Fırat Üniversitesi Müh. Bil. Dergisi
34(2), 699-707, 2022

Araştırma Makalesi
https://doi.org/10.35234/fumbd.1122947

Derin Takviyeli Öğrenme Tabanlı Bilinmeyen Ortamda Çoklu Robot Navigasyonu

Niyazi Furkan BAR[1*], Mehmet KARAKÖSE[2]
[1,2] Bilgisayar Mühendisliği, Mühendislik Fakültesi, Fırat Üniversitesi, Elazığ, Türkiye
[*1] nfbar@firat.edu.tr, [2] mkarakose@firat.edu.tr

(Geliş/Received: 30/05/2022; Kabul/Accepted: 03/09/2022)

Öz: Günümüzde mobil robotların navigasyon problemini derin takviyeli öğrenme (DRL) ile çözmeye çalışmak ilgi çekici konulardan birisi haline gelmiştir. Tekli mobil robotlarda DRL yüksek başarı oranlarına ulaşmıştır. Çoklu robot sistemlerinde ise, problemin karmaşıklığı üstel bir şekilde arttığı için maliyeti yüksek ve daha zorlu iş haline gelmektedir. Bu çalışmada ise DRL ile çoklu robot navigasyonu problemine çözüm getirilmeye çalışılmıştır. Önerilen yaklaşımdaki sistemde eşzamanlı bir ortam, bu ortamda birden fazla robot, hedef, engel bulunmaktadır. Ortamda robotlar sırasıyla eylem seçerek, hareket ederler. Aynı zamanda robotlar kendilerinden başka robotlar için dinamik bir engel işlevi görmektedir. Robotlar kendi hedeflerine en kısa yoldan herhangi bir çarpışma yaşamadan ulaşmaya çalışırlar. Aynı zamanda robotlar bir başka robotla çarpışmayacak şekilde veya bir başka robotun rotasını uzatmayacak şekilde yol planlaması yapmaya çalışırlar. Bunları sağlayabilmek için çok ajanlı deep q-network (DQN) algoritması, hedefe yönelik bir durum verisi, güçlendirilmiş adaptif ödül mekanizması kullanılmıştır. Önerilen yaklaşım doğrultusunda oluşturulan sistem tek bir robotun navigasyon başarısı, çoklu robot sisteminin navigasyon başarısı, birim-kare başına düşen robot sayısına göre başarı oranı olarak değerlendirilmiştir. Bu değerlendirmeler önerilen yaklaşımın performansını doğrulamıştır.

Anahtar kelimeler: bilinmeyen ortamda navigasyon, çoklu robot navigasyonu, çok ajanlı derin takviyeli öğrenme.

Multi-Robot Navigation in Unknown Environment Based on Deep Reinforcement Learning

Abstract: Nowadays, trying to solve the navigation problem of mobile robots with deep reinforcement learning (DRL) has become one of the interesting topics. DRL has achieved high success rates in single mobile robots. In multi-robot systems, the complexity of the problem increases exponentially, making it a costly and more demanding task. In this study, it has been tried to solve the multi-robot navigation problem with DRL. In the system in the proposed approach, there is a synchronous environment and more than one robot, target and obstacle in this environment. The robots in the environment move by selecting an action, respectively. At the same time, the robots as a dynamic obstacle for other robots. The robots try to reach their targets in the shortest path without any collision. At the same time, the robots try to plan paths so that they do not collide with another robot or extend the path of another robot. In order to provide these, multi-agent deep q-network (DQN) algorithms, target-oriented state data, and reinforced adaptive reward mechanism were used. The system in the proposed approach was evaluated as the navigation success of a single robot, the navigation success of the multi-robot system, and the success rate according to the number of robots per unit square. These evaluations confirmed the performance of the proposed approach.

Key words: navigation in unknown environment, navigation for multi-robots, multi-agent deep reinforcement learning.

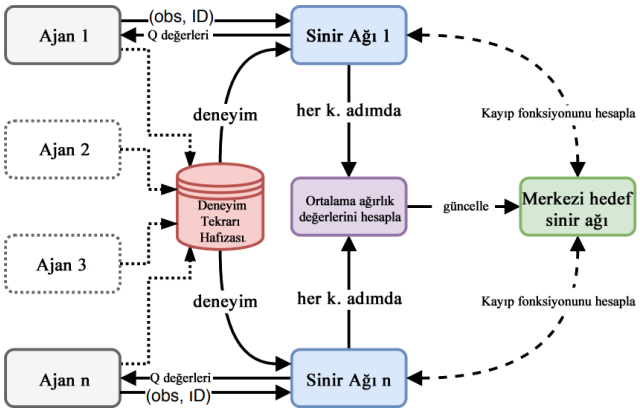
1. Giriş

Günümüzde çoklu robot sistemlerinde navigasyon için geleneksel yöntemler yetersiz kalmaya başlamıştır. Bu yüzden alternatif olarak yeni yöntemler geliştirilmeye çalışılmaktadır. Çoklu robot sistemleri savunma sanayiden depo otomasyonlarına kadar geniş bir kullanım alanının olması bu konuyu yapay zeka ve robotik alanlardaki ilgi çekici konulardan birisi haline getirmiştir. Navigasyon problemi, robotların hedeflerine herhangi bir engelle çarpışmadan en kısa yoldan ulaşmasıyla ilgilenirken, çoklu robotlardaki navigasyon problemi buna ek olarak robotların birbirleriyle işbirliği içerisinde ulaşmasıyla ilgilenir [1]. Bu problem tekli robot sistemlerinde, derin takviyeli öğrenmeyle çözülme konusunda yüksek başarı oranlarına ulaşmıştır [2-6]. Hatta bu çalışmalar gerçek hayatta uygulanmaya başlanmıştır [7]. Yeni eğilim ise bu problemin çoklu robot sistemleri üzerinde çözülmeye çalışılmasıdır. Çoklu robot sistemlerinde, problemin karmaşıklığının üstel bir biçimde artması, maliyet ve kaynak ihtiyacı artışını beraberinde getirmektedir. Çoklu robot sistemlerinde maliyeti azaltmak için başarılı bir öğrenme sürecini daha hızlı gerçekleştirecek ve kaynak ihtiyacını minimize edecek yaklaşımlara ihtiyaç vardır.

* Sorumlu yazar: nfbar@firat.edu.tr. Yazarların ORCID Numarası: [1] 0000-0002-3393-004X, [2] 0000-0002-3276-3788

Bu çalışmada derin takviyeli öğrenme ile bilinmeyen bir ortamda birden fazla robotun navigasyon problemine çözüm getirilmeye çalışıldı. Bu probleme çözüm getirilmeye çalışılırken sadece navigasyon problemi değil, robotların işbirliği sağlayarak rota oluşturması gibi çoklu robot sistemleri için önemli noktalar da göz önüne alınmıştır. Bu çalışmayı gerçekleştirmek için değer tabanlı bir derin takviyeli öğrenme algoritması olan DQN algoritması kullanıldı. İlk defa 2015 yılında V. Mnih ve arkadaşları [8] tarafından sunulan DQN algoritması, yapılan değişiklikler ile çok ajanlı hale getirildi. Birden fazla robotun, hedefin ve engellerin bulunacağı 2-boyutlu senkron çalışan bir ortam tasarlandı. Ortam görüntüsü ajana durum verisi olarak verilerek, evrişimsel sinir ağı (CNN) ile öğrenme sağlandı. Bu sayede robot herhangi bir pozisyon bilgisine sahip olmadan, bilinmeyen bir ortamda navigasyon gerçekleştirdi. Bunlara ek olarak önceki çalışmalarımızdan [2] ve literatürdeki diğer çalışmalardan gözlemlediğimiz üzere durum verisinin çeşitlendirilerek ödül mekanizmasında kullanılması ve adaptif ödül mekanizmalarının daha hızlı ve başarılı bir öğrenme gerçekleştirmektedir. Bu bilgiyle ortam görüntüsü 4-katmanlı bir görüntü olarak verilerek, adaptif bir ödül mekanizması kullanılmıştır. Önerilen yaklaşımla çoklu robot sistemlerinin getirdiği zorluklar olan maliyet ve kaynak ihtiyacı artışı, hız düşüklüğü çözülerek başarılı sonuçlar elde edilmeye çalışılmıştır. Elde edilen sonuçlar grafikler ve tablolar şeklinde verilerek detaylı bir şekilde açıklanarak, bazı noktalarda yorumlara yer verilmiştir.

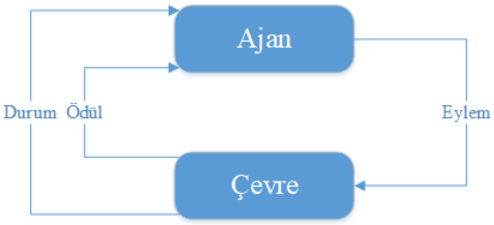
Çoklu robot sistemleri konusunda yapılmış çalışmalardan birisi olan [9]'daki çalışmada DQN algoritması kullanılarak, ortak bir deneyim tekrarı hafızası oluşturulmuş ve her ajan için ayrı bir yapay sinir ağı oluşturulmuştur. Her ajan için ayrı oluşturulan yapay sinir ağlarının belirlenen adım sayısına ulaşınca hedef ağ tarafından ağırlıkları eşitlenmektedir. [9]'daki sistemin blok diyagramı Şekil 1'de verilmiştir. [4]'teki çalışmada ajanlar toplu halde bir görseli oluşturmaya çalışırlar. Bu sayede hem navigasyon hem de işbirliği gerçekleştirmiş olurlar. Bu çalışmayla benzer olarak 2-boyutlu bir ortam ve yapay sinir ağlarında CNN, FFNN katmanlar kullanmışlardır. [9]'daki çalışmada elde edilen sonuçlar incelendiğinde tek bir robotun navigasyon başarısı ~95%, çoklu robotların navigasyon başarısı ~90% olmuştur. Bu çalışmada 10.000 eğitim bölümü gerçekleştilmiştir ve DQN hiperparametreleri deneyim tekrarı hafızası 1.000.000, mini-batch sayısı ise 128 olarak belirlenmiştir. [9]'daki çalışma, bu çalışmadan tekli robotların navigasyon başarısından daha az, çoklu robotların navigasyon başarısından daha yüksek başarıya ulaşmıştır eğitim sayısı, mini-batch sayısı ve deneyim tekrarı hafızası boyutumuz göz önüne alındığında aradaki farkın kabul edilebilir düzeyde olduğunu göstermektedir.



Şekil 1. [9]'daki çalışmada önerilen yaklaşımın blok diyagramı

2. Derin Takviyeli Öğrenme

Takviyeli öğrenme, kümülatif bir ödülü en üst düzeye çıkarmak için ajanın ortamda belirlenen eylemler arasından en uygun olanını seçmesini sağlamak için kullanılan bir makine öğrenmesi yöntemidir [10-11]. Bunu yapabilmek için Markov Karar Sürecini kullanır [11]. Markov Karar Süreci temel olarak şu şekilde çalışır: Ajan eylem kümesinden bir eylem seçer ve bunu ortama gönderir. Ortam da seçilen eylem karşılığında geri dönüt olarak yeni durumu ve ödül değerini ajana gönderir. Markov Karar Süreci Şekil 2'de gösterildiği gibidir.



Şekil 2. Markov Karar Süreci

Derin takviyeli öğrenme ise takviyeli öğrenmede kullanılan algoritmaların derin sinir ağlarıyla güçlendirilmiş halidir. Derin takviyeli öğrenme, takviyeli öğrenme ile çözülmesi zor olan problemlerin veya zor optimizasyon problemlerinin çözülmesini derin öğrenmenin prensiplerini kullanarak daha basit hale getirir [10]. Derin takviyeli öğrenmede yapılan eylem sonucu geri dönüt olarak alınan sayısal ödül değeri sinir ağının ağırlıklarını güncellemek için kullanılır. Bu sayede sonraki eylemlerini seçerken en yüksek ödül değerini getirecek eylemi seçer. Sinir ağı güncellenirken Q-fonksiyonu adı verilen bir fonksiyon ile güncellenir [12]. Bu fonksiyon Denklem 1'de verilmiştir. Bu fonksiyonda indirim faktörü adı verilen bir parametre vardır. Bu parametre (0,1] aralığında değer alır [10-11] ve bu parametrenin Q-fonksiyonuna dahil edilmesiyle ajanın gelecekte alabileceği ödül değerinin etkisi arttırılabilir veya azaltılabilir. Bu sayede ajan uzun-kısa vadeli planlar yapabilir veya hiç plan yapmaz. İndirim faktörü parametresinin 0'a yaklaşmasıyla ajanın kısa vadeli planlar, 1'e yaklaşmasıyla ajanın uzun vadeli planlar yapabilmesini sağlar, 0 olmasıyla plan yapmamasını sağlar.

$$Q(s,a) = Q(s,a) + \alpha[R_t + \gamma * max[Q(s_{t+1},a)] - Q(s,a)] \quad (1)$$

Denklem 1'de belirtilen s durumu, a eylemi, α öğrenme oranını, R_t t zamanındaki ödül değerini, γ: indirim faktörünü, $max[Q(s_{t+1},a)]$ t+1 zamanındaki durum ve aynı eylemde alınabilecek en yüksek Q değerini temsil eder.

2.1. Keşif-Sömürü

Derin takviyeli öğrenmede ortaya çıkan büyük problemlerden birisidir. Bu problem özünde bir ikilemdir. Keşif, ajanın yeni eylemler deneyerek, yeni durum-eylem çiftleri öğrenmesidir. Sömürü ise ajanın durum için sinir ağından en yüksek ödülü getirecek eylemi seçmesidir. Bu problemin ikilem olmasındaki ana nokta şudur; Ajan daha yüksek ödül elde etmek için daha önce deneyerek öğrendiği en yüksek ödüllü eylemi seçmelidir yani "sömürü" işlemi yapmalıdır fakat yüksek ödül değeri getirecek eylemleri öğrenebilmesi için de bu eylemleri denemesi gerekmektedir yani "keşif" işlemi yapmalıdır. Bu ikilemi çözmek için geliştirilmiş Epsilon-Greedy, Upper Confidence Bounds, Boltzmann Exploration gibi birçok yöntem vardır. Bu yöntemlerden en çok kullanılanı ise Epsilon-Greedy yöntemidir [13].

2.1.1. Epsilon-Greedy yöntemi

Bu yöntemde bir keşif sömürü oranı belirlenir ve bu oran doğrultusunda toplam eğitim sayısında kaç bölüm keşif odaklı, kaç bölüm sömürü odaklı bir seçim yapılır. Ayrıca ε ile gösterilen (0,1) aralığında değer alan, her bölümde keşif sömürü oranına göre yeniden hesaplanan bir değişken vardır. Bu değişken eylem seçerken kullanılır. Eylem seçimi sırasında rastgele bir sayı üretilerek, bu değişken ile karşılaştırılır. Bu karşılaştırma sonucunda değişken, rastgele üretilen sayıdan büyükse eylem kümesinden rastgele bir eylem seçilerek keşif odaklı bir seçim yapılmış olur ve böylece ajan yeni bir durum-eylem çifti öğrenir. Karşılaştırma sonucunda değişkenin rastgele üretilen sayıdan büyük olmaması durumunda sinir ağı kullanılarak en yüksek ödül değerini getirecek olan eylem seçilerek sömürü odaklı bir seçim yapılmış olur. Bu yöntemin eylem seçimi Denklem 2'deki gibidir.

$$a_t = \begin{cases} rand\ action\ from\ A(s) & ,eğer\ rastgele\_sayi < \varepsilon \\ argmax_{a \epsilon A(s)} Q(s,a), & aksi\ durumda \end{cases} \quad (2)$$

Denklem 2'de belirtilen ε keşif sömürü oranına göre hesaplanan değişkeni, $A(s)$ eylem kümesini, $Q(s,a)$ öğrenilen durum-eylem çiftlerinin Q değerleri kümesini temsil eder.

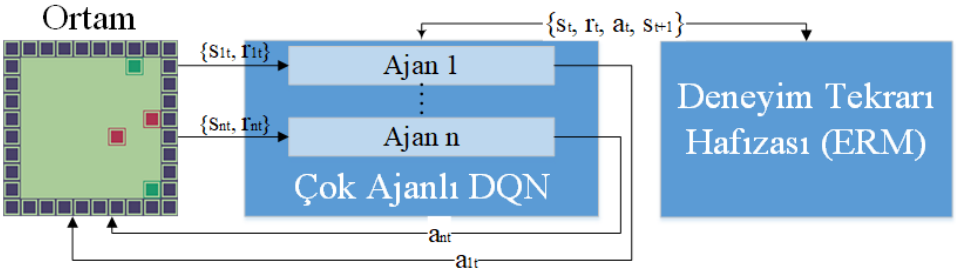
2.2. DQN algoritması

Takviyeli öğrenmedeki Q-learning algoritmasının derin takviyeli öğrenmedeki karşılığıdır. Bu yüzden Deep Q-learning algoritması olarak da bilinir. Takviyeli öğrenmedeki karşılığı olan Q-learning algoritmasında durum-eylem çiftlerine göre verilecek Q değerleri, önceden hazırlanan bir tabloda tutulur ve ajan bu tabloyu kullanarak eylem seçimi gerçekleştirir. Fakat Q değerlerini barından ve Q-tablo olarak adlandırılan bu tablonun kapasite, maliyet ve her durumu önceden bilerek Q-tablosunu hazırlama gibi sorunları vardır. Bu sorunları ortadan kaldırmak için Q değerlerini tahmin eden, durumları genelleştirerek önceden bilme ve tablo hazırlama zorunluluklarını ortadan kaldıran sinir ağı kullanan DQN algoritması ortaya çıkmıştır [12].

DQN algoritmasında Q değerlerini tahmin etmek için bir sinir ağı ile yakınsamaya çalışılan doğrusal olmayan Q-fonksiyonu, sadece son deneyimi öğrenmesiyle aradaki korelasyondan büyük derecede etkilenme riski taşır [14]. Bu riskin bir sonucu olarak da kararlılık sorunu ortaya çıkar [15]. Bu sorunu çözmek için deneyim tekrarı hafızası (ERM) adı verilen bir yöntem geliştirilmiştir. Bu yöntemde ajanın deneyimleri (durum, eylem, ödül, sonraki durum) depolanır. Her bir eğitim bölümünde bu hafızadan rastgele deneyimler seçilerek, mini-batchler şeklinde öğrenme işlemi gerçekleştirilir. Bu sayede sadece o anki deneyim ile değil geçmişteki deneyimler de kullanılarak daha kararlı bir şekilde sinir ağının ağırlıkları güncellenir.

3. Önerilen Yaklaşım

Derin takviyeli öğrenmeyle bilinmeyen bir ortamda navigasyon işlemi yapmanın birçok zorluğu olduğu gibi bu işi çoklu robot sistemlerinde yapmanın getirdiği başka zorluklar da vardır [16-17]. Bunlar yanlış hedefe ulaşmama, diğer robotlarla çarpışmama, diğer robotlarla işbirliği içerisinde en uygun yol planlamasının yapılması gibi zorluklardır. Diğer robotlarla çarpışmama zorluğu, sabit bir engelle çarpışmamadan daha zorludur çünkü diğer robotlar da hedefine ulaşmak için hareket etmektedir. Bu doğrultuda çözmek istenilen navigasyon problemi, çoklu robot sistemleri için ele alınınca problemin karmaşıklık derecesi üstel olarak artmaktadır. Bu zorlu problemi çözebilmek için DQN algoritması çok ajanlı hale getirilerek, durum verisi, ödül mekanizması ve sinir ağı güçlendirildi. Önerilen yaklaşımda aynı ortam içerisinde birden fazla ajan bulunur. Bu ajanlar eylemleri seçmek için ortak bir sinir ağı, deneyimlerini depolamak için ortak bir deneyim tekrarı hafızası kullanırlar. Önerilen yaklaşımda klasik ödül mekanizması yerine performansı doğrulanmış çok durumlu adaptif ödül mekanizması kullanılır. Ödül mekanizması en az adımda, ajanın kendisini tekrarlamayacağı şekilde tasarlanmıştır. Durum verileri hem ödül sistemindeki adaptifliği sağlamak hem de ajanların kendilerini ve hedeflerini görebilmeleri için özelleştirilmiştir. Bütün bunların sayesinde çoklu robot sistemlerinde karmaşıklığı üstel artan navigasyon probleminin, daha hızlı ve başarılı bir öğrenme gerçekleştirilerek çözülmesi hedeflendi. Önerilen yaklaşım Şekil 3'teki gibidir.

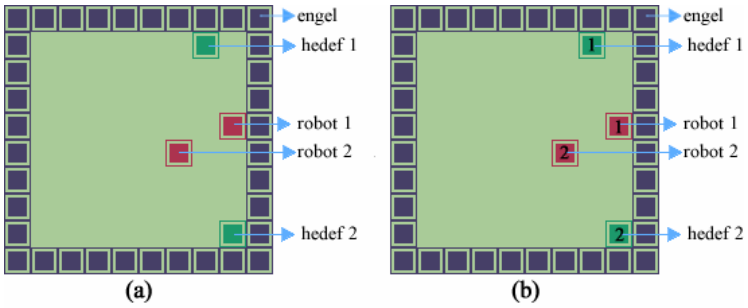


Şekil 3. Önerilen yaklaşımın blok diyagramı, burada s_{1t} birinci ajanın t zamanındaki durum verisini, r_{1t} birinci ajanın t zamanındaki ödül değerini, a_{1t} birinci ajanın t zamanındaki seçtiği eylemi, s_{nt} n-inci ajanın t zamanındaki durum verisini, r_{nt} n-inci ajanın t zamanındaki ödül değerini, a_{nt} n-inci ajanın t zamanındaki seçtiği eylemi, s_t herhangi bir ajanın t zamanındaki durum verisini, r_t herhangi bir ajanın t zamanındaki ödül değerini, a_t herhangi bir ajanın t zamanındaki seçtiği eylemi, s_{t+1} herhangi bir ajanın t+1 zamanındaki durum verisini temsil eder.

3.1. Simülasyon

3.1.1. Ortam

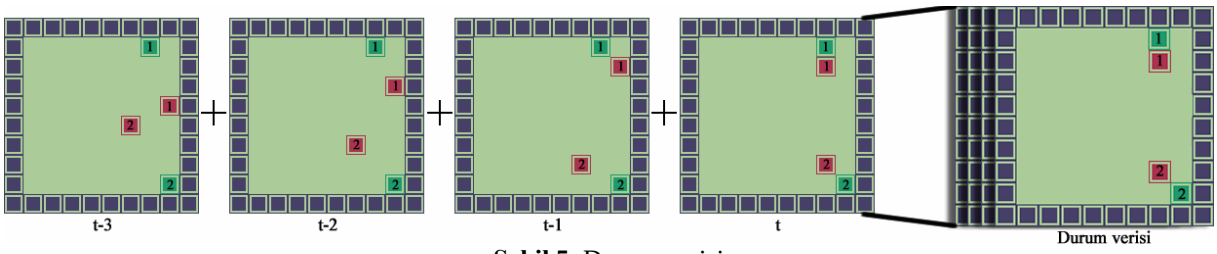
Önerilen yaklaşım için 10×10 boyutunda 2-boyutlu bir ortam oluşturulmuştur. Ortamda 2 robot ve 2 hedef bulunmaktadır. Her bölüm başlangıcında robotlar ve hedefler rastgele boş pozisyonlarda oluşturulur ve robotlara hedefleri tanımlanır. Bir robot için bölüm herhangi bir engelle veya bir diğer robot ile çarpışması durumunda sonlandırılır. Durum verisi başlığı altında bahsedilen ortamın robot için manipüle edilmiş haliyle beraber ortam Şekil 4'te verildiği gibidir.



Şekil 4. (a) simülasyon ortamı, (b) robotun gördüğü ortam

3.1.2. Durum verisi

Problemin çözümünde DQN ajanlarına durum verisi olarak ortamın o anki görüntüsü ve 3 adım önceki görüntüleri birleştirilerek 4-katmanlı bir görüntü olarak verildi. Bu görüntü Şekil 5'te verilmiştir. Ortamın önceki görüntüleri de durum verisine eklenerek, ajanın aynı eylemleri tekrarladığında ceza verilebilmesi sağlandı. Bu sayede ajan aynı eylemleri tekrarlayarak, aynı durumları elde edip bir çıkmaza girmesi engellendi ve daha verimli bir eğitim süreci yapılması hedeflendi. Ajanlara ortam görüntüsü verildiğinden ve ortamın çok ajanlı bir ortam olmasından kaynaklı ajanlara verilen durum verisinde robot ve hedeflere ID verilerek ayırt edilebilmesi sağlandı. Bu sayede robotun hedefe ulaşmasındaki performansın arttırılması ve robotlar arasında yol planlaması yaparken işbirliği sağlanması hedeflendi.



Şekil 5. Durum verisi

3.1.3. Eylem Kümesi

Robotun navigasyonu için yukarı git, aşağı git, sağa git, sola git olmak üzere dört farklı eylem tanımlanmıştır. Ortam 2-boyutlu tasarlandığı için X ve Y koordinatlarında mevcut koordinatlarla toplama işlemi yapılarak hareket sağlanmıştır. Yukarı için (0,1), aşağı için (0,-1), sağ için (1,0), sol için (-1,0) koordinatlarıyla işlem gerçekleştirilmiştir.

3.1.4 Ödül Mekanizması

Ödül mekanizması tasarlanırken 4 farklı duruma odaklanılmıştır. Bu durumlar, robotun kendi hedefine ulaşması, hedefine ulaşmaya çalışırken en kısa yoldan ulaşması, herhangi bir engel veya bir diğer robotla çarpışması ve tekrarlı eylemler gerçekleştirmesidir. Robotun bir diğer robotun hedefine ulaşması durumu istenilmeyen bir durum olduğu için o durumda çarpışma yaşamış gibi bir ödül değeri verildi. Çoğu derin takviyeli öğrenme çalışmasında kullanılan istenilen gerçekleşince +1, istenmeyen gerçekleşince -1 ödül değeri verilen ödül mekanizması geliştirilip, güçlendirilerek farklı durumlar ve hassasiyete sahip ödül değerleri eklenerek adaptif ödül mekanizması elde edildi. Bu ödül mekanizması Denklem 3'teki gibidir.

$$odul = \begin{cases} \mathbf{+1} & ,hedefe\ ulaşma\ için \\ \mathbf{-1} & ,çarpışma\ için \\ \boldsymbol{(d_{t-1} - d_t)/10} & ,her\ adım\ için \\ \mathbf{-0.5} & ,tekrarlı\ eylem\ için \end{cases} \quad (3)$$

Denklem 3'te belirtilen d_{t-1} t-1 zamanındaki robot ile hedef arasındaki uzaklığı, d_t t zamanındaki robot ile hedef arasındaki uzaklığı temsil eder.

3.2. Çok ajanlı DQN ve mimarisi

Aynı ortamda birden fazla ajan bulunacağı için DQN algoritması geliştirilerek birden fazla ajanın aynı ortamda bulunabilmesi sağlandı. DQN algoritmasındaki bölümler içerisindeki adımların işletildiği döngünün içerisine ajanlar için bir döngü eklendi. Bu döngü sayesinde bölüm içerisindeki her bir adımda ajanlar sırayla eylemlerini seçerler ve bunu ortak deneyim tekrarı hafızasına gönderirler. Ortak deneyim tekrarı hafızası sayesinde ajanlar sadece kendi deneyimlerini değil, diğer ajanların deneyimlerini de öğrenerek genel bir öğrenme gerçekleştirirler. Çok ajanlı hale getirilen DQN algoritmasının sözde kodu Tablo 1'de verilmiştir.

	Algoritma 1: Çok ajanlı DQN
1.	Initialize replay memory D with capacity N
2.	Initialize Q network with random weights θ
3.	Initialize target Q network with weights θ - = θ
4.	repeat (for each episode):
5.	Observe initial state s0
6.	repeat (for each step of episode):
7.	repeat (for each agent in step):
8.	select action a_t by $a_t = \begin{cases} rand\ action\ from\ A(s)\ ,if\ rand < \varepsilon \\ argmax_{a \epsilon A(s)} Q(s,a) \quad ,otherwise \end{cases}$
9.	Execute action a_t
10.	Observe reward $r_t$and new state s_{t+1}
11.	Store transition (s_t, a_t, r_t, s_{t+1}) in D
12.	until agent
13.	Sample mini-batch of transitions from D
14.	Calculate target for each transition: $y_j = \begin{cases} r_j & ,is\ terminal \\ r_j + \gamma max Q(s_{j+1}, a') & ,is\ nonterminal \end{cases}$
15.	Perform a descent step on $\left(y_j - Q(s_j, a_j; \theta)\right)^2$ with respect to the Q network parameters θ
16.	until step
17.	until episode

DQN algoritmasında Q-değerlerini tahmin etmek için kullanılan sinir ağı tasarlanırken giriş katmanı olarak evrişimsel katman kullanıldı. Bunun sebebi evrişimsel katmanların matrislerle, görüntülerle daha başarılı sonuçlar elde etmesidir. Bu sinir ağında 2 tane evrişimsel [18], 3 tane tam bağlantılı[19] katman olmak üzere toplam 5 katman vardır. Bu sinir ağının giriş katmanına 4 katmanlı 2-boyutlu matris verilmiştir, çıkış katmanına eylem kümesi boyutu olan 4 nöron tanımlanmıştır. Bu katmanlarda aktivasyon fonksiyonu olarak ReLU [18] fonksiyonu,

optimizasyon yöntemi olarak RMSprop [19], kayıp fonksiyonu olarak ise MSE [20] kullanılmıştır. Tasarlanan sinir ağının mimarisi Tablo 2'de verilmiştir.

Tablo 2. Sinir ağının mimarisi

	Katman Tipi	**Giriş**	**Çıkış**
Giriş	Conv2D	(4,10,10)	(16,8,8)
Gizli	Conv2D	(16,8,8)	(32,6,6)
Gizli	Dense (FC)	(32,6,6)	(1,1152)
Gizli	Dense (FC)	(1,1152)	(1,256)
Çıkış	Dense (FC)	(1,256)	(1,4)

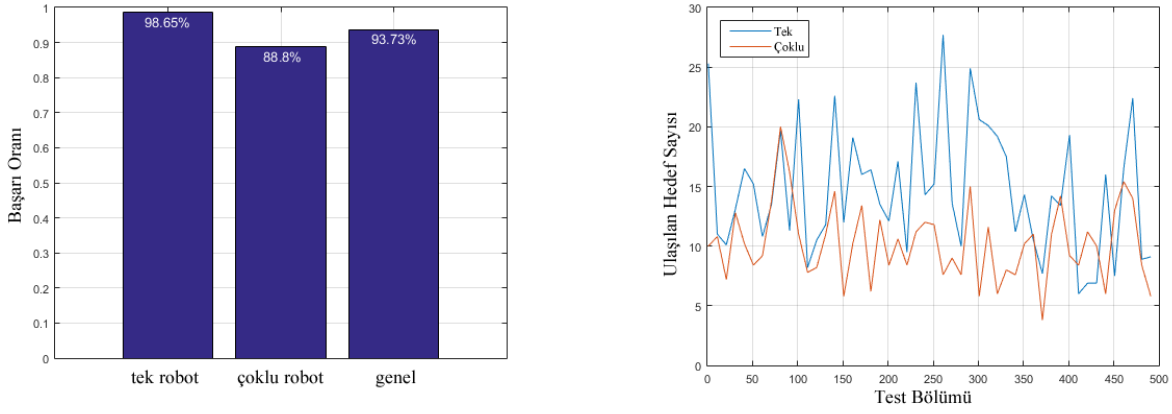
4. Simülasyon Sonuçları

Önerilen yaklaşım başlığı altında detaylı bir şekilde anlatılan sistem, Tablo 2'deki parametrelerle 1000 bölüm eğitilmiştir. Bu eğitim sonucu elde edilen sinir ağı modelleri 500 bölüm test edilmiştir. Bu parametrelerin sistem üzerindeki etkilerine değinecek olursak; belirlenen keşif sömürü oranı, ilk ve son epsilon değerleri doğrultusunda belirlenen eğitim bölüm sayısının yarısına kadar ilk epsilon değerinden, son epsilon değerine azalarak devam edecek ve eğitim bölüm sayısının yarısından sonra sürekli son epsilon değeriyle devam edecektir. İndirim faktörünün 1'e yakın şekilde belirlenmesiyle ajanın uzun vadeli planlar yapabilmesine imkan tanınmıştır. Çünkü navigasyon probleminde robotun hedefe ulaşana kadar birden fazla adım sayısı gereklidir. Bu yüzden uzun vadeli plan yapmalıdır. Maksimum adım sayısı kısıtıyla bölüm içerisindeki eğitim sayısı 64×1000 olarak kısıtlansa da sistemin tıkanmasını engelleyerek, daha verimli bir eğitim gerçekleştirilmiştir. Aynı zamanda bu kısıt ile ortamda tek bir ajan kaldığında sadece onun başarısının artması yerine, işbirliğinin başarısının artması hedeflenmiştir.

Tablo 1. Sistem parametreleri

Parametre Adı	**Değer**	**Parametre Adı**	**Değer**
İlk $\boldsymbol{\varepsilon}$	1.0	Öğrenme oranı	0.001
Son $\boldsymbol{\varepsilon}$	0.1	Keşif-sömürü oranı	50%
mini-batch boyutu	64	ERM boyutu	1000
$\boldsymbol{\gamma}$:indirim faktörü	0.95	Maksimum adım sayısı	1000

Eğitim sonucu elde edilip 500 bölüm test uygulanan model, hedefe ulaşma başarı oranı ve artarda ulaştığı hedef sayısı olarak değerlendirilmiştir. Robotun en az 1 hedefe ulaşması, robot için başarı olarak adlandırılır. Yapılan test için başarı oranları, tek bir robotun başarısı, çoklu robotların başarısı ve genel başarı oranı olarak Şekil 6 (a)'da verilmiştir. Robotların artarda ulaştığı hedef sayısının grafiği ise Şekil 6 (b)'de, tek bir robotun ve çoklu robotların artarda ulaştığı hedef sayısı olarak verilmiştir.



(a) **(b)**

Şekil 6. (a) başarı oranları, (b) ulaşılan hedef sayısı

Önerilen yaklaşım ile eğitilen sinir ağı modeli Şekil 6 (a)'dan görülebileceği üzere ortamda tek başınayken 98.65%'lik bir başarı oranı, ortamda birden fazla robot varken 88.8%'lik bir başarı elde etmiştir. Önerilen yaklaşım hem tek bir robot için hem de çoklu robot sistemleri için yeterli ölçüde başarı oranına ulaşmış olup, yaklaşımın genel başarı oranı 93.73% olmuştur. Önerilen yaklaşım ile eğitilen sinir ağlarının artarda görev tamamlama grafiği Şekil 6 (b)'de verilmiştir. Buradaki grafikten de görülebileceği üzere başarı oranlarıyla paralel olarak, görev tamamlama sayılarında da farklar vardır. Ortamda tek bir robot varken artarda görev tamamlama sayısı ortalama olarak 19.76 olurken, ortamda birden fazla robot olduğunda bu sayı 5.08 olmaktadır. Önerilen yaklaşımın başarılı olduğu görüldükten sonra ortam boyutu ve ortamdaki robot-hedef yoğunluğunun sistem başarısını nasıl etkilediğini gözlemlemek için bir dizi daha test gerçekleştirildi. Bu testlerde ideal robot sayısına göre ideal ortam boyutu ve ideal robot-hedef yoğunluğu ortaya çıkarılmaya çalışılmıştır. Bu test ortamdaki ajan sayısı arttırılarak gerçekleştirilmiştir. Testin sonuçları Tablo 3'te verilmiştir.

Tablo 2. Ortamdaki robot yoğunluğu test sonuçları

Robot Sayısı	1	2	3	4
Boş alan boyutu	79 br^2	77 br^2	75 br^2	73 br^2
Robot başına düşen boş alan boyutu	79 br^2	38.5 br^2	25 br^2	18.25 br^2
Başarı oranı	98.65%	88.8%	67.85%	48.26%
Ortalama gerçekleştirilen görev sayısı	13.73	9.21	4.51	2.84

Not: Toplam alan 10×10 ortamdan dolayı 100 birim-karedir. Robotlar, hedefler, engeller 1 birim-karelik yer kaplamaktadır.

Tablo 3'te görüleceği üzere robot başına düşen boş alan boyutu azaldıkça başarı oranı ve artarda ulaşılan hedef sayısı azalmaktadır. Bu oranlara etki eden bir diğer faktör ise robot sayısının çoğalmasıyla problemin karmaşıklığının artması ve yeni daha önce deneyimlenmeyip, genelleştirilemeyen durum verilerinin oluşmasıdır. Çoklu robot sistemleri kurarken ortamdaki robot yoğunluğunun hesaplanarak bu yaklaşım uygulandığında yüksek başarı elde eden, verimli çoklu robot sistemlerinin kurulabileceği gözlenmiştir.

5. Sonuç

Bu çalışmada derin takviyeli öğrenme tabanlı çoklu robot sistemlerinde navigasyon için bir yaklaşım önerilmiştir. Önerilen yaklaşımda klasik DQN algoritması geliştirilerek, çok ajanlı bir yapı haline getirilmiştir. Daha kısa eğitim süresinde daha iyi başarılar elde etmek için sadece o anki değil önceki anların görüntülerini de içeren ve bu robota göre manipüle edilmiş bir durum verisi, güçlendirilmiş ödül mekanizması kullanılmıştır. Bu durum verisiyle ajana adeta hafıza kazandırılmış olup, sürekli aynı durum ve eylemleri tekrar ettiğinde ceza verilmiş ve böylece daha verimli eğitim süreci gerçekleştirilmiştir. Güçlendirilmiş ödül mekanizmasında, ortamla bağlantılı ödül verilerek robotun her hareket yaptığında hedefe yaklaşmasıyla pozitif, uzaklaşmasıyla negatif ödül değeri verilerek en kısa yoldan hedefine ulaşması sağlanmıştır. Sistemde kullanılan parametrelerle, özellikle de maksimum adım kısıtı parametresiyle eğitim sürecinin verimi arttırıldığı gibi, tek bir robotun başarısının artması yerine robotları arasındaki işbirliğinin artırılması sağlanmıştır.

Kaynaklar

[1] J. Yu and S. M. LaValle, "Optimal Multirobot path planning on graphs: Complete algorithms and effective heuristics," IEEE Transactions on Robotics, vol. 32, no. 5, pp. 1163–1177, 2016.

[2] N. F. Bar, H. Yetis, and M. Karakose, "Deep Reinforcement Learning Approach with adaptive reward system for robot navigation in Dynamic Environments," Interdisciplinary Research in Technology and Management, pp. 349–355, 2021.

[3] M. Pfeiffer et al., "Reinforced Imitation: Sample Efficient Deep Reinforcement Learning for Map-less Navigation by Leveraging Prior Demonstrations." arXiv, 2018. doi: 10.48550/ARXIV.1805.07095.

[4] T. Xuan Tung and T. Dung Ngo, "Socially Aware Robot Navigation Using Deep Reinforcement Learning," 2018 IEEE Canadian Conference on Electrical & Computer Engineering (CCECE), 2018, pp. 1-5, doi: 10.1109/CCECE.2018.8447854.

[5] S. -H. Han, H. -J. Choi, P. Benz and J. Loaiciga, "Sensor-Based Mobile Robot Navigation via Deep Reinforcement Learning," 2018 IEEE International Conference on Big Data and Smart Computing (BigComp), 2018, pp. 147-154, doi: 10.1109/BigComp.2018.00030.

[6] X. Qiu, K. Wan and F. Li, "Autonomous Robot Navigation in Dynamic Environment Using Deep Reinforcement Learning," 2019 IEEE 2nd International Conference on Automation, Electronics and Electrical Engineering (AUTEEE), 2019, pp. 338-342, doi: 10.1109/AUTEEE48671.2019.9033166.

[7] H. Surmann, C. Jestel, R. Marchel, F. Musberg, H. Elhadj, and M. Ardani, "Deep Reinforcement learning for real autonomous mobile robot navigation in indoor environments." arXiv, 2020. doi: 10.48550/ARXIV.2005.13857.

[8] V. Mnih et al., "Human-level control through deep reinforcement learning," Nature, vol. 518, no. 7540. Springer Science and Business Media LLC, pp. 529–533, Feb. 25, 2015. doi: 10.1038/nature14236.

[9] E. A. Oury Diallo and T. Sugawara, "Multi-Agent Pattern Formation: a Distributed Model-Free Deep Reinforcement Learning Approach, " 2020 International Joint Conference on Neural Networks (IJCNN), 2020, pp. 1-8.

[10] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction. MIT Press, 2018.

[11] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau, "An Introduction to Deep Reinforcement Learning," Foundations and Trends® in Machine Learning, vol. 11, no. 3–4. Now Publishers, pp. 219–354, 2018. doi: 10.1561/2200000071.

[12] O. Anschel, N. Baram, and N. Shimkin, "Averaged-DQN: Variance Reduction and Stabilization for Deep Reinforcement Learning." arXiv, 2016. doi: 10.48550/ARXIV.1611.01929.

[13] M. Tokic, "Adaptive ε-Greedy Exploration in Reinforcement Learning Based on Value Differences," KI 2010: Advances in Artificial Intelligence. Springer Berlin Heidelberg, pp. 203–210, 2010. doi: 10.1007/978-3-642-16111-7_23.

[14] L.-J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," Machine Learning, vol. 8, no. 3–4. Springer Science and Business Media LLC, pp. 293–321, May 1992. doi: 10.1007/bf00992699.

[15] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized Experience Replay." arXiv, 2015. doi: 10.48550/ARXIV.1511.05952.

[16] Z. Tan and M. Karakose, "On-policy deep reinforcement learning approach to multi agent problems," Interdisciplinary Research in Technology and Management, pp. 369–376, 2021.

[17] Z. Tan and M. Karaköse, "Proximal Policy Based Deep Reinforcement Learning Approach for Swarm Robots," 2021 Zooming Innovation in Consumer Technologies Conference (ZINC), 2021, pp. 166-170, doi: 10.1109/ZINC52049.2021.9499288.

[18] A. Shrestha and A. Mahmood, "Review of Deep Learning Algorithms and Architectures", IEEE Access, vol. 7, pp. 53040-53065, 2019.

[19] E. Bochinski, T. Senst and T. Sikora, "Hyper-parameter optimization for convolutional neural network committees based on evolutionary algorithms," 2017 IEEE International Conference on Image Processing (ICIP), Beijing, 2017, pp. 3924-3928.

[20] J. Brownlee, "Loss and Loss Functions for Training Deep Learning Neural Networks", Machine Learning Mastery, 2019. [Online]. Available: https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/. [Accessed: 12-May- 2022].