

# *ΕΡΓΑΣΙΑ ΜΑΘΗΜΑΤΟΣ*

---

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

Τμήμα Πληροφορικής



Μάθημα: «ΕΥΦΥΕΙΣ ΠΡΑΚΤΟΡΕΣ (80 εξ.)»

Π18101 – ΑΝΑΣΤΑΣΙΑ ΙΩΑΝΝΑ ΜΕΞΑ

[anastasia.mexa@yahoo.gr](mailto:anastasia.mexa@yahoo.gr)

Π18078 – ΑΘΑΝΑΣΙΑ ΚΟΜΜΑΤΙΔΟΥ

[nancygianna11@gmail.com](mailto:nancygianna11@gmail.com)

Π18123 – ΒΑΣΙΛΙΚΗ ΠΑΣΙΑ

[basia111@windowslive.com](mailto:basia111@windowslive.com)

Τηλέφωνο επικοινωνίας: 6970534855



## Εκφώνηση της άσκησης

### ML-Agents (Machine Learning Agents)

#### Περιγραφή

Οι πράκτορες Μηχανικής Μάθησης της Unity (Unity ML-Agents), αποτελούν μια ενδιαφέρουσα εναλλακτική εκπαίδευσης ευφυών Πρακτόρων μέσα στο περιβάλλον της Unity. Πραγματοποιώντας μια εργασία στα πλαίσια αυτά εξοικειώνεται κανείς με τις τεχνικές της μηχανικής μάθησης όπως αυτές υλοποιούνται στο πλαίσιο της πλατφόρμας της Unity

<https://unity.com/products/machine-learning-agents>

«Unity Machine Learning Agents, Train and embed intelligent agents by leveraging state-of-the-art deep learning technology»

Create an intelligent game experience Create realistic and complex AI environments to train models

Υπάρχουν πολλά έτοιμα παραδείγματα πολύ ενδιαφέροντων εφαρμογών ML-Agents τόσο μέσα στην Unity, όσο και σε σχετικά videos στο YouTube :

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

<https://www.youtube.com/watch?v=zPFU30tbyKs>

<https://towardsdatascience.com/an-introduction-to-unity-ml-agents-6238452fcf4c>

<https://gamedevacademy.org/unity-machine-learning-agents-tutorials/>

Υπάρχει επίσης μεγάλη ερευνητική δραστηριότητα που έχει δημοσιευθεί ως επιστημονικά άρθρα, ή μεταπτυχιακές διατριβές :

<https://arxiv.org/pdf/1809.02627.pdf>

<https://skemman.is/bitstream/1946/37111/1/An%20Evaluation%20of%20Unity%20MLAgents%20Toolkit%20for%20Learning%20Boss%20-%20MSc%20Thesis.pdf>

Έχουν ήδη σε αρχίσει να εμφανίζονται και σχετικά ξενόγλωσσα βιβλία :

«Learn Unity ML-Agents – Fundamentals of Unity Machine Learning: Incorporate new powerful ML algorithms such as Deep Reinforcement Learning for games», Transform games into environments using machine learning and Deep learning with Tensorflow, Keras, and Unity

[https://books.google.gr/books?hl=el&lr=&id=OMNiDwAAQBAJ&oi=fnd&pg=PP1&dq=Unity+Machine+Learning+Agents&ots=CjFPRA-zd-&sig=Os4n85rWeFL3Hugl5GXnpQsAKQ&redir\\_esc=y#v=onepage&q=Unity%20Machine%20Learning%20Agents&f=false](https://books.google.gr/books?hl=el&lr=&id=OMNiDwAAQBAJ&oi=fnd&pg=PP1&dq=Unity+Machine+Learning+Agents&ots=CjFPRA-zd-&sig=Os4n85rWeFL3Hugl5GXnpQsAKQ&redir_esc=y#v=onepage&q=Unity%20Machine%20Learning%20Agents&f=false)

Στην εργασία αυτή ζητείται να αξιοποιήσετε την σχετική τεχνολογία αναπτύσσοντας μια νέα εφαρμογή ή αξιοποιώντας μια υπάρχουσα και επεμβαίνοντας σε αυτή για να αναπτύξετε κάτι που ταιριάζει στα δικά σας ενδιαφέροντα.

#### Τεκμηρίωση

Η τεκμηρίωση της εφαρμογής θα περιλαμβάνει τα εξής:

1. Περιγραφή του προβλήματος
2. Περιγραφή της θεωρητικής βάσης της εφαρμογής, συμπεριλαμβανομένων των δομών δεδομένων και αναπαράστασης γνώσης που υιοθετήθηκαν, των αλγορίθμων και μεθοδολογιών που χρησιμοποιήθηκαν, καθώς και των προσαρμογών και μεταβολών που έγιναν στα παραπάνω προκειμένου να είναι δυνατή η εφαρμογή τους στο συγκεκριμένο πρόβλημα.
3. Περιγραφή σημαντικών σχεδιαστικών αποφάσεων και στοιχείων υλοποίησης
4. Ολοκληρωμένη περιγραφή μίας παραδειγματικής εκτέλεσης και των αποτελεσμάτων της



- 
- 5. Αναλυτική περιγραφή της διαδικασίας εγκατάστασης
  - 6. Περιγραφή πρόσθετων δυνατοτήτων της εφαρμογής, εάν υπάρχουν
  - 7. Αναλυτική περιγραφή της συμβολής του κάθε μέλους της ομάδας
  - 8. Αναλυτική περιγραφή ανοικτών θεμάτων, ανεπίλυτων προβλημάτων και πιθανοτήτων εμφάνισης σφαλμάτων κατά την εκτέλεση
- Είναι σημαντικό να υπάρχουν αναλυτικά και επεξηγημένα screenshots από την εκτέλεση της εφαρμογής

#### Παραδοτέα

- 1. Η εφαρμογή σε εκτελέσιμη μορφή (πρέπει να έχει project και να έχει γίνει build)
- 2. Πηγαίος κώδικας για το σύνολο της εφαρμογής
- 3. Τεκμηρίωση όπως περιγράφεται προηγουμένως
- 4. Τουλάχιστον ένα παράδειγμα το οποίο θα συνοδεύεται από αντίστοιχο αρχείο στατιστικών για συγκεκριμένα απαιτούμενα αγαθά ανά χωριό.
- 5. video + powerpoint



## ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

|     |  |    |
|-----|--|----|
| 1   | Περιγραφή του προβλήματος.....                                   | 5  |
| 2   | Αναλυτική περιγραφή της εφαρμογής .....                          | 5  |
| 2.1 | Agent.....   | 5  |
| 2.2 | Θεωρητικό υπόβαθρο για την διαδικασία της εκπαίδευσης.....       | 6  |
| 2.3 | Επεξήγηση κώδικα του agent .....                                 | 7  |
| 2.4 | Assets που χρησιμοποιήσαμε.....                                  | 11 |
| 2.5 | Μουσική.....   | 12 |
| 2.6 | Λογότυπα και εικονίδιο εφαρμογής.....                            | 12 |
| 3   | Εγκατάσταση του ML Agents .....                                  | 13 |
| 4   | Εκπαίδευση του μοντέλου .....                                    | 16 |
| 5   | Σχολιασμός αποτελεσμάτων εκπαίδευσης.....                        | 18 |
| 6   | Περιγραφή μίας παραδειγματικής εκτέλεσης.....                    | 19 |
| 7   | Αναλυτική περιγραφή της συμβολής του κάθε μέλους της ομάδας..... | 21 |
| 8   | Βιβλιογραφικές Πηγές.....  | 21 |



## 1 Περιγραφή του προβλήματος

Στην παρούσα εργασία αξιοποιήσαμε την τεχνολογία των πρακτόρων Μηχανικής Μάθησης της Unity (Unity ML-Agents), για την ανάπτυξη μιας εφαρμογής, στο πλαίσιο της οποίας οι ευφυείς πράκτορες εκπαιδεύονται με την χρήση της βιβλιοθήκης αυτής. Η εφαρμογή μας είναι βασισμένη στο ήδη υπάρχον εκπαιδευτικό παράδειγμα Worm, που υπάρχει ανεβασμένο στο [GitHub](#) της Unity ML – Agents βιβλιοθήκης, το οποίο και μας έχετε παραθέσει στην εκφώνηση της εργασίας. Η εργασία αναπτύχθηκε στην Unity 2020.3.20f1.

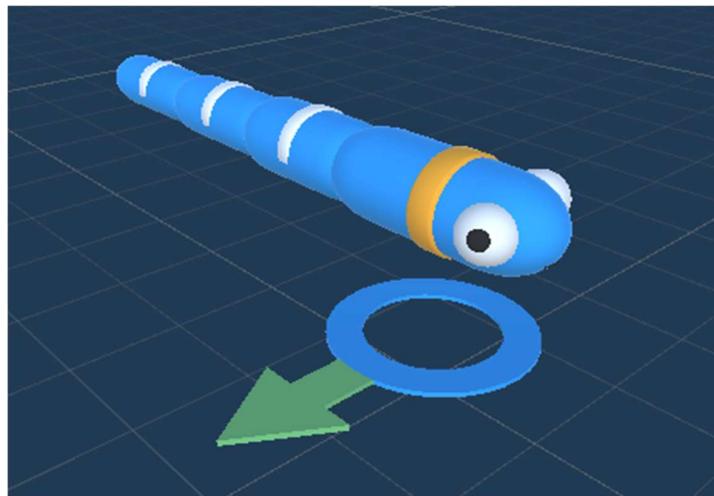
Το περιβάλλον που αναπτύξαμε προσομοιώνει έναν κήπο μέσα στον οποίο υπάρχει ο πράκτορας «σκουλήκι», σκοπός του οποίου είναι να κινήσει το σώμα του και να μετακινηθεί προς την κατεύθυνση του μήλου (στόχος).

## 2 Αναλυτική περιγραφή της εφαρμογής

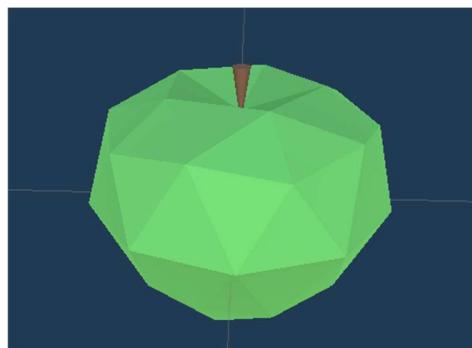
Παρακάτω θα αναλυθούν σε μορφή υπο-ενοτήτων τα βασικά χαρακτηριστικά/μέρη της εφαρμογής.

### 2.1 Agent

Ο πράκτορας (agent) που χρησιμοποιούμε απεικονίζεται παρακάτω και αποτελείται από τέσσερα τμήματα σώματος (1 κεφάλι και 3 body parts) και από τρεις αρθρώσεις joints.



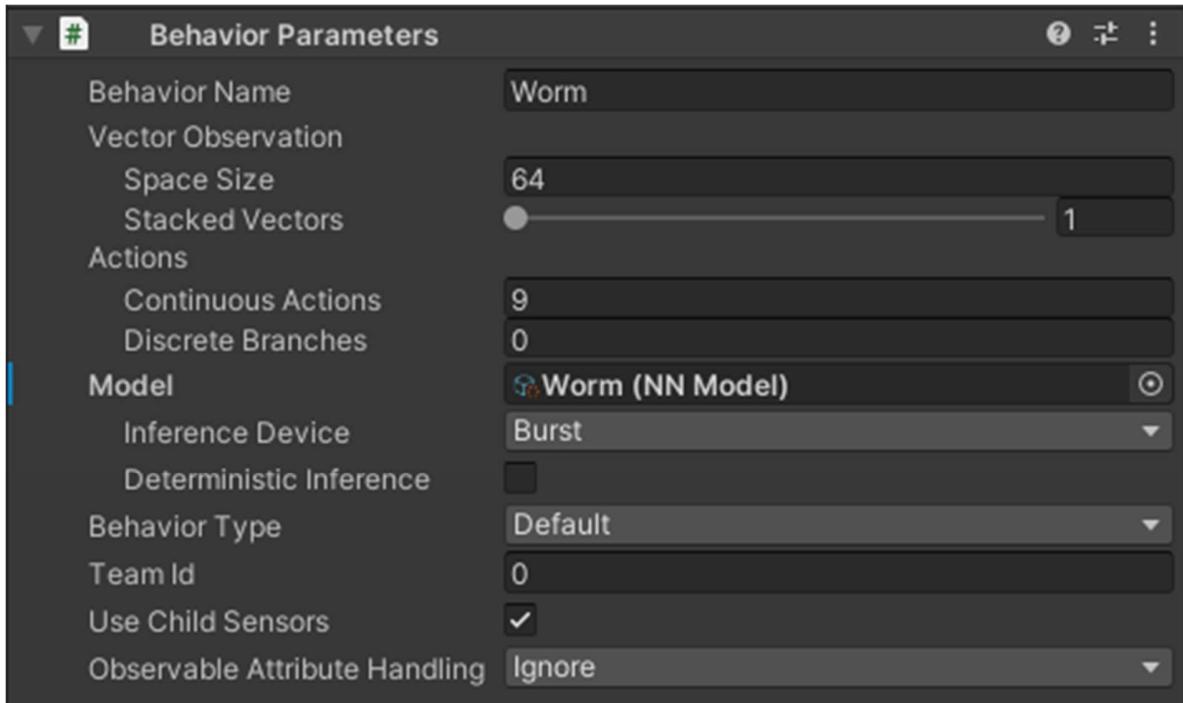
Σκοπός του πράκτορα είναι να κινήσει το σώμα του προς την κατεύθυνση του στόχου (μήλου). Το asset που χρησιμοποιήσαμε για την απεικόνιση του στόχου, φαίνεται παρακάτω.





Ο agent χαρακτηρίζεται από κάποιες παραμέτρους συμπεριφοράς (Behavior Parameters), οι οποίες καθορίζουν τον τρόπο με τον οποίο ο πράκτορας λαμβάνει τις αποφάσεις.

Όσον αφορά το Vector Observation space, έχουμε 64 μεταβλητές που αντιστοιχούν στη θέση, την περιστροφή, την ταχύτητα και τις γωνιακές ταχύτητες κάθε άκρου συν την επιτάχυνση και τη γωνιακή επιτάχυνση του σώματος. Ενώ για τα Actions έχουμε 9 continuous actions, που αντιστοιχούν σε περιστροφές των αρθρώσεων.



Το περιβάλλον μας περιέχει 10 πράκτορες με τις ίδιες παραμέτρους συμπεριφοράς, για την καλύτερη και γρηγορότερη εκπαίδευση του μοντέλου.

Λειτουργία ανταμοιβής πράκτορα (Agent Reward Function) εξαρτάται από τρεις παράγοντες:

1. Αν ο agent ακουμπήσει τον στόχο.
2. Αν η ταχύτητα με την οποία κινείται ο agent ταυτίζεται με την επιθυμητή.
3. Αν η κατεύθυνση του σώματος του agent ταυτίζεται με την κατεύθυνση του στόχου, δηλαδή αν κινείται προς τον στόχο.

## 2.2 Θεωρητικό υπόβαθρο για την διαδικασία της εκπαίδευσης

Με την χρήση των ML-Agents, είναι δυνατό να εκπαιδεύσουμε την συμπεριφορά του πράκτορα χρησιμοποιώντας μια ποικιλία μεθόδων. Για να επιτευχθεί η εκπαίδευση, πρέπει σε κάθε στιγμή του παιχνιδιού (περιβάλλοντος) να ορίζονται τρεις οντότητες. Συγκεκριμένα,

### 1. Observations (Παρατηρήσεις):

Περιέχει μόνο πληροφορίες που γνωρίζει ο πράκτορας και είναι συνήθως ένα υποσύνολο της κατάστασης περιβάλλοντος, δηλαδή τι αντιλαμβάνεται ο πράκτορας για το περιβάλλον. Οι παρατηρήσεις είναι αριθμητικές και συνεχείς και μετρούν τα χαρακτηριστικά του περιβάλλοντος από τη σκοπιά του πράκτορα.



## 2. Actions (Δράσεις):

Αφορούν ποιες ενέργειες/δράσεις μπορεί να κάνει ο πράκτορας. Είναι πάλι αριθμητικές και συνεχείς και προσδιορίζουν την κίνηση του πράκτορα στον χώρο.

## 3. Reward signals (Σήματα ανταμοιβής):

Είναι μια αριθμητική τιμή που υποδεικνύει πόσο καλά τα πάει ο πράκτορας. Σημειώνεται ότι το σήμα ανταμοιβής δεν χρειάζεται να παρέχεται κάθε στιγμή, αλλά μόνο όταν ο πράκτορας εκτελεί μια ενέργεια που είναι καλή ή κακή. Επομένως πρέπει να ρυθμιστεί με τρόπο που η μεγιστοποίηση της ανταμοιβής δημιουργεί την επιθυμητή βέλτιστη συμπεριφορά.

Αφού οριστούν αυτές οι τρεις οντότητες, μπορούμε να ξεκινήσει η διαδικασία της εκπαίδευσης. Αυτή επιτυγχάνεται με την προσομοίωση του περιβάλλοντος για πολλές δοκιμές όπου ο agent, με την πάροδο του χρόνου, μαθαίνει ποια είναι η βέλτιστη ενέργεια που πρέπει να κάνει για κάθε παρατήρηση που μετράει, μεγιστοποιώντας τη μελλοντική του ανταμοιβή.

### 2.3 Επεξήγηση κώδικα του agent

Το script που αφορά την λειτουργία του πράκτορα μας, ονομάζεται *WormAgent.cs* και είναι υπεύθυνο για την εκτέλεση των ενεργειών του πράκτορα και για την αντίστοιχη ανταμοιβή του.

Οι αρχικές εντολές του κώδικα αφορούν τον ορισμό βασικών μεταβλητών, όπως την μέγιστη ταχύτητα που μπορεί να λάβει ο πράκτορας, τον ορισμό του αντικειμένου στόχου, τον ορισμό των μελών του σώματος του πράκτορα, τον ορισμό της κατεύθυνσης που κινείται ο πράκτορας και την αρχική του τοποθεσία.

```
1  using UnityEngine;
2  using Unity.MLAgents;
3  using Unity.MLAgents.Actuators;
4  using Unity.MLAgents.Examples;
5  using Unity.MLAgents.Sensors;
6
7  [RequireComponent(typeof(JointDriveController))] // Required to set joint forces
8  public class WormAgent : Agent
9  {
10     const float m_MaxWalkingSpeed = 10; //The max walking speed
11
12     [Header("Target Prefabs")] public Transform TargetPrefab; //Target prefab to use in Dynamic envs
13     private Transform m_Target; //Target the agent will walk towards during training.
14
15     [Header("Body Parts")] public Transform bodySegment0;
16     public Transform bodySegment1;
17     public Transform bodySegment2;
18     public Transform bodySegment3;
19
20     //This will be used as a stabilized model space reference point for observations
21     //Because ragdolls can move erratically during training, using a stabilized reference transform improves learning
22     OrientationCubeController m_OrientationCube;
23
24     //The indicator graphic gameobject that points towards the target
25     DirectionIndicator m_DirectionIndicator;
26     JointDriveController m_JdController;
27
28     private Vector3 m_StartingPos; //starting position of the agent
29 }
```



Στην συνέχεια, η μέθοδος **Initialize()** αρχικοποιεί τις μεταβλητές που ορίστηκαν προηγουμένως και καλεί την μέθοδο **SpawnTarget()** η οποία τοποθετεί ένα αντικείμενο στόχο (μήλο) μέσα στο περιβάλλον.

```
30     public override void Initialize()
31     {
32         SpawnTarget(TargetPrefab, transform.position); //spawn target
33
34         m_StartingPos = bodySegment0.position;
35         m_OrientationCube = GetComponentInChildren<OrientationCubeController>();
36         m_DirectionIndicator = GetComponentInChildren<DirectionIndicator>();
37         m_JdController = GetComponent<JointDriveController>();
38
39         UpdateOrientationObjects();
40
41         //Setup each body part
42         m_JdController.SetupBodyPart(bodySegment0);
43         m_JdController.SetupBodyPart(bodySegment1);
44         m_JdController.SetupBodyPart(bodySegment2);
45         m_JdController.SetupBodyPart(bodySegment3);
46     }
47
48
49     /// <summary>
50     /// Spawns a target prefab at pos
51     /// </summary>
52     /// <param name="prefab"></param>
53     /// <param name="pos"></param>
54     void SpawnTarget(Transform prefab, Vector3 pos)
55     {
56         m_Target = Instantiate(prefab, pos, Quaternion.identity, transform.parent);
57     }
58 }
```

Για την δημιουργία ενός πράκτορα, κάνουμε extend τις παρακάτω μεθόδους της Agent κλάσης.

- **OnEpisodeBegin()**

Καλείται στην αρχή του επεισοδίου, συμπεριλαμβανομένης της αρχής της προσομοίωσης. Ο σκοπός της είναι να επαναφέρει τα μέλη του σώματος του πράκτορα στις αρχικές καταστάσεις.

```
59     /// <summary>
60     /// Loop over body parts and reset them to initial conditions.
61     /// </summary>
62     public override void OnEpisodeBegin()
63     {
64         foreach (var bodyPart in m_JdController.bodyPartsList)
65         {
66             bodyPart.Reset(bodyPart);
67         }
68
69         //Random start rotation to help generalize
70         bodySegment0.rotation = Quaternion.Euler(0, Random.Range(0.0f, 360.0f), 0);
71
72         UpdateOrientationObjects();
73     }
74 }
```



- **CollectObservations() και CollectObservationBodyPart()**

Καλούνται σε κάθε βήμα που ο πράκτορας πρέπει να πάρει μια απόφαση, κατά αυτόν τον τρόπο είναι δυνατή η συλλογή των παρατηρήσεων του πράκτορα για το περιβάλλον. Οι πληροφορίες που συλλέγονται αφορούν την θέση, την περιστροφή, την ταχύτητα και τις γωνιακές ταχύτητες κάθε άκρου συν την επιτάχυνση και τη γωνιακή επιτάχυνση του σώματος.

```
75  //<summary>
76  /// Add relevant information on each body part to observations.
77  //</summary>
78  public void CollectObservationBodyPart(BodyPart bp, VectorSensor sensor)
79  {
80      //GROUND CHECK
81      sensor.AddObservation(bp.groundContact.touchingGround ? 1 : 0); // Whether the bp touching the ground
82
83      //Get velocities in the context of our orientation cube's space
84      //Note: You can get these velocities in world space as well but it may not train as well.
85      sensor.AddObservation(m_OrientationCube.transform.InverseTransformDirection(bp.rb.velocity));
86      sensor.AddObservation(m_OrientationCube.transform.InverseTransformDirection(bp.rb.angularVelocity));
87
88
89      if (bp.rb.transform != bodySegment0)
90      {
91          //Get position relative to hips in the context of our orientation cube's space
92          sensor.AddObservation(
93              m_OrientationCube.transform.InverseTransformDirection(bp.rb.position - bodySegment0.position));
94          sensor.AddObservation(bp.rb.transform.localRotation);
95      }
96
97      if (bp.joint)
98          sensor.AddObservation(bp.currentStrength / m_3dController.maxJointForceLimit);
99  }
100
101  public override void CollectObservations(VectorSensor sensor)
102  {
103      RaycastHit hit;
104      float maxDist = 10;
105      if (Physics.Raycast(bodySegment0.position, Vector3.down, out hit, maxDist))
106      {
107          sensor.AddObservation(hit.distance / maxDist);
108      }
109      else
110          sensor.AddObservation(1);
111
112      var cubeForward = m_OrientationCube.transform.forward;
113      var velGoal = cubeForward * m_MaxWalkingSpeed;
114      sensor.AddObservation(m_OrientationCube.transform.InverseTransformDirection(velGoal));
115      sensor.AddObservationQuaternion.Angle(m_OrientationCube.transform.rotation,
116                                              m_3dController.bodyPartsDict[bodySegment0].rb.rotation) / 180);
117      sensor.AddObservationQuaternion.FromToRotation(bodySegment0.forward, cubeForward));
118
119      //Add pos of target relative to orientation cube
120      sensor.AddObservation(m_OrientationCube.transform.InverseTransformPoint(m_Target.transform.position));
121
122      foreach (var bodyPart in m_3dController.bodyPartsList)
123      {
124          collectObservationBodyPart(bodyPart, sensor);
125      }
126  }
```



- **TouchedTarget()**

Καλείται όταν ο πράκτορας αγγίζει το μήλο και αυξάνεται το reward.

```

128     /// <summary>
129     /// Agent touched the target
130     /// </summary>
131     public void TouchedTarget()
132     {
133         AddReward(1f);
134     }

```

- **OnActionReceived()**

Καλείται κάθε φορά που ο πράκτορας λαμβάνει μια ενέργεια που πρέπει να εκτελέσει. Ενημερώνονται καταλλήλως οι περιστροφές των αρθρώσεων του πράκτορα, ανάλογα με τις τιμές των συνεχών δράσεων.

```

136     public override void OnActionReceived(ActionBuffers actionBuffers)
137     {
138         // The dictionary with all the body parts in it are in the jdController
139         var bpDict = m_JdController.bodyPartsDict;
140
141         var i = -1;
142         var continuousActions = actionBuffers.ContinuousActions;
143         // Pick a new target joint rotation
144         bpDict[bodySegment0].SetJointTargetRotation(continuousActions[++i], continuousActions[++i], 0);
145         bpDict[bodySegment1].SetJointTargetRotation(continuousActions[++i], continuousActions[++i], 0);
146         bpDict[bodySegment2].SetJointTargetRotation(continuousActions[++i], continuousActions[++i], 0);
147
148         // Update joint strength
149         bpDict[bodySegment0].SetJointStrength(continuousActions[++i]);
150         bpDict[bodySegment1].SetJointStrength(continuousActions[++i]);
151         bpDict[bodySegment2].SetJointStrength(continuousActions[++i]);
152
153         //Reset if Worm fell through floor;
154         if (bodySegment0.position.y < m_StartingPos.y - 2)
155         {
156             EndEpisode();
157         }
158     }
159

```

- **FixedUpdate()**

Αποδίδει το κατάλληλο reward στον πράκτορα. Η ανταμοιβή ορίζεται από το γινόμενο των κανονικοποιημένων τιμών της ταύτισης της ταχύτητας, με την οποία κινείται ο agent σε σχέση με την επιθυμητή, και της ταύτισης της κατεύθυνσης του σώματος του agent σε σχέση με τον στόχο.

```

160     void FixedUpdate()
161     {
162         UpdateOrientationObjects();
163
164         var velReward =
165             GetMatchingVelocityReward(m_OrientationCube.transform.forward * m_MaxWalkingSpeed,
166             m_JdController.bodyPartsDict[bodySegment0].rb.velocity);
167
168         //Angle of the rotation delta between cube and body.
169         //This will range from (0, 180)
170         var rotAngle = Quaternion.Angle(m_OrientationCube.transform.rotation,
171             m_JdController.bodyPartsDict[bodySegment0].rb.rotation);
172
173         //The reward for facing the target
174         var facingRew = 0f;
175         //If we are within 30 degrees of facing the target
176         if (rotAngle < 30)
177         {
178             //Set normalized facingReward
179             //Facing the target perfectly yields a reward of 1
180             facingRew = 1 - (rotAngle / 180);
181         }
182
183         //Add the product of these two rewards
184         AddReward(velReward * facingRew);
185     }
186

```



- ***GetMatchingVelocityReward()***

Επιστρέφει την κανονικοποιημένη τιμή της διαφοράς της ταχύτητας με την οποία κινείται ο agent σε σχέση με την επιθυμητή.

```

187    /// <summary>
188    /// Normalized value of the difference in actual speed vs goal walking speed.
189    /// </summary>
190    public float GetMatchingVelocityReward(Vector3 velocityGoal, Vector3 actualVelocity)
191    {
192        //distance between our actual velocity and goal velocity
193        var velDeltaMagnitude = Mathf.Clamp(Vector3.Distance(actualVelocity, velocityGoal), 0, m_MaxWalkingSpeed);
194
195        //return the value on a declining sigmoid shaped curve that decays from 1 to 0
196        //This reward will approach 1 if it matches perfectly and approach zero as it deviates
197        return Mathf.Pow(1 - Mathf.Pow(velDeltaMagnitude / m_MaxWalkingSpeed, 2), 2);
198    }

```

- ***UpdateOrientationObjects()***

Ενημερώνει τις καταστάσεις των OrientationCube και DirectionIndicator αντικειμένων.

```

200    /// <summary>
201    /// Update OrientationCube and DirectionIndicator
202    /// </summary>
203    void UpdateOrientationObjects()
204    {
205        m_OrientationCube.UpdateOrientation(bodySegment0, m_Target);
206        if (m_DirectionIndicator)
207        {
208            m_DirectionIndicator.MatchOrientation(m_OrientationCube.transform);
209        }
210    }
211
212

```

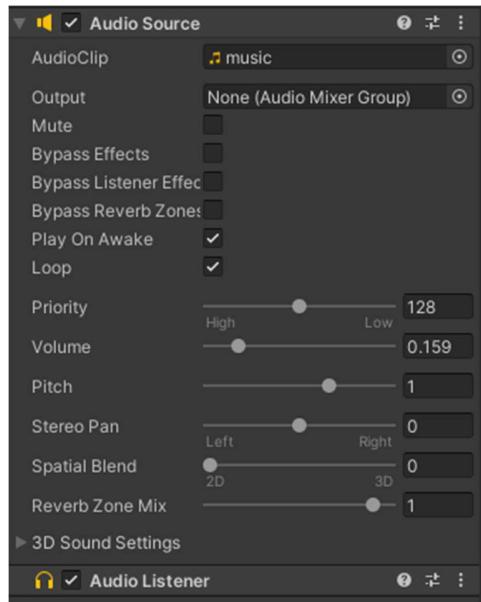
## 2.4 Assets που χρησιμοποιήσαμε

| Asset                        | Έκδοση | Link  |
|------------------------------|--------|---|
| Low Poly Tree Pack           | 1.3    | <a href="https://assetstore.unity.com/packages/3d/vegetation/trees/low-poly-tree-pack-57866">https://assetstore.unity.com/packages/3d/vegetation/trees/low-poly-tree-pack-57866</a>               |
| Low Poly Rock Pack           | 1.3    | <a href="https://assetstore.unity.com/packages/3d/environments/low-poly-rock-pack-57874">https://assetstore.unity.com/packages/3d/environments/low-poly-rock-pack-57874</a>                       |
| Low Poly Free Vegetation Kit | 1.1.1  | <a href="https://assetstore.unity.com/packages/3d/environments/low-poly-free-vegetation-kit-176906">https://assetstore.unity.com/packages/3d/environments/low-poly-free-vegetation-kit-176906</a> |
| Low Poly Fence Pack          | 1.1    | <a href="https://assetstore.unity.com/packages/3d/props/exterior/low-poly-fence-pack-61661">https://assetstore.unity.com/packages/3d/props/exterior/low-poly-fence-pack-61661</a>                 |
| Simple Foods                 | 1.0    | <a href="https://assetstore.unity.com/packages/3d/props/food/simple-foods-207032">https://assetstore.unity.com/packages/3d/props/food/simple-foods-207032</a>                                     |



## 2.5 Μουσική

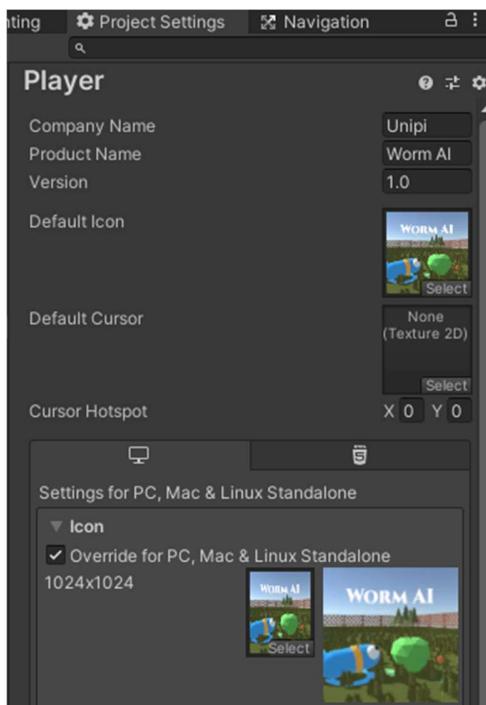
Για να προσθέσουμε μουσική υπόκρουση στην εφαρμογή μας, προσθέσαμε στην Main Camera, έναν Audio Listener και ένα Audio Source component, ορίζοντας στο πεδίο AudioClip το μουσικό κομμάτι που επιθυμούμε.



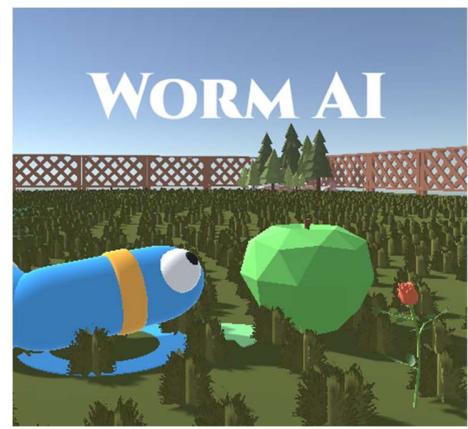
## 2.6 Λογότυπα και εικονίδιο εφαρμογής

Αρχικά, έχουμε δημιουργήσει έναν φάκελο στο project με όνομα Logos, στον οποίο εμπεριέχονται οι εικόνες μας.

Για να ορίσουμε εικονίδιο για την εφαρμογή (.exe) στο μενού Project Settings την Unity, δώσαμε την κατάλληλη εικόνα ως Default Icon όπως φαίνεται παρακάτω.



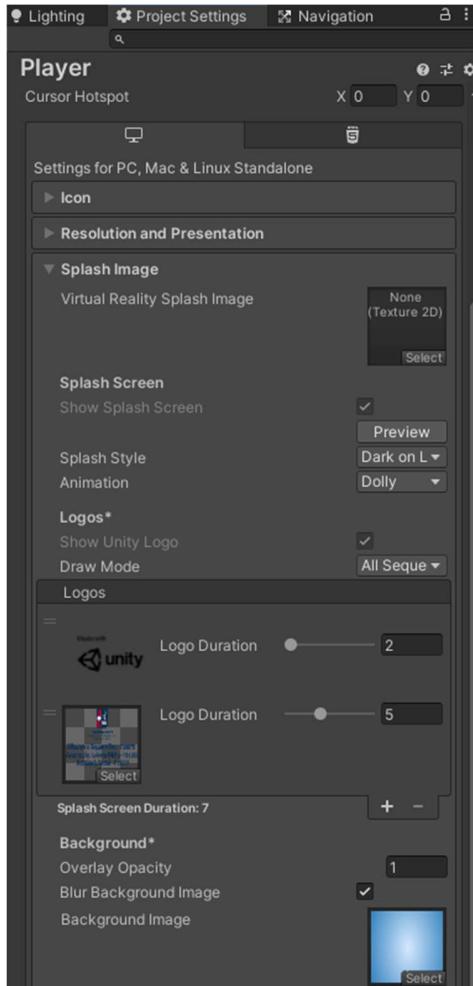
Το .exe αρχείο βρίσκεται στον φάκελο του project στο path "Project\Builds\Worm AI.exe"





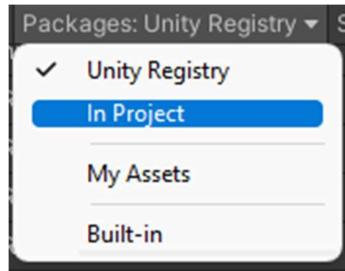
Επίσης, κατά την έναρξη της εφαρμογής εμφανίζονται τα στοιχεία μας μαζί με το λογότυπο της Unity.

Για να το επιτύχουμε αυτό, πάλι μέσω του μενού Project Settings την Unity, δώσαμε τις κατάλληλες εικόνες στο υπο-μενού Splash Image όπως φαίνεται παρακάτω.



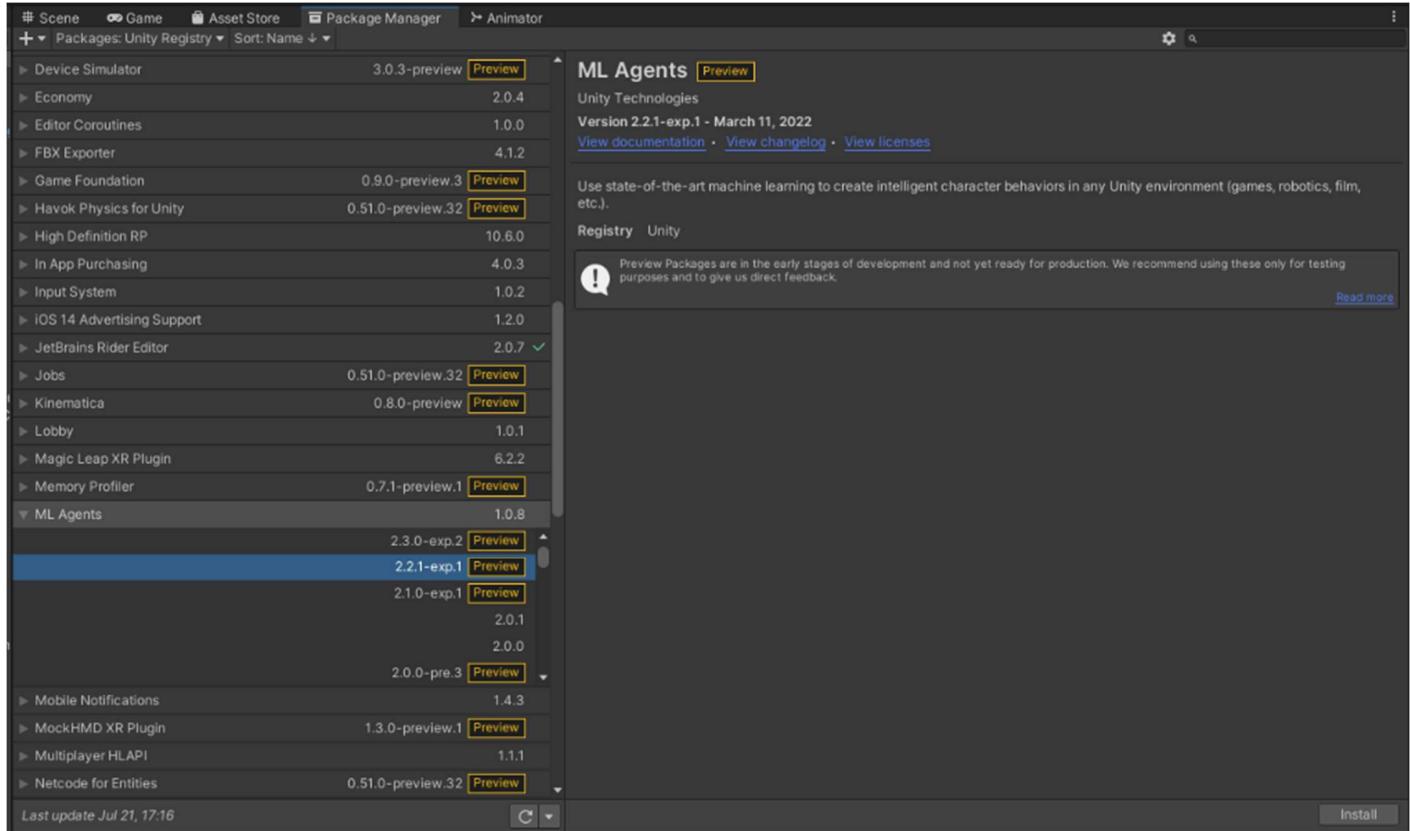
### 3 Εγκατάσταση του ML Agents

Αρχικά εγκαθιστούμε το **com.unity.ml-agents** Unity package, πηγαίνοντας στο Package Manager και επιλέγοντας από το drop down Packages μενού την επιλογή “Unity Registry”.



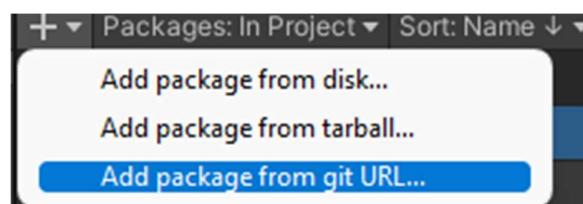


Στην συνέχεια επιλέγουμε την επιλογή ML Agents και συγκεκριμένα την έκδοση 2.2.1-exp.1, όπως φαίνεται παρακάτω.



Πατάμε το κουμπί Install.

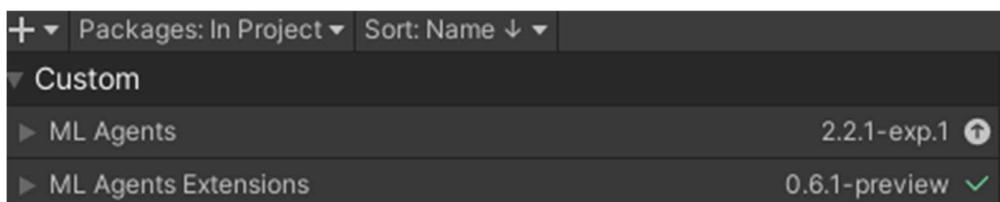
Στην συνέχεια πρέπει να κατεβάσουμε **com.unity.ml-agents.extensions** Unity package, πηγαίνοντας στο Package Manager και επιλέγοντας από το drop down + μενού την επιλογή "Add package from git URL"



Στο παράθυρο που θα εμφανιστεί πληκτρολογούμε την εντολή

git+https://github.com/Unity-Technologies/ml-agents.git?path=com.unity.ml-agents.extensions#release\_19

Μόλις ολοκληρώσουμε την εγκατάσταση των δύο βιβλιοθηκών πρέπει στο project μας να εμφανιστεί το εξής:





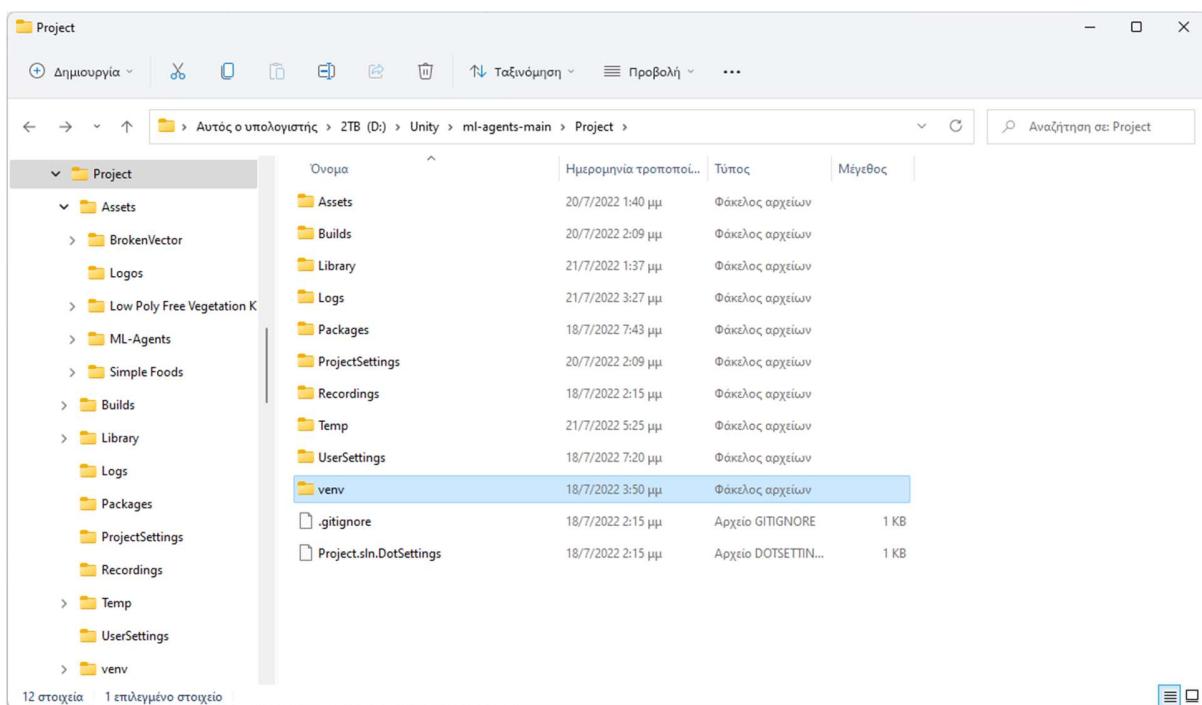
Πλέον, μπορούμε να δημιουργήσουμε στο φάκελο του project ένα virtual environment για την python. Ανοίγοντας ένα παράθυρο PowerShell, πηγαίνουμε στο κατάλληλο path που βρίσκεται το project της Unity και πληκτρολογούμε την παρακάτω εντολή.

```
Windows PowerShell (x86)
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\anast> Set-Location -Path D:\Unity\ml-agents-main\Project
PS D:\Unity\ml-agents-main\Project> py -m venv venv
```

Το αποτέλεσμα την οποίας είναι η δημιουργία ενός φακέλου με όνομα venv, με τα περιεχόμενα του virtual environment.



Για να ενεργοποιήσουμε το περιβάλλον πρέπει να εκτελέσουμε το αρχείο με όνομα Activate.ps1 και μόλις ενεργοποιηθεί, εκτελούμε την εντολή pip install mlagents, ώστε να κατεβάσουμε την βιβλιοθήκη.

```
Administrator: Windows PowerShell (x86)
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\WINDOWS\system32> Set-Location -Path D:\Unity\ml-agents-main\Project
PS D:\Unity\ml-agents-main\Project> cd venv\Scripts
PS D:\Unity\ml-agents-main\Project\venv\Scripts> .\Activate.ps1
(venv) PS D:\Unity\ml-agents-main\Project\venv\Scripts> pip install mlagents
Collecting mlagents
  Using cached https://files.pythonhosted.org/packages/9d/b9/48d2afc106541546607bb90338541417e8fcfa456fd7be0c4ec7a23849ca/mlagents-0.28.0-py3-none-any.whl
Collecting tensorboard>=1.15 (from mlagents)
  Using cached https://files.pythonhosted.org/packages/ee/0d/23812e6ce63b3d87c39bc9fee83e28c499052fa83fdddd7dae21a6d620/tensorboard-2.9.1-py3-none-any.whl
Collecting numpy<2.0,>=1.13.3 (from mlagents)
  Using cached https://files.pythonhosted.org/packages/97/9f/da37cc4a188a1d5d203d65ab28d6504e17594b5342e0c1dc5610ee6f4535/numpy-1.21.6-cp37-cp37m-win_amd64.whl
Collecting attrs>=19.3.0 (from mlagents)
```



## 4 Εκπαίδευση του μοντέλου

Μόλις ολοκληρωθεί η εγκατάσταση όλων των παραπάνω, μπορούμε να ξεκινήσουμε την διαδικασία εκπαίδευσης του μοντέλου. Για να το πραγματοποιήσουμε αυτό, μέσω του PowerShell μεταφερόμαστε μέσα στο virtual environment που έχουμε δημιουργήσει και πληκτρολογούμε την παρακάτω εντολή.

```
Administrator: Windows PowerShell (x86)
(venv) PS D:\Unity\ml-agents-main> mlagents-learn .\config\ppo\Worm.yaml --run-id=FirstWormTraining

Version information:
ml-agents: 0.28.0,
ml-agents-envs: 0.28.0,
Communicator API: 1.5.0,
PyTorch: 1.12.0+cpu
[INFO] Listening on port 5004. Start training by pressing the Play button in the Unity Editor.
[INFO] Connected to Unity environment with package version 2.2.1-exp.1 and communication version 1.5.0
[INFO] Connected new brain: Worm?team=0
[INFO] Hyperparameters for behavior name Worm:
  trainer_type: ppo
```

Η εντολή ξεκινάει την διαδικασία της εκπαίδευσης με παραμέτρους οι οποίες ορίζονται μέσα στο αρχείο Worm.yaml και ως id του training ορίζουμε το “FirstWormTraining”.

Τα περιεχόμενα του Worm.yaml αρχείου:

```
Administrator: Windows PowerShell (x86)
[INFO] Connected new brain: Worm?team=0
[INFO] Hyperparameters for behavior name Worm:
  trainer_type: ppo
  hyperparameters:
    batch_size: 2024
    buffer_size: 20240
    learning_rate: 0.0003
    beta: 0.005
    epsilon: 0.2
    lambda: 0.95
    num_epoch: 3
    learning_rate_schedule: linear
    beta_schedule: linear
    epsilon_schedule: linear
  network_settings:
    normalize: True
    hidden_units: 512
    num_layers: 3
    vis_encode_type: simple
    memory: None
    goal_conditioning_type: hyper
    deterministic: False
  reward_signals:
    extrinsic:
      gamma: 0.995
      strength: 1.0
    network_settings:
      normalize: False
      hidden_units: 128
      num_layers: 2
      vis_encode_type: simple
      memory: None
      goal_conditioning_type: hyper
      deterministic: False
    init_path: None
    keep_checkpoints: 5
    checkpoint_interval: 500000
    max_steps: 7000000
    time_horizon: 1000
    summary_freq: 30000
    threaded: False
    self_play: None
```



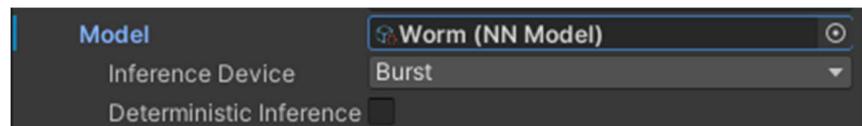
Αν εμφανιστεί το λογότυπο της Unity έχουν πάει όλα σωστά και πλέον μπορούμε να πατήσουμε το κουμπί run μέσα από τον editor της Unity για την έναρξη της εκπαίδευσης.

Κατά την διάρκεια της εκπαίδευσης εμφανίζονται στην κονσόλα πληροφορίες για την εξέλιξή της. Συγκεκριμένα, τα βήματα, ο χρόνος που έχει περάσει από την έναρξη της διαδικασίας, το Mean Reward και το Reward.

```
Administrator: Windows PowerShell (x86)
please. Consider `x.mT` to transpose batches of matrices or `x.permute(*torch.arange(x.ndim - 1, -1, -1))` to reverse the
en\src\ATen\native\TensorShape.cpp:2985.)
    return (tensor.T * masks).sum() / torch.clamp(
[INFO] Worm. Step: 60000. Time Elapsed: 68.732 s. Mean Reward: 0.301. Std of Reward: 0.402. Training.
[INFO] Worm. Step: 90000. Time Elapsed: 103.414 s. Mean Reward: 0.419. Std of Reward: 0.549. Training.
[INFO] Worm. Step: 120000. Time Elapsed: 138.771 s. Mean Reward: 0.289. Std of Reward: 0.358. Training.
[INFO] Worm. Step: 150000. Time Elapsed: 174.331 s. Mean Reward: 0.469. Std of Reward: 0.477. Training.
[INFO] Worm. Step: 180000. Time Elapsed: 209.602 s. Mean Reward: 0.567. Std of Reward: 0.798. Training.
[INFO] Worm. Step: 210000. Time Elapsed: 245.935 s. Mean Reward: 0.820. Std of Reward: 0.769. Training.
[INFO] Worm. Step: 240000. Time Elapsed: 281.667 s. Mean Reward: 1.258. Std of Reward: 1.062. Training.
[INFO] Worm. Step: 270000. Time Elapsed: 316.752 s. Mean Reward: 1.304. Std of Reward: 1.114. Training.
[INFO] Worm. Step: 300000. Time Elapsed: 352.527 s. Mean Reward: 2.368. Std of Reward: 1.626. Training.
[INFO] Worm. Step: 330000. Time Elapsed: 387.733 s. Mean Reward: 2.376. Std of Reward: 1.945. Training.
[INFO] Worm. Step: 360000. Time Elapsed: 422.778 s. Mean Reward: 3.717. Std of Reward: 2.757. Training.
[INFO] Worm. Step: 390000. Time Elapsed: 458.376 s. Mean Reward: 6.062. Std of Reward: 4.028. Training.
[INFO] Worm. Step: 420000. Time Elapsed: 493.358 s. Mean Reward: 6.456. Std of Reward: 3.514. Training.
[INFO] Worm. Step: 450000. Time Elapsed: 529.046 s. Mean Reward: 8.463. Std of Reward: 4.394. Training.
[INFO] Worm. Step: 480000. Time Elapsed: 563.378 s. Mean Reward: 12.729. Std of Reward: 4.071. Training.
[INFO] Exported results\FirstWormTraining\Worm\Worm-499000.onnx
[INFO] Worm. Step: 510000. Time Elapsed: 598.762 s. Mean Reward: 15.540. Std of Reward: 5.742. Training.
[INFO] Worm. Step: 540000. Time Elapsed: 632.905 s. Mean Reward: 20.451. Std of Reward: 5.411. Training.
[INFO] Worm. Step: 570000. Time Elapsed: 667.170 s. Mean Reward: 25.665. Std of Reward: 6.803. Training.
[INFO] Worm. Step: 600000. Time Elapsed: 701.054 s. Mean Reward: 30.820. Std of Reward: 5.661. Training.
[INFO] Worm. Step: 630000. Time Elapsed: 736.039 s. Mean Reward: 33.223. Std of Reward: 6.984. Training.
[INFO] Worm. Step: 660000. Time Elapsed: 770.286 s. Mean Reward: 37.909. Std of Reward: 10.993. Training.
[INFO] Worm. Step: 690000. Time Elapsed: 825.668 s. Mean Reward: 41.968. Std of Reward: 11.689. Training.
[INFO] Worm. Step: 720000. Time Elapsed: 860.425 s. Mean Reward: 47.888. Std of Reward: 10.091. Training.
[INFO] Worm. Step: 750000. Time Elapsed: 895.262 s. Mean Reward: 54.675. Std of Reward: 10.735. Training.
[INFO] Worm. Step: 780000. Time Elapsed: 930.752 s. Mean Reward: 60.720. Std of Reward: 9.868. Training.
[INFO] Worm. Step: 810000. Time Elapsed: 964.962 s. Mean Reward: 69.404. Std of Reward: 8.170. Training.
[INFO] Worm. Step: 840000. Time Elapsed: 998.867 s. Mean Reward: 71.827. Std of Reward: 15.088. Training.
[INFO] Worm. Step: 870000. Time Elapsed: 1032.672 s. Mean Reward: 80.383. Std of Reward: 12.804. Training.
[INFO] Worm. Step: 900000. Time Elapsed: 1066.896 s. Mean Reward: 93.349. Std of Reward: 10.253. Training.
[INFO] Worm. Step: 930000. Time Elapsed: 1100.711 s. Mean Reward: 95.794. Std of Reward: 18.738. Training.
[INFO] Worm. Step: 960000. Time Elapsed: 1134.824 s. Mean Reward: 103.167. Std of Reward: 13.162. Training.
[INFO] Worm. Step: 990000. Time Elapsed: 1168.558 s. Mean Reward: 103.331. Std of Reward: 20.318. Training.
[INFO] Exported results\FirstWormTraining\Worm\Worm-999000.onnx
[INFO] Worm. Step: 1020000. Time Elapsed: 1202.824 s. Mean Reward: 115.617. Std of Reward: 14.125. Training.
[INFO] Worm. Step: 1050000. Time Elapsed: 1236.476 s. Mean Reward: 112.461. Std of Reward: 19.082. Training.
[INFO] Worm. Step: 1080000. Time Elapsed: 1272.083 s. Mean Reward: 124.315. Std of Reward: 15.259. Training.
[INFO] Worm. Step: 1110000. Time Elapsed: 1307.168 s. Mean Reward: 125.476. Std of Reward: 14.198. Training.
[INFO] Worm. Step: 1140000. Time Elapsed: 1342.455 s. Mean Reward: 128.442. Std of Reward: 17.875. Training.
[INFO] Worm. Step: 1170000. Time Elapsed: 1377.288 s. Mean Reward: 135.747. Std of Reward: 16.565. Training.
[INFO] Worm. Step: 1200000. Time Elapsed: 1411.256 s. Mean Reward: 132.107. Std of Reward: 25.034. Training.
[INFO] Worm. Step: 1230000. Time Elapsed: 1445.088 s. Mean Reward: 150.043. Std of Reward: 16.021. Training.
[INFO] Worm. Step: 1260000. Time Elapsed: 1478.696 s. Mean Reward: 150.121. Std of Reward: 19.340. Training.
[INFO] Worm. Step: 1290000. Time Elapsed: 1513.086 s. Mean Reward: 162.733. Std of Reward: 16.518. Training.
[INFO] Worm. Step: 1320000. Time Elapsed: 1547.063 s. Mean Reward: 170.442. Std of Reward: 20.128. Training.
[INFO] Worm. Step: 1350000. Time Elapsed: 1581.106 s. Mean Reward: 176.171. Std of Reward: 21.883. Training.
[INFO] Worm. Step: 1380000. Time Elapsed: 1617.592 s. Mean Reward: 175.347. Std of Reward: 21.567. Training.
[INFO] Worm. Step: 1410000. Time Elapsed: 1652.531 s. Mean Reward: 184.973. Std of Reward: 23.433. Training.
[INFO] Learning was interrupted. Please wait while the graph is generated.
[INFO] Exported results\FirstWormTraining\Worm\Worm-1410000.onnx
[INFO] Copied results\FirstWormTraining\Worm\Worm-1410000.onnx to results\FirstWormTraining\Worm.onnx.
(venv) PS D:\Unity\ml-agents-main>
```

Εμείς διακόψαμε την διαδικασία της εκπαίδευσης μετά από 27 λεπτά, όπου το Mean Reward είχε φτάσει στο 184.973. Προφανώς, είναι λογικό ότι αν αφήναμε περισσότερη ώρα το μοντέλο να εκπαιδευτεί, θα είχαμε καλύτερα αποτελέσματα.

Το αποτέλεσμα της εκπαίδευσης είναι το αρχείο Worm.onnx, το οποίο και δίνουμε στον agent μέσω του Unity editor.



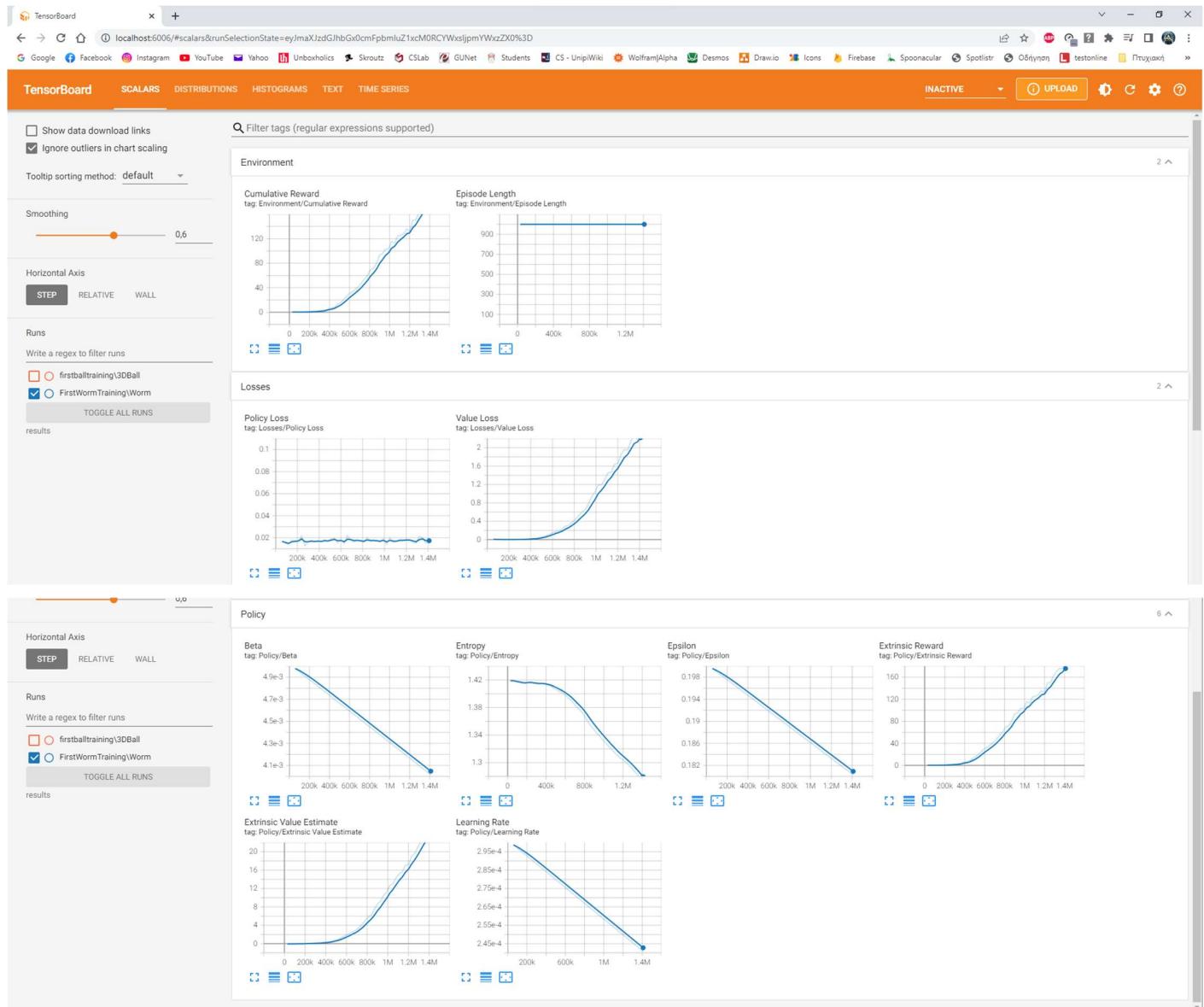


## 5 Σχολιασμός αποτελεσμάτων εκπαίδευσης

Για την καλύτερη κατανόηση των αποτελεσμάτων της εκπαίδευσης του μοντέλου χρησιμοποιήσαμε το TensorBoard για την οπτικοποίησή τους. Για να έχουμε πρόσβαση στα αποτελέσματα αυτά, εκτελέσαμε την παρακάτω εντολή στο PowerShell.

```
(venv) PS D:\Unity\ml-agents-main> tensorboard --logdir results
TensorFlow installation not found - running with reduced feature set.
Serving TensorBoard on localhost; to expose to the network, use a proxy or pass --bind_all
TensorBoard 2.9.1 at http://localhost:6006/ (Press CTRL+C to quit)
```

Συνεπώς, αν συνδεθούμε στο link που μας παραπέμπει το PowerShell έχουμε τα εξής αποτελέσματα/γραφήματα.



Παρακάτω αναλύεται το περιεχόμενο των γραφημάτων



### **Environment:**

- **Cumulative Reward** είναι η μέση αθροιστική ανταμοιβή για κάθε επεισόδιο σε όλους τους πράκτορες. Το γεγονός ότι αυξάνεται, υποδεικνύει ότι η εκπαίδευσή μας ήταν επιτυχημένη.
- **Episode Length** είναι η μέση διάρκεια κάθε επεισοδίου στο περιβάλλον για όλους τους πράκτορες. Προφανώς σε εμάς είναι σταθερό, αφού δεν αλλάζουμε την διάρκεια των επεισοδίων.

### **Losses:**

- **Policy Loss** σχετίζεται με το πόσο αλλάζει η διαδικασία λήψης αποφάσεων για ενέργειες. Το γεγονός ότι παραμένει σχετικά σταθερή σε ένα πολύ μικρό μέγεθος (της τάξης 0.02), υποδεικνύει ότι η εκπαίδευσή μας ήταν επιτυχημένη.
- **Value Loss** σχετίζεται με το πόσο καλά το μοντέλο είναι σε θέση να προβλέψει την τιμή κάθε κατάστασης. Το γεγονός ότι αυξάνεται, υποδεικνύει ότι η εκπαίδευσή μας ήταν επιτυχημένη.

### **Policy:**

- **Entropy** αφορά το όσο τυχαίες είναι οι αποφάσεις του μοντέλου. Θα πρέπει να μειώνεται αργά κατά τη διάρκεια μιας επιτυχημένης διαδικασίας εκπαίδευσης. Εάν μειωθεί πολύ γρήγορα, η βήτα παράμετρος θα πρέπει να αυξηθεί.
- **Extrinsic Reward** αυτό αντιστοιχεί στη μέση αθροιστική ανταμοιβή που λαμβάνεται από το περιβάλλον ανά επεισόδιο.
- **Extrinsic Value Estimate** η εκτίμηση μέσης αξίας για όλες τις καταστάσεις του πράκτορα. Θα πρέπει να αυξηθεί κατά τη διάρκεια μιας επιτυχημένης προπόνησης.
- **Learning Rate** πόσο μεγάλο βήμα κάνει ο αλγόριθμος εκπαίδευσης καθώς αναζητά τη βέλτιστη πολιτική. Θα πρέπει να μειώνεται με την πάροδο του χρόνου.

## **6 Περιγραφή μίας παραδειγματικής εκτέλεσης**

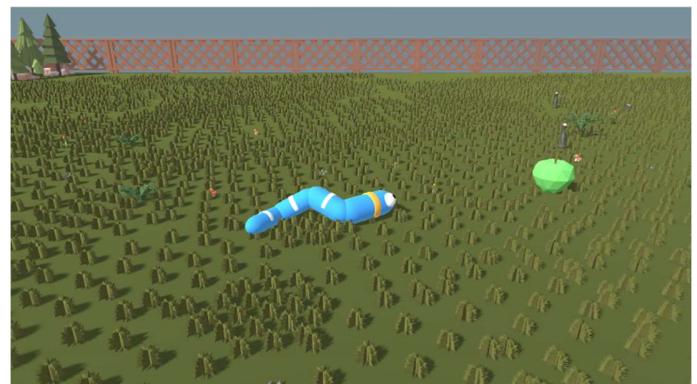
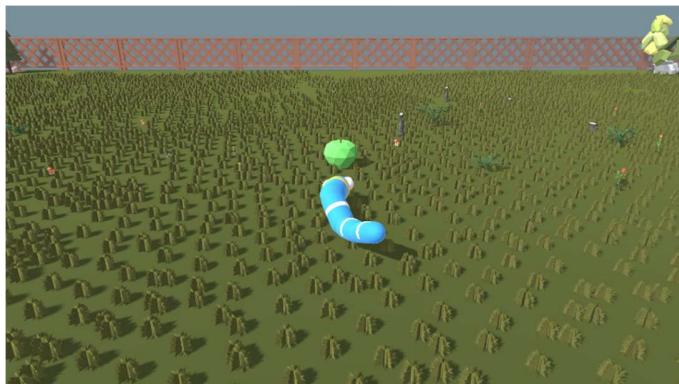
Κατά την έναρξη της εφαρμογής, εμφανίζεται πρώτα το λογότυπο της Unity και τα στοιχεία της ομάδας.





Στην συνέχεια αυτόματα μεταφερόμαστε στο εικονικό περιβάλλον που έχουμε δημιουργήσει, στο οποίο ο ήδη εκπαιδευμένος agent προσπαθεί να κινηθεί και να ακουμπήσει το μήλο (στόχος), το οποίο τοποθετείται κάθε φορά σε καινούργια θέση μέσα στον χώρο.

Ακολουθούν μερικά ενδεικτικά screenshots που απεικονίζουν την συμπεριφορά του πράκτορα, ο οποίος κινείται προς το μήλο, ανεξαρτήτως της θέσης του.



Να σημειωθεί ότι η κάμερα ακολουθεί πάντα τον agent, δηλαδή είναι «κλειδωμένη» σε αυτόν. Επιπλέον, τα αντικείμενα που διακοσμούν το έδαφος, δεν έχουν mesh collider ώστε να είναι πιο εύκολες οι κινήσεις που πρέπει να εκτελέσει ο agent για να φτάσει στον στόχο του, χωρίς να τον εμποδίζουν.

Μόλις ο agent ακουμπήσει το μήλο, τότε αυτό εξαφανίζεται από την προηγούμενη θέση του και τοποθετείται τυχαία κάπου αλλού στον χώρο. Για αυτό το λόγο, ο agent πλέον περιστρέφεται προς την σωστή κατεύθυνση με σκοπό να «στοχεύει» τον νέο στόχο και ξεκινά να κινείται προς αυτόν, μέχρι να τον φτάσει και πάλι και να συνεχιστεί η ίδια διαδικασία.





## 7 Αναλυτική περιγραφή της συμβολής του κάθε μέλους της ομάδας

Η εργασία έχει υλοποιηθεί εξ ολοκλήρου συνεργατικά, κατά την διάρκεια βιντεοκλήσεων και διαμοιρασμού της οθόνης. Αυτό συνέβη διότι, μόνο ένας από τους υπολογιστές της ομάδας διαθέτει την κατάλληλη υπολογιστική ισχύ για να μπορεί να υποστηρίξει τόσο τα εργαλεία της Unity, όσο και την διαδικασία εκπαίδευσης του πράκτορα. Επίσης, η ομάδα μας αποτελείται από φοιτήτριες διαφορετικών κατευθύνσεων. Συγκεκριμένα, οι **ΑΝΑΣΤΑΣΙΑ ΙΩΑΝΝΑ ΜΕΞΑ – Π18101** και **ΒΑΣΙΛΙΚΗ ΠΑΣΙΑ – Π18123** είμαστε από την κατεύθυνση **ΠΣΥ**, ενώ η **ΑΘΑΝΑΣΙΑ ΚΟΜΜΑΤΙΔΟΥ – Π18078** είναι από την κατεύθυνση **ΤΛΕΣ**. Όπως είναι λογικό, μόνο ένα μέλος της ομάδας είχε προηγούμενη εμπειρία για το λογισμικό της Unity και γενικότερα για τον σχεδιασμό εικονικών περιβαλλόντων/παιχνιδιών. Ωστόσο, τα υπόλουτα μέλη της ομάδας, θέλαμε να αποκτήσουμε και εμείς γνώσεις και εμπειρία πάνω στο αντικείμενο και να ανακαλύψουμε τον κόσμο της Unity και των γραφικών γενικότερα, προτού αποφοιτήσουμε. Συμπεραίνοντας, για τους παραπάνω λόγους η ομάδα μας είναι τριών ατόμων και η εργασία πραγματοποιήθηκε πλήρως ομαδικά και συνεργατικά.

## 8 Βιβλιογραφικές Πηγές

1. Όλο το περιεχόμενο των διαλέξεων του μαθήματος και τα αρχεία που μας έχετε ανεβάσει στο Teams
2. <https://github.com/Unity-Technologies/ml-agents>
3. <https://www.youtube.com/watch?v=zPFU30tbyKs>
4. <https://www.youtube.com/watch?v=GhS6-vvhOy8>
5. <https://towardsdatascience.com/an-introduction-to-unity-ml-agents-6238452fcf4c>
6. <https://gamedevacademy.org/unity-machine-learning-agents-tutorials/>
7. <https://arxiv.org/pdf/1809.02627.pdf>
8. <https://skemman.is/bitstream/1946/37111/1/An%20Evaluation%20of%20Unity%20MLAgents%20Toolkit%20for%20Learning%20Boss%20-%20MSc%20Thesis.pdf>
9. [https://books.google.gr/books?hl=el&lr=&id=OMNiDwAAQBAJ&oi=fnd&pg=PP1&dq=Unity+Machine+Learning+Agents&ots=CjFPRA-zd&sig=Os4n85rWeFL3HUGl5GXnpQsAKQ&redir\\_esc=y#v=onepage&q=Unity%20Machine%20Learning%20Agent&f=false](https://books.google.gr/books?hl=el&lr=&id=OMNiDwAAQBAJ&oi=fnd&pg=PP1&dq=Unity+Machine+Learning+Agents&ots=CjFPRA-zd&sig=Os4n85rWeFL3HUGl5GXnpQsAKQ&redir_esc=y#v=onepage&q=Unity%20Machine%20Learning%20Agent&f=false)