAgent(ActionSegment<int> act)에서 처리된다.
- 행동은 이산형(Discrete) 행동 공간으로 설정되어 있으며,
  아래과 같은 동작이 가능하다:

  - 1: 앞으로 이동.
  - 2: 뒤로 이동.
  - 3: 오른쪽으로 회전.
  - 4: 왼쪽으로 회전.
  - 5: 왼쪽으로 이동(평행이동).
  - 6: 오른쪽으로 이동(평행이동).

```
/// <summary>
/// Moves the agent according to the selected action.
/// </summary>
public void MoveAgent(ActionSegment<int> act)
{
    var dirToGo = Vector3.zero;
    var rotateDir = Vector3.zero;

    var action = act[0];

    switch (action)
    {
        case 1:
            dirToGo = transform.forward * 1f;
            break;
        case 2:
            dirToGo = transform.forward * -1f;
            break;
        case 3:
            rotateDir = transform.up * 1f;
            break;
        case 4:
            rotateDir = transform.up * -1f;
            break;
        case 5:
            dirToGo = transform.right * -0.75f;
            break;
        case 6:
            dirToGo = transform.right * 0.75f;
            break;
    }
    transform.Rotate(rotateDir, Time.fixedDeltaTime * 200f);
    m_AgentRb.AddForce(dirToGo * m_PushBlockSettings.agentRunSpeed,
        ForceMode.VelocityChange);
}
```

코드 설명

3. Reward (보상)

- 보상은 에이전트가 학습 과정에서 자신의 행동이 얼마나 효과적인지 평가받는 기준이다.
- 긍정적인 보상은 원하는 행동을 강화하고, 부정적인 보상(페널티)은 잘못된 행동을 억제한다.

1) 목표 달성 보상

```
/// </summary>
public void ScoredAGoal()
{
    // We use a reward of 10.
    AddReward(10f);
```

2) 시간 기반 페널티

```
/// </summary>
public override void OnActionReceived(ActionBuffers actionBuffers)

{
    // Move the agent using the action.
    MoveAgent(actionBuffers.DiscreteActions);

    // Penalty given each step to encourage agent to finish task quickly.
    AddReward(-1f / MaxStep);
}
```

3) 잘못된 행동 페널티

```
void OnCollisionEnter(Collision collision)
{
    if (collision.collider.CompareTag("PenaltyZone"))
    {
        // 감점 부여
        AddReward(-1f/MaxStep); / -1/MaxStep 감점
        Debug.Log("Penalty applied: Collided with PenaltyZone!");
    }
}
```

코드 설명

기본 예제 수정

- 에이전트가 장애물에 닿으면 페널티를 주는 부분 추가

```
void OnCollisionEnter(Collision collision)
{
    if (collision.collider.CompareTag("PenaltyZone"))
    {
        // 감점 부여
        AddReward(-1f/MaxStep); // -1/MaxStep 감점
        Debug.Log("Penalty applied: Collided with PenaltyZone!");
    }
}
```

- 블록을 밀었을 때의 보상 증가

```
/// </summary>
public void ScoredAGoal()
{
    // We use a reward of 10.
    AddReward(10f);
```

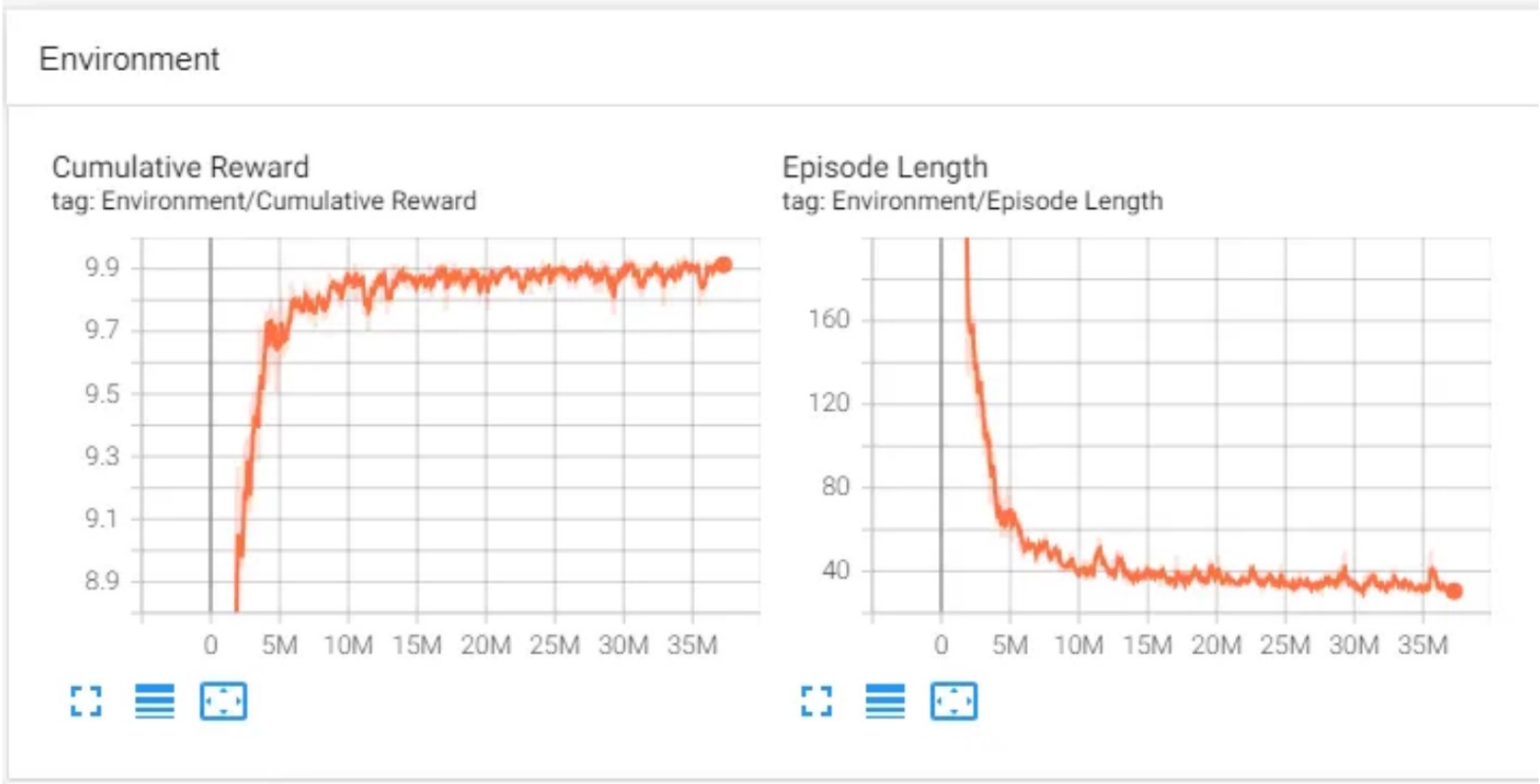
학습 과정 및 결과

- 학습 컴퓨팅 환경
  - CPU: Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz
  - RAM: 32GB
  - CPU: NVIDIA GeForce RTX 2070
- 학습 시간
  - 8시간
- 학습 파라미터
  - DQN vs PPO
  - Max_steps: 200,000-> 100,000,000

```
behaviors:
  PushBlock:
    trainer_type: ppo
    hyperparameters:
      batch_size: 128
      buffer_size: 2048
      learning_rate: 0.0003
      beta: 0.01
      epsilon: 0.2
      lambd: 0.95
      num_epoch: 3
      learning_rate_schedule: linear
    network_settings:
      normalize: false
      hidden_units: 256
      num_layers: 2
      vis_encode_type: simple
    reward_signals:
      extrinsic:
        gamma: 0.99
        strength: 1.0
    keep_checkpoints: 5
    max_steps: 100000000
    time_horizon: 64
    summary_freq: 60000
```

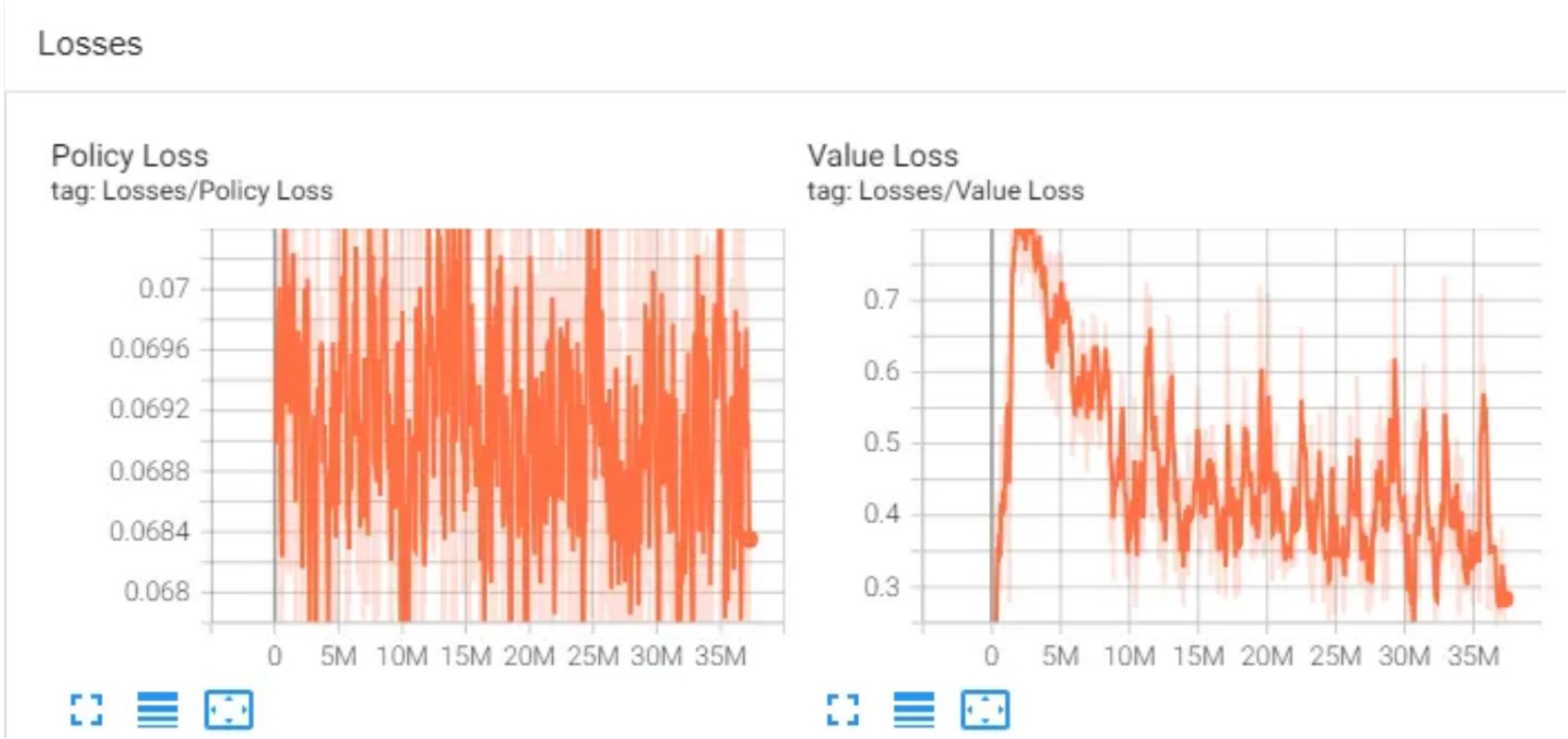
학습 과정 및 결과

- 학습 결과 시각화



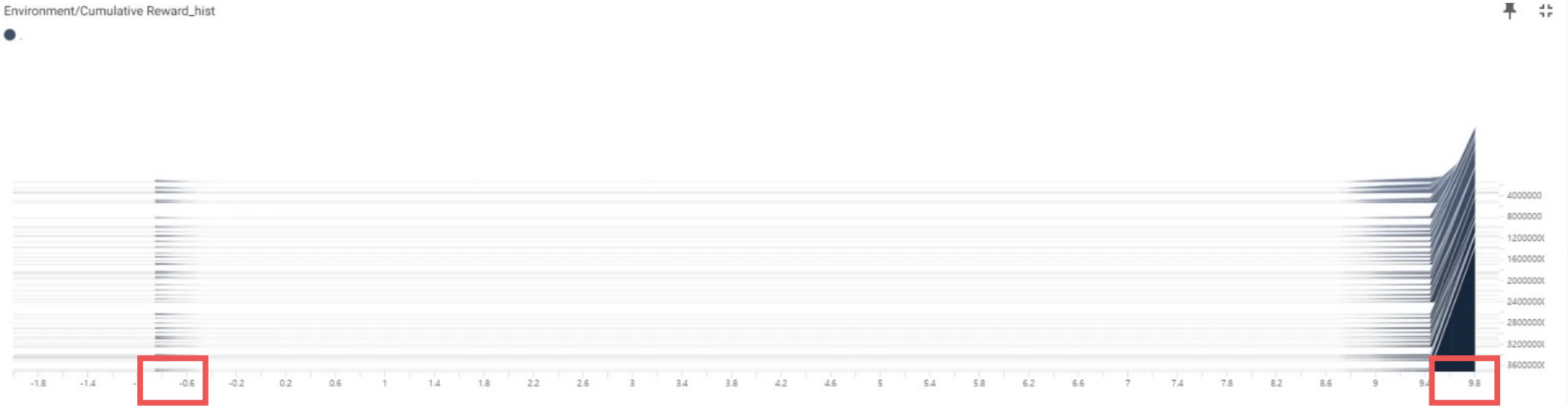
학습 과정 및 결과

- 학습 결과 시각화



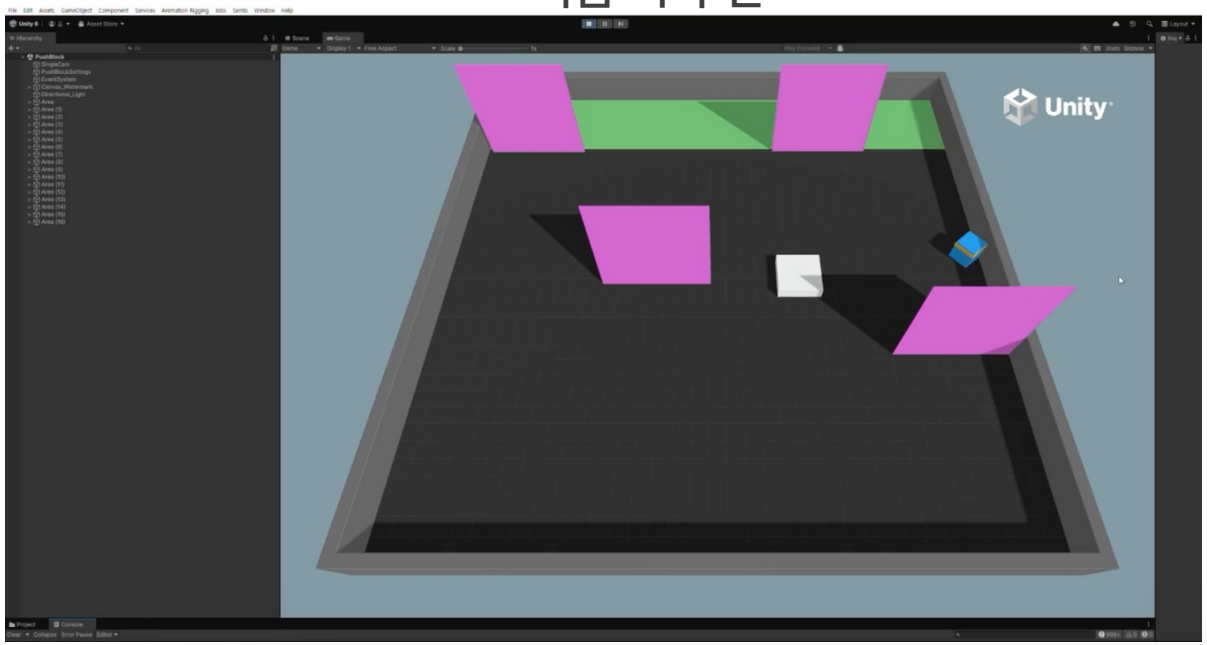
학습 과정 및 결과

- 학습 결과 시각화

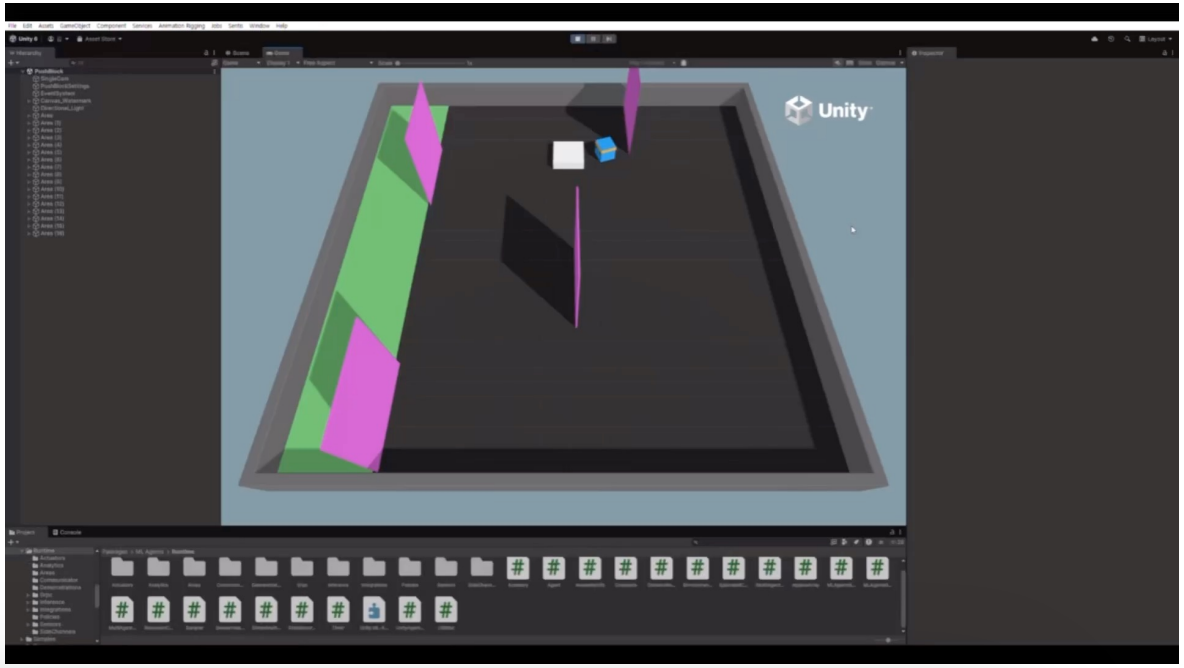


시연

학습 시작 전



학습 완료



그래프의 결과와 **에이전트 목표 달성 관찰을** 통하여 학습이 제대로 완료됨을 알 수 있다.