1_2

March 22, 2021

[1]:
```
# install required system dependencies
# install required system dependencies
!apt-get install -y xvfb x11-utils
!apt-get install x11-utils > /dev/null 2>&1
!pip install PyVirtualDisplay==2.0.* \
  PyOpenGL==3.1.* \
  PyOpenGL-accelerate==3.1.* \
  gym[box2d]==0.17.*
!pip install pyglet
```

```
Reading package lists… Done
Building dependency tree
Reading state information… Done
The following additional packages will be installed:
  libxxf86dga1
Suggested packages:
  mesa-utils
The following NEW packages will be installed:
  libxxf86dga1 x11-utils xvfb
0 upgraded, 3 newly installed, 0 to remove and 30 not upgraded.
Need to get 993 kB of archives.
After this operation, 2,981 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/main amd64 libxxf86dga1 amd64
2:1.1.4-1 [13.7 kB]
Get:2 http://archive.ubuntu.com/ubuntu bionic/main amd64 x11-utils amd64
7.7+3build1 [196 kB]
Get:3 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 xvfb amd64
2:1.19.6-1ubuntu4.8 [784 kB]
Fetched 993 kB in 1s (1,316 kB/s)
Selecting previously unselected package libxxf86dga1:amd64.
(Reading database … 160980 files and directories currently installed.)
Preparing to unpack …/libxxf86dga1_2%3a1.1.4-1_amd64.deb …
Unpacking libxxf86dga1:amd64 (2:1.1.4-1) …
Selecting previously unselected package x11-utils.
Preparing to unpack …/x11-utils_7.7+3build1_amd64.deb …
Unpacking x11-utils (7.7+3build1) …
Selecting previously unselected package xvfb.
Preparing to unpack …/xvfb_2%3a1.19.6-1ubuntu4.8_amd64.deb …
```

```
Unpacking xvfb (2:1.19.6-1ubuntu4.8) …
Setting up xvfb (2:1.19.6-1ubuntu4.8) …
Setting up libxxf86dga1:amd64 (2:1.1.4-1) …
Setting up x11-utils (7.7+3build1) …
Processing triggers for man-db (2.8.3-2ubuntu0.1) …
Processing triggers for libc-bin (2.27-3ubuntu1.2) …
/sbin/ldconfig.real: /usr/local/lib/python3.7/dist-
packages/ideep4py/lib/libmkldnn.so.0 is not a symbolic link

Collecting PyVirtualDisplay==2.0.*
  Downloading https://files.pythonhosted.org/packages/ad/05/6568620fed440941b704
664b9cfe5f836ad699ac7694745e7787fbdc8063/PyVirtualDisplay-2.0-py2.py3-none-
any.whl
Requirement already satisfied: PyOpenGL==3.1.* in /usr/local/lib/python3.7/dist-
packages (3.1.5)
Collecting PyOpenGL-accelerate==3.1.*
  Downloading https://files.pythonhosted.org/packages/a2/3c/f42a62b7784c04
b20f8b88d6c8ad04f4f20b0767b721102418aad94d8389/PyOpenGL-accelerate-3.1.5.tar.gz
(538kB)
     |                 | 542kB 5.5MB/s
Requirement already satisfied: gym[box2d]==0.17.* in
/usr/local/lib/python3.7/dist-packages (0.17.3)
Collecting EasyProcess
  Downloading https://files.pythonhosted.org/packages/48/3c/75573613641c90c6d094
059ac28adb748560d99bd27ee6f80cce398f404e/EasyProcess-0.3-py2.py3-none-any.whl
Requirement already satisfied: numpy>=1.10.4 in /usr/local/lib/python3.7/dist-
packages (from gym[box2d]==0.17.*) (1.19.5)
Requirement already satisfied: pyglet<=1.5.0,>=1.4.0 in
/usr/local/lib/python3.7/dist-packages (from gym[box2d]==0.17.*) (1.5.0)
Requirement already satisfied: cloudpickle<1.7.0,>=1.2.0 in
/usr/local/lib/python3.7/dist-packages (from gym[box2d]==0.17.*) (1.3.0)
Requirement already satisfied: scipy in /usr/local/lib/python3.7/dist-packages
(from gym[box2d]==0.17.*) (1.4.1)
Collecting box2d-py~=2.3.5; extra == "box2d"
  Downloading https://files.pythonhosted.org/packages/87/34/da5393985c3ff9
a76351df6127c275dcb5749ae0abbe8d5210f06d97405d/box2d_py-2.3.8-cp37-cp37m-manylin
ux1_x86_64.whl (448kB)
     |                 | 450kB 7.0MB/s
Requirement already satisfied: future in /usr/local/lib/python3.7/dist-
packages (from pyglet<=1.5.0,>=1.4.0->gym[box2d]==0.17.*) (0.16.0)
Building wheels for collected packages: PyOpenGL-accelerate
  Building wheel for PyOpenGL-accelerate (setup.py) … done
  Created wheel for PyOpenGL-accelerate:
filename=PyOpenGL_accelerate-3.1.5-cp37-cp37m-linux_x86_64.whl size=1599126
sha256=1e56fc3d26c289b391c496f3f876dc66f5ceb232ea30e48917ad0ff694596f78
  Stored in directory: /root/.cache/pip/wheels/bd/21/77/99670ceca25fddb3c2b60a7a
e44644b8253d1006e8ec417bcc
Successfully built PyOpenGL-accelerate
```

```
Installing collected packages: EasyProcess, PyVirtualDisplay, PyOpenGL-
accelerate, box2d-py
Successfully installed EasyProcess-0.3 PyOpenGL-accelerate-3.1.5
PyVirtualDisplay-2.0 box2d-py-2.3.8
Requirement already satisfied: pyglet in /usr/local/lib/python3.7/dist-packages
(1.5.0)
Requirement already satisfied: future in /usr/local/lib/python3.7/dist-packages
(from pyglet) (0.16.0)
```

[3]:
```
## Library ##
import gym
import numpy as np
import base64
import io
import IPython
```

[114]:
```
import random
MAX_REWARD = 5000000

class World(gym.Env):
    def __init__(self, position=[0, 0], max_moves=30, grid_size=[2, 2],␣
 ↪dirt=[[1, 1], [1, 1]]):
        """
            Defaults grid: 2 x 2.
            Map: [[1, 1],
                  [1, 1]].
        """
        metadata = {'render.mode': ['human']}
        super(World, self).__init__()

        self.initial_position = position
        self.max_moves = max_moves
        self.grid_size = grid_size
        self.initial_dirt = dirt

        self.reward_range = (0, MAX_REWARD)

    def reset(self):
        self.score = 0
        self.move = 0
        self.position = self.initial_position
        self.dirt = self.initial_dirt

        observation = self._next_observation()
        self.visited = set()
        self.visited.add((self.position[0], self.position[1]))
        return observation

    def step(self, action):
      self.perform_action(action)
      reward = self.score
      observation = self._next_observation()
      done = self.move == self.max_moves
      info = {}

      return observation, reward, done, info
    def _next_observation(self):
        obs = {'C': 0, 'R' : 0, 'L' : 0, 'U' : 0, 'D' : 0}
        obs['C'] = dirt[self.position[0]][self.position[1]]
        if self._crosses_boundary('R') == False:
            obs['R'] = dirt[self.position[0]][self.position[1] + 1]
        if self._crosses_boundary('L') == False:
            obs['L'] = dirt[self.position[0]][self.position[1] - 1]
        if self._crosses_boundary('U') == False:
            obs['U'] = dirt[self.position[0] - 1][self.position[1]]
        if self._crosses_boundary('D') == False:
            obs['D'] = dirt[self.position[0] + 1][self.position[1]]
        next_action = self._action_space()

        return [next_action, self.position, obs]
    def perform_action(self, action):
        self.move += 1
        if action == 'R':
            self.position[1] += 1
            self.score = 0
        if action == 'L':
            self.position[1] -= 1
            self.score = 0
        if action == 'U':
            self.position[0] -= 1
            self.score = 0
        if action == 'D':
            self.position[0] += 1
            self.score = 0
        if action == 'S':
            self.score += self.dirt[self.position[0]][self.position[1]]
            self.dirt[self.position[0]][self.position[1]] = 0

        self.visited.add((self.position[0], self.position[1]))

    def _crosses_boundary(self, action):
        """This function checks if action taken by the agent will cross␣
↪boundary.
```

```
        Returns:
            boolean: True if boundary will be crossed
        """
        if action == 'R':
            if self.position[1]+1 > self.grid_size[1]-1:
                return True
        if action == 'L':
            if self.position[1]-1 < 0:
                return True
        if action == 'U':
            if self.position[0]-1 < 0:
                return True
        if action == 'D':
            if self.position[0]+1 > self.grid_size[0]-1:
                return True
        return False

    def _action_space(self):
        if self.dirt[self.position[0]][self.position[1]] > 0:
            return ['S']
        else:
            ls = ['R', 'L', 'U', 'D']
            for i in ['R', 'L', 'U', 'D']:
                if self._crosses_boundary(i):
                   ls.remove(i)
            return ls


    def render(self):
        if self.move % 5 == 0:
            self.print_dirt()

    def print_dirt(self):
        """This function prints the current world representation with dirt in
        each tile
        """
        part = self.dirt[:self.position[0]]
        print()
        for row in part:
            print(*row, sep=", ")

        current = self.dirt[self.position[0]]
        print(*current[:self.position[1]], sep=", ", end=" ")
        print("["+str(self.dirt[self.position[0]][self.position[1]])
              + "]", end=" ")
        print(*current[self.position[1]+1:], sep=", ")
        part = self.dirt[self.position[0]+1:]
        for row in part:
            print(*row, sep=", ")
        print()
```

[109]:
```
def policy(past_action, reward, obs):
    action_space, position, sur = obs
    # if robot is not cleaning
    print(obs)
    if past_action != 'S':
      # if robot is cleaned area (surroungding cells are cleaned
      if (sur['C'] == 0) and (sur['L'] == 0) and (sur['R'] == 0) and (sur['U']␣
↪== 0) and (sur['D'] == 0):
          if reward > 0:
            reward -= 0.01
    return reward, random.choice(action_space)
```

[115]:
```
    #Map information
    grid = [8, 5]
    dirt = [[0, 0.5, 0.8, 0.1, 0.1],
            [0.1, 0, 0.5, 0.5, 0.5],
            [0.3, 0.5, 0.4, 0.3, 0.2],
            [0.3, 0.1, 0.7, 0.8, 0.2],
            [0, 0, 0.2, 0.8, 0.3],
            [0, 0, 0, 0.5, 0.1],
            [0, 0, 0, 0.5, 0.1],
            [0, 0, 0, 0.2, 0]]
    moves = 30
    pos = [6, 0]

    # Random setting
    seed = 1
    random.seed(seed)


    #Env load
    env = World(pos, moves, grid, dirt)
    obs = env.reset()
    env.render()

    #inititial paramettters
    past_action = ''
    reward = 0
    for _ in range(30):
      # action = random.choice(env._action_space())
      reward, action = policy(past_action, reward, obs)
      obs, r, d, i = env.step(action) # Take action
```

```python
    reward += r
    print(action, round(reward, 5))

    past_action = action
    env.render()
    if d:
        env.reset()
```

```
0, 0.5, 0.8, 0.1, 0.1
0.1, 0, 0.5, 0.5, 0.5
0.3, 0.5, 0.4, 0.3, 0.2
0.3, 0.1, 0.7, 0.8, 0.2
0, 0, 0.2, 0.8, 0.3
0, 0, 0, 0.5, 0.1
 [0] 0, 0, 0.5, 0.1
0, 0, 0, 0.2, 0

[['R', 'U', 'D'], [6, 0], {'C': 0, 'R': 0, 'L': 0, 'U': 0, 'D': 0}]
R 0
[['R', 'L', 'U', 'D'], [6, 1], {'C': 0, 'R': 0, 'L': 0, 'U': 0, 'D': 0}]
R 0
[['R', 'L', 'U', 'D'], [6, 2], {'C': 0, 'R': 0.5, 'L': 0, 'U': 0, 'D': 0}]
U 0
[['R', 'L', 'U', 'D'], [5, 2], {'C': 0, 'R': 0.5, 'L': 0, 'U': 0.2, 'D': 0}]
R 0
[['S'], [5, 3], {'C': 0.5, 'R': 0.1, 'L': 0, 'U': 0.8, 'D': 0.5}]
S 0.5

0, 0.5, 0.8, 0.1, 0.1
0.1, 0, 0.5, 0.5, 0.5
0.3, 0.5, 0.4, 0.3, 0.2
0.3, 0.1, 0.7, 0.8, 0.2
0, 0, 0.2, 0.8, 0.3
0, 0, 0 [0] 0.1
0, 0, 0, 0.5, 0.1
0, 0, 0, 0.2, 0

[['R', 'L', 'U', 'D'], [5, 3], {'C': 0, 'R': 0.1, 'L': 0, 'U': 0.8, 'D': 0.5}]
D 0.5
[['S'], [6, 3], {'C': 0.5, 'R': 0.1, 'L': 0, 'U': 0, 'D': 0.2}]
S 1.0
[['R', 'L', 'U', 'D'], [6, 3], {'C': 0, 'R': 0.1, 'L': 0, 'U': 0, 'D': 0.2}]
D 1.0
[['S'], [7, 3], {'C': 0.2, 'R': 0, 'L': 0, 'U': 0, 'D': 0}]
S 1.2
```

```
[['R', 'L', 'U'], [7, 3], {'C': 0, 'R': 0, 'L': 0, 'U': 0, 'D': 0}]
R 1.2

0, 0.5, 0.8, 0.1, 0.1
0.1, 0, 0.5, 0.5, 0.5
0.3, 0.5, 0.4, 0.3, 0.2
0.3, 0.1, 0.7, 0.8, 0.2
0, 0, 0.2, 0.8, 0.3
0, 0, 0, 0, 0.1
0, 0, 0, 0, 0.1
0, 0, 0, 0 [0]

[['L', 'U'], [7, 4], {'C': 0, 'R': 0, 'L': 0, 'U': 0.1, 'D': 0}]
U 1.2
[['S'], [6, 4], {'C': 0.1, 'R': 0, 'L': 0, 'U': 0.1, 'D': 0}]
S 1.3
[['L', 'U', 'D'], [6, 4], {'C': 0, 'R': 0, 'L': 0, 'U': 0.1, 'D': 0}]
U 1.3
[['S'], [5, 4], {'C': 0.1, 'R': 0, 'L': 0, 'U': 0.3, 'D': 0}]
S 1.4
[['L', 'U', 'D'], [5, 4], {'C': 0, 'R': 0, 'L': 0, 'U': 0.3, 'D': 0}]
D 1.4

0, 0.5, 0.8, 0.1, 0.1
0.1, 0, 0.5, 0.5, 0.5
0.3, 0.5, 0.4, 0.3, 0.2
0.3, 0.1, 0.7, 0.8, 0.2
0, 0, 0.2, 0.8, 0.3
0, 0, 0, 0, 0
0, 0, 0, 0 [0]
0, 0, 0, 0, 0

[['L', 'U', 'D'], [6, 4], {'C': 0, 'R': 0, 'L': 0, 'U': 0, 'D': 0}]
L 1.39
[['R', 'L', 'U', 'D'], [6, 3], {'C': 0, 'R': 0, 'L': 0, 'U': 0, 'D': 0}]
D 1.38
[['R', 'L', 'U'], [7, 3], {'C': 0, 'R': 0, 'L': 0, 'U': 0, 'D': 0}]
L 1.37
[['R', 'L', 'U'], [7, 2], {'C': 0, 'R': 0, 'L': 0, 'U': 0, 'D': 0}]
U 1.36
[['R', 'L', 'U', 'D'], [6, 2], {'C': 0, 'R': 0, 'L': 0, 'U': 0, 'D': 0}]
L 1.35

0, 0.5, 0.8, 0.1, 0.1
0.1, 0, 0.5, 0.5, 0.5
0.3, 0.5, 0.4, 0.3, 0.2
0.3, 0.1, 0.7, 0.8, 0.2
0, 0, 0.2, 0.8, 0.3
```

```
0, 0, 0, 0, 0
0 [0] 0, 0, 0
0, 0, 0, 0, 0

[['R', 'L', 'U', 'D'], [6, 1], {'C': 0, 'R': 0, 'L': 0, 'U': 0, 'D': 0}]
R 1.34
[['R', 'L', 'U', 'D'], [6, 2], {'C': 0, 'R': 0, 'L': 0, 'U': 0, 'D': 0}]
U 1.33
[['R', 'L', 'U', 'D'], [5, 2], {'C': 0, 'R': 0, 'L': 0, 'U': 0.2, 'D': 0}]
R 1.33
[['R', 'L', 'U', 'D'], [5, 3], {'C': 0, 'R': 0, 'L': 0, 'U': 0.8, 'D': 0}]
R 1.33
[['L', 'U', 'D'], [5, 4], {'C': 0, 'R': 0, 'L': 0, 'U': 0.3, 'D': 0}]
L 1.33

0, 0.5, 0.8, 0.1, 0.1
0.1, 0, 0.5, 0.5, 0.5
0.3, 0.5, 0.4, 0.3, 0.2
0.3, 0.1, 0.7, 0.8, 0.2
0, 0, 0.2, 0.8, 0.3
0, 0, 0 [0] 0
0, 0, 0, 0, 0
0, 0, 0, 0, 0

[['R', 'L', 'U', 'D'], [5, 3], {'C': 0, 'R': 0, 'L': 0, 'U': 0.8, 'D': 0}]
R 1.33
[['L', 'U', 'D'], [5, 4], {'C': 0, 'R': 0, 'L': 0, 'U': 0.3, 'D': 0}]
U 1.33
[['S'], [4, 4], {'C': 0.3, 'R': 0, 'L': 0.8, 'U': 0.2, 'D': 0}]
S 1.63
[['L', 'U', 'D'], [4, 4], {'C': 0, 'R': 0, 'L': 0.8, 'U': 0.2, 'D': 0}]
U 1.63
[['S'], [3, 4], {'C': 0.2, 'R': 0, 'L': 0.8, 'U': 0.2, 'D': 0}]
S 1.83

0, 0.5, 0.8, 0.1, 0.1
0.1, 0, 0.5, 0.5, 0.5
0.3, 0.5, 0.4, 0.3, 0.2
0.3, 0.1, 0.7, 0.8 [0]
0, 0, 0.2, 0.8, 0
0, 0, 0, 0, 0
0, 0, 0, 0, 0
0, 0, 0, 0, 0
```