Environment "Find the Treasure" Documentation

Author: Shuo Jiang (jiang.shuo@husky.neu.edu)
License: Northeastern University, Lab for Learning and Planning in Robotics(LLPR)

Introduction

The document introduces an 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

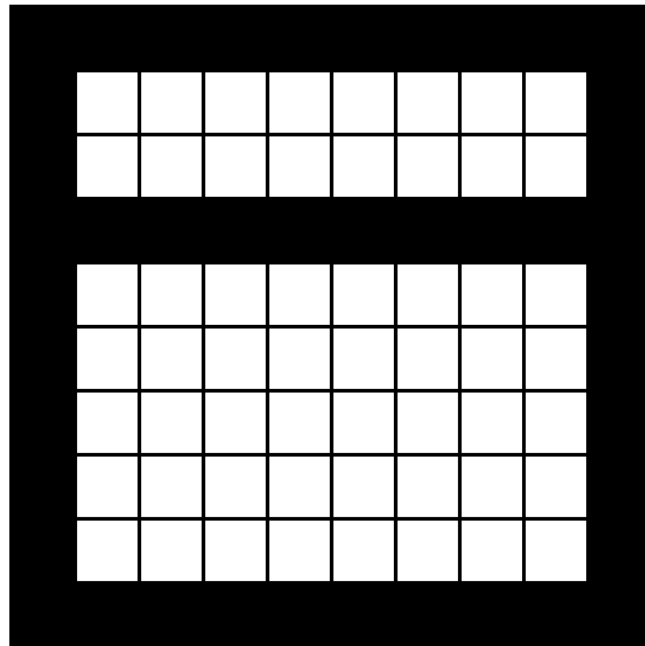The task of the environment is explained as below

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 red and green, and the treasure will be denoted using blue.
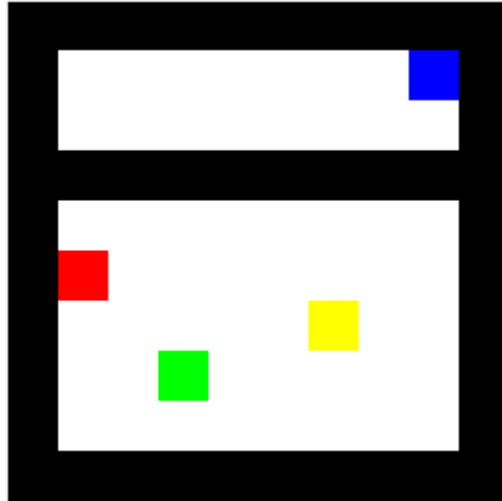
Figure 2

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 gets the access to the smaller room to get treasure. If any agent stands on the treasure position, the system resets, which means the two agents are randomly relocated in the larger room, as shown in Figure 3
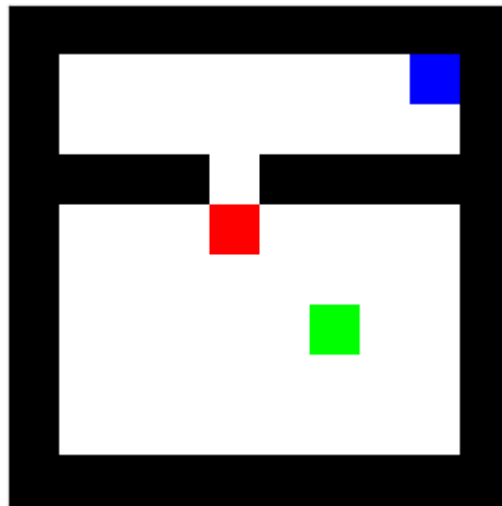

Figure 3. When an agent stands on the lever, the secret door opens

Coordinate System
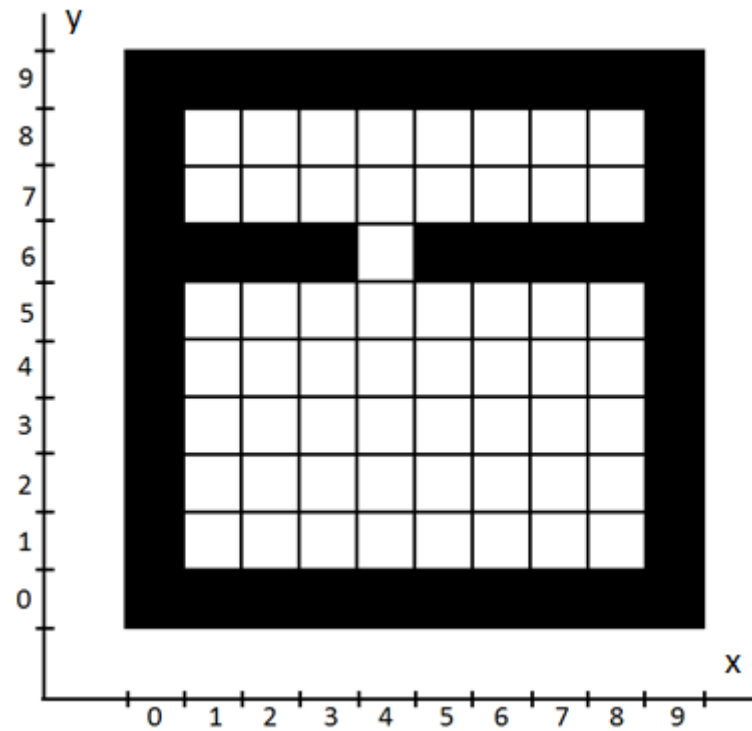
The coordinate system is show as

Figure. 4 Coordinate System

Each time any agent finds the treasure will be rewarded 100 for each agent. Any moving action will be rewarded -1 (except "wait"), bump to the wall will be rewarded -20.

Action Space

There are 5 possible action for the two agents

⬆	go up	0
⬇	go down	1
⬅	go left	2
➡	go right	3
↻	wait	4

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. The five actions are indexed by an integer 0 - 4.

Observation

Each agent can only observe what is directly visible in its vision. Which means it cannot observe something which is blocked by other things. The following example shows
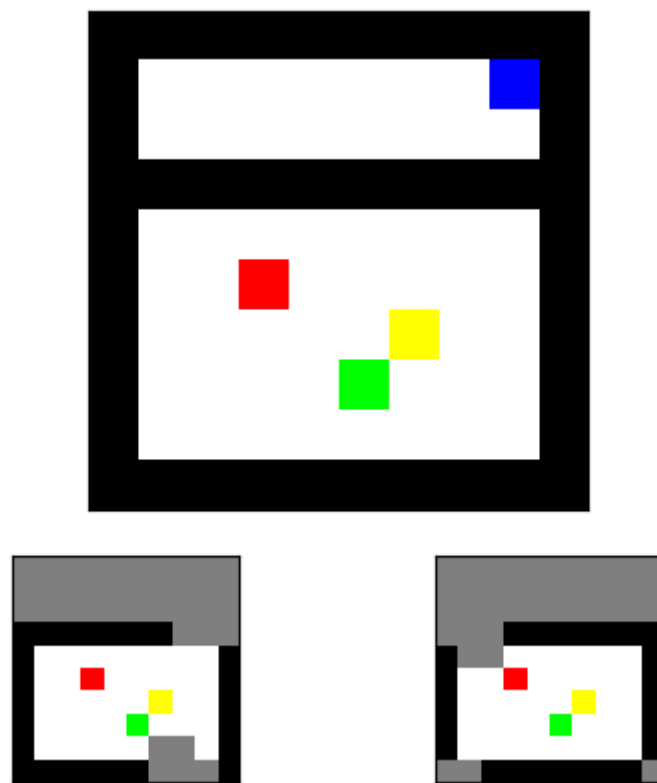

Figure. 6

The top plot shows the fully observed environment. The bottom left plot is the view of agent 1 and bottom right plot is the view of agent 2. Readers can see in the agent 1(red)'s view, the area behind agent 2(green) is unobservable. 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_FindTreasure.py" and a class " EnvFindTreasure" is defined.

The class has the following interfaces:

```
__init__()
obs_list = get_obs()
obs = get_full_obs()
reward_list, done = step(action_list)
reset()
set_agt1_at(new_pos)
set_agt2_at(new_pos)
```

```
__init__()
```
initialize the object, important variables are
```
agent1_pos
agent2_pos
```
These are programmed by a list as agent_pos = [3, 1]
Notice: Do not set value to agent_pos directly, use `set_agt1_at` function instead.

```
get_obs()
```
This function returns a list contains current views of agent and agent2.
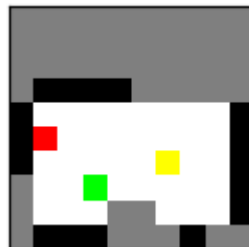one example is the view of agent 1 is shown as

Figure. 7

Each returned observation is a numpy array with size (10,10, 3) as an image. obs_list = [obs1, obs2]

```
get_full_obs()
```
return global view with no fog, a numpy array with size (10,10, 3)

```
step(action_list)
```
The function updates the environment by feeding the actions for agent1 and agent2. The actions are indexed by integers as

⬆	go up	0
⬇	go down	1
⬅	go left	2
➡	go right	3
↻	wait	4

Figure. 8

The size of valid action is 5. The returned values of the function are reward_list, done, which are the rewards of agent 1 and agent 2 in the given step, and whether the episode is done. reward_list = [reward1, reward2]

```
reset()
```
This function randomly relocates the two agents in the large room

```
set_agt1_at(new_pos)
```
This function set agent 1 at given position.

Example

Here is an example using test function, and it is in "test_FindTreasure.py". The actions for agents are random chosen, click and run.

```python
from env_FindTreasure import EnvFindTreasure
import random

if __name__ == '__main__':
    env = EnvFindTreasure()

    max_iter = 1000
    for i in range(max_iter):
```

```python
print("iter= ", i)
action_list = [random.randint(0, 4), random.randint(0, 4)]
reward_list, done = env.step(action_list)

env.plot_scene()
```