27.04.2020

Projekt “Automatyczny kelner ” raport nr. 2.

Opis zadania: Zadaniem automatycznego kelnera jest przyjmowanie zamówień i dostarczanie posiłków klientom. Agent rozpoznaje przygotowany w kuchni posiłek, a następnie na podstawie historii zamówień wybiera stolik, do którego należy go dostarczyć.

1. Dotychczas wykonane zadania:
    - wykorzystanie biblioteki PyGame do tworzenia graficznego interfejsu
    - tworzenie okna o rozmiarze podanym w pixelach
    - rysowanie kraty
    - menadżer rysowania obiektów ze zbiorem tych obiektów
    - stoliki z 4 stanami zamówień
    - kelner poruszający się po planszy i zbierający zamówienia ze stolików
    - wykrywanie kolizji obiektów
    - cache’owanie sprite’ów
    - wprowadzenie algorytmu A*
    - wprowadzenie algorytmu BFS

2. Technologie:
    - Python 3.7
    - realizacja w IDE Pycharm Professional
    - Git
    - biblioteka PyGame

3. Do zrobienia:
    ○ implementa
    cja algorytmu do rozpoznawania obrazu z wykorzystaniem sieci neuronowych np. convolutional neural network
    ○ Przebudowa sposobu reprezentowania wiedzy, dostępu do wiedzy i aktualizowania jej
    ○ Obrazy imitujące zamówienia(dania)
    ○ Refactoring kodu aby spełniał główne standardy PEP 8

4. Obiekty:
    - Agent – obiekt posiadający zdolność poruszania się po mapie za pomocą metod moveUp, moveDown, moveLeft, moveRight. Posiada możliwość zebrania zamówień – collectOrders

    - Table – obiekt reprezentujący stół przy którym będą składane zamówienia i do którego podchodzić będzie kelner.

    - GridBoard – obiekt reprezentujący kratę po której przemieszcza się agent. Konstruktor przyjmuje argumenty do rozdzielczości (width, height) i cellSize – dzięki

czemu ruchy są obliczane w kratkach a nie w pixelach - Każdy powyższy obiekt dziedziczy z klasy Drawable szereg metod jak draw, getX, getY, setX, sety

5. Planowanie ruchu:

Ruchy kelnera generowane są przy pomocy algorytmu A*. Implementacja algorytmu umożliwia pracę w dwóch trybach. Tryb pierwszy to tylko ruchy do przodu, do tyłu, na prawo i na lewo. Tryb drugi bardziej zaawansowany, pozwala dodatkowo na wykonywanie ruchów na skos, czyli lewo-przód, prawo-przód, lewo-tył, prawo-tył. Nasza modyfikacja algorytmu A* uwzględnia również liczbę obrotów, które trzeba wykonać, by zmienić kierunek ruchu, odpowiednio dla trybu pierwszego jak i drugiego.

Heurystyka wykorzystywana przez algorytm jest dopuszczalna. Opiera się o funkcję liczącą odległość z bieżącego położenia do celu, po linii prostej. Natomiast odległość od początku do bieżącego położenia to suma liczby wykonanych kroków i wykonanych obrotów. Poniżej znajduje się tablica możliwych zwrotów kelnera (stan (0, 0) nie jest osiągalny). Na przykład zmiana zwrotu z pozycji (-1, 1) do (1, 0) wydłuża drogę o 3 jednostki, gdyż potrzeba wykonać 3 obroty, by zmienić zwrot ruchu.

(-1,-1) lewo góra	(0,-1) góra	(1,-1) prawo góra
(-1, 0) lewo	(0, 0) pusty	(1, 0) prawo
(-1, 1) lewo dół	(0, 1) dół	(1, 1) prawo dół

Przez podejście kelnera rozumiemy stan, w którym kelner jest w najbliższym sąsiedztwie stolika. Najbliższe sąsiedztwo stolika to kratki nad, pod, z lewej i prawej strony stolika. Kratki po przekątnych nie są najbliższym sąsiedztwem.

Za każdym razem algorytm A* wylicza ścieżki do sąsiedztwa wszystkich stolików (które tego wymagają) i wybiera najkrótszą z nich. Kelner najpierw zmienia zwrot (o ile to konieczne), a następnie realizuje tylko pierwszy krok ze znalezionej ścieżki. Po tym algorytm ponownie wyszukuje najbliższy stolik. Dzięki takiemu podejściu, kelner zawsze kieruje się w stronę najbliższego stolika, który napotyka na swojej drodze, co związane jest z dynamicznymi zmianami stanów stolików. Po wykonaniu ostatniego kroku kelner może znaleźć się w sąsiedztwie więcej niż jednego stolika wymagającego interakcji. W takiej sytuacji inteligentny kelner wybiera ten stolik, który wymaga najmniejszej liczby obrotów, aż do momentu obsłużenia wszystkich. Sposób w jaki została wykonana implementacja umożliwia jednoczesne wprowadzenie wielu zsynchronizowanych kelnerów, co można zobaczyć w bieżącej wersji na Git.

6. Dodatkowe funkcje:

○ Stolik posiada 4 stany związane z zamówieniem:

- NotReady (książka) zamówienie nie jest przygotowane, klienci zastanawiają się nad wyborem potraw z menu, podejście kelnera nie powoduje żadnej akcji

- Ready (check mark) klienci są gotowi do złożenia zamówienia i oczekują na przybycie kelnera, podejście kelnera powoduje przekazanie zamówienia i przejście do następnego stanu

- Waiting (talerze) klienci oczekują realizacji złożonego wcześniej zamówienia

- Served klienci otrzymali zamówione posiłki

7. Uwagi: Brak uwag

Obecny program: