TS3 – Własne środowisko - Paweł Malisz, Mikołaj Białek

1. Opis gry

Implementowana przez nas gra polega na zmienianiu pasów jazdy przez samochód w zależności od przeszkód napotykanych na drodze. Samochód porusza się po jednym z dwóch dostępnych pasów. Na pasach mogą pojawiać się przeszkody, które należy ominąć poprzez zmianę pasu ruchu. Wygrana gry polega na dojechaniu do końca trasy bez uderzenia w żadną przeszkodę.

2. Cel ćwiczenia

Celem naszego ćwiczenia jest napisanie własnego środowiska wyżej opisanej gry w bibliotece Gymnasium oraz nauczenie agenta jej przechodzić.

3. Opis problemu

Na grę składają się dyskretne zbiory akcji i obserwacji. Zbiór akcji zawiera dwie liczby, 0 i 1, które oznaczają odpowiednio pasy, na których ma się znaleźć samochód.

Natomiast zbiór obserwacji jest słownikiem składającym się z dwóch informacji. Klucz ‘agent’ zawiera informacje o pozycji samochodu w danym momencie (liczba 0 lub 1), a klucz ‘obstacle’ to tablica 2-wymiarowa, gdzie pierwszym wymiarem jest numer pasa ruchu (0 lub 1), a drugim informacja czy na danym pasie, na danej pozycji znajduje się przeszkoda (0 – nie lub 1 - tak).

Za przejechanie do końca bez uderzenia w żadną przeszkodę agent dostaje nagrodę (+1 punkt), a za zderzenie z przeszkodą karę (-1 punkt).

4. Rozwiązanie problemu

4.1. Inicjalizacja i reset

W tym momencie musimy zrobić kilka rzeczy:

- Zerujemy liczników punktów (jest to ilość przejechanych pól),
- Losujemy położenie agenta (0 lub 1),
- Losujemy położenie przeszkód, uwzględniając jedno miejsce wolne, oraz wolny wiersz między poszczególnymi przeszkodami.

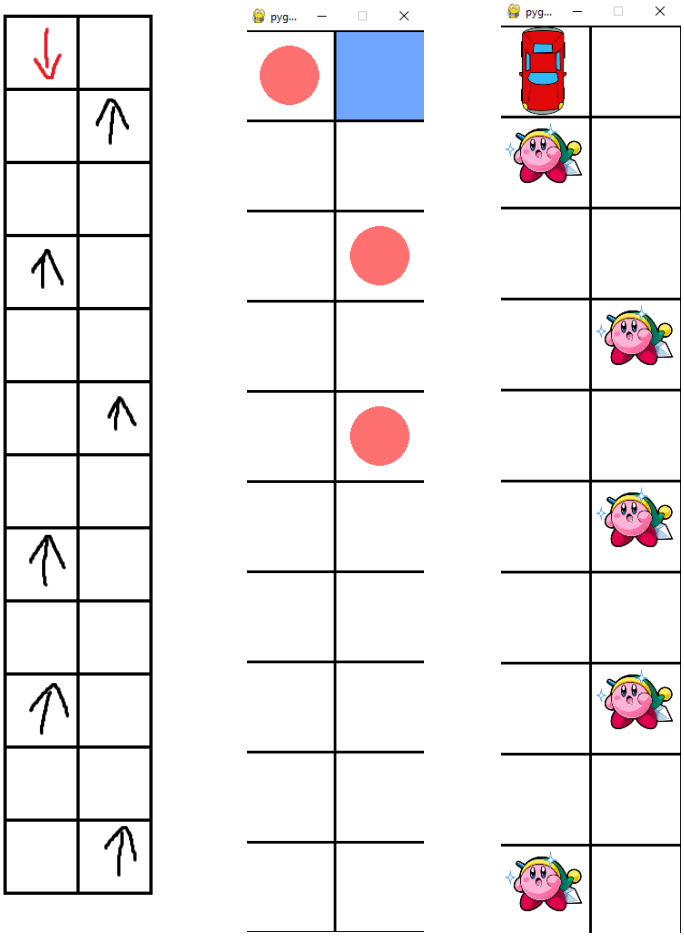
4.2. Funkcja kroku

Funkcja zwracają obserwacje, nagrodę, flagę czy środowisko skończone oraz informację o punktach. W jej środku wykonywane są poszczególne kroki:

1. Ustaw agenta na pozycji po wprowadzonej akcji.
2. Sprawdź czy agent po ustawieniu jest na pozycji przeszkody, jeśli tak to nagroda -1 i koniec.
3. Jeśli nie jest, to podnieś punkty o 1, oraz usuń pierwszy rząd drogi i dodaj pustą na sam koniec trasy [0, 0].
4. Jeśli punkty osiągną wielkość trasy to nagroda osiąga wartość 1 i koniec.

4.3. Funkcja do rysowania

Początkowy zamysł vs końcowy efekt:



Generujemy okno o rozmiarach 200 x 1000. Oraz mapę z kratkami 100 x 100 oraz ramce 3 pikseli.

Następnie na podstawie agenta generujemy samochód u góry na pozycji 0 lub 1. Później na podstawie przeszkód dodajemy je do mapy. Obok kod dodający agenta oraz przeszkody:

```
canvas = pygame.Surface((self.x, self.y))
canvas.fill((255, 255, 255))

carImg = pygame.image.load('car.png')
obstancleImg = pygame.image.load('obstancle.png')

# Rysowanie pojazdu
if self._agent_location == 0:
    canvas.blit(carImg, (15, 0))
else:
    canvas.blit(carImg, (self.pix_square_size + 15, 0))

# Rysowanie przeszkód
i = 0
for x in self._obstacle_location[0:10]:
    if not np.array_equal(x, [0, 0]):
        if(x[0] == 0):
            canvas.blit(obstancleImg, (self.pix_square_size, i * self.pix_square_size))
        else:
            canvas.blit(obstancleImg, (0, i * self.pix_square_size))
    i = i + 1
```

5. Testowanie środowiska

Środowisko musimy najpierw wykonać za pomocą poniższej komendy:

pip install -e .

Następnie możemy wykonać przykładowe jego użycie za pomocą tego kodu (plik test.py):

```
import gymnasium
import driver_game
import random

size = 10
env = gymnasium.make("DriverGame-v0", size=size, render_mode = "human")
env.reset()

# Rozwiązanie gry - przykład 1
observation, reward, done, _, info = env.step(0)
while not done:
    step = random.randint(0, 1)
    if observation["obstacle"][0][0]:
        step = 1
    if observation["obstacle"][0][1]:
        step = 0

    observation, reward, done, _, info = env.step(step)
    print(observation, reward, done, info)
```

Parametr **size** służy do wytyczenia długości trasy, a **render_mode** służy do włączenia okna z trybem graficznym gry.