Environment "Catch the Pigs" Documentation

Author: Shuo Jiang (jiangshuo_sd@126.com)

Introduction

The document introduces a environment for multi agent study as two agents are trying to catch a moving pig in the pigsty. 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

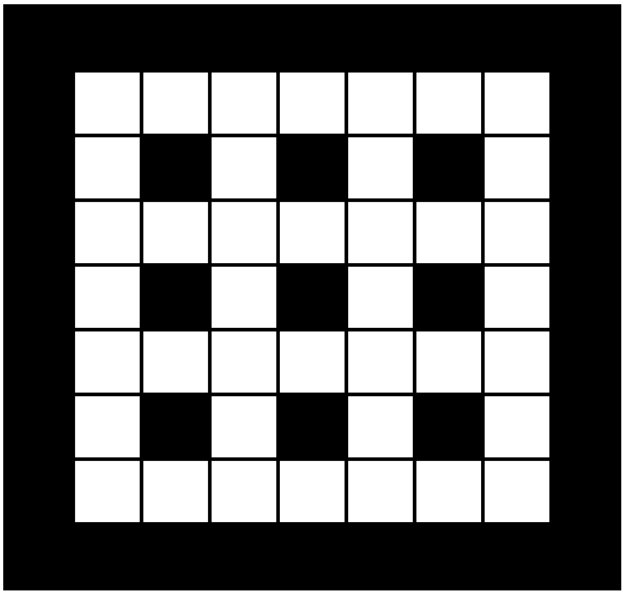


Figure 1

The environment works like a pigsty, two agents(agent 1 and agent2) will run in the pigsty and try to catch the moving pig. The two agents, agent 1 and agent2 will be denoted using color red and blue, and will be shown as red and blue rectangles in the maze, and the pig will be denoted using color green like:

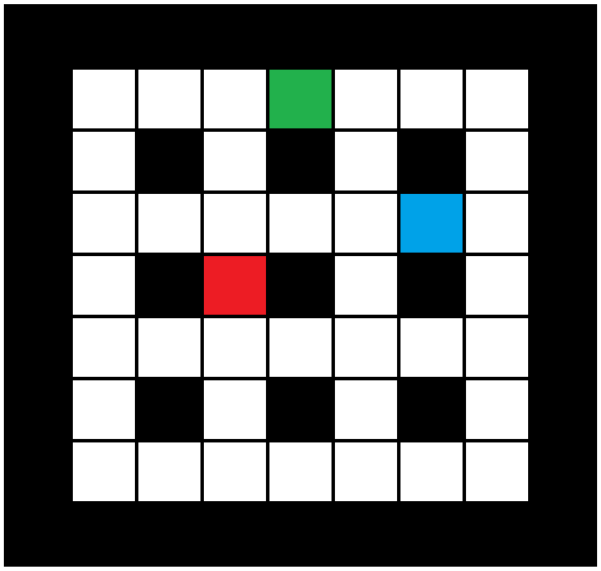


Figure 2

blocks in black are obstacles that agents could not step on and white ones are free spaces. The coordinate system is show as

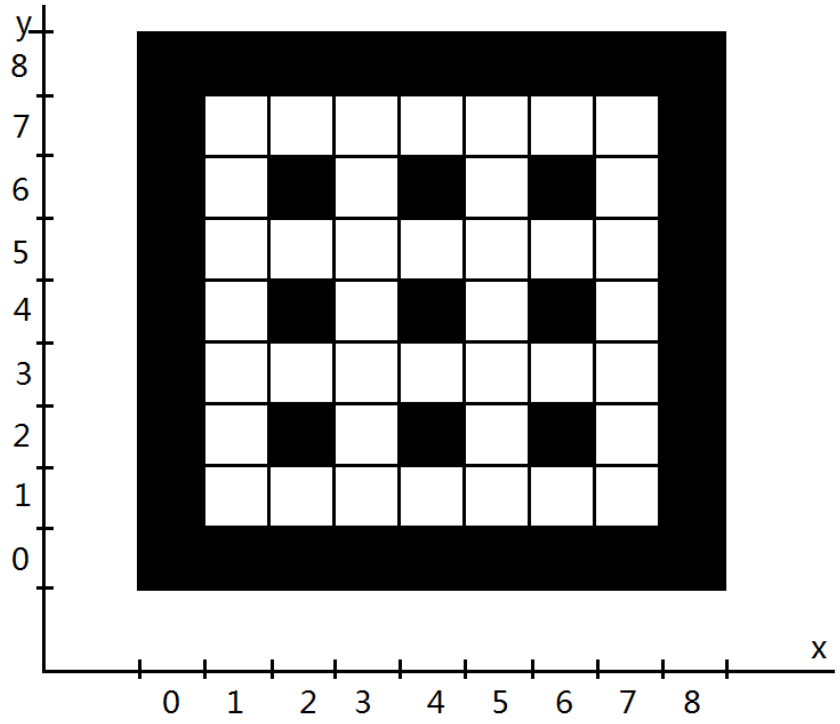


Figure 3

The two agents and the pig will be at random positions at the beginning with no overlapping. When the two agents catch the pig, the environment will reset and the three things will be set at

random positions again.

There are 5 possible actions for the two agents

Figure 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.

Actions for the pig are similar but without catch action

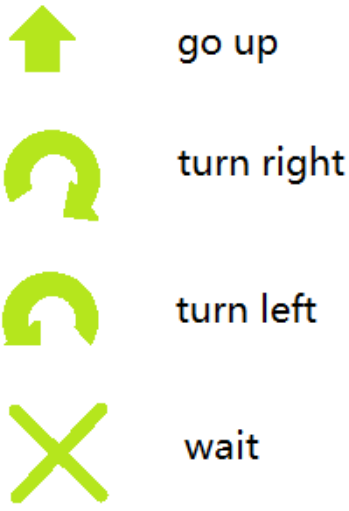


Figure 5

We assume the pig is strong and only when two agents simultaneously using catch action, the pig can be caught. When the two agents using action catch, the pig must be in the 1 nearby position

of any one. (means the only when agents are close enough that catch action can work, and simultaneous catching can succeed). When the pig is caught, the environment resets and all the three things will be randomly relocated. And the agent 1 and agent 2 will get a positive reward of 50 and pig gets negative reward of -50.

Also the environment can be locally observed by each thing. Each thing (agents and pig) has 4 orientations, and they can only observe about 90 degree of environment in front of it. Orientation will be indicated by an integer 0, 1, 2, 3 as shown in graph.

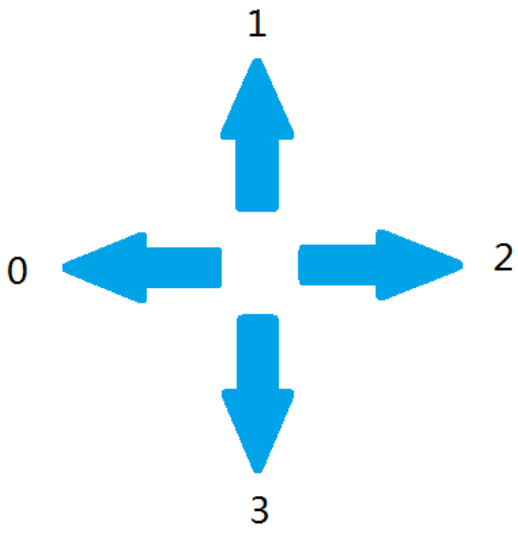


Figure 6

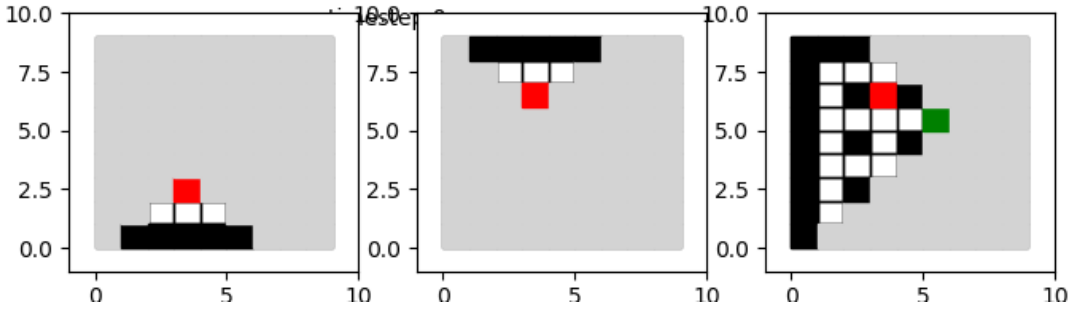


Figure 6

An example of observation is shown in Figure 6. From left to right are observations of agent 1, agent 2 and agent 3. As shown in the graph, agent 1 is facing down, agent 2 is facing up and pig is facing left. The light grey areas are unobserved areas.

So the observation is a numpy array. Using the pig's view as example, the free space is 0, obstacle is 1, agent 1 and agent 2 are denoted as 2 (in observation we do not distinguish agent 1 and agent 2), pig is denoted as 3. Unknown areas are denoted as 4.

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

Figure 7

Class Organization

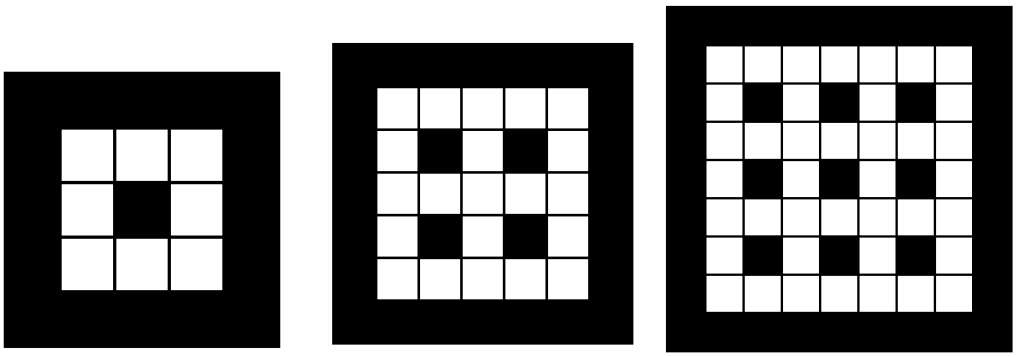
The environment is programmed in python 3.6 and has very simple interfaces. The file contains the environment is "env_CatchPigs.py" and a class " EnvCatchPigs
" is defined.

The class has the following interfaces:

__init__(self, size):
get_agt1_obs(self):
get_agt2_obs(self):
get_pig_obs(self)
step(self, action1, action2, action_pig):
reset(self):
plot_scene(self):
print_info(self)

__init__(self, size):
initialize the object, important variables are
`agt1_pos`
`agt1_ori`

`agt2_pos`
`agt2_ori`
`pig_pos`
`pig_ori`
These are programmed by a list as agt_pos=[3, 1], agt_ori=0
Also, the parameter size controls how large the map is, it should be assigned as an odd integer no smaller than 5. and the generated maps are as



size=5 size=7 size =9

get_agt1_obs(self):
These function return the current vision of agent 1 by returning a array of size (9, 9), indexed as

step(self, action1, action2, action_pig):
The function update the environment by feeding the actions for agent1 and agent2 and pig. The actions are expressed by integers as shown in Figure 4 and 5.

reset(self):
This function randomly relocates the two agents and the pig

plot_scene(self):
This function plot the current state of the whole environment, and uses three subplots to show the views of each agent and pig

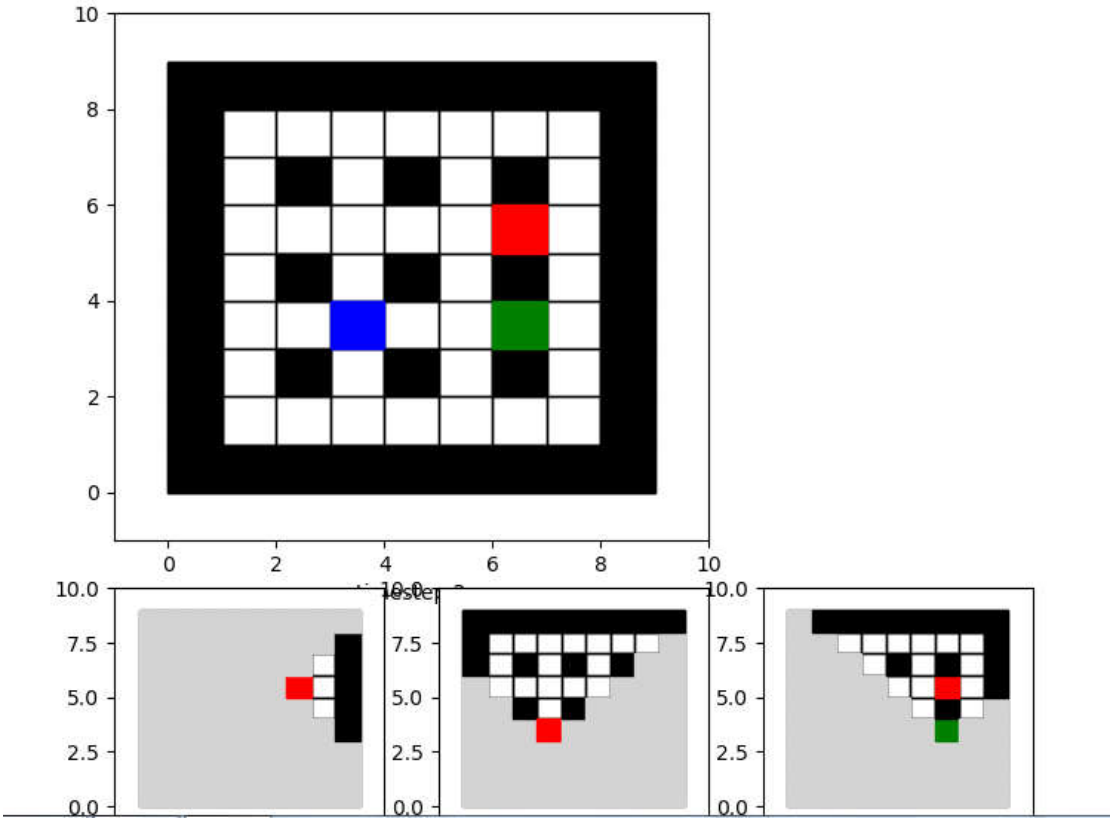


Figure 8

The global positions of agent1 (red) and 2 (blue) and pig (green) are shown as figure 8 top. The visions of things are listed at bottom, from left to right are view of agent 1, view of agent 2 and view of pig. In the view plot, the two agents are using the same color (red). Unobserved area are shown using light grey.

print_info(self)

This function prints some information on the console like the positions and orientations of agents and pig.

Example

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

```
from env_CatchPigs import EnvCatchPigs
import random

env = EnvCatchPigs(7)
max_iter = 10000
for i in range(max_iter):
    print("iter= ", i)
```

```
    reward_1, reward_2, reward_pig, obs_1, obs_2, obs_pig =
env.step(random.randint(0, 4), random.randint(0, 4), random.randint(0, 3))
    env.plot_scene()
    env.print_info()
```