

Environment "Find the Treasure" Documentation

Author: Shuo Jiang (jiangshuo_sd@126.com)

License: Northeastern University, Lab for Learning and Planning in Robotics(LLPR)

Introduction

The document introduces a environment for multi agent study as two agents are trying to find the treasure in a room. The code is implemented in python and provided with simple interfaces. The environment is decoupled with learning method so reader can try any algorithms on.

Scene Description

The task of the environment is explained as below

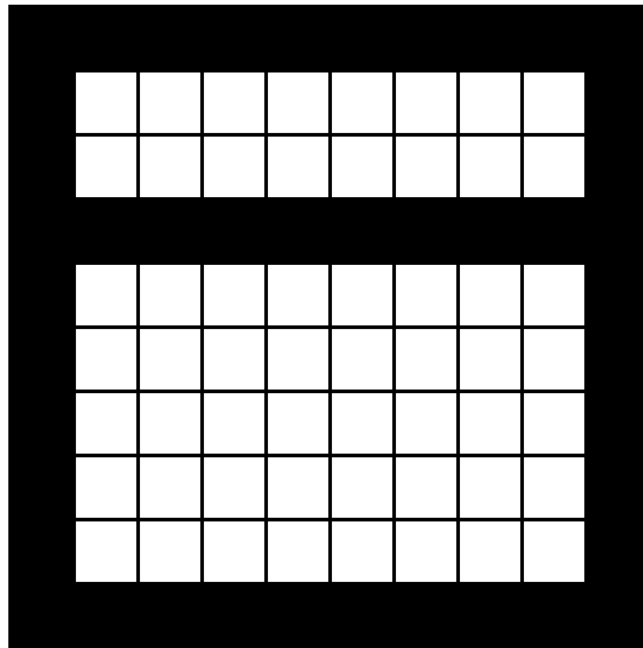


Figure 1

The environment consists of two rooms, two agents(agent 1 and agent2) will run in the larger room and the treasure is in the smaller room. The two agents, agent 1 and agent2 will be denoted using color **green**, and the treasure will be denoted using **red**.

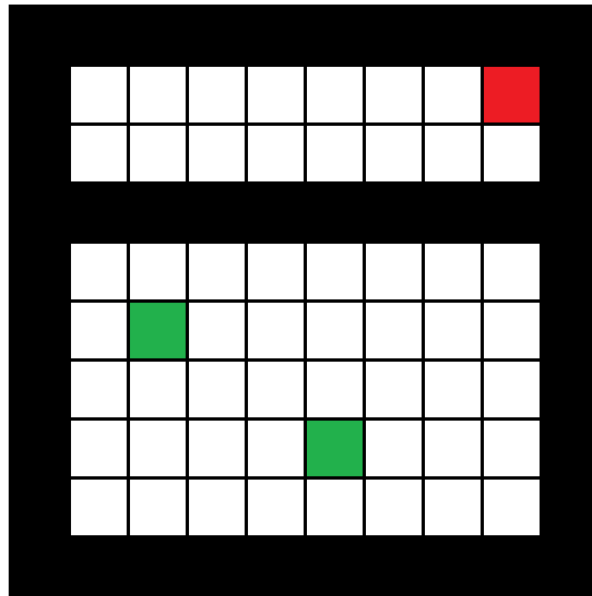


Figure 2

blocks in black are obstacles that agents could not step on and white ones are free spaces. We can see that the agents and the treasure are in separated rooms. There is a secret door on the wall in the middle. In the larger room, there will be a lever, denoted by yellow. If any of the agent is standing on the lever position, the secret door will open and the other agent get the access to the smaller room to get treasure. If any agent stands on the treasure position, the system reset, which means the two agents are randomly located in the larger room, as shown in Figure 3 and Figure 4

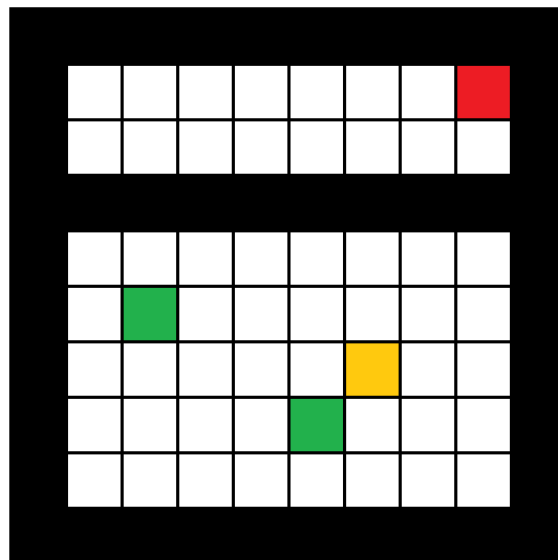


Figure 3. The lever position

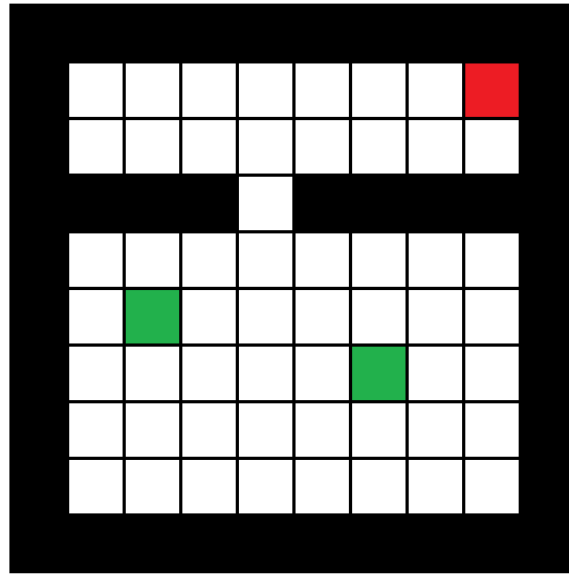


Figure 4. When an agent stands on the lever, the secret door opens

The coordinate system is show as

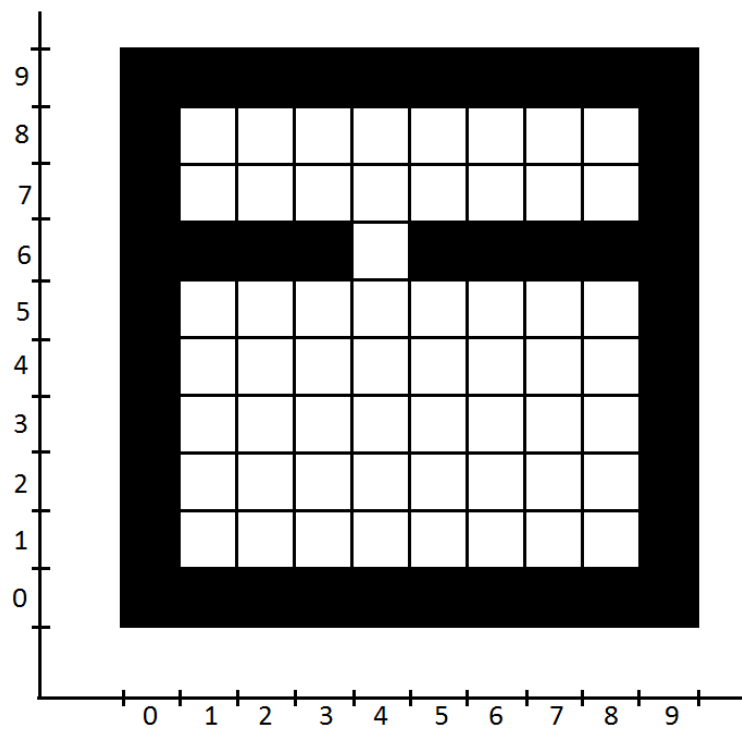


Figure 5

Each time any agent find the treasure will be rewarded 100 as the joint reward. Any moving action will be rewarded -1.

There are 5 possible action for the two agents

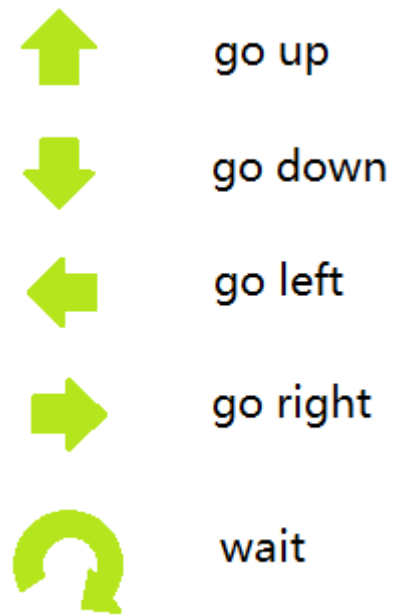


Figure 6

when the agent moves, it will move to the free space of 1 distance near it. However, when there is a block or the other agent is in its way, the action won't work.

Each agent will only observe what is directly visible in its vision. Which mean it can not observe something which is obstructed by other things. The following example shows

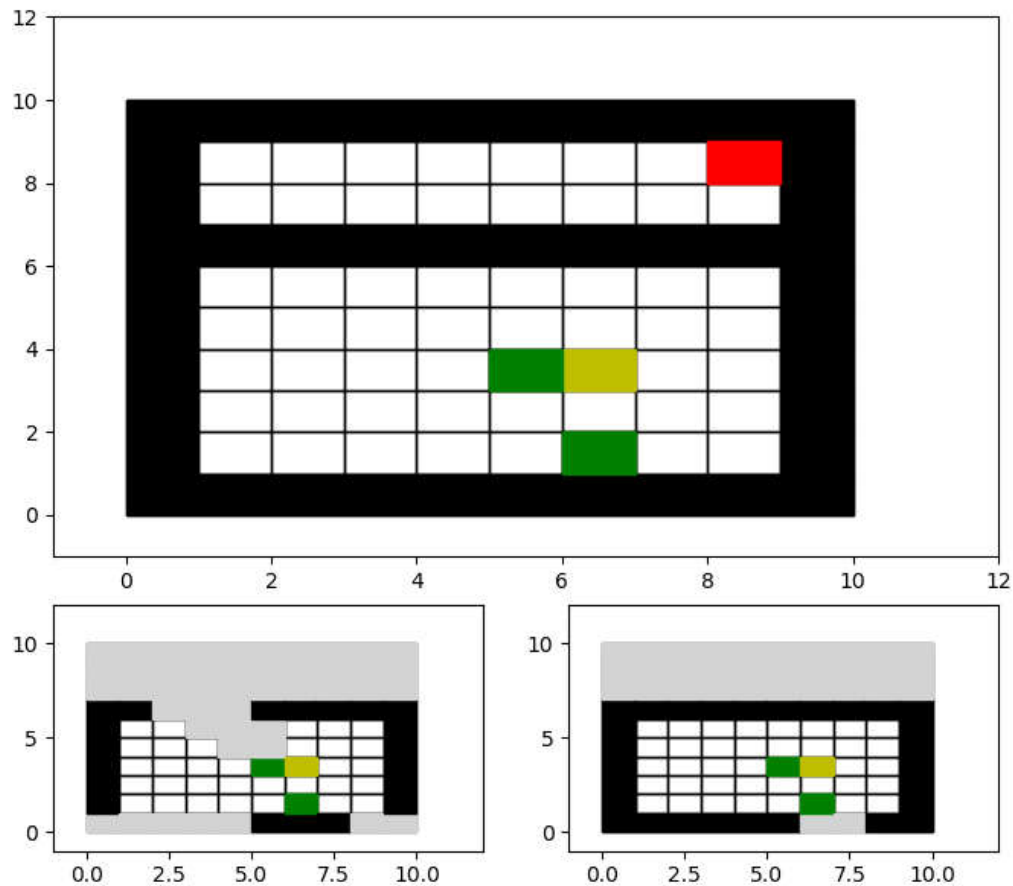


Figure 7

The top graph shows the fully observable environment. The bottom left graph is the vision of agent 1 and bottom right graph is the vision of agent 2. Readers can see in the agent 1's vision, agent 1 is at the bottom wall and agent 2 is near the lever, the area behind agent 2 is unobservable. Also the small room is totally unobservable unless the secret door opens.

Class Organization

The environment is programmed in python 3.6 and has very simple interfaces. The file contains the environment is "env_FindGoals.py" and a class " EnvFindTreasure" is defined.

The class has the following interfaces:

```
__init__(self):
get_agt1_obs(self):
get_agt2_obs(self):
step(self, action1, action2):
reset(self):
print_info(self):
plot_scene(self)
```

`__init__(self):`

initialize the object, important variables are

`agent1_pos`

`agent2_pos`

These are programmed by a list as `agent_pos=[3, 1]`

`get_agt1_obs(self):`

These function return the current vision of agent 1.

one example is the view of agent 1 is shown as

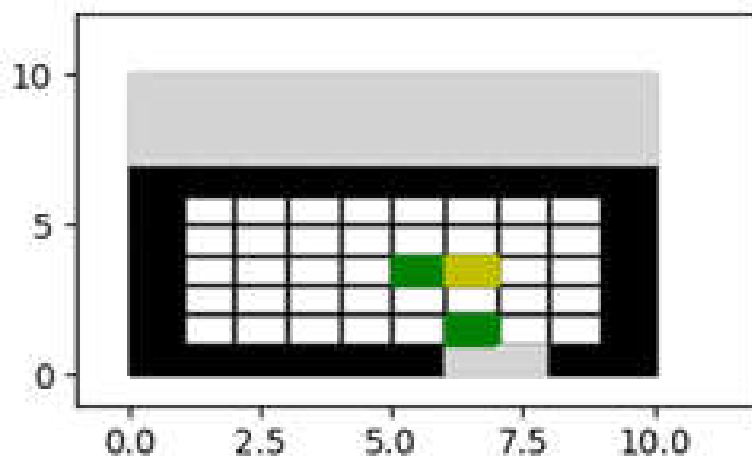


Figure 8

The returned observation is a numpy array. Each element is assigned a non-negative integer. 0 for free space, 1 for wall, 2 for agent, 3 for lever, 4 for unobserved area, 5 for treasure.

For example the observation of the Figure 8 is actually.

4	4	4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4	4	4
1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	2	3	0	0	1
1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	2	0	0	1
1	1	1	1	1	1	4	4	1	1

`step(self, action1, action2):`

The function update the environment by feeding the actions for agent1 and agent2. The actions are expressed by integers as






	go up	0
	go down	1
	go left	2
	go right	3
	wait	4

Figure 9

So the size of valid action is 5. The returned values of the function are reward, obs_1 and obs_2, which are the joint reward (sum of reward of agent 1 and agent 2) in the given step, the observation vector of agent 1 and agent 2.

`reset(self):`

This function randomly relocates the two agents in the large room

`print_info(self):`

print the current positions of agent 1 and agent2 in console

`plot_scene(self)`

This function plot the current state of the whole environment, and uses three subplots to show the views of each agent

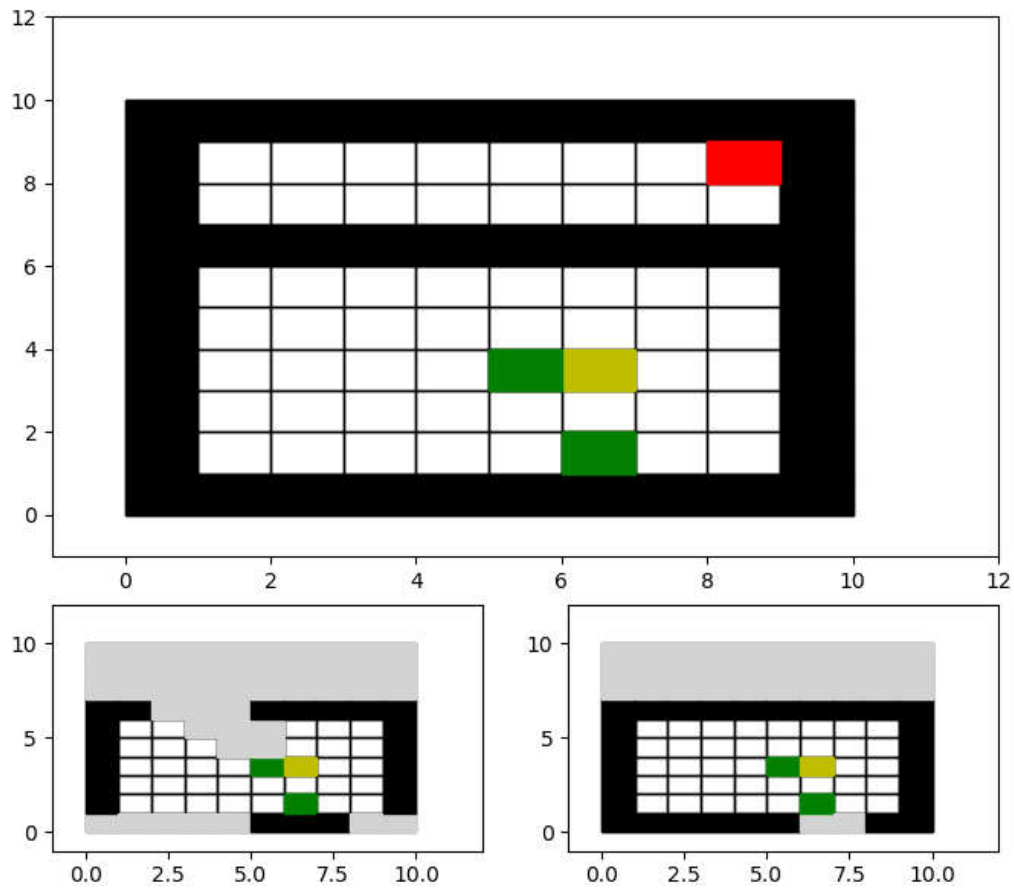


Figure 10

The global positions of agent1 and agent2 are shown as figure 10 top. The vision of agent 1 is shown in figure 10 bottom left, the light gray area is invisible to agent1. Also the figure 10 bottom right is the vision of agents 2.

Example

Here is an example using test function, and it is in "test_FindTreasure.py"

```
from env_FindTreasure import EnvFindTreasure
import random

env = EnvFindTreasure()

max_iter = 1000
for i in range(max_iter):
    print("iter= ", i)
    reward, obs_1, obs_2 = env.step(random.randint(0, 4), random.randint(0, 4))
```



```
env.plot_scene()  
env.print_info()
```