February 20, 2020

Reinforcement Learning Penguins (Part 3/4) | Unity ML-Agents

SCENE CONSTRUCTION

In this tutorial, you will create a Scene that includes a penguin, a baby penguin, several fish, and a small, enclosed area where the penguin can move around, collect fish, and feed the baby. You will use the C# scripts written in the previous tutorial to enable ML-Agents to learn and make intelligent decisions.

CREATE THE BABY PENGUIN PREFAB

In this section, you'll create a Prefab for the baby penguin.

- Double-click on the BabyPenguin Prefab in the *Prefabs* folder to open it.
- Add a Rigidbody component.
- Set the Rigidbody Constraints to lock position in X and Z and to lock rotation in X, Y, and Z so that the baby doesn't accidentally get knocked around, but can still be affected by gravity
- Add a Sphere Collider.
- Set the Sphere Collider Center to (0, 0.24, 0).
- Set the Sphere Collider Radius to 0.25.
- Change the tag to "baby" (you will need to create a new tag first)

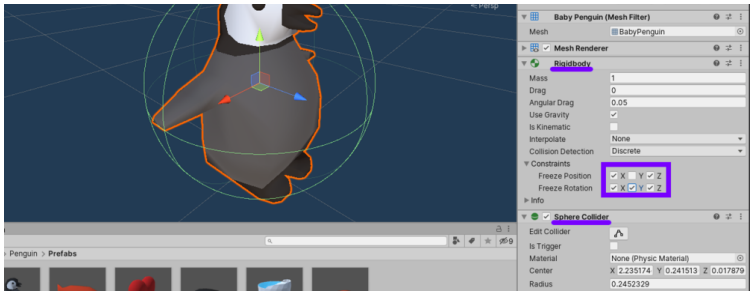

Figure 01: The BabyPenguin, with all changes in the Inspector highlighted

CREATE THE FISH PREFAB

In this section, you'll create a Prefab for the fish that the penguin agent needs to catch.

- Double-click on the Fish Prefab in the *Prefabs* folder to open it.
- Add a Rigidbody component.
- Set the Rigidbody Constraints to lock rotation in X and Z so that it doesn't flip over
- Add a Capsule Collider that encapsulates the fish.
- Set the Capsule Collider Center to (0, 0, -0.08).
- Set the Capsule Collider Radius to 0.15.
- Set the Capsule Collider Height to 0.66.
- Set the Capsule Collider Direction to Z-axis.
- Change the tag to "fish" (you will need to create a new tag first).
- Attach the Fish script.

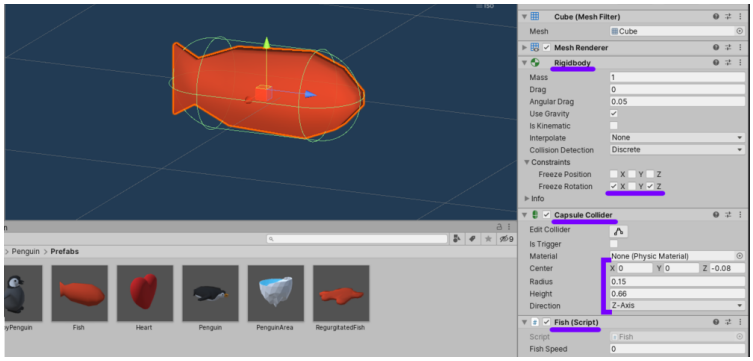

Figure 02: The fish, with all changes in the Inspector highlighted

CREATE THE PENGUIN AGENT

In this section, you'll create a penguin agent that can observe its environment and take actions using deep learning. We won't train it yet. We're just setting it up for now.

- Double-click on the Penguin Prefab in the *Prefabs* folder to open it.
- Add a Rigidbody component.
- Set the Rigidbody Constraints to lock rotation in X and Z so that it doesn't flip over.
- Add a Capsule Collider.
- Set the Capsule Collider Center to (0, 0.22, 0.13).
- Set the Capsule Collider Radius to 0.24.
- Set the Capsule Collider Height to 1.41.
- Set the Capsule Collider Direction to Z-axis.

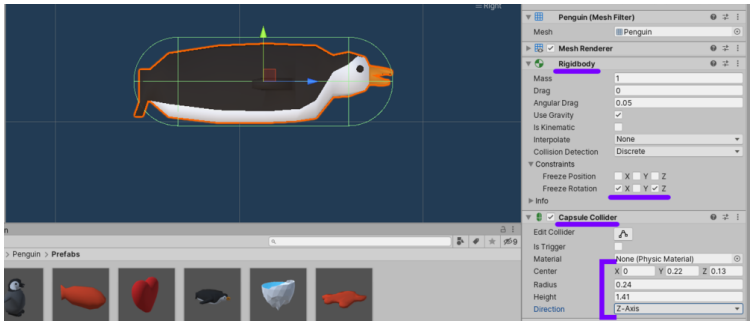

Figure 03: The Penguin, with changes for Rigidbody and Capsule Collider in the Inspector highlighted

- Attach the PenguinAgent script.
- If the BehaviorParameters script is not added automatically, add that as well.
- Set up Behavior Parameters **(Figure 04)**:
    - Behavior Name: PenguinLearning
    - Space Size: 8
    - Branches Size: 2
    - Branch 0 Size: 2
    - Branch 1 Size: 3
- Set up Penguin Agent **(Figure 04)**:
    - Max Step: 5000
    - Decision Interval: 4
    - Drag the Heart Prefab from the Project tab into the appropriate box.
    - Drag the Regurgitated Fish Prefab from the Project tab into the appropriate box.

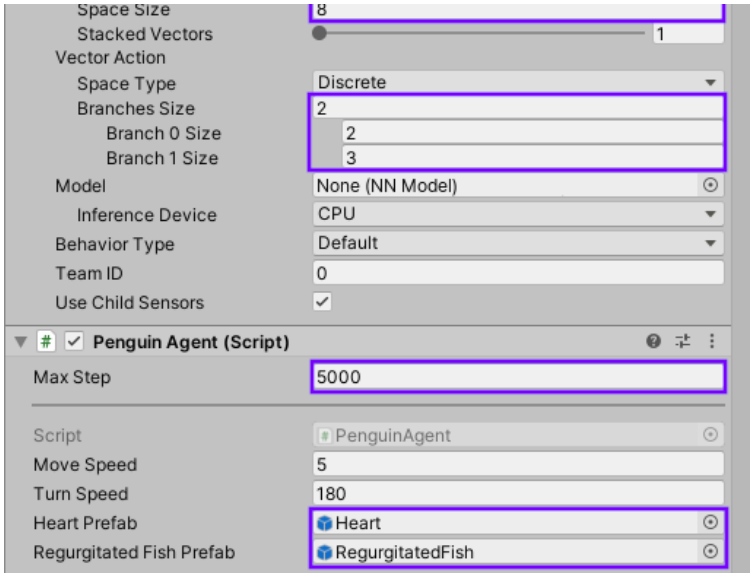

Figure 04: Inspector changes for the Behavior Parameters and PenguinAgent components on the Penguin prefab

The Behavior Name will need to match configuration files that you will make in the Training and Inference tutorial. Make sure that it's spelled exactly the same or else training will not work.

The Vector Observation Space Size corresponds to the AddVectorObs() calls we made in CollectObservations() in PenguinAgent.cs. We have eight total values.

The Vector Action Branches Size indicates how many actions are possible. In our case, these are Move Forward and Turn, so we have two. Branch 0 is Move Forward, and we have two options: don't move and move forward. Branch 1 is Turn, and we have three options: turn left, don't turn, and turn right.

Max Steps means the agent will reset automatically after 5,000 steps. Decision Interval of 4 means that the agent will be asked what to do every four steps and will carry out that action between each request.

- Set up the RayPerceptionSensorComponent3D **(Figure 05)**:
  - Detectable tags size: 3
  - Element 0: baby
  - Element 1: fish
  - Element 2: Untagged

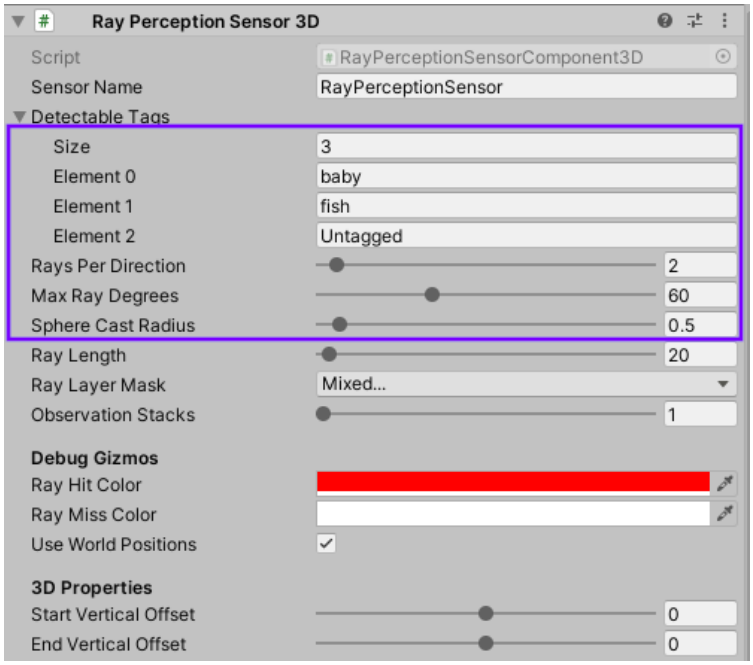

Figure 05: Inspector changes for the RayPerceptionSensorComponent3D on the Penguin

Detectable Tags tells the sensor which tags to report collisions for. **Rays Per Direction** tells the sensor to cast two rays to either side of center, as well as straight ahead. **Max Ray Degrees** tells the sensor to spread the two rays out 60 degrees in either direction. The total spread is 120 degrees, 30 degrees between each ray. **Sphere Cast Radius** projects a sphere of radius 0.5 along each ray to test collisions, not just a single point in space.

The screenshot below **(Figure 06)** shows spherecasts, two of which have hit fish and two of which have hit the

that Unity Physics is not flawless, but fortunately our ML-Agents are robust enough to work despite occasional misleading or incomplete information.

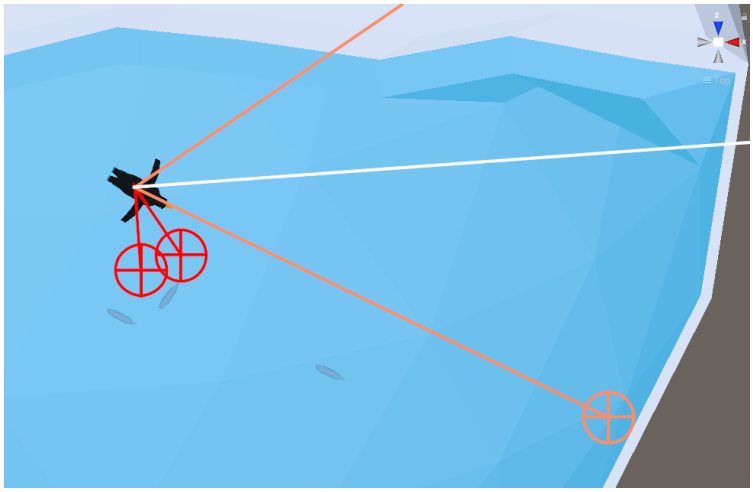

Figure 06: Penguin spherecasts as seen in the Scene view while the game is playing

PENGUIN AREA

In this section, you'll create an enclosed space for the penguin agent to catch fish and feed its baby. ML-Agents can work in environments of any shape and size, but this tutorial uses a small, confined space to keep things simple.

- Double-click on the PenguinArea Prefab in the *Prefabs8* folder to open it.
- Hold the Ctrl key on your keyboard and click on InvCylinder_Collider, RockCollider_01 and RockCollider_02 to select all three **(Figure 07)**. These 3d Meshes are intended to be simplified Mesh Colliders and will not be visible.

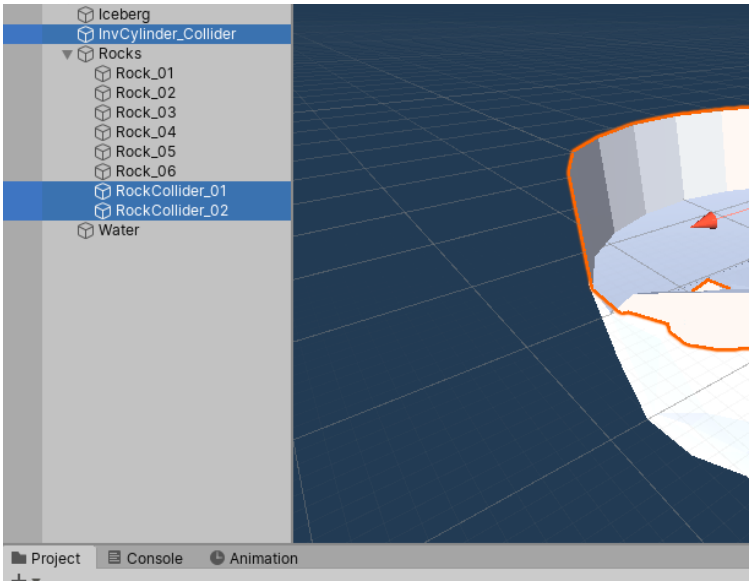

Figure 07: The three colliders selected

- Remove the Mesh Renderer component using the drop-down menu in the Inspector as shown **(Figure 08)**. This will make the Colliders invisible to the camera.

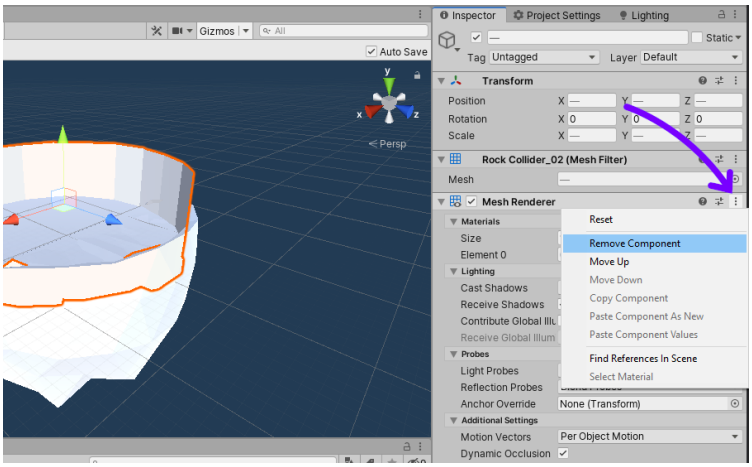

Figure 08: Removing the Mesh Renderer from the Collider objects

- Click the **Add Component** button and add a Mesh Collider component to all three objects **(Figure 09)**.

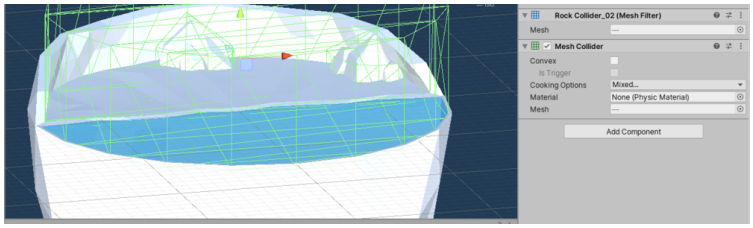

Figure 09: Newly added Mesh Colliders on the three collider objects

- Create a new folder called *Materials* inside Assets\Penguin.
- Create a new Material called *Snow* in the *Materials* folder.
- Set the Snow Albedo/Color to 255, 255, 255, 255.
- Set the Snow Smoothness to 0.
- Create a new Material called *Water* in the *Materials* folder.
- Set the Water Rendering Mode to **Transparent**.
- Set the Water Albedo/Color to 0, 200, 255, 165.
- Set the Water Smoothness to 0.
- Apply the Snow Material to the iceberg and rocks in the area. (You can apply Materials to multiple objects at once by Ctrl + clicking each object and dragging the Material into the Inspector tab).
- Apply the Water Material to the water.

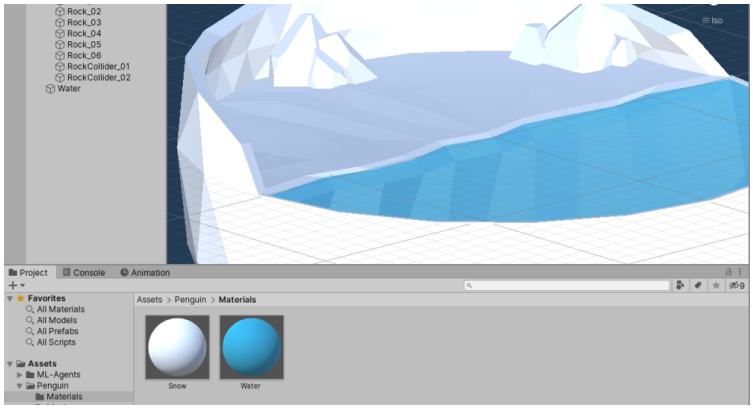

Figure 10: The PenguinArea with Snow and Water Materials applied

- Create a new Text Mesh Pro object by right-clicking on PenguinArea, and choosing **3D Object > Text - TextMeshPro (Figure 11)**.

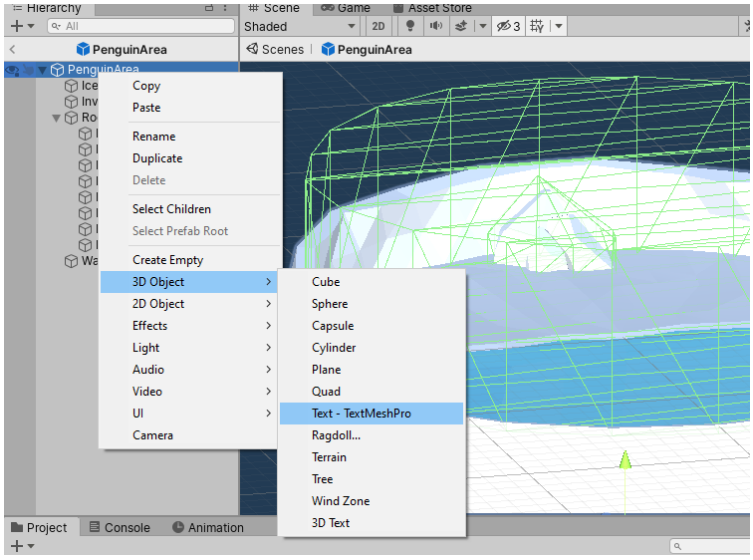

Figure 11: Adding a Text - TextMeshPro

- Click the **Import TMP Essentials** button if prompted.

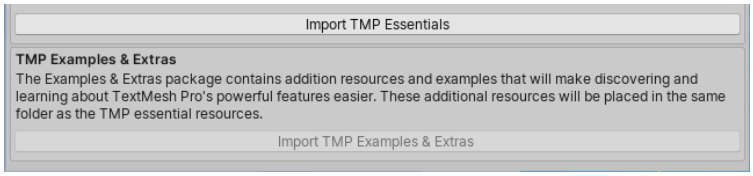

Figure 12: The TMP Importer window

You should now have a 2D text object floating in 3D space.

- Rename the object to *Cumulative Reward (TMP)*.
- Move the text so that you can see it:
    - PosX: 7
    - PosY: 2
    - PosZ: 11
    - Width: 20
    - Height: 5
    - Rotation (0, 30, 0)

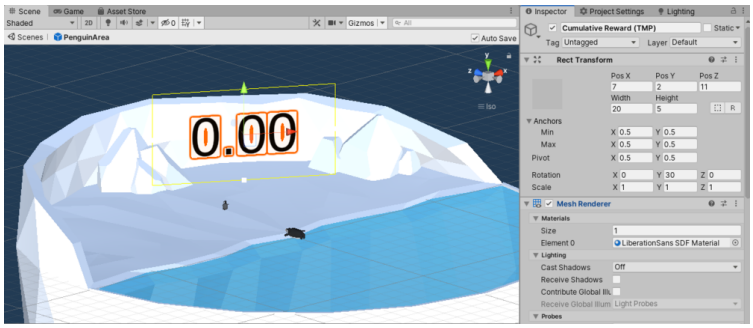

Figure 13: The Cumulative Reward text placed in the PenguinArea

- Set the Text settings:
    - Default text: 0.00
    - Font Size: 30
    - Vertex Color: Black
    - Alignment: Center Horizontal and Center Vertical

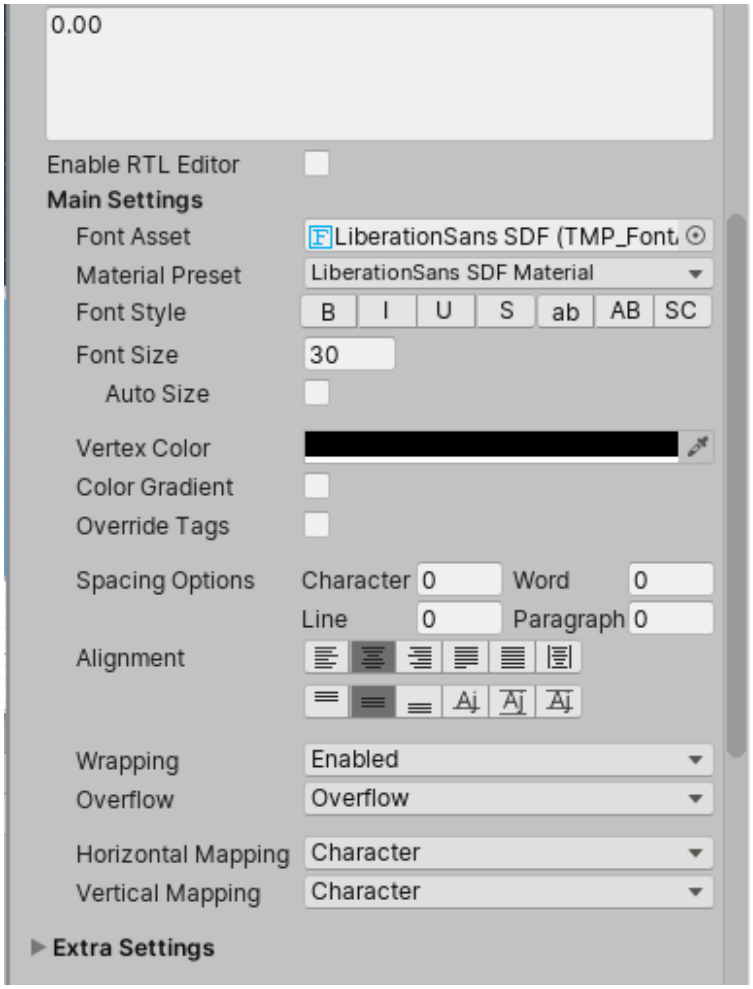

Figure 14: TextMeshPro settings in the Inspector

- Add a BabyPenguin Prefab to the area.
- Add a Penguin (agent) Prefab to the area.

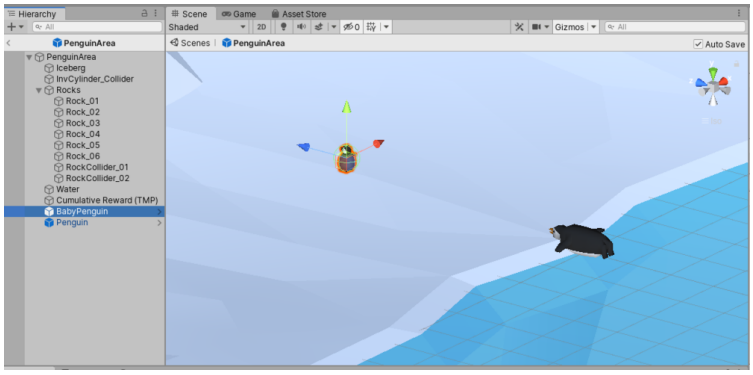

Figure 15: The PenguinArea (in the Prefab editor) with BabyPenguin and Penguin Prefabs added

- Add a PenguinArea script component to the

- Drag the BabyPenguin from the Hierarchy to the **Penguin Baby** field in the PenguinArea.
- Drag the Cumulative Reward (TMP) to the **Cumulative Reward Text** field in the PenguinArea.
- Drag the Fish Prefab from the Project tab to the **Fish Prefab** field in the PenguinArea.

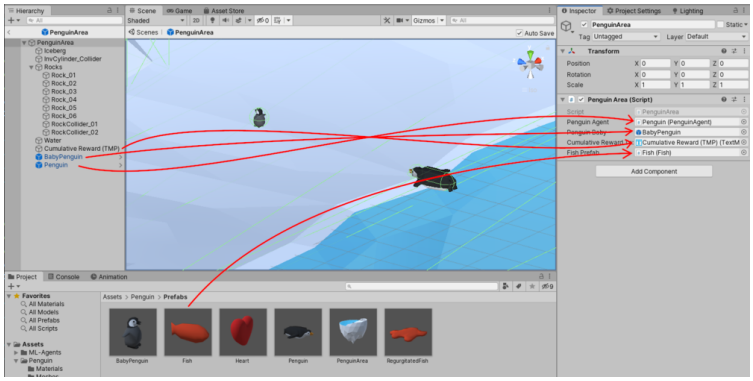

Figure 16: Drag the four objects to their respective fields

- Exit the Prefab editor by clicking on the **Back (<)** data-preserve-html-node="true" arrow in the Hierarchy tab.

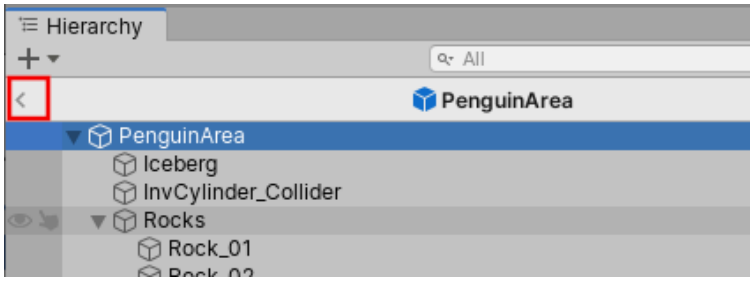

Figure 17: Click the arrow to exit the Prefab editor

- Expand the PenguinArea in the Scene view so that you can see its children objects.
- Select the Penguin in the PenguinArea.
- Change the Behavior Type **(Figure 19)** of the Behavior Parameters component to **Heuristic Only** (you will revert this change after you test it).

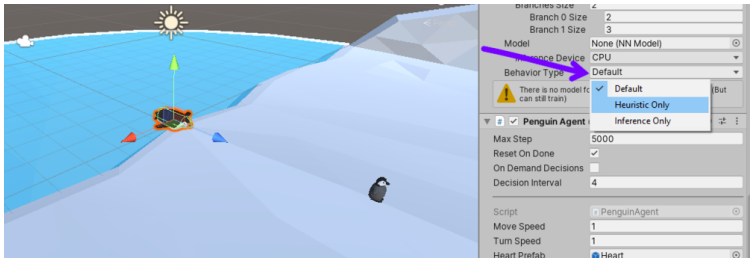

Figure 18: Changing the Behavior Type to Heuristic Only

This will not modify the Prefab, but it will allow you to test the Scene by playing as the penguin.

- Move the Main Camera so that it's pointing at the area:
  - Position: (-8, 9, -20)
  - Rotation: (30, 22, 0)
- Press the **Play** button at the top center of the Unity Editor.

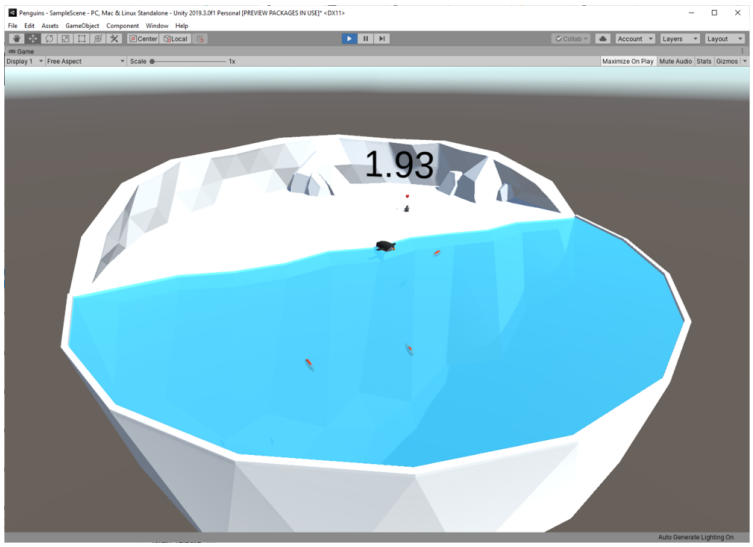

Figure 19: Changing the Behavior Type to Heuristic Only

You should be able to control the movement of the penguin with the W, A, and D keys on your keyboard. Try to pick up fish and deliver them to the baby. Once you are within the six meter feed_radius, the penguin should feed the baby and the Cumulative Reward text

- Change the Behavior Type back to **Default** after you've successfully tested the penguin.

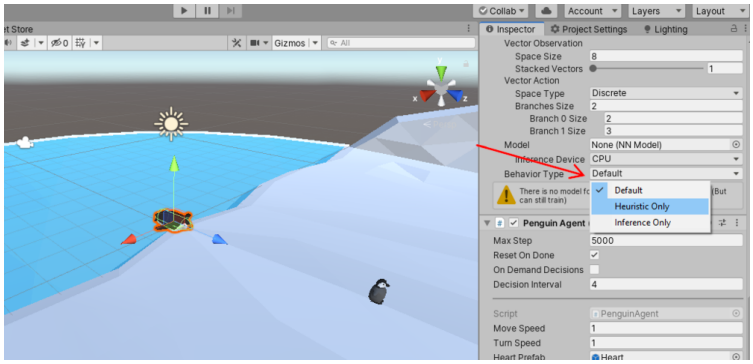

Figure 20: Changing the Behavior Type back to Default

Before moving on to training, you should create several copies of the area so that several penguins can train simultaneously. This step is optional, but should help training go faster.

- Duplicate the PenguinArea seven times so that you have a total of eight areas.
- Space out the areas by about 40 meters in X and Z so that they do not overlap.

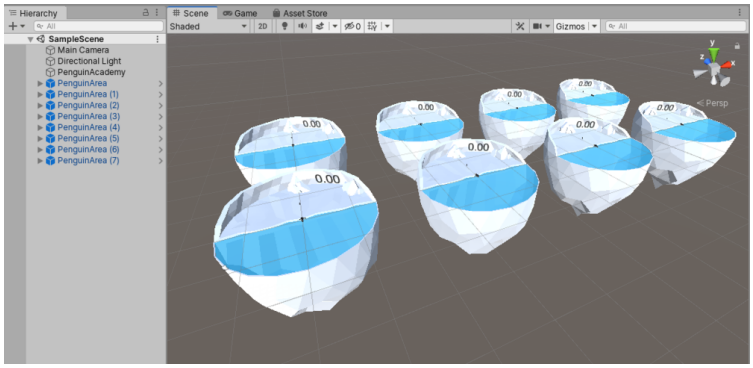

Figure 21: Eight instances of the completed PenguinArea, ready for training

CONCLUSION

You should now have Prefabs for the penguin agent, baby penguin, fish, and area as well as a Scene set up

IMMERSIVE LIMIT

TUTORIAL PARTS

Reinforcement Learning Penguins (Part 1/4)

Reinforcement Learning Penguins (Part 2/4)

Reinforcement Learning Penguins (Part 3/4)

Reinforcement Learning Penguins (Part 4/4)

COMMENTS (3) Most Liked Subscribe via e-mail

Preview Post Comment...

david familian A day ago

· 0 Likes

hi

great tutorial!!! I have upgraded to .16. which is the latest?
this is supposed to be a release candidate. I fixed everything and it works in heuristic. but in default Penguin barely moves. I discovered that vectorAction[0] is always 0

```
using MLAgents.Sensors;
using UnityEngine.Serialization;
using MLAgents.SideChannels;
public class PenguinAgent : Agent
{
[Tooltip("How fast the agent moves forward")]
public float moveSpeed = 5f;

[Tooltip(&quot;How fast the agent tu
public float turnSpeed = 180f;

[Tooltip(&quot;Prefab of the heart t
public GameObject heartPrefab;

[Tooltip(&quot;Prefab of the regurg
public GameObject regurgitatedFishPr

private PenguinArea penguinArea;
new private Rigidbody rigidbody;
private GameObject baby;
private bool isFull; // If true, pen
private float feedRadius = 0f;

/// &lt;summary&gt;
/// Initial setup, called when the a
/// &lt;/summary&gt;
public override void Initialize()
&#123;
    base.Initialize();
    penguinArea = GetComponentInPare
    baby = penguinArea.penguinBaby;
    rigidbody = GetComponent&lt;Rig
&#125;
```

```
/// Perform actions based on a vecto
/// &lt;/summary&gt;
/// &lt;param name=&quot;vectorActio
public override void OnActionReceive
&#123;
    // Convert the first action to
    vectorAction[0] = 1;
    float forwardAmount = vectorAct

    //Debug.Log(vectorAction[0]);
    //float forwardAmount = .25f;

    // Convert the second action to
    float turnAmount = 0f;
    if (vectorAction[1] == 1f)
    &#123;
        turnAmount = -1f;
    &#125;
    else if (vectorAction[1] == 2f)
    &#123;
        turnAmount = 1f;
    &#125;

    // Apply movement
    rigidbody.MovePosition(transfor
    transform.Rotate(transform.up *

    // Apply a tiny negative reward
    if (maxStep &gt; 0) AddReward(-
&#125;

/// &lt;summary&gt;
/// Read inputs from the keyboard a
```

```
/// &lt;/summary&gt;
/// &lt;returns&gt;A vectorAction a
//public override float[] Heuristic
public override void Heuristic(floa
&#123;
    actionsOut[0] = 0f;
    actionsOut[1] = 0f;
    if (Input.GetKey(KeyCode.W))
    &#123;
        // move forward
        actionsOut[0] = 1f;
    &#125;
    if (Input.GetKey(KeyCode.A))
    &#123;
        // turn left
        actionsOut[1] = 1f;
    &#125;
    else if (Input.GetKey(KeyCode.D
    &#123;
        // turn right
        actionsOut[1] = 2f;
    &#125;
    //else if (Input.GetKey(KeyCode
    //&#123;
    //    // turn right
    //    actionsOut[0] = 2f;
    //&#125;

&#125;
```

```
/// &lt;/summary&gt;
//public override void CollectObser
public override void CollectObserva
&#123;
    //sensor.AddObservation(gameObj
    //replace all calls to AddVecto
    //sensor.AddOneHotObservation()
    // Whether the penguin has eate
    sensor.AddObservation(isFull);

    // Distance to the baby (1 floa
    sensor.AddObservation(Vector3.D

    // Direction to baby (1 Vector3
    sensor.AddObservation((baby.tra

    // Direction penguin is facing
    sensor.AddObservation(transform

    // 1 + 1 + 3 + 3 = 8 total valu
&#125;

private void FixedUpdate()
&#123;
    // Request a decision every 5 s
    // but for the steps in between
    // of the previous decision
    if (StepCount % 5 == 0)
    &#123;
        RequestDecision();
    &#125;
    else
    &#123;
```

```
        // Test if the agent is close e
        if (Vector3.Distance(transform.
        &#123;
            // Close enough, try to fee
            RegurgitateFish();
        &#125;
    &#125;

    /// &lt;summary&gt;
    /// When the agent collides with so
    /// &lt;/summary&gt;
    /// &lt;param name=&quot;collision&
    private void OnCollisionEnter(Colli
    &#123;
        if (collision.transform.Compare
        &#123;
            // Try to eat the fish
            EatFish(collision.gameObjec
        &#125;
        else if (collision.transform.Co
        &#123;
            // Try to feed the baby
            RegurgitateFish();
        &#125;
    &#125;

    /// &lt;summary&gt;
    /// Check if agent is full, if not,
    /// &lt;/summary&gt;
    /// &lt;param name=&quot;fishObject
    private void EatFish(GameObject fis
    &#123;
```

```
        penguinArea.RemoveSpecificFish(

        AddReward(1f);
    &#125;
    /// &lt;summary&gt;
    /// Reset the agent and area
    /// &lt;/summary&gt;+
    ///
    public override void OnEpisodeBegin
    &#123;
        isFull = false;
        penguinArea.ResetArea();
        feedRadius = (int)SideChannelUt
        //feedRadius = Academy.Instance
    &#125;
    /// &lt;summary&gt;
    /// Check if agent is full, if yes,
    /// &lt;/summary&gt;
    private void RegurgitateFish()
    &#123;
        if (!isFull) return; // Nothing
        isFull = false;

        // Spawn regurgitated fish
        GameObject regurgitatedFish = In
        regurgitatedFish.transform.parer
        regurgitatedFish.transform.posi
        Destroy(regurgitatedFish, 4f);

        // Spawn heart
        GameObject heart = Instantiate&
        heart.transform.parent = transf
```

```
        AddReward(1f);

        if (penguinArea.FishRemaining &
        &#123;
            EndEpisode();
        &#125;
    &#125;

    }
```

DavidVella98 -

A month ago · 0 Likes

Hi I'm trying to follow your tutorial however I'm getting multiple errors, one of which is:

Assets\Assets\Game\Scripts\PenguinAgent.cs(92,26): error CS0115: 'PenguinAgent.AgentReset()': no suitable method found to override

Adam Kelly

A month ago · 0 Likes

Are you using the v0.14 release? If so, double check to make sure you are inheriting from the Agent class.

If you are using the master branch of the repo, you should remove the package and install v0.14 because they change it on a daily basis and this tutorial will not work

PREVIOUS

Reinforcement Learning Penguins (Part 4/4) | Unity ML-Agents

NEXT

Reinforcement Learning Penguins (Part 2/4) | Unity ML-Agents