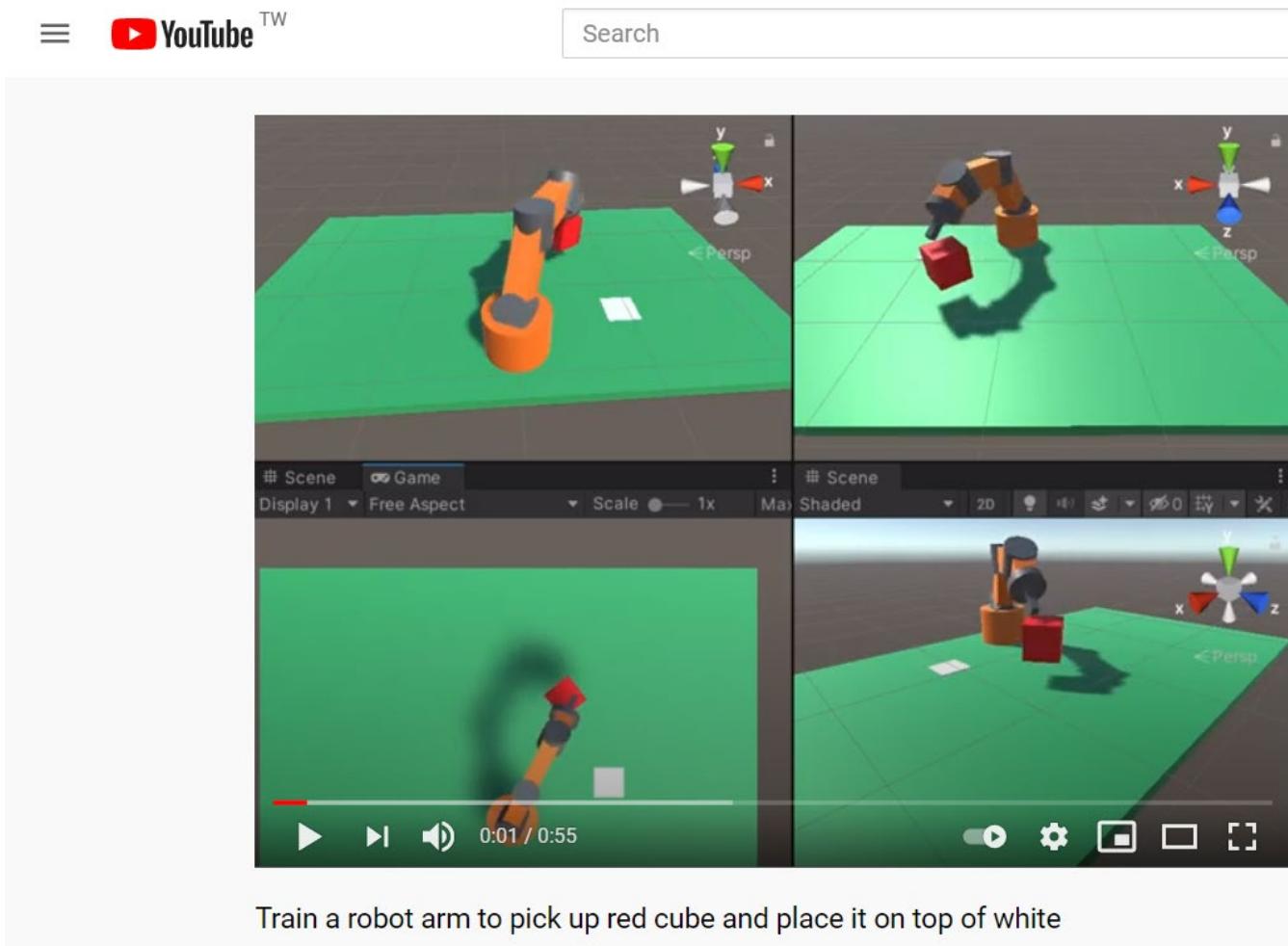


Youtube demo video

Youtube search: Train a robot arm to pick up red cube and place it on top of white



<https://youtu.be/-NVNT-YTv1s>

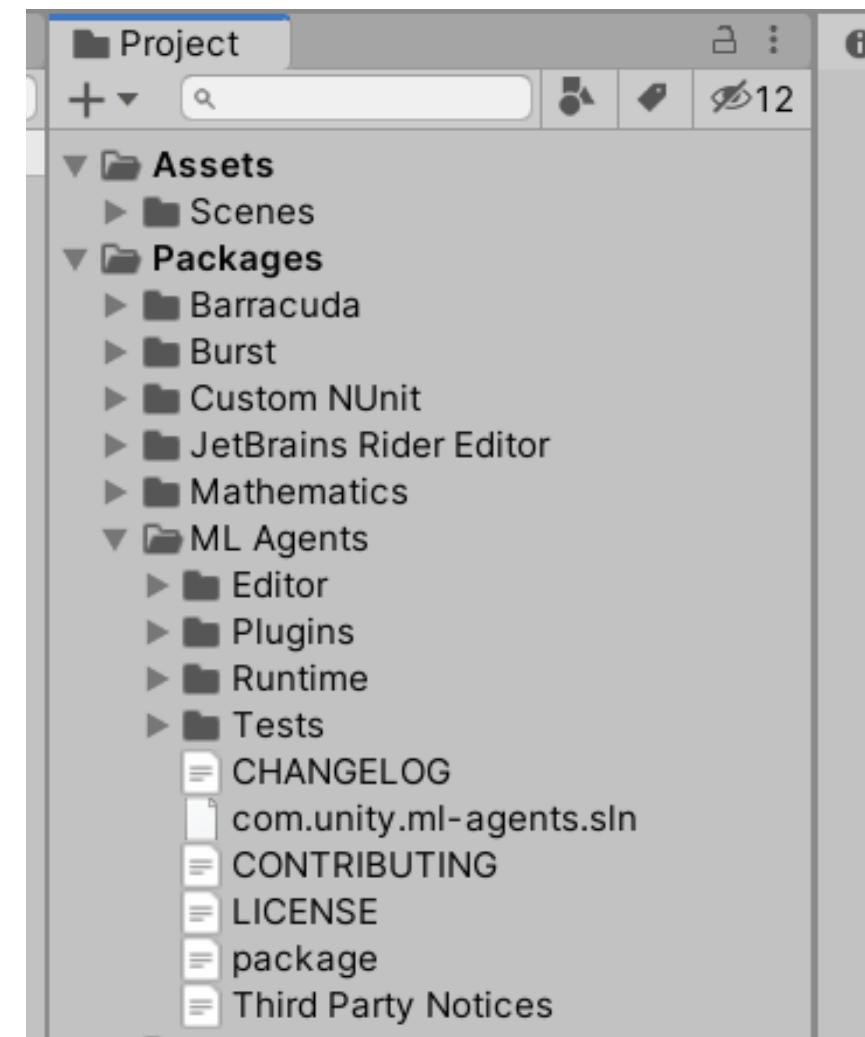
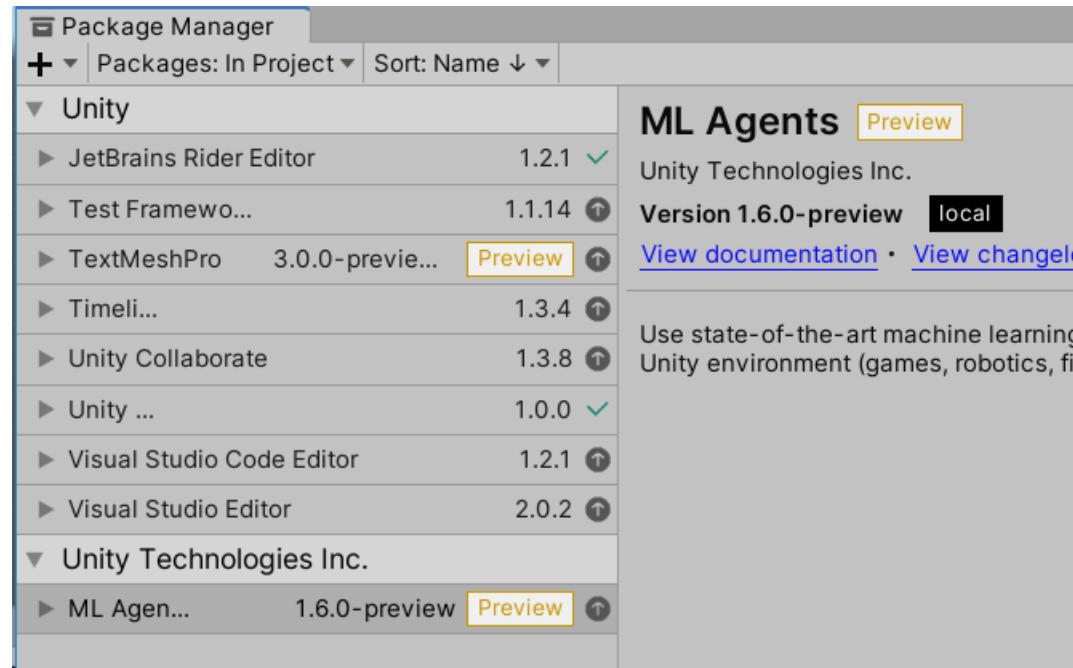
Download and save ML Agent to C:\

This will make it convenient to type commands to train
and monitor performance

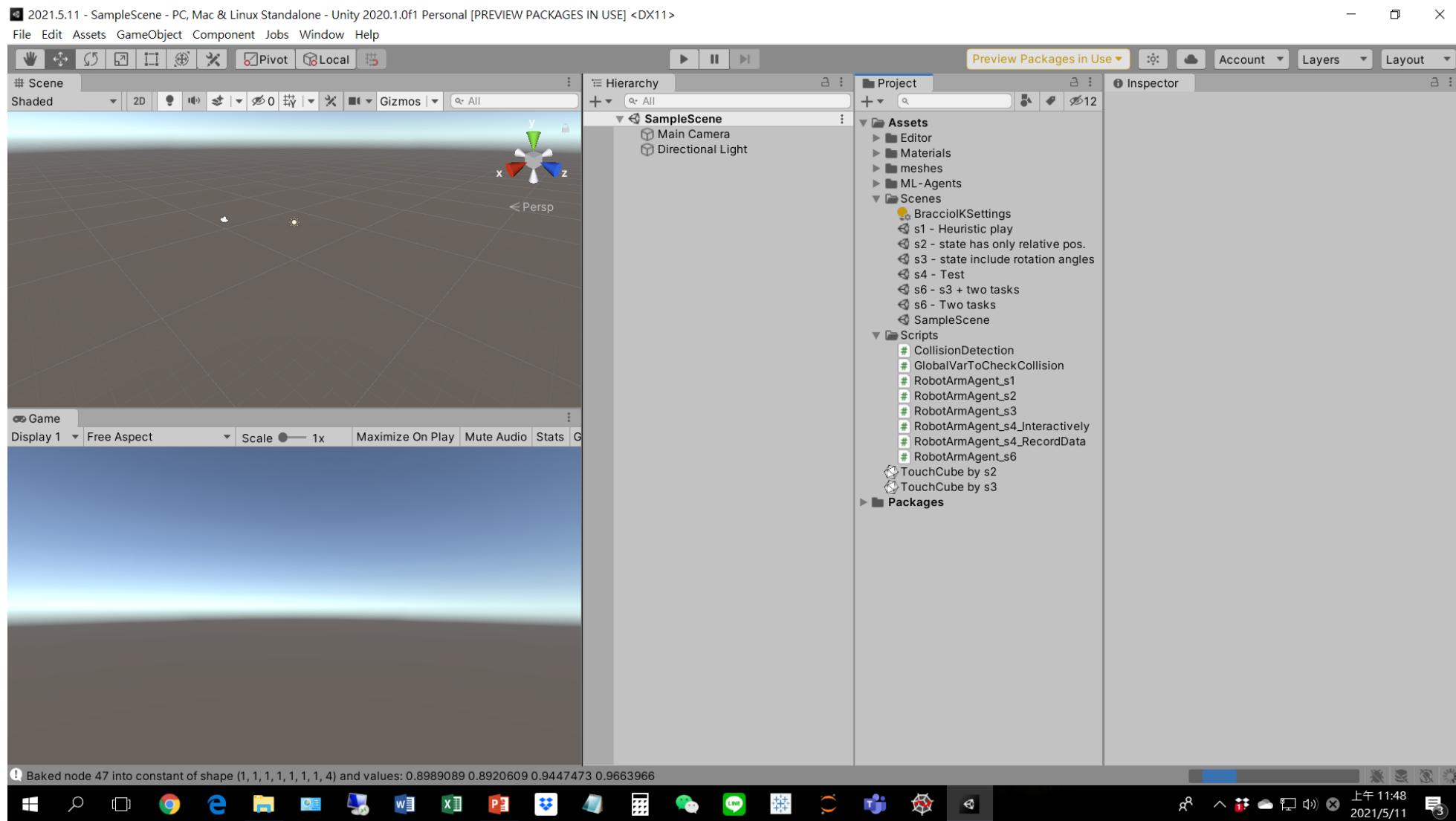
```
cd C:\ml-agents-release_10\config\ppo  
mlagents-learn TouchCube.yaml --run-id=1 --force
```

```
cd C:\ml-agents-release_10\config\ppo\results  
tensorboard --logdir=1
```

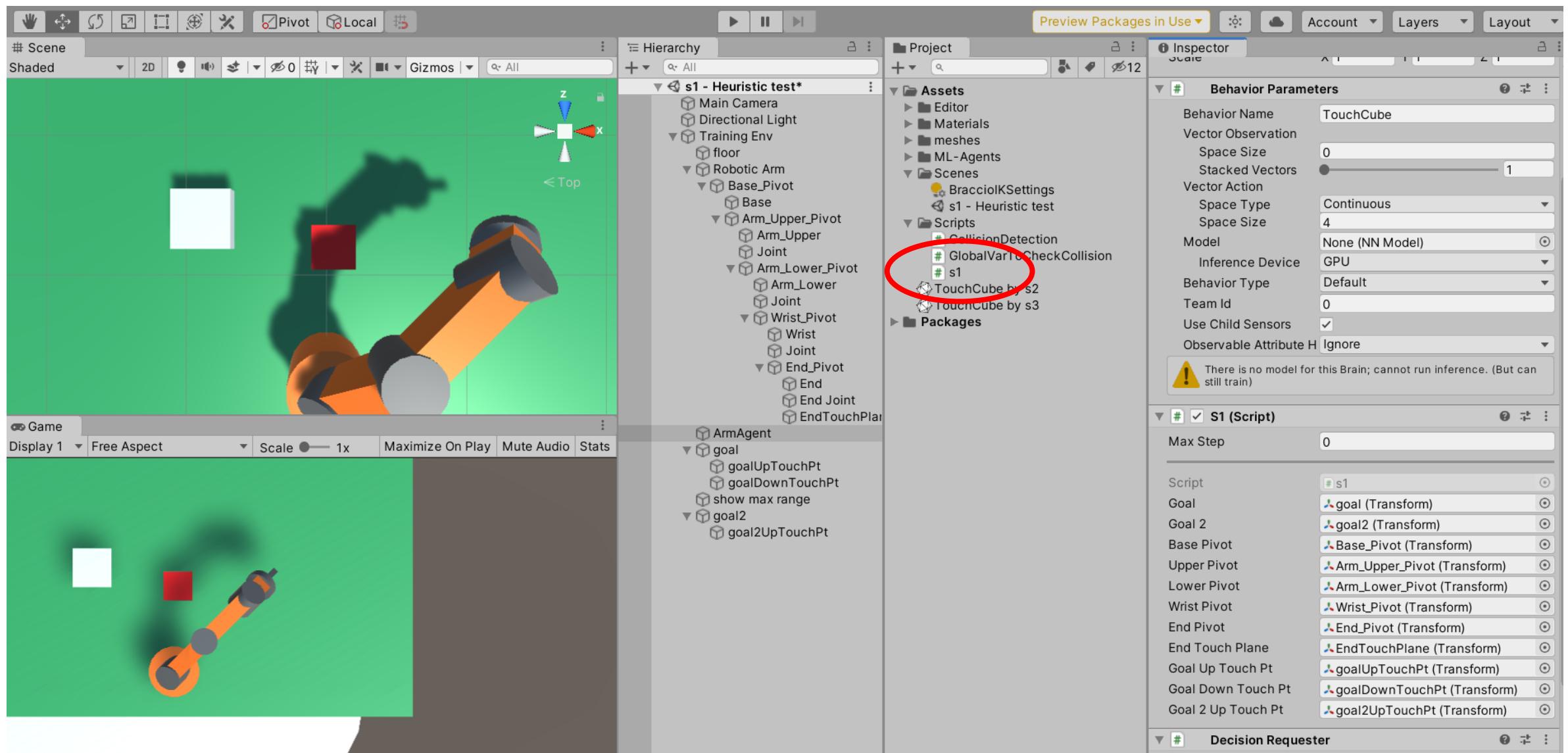
Create a new Unity project and import ML Agent package to this new project



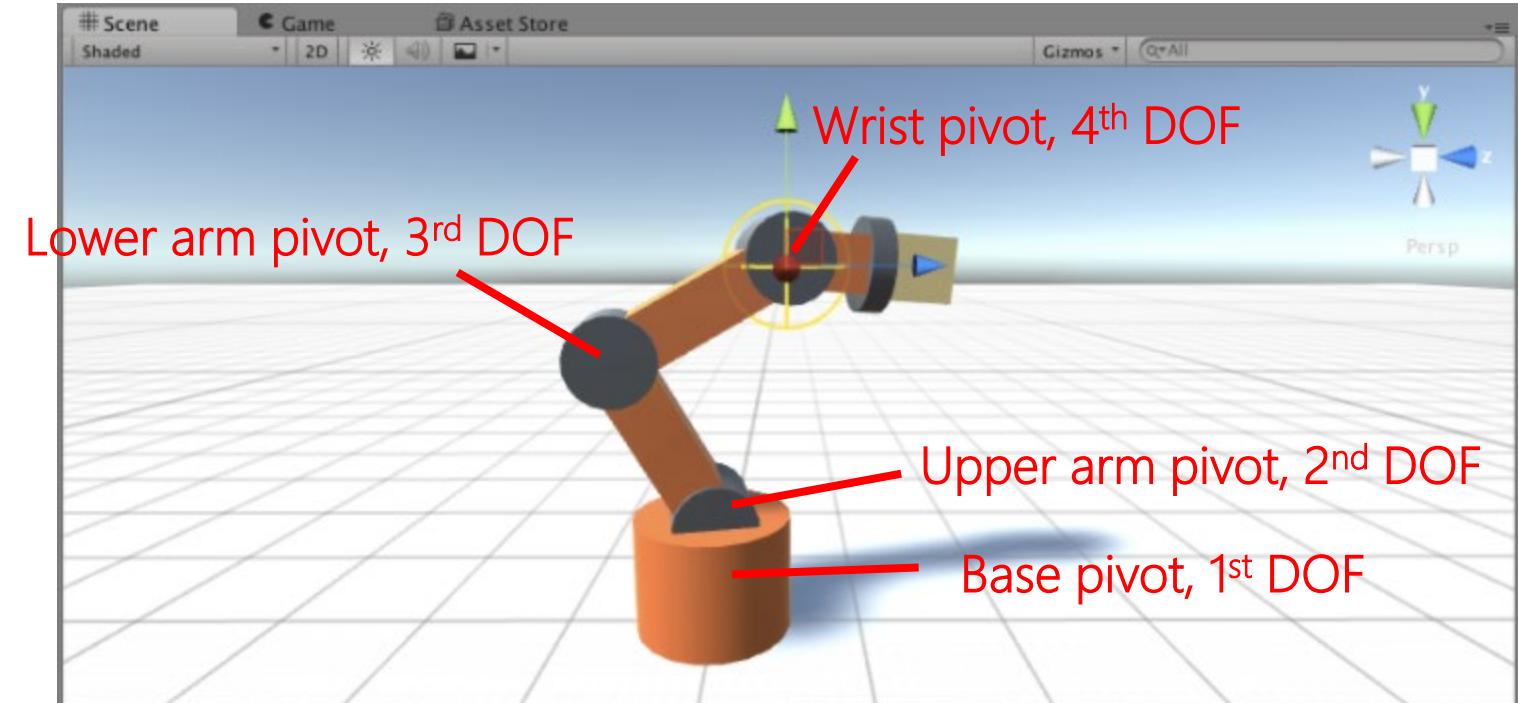
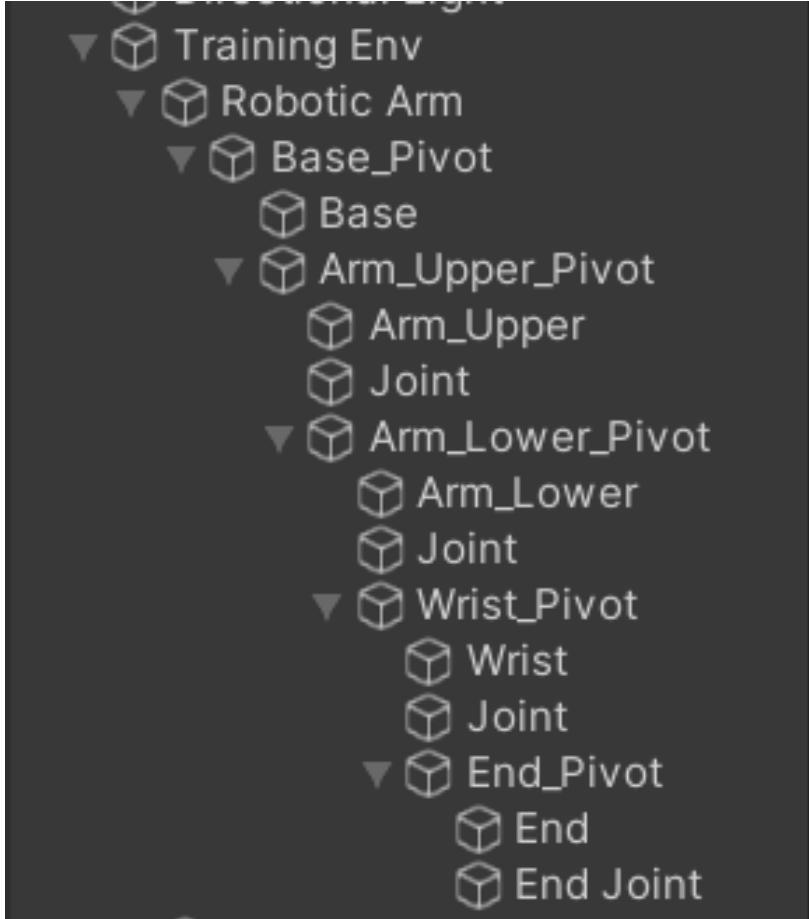
Import Robot arm package to Unity project



Open scene "s1 - Heuristic play"



This Unity project contains a Braccio robot arm

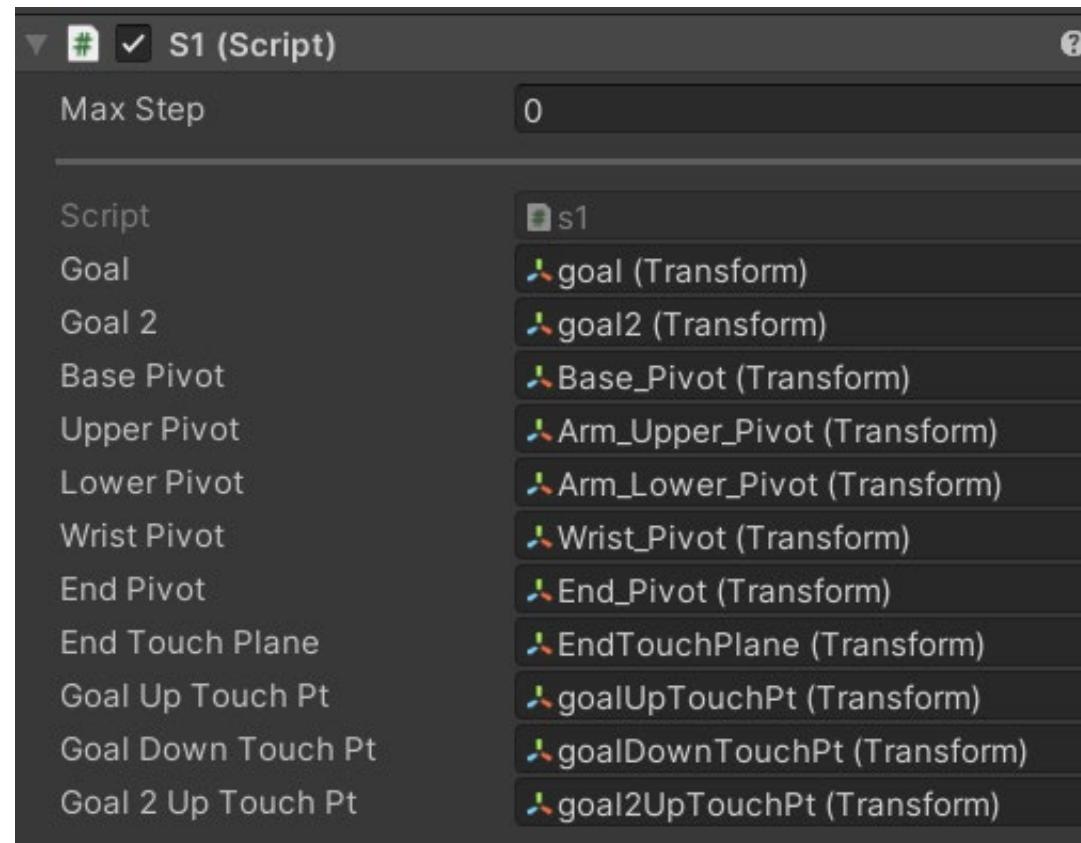


<https://github.com/tanyuan/braccio-ik-unity>

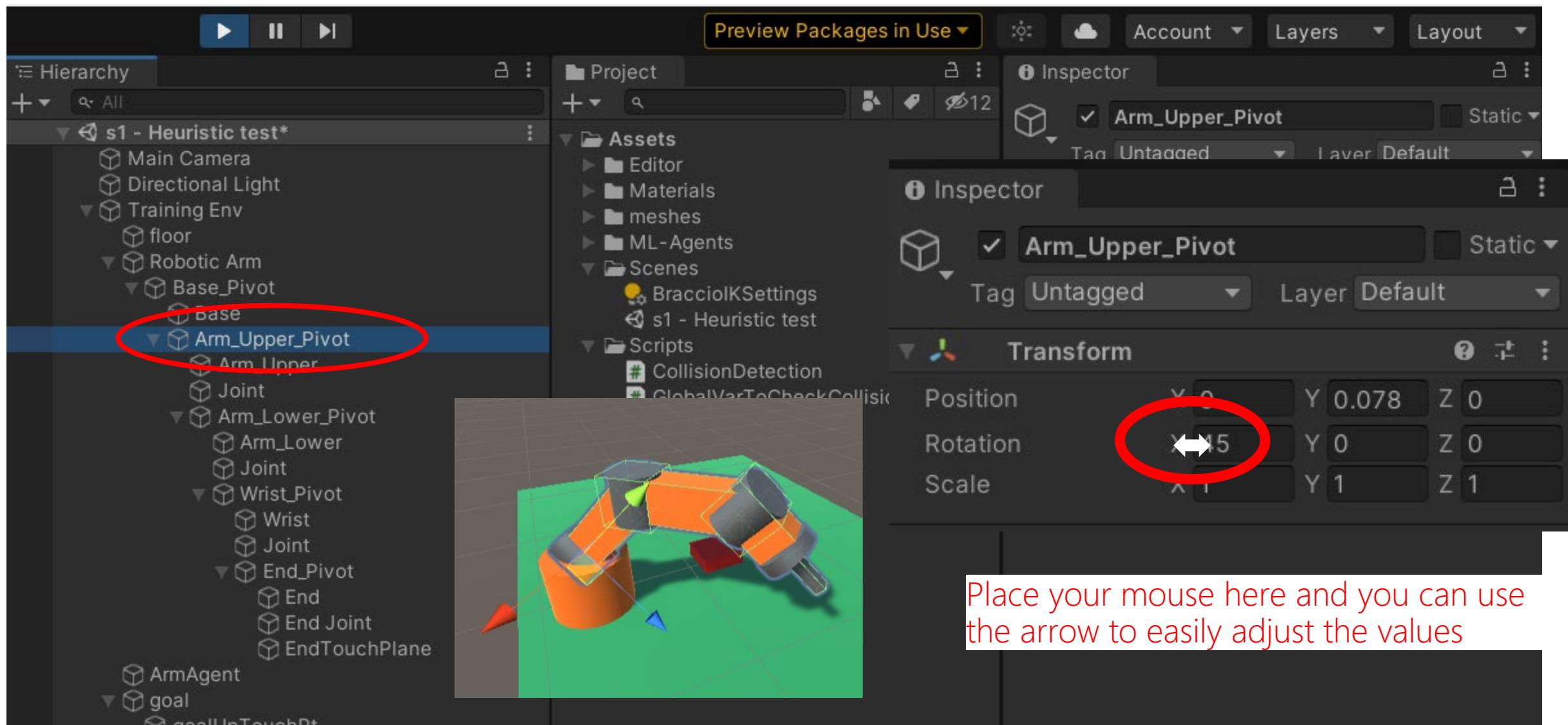
(1) Manually operates the robot arm in VE

Public variables to link agent script with scene objects

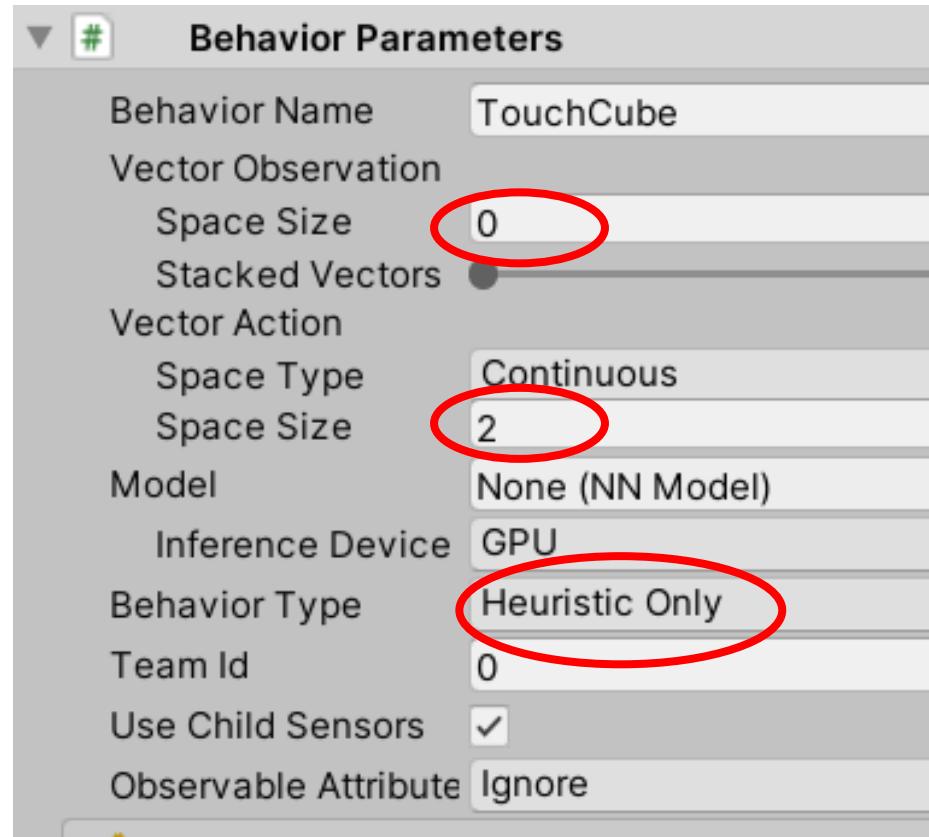
```
public Transform goal, goal2;  
public Transform BasePivot, UpperPivot, LowerPivot, WristPivot, EndPivot;  
public Transform EndTouchPlane, goalUpTouchPt, goalDownTouchPt, goal2UpTouchPt;  
int stage = 1;
```



Play and rotate robot arm by changing "Rotation" angle in the Inspector window



Behavior parameters

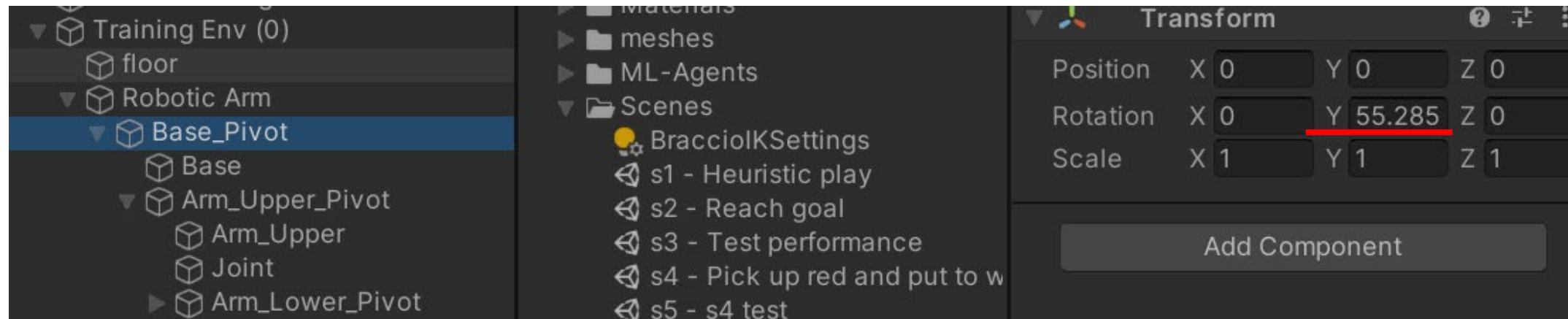


Play and use Up/Down, L/R keys to rotate arm

```
actionsOut[0] = Input.GetAxis("Horizontal");  
actionsOut[1] = Input.GetAxis("Vertical");
```

```
BasePivot.Rotate(0, vectorAction[0] * speed, 0);  
float RotationAngle = UnityEditor.TransformUtils.GetInspectorRotation(BasePivot).y;  
  
UpperPivot.Rotate(vectorAction[1] * speed, 0, 0);  
//float RotationAngle = UnityEditor.TransformUtils.GetInspectorRotation(UpperPivot).x;
```

Check whether arm rotation is out of range



```
float BaseRotationAngle = UnityEditor.TransformUtils.GetInspectorRotation(BasePivot).y;
float UArmRotationAngle = UnityEditor.TransformUtils.GetInspectorRotation(UpperPivot).x;
float LArmRotationAngle = UnityEditor.TransformUtils.GetInspectorRotation(LowerPivot).x;
float WRotationAngle = UnityEditor.TransformUtils.GetInspectorRotation(WristPivot).x;
if ((BaseRotationAngle >= -90 && BaseRotationAngle <= 90) &&
    (UArmRotationAngle >= 0 && UArmRotationAngle <= 90) &&
    (LArmRotationAngle >= 0 && LArmRotationAngle <= 90) &&
    (WRotationAngle >= 0 && WRotationAngle <= 90))
{
    return true;
}
```

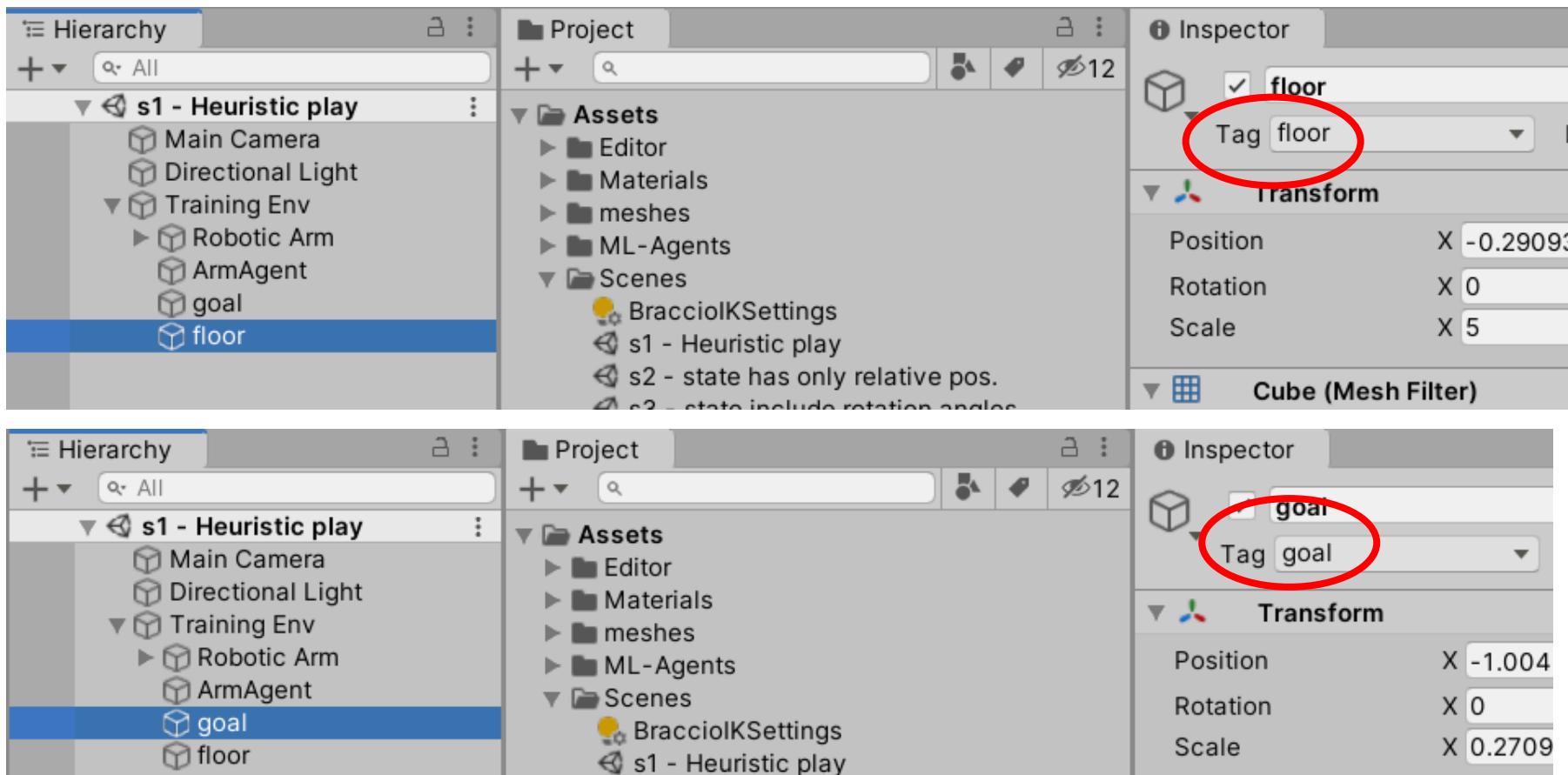
Collision detections

Use OnTriggerEnter/Exit to record collisions between robot arm and other scene objects

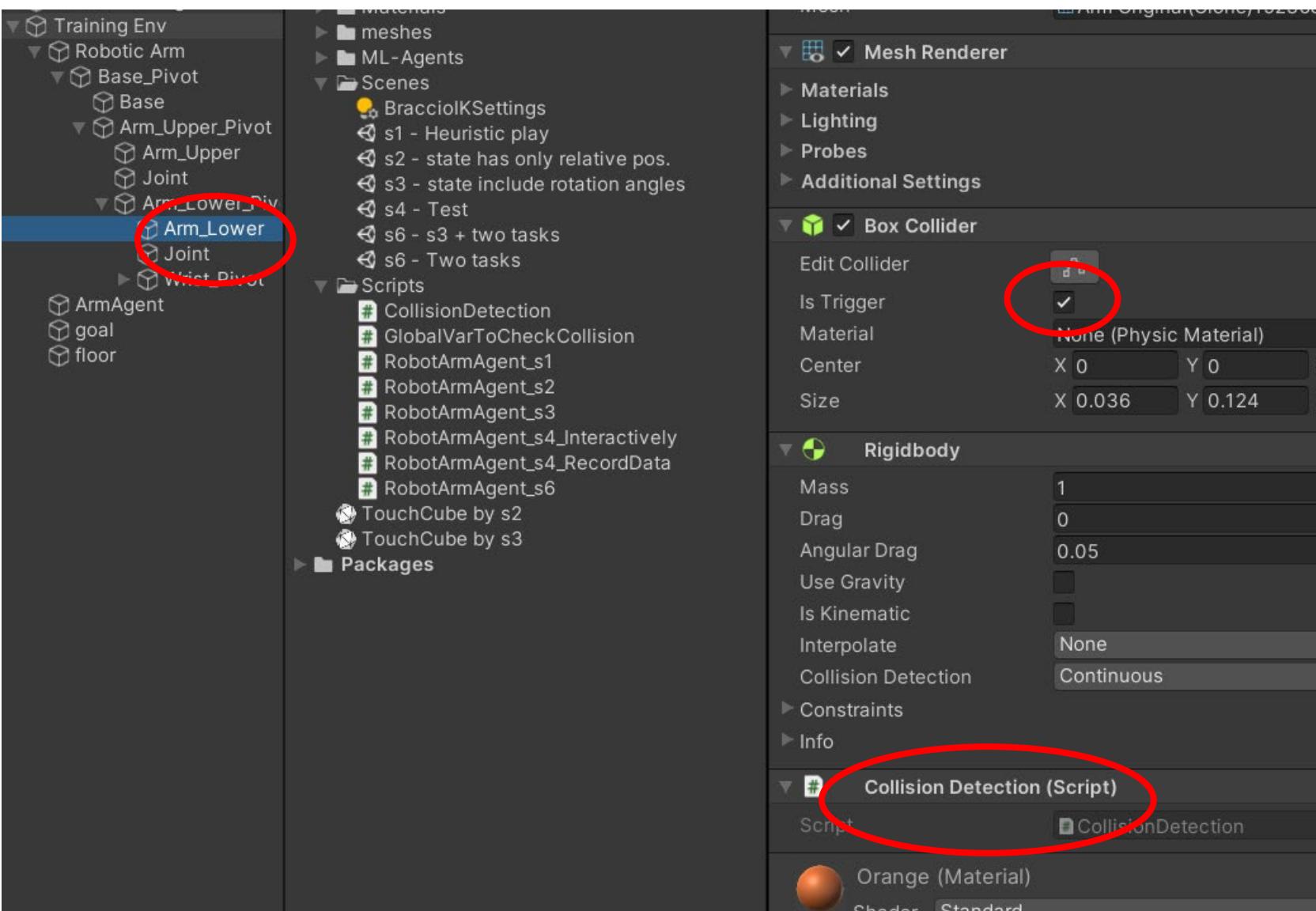
```
|public class CollisionDetection : MonoBehaviour  
{  
    public bool CollisionHappen;  
    ⌚Unity Message | 0 個參考  
    void OnTriggerEnter(Collider other)  
    {  
        if (other.gameObject.tag == "floor" || o  
        {  
            CollisionHappen = true;  
        }  
    }  
  
    void OnTriggerExit(Collider other)  
    {  
        if (other.gameObject.tag == "floor" || o  
        {  
            CollisionHappen = false;  
        }  
    }  
}
```

Add tags to floor and goal object

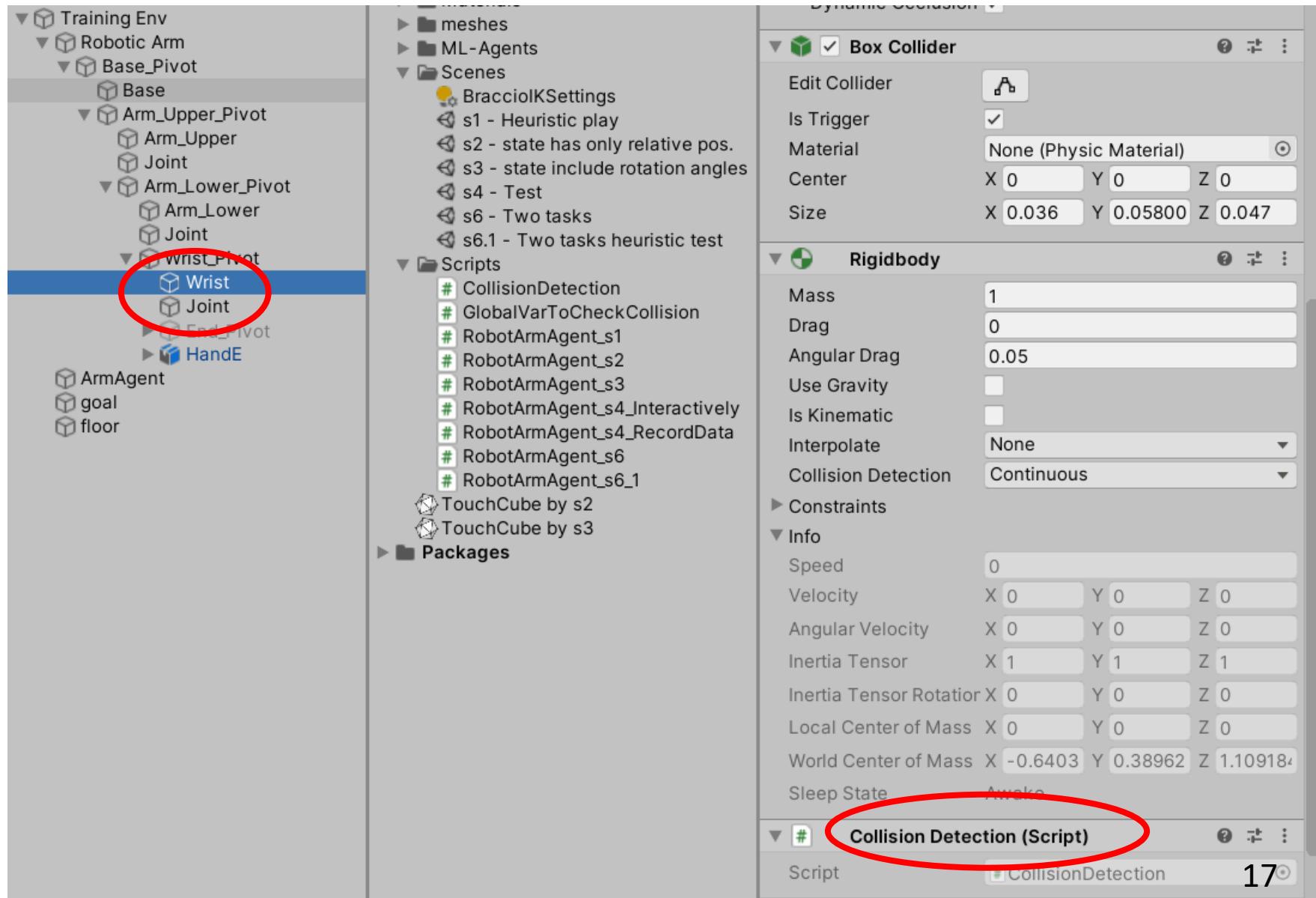
```
if (other.gameObject.tag == "floor" ||  
{  
    CollisionHappen = true;
```



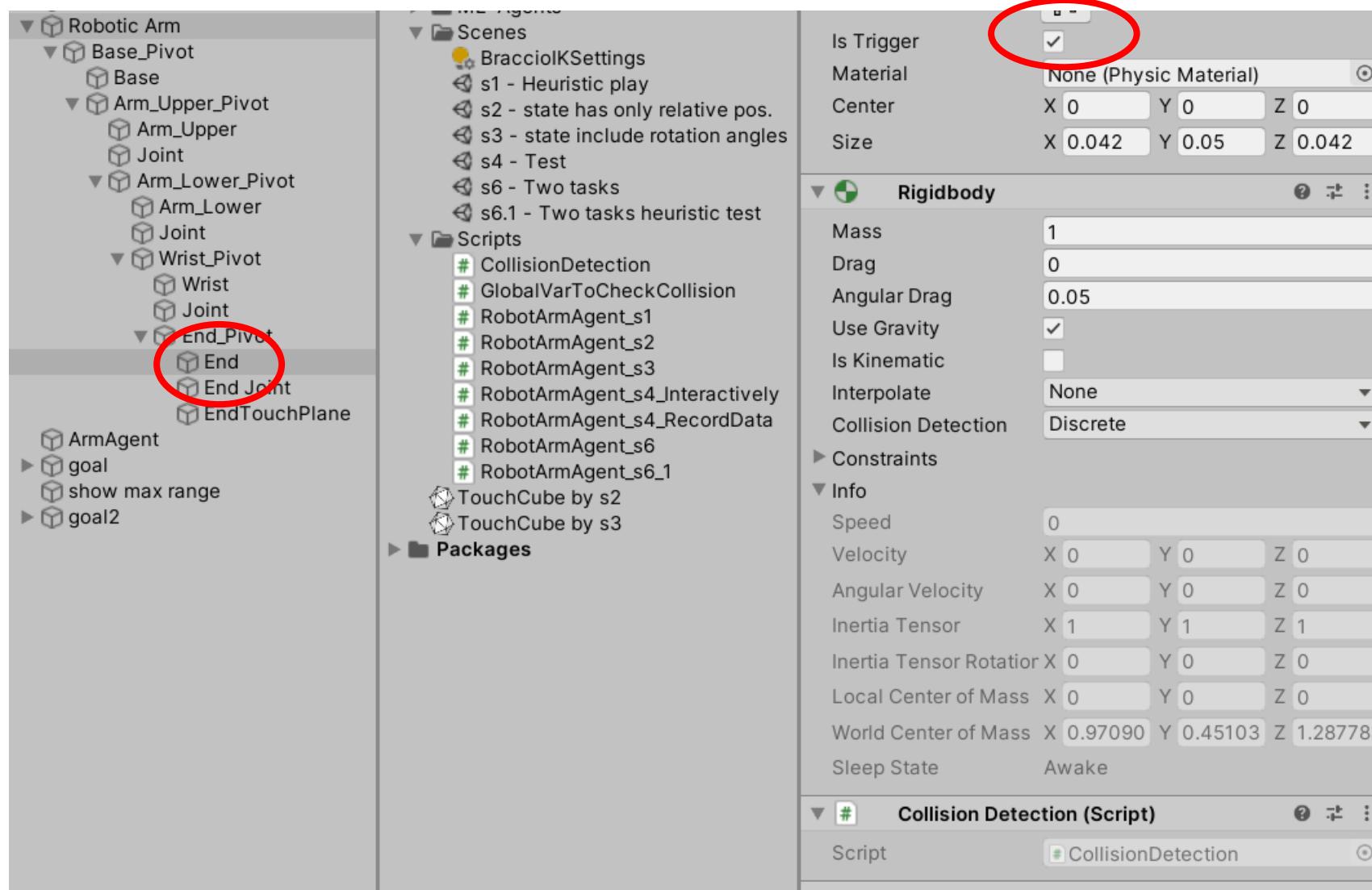
Add trigger collider, Rigid body, and collision detection script to Lower Arm



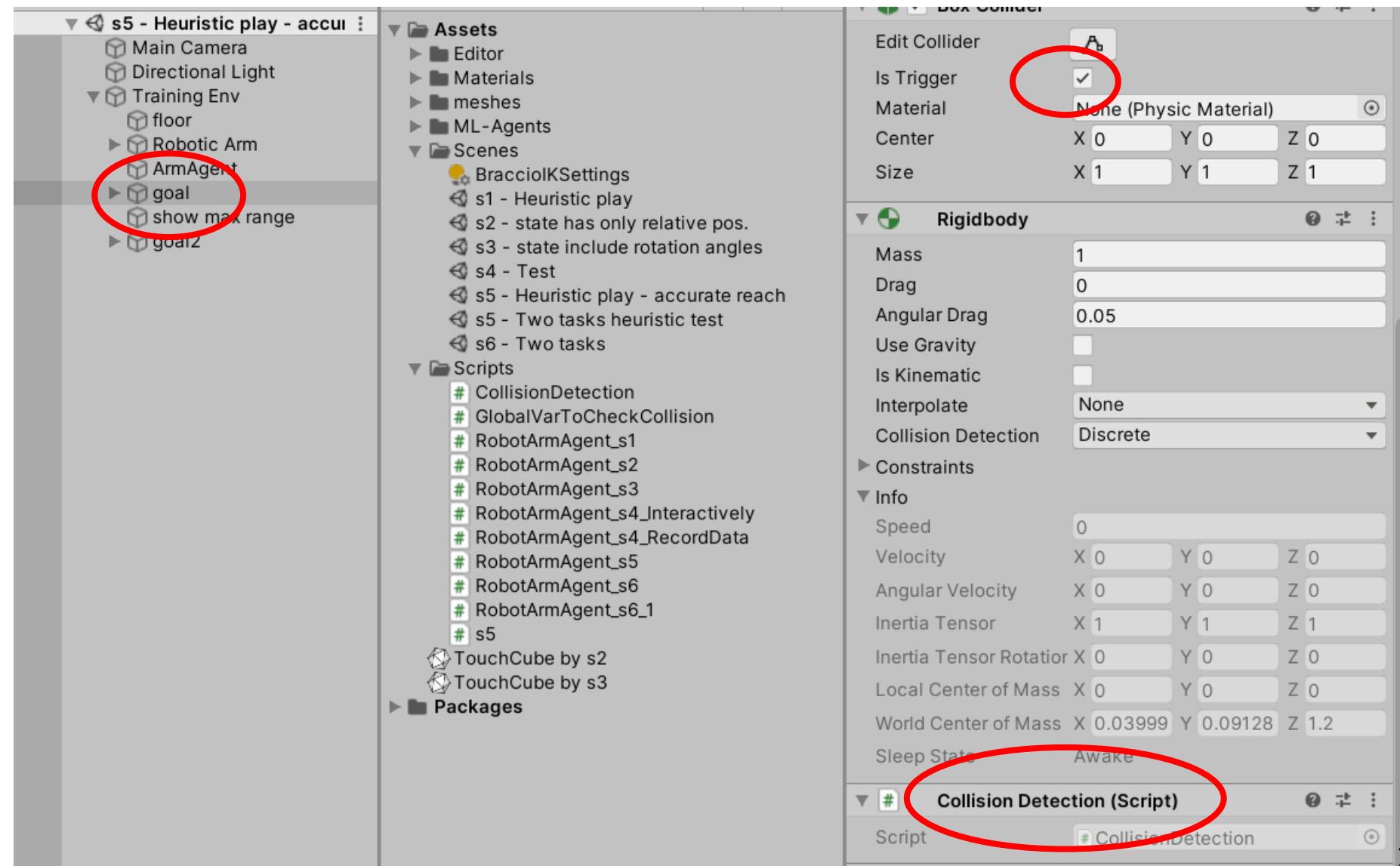
Add trigger collider, Rigid body, and collision detection script to Wrist



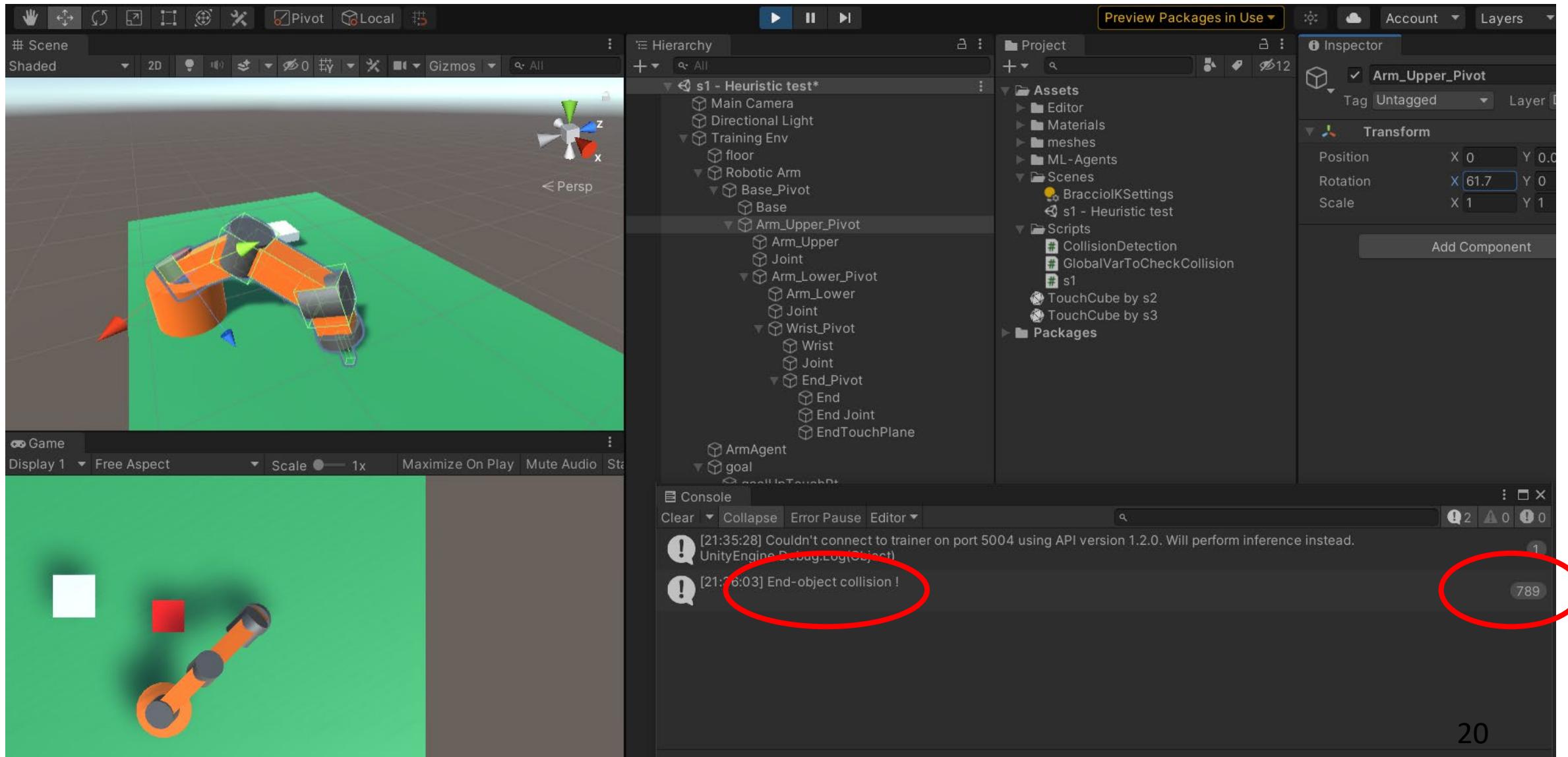
Add trigger collider, Rigid body, and collision detection script to Robot End



Add trigger collider, Rigid body, and collision detection script to goal object

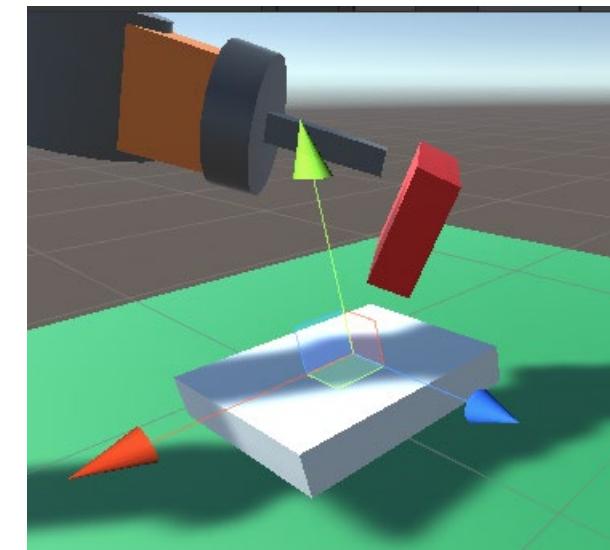
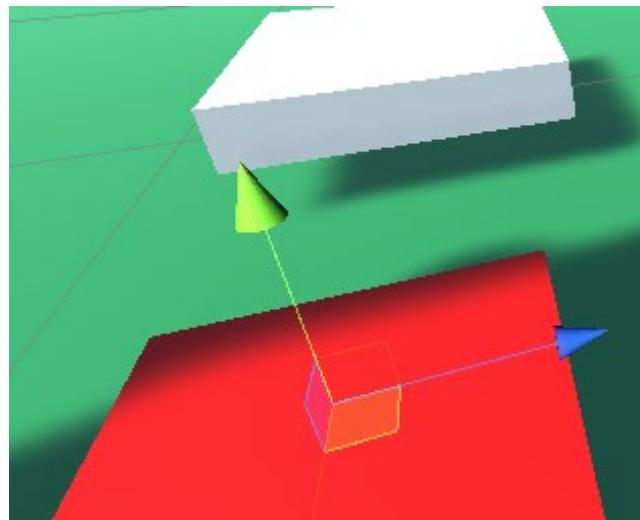
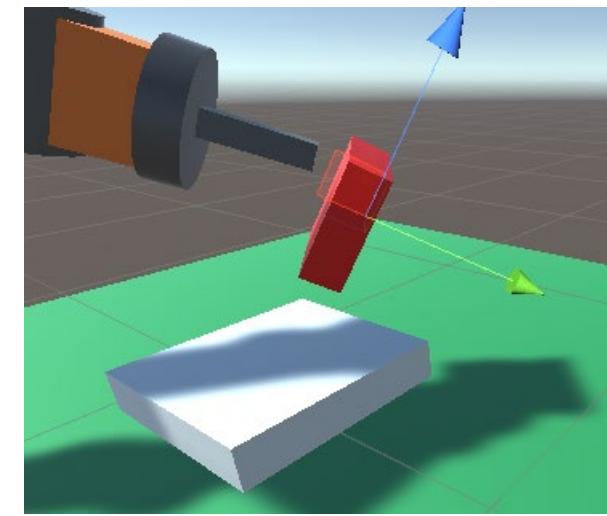
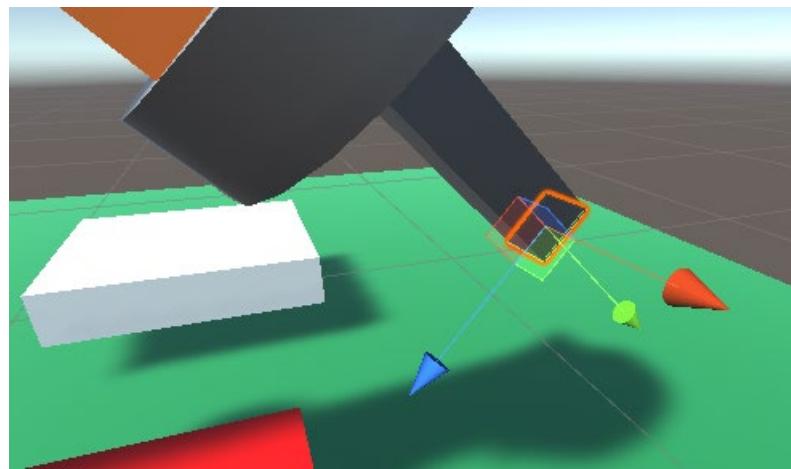
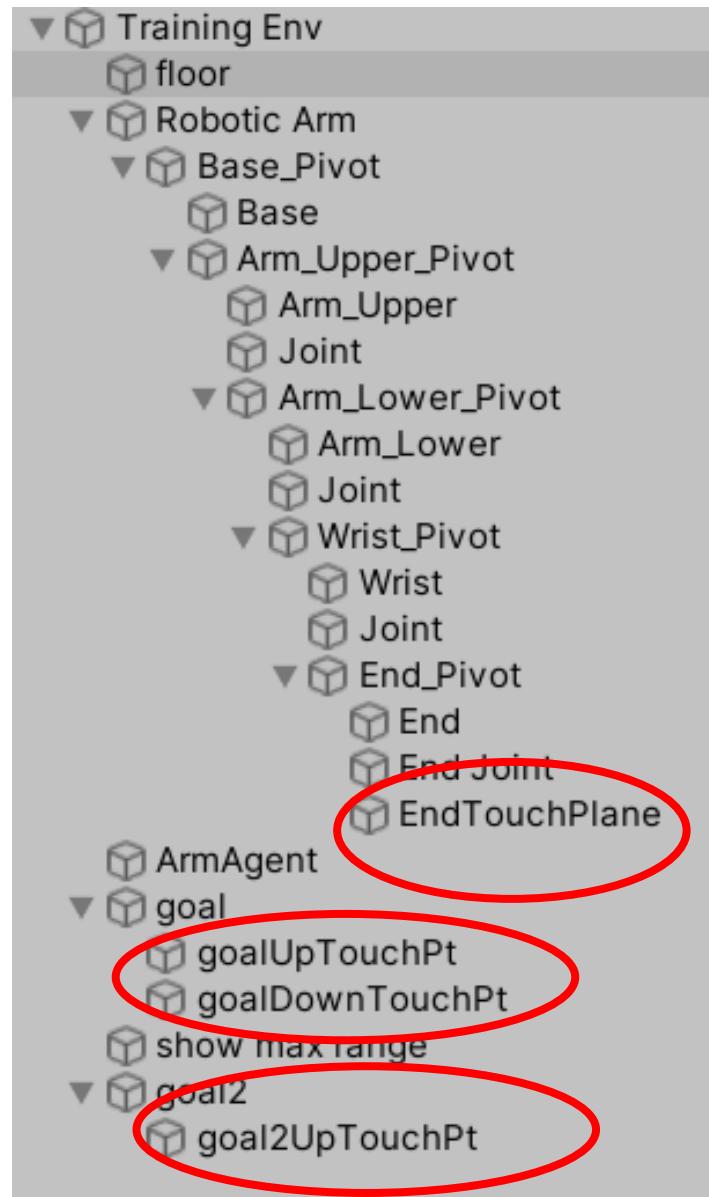


Control the robot arm to collide with the floor or goal and check the error message



Point-based touch detection

Assign points for reach-detection

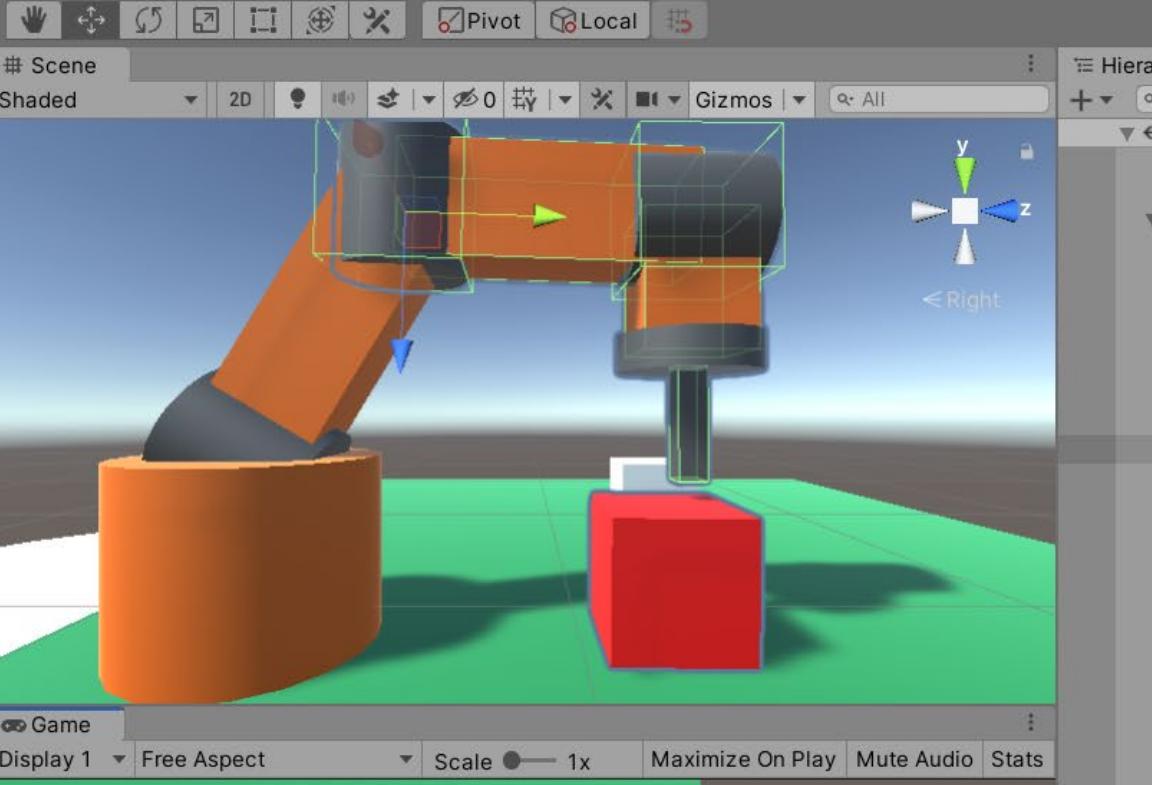


Point-based touch detection

```
bool PointTouch(Transform pt1, Transform pt2, float threshold)
{
    //string msg;
    float dx, dy, dz;

    dx = Mathf.Abs(pt1.position.x - pt2.position.x);
    dz = Mathf.Abs(pt1.position.z - pt2.position.z);
    dy = pt1.position.y - pt2.position.y;
    //msg = dx.ToString() + ", " + dy.ToString() + ", " + dz.ToString();
    //print(msg);
    if (dy > 0 && dy < threshold && dx < threshold && dz < threshold)
        return true;
    else
        return false;
}
```

Manually control robot arm to touch goal (avoid collision!)



The Unity Editor interface is shown, featuring the Scene View, Hierarchy View, Project View, Inspector View, and Console View.

Scene View: Displays a 3D scene with a robotic arm (orange and black) positioned above a red cube. A green wireframe cube indicates the current position of the arm's end effector.

Hierarchy View: Shows the project structure with scenes like "s5 - Heuristic play - accur" and objects like "Base_Pivot", "Arm_Upper_Pivot", "Arm_Lower_Pivot", and "goal".

Inspector View: Shows the transform components for the "Arm_Lower_Pivot" object. The "Scale" field for the X-axis is highlighted with a red circle.

Console View: Displays log messages, with the last two entries circled in red:

- [16:28:18] Goal 1!
- [16:28:18] 0.09907154

Text Labels:

- 2DOF using keyboards**: Located in the bottom-left corner of the Scene View.
- Base: ← and →**: Control instructions for the base joint.
- wrist: ↑ and ↓**: Control instructions for the wrist joint.
- 2DOF using Inspector window**: Located in the middle-right area of the screen.
- Adjust Upper/Lower arm in Inspector window**: Instruction for manually controlling the arm segments.

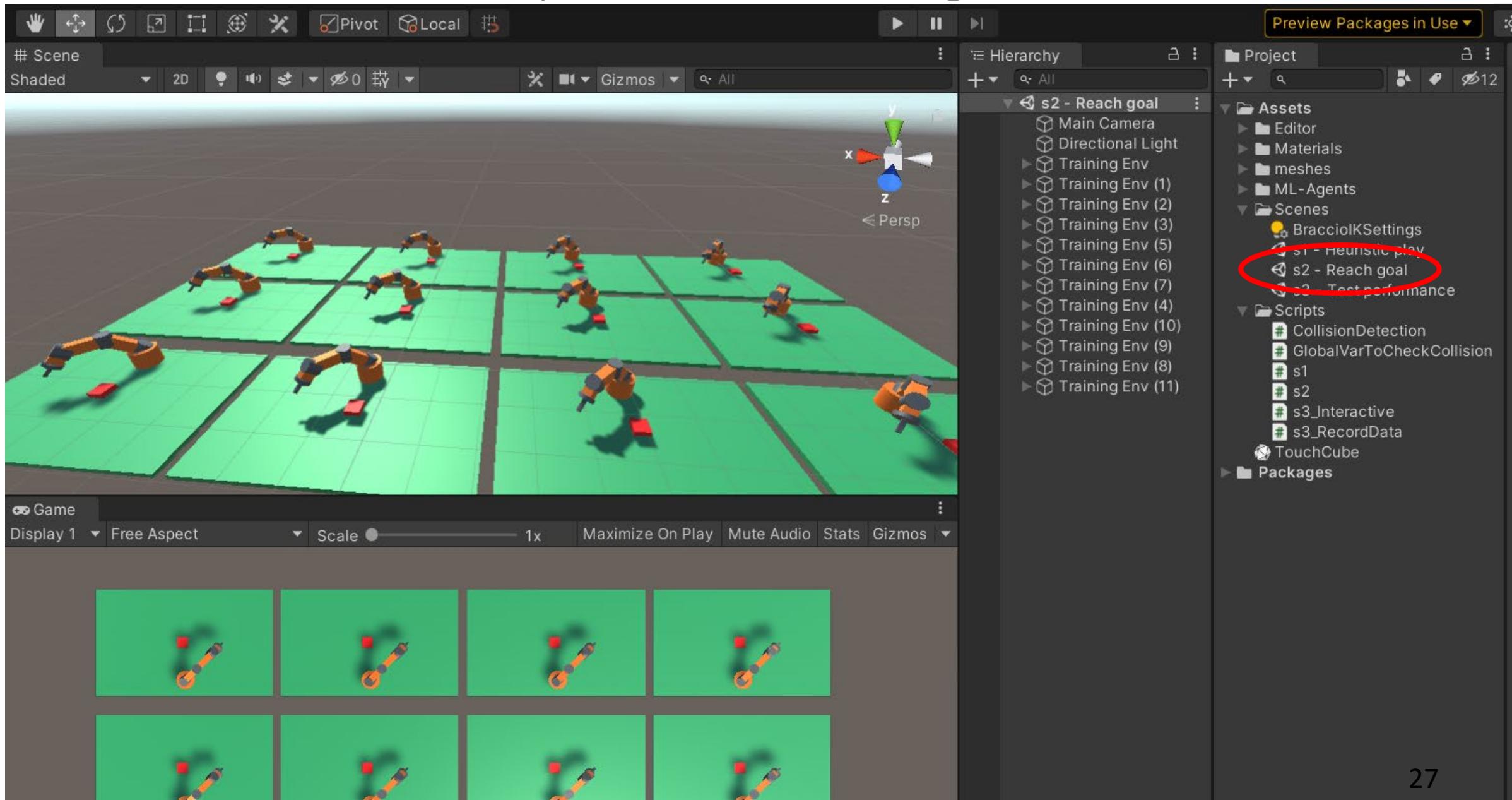
(2) Train robot arm to reach goal

HW4(1)

- Describe the training environment ("s2 – Reach goal")
- Describe the agent script (s, a, r)
- Show tensor board plots and discuss your training performance
- Describe your test performance



4. Open "s2 – Reach goal"



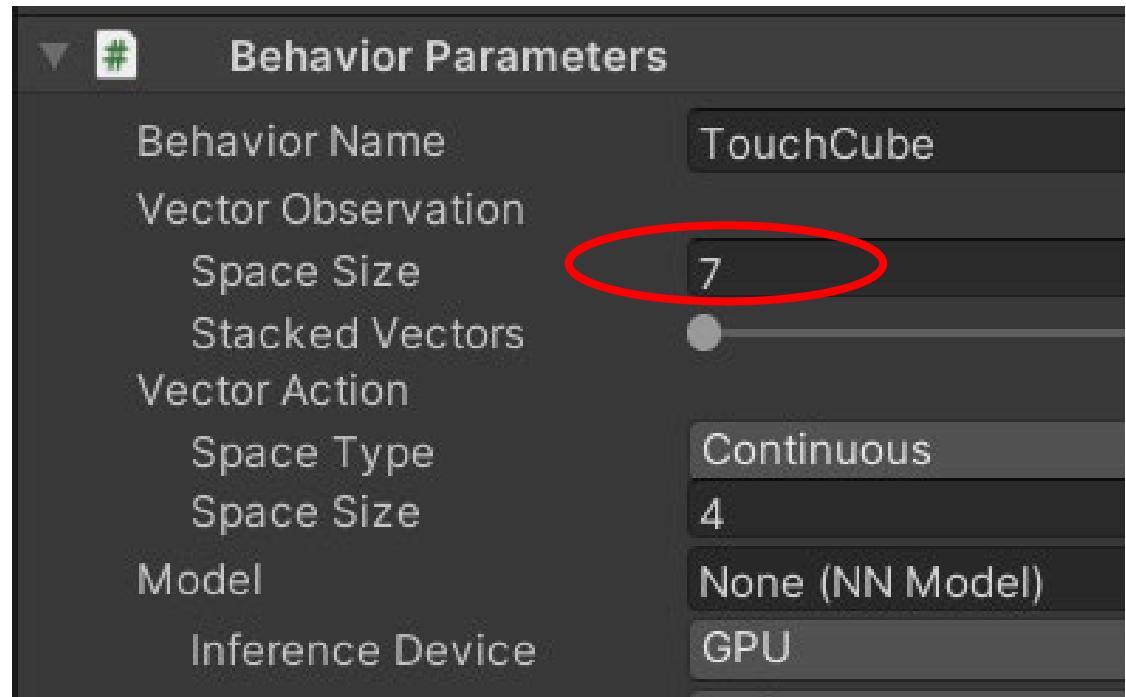
Use polar system to randomly place goals

```
float radius = UnityEngine.Random.Range(radiusMin, radiusMax);
float theta = (UnityEngine.Random.Range(thetaMin, thetaMax) / 180.0f) * Mathf.PI;
    to radians
float x = radius * Mathf.Sin(theta);
float z = radius * Mathf.Cos(theta);
return new Vector3(x, y, z);
```

State has 7 variables

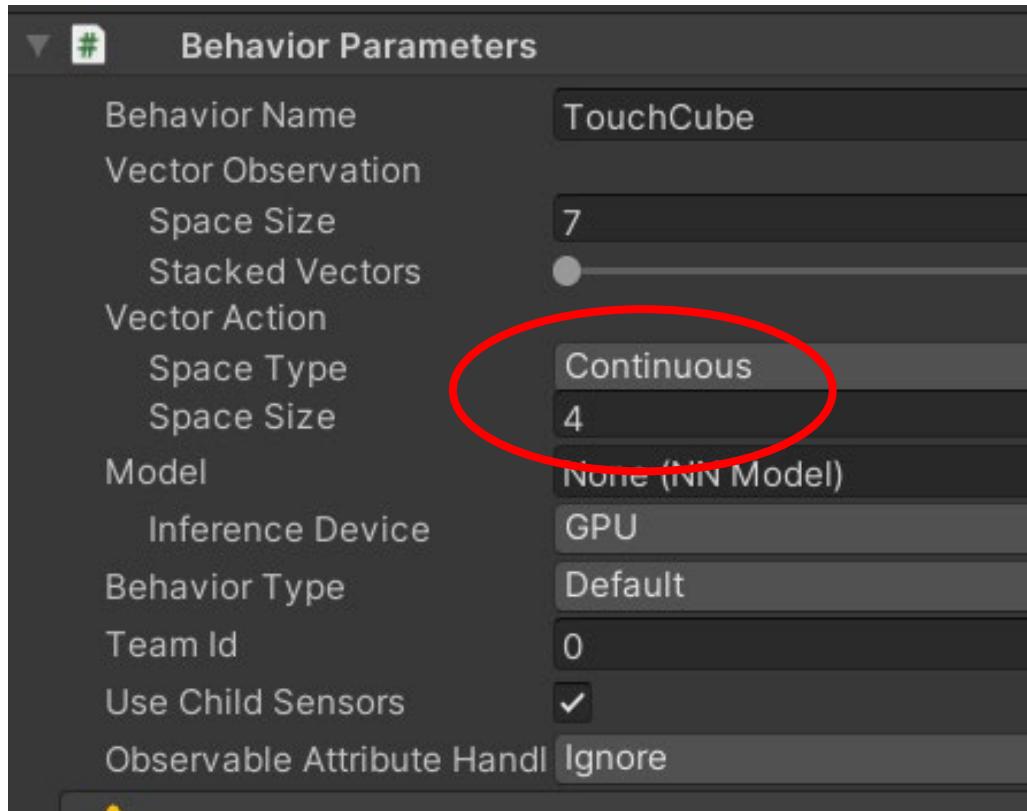
```
sensor.AddObservation(EndTouchPlane.position - goalUpTouchPt.transform.position);  
  
float BaseRotationAngle = UnityEditor.TransformUtils.GetInspectorRotation(BasePivot).y;  
float UArmRotationAngle = UnityEditor.TransformUtils.GetInspectorRotation(UpperPivot).x;  
float LArmRotationAngle = UnityEditor.TransformUtils.GetInspectorRotation(LowerPivot).x;  
float WRotationAngle = UnityEditor.TransformUtils.GetInspectorRotation(WristPivot).x;
```

```
sensor.AddObservation(BaseRotationAngle);  
sensor.AddObservation(UArmRotationAngle);  
sensor.AddObservation(LArmRotationAngle);  
sensor.AddObservation(WRotationAngle);
```



Action has 4 values

```
BasePivot.Rotate(0, vectorAction[0] * speed, 0);  
UpperPivot.Rotate(vectorAction[1] * speed, 0, 0);  
LowerPivot.Rotate(vectorAction[2] * speed, 0, 0);  
WristPivot.Rotate(vectorAction[3] * speed, 0, 0);
```



Reward

Punish every step to
avoid 耍廢

```
float speed = 1.0f;  
AddReward(-0.005f);  
string msg;
```

HW

What will happen if do
not end episode ?

```
if (!Rotation_in_range())  
{  
    /*  
    msg = System.DateTime.Now.ToShortTimeString();  
    msg = msg + trainingVE.name + "Angle out of range error ! \n";  
    print(msg); */  
    AddReward(-5.0f);  
    EndEpisode();  
}  
  
if (LowerArmObj.GetComponent<CollisionDetection>().CollisionHappen ||  
    WristObj.GetComponent<CollisionDetection>().CollisionHappen ||  
    EndObj.GetComponent<CollisionDetection>().CollisionHappen)  
{  
    /*  
    msg = System.DateTime.Now.ToShortTimeString();  
    msg = msg + trainingVE.name + MyGlobalVar.LowerArmCollisionHappens.  
    msg = msg + MyGlobalVar.WristCollisionHappens.ToString() + ", " +  
        MyGlobalVar.EndCollisionHappens.ToString() + ", " +  
        MyGlobalVar.goalCollisionHappens.ToString()+"\n";  
    print(msg);*/  
    AddReward(-5.0f);  
    EndEpisode();  
}
```

Reward when reach goal

HW

- 0.5 will succeed, but the behavior is too rough
- 0.05 is too difficult and training will fail
- In your HW, try 0.25 or 0.1?

```
if(PointTouch(EndTouchPlane, goalUpTouchPt, 0.5f))  
{  
    msg = System.DateTime.Now.ToString("T");  
    msg = msg + trainingVE.name + " Goal 1! \n";  
    print(msg);  
    AddReward(20.0f);  
    EndEpisode();  
}
```

Training configuration file

```
TouchCube:  
    trainer_type: ppo  
    hyperparameters:  
        batch_size: 2048  
        buffer_size: 20480  
        learning_rate: 0.0003  
        beta: 0.001  
        epsilon: 0.2  
        lambd: 0.95  
        num_epoch: 3  
        learning_rate_schedule:  
            network_settings:  
                normalize: true  
                hidden_units: 512  
                num_layers: 3  
                vis_encode_type: s  
            reward_signals:  
                extrinsic:  
                    gamma: 0.995  
                    strength: 1.0  
    keep_checkpoints: 5  
    max_steps: 5000000  
    time_horizon: 2000  
    summary_freq: 30000  
    threaded: true
```

Print environment number that reaches goal



```
msg = System.DateTime.Now.ToString("HH:mm:ss");
msg = msg + trainingVE.name + " Goal 1! ==> " + distToGoal.ToString() + "\n";
print(msg);
AddReward(20.0f);
EndEpisode();
```

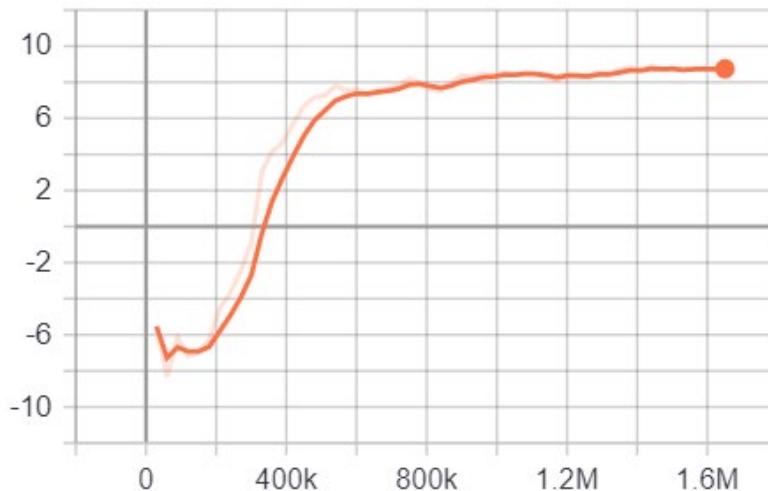
```
[16:38:06] 下午 04:38Training Env Goal 1! ==> 0.05521972
[16:38:07] 下午 04:38Training Env (10) Goal 1! ==> 0.09773364
[16:38:08] 下午 04:38Training Env (6) Goal 1! ==> 0.08763794
[16:38:09] 下午 04:38Training Env (6) Goal 1! ==> 0.09873476
[16:38:22] 下午 04:38Training Env (5) Goal 1! ==> 0.08890654
[16:38:22] 下午 04:38Training Env (4) Goal 1! ==> 0.09917516
[16:38:44] 下午 04:38Training Env (9) Goal 1! ==> 0.09907977
[16:38:45] 下午 04:38Training Env Goal 1! ==> 0.09666377
```

Results after 1.7M steps, looks promising

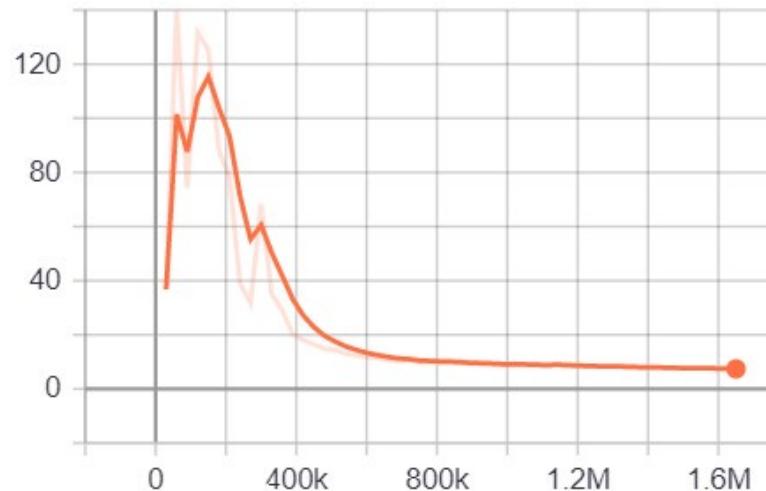
```
TouchCube. Step: 510000. Time Elapsed: 667.226 s. Mean Reward: 7.267. Std of Reward: 12.186. Train
TouchCube. Step: 540000. Time Elapsed: 707.059 s. Mean Reward: 7.803. Std of Reward: 12.211. Train
TouchCube. Step: 570000. Time Elapsed: 755.981 s. Mean Reward: 7.567. Std of Reward: 12.226. Train
TouchCube. Step: 600000. Time Elapsed: 796.691 s. Mean Reward: 7.605. Std of Reward: 12.246. Train
TouchCube. Step: 630000. Time Elapsed: 848.452 s. Mean Reward: 7.315. Std of Reward: 12.247. Train
TouchCube. Step: 660000. Time Elapsed: 890.687 s. Mean Reward: 7.594. Std of Reward: 12.260. Train
TouchCube. Step: 690000. Time Elapsed: 947.977 s. Mean Reward: 7.625. Std of Reward: 12.267. Train
TouchCube. Step: 720000. Time Elapsed: 988.268 s. Mean Reward: 7.792. Std of Reward: 12.319. Train
TouchCube. Step: 750000. Time Elapsed: 1037.415 s. Mean Reward: 8.202. Std of Reward: 12.260. Train
TouchCube. Step: 780000. Time Elapsed: 1088.936 s. Mean Reward: 7.957. Std of Reward: 12.165. Train
TouchCube. Step: 810000. Time Elapsed: 1141.701 s. Mean Reward: 7.619. Std of Reward: 12.277. Train
TouchCube. Step: 840000. Time Elapsed: 1182.285 s. Mean Reward: 7.524. Std of Reward: 12.280. Train
TouchCube. Step: 870000. Time Elapsed: 1222.890 s. Mean Reward: 7.983. Std of Reward: 12.273. Train
TouchCube. Step: 900000. Time Elapsed: 1274.109 s. Mean Reward: 8.380. Std of Reward: 12.261. Train
TouchCube. Step: 930000. Time Elapsed: 1315.381 s. Mean Reward: 8.268. Std of Reward: 12.277. Train
TouchCube. Step: 960000. Time Elapsed: 1366.218 s. Mean Reward: 8.481. Std of Reward: 12.253. Train
TouchCube. Step: 990000. Time Elapsed: 1413.950 s. Mean Reward: 8.361. Std of Reward: 12.270. Train
ation.py:93] Converting to results\1\TouchCube\TouchCube-999992.onnx
ation.py:105] Exported results\1\TouchCube\TouchCube-999992.onnx
TouchCube. Step: 1020000. Time Elapsed: 1468.478 s. Mean Reward: 8.561. Std of Reward: 12.284. Train
TouchCube. Step: 1050000. Time Elapsed: 1512.854 s. Mean Reward: 8.413. Std of Reward: 12.279. Train
TouchCube. Step: 1080000. Time Elapsed: 1563.380 s. Mean Reward: 8.533. Std of Reward: 12.259. Train
TouchCube. Step: 1110000. Time Elapsed: 1605.463 s. Mean Reward: 8.456. Std of Reward: 12.268. Train
TouchCube. Step: 1140000. Time Elapsed: 1654.158 s. Mean Reward: 8.288. Std of Reward: 12.286. Train
TouchCube. Step: 1170000. Time Elapsed: 1696.093 s. Mean Reward: 8.091. Std of Reward: 12.291. Train
TouchCube. Step: 1200000. Time Elapsed: 1746.003 s. Mean Reward: 8.472. Std of Reward: 12.270. Train
TouchCube. Step: 1230000. Time Elapsed: 1787.666 s. Mean Reward: 8.369. Std of Reward: 12.276. Train
TouchCube. Step: 1260000. Time Elapsed: 1837.945 s. Mean Reward: 8.287. Std of Reward: 12.287. Train
TouchCube. Step: 1290000. Time Elapsed: 1879.433 s. Mean Reward: 8.588. Std of Reward: 12.260. Train
TouchCube. Step: 1320000. Time Elapsed: 1921.159 s. Mean Reward: 8.455. Std of Reward: 12.275. Train
TouchCube. Step: 1350000. Time Elapsed: 1971.810 s. Mean Reward: 8.685. Std of Reward: 12.258. Train
TouchCube. Step: 1380000. Time Elapsed: 2013.756 s. Mean Reward: 8.849. Std of Reward: 12.244. Train
TouchCube. Step: 1410000. Time Elapsed: 2063.943 s. Mean Reward: 8.627. Std of Reward: 12.264. Train
TouchCube. Step: 1440000. Time Elapsed: 2105.873 s. Mean Reward: 8.891. Std of Reward: 12.241. Train
TouchCube. Step: 1470000. Time Elapsed: 2156.424 s. Mean Reward: 8.702. Std of Reward: 12.264. Train
TouchCube. Step: 1500000. Time Elapsed: 2198.716 s. Mean Reward: 8.772. Std of Reward: 12.257. Train
```

Results after 1.7M steps, looks promising

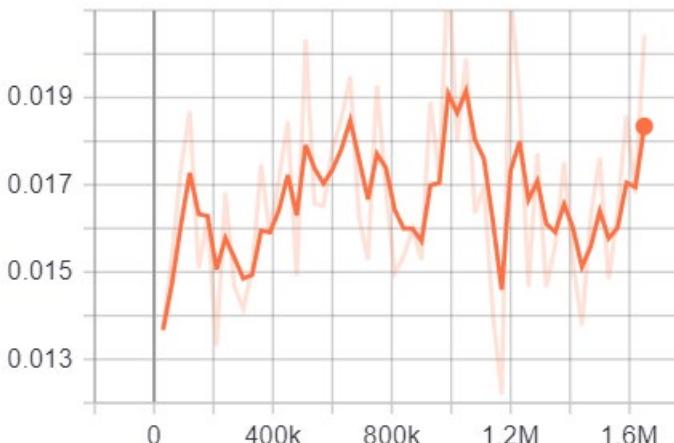
Cumulative Reward
tag: Environment/Cumulative Reward



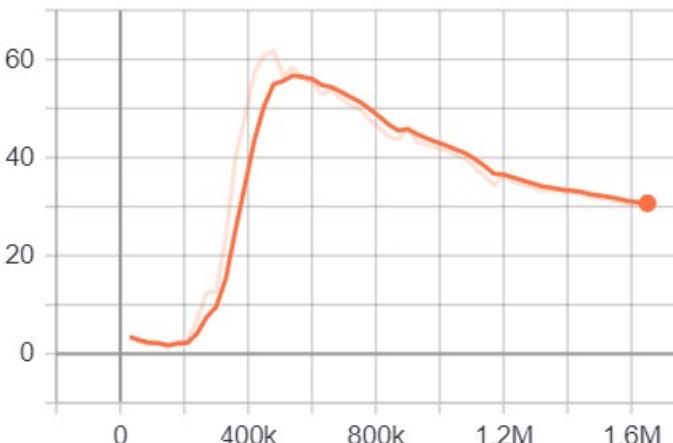
Episode Length
tag: Environment/Episode Length



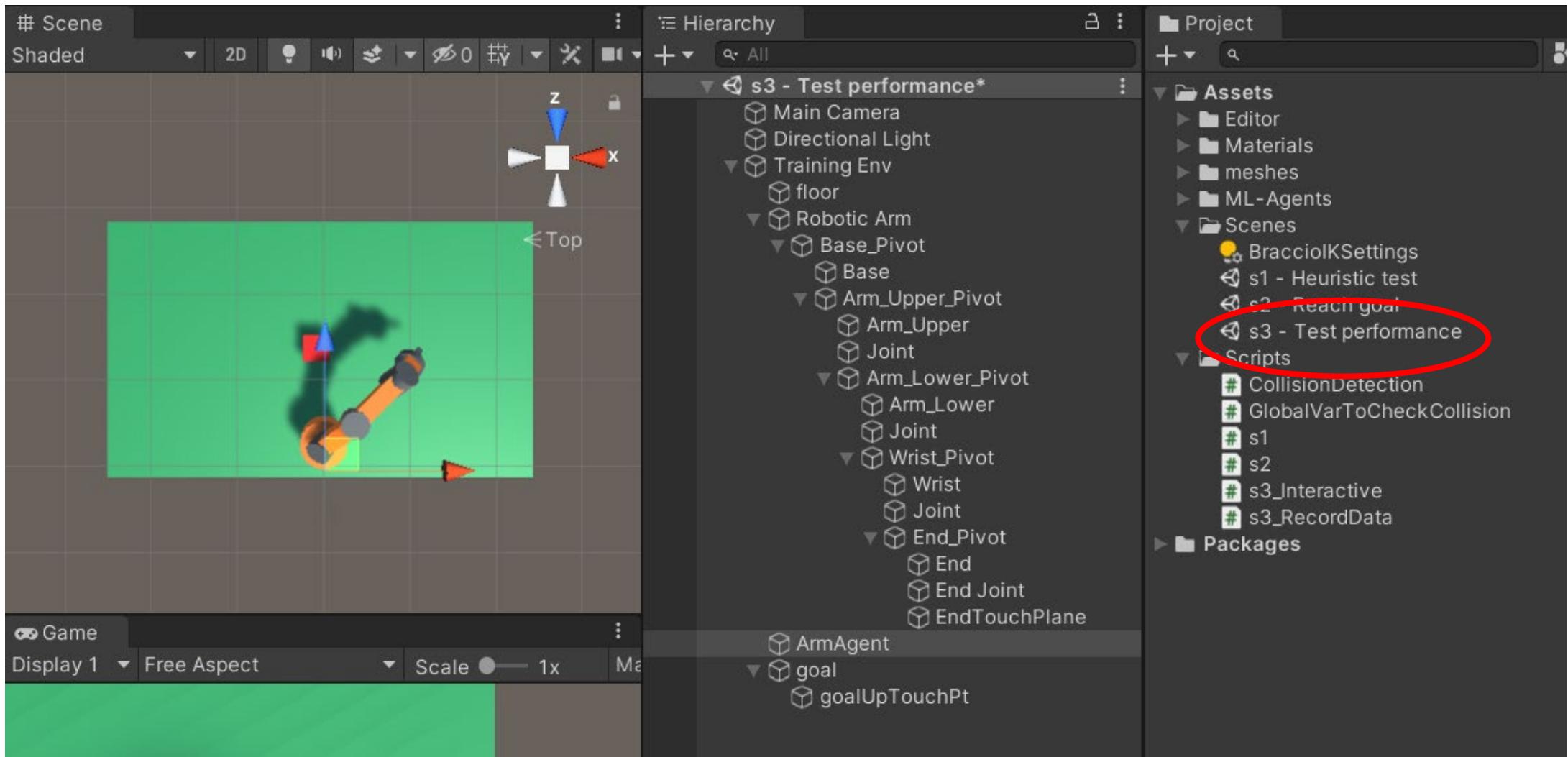
Policy Loss
tag: Losses/Policy Loss



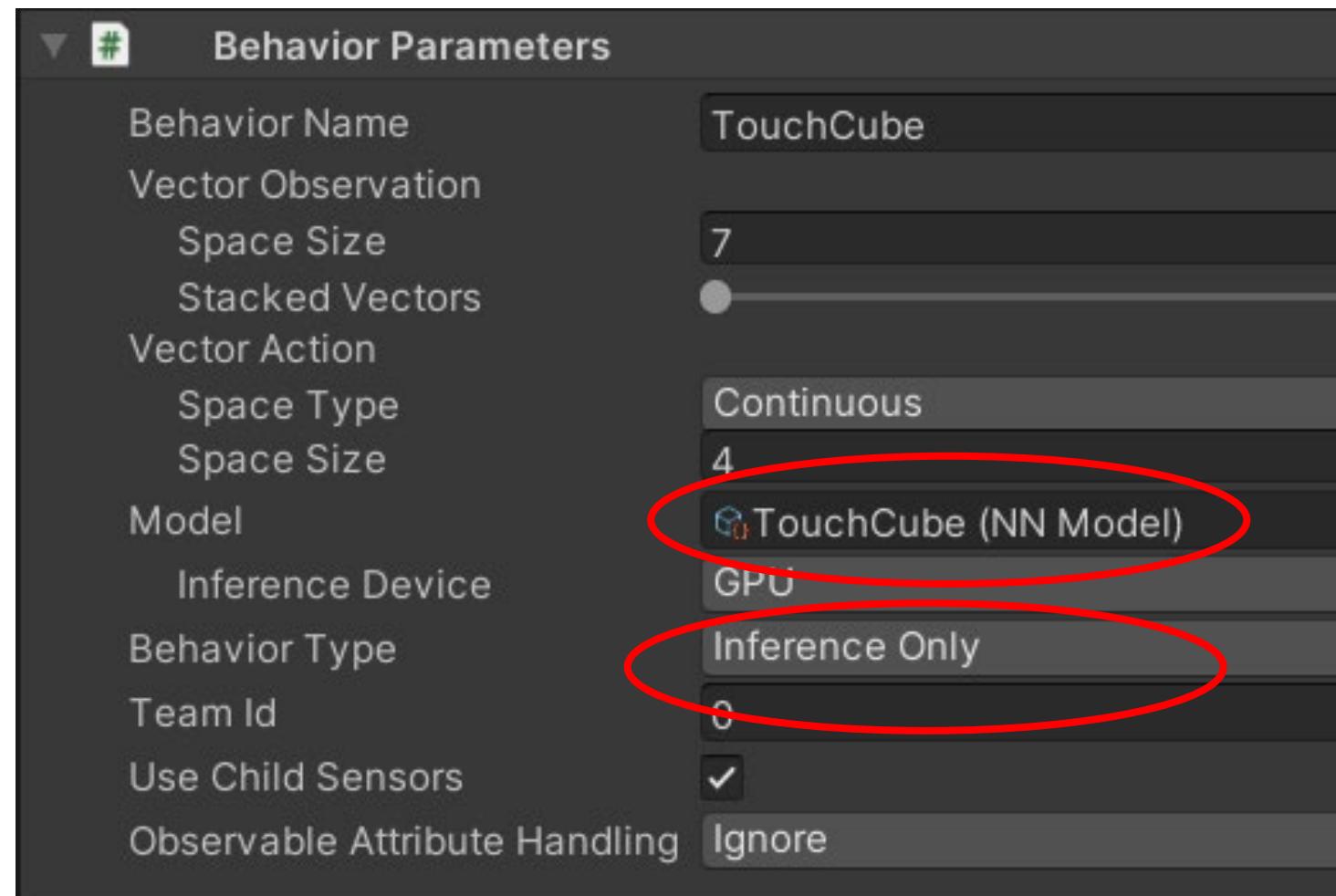
Value Loss
tag: Losses/Value Loss



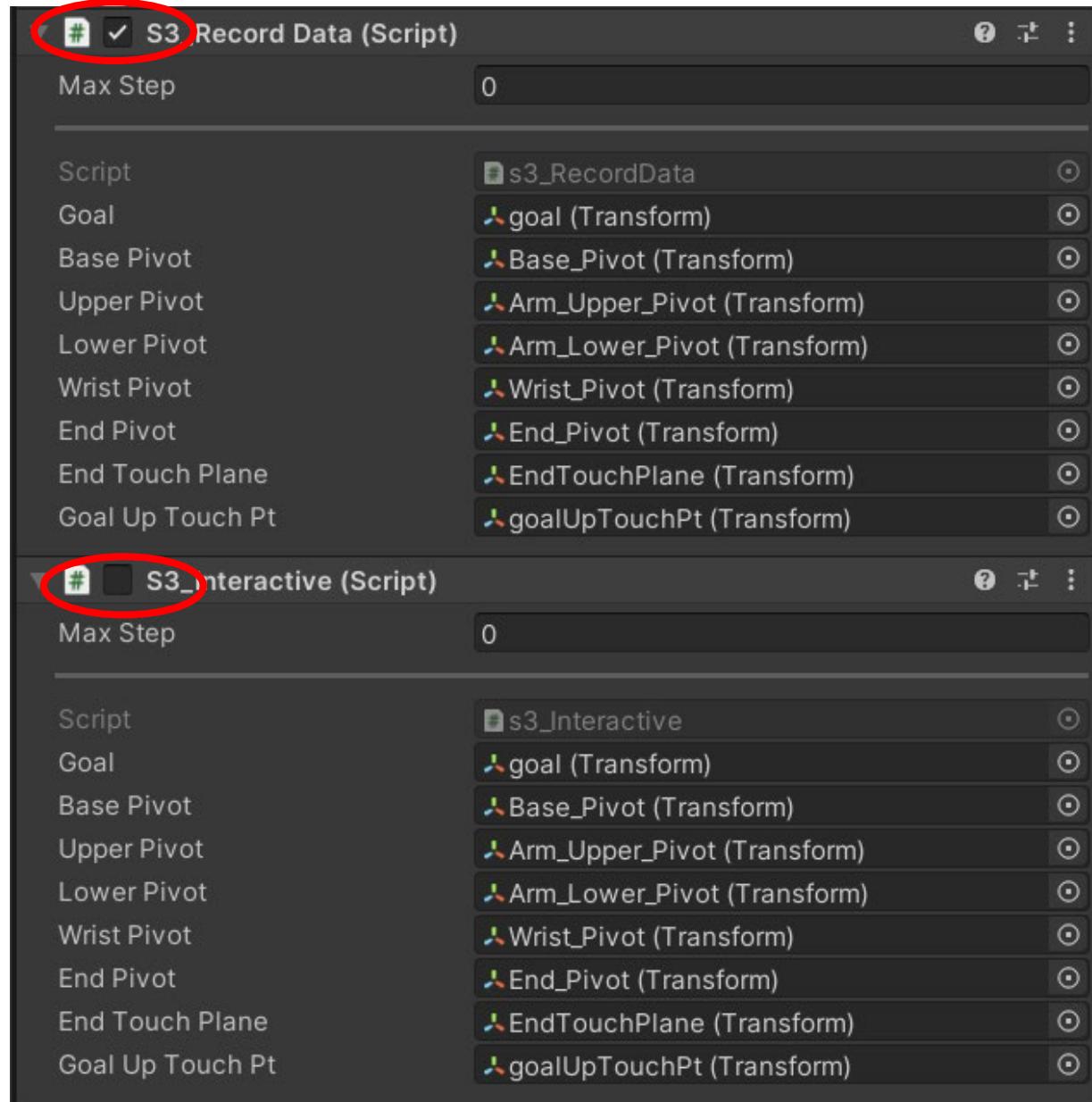
Open "s3 – Test performance"



Assign trained NN



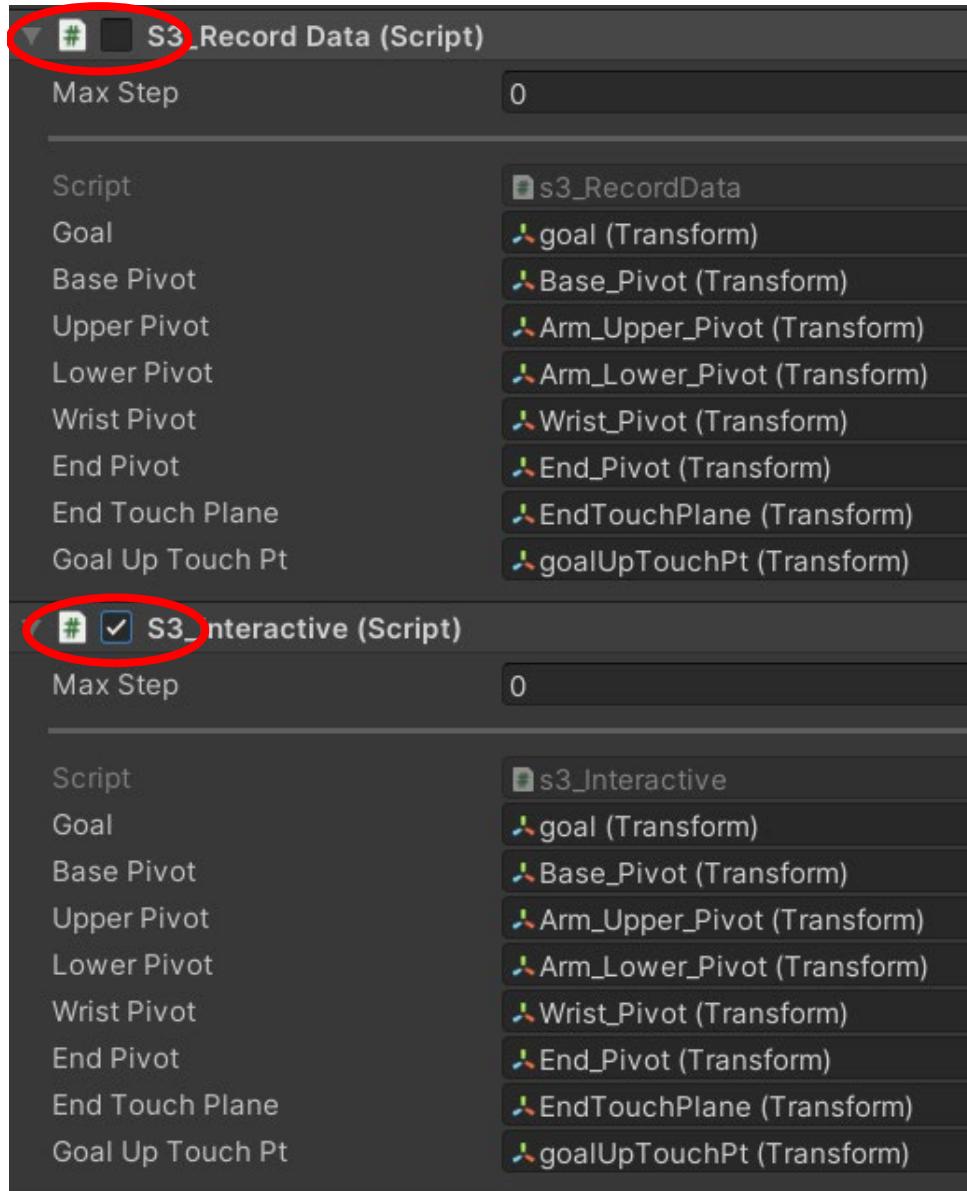
2 types of tests – (1) data recording



Uncheck interactive test

2 types of tests – (2) interactive test

Uncheck data-recording
test



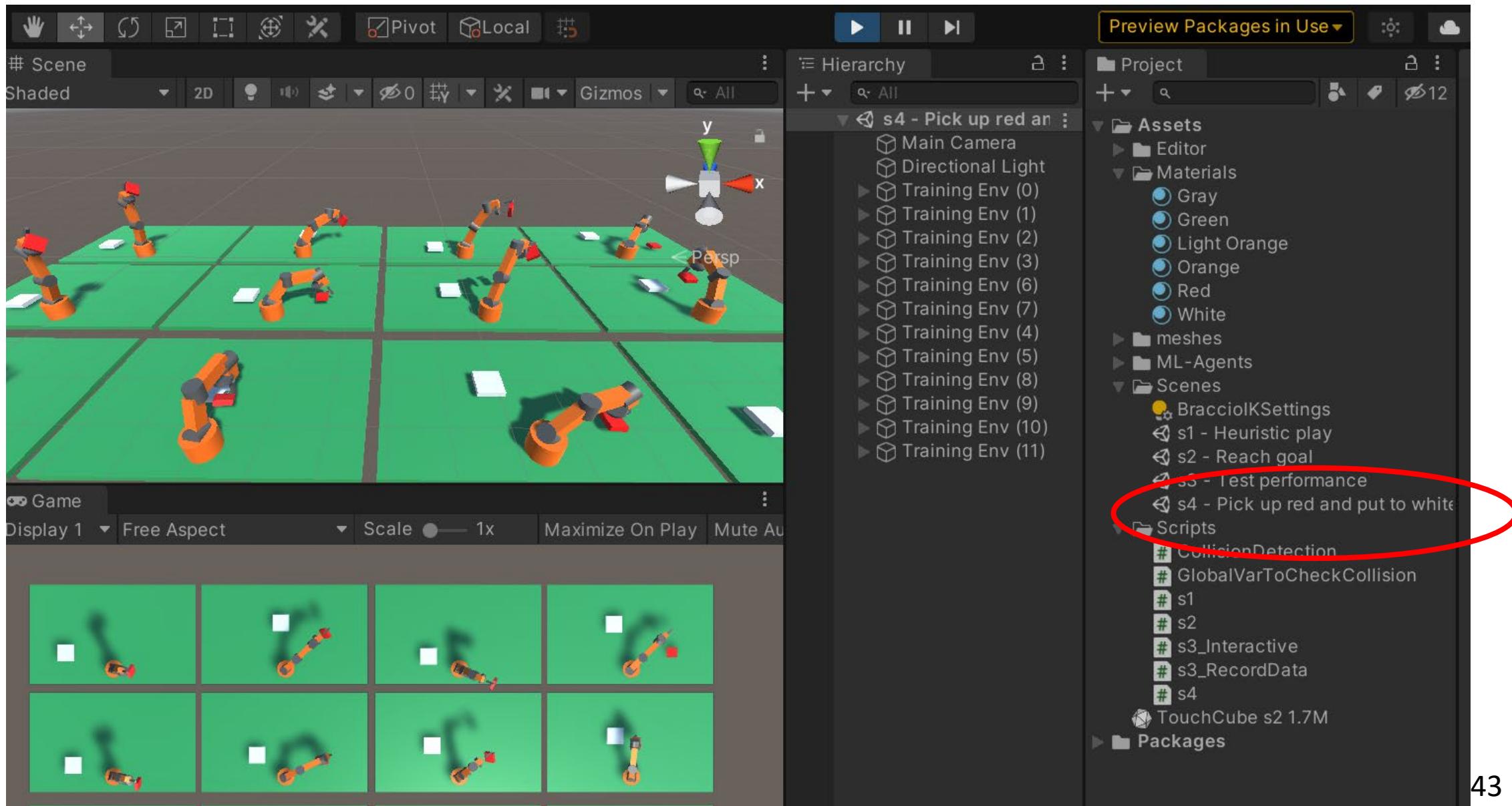
(3) Train robot arm to pick up the red cube and place it on top of the white cube

HW4(2)

- Describe the training environment ("s4 – Pick up red and put to white")
- Describe the agent script (s, a, r)
- Show tensor board plots and discuss your training performance
- Describe your test performance



Open "s4 – Pick up red and put to white"



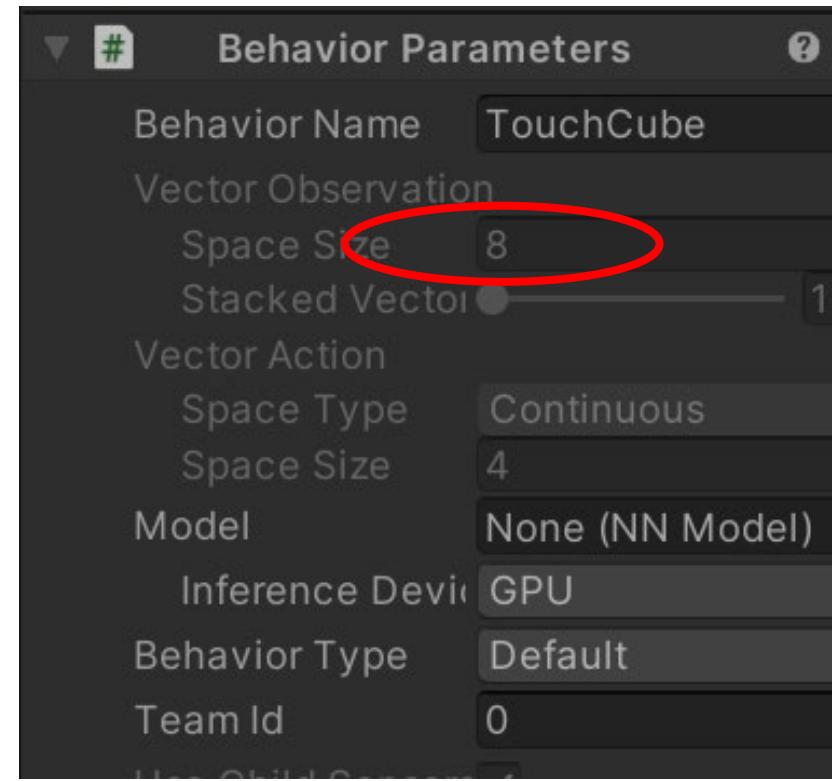
State has 8 variables

```
sensor.AddObservation(stage);

if (stage == 1)
    sensor.AddObservation(EndTouchPlane.position - goalUpTouchPt.posi
else //stage =2
    sensor.AddObservation(goalDownTouchPt.position - goal2UpTouchPt.p

float BaseRotationAngle = UnityEditor.TransformUtils.GetInspectorRota
float UArmRotationAngle = UnityEditor.TransformUtils.GetInspectorRota
float LArmRotationAngle = UnityEditor.TransformUtils.GetInspectorRota
float WRotationAngle = UnityEditor.TransformUtils.GetInspectorRotatio

sensor.AddObservation(BaseRotationAngle);
sensor.AddObservation(UArmRotationAngle);
sensor.AddObservation(LArmRotationAngle);
sensor.AddObservation(WRotationAngle);
```



Action has 4 values

```
BasePivot.Rotate(0, vectorAction[0] * speed, 0);  
UpperPivot.Rotate(vectorAction[1] * speed, 0, 0);  
LowerPivot.Rotate(vectorAction[2] * speed, 0, 0);  
WristPivot.Rotate(vectorAction[3] * speed, 0, 0);
```

4 types of rewards

AddReward(-0.005f); //avoid 要廢

```
if (!Rotation_in_range())
{
    /*
    msg = trainingVE.name +
    print(msg); */
    AddReward(-5.0f);
    EndEpisode();
}

else if (LowerArmObj.GetComponent<CollisionDetection>()
    WristObj.GetComponent<CollisionDetection>()
    EndObj.GetComponent<CollisionDetection>()
    GoalObj.GetComponent<CollisionDetection>()
    print(msg);
    AddReward(-5.0f);
    EndEpisode();
}
```

4 types of rewards

```
// if no rotation out of range and no collision, give awards
if (stage ==1 && PointTouch(EndTouchPlane, goalUpTouchPt , 0.1f))
{
    msg = trainingVE.name + " Goal 1! \n";
    Debug.Log(msg);
    stage = 2;
    AddReward(15.0f);
    goal.transform.parent = EndPivot.transform; //grab goal
}
else if (PointTouch(goalDownTouchPt , goal2UpTouchPt , 0.3f))
{
    msg = trainingVE.name + " Goal 2! \n";
    Debug.Log(msg);
    AddReward(100.0f);
    EndEpisode();
}
```

Training configuration file

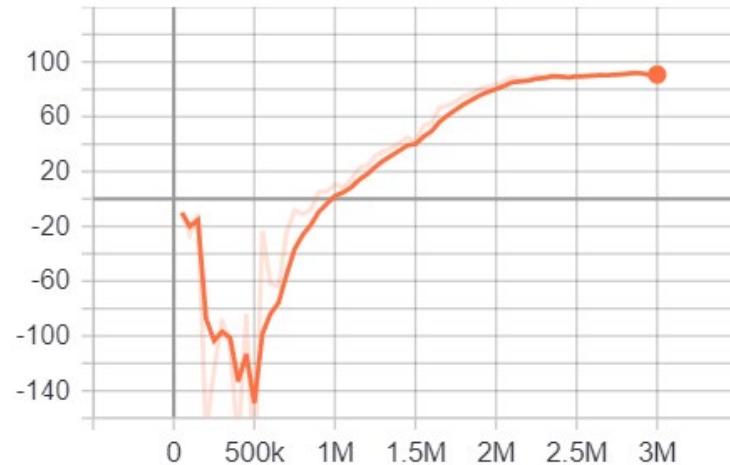
```
TouchCube:  
    trainer_type: ppo  
    hyperparameters:  
        batch_size: 2048  
        buffer_size: 20480  
        learning_rate: 0.0003  
        beta: 0.001  
        epsilon: 0.2  
        lambd: 0.95  
        num_epoch: 3  
        learning_rate_schedule:  
            network_settings:  
                normalize: true  
                hidden_units: 512  
                num_layers: 3  
                vis_encode_type: s  
            reward_signals:  
                extrinsic:  
                    gamma: 0.995  
                    strength: 1.0  
    keep_checkpoints: 5  
    max_steps: 5000000  
    time_horizon: 2000  
    summary_freq: 30000  
    threaded: true
```

I quit at 3M, looks promising

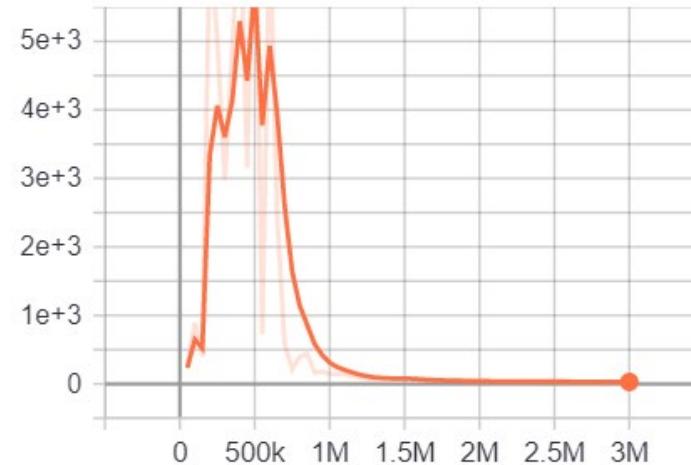
```
TouchCube. Step: 1400000. Time Elapsed: 1521.994 s. Mean Reward: 40.746. Std of Reward: 56.700. Training.  
TouchCube. Step: 1450000. Time Elapsed: 1596.218 s. Mean Reward: 44.816. Std of Reward: 57.113. Training.  
TouchCube. Step: 1500000. Time Elapsed: 1654.829 s. Mean Reward: 41.636. Std of Reward: 56.026. Training.  
[atation.py:93] Converting to results\1\TouchCube\TouchCube-1499998.onnx  
[atation.py:105] Exported results\1\TouchCube\TouchCube-1499998.onnx  
TouchCube. Step: 1550000. Time Elapsed: 1710.662 s. Mean Reward: 53.289. Std of Reward: 58.197. Training.  
TouchCube. Step: 1600000. Time Elapsed: 1772.814 s. Mean Reward: 55.664. Std of Reward: 57.972. Training.  
TouchCube. Step: 1650000. Time Elapsed: 1839.616 s. Mean Reward: 66.710. Std of Reward: 57.145. Training.  
TouchCube. Step: 1700000. Time Elapsed: 1905.361 s. Mean Reward: 68.004. Std of Reward: 56.702. Training.  
TouchCube. Step: 1750000. Time Elapsed: 1974.348 s. Mean Reward: 71.199. Std of Reward: 55.970. Training.  
TouchCube. Step: 1800000. Time Elapsed: 2034.998 s. Mean Reward: 75.024. Std of Reward: 54.928. Training.  
TouchCube. Step: 1850000. Time Elapsed: 2093.756 s. Mean Reward: 77.129. Std of Reward: 54.333. Training.  
TouchCube. Step: 1900000. Time Elapsed: 2153.092 s. Mean Reward: 80.611. Std of Reward: 52.868. Training.  
TouchCube. Step: 1950000. Time Elapsed: 2214.371 s. Mean Reward: 82.048. Std of Reward: 52.347. Training.  
TouchCube. Step: 2000000. Time Elapsed: 2273.202 s. Mean Reward: 83.202. Std of Reward: 51.604. Training.  
[atation.py:93] Converting to results\1\TouchCube\TouchCube-1999992.onnx  
[atation.py:105] Exported results\1\TouchCube\TouchCube-1999992.onnx  
TouchCube. Step: 2050000. Time Elapsed: 2335.471 s. Mean Reward: 85.839. Std of Reward: 50.198. Training.  
TouchCube. Step: 2100000. Time Elapsed: 2396.409 s. Mean Reward: 89.061. Std of Reward: 48.055. Training.  
TouchCube. Step: 2150000. Time Elapsed: 2460.368 s. Mean Reward: 86.739. Std of Reward: 49.768. Training.  
TouchCube. Step: 2200000. Time Elapsed: 2521.083 s. Mean Reward: 87.105. Std of Reward: 49.479. Training.  
TouchCube. Step: 2250000. Time Elapsed: 2583.198 s. Mean Reward: 89.747. Std of Reward: 47.747. Training.  
TouchCube. Step: 2300000. Time Elapsed: 2643.755 s. Mean Reward: 89.104. Std of Reward: 48.263. Training.  
TouchCube. Step: 2350000. Time Elapsed: 2705.644 s. Mean Reward: 91.258. Std of Reward: 46.869. Training.  
TouchCube. Step: 2400000. Time Elapsed: 2770.443 s. Mean Reward: 88.986. Std of Reward: 48.436. Training.  
TouchCube. Step: 2450000. Time Elapsed: 2833.605 s. Mean Reward: 88.044. Std of Reward: 49.433. Training.  
TouchCube. Step: 2500000. Time Elapsed: 2895.328 s. Mean Reward: 90.368. Std of Reward: 48.504. Training.  
[atation.py:93] Converting to results\1\TouchCube\TouchCube-2499995.onnx  
[atation.py:105] Exported results\1\TouchCube\TouchCube-2499995.onnx  
TouchCube. Step: 2550000. Time Elapsed: 2956.370 s. Mean Reward: 89.639. Std of Reward: 48.371. Training.  
TouchCube. Step: 2600000. Time Elapsed: 3028.253 s. Mean Reward: 90.657. Std of Reward: 47.175. Training.  
TouchCube. Step: 2650000. Time Elapsed: 3095.190 s. Mean Reward: 90.670. Std of Reward: 47.247. Training.  
TouchCube. Step: 2700000. Time Elapsed: 3157.692 s. Mean Reward: 90.483. Std of Reward: 47.232. Training.  
TouchCube. Step: 2750000. Time Elapsed: 3226.099 s. Mean Reward: 91.113. Std of Reward: 46.861. Training.  
TouchCube. Step: 2800000. Time Elapsed: 3285.665 s. Mean Reward: 91.785. Std of Reward: 46.426. Training.  
TouchCube. Step: 2850000. Time Elapsed: 3349.785 s. Mean Reward: 92.851. Std of Reward: 46.171. Training.  
TouchCube. Step: 2900000. Time Elapsed: 3413.532 s. Mean Reward: 91.497. Std of Reward: 46.714. Training.  
TouchCube. Step: 2950000. Time Elapsed: 3480.496 s. Mean Reward: 89.342. Std of Reward: 48.568. Training.  
TouchCube. Step: 3000000. Time Elapsed: 3545.361 s. Mean Reward: 90.762. Std of Reward: 47.138. Training.
```

I quit at 3M, looks promising

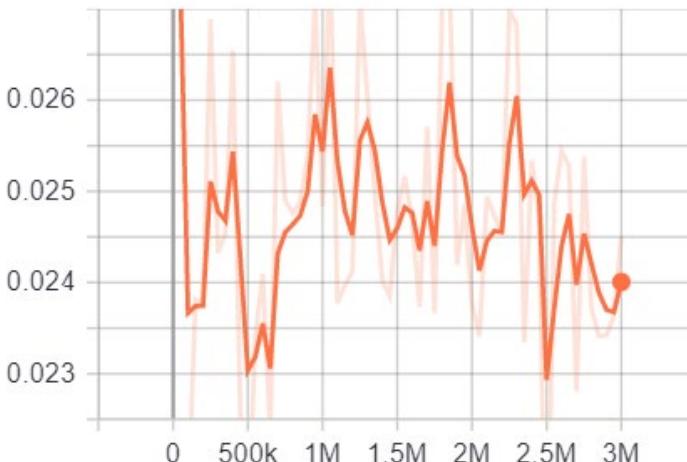
Cumulative Reward
tag: Environment/Cumulative Reward



Episode Length
tag: Environment/Episode Length



Policy Loss
tag: Losses/Policy Loss



Value Loss
tag: Losses/Value Loss

