Environment "GoTogether" Documentation

Author: Shuo Jiang (jiang.shuo@husky.neu.edu)

Introduction

The document introduces an environment for multi agent study as two agents (or more) search for a way to the goal. 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.

Scenario Description

The task of the environment is explained as below

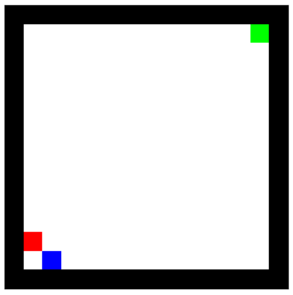

Figure 1

In the environment, agents (marked using color red and blue, the positions can overlap) will search for a way to the goal position (green). Where the black blocks are walls that agents cannot go through.

The coordinate system is shown as

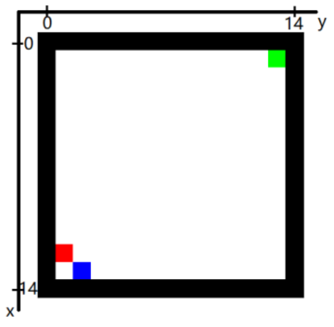

Figure 2

The area can be set as any large, and agents are born at left bottom.

There are 4 possible actions for the two agents go up/down/left/right, using an integer as 0/1/2/3.

The observation is an image (numpy array) as figure 1, with format (width = map size, height = map size, rgb_channel = 3)

When agents are both at the green position, it gets reward of 10. The important thing in the environment is that the two agents should keep in certain range. When the agents are too close or too far apart, they will get -0.5 reward, that is why it is called 'go together'

Class Organization

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

The class has the following interfaces:

__init__(self, size)
reset(self)
state = get_state(self)
obs = get_global_obs(self)
step(self, action_list)
render(self)

__init__(self)
initialize the object, important variables are
size: can be change, in example is 15.

get_ state (self):
return the normalized positions for all agents, which is a numpy array of shape [1, 4].

get_global_obs (self):
This function return a list of current visions of the scenario as an image, with form (width = size, height = size, rgb_channel = 3) as shown in Figure1

step(self, action_list):
The function updates the environment by feeding the actions for each agent. The actions are

denoted by integers 0/1/2/3. action_list = [action1, action2, ...]. The return is shared reward as an integer

reset(self):
This function relocates the agents to the initial position

render(self)
This function plot scene in animation, call in each step

Example

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

```
from env_GoTogether import EnvGoTogether
import random

if __name__ == '__main__':
    env = EnvGoTogether(15)
    max_iter = 100000
    for i in range(max_iter):
        print("iter= ", i)
        env.plot_scene()
        action_list = [random.randint(0, 3), random.randint(0, 3)]
        reward, done = env.step(action_list)
        if done:
            print('find goal, reward', reward)
            env.reset()
```