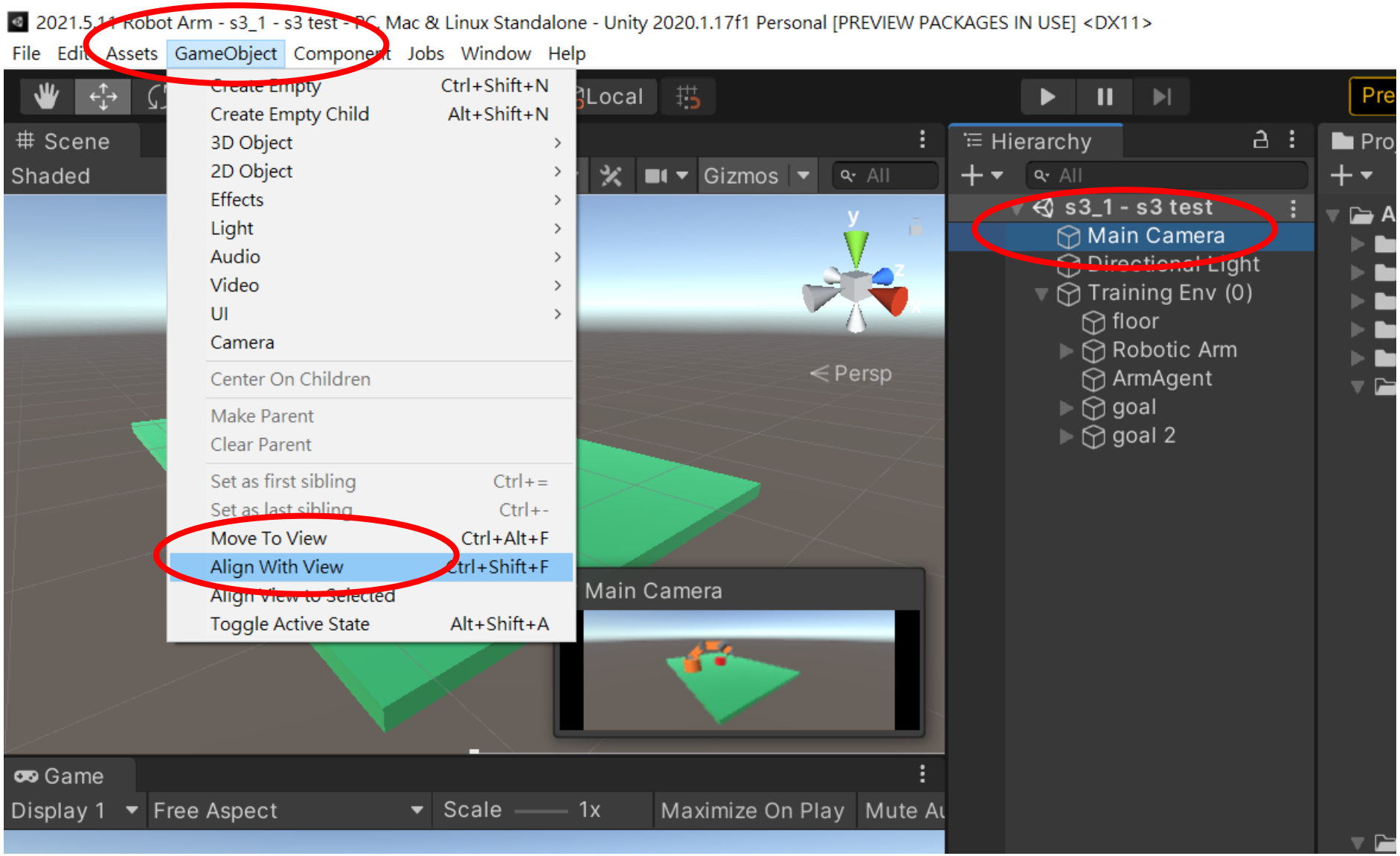
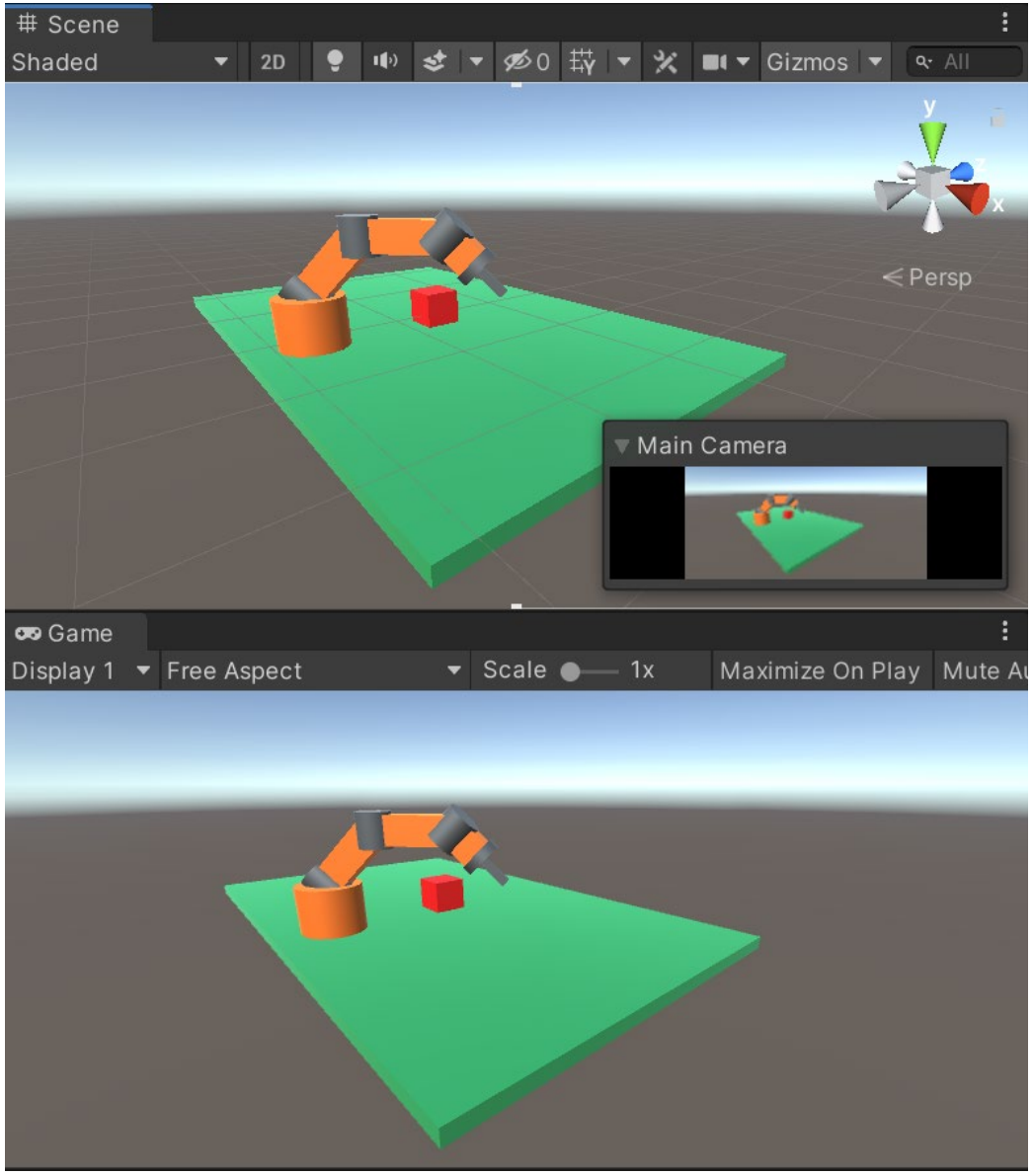
Unity tricks

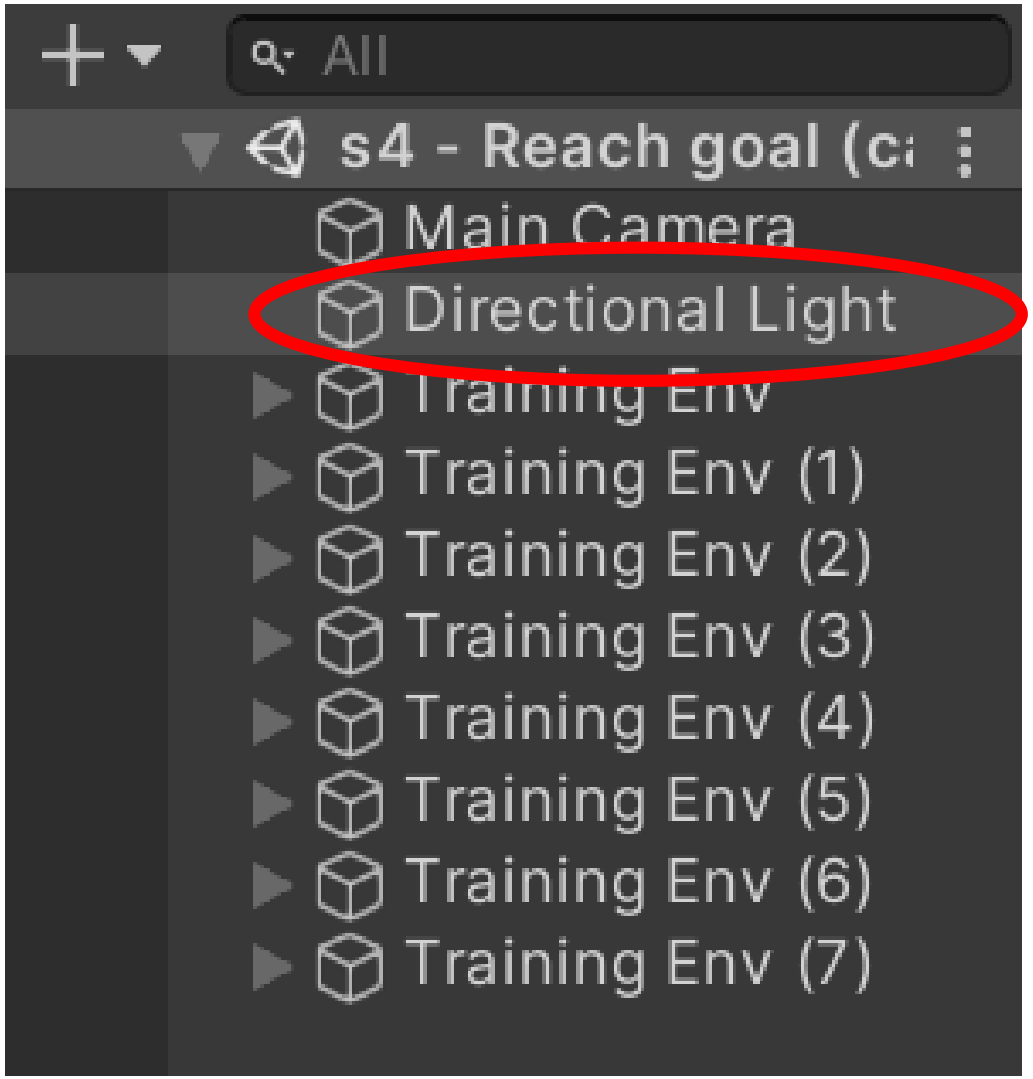
Align main camera with scene view to set Game window view

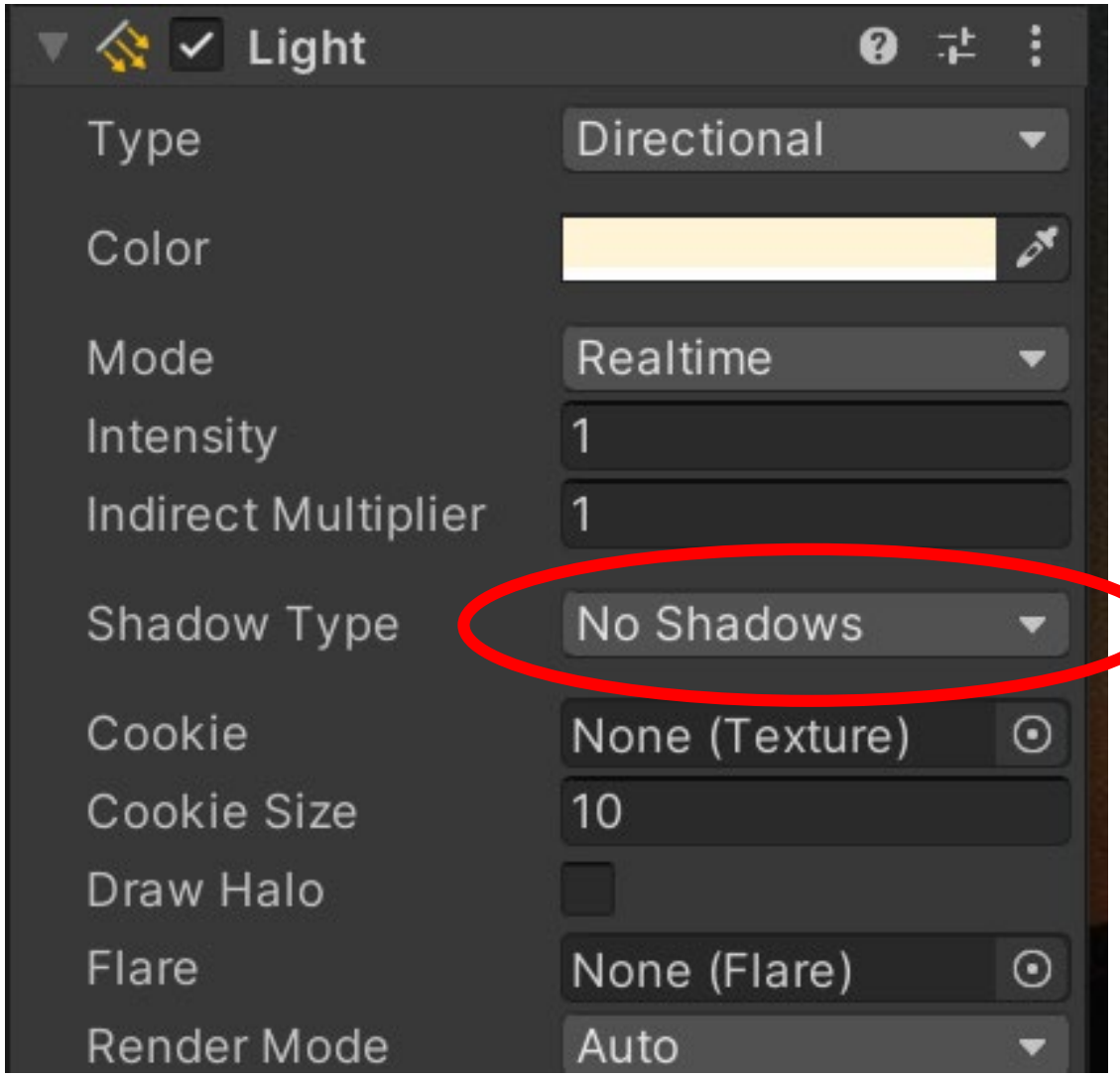


Game window view now is the same as scene view

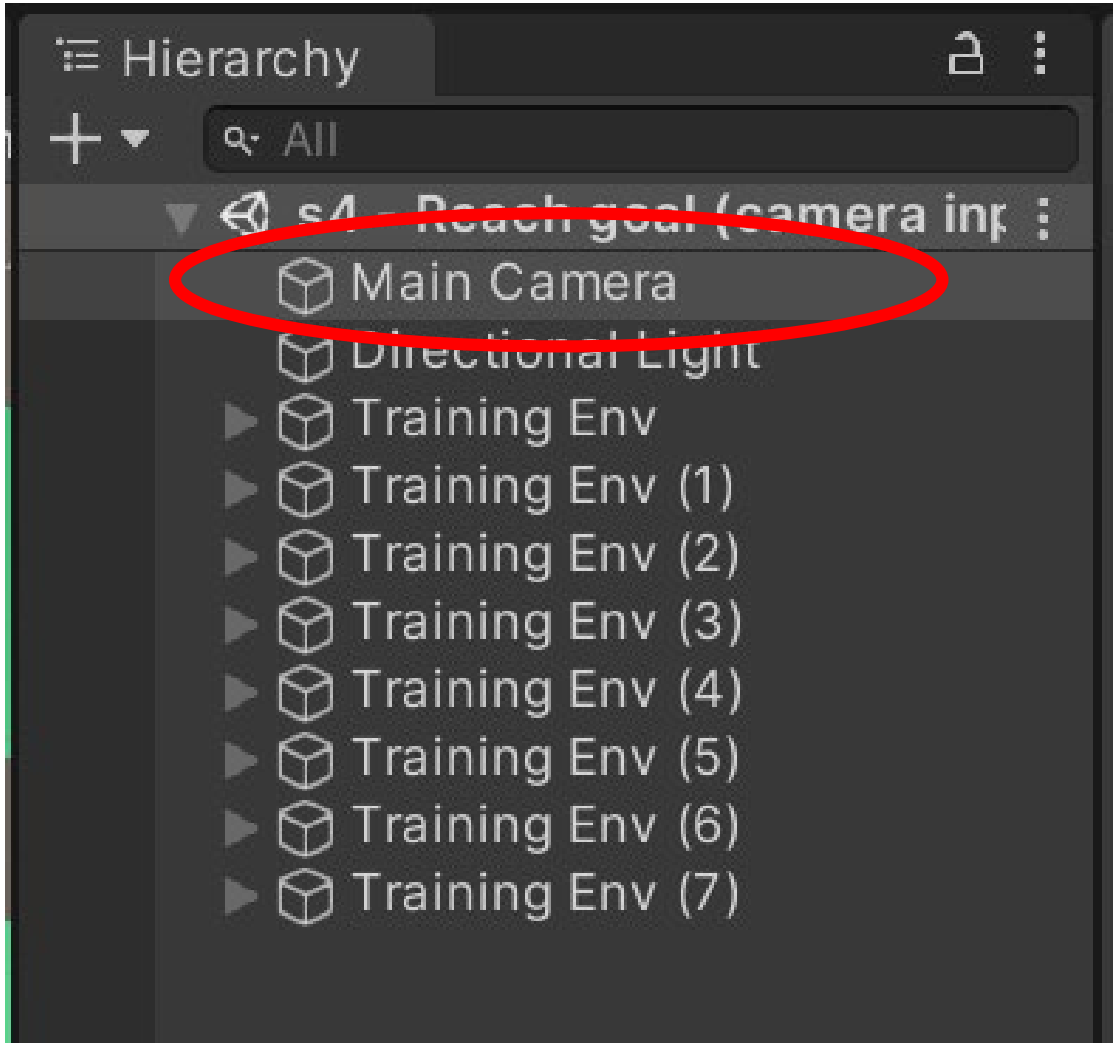


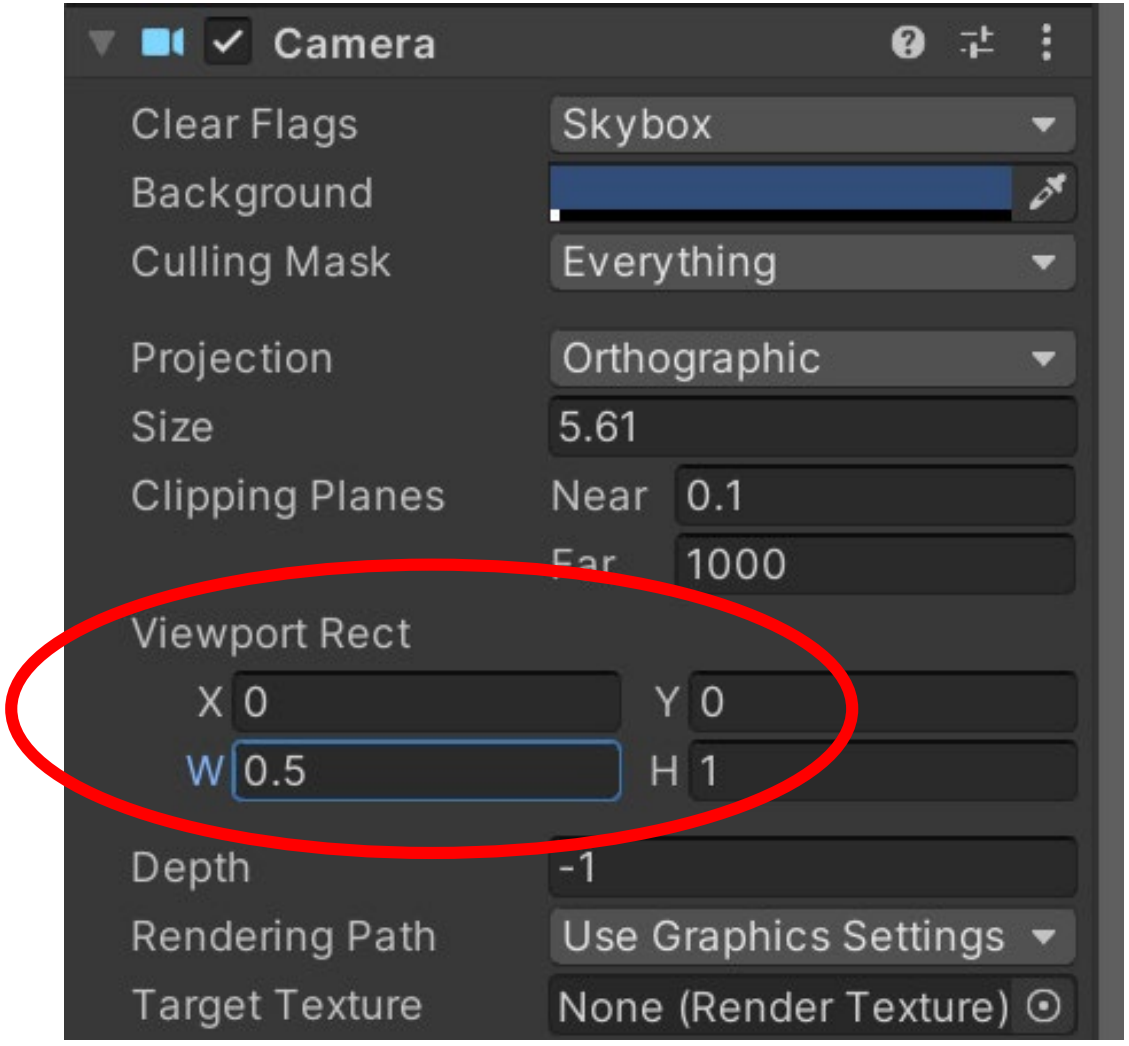
Turn on/off shadow



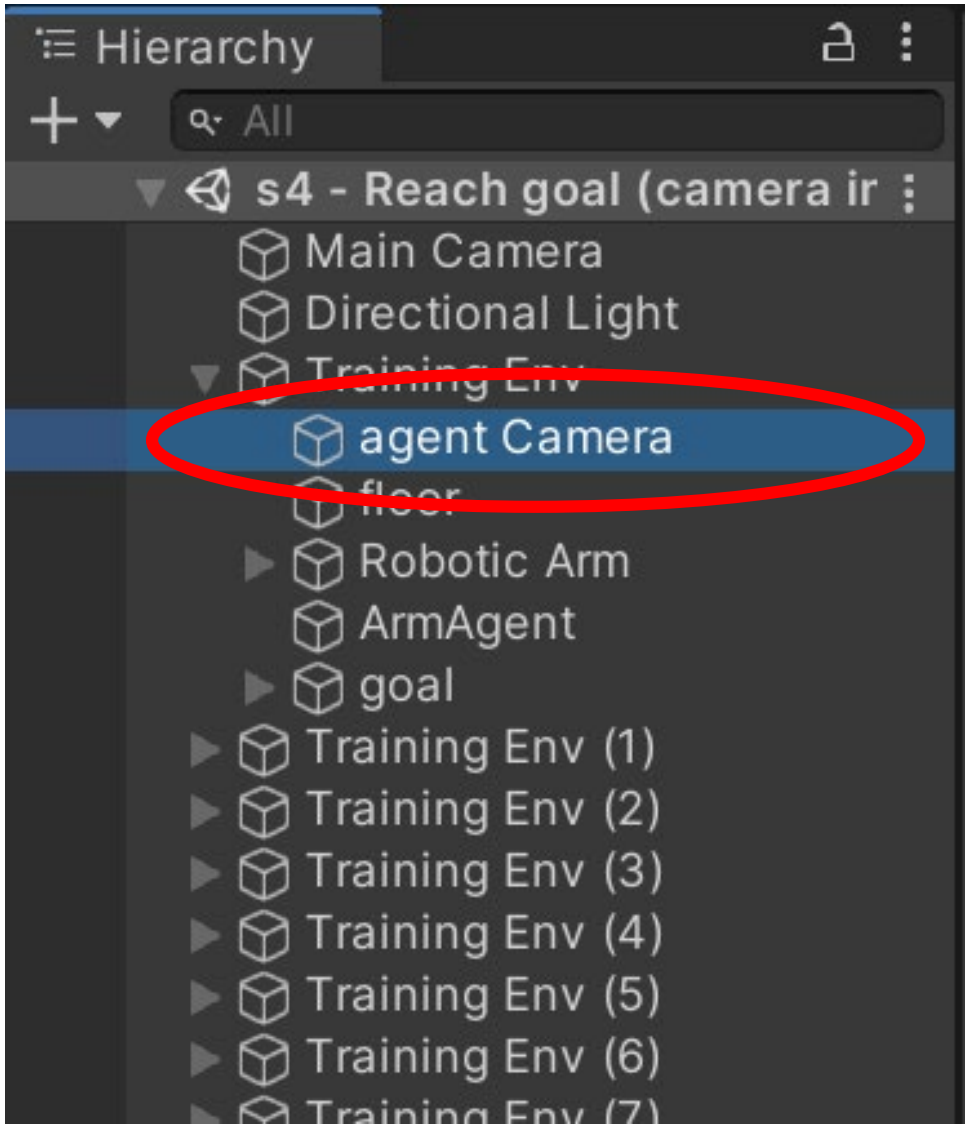


Camera view port adjustment





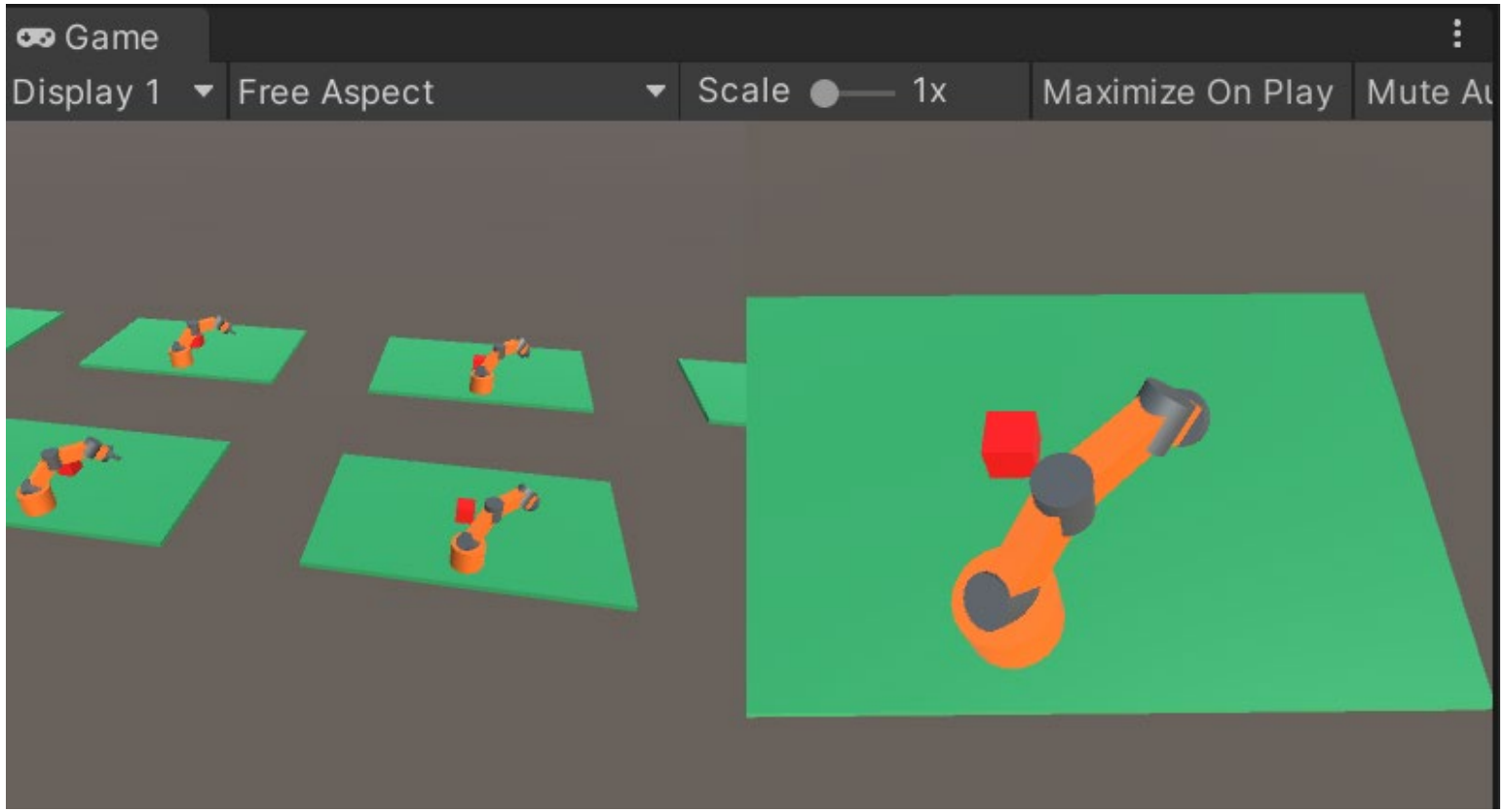
Camera view port adjustment



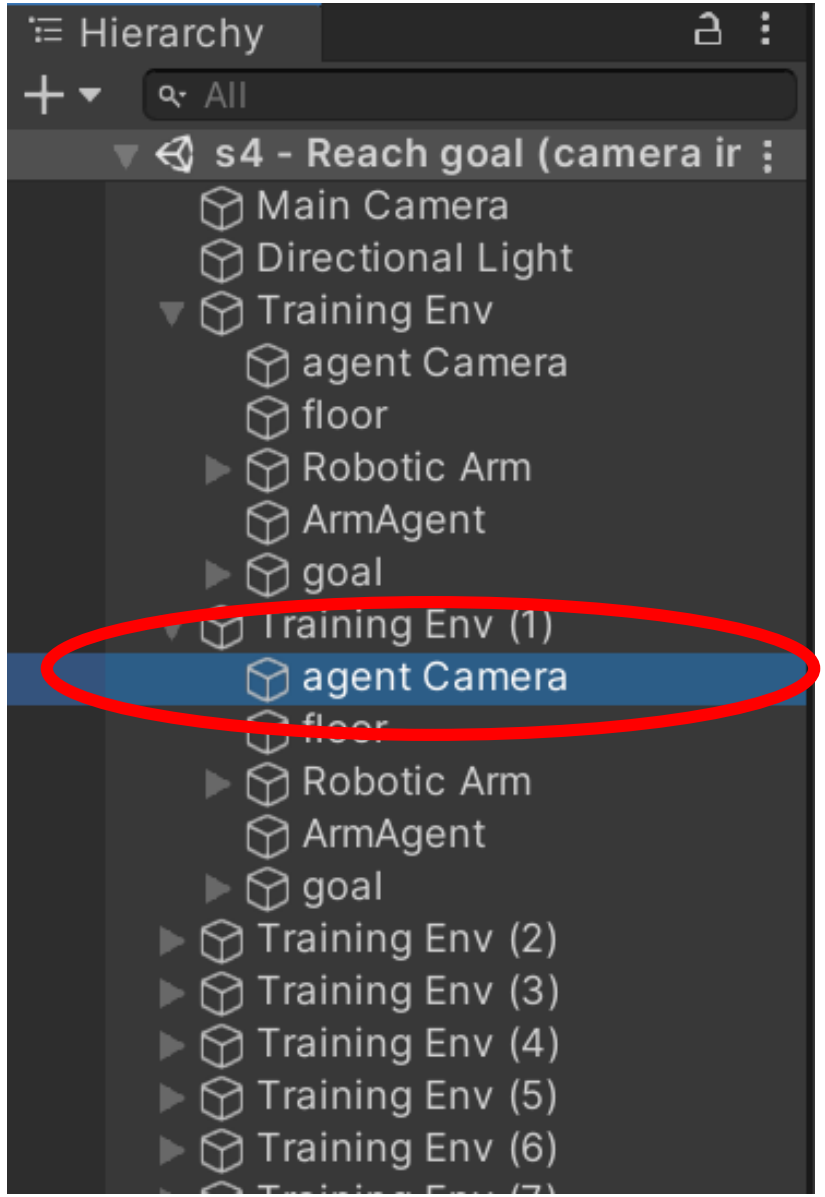
Camera view port adjustment

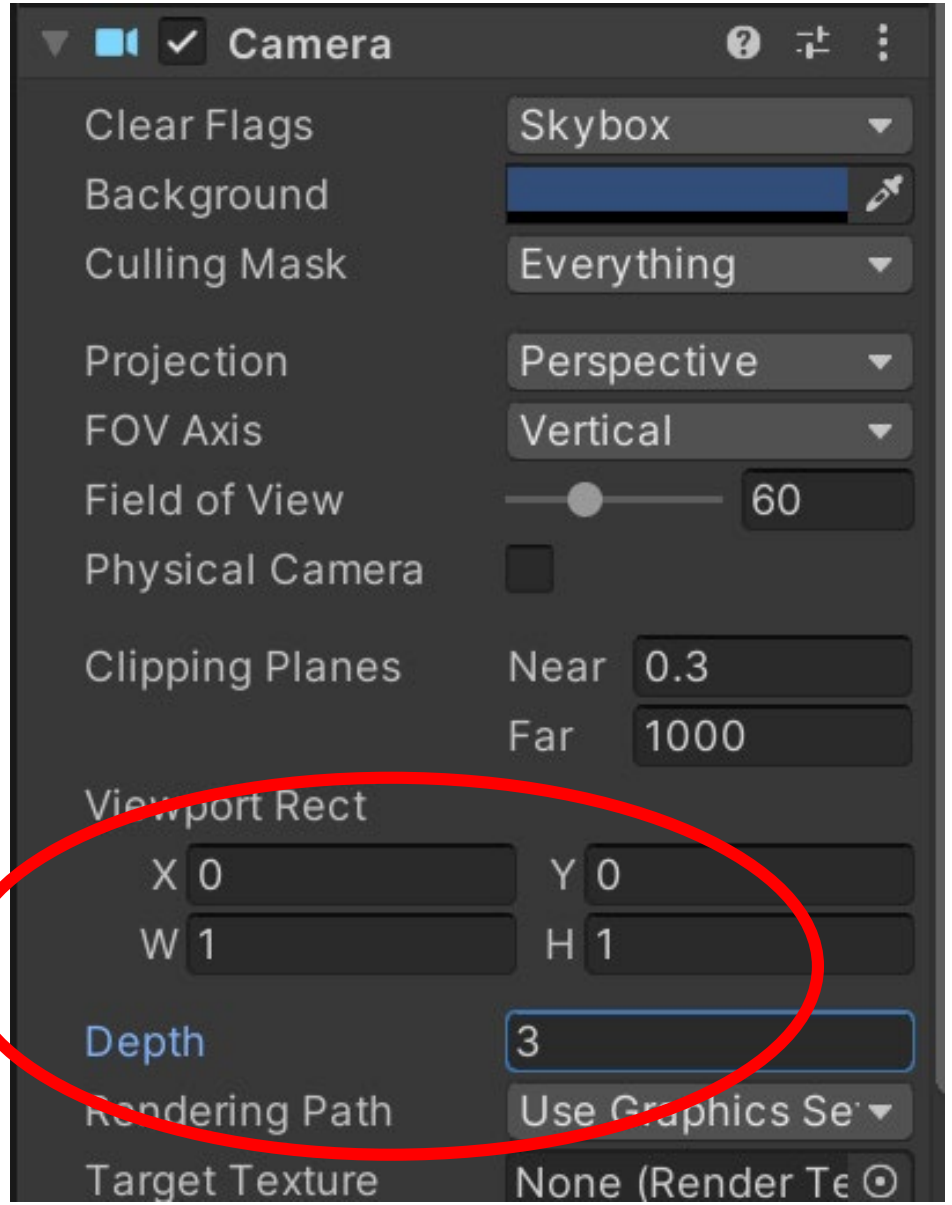
Main camera
(x=0, y=0), (w=0.5, h=1)

Agent camera
(x=0.5, y=0), (w=0.5, h=1)



Camera depth

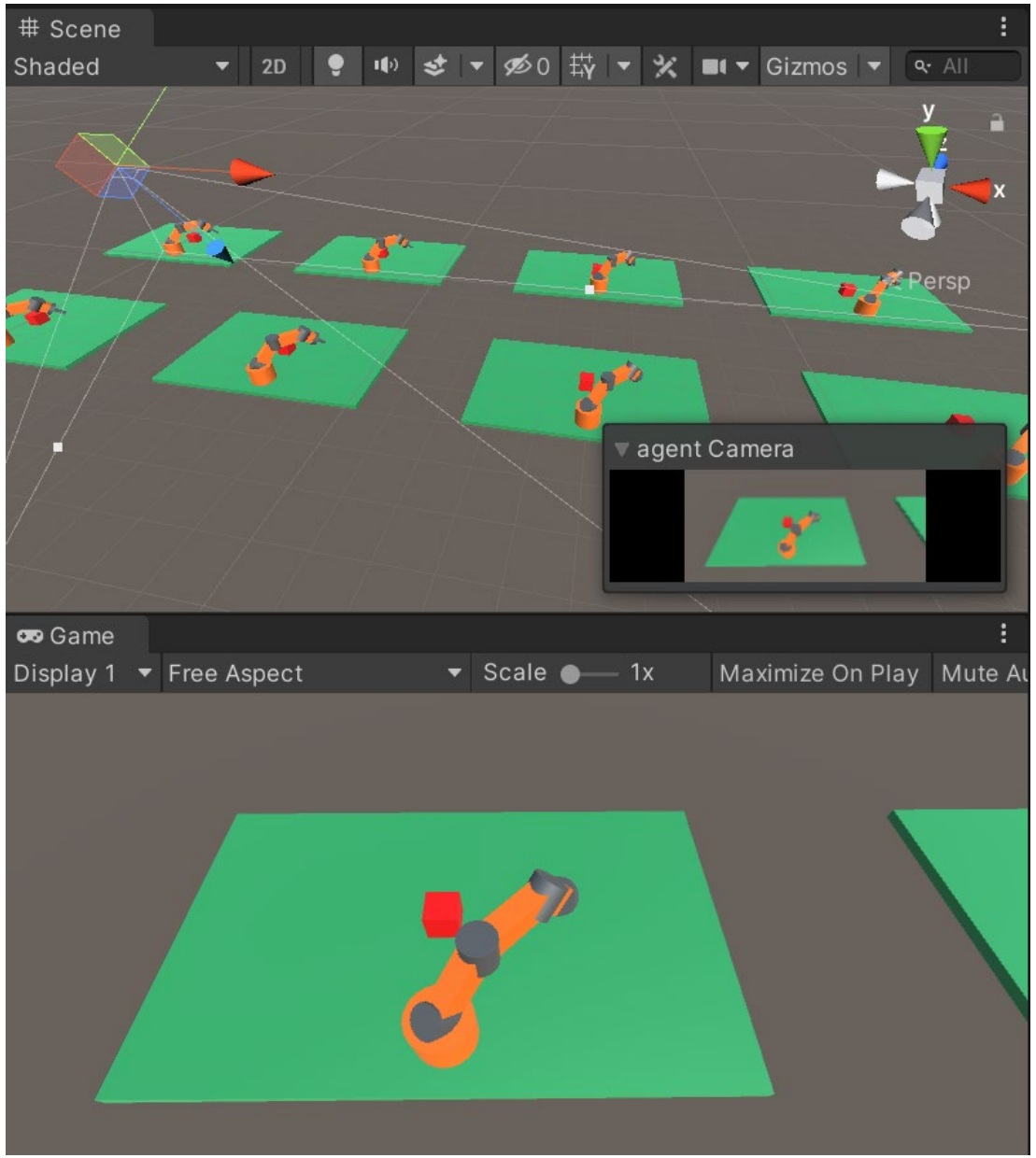




Game view display the camera with largest depth

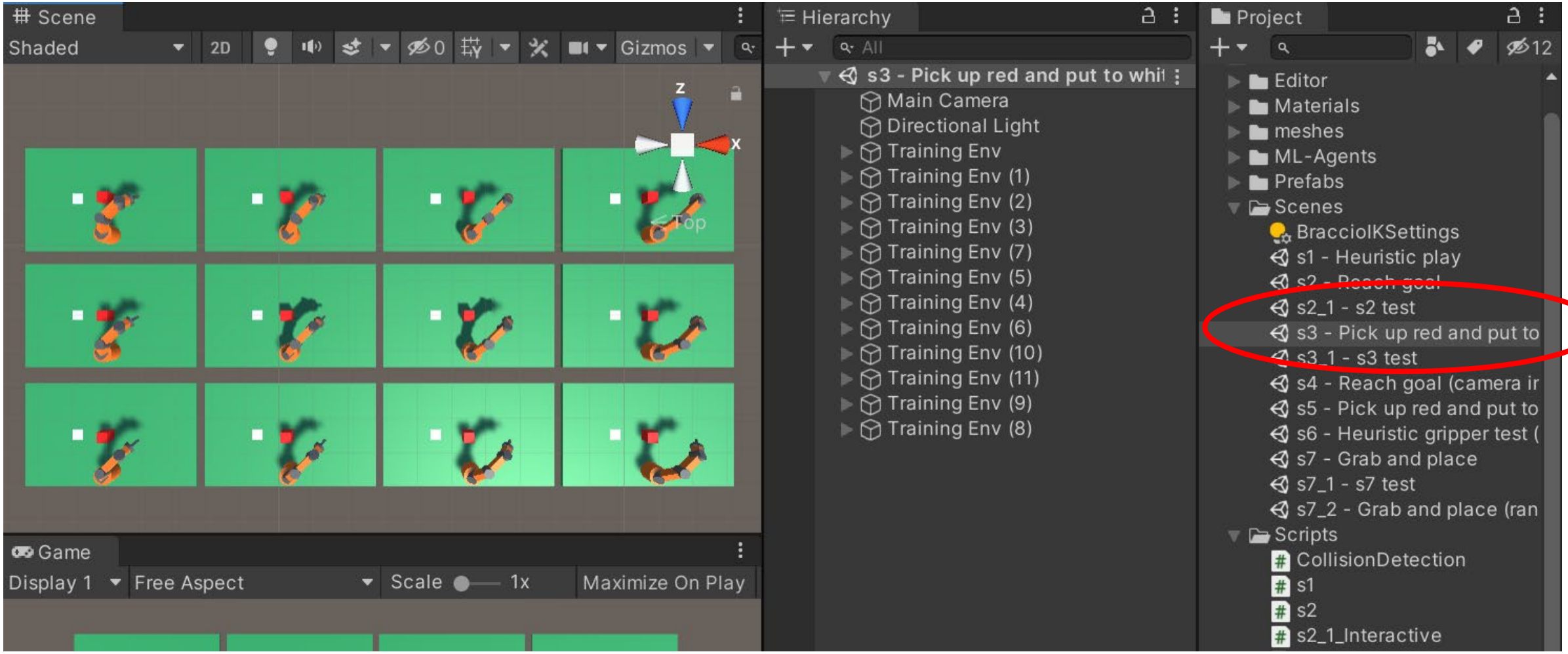
Main camera
(x=0, y=0), (w=0.5, h=1)
depth = -1

Agent camera
(x=0, y=0), (w=1, h=1)
depth = 0

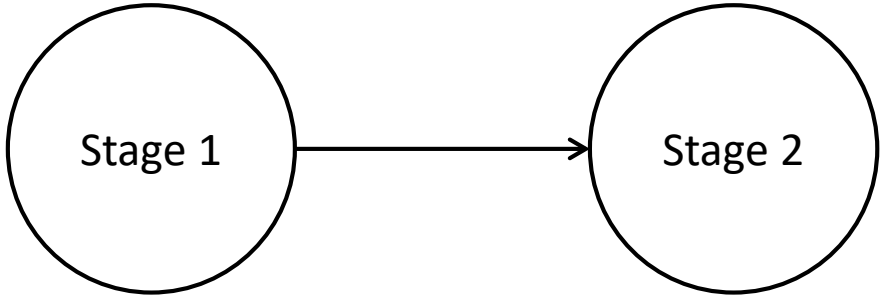


Scene s3 – Pick up red object and place it on top of white area using Δd information

Open "s3 – Pick up red and put to white"



Training setting



$$s = (\Delta x, \Delta y, \Delta z, \theta_{\mathrm{B}}, \theta_{\mathrm{U}}, \theta_{\mathrm{L}}, \theta_{\mathrm{W}})$$

$$r = \begin{cases} -0.005 & \text{per step} \\ -5 & \text{collision, out of range} \\ +20 & d_{stage1} \leq 0.5,\ d_{stage2} \leq 0.5 \end{cases}$$

$$a = (\Delta\theta_B, \Delta\theta_U, \Delta\theta_L, \Delta\theta_W)$$

NN: 8-512-512-512-4

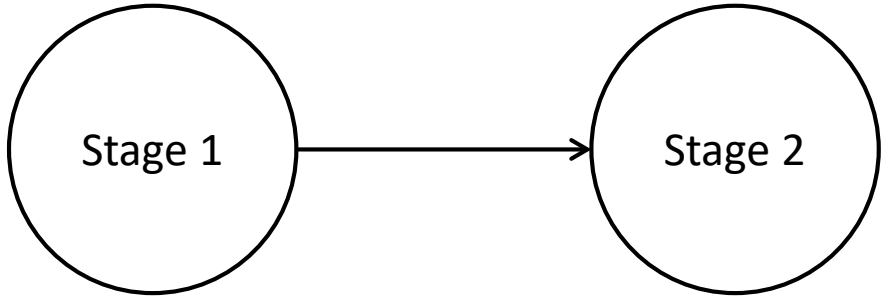
No. of training environment = 9
Time horizon = 2000
Buffer size = 20480
Batch size = 2048

Goal initialize = randomly positioned in polar system θ=-80~80, r=0.8~1.5

Goal2 initialize = same as goal 1

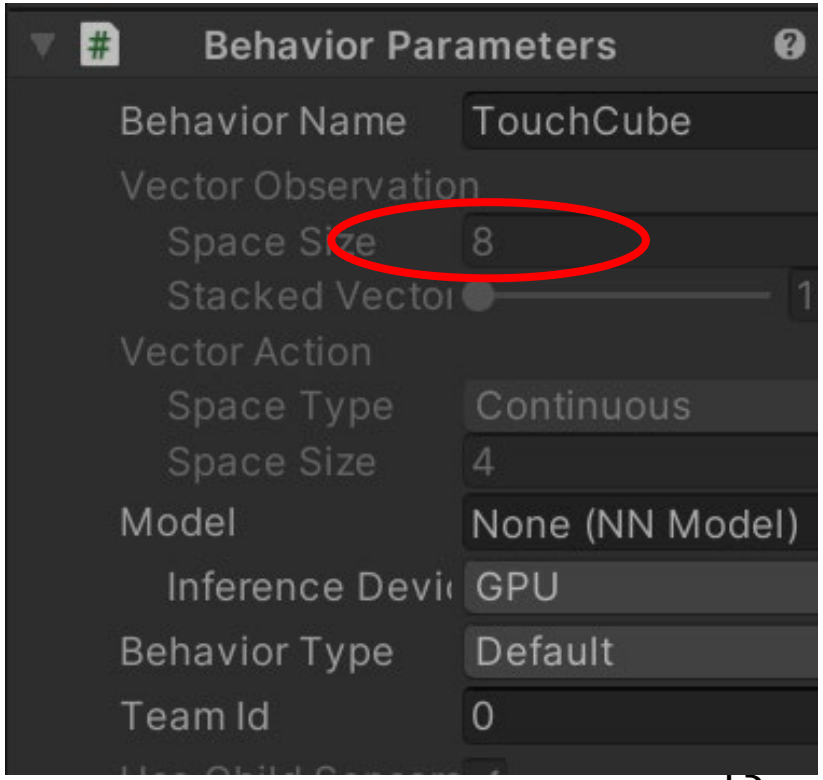
Arm initialize: $(\theta_B = 0, \theta_U = 45, \theta_L = 45, \theta_W = 45)$

State

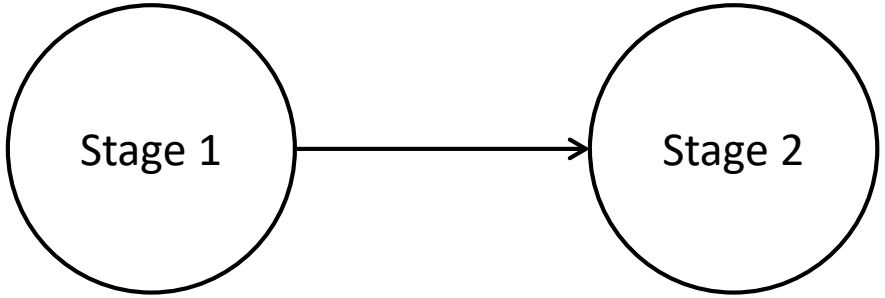


$$s = (\Delta x, \Delta y, \Delta z, \theta_{\mathrm{B}}, \theta_{\mathrm{U}}, \theta_{\mathrm{L}}, \theta_{\mathrm{W}})$$

```
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);
```



Rewards

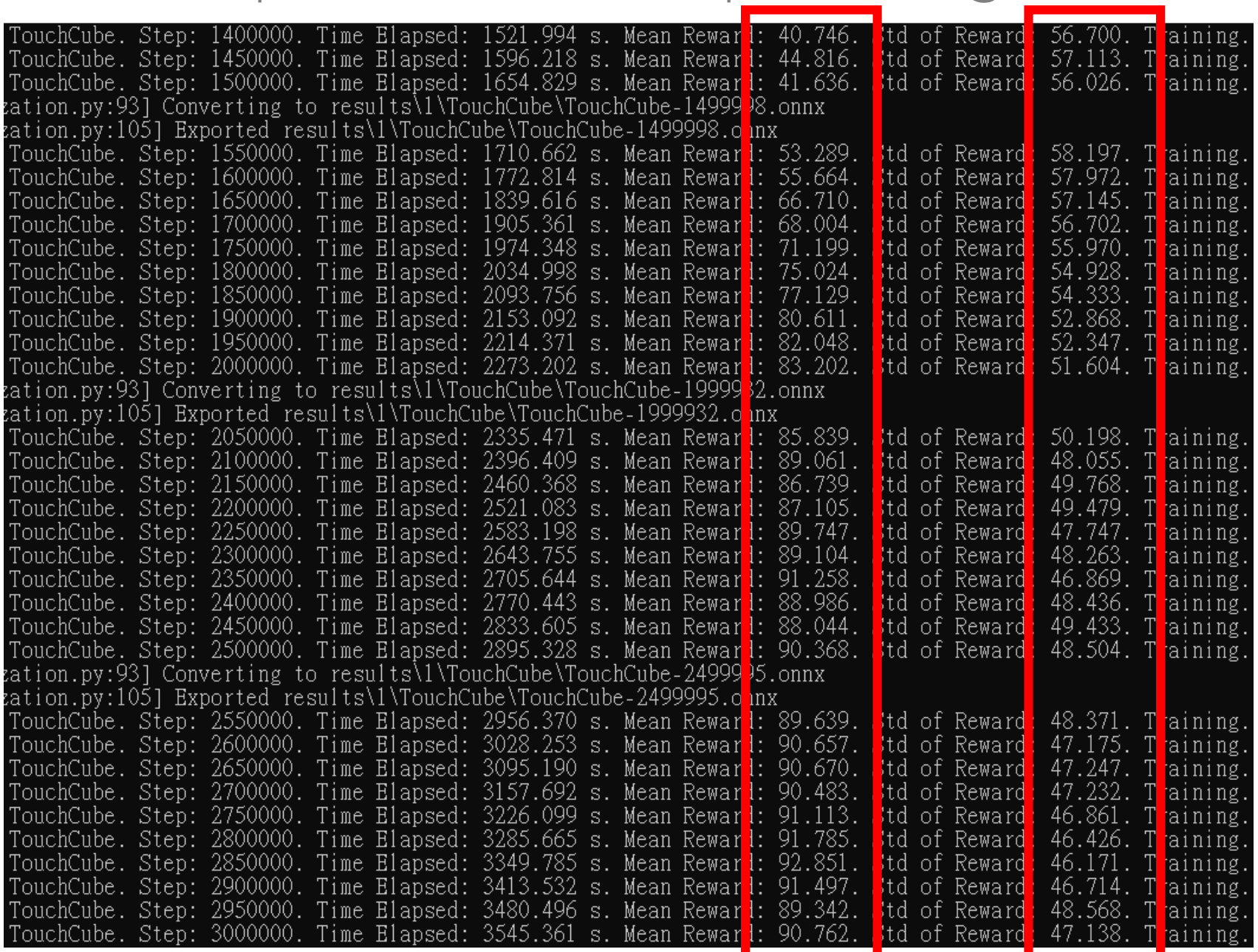


$$r = \begin{cases} -0.005 & \text{per step} \\ -5 & \text{collision, out of range} \\ +20 & d_{stage1} \leq 0.5,\ d_{stage2} \leq 0.5 \end{cases}$$

```
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
}
```

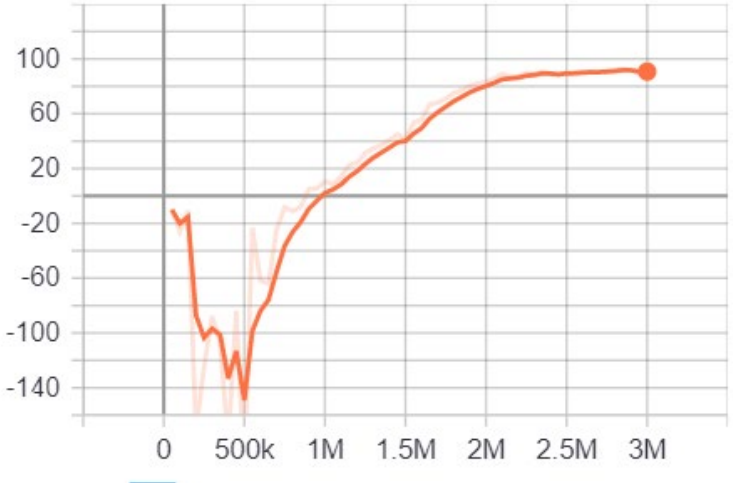
```
else if (PointTouch(goalDownTouchPt, goal2UpTouchPt, 0.3f))
{
    msg = trainingVE.name + " Goal 2! \n";
    Debug.Log(msg);
    AddReward(100.0f);
    EndEpisode();
}
```

I quit at 3M, looks promising

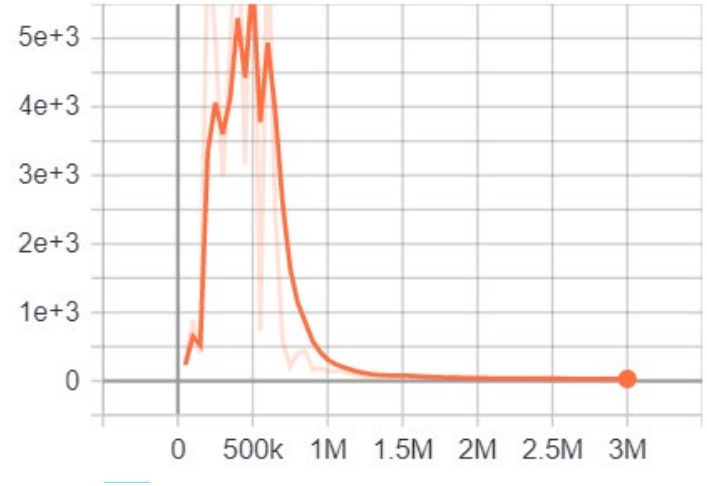


I quit at 3M, looks promising

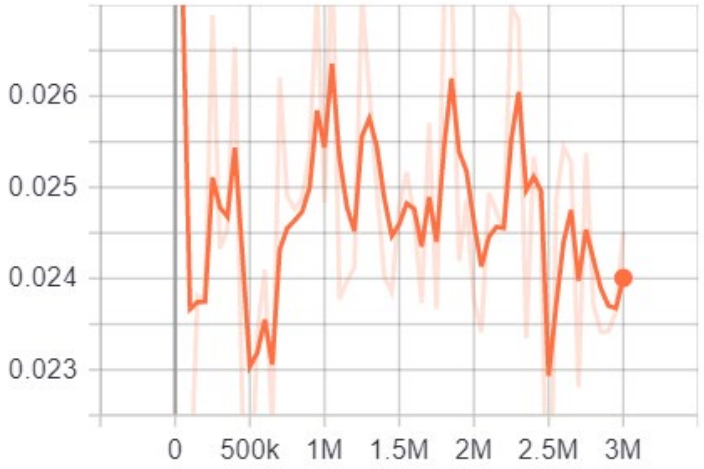
Cumulative Reward
tag: Environment/Cumulative Reward



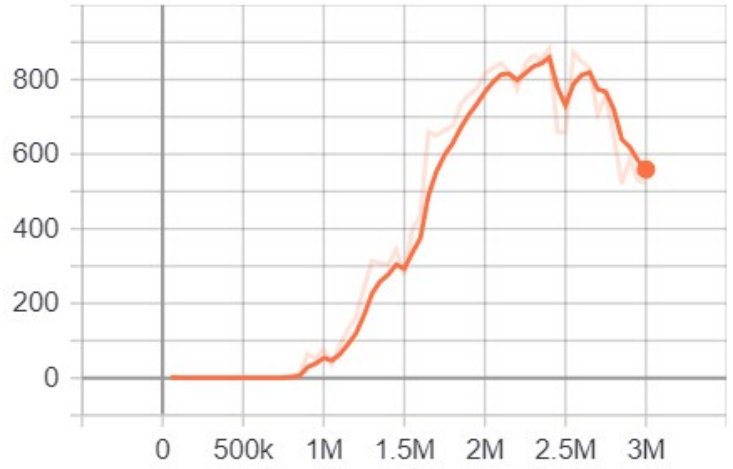
Episode Length
tag: Environment/Episode Length



Policy Loss
tag: Losses/Policy Loss

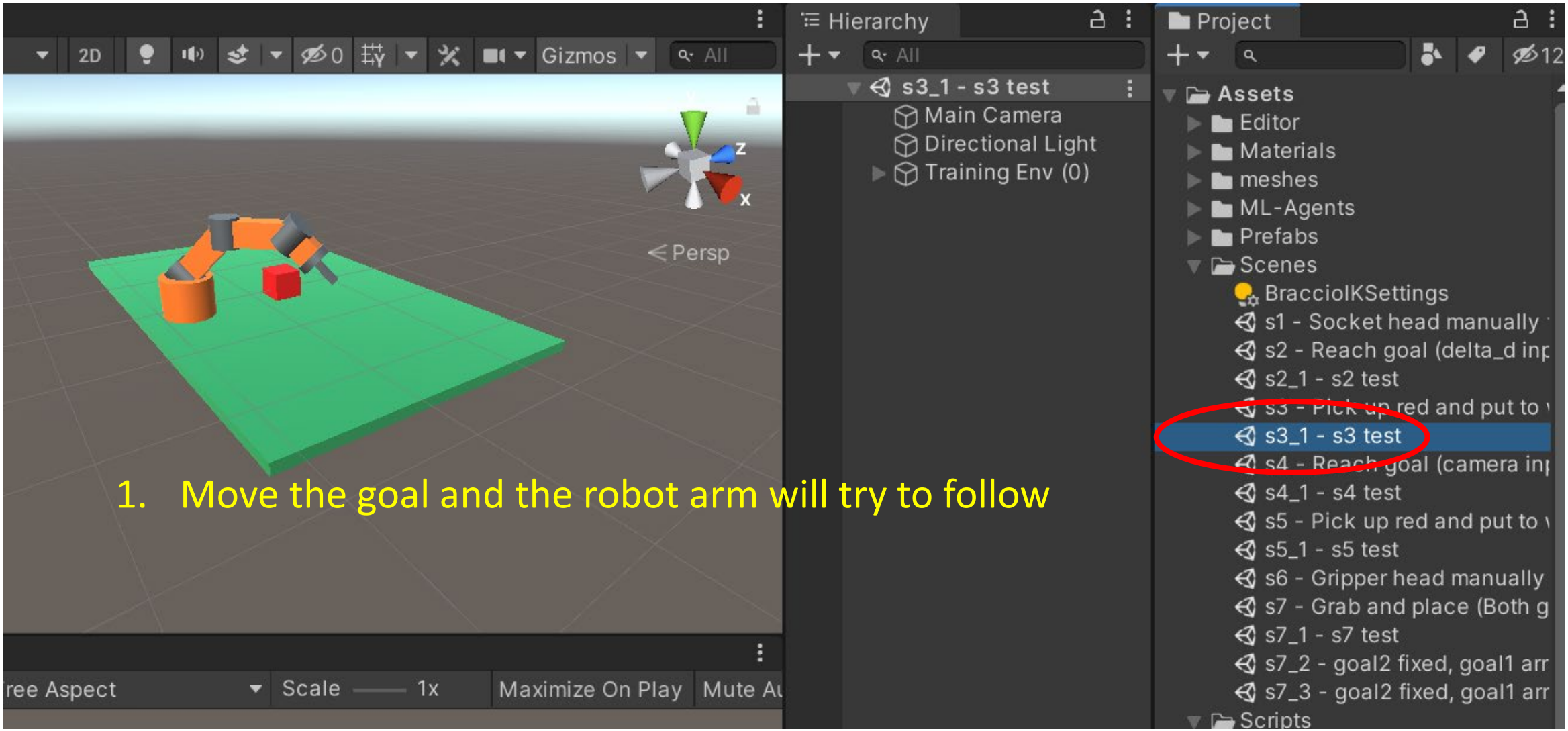


Value Loss
tag: Losses/Value Loss



Test – Play with robot arm

Open "s3_1"



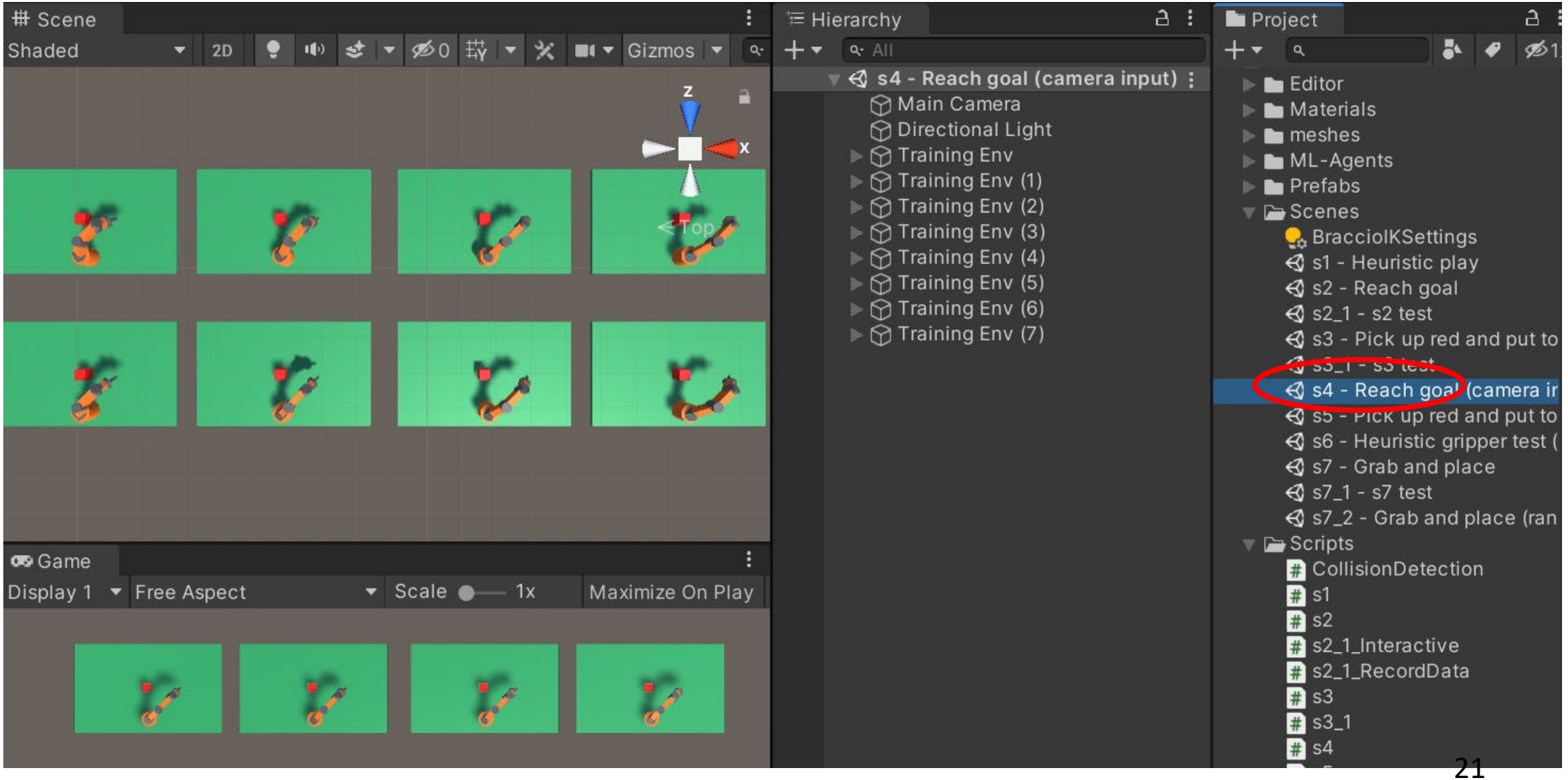
1. Move the goal and the robot arm will try to follow

HW4(2)

- Describe training setting
- Show tensor board plots and discuss your training performance
- Describe test performance (recorded data, interactive test)

Scene s4 – Reach goal using camera image

Open s4


Scene
Shaded
2D
Gizmos
Top
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Hierarchy
All
s4 - Reach goal (camera input)
Main Camera
Directional Light
Training Env
Training Env (1)
Training Env (2)
Training Env (3)
Training Env (4)
Training Env (5)
Training Env (6)
Training Env (7)
Project
Editor
Materials
meshes
ML-Agents
Prefabs
Scenes
BraccioIKSettings
s1 - Heuristic play
s2 - Reach goal
s2_1 - s2 test
s3 - Pick up red and put to
s4 - Reach goal (camera ir
s6 - Heuristic gripper test (
s7 - Grab and place
s7_1 - s7 test
s7_2 - Grab and place (ran
Scripts
CollisionDetection
s1
s2
s2_1_Interactive
s2_1_RecordData
s3
s3_1
s4

Training setting

s = feature map vector from a CNN, size = 2592

Input image to the CNN is captured by a camera from top, size = 84x84x3

$$a = (\Delta\theta_B, \Delta\theta_U, \Delta\theta_L, \Delta\theta_W)$$

$$r = \begin{cases} -0.005 & \text{per step} \\ -5 & \text{collision, out of range} \\ +20 & \text{goal, } d \leq 0.5 \end{cases}$$

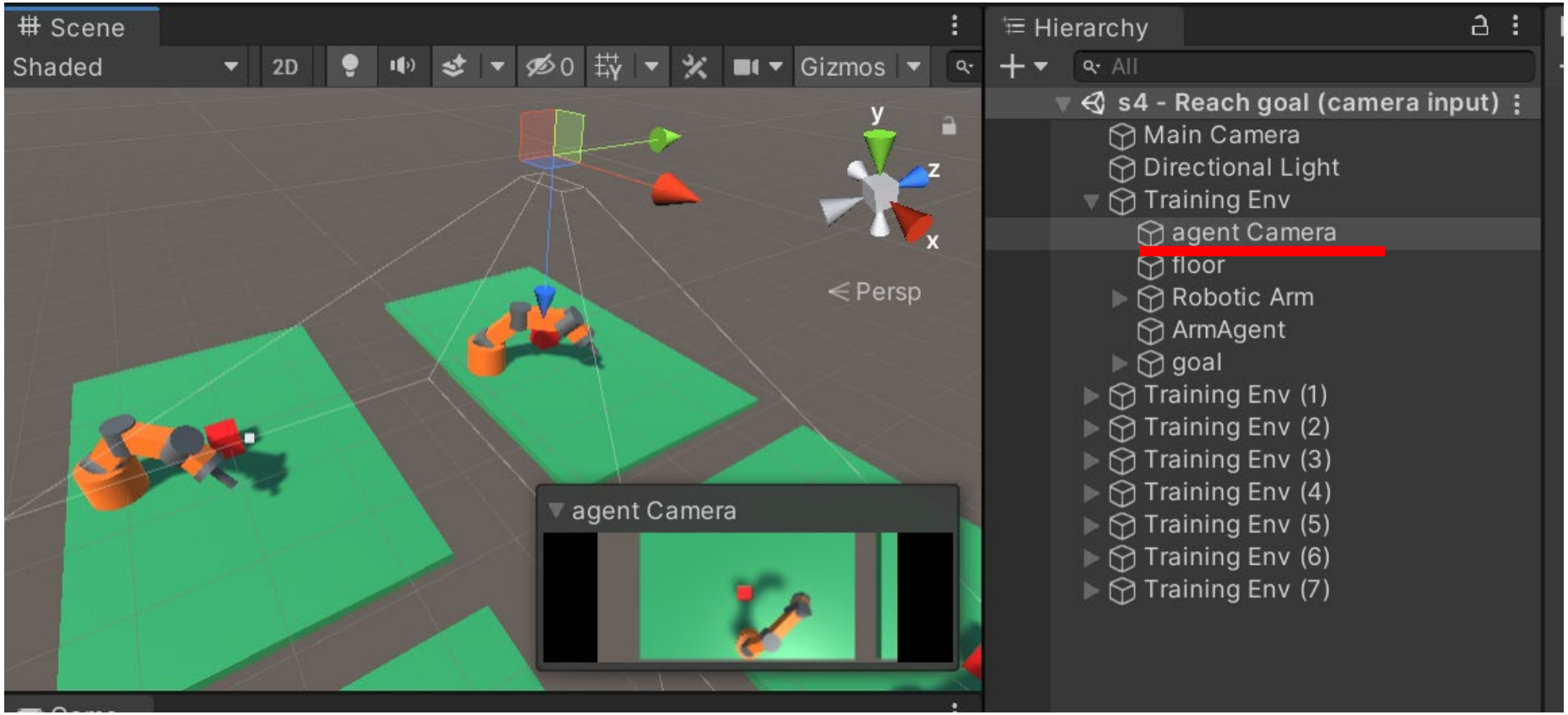
Try 0.3? 0.25

No. of training environment = 8

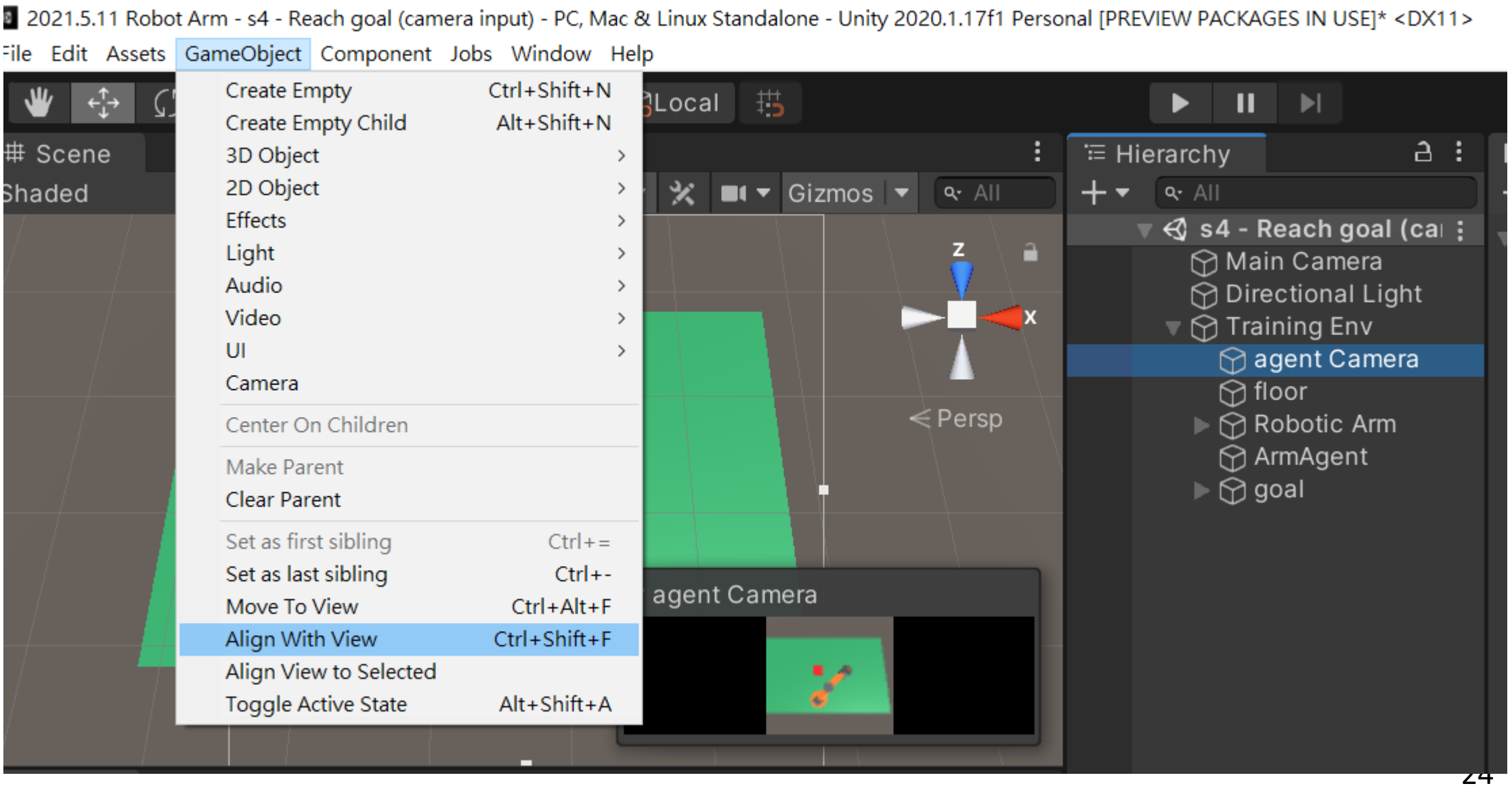
Goal initialize = randomly positioned in polar system θ=-80~80, r=0.8~1.5

Arm initialize: $(\theta_b = 0, \theta_u = 45, \theta_l = 45, \theta_w = 45)$

Add camera sensor to the robot agent

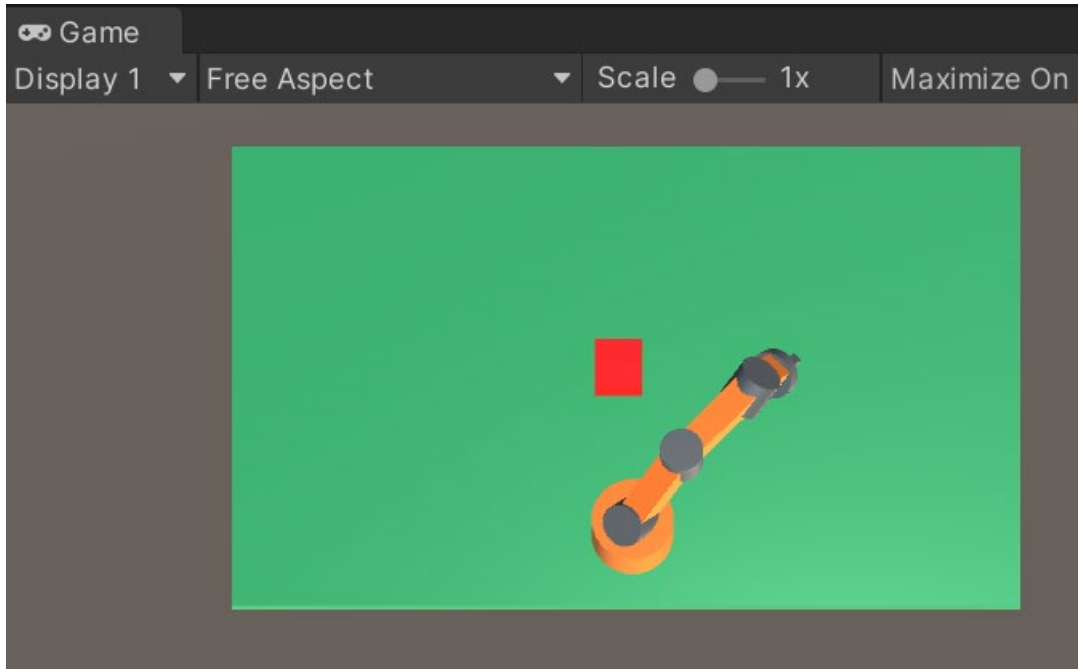


Align agent camera with scene view

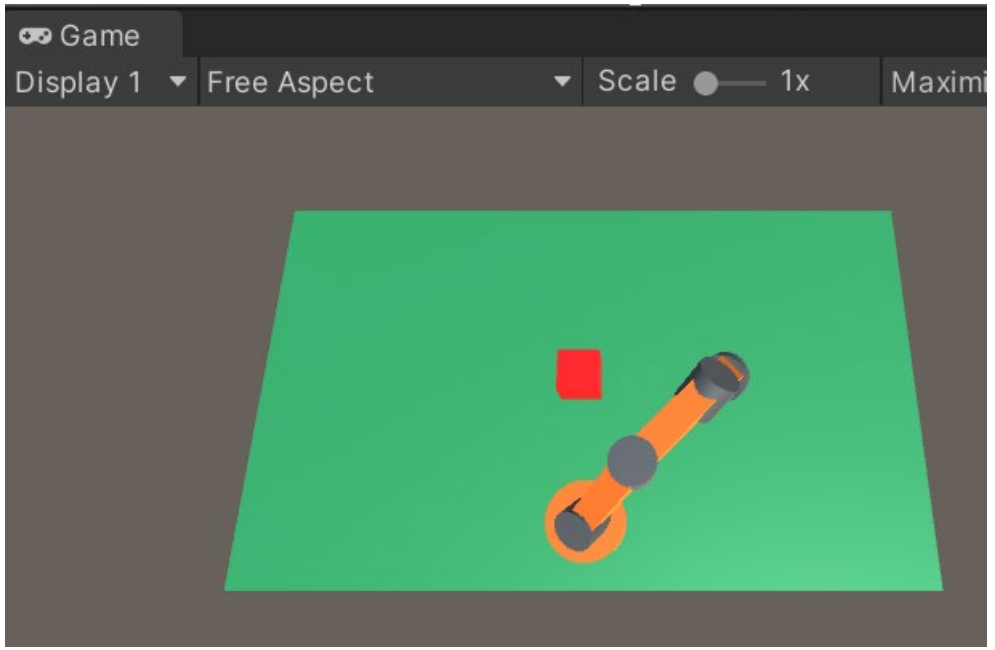


How to place agent camera?

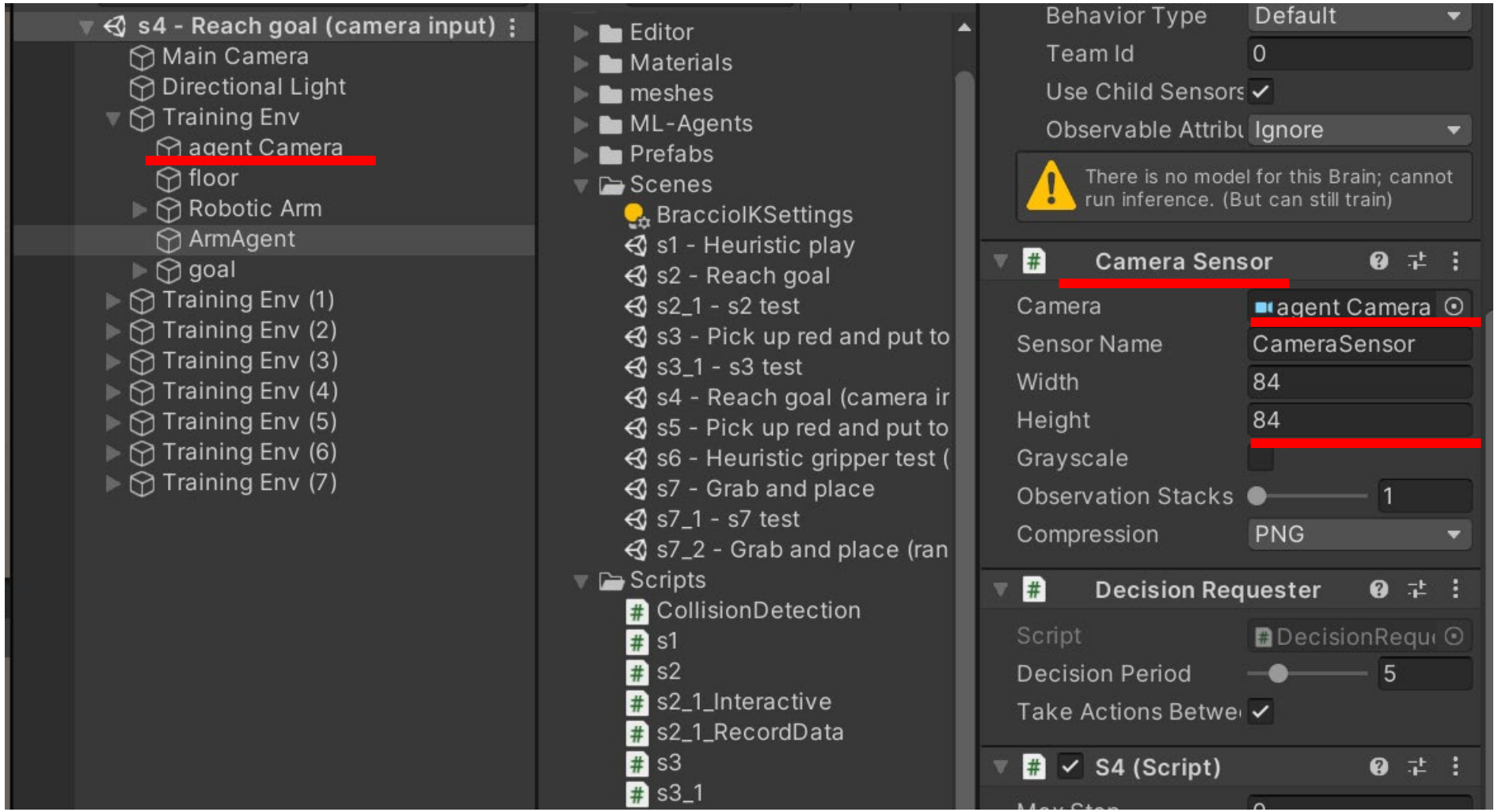
Orthographic, size=5



Perspective, DOF=60

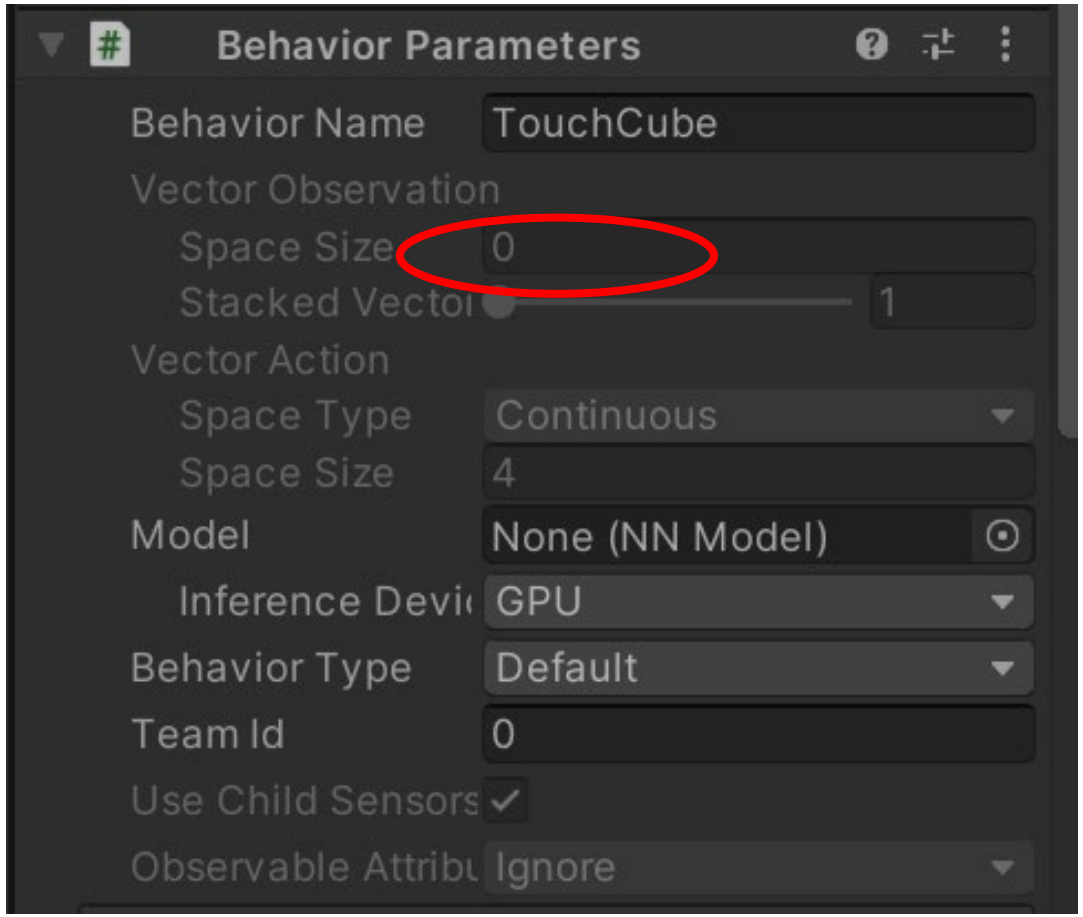


Add camera sensor to the robot agent



Vector observation = 0

```
public override void CollectObservations(VectorSensor sensor)
{

}
```

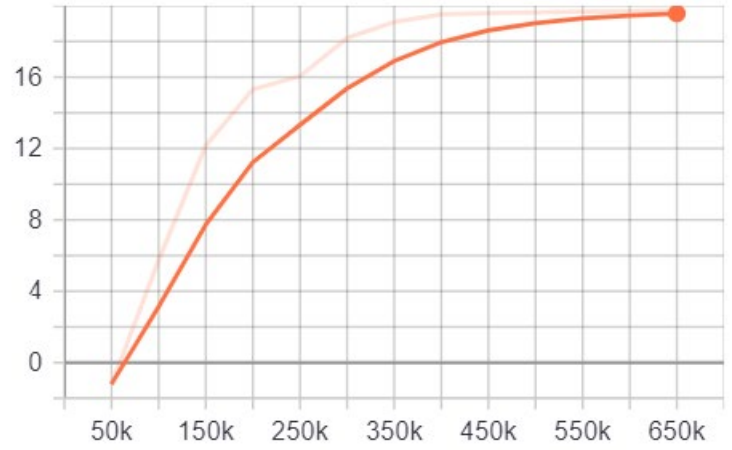


Looks good shortly (600K only)

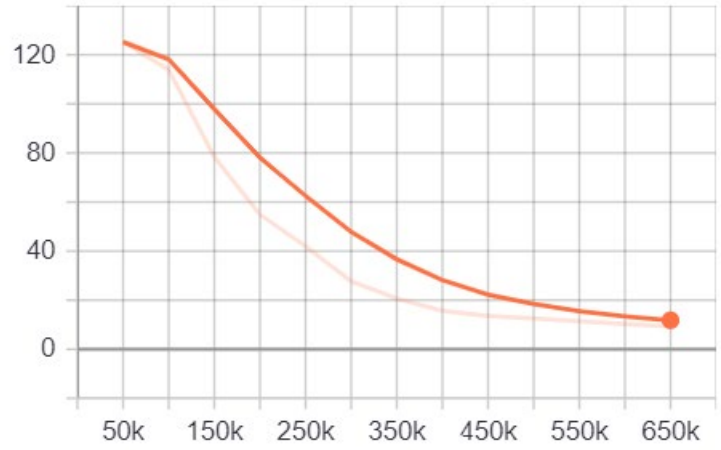
```
TouchCube. Step: 50000. Time Elapsed: 222.493 s. Mean Reward: -1.205. Std of Reward: 12.602. Training.
TouchCube. Step: 100000. Time Elapsed: 417.921 s. Mean Reward: 5.743. Std of Reward: 13.921. Training.
TouchCube. Step: 150000. Time Elapsed: 618.958 s. Mean Reward: 12.166. Std of Reward: 11.773. Training.
TouchCube. Step: 200000. Time Elapsed: 819.051 s. Mean Reward: 15.312. Std of Reward: 9.549. Training.
TouchCube. Step: 250000. Time Elapsed: 1027.214 s. Mean Reward: 16.052. Std of Reward: 9.106. Training.
TouchCube. Step: 300000. Time Elapsed: 1230.887 s. Mean Reward: 18.197. Std of Reward: 5.736. Training.
TouchCube. Step: 350000. Time Elapsed: 1446.545 s. Mean Reward: 19.086. Std of Reward: 3.321. Training.
TouchCube. Step: 400000. Time Elapsed: 1662.651 s. Mean Reward: 19.512. Std of Reward: 1.555. Training.
TouchCube. Step: 450000. Time Elapsed: 1899.574 s. Mean Reward: 19.595. Std of Reward: 1.210. Training.
TouchCube. Step: 500000. Time Elapsed: 2154.839 s. Mean Reward: 19.635. Std of Reward: 0.976. Training.
zation.py:93] Converting to results\1\TouchCube\TouchCube-499992.onnx
ges\mlagents\trainers\torch\distributions.py:163: TracerWarning: Converting a tensor to a Python index mig
n't record the data flow of Python values, so this value will be treated as a constant in the future. This
e to other inputs!
] * inputs.shape[0], axis=0)
ges\mlagents\trainers\torch\networks.py:352: TracerWarning: torch.Tensor results are registered as constar
is warning if you use this function to create tensors out of constant variables that would be the same eve
her case, this might cause the trace to be incorrect.
y_size]),
zation.py:105] Exported results\1\TouchCube\TouchCube-499992.onnx
TouchCube. Step: 550000. Time Elapsed: 2391.675 s. Mean Reward: 19.671. Std of Reward: 0.946. Training.
TouchCube. Step: 600000. Time Elapsed: 2647.336 s. Mean Reward: 19.707. Std of Reward: 0.785. Training.
```

Good results only after 650K

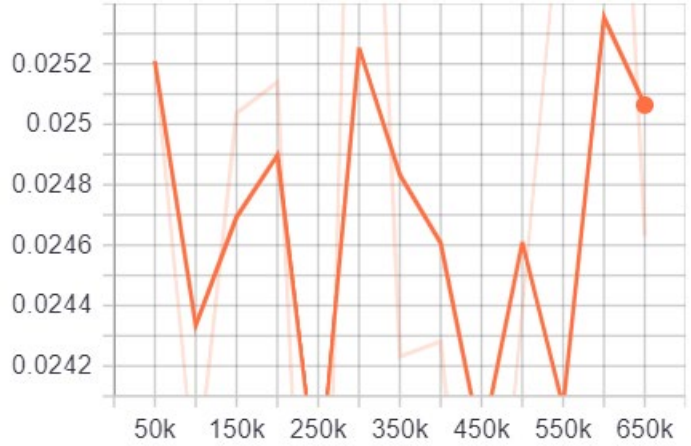
Cumulative Reward
tag: Environment/Cumulative Reward



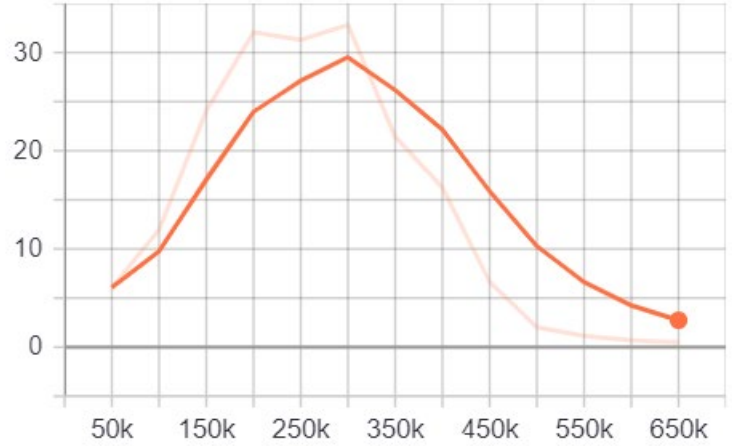
Episode Length
tag: Environment/Episode Length



Policy Loss
tag: Losses/Policy Loss

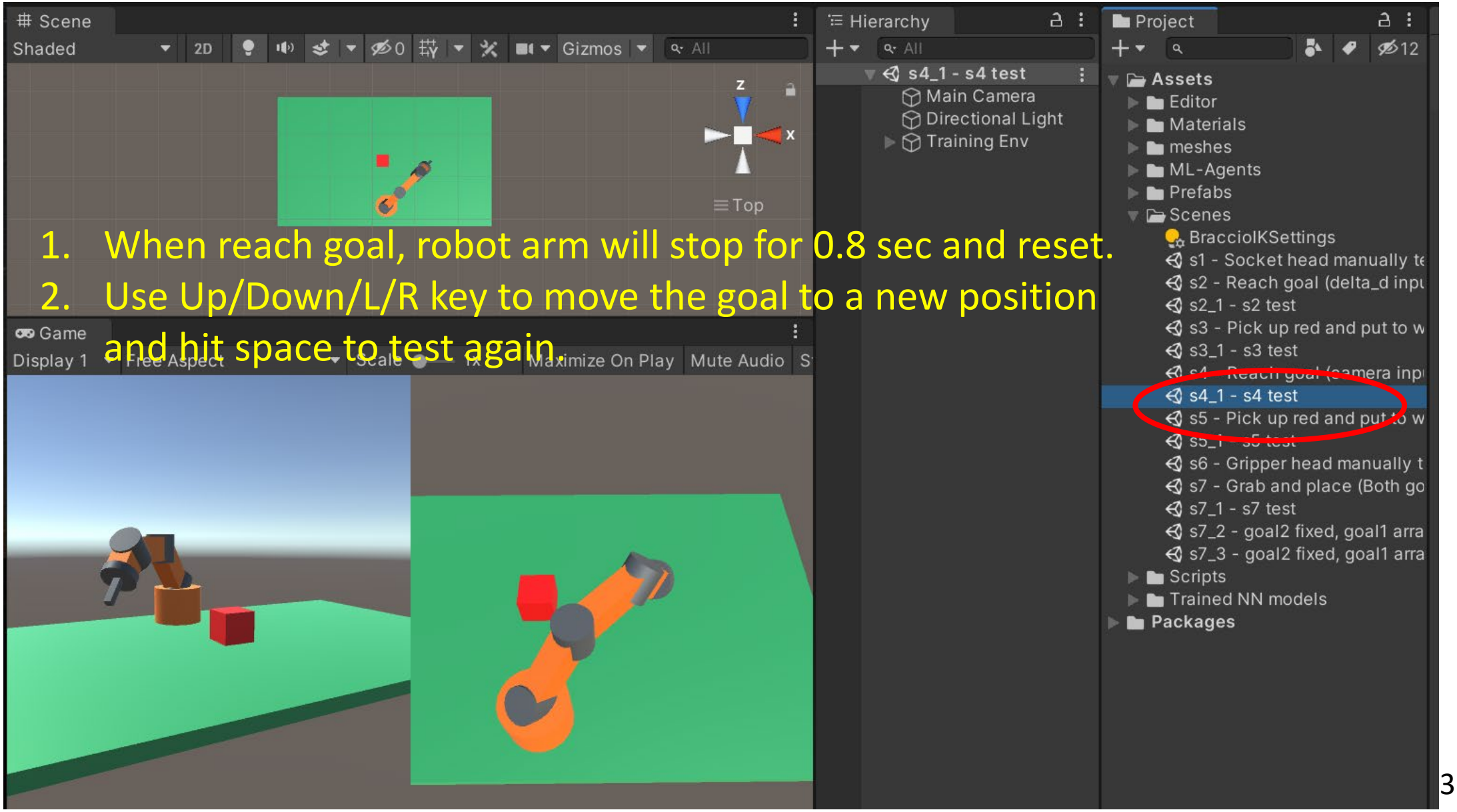


Value Loss
tag: Losses/Value Loss



Test – Play with robot arm

Open "s4_1"



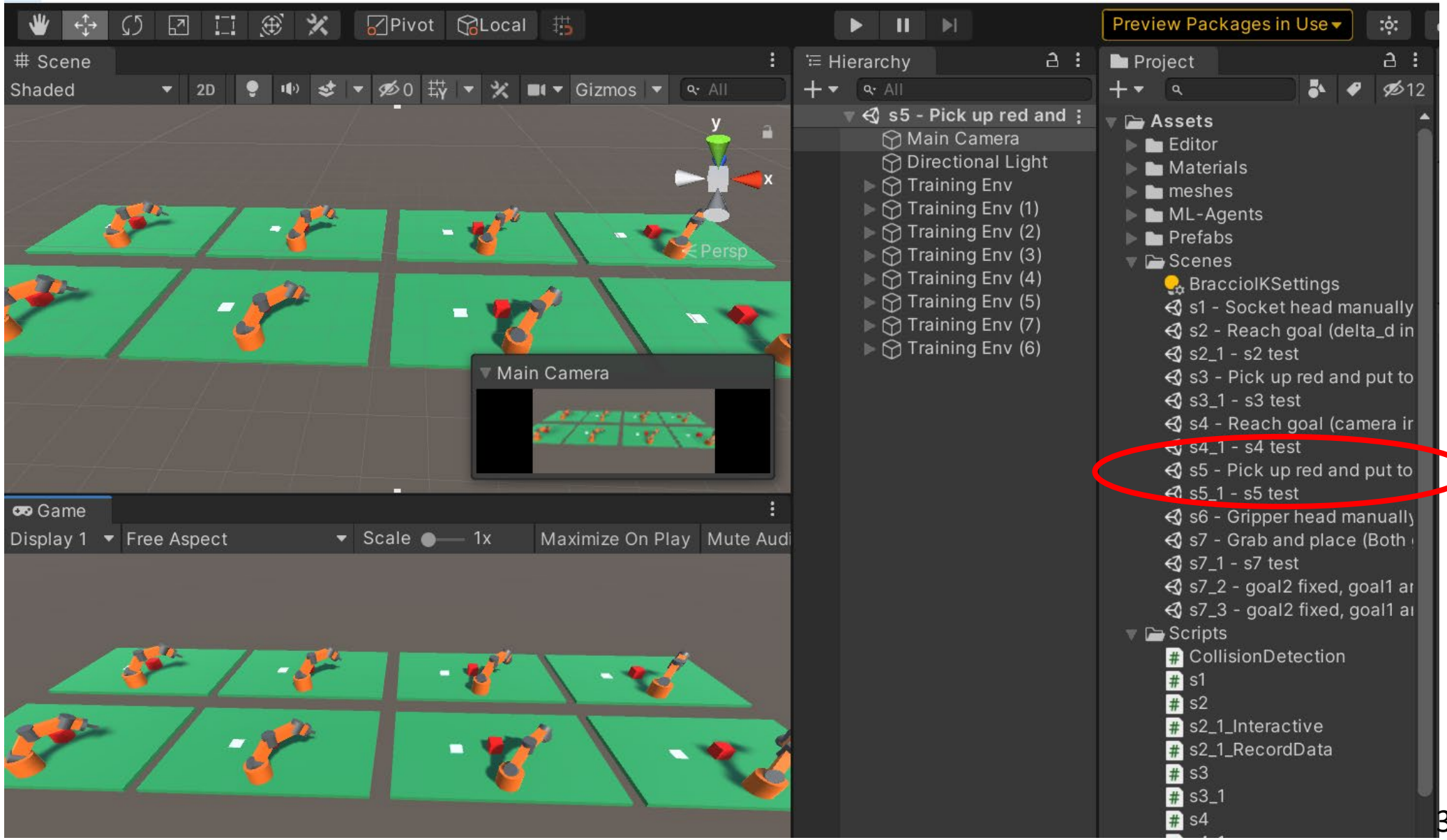
1. When reach goal, robot arm will stop for 0.8 sec and reset.
2. Use Up/Down/L/R key to move the goal to a new position and hit space to test again.

HW5(1)

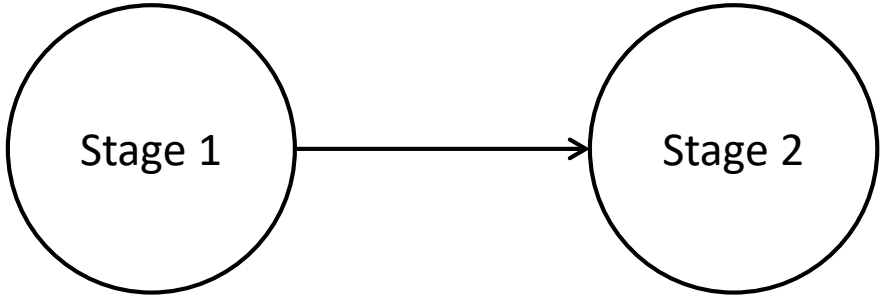
- Describe training setting
- Show tensor board plots and discuss your training performance
- Describe test performance (recorded data, interactive test)

Scene s5 – Pick red cube and place it on top of the white cube using camera image

Open s5


Pivot
Local
Preview Packages in Use
Scene
Shaded
2D
Gizmos
Persp
Main Camera
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Hierarchy
s5 - Pick up red and
Main Camera
Directional Light
Training Env
Training Env (1)
Training Env (2)
Training Env (3)
Training Env (4)
Training Env (5)
Training Env (7)
Training Env (6)
Project
Assets
Editor
Materials
meshes
ML-Agents
Prefabs
Scenes
BracciolKSettings
s1 - Socket head manually
s2 - Reach goal (delta_d in
s2_1 - s2 test
s3 - Pick up red and put to
s3_1 - s3 test
s4 - Reach goal (camera ir
s4_1 - s4 test
s5 - Pick up red and put to
s5_1 - s5 test
s6 - Gripper head manually
s7 - Grab and place (Both
s7_1 - s7 test
s7_2 - goal2 fixed, goal1 ar
s7_3 - goal2 fixed, goal1 ar
Scripts
CollisionDetection
s1
s2
s2_1_Interactive
s2_1_RecordData
s3
s3_1
s4

Training setting



s = feature map vector from a CNN, size = 2592

Input image to the CNN is captured by a camera from top, size = 84x84x3

In my script, I use +15 for goal 1 and +100 for goal 2. But +20 for both goal 1 and goal 2 should be OK.

$$r = \begin{cases} -0.005 & \text{per step} \\ -5 & \text{collision, out of range} \\ +20 & d_{stage1} \leq 0.5,\ d_{stage2} \leq 0.5 \end{cases}$$

0.5 is easier to succeed. But the resulted behavior is unreal, like long-distance socket. Try 0.3 or 0.25 for your HW.

$$a = (\Delta\theta_B, \Delta\theta_U, \Delta\theta_L, \Delta\theta_W)$$

No. of training environment = 8

NN: ?-512-512-512-4

Time horizon = 2000

Buffer size = 20480

Batch size = 2048

Goal initialize = randomly positioned in polar system θ=-80~80, r=0.8~1.5

Goal2 initialize = same as goal 1

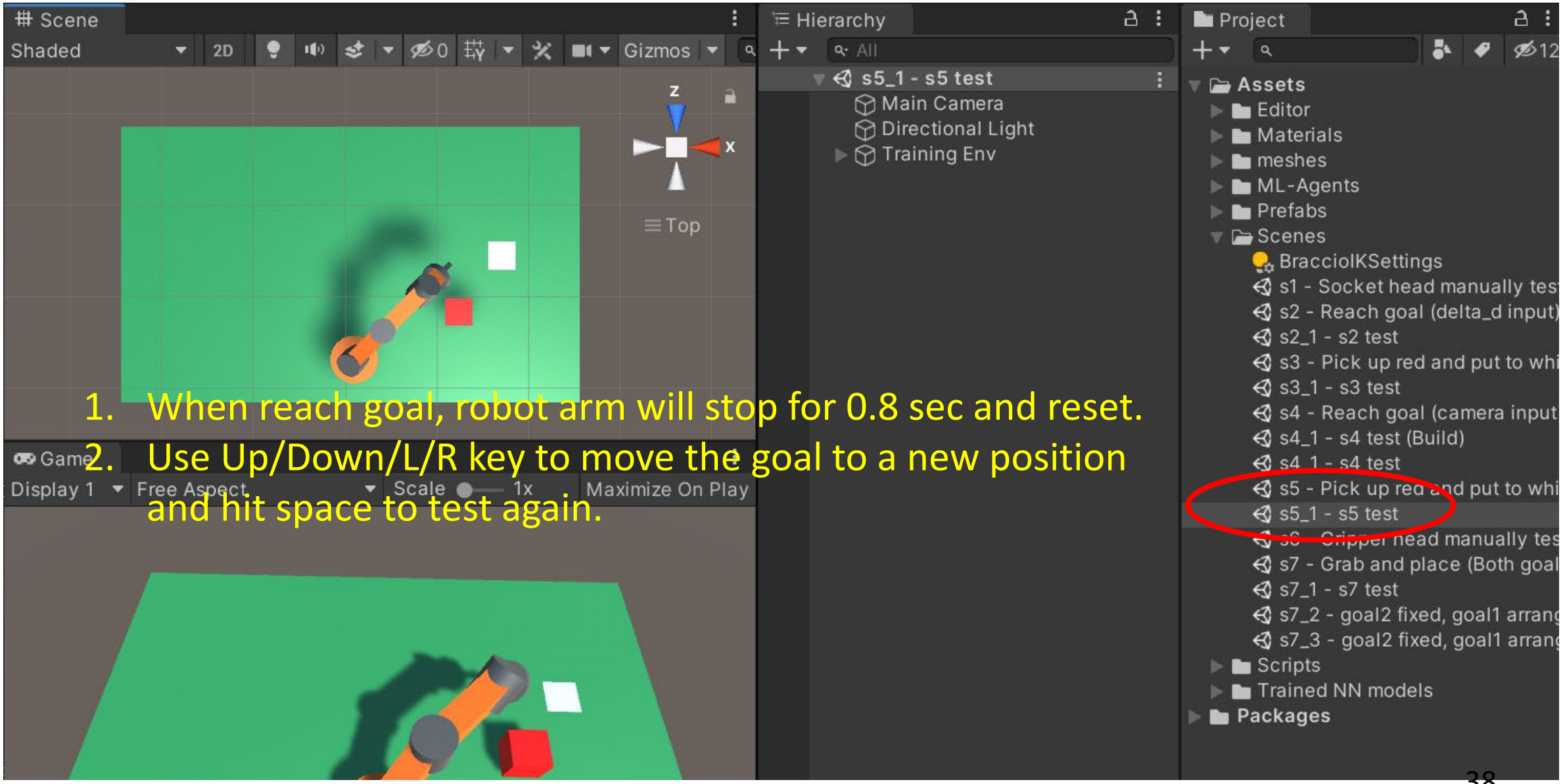
Arm initialize: $(\theta_B = 0, \theta_U = 45, \theta_L = 45, \theta_W = 45)$

Intermediate results after 2.4M

```
TouchCube. Step: 900000. Time Elapsed: 3622.903 s. Mean Reward: 42.232. Std of Reward: 51.025. 
TouchCube. Step: 950000. Time Elapsed: 3817.672 s. Mean Reward: 43.436. Std of Reward: 52.013. 
TouchCube. Step: 1000000. Time Elapsed: 4012.238 s. Mean Reward: 40.262. Std of Reward: 51.225.
ation.py:93] Converting to results\1\TouchCube\TouchCube-999986.onnx
ation.py:105] Exported results\1\TouchCube\TouchCube-999986.onnx
TouchCube. Step: 1050000. Time Elapsed: 4198.270 s. Mean Reward: 41.678. Std of Reward: 51.639.
TouchCube. Step: 1100000. Time Elapsed: 4396.988 s. Mean Reward: 43.416. Std of Reward: 51.913.
TouchCube. Step: 1150000. Time Elapsed: 4593.338 s. Mean Reward: 43.914. Std of Reward: 51.657.
TouchCube. Step: 1200000. Time Elapsed: 4792.435 s. Mean Reward: 40.471. Std of Reward: 50.521.
TouchCube. Step: 1250000. Time Elapsed: 4984.331 s. Mean Reward: 42.493. Std of Reward: 51.315.
TouchCube. Step: 1300000. Time Elapsed: 5197.395 s. Mean Reward: 43.941. Std of Reward: 51.623.
TouchCube. Step: 1350000. Time Elapsed: 5403.109 s. Mean Reward: 47.774. Std of Reward: 51.834.
TouchCube. Step: 1400000. Time Elapsed: 5608.913 s. Mean Reward: 47.696. Std of Reward: 52.174.
TouchCube. Step: 1450000. Time Elapsed: 5836.131 s. Mean Reward: 45.999. Std of Reward: 51.536.
TouchCube. Step: 1500000. Time Elapsed: 6037.497 s. Mean Reward: 45.637. Std of Reward: 51.625.
ation.py:93] Converting to results\1\TouchCube\TouchCube-1499845.onnx
ation.py:105] Exported results\1\TouchCube\TouchCube-1499845.onnx
TouchCube. Step: 1550000. Time Elapsed: 6243.114 s. Mean Reward: 46.052. Std of Reward: 52.346.
TouchCube. Step: 1600000. Time Elapsed: 6435.717 s. Mean Reward: 48.292. Std of Reward: 52.147.
TouchCube. Step: 1650000. Time Elapsed: 6634.507 s. Mean Reward: 45.468. Std of Reward: 51.541.
TouchCube. Step: 1700000. Time Elapsed: 6835.629 s. Mean Reward: 49.736. Std of Reward: 51.866.
TouchCube. Step: 1750000. Time Elapsed: 7030.518 s. Mean Reward: 50.424. Std of Reward: 51.975.
TouchCube. Step: 1800000. Time Elapsed: 7232.622 s. Mean Reward: 52.385. Std of Reward: 52.520.
TouchCube. Step: 1850000. Time Elapsed: 7427.402 s. Mean Reward: 53.477. Std of Reward: 52.988.
TouchCube. Step: 1900000. Time Elapsed: 7624.994 s. Mean Reward: 49.200. Std of Reward: 52.777.
TouchCube. Step: 1950000. Time Elapsed: 7820.621 s. Mean Reward: 51.300. Std of Reward: 52.514.
TouchCube. Step: 2000000. Time Elapsed: 8031.034 s. Mean Reward: 51.976. Std of Reward: 52.098.
ation.py:93] Converting to results\1\TouchCube\TouchCube-1999899.onnx
ation.py:105] Exported results\1\TouchCube\TouchCube-1999899.onnx
TouchCube. Step: 2050000. Time Elapsed: 8223.305 s. Mean Reward: 51.876. Std of Reward: 52.860.
TouchCube. Step: 2100000. Time Elapsed: 8423.297 s. Mean Reward: 49.998. Std of Reward: 52.224.
TouchCube. Step: 2150000. Time Elapsed: 8609.655 s. Mean Reward: 51.952. Std of Reward: 52.659.
TouchCube. Step: 2200000. Time Elapsed: 8819.758 s. Mean Reward: 54.323. Std of Reward: 53.211.
TouchCube. Step: 2250000. Time Elapsed: 9014.827 s. Mean Reward: 48.848. Std of Reward: 52.707.
TouchCube. Step: 2300000. Time Elapsed: 9210.316 s. Mean Reward: 52.215. Std of Reward: 52.870.
TouchCube. Step: 2350000. Time Elapsed: 9401.985 s. Mean Reward: 49.891. Std of Reward: 52.837.
TouchCube. Step: 2400000. Time Elapsed: 9609.943 s. Mean Reward: 49.739. Std of Reward: 52.804.
```

Test – Play with robot arm

Open "s5_1"



1. When reach goal, robot arm will stop for 0.8 sec and reset.
2. Use Up/Down/L/R key to move the goal to a new position and hit space to test again.

HW5(2)

- Describe training setting
- Show tensor board plots and discuss your training performance
- Describe test performance (recorded data, interactive test)