PROJE 3: Q-Learning Algoritması ile Yol Planlaması

Erdem Nayın 180202050 – Melih Yeşilyurt 180202060

Bilgisayar Mühendisliği Bölümü

Kocaeli Üniversitesi

erdem_nayin@hotmail.com – melih58yesilyurt@outlook.com

Proje, masaüstü uygulaması. Bu masaüstü uygulaması Q-Learning pekiştirmeli öğrenme algoritmasını kullanarak kısıtları daha önceden belirlenen labirent içerisinde, kullanıcı tarafından belirlenen başlangıç ve bitiş noktaları arasındaki optimum rota bulma işlemini gerçekleştirir.

1.Yazılım Dili ve Geliştirme Ortamı

Projenin kodları PyCharm kullanılarak Python dilinde yazılmış ve geliştirilmiştir.

2. Geliştirme Aşamaları

a. Uygulama Ara Yüzünün Tasarlanması

Bu projedeki aşamaları gerçekleştirmek adına ilk olarak uygulamamızın kullanıcı ile etkileşime gireceği kısım olan kullanıcı ara yüzü tasarlandı. Ara yüzün ilk kısmı, kullanıcının labirent içerisinde başlangıç ve bitiş noktalarını koordinatlar biçiminde girebileceği bir ekran olarak tasarlandı. Ara yüzün ikinci kısmı kullanıcının algoritmanın çalışırken ajanın yaptığı hamleleri labirent üzerinde anlık olarak gözlemleyebileceği şekilde tasarlandı. Mavi kutu başlangıç noktası, yeşil kutu varış noktası, kırmızı kutular engeller, beyaz kutular ise engel olmayan boş geçişler, program sonundaki mor kutular ise başlangıç ve bitiş noktası arasındaki en optimum yol olarak tasarlandı.

b. Q-Learning Algoritmasının Tasarlanması

Bu aşamada verilen iki koordinat noktası arasında en optimum yolu bulma işleminde kullanılan Q-Learning pekiştirmeli öğrenme algoritmasının yazımı gerçekleştirildi. Algoritmanın kıstasları daha önceden belirlenen şekilde indirim faktörü gamma 0.9, engellere çarparsa -5 ödül puanı, bitiş noktasına ulaşırsa +5 ödül puanı, diğer temiz geçişler ise +3 ödül puanı olacak şekilde tasarlandı. Q-Learninq algoritmasını geliştirirken başka insanların hazırladığı projelerden faydalandık.

c. Ajan Hareketlerinin Grafiğe Dökülmesi

Bu aşamada ajanın matris üzerinde yaptığı hareketlerin sonuçları bağlamında episode via cost ve episode via step grafikleri matplotlib pylab kütüphanesi kullanılarak tasarlandı. Grafik çıktıları uygulama bitiminde uygulamanın bulunduğu klasöre ilgili grafik adında resim dosyalarına çıkarılarak elde edildi.

d. Matrisin Metin Dosyasına Yazılması

Bu aşamada programı her başlatışımızda rastgele olarak oluşturulan 50x50 boyutlarındaki matrisimizin üzerindeki her bir koordinat ve bulunulan koordinatta engel olup olmadığının bilgisi, uygulamanın bulunduğu klasöre metin dosyası şeklinde yazdırıldı.

e. Başlangıç ve Bitiş Noktası Arasındaki Yolun Çizdirilmesi

Bu aşamada Q-Learning algoritmasının sonlanıp optimal rotayı bulmasının ardından algoritmanın geçtiği en son rotayı kaydedip ara yüzde başlangıç ve bitiş noktaları arasını dolduran mor kutucuklar şeklinde tasarlandı..

3. Uygulamanın Çalışması

Uygulama açıldığında kullanıcının martis üzerinde başlangıç ve bitiş koordinatlarını girebileceği iki ayrı TextBox görünür. TextBoxlara girilen değerler sırasıyla x ve y koordinatları ve bunları birbirinden ayıran virgül şeklinde girilmelidir. İstenilen değerler girildikten sonra Onayla buttonuna basılarak ilgili matrisin içerisine bu koordinatlar gönderilir.

Daha sonra kullanıcıdan gönderilen noktalar haricinde yer alan ve toplam kare sayısının yüzde otuzunu kapsayacak şekilde engeller ve engel

olmayan kutular oluşturulur.

Tüm bu noktalar 50x50 boyutlarındaki bir matrisle ve farklı renklerle kullanıcıya gösterilir.

Daha sonra Q-Learning algoritması çalışır ve ajanın yaptığı her hareket ara yüz üzerinden görüntülenebilir. Ajanın hareketleri aynı zamanda konsoldan görüntülenebilen çıktılar şeklinde de görüntülenebilir.

Bir süre algoritma çalışır ve kullanıcının girdiği başlangıç ve bitiş noktaları arasındaki optimal rotayı mor renkli olacak şekilde ara yüzde gösterir ve ekran, kullanıcı uygulamayı kapatana kadar bu şekilde bekler.

3. Kullanılan Classlar

A. RunRL

Bu class uygulamanın çalışmasını sağlar. Ayrıca grafiklerin çizdirilmesini bu class gerçekleştirmektedir. Kontrol paneli arayüzü burada oluşturulur. Projeyi çalıştırmak için burası başlatıldığından dolayı diğer classları burada çağırır.

B. Functions

Bu class haritada oluşturulan engelleri random bir şekilde dağıtmayı ve oluşturmayı sağlar.

C. GridWorld

Bu class Q-Learning algoritmasının çalışması için gerekli olan fonksiyonları içinde bulundurur. Ayrıca oluşan labirentin arayüzde gösterilmesini sağlar. Tespit edilen en kısa yol burada oluşturulmaktadır.

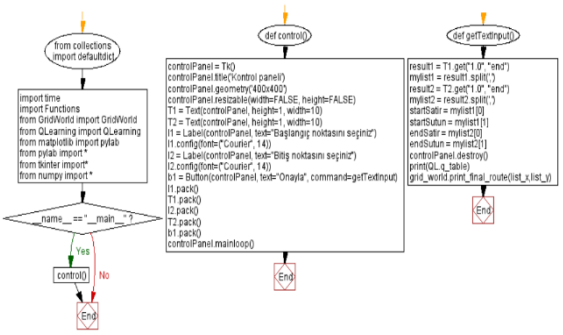
D. Graph

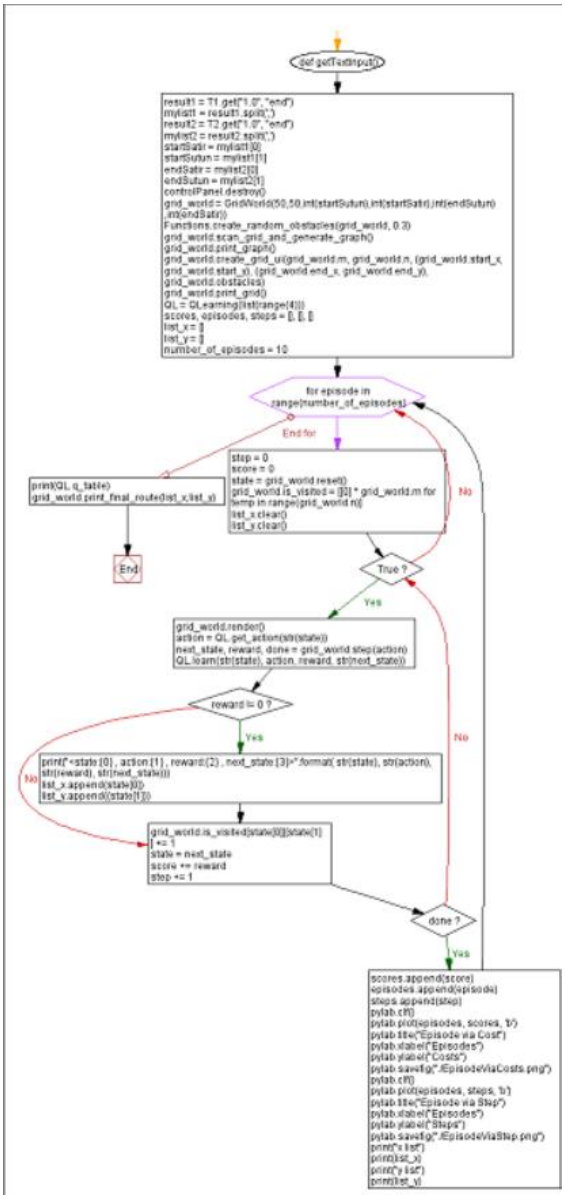
Bu class bütün haritanın graf şeklinde terminal üstünden gösterilmesini sağlar.

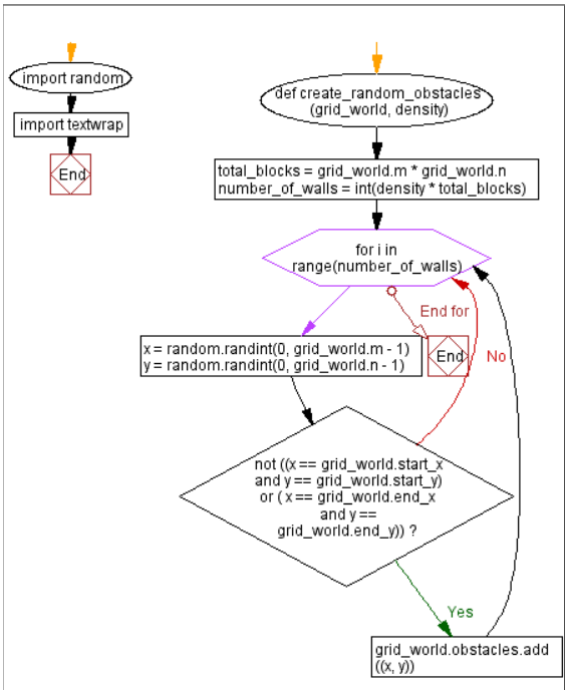
E. QLearning

Bu class Q-Learning pekiştirmeli öğrenme algoritmasının çalışmasını sağlar. Epsilon değerinin azalması Q-Learning algoritmasının formülünün uygulanması gibi kısımlar bu classta çalışır.

4.Akış Diyagramı



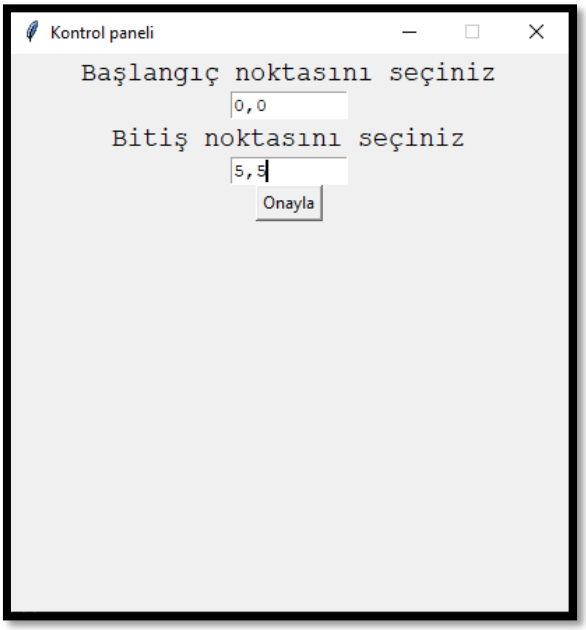


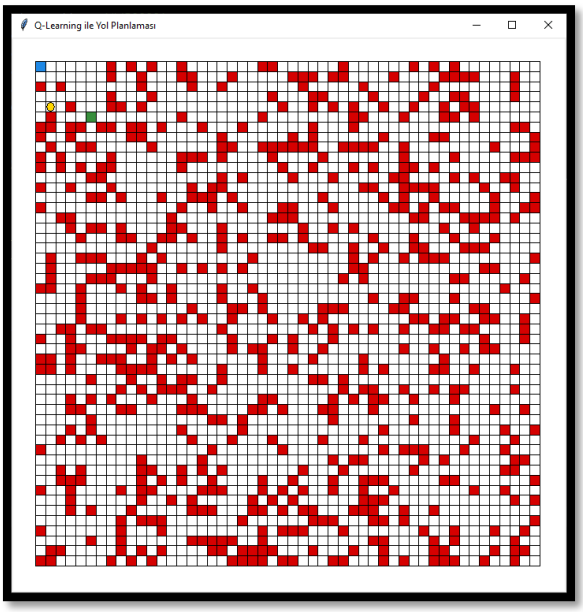


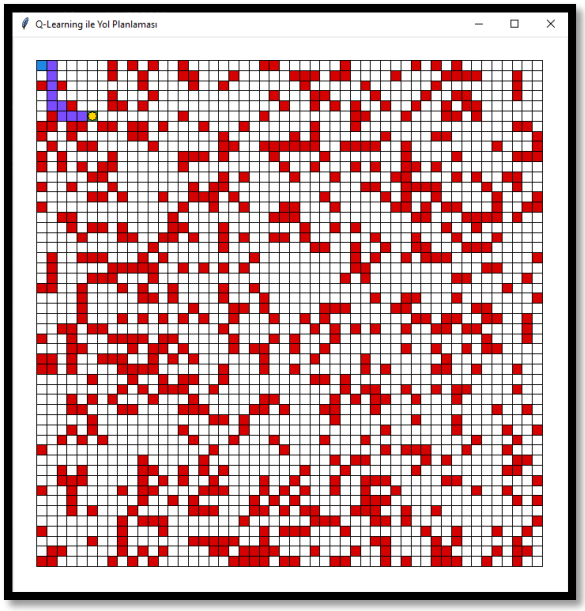
5. Karşılaşılan Problemler

1. Daha önce yapay öğrenme yöntemleriyle hiç çalışmadığımız için Q-Learning algoritmasını oluşturmak biraz zor oldu. Araştırmalarımız sonucu ulaştığımız algoritmalardan bir tanesini seçerek kendimize göre düzenledik ve kullandık.
2. Ajanın yaptığı hareketlerin sonuçlarını grafik üzerinde göstermekte sıkıntı yaşadık. Daha sonra matplotlib pylab üzerinde yapılan grafik çizdirme örneklerinden yararlanarak kendi grafiklerimizi çizdirmeyi başardık
3. Kullanıcının matrisin gidişatını görüntüleyeceği ara yüzü tasarlamada sorun yaşadık. OpenAI hazır environmentlarından mı yararlansak kendimiz mi çizdirsek diye düşündük. Hazır environmentları implement ederken zorlukla yaşadık ve pythonda bulunan tkinter kütüphanesini kullanarak ara yüzü tasarladık.

6.Deneysel Sonuçlar







7.Kaynakça

- https://www.geeksforgeeks.org/q-learning-in-python/
- https://medium.com/@sddkal/python-ve-makine-%C3%B6%C4%9Frenmesi-q-learning-temelleri-181d29326782
- https://github.com/shiluyuan/Reinforcement-Learning-in-Path-Finding
- https://towardsdatascience.com/finding-shortest-path-using-q-learning-algorithm-1c1f39e89505
- http://technobium.com/reinforcement-learning-q-learning-java/
- https://kunuk.wordpress.com/2010/09/24/q-learning/
- http://burlap.cs.brown.edu/tutorials/cpl/p3.html
- https://blog.floydhub.com/an-introduction-to-q-learning-reinforcement-learning/
- https://amunategui.github.io/reinforcement-learning/index.html
- https://www.youtube.com/watch?v=YXPyB4XeYLA
- https://www.youtube.com/watch?v=pc7XhHxSgrM
- https://www.youtube.com/watch?v=aOPB8Mseuzs
- https://www.youtube.com/watch?v=nsLTQj-l_18
- https://www.youtube.com/watch?v=EnCoYY087Fc
- https://medium.com/deep-learning-turkiye/q-learninge-giriş-6742b3c5ed2b
- https://github.com/MattChanTK/gym-maze
- https://github.com/openai/gym/blob/master/docs/environments.md
- https://www.youtube.com/watch?v=SIqXxtKtbtY
- https://www.youtube.com/watch?v=psDlXfbe6ok
- https://www.youtube.com/watch?v=UCbnYXIRrfA
- https://www.youtube.com/watch?v=LUEEmovuuX4

- https://github.com/gacutcut/Reinforcement_learning_path_finding
- https://github.com/openai/gym/wiki/Table-of-environments
- https://www.samyzaf.com/ML/rl/qmaze.html
- https://github.com/erikdelange/Reinforcement-Learning-Maze
- https://www.youtube.com/watch?v=qe0AUmpeyTs
- https://www.youtube.com/watch?v=uTPzj6eK43A
- https://github.com/Hsn37/MazeQLearning
- https://firsttimeprogrammer.blogspot.com/2016/09/getting-ai-smarter-with-q-learning.html
- https://towardsdatascience.com/a-beginners-guide-to-q-learning-c3e2a30a653c
- https://github.com/ChathurangiShyalika/Agent_Path_Following
- https://becominghuman.ai/q-learning-a-maneuver-of-mazes-885137e957e4
- https://github.com/sichkar-valentyn/Reinforcement_Learning_in_Python
- https://github.com/siddharth691/Path-Planning-using-Markov-Decision-Process
- https://github.com/mihai-t/maze-qlearning
- https://www.geeksforgeeks.org/graph-plotting-in-python-set-1/
- https://stackoverflow.com/questions/741877/how-do-i-tell-matplotlib-that-i-am-done-with-a-plot
- https://github.com/guroosh/CS7IS2-AI-project