HİYERARŞİK BAŞLANGIÇ POZİSYONLU DERİN Q-AĞI ALGORİTMASI İLE MOBİL ROBOT UYGULAMASI

EMRE ERKAN

KASIM 2022

DİYARBAKIR

DİCLE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HİYERARŞİK BAŞLANGIÇ POZİSYONLU DERİN Q-AĞI ALGORİTMASI İLE MOBİL ROBOT UYGULAMASI

EMRE ERKAN

DİCLE ÜNİVERSİTESİ LİSANSÜSTÜ EĞİTİM-ÖĞRETİM VE SINAV YÖNETMELİĞİNİN BİR PARÇASI OLARAK
ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANA BİLİM DALINDA
DOKTORA TEZİ
OLARAK HAZIRLANMIŞTIR

KASIM 2022

DİYARBAKIR

HİYERARŞİK BAŞLANGIÇ POZİSYONLU DERİN Q-AĞI ALGORİTMASI İLE MOBİL ROBOT UYGULAMASI

Emre ERKAN tarafından Dicle Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliği'nin bir parçası olarak hazırlanan bu çalışma, aşağıda bilgileri yazılı jüri üyeleri tarafından değerlendirilerek **Elektrik Elektronik Mühendisliği Ana Bilim Dalı**'nda **Doktora Tezi** olarak kabul edilmiştir.

Prof. Dr. Neslihan DALKILIÇ
Müdür, **Fen Bilimleri Enstitüsü**

Dr. Öğr. Üyesi Muhammet Ali ARSERİM
Danışman, **Elektrik Elektronik Mühendisliği Bölümü, Dicle Üniversitesi**

Sınav Jürisi:

Doç. Dr. Davut İZCİ (*)
Bilgisayar Mühendisliği Bölümü, Batman Üniversitesi

Dr. Öğr. Üyesi Muhammet Ali ARSERİM (**)
Elektrik Elektronik Mühendisliği Bölümü, Dicle Üniversitesi

Dr. Öğr. Üyesi Abdulnasır YILDIZ
Elektrik Elektronik Mühendisliği Bölümü, Dicle Üniversitesi

Dr. Öğr. Üyesi Ömer TÜRK
Mardin Meslek Yüksekokulu, Mardin Artuklu Üniversitesi

Dr. Öğr. Üyesi Cafer BUDAK
Elektrik Elektronik Mühendisliği Bölümü, Dicle Üniversitesi

ONAY

Savunma Tarihi: 11 / 11 / 2022

(*) Sınav Jürisi kısmının birinci satırına Jüri Başkanının bilgilerini yazınız.
(**) Sınav Jürisi kısmının ikinci satırına Tez Danışmanının bilgilerini yazınız.

Aileme…

Dicle Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırlanan bu tez çalışmasında yer alan tüm bilgilerin akademik kurallara ve etik ilkelere uygun olarak elde edildiğini ve sunulduğunu beyan ederim. Ayrıca, bahse konu bu kural ve ilkelerin gerektirdiği üzere, bu çalışmada özgün olmayan tüm bilimsel içerikleri kurallara uygun biçimde alıntılayıp kaynak gösterdiğimi beyan ederim. Beyanımla çelişen herhangi bir delil bulunduğu takdirde tüm sorumluluğu üstleneceğimi kabul ederim.

Ad, Soyad: Emre ERKAN

İmza: …….

TEŞEKKÜR

Öncelikle tez çalışmamın her aşamasında bilgisi, birikimi ve tecrübesiyle bana rehberlik eden danışmanım Dr. Öğr. Üyesi Muhammet Ali ARSERİM'e çok teşekkür ederim. Çalışma konusunun belirleme aşamasındaki yönlendirmelerinden dolayı Prof. Dr. Ayşegül UÇAR'a teşekkürlerimi sunarım. Tez çalışmam süresince hazırladığım raporları değerlendirerek çalışmamın daha nitelikli hale gelmesini sağlayan tez izleme jüri üyelerim Dr. Öğr. Üyesi Abdulnasır YILDIZ ve Dr. Öğr. Üyesi Ömer TÜRK'e teşekkür ederim. Tez kapsamında hazırlamış olduğum bilimsel yayına vermiş olduğu katkılardan dolayı Doç. Dr. Davut İZCİ'ye teşekkür ederim. Tez çalışmama yapmış olduğu olumlu eleştiri ve yönlendirmelerinden dolayı Dr. Öğr. Üyesi Cafer BUDAK'a teşekkür ederim.

Çalışmalarım süresince her türlü fedakârlığı gösteren eşim Büşra AKÇINAR ERKAN'a, oğullarım Musab Yusuf ve Ahmet Erdem'e teşekkürü bir borç bilirim. Son olarak hayatımın her aşamasında verdikleri sonsuz desteklerden dolayı anne ve babama sonsuz teşekkürlerimi ve minnetlerimi sunarım.

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

SİMGELER VE KISALTMALAR LİSTESİ

Simge	**Açıklama**
V	Değer
V^*	Optimum Değer
Q	Q-Değeri
Q^*	Optimum Q-Değeri
S	Durum
A	Eylem
R	Ödül
π	Politika
T	Geçiş fonksiyonu
γ	İndirim Faktörü
α	Öğrenme Oranı
ε	Epsilon Greedy

Kısaltma	**Açıklama**
A3C	Asenkron Avantaj Aktör-Kritik
ASCII	American Standard Code for Information Interchange
BM	Boltzmann Makinaları
COCO	Common Objects in Context
CSS	Chirp Spread Spectrum
DDPG	Derin Determistik Politika Gradyant
DİA	Derin İnanç Ağları
DOK	Derin Oto Kodlayıcılar
DÖ	Derin Öğrenme
DPÖ	Derin Pekiştirmeli Öğrenme
DQA	Derin Q-Ağı
DYSA	Derin Yapay Sinir Ağları

EEPROM	Electrically Erasable Programmable Read-Only Memory
ELU	Exponential Linear Unit
ESA	Evrişimsel Sinir Ağı
GİB	Grafik İşlemci Birimi
HBP-DQA	Hiyerarşik Başlangıç Pozisyonlu Derin Q-Ağı
I2C	Inter-Integrated Circuit
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IoT	Internet of Things
KHK	Kare Hata Kaybı
LoRa	Long Range
LSTM	Long Short-Term Memory
MKS	Markov Karar Süreci
MRKK	Mobil Robot Kontrol Kartı
NAF	Normalize Avantaj Fonksiyonları ile Q-Öğrenme
PER	Prioritized Experience Replay
PÖ	Pekiştirmeli Öğrenme
PPO	Yaklaşık Politika Optimizasyonu
PWM	Pulse-Width Modulation
RAM	Random-Access Memory
ReLU	Rectified Linear Unit
RF	Radio Frequency
RFKK	RF Kontrol Kartı
SAC	Soft Aktör-Kritik
SPI	Serial Peripheral Interface
TD3	Çift Gecikmeli Derin Determistik Politika Gradyant
TRPO	Güven Bölgesi Politika Optimizasyonu
TSA	Tekrarlayan Sinir Ağları
UART	Universal Asynchronous Receiver-

	Transmitter
UKVH	Uzun-Kısa Vadeli Hafıza Ağları
UKVH	Uzun-Kısa Vadeli Hafıza
USART	Universal Synchronous and Asynchronous Receiver-Transmitter
YSA	Yapay Sinir Ağları

ÖZET

HİYERARŞİK BAŞLANGIÇ POZİSYONLU DERİN Q-AĞI ALGORİTMASI İLE MOBİL ROBOT UYGULAMASI

ERKAN, Emre
Doktora, Elektrik Elektronik Mühendisliği Bölümü
Danışman: Dr. Öğr. Üyesi Muhammet Ali ARSERİM
Kasım 2022, 103 sayfa

Derin öğrenmedeki kayda değer ilerleme pekiştirmeli öğrenmeyi de önemli ölçüde etkilemiştir ve her iki yöntemin birleşimi olan Derin Pekiştirmeli Öğrenme (DPÖ) yöntemini ortaya çıkarmıştır. DPÖ'nün bir veri setine ihtiyaç duymaması ve insan uzmanların performansını aşabilecek potansiyele sahip olması yapay zeka alanında önemli gelişmeler olmasına yol açmaktadır. Ancak bir DPÖ ajanı eğitilirken ortam ile çok sayıda etkileşime girmesi gerektiğinden, doğrudan gerçek ortamda eğitilmesi, uzun eğitim süresi, yüksek maliyet ve oluşabilecek maddi hasarlardan dolayı zordur. Bu sebepten dolayı gerçek dünya uygulamaları için DPÖ ajanlarının eğitiminin büyük bir kısmı ya da tamamı sanal ortamlarda gerçekleştirilmektedir. Yapılan bu çalışmada, seyrek ödüllerin bulunduğu ayrık yapıdaki bir sanal ortamda eğitilen DPÖ ajanının ağ parametreleri kullanılarak bir mobil robotun, gerçek dünya ortamında hedefine ulaşma problemine odaklanılmıştır. DPÖ ajanının eğitimi için Minimalistic Gridworld sanal ortamı kullanılmıştır ve bilindiği kadarıyla bu çalışma Minimalistic Gridworld sanal ortamı için ilk gerçek dünya uygulamasıdır. Ortam genişlediğinde, klasik DQA algoritmasına göre daha yüksek performansa sahip bir DPÖ algoritması oluşturulmuştur. Gerçek dünya uygulamasında kullanılmak üzere düşük maliyetli bir mobil robot tasarlanmıştır. Önerilen tasarımda mobil robot, merkezi bir bilgisayardan uzun menzilli olarak kontrol edilebilir yapıdadır. Sanal ortam ile gerçek ortamı eşleştirebilmek için mobil robot ve hedefin pozisyonunu ile mobil robotun rotasyonunu tespit edebilen algoritmalar oluşturulmuştur. Sanal ortamda eğitilen modelin, gerçek ortamda daha verimli şekilde kullanılması sağlanmıştır. Sonuç olarak, sadece ortamın üstten görüntüsünü kullanan, başlangıç pozisyonu ve rotasyonundan bağımsız olarak hedefine ulaşabilen DPÖ tabanlı bir mobil robot geliştirilmiştir. Bu çalışmanın gerçek ortam deneylerinde DPÖ tabanlı mobil robot, farklı başlangıç koşullarıyla ile başlatılmış ve mobil robot tüm deneylerde hedefe başarılı bir şekilde ulaşabildiği gözlenmiştir.

Anahtar Kelimeler: Derin pekiştirmeli öğrenme, Derin Q-ağı, Mobil robot, Nesne tespiti

ABSTRACT

MOBILE ROBOT APPLICATION WITH HIERARCHICAL START POSITION DEEP Q-NETWORK ALGORITHM

ERKAN, Emre
PhD of Science in Department of Electrical and Electronics Engineering
Supervisor: Dr. Muhammet Ali ARSERİM
November 2022, 103 pages

The remarkable progress of deep learning has also significantly affected the reinforcement learning and resulted in deep reinforcement learning (DRL) method, which is a combination of both methods. DRL does not need a data set and has the potential beyond the performance of human experts, resulting in significant developments in the field of artificial intelligence. However, because a DRL agent has to interact with the environment a lot while it is trained, it is difficult to be trained directly in the real environment due to the long training time, high cost, and possible material damage. Therefore, most or all of the training of DRL agents for real-world applications is conducted in virtual environments. In this study, a DRL agent was trained in a discrete virtual environment with sparse rewards and focused on the real-world targeting problem of a mobile robot using these DRL network parameters. The Minimalistic Gridworld virtual environment was used for the training of the DRL agent, and as far as is known, this study is the first real-world application for the Minimalistic Gridworld virtual environment. A DRL algorithm with higher performance than the classical Deep Q-network algorithm was created with the expanded environment. A low-cost mobile robot was designed for use in a real-world application. In the proposed design, the mobile robot can be controlled from a central computer in a long range. To match the virtual environment with the real environment, algorithms that can detect the position of the mobile robot and the target, as well as the rotation of the mobile robot was created. The model trained in the virtual environment was enabled to be used more efficiently in the real environment. As a result, a DRL-based mobile robot was developed which used only the top view of the environment and could reach its target regardless of its initial position and rotation. In the real environment experiments of this study, the DRL-based mobile robot was started with different initial conditions and it was observed that the mobile robot could successfully reach the target in all experiments.

Keywords: Deep reinforcement learning, Deep Q-network, Mobile robot, Object detection

1. GİRİŞ

Yol planlama, robot araştırmalarının önemli bir konusudur. Bir mobil robotun otonom yol planlama görevini, başlangıç konumundan bağımsız olarak hedefe sorunsuz ve çarpmadan yapması istenir [1]. İlerleyen zamanlarda robot-insan etkileşiminin daha da artacağı öngörülmektedir [2,3]. Bu durumla beraber, robotlarla kullandığımız ortak alanlarda çoğalacağından yol planlamanın önemi daha da artmaktadır.

1.1 Problemin Tanımı

Literatürde A-yıldız, karınca koloni optimizasyonu ve yapay potansiyel alan gibi geleneksel yol planlama yöntemleri mevcuttur [4]. Ancak mobil robotun bulunduğu ortam karmaşıklaştıkça geleneksel yöntemler gerçek zamanlı görevleri karşılayamayacak duruma gelmektedirler [5]. Derin öğrenmenin (DÖ) yol planlama problemlerine uygulanması ile başarılı sonuçlar elde edilmiştir [6]. Bununla birlikte, DÖ'nün ortamı öğrenebilmesi için etiketli veri örneklerine ihtiyaç duymaktadır. Veri toplama ve etiketleme işlemi ise zaman alıcı bir görevdir [7]. Derin pekiştirmeli öğrenme (DPÖ), bahsedilen zorlukları aşabilecek yapısıyla son yıllarda araştırmacıların sıklıkla başvurdukları yöntemlerden biridir.

Pekiştirmeli öğrenme (PÖ), sayısal ödül sinyallerini en üst düzeye çıkarmak için hangi durumda hangi eylemin yapılması gerektiğini öğrenmeye odaklanır [8]. PÖ ajanına hangi eylemi yapacağı söylenmez bunun yerine hangi eylemlerin daha ödüllendirici olduğunu deneyerek öğrenmesi istenir [9]. Yeteri kadar ortam ile etkileşimde bulunan PÖ ajanı içinde bulunduğu durumda hangi hareketin kümülatif ödüle daha yüksek katkıda bulunduğunu, yani en iyi durum-eylem çiftini öğrenir. Ancak ortamdaki durum ve eylem sayısı büyüdükçe boyutsallık lanetinden (Curse of Dimensionality) dolayı ajanın ortamı keşfetmesi maliyetli ve zordur [10]. Bu nedenle, eylem ve durum uzayları yüksek boyutlara sahip bazı gerçek dünya problemlerini geleneksel pekiştirmeli öğrenme ile çözmek olanaksız hale gelebilmektedir [11].

DÖ'deki kayda değer ilerleme, PÖ'yi de önemli ölçüde etkilemiştir ve her iki yöntemin birleşimi olan DPÖ yöntemini ortaya çıkarmıştır [12]. DPÖ ile atari oyunları [13,14] masa oyunları [15,16] otonom sürüş [17], insanlarla etkileşime

geçen robotlar [18], insansız hava araçları [19], insansız yüzey araçları [20], robot navigasyon problemleri [21], robotik cerrahi [22] gibi çok sayıda duruma sahip karmaşık sistemlerde önemli başarılar elde edilmiştir.

1.2 Çalışmanın Amacı ve Katkıları

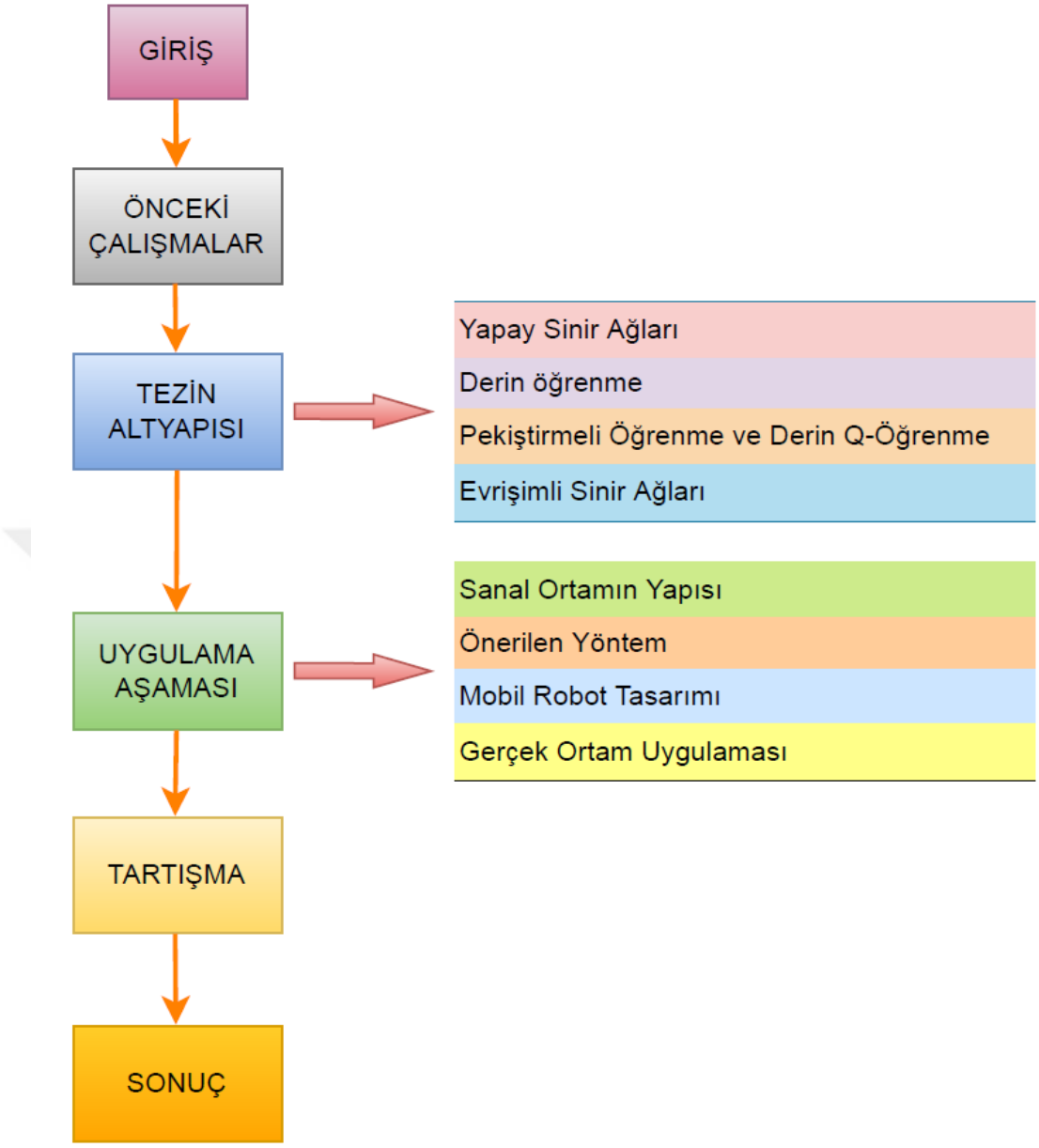
Yapılan bu çalışmada, seyrek ödüllerin bulunduğu ayrık bir yapıda olan sanal ortamda eğitilen DPÖ ajanının ağ parametreleri kullanılarak bir mobil robotun, gerçek dünya ortamında hedefine ulaşma problemine odaklanılmıştır.

Genel olarak, bu çalışmanın katkıları şu şekilde özetlenebilir:

- Seyrek ödüllerin olduğu büyük ortamlar için klasik DQA algoritmasına göre yakınsama hızı ve başarı oranını yükselten geliştirilmiş bir DQA algoritması önerildi.
- Minigrid ortamında eğitilen ajana ait ağ parametreleri ilk kez gerçek ortamda kullanıldı.
- Mobil robotun rotasyonunu tespit edebilen, piksel tabanlı bir görüntü işleme algoritması önerildi.
- Sanal ortamda eğitilen modelin, gerçek ortamda daha verimli şekilde kullanılması sağlandı.
- Düşük maliyeti ve merkezi bir bilgisayardan uzun menzilli olarak kontrol edilebilir yapısıyla, sürü robotiği ve çok ajanlı sistemlerde de kullanılma potansiyeline sahip bir mobil robot tasarlandı.

1.3 Tez Düzeni

Bu tez çalışmasının ikinci bölümünde, DPÖ yöntemi kullanılarak mobil robotların yol planlamasıyla ilgili yapılan son çalışmalara değinilmektedir. Tezin alt yapısını oluşturan üçüncü bölümünde, yapay sinir ağları (YSA), DÖ, DPÖ ve evrişimli sinir ağları (ESA) hakkında gerekli bilgiler verilmektedir. Tezin uygulama aşamasının açıklandığı dördüncü bölümde, kullanılan sanal ortamın yapısı, önerilen yöntem, tasarlanan mobil robot ve yapılan gerçek ortam uygulaması hakkında gerekli açıklamalar yapılmaktadır. Tezin beşinci bölümünde, yapılan uygulamanın hesaplama ve performans analizleri sunulmaktadır. Tezin altıncı bölümünde, tez çalışması sonuçlandırılmaktadır. Tez çalışmasının aşamaları Şekil 1.1'de gösterilmektedir.



Şekil 1.1- Tez çalışmasının aşamaları

2. ÖNCEKİ ÇALIŞMALAR

Zheng ve ark. [1], bir mobil robotun, haritasız ortam için yol planlama sürecini optimize eden bir yöntem önermişlerdir. Önerilen yöntemde, long short-term memory'nin (LSTM) özel yapısına dayalı bir bellek modülü tanıtılmıştır. Robotun uzun süreli hafıza kapasitesini artıran bu yöntem sayesinde, eğitim için gerekli olan deneme sayısı ve hesaplama süresi kısaltıldığı gösterilmiştir.

Gong ve ark. [5], statik engeller bulunan ortamlarda, mobil robotun yol planlama probleminin çözümü için DDPG ağının, LSTM ile optimize edilmesini önermişlerdir. Hızlı bir eğitim için karışık gürültü ve daha makul ödül fonksiyonu kullanılmıştır. Önerilen algoritma ile DDPG tabanlı algoritmalar [23,24] deneysel olarak karşılaştırılmıştır. Önerilen algoritmanın, karmaşık bir ortamda keşif verimliliği, optimum yol ve süre açısından avantajları gösterilmiştir.

Zhou ve ark. [7], devriye robotlarının yol planlama problemi için iyileştirilmiş bir DQA algoritması önerilmiştir. Ödül ceza fonksiyonları geliştirilmiş ve yeni ödül değer noktaları eklenerek durum-eylem alanı optimize edilerek seyrek ödül problemi çözülmüştür. Önerilen algoritmanın klasik DQA algoritmasına göre üstünlükleri deneysel olarak gösterilmiştir.

Wang ve ark. [23], mobil robot yol planlamada, zayıf keşif yeteneği ve seyrek ödül problemine odaklanılmış ve gelişmiş bir DQA yöntemi önerilmiştir. Durum-eylem alanını optimize etmek için yapay potansiyel alan yöntemini, ödül fonksiyonu ile birleştirerek ödül fonksiyonu iyileştirilmiştir. Önerilen yöntem ile klasik DQA'nın performansları karşılaştırılmıştır.

Xing ve ark. [24], alan bölmeli DQA yöntemini önermişlerdir. Önerilen yöntem ile eğitilen mobil kablosuz güç aktarım robotu, çok sayıda IoT cihazını adil bir şekilde şarj etmesi için en uygun yolu tespit edebildiği gösterilmiştir.

Huang ve ark. [25], bir mobil robotun, bilinmeyen dinamik bir ortamda yol planlama sürecinde karşılaşılan anormal ödül probleminin çözümü için iki ödül eşiği belirleyen

bir yöntem önerilmiştir. Önerilen yöntem ile değer tabanlı DPÖ algoritmalarında yapılan iyileştirme deneysel olarak gösterilmiştir.

Yu ve ark. [26], Sinir ağlarına ve hiyerarşik PÖ'ye dayalı mobil robot yol planlama yöntemi önermişlerdir. Önerilen yöntemin performansı, statik engellerin bulunduğu bir ortam için değerlendirilmiştir.

Wang ve ark. [27], mobil robot yol planlaması için PER ile DDQN tabanlı bir yöntem önermişlerdir. Simülasyon deneyleri ile bu yöntemin bilinmeyen ortamlarda, klasik DQA'ya göre daha iyi yakınsama hızına ve başarı oranına sahip olduğu gösterilmiştir.

Zhang ve ark. [28], statik engellerin bulunduğu ortamlar için DQA algoritmasının eğitimini olumsuz etkileyen benzer örnek fazlalığı problemlerine odaklanmışlardır. Benzerlik tarama matrisi kullanarak eğitim örneklerini optimize eden bir algoritma önermişlerdir. Böylece daha faydalı eğitim örnekleri kullanılabilmiştir.

Ruan ve ark. [29], engellerden kaçan mobil robotlar için D3QN tabanlı, LN-D3QN algoritması önermişlerdir. Önerilen algoritma sinir ağının katmanları normalleştirilerek, sinir ağının eğitimi hızlandırdığı tespit edilmiştir. Ayrıca öncelikli deneyim tekrarı kullanılarak öğrenme süresi azaldığı görülmüştür. LN-D3QN algoritması ile eğitilen robot, D3QN algoritması ile eğitilen robota göre bilinmeyen ortamlara daha hızlı uyum sağlayabildiği deneylerle gösterilmiştir.

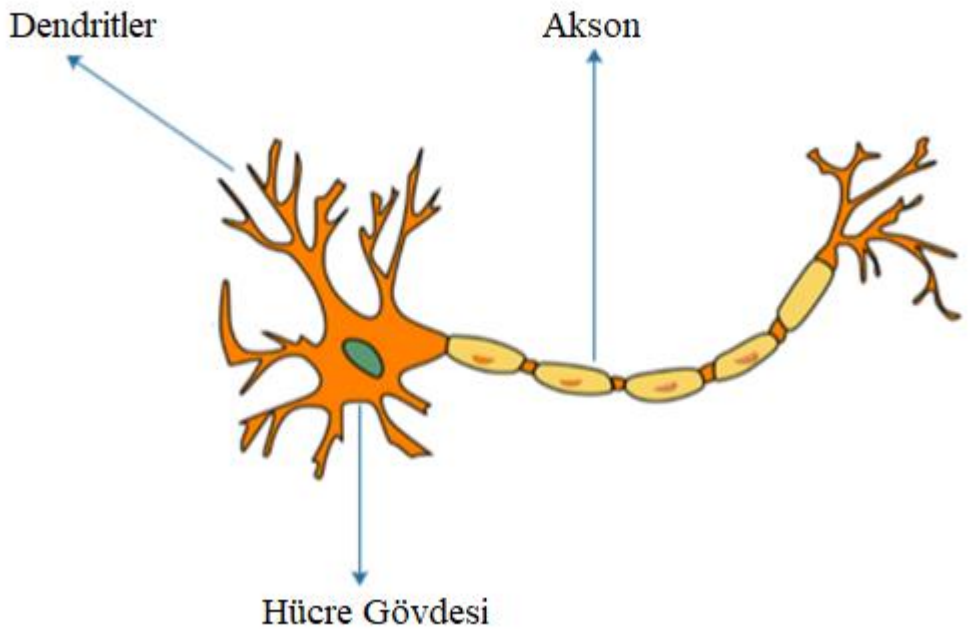
Kim ve ark. [30], bir mobil robotun, yol planlama problemini çözmek için Gated Recurrent Unit (GRU) ile birleştirilmiş DQA yöntemini önermişlerdir.

3. TEZİN ALTYAPISI

Tezin bu bölümünde, tezin altyapısını oluşturan gerekli teorik bilgiler sunulmaktadır.

3.1 Yapay Sinir Ağları

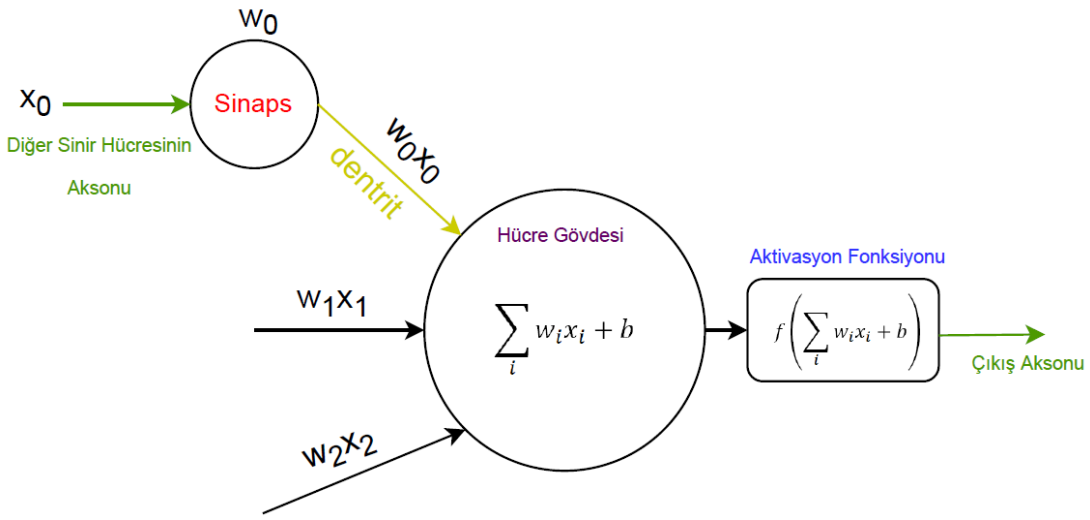
Beynin yapı taşları nöron adı verilen sinir hücreleridir. Nöronların elektriksel ve kimyasal sinyalleri birbirlerine aktarma özellikleri vardır. Bir nöron Şekil 3.1'de görüldüğü gibi gövde (soma), akson ve dendrit olarak adlandırılan üç ana bölümden oluşmaktadır [31].



Şekil 3.1 Nöronun yapısı [32]

Nöronda bilgi, elektriksel sinyaller olarak iletilmektedir. Bu elektriksel sinyallere aksiyon potansiyeli adı verilmektedir. Nöron gelen bilgiyi dendritler vasıtasıyla almakta ve hücre gövdesinde toplamaktadır. Bu toplam sonucunda aksiyon potansiyeli yeterli düzeye ulaşması durumunda bilgi diğer sinir hücrelerine aktarılmaktadır [32]. Bir sinir hücresinin temel çalışma prensibi bu şekildedir.

İnsan sinir sistemi milyarlarca nörondan oluşmakta ve bu nöronlar birbirlerine sinapslar ile bağlanmaktadır. YSA, insan sinir sisteminin matematiksel olarak basit bir modelidir [33]. YSA'yı oluşturan yapay nöronların genel yapısı Şekil 3.2'de gösterilmektedir.



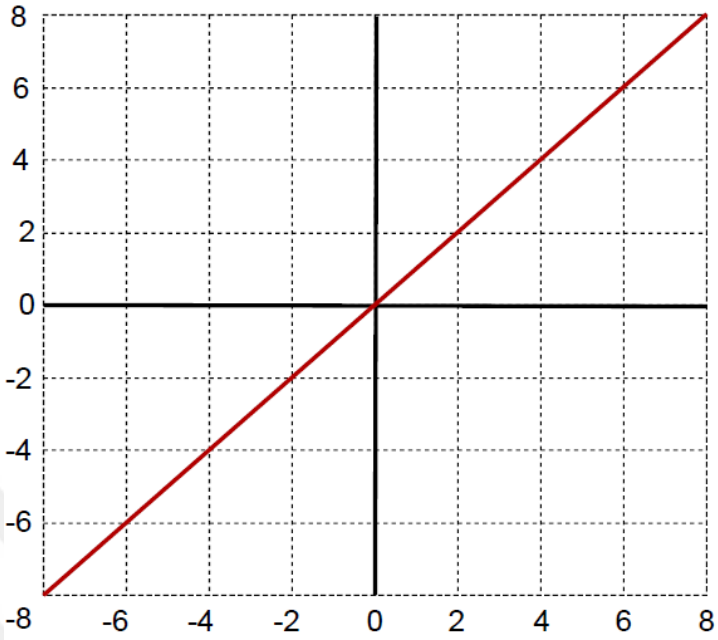
Şekil 3.2 Yapay nöronun yapısı [33]

Biyolojik nöronda, diğer sinir nöronlardan gelen sinyaller dendritler vasıtasıyla alınarak, aksona çıkış sinyalini iletmektedir. Akson kolları diğer nöronların dendritlerine, sinapslar ile bağlanmaktadır. Yapay nöronda ise gelen sinyaller x_0, x_1, … , x_n ile gösterilmekte, sinapslar, w_0, w_1, …, w_n ağırlıklarıyla temsil edilmektedir. Bilgi sinyali ile ağırlıkların çarpımı (x_0*w_0 , x_1*w_1, …, x_n*w_n) dendrit işlevini gerçekleştirmektedir. Bir nöronun, diğer bir nörona etkisi ağırlıklar vasıtasıyla kontrol edilmektedir. Biyolojik nöronda dendritlerden gelen bilgi sinyalleri hücre gövdesinde toplanmakta ve toplanan bu sinyallerin büyüklüğü belli bir eşik değerin üzerinde ise nöron aktifleşerek oluşturduğu bilgi sinyalini diğer nörona akson boyunca iletmektedir. Bu işlem yapay nöronda aktivasyon fonksiyonu (f) tarafından yapılmaktadır [33].

Aktivasyon fonksiyonları, nörona gelen girdi sinyaline sabit bir matematiksel işlem uygulayarak çıktı sinyali üretmektedir [33]. YSA'ların karmaşık yapıları öğrenmesinde ve girdiler ile çıktılar arasındaki bağıntıyı anlamlandırmasında aktivasyon fonksiyonları önemli rol oynamaktadır [34].

Aktivasyon fonksiyonları doğrusal ve doğrusal olmayan aktivasyon fonksiyonları olarak iki ana gruba ayrılabilir. Doğrusal aktivasyon fonksiyonları $f(x) = x$ olarak ifade edilen fonksiyonlardır. Bu fonksiyonlar, karmaşık veya çok parametreye sahip yapılarda başarılı değillerdir. Yapısı gereği sadece basit doğrusal regresyon

problemlerinde başarılı olabilirler. Doğrusal aktivasyon fonksiyonuna ait grafik Şekil 3.3'te gösterilmektedir.

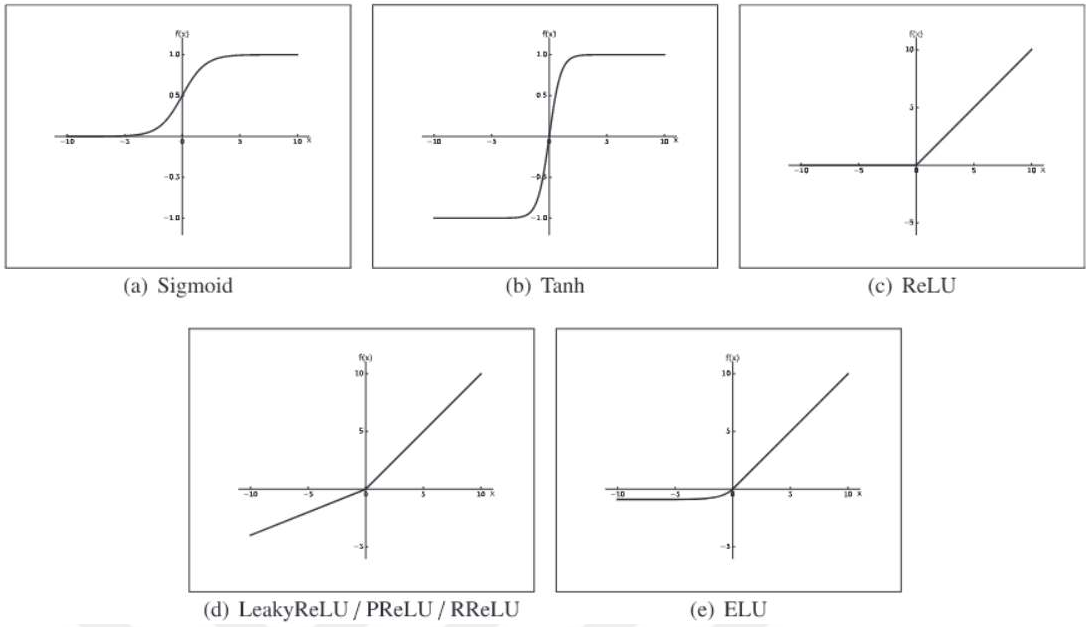


Şekil 3.3 Doğrusal aktivasyon fonksiyonu

Öğrenilmesi istenen veri türü görüntü, video, ses gibi daha karmaşık yapıya sahip ise doğrusal olmayan aktivasyon fonksiyonları kullanılmalıdır. Çünkü doğrusal olmayan aktivasyon fonksiyonları birden fazla dereceye sahip eğrilerle ifade edildiğinden girdiler ile çıktılar arasında doğrusal olmayan bağlantılar kurma imkânı sağlayabilmektedir. Böylelikle YSA herhangi bir karmaşık fonksiyonun çözümünü yakınsama ile bulabilmektedir. Yani insan tarafından çözülmesi gereken problemlerin birçoğu, yeterli parametreye sahip ve uygun aktivasyon fonksiyonu kullanan YSA'lar tarafından bir çeşit fonksiyonel yapıya dönüştürülerek çözülebilmektedir. Şekil 3.4'te sıklıkla kullanılan doğrusal olmayan aktivasyon fonksiyonları görülmektedir.

Şekil 3.4 (a)'da gösterilen sigmoid aktivasyon fonksiyonuna ait matematiksel ifade denklem (3.1)'de görülmektedir. Sigmoid aktivasyon fonksiyonu aldığı girdi sinyalini [0,1] aralığında çıktı sinyaline dönüştürmektedir. Nöronun çıktı sinyalinin bire yaklaşması nöronun ağdaki etkisinin artması, sıfıra yaklaşması ise ağdaki etkisinin azalması anlamına gelmektedir.

$$f(x) = \frac{1}{1+e^{-x}} \tag{3.1}$$



Şekil 3.4 Doğrusal olmayan aktivasyon fonksiyonları [35]

Şekil 3.4 (b)'de gösterilen tanh aktivasyon fonksiyonuna ait matematiksel ifade denklem (3.2)'de görülmektedir. Tanh aktivasyon fonksiyonu, sigmoid aktivasyon fonksiyonuna benzer bir grafik üretir fakat çıktı sinyali [-1,1] aralığındadır. Ancak hem sigmoid hem de tanh aktivasyon fonksiyonları günümüzde popülerliğini kaybetmeye başlamıştır. Derin öğrenmenin, karmaşık problemlerin çözümünde kullanılmasıyla beraber derin öğrenmenin temelini oluşturan çok sayıda nöron ve katman içeren YSA'lar modellenmektedir. Sigmoid ve tanh aktivasyon fonksiyonları çok katmanlı YSA'larda kullanılması geri yayılım esnasında gradyanın sıfıra yaklaşmasına sebebiyet verdiğinden dolayı kullanılması uygun değildir [36].

$$f(x) = \frac{2}{1+e^{-2x}} - 1 \tag{3.2}$$

Şekil 3.4 (c)'de gösterilen ReLU aktivasyon fonksiyonuna ait matematiksel ifade denklem (3.3)'te görülmektedir. ESA mimarilerinde yaygın olarak kullanılan ReLU aktivasyon fonksiyonu, giriş sinyali sıfırdan küçükse, çıkış sinyali olarak sıfır üretmekte, giriş sinyali sıfırdan büyükse giriş sinyalini çıkışa aktarmaktadır. ReLU aktivasyon fonksiyonu, hızlı yakınsama ve düşük maliyetli işlemler gerektirmesinden dolayı sıklıkla tercih edilmektedir. Ayrıca Şekil 3.4 (d)'de gösterilen ReLU aktivasyon fonksiyonundan türetilmiş LeakyReLU, PReLU ve RReLU aktivasyon fonksiyonları da vardır. İlgili aktivasyon fonksiyonlara ait matematiksel ifade denklem (3.4)'te gösterilmektedir.

$$f(x) = \begin{cases} x & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{3.3}$$

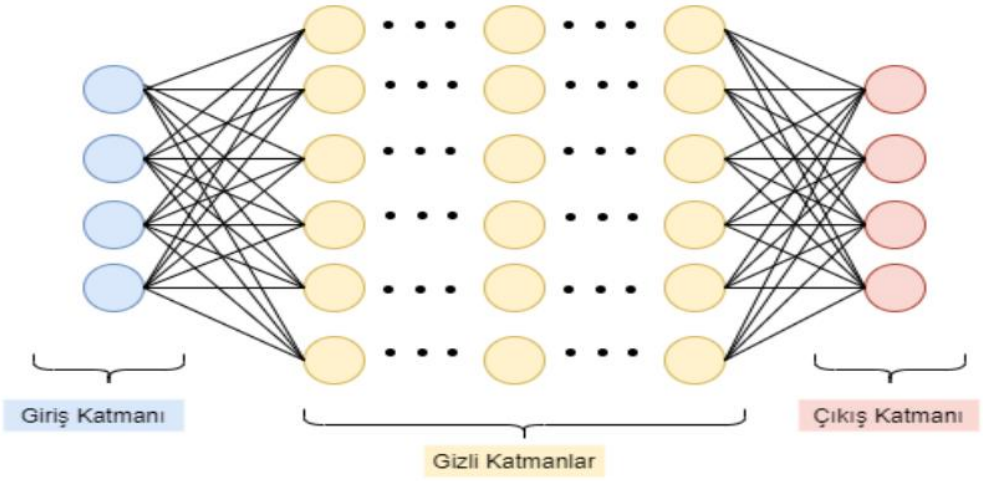
$$f(x) = \begin{cases} x & x \geq 0 \\ \alpha x & x < 0 \end{cases} \tag{3.4}$$

Şekil 3.4 (e)'de gösterilen ELU aktivasyon fonksiyonuna ait matematiksel ifade denklem (3.5)'te görülmektedir. ELU aktivasyon fonksiyonu düşük işlem maliyeti, hızlı yakınsama ve doğru sonuç üretme eğilimindedir. ELU aktivasyon fonksiyonunda pozitif olması gereken alfa parametresi mevcuttur.

$$f(x) = \begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases} \tag{3.5}$$

3.2 Derin Öğrenme

DÖ'nün alt yapısını YSA'lar oluşturmaktadır ancak DÖ, YSA'lardan farklı olarak çok sayıda doğrusal olmayan şekilde birbirine bağlı gizli katmandan oluşmaktadır. DÖ sinir ağının genel yapısı Şekil 3.5'te gösterilmektedir.



Şekil 3.5 Derin öğrenme sinir ağı [37]

DÖ'nün alt yapısı geçmişe dayanmasına rağmen etkin bir şekilde kullanılma imkânı çok daha sonra ortaya çıkmıştır. DÖ modellerinin eğitilmesi için büyük miktarda verinin işlenmesi gerektiğinden, verinin elde edilmesi, veri işleme süresi ve kapasitesi DÖ'nün gelişmesini engelleyen faktörlerdi. Gelişen teknolojiyle beraber, erişilebilen veri miktarı çoğaldı, bilgisayarların veri işleme kapasiteleri ve hızları arttı, paralel işlem yapma yeteneğine sahip grafik işlemci birimlerinin (GİB) gelişmesiyle beraber ise veri çok daha kolay işlenebilir hale geldi.

DÖ karmaşık problemleri uzman insanlar seviyesinde çözme kabiliyetine sahiptir. Günümüzde DÖ kullanılarak, nesne tanıma [38], görüntü işleme [39], ses tanıma [40], doğal dil işleme [41], Otonom sürüş [42], robotik [43], hastalıkların teşhisi [44,45] gibi birçok farklı alanda başarılı sonuçlar elde edilmektedir.

DÖ'deki gelişmeler ile birlikte çok sayıda DÖ mimarisi ortaya çıkmıştır. Bunlardan bazıları, tekrarlayan sinir ağları (TSA), Boltzmann makinaları (BM), uzun-kısa vadeli hafıza ağları (UKVH), derin inanç ağları (DİA), derin oto kodlayıcılar (DOK) ve ESA'dır. Kullanılan verinin türü, verinin miktarı ve çalışmanın konusu, DÖ mimarisinin seçiminde önemli rol oynamaktadır [37].

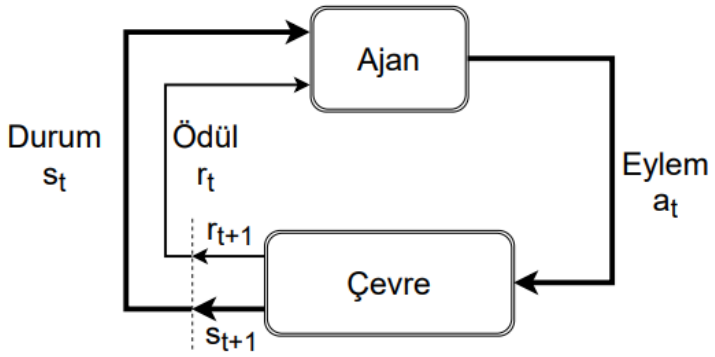
Tezin uygulama aşamasında, nesne tespiti için ESA mimarisi kullanıldığından dolayı tezin bu bölümünde ESA'larla ilgili bilgiler sunulmaktadır.

3.3 Pekiştirmeli Öğrenme ve Derin Q-Öğrenme

Bu bölümde pekiştirmeli öğrenme ve derin Q-öğrenme hakkında gerekli arka plan bilgisi sunulmaktadır.

3.3.1 Pekiştirmeli öğrenmenin temel fikri

PÖ, modellerin ön bilgiye ihtiyaç duymadan karmaşık ve/veya belirsiz ortamlarda kararlar dizisi almayı öğrendiği bir makine öğrenmesi eğitim yöntemidir. PÖ algoritmaları, herhangi bir dış denetim veya modelleyiciden, sorunun nasıl çözüleceğine dair ipuçları olmadan bir sorun için en iyi çözümü bulmak için yinelemeli bir deneme-yanılma süreci uygular. Algoritma yalnızca yapılan eylemler sonucunda elde edilen ödüller aracılığıyla hangi eylemlerin en uygun olduğunu öğrenir. Algoritmada iyi bir eylem yapıldığında olumlu bir ödül, kötü bir eylem yapıldığında ise ceza (olumsuz bir ödül) puanı alınır. Öğrenme süreci çok sayıda deneme-yanılma eylemlerinden oluşmaktadır. Algoritma zamanla deneyim kazanır ve hangi eylemlerin daha iyi sonuçlara yol açtığını öğrenir.



Şekil 3.6 Pekiştirmeli öğrenme şeması

PÖ üç temel unsurdan oluşur. Bunlar Şekil 3.6'da gösterildiği gibi ajan, çevre ve ödüllerdir. Ajan, eylemleri gerçekleştiren varlıktır. Çevre, ajanın içinde bulunduğu dünyadır. Çevre ajanın yaptığı eylemlere yanıt verir. Ajan bir eylem gerçekleştirdiğinde, ajanın o anki durumu değişir ve çevrede yeni bir durum oluşur. Böylece yeni zaman durumuna geçen ajan, gerçekleştirmiş olduğu eyleme göre çevrenin sağlamış olduğu olumlu ya da olumsuz skaler büyüklükte bir ödül alır. Alınan bu ödüller, ajanın belirli bir eylemin ne kadar iyi olduğunu ölçmesini sağlar.

PÖ terimleri ile ilgili kısa bilgiler aşağıda sunulmaktadır.

Durum (S): S, ajana göre çevrenin mevcut durumunu göstermektedir.

Eylemler ($\boldsymbol{A}$): A, ajan s durumundayken gerçekleştirebileceği tüm olası eylemler kümesini göstermektedir. Eylem kümeleri ayrık ve sonludur. Her bir zaman diliminde ajan tek bir eylem seçebilir.

Ödül (R): R, ajanın gerçekleştirmiş olduğu eyleme göre çevrenin verdiği anlık geri bildirimdir. Ödüller, ajanın başarısını ya da başarısızlıklarını değerlendirmek için kullanılmaktadır.

Politika ($\boldsymbol{\pi}$): Ajanın içinde bulunduğu duruma göre seçmesi gereken eylemi belirleme stratejisidir.

Geçiş fonksiyonu ($\boldsymbol{T}$): Ajanın belli bir eylem gerçekleştirmesi ile içinde bulunduğu çevrenin değişikliğini göstermektedir.

İndirim faktörü ($\boldsymbol{\gamma}$): Ajanın çevreden aldığı ödüller anında ya da gecikmeli olabilir. Gecikmeli ödüller, ajanın içinde bulunduğu durumlara göre seçmiş olduğu ardışık hareketler sonucunda elde etmiş olduğu ödüllerdir. Bu ödüller, anlık ödüllere eşit ya da daha az değerlidir. Gecikmeli ödüllerin değeri γ faktörü ile belirlenir. Gecikmeli ödüllerin değerleri, 0-1 aralığında değer alabilen γ faktörü oranında indirime tabi tutularak belirlenir. γ'nın değerinin 1'e yaklaşması gecikmeli ödüllerin değerini

arttırırken 0'a yaklaşması gecikmeli ödüllerin değerini düşürmektedir. γ'nın değerinin 1 olması durumunda ise gecikmeli ödüller herhangi bir indirime tabi tutulmazlar yani gecikmeli ödüller de anlık ödüllerle aynı oranda değerli olmaktadır.

Değer (V): Ajanın yapmış olduğu ardışık eylemler sonucu γ oranına göre indirim uygulanmış ödüllerin getirisidir. $V^{\pi}(s)$, π politikası kapsamında mevcut durumun beklenen uzun vadeli getirisidir.

Q değeri (Q): $Q^{\pi}(s,a)$, mevcut durumda π politikası kapsamında eyleme geçmenin uzun vadeli getirisidir.

3.3.2 Markov karar süreci

Markov Karar Süreci (MKS), eylemlerin sadece bir sonraki ödülü değil gelecek durumları da etkilediği sıralı karar verme sürecinin formülize edilmiş halidir. Aynı zamanda PÖ probleminin matematiksel olarak idealleştirilmiş biçimidir. PÖ ajanları genellikle MKS olarak modellenmektedir [8].

Bir eylem seçme probleminin çıktısı sadece şimdiki duruma bağlı ve önceki durumlara, eylemlere, ödüllere bağlı değilse, bu problem Markov özelliği göstermektedir. Ancak gerçek hayat problemleri PÖ ile çözülebilmesi için tamamen MKS özelliği göstermese de MKS'ye yakın tahminlerinin olması gerekmektedir. Eğer problemin iyi bir MKS tahmini yoksa ajanın PÖ ile eğitilmesi zordur [46], çünkü PÖ'de ajanın problemi çözebilmesi için sadece anlık durum bilgisi verilir ve uygun eylemi seçmeyi öğrenmesi beklenir.

Bir MKS <S,A,R,T> parametreleriyle tanımlanabilir; burada $S = \{s_1, \dots, s_n\}$ sonlu bir durumlar kümesini ve $A = \{a_1, \dots, a_m\}$ sonlu eylemler kümesini göstermektedir. T: $S\ x\ A\ x\ S \rightarrow [0,1]$ fonksiyonu, s durumunda a eylemi gerçekleştirme ve s' durumunda sona erme olasılığını belirleyen geçiş fonksiyonunu tanımlamaktadır. s durumunda a eylemi gerçekleştirmenin ve s' durumuna geçmenin ödülü, R:$S\ x\ A\ x\ S \rightarrow \mathbb{R}$ ödül fonksiyonu ile temsil edilmektedir.

Bir eylemin sonucu yalnızca önceki duruma ve eyleme bağlıysa, sistem MKS özelliğini yerine getirir;

$$P(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \dots) = (s_{t+1}|s_t, a_t) \tag{3.6}$$

Ajanın belli bir durumda hangi eylemi seçeceği politika ile belirlenmektedir. π politikasına göre her durum bir eylem ile eşleşmektedir: $\pi = S \rightarrow A$. Bir ajan, π politikasına göre hareket ederse, $a_0 = \pi(s_0)$ eylemini seçerek s_1 durumuna geçecektir. Bu eylem sonucunda ajan, $r_0 = R(s_0, a_a, s_1)$ ödülünü alacaktır.

Toplam ödül, anında ve gecikmeli ödüllerin kümülatif toplamına eşittir. Gecikmeli ödüllere her bir zaman dilimi için $\gamma \epsilon [0,1]$ indirim faktörü uygulanır. Ajanın amacı, beklenen ödülü zaman içinde en üst düzeye çıkarmaktır. Toplam ödül, denklem (3.7)'de gösterildiği gibi hesaplanmaktadır.

$$R_t = E[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots] = E[\sum_{k=0}^{\infty} \gamma^k r_{t+k}] \quad (3.7)$$

MKS'de durum değeri fonksiyonu ve durum-eylem değeri fonksiyonu olmak üzere iki farklı değer fonksiyonu tanımlanabilir;

$V^{\pi}(s)$, π politikası altındaki s durumunun değerini göstermektedir. Durum değer fonksiyonu denklem (3.8)'de gösterilmektedir.

$$V^{\pi}(s) = E_{\pi}[R_t | s_t = s]$$

$$= E_{\pi}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \middle| s_t = s\right]$$

$$= E_{\pi}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots | s_t = s]$$

$$= E_{\pi}[r_t + \gamma V^{\pi}(s_{t+1}) | s_t = s]$$

$$= \sum_{s' \in S} T(s, \pi(s), s')\big(R(s, a, s') + \gamma V^{\pi}(s')\big) \quad (3.8)$$

Burada s' sonraki durumu belirtir. Bu denklem aynı zamanda Bellman denklemi olarak da bilinir. s durumunun değeri ile ardıl durumlar arasındaki ilişkiyi tanımlar. $Q^{\pi}(s, a)$, π politikası altında, durum-eylem değer fonksiyonunu göstermektedir. $Q^{\pi}(s, a)$ durum-eylem çiftinin ne kadar iyi olduğunun bir ölçüsüdür. Durum-eylem değer fonksiyonunu denklem (3.9)'da gösterilmektedir.

$$Q^{\pi}(s, a) = E[R_t | s_t = s, a_t = a]$$

$$= E_{\pi}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \middle| s_t = s, a_t = a\right]$$

$$= \sum_{s' \in S} T(s, \pi(s), s')(R(s, a, s') + \gamma V^{\pi}(s')) \quad (3.9)$$

Tüm durumlar için diğer politikalara eşit veya daha büyük beklenen değerlerle sonuçlanan politikaya optimal politika denir ve π^* ile gösterilmektedir. Optimal politikaya göre değer fonksiyonları aşağıdaki gibi tanımlanır.

$$V^*(s) = \max_{a \epsilon A} \sum_{s' \in S} T(s, \pi(s), s')\big(R(s, a, s') + \gamma V^{\pi}(s')\big) \quad (3.10)$$

$$Q^*(s, a) = \sum_{s' \in S} T(s, \pi(s), s')(R(s, a, s') + \gamma \max_{a \epsilon A} Q^*(s', a')) \quad (3.11)$$

T ve R biliniyorsa, denklem (3.10)'daki özyinelemeli yapı kullanılarak optimal politika bulunabilir. Model tabanlı olan bu yaklaşımda ajanın, çevre ile doğrudan etkileşime girmesi gerekmez. Ancak çoğu sitemde T ve R'yi önceden belirlemek zordur. Böyle durumlar için PÖ algoritmaları uygulanabilir. Deneme-yanılma süreçlerini kullanan PÖ algoritmaları modelsiz yaklaşımlardır.

Optimal değer fonksiyonu ve politikası yalnızca Q fonksiyonu cinsinden ifade edilebilir:

$$V^*(s) = \max_{a \epsilon A} Q^*(s, a) \quad (3.12)$$

$$\pi^*(s) = \arg max_{a \epsilon A} Q^*(s, a) \quad (3.13)$$

Modelsiz olan bu yaklaşım için T veya R'yi bilmeye gerek yoktur. Optimal politikayı bulabilmek için Q-fonksiyonu yeterli olmaktadır. Q-fonksiyonu deneme-yanılma ile yinelemeli olarak tahmin edilmekte ve güncellenmektedir.

3.3.3 Pekiştirmeli öğrenme algoritmaları

Modelsiz PÖ algoritmaları, değer fonksiyon tabanlı veya politika arama tabanlı olarak sınıflandırılabilir. Ayrıca hem değer fonksiyon tabanlı hem de politika arama tabanlı hibrit aktör-eleştirmen yaklaşımları da mevcuttur.

Ayrıca, değer temelli yaklaşımlar, politikaya dayalı ve politika dışı algoritmalar olarak ayrılabilir. Politika dışı algoritmalarda, eğitim sırasında optimal olmayan politikalara göre eylemler seçilebilir. Politikaya dayalı algoritmalar da ise optimal olduğu tahmin edilen politika seçilmektedir.

Bu tez için ajanın eğitiminde, sanal ortam ayrık ve sınırlı sayıda durum içermesinden dolayı Derin Q-Ağı (DQA) algoritması ve bu tez kapsamında önerilen DQA temelli

bir algoritma kullanılmaktadır. Kullanılan algoritmalara ait ayrıntılı bilgi tezin ilerleyen kısımlarında verilecektir.

Temel PÖ algoritmalarının karşılaştırılması Tablo 3.1'de verilmektedir.

Tablo 3.1 Pekiştirmeli öğrenme algoritmalarının karşılaştırılması

Algoritma	Yaklaşım	Eylem ve Durum Uzayı
Monte Carlo	Değer temelli	Ayrık
Q-learning (Q-öğrenme)	Değer temelli	Ayrık
SARSA (Durum-eylem-ödül-durum-eylem)	Politikaya dayalı	Ayrık
Q-learning-Lambda (Q-öğrenme-Lambda)	Değer temelli	Ayrık
SARSA- Lambda (Durum-eylem-ödül-durum-eylem- Lambda)	Politikaya dayalı	Ayrık
DQN (Derin Q-Ağı)	Değer temelli	Ayrık
DDPG (Derin deterministik politika gradyant)	Değer temelli	Sürekli
A3C (Asenkron avantaj aktör-kritik)	Politikaya dayalı	Sürekli
NAF (Normalize avantaj fonksiyonları ile Q-öğrenme)	Değer temelli	Sürekli
TRPO (Güven bölgesi politika optimizasyonu)	Politikaya dayalı	Sürekli
PPO (Yaklaşık Politika Optimizasyonu)	Politikaya dayalı	Sürekli
TD3 (Çift gecikmeli derin deterministik politika gradyant)	Değer temelli	Sürekli
SAC (Soft aktör-kritik)	Değer temelli	Sürekli

3.3.4 Tablolu Q-öğrenme

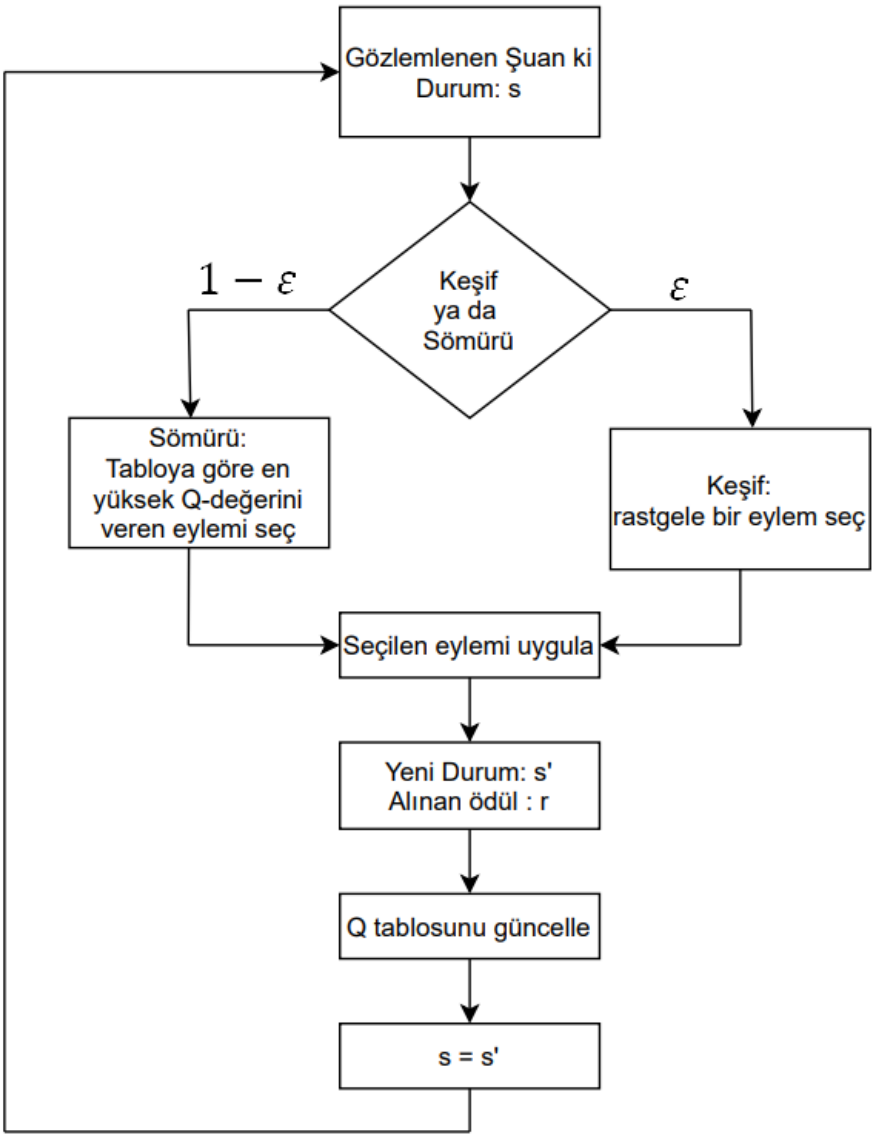
Q-öğrenme algoritmasında, olası tüm durum-eylem çiftlerinin ne kadar değerli olduğunu gösteren Q-değerleri bir arama tablosuna yerleştirilir. Q-değerleri mümkün olduğunca çok durum-eylem çifti denenerek güncellenir. Her t adımındaki güncellemeler aşağıdaki denklem kullanılarak gerçekleştirilir:

$$\underbrace{Q_{t+1}(s_t, a_t)}_{yeni\ Q-değeri} = \underbrace{Q_t(s_t, a_t)}_{eski\ Q-değeri} + \underbrace{\alpha}_{\text{öğrenme oranı}} \overbrace{\left[\underbrace{r_t + \gamma \underbrace{\max_{a' \epsilon A} Q_t(s_{t+1}, a)}_{Optimum\ gelecek\ Q-değeri\ yaklaşımı} - \underbrace{Q_t(s_t, a_t)}_{eski\ Q-değeri}}_{yeni\ değer\ (zamansal\ fark\ hedefi)}\right]}^{zamansal\ fark} \quad (3.14)$$

Algoritma, eski Q-değerini ve zamansal fark değerini içeren Bellman denklemini kullanarak yeni Q-değerini günceller. Öğrenme hızı $a\epsilon[0,1]$ yeni bilgilerin öğrenilme

hızını belirler. α'nın 0'a yakın olması algoritmanın öğrenememesi, α'nın 1'e yakın olması algoritmanın öğrendiği bilgileri daha az kullanması anlamına gelmektedir.

Şekil 3.7'de algoritmaya ait akış diyagramı verilmektedir. Algoritmaya göre her durum-eylem çifti sonsuz kez ziyaret edilirse, Q-öğrenme optimal bir politikaya yakınsar. Pratikte sonsuz kez işlem yapılması yerine modelin performansı kabul edilebilir bir eşiğin üstüne çıkıncaya kadar işlem yapılır.



Şekil 3.7 Q-öğrenme akış diyagramı

3.3.5 Keşif ve sömürü

Pekiştirmeli öğrenme algoritmalarında sistemin öğrenmesi ve öğrendiği bilgiyi kullanabilmesi için keşif ve sömürü arasında iyi bir denge olmalıdır. PÖ'de daha

değerli durum-eylem çiftlerini tespit edebilmek için o zamana kadar karşılaşılmamış durum-eylem kombinasyonları keşfedilmelidir. Mevcut duruma ait optimal çözüm bulunduktan sonra ise elde edilen ödülü maksimize edebilmek için mevcut durumdaki en iyi eylem seçilerek sömürü gerçekleştirilmelidir. Aşırı keşif, ajanın ödül alma kapasitelerini sınırlandırmasına ve dolayısıyla performanslarını düşürmesine yol açar. Aşırı sömürü, ajanın potansiyel olarak daha iyi politikaları öğrenmemesine yol açar [47].

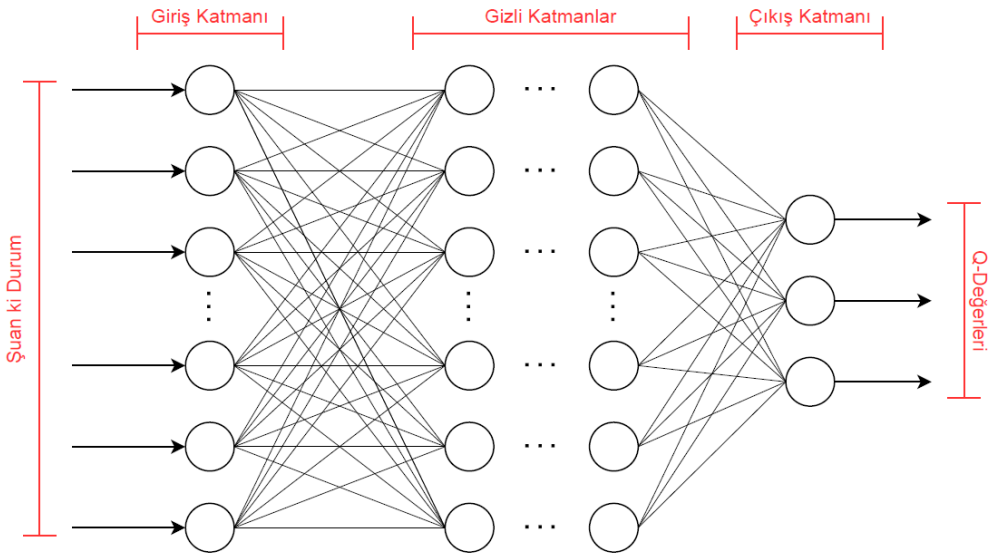
PÖ'de farklı keşif stratejileri olmakla birlikte en bilinen stratejilerden biri $\varepsilon - greedy$ (açgözlü ε) ile eylem seçimidir. ε, [0,1] aralığında değerler alabilmektedir. ε, keşif yapma ihtimalini temsil ederken $1 - \varepsilon$, sömürü gerçekleştirme ihtimalini temsil etmektedir. ε'un 1'e yakın olması ajanın çevrede rastgele hareketler yaparak keşif yapma ihtimalini arttırırken, ε'un sıfıra yakın olması ajanın öğrendiği en iyi eylemi yapma ihtimalini arttırmaktadır. Genellikle eğitimin başlangıcında ε'nun değeri bire yakın olarak ayarlanır ve eğitimin ilerleyen safhalarında çok küçük oranda değeri azaltılır. Eğitim ilerledikçe ε sıfıra yaklaşır, isteğe bağlı olarak bu değer sıfıra kadar azaltılabilir ya da küçük bir değerde sabitlenebilir. Eğitimin başlangıcında ajanın eylem seçimi için keşif ön plandayken, eğitimin ilerleyen aşamalarında ajanın eylem seçiminde öğrenilen bilgi ön plana çıkmaktadır. Hatta ε, sıfıra kadar azaltılırsa, ajan eylem seçimi için sadece öğrendiği bilgiyi kullanacaktır. Keşif-sömürü dengesi, algoritmanın performansını önemli ölçüde etkileyebilmektedir.

3.3.6 Derin Q-ağı

Pekiştirmeli öğrenmede, durum-eylem uzayı büyüdükçe, boyutsallık laneti olarak adlandırılan etkiden dolayı ajanın ortamı keşfetmesi zorlaşabilir. Bu nedenle, durum-eylem uzayı yüksek boyutlara sahip bazı gerçek dünya problemlerini geleneksel pekiştirmeli öğrenme ile çözmek olanaksız hale gelebilmektedir. Çünkü her durum-eylem çiftini depolamak için çok fazla depolama alanına ihtiyaç duyulacak ve hesaplama maliyetleri ve öğrenme süreleri katlanarak artacaktır. Ayrıca algoritmanın problemi çözebilmesi için her durum-eylem çiftini defalarca kez ziyaret etmesi gerekmektedir. Böyle bir durum, büyük durum-eylem uzayına sahip problemler için yapılabilmesi çok zor olduğundan en uygun politikanın bulunması da zor veya imkânsız hale gelecektir.

Boyutsallık sorununu çözmek için YSA iyi bir yaklaşımdır. YSA yaklaşımında tüm durum eylem uzayını ziyaret etmeye gerek yoktur.

Boyutsallık sorununu çözmek için farklı yaklaşımlar mevcuttur. En önemli yaklaşımlardan biri Derin Yapay Sinir Ağlarının (DYSA) kullanımıdır. DYSA'nın kullanılarak en uygun Q değerini bulma yaklaşımı, DQA olarak isimlendirilir. YSA'larının karmaşık ve doğrusal olmayan fonksiyonları tahmin etme yetenekleri, DQA'yı kullanışlı bir araç haline getirmektedir. DQA'nın genel yapısı Şekil 3.8'de gösterilmektedir.



Şekil 3.8 DQA'nın genel yapısı

DQA'nın da kullandığı DYSA'ları birden çok gizli katmana sahip YSA'dır. YSA'ya daha fazla gizli katman eklenmesi eğitim süresini arttırmakla birlikte daha karmaşık problemleri çözebilme yeteneği sağlayabilir. Yani, gizli katman ve nöron sayısı problemin karmaşıklığına göre belirlenmelidir.

DQA, denklem (3.14)'deki yaklaşımla Q-değerini güncellemek için DYSA'yı kullanmaktadır. DYSA'da ağın ağırlıklarını güncellemek için genellikle gradyan iniş ve geri yayılım yöntemlerinden yararlanılır. Ağın ağırlıklarını güncelleyebilmek için tahmini Q değeri ile gerçek Q değeri arasındaki farkı ölçen bir maliyet fonksiyonuna ihtiyacımız vardır. Amaç bu iki değer arasındaki hata fonksiyonunu en aza indirmektir. Gerçek Q değeri bilinmediğinden, denklem (3.14)'deki zamansal fark

hedefi kullanılarak Q-değeri tekrar tahmin edilebilir. İndirimli gelecek ödülleri de dâhil olmak üzere tüm gelecek zaman adımlarının toplam beklenen ödülünü temsil eder. Ardından hedef değeri yinelemeli olarak güncellenebilir.

Kare hata kaybı (KHK) fonksiyonunu kullanılarak kayıp fonksiyonu aşağıdaki gibi kurulabilir:

$$L = \frac{1}{2}\left[Q(s_t, a_t) - \left(r_t + \gamma V(s_{t+1})\right)\right] \tag{3.15}$$

KHK fonksiyonunun türevi alınırsa gradyan iniş gerçekleştirilmiş olur:

$$\frac{\delta L}{\delta Q(s_t, a_t)} = Q(s_t, a_t) - \left(r_t + \gamma V(s_{t+1})\right) \tag{3.16}$$

Denklem (3.16) kullanılarak, Q değeri için bir güncelleme kuralı oluşturulabilir:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) - \alpha\left[Q(s_t, a_t) - \left(r_t + \gamma V(s_{t+1})\right)\right] \tag{3.17}$$

Bir sonraki durumun değeri için en iyi tahminin, o durumda beklenen en iyi eylemin değeri, denklem (3.18) ile belirlenir.

$$V(s) = \max_{a \epsilon A} Q_{hedef}(s, a) \tag{3.18}$$

Şekil 3.8'de de görülebileceği gibi DYSA'da giriş, ortamın çoklu ölçüm değerlerinden oluşan o anki durumudur. DYSA'da çıkış, her olası eylemle ilişkilendirilen Q değerleridir. Yani DYSA, mevcut durumu kullanarak eylemlerin Q-değerlerini tahmin etmektedir.

Q-değerlerinin tahmininde, DYSA kullanımıyla beraber, farklı kararsızlıklar veya en uygun politikaya yakınsama sorunları ortaya çıkabilir .

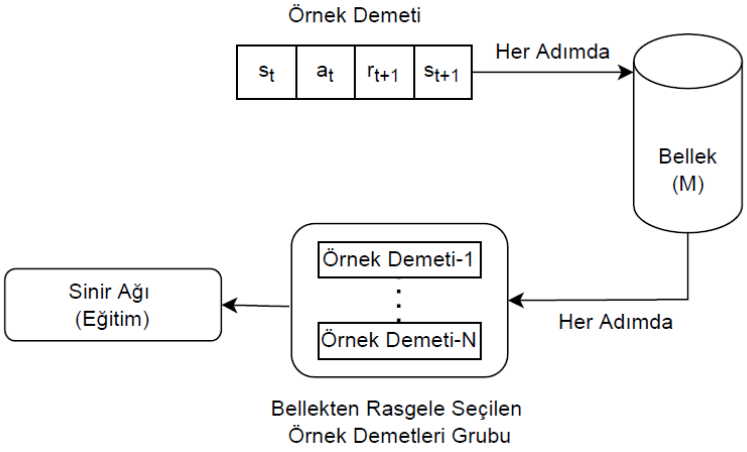
Denetimli öğrenme algoritmalarının eğitiminde kullanılan veri setindeki örnekler bir birinden bağımsız ve homojen şekilde dağıtıldığını varsayar. Ancak PÖ'de, bir sonraki durum-eylem çifti örneği (s_{t+1},a_{t+1}) büyük ölçüde önceki çifte (s_t,a_t) bağlıdır. Ayrıca, Q_t her zaman adımında yinelemeli olarak güncellendiğinden, örneğin (s_t,a_t,r_t,s_{t+1}) örnekleme dağılımı da homojen değildir.

Q-değerlerinin tahmininde, DYSA kullanımıyla ortaya çıkan bir diğer zorluk ise Q fonksiyonunu güncellemek için yeni örnekler kullanıldığında, ağ eski örneklerden daha önce öğrenilen görevleri unutabilir [48].

Yakınsama sorunlarını çözmek için farklı yöntemler önerilmiştir. En yaygın kullanılan ikisi, ilk olarak atari oyunları üzerinde uygulanan DQA için önerilen deneyim tekrarı ve hedef ağlardır [49].

3.3.6.1 Deneyim tekrarı

PÖ'de, durum-eylem çifti örneği (s_{t+1},a_{t+1}) büyük ölçüde bir önceki durum-eylem çifti (s_t,a_t) ile ilişkilidir. Deneyim tekrarı, örneklerin arasındaki ilişkiyi önlemek için kullanılan bir yöntemdir. Deneyim tekrarında örnekler, (s_t,a_t,r_t,s_{t+1}) oluşur; burada s_t mevcut durumdur, a_t seçilen eylemdir, r_t, s durumunda a eylemi seçmenin ödülüdür ve s_{t+1}, s durumunda a eylemi seçmenin sonucunda oluşan durumdur. Genellikle, sinir ağı güncellemeleri her deneyimden hemen sonra gerçekleştirilir. Deneyim tekrarında ise deneyimler ilk olarak deneyim tekrarı hafızasında (M) depolanır daha sonra her eğitim adımında, bellekten rastgele seçilmiş bir örnek grubu alınır ve ağı eğitmek için bu grup kullanılır. Bu gruptaki örnek demetleri rastgele seçildiği için seçilen örnek demetleri artık doğrudan ilişkili olmamaktadır. Bu sayede daha iyi bir yakınsama sağlanmaktadır [49]. Deneyim tekrarı ile ilgili süreç Şekil 3.9'da gösterilmektedir.



Şekil 3.9 Deneyim tekrarı

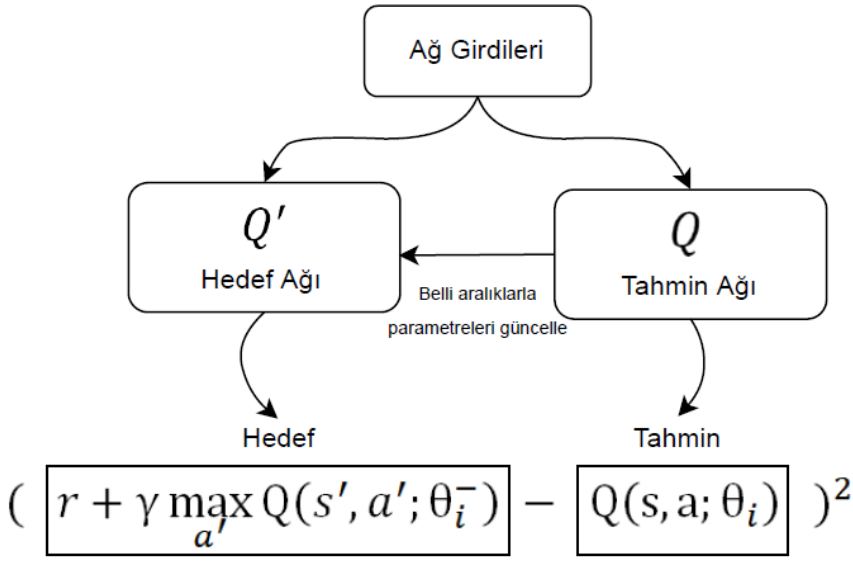
Deneyim tekrarını kullanmak için bir belleğe ihtiyaç duyulmaktadır. Sistemin kullanması gereken bellek boyutu önceden belirlenmelidir. Belleğin boyutu, eski örnek demetleri çıkarmadan önce kaç tane örnek demetinin saklanabileceğini göstermektedir. Genellikle, bellek dolduğunda, en eski örnek demeti kaldırılır. Ayrıca, eğitim için kullanılan örnek demetlerinin bulunduğu örnek grubun boyutu da önceden belirlenmelidir. Deneyim tekrarında, aynı örnek demetleri ile birkaç kez karşılaşılabilir. Örnek grubunun boyutu ne kadar büyükse ve örnek demeti ile ne

kadar sık karşılaşılırsa, bir örnek demetiyle o kadar sık karşılaşılır. Yeni deneyimler kazanmak hesaplama açısından maliyetli olduğundan, bir deneyimle birden çok kez karşılaşmak faydalıdır. Algoritma yinelemeli olarak çalıştığından ve Q-değeri yavaş bir şekilde güncellediğinden, bir örnekle birden çok kez karşılaşmak Q-değerine daha hızlı yakınsamayı sağlamaktadır. Deneyim tekrarı, özellikle nadir fakat değerli deneyimleri yeniden kullanma açısından da önemlidir.

Deneyim tekrarının bazı dezavantajları da vardır. Dezavantajlarından biri, büyük bellek kullanımı maliyetli olabilmektedir [50]. Diğer bir dezavantajı ise ortamın hızlı şekilde değiştiği durumlarda belirli durum-eylem çiftleri için eski ödüllerin artık doğru olmayabilmesidir. Ayrıca kritik öneme sahip deneyimler ile normal deneyimleri ayırt edebilecek bir yapı içermemektedir.

3.3.6.2 Hedef ağ

Hareketli hedefler probleminden kaçınmak için hedef ağ adı verilen bir yaklaşım uygulanabilir [49]. Bu yöntemde, farklı θ parametrelerine sahip iki ayrı Q ağı tutulmaktadır: bir tahmin ağı $Q(s,a,\theta)$ ve bir hedef ağı $Q'(s,a,\theta^T)$. Tahmin Q ağı θ parametresini güncellemek için deneyimlerini kullanmaktadır. Hedef ağ, hedef Q değerlerini tahmin etmek için kullanılmaktadır. Hedef Q ağı, donma aralığı olarak adlandırılan birkaç yineleme için parametreler güncellemeden tutulur. Donma aralığı sona erdiğinde tahmin ağının parametreleri hedef ağa kopyalayarak senkronize edilir. Bu sayede hedef Q ağının parametreleri, her zaman adımı yerine donma aralığının sonunda yalnızca bir kez güncellenmektedir. Yaklaşımın şeması Şekil 3.10'da gösterilmektedir. Mevcut Q değerleri ve hedef Q değerleri artık aynı anda değişmediğinden, ağın kararları daha istikrarlı hale gelmektedir [51].



Şekil 3.10 Hedef ağın güncellenmesi

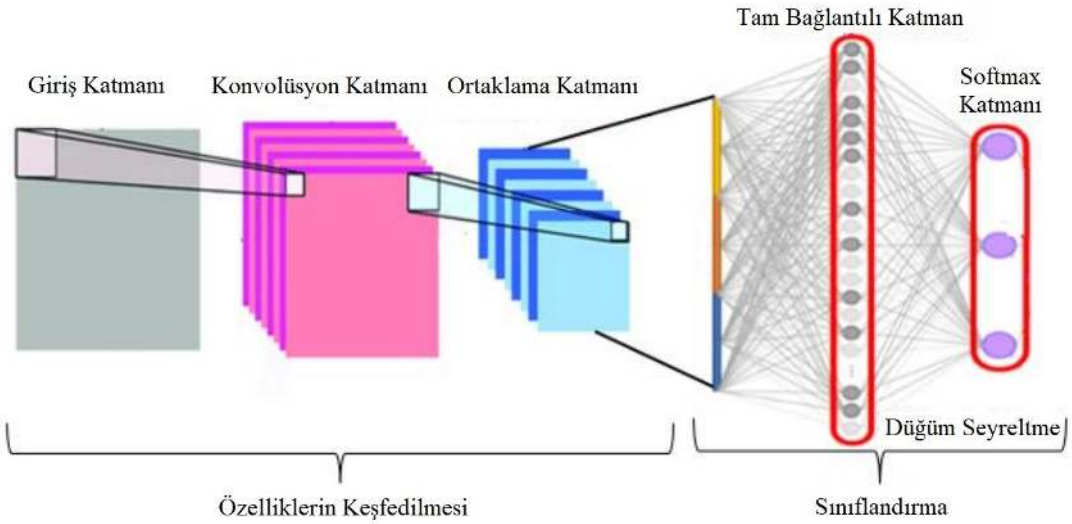
3.4 Evrişimli Sinir Ağları

Derin öğrenme ile görüntü sınıflandırmanın temelini ESA'lar oluşturmaktadır [52]. ESA'lar etkinliğini, 2012 yılında yapılan ImageNet Large Scale Visual Recognition Challenge (ILSVRC) yarışmasında kanıtladı [53]. Bu başarıdan sonra ESA ile yapılan çalışmalar ivme kazandı [54-58].

ESA'lar ile yapılan çalışmalar, ILSVRC yarışmasından çok daha öncesine dayanmasına rağmen ESA'ların bu yarışmadan sonra dikkat çekici başarılar elde etmesinin üç temel sebebi;

- Gelişen teknolojiyle beraber bilgisayar ekran kartlarının hesaplama güçlerinin katlanarak artması,
- Büyük miktarda verinin toplanabilmesi ve bu verilere kolaylıkla ulaşılabilmesi,
- Çok sayıda ESA modelinin ortaya çıkması olarak özetlenebilir.

ESA mimarileri, geleneksel görüntü sınıflandırma yöntemlerinden farklı olarak ön işlem, özellik çıkarımı ve öznitelik seçimini işlemlerini kendi içinde otomatik olarak gerçekleştirirler. ESA mimarilerinin genel yapısı Şekil 3.11'de görülebileceği gibi özelliklerin keşfedilmesi ve sınıflandırma olarak iki bölümden oluşmaktadır. Her bir bölüm ise kendi içinde katmanlardan oluşmaktadır.



Şekil 3.11 ESA mimarisinin genel yapısı [33]

3.4.1 ESA'nın katmanları

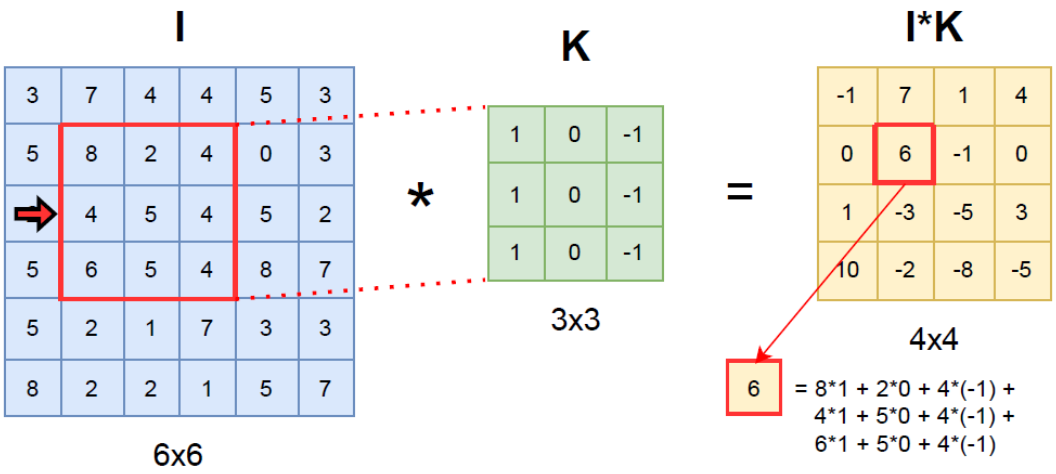
3.4.1.1 Giriş katmanı

ESA modelinin ilk katmanı olan giriş katmanında, belli bir boyuttaki görüntü verileri herhangi bir ön işleme tabi tutulmadan ham haliyle ağa verilmektedir. Ağda kullanılan görüntü verileri RGB ya da gri ölçekli olabilmektedir. Bu katmanda kullanılan görüntünün boyutu, tasarlanan ESA modelinin başarısını doğrudan etkilemektedir. Giriş görüntü boyutunun büyük seçilmesi, bellek ihtiyacının ve eğitim süresinin artmasına sebep olmaktadır. Giriş görüntü boyutunun küçük seçilmesi ise bellek ihtiyacını azaltmakta ve eğitim süresini kısaltmaktadır ancak gereğinden fazla küçük boyutta seçilen görüntü ağın performansını düşürebilmektedir. Giriş görüntüsünün boyutu seçilirken donanımsal hesaplama maliyeti ve ağın performansı göz önünde bulundurulmalıdır [33].

3.4.1.2 Evrişim katmanı

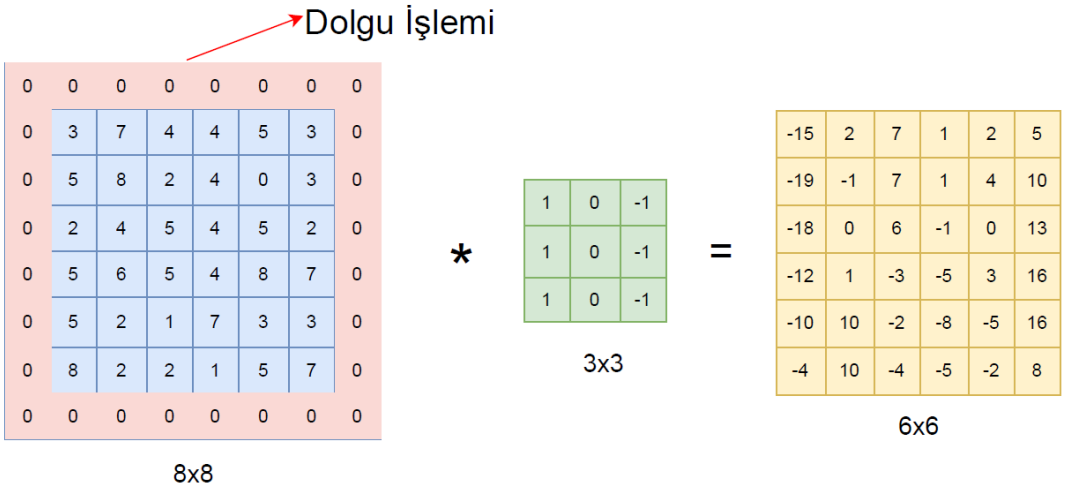
Evrişim katmanı ESA'nın temelini oluşturmaktadır. Bu katman, giriş görüntüsünün düşük ve yüksek seviyeli özelliklerini çıkarmak için görüntüye farklı filtrelerin uygulandığı katmandır. Filtreler genellikle 2x2, 3x3, 5x5 gibi boyutlardan oluşmaktadır. Filtre, görüntünün en üst sol köşe pozisyonundan başlayarak bütün görüntü tarayacak şekilde kaydırılarak görüntünün tamamına uygulanmaktadır. Bu katmanın performansını, filtre boyutu, filtre sayısı ve adım aralığı parametreleri belirlemektedir. Örnek bir evrişim işlemi Şekil 3.12'de ile gösterilmektedir. Burada

I, iki boyutlu görüntüyü, K, h × w boyuttaki filtreyi ve I ∗ K ise evrişim işlemini göstermektedir.



Şekil 3.12 Evrişim işlemi

Ancak Şekil 3.12'de de görülebileceği gibi filtre boyutu ve adım aralığına bağlı olarak evrişim işleminden sonra elde edilen çıkış görüntüsünün boyutu giriş görüntüsünden daha küçük olmaktadır. Her bir evrişim katmanından sonra elde edilen görüntünün boyutu azalmasından dolayı orijinal görüntülerden elde edilen özniteliklerin kaybolmasına yol açmaktadır. Bu problemin çözümü için evrişim işleminden önce görüntüye dolgu işlemi uygulanmaktadır. Dolgu işlemi, görüntünün boyut kaybına uğramaması için yeterli sayıda sıfır ile bir çerçeve olacak şekilde doldurulmasıdır. Şekil 3.12'de gösterilen giriş görüntüsünün boyut kaybına uğramaması için yapılan dolgu işlemi Şekil 3.13'te gösterilmektedir.



Şekil 3.13 Dolgu işlemi ile görüntü boyutunun korunması

Giriş görüntüsüne uygulanan evrişim işleminden sonra çıkış görüntüsünün boyutu denklem (3.19) ile hesaplanır.

$$O = \frac{(I-K)+2P}{S} + 1 \tag{3.19}$$

Denklemde;

O: Çıkış görüntüsünün boyutu,

I: Giriş görüntüsünün boyutu,

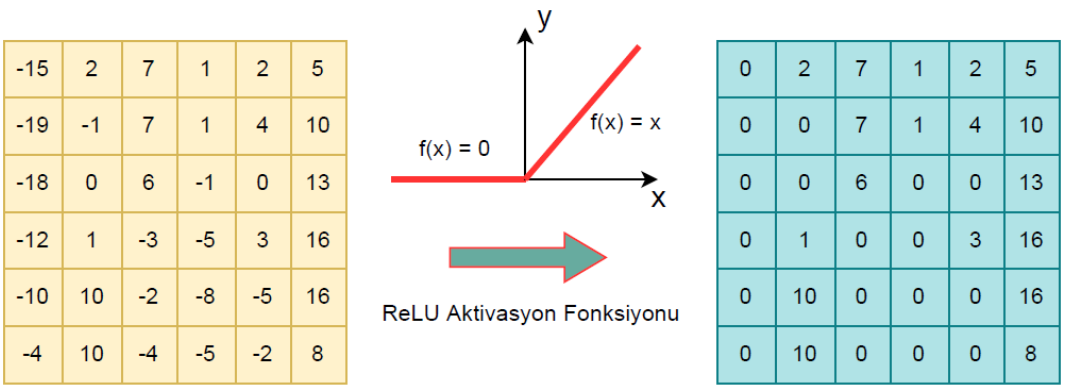
K: Filtre boyutu,

S: Adım aralığı,

P: Dolgu işlemini göstermektedir.

3.4.1.3 Aktivasyon katmanı

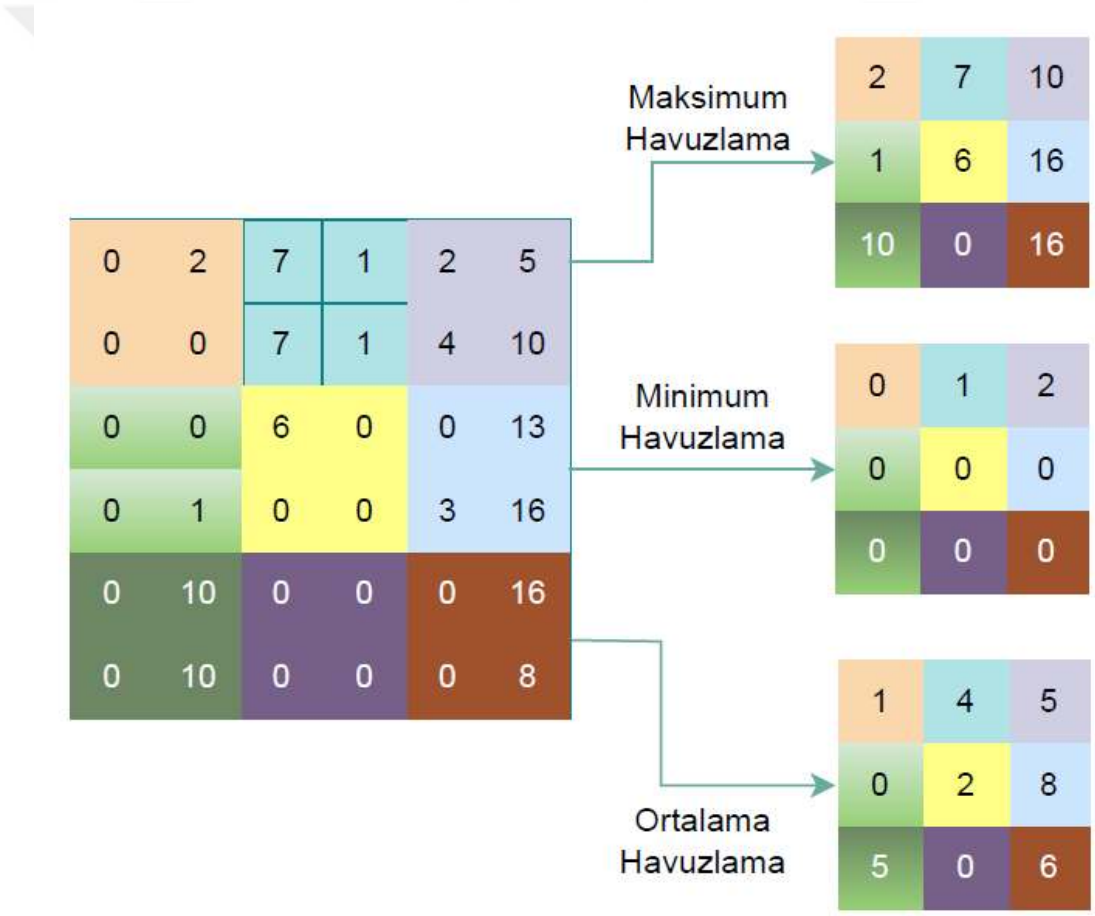
Evrişim katmanından sonra elde edilen görüntülerin daha kullanışlı hale gelmesi için genellikle elde edilen görüntülere aktivasyon fonksiyonu uygulanmaktadır. ESA'larda genellikle karmaşık özelliklerin öğrenilmesinde daha başarılı olan doğrusal olmayan aktivasyon fonksiyonları tercih edilmektedir. ESA'larda en çok tercih edilen aktivasyon fonksiyonlarından biri ReLU'dur. ReLU aktivasyon fonksiyonu sayesinde negatif veriler sıfıra dönüştürülürken, pozitif değerler değiştirilmeden doğrudan çıkışa aktarılır. ReLU aktivasyon fonksiyonu, basit yapısından dolayı çok hızlı hesaplanabilmektedir. Bu sayede çok fazla işlem gücüne ihtiyaç duyulmadan ağın doğrusal olmayan bir yapıya dönüştürülmesi sağlamaktadır. ReLU katmanının görüntü verisi üzerindeki etkisi Şekil 3.14'te gösterilmektedir.



Şekil 3.14 ReLU katmanının görüntü verisi üzerindeki etkisi

3.4.1.4 Havuzlama katmanı

Havuzlama katmanı, hesaplama yükünün düşürülmesi, görseldeki küçük pozisyon değişikliklerine karşı sistemin dayanıklı hale gelmesi ve modelin ezberlenmesinin önlenmesi gibi avantajları vardır. Ancak havuzlama işleminden sonra boyut azaltılmasıyla bilgilerin kaybolması ve sınıflandırma performansının düşmesi gibi sorunlarla karşılaşılabilmektedir. Havuzlama işlemi, belli bir çerçeve boyutu boyunca ve belli bir adım aralığıyla evrişim işlemine tabi tutulmuş görüntüye uygulanır. Maksimum, minimum, ortalama gibi havuzlama yöntemleri vardır. Örnek olarak 2x2 boyutlu çerçeve ve iki adım aralığı ile yapılan farklı havuzlama işlemleri Şekil 3.15'te gösterilmektedir.

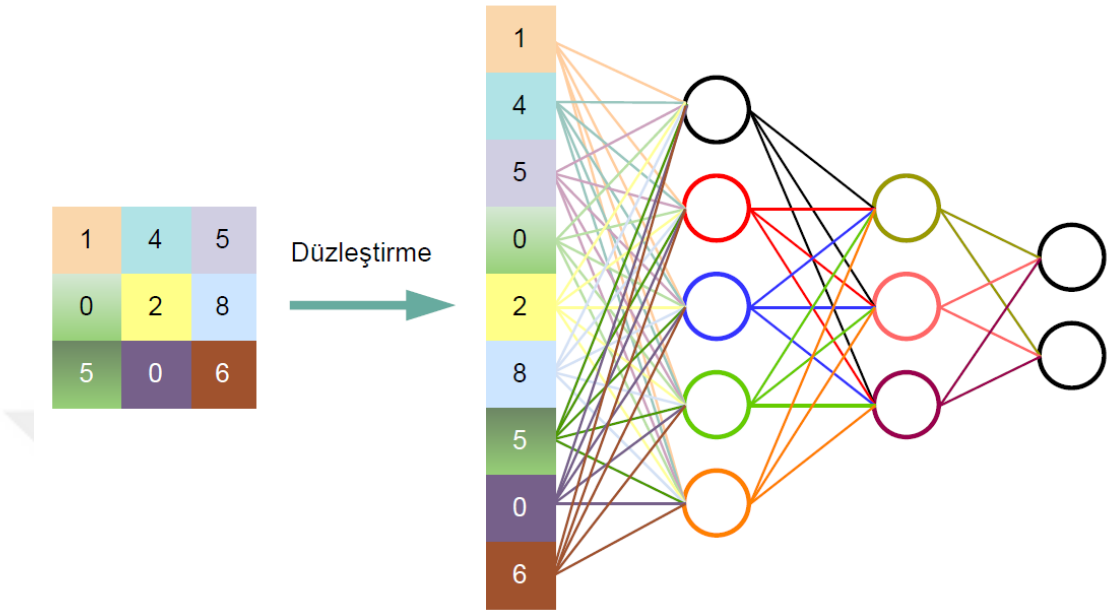


Şekil 3.15 Havuzlama işlemi

3.4.1.5 Tam bağlı katman

ESA'ların son katmanını genellikle tam bağlı katman oluşturmaktadır. Bu katman kendinden önceki katmanın bütün alanlarına bağlantılıdır. Tam bağlı katmandan önceki katmanda veriler matris formundadır. Matris formundaki veriler Şekil 3.16'da

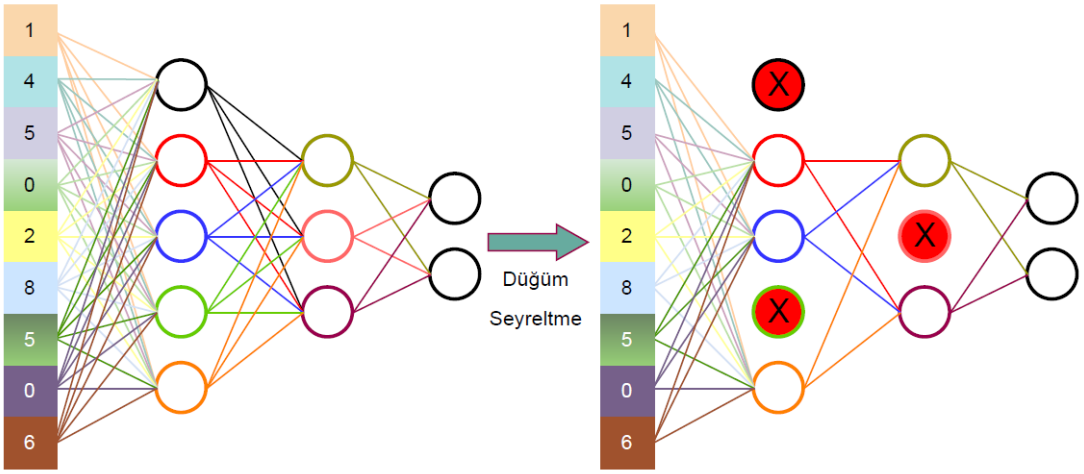
gösterildiği gibi düzleştirilerek vektör formuna getirilmekte ve sınıflandırmanın gerçekleşebilmesi için sinir ağına bağlanmaktadır.



Şekil 3.16 Tam bağlı katman

3.4.1.6 Düğüm seyreltme katmanı

Düğüm seyreltme, modelin veriyi ezberlemesini önlemek için kullanılmaktadır. Düğüm seyreltme işlemi, ağın bazı düğümlerinin ve bağlantılarının rastgele kaldırılması ile Şekil 3.17'de gösterildiği gibi gerçekleşmektedir. Ayrıca düğüm seyreltme işleminden sonra parametre sayısı azalacağından modelin eğitim süresini de kısaltmaktadır.

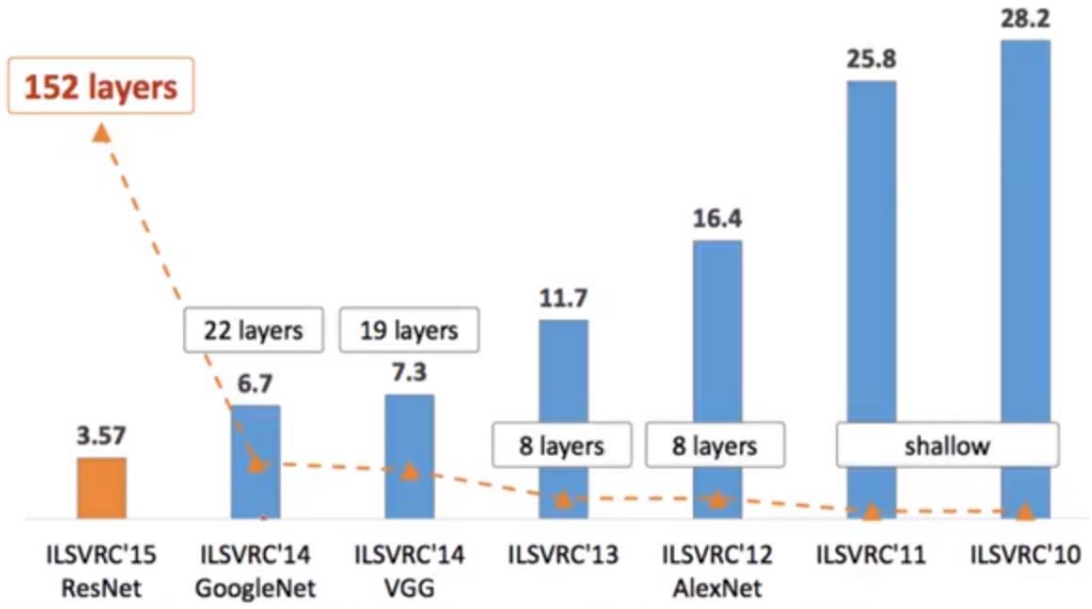


Şekil 3.17 Düğüm seyreltme işlemi

3.4.2 ESA modelleri

Nesne algılamada, makine öğrenimi ya da derin öğrenme tabanlı yaklaşımlar kullanılmaktadır. Geleneksel makine öğrenimi tabanlı yöntemlerde, bir nesneye ait piksellerden oluşan alanı tespit etmek için görüntünün histogram, kenar ve köşe gibi özellikleri görüntü işleme teknikleri ile belirlenir. Önceden aynı özellikler için etiketli veriler ile eğitilen regresyon modeli kullanılarak görüntüdeki nesne konumuyla birlikte tespit edilebilmektedir. Derin öğrenme tabanlı yöntemler ise veri ön işleme sürecine ihtiyaç duymayan ESA'ları kullanmaktadır.

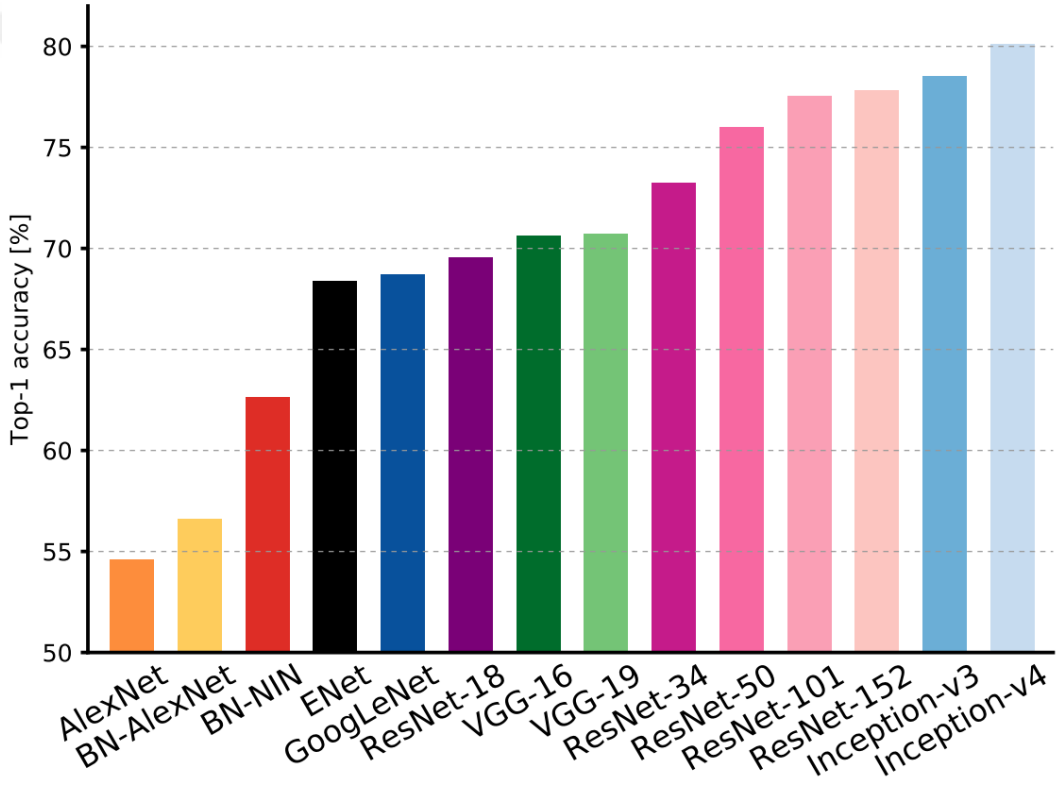
Zaman içinde birçok ESA modeli geliştirilmiştir. ILSVRC yarışması ESA'ların nesne algılamada ne kadar ilerlediğini görmek açısından önemlidir. Şekil 3.18'de yıllara göre ILSVRC yarışmasında birinci olan ESA modellerinin top-5 hata oranları gösterilmektedir. ILSVRC yarışmasında modellerin eğitimi için 1,2 milyon görüntü ve 1000 sınıftan oluşan veri seti kullanılmaktadır. Modeller 1000 farklı nesne için 0-1 aralığında olasılık değeri üretmektedir ve tüm değerlerin toplamı bire eşittir. Top-5 hata ise tahmin edilen nesnenin en yüksek beş olasılığa sahip nesnenin içinde olmadığını göstermektedir. Zaman içinde top-5 hata oranının %3,57'ye kadar düştüğü görülmektedir. Aynı veri seti için insan gözünün top-5 hata oranı % 5,1 olduğu düşünüldüğünde ESA modellerinin nesne tanımadaki başarısının insan yeteneğini aştığı söylenebilir [59].



Şekil 3.18 Yıllar içinde ILSVRC yarışmasında birinci olan modellerin top-5 hata oranı [59]

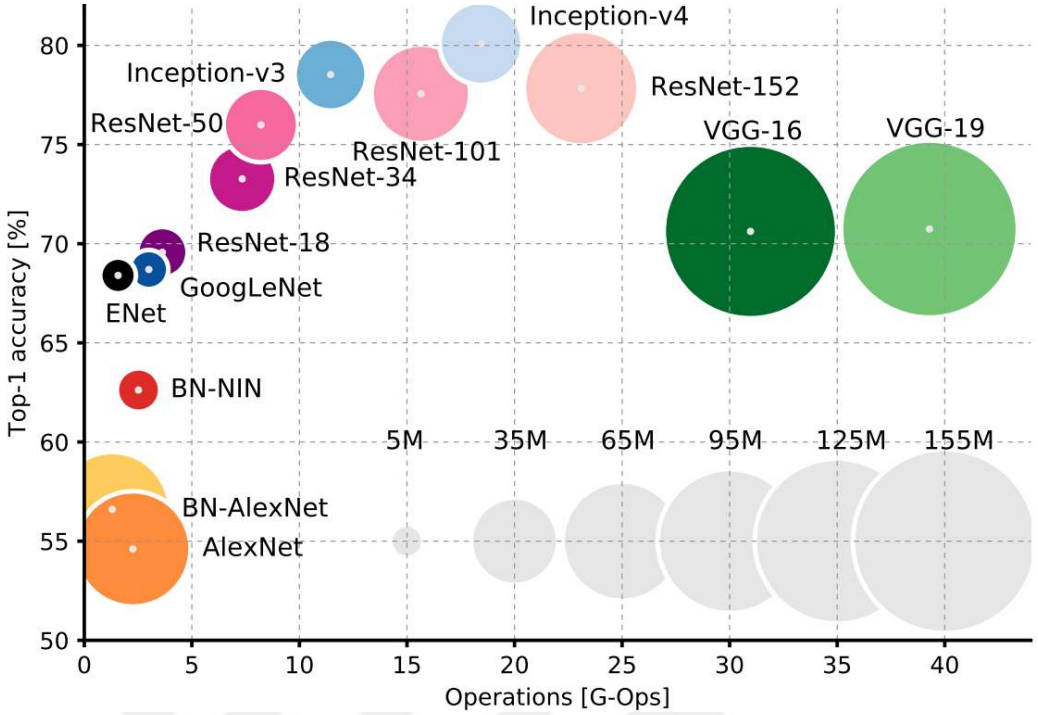
ESA sınıflandırma modellerini, tek aşamalı ve çok aşamalı nesne tespit modelleri olarak ayırabiliriz. SSD, ResNet, YOLO gibi tek aşamalı nesne tespit modellerinde nesnenin sınırlarını belirleyen kutu tahmini ve sınıflandırma tek aşamada yapılırken, RCNN, Fast RCNN, Faster RCNN gibi çok aşamalı nesne tespit modellerinde ise ilk aşamada sınırlayıcı kutular belirlenmekte, ikinci aşamada nihai sınıflandırma yapılmaktadır.

ESA modelinin seçiminde doğruluk oranı önemli bir parametredir. Şekil 3.19'da bazı ESA modellerine ait Top-1 doğruluk oranı verilmektedir. Top-1 doğruluk oranı, modelin en yüksek olasılığa sahip tahmininin doğruluk oranını göstermektedir.



Şekil 3.19 Bazı ESA modellerine ait top-1 doğruluk oranı [60]

Ayrıca ESA modelinin seçiminde, modelin boyutu, tahmin ve eğitim hızı yapılacak çalışmaya göre önem arz etmektedir. Şekil 3.20'de bazı ESA modelleri daha kapsamlı şekilde karşılaştırılmaktadır. Grafiğin y-ekseni, modelin Top-1 doğruluk oranını, grafiğin x-ekseni ise modelin tahmin için gerekli işlem yükünü göstermektedir. Ayrıca model parametre sayılarının $5x10^6$-$155x10^6$ aralığında değiştiği görülmektedir.



Şekil 3.20 Bazı ESA modellerinin karşılaştırılması [60]

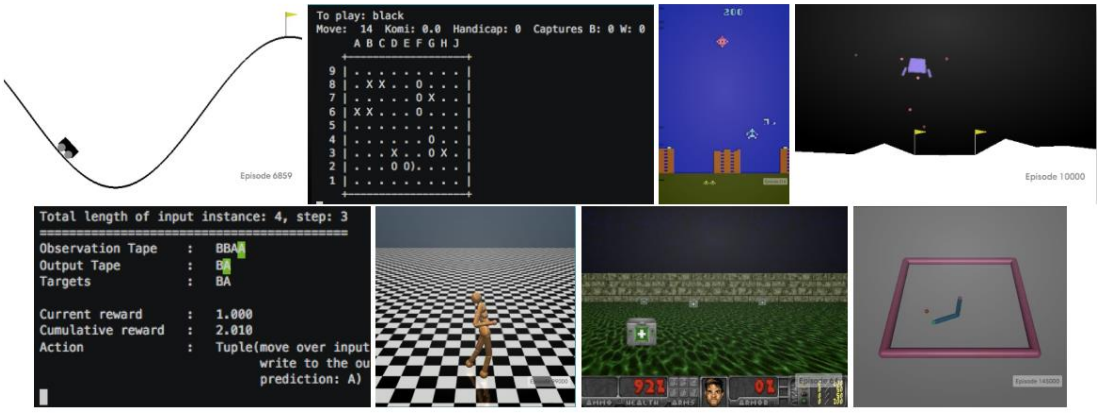
Bu tez kapsamında yapılan uygulamada EfficientDet ESA modeli kullanılmıştır. EfficientDet ESA modeline ait açıklamalar bir sonraki bölümde verilmektedir.

4. TEZİN UYGULAMA AŞAMASI

4.1 Sanal Ortamın Yapısı

DPÖ algoritmalarının bir veri setine ihtiyaç duymaması ve insan uzmanların performansını aşabilecek potansiyele sahip olması yapay zeka alanında önemli gelişmeler olmasına yol açmaktadır. Ancak bir DPÖ ajanı eğitilirken ortam ile çok sayıda etkileşime girmesi gerektiğinden, doğrudan gerçek ortamda eğitilmesi, uzun eğitim süresi, yüksek maliyet ve oluşabilecek maddi hasarlardan dolayı zordur. Bu sebepten dolayı gerçek dünya uygulamaları için DPÖ ajanlarının eğitiminin büyük bir kısmı ya da tamamı sanal ortamlarda gerçekleştirilmektedir.

Bu tez kapsamında yapılan uygulamadaki PÖ ajanının eğitimi için OpenAI Gym alt yapısı kullanılmaktadır. OpenAI Gym, pekiştirmeli öğrenme araştırmaları için hazırlanmış bir araç setidir. PÖ algoritmaların için ortak bir arayüz oluşturan OpenAI Gym'de birçok farklı görev ve ortam bulunmaktadır [61]. OpenAI Gym'de bulunan bazı ortamlar Şekil 4.1'de gösterilmektedir. Bu ortamlar kısmen gözlemlenebilir bir Markov karar süreci olarak düşünülebilir. OpenAI Gym'de bulunan tüm ortamlar için PÖ ajanı her zaman adımında bir hareket gerçekleştirdiğinde, ajanın ortamdaki yeni durum bilgisi, yapılan hareket sonrası ajanın aldığı ödül bilgisi ve bölümün tamamlandığına ya da tamamlanmadığına dair bilgi alınmaktadır.



Şekil 4.1 OpenAI Gym'de bulunan bazı ortamlar [61]

Sanal uygulama ortamı olarak OpenAI Gym platformlarından biri olan Minimalistic Gridworld Environment (MiniGrid) kullanılmaktadır. MiniGrid Environment

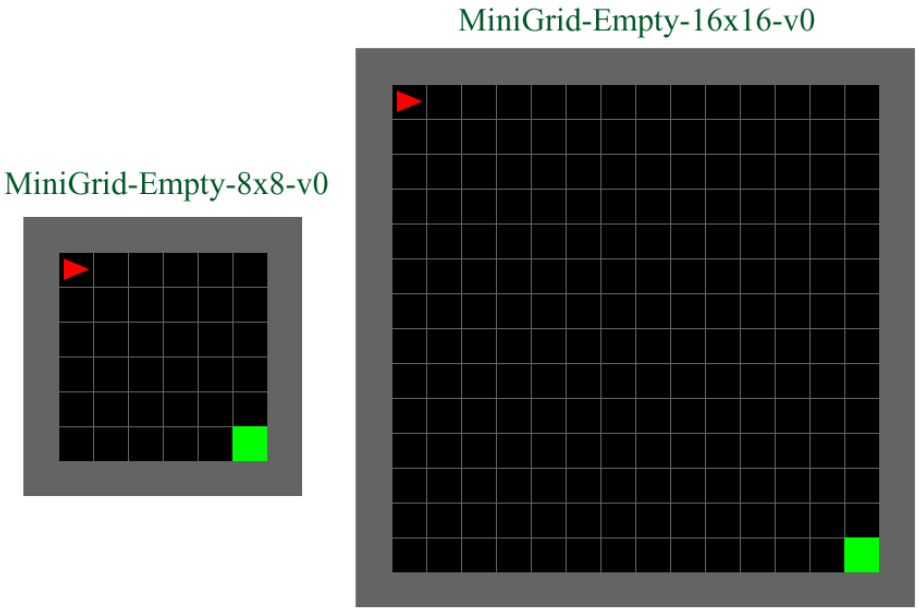
platformunun altında artan zorluk seviyelerine sahip çok sayıda ortam mevcuttur [62].

Bu ortamlar için bazı görevler;

- Farklı boyutlarda olan boş bir odanın bulunduğu ortamda ajanın hedefe ulaşması,
- Birbirine bağlı dört odanın bulunduğu ortamda ajanın hedefe ulaşması,
- Kilitli kapılar ve kapılara ait anahtarların olduğu çok sayıda odanın bulunduğu ortamda ajanın hedefe ulaşması,
- Dil talimatlarıyla ajanın istenen görevi yapması,
- Sabit engellerin bulunduğu ortamda ajanın hedefe ulaşması,
- Dinamik engellerin bulunduğu ortamda ajanın hedefe ulaşması şeklinde sıralanabilir.

MiniGrid Environment platformunda doğal dil işleme [63-65] ve ödül sinyallerinin seyrek olduğu ortamlar için pekiştirmeli öğrenme ile ilgili çalışmalar mevcuttur [66-73]. Ancak yapılan çalışmalar sanal çevre ile sınırlandırılmıştır. Bu çalışmada ise sanal ortamda eğitilen ajanı kullanarak gerçek ortam uygulaması yapılmasına odaklanılmaktadır.

Uygulamada Şekil 4.2 ‘de gösterilen ortamlar kullanılmaktadır. Her iki ortam da boş bir odayı temsil etmektedir ve ajanın amacı, seyrek ödül sağlayan ortamda yeşil renkli hedef parseline ulaşmaktır. İki ortam arasındaki tek fark ortamı oluşturan parsel sayısıdır. Bundan dolayı sanal ortam analizinin yapıldığı bu bölümde MiniGrid-Empty-8x8-v0 ortamının yapısı açıklanmaktadır.



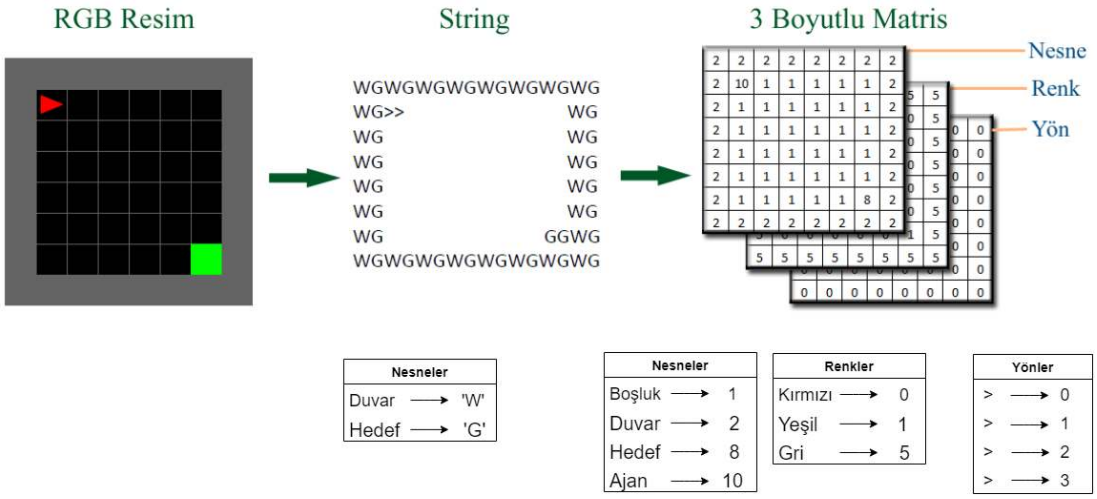
Şekil 4.2 Uygulamada kullanılan ortamlar

4.1.1 Durumlar

MiniGrid-Empty-8x8-v0 ortamı 8x8 parselden oluşmaktadır. Sanal ortamın durumu 3 farklı şekilde alınabilmektedir. Bunlar Şekil 4.3'te gösterildiği gibi;

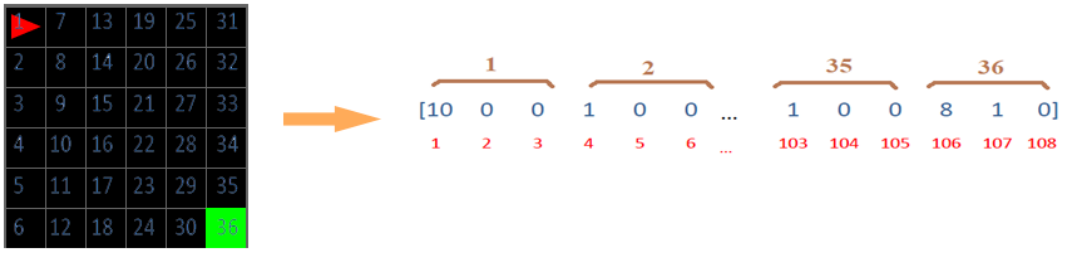
- Varsayılan olarak 32x32 piksel boyutundaki parsellerin birleşiminden oluşan RGB resim,
- Her bir parselin 2 karakter ile kodlandığı string yapı,
- Her bir parselin sayısal değerlerle kodlandığı 3-boyutlu matris şeklinde olabilmektedir.

Uygulamada sanal ortamın durumu 3 boyutlu matris olarak alınmaktadır. Çünkü RGB resmine göre parametre sayısı çok daha azdır ve parametreler sayısal değerlerden oluştuğundan dolayı, bunları YSA kullanımına uygun hale getirmek kolaydır. YSA'da girilen veri sayısı arttıkça ağ karmaşıklaşmaktadır ve ağı daha basit hale getirmek için hiçbir zaman durumunu değişmeyen dış duvar kaldırılmaktadır. 3 boyutlu matris yapısına sahip ortamın durum parametreleri, YSA'da kullanılabilmesi için Şekil 4.4'teki gibi düzleştirilmektedir.



Şekil 4.3 Sanal ortam durumunun üç farklı şekilde alınması

Şekil 4.4'te gösterildiği gibi her parsel üç parametre (nesne, renk, yön) almaktadır. Bu parametreler şekildeki gibi art arda eklenerek ortamın durumunu gösteren ve 108 parametreden oluşan bir dizi elde edilmektedir. MiniGrid-Empty-16x16-v0 ortamı için oluşturulan dizinin parametre sayısı ise 588'dir.



Şekil 4.4 Sanal ortam durum parametrelerinin düzleştirilmesi

4.1.2 Hareketler

Uygulamada ajan 3 farklı hareket yapabilmektedir. Ajanın yaptığı her bir hareket için bir sayısal değer tanımlıdır. Bunlar;

- 90 derece sola dön→0
- 90 derece sağa dön→1
- Bir birim ileri git→2'dir.

4.1.3 Ödül fonksiyonu

Uygulamada ajanın bir bölümde alabileceği toplam ödül değeri sıfır ile bir aralığındadır. Ajanın bölümü tamamlayabilmesi için ya ajanın hedefine varması ya da maksimum sayıda adım atması gerekmektedir. Uygulama için maksimum adım

sayısı 50 olarak belirlenmiştir. Ajanın hedefine varması durumunda ödül fonksiyonu denklem (4.1)'de gösterildiği gibidir.

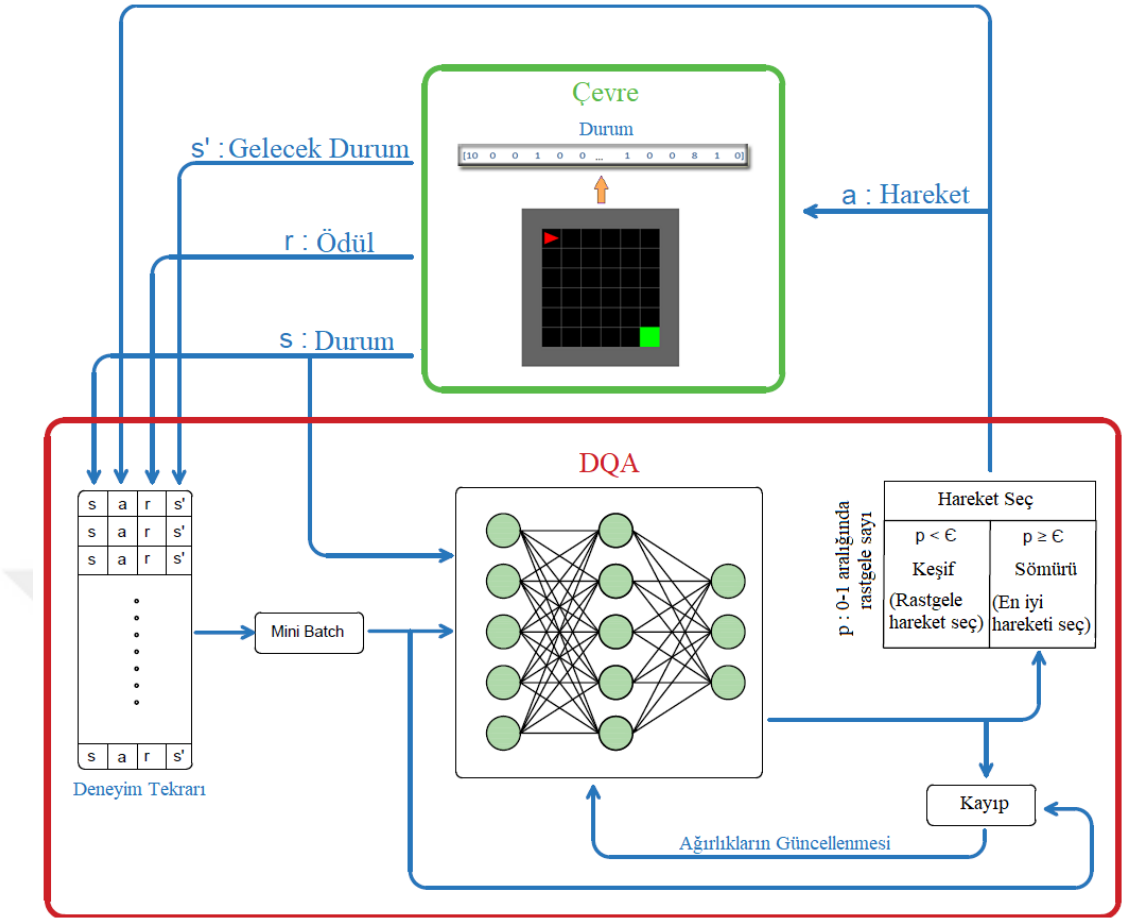
$$R = 1 - 0{,}9 * \frac{Adım\ sayısı}{Maksimum\ adım\ sayısı} \tag{4.1}$$

Ödül fonksiyonuna göre adım sayısı arttıkça kazanılan ödül azalmaktadır, ajan ödülünü maksimize edebilmesi için en kısa yolu bulması gerekmektedir. Eğer ajan maksimum sayıda adım atarsa bölüm sonlanmakta ve ajan başarısız olduğundan dolayı sıfır puan ödül almaktadır.

4.2 Derin Q-Ağı ve Hiyerarşik Başlangıç Pozisyonlu Derin Q-Ağı

Derin öğrenmedeki ilerleme ile MKS'nin derin sinir ağları ile çözülmesi mümkün olmaktadır [74]. Böyle bir yöntem olan DQA, bazı alanlarda insan seviyesinde başarılı olmaktadır [13]. DQA denetimli öğrenme ile pekiştirmeli öğrenme tekniklerini birleştiren bir derin pekiştirmeli öğrenme algoritmasıdır [13]. Oyunlar [49], robotik [75] ve doğal dil işleme [76] gibi alanlarda sıklıkla kullanılan DQA'nın yapısı Şekil 4.5'te gösterilmektedir.

DQA algoritmasında ajan, eğitimin başlangıç aşamasında rastgele hareketler yaparak ortam hakkındaki tecrübelerini deneyi tekrarı hafızasında kayıt altına almaktadır. Bu kayıt alanına, bir durumda (s) yapılan hareketin (a) ne kadar ödül (r) getirdiği ve hareket sonrasındaki yeni durum (s') bilgileri kaydedilir. Bu şekilde elde edilen tecrübeler yeterli sayıya ulaştığında mini batch'in boyutu kadar tecrübe deneyi tekrarı hafızasından rastgele seçilerek Q-ağının eğitiminde kullanılmaktadır. Böylece ağ, yerel minimumlardan etkilenmemektedir. Ayrıca DQA algoritmasında ajan yapacağı hareketi Epsilon-Greedy'yi ($\in$) kullanarak belirlemektedir. $\in$, 0 ile 1 arasında değerler alabilmektedir. $\in$'nun 1'e yakın olması ajanın keşfe odaklandığını, 0'a yakın olması ise tecrübeleri ile belirlediği en iyi hareketi yapmaya odaklandığını göstermektedir. Bu sebepten dolayı eğitimin başlangıç aşamasında $\in$ değeri 1'e yakın seçilerek ajanın ortamı keşfetmesi sağlanır. Bölümler ilerledikçe $\in$ değeri oransal olarak azaltılır ve ajanın tecrübelerini kullanarak en iyi hareketi seçmesi sağlanmaktadır. DQA algoritması açıklanan yapısından dolayı ayrık ve küçük boyutlu ortamlar için oldukça kullanışlıdır.



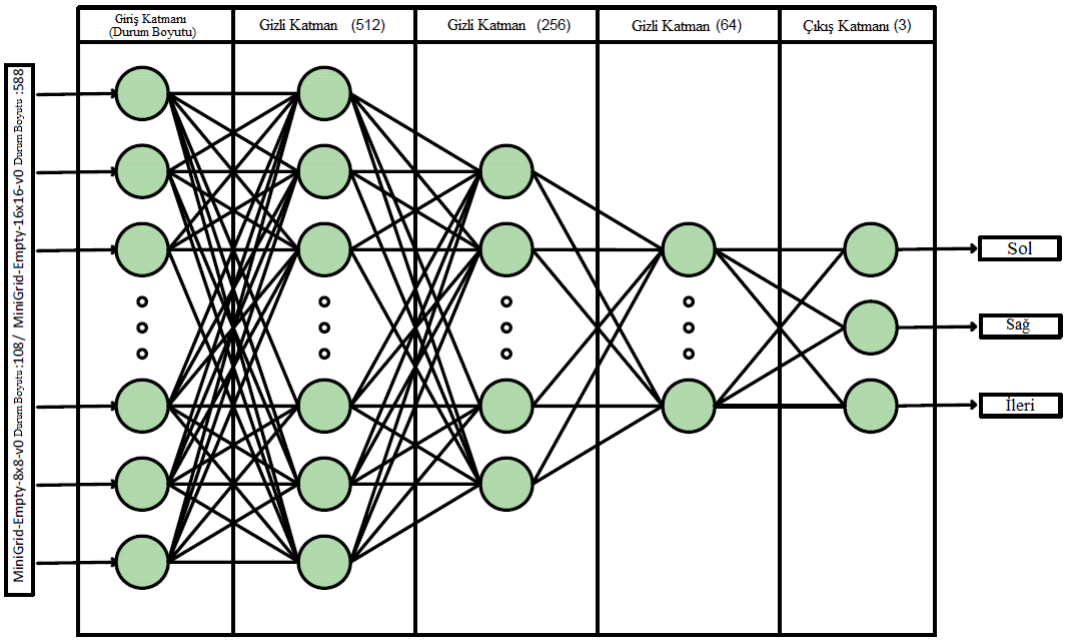
Şekil 4.5 DQA'nın yapısı

Yapılan uygulamada sanal ortam olarak MiniGrid-Empty-8x8-v0 ve MiniGrid-Empty-16x16-v0 ortamları kullanılmaktadır. Ajan belirtilen ortamlar için, DQA algoritması ve tez kapsamında önerilen hiyerarşik başlangıç pozisyonlu derin Q-ağı (HBP-DQA) algoritması ile ayrı ayrı eğitilmiş ve algoritma performansları incelenmiştir. Her iki algoritma için de Şekil 4.6'da gösterilen ağ mimarisi kullanılmaktadır.

Gerçek ortam uygulamasında mobil robotun başlangıç pozisyonu ve yönü ne olursa olsun, mobil robotun hedefine varması istenmektedir. Bundan dolayı DQA ile yapılan eğitimde ajanın pozisyonu ve yönü her yeni bölümde rastgele seçilmektedir.

HBP-DQA ile yapılan eğitimde ise ajanın hedefi rastgele hareketlerle bulma ihtimalinin yüksek olduğu bir bölge belirlenerek, eğitim bu bölgede başlatılmaktadır. Ajan, her yeni bölümde belirtilen bölgede kalma koşuluyla rastgele pozisyon ve yönde başlatılmaktadır. Bu bölge için ağ yeterince eğitildikten sonra eğitim için yeni

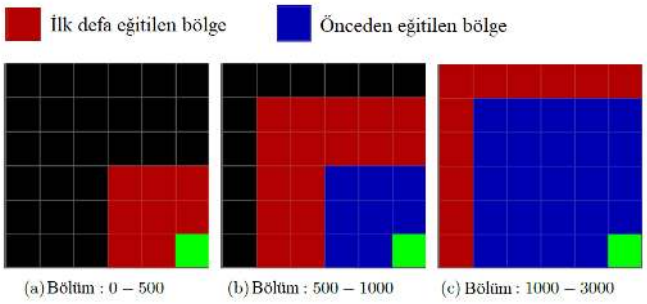
bir bölge belirlenmekte ve ∈ değeri büyütülerek ajanın yeniden keşif yapması sağlanmaktadır. Bu aşamada 2 bölge vardır; ilk defa eğitimin yapıldığı bölge ve daha önce eğitim yapılmış olan bölge. Eğitim esnasında her iki bölgede kullanılmaktadır. Her yeni bölümde, 2 bölümden biri dönüşümlü olarak seçilmekte ve ajan rastgele pozisyon ve yönde başlatılmaktadır. Bu şekilde ağın, eski bilgileri güncel tutularak yeni bölgelerde keşif yapılabilmektedir. Bu aşamada ağ yeterince eğitildikten sonra ilk defa eğitilen bölge daha önce eğitilmiş bölgenin içine dahil edilmektedir. ∈ değeri tekrar büyütülmekte ve yeni bir keşif bölgesi belirlenmektedir. Böylelikle ağ kademeli olarak eğitilmektedir. Bu aşamalar ağ, ortamın tamamını öğreninceye kadar devam etmektedir. HBP-DQA algoritmasının MiniGrid-Empty-8x8-v0 ortamında uygulanması Şekil 4.7'de gösterilmektedir.



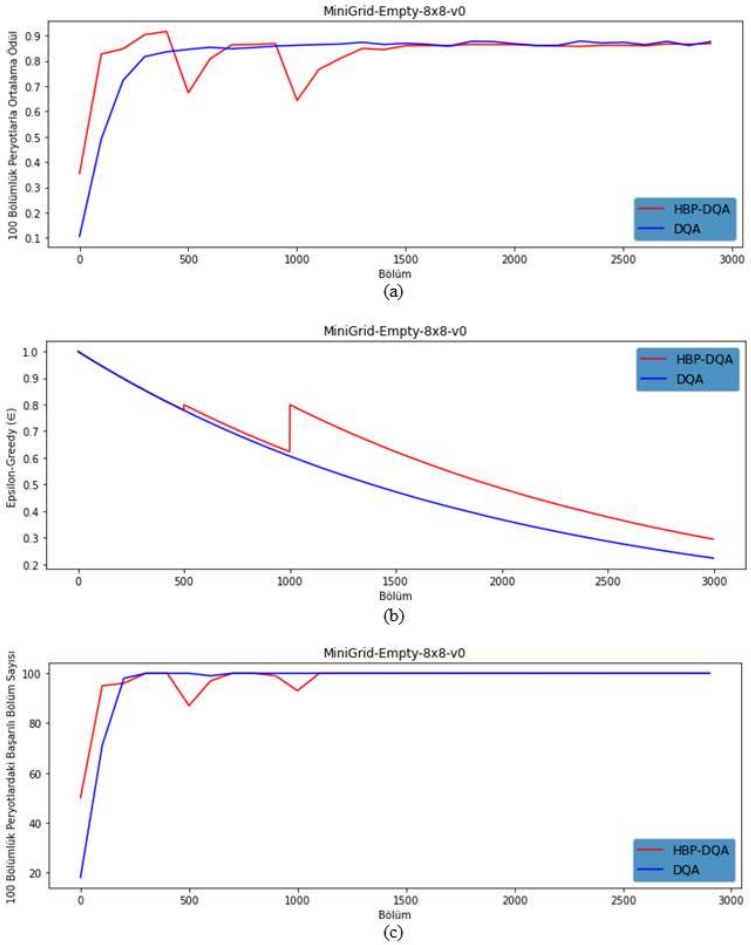
Şekil 4.6 Q-ağı mimarisi

MiniGrid-Empty-8x8-v0 ortamı için HBP-DQA ve DQA karşılaştırma grafikleri Şekil 4.8'de gösterilmektedir. HBP-DQA algoritması her yeni bölge tanımında ∈ değeri büyütülerek keşfe yoğunlaşması sağlanmaktadır. Bu esnada ajanın keşfe yoğunlaştığından, ajanın başarısız olduğu ya da düşük puan aldığı bölümler olmakta ve ortalama ödül miktarında düşüşler görülmektedir (Bkz. Şekil 4.8 (a) (b)). Ancak 1500'lü bölümlerden sonra Şekil 4.8 (a)'da görüldüğü gibi alınan ödüller benzerlik göstermektedir ve bu aşamadan sonra Şekil 4.8 (c)'de görüldüğü gibi hem HBP-

DQA hem de DQA algoritmalarıyla eğitilen ajan geri kalan tüm bölümlerde başarılı olmuştur.



Şekil 4.7 HBP-DQA algoritmasının MiniGrid-Empty-8x8-v0 ortamına uygulanması



Şekil 4.8 MiniGrid-Empty-8x8-v0 ortamı için HBP-DQA ve DQA karşılaştırılması (a) Bölüm / Ortalama ödül grafiği, (b) Bölüm / Epsilon-Greedy grafiği, (c) 100 bölümlük periyotlarda başarılı bölüm sayısı

DQA ve HBP-DQA'nın performans verileri Tablo 4.1'de gösterilmektedir. Her iki algoritmada da 3000 bölüm sonunda başarılı sonuçlar elde edilmiştir. HBP-DQA'nın

toplam adım sayısı DQA'ya göre kısmen az olsa da bariz bir avantaj sağlamamaktadır.

Tablo 4.1 DQA ve HBP-DQA Karşılaştırılması (MiniGrid-Empty-8x8-v0)

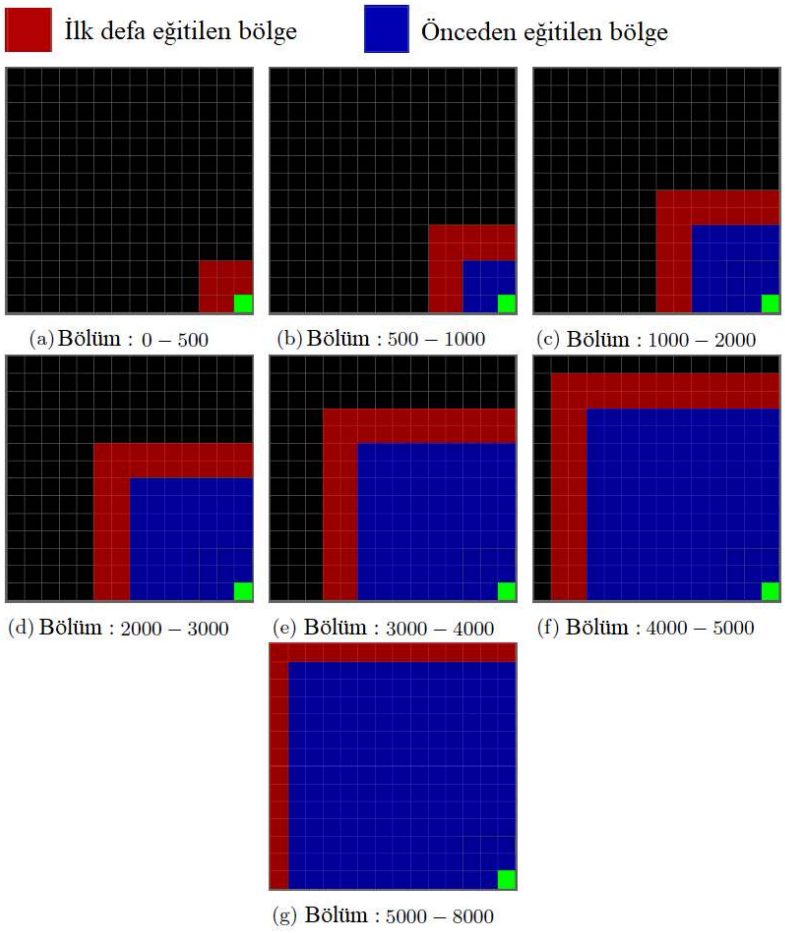
MiniGrid-Empty-8x8-v0						
	DQA	**Bölümler**	1-500	500-1K	1K-2K	2K-3K
		Toplam Adım Sayısı	10646	4127	7373	7305
		Başarılı Bölüm Sayısı	387	499	1000	1000
		Başarı Oranı	77,40%	99,80%	100%	100%
		Toplam Adım Sayısı				29451
		Her Bölüm için Ortalama Adım Sayısı				9,82
	HBP-DQA	**Bölümler**	1-500	500-1K	1K-2K	2K-3K
		Toplam Adım Sayısı	6083	5037	9847	7659
		Başarılı Bölüm Sayısı	441	483	993	1000
		Başarı Oranı	88,20%	96,60%	99,30%	100%
		Toplam Adım Sayısı				28626
		Her Bölüm için Ortalama Adım Sayısı				9,54

MiniGrid-Empty-8x8-v0 ortamı alan bakımından büyük olmadığından dolayı DQA algoritması ortamı öğrenmede başarılı olmuştur.

Algoritma performansları daha büyük alana sahip olan MiniGrid-Empty-16x16-v0 ortamı için de karşılaştırılmaktadır. HBP-DQA algoritmasının MiniGrid-Empty-16x16-v0 ortamında uygulanması Şekil 4.9'da gösterilmektedir.

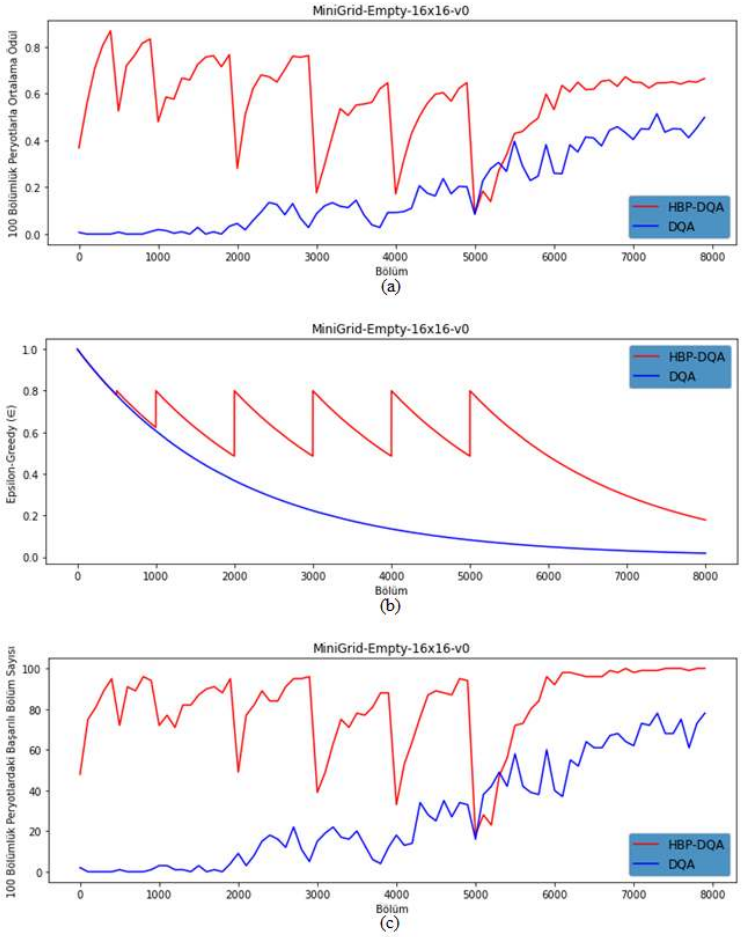
MiniGrid-Empty-16x16-v0 ortamı için DQA ve HBP-DQA'ya ait karşılaştırma grafikleri Şekil 4.10'da gösterilmektedir. Şekil 4.10 (c)'de görüldüğü gibi DQA algoritması ile eğitilen ajanın ilk 2000 bölümde başarılı olduğu bölüm sayısı oldukça azdır. Eğitimin ilerleyen aşamalarında başarı artma eğilimindedir ancak Şekil 4.10 (b)'de de görüldüğü gibi $\in$ değeri sıfıra yaklaşmaktadır. $\in$ sıfıra yaklaştıkça ajan yeni yollar keşfetmek yerine deneyimlerine güvenmekte ve eğitilen ağın verdiği sonuca göre hareket etmektedir. 8000 bölüm sonunda ise keşif yapma ihtimali sıfıra yaklaşmasına rağmen başarı oranı %70 civarındadır. HBP-DQA algoritmasında ise her yeni bölge tanımında $\in$ değeri büyütülerek keşfe yoğunlaşması sağlanmaktadır. Bu esnada ajanın keşfe yoğunlaştığından, ajanın başarısız olduğu ya da düşük puan aldığı bölümler olmakta ve ortalama ödül miktarında düşüşler görülmektedir (Bkz. Şekil 4.10 (a) (b)). Ancak her $\in$ güncellemesinden sonra ajan, ortamı kısa zamanda

öğrenmekte ve başarı oranı yükselmektedir. 6000'li bölümlerden sonra ödül aralığının daralma eğiliminde olduğu (Bkz. Şekil 4.10 (a)) yani ajanın kısa yollar keşfettiği ve başarı oranının %100'e yaklaştığı görülmektedir (Bkz. Şekil 4.10 (c)).



Şekil 4.9 HBP-DQA algoritmasının MiniGrid-Empty-16x16-v0 ortamına uygulanması

DQA ve HBP-DQA'nın performans verileri Tablo 4.2'de gösterilmektedir. Buna göre HBP-DQA'nın DQA'ya göre başarılı bölüm oranı oldukça yüksektir. DQA'nın 8000 bölüm sonundaki toplam adım sayısı HBP-DQA'dan yaklaşık %84 daha fazladır. Yani başarı oranı bakımından düşük olan DQA'nın eğitim süresi de HBP-DQA'ya göre oldukça uzundur. Ayrıca 8000 bölüm sonunda DQA'da Epsilon-Greedy değeri 0,018'e kadar düşmektedir, HBP-DQA için bu değer 0,178'dir. Yani HBP-DQA'nın daha kısa yolları keşfetme ihtimali yüksektir.



Şekil 4.10 MiniGrid-Empty-16x16-v0 ortamı için HBP-DQA ve DQA karşılaştırılması (a) Bölüm / Ortalama ödül grafiği, (b) Bölüm / Epsilon-Greedy grafiği, (c) 100 bölümlük periyotlarda başarılı bölüm sayısı

Tablo 4.2 DQA ve HBP-DQA Karşılaştırılması (MiniGrid-Empty-16x16-v0)

MiniGrid-Empty-16x16-v0	**DQA**	**Bölümler**	1-500	500-1K	1K-2K	2K-3K	3K-4K	4K-5K	5K-6K	6K-7K	7K-8K
		Toplam Adım Sayısı	24973	24912	49430	46311	45486	42255	37297	32136	28881
		Başarılı Bölüm Sayısı	2	2	16	119	144	261	424	569	708
		Başarı Oranı	0,40%	0,40%	1,60%	11,90%	14,40%	26,10%	42,40%	56,90%	70,80%
		Toplam Adım Sayısı			331681						
		Her Bölüm için Ortalama Adım Sayısı			41,46						
	HBP-DQA	**Bölümler**	1-500	500-1K	1K-2K	2K-3K	3K-4K	4K-5K	5K-6K	6K-7K	7K-8K
		Toplam Adım Sayısı	8806	7157	17491	19176	26923	26424	34068	20562	19607
		Başarılı Bölüm Sayısı	388	442	835	842	709	766	577	970	994
		Başarı Oranı	77,60%	88,40%	83,50%	84,20%	70,90%	76,60%	57,70%	97,00%	99,40%
		Toplam Adım Sayısı			180214						
		Her Bölüm için Ortalama Adım Sayısı			22,53						

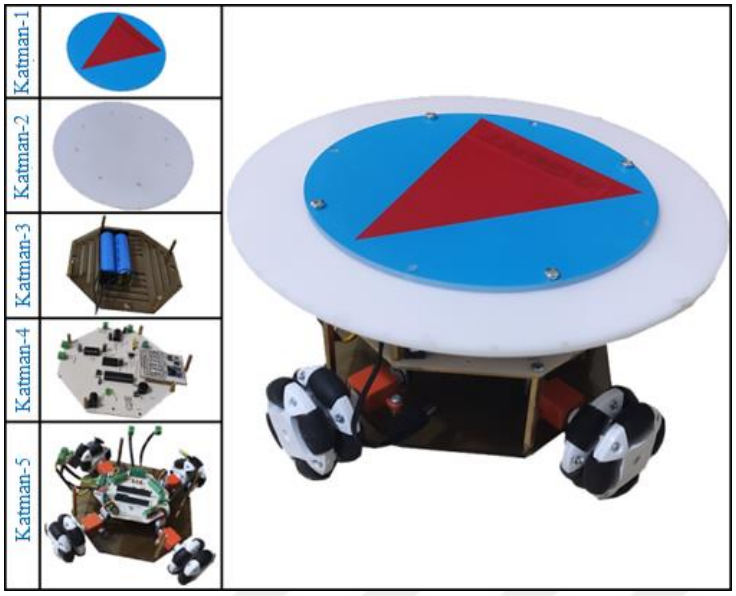
4.3 Mobil Robot Tasarımı

4.3.1 Mobil robotun genel yapısı

Çalışmanın amacı doğrultunda, sanal ortamda eğitilen ajanın eğitim verileri kullanılarak gerçek ortam uygulaması yapılması istendiğinden, çalışmaya uygun bir mobil robot tasarlamak gerekmektedir. Sanal ortam ayrık bir ortam olduğu için ajanın başlangıç yönü ve pozisyonu kısıtlı sayıdadır ancak gerçek ortam sürekli bir ortam olduğundan mobil robotun başlangıç yönü ve pozisyonun alabileceği değer sayısı sınırsızdır. Bu sebepten dolayı mobil robot, fazla manevra yapmasına gerek kalmadan sanal ortamdaki ajanın pozisyonuna yaklaşık bir pozisyona geçebilmesi gerekmektedir. Mobil robotlarda manevra kabiliyeti önemli biri özelliktir [77] ve omni tekerlekli yapıya sahip olan mobil robotlar manevra kabiliyetleri açısından normal tekerlekli mobil robotlara göre önemli avantajları vardır. Omni tekerlekli robotlar, yeniden yönlendirme olmaksızın bir pozisyondan diğerine geçerken doğrudan hareket edebilme özelliğinden dolayı yüksek manevra kabiliyetine sahiptir [78]. Bu sebepten dolayı gerçek ortam uygulamasında kullanılabilmek üzere Şekil 4.11'de gösterilen yüksek manevra kabiliyetine sahip, omni tekerlekli bir mobil robot tasarlanmıştır.

Mobil robot, beş katmandan oluşmaktadır;

- 1. ve 2. katman görsel katmanlardır, bu katmanlar robotun konumu ve rotasyonunun tespiti için kullanılmaktadır, sonraki bölümlerde kullanım amaçları açıklanacaktır.
- 3. katmanda, robotun enerji ihtiyacını sağlayan 2 adet seri bağlı 2500 mAH akım kapasitesine ve 3,7 V gerilim değerine sahip Li-ion pil bulunmaktadır.
- 4. katmanda, RF kontrol kartı vasıtasıyla bilgisayardan gelen komutları uygulayan robot kontrol kartı bulunmaktadır.
- 5. katman, 4 adet N20 enkoderli DC motor ile bu motorların kontrol ettiği, 8 silindirli 50 mm çapında 4 adet omni tekerleğin bulunduğu katmandır.



Şekil 4.11 Omni tekerlekli mobil robot

4.3.2 Mobil robot kontrol kartı

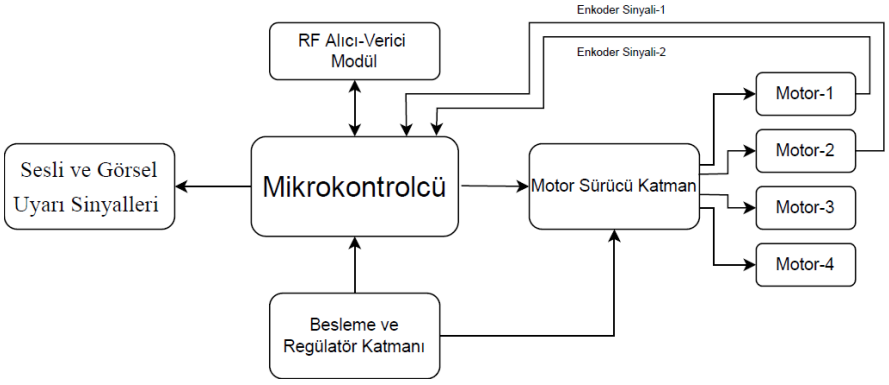
Tasarlanan mobil robotun yapacağı hareketler, RF kontrol kartı (RFKK) vasıtasıyla bilgisayardan gelen komutlara göre gerçekleşmektedir. Yani mobil robot kontrol kartı (MRKK), bilgisayarda çalıştırılan algoritmaların verdiği çıktılara göre mobil robotun hareket etmesini sağlamaktadır.

MRKK'nin genel özellikleri;

- Mikrokontrolcü tabanlı yapıya sahiptir,
- Bilgisayardan aldığı RF sinyalleri ile hareketlerini gerçekleştirmektedir,
- Dört farklı motorun yön ve hız kontrolü yapabilmektedir,
- Sesli ve ışıklı uyarılar verebilecek yapıdadır.

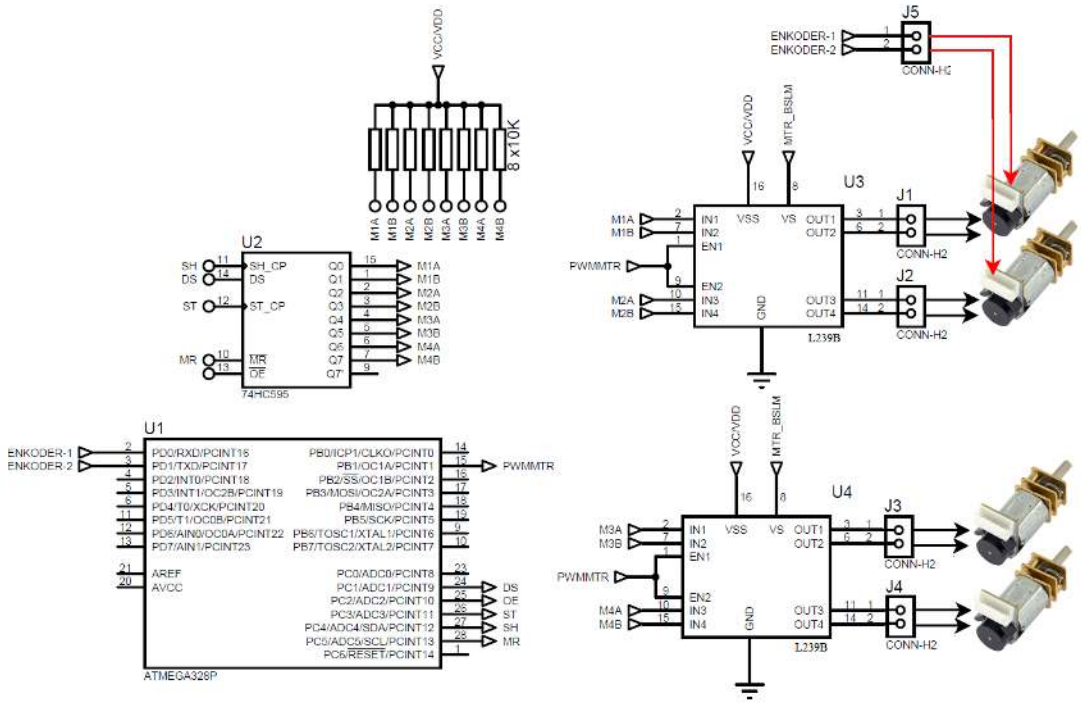
MRKK'ye ait çalışma diyagramı Şekil 4.12'de gösterilmektedir.

MRKK'de 8-bitlik Atmega328p mikrodenetleyicisi kullanılmaktadır. Atmega328p, 23 adet dijital genel amaçlı giriş/çıkış portu, altı adet 10-bitlik analog girişi, altı adet PWM çıkışı, programlanabilir seri USART, SPI ve I^2C donanımsal iletişim desteği, iki adet harici kesme girişi, üç adet zamanlayıcı/sayıcı kesmesi özelliklerine sahiptir. Ayrıca 20 MHz' e kadar çalışma hızı, 32 KB flash bellek, 1 KB EEPROM ve 2 KB dahili statik RAM bulunmaktadır.



Şekil 4.12 Mobil robot kontrol kartına ait çalışma diyagramı

MRKK'deki motor sürücü katmana ait devre şeması Şekil 4.13'de görülmektedir. Bu katman dört adet N20 enkoderli DC motorun yönünün ve hızının kontrol edildiği bölümdür. Bu katmanda bulunan L293B entegresi motor sürücü olarak kullanılmaktır. L293B entegresinde bulunan IN1, IN2, IN3 ve IN4 dijital girişlerine uygulanan sinyaller ile motorlar ileri-geri hareket ettirilebilmekte veya durdurulabilmektedir. Dört çıkış kanalı bulunan bu entegrenin her bir çıkış kanalının pik akımı 2 amperdir, sürekli akımı ise 1A'dir. Entegrede bulunan VS girişi, motorların beslemesi için kullanılmaktadır ve 4,5-36 volt arasında gerilim uygulanabilmektedir. EN1 ve EN2 girişlerine uygulanan PWM sinyalleri ile motorların hızı kontrol edilebilmektedir. Ayrıca entegrede aşırı sıcaklık koruması mevcuttur.



Şekil 4.13 Motor sürücü katmanına ait devre şeması

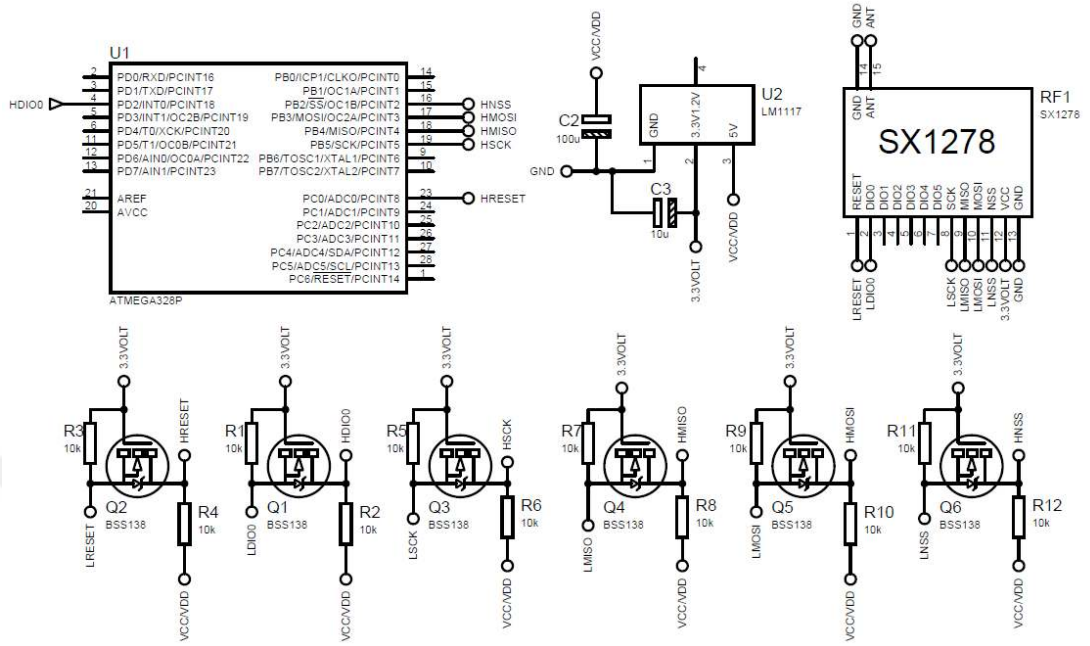
MRKK'deki motor sürücü katmanında 74HC595 entegresi kullanılmaktadır. Kaydırmalı kaydedici (Shift Register) olan bu entegre, mikrokontrolcüden aldığı seri sinyalleri paralel sinyallere çevirmektedir. Dahili kaydırmalı kaydedicisi sayesinde sekiz bitlik seri bilgi sinyallini aldıktan sonra istenilen zamanda sekiz bit olarak paralel çıkış verebilmektedir. Bu entegre kullanılarak daha az sayıda dijital pin ile motorlar güvenli bir şekilde kontrol edilebilir.

Ayrıca iki motorun enkoder sinyal uçları etmega328p entegresinin D0 ve D1 pinlerine bağlanmıştır. Bu pinler port değişiklik kesmesini desteklemektedir. Port değişiklik kesmesi kullanmak suretiyle enkoder sinyalleri sayılabilmektedir. Bu sayede enkoder sinyallerini kullanarak mobil robotun pozisyon ve yön kontrolü yapılabilmektedir.

MRKK ile bilgisayarın kablosuz haberleşebilmesi için SX1278 LoRa modülü kullanılmaktadır. LoRa teknolojisi, lisanssız spektrum üzerinde çalışmaktadır. Türkiye için lisans gerektirmeyen frekans bandı 433 MHz'dir. Bu teknoloji ile uzun mesafelerde, düşük veri hızında, yarı çift yönlü (half-duplex) haberleşme gerçekleştirilebilir. LoRa, CSS modülasyon tekniğini kullanmaktadır [79]. LoRa, kanal gürültüsü ve sinyal bozulmalarına karşı dayanıklı olduğundan çok uzak mesafelerde bile sağlıklı haberleşme gerçekleştirilebilir [80]. Ancak LoRa'nın uzun mesafe veri iletimi ve düşük güç tüketimi gibi avantajlarının yanında kısa mesafe haberleşme yöntemlerine göre düşük veri hızı ve yüksek gecikme oranı gibi dezavantajları da vardır [81]. Yapılan uygulamada kablosuz haberleşme ile bilgisayardan, mobil robota sadece hareket komutlar gönderileceğinden dolayı uygulamanın bahsedilen dezavantajlardan etkilemesi beklenmemektedir.

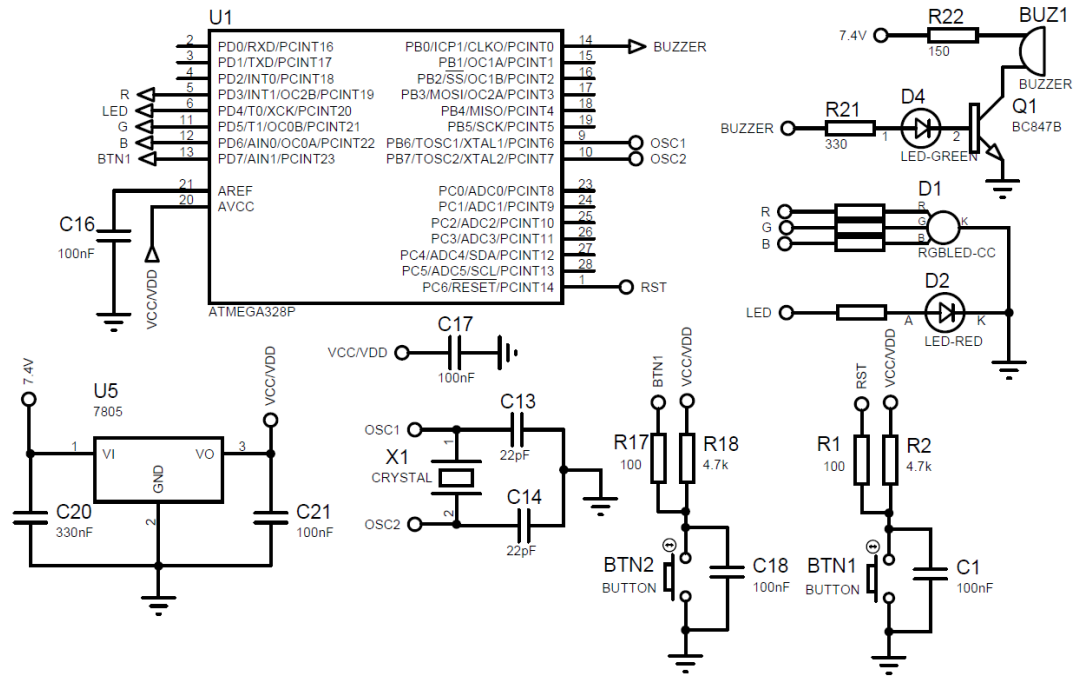
MRKK'deki RF alıcı-verici katmana ait devre şeması Şekil 4.14'de görülmektedir. SX1278 modülü ile mikrokontrolcü aralarında SPI ile haberleşmektedir. Bu modül 1,8-3,6 volt aralığında gerilim ile çalışabilmektedir. Mikrokontrolcü ise 5 volt gerilimle çalıştığından dolayı modülün besleme gerilimi LM1117 regülatör entegresi ile 3,3 volt olarak ayarlanmıştır. Mikrokontrolcü ile modülün dijital sinyal uçlarının gerilim seviyeleri farklıdır, modülün bu farktan dolayı zarar görmemesi için sinyal

uçlarında bss138lt1g n kanal mosfet kullanılarak lojik seviye dönüşümü yapılmaktadır.



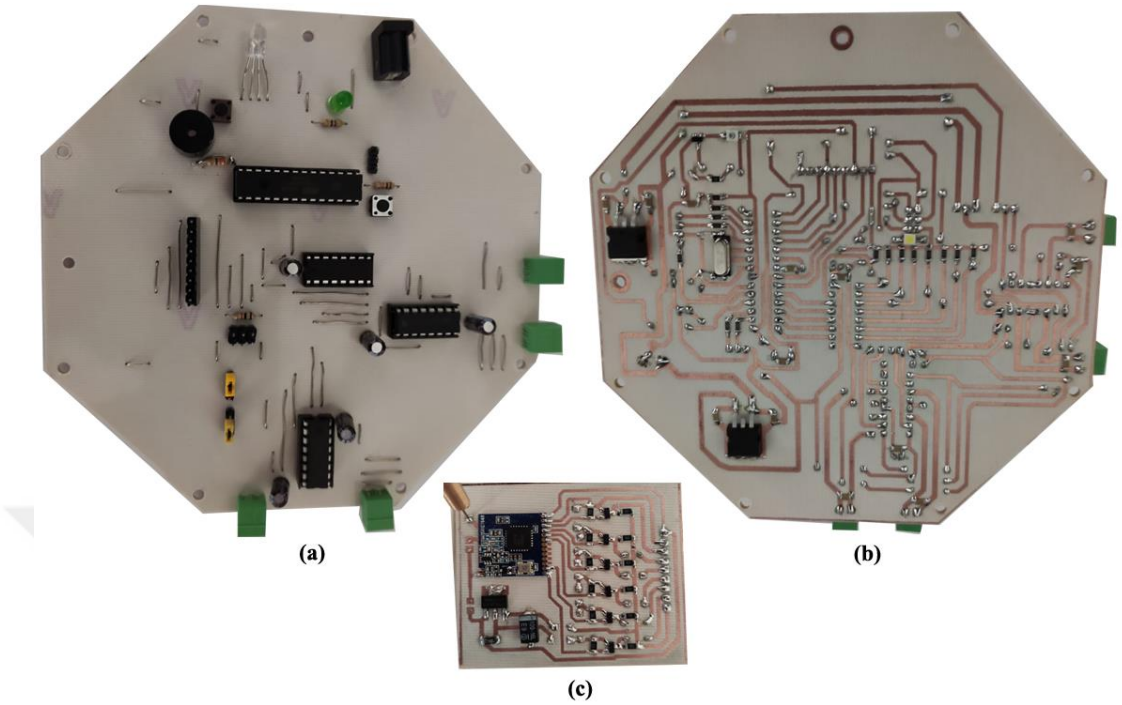
Şekil 4.14 RF alıcı-verici katmanına ait devre şeması

MRKK'deki diğer devre elemanlarına ait devre şeması Şekil 4.15'te görülmektedir. Devrede sesli uyarılar için buzzer, görsel uyarılar için RGB led kullanılmaktadır. Mikrokontrolcünün besleme gerilimi ise 7805 entegresiyle sabitlenmektedir.



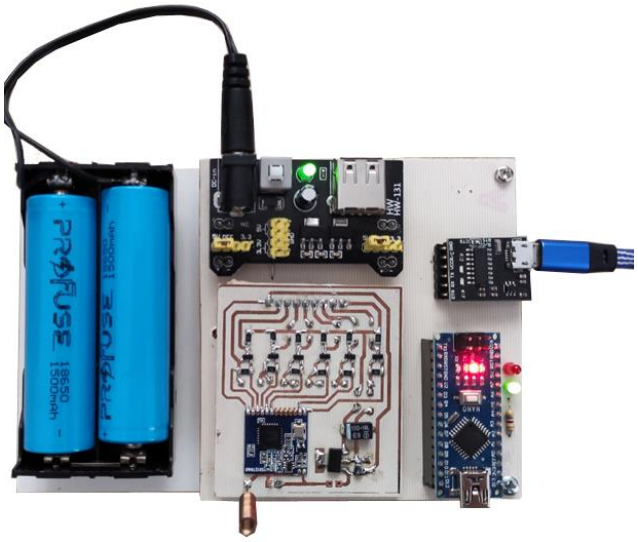
Şekil 4.15 MRKK'daki diğer devre elemanlarına ait devre şeması

Şekil 4.16'da MRKK'nin uygulama devresi gösterilmektedir.



Şekil 4.16 MRKK'na ait görüntüler (a) MRKK'nın üstten görünümü (b) MRKK'nın alttan görünümü (c) SX1278 modülü ve seviye dönüştürücü devre

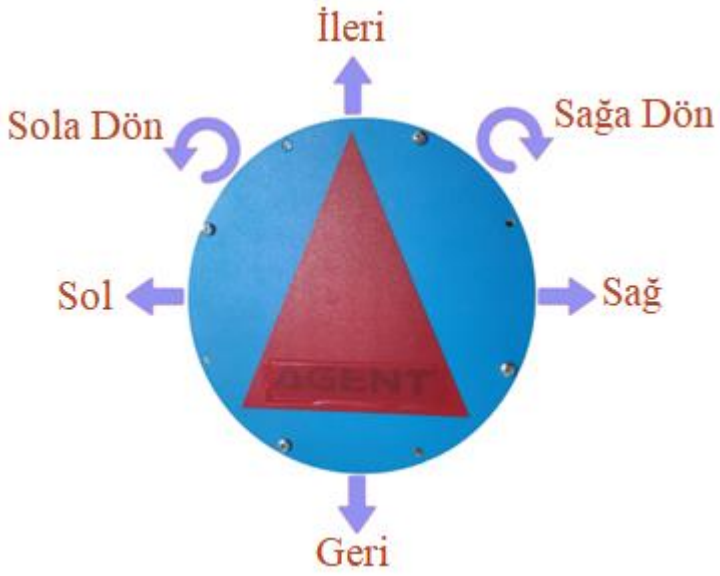
Şekil 4.17'de MRKK'nin bilgisayarla haberleşmesini sağlayan RF kumanda devresi gösterilmektedir. LoRa teknolojisini kullanan RF kumanda devresinin enerjisi 2 adet seri bağlı 1500 mAH akım kapasitesine ve 3,7 V gerilim değerine sahip Li-ion pil ile sağlanmaktadır. Devre, arduino nona geliştirme kartı, SX1278 modül, gerilim regülatör modülü ve USB/TTL dönüştürücü modülden oluşmaktadır.



Şekil 4.17 RF kumanda devresi

4.3.3 Mobil robotun hareketleri ve kalibrasyonu

Bu çalışmada ajan, empty environment sanal ortamında eğitilmiştir. Bu ortam için ajanın yapabileceği üç hareket (bir birim ileri git, 90ºsağa dön, 90º sola dön) vardır. Empty environment ortamı ayrık bir ortamdır. Ancak gerçek ortam sürekli bir ortam olduğu için mobil robot, açısal olarak ve pozisyon olarak sayısız farklı konumda olabilir. Mobil robotun ayrık ortamdan aldığı bilgileri kullanabilmesi için gerçek ortamında bir nevi ayrık ortama benzetilmesi gerekmektedir. Yani mobil robot, konumunu istenilen bir pozisyona getirebilecek şekilde hareket edebilmelidir. Mobil robot, istenen konuma gidebilmesi için Şekil 4.18'de gösterildiği gibi altı farklı hareket yapabilecek şekilde tasarlanmıştır. Mobil robot 0-360 derece aralığında, yaklaşık üç derecelik hassasiyet ile dönüşler ve yaklaşık bir cm'lik hassasiyetle doğrusal hareketler yapabilmektedir.

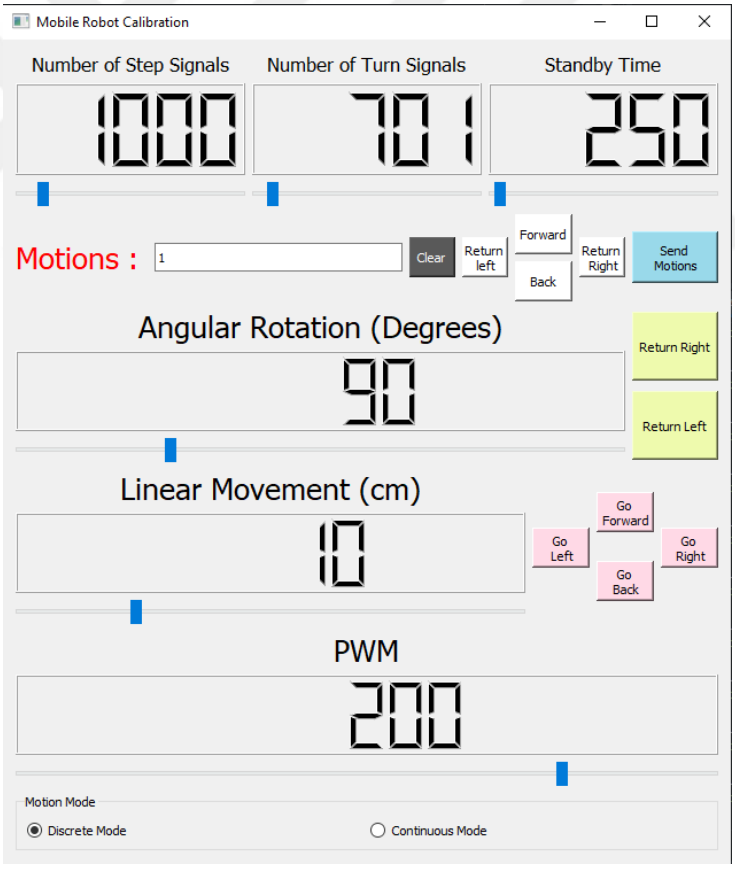


Şekil 4.18 Mobil Robotun Hareketleri

Tasarlanan mobil robotta Şekil 4.19'da gösterilen sekiz silindirli, 50 mm çapında dört adet omni tekerlek kullanılmaktadır. Ancak omni tekerlekli robotlar yüksek manevra kabiliyetine sahip olmasına rağmen bulundukları zemin hareketlerini olumsuz yönde etkileyebilmektedir [82]. Tasarlanan mobil robotta enkoderli motorlar kullanılmasına rağmen enkoder sinyalleri robotun pozisyonunu garanti edemez. Çünkü robotun bulunduğu zemin ve omni tekerleklerin duruş pozisyonlarından dolayı kalkış esnasında tekerleklerin bir miktar boşta dönmesi gibi istenmeyen durumlar ile karşılaşılabilir.

Şekil 4.19 Omni tekerlek

Bahsedilen sorunları nispeten azaltabilmek için Şekil 4.20'de gösterilen mobil robot kalibrasyon programı hazırlanmıştır. Python dilinde hazırlanan program UART haberleşme protokolünü kullanarak RF kumandaya erişebilmekte ve mobil robota, RF kumanda aracılığıyla istenilen hareket komutlarını gönderilebilmektedir.

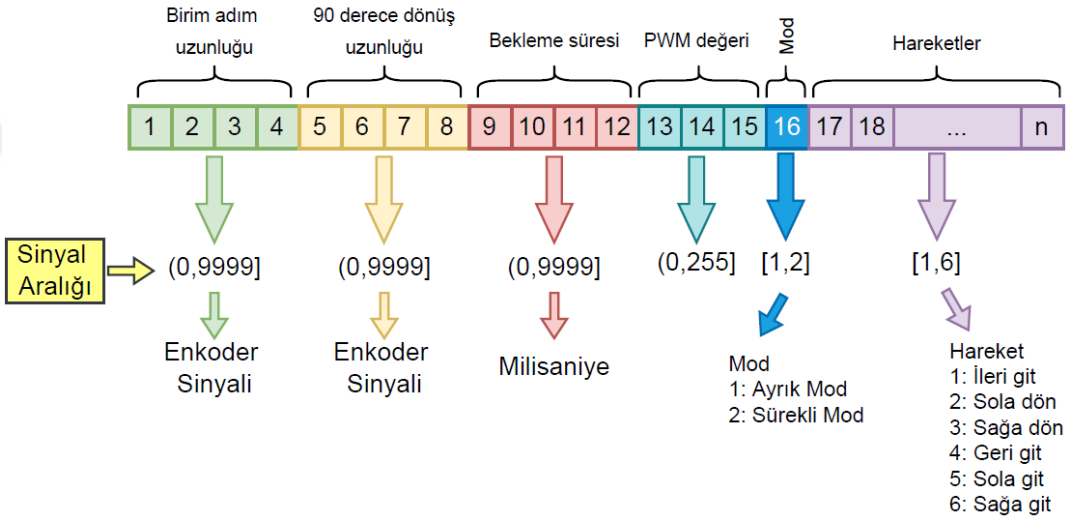


Şekil 4.20 Mobil robot kalibrasyon programı

Mobil robot kalibrasyon programı kullanılarak mobil robotun bir birim adım (bu çalışma için 30 cm olarak belirlenmiştir) için enkoderden gelen sinyal sayısı ve 90^o dönüş için enkoderden gelen sinyal sayısı bulunabilmektedir. Çoklu hareketler

gönderilebilmekte ve seçilen moda (ayrık, sürekli) göre hareketler arasında geçişte belirtilen süre kadar beklenilir ya da beklenilmeyebilinir. Farklı değerlerde açısal ve doğrusal hareketler belirlenebilir ve PWM ile motorların hız kontrolü yapılabilir.

Mobil robotun, yukarıda belirtilen özelliklerini kontrol edebilmek için Şekil 4.21'deki yapıda bir veri paketi tasarlanmıştır. Bu veri paketi robotun yapması gereken harekete göre hazırlanarak RF sinyalleri ile gönderilmektedir, robot ise bu sinyalleri çözerek gerekli hareketleri yapabilmektedir.



Şekil 4.21 RF veri paketi

Veri paketinde kullanılan her bir karakter ASCII kodlama sistemine göre kodlanmakta ve sadece rakamların kod karşılıkları ('0'=0x30, '1'=0x,31, … ,'9'=0x39) kullanılmaktadır. Bir veri paketi, mobil robotun;

- Doğrusal mı (ileri, geri, sağ ve sol) ya da dairesel mi(sola dön, sağa dön) hareket edeceğini,
- Doğrusal veya dairesel olma durumuna göre enkoder sinyalinin ne olması gerektiği,
- Her bir hareket sonunda beklenip beklenmeyeceğini ve beklenecekse, ne kadar süre bekleneceğini,
- Motorların PWM değerlerinin ne olması gerektiğini ve
- Çoklu hareket yapacaksa bunların neler olduğu gibi robotun nasıl hareket edeceğine dair tüm bilgileri içermektedir.

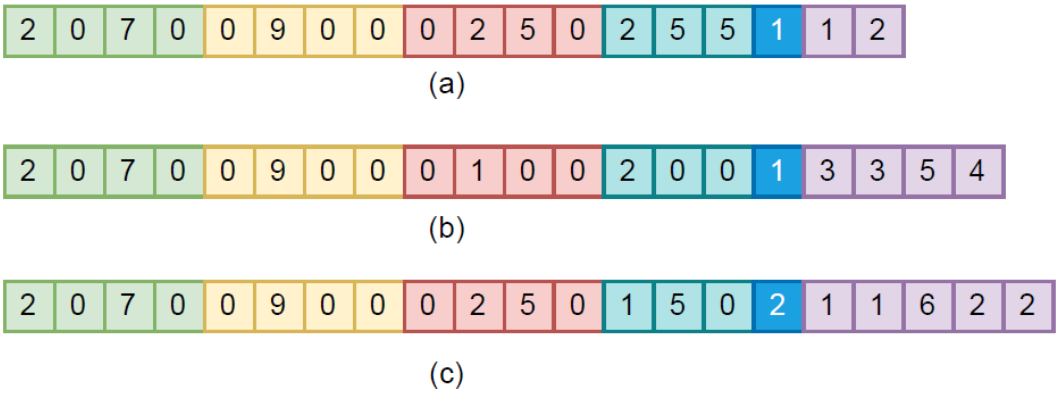
Omni tekerlekli mobil robotlar, çalıştırıldıkları zeminden etkilenme eğilimleri vardır. Yani omni tekerlekli mobil robotlarda enkoderli motorlar kullanılsa dahi farklı

zeminlerde, aynı değerde ölçülen enkoder sinyal için değişen pozisyon değeri farklı olabilir. Bu durumun etkisini en aza indirmek için mobil robotun çalıştırılacağı zeminde, robotun yapacağı temel hareketler için Tablo 4.3'te gösterilen enkoder sinyal değerleri deneysel olarak tespit edilmiş ve uygulama esnasında bu değerler kullanılmıştır.

Tablo 4.3 Deneysel olarak elde edilen enkoder sinyal değerleri

Hareket	**Enkoder Sinyali**
1 cm Doğrusal Hareket	22
30 cm Doğrusal Hareket (Bir Birim)	2070
3 Derece Dönüş Hareketi	10
90 Derece Dönüş Hareketi	900

Şekil 4.22'de bazı örnek veri paketleri verilmektedir. Şekil 4.22 (a)'da verilen veri paketine göre robot sırasıyla bir birim ileri gittikten sonra 90 derece sola dönmesi istenmektedir. PWM değeri 255 olan bu veri paketinde, iki hareket arasındaki bekleme süresi 250 ms'dir. Şekil 4.22 (b)'de verilen veri paketine göre robot sırasıyla iki kez 90 derece sağa dönmesi, bir birim sola gitmesi ve bir birim geri hareket etmesi istenmektedir. PWM değeri 200 olan bu veri paketinde, hareketler arası bekleme süresi 100 ms'dir. Şekil 4.22 (c)'de verilen veri paketine göre robot sırasıyla iki kez bir birim ileri gitmesi, bir birim sağa gitmesi ve iki kez 90 derece sola dönmesi istenmektedir. PWM değeri 150 olan bu veri paketinde, hareketler arasında bekleme süresi yoktur.

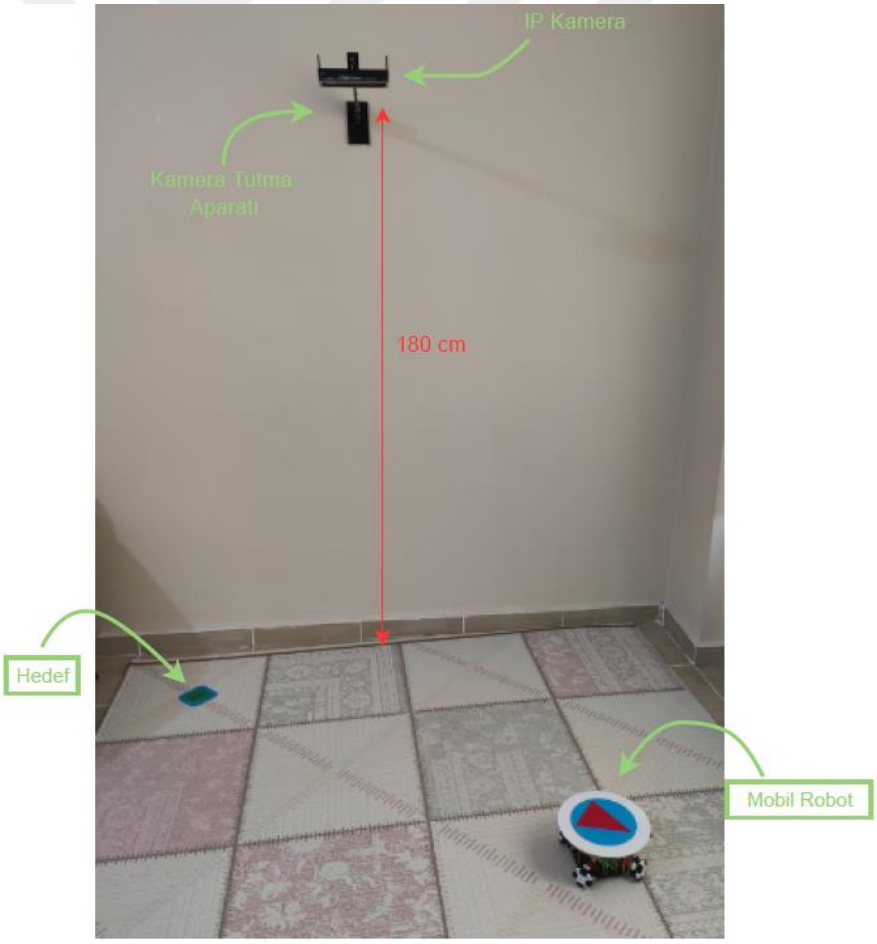


Şekil 4.22 Örnek veri paketleri

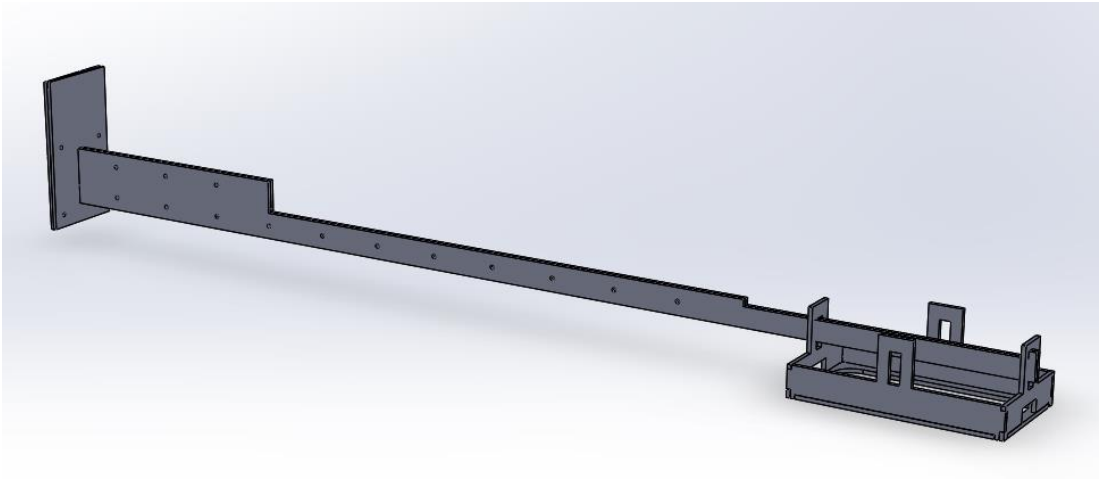
4.4 HBP-DQA ile Gerçek Ortam Uygulaması

Bu bölümde, ayrık empty environment ortamında eğitilen HBP-DQA'nın, gerçek ortamda da kullanılabileceği gösterilmektedir. Burada HBP-DQA'nın eğitimi ayrık bir ortamda yapılmıştır ve uygulamanın yapılacağı gerçek ortamın ayrık bir yapıya benzetilerek HBP-DQA uygulaması yapılmıştır.

Ayrık yapıya benzetilecek olan gerçek ortam Şekil 4.23'de gösterilmektedir. Ortam ile ilgili veriler, ortamın üsten görüntüsünü alan IP kamera vasıtasıyla gerçekleştirilmektedir. IP kameranın uygun şekilde görüntü alabilmesi için Şekil 4.24'te gösterilen kamera tutma aparatı hazırlanmıştır. Uygulama için yerden 180 cm uzağa monte edilen bu aparat sayesinde IP kamera duvardan 80 cm uzağa yerleştirilebilmektedir. Bu sayede uygulama yapılan ortamın üstten görüntüsü alınabilmektedir.

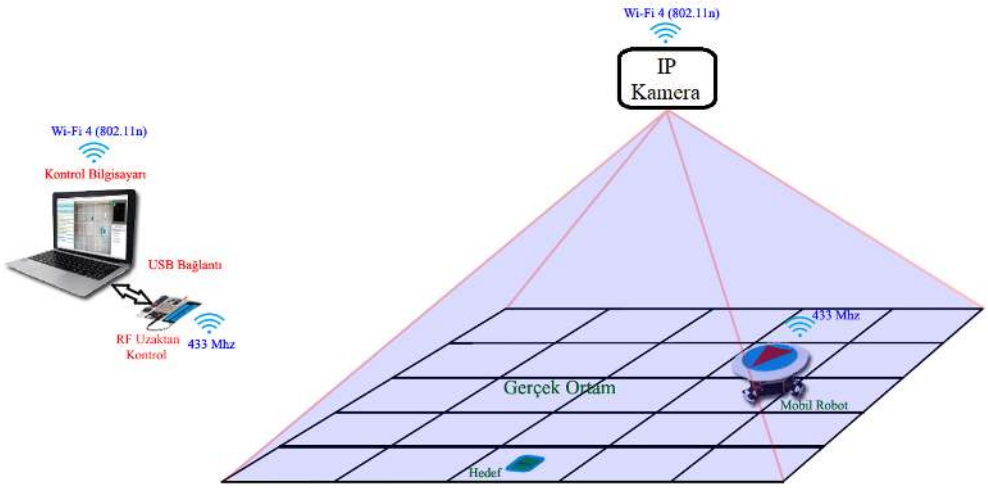


Şekil 4.23 Gerçek ortamın yapısı



Şekil 4.24 Kamera tutma aparatı

Ayrık yapıya benzetilen gerçek ortamda kullanılan mobil robot, Şekil 4.25'te gösterildiği gibi hareket komutlarını RF sinyalleri ile bilgisayardan almaktadır.



Şekil 4.25 Ayrık yapıya benzetilen gerçek ortam

Gerçek ortamın ayrık yapıya benzetilmesi;

- Evrişimli sinir ağları ile mobil robot ve hedefin tespit edilmesi,
- Gerçek ortamın, sanal ortamdaki gibi parsellere ayrılması,
- Mobil robotun pozisyon ve açısının ayrık ortama uyacak şekilde düzeltilmesi,
- Gerçek ortam ile sanal ortamın eşleştirilmesi, aşamalarıyla gerçekleşmektedir.

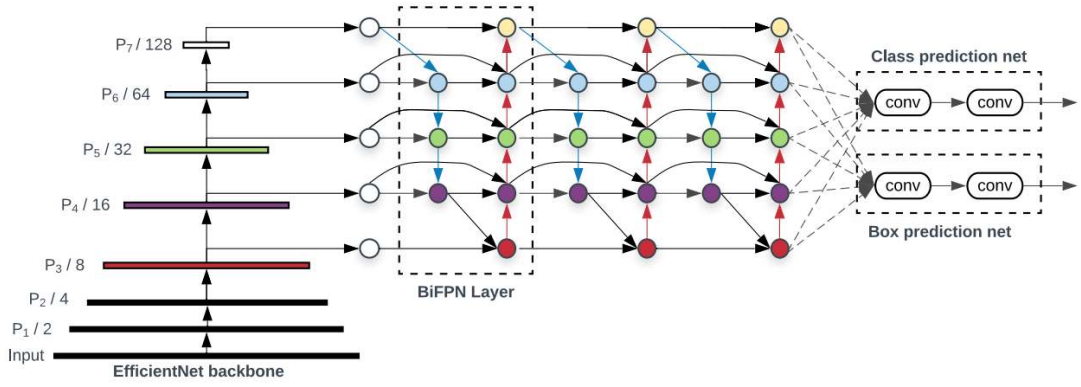
Bu aşamaları gerçekleştirerek, mobil robotun sanal ortamda eğitilmiş HBP-DQA'yı kullanabilmesi ve hedefine otonom olarak ilerlemesini sağlayan bir uygulama

geliştirilmiştir. Python dilinde hazırlanan bu uygulama Şekil 4.26'da görülmektedir. Uygulamanın aşamalarla ilgili açıklamalar bu bölüm altında yapılmaktadır.

Şekil 4.26 Mobil robotun otonom olarak hareket etmesini sağlayan uygulama

4.4.1 Evrişimli sinir ağları ile nesnelerin tespit edilmesi

Uygulamada nesne tespiti için EfficientDet ESA modeli kullanılmaktadır. Omurga ağı olarak EffiCientNet modelini kullanan bu model, ağ genişliğini, derinliğini ve giriş çözünürlüğünü birlikte ölçekleyerek görüntü sınıflamada iyi bir performans göstermektedir. Ayrıca modelin parametre sayısı optimize edildiğinden dolayı sistem kaynaklarının kullanımı açısından da avantajlıdır [83-85]. EfficientDet ailesinin genel yapısı Şekil 4.27'de görülmektedir.



Şekil 4.27 EfficientDet ailesinin genel yapısı [83]

EfficientDet ailesi, parametre sayısı değişen sekiz modelden oluşmaktadır. EfficientDet ailesine ait modeller ve nesne tespiti için gereken süreler Tablo 4.4'te gösterilmektedir [86]. EfficientDet modelleri COCO 2017 veri kümesi kullanılarak eğitilmiş modellerdir. COCO veri kümesi 123.287 görüntü ve 886.284 örnek'ten oluşmaktadır. Veri kümesinde 80 adet tanımlı nesne bulunmaktadır [87]. Uygulama

için parametre sayısı en az ve nesne tespitinde en hızlı olan EfficientDet D0 modeli seçilmiştir.

Tablo 4.4 EfficientDet ailesine ait modellerinin nesne tespiti için harcanan süreler

Model	Giriş Görüntüsü (piksel)	Hız (ms)
EfficientDet D0	512x512	39
EfficientDet D1	640x640	54
EfficientDet D2	768x768	67
EfficientDet D3	896x896	95
EfficientDet D4	1024x1024	133
EfficientDet D5	1280x1280	222
EfficientDet D6	1280x1280	268
EfficientDet D7	1536x1536	325

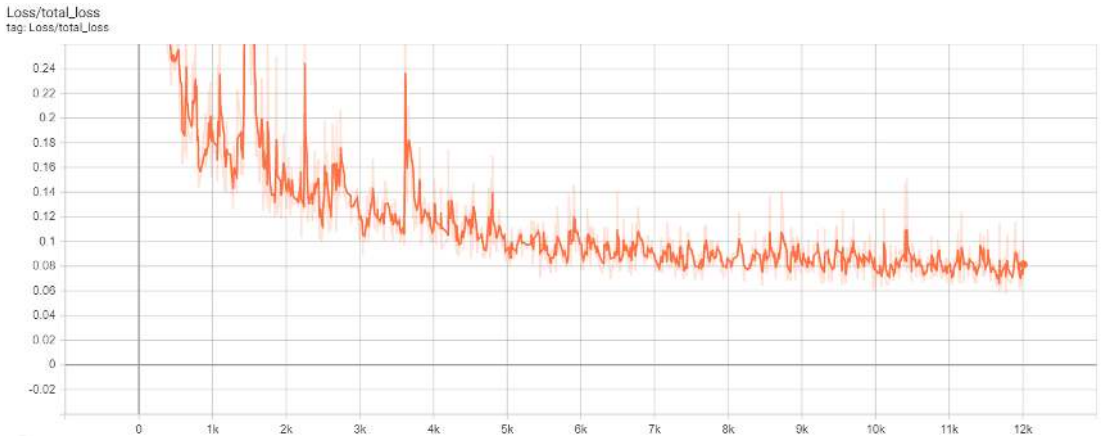
ESA modeli, Şekil 4.28'de gösterilen ajan ve hedef nesnelerinin tespiti için eğitilmiştir. Farklı çözünürlük ve ışık koşullarında elde edilen, eğitim için 117 (Bkz. Şekil 4.29), test için 32 (Bkz. Şekil 4.30) adet görsel kullanılmıştır.

Şekil 4.28 Ajan ve hedef nesnesi

Şekil 4.29 ESA modelinin eğitimi için kullanılan görseller

Şekil 4.30 ESA modelinin testi için kullanılan görseller

12.000 epoch eğitilen modele ait total-loss grafiği Şekil 4.31'de gösterilmektedir. Grafiğe göre 5.000 epoch sonrasında modelin büyük ölçüde eğitildiği görülmektedir.



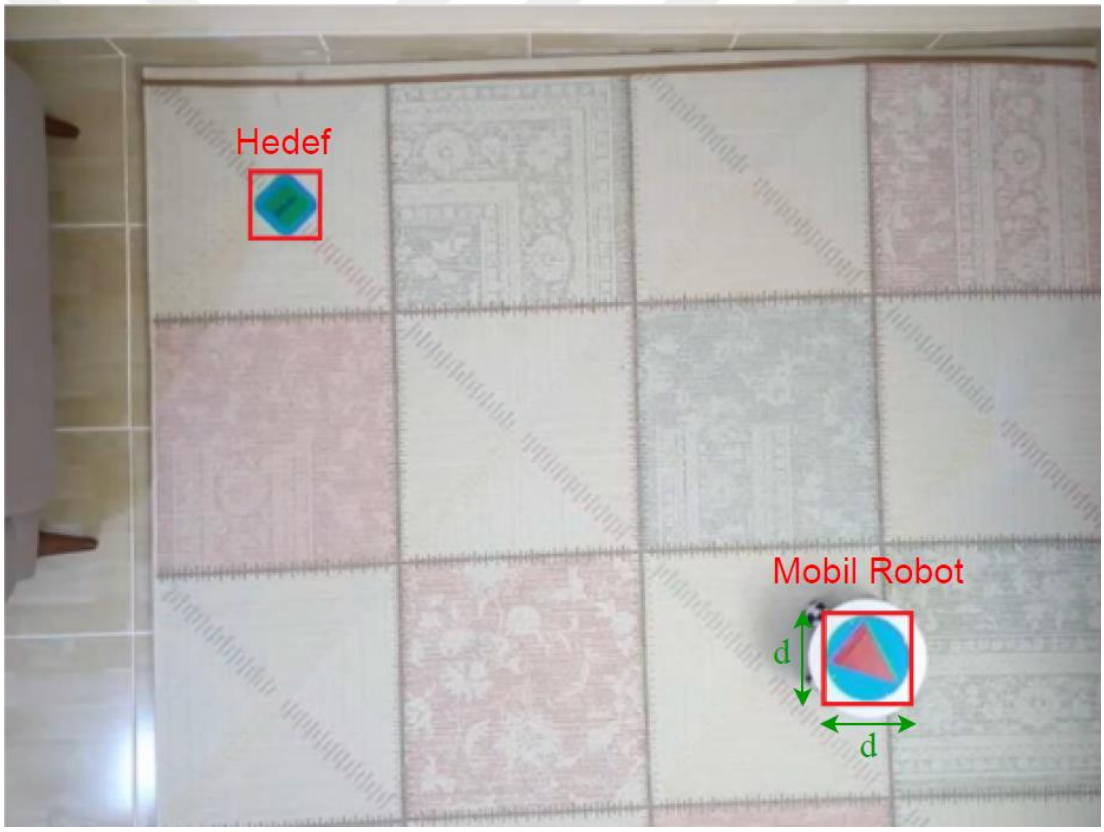
Şekil 4.31 Total loss grafiği

Eğitilen ESA modeli kullanılarak, ortamda ajan ve hedefin olup olmadığı, nesnelerin görüntüdeki pozisyon ve merkez bilgileri tespit edilmektedir. Ayrıca ajan görseli referans alınarak, ortamın piksel mesafe dönüşümü yapılabilmektedir.

4.4.2 Gerçek ortamın parsellere ayrılması

HBP-DQA'nın eğitildiği ortam ayrık bir ortamdır. Fakat mobil robotun, eğitilen HBP-DQA'yı kullanabilmesi için gerçek ortamın, ayrık ortama benzetilmesi gerekmektedir. Benzetim işleminin aşamaları aşağıda açıklanmaktadır;

Aşama-1: Gerçek ortam verileri, sadece ortamın üsten alınan görüntüsü ile elde edilmektedir. Bu görüntüden ESA kullanılarak mobil robotun ve hedefin konumları Şekil 4.32'deki gibi tespit edilmektedir.



Şekil 4.32 Mobil robotun ve hedefin ESA ile tespit edilmesi

Aşama-2: Sonraki aşamalarda kullanılmak üzere hedefin merkezi $(Goal_x, Goal_y)$, mobil robotun merkezi $(Agent_x, Agent_y)$ hesaplanır. Gerçek ortamdaki her bir parselin alanı 30x30 cm^2 olarak belirlenmektedir. Yani mobil robotun bir parselin merkezinden diğer parselin merkezine geçebilmesi için 30 cm yol kat etmesi gerekmektedir. 30 cm olarak belirlenen adım mesafesi $(Step_d)$, mobil robotun

görseli referans alınarak hesaplanmaktadır. Daire şeklinde olan mobil robot görselinin çapı 16 cm'dir, ajanı mobil robotu tespit eden çerçevenin bir kenarının (d) uzunluğu da yaklaşık olarak mobil robot görselinin çapına eşittir. Yani 1 cm, yaklaşık olarak d/16 pikseldir. Mobil robotun, piksel cinsinden adım mesafesi denklem (4.2)'de gösterilmektedir.

$$Step_d = 30\ cm \ \rightarrow \ Step_d = \frac{30}{16} * d\ \ piksel \tag{4.2}$$

Aşama-3: Ortam görüntüsünün eni ($Image_w$) ve boyu ($Image_h$) olmak üzere, gerçek ortamdaki parsellere ait merkezler denklem (4.3) kullanılarak bulunmaktadır. Gerçek ortam parsellere ayırırken ajanın konumunun sabit olmamasından dolayı hedefin merkezi referans alınarak, parsel merkezlerinin ($G_{centers}$) olduğu bir liste oluşturulur ve Şekil 4.33'de görüldüğü gibi gerçek ortam parsellere bölünür.

$$G_{centers} = \{a, b |\ a, b\ \in\ N\ and\ (0 < Goal_x \pm a * Step_d < Image_w, 0 < Goal_y \pm b * Step_d < Image_h)\} \tag{4.3}$$

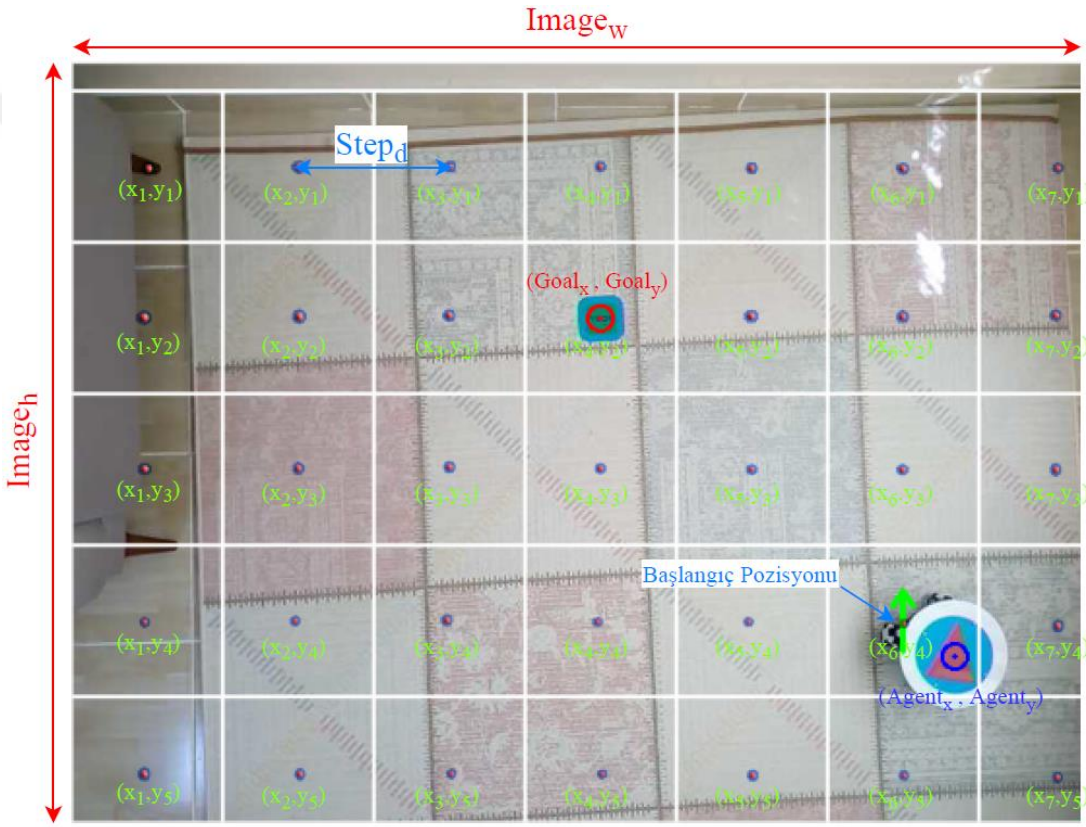
Şekil 4.33 Gerçek ortamın parsellere bölünmesi

Aşama-4: Parsellere bölme işleminde hedefin merkezi referans olarak kullanıldığından dolayı mobil robotun konumu herhangi bir parselin merkezi ile

eşleşmeyebilir. Bu sebepten dolayı mobil robotun en yakınındaki parsele uygun şekilde konumlanması gerekmektedir. Denklem (4.4) kullanılarak mobil robotun, parsel merkezlerine uzaklıkları ($Center_d$) listesi oluşturulur.

$$Center_d = \{(k,l) \in G_{centers} \,|and\, \sqrt{(Agent_x - x_k)^2 - (Agent_y - y_l)^2}\,\} \qquad (4.4)$$

$Center_d$ listesindeki minimum değeri oluşturan $G_{centers}$ listesinin elemanı mobil robota en yakın parsel merkezini göstermektedir. Bu parsel aynı zamanda mobil robotun başlangıç pozisyonunu göstermektedir. Şekil 4.34'te mobil robotun başlangıç pozisyonun belirlenmesi görülmektedir.



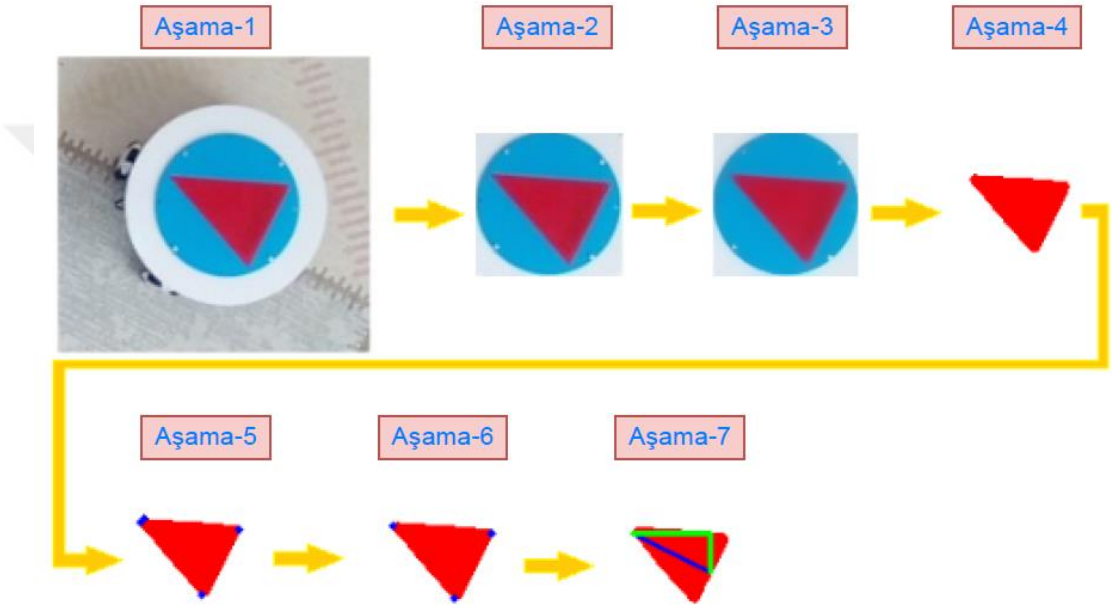
Şekil 4.34 Mobil robotun başlangıç pozisyonunun belirlenmesi

4.4.3 Mobil robotun rotasyonunun bulunması

ESA'ların çok iyi öznitelik çıkarabilen yapıları sayesinde görüntü tanıma alanında başlıca yöntem haline gelmiştir. ESA'ların çoğu görüntünün açısı ile ilgilenmemekte, sadece görüntü tanımaya odaklanmaktadırlar. Ancak robot uygulamalarında açı bilgisi de çok önemlidir [88].

Bu çalışmada ajan HBP-DQA ile ayrık bir sanal ortamda eğitilmiştir. Sanal ortam için sadece 4 yön bulunmaktadır. Gerçek ortamda ise mobil robot, açısal olarak sınırsız şekilde pozisyon alabilir. Mobil robotun sanal ortamdaki bilgileri

kullanabilmesi için yönünü sanal ortamdaki en uygun yöne benzetecek şekilde ayarlaması gerekmektedir. Bu sebepten dolayı mobil robotun açısını tespit edebilecek bir yapıya ihtiyaç duyulmaktadır. ESA tarafından mobil robotun tespiti için kullanılan görsel aynı zamanda mobil robotun açısının tespiti için oluşturulan algoritmada da kullanılmaktadır. Mobil robotun üstünde bulunan ikizkenar üçgenin yönü ve açısı aynı zamanda mobil robotun yönü ve açısıyla aynıdır. Algoritmanın amacı görseldeki üçgenin yönünü ve açısını bularak mobil robotun yönünü ve açısını tespit etmektir. Algoritmaya ait aşamalar Şekil 4.35'te gösterilmektedir.



Şekil 4.35 Mobil robotun açısının tespit edilmesi

Aşama-1: Mobil robotun ve hedefin bulunduğu ortamın üsten görüntüsü alınmaktadır.

Aşama-2: ESA vasıtasıyla mobil robotun koordinatları belirlenir ve görüntüden bu kısım alınır. Mobil robot görselinin hemen altında, beyaz bir katman kullanılmaktadır, bu katman robotun altında kalan kısmı görsel olarak kapatmaktadır. Böylelikle, algoritma çalışırken zemindeki renklerden etkilenmemekte ve mobil robot her türlü zemin renginde algoritmayı kullanabilmesi sağlanmaktadır.

Aşama-3: Görseldeki gürültüyü azaltmak ve kenarları belirginleştirmek için Bilateral Filtresi uygulanır [89].

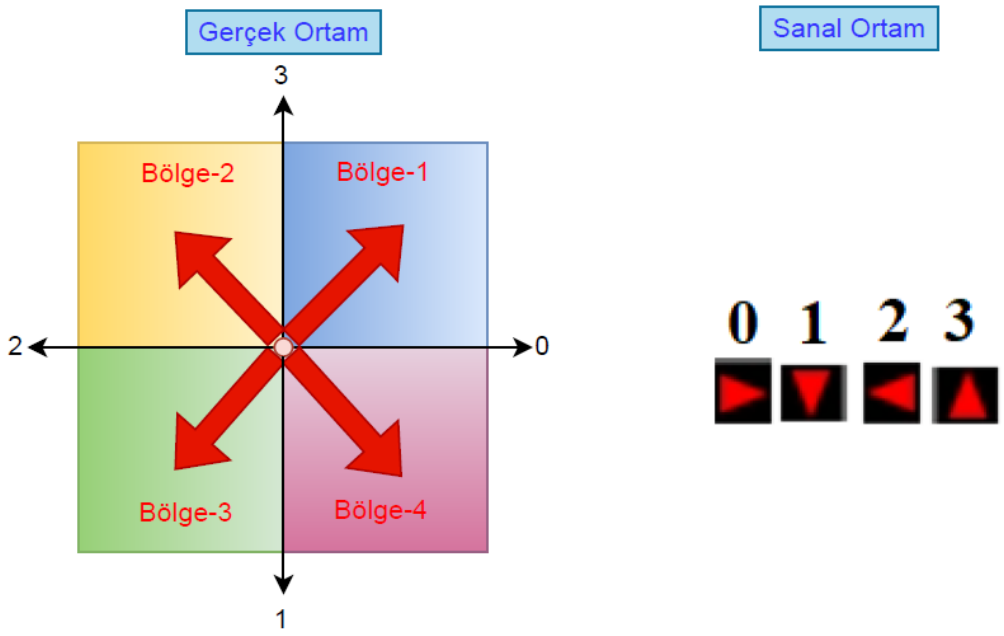
Aşama-4: Mobil robot görseli daire şeklinde olduğu için ESA ile belirlenen çerçeve yaklaşık olarak kareye benzemektedir, çerçevenin merkezi her zaman üçgen şeklinin içinde kalmaktadır. Görselin merkezindeki piksel ve sekiz komşusunun RGB değerleri alınarak R, G, B değerleri ayrı ayrı listelenmektedir. Her bir listenin içindeki değerler küçükten büyüğe sıralanmaktadır. R, G, B için oluşturulan listelerin 5. elemanları alınarak üçgen görselinin referans RGB değerleri elde edilmektedir. Bu işlemin amacı farklı ışık ortamlarında çalışabilecek bir yapı kurmaktır. Referans RGB değerlerine yakın olan piksellere kırmızı diğer piksellere beyaz renk değeri verilerek iki renkli bir görüntü elde edilir.

Aşama-5: Görüntü sağdan, soldan, üstten ve alttan ayrı ayrı taranarak kırmızı rengin ilk tespit edildiği pikseller belirlenmektedir. Bu şekilde dört nokta elde edilir.

Aşama-6: Belirlenen dört nokta içindeki en yakın iki nokta aynı köşeyi göstermektedir. İki noktadan sadece biri kullanılabilmektedir. Seçim yapabilmek için iki noktanın ayrı ayrı diğer noktalara uzaklıkları hesaplanır ve diğer noktalara daha uzak olan nokta 3. Köşe noktası olarak seçilmektedir. Çünkü daha uzakta olması, noktanın daha uçta olması anlamına gelmektedir.

Aşama-7: Köşe noktalarının birbirleriyle olan mesafeleri hesaplanmaktadır. Üçgenin ikiz kenarları, taban kenarından daha uzun belirlendiği için en yakın 2 köşe noktasının bulunduğu kenar, üçgenin tabanını göstermektedir diğer köşe ise üçgenin tepe noktasını göstermektedir. Tepe noktası ile tabanın orta noktasına bir çizgi çekilerek Şekil 4.35 aşama-7'deki gibi Pisagor üçgeni oluşturularak, Arctanjant ile üçgenin açısı dolayısıyla mobil robotun açısı bulunmaktadır.

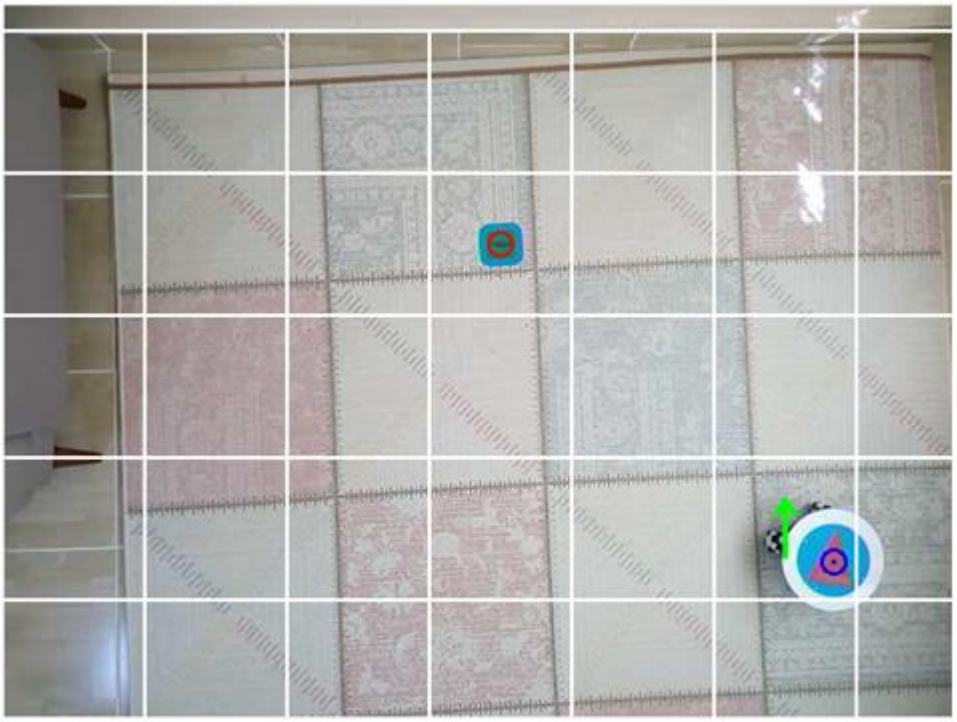
Eğer bulunan açı sıfır derece ise mobil robotun gerçek ortam ile sanal ortam pozisyonu eşleşebilmektedir. Eğer bulunan açı sıfır dereceden farklı ise mobil robotun yönü Şekil 4.36'da gösterildiği gibi dört bölgeden birinde olabilmektedir. Üçgenin yönü mobil robotun hangi bölge içinde olduğunu belirlemektedir. Mobil robotun döneceği yön, hangi yöne açısal olarak yakın olduğu karşılaştırılarak belirlenmektedir.



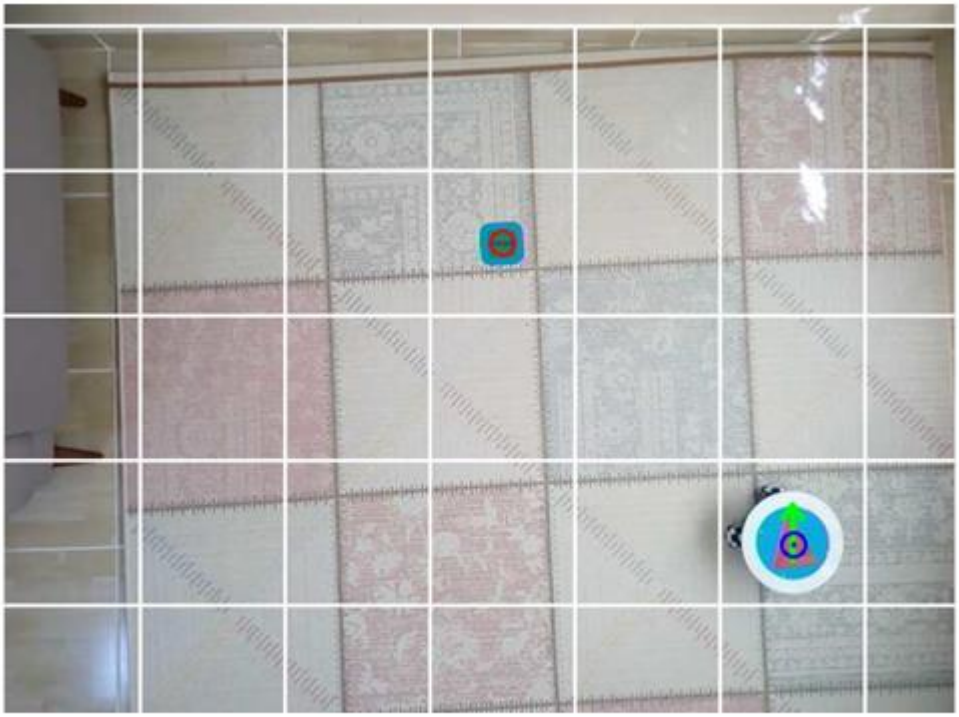
Şekil 4.36 Gerçek ortam ile sanal ortamda yönler

Algoritma ile mobil robotun hangi bölgede olduğu, robotun sanal ortamdaki en uygun yön ile eşleşmesi için açısal olarak sağa veya sola ne kadar dönmesi gerektiği ve rotasyonun düzeltilmesinden sonra mobil robotun hangi pozisyonda olacağı belirlenmektedir.

Mobil robot başlangıçta Şekil 4.37 (a)'da görüldüğü gibi herhangi bir pozisyonda olabilir. Mobil robot Şekil 4.37 (b)'de görüldüğü gibi otonom bir şekilde, bu bölümde belirlenen değerleri kullanarak açısını uygun hale getirmekte ve daha sonra önceki bölümde belirlenen başlangıç pozisyonuna, lineer hareketler (sağ, sol, ileri, geri) ile konumlamaktadır. Mobil robotun, açı ve konumunu düzeltmesi sonucu, hem mobil robot hem de hedef Şekil 4.37 (b)'de gösterildiği gibi bulundukları parselin merkezine yerleşmektedirler. Aynı zamanda mobil robot, sanal ortamda olduğu gibi alabileceği 4 yönden birine konumlanmaktadır.



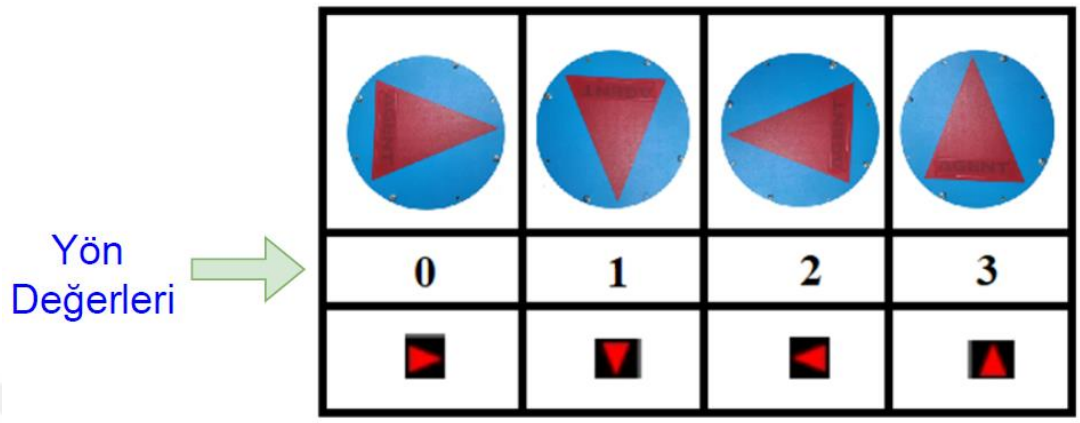
(a)



(b)

Şekil 4.37 (a) Mobil robotun sanal ortam ile eşleşmeyen başlangıç pozisyonu (b) Mobil robotun pozisyonunu otonom olarak uygun başlangıç pozisyonuna getirmesi

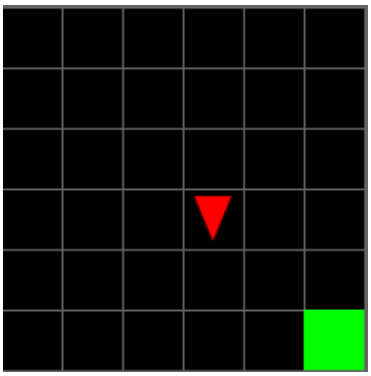
Mobil robot ve hedef sadece parsellerin merkezlerinde konumlandığı ve Şekil 4.38'de gösterildiği gibi mobil robotun alabileceği dört yön bulunduğundan dolayı gerçek ortam ile ayrık sanal ortam eşleştirilebilir duruma gelmektedir.



Şekil 4.38 Mobil robotun ve sanal ajanın alabileceği yönler

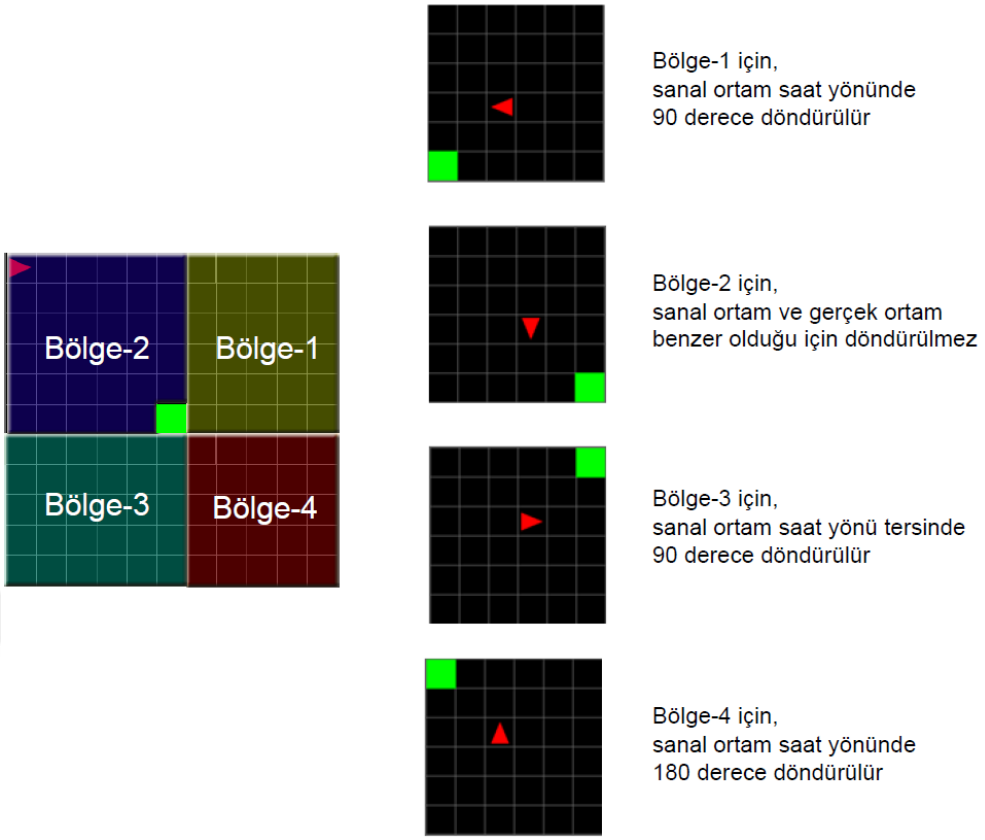
4.4.4 Gerçek ortam ile sanal ortamın eşleştirilmesi ve yol planının belirlenmesi

Sanal ortamda eğitilen HBP-DQA'yı, gerçek ortamda kullanabilmek için mobil robotun pozisyonu ile sanal ajanın pozisyonunu ve gerçek hedefin pozisyonu ile sanal hedefin pozisyonunu eşleştirmek gerekmektedir. Sanal ortam için ajan başlangıçta herhangi bir pozisyonda ve yönde olabilmektedir, hedefin konumu ise Şekil 4.39'daki gibi sanal ortamın sağ alt köşesindedir.



Şekil 4.39 Sanal ortam

Ancak gerçek ortamda mobil robot, hedefin çevresinde herhangi bir pozisyonda ve yönde olabilmektedir. Yani gerçek ortam ile sanal ortamın eşleşemeyeceği birçok durum ortaya çıkmaktadır. Bu problemi çözebilmek için, mobil robotun, hedefe olan konumuna göre gerçek ortam Şekil 4.40'daki gibi dört bölgeye ayrılmaktadır. Sanal ortam ise gerçek ortam ile eşleşebilecek şekilde döndürülmektedir.



Şekil 4.40 Gerçek ortamdaki bölgelerin elde edilebilmesi için sanal ortamın döndürülmesi

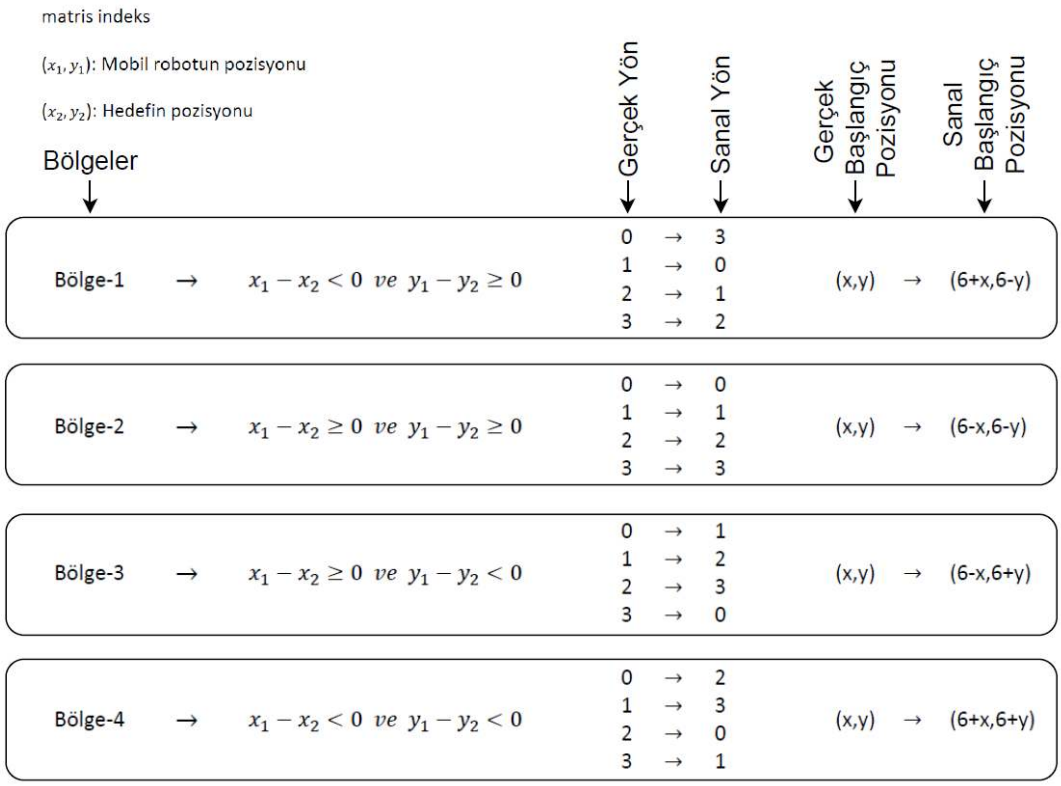
Döndürülen sanal ortam ile gerçek ortamı eşleştirebilmek için bölgeye göre sanal ajanın başlangıç yönü ve pozisyonu Şekil 4.41'deki gibi dönüştürülmektedir.

Bu dönüşümler sonucunda gerçek ortamdaki tüm durumlar sanal ortam ile eşleştirilebilmekte ve eğitilmiş HBP-DQA, gerçek ortam için Şekil 4.42'deki gibi kullanılabilmektedir.

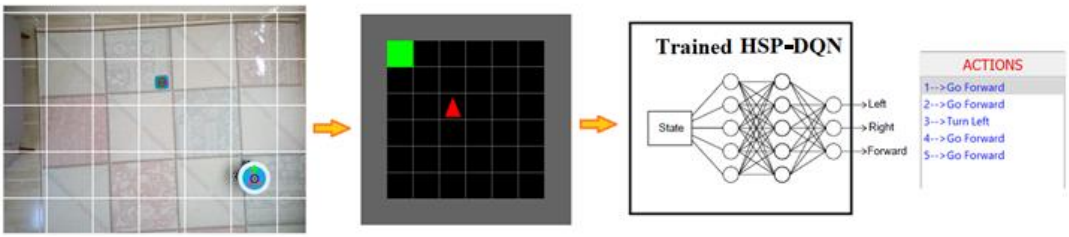
Mobil robot harekete geçmeden önce yol planı sanal ortamda belirlenmektedir. Eğitilmiş HBP-DQA ile ajanın o anki durumu için bir hareket üretilmekte ve sanal ortamda uygulanarak, ajanın yeni pozisyonuna geçmesi sağlanmaktadır. Ajan hedefine varıncaya kadar HBP-DQA hareket üretmeye devam etmektedir ve yapılan bu hareketler mobil robotun kullanabilmesi için listelenmektedir. Bu şekilde mobil robotun yol planı çıkarılmaktadır.

Yol planı çıkarıldıktan sonra yapması gereken hareketler mobil robota sırasıyla gönderilmektedir. Her yapılan hareket sonrası mobil robotun açı ve pozisyon

kontrolü yapılarak sapmalar düzeltilmektedir. Mobil robotun o anki hareketini tamamlayıp tamamlamadığı ise arka plan fark yöntemi ile belirlenmektedir. Mobil robota, tamamladığı her hareket sonrası bir sonraki yapması gereken hareket gönderilerek hedefe varması sağlanmaktadır.



Şekil 4.41 Gerçek ortama göre sanal ortamın dönüştürülmesi



Şekil 4.42 Gerçek ortamda HBP-DQA kullanımı

5. TARTIŞMA

DQA algoritması MiniGrid-Empty-8x8-v0 ortamında başarılı olmasına rağmen MiniGrid-Empty-16x16-v0 ortamında benzer bir başarı elde edememektedir. Çünkü ajan, MiniGrid-Empty ortamlarında sadece hedefe vardığında ödül alabilmektedir. Eğer ortam MiniGrid-Empty-16x16-v0 gibi geniş bir alana sahip ve ödüller seyrek ise DQA algoritmasının rastgele hareketlerle ağı eğitebilecek kadar anlamlı veri üretebilmesi zordur ve yakınsama hızı çok yavaştır. Önerilen algoritmada, ajanın başlangıç pozisyonu, ödüle ulaşma olasılığının büyük olduğu bölgede başlatılarak ağın anlamlı veriye ulaşma ihtimali arttırılmakta ve ağ ortamda yolları öğrendikçe ajanın başlayabileceği alan kademeli olarak genişletilmektedir. Böylece seyrek ödüllerin olduğu geniş alanlar için ağın yakınsama probleminin üstesinden gelinmektedir. Çalışmanın performans analizleri MiniGrid-Empty-8x8 ortamından daha büyük bir ortam olan MiniGrid-Empty-16x16 için yapılmaktadır.

Yapılan çalışmanın uygulamasında Tablo 5.1'de donanım özellikleri belirtilen, Windows 10 Pro 64 Bit işletim sistemine sahip bilgisayar kullanılmaktadır.

Tablo 5.1 Uygulamada kullanılan bilgisayarın özellikleri

Donanım	Özellik
CPU	Intel™ Core™ i7-9750H CPU @ 2.60GHz 2.60 GHz
Ekran Kartı	NVIDIA GeForce RTX 2070 8GB
RAM	16 GB 2667 MHz

DQA ve HBP-DQA algoritmalarında kullanılan sinir ağları aynı özelliklere sahiptir. Sinir ağının yapısına ait bilgiler Tablo 5.2'de ve algoritmalara ait hiper parametreler Tablo 5.3'de gösterilmektedir.

Tablo 5.2 DQA ve HBP-DQA'da kullanılan sinir ağın özellikleri

Katmanlar	Nöron sayısı	Aktivasyon fonksiyonu	Parametreler
Giriş Katmanı	512	ReLU	301568
Gizli Katman-1	256	ReLU	131328
Gizli Katman-2	64	ReLU	16448
Çıkış Katmanı	3	Linear	195

Tablo 5.2 (Devam)

	Toplam Parametre	449.539
Optimizer: RMSprop algorithm		
Loss: Mean Squared Error		
Learning rate: 0,00025		
Rho: 0,95		
Epsilon: 0,01		

Tablo 5.3 DQA ve HBP-DQA algoritmalarına ait hiper parametreler

Hiper Parametreler	Değer
Toplam Bölüm Sayısı	8000
Maksimum Adım Sayısı	50
Bellek	2000
Epsilon Greedy Başlangıç Değeri	1.0
Minimum Epsilon Greedy	0,001
Epsilon Greedy Azaltma Oranı	0,9995
Mini Batch	64

DQA algoritmasında, ajanın başlayabileceği farklı durum sayısı 780'dir. Sinir ağı eğitilirken her yeni bölümde 780 durumdan biri başlangıç durumu rastgele olarak seçilmektedir. Önerilen algoritmada ise bölümler ilerledikçe eğitimin başlayabileceği alan kademeli olarak büyütülmektedir. Bununla birlikte ajanın başlayabileceği farklı durum sayısı Tablo 5.4'te gösterildiği gibi artmaktadır.

Tablo 5.4 Eğitim alanının kademeli olarak büyütülmesi

Eğitimin kademeleri	Bölüm Aralığı	Alan	Ajanın başlayabileceği farklı durum sayısı
1.Kademe	1-500	3x3	32
2.Kademe	501-1000	5x5	96
3.Kademe	1001-2000	7x7	192
4.Kademe	2001-3000	9x9	320
5.Kademe	3001-4000	11x11	480
6.Kademe	4001-5000	13x13	672
7.Kademe	5001-8000	14x14	780

Her iki algoritmada yukarıda belirtilen koşullarda, 8000 bölüm eğitildikten sonra elde edilen modellerin performansları incelendi. Algoritmaların, sinir ağının eğitimi için harcadığı süreler Tablo 5.5'te gösterilmektedir. Önerilen algoritmanın eğitimi, DQA algoritmasına göre %45,78 daha kısa sürede tamamlanmaktadır. DQA algoritmasında ajan ortamda daha fazla hareket etmesine rağmen, önerilen algoritmaya göre daha az sayıda hedefe vardığından dolayı bölümlerde harcanan süre artmaktadır.

Tablo 5.5 Algoritmaların eğitim süreleri

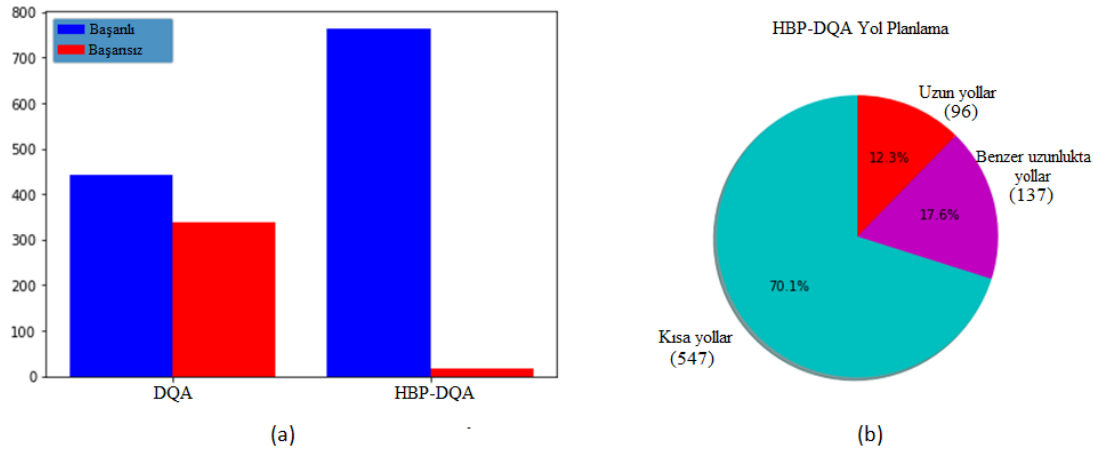
Algoritma	**Bölüm Sayısı**	**Toplam Adım**	**Eğitim Süresi**
DQA	8000	331681	8,3 saat
HBP-DQA	8000	180214	4,5 saat

Eğitilen modellerin başarısını karşılaştırmak için 780 farklı başlangıç koşulu bulunan ortamda tüm koşullar test edildi ve Tablo 5.6'daki sonuçlar elde edildi. Önerilen algoritmanın başarı oranı DQA algoritmasından %41,28 daha fazladır. Algoritmaların başarılarının karşılaştırıldığı grafik Şekil 5.1 (a)'da gösterilmektedir.

Tablo 5.6 Algoritmaların başarı oranları

Algoritma	**Başarılı**	**Başarısız**	**Başarı Oranı (%)**
DQA	442	338	56,67
HBP-DQA	764	16	97,95

Ayrıca algoritmaların tespit ettiği yolların uzunlukları da incelendi. DQA ve önerilen algoritma ile eğitilen modeller aynı başlangıç koşulları ile başlatılarak bulunan yol uzunlukları Şekil 5.1 (b)'de gösterildiği gibi karşılaştırıldı. Önerilen algoritma, farklı başlangıç koşullarının %70.1'inde DQA algoritmasına göre daha kısa yol bulmaktadır.



Şekil 5.1 HBP-DQA'nın sanal ortam (MiniGrid-Empty-16x16-v0) performansı

Yapılan testler sonucunda önerilen algoritmanın, seyrek ödüllerin bulunduğu geniş ortamlar için DQA algoritmasına göre başarı oranı ve yakınsama hızının daha iyi olduğu görülmektedir. Ayrıca geliştirilmiş DPÖ tabanlı güncel bazı algoritmaların başarı oranı üzerindeki etkileri Tablo 5.7'de gösterilmektedir.

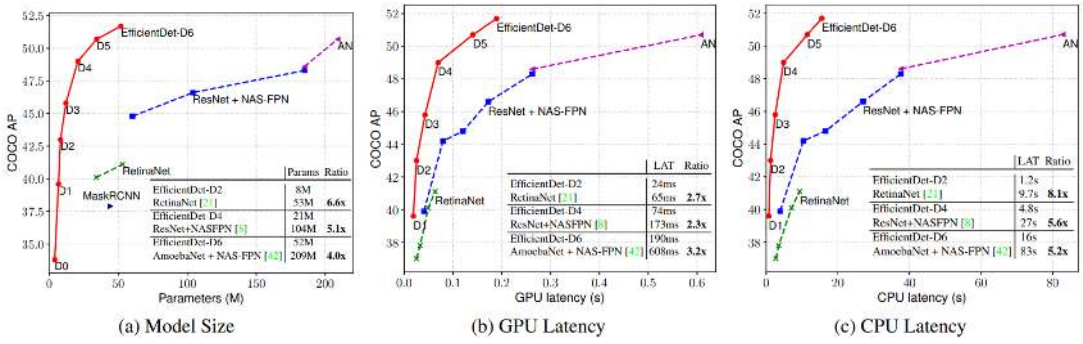
Tablo 5.7 Geliştirilmiş DPÖ tabanlı algoritmaların başarı oranı üzerindeki etkisi

Referans	**Yazarlar**	**Karşılaştırılan Algoritma (Başarı Oranı %)**	**Önerilen Algoritma (Başarı Oranı %)**	**Simülasyon**	**Gerçek Sistem**
[90]	Pfeiffer ve ark. (2018)	IL or CPO (37)	IL+CPO (90)	E	E
[91]	Hsu ve ark. (2018)	A3C (17,5)	LSTM + DRL (49,5)	E	E
[92]	Long vr ark. (2018)	NH-ORCA (78)	Parallel PPO (96,5)	E	H
[93]	Wang ve ark. (2019)	DDPG (42,67)	Fast-RDPG (97,42)	E	H
[94]	Lin ve ark. (2019)	PPO (23)	Improved PPO (88)	E	E
[95]	Wang ve ark. (2020)	POfD (58,5)	LwH (96,5)	E	H
[96]	Leiva ve Ruiz-del-Solar (2020)	DDPG (69)	PCL-LSTM (88)	E	E
[27]	Wang ve ark. (2020)	DQN (63,4)	DDQN with PER (81.1)	E	H

Tablo 5.7 (Devam)

[30]	Kim ve ark. (2021)	DQN (46)	DQN+GRU+action skipping (87)	E	H
[28]	Zhang ve ark. (2021)	DQN (75)	improved DQN (90)	E	H
[7]	Zhou ve ark. (2022)	DDPG (84,25)	LDDPG+D (94,25)	E	H
[5]	Gong ve ark. (2022)	DDPG (83,5)	MN-LSTM-DDPG (90.5)	E	H
[24]	Xing ve ark (2022)	DQN (75)	Area Division DQN (100)	E	E

Yapılan çalışmanın gerçek ortam uygulamasında sistem girdisi olarak sadece ortamın üstten alınan görüntüsü kullanılmaktadır. Uygulamanın gerçek zamanlı çalışması istendiğinden bu görüntünün verimli bir şekilde kullanılması gerekmektedir. Nesne tespiti için kullanılan modelin, nesne tespit yeteneği, hızı ve model boyutu çok önemlidir. Bu kriterler göz önüne alındığında nesne tespiti için EfficientDet-D0 modeli kullanılmıştır. EfficientDet modelinin, orijinal makalesinde sunulan performans analizleri Şekil 5.2'de gösterilmektedir. Modelin, model boyutu, hızı ve nesne tespiti açısından kısıtlı kaynağa sahip mobil robot uygulamaları için uygun olduğu düşünülmektedir.



Şekil 5.2 EfficientDet Modelinin COCO veri seti üzerindeki performansı [83]

EfficientDet modelinin, uygulamamızdaki performansı da test edilmektedir. Model, bölüm 4.4.1'de belirtilen şekilde eğitilmiştir. Eğitilen model, yerden 180 cm yüksekliğe monte edilen IP kameradan alınan görüntü ile nesneleri tespit etmektedir. Alınan görüntü, 800*600 piksel çözünürlüğünde ve 2,1x1,6 m^2'lik alanı kapsamaktadır. Yapılan testlerde, ortam aydınlatmasının uygun olması koşuluyla

model, mobil robot ve hedef nesnelerini başarılı şekilde tespit edebilmektedir. Nesne tanıma sürecinin 820~920 ms aralığında tamamlandığı tespit edilmiştir.

MiniGrid-Empty-16x16 sanal ortamında 780 farklı başlangıç koşulu vardır. Bölüm 4.4.4'te açıklanan yapı kullanılarak hedefin sadece belli bir pozisyonda olması yerine ortamda herhangi bir pozisyonda olabilmesi sağlanmaktadır. Yani hedef başlangıçta 196 farklı pozisyonda konumlanabilmektedir. Böylelikle eğitilen ağ önceki başarısını, 780x196=152.880 farklı başlangıç koşulu için de sağlamaktadır. Önerilen bölgesel eşleştirme işleminin tamamlanması ise 1ms'nin altında sürmektedir.

Gerçek ortam uygulamasında yapılan işlemler aşamalı olarak gerçekleşmektedir. Yapılan işlemlerin, deneysel olarak tespit edilen işlem süreleri Tablo 5.8'de gösterilmektedir. Kurulan sistem, 1~1,5 saniye aralığında ortamın durumunu algılayarak, robotun bir eylem yapmasını sağlayabilmektedir.

Tablo 5.8 Gerçek ortam işlem süreleri

İşlem	**Min. Süre (ms)**	**Maks. Süre (ms)**
Nesne tespiti	820	920
Ortamı parsellere ayırma	15	16
Robotun açısının tespiti	40	60
Alan tespiti	0	1
Yol planı	100	300
Hareket sinyalinin, bilgisayardan kumandaya, kumandadan robota gönderilmesi	100	150
Robotun gelen hareket sinyalini çözmesi	10	20
	1085 ms < Toplam Süre < 1467ms	

Gerçek ortam uygulamasında, giriş verisi, sadece ortamın üsten alınan görüntüsüdür. Bu görüntü kullanılarak işlem performansı gerektiren, anlamsal veri çıkarma, mobil robotun hareketini belirleme ve belirlenen hareketin mobil robota bildirme işlemleri merkezi bir bilgisayar tarafından gerçekleştirilmektedir. Mobil robot ise merkezi bilgisayardan aldığı komutlar doğrultusunda hareket uygulayıcı olarak görev yapmaktadır. Mobil robotun yapacağı işlemler hesaplama yükü gerektirmediğinden, Tablo 5.9'da da görülebileceği gibi düşük maliyetli bir mobil robot tasarlanmıştır.

Tablo 5.9 Mobil robotun maliyeti

Ürün	Fiyat
Robotun gövdesi + omni tekerlek (4 adet)	20$
N20 Micro Encoder Gear Motor 6V 105rpm (4 adet)	24$
Elektronik Komponentler	28$
Toplam	**72$**

Bilgisayar-mobil robot kablosuz haberleşmesi LoRa tabanlı modülasyon ile sağlanmaktadır. LoRa geniş kapsama alanı, düşük güç tüketimi, düşük maliyeti ve lisans gerektirmemesi gibi avantajları vardır [97]. Ancak 290 kbps veri taşıma hızına sahip olan LoRa birçok kablosuz haberleşme yöntemine göre daha yavaştır [98]. Yine de yapılan uygulamada sadece robotun hareketine dair küçük boyutlu data sinyalleri gönderildiğinden LoRa, yapılan uygulama için ideal bir haberleşme yöntemidir.

Tez çalışmasının gerçek ortam deneylerinde, DPÖ tabanlı mobil robot, Bölüm 4.4.4'te belirtilen 4 bölge için farklı başlangıç şartları ile başlatıldı. https://youtu.be/bEAmJF6lD4g'da da görülebileceği gibi mobil robotun tüm deneylerde hedefine başarılı bir şekilde ulaştığı tespit edildi. Yapılan deneylerde tespit edilen diğer hususlar aşağıdaki gibidir;

- Mobil robotun omni tekerlekli yapısı manevra açısından avantajlı olsa da robotun çalıştığı zeminden etkilenmeye yatkındır. Bu durum robotun hedefine varmasına engel olmasa da robotun hedefe ulaşma süresini olumsuz etkilemektedir.
- Kurulan sistemin adaptif yapısı sayesinde kameranın zeminden yüksekliği değiştirilerek sistemin çalışma alanı değiştirilebilmektedir. Ancak kameranın kapsadığı alanın aşırı büyütülmesi veya ortamın uygun şekilde aydınlatılmaması ortamdaki nesnelerin tespit edilmesini olumsuz etkileyebilmektedir.

Engelsiz bir ortam için yapılan bu çalışmanın, gelecekte yapacağımız çalışmalar için sağlam bir altyapı oluşturduğuna inanmaktayız. Kurulan bu yapı ile çok sayıda robotun uzun menzilli olarak tek bir merkezden kontrol edilme imkânı vardır. Çoklu robot uygulamaları önemli bir araştırma konusudur [99-101]. Ayrıca çok sayıda

robot kullanımı daha fazla veri çeşitliliği sağlayabildiğinden global politika eğitiminin hızlandırması açısından da önemlidir [102]. Tasarlanan robot, maliyet yönünden erişilebilir ve uzun menzilli kontrol edilebilir olmasından dolayı sürü robotiği ve çok ajanlı sistemlerde de kullanılma potansiyeline sahiptir.

6. SONUÇLAR

Bu tezde, seyrek ödüllerin olduğu büyük ortamlar için klasik DQA algoritmasına göre yakınsama hızını ve başarı oranını yükselten geliştirilmiş bir DQA algoritması olan HBP-DQA algoritması önerilmiştir.

Çalışmanın uygulama aşamasında kullanılmak üzere, düşük maliyetli ve merkezi bir bilgisayardan uzun menzilli olarak kontrol edilebilen, Omni tekerlekli bir mobil robot tasarlanmıştır.

Minimalistic gridworld sanal ortamında HBP-DQA algoritması kullanılarak eğitilen ajanın ağ parametreleri kullanılarak, bir mobil robotun hedefine ulaşması gerçek bir ortam için sağlanmıştır. Bu çalışma, minimalistic gridworld sanal ortamında eğitilen bir ajanın ağ parametrelerinin gerçek ortamda kullanıldığı ilk çalışma olma özelliğini taşımaktadır.

Tezin uygulama aşamasında, ortam bilgisi olarak sadece ortamın üstten alınan görüntüsü kullanılmıştır. Mobil robot ve hedefin konumlarının tespit edilmesi için kaynak kullanımı ve hız etmenleri göz önünde bulundurularak EfficientDet D0 ESA modeli kullanılmıştır. Ayrıca mobil robotun rotasyonunu hassas bir şekilde tespit edebilen, piksel tabanlı bir görüntü işleme algoritması önerilmiştir.

Sanal ortamda eğitilen modelin, gerçek ortamda daha verimli şekilde kullanılmasını sağlayan bir algoritma oluşturulmuştur. Algoritma, sanal ortamda sabit pozisyonlu hedef için eğitilen modeli, gerçek ortamda hedefin farklı pozisyonda olması durumunda da kullanılmasını sağlamaktadır. Algoritma aynı zamanda ayrık yapıya sahip sanal ortam ile gerçek ortamı eşleştirebilmektedir.

Tezin gerçek ortam deneylerinde DPÖ tabanlı mobil robot, farklı başlangıç koşulları için test edilmiştir ve mobil robotun tüm testlerde hedefine başarılı bir şekilde ulaştığı tespit edilmiştir.

Yapılan çalışma, gelecekte yapılması olası, mobil robotun hedefine optimum şekilde ulaşabilmesini sağlayan, sıralı görevler içeren ya da birden fazla mobil robotun

kontrol edildiği DPÖ algoritmalarının, gerçek dünya uygulamaları için altyapı oluşturmaktadır.

KAYNAKLAR

[1] J. Zheng, S. Mao, Z. Wu vd. "Improved Path Planning for Indoor Patrol Robot Based on Deep Reinforcement Learning," Symmetry 2022, vol. 14, no. 1, Jan. 2022, s. 1-12.

[2] P. K. R. Maddikunta, QV Pham, B Prabadevi vd. "Industry 5.0: A survey on enabling technologies and potential applications," Journal of Industrial Information Integration, vol. 26, Mar. 2022, s. 1-19.

[3] W. Zehra, A. R. Javed, Z. Jalil vd. "Cross corpus multi-lingual speech emotion recognition using ensemble learning," Complex & Intelligent Systems, vol. 7, no. 4, Jan. 2021, s. 1845–1854.

[4] K. Zhu ve T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," Tsinghua Science and Technology, vol. 26, no. 5, Oct. 2021, s. 674–691.

[5] H. Gong, P. Wang, C. Ni vd."Efficient Path Planning for Mobile Robot Based on Deep Deterministic Policy Gradient," Sensors 2022, vol. 22, no. 9, May 2022, s. 3579.

[6] L. Zhang, Y. Zhang, ve Y. Li, "Path planning for indoor Mobile robot based on deep learning," Optik, vol. 219, Oct. 2020, s. 165096.

[7] Q. Zhou, L. Lyu, ve H. Liu, "Deep Reinforcement Learning with Long-Time Memory Capability for Robot Mapless Navigation," Computer Supported Cooperative Work in Design, (s. 1215–1220), 2022.

[8] A. G. Sutton, Richard S and Barto, Reinforcement learning: An introduction. MIT Press, 1998.

[9] Y. Duan, X. Chen, R. Houthooft, J. Schulman, ve P. Abbeel, "Benchmarking Deep Reinforcement Learning for Continuous Control," International conference on machine learning, (s. 1329–1338), 2016.

[10] T. P. Lillicrap, J. J. Hunt, A. Pritzel vd. "Continuous control with deep reinforcement learning," ICLR 2016, (s. 1–14), 2016.

[11] H. Li, R. Cai, N. Liu, vd. "Deep reinforcement learning: Algorithm, applications, and ultra-low-power implementation," Nano Communication Networks, vol. 16, Jun. 2018, s. 81–90.

[12] A. Heuillet, F. Couthouis, ve N. Díaz-Rodríguez, "Explainability in deep reinforcement learning," Knowledge-Based Systems, vol. 214, Dec. 2021, s. 106685.

[13] V. Mnih K. Kavukcuoglu, D. Silver vd. "Human-level control through deep reinforcement learning," Nature, vol. 518, no. 7540, Feb. 2015, s. 529–533.

[14] V. Mnih K. Kavukcuoglu, D. Silver vd. "Playing Atari with Deep Reinforcement Learning," arXiv, Dec. 2013, Erişim tarihi: 2 Ağustos 2021. https://arxiv.org/abs/1312.5602v1.

[15] D. Silver, T. Hubert,J. Schrittwieser vd. "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm," arXiv, Erişim tarihi: 2 Ağustos 2021. https://arxiv.org/abs/1712.01815v1.

[16] D. Silver, J. Schrittwieser, K. Simonyan vd. “Mastering the game of Go without human knowledge,” Nature, vol. 550, no. 7676, Oct. 2017, s. 354-359.

[17] X. Pan, Y. You, Z. Wang, vd. “Virtual to Real Reinforcement Learning for Autonomous Driving,” British Machine Vision Conference 2017, (s.1-13), 2017.

[18] A. H. Qureshi, Y. Nakamura, Y. Yoshikawa vd. “Intrinsically motivated reinforcement learning for human–robot interaction in the real-world,” Neural Networks, vol. 107, Nov. 2018, s. 23–33.

[19] T. GUO, N. JIANG, B. LI vd. “UAV navigation in high dynamic environments: A deep reinforcement learning approach,” Chinese Journal of Aeronautics, vol. 34, no. 2, Feb. 2021, s. 479–489.

[20] J. Woo, C. Yu ve N. Kim, “Deep reinforcement learning-based controller for path following of an unmanned surface vehicle,” Ocean Engineering, vol. 183, Jul. 2019, s. 155–166.

[21] J. Wang, S. Elfwing ve E. Uchibe, “Modular deep reinforcement learning from reward and punishment for robot navigation,” Neural Networks, vol. 135, Mar. 2021, s. 115–126.

[22] P. N. Srinivasu, A. K. Bhoi, R. H. Jhaveri vd. “Probabilistic Deep Q Network for real-time path planning in censorious robotic procedures using force sensors,” Journal of Real-Time Image Processing, vol. 18, no. 5, Oct. 2021, s. 1773-1785.

[23] W. Wang, Z. Wu, H. Luo vd. “Path Planning Method of Mobile Robot Using Improved Deep Reinforcement Learning,” Journal of Electrical and Computer Engineering, vol. 2022, Apr. 2022, s.1-7.

[24] Y. Xing, R. Young, G. Nguyen vd. “Optimal Path Planning for Wireless Power Transfer Robot Using Area Division Deep Reinforcement Learning,” Wireless Power Transfer, vol. 2022, Mar 2022, s. 1-10.

[25] R. Huang, C. Qin, J. L. Li vd. “Path planning of mobile robot in unknown dynamic continuous environment using reward-modified deep Q-network,” Optimal Control Applications and Methods, Aug 2021, s. 1-18.

[26] J. Yu, Y. Su ve Y. Liao, “The Path Planning of Mobile Robot by Neural Networks and Hierarchical Reinforcement Learning,” Frontiers in Neurorobotics, vol. 14, Oct. 2020, s. 63.

[27] Y. Wang, Y. Fang, P. Lou vd. “Deep Reinforcement Learning based Path Planning for Mobile Robot in Unknown Environment,” Journal of Physics: Conference Series, vol. 1576, no. 1, (s. 1-7), 2020.

[28] H. Zhang, P. Wang, C. Ni vd. “Deep Q network algorithm based on sample screening,” International Conference on Computer Application and Information Security, (s. 130-135), 2022.

[29] X. Ruan, C. Lin, J. Huang vd. “Obstacle avoidance navigation method for robot based on deep reinforcement learning,” ITOEC 2022, (s. 1633-1637), 2022.

[30] I. Kim, S. H. Nengroo ve D. Har, “Reinforcement Learning for Navigation of Mobile Robot with LiDAR,” ICECA 2021, (s. 148-154), 2021.

[31] U. Taşkıran, "Temporomandibular Eklem Bozukluklarının Belirlenmesinde Sinyal İşleme ve Yapay Zeka Teknikleri Kullanılması," Doktora Tezi, Konya Teknik Üniversitesi, Lisansüstü Eğitim Enstitüsü, 2019.

[32] F. Varçın, "Ağırlıklandırılmış Derin Evrişimsel Sinir Ağları Topluluğu ile Böcek Türlerinin Sınıflandırılması," Doktora Tezi, Kırıkkale Üniversitesi, Fen Bilimleri Enstitüsü, 2020.

[33] K. Uyar, "Konvolüsyonel Sinir Ağlarında Ağ Eğitiminin İyileştirilmesi," Doktora Tezi, Selçuk Üniversitesi, Fen Bilimleri Enstitüsü, 2022.

[34] F. Daş, "Görüntü Tabanlı Otomatik Yol Tespiti ve Modellenmesi," Doktora Tezi, Eskişehir Teknik Üniversitesi, Lisansüstü Eğitim Enstitüsü, 2020.

[35] Q. Zhang, M. Zhang, T. Chen vd. "Recent advances in convolutional neural network acceleration," Neurocomputing, vol. 323, Jan. 2019, s. 37–51.

[36] C. Bircanoglu ve N. Arica, "A comparison of activation functions in artificial neural networks," Signal Processing and Communications Applications Conference, (s. 1-4), 2018.

[37] M. Yıldırım, "Derin Öğrenme Yöntemleri Kullanılarak Görüntüler Üzerinde İnsan Hareketlerinin ve Hastalıklarının Sınıflandırması," Doktora Tezi, Fırat Üniversitesi, Fen Bilimleri Enstitüsü, 2021.

[38] X. Wu, D. Sahoo ve S. C. H. Hoi, "Recent advances in deep learning for object detection," Neurocomputing, vol. 396, Jul. 2020, s. 39–64.

[39] L. Jiao ve J. Zhao, "A Survey on the New Generation of Deep Learning in Image Processing," IEEE Access, vol. 7, Nov. 2019, s. 172231–172263.

[40] J. Li, W. Dai, F. Metze vd. "A comparison of Deep Learning methods for environmental sound detection," International Conference on Acoustics, Speech and Signal Processing - Proceedings, (s. 126–130), 2017.

[41] D. W. Otter, J. R. Medina, ve J. K. Kalita, "A Survey of the Usages of Deep Learning for Natural Language Processing," IEEE transactions on neural networks and learning systems, vol. 32, no. 2, Feb. 2021, s. 604-624.

[42] A. Gupta, A. Anpalagan, L. Guan vd. "Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues," Array, vol. 10, Jul. 2021, s. 1-20.

[43] I. Lenz, H. Lee ve A. Saxena, "Deep learning for detecting robotic grasps," The International Journal of Robotics Research, vol. 34, no. 4–5, Mar. 2015, s. 705–724.

[44] S. Liu, S. Liu, W. Cai vd. "Early diagnosis of Alzheimer's disease with deep learning," ISBI 2014, (s. 1015-1018), 2014.

[45] H. Li, Y. Pan, J. Zhao vd. "Skin disease diagnosis with deep learning: A review," Neurocomputing, vol. 464, Nov. 2021, s. 364–393.

[46] S. Nissen, Large Scale Reinforcement Learning using Q-SARSA(λ) and Cascading Neural Networks, Masters Thesis, University of Copenhagen, Department of Computer Science, 2007.

[47] K. L. A. Yau, J. Qadir, H. L. Khoo vd. “A Survey on Reinforcement Learning Models and Algorithms for Traffic Signal Control,” ACM Computing Surveys, vol. 50, no. 3, Jun. 2017, s. 1–38.

[48] C. Schneider, “Intelligent signalized intersection management for mixed traffic using Deep Q-Learning,” Masters Thesis, Delft University of Technology, Complex Systems Engineering and Management, 2020.

[49] V. Mnih K. Kavukcuoglu, D. Silver vd. “Human-level control through deep reinforcement learning,” Nature, vol. 518, no. 7540, Feb. 2015, s. 529–533

[50] S. J. Park ve D. R. Shires, “Learning optimal actions with imperfect images,” Real-Time Image Processing and Deep Learning, (s. 96-102) 2019.

[51] R. Zhang, A. Ishikawa, W. Wang vd. “Using Reinforcement Learning with Partial Vehicle Detection for Intelligent Traffic Signal Control,” IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 1, Jan. 2021, s. 404-415.

[52] A. Krizhevsky, I. Sutskever, ve G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” Communications of the ACM, vol. 60, no. 6, Jun. 2017, s. 84–90.

[53] O. Russakovsky, J. Deng, H. Su vd. “ImageNet Large Scale Visual Recognition Challenge,” International journal of computer vision, vol. 115, no. 3, Apr. 2015, s. 211-252.

[54] K. He, G. Gkioxari, P. Dollár vd. “Mask R-CNN,” in Proceedings of the IEEE international conference on computer vision, (s. 2961–2969), 2017.

[55] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, , 2016, s. 11371149.

[56] J. Redmon, S. Divvala, R. Girshick vd. “You Only Look Once: Unified, Real-Time Object Detection,” in Proceedings of the IEEE conference on computer vision and pattern recognition, (s. 779–788), 2016.

[57] R. Girshick, “Fast R-CNN,” in Proceedings of the IEEE international conference on computer vision, (s. 1440–1448), 2015.

[58] R. Girshick, J. Donahue, T. Darrell vd. “Rich feature hierarchies for accurate object detection and semantic segmentation,” in Proceedings of the IEEE conference on computer vision and pattern recognition, (s. 580-587), 2014.

[59] L. Mengxi, J. Yongfeng ve S. Jiuxu, “Research of Image Recognition and Classification Based on NIN Model,” CGDIP 2018, (s.1-6), 2020.

[60] A. Canziani, A. Paszke, ve E. Culurciello, “An Analysis of Deep Neural Network Models for Practical Applications,”, arXiv, May 2016, s. 1-7.

[61] G. Brockman, V. Cheung, L. Pettersson vd. “OpenAI Gym,” arXiv, 2016, s. 1-4.

[62] M. Chevalier-Boisvert vd. “BABYAI: A PLATFORM TO STUDY THE SAMPLE EFFI-CIENCY OF GROUNDED LANGUAGE LEARNING,” 2019, Erişim tarihi: 21 Haziran 2021. https://github.com/mila-iqia/babyai/tree/iclr19.

[63] J. D. Co-Reyes, A. Gupta, S. Sanjeev vd. “Guiding Policies with Language via Meta-Learning,” ICLR 2019, (s. 1–10), 2019.

[64] M. Mul, D. Bouchacourt, F. A. I. Research vd. “Mastering emergent language: learning to guide in simulated navigation,” arXiv:1908.05135, pp. 1–14, 2019, Erişim tarihi: 24 Haziran 2021. https://github.com/mila-udem/babyai.

[65] M. Hutsebaut-Buysse, K. Mets, ve S. Latré, “Fast Task-Adaptation for Tasks Labeled Using Natural Language in Reinforcement Learning,” arXiv:1910.04040, 2019, s. 1–10.

[66] A. Goyal, R. Islam, D. Strouse vd. “Infobot: Transfer and exploration via the information bottleneck,” ICLR 2019, (s. 1–21), 2019.

[67] M. Al-Shedivat, L. Lee, R. Salakhutdinov, and E. P. Xing, “On the Complexity of Exploration in Goal-Driven Navigation,” arXiv Prepr., vol. abs/1811.0, Nov. 2018, s. 1-7.

[68] R. Chourasia ve A. Singla, “Unifying Ensemble Methods for Q-learning via Social Choice Theory,” NeurIPS2019 LIRE Workshop, (s. 1–11), 2019.

[69] W. Shang, A. Trott, S. Zheng vd. “Learning World Graphs to Accelerate Hierarchical Reinforcement Learning,” arXiv:1907.00664v1, Jul. 2019, s. 1-13.

[70] A. Goyal, S. Sodhani, J. Binas vd. “Reinforcement Learning with Competitive Ensembles of Information-Constrained Primitives,” arXiv Prepr. arXiv1906.10667, Jun 2019, 1-21.

[71] D. Chen, Q. Yan, S. Guo vd.“Learning Effective Subgoals with Multi-Task Hierarchical Reinforcement Learning,” Scaling-Up Reinforcement Learning, (s. 1-7), 2019.

[72] R. Raileanu ve T. Rocktäschel, “RIDE: REWARDING IMPACT-DRIVEN EXPLORATION FOR PROCEDURALLY-GENERATED ENVIRONMENTS,” in ICLR 2020, (s. 1–21), 2020.

[73] A. Campero, R. Raileanu, J. B. Tenenbaum vd. “LEARNING WITH AMIGO: ADVERSARIALLY MOTIVATED INTRINSIC GOALS,” in ICLR 2021, (s. 1–19) 2021.

[74] H. Zhu, I. Ch. Paschalidis, M. E. Hasselmo vd. “Feature extraction in Q-learning using neural networks,” in 2017 IEEE 56th Annual Conference on Decision and Control (CDC), (s. 3330–3335), 2017.

[75] S. Phaniteja, P. Dewangan, P. Guhan vd. “A deep reinforcement learning approach for dynamically stable inverse kinematics of humanoid robots,” in 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), (s. 1818–1823), 2017.

[76] A. R. Sharma ve P. Kaushik, “Literature survey of statistical, deep and reinforcement learning in natural language processing,” in Proceeding - IEEE International Conference on Computing, Communication and Automation, ICCCA 2017, (s. 350-354), 2017.

[77] A. Kilin, P. Bozek, Y. Karavaev vd. “Experimental investigations of a highly maneuverable mobile omniwheel robot,” International Journal of Advanced Robotic Systems, vol. 14, no. 6, Aug. 2017, s. 1–9.

[78] M. O. Tătar, C. Popovici, D. Mândru vd. “Design and development of an autonomous omni-directional mobile robot with Mecanum wheels,” in 2014 IEEE International Conference on Automation, Quality and Testing, Robotics, (s. 1-6), 2014.

[79] Y. Kabalcı ve M. Ali, “Emerging LPWAN technologies for smart environments: An outlook,” in 1st Global Power, Energy and Communication Conference (GPECOM), (s. 24-29), 2019.

[80] E. Aras, G. S. Ramachandran, P. Lawrence vd. “Exploring the security vulnerabilities of LoRa,” in 2017 3rd IEEE International Conference on Cybernetics (CYBCONF), (s. 1-6), 2017.

[81] S. YASİNTİMUR ve V. TAVAS, “NB-IOT VE LORA HABERLEŞME TEKNOLOJİLERİNİN KARŞILAŞTIRMALI PERFORMANS ANALİZİ,” İstanbul Ticaret Üniversitesi Fen Bilim. Derg., vol. 20, no. 40, Dec. 2021, s. 297-313.

[82] K. Krinkin, E. Stotskaya, ve Y. Stotskiy, “Design and implementation Raspberry Pi-based omni-wheel mobile robot,” in 2015 Artificial Intelligence and Natural Language and Information Extraction, Social Media and Web Search FRUCT Conference (AINL-ISMW FRUCT), (s. 39–45), 2015.

[83] M. Tan, R. Pang ve Q. V. Le, “Efficientdet: Scalable and efficient object detection,” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, (s. 10781-10790), 2020.

[84] M. Tan ve Q. V Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” Proceedings of the 36th International Conference on Machine Learning, (s. 6105-6114), 2019.

[85] M. Moghaddam, M. Charmi, ve H. Hassanpoor, “Jointly human semantic parsing and attribute recognition with feature pyramid structure in EfficientNets,” IET Image Processing, 2021, vol. 15, no. 10, Mar. 2021, s.2281-2291.

[86] “TensorFlow 2 Detection Model Zoo,”, Erişim tarihi: 2 Haziran 2022. https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md

[87] “COCO - Common Objects in Context,”, Erişim tarihi: 2 Haziran 2022. https://cocodataset.org/

[88] Y. Zhou, J. Shi, X. Yang vd. “Rotational Objects Recognition and Angle Estimation via Kernel-Mapping CNN,” IEEE Access, vol. 7, Aug. 2019, s. 116505–116518.

[89] M. Elad, “On the origin of the bilateral filter and ways to improve it,” IEEE Transactions on Image Processing, vol. 11, no. 10, Oct. 2002, s. 1141–1151.

[90] M. Pfeiffer, S. Shukla, M. Turchetta vd. “Reinforced Imitation: Sample Efficient Deep Reinforcement Learning for Mapless Navigation by Leveraging Prior Demonstrations,” IEEE Robotics and Automation Letters, vol. 3, no. 4, Sep. 2018, s. 4423 - 4430

[91] S.-H. Hsu, S.-H. Chan, P.-T. Wu vd. “Distributed Deep Reinforcement Learning Based Indoor Visual Navigation,” in International Conference on Intelligent Robots and Systems (IROS), (s. 2532–2537), 2018.

[92] P. Long, T. Fan, X. Liao vd. “Towards Optimally Decentralized Multi-Robot Collision Avoidance via Deep Reinforcement Learning,” in 2018 IEEE International Conference on Robotics and Automation (ICRA), (s. 6252-6259), 2018.

[93] C. Wang, J. Wang, Y. Shen, and X. Zhang, “Autonomous Navigation of UAVs in Large-Scale Complex Environments: A Deep Reinforcement Learning Approach,” IEEE Transactions on Vehicular Technology, vol. 68, no. 3, Mar. 2019, s. 2124–2136.

[94] J. Lin, X. Yang, P. Zheng vd. “End-to-end Decentralized Multi-robot Navigation in Unknown Complex Environments via Deep Reinforcement Learning,” 2019 IEEE International Conference on Mechatronics and Automation (ICMA), (s. 2493-2500), 2019.

[95] C. Wang, J. Wang, J. Wang vd. “Deep-Reinforcement-Learning-Based Autonomous UAV Navigation With Sparse Rewards,” IEEE Internet of Things Journal, vol. 7, no. 7, Feb. 2020, s. 6180-6190.

[96] F. Leiva ve J. Ruiz-Del-Solar, “Robust RL-Based Map-Less Local Planning: Using 2D Point Clouds as Observations,” IEEE Robotics and Automation Letters, vol. 5, no. 4, Jul 2020, s. 5787 - 5794.

[97] Q. M. Qadir, T. A. Rashid, N. K. Al-Salihi vd. “Low power wide area networks: A survey of enabling technologies, applications and interoperability needs,” IEEE Access, vol. 6, Nov. 2018, s. 77454–77473.

[98] M. Shahjalal, M. K. Hasan, M. M. Islam vd. “An Overview of AI-Enabled Remote Smart- Home Monitoring System Using LoRa,” in 2020 International Conference on Artificial Intelligence in Information and Communication, ICAIIC 2020, (s. 510–513), 2020.

[99] G. Sartoretti, J. Kerr, Y. Shi vd. “PRIMAL: Pathfinding via Reinforcement and Imitation Multi-Agent Learning,” IEEE Robotics and Automation Letters, vol. 4, no. 3, Jul. 2019, s. 2378–2385.

[100] J. Lin, X. Yang, P. Zheng vd. “End-to-end Decentralized Multi-robot Navigation in Unknown Complex Environments via Deep Reinforcement Learning,” ICMA 2019, (s. 2493–2500), 2019.

[101] T. Fan, P. Long, W. Liu vd. “Fully Distributed Multi-Robot Collision Avoidance via Deep Reinforcement Learning for Safe and Efficient Navigation in Complex Scenarios,” arXiv Prepr. arXiv1808.03841, Aug. 2018, s.1-30.

[102] A. Yahya, A. Li, M. Kalakrishnan vd.“Collective robot reinforcement learning with distributed asynchronous guided policy search,” in IEEE International Conference on Intelligent Robots and Systems, (s. 79–86), 2017.

ÖZGEÇMİŞ

Kişisel Bilgiler		
Soyad, Ad		ERKAN, Emre
Web sayfası		https://www.researchgate.net/profile/Emre-Erkan
Eğitim Bilgileri		
Derece	Kurum	Mezuniyet Yılı
Yüksek Lisans	Kahramanmaraş Sütçü İmam Üniversitesi, Elektrik-Elektronik Mühendisliği ABD	2015
Lisans	Mustafa Kemal Üniversitesi, Elektrik-Elektronik Mühendisliği	2008
Lise	Ahlat Selçuklu Lisesi	2002
İş Deneyimi		
Dönem (Yıl)	Şirket, Kurum	Görev
2009-2011	Batman Üniversitesi	Mühendis
2011-Halen	Batman Üniversitesi	Öğretim Görevlisi
Yabancı Dil		
İngilizce		
Yayınlar		
1. E. Erkan ve M. A. Arserim, "Mobile Robot Application with Hierarchical Start Position DQN," Computational Intelligence and Neuroscience, vol. 2022, Sep. 2022, s. 1–21,		

DİCLE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
TEZ BENZERLİK BİLDİRİMİ FORMU

Öğrencinin Adı, Soyadı	Emre ERKAN		
Öğrenci No	17805601		
Ana Bilim Dalı	Elektrik Elektronik Mühendisliği		
Program Türü	Proje ☐	Yüksek Lisans ☐	Doktora ☒
Tez Danışmanı (Ünvanı, Adı, Soyadı)	Dr. Öğr. Üyesi Muhammet Ali ARSERİM		
Tez Başlığı	Hiyerarşik Başlangıç Pozisyonlu Derin Q-Ağı Algoritması ile Mobil Robot Uygulaması		
RAPOR BİLGİLERİ			
Raporlama Aşaması	Tez Savunma Sınavı Sonrası		
Sayfa Sayısı	103		
Raporlama Tarihi	17/11/2022		
Benzerlik Oranı (%)	7		

Yukarıda bilgileri verilen tez çalışmamın toplam 103 sayfalık kısmına ilişkin, 17/11/2022 tarihinde tez danışmanım tarafından *Turnitin* isimli intihal tespit programından aşağıda belirtilen filtrelemeler uygulanarak alınmış olan intihal raporuna göre, tezimin benzerlik oranı %7 olarak tespit edilmiştir.

Uygulanan filtrelemeler:

☐Başlangıç Bölümleri (Kabul ve Onay sayfası, Teşekkür sayfası, Özet/Abstract) hariç

☒Kaynaklar hariç

☐Alıntılar hariç/dâhil

☐Diğer (Açıklayınız)

Tezimin benzerlik oranı, Dicle Üniversitesi Fen Bilimleri Enstitüsü İntihal Raporu Uygulama Esaslarında belirtilen üst sınır benzerlik oranını aşmamaktadır. Benzerlik oranım üst sınır benzerlik oranının altında olsa dahi aksinin tespit edilmesi durumunda her türlü yasal sorumluğu kabul ettiğimi ve hukuki sonuçlarına razı olduğumu bildirir,
gereğini arz ederim.

Öğrencinin Adı, Soyadı: Emre ERKAN
Tarih:17.11.2022
İmza:

Danışman Adı, Soyadı: Dr. Öğr. Üyesi Muhammet Ali ARSERİM İmza:

Tarih: 17.11.2022

Ana Bilim Dalı Başkanı Adı, Soyadı: Doç. Dr. Bilal GÜMÜŞ İmza:

Tarih: 17.11.2022

Ek: Benzerlik Nihai Raporu ilk sayfa çıktısı