

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MAKİNA MÜHENDİSLİĞİ ANABİLİM DALI

**KABLO İLE SÜRÜLEN EKSİK SINIRLANDIRILMIŞ ORTAMSAL BİR
PARALEL ROBOTUN PEKİŞTİRMELİ ÖĞRENME İLE YÖRÜNGE
PLANLAMASI**

YÜKSEK LİSANS TEZİ

Kadir İbrahim ERTÜRK

TEMMUZ 2025
TRABZON



KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MAKİNA MÜHENDİSLİĞİ ANABİLİM DALI

**KABLO İLE SÜRÜLEN EKSİK SINIRLANDIRILMIŞ ORTAMSAL BİR
PARALEL ROBOTUN PEKİŞTİRMELİ ÖĞRENME İLE YÖRÜNGE
PLANLAMASI**

Kadir İbrahim Ertürk

Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsünde
"MAKİNA YÜKSEK MÜHENDİSİ"
Unvanı Verilmesi İçin Kabul Edilen Tezdir.

Tezin Enstitüye Verildiği Tarih : 03 / 07 / 2025

Tezin Savunma Tarihi : 18 / 07 / 2025

Tez Danışmanı : Dr. Öğr. Üyesi Caner SANCAK

Trabzon 2025

ÖNSÖZ

“Kablo ile Sürülen Eksik Sınırlandırılmış Ortamsal Bir Paralel Robotun Pekiştirmeli Öğrenme ile Yörünge Planlaması” isimli bu tez, Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü Makina Mühendisliği Anabilim Dalı, Yüksek Lisans Programı’nda hazırlanmıştır.

Tez süreci boyunca desteğini ve bilgisini hiçbir zaman esirgemeyen, tecrübeleriyle bana her daim yol gösteren değerli danışmanım Dr. Öğr. Üyesi Caner SANCAK’a teşekkürü bir borç bilirim. Ayrıca çalışmalarım sırasında kıymetli katkıları ve yol göstericiliğiyle yanımda olan Dr. Öğr. Üyesi Mustafa Yavuz COŞKUN’a teşekkür ederim.

Hayatım boyunca elimi hiç bırakmayan, her kararında bilgi ve tecrübeleriyle bana destek olan, maddi ve manevi desteğini daima hissettiğim ve bugünlere gelmemde en büyük paya sahip olan sevgili annem Tülay ERTÜRK’e ve babam Adem ERTÜRK’e, kardeşlerim Kerem Akif ERTÜRK ve Selim Erdem ERTÜRK’e minnettarlığımı sunarım.

Bu tez çalışmasının bundan sonraki araştırmalara katkı sağlamasını temenni ederim.

Kadir İbrahim ERTÜRK
Trabzon 2025

TEZ ETİK BEYANNAMESİ

Yüksek Lisans Tezi olarak sunduğum “Kablo ile Sürülen Eksik Sınırlandırılmış Ortamsal Bir Paralel Robotun Pekiştirmeli Öğrenme ile Yörünge Planlaması” başlıklı bu çalışmayı baştan sona kadar danışmanım Dr. Öğr. Üyesi Caner SANCAK’ın sorumluluğunda tamamladığımı, verileri ve örnekleri kendim topladığımı, deneyleri ve analizleri ilgili laboratuvarlarda yaptığımı, başka kaynaklardan aldığım bilgileri metinde ve kaynakçada eksiksiz olarak gösterdiğimi, çalışma sürecinde bilimsel araştırma ve etik kurallara uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul ettiğimi beyan ederim. 18/07/2025

Kadir İbrahim ERTÜRK

İÇİNDEKİLER

	<u>Sayfa No</u>
ÖNSÖZ	III
TEZ ETİK BEYANNAMESİ.....	IV
İÇİNDEKİLER.....	V
ÖZET	VIII
SUMMARY	IX
ŞEKİLLER DİZİNİ	X
TABLolar DİZİNİ.....	XIII
SEMBOLLER VE KISALTMALAR DİZİNİ	XIV
1. GİRİŞ.....	1
1.1. Paralel Robotlar.....	2
1.2. Tez İçeriği.....	5
1.3. Kabloyla Sürülen Paralel Robotlar.....	6
1.3.1. KSPR Uygulama Alanları.....	7
1.3.2. Kinematik Model	9
1.4. Literatür Özeti	11
1.5. Makine Öğrenmesi	16
1.5.1. Derin Öğrenme.....	17
1.5.2. Pekiştirmeli Öğrenme	18
1.5.3. Markov Karar Süreci.....	19
1.5.4. Q Öğrenme	21
1.5.5. Derin Sinir Ağları.....	23
1.6. Derin Q Ağı	27
1.6.1. Epsilon-Greedy Dengesi	29
1.6.2. Ödül Fonksiyonunun Seçimi.....	31
1.6.3. Hiperparametrelerin Seçimi	32
1.6.4. Gözlemlerin Belirlenmesi	33
1.6.5. Azaltma Faktörü Seçimi.....	33
1.6.6. Öğrenme Oranı Seçimi.....	34
1.6.7. Hedef Ağ Güncelleme Sıklığı ve Yumuşatma Faktörü	36
2. YAPILAN ÇALIŞMALAR.....	38
2.1. Ortamsal KSPR Tasarımı ve Deney Düzenegi Kurulumu	38

2.1.1. Uç İşlemci Bağlantısı, Tasarımı ve Kontrolü.....	42
2.1.2. Ortamsal KSPR 'nin Kinematik Modeli	44
2.1.3. Kamera Kalibrasyonu	48
2.2. Pekiştirmeli Öğrenme Ortamının Oluşturulması.....	49
2.3. Engelsiz Benzetim Ortamında Yol Planlama.....	53
2.3.1. Kullanılan Gözlemler, Ödül Fonksiyonu ve Sınırlamalar	53
2.4. Küresel Engel Yaklaşımı.....	55
2.4.1. Sabit Çaplı Küresel Engel İçeren Benzetim Ortamında Yol Planlama.....	56
2.5. Pekiştirmeli Öğrenme Ajanı ile Deneysel Konumlama	58
2.5.1. Değişken Çaplı Küresel Engel İçeren Benzetim Ortamında Yol Planlama.....	58
2.5.2. Görüntü İşleme ile Öznitelik Çıkarma.....	60
2.6. KSPR ile Tut-Yerleştir Uygulaması.....	66
2.7. Hiperparametre Optimizasyonu	68
2.7.1. Sabit Çaplı Küresel Engelden Kaçınma Uygulaması İçin Hiperparametre Optimizasyonu	69
2.7.2. Değişken Çaplı Küresel Engelden Kaçınma Uygulaması İçin Hiperparametre Optimizasyonu	70
3. BULGULAR.....	72
3.1. Engelsiz Benzetim Ortamında Pekiştirmeli Öğrenme Eğitimi Bulguları	72
3.2. Sabit Çaplı Küresel Engel İçin Hiperparametre Optimizasyonun Eğitim Sonuçlarına Etkisi.....	74
3.2.1. Öğrenme Oranı Optimizasyonu	74
3.2.2. Hedef Ağ Güncelleme Sıklığı Optimizasyonu.....	78
3.2.3. Azaltma Faktörü Optimizasyonu	83
3.2.4. Hiperparametre Optimizasyonunun Başarımı.....	87
3.3. Değişken Çaplı Küresel Engel İçin Hiperparametre Optimizasyonun Eğitim Sonuçlarına Etkisi.....	89
3.3.1. Öğrenme Oranı Optimizasyonu	89
3.3.2. Hedef Ağ Güncelleme Sıklığı Optimizasyonu.....	92
3.3.3. Azaltma Faktörü Optimizasyonu	94
3.4. Tut-Yerleştir Uygulaması Bulguları.....	99
4. SONUÇLAR.....	102
5. ÖNERİLER.....	104

6. KAYNAKLAR	106
ÖZGEÇMİŞ	



Yüksek Lisans Tezi

ÖZET

KABLO İLE SÜRÜLEN EKSİK SINIRLANDIRILMIŞ ORTAMSAL BİR PARALEL ROBOTUN PEKİŞTİRMELİ ÖĞRENME İLE YÖRÜNGE PLANLAMASI

Kadir İbrahim ERTÜRK

Karadeniz Teknik Üniversitesi
Fen Bilimleri Enstitüsü
Makina Mühendisliği Anabilim Dalı
Danışman: Dr. Öğr. Üyesi Caner SANCAK
2025, 110 Sayfa

Bu tezde, kablo ile sürülen eksik sınırlandırılmış ortamsal bir paralel robotun uç işlemcisinin ve kablolarının çalışma ortamındaki engele çarpmadan hareket etmesini sağlamak amacıyla, bir pekiştirmeli öğrenme yöntemi olan Derin Q Ağı kullanılarak yörünge planlaması yapılmış ve deneysel olarak uygulanmıştır. Bu amaçla yapılan çalışmalarda, benzetim ortamında oluşturulan robot modeli üzerinde pekiştirmeli öğrenme eğitimleri gerçekleştirilmiş, eğitimler sonucu elde edilen pekiştirmeli öğrenme ajanı ile uç işlemci ve kabloların engele temas etmeden hedefe ulaşması sağlanmıştır. Benzetim ortamında eğitilen yapay sinir ağları, geliştirilen deneysel KSPR sistemde kullanılmış ve böylece gerçek ortam performansları test edilmiştir. Gerçek ortamdaki objelerin konum ve boyut bilgileri, bir RGB-D kameradan alınan derinlik ve renk verilerinin işlenmesiyle elde edilmiş; bu veriler pekiştirmeli öğrenme ajanına gözlem olarak verilmiştir. Yapılan bu testlerde ajan, engelsiz ortamda %98, engelli ortamda ise %80 başarımla çarpışma olmadan hedefe ulaşmıştır. Ek olarak engelli ortam için eğitilen ajan, örnek bir tut-yerleştir senaryosu üzerine uygulanarak test edilmiş; böylece sistemin uygulamalarda kullanılabilirliği ve gerçek görev koşullarına uyarlanabilirliği ortaya konmuştur.

Anahtar Kelimeler: Kablo İle Sürülen Paralel Robot (KSPR), Pekiştirmeli Öğrenme (PÖ), Yol Planlama, Tut-Yerleştir Uygulaması, Otonom Robot

Master Thesis

SUMMARY

TRAJECTORY PLANNING WITH REINFORCEMENT LEARNING FOR A SPATIAL UNDER-CONSTRAINED CABLE-DRIVEN PARALLEL ROBOT

Kadir İbrahim ERTÜRK

Karadeniz Technical University
The Graduate School of Natural and Applied Sciences
Mechanical Engineering Sciences Graduate Program
Supervisor: Asst. Prof. Dr. Caner SANCAK
2025, 110 Pages

In this thesis, trajectory planning is carried out using a reinforcement learning method, the Deep Q-Network (DQN), to ensure that the end-effector and cables of an under-constrained spatial cable-driven parallel robot move in the workspace without colliding with obstacles. For this purpose, reinforcement learning training is performed on the robot model created in the simulation environment, and the end-effector and cables reach the target without contacting obstacles using the reinforcement learning agent obtained as a result of the training. The artificial neural networks trained in the simulation environment are used on the developed experimental CDPR system, thus testing their performance in the real environment. The position and size information of real-world objects is obtained by processing the depth and color data from an RGB-D camera, and these data are provided to the reinforcement learning agent as observations. In these tests, the agent reaches the target without collision with a success rate of 98% in an obstacle-free environment and 80% in an environment with an obstacle. In addition, the agent trained for the obstacle environment is applied to a sample pick-and-place scenario, thereby demonstrating the applicability of the system in real tasks and its adaptability to actual operating conditions.

Key Words: Cable-Driven Parallel Robot (CDPR), Reinforcement Learning (RL), Path Planning, Image Processing, Pick and Place, Autonomous Robot

ŞEKİLLER DİZİNİ

	<u>Sayfa No</u>
Şekil 1. Seri ve Paralel Robot.....	2
Şekil 2. a) Uçuş simülatörü b) Sürüş simülatörü c) Çok eksenli malzeme deneyi d) Deprem simülatörü	3
Şekil 4. Delta robot çalışma alanı.....	4
Şekil 5. Kabloyla Sürülen Paralel Robot.....	5
Şekil 6. (a) Düzlemsel KSPR, (b) Ortamsal KSPR.....	6
Şekil 7. (a) RoboCrane, (b) SkyCam, (c) FAST Radyo Teleskopu h, (d) Çoklu Vinç Sistemi, (e)Kabloyla Sürülen Kol Rehabilitasyon Robotu, (f) SPIDERobot, (g) Kabloyla Sürülen 3B Yazıcı, (h) Kabloyla Sürülen Hareket Simülatörü	8
Şekil 8. KSPR Genel Kinematik Yapı.....	9
Şekil 9. PÖ yapısı ve ajanın çevreyle etkileşimi	18
Şekil 10. Aktivasyon Fonksiyonları	24
Şekil 11. Yapay sinir ağı güncelleme işlemleri	25
Şekil 12. DQA Algoritması (URL-9, 2025)	28
Şekil 13. Deney düzeneği fiziki kurulumu	38
Şekil 14. Deney düzeneği şematik kurulum.....	39
Şekil 15. Endüstriyel servo motor ve motor sürücülerini	40
Şekil 16. Simulink/Realtime servo motor kontrolü	41
Şekil 17. Pekiştirmeli öğrenme ajanının entegrasyonu	41
Şekil 18. Tutucu Uç Ölçüleri.....	43
Şekil 19. (a) Arduino geliştirici kart, (b) mini servo motor.....	44
Şekil 20. Dört kablo ile sürülen KSPR ve kinematik modeli	46
Şekil 21. Matlab/Simulink PÖ Ortam Kurulumu	50
Şekil 22. Birim küp üzerindeki aksiyon yönleri ve aksiyon vektörü	52
Şekil 23. Aksiyon adım aralığına bağlı çalışma uzayı çözünürlüğü.....	52
Şekil 24. Küresel engel yaklaşımı ve güvenli alan.....	55
Şekil 25. Renk filtresi uygulanmamış nokta bulutu	61
Şekil 26. Mavi renkteki objeye göre uygulanmış filtre	61
Şekil 27. Yeşil renge göre uygulanmış filtre	62
Şekil 28. Kırmızı renge göre uygulanmış filtre	63
Şekil 29. Dikdörtgen prizma engelin, yeşil dairesel objenin ve düzlemsel referans noktasının küre içine alınması	64

Şekil 30. Dairesel engelin, yeşil dairesel objenin ve düzlemsel referans noktasının küre içine alınması.....	65
Şekil 31. Sanal kürenin engelin tamamını kapsayamaması.....	65
Şekil 32. Sanal kürenin engelin tamamını kapsaması	66
Şekil 33. Otonom tut-yerleştir uygulamasında kullanılan Simulink/Realtime kontrol yapısı.....	67
Şekil 34. Noktadan Noktaya Engelsiz PÖ Ajan Eğitimi	72
Şekil 35. Rastgele başlangıç ve referans konumlarıyla yapılan denelemler.....	73
Şekil 36. Deney 1, Öğrenme Oranı: $2 * 10 - 4$	74
Şekil 37. Deney 2, Öğrenme Oranı: $3 * 10 - 4$	75
Şekil 38. Deney 3, Öğrenme Oranı: $4 * 10 - 4$	76
Şekil 39. Deney 4, Öğrenme Oranı: $5 * 10 - 4$	76
Şekil 40. Deney 5, Öğrenme Oranı: $6 * 10 - 4$	77
Şekil 41. Deney 6, Öğrenme Oranı: $1 * 10 - 3$	77
Şekil 42. Deney 7, Hedef Ağ Güncelleme Sıklığı: Her adımda (1)	78
Şekil 43. Deney 8, Hedef Ağ Güncelleme Sıklığı: 3 adımda 1 defa	79
Şekil 44. Deney 9, Hedef Ağ Güncelleme Sıklığı: 5 adımda 1 defa	80
Şekil 45. Deney 10, Hedef Ağ Güncelleme Sıklığı: 10.....	81
Şekil 46. Deney 11, Hedef Ağ Güncelleme Sıklığı: 20 adımda 1 defa	82
Şekil 47. Deney 12, Hedef Ağ Güncelleme Sıklığı: 50 adımda 1 defa	82
Şekil 48. Deney 13, Azaltma faktörü $\gamma=0.999$	84
Şekil 49. Deney 14, Azaltma faktörü $\gamma=0.996$	85
Şekil 50. Deney 15, Azaltma faktörü $\gamma=0.993$	85
Şekil 51. Deney 16, Azaltma faktörü $\gamma=0.990$	86
Şekil 52. Deney 17, Azaltma faktörü $\gamma=0.987$	87
Şekil 53. Noktadan noktaya sabit çaplı engel benzetimi konum sonuçları	88
Şekil 54. Noktadan noktaya sabit çaplı engel benzetiminin 3B görselleştirilmesi.....	88
Şekil 55. Deney 7, Öğrenme oranı $1 * 10 - 3$	89
Şekil 56. Deney 8, Öğrenme Oranı $5 * 10 - 3$	90
Şekil 57. Deney 9, Öğrenme oranı $5 * 10 - 4$	91
Şekil 58. Deney 10, Öğrenme oranı $1 * 10 - 5$	91
Şekil 59. Deney 11, Hedef ağın 320 adımda bir güncellenmesi.....	92
Şekil 60. Deney 12, Hedef ağın 960 adımda bir güncellenmesi.....	93
Şekil 61. Deney 13, Hedef ağın 1280 adımda bir güncellenmesi.....	93

Şekil 62. Deney 1, Azaltma faktörü $\gamma=0.99$	94
Şekil 63. Deney 2, Azaltma faktörü $\gamma=0.95$	95
Şekil 64. Deney 3, Azaltma faktörü $\gamma=0.90$	95
Şekil 65. Deney 4, Azaltma faktörü $\gamma=0.85$	96
Şekil 66. Deney 5, Azaltma faktörü $\gamma=0.80$	97
Şekil 67. Deney 6, Azaltma faktörü $\gamma=0.75$	97
Şekil 68. Uç işlemcinin engele çarpmadan topa yönelmesi ve beklemesi	100
Şekil 69. Uç işlemcinin engele çarpmadan topu referansa götürmesi ve bırakması	100



TABLÖLAR DİZİNİ

	<u>Sayfa No</u>
Tablo 1. Matlab DQA algoritması sözde kodu	51
Tablo 2. Sabit engel çaplı eğitim için uygulanan hiperparametreler	70
Tablo 3. Değişken çaplı eğitim için uygulanan hiperparametreler	71



SEMBOLLER VE KISALTMALAR DİZİNİ

$\|l_i\|$: Kablo uzunlukları

d_i : Kabloların engel yüzeyine dik uzaklıkları

$\bar{Q}$: Uç işlemci parmak merkezi

X_i : Kablolar üzerinde engele en yakın nokta konumu

θ^- : Hedef ağ

R_t : Markov Karar Sürecinde Ödül

S_t : Markov Karar Sürecinde Durum

$\frac{d}{dt}$: Türev işlemi

D_P^O : Dönüşüm matrisi

b_i : Kabloların sabit gövdeye bağlantı konumları

l_i : Kablo vektörleri

$\bar{q}$: Uç işlemci merkezinden tutucu parmak merkezine vektör

q_i^P : A_i noktalarının $\{P\}$ 'ye göre konumları

θ_i : Motor dönüş açısı

$\{O\}$: Referans koordinat sistemi

$\{P\}$: Yerel koordinat sistemi

A : Aksiyonlar kümesi

a : Mevcut aksiyon

A_i : Kabloların hareketli platforma bağlantı noktaları

b : Bias değeri

B_i : Kabloların sabit gövdeye bağlantı noktaları

C : Tambur çevresi

D : Deneyim tekrarlama belleği

D : Tambur çapı

DDPG: Deep Deterministic Policy Gradient Algoritması

DQA: Derin Q Ağı Algoritması

e : Çevre modeli

e : Engel merkezi konum vektörü

IR: Infrared

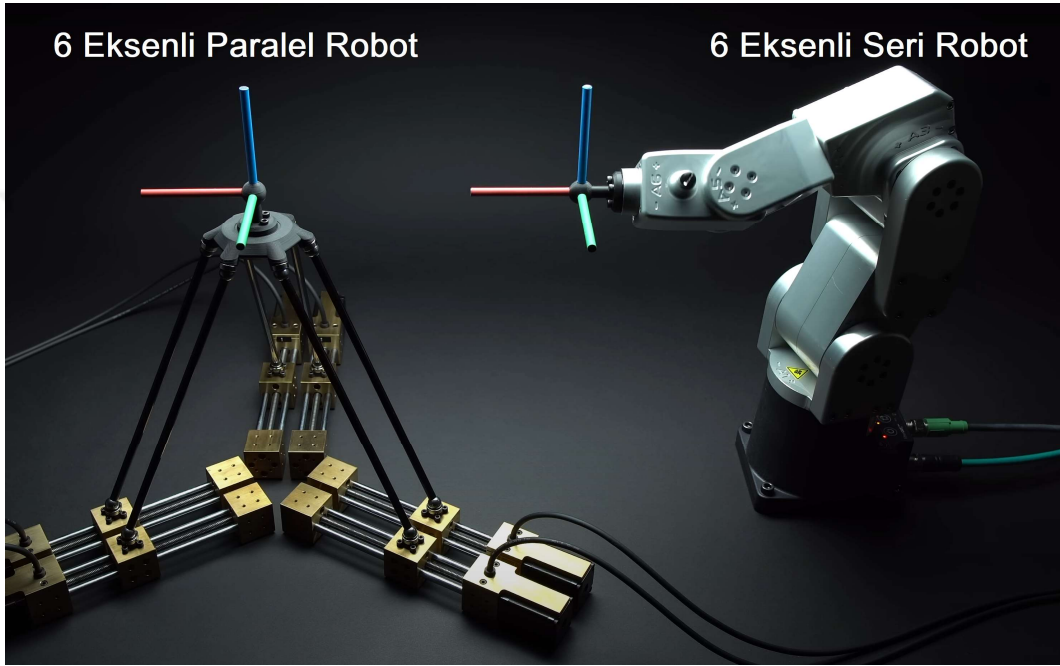
j : Piksel yoğunluk merkezi

k: Uç işlemci merkezinden engel merkezine vektör
MKS: Markov Karar Süreci
 $P(s'/s,a)$: Geçiş olasılık fonksiyonu
P: Kablo bağlantı noktası
p: Uç işlemci merkez konum vektörü
PÖ: Pekiştirmeli Öğrenme
q: Kablo bağlantı noktasından uç işlemci merkezine vektör
Q: Küresel uç işlemci merkezi
R: Dönme matrisi
r: Ödül
r: Referans merkezi konum vektörü
RGB-D: Kırmızı Yeşil Mavi Renk ve Derinlik
S: Durumlar kümesi
s: Mevcut durum
 s' : Bir sonraki durum
SAC: Soft Actor Critic Algoritması
u: Uç işlemci parmak merkezinden referans merkezine vektör
UDP: User Datagram Protocol
v: Değer fonksiyonu
w: Nöron ağırlık değeri
x: X eksen
y: Y eksen
YSA: Yapay Sinir Ağı
z: Z eksen
 γ : Azaltma faktörü
 ϵ : Epsilon olasılık değeri
 θ : Ana ağ
 π : Politika
 α : Öğrenme oranı
 δ : Kamera derinlik verisi hata oranı
 δt : Zamansal fark hatası
 τ : Yumuşatma faktörü

1. GİRİŞ

Otomasyon fikrinin tarihsel kökenleri Antik Yunan'a ve hatta Mezopotamya gibi daha eski uygarlıklara kadar uzanır. Otomatlar ve mekanik sistemlere dair ilk çalışmalar; insan müdahalesini azaltarak belirli görevleri makinelerin kendi kendine gerçekleştirebilmesini sağlamak, doğadaki canlıların hareketlerini taklit etmek ve zaman zaman da eğlence amacı gütmek üzere yapılmıştır. Zaman ilerledikçe teknolojik gelişmelerin etkisiyle ve ihtiyaçlar doğrultusunda robotik kavramı ortaya çıkmıştır. Günümüzde robotik bilimi; endüstriyel otomasyon, tıp, savunma sanayisi, tarım, keşif sistemleri ve hizmet sektörü gibi çok geniş kullanım alanlarına sahiptir. Bu alanlardaki hızlı gelişimler, robotik teknolojilere olan ihtiyacı günden güne artırmakta ve bu durum sistemlerin çok disiplinli bir mühendislik yaklaşımıyla ele alınmasını zorunlu kılmaktadır (Kurfess, 2005). Bu sebeple robotik bilimi; makine mühendisliğini, elektrik-elektronik mühendisliğini, bilgisayar mühendisliğini, kontrol mühendisliğini ve yazılım mühendisliğini bir arada barındıran çok disiplinli ve mühendislik temelli bir bilimsel alan haline gelmiştir. Robot kavramı, belirli bir görevi gerçekleştirmek üzere tasarlanan; mekanik, elektrik ve elektronik bileşenlerden oluşan, tamamen otomatik, yarı otomatik ya da insan müdahalesiyle çalışabilen sistemleri ifade eder. Robotlar; kullanım amaçları, hareket ve yük taşıma kabiliyetleri, çalışma alanları, hassasiyet, tekrarlanabilirlik ve çalışma hızları gibi ihtiyaçlar göz önüne alınarak tasarlanmakta ve üretilmektedirler. Robotların belirtilen temel özelliklerini belirleyen en önemli faktör ise kinematik yapılarıdır. Kinematik yapılarına göre robotlar genellikle seri ve paralel robotlar olmak üzere iki ana gruba ayrılır. Şekil 1'de görüldüğü gibi seri robotlar; genellikle sabit bir tabana bağlı, rijit ve birbirine ardışık olarak eklenmiş uzuvlardan oluşmaktadırlar. Çoğu zaman 6 serbestlik derecesine sahip olup hassas görevlerde yaygın olarak tercih edilirler ve endüstride en sık kullanılan robot tipidirler. Ancak bu yapıları, yüksek atalet nedeniyle hareket hızında kısıtlamalara yol açar ve bağlantıların uç uca olması, küçük mafsallı hatalarının uç işlemcide birikerek büyük konum hatalarına dönüşmesine ve ayrıca istenmeyen titreşimlere neden olur. Paralel robotlar ise yine Şekil 1'de gösterilmiş olup uç işlemcinin birden fazla bağlantı ile desteklendiği kapalı kinematik zincir yapısına sahiptirler. Bu yapı sayesinde sistemin genel yük taşıma kapasitesi daha yüksektir ve hata miktarı daha azdır. Böylece uç işlemci konumu hassas bir şekilde kontrol edilebilir. Uzuvları hareket ettiren eyleyiciler genellikle sabit bir platformda bulunduklarından dolayı hareketli parçalar

daha hafiftir. Bu yapı, robotun daha hızlı hareket etmesine ve yüksek ivmelere ulaşmasına imkân tanır. Ayrıca sistemin rijitliğinin yüksek olmasına, titreşimlerin daha az olmasına ve dinamik davranışların daha kararlı olmasına katkı sağlar. Bu özellikler sayesinde paralel robotlar; hassasiyetin, hızın ve yük taşımanın önemli olduğu uygulamalarda tercih edilir. Tüm bu avantajlarının yanında paralel yapılar genellikle daha karmaşık bir mekanik tasarıma sahiptir (Tsai, 1999).



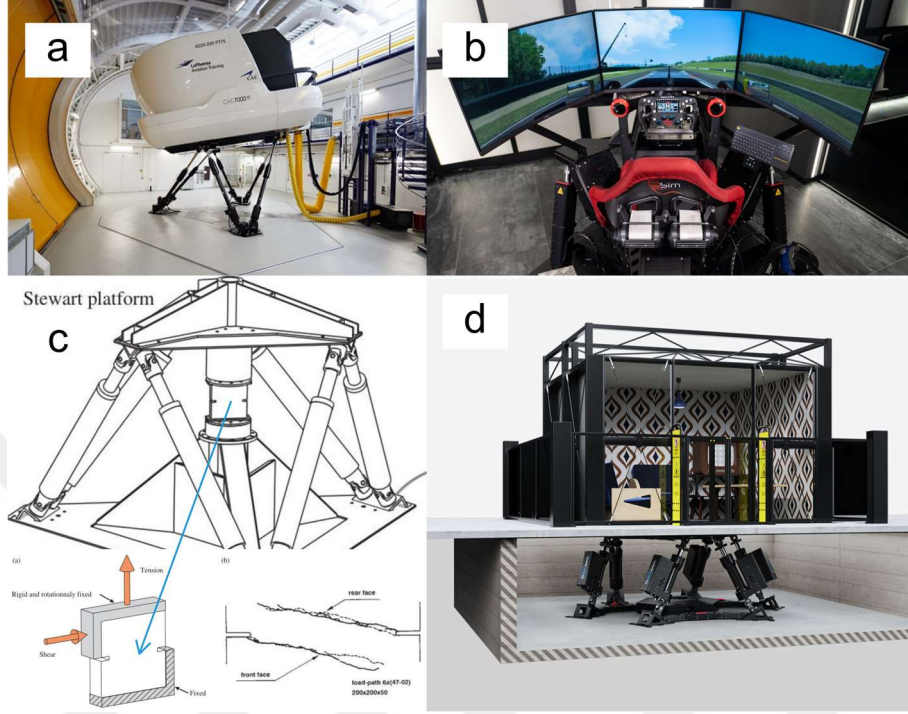
Şekil 1. Seri ve paralel robot (URL-8, 2025)

1.1. Paralel Robotlar

Paralel robotlar genellikle çok uzuvlu kapalı kinematik zincirlerle sabit bir tabana bağlı hareketli platformdan oluşurlar ve uzuvların aynı anda aktifleştirilmesi ile uç işlemciyi hareket ettirirler. Uç işlemciyi hareket ettirmek için tüm uzuvların senkronize olarak hareket etmesi oldukça önemlidir. Ayrıca yüksek rijitliğe sahip olup hız ve hassasiyet gerektiren uygulamalar için oldukça uygundurlar (Merlet, 2005).

Paralel robotların en bilinen örneği olan Stewart platformu, sabit bir gövdeye 6 adet değişken uzunluklu uzvun küresel mafsallarla bağlandığı bir platformdan oluşur. 6 serbestlik derecesine ve yüksek yük taşıma kapasitesine sahiptirler. Konum ve yönelim olarak hassas şekilde kontrol edilebilirler. Bu özellikleri sayesinde Şekil 20’de gösterildiği gibi uçuş, sürüş

ve deprem simülatörlerinde, endüstriyel uygulamalarda ve çok eksenli yükleme gerektiren malzeme deneyleri (Nierenberger vd., 2014) gibi birçok alanda kullanılmaktadırlar.

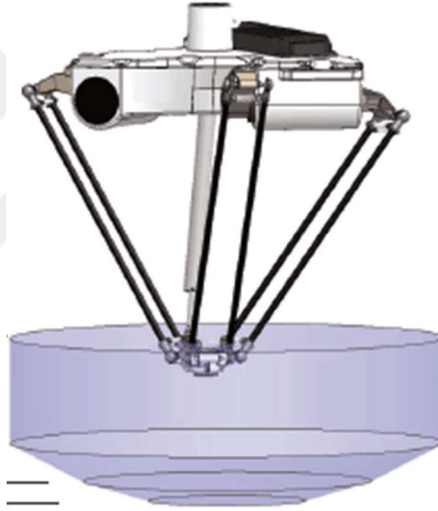


Şekil 2. a) Uçuş simülatörü (URL-1, 2025), b) Sürüş simülatörü (URL-2, 2025), c) Çok eksenli malzeme deneyi (URL-3, 2025) d) Deprem simülatörü (URL-4, 2025)

Şekil 3’te gösterilen paralel robot örneği, delta robot yapısına ve sabit tabana bağlı üç döner eyleyici tarafından kontrol edilen simetrik uzuvlara sahiptir. Eyleyici, sabit bir uç platforma bağlı olan kısa döner kolları hareket ettirirler. Uzuvların paralelogram yapıya sahip olması, uç platformun her zaman zemine paralel olarak hareket etmesine sebep olur. Bu sebeple sadece üç öteleme serbestlik derecesine sahiptir (Laribi vd., 2007). Yapılarında düşük atalete sahip uzuvlar tercih edilir ve genellikle yüksek hızlı tut-yerleştir uygulamalarına uygun olduklarından endüstride en çok kullanılan paralel robot türlerinden biridirler. Fakat Şekil 4’te görüldüğü gibi çalışma uzayları kısıtlı olup yüksek hızlar için tasarlandıklarından ağır yükler taşıyamamaları ve sadece üç öteleme serbestlik derecesine sahip olmaları dezavantajları olarak sıralanabilir.

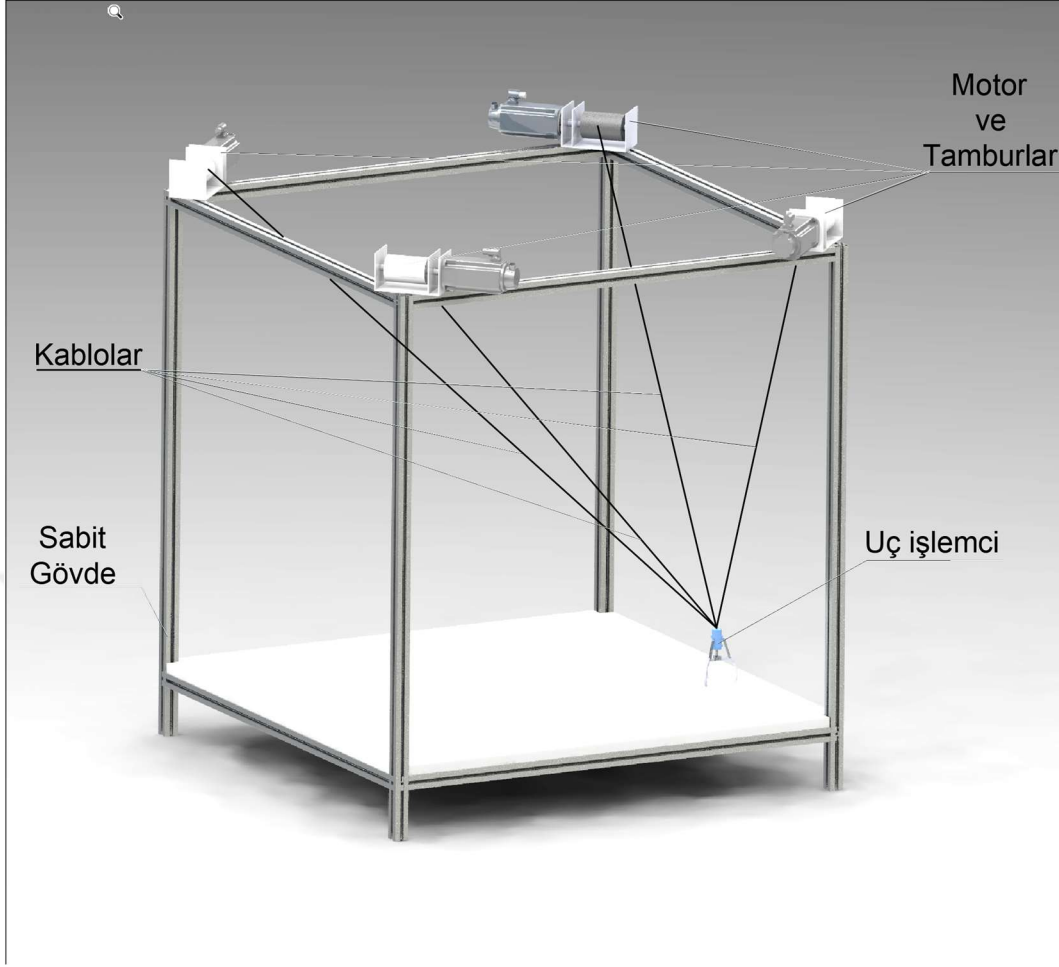


Şekil 3. Delta robot (URL-5, 2025)



Şekil 4. Delta robot çalışma alanı (URL-6, 2025)

Kabloyla sürülen paralel robotlar (KSPR), rijit uzuvlar yerine kabloların kullanıldığı paralel robotların özel bir türüdür. Uç işlemci, genellikle kabloların tamburlara sarılıp bırakılmasıyla kısaltılarak hareket ettirilir. Şekil 5'te gösterildiği gibi sabit uzuvlar yerine kabloların kullanılması, motor ve tambur gibi bileşenlerin sabit noktalara konumlandırılması KSPR'lerin geniş bir çalışma alanına sahip olmasına ve ataletsel etkilerin önemli ölçüde azalmasına, dolayısıyla yüksek hızlara ulaşılmasına imkân tanır (Qian vd., 2018).



řekil 5. Kabloyla srlen paralel robot

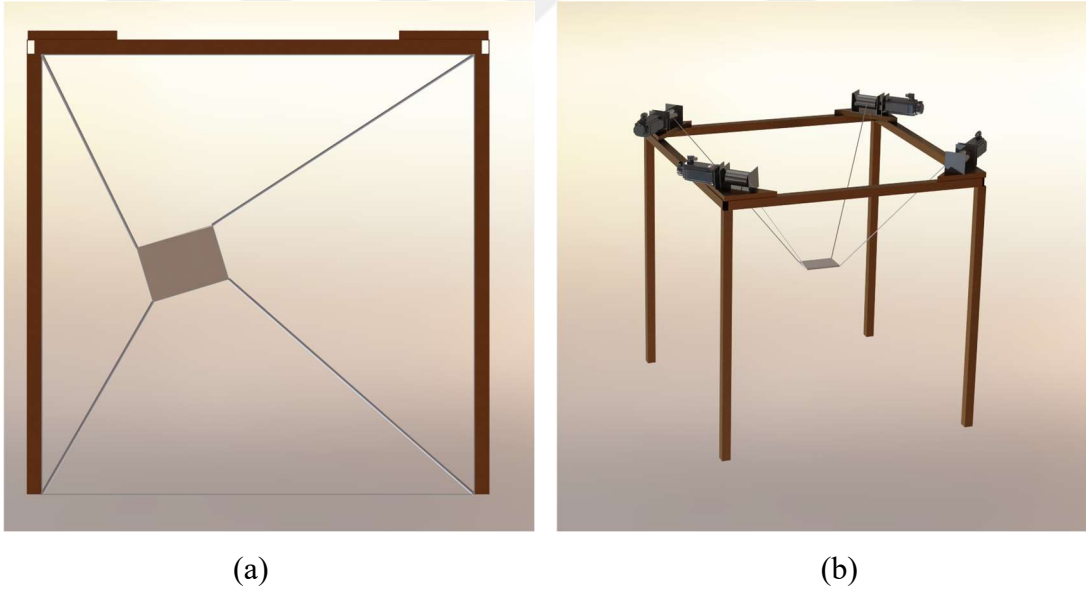
1.2. Tez İerięi

Bu tez alıřmasının ilk blmnde, KSPR'ler hakkında temel bilgiler sunulmuř, sistemin genel yapısı ve kullanım alanları aıklanmıřtır. Ayrıca, tezde kullanılan yntemlere iliřkin temel kavramlara yer verilmiř; literatr zetiyle birlikte mevcut arařtırmalardaki eksiklikler vurgulanmıř ve alıřmanın kapsamı ortaya konmuřtur. İkinici blmde ise, tez kapsamında yrtlen alıřmalar detaylı ve sistematik biimde aıklanmıřtır. alıřmada kullanılan KSPR'nin kinematik modeli oluřturulmuř, kullanılan P yntemi ve bu yntemin KSPR'ye yol planlama amacıyla uygulanıřı adım adım ele alınmıřtır. Deney dzeneęinin kurulumu kapsamında, KSPR sistemi ile RGB-D kameranın entegrasyonu gerekleřtirilmiř; alıřma ortamındaki nesnelerin konumlarının belirlenmesi iin geliřtirilen grnt iřleme algoritmalarıyla znitelik ıkarımı yapılmıřtır. Elde edilen gzlem verilerinin P ajanına aktarılması ve sistemin btnsel olarak kontrol ana hatlarıyla anlatılmıřtır. nc

bölümde, gerçekleştirilen çalışmalar sonucunda elde edilen veriler, grafikler ve tablolar eşliğinde sunulmuş; bu veriler ayrıntılı olarak analiz edilerek, geliştirilen yöntemin başarımları oranı ve etkinliği değerlendirilmiştir. Son bölümde ise, genel bir değerlendirme yapılarak çalışmanın güçlü yönleri ve geliştirilebilecek tarafları tartışılmıştır.

1.3. Kabloyla Sürülen Paralel Robotlar

Paralel robotların özel bir türü olan kabloyla sürülen paralel robotlar, rijit uzuvlar yerine kabloların kullanıldığı bir robot tipidir. Yapılarında rijit uzuvlar yerine kabloların kullanılmasından dolayı sadece çekme kuvvetini iletebilirler (Alp & Agrawall, 2002). Şekil 6'da görüldüğü gibi KSPR'ler çalışma ortamlarına göre düzlemsel ve ortamsal olarak ikiye ayrılırlar. Düzlemsel KSPR'ler üç serbestlik derecesine sahiptirler ve uç işlemcileri yalnızca düzlemsel olarak hareket eder. Üç serbestlik derecesine sahip olup kullanılan kablolar bağlantı geometrisi nedeniyle tüm serbestlik derecelerindeki hareket kablo uzunluklarıyla sınırlandırıldığı için tam sınırlandırılmış olarak adlandırılırlar. Ortamsal KSPR'ler altı serbestlik derecesine sahiptirler ve üç boyutlu çalışma alanı içerisinde hareket ederler.



Şekil 6. (a) Düzlemsel KSPR, (b) Ortamsal KSPR

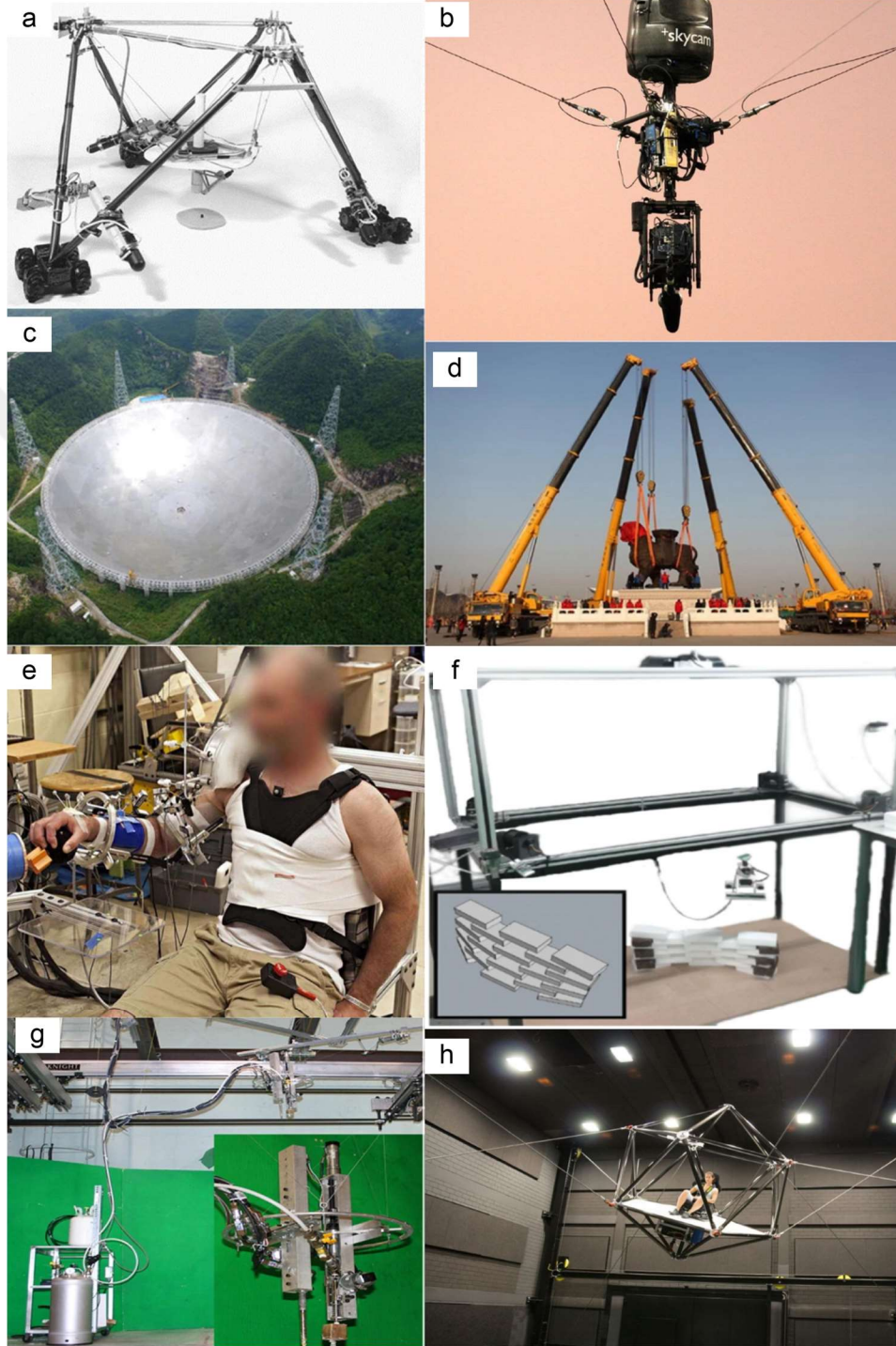
KSPR'ler, uç işlemcinin serbestlik derecelerindeki hareketlerinin fiziksel olarak sınırlandırılıp sınırlandırılmamasına göre “tam kısıtlanmış” veya “eksik kısıtlanmış” olarak sınıflandırılır. Eğer uç işlemciye bağlanan kablolar, tüm serbestlik derecelerinde hareketi

fiziksel olarak engelleyebiliyorsa sistem tam kısıtlanmış kabul edilir; aksi halde eksik kısıtlanmış olur. Bununla birlikte, sistemde kullanılan kablo veya eyleyici sayısı serbestlik derecesinden fazla olduğunda sistem “artıksıl eyletilmiş” olarak adlandırılır. Ancak kablo sayısının fazla olması, tek başına tam kısıtlanma veya tüm serbestlik derecelerinin bağımsız olarak kontrol edilebileceği anlamına gelmez. Özellikle kabloların tümü belirli bir yönden (örneğin sadece üstten) bağlandığında, teorik olarak artıksıl eyleyici bulunsu dahi uç işlemcinin tüm serbestlik dereceleri üzerinde tam kontrol sağlanamayabilir; bu durumda sistem eksik kısıtlanmış ve artıksıl eyletilmiş olur, fakat pratikte tüm hareket eksenleri bağımsız olarak kontrol edilemez. Dolayısıyla, kablo sayısı ile sistemin kısıtlanma ve eyletme durumu arasında doğrudan bir ilişki olmayıp, kablo bağlantılarının geometrisi ve yönelimi de dikkate alınmalıdır

1.3.1. KSPR Uygulama Alanları

KSPR’ler hem endüstriyel hem de akademik anlamda birçok uygulama alanına sahiptirler. KSPR’ler üzerine yapılan ilk araştırmalar, 1984 yılında Amerika’da sualtı operasyonlarına yönelik bir paralel manipülatörün geliştirilmesiyle başlamıştır. Ardından, Şekil 7a’da görülen 1989 yılında ABD Ulusal Standartlar ve Teknoloji Enstitüsü (NIST) tarafından geliştirilen RoboCrane projesi (James vd., 1993), yalın tasarımı ve çok yönlü kullanımı ile özellikle işleme makineleri, liman yük taşımacılığı, köprü inşaatı ve kaynak işlemleri gibi çeşitli alanlarda yaygın biçimde kullanılmıştır. 1980’lerin sonunda ise August Design Company tarafından geliştirilen, dört kabloyla sürülen ve 44.8 km/saat hıza ulaşabilen SkyCam sistemi (Tanaka vd., 1988), büyük ölçekli canlı yayınlarda ve özellikle yüksek hızlı takip gerektiren spor karşılaşmalarında yaygın olarak kullanılmaya başlanmıştır (Şekil 7b). Şekil 7c’de Çin’de 500 metre çapındaki FAST (Duan, 1999) radyo teleskobunun hassas konumlandırılması için kablo tahrikli taşıma sistemi geliştirilmiştir. Bu sistem KSPR teknolojisinin en başarılı uygulamalarından biri olmuştur. Şekil 7d’de çoklu mobil vinçlerden oluşan bir KSPR geliştirilmiş ve inşaatlarda yüklerin dengeli taşınabilmesine olanak sunmuştur (Zi vd., 2015). Şekil 7e’de görüldüğü gibi üst kol sinir rehabilitasyonu için geliştirilen kablo tahrikli bir dış iskelet olarak kullanılmıştır (Mao vd., 2014). Şekil 7f’de görülen SPIDERobot isimli robot, mimari çalışmalarda otonom tut-yerleştir uygulamaları için geliştirilmiştir (Pinto vd., 2017). Şekil 7g’de ise geniş çalışma alanına sahip 6 serbestlik dereceli bir KPSR ile 3 boyutlu bir yazıcı tasarlanmış ve kullanılmıştır (Barnett & Gosselin,

2015). Şekil 7h’de ise Max Plank Enstitüsü tarafından geliştirilen bir hareket simülatörü görülmektedir (Miermeister vd., 2016).

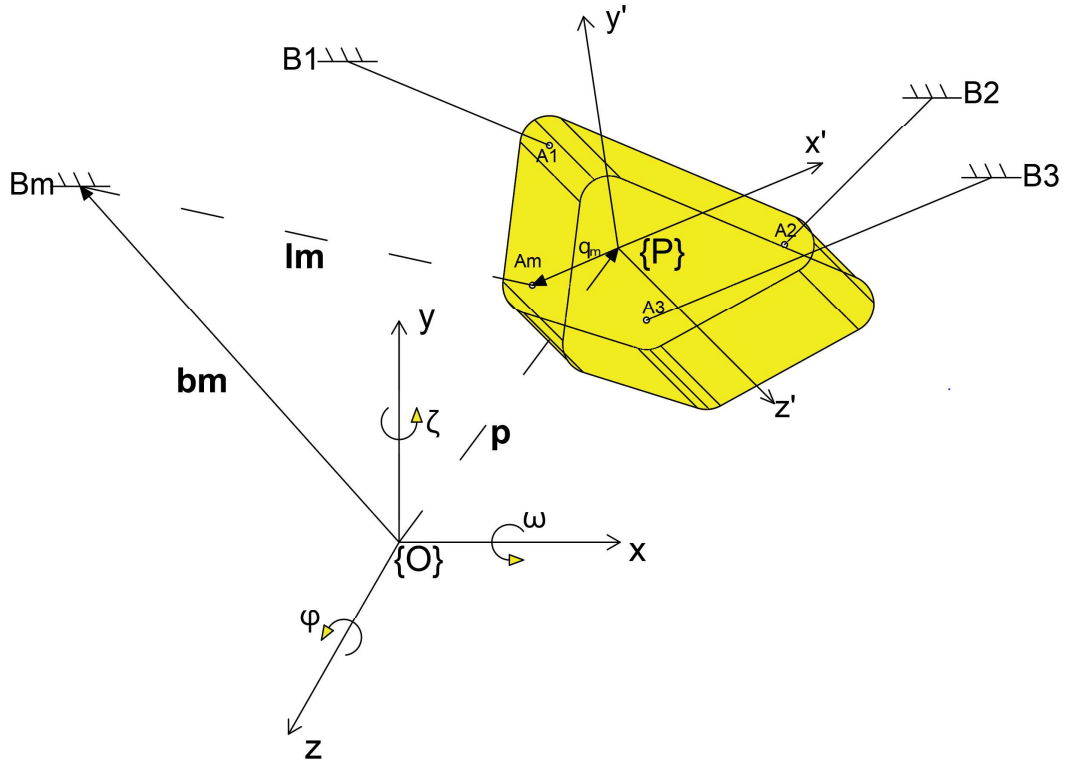


Şekil 7. (a) RoboCrane, (b) SkyCam, (c) FAST radyo teleskopu h, (d) Çoklu vinç sistemi, (e)Kabloyla sürülen kol rehabilitasyon robotu, (f) SPIDERobot, (g) Kabloyla sürülen 3B yazıcı, (h) Kabloyla sürülen hareket simülatörü

1.3.2. Kinematik Model

KSPR'ler bir paralel robot türü olduklarından bu robotlar için ileri kinematik işlem, kablo uzunlukları bilinirken uç işlemci konumunun hesaplanmasıdır. Ters kinematik işlem ise uç işlemcinin konumunun bulunduğu durumda kablo uzunluklarının hesaplanması işlemidir. Fakat paralel robotlarda ileri kinematik denklemler birden fazla sayıda çözüm içerdiğinden KSPR'lerin de ileri kinematiklerini hesaplamak oldukça zordur ve farklı sayısal yöntemler kullanmayı gerektirir. Ancak ters kinematiklerini hesaplamak çok daha kolaydır. Kablo çıkış noktalarından uç işlemciye çizilen vektörlerle kablo uzunlukları ve yönelimleri kolaylıkla hesaplanabilir. Bu çalışmada ise robotun konum kontrolü için ters kinematik çözüm ele alınmıştır.

KSPR sabit çerçeve, kablolar ve uç işlemci ya da platform olmak üzere üç ana bileşenden oluşur. Şekil 8'de tabandaki referans koordinat sistemi $\{O\}$, uç işlemciye bağlı yerel koordinat sistemi $\{P\}$, kabloların sabit gövdeye bağlantı noktaları B_i , kabloların hareketli platforma bağlantı noktaları A_i , uç işlemci merkez konum vektörü $\mathbf{p}$, uç işlemci merkezinden kablo bağlantılarına vektörler $\mathbf{q}_i$ ile gösterilmiştir.



Şekil 8. KSPR genel kinematik yapı

{P}'den {O}'ya dönüşüm matrisi ile şu şekilde ifade edilir:

$$D_P^O = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \quad (1)$$

Burada dönme matrisi R şöyle gösterilir:

$$R = \begin{bmatrix} \cos \omega \cos \zeta & -\sin \omega \cos \varphi + \cos \omega \sin \zeta \sin \varphi & \sin \omega \sin \varphi + \cos \omega \sin \zeta \cos \varphi \\ \sin \omega \cos \zeta & \cos \omega \cos \varphi + \sin \omega \sin \zeta \sin \varphi & -\cos \omega \sin \varphi + \sin \omega \sin \zeta \cos \varphi \\ -\sin \zeta & \cos \zeta \sin \varphi & \cos \zeta \cos \varphi \end{bmatrix} \quad (2)$$

A noktalarının {P}'ye göre konumları ise şu şekilde ifade edilir:

$$q_i^P = \begin{bmatrix} q_{i,x}^P \\ q_{i,y}^P \\ q_{i,z}^P \end{bmatrix} \quad i = 1, 2, \dots, m \quad (3)$$

Kabloların sabit gövdeye bağlantı konumları ise şu şekilde gösterilir:

$$b_i = \begin{bmatrix} b_{i,x}^P \\ b_{i,y}^P \\ b_{i,z}^P \end{bmatrix} \quad i = 1, 2, \dots, m \quad (4)$$

Bu durumda kablolar için vektör kapalılık eşitliği şu şekilde yazılabilir:

$$\overrightarrow{OB_i} + \overrightarrow{B_iA_i} = \overrightarrow{OP} + \overrightarrow{PA_i} \quad i = 1, 2, \dots, m \quad (5)$$

Böylece kablolar vektörel olarak Denklem 6'da kablo uzunlukları skaler olarak Denklem 7'deki gibi ifade edilir:

$$l_i = p + R q_i^P - b_i, \quad i = 1, 2, \dots, m \quad (6)$$

$$\|l_i\| = \|\overline{B_i A_i}\| = \|\overline{OP} + \overline{PA_i} - \overline{OB_i}\| = \|\mathbf{p} + \mathbf{R} \mathbf{q}_i^P - \mathbf{b}_i\| \quad i = 1, 2, \dots, m \quad (7)$$

1.4. Literatür Özeti

Robotikte yol planlama; bir robotun çalışma ortamı içerisinde, fiziksel ve geometrik sınırlar dahilinde, başlangıç konumundan hedef konuma engellere çarpmadan optimum şekilde yol oluşturmayı amaçlar (Yang vd., 2016). Fakat amaç her zaman iki konum arasında yalnızca yol belirlemek olmayabilir. Yol planlama işlemi, mevcut sistemin kısıtlamalarına bağlı olarak farklı optimizasyon amaçları içerebilir. Bu amaçlar arasında en kısa mesafeyi bulmak, hedefe en kısa sürede varmak, sistemin dinamik özelliklerine uygun bir şekilde yumuşak ve kararlı rotalar üretmek veya enerji sarfiyatını en aza indirmek gibi farklı amaçlar yer alabilir. Bu nedenlerle yol planlama; yalnızca konumlar arasında bir yol belirlemek değil, aynı zamanda bu yolun stratejik anlamda istenen kriterlere uygun olarak belirlenmesini de kapsar. Bu kapsamda noktalar arası hareket eden her türlü robot için yol planlama yöntemleri kullanılabilir ve uygulanabilir. Ayrıca bir robotun insan müdahalesi olmadan hareket edebilmesi ve verilen görevleri sorunsuz bir şekilde yerine getirebilmesi için yol planlama yapabilmesi bir zorunluluktur. Yol planlama yöntemlerine bakıldığında geleneksel yol planlama yöntemleri, robotların farklı kullanım alanlarında geniş bir yelpazede uygulanmaktadır. Bunlar arasında mobil robotlar, otonom kara ve hava araçları, insansız hava araçları, endüstriyel robot kolları ve servis robotları gibi farklı uygulama alanları bulunmaktadır (Karur vd., 2021).

Geleneksel yol planlama yöntemleri, ortamın haritalanmış olduğu varsayımıyla çalışır ve robotun hareket edeceği ortamı genellikle bir grafik ya da bir hücre dizisi olarak modeller. Bu yöntemlerin başında A* algoritması (Hart vd., 1968), Dijkstra algoritması (Dijkstra, 1959), RRT algoritması (LaValle & Kuffner Jr, 2001) ve D* algoritması (Stentz, 1994) gelir. A* algoritması, sezgisel bir fonksiyon yardımıyla hedefe ulaşmak için en kısa yolu arayarak hızlı ve optimal sonuçlar üretir. Ancak büyük ölçekli ve karmaşık ortamlarda hesaplama maliyeti yüksek olabilir. Dijkstra algoritması, sezgisel bilgi kullanmadan kesin olarak en kısa yolu bulur ancak çalışma uzayındaki tüm noktaları değerlendirdiğinden hesaplama maliyeti çok yüksektir. RRT algoritması, rastgele örneklemelerle ortamı keşfettiği için yüksek boyutlu ve karmaşık ortamlarda hızlı ve pratik çözümler sağlar fakat optimal çözüme

ulařma garantisi yoktur. D* algoritması ise ortamda dinamik deęiřimler olduęunda mevcut yolu hızlı bir řekilde gncelleyebilir ancak yoğun dinamik deęiřimler ve karmařık yapılar da hesaplama yk artabilir. zellikle fabrika ii lojistik sistemlerde, depo ii otomatik tařıma aralarında ve iki boyutlu zemin hareketlerinde A* veya Dijkstra gibi algoritmalar sıklıkla tercih edilir. Dinamik engellerin bulunduęu karmařık ortamlarda ise genellikle D* algoritması kullanılırken, keřif ve navigasyon amalı insansız hava araları gibi  boyutlu ortamlarda hızlı zm saęladıęı iin RRT algoritması tercih edilmektedir. Ancak geleneksel yntemlerin  boyutlu ortamlardaki uygulamalarında nemli dezavantajları mevcuttur.  boyutlu ortamların karmařık yapısı nedeniyle ortamın grafik tabanlı gsteriminin boyutu nemli lde artmakta, bu durum hesaplama maliyetini ciddi oranda ykseltmektedir. Ayrıca yksek boyutlu arama alanlarında geleneksel yntemlerin performansı dřerek gerek zamanlı uygulamalarda gerekli olan hız ve tepki srelerini karřılaması gleřmektedir. Buna ek olarak ortamın dinamik yapıda olması veya srekli gncellenmesi gereken karmařık engeller iermesi durumunda bu yntemlerin sık sık yeniden hesaplama yapması gerekebilir. Bu durum, zellikle yksek boyutlu ve dinamik yapıdaki grevlerde klasik yntemlerin etkinlięini nemli lde azaltmaktadır (Choset vd., 2005).

KSPR'lerde bir noktadan dięerine hareket ederken hem u iřlemci hem de kablolar hareket ettięinden ortamsal bir KSPR'de yol planlama, dięer robot trlerinden ok daha farklı gereksinimlere sahiptir.  boyutlu alıřma ortamları iin kullanılan klasik yol planlama yntemlerinde hareket eden robot bir nokta, daire veya kre gibi bir btn olarak varsayılarak engellere arpmadan btncl bir biimde yol planı oluřturulmaktadır. Fakat ortamsal KSPR'lerde hem u iřlemcinin hem de kabloların alıřma ortamı ierisindeki engellere temas etmeden hedef konuma ulařması gerektięinden geleneksel yol planlama yntemleri bu noktada yetersiz kalmaktadır. Literatr zetinde verildięi zere yapılan alıřmalara bakıldıęında klasik yntemlerin KSPR'ler iin olduka verimsiz ve yetersiz olduęu, gerek zamanlı uygulamalar iin uygun olmadıęı aıka anlařılmaktadır. Kablo gerilimlerinin u iřlemci konumuna gre farklılık gstermesi sebebiyle KSPR'ler iin sadece kablo ve u iřlemcinin engellere teması deęil, takip edilen yol da olduka nem arz etmektedir. Bu sebeplerle klasik yol planlama algoritmalarının yerine bir makine ęrenmesi yntemi olan derin P ynteminin kullanılması KSPR'lerin yol planlamasında avantaj saęlayacaktır.

KSPR'lerde PÖ yöntemi KSPR dinamiğinin kontrolü amacıyla da kullanılmaktadır. Xiong vd. (2020) ters kinematik çözümü ve DDPG algoritmasını hibrit olarak kullanmış ve klasik kontrolden daha iyi sonuçlar elde etmiştir. Grimshaw & Oyekan (2021) ortamsal KSPR'nin uç işlemcisine bağlı düz bir plaka üzerine yerleştirilen topun plaka üzerindeki konumunun görsel sensörlerle belirleyerek eş zamanlı olarak topun plaka üzerindeki konum kontrolünü PÖ ile gerçekleştirmişlerdir. Sancak vd. (2022) düzlemsel KSPR'nin noktadan noktaya ve dinamik referans konumu takibinde bir PÖ algoritması olan DDPG algoritmasını kullanmışlardır. Yapılan dinamik kontrol çalışmalarında kullanılan PÖ yöntemlerinin göstermiş olduğu başarımlar KSPR'lerin yörünge planlama işlemlerinde kullanılabilirliği açısından bir araştırma alanı doğurmaktadır.

KSPR'lerin uç işlemci ve kablolarının çalışma alanı içerisindeki objelere çarpmadan yörünge planlaması ve takibindeki çalışmalara bakıldığında Lahouar vd. (2009) tarafından gerçekleştirilen çalışmada, ortamsal KSPR'nin yol planlaması için iki modlu bir kontrol algoritması geliştirilmiştir. Bu algoritmada, uç işlemci ve kablolar engelden uzaktayken hedefe doğru herhangi bir çarpışma sorgulaması yapılmadan doğrudan bir yol izlenirken, engele yaklaşıldığında ikinci mod devreye girerek engellere çarpmadan alternatif güzergâhlar aranmaktadır. Yapılan çalışmada, çalışma ortamındaki engeller çok yüzlü olarak modellenmiş, kablo ve uç işlemci ile engeller arasındaki mesafe hesaplamaları bu geometrik tanıma göre gerçekleştirilmiştir. Hem uç işlemcinin hem de kabloların engellerle temasını tespit eden özel bir algoritma geliştirilmiş, böylece çarpışma riski her iki unsur için de değerlendirilmiştir. Sunulan yaklaşımda yol planlama işlemi önceden tanımlanmış, benzetim ortamına manuel olarak yerleştirilen engellerle sınırlı kalmakta ve yalnızca düzlemsel hareket dikkate alınmaktadır. Dolayısıyla ortam üç boyutlu olsa da önerilen yol planlama ve çarpışma engelleme süreci yalnızca düzlemsel olarak gerçekleştirilmiştir. Deney düzeneğine bulunan engeller, benzetim ortamındaki aynısı olacak şekilde yerleştirilmiş ve bu sebeple ortamı algılayacak herhangi bir sensör kullanılmamıştır. Korayem vd. (2014) üçgen tabanlı bir uç işlemcinin altı kablo ile desteklendiği ortamsal bir kabloyla asılı paralel robot sisteminin kontrolü için State Dependent Riccati Equation (SDRE) yöntemini kullanmışlardır. Simülasyon ortamında engel kaçınma, Yapay Potansiyel Alan (Warren, 1989) yöntemiyle gerçekleştirilmiş ve uç işlemcinin çarpışmasız biçimde hedefe ulaşması sağlanmıştır. Ayrıca hareketli sınırlar, varyasyonlar hesabı ile modellenerek başlangıç ve hedef konfigürasyonları bu sınırlar üzerinden optimal şekilde seçilmiştir. Ancak deneysel uygulamalarda ortam algısı sınırlı tutulmuştur. Gerçek deney düzeneğinde

kullanılan kamera ve lazer sistemi yalnızca uç işlemcinin konum ve oryantasyon bilgisini doğrulamak amacıyla kullanılmış; engel algılama ya da haritalama görevinde kullanılmamıştır. Fiziksel deneylerde kullanılan engel, sabit bir konuma yerleştirilmiş ve kontrol bu konum önceden bilinerek yapılandırılmıştır. Zi vd. (2015) çoklu mobil vinçlerden oluşan kabloyla asılı paralel robot sistemi için engellere çarpmadan yol planlama ve otomatik platform seviyeleme yaklaşımı geliştirmişlerdir. Çalışmada ortam, çalışma alanını düzenli aralıklı hücrelere bölen ve her hücrenin konumunu ve durumunu ayrı ayrı analiz etmeye imkân tanıyan bir ızgara tabanlı ortam modeli ile modellenmiştir. Yol planlama çalışması, Yapay Potansiyel Alan yöntemi ve ızgara tabanlı yöntemlerin (Krishnaswamy & Stentz, 1995) birleşimiyle gerçekleştirilmiştir. Dört noktadan destekli otomatik platform seviyeleme algoritması ile yükün taşınması sırasında platformun yatayda dengede kalması sağlanmıştır. Tüm yöntemler Matlab ve LabVIEW tabanlı bir ortak benzetim ortamında test edilerek çoklu vinçten oluşan kabloyla sürülen paralel robotun karmaşık ortamlarda güvenli ve hassas şekilde hareket edebildiği gösterilmiştir. Ancak yol planlama ve engel kaçınma algoritmasının uygulaması esas olarak iki boyutlu hareketle sınırlı kalmış, sistem çoğunlukla X ve Y eksenlerinde hareket etmiş, Z ekseninde ise yükseklik genellikle sabit tutulmuştur. Pinto vd. (2017) tarafından yapılan çalışmada mimari projelerin otonom olarak inşası için tasarlanan dört kabloyla sürülen SPIDERobot adlı kabloyla sürülen paralel robotun, görsel geri beslemeyle yol planlaması ve hareket kontrolü çalışılmıştır. Geliştirilen görsel tabanlı yol planlama algoritması ile ortamda bulunan engeller ve objeler, RGB-D kamera ile algılanmıştır. Algılanan engeller, üç boyutlu ızgara tabanlı bir harita üzerinde simülasyon ortamında yeniden modellenmiştir. Model üzerinde A* algoritmasına dayalı yol planlama yöntemiyle uç işlemcinin ve kabloların engellere çarpmadan izleyeceği güzergâh belirlenmiştir. Uç işlemci ve parçaların konumu, görsel segmentasyon ve derinlik verileriyle tespit edilmiştir. Çalışmada yol planlama algoritması ortalama 2 ms gibi oldukça kısa bir sürede tamamlanmış ve planlanan yol, oluşturulan sanal modele yaklaşık 755 ms'de yüklenmiştir. Ancak A* algoritmasının çalışabilmesi için ortamın ve engellerin modelinin önceden oluşturulmuş olması gerekmektedir. Bu nedenle bu çalışmada çalışma ortamı ve engeller, bilgisayar ortamında önceden modellenmiştir. Model oluşturma süreci; kamera verilerinden elde edilen nokta bulutu veya derinlik verilerinin işlenmesini, 3B ızgara tabanlı harita oluşturulmasını ve objelerin segmentasyonunu gerektirir. Bu işlemler yüksek miktarda veri ve karmaşık hesaplama gerektirdiği için ciddi bir işlem gücüne ve zamana ihtiyaç duyar. Sonuç olarak ortamın modellenmesi ve yol planlamasının her adımda yeniden yapılması,

gerçek zamanlı uygulamalar için uygun değildir. Ortamda meydana gelebilecek ani değişiklikler veya yeni engeller, anında sisteme aktarılamaz ve sistemin çevresel değişikliklere hızlı tepki vermesi mümkün olmaz. Bak vd. (2019) 8 kabloyla sürülen tam sınırlandırılmış KSPR için geleneksel RRT* algoritmasına karşılık değiştirilmiş bir RRT algoritması önermişlerdir. Benzetim ortamında yapılan çalışmada değiştirilmiş RRT algoritması, RRT*'a göre daha kısa hesaplama süresi sunmuştur. Fakat yol planlama, iki nokta arasında herhangi bir çarpışma denetimi olmadan rastgele yapıldığından planlanan her bir yol için engellerle temas olup olmadığının tek tek tespit edilmesi gerekmektedir. Algoritma tarafından üretilen yolların sadece %38'inde çarpışma olmadan hedefe ulaşabilmiş ve bu anlamda hesaplama süreleri her bir yol için 0.05 saniye ile 7 saniye arasında değişkenlik göstermiştir. Mishra vd. (2021) 8 kablo ile sürülen ve 6 serbestlik derecesine sahip KSPR için AFG-RRT* algoritmasını kullanmış ve benzetim ortamında yapılan çalışmada engel ve başlangıç konumları değiştirilmeden tek bir çalışma ortamı için çarpışma olmayan yola yaklaşık 13 dakikada ulaşılmıştır. Daha sonra benzetim ortamındaki engelleri gerçek ortama el ile yerleştirerek belirlenen yörüngeyi test etmişler ve benzer sonuçlar elde etmişlerdir. Çalışma ortamının algılanması ve engellerin tespiti söz konusu değildir. Xu & Park (2021) 8 kabloyla sürülen tam sınırlandırılmış ortamsal bir KSPR'de hareketli engellerden kaçınma problemini çözmek amacıyla, klasik RRT algoritmasını hedefe yönelimli örnekleme, dinamik adım büyüklüğü ve kabloya özgü kısıtlar ile geliştirerek dinamik ortamlara uyarlanabilir bir yol planlama yöntemi sunmuştur. Yöntemde, uç işlemci ve kablolar için çarpışma tespiti süpürülen hacim yaklaşımı ve Gilbert–Johnson–Keerthi algoritmasıyla gerçekleştirilmiş, ayrıca hem uç işlemcinin hem de engellerin hız bilgileri kullanılarak en yakın yaklaşma zamanı ve mesafesi öngörülmüştür. Simülasyon ve toplu test sonuçları, önerilen algoritmanın önceki APG-RRT yöntemine göre uç işlemcinin salınım süresini %36, yol maliyetini ise %32 oranında azalttığını göstermiştir. Yöntemin gerçek ortamda uygulanabilirliği, hareketli engel olarak kullanılan bir drone ile yürütülen ve iki adet web kamera ile üç boyutlu konumu algılanan deneysel çalışmalarla doğrulanmıştır. Çalışma, KSPR'ler için hem uç efektörün hem de kabloların gerçek zamanlı ve dinamik engellerle çarpışmasız şekilde hareketini sağlayan özgün bir yol planlama çözümü ortaya koymuştur. Ancak çevredeki ani değişikliklerde algoritmanın yeniden yol planlaması gerekmekte ve bu da zaman kaybına yol açmaktadır. Ayrıca APG-RRT'nin teorik olarak optimal çözüme ulaşma garantisi yoktur ve uygulama aşamasında kodlama ile entegrasyon işlemleri, klasik RRT'ye kıyasla daha zordur. Bu nedenlerle, APG-RRT'nin seçimi

yapılırken uygulama gereksinimleri ve kısıtları dikkatlice değerlendirilmelidir. Liu vd. (2023) tarafından gerçekleştirilen çalışmada dört mobil robotun hareketiyle konumlandırılan ve sabit uzunluktaki kablolar kullanılan bir KSPR sisteminde, dinamik engellerden kaçınmak amacıyla PÖ algoritması kullanılmıştır. Ancak uç işlemcinin, mobil robotların ve engellerin pozisyon bilgileri gerçek zamanlı olarak laboratuvara kurulu hareket yakalama sistemiyle elde edilmiştir. Bu da sistemin gerçek saha uygulamalarında kullanılabilirliğini ciddi biçimde kısıtlamaktadır. Hareket yakalama sistemi sayesinde yüksek doğruluk elde edilse de pratik uygulamalarda yaygın sensörler (kamera, lidar, ultrasonik vb.) yerine bu yöntemin tercih edilmesi, pozisyon ölçümündeki hata ve gecikmelere karşı sistemin dayanıklılığının test edilmemesi anlamına gelmektedir. Ayrıca kablo uzunluklarının sabit tutulması, uç işlemci hareketinin yalnızca mobil robotların hareketiyle sağlanmasına neden olmakta ve robotun hareket uzayını daraltmaktadır. Sonuç olarak bu çalışma, PÖ tabanlı yol planlama açısından oldukça yenilikçi bir çalışma olsa da ortam algılama ve pratik uygulama açısından önemli sınırlamalar taşımaktadır.

Literatürde düzlemsel ve ortamsal KSPR'lerin PÖ yöntemiyle kontrolü mevcuttur. Fakat, uç işlemci ve kabloların geleneksel yöntemler kullanılarak engellerden kaçınması gibi çalışmalar mevcut olmasına rağmen, aktif kablo uzunluklarıyla kontrol edilen KSPR'ler için PÖ ile yol planlama çalışması bulunmamaktadır. Bu tez çalışmasında kablo uzunlukları aktif olarak kontrol edilen 4 kablo ile sürülen eksik sınırlandırılmış ortamsal bir paralel robotun uç işlemci ve kablolarının kamera verilerini kullanarak gerçek zamanlı obje tespiti ve PÖ yardımıyla engelden kaçınması amaçlanmıştır. Tez kapsamında robotun çalışma ortamı içerisinde bulunan obje ve engeli sensörlerle tespit ederek üç boyutlu uzayda konumlarının belirlenmesi, kabloların ve uç işlemcinin konumu belirlenen engele çarpmadan taşınacak objeyi tutarak bir noktadan diğer bir noktaya taşınması ve yerleştirilmesi çalışmaları yapılmıştır. Bu görevde robotun yörünge planlamasında geleneksel yöntemler kullanılmadan sadece PÖ yöntemi kullanılarak kurulan sistem gerçek zamanlı olarak çalışabilen otonom bir tut yerleştir robotu olarak görev yapabilmektedir.

1.5. Makine Öğrenmesi

Makine öğrenmesi; bilgisayar sistemlerinin verilerden deneyim kazanarak performansını geliştirmesini sağlayan bir öğrenme yöntemi olup günümüzde sağlık, finans, pazarlama, üretim, robotik gibi pek çok farklı alanda aktif olarak kullanılmaktadır. Makine

öğrenmesinde temel amaç, mevcut verilerden yola çıkarak bir model oluşturmak ve bu modelin daha önce görmediği veriler üzerinde yüksek doğrulukla tahminler yapabilmesini sağlamaktır. Makine öğrenmesi genel olarak üç ana türe ayrılır: denetimli öğrenme, denetimsiz öğrenme ve PÖ. Denetimli öğrenmede sistem, etiketlenmiş yani girişlere karşılık doğru çıkış bilgileri verilmiş veriler üzerinden bir model üretir. Bu model, eğitim verilerinin dışında yeni verilere karşılık gelen çıktıları tahmin etmekte kullanılır. Robotik sistemlerde obje tanıma, hareket sınıflandırma, hata tespiti gibi amaçlarla kullanılır. Denetimsiz öğrenmede ise veriler etiketlenmemiştir ve amaç, veri içindeki yapıları veya örüntüleri keşfetmek ve bu yapıları kümelemektir. Denetimsiz öğrenme; robotikte etiketlenmemiş verilerle ortam keşfi, obje segmentasyonu, anomali tespiti, hareket örüntülerinin analizi ve özellik çıkarımı gibi görevlerde kullanılır. Pekiştirmeli öğrenmede denetimli ve denetimsiz öğrenmeden farklı olarak herhangi bir veri seti bulunmaz. Ajanın çevresiyle etkileşimde bulunarak nasıl davranması gerektiğini kendi kendine öğrendiği bir öğrenme yöntemidir. Bu süreçte ajan; çevrenin o anki durumunu gözlemler, bir aksiyon seçer ve bu eyleme karşılık çevreden bir ödül alır. Yaptığı aksiyonlar ve aldığı ödüller sonucu hangi durumda hangi aksiyonu yapması gerektiğini öğrenir. Asıl amaç, uzun vadede alınan toplam ödülün zaman içinde en yüksek seviyeye ulaşmasını sağlayacak bir strateji (politika) geliştirmektir (François-Lavet vd., 2018).

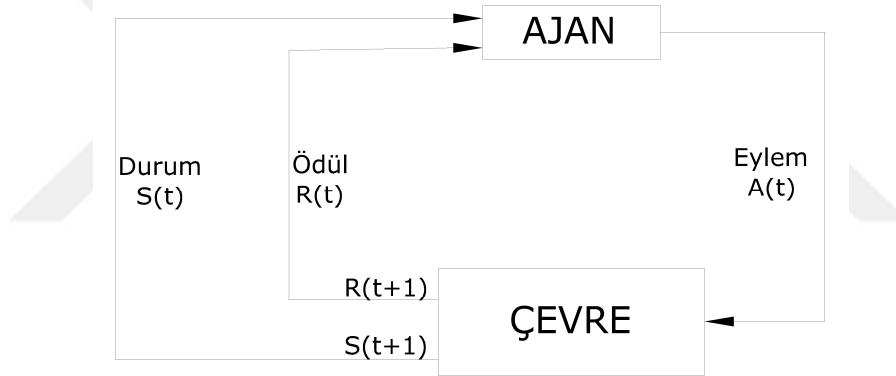
1.5.1. Derin Öğrenme

Derin öğrenme, makine öğrenmesinin bir alt dalı olup çok katmanlı yapay sinir ağları kullanarak yüksek boyutlu ve karmaşık verilerden anlamlı örüntüler çıkarmayı amaçlayan yöntemleri içerir (Goodfellow vd., 2016). Temel olarak geleneksel makine öğrenmesi yöntemlerinin sınırlı kaldığı karmaşık durumlar için geliştirilmiştir. Bu yöntemler, verileri birçok ardışık katmanda adım adım işleyerek basit ham bilgilerden daha üst düzeyde anlamlı soyutlamalar üretir (Lecun vd., 2016). Derin öğrenme algoritmaları, büyük veri kümelerinde üst düzey karmaşık ilişkileri öğrenebilmek için doğrusal olmayan aktivasyon fonksiyonları ve katmanlar arasında ağırlıkların ayarlandığı geri yayılım (backpropagation) algoritması kullanır (Rumelhart vd., 1986). Derin öğrenme yöntemleri arasında derin sinir ağları (Deep Neural Networks, DNN), evrişimli sinir ağları (Convolutional Neural Networks, CNN) ve tekrarlayan sinir ağları (Recurrent Neural Networks, RNN) gibi farklı ağ mimarileri bulunur (Goodfellow vd., 2016). Bu yöntemler; özellikle görüntü işleme, doğal dil işleme, ses

tanıma, robotik kontrol ve karmaşık zaman serisi tahminleri gibi yüksek boyutlu veri işleyen birçok uygulama alanında etkili sonuçlar üretmektedir (Lecun vd., 2016).

1.5.2. Pekiştirmeli Öğrenme

Pekiştirmeli öğrenme, bir ajanın çevresiyle etkileşime girerek uzun vadede en yüksek toplam ödülü elde etmesini sağlayacak eylemleri öğrenmesini amaçlayan bir makine öğrenmesi türüdür. Bu yöntemde öğrenme, davranışsal psikolojiden esinlenen bir şekilde çevreden alınan sayısal ödülün en üst düzeye çıkarılması üzerine kuruludur. Ajan, çevre ile sürekli etkileşim kurar ve yaptığı her eylemin sonucunda aldığı ödül ve gözlem sinyallerine dayanarak hangi davranışların avantajlı olduğunu zamanla keşfeder (Plaat, 2022). Bu yapı Şekil 9’da gösterilmiştir.



Şekil 9. PÖ yapısı ve ajanın çevreyle etkileşimi

PÖ yapısı, danışmanlı ve danışmansız öğrenme yöntemlerinden farklıdır. PÖ’de etiketli veri seti bulunmaz, sistem doğru yanıtları önceden bilmez. Bunun yerine ajan, kendi deneyimlerini oluşturarak çevreyi anlamaya çalışır ve bu deneyimlerden gelecekteki eylemlerine yön verir. Bu süreçteki temel bileşenler; ajan, durum (s), aksiyon (a), ödül (r), politika (π), değer fonksiyonu (v) ve çevre modelinden (e) oluşur. Ajan, içinde bulunduğu duruma karşılık bir aksiyon seçer; bu seçim ajan politikasına göre yapılır. Seçilen aksiyon sonucunda çevre bir ödül üretir ve ajanı yeni bir duruma geçirir. Ödül sinyali, seçilen eylemin ne kadar başarılı olduğunu sayısal olarak yansıtırken değer fonksiyonları, ajanın gelecekteki ödül beklentisini tahmin etmesine olanak tanır (Plaat, 2022).

Bu öğrenme süreci çoğunlukla Markov Karar Süreci (MKS) olarak adlandırılan bir yapı ile modellenir. MKS; ajanın karar verdiği her adımda içinde bulunduğu durumu, uyguladığı eylemi ve bunun sonucunda ulaştığı yeni durumu ve aldığı ödülü tanımlar. Markov özelliğine göre sistemin bir sonraki durumu, yalnızca mevcut durum ve seçilen eyleme bağlıdır; geçmişteki durumlar doğrudan etkili değildir. Bu özellik, öğrenme algoritmalarının bellek gereksinimini azaltır, sistemin karmaşıklığı artsa bile ajanın her adımda yalnızca güncel durumunu dikkate alarak karar vermesine imkân tanır (Plaat, 2022).

1.5.3. Markov Karar Süreci

Pekiştirmeli öğrenmede eylemleri gerçekleştiren ve öğrenen yapı ajan, ajanın etkileşimde bulunduğu ortam ya da sistem çevre, ajanın çevrede içinde bulunduğu anlık durum s , ajanın anlık mevcut durumunda seçebileceği komutlar aksiyon(a), bir aksiyon gerçekleştirildikten sonra çevreden alınan geri bildirim ödül(r), ajanın hangi durumda hangi eylemi seçeceğine karar veren strateji politika(π) olarak tanımlanmıştır. Bu bileşenler Markov Karar Süreci (MKS)'nin çerçevesini oluşturmaktadır. Markov özelliği gereği sistemin gelecekteki durumu yalnızca mevcut duruma ve seçilen eyleme bağlı olduğunu, geçmişe ait daha önceki durumlara doğrudan hedef olmadığını ifade eder. Bu sayede sistem zamanla karmaşıklaşırsa bile karar mekanizması, her adımda yalnızca o anki durumu esas alarak öğrenme gerçekleştirir (Plaat, 2022). MKS'ye göre PÖ problemleri aşağıdaki şekilde Denklem (8) ile tanımlanır:

$$M=(S,A,P,R,\gamma) \quad (8)$$

Burada S sonlu ya da sürekli durumlar kümesini, A aksiyonlar kümesini ifade eder. $P(s'/s,a)$ geçiş olasılık fonksiyonudur ve ajan s durumunda a eylemini uyguladığında bir sonraki durumun s' olma olasılığını ifade eder. $R(s,a,s')$ beklenen ödül fonksiyonudur ve ajan s durumunda a eylemini yaptıktan sonra s' durumuna geçtiğinde aldığı ödülü ifade eder. Gelecekteki ödüllerin anlık değeri üzerindeki etkiyi belirleyen katsayı ise azaltma faktörüdür ve $\gamma \in [0,1]$ ile ifade edilir. Bu yapıya göre ajan, bulunduğu S_t durumunda politikasına göre bir A_t eylemi seçer. Çevreden bu eyleme karşılık bir R_{t+1} ödülü alır ve ajan, yeni bir S_{t+1} durumuna geçer. Ajan, yeni durumu gözlemleyerek öğrenme algoritması doğrultusunda politikasını günceller ve süreci tekrar eder. Politika, ajanın belirli bir durumda hangi eylemi

seçeceğini belirleyen stratejidir ve genellikle π ile gösterilir. Politika, deterministik ya da stokastik olabilir. Deterministik politikada belirli bir durumda her zaman aynı aksiyon seçilir. Stokastik politikada ise her aksiyon bir olasılık dağılımına göre seçilir. Ajanın politikayı ne kadar iyi uyguladığını değerlendiren temel kavram, durum-değer fonksiyonudur ve Denklem (9)'da gösterilmiştir (Plaet, 2022).

$$[V^\pi(s) = E_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s]] \quad (9)$$

Ajanın belirli bir durumda belirli bir aksiyon seçtiğinde, gelecekte ne kadar ödül beklediğini tahmin eder. Burada $V^\pi(s)$, ajanın s durumundayken π politikasını takip ederek uzun vadede alacağı beklenen ödül değeridir. γ ise daha önce bahsedildiği gibi gelecekteki ödüllerin etkisini belirleyen azaltma faktörüdür. Belirli bir durumda belirli bir aksiyon seçildiğinde ve sonrasında politikaya sadık kalındığında elde edilecek olan toplam ödül ise aksiyon-değer fonksiyonu ile ifade edilir ve Denklem (10)'da gösterilmiştir (Plaet, 2022).

$$Q^\pi(s, a) = E_\pi[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a] \quad (10)$$

Bu fonksiyon, belirli bir eylemin o anki durumda uzun vadeli katkısının ölçülmesini sağlar. Politika π ile birlikte bu değer fonksiyonları, ajan için hem karar verme hem de öğrenme mekanizmasının temelini oluşturur. Pekiştirmeli öğrenme algoritmaları, bu değer fonksiyonlarını değer tabanlı yöntemlerle doğrudan öğrenebilir ya da politika tabanlı yöntemlerle yalnızca politikayı iyileştirmeye çalışabilir. Bahsedilen değer fonksiyonlarının davranışlarını temsil eden matematiksel temeli Bellman eşitliği oluşturur. Bir politikaya göre herhangi bir durumun değeri, o durumdayken alınan ödül ile sonraki durumda elde edilmesi beklenen değer toplamıdır (Plaet, 2022). Bu tanıma göre durum-değer fonksiyonu için Bellman eşitliği Denklem (11) ile ifade edilir:

$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \sum_{s' \in \mathcal{S}} \sum_{r \in \mathcal{R}} p(s', r|s, a) [r + \gamma V^\pi(s')] \quad (11)$$

Eşitliğe göre $\pi(a|s)$ ifadesi, ajan s durumundayken a eyleminin seçilme olasılığı yani politikayı belirtir. Çevrenin modeli ise s durumunda a eylemiyle s' durumuna geçildiğinde r ödülü alma olasılığı $p(s', r|s, a)$ ile ifade edilir. Azaltma faktörü γ ve bir sonraki durumun değeri $V^\pi(s')$ ile gösterilmiştir. Bir durum ve aksiyon çiftinin değeri ise o eylemi gerçekleştirdikten sonra alınan ödül ile ulaşılan yeni durumda uygulanacak politikanın getirisinin toplamıdır (Plaat, 2022). Bu durumda aksiyon-değer fonksiyonu için Bellman eşitliği Denklem (12) ile ifade edilir.

$$Q^\pi(s, a) = \sum_{s' \in \mathcal{S}} \sum_{r \in \mathcal{R}} p(s', r|s, a) \left[r + \gamma \sum_{a' \in \mathcal{A}} \pi(a'|s') Q^\pi(s', a') \right] \quad (12)$$

Aksiyon-değer fonksiyonu, ajan için durum-aksiyon çiftlerinin değerini öğrenmesini sağlar. Ajan s durumundayken a eylemini seçtiğinde ve π politikasını kullandığında elde etmeyi beklediği toplam indirgenmiş ödül $Q^\pi(s', a')$ ile gösterilmiştir (Plaat, 2022).

Bu noktaya kadar Markov Karar Süreci çerçevesinde PÖ temelleri, durum-değer ve aksiyon-değer fonksiyonları ile bu fonksiyonların Bellman eşitlikleri üzerinden nasıl biçimlendiği detaylıca anlatılmıştır. Ancak bu yapının etkin biçimde çalışabilmesi için ajanın çevre ile etkileşime girerek bu fonksiyonları nasıl öğreneceği yani değeri sayısal olarak nasıl tahmin edeceği de tanımlanmalıdır. Teorik olarak Bellman eşitlikleri, optimal politikanın elde edilmesine dair güçlü bir temel oluştursa da çoğu durumda geçiş olasılıkları veya çevre modeli gibi çevresel parametreler önceden bilinmez. Bu nedenle modelin bilinmediği ortamlarda, bu değerleri doğrudan deneyimlerden yola çıkarak yaklaşık olarak öğrenen algoritmalar geliştirilmiştir. Bu bağlamda duruma ve eyleme karşılık gelen Q -değerlerini doğrudan öğrenmeyi amaçlayan ve çevre modeline ihtiyaç duymayan temel algoritmalarından biri olan Q -Öğrenme yöntemi ortaya çıkmıştır.

1.5.4. Q Öğrenme

Q öğrenme, çevrenin geçiş modeli veya ödül dağılımına ihtiyaç duymadan optimal bir politika öğrenmeyi hedefleyen modelden bağımsız bir PÖ algoritmasıdır. Bu yönüyle MKS'de kullanılan Bellman denkleminin doğrudan uygulanamayacağı durumlarda önemli bir avantaj sağlar. Q -öğrenme, zamansal farka dayalı öğrenme olup her deneyimden sonra

Q değerlerini güncelleyerek zaman içinde optimal aksiyon-değer fonksiyonuna yakınsamayı amaçlar.

Bu yöntemde her durum-aksiyon çifti için beklenen toplam getiri, bir tablo yapısında saklanır ve bu tabloya Q tablosu denir. Ajan, bulunduğu durum için Q tablosuna bakarak hangi eylemin daha fazla ödül sağlayacağını tahmin eder ve bu tahmin doğrultusunda karar verir. Öğrenme süreci ilerledikçe tablo, çevreden alınan ödül ve geçiş bilgilerine göre güncellenir. Böylece ajan, daha önce keşfettiği deneyimlere dayanarak karar alma kabiliyetini geliştirir. Q-değerlerinin güncellenmesi Denklem (13)'teki şekilde gerçekleştirilir:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \cdot \delta t \quad (13)$$

$Q(s_t, a_t)$ ajanın s_t durumunda a_t eylemini seçtiğinde ne kadar ödül alacağını tahmin ettiği değer, α öğrenme oranı, δt zamansal fark hatası yani yeni gözlemden elde edilen hedef Q değeri ile tahmin edilen Q değeri arasındaki fark olarak tanımlanır. Zamansal fark hatası ise şu Denklem (14) ile açıklanır:

$$\delta t = \left[r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right] \quad (14)$$

Burada s_t ajanın t anındaki mevcut durumunu, a_t bu durumda seçilen eylemi, r_{t+1} bu eylemin çevreden aldığı anlık ödülü, s_{t+1} aksiyon sonrası ulaşılan yeni durumu, $Q(s_t, a_t)$ mevcut durum-aksiyon için uzun vadeli ödülün miktarı, $\max_{a'} Q(s_{t+1}, a')$ yeni durumda seçilebilecek eylemlerin tamamı için beklenen en yüksek ödül, γ ise azaltma faktörünü temsil eder. Bu formül, mevcut Q değerini önceki değeriyle yeni örneklemden elde edilen tahmini değer ağırlıklı ortalamasıyla günceller.

Güncelleme sırasında ajan, öncelikle mevcut $Q(s_t, a_t)$ tahminini dikkate alır. Ardından aldığı ödül ve bir sonraki durumdaki maksimum Q değerine ($\max_{a'} Q(s_{t+1}, a')$) bakarak hedef değeri belirler. Bu süreçte kullanılan ($\max_{a'} Q(s_{t+1}, a')$) terimi yalnızca bir sonraki adımı değil aynı zamanda geçmişte öğrenilmiş uzun vadeli ödül tahminlerini de kapsayan bir deneyimi temsil eder. Dolayısıyla güncelleme, her ne kadar yalnızca bir adım ileriye dönük gibi görünse de bu değer, zincirleme biçimde daha uzun vadeli beklentileri de

içine alır. Böylece mevcut tahmin ile hedef değer arasındaki fark hesaplanarak bu fark kadar bir düzeltme yapılır. Bu düzeltme, Q tablosundaki ilgili hücreye doğrudan uygulanır yani tablo üzerinde s_t, a_t konumunda bulunan Q değeri, elde edilen yeni bilgiye göre yerine yazılır. Bu güncelleme süreci, ajan çevreyle sürekli etkileşim hâlindeyken tekrarlanır ve zamanla Q tablosundaki tüm değerler daha doğru tahminlere ulaşacak şekilde güncellenir. Böylece tablo giderek en iyi politika için optimal Q değerlerini yansıtmaya başlar (Plaatt, 2022).

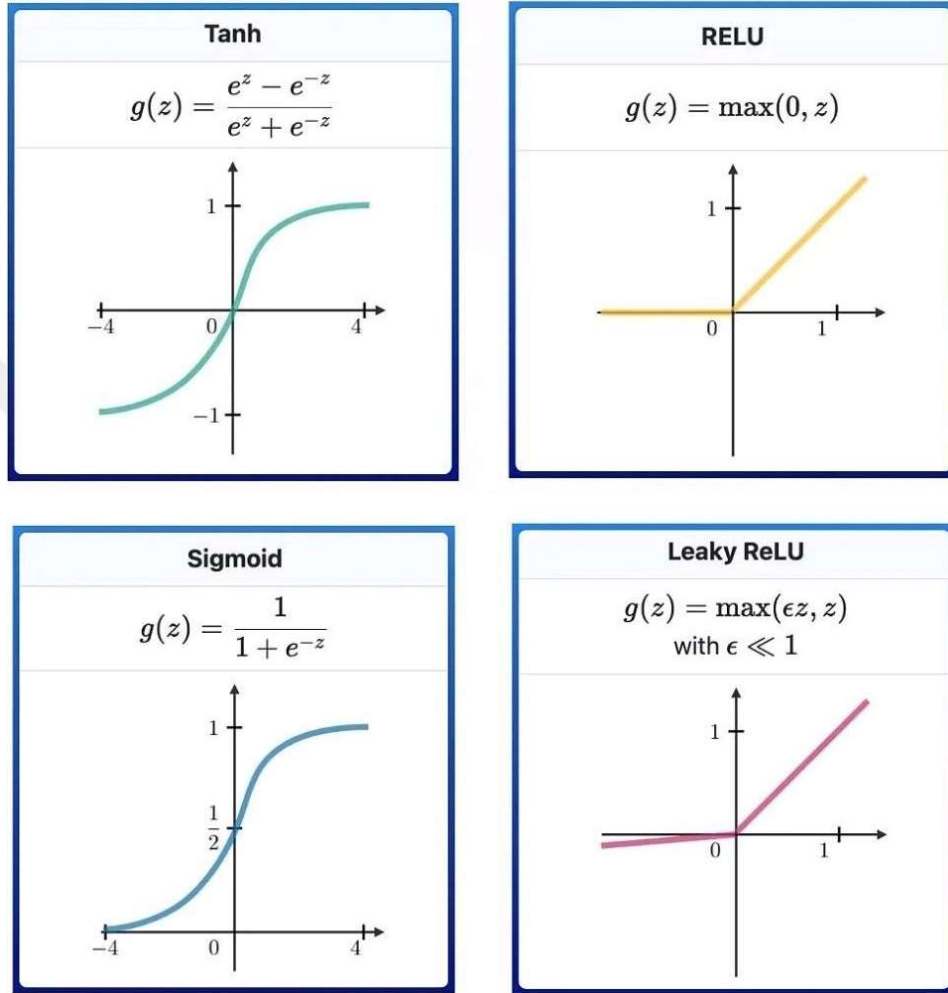
Ancak tablo tabanlı bu yapı yalnızca durum ve aksiyon uzayının küçük olduğu problemler için uygundur. Durum sayısı ya da aksiyon sayısı büyüdükçe tablo boyutu da katlanarak artar. Bu durumda hem bellek kullanımı hem de hesaplama maliyetleri ciddi oranlarda artar. Ayrıca daha önce hiç karşılaşılmamış bir duruma ilişkin Q değeri bulunamayacağından genelleme yapmak mümkün değildir. Bu problemler dikkate alındığında Q tablosunun yerine yaklaşım fonksiyonu olarak yapay sinir ağlarının kullanıldığı bir yöntem olan Derin Q Ağları (DQA) önerilmiştir. Bu yöntemde sinir ağı, Q tablosunun yerini alır ve her durumu girdi olarak alıp tüm olası eylemlerin Q değerlerini çıktı olarak üretir. Bu sayede yüksek boyutlu durum uzaylarında benzer durumlar arasında genelleme yapabilir ve klasik Q öğrenmenin durum-aksiyon tablosunun çok büyümesiyle ortaya çıkan ölçeklenebilirlik problemi aşılmış olur.

1.5.5. Derin Sinir Ağları

Derin öğrenmede kullanılan derin sinir ağları; temel olarak bir girdi katmanı, bir veya daha fazla gizli katman ve bir çıktı katmanından oluşur. Her bir katman, bir önceki katmandan aldığı verileri işleyerek bir sonraki katmana aktarır ve bu işlem ileri yayılım olarak adlandırılır.

Ajanın içinde bulunduğu s durumu, bir gözlem verisi olarak sinir ağına verilir ve veri ilk katmandaki nöronlara iletilir. Her nöron, aldığı girdiye bir w ağırlığı ve bir b biası uygular. Ardından bir aktivasyon fonksiyonu ile çıktı üretir. Yapay sinir ağlarında aktivasyon fonksiyonları, her bir nöronun ürettiği çıktıya doğrusalsızlık kazandırmak amacıyla kullanılır. Eğer ağı tüm katmanları yalnızca doğrusal işlemlerden oluşsaydı kaç katman eklenirse eklensin ağı genel davranışı yine doğrusal bir fonksiyon olurdu. Bu durumda daha karmaşık yapıları ya da verilerdeki ince desenleri ayırt etme yeteneği kazanamazdı. Aktivasyon fonksiyonları sayesinde ağı; girişler ile çıktılar arasındaki

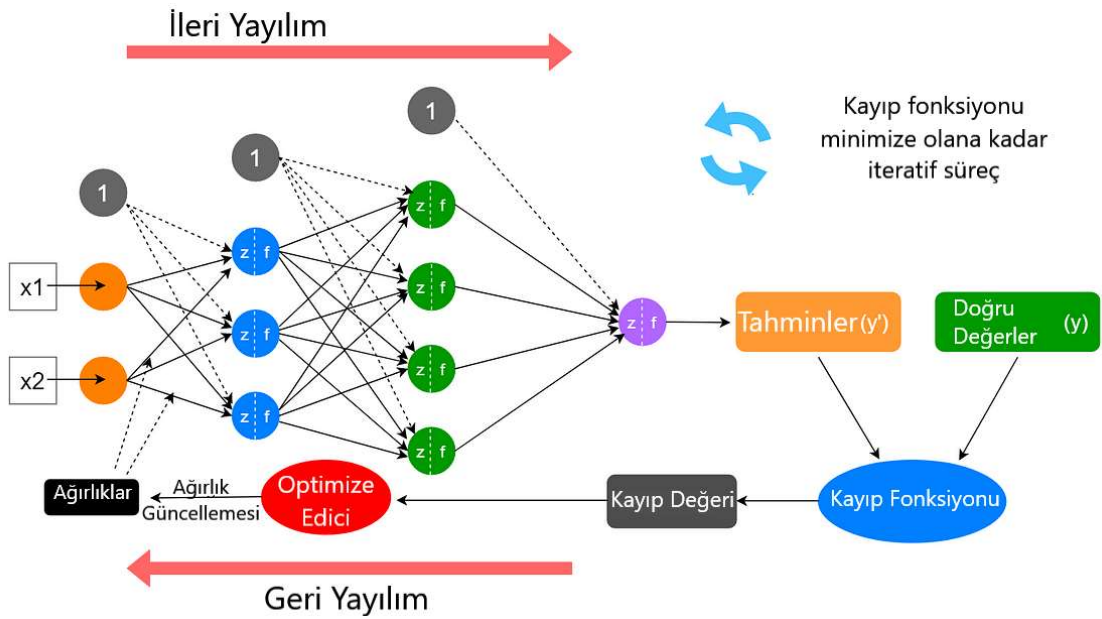
karmaşık, doğrusal olmayan ilişkileri modelleyebilir hâle gelir. Aktivasyon fonksiyonları, Şekil 10'da görüldüğü gibi gelen değerleri fonksiyonun cinsine göre değiştirerek bir çıktı değeri sağlar.



Şekil 10. Aktivasyon fonksiyonları (URL-7, 2025)

Örneğin RELU fonksiyonu, sıfırın altındaki tüm değerleri sıfıra eşitlerken sıfırın üstündeki değerleri olduğu gibi bırakır. Böylece ağıın lineer olması önlenir. Ayrıca aktivasyon fonksiyonları, ağıın katmanları arasında bilgi dönüşümünü düzenler; ağıın karar sınırlarını esnetmesini sağlar ve geri yayılım sırasında türevlenebilirlik özelliği ile gradyan hesaplamasına olanak tanır. Özellikle sigmoid, tanh ve ReLU gibi fonksiyonlar hem diferansiyellenebilir yapıları hem de bilgi akışını yönlendirebilmeleri nedeniyle derin öğrenme modellerinde yaygın olarak tercih edilirler (Nwankpa vd., 2018).

Derin sinir ağlarında ağırlıkların güncellenmesi için ağıın tahmin ettiği çıktı değeri ile elde edilen asıl değerin karşılaştırılarak aradaki farktan doğan hatanın hesaplanması gerekir. Bu hata, kayıp fonksiyonu ile ölçülür. Hesaplanan hata kullanılarak, sinir ağının tüm katmanlarında geriye doğru iletilerek her ağırlığın ne kadar değişmesi gerektiği hesaplanır ve buna da geri yayılım adı verilir. Geri yayılımda, çıktıyla beklenen değer arasındaki farktan doğan hata hesaplanarak zincirleme türev kuralları yardımıyla ağıın ağırlıklarına yayılır ve parametreler güncellenir. Bahsedilen işlemler Şekil 11’de gösterilmiştir.



Şekil 11. Yapay sinir ağı güncelleme işlemleri (URL-10, 2025)

Matematiksel olarak gösterecek olursak elimizde bir giriş katmanı, iki gizli katman ve bir çıkış katmanlı bir ağ olsun. Bu durumda:

- Giriş katmanı: x (veri)
- 1. Gizli Katman: ağırlık $W^{[1]}$, bias $b^{[1]}$, aktivasyon $a^{[1]}$
- 2. Gizli Katman: ağırlık $W^{[2]}$, bias $b^{[2]}$, aktivasyon $a^{[2]}$
- Çıkış katmanı: ağırlık $W^{[3]}$, bias $b^{[3]}$, çıktı $\hat{y} = a^{[3]}$

Burada ortalama kare hatası kullanılan kayıp fonksiyonu Denklem (15) ile gösterilir:

$$L = \frac{1}{2} (y_{\text{hedef}} - y_{\text{tahmin}})^2 \quad (15)$$

Öncelikle ileri yayılıma göre giriş değerleri, ağırlıklarla çarpılır ve bias değerleri eklenir.

İlk katman için:

$$z^{[1]} = W^{[1]}x + b^{[1]} \quad (16)$$

$$a^{[1]} = f(z^{[1]}) \quad (17)$$

İkinci katman için:

$$z^{[2]} = W^{[2]}a^{[1]} + b^{[2]} \quad (18)$$

$$a^{[2]} = f(z^{[2]}) \quad (19)$$

Çıkış katmanı için:

$$z^{[3]} = W^{[3]}a^{[2]} + b^{[3]} \quad (20)$$

$$a^{[3]} = \hat{y} = f(z^{[3]}) \quad (21)$$

Böylece x girdisine göre $\hat{y}$ çıktısı yani Q değeri hesaplanmış olur. Fakat asıl Q değeri ile ağın çıktısı olan Q değerinin birbirine yakınsaması için ağ parametrelerinin güncellenmesi gerekir ve burada geri yayılım devreye girer. Zincir kuralına göre kayıp fonksiyonunun önce çıktıya göre türevi, sonra her katmandaki aktivasyonlara göre türevi, sonra da o katmandaki ağırlıklara göre adım adım türevi alınır.

-n. katmanı için;

Hata sinyali:

$$\delta^{[n]} = (\hat{y} - y) \cdot f'^{[n]}(z^{[n]}) \quad (22)$$

Gradyan Hesabı:

$$\frac{\partial L}{\partial W^{[n]}} = \delta^{[n]} \cdot (a^{[n-1]})^T, \frac{\partial L}{\partial b^{[n]}} = \delta^{[n]} \quad (23)$$

Böylece çıkış katmanının ağırlıkları ve bias değerinin kayıp fonksiyonu üzerindeki etkisi gradyan ile hesaplanmış olur. Aynı hesaplamalar diğer katmanlar için de yapıldıktan sonra ağırlıkların güncellenmesi şu şekilde gerçekleşir:

Tüm katmanlar için:

$$W^{[n]} \leftarrow W^{[n]} - \alpha \cdot \frac{\partial L}{\partial W^{[n]}} \quad (24)$$

$$b^{[n]} \leftarrow b^{[n]} - \alpha \cdot \frac{\partial L}{\partial b^{[n]}} \quad (25)$$

Böylece geri yayılım algoritması sayesinde her bir katmandaki ağırlıklar, çıktıdaki toplam hataya olan etkilerine göre güncellenir. Bu sistematik güncelleme mekanizması sayesinde sinir ağı, zamanla istenen çıktı ile ürettiği çıktı arasındaki farkı azaltmayı öğrenir ve doğruluğu yüksek tahminler yapabilecek şekilde eğitilmiş hale gelir (Plaatt, 2022).

1.6. Derin Q Ağı

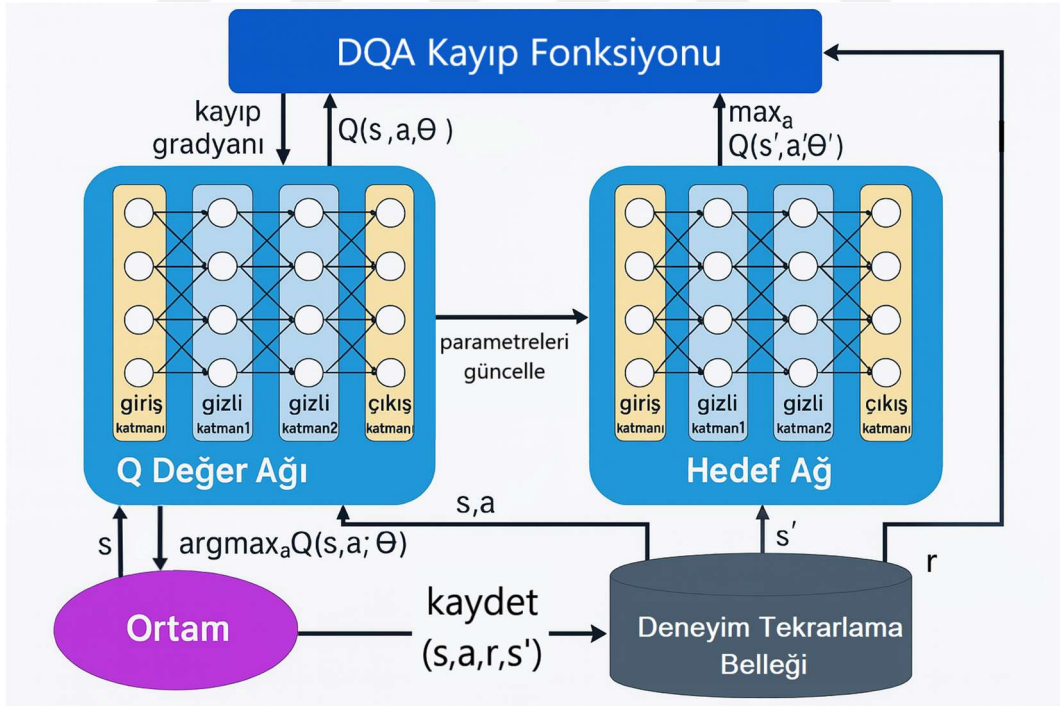
Klasik Q öğrenme algoritmasında her bir durum-aksiyon çifti için beklenen ödül değeri, doğrudan bir Q tablosunda saklanır. Ancak bu yapı, yüksek boyutlu durum uzaylarında hem bellek hem de genelleme açısından yetersiz kalmaktadır. Bu problemi çözmek amacıyla önerilen Derin Q Ağı algoritmasında Q değerlerinin hesaplanması, Q tablosu yerine yapay bir sinir ağı (YSA) tarafından gerçekleştirilir. Bu ağ, verilen bir s durumu için tüm eylemlere karşılık gelen Q değerlerini çıktı olarak üretir.

DQA algoritmasında öğrenmenin daha kararlı ve dengeli bir şekilde ilerlemesini sağlamak için ana ağ ve hedef ağ olmak üzere iki ayrı sinir ağı kullanılır. Ana ağ, ajan tarafından gerçek zamanlı olarak kullanılan ağdır ve her durumda hangi aksiyonun seçileceğine karar verir. Ajan, içinde bulunduğu durumun gözlem vektörünü ana ağa verdiğinde ağ, her olası eylemin Q değerlerini hesaplar ve en yüksek değere sahip aksiyonu seçerek uygular. Hedef ağ ise sadece hedef Q değeri hesaplamasında kullanılır. Bu ağ, doğrudan öğrenme sürecine dahil edilmez ve sadece Q güncelleme hedefini üretmek amacıyla kullanılır. Böylece hem hedef hem de tahmin aynı ağdan alınmadığı için geri

yayılım sırasında ortaya çıkabilecek sapmalar azaltılır ve daha kararlı bir öğrenme elde edilir. Hedef ağıın parametreleri, belirli bir zaman aralığında ve kademeli olarak ana ağıın parametreleriyle güncellenir. Bu güncelleme sayesinde hedef Q değerlerinin ani değişikliklerden etkilenmesi önlenir. Öğrenme sürecinde ayrıca deneyim tekrarlama (experience replay) kullanılır. Ajanın çevre ile etkileşimi sonucunda elde ettiği her bir durum, aksiyon, ödül ve bir sonraki durum bir bellekte saklanır. Eğitim sırasında bu bellekteki deneyimlerden rastgele örnekler Denklem (26)'da görüldüğü gibi seçilerek eğitim verisi oluşturulur. Bu sayede korelasyonlu örneklerle eğitimin sapması engellenmiş olur ve geçmişteki deneyimlerin tekrar değerlendirilmesi sağlanır.

$$D = \{(s_i, a_i, r_i, s'_i)\} \quad (26)$$

Bu yapı aynı zamanda veri verimliliğini de artırır çünkü her deneyimden birden fazla kez öğrenme yapılabilir. Bahsedilen yapı Şekil 12'de gösterilmiştir.



Şekil 12. DQA algoritması (URL-9, 2025)

DQA'da öğrenme adımları Şekil 12'de görüldüğü gibi tekrarlama belleğinden (replay buffer'dan) rastgele seçilen her deneyim kümesi için öncelikle mevcut durum ve aksiyon çifti ana ağa verilerek tahmini Q değeri hesaplanır.

$$Q(s_t, a_t; \theta) \quad (\text{Tahmin}) \quad (27)$$

Ardından bir sonraki durum hedef ağa verilerek tüm aksiyonların Q değerleri elde edilir ve bu değerler arasından en yüksek olanı seçilerek hedef Q değeri Denklem (28) şeklinde hesaplanır.

$$y_t = r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-) \quad (28)$$

Kayıp fonksiyonu, bu hedef ile tahmin edilen Q değeri arasındaki farkın karesi olarak belirlenir. Bu kayıp, geri yayılım yöntemiyle ana ağın ağırlıklarının güncellenmesinde kullanılır.

$$L(\theta) = (y_t - Q(s_t, a_t; \theta))^2 \quad (29)$$

Bu süreç tekrarlanarak ağ, zamanla optimal Q değerlerine yakınsamaya başlar ve öğrenme gerçekleşir (Plaatt, 2022).

1.6.1. Epsilon-Greedy Dengesi

PÖ algoritmalarında ajanların çevreyle etkileşim kurarak bilgi edinmeleri, temel öğrenme sürecini oluşturur. Bu bağlamda epsilon-greedy yöntemi, öğrenme ve keşif arasında denge kurmak amacıyla yaygın şekilde kullanılan bir yöntemdir. Özellikle DQA gibi değer tabanlı algoritmalarda bu yöntem oldukça önemlidir. Epsilon-greedy politikasında ajan, her zaman en yüksek ödülü vereceği tahmin edilen aksiyonu seçmek yerine her adımda ϵ (epsilon) olasılığıyla rastgele bir aksiyon seçerken $(1 - \epsilon)$ olasılığıyla Q-ağının en yüksek değeri önerdiği aksiyonu uygular. Bu sayede hem keşif hem de sömürü aynı anda yürütülmüş olur. Keşif, ajanın henüz hakkında yeterli bilgiye sahip olmadığı durumları ve aksiyonları

denemesi anlamına gelir. Ajan, bu rastgele seçimlerle ortamın daha önce karşılaşmadığı bölgelerini keşfeder ve daha iyi politikaların var olup olmadığını öğrenme şansı elde eder. Keşif sayesinde ajan, kısa vadede ödül getirmeyen ama uzun vadede yüksek kazanç sağlayabilecek alternatifleri öğrenebilir (Plaata, 2022).

Sömürü ise ajanın mevcut bilgilerine dayanarak kendisine en yüksek ödülü getireceği tahmin edilen aksiyonu seçmesidir. Bu durumda ajan, daha önce denediği ve başarılı sonuçlar aldığı eylemleri tekrarlar ve öğrenme sürecinden ziyade kazanç elde etmeye odaklanır. Eğitimin ilk bölümlerinde epsilon değeri genellikle 1 gibi yüksek bir değer seçilir. Bu da ajanın başlangıçta tamamen rastgele aksiyonlar seçmesine neden olur. Bu durumda ajan, başlangıçta ortam hakkında bilgi sahibi olmamasına rağmen rastgele aksiyonlar seçerek farklı durumlarla karşılaşır ve deneyim havuzunu zenginleştirir. Bu aşamada olumsuz sonuçlar da dahil olmak üzere farklı ödül-ceza kombinasyonları gözlemlenir ve öğrenme tabanı oluşturulur. Ancak epsilon değerinin sürekli yüksek kalması, ajanın öğrenme sürecinde optimal politikaya ulaşmasını engelleyebilir. Sürekli rastgele seçim yapan bir ajan, elde ettiği bilgiyi yeterince kullanamayıp kararsız davranışlar sergiler. Öte yandan epsilon değeri çok düşük tutulursa ajan, yalnızca geçmişte ödül getirdiğini öğrendiği aksiyonları seçmeye başlar. Bu durum, ajanın daha iyi olasılıkları keşfetmesini engeller ve sistemin yerel maksimumlara takılmasına yol açabilir. Bu nedenle epsilon değerinin kontrollü bir şekilde azaltılması, öğrenmenin ilerleyen aşamalarında daha kararlı ve verimli bir davranış biçimi geliştirilmesini sağlar. Genellikle doğrusal veya üssel azalma stratejileri tercih edilir. Böylece ajan, başlangıçta keşif yaparak çevreyi tanır ve sonrasında ise edindiği bilgilere dayanarak daha kararlı aksiyonlar almaya başlar (Plaata, 2022) .

DQA algoritmasının yapısı gereği aksiyon seçiminde epsilon-greedy stratejisini doğrudan içinde barındırır. Bu yöntem sayesinde ajan, öğrenme süreci boyunca deneme-yanılma yoluyla hem çevreyi tanımakta hem de zamanla ödülü maksimize edecek politikayı öğrenmektedir. DDPG, SAC gibi sürekli aksiyon uzaylı diğer algoritmalarda keşif için genellikle Ornstein-Uhlenbeck veya Gaussian noise gibi gürültü eklenmesi tercih edilirken DQA'da bu denge epsilon üzerinden sağlanır (Plaata, 2022).

Sonuç olarak epsilon-greedy stratejisi, DQA tabanlı sistemlerde keşif ve sömürü arasında etkin bir denge kurarak öğrenmenin başarısını doğrudan etkileyen temel bileşenlerden biridir. Bu dengenin doğru şekilde kurulması, ajanın hem çevreyi yeterince keşfetmesini hem de öğrendiği bilgiyi kararlı biçimde uygulamasını mümkün kılar.

1.6.2. Ödül Fonksiyonunun Seçimi

PÖ’de ödül fonksiyonu, ajanın hedef davranışları öğrenmesinde yönlendirici en temel etkenlerden biridir. Ajan, çevresiyle etkileşim kurarken yalnızca gözlemler ve bu gözlemler sonucunda gerçekleştirdiği aksiyonlara karşılık aldığı ödüller üzerinden kararlarını yönlendirir. Her aksiyonun sonucunda alınan ödül veya ceza, ajanın gelecekte aynı durumda benzer bir aksiyonu tekrarlayıp tekrarlamayacağına karar vermesini sağlar (Plaat, 2022). Bu nedenle ödül fonksiyonu, sistemin istenen davranışı öğrenip gerçekleştirebilmesi için ajanın hangi durumda neyi yapıp neyi yapmamasını istediğimize göre matematiksel olarak tasarlanmalıdır. Tasarlanan ödül fonksiyonu, yalnızca hedefe ulaşıldığında ödül vermek gibi çok sade olabilir. Fakat bu durumda ajanın hedefe ulaşması çok uzun sürebilir ya da hiç ulaşamayabilir. Bu tür ödül yapıları, özellikle durum uzayı genişse ve hedef nadiren ulaşılan bir noktadaysa öğrenmeyi büyük ölçüde yavaşlatır. Bu sorunu çözmek adına hedefe yaklaşmayı teşvik eden ara ödül yapıları kullanılabilir. Örneğin ajan, her adımdaki konum hatasına (hedefle arasındaki mesafe farkına) göre ödül veya ceza alacak şekilde yapılandırılırsa yalnızca hedefe ulaştığında değil, hedefe yaklaşırken de pozitif ödül alır. Benzer şekilde hedeften uzaklaştığında cezalandırılırsa ajan bu davranıştan da kaçınmayı öğrenir. Böylece ajan, rastgele hareketler yerine istikrarlı bir şekilde hedefe yönelme eğilimi kazanır. Ancak ödül fonksiyonunun dikkatli tasarlanmaması durumunda ajanın ödül fonksiyonundaki açıkları kullanarak hatalı yollarla yüksek ödül alması da mümkündür. Örneğin ödül fonksiyonu 3 parçadan oluşuyorsa ve bunlardan yalnızca biri yüksek ödül getiriyor ve diğer ikisi ise küçük cezalar veriyorsa ajan, ödül getiren kısmı maksimize etmeye çalışabilir. Bu durum, toplamda sistem açısından istenmeyen bir davranışla sonuçlanabilir. Ajan, matematiksel olarak avantajlı görünen ancak mantıksal olarak hatalı bir strateji geliştirmiş olur. Bu nedenle ödül fonksiyonu yalnızca hedefi değil, hedefe ulaşma biçimini de yönlendirecek şekilde tasarlanmalıdır. Cezaların hangi durumlarda verileceği, ödül-ceza oranlarının dengesi ve her ödül bileşeninin etkisi dikkatle gözlemlenmelidir. Eğitim süresince ajanın davranışları analiz edilmeli, ortaya çıkan yanlış yönelimler incelenmeli ve gerektiğinde ödül fonksiyonu yeniden yapılandırılmalıdır. Bu süreç iteratif ve deneysel bir yaklaşımla yürütülmelidir.

Sonuç olarak ödül fonksiyonu, yalnızca sistemin hedefe ulaşmasını değil aynı zamanda bu hedefe nasıl ulaşacağını da belirler. Ajanın hangi durumlarda ödüllendirileceği, hangi eylemlerin tercih edilmemesi gerektiği gibi önemli kararlar bu fonksiyonla

şekillendirilir. Bu nedenle PÖ uygulamalarında başarılı sonuçlar elde edebilmek için ödül fonksiyonu dikkatle tasarlanmalı ve sistematik olarak test edilmelidir.

1.6.3. Hiperparametrelerin Seçimi

Pekiştirmeli öğrenmede, bir ajanın başarılı bir şekilde öğrenebilmesi; yalnızca gözlem vektörlerinin doğru seçimi ve ödül fonksiyonunun uygun tasarımıyla sınırlı değildir. Sistem davranışını büyük oranda belirleyen bir diğer temel unsur da hiperparametrelerdir. Hiperparametreler, öğrenme algoritmasının dışsal ayarları olarak düşünülebilir ve doğrudan öğrenme sürecinin dinamiklerine yön verirler. Ajanın çevreyi keşfetme hızı, geçmiş deneyimlerinden ne ölçüde faydalandığı, hangi sıklıkla güncelleme yaptığı, öğrenme oranı gibi birçok temel süreç bu parametrelerle kontrol edilir. Dolayısıyla hiperparametreler; öğrenmenin kararlılığı, hızı, genelleme kapasitesi ve optimum politikaya yakınsama başarısı üzerinde doğrudan etkilidir.

Özellikle bu çalışmada olduğu gibi, yüksek boyutlu gözlem ve aksiyon uzayına sahip bir KSPR'nin engele çarpmadan yol planlaması probleminde, uygun hiperparametre ayarları kritik bir öneme sahiptir. Bu tür ortamlarda, yanlış seçilen hiperparametreler yalnızca öğrenmeyi yavaşlatmakla kalmaz; aynı zamanda ajanın hiç öğrenememesine, öğrenmeyi erken sonlandırmasına ya da düşük kaliteli yerel optimum politikalara takılı kalmasına neden olabilir. Ayrıca kullanılan gözlemlerin çeşitliliği ve ödül fonksiyonunun yapısı ile hiperparametreler arasında karşılıklı bir denge ilişkisi bulunmaktadır. Örneğin, hedefe yaklaşmayı teşvik eden yönlendirici ödüllerin kullanıldığı bir ortamda ajanın hem kısa vadeli ödüllere hem de uzun vadeli ödüllere duyarlı hale getirilmesi gerekmektedir. Bu durumda öğrenme oranı, hedef ağ güncelleme sıklığı ve azaltma faktörü gibi hiperparametrelerin tanımlanan ödül fonksiyonuyla uyumlu ve dengeli şekilde seçilmesi kritik öneme sahiptir.

Sonuç olarak hiperparametreler, etkili bir öğrenme süreci üzerinde belirleyici bir rol oynar. Bu nedenle her uygulama ve senaryo için hiperparametrelerin özenle belirlenmesi gerekir. Parametre seçimi rastgele değil; sistematik denemeler, sayısal analizler ve eğitim süresince ajanın sergilediği davranışların dikkatle izlenmesi yoluyla yapılmalıdır. Bu gözlemler doğrultusunda hiperparametreler, kademeli bir şekilde optimize edilerek daha kararlı ve verimli bir öğrenme sağlanır.

1.6.4. Gözlemlerin Belirlenmesi

PÖ sistemlerinde bir ajanın ortam hakkında bilgi edinmesini ve hedefe ulaşmasını sağlayan temel unsurlar, gözlemler ve ödül fonksiyonudur. Ajan, ortamın dinamiklerine veya modeline dair doğrudan bir bilgiye sahip değildir. Bunun yerine sadece aldığı gözlemler üzerinden aksiyonlar üretir ve bu aksiyonların sonucunda aldığı ödüller aracılığıyla davranışlarını günceller. Bu nedenle verilen gözlemlerin ortamı doğru bir şekilde temsil etmesi, öğrenme açısından kritik öneme sahiptir.

Ajanın temel amacı, belli bir durumda hangi aksiyonu seçtiğinde hangi ödülü aldığını öğrenerek uzun vadede en yüksek toplam ödülü elde edecek politikayı geliştirmektir. Bu sürecin sorunsuz ilerlemesi için gözlem vektörlerinin ajan tarafından anlamlı bir şekilde işlenebilmesi gerekir. Ancak gözlem boyutunun fazla olması, her zaman daha iyi sonuçlara yol açmaz. Yüksek boyutlu gözlem uzayları, öğrenme sürecini zorlaştırabilir. Çünkü bu durumda, gözlemlerin işlenmesi için daha karmaşık ve geniş yapay sinir ağları gerekir. Bu da hem hesaplama maliyetini artırır hem de aşırı öğrenme veya genelleme zayıflığı gibi sorunlara yol açabilir. Ayrıca eğitim süresi ciddi şekilde uzayabilir ve kaynak ihtiyacı önemli ölçüde artabilir. Bu nedenle PÖ sistemlerinde gözlem uzayı, dikkatli ve dengeli şekilde tasarlanmalıdır. Ajanın görevini yerine getirebilmesi için sadece anlamlı bilgileri içeren optimum gözlem kümesi sunulmalıdır. Gözlem tasarımı yapılırken gözlemlerin; ajanın çevresel durumu yeterince temsil edebilmesi, gözlem vektöründeki her bileşenin öğrenme sürecine katkı sağlaması, boyutsal olarak verimli ve öğrenilebilir olması göz önünde bulundurulması gereken kriterler arasındadır (Plaat, 2022).

Sonuç olarak gözlemler ne kadar anlamlı ve iyi yapılandırılmış olursa ajan da o kadar verimli, hızlı ve doğru bir şekilde öğrenme sürecini tamamlayabilir.

1.6.5. Azaltma Faktörü Seçimi

Pekiştirmeli öğrenme algoritmalarında en kritik hiperparametrelerden biri azaltma faktörüdür. Bu parametre, ajanın karar verirken ne ölçüde uzun vadeli ödülleri dikkate alacağını belirler. Diğer bir deyişle ajan, yalnızca anlık ödüllere mi yoksa gelecekteki olası ödülleri de hesaba katarak mı hareket edeceğini gamma değeriyle belirler. Bu durum, Bellman denkleminde şu şekilde ifade edilir:

$$Q(s, a) = r + \gamma \cdot \max_{a'} Q(s', a') \quad (30)$$

Burada $Q(s, a)$ ajanın mevcut durumda s durumundayken a aksiyonunu almasının beklenen değeridir. Alınan anlık ödül r , bir sonraki durum s' , bir sonraki adımda seçilebilecek aksiyonlar a' , azaltma faktörü ise γ ile gösterilmiştir. Denklemden görüldüğü gibi γ katsayısı, gelecekte alınacak maksimum Q değerinin anlık karar üzerindeki etkisini belirler. Eğer γ değeri 0.99 gibi yüksek seçilirse ajan uzun vadeli ödülleri daha fazla dikkate alır ve stratejik, planlı davranışlar sergiler. Buna karşılık γ değeri 0.5 gibi düşük bir değerdeyse ajan daha çok kısa vadeli ödüllere odaklanır ve açgözlü davranışlar sergileyerek anlık en yüksek ödül getiren aksiyonları seçer. Bu dengeğin hassas bir şekilde ayarlanması gerekmektedir. Eğer azaltma faktörü çok yüksek seçilirse ajan sürekli gelecek ödüllere odaklanarak anlık ödülleri görmezden gelebilir. Bu durum öğrenmenin yavaşlamasına, önündeki engeli fark edememesine veya davranışlarının kararsızlaşmasına neden olabilir. Azaltma faktörünün çok düşük seçilmesi de ajanı sadece kısa vadeli ödüllere odaklayarak asıl hedefine ulaşmasına engel olabilir. Ajan, yerel maksimumlara takılabilir. Örneğin bir hedefe giden birden fazla yol varsa ajan anlık getirisi yüksek olan yolu tercih edebilir fakat diğer muhtemel daha kısa, verimli ve uzun vadede yüksek ödül getirecek yolları öğrenemez (Plaatt, 2022).

Sonuç olarak azaltma faktörü, ajanın öğreneceği politikada büyük öneme sahip olup hassas bir şekilde ayarlanmalıdır. Özellikle bu tez çalışmasındaki gibi büyük gözlem uzayı ve aksiyon uzayına sahip yol planlama çalışmalarında bu dengeyi yakalamak kritik öneme sahiptir.

1.6.6. Öğrenme Oranı Seçimi

Özellikle Q -öğrenme temelli algoritmalarda, ajan her zaman çevresiyle etkileşime girerek öğrenmektedir. Ajan, belirli bir durumda (s_t) bir aksiyon (a_t) seçer ve bunun sonucunda hem yeni bir duruma geçer (s_{t+1}) hem de çevreden bir ödül (r_t) alır. Bu süreçte ajan, geçmişteki deneyimlerinden yola çıkarak hangi aksiyonların uzun vadede daha faydalı olduğunu öğrenmeye çalışır.

Bu öğrenme, Q değerlerinin sürekli olarak güncellenmesiyle gerçekleşir. Q değeri, belirli bir durumda belirli bir aksiyon seçildiğinde uzun vadede beklenen toplam ödülü temsil eder. İşte bu güncelleme sırasında öğrenme oranı (α) devreye girer. Yeni hesaplanan

Q değerinin eski Q değeriyle hangi oranda birleştirileceğini, yani eski değerin ne kadar korunup ne kadarının yeni bilgiye göre değişeceğini belirleyen katsayıdır. Matematiksel olarak bu süreç şu şekilde ifade edilir:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)] \quad (31)$$

Bu denklemde α katsayısı, yeni bilgi ile mevcut bilginin ne ölçüde harmanlanacağını belirler. Azaltma faktörü(γ), gelecekteki ödüllerin anlık değere etkisini kontrol eder. Köşeli parantez içindeki ifadenin tamamı ise hedef Q değeridir.

Öğrenme oranı yüksek seçildiğinde ajan, aldığı ödül ve yeni gözlemle birlikte Q değerini çok hızlı şekilde değiştirir. Bu sayede öğrenme, başlarda daha hızlı başlar. Ortam dinamikleri aniden değişirse, örneğin farklı hedefler veya değişen engeller mevcutsa, ajan yeni duruma daha hızlı uyum sağlayabilir. Fakat öğrenme oranı, olması gerekenden çok daha yüksek seçilmişse ajan yeni bilgilere fazla güvendiğinden Q değerleri değişken olur. Bu da kararsız politikalara yol açabilir. Anlık ödül sinyalleri gereğinden fazla etkili olur. Rastgele alınan yüksek bir ödül, Q değerini aşırı şekilde artırabilir. Ayrıca sürekli yeni deneyimlerle yüksek oranda güncellenen ağ, öğrendiği deneyimleri unutabilir.

Öğrenme oranı düşük seçildiğinde Q değerleri zamanla yavaş bir şekilde güncellenir. Bu durum, ajanın geçmiş deneyimlerini hızlıca unutmadan değerlendirmesine ve öğrenilen politikanın daha kararlı hale gelmesine olanak sağlar. Ajan, anlık ödül değişimlerinden aşırı etkilenmeden genellenebilir bir politika geliştirmeye çalışır. Bu sayede eğitim sürecinde iniş çıkışlar ve hatalı genellemeler azaltılmış olur. Ancak öğrenme oranının aşırı düşük belirlenmesi, bazı önemli sorunlara yol açabilir. Öncelikle, Q değerlerinin çok yavaş güncellenmesi nedeniyle öğrenme süreci ciddi şekilde yavaşlar. Ajan, uzun süre boyunca benzer davranışları sürdürür ve yeterli deneyim elde etse bile bu deneyimler model üzerinde yeterli etkiyi yaratamayabilir. Bu durum, yanlış öğrenilmiş davranışların uzun süre korunmasına ve hatalı stratejilerin düzeltilmesinde güçlüğü neden olur. Ek olarak, öğrenme ortamında meydana gelebilecek dinamik değişimlere karşı adaptasyon kabiliyeti de azalır. Ortamda hedefin yerinin değişmesi, yeni engellerin ortaya çıkması gibi durumlarda düşük öğrenme oranına sahip bir ajan, bu yeni koşullara oldukça geç yanıt verir. Sonuç olarak öğrenme oranı seçimi; öğrenme hızının, kararlılığın ve çevresel adaptasyonun dengelenmesi açısından kritik öneme sahiptir (Plaata, 2022).

1.6.7. Hedef Ağ Güncelleme Sıklığı ve Yumuşatma Faktörü

DQA algoritmasında hedef Q değerlerinin kararlı ve tutarlı olabilmesi için ayrı bir hedef ağ kullanılır. Bu hedef ağ, doğrudan güncellenmek yerine ana ağın parametreleriyle belirli aralıklarla ve belirli oranda güncellenerek daha yumuşak bir öğrenme süreci sağlar. Bunu sağlayan iki parametre güncelleme sıklığı ve yumuşatma faktörüdür. Hedef ağın güncellenmesi iki şekilde yapılabilir. Birisi ana ağın parametrelerinin hedef ağa belirli bir sıklıkla kopyalanmasıdır. Diğer seçenek ise belirli sıklıkla kopyalama işlemi yapılırken ana ağ parametrelerine $[0,1]$ aralığında belirli bir katsayı uygulanarak hedef ağ parametrelerini güncellemektir. Böylece ağ her güncellemede ana ağın parametrelerinin sadece belirlenen bir kısmı ile güncellenir ve şu şekilde ifade edilir:

$$\theta^- \leftarrow \tau \cdot \theta + (1 - \tau) \cdot \theta^- \quad (32)$$

Burada θ ana ağın parametrelerini, θ^- ise hedef ağın parametrelerini temsil eder. Bu ifade, hedef ağın yeni parametrelerinin ana ağ parametreleriyle bir birleştirme sonucu elde edildiğini gösterir. Buradaki $\tau \in [0,1]$ yumuşatma faktörünü ifade eder ve değeri ne kadar küçükse, hedef ağ, ana ağı o kadar yavaş takip eder. Hedef ağın her adımda güncellenmesi, yani çok sık güncelleme, hedef ağı ana ağa gereğinden fazla yaklaştırarak kararsızlığa neden olabilir. Bu durumda, hedef Q değerleri öğrenme sürecinde sürekli değiştiği için Bellman sabitliğini bozabilir. Sonuç olarak ağ, kararlı davranışlar sergileyemez ve optimal politikayı öğrenemez.

Eğer çok seyrek güncellemeler kullanılırsa hedef ağ, ana ağdan aşırı uzaklaştırır. Bu durumda hedef ağ, mevcut çevre dinamiklerine ve güncel politika değişikliklerine uyum sağlayamaz. Ajan, eski hedef değerlere göre davranış geliştirdiğinden dolayı aldığı ödül ile beklenen Q değeri arasında ciddi farklar oluşur. Bu fark büyür ve ağın öğrenmesi güçleşir (Plaat, 2022). Eğer çok seyrek güncellemeler kullanılırsa hedef ağ, ana ağdan aşırı uzaklaştırır. Bu durumda hedef ağ, mevcut çevre dinamiklerine ve güncel politika değişikliklerine uyum sağlayamaz. Ajan, eski hedef değerlere göre davranış geliştirdiğinden dolayı aldığı ödül ile beklenen Q değeri arasında ciddi farklar oluşur. Bu fark büyür ve ağın öğrenmesi güçleşir (Plaat, 2022). Sonuç olarak hem hedef ağ güncelleme sıklığı hem de yumuşatma faktörü, öğrenmenin kararlılığı açısından dikkatlice seçilmelidir. Seçilecek olan

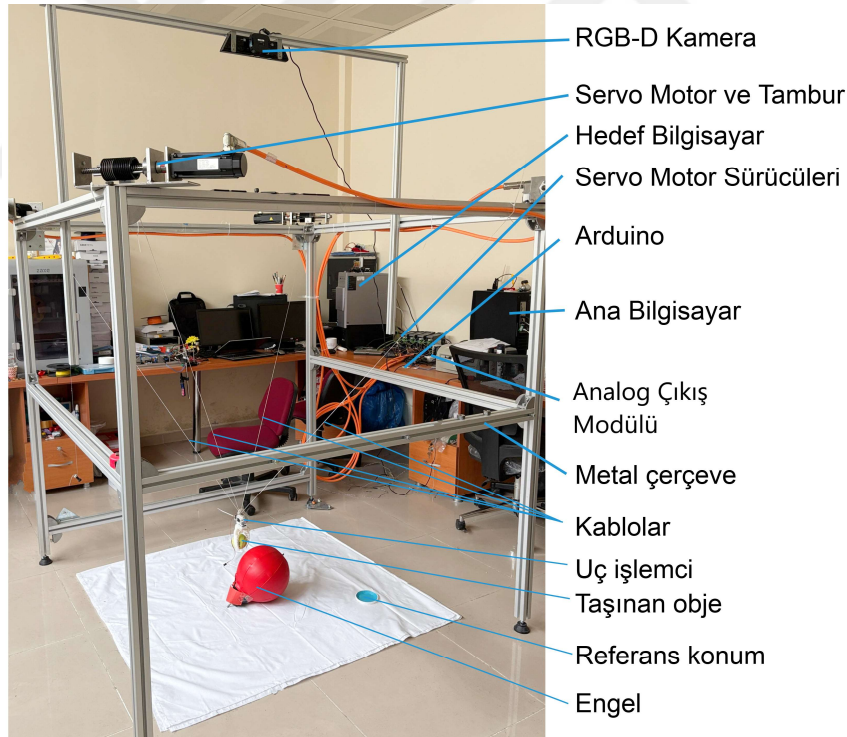
değerler, problem türüne ve gözlem/ödöl yapısına göre deneysel olarak optimize edilerek optimum öğrenme sağlanmalıdır.



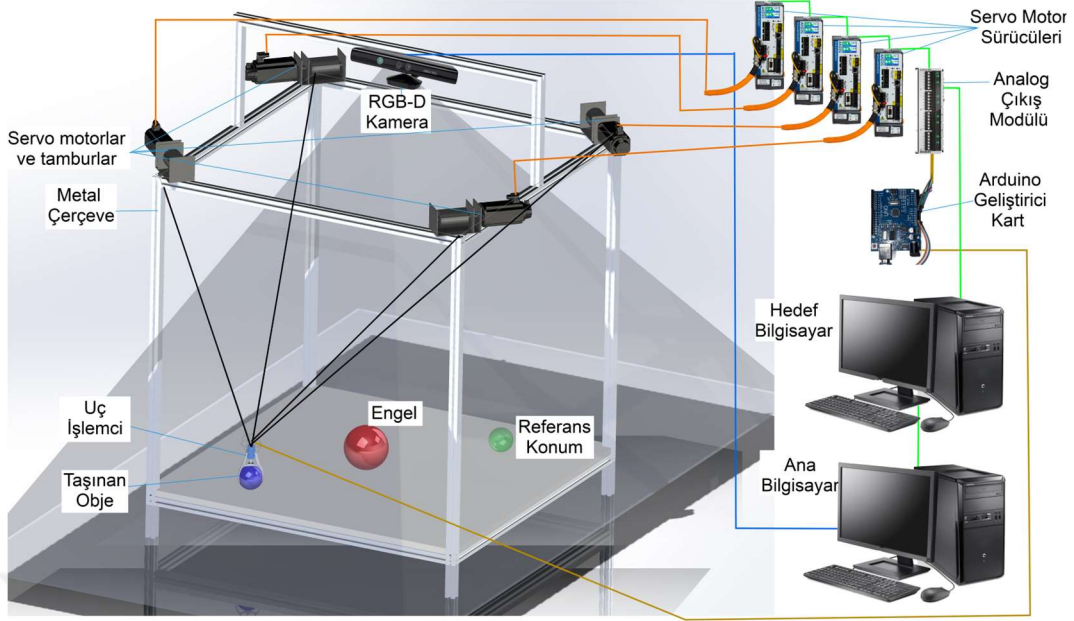
2. YAPILAN ÇALIŞMALAR

2.1. Ortamsal KSPR Tasarımı ve Deney Düzeneği Kurulumu

Bu çalışmada, dört kablo ile sürülen ve üç boyutlu bir hacim içerisinde asılı şekilde hareket eden ortamsal bir paralel robot tasarlanmıştır. Tasarlanan deney düzeneği 1.5m x 1.5m x 1.5m boyutlarında alüminyum sigma profil çerçeve kullanılarak üretilmiş olup deney düzeneğinde dört adet servo motor ve dört adet servo motor sürücüsü, bir adet ana bilgisayar, bir adet hedef bilgisayar, bir adet RGB-D kamera, bir adet Arduino geliştirici kart, bir adet analog çıkış modülü ve bir adet küçük servo motor entegre edilmiş bir uç işlemci kullanılmıştır. Kullanılan bileşenlerin deney düzeneği üzerindeki gösterimleri Şekil 13'te, şematik gösterimleri Şekil 14'te gösterilmiştir.



Şekil 13. Deney düzeneği fiziki kurulumu



Şekil 14. Deney düzeneği şematik kurulum

Deney düzeneğinde kullanılan ana bilgisayar, hedef bilgisayarda çalışacak olan kontrol sisteminin tasarımında, verilerin gözlemlenmesinde ve RGB-D kameradan alınan verilerin işlenerek hedef bilgisayara gönderilmesinde rol almaktadır. Ana bilgisayar ile hedef bilgisayar birbirlerine ethernet protokolü üzerinden bağlanmaktadır ve veri alış-verişi bu şekilde yapılmaktadır. Hedef bilgisayar ise servo motor sürücülere EtherCAT protokolü ile bağlı olup hedef bilgisayardan aldığı konum verilerini motorlara aktarmakta ve uç işlemciyi hareket ettirmektedir. Kullanılan RGB-D kamera ana bilgisayara USB üzerinden bağlı olup kamera verilerinin işlenmesi ana bilgisayar üzerinde gerçekleşmektedir. Verilerin işlenmesi sonucu elde edilen konum bilgileri ise UDP bağlantı protokolü ile ana bilgisayardan hedef bilgisayara gönderilmektedir. Uç işlemcide, parmakları açıp kapatmak için kullanılan küçük servo motor bir Arduino geliştirici kartı ile kontrol edilmektedir. Uç işlemcinin açılıp kapanabilmesi için servo motor sürücülere EtherCAT ile bağlı analog çıkış modülü bulunmaktadır. Bu modül analog sinyaller üretmekte ve üretilen sinyaller Arduino kart tarafından değerlendirilerek küçük servo motora komutlar göndermekte böylece uç işlemcinin açılıp kapanması sağlanmaktadır.

Bu çalışmada KSPR'yi hareket ettirmek üzere dört adet Kollmorgen AKM 24D servo motor ve Kollmorgen AKD P306 motor sürücüler kullanılmıştır. Kullanılan motor ve sürücüler Şekil 15'te gösterilmiştir.



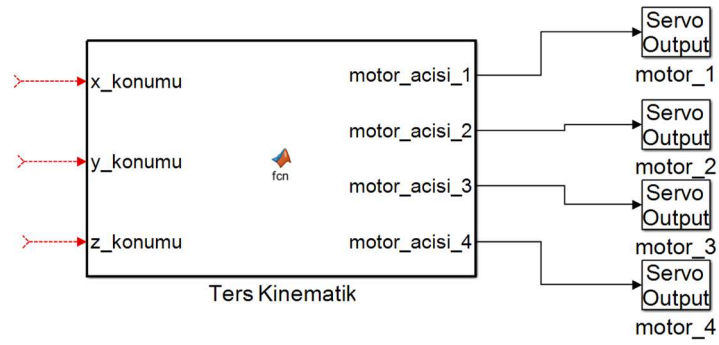
Şekil 15. Endüstriyel servo motor ve motor sürücüler

Deney düzeneğinde kullanılan Kollmorgen AKD serisi servo sürücüler ve Beckhoff analog/dijital sinyal modülleri dijital EtherCAT iletişim protokolü üzerinden hedef bilgisayara bağlanmıştır. Sürücüler, gerekli kontrol komutlarını doğrudan hedef bilgisayarda çalışan Simulink/Realtime üzerinden gerçek zamanlı olarak almaktadırlar. Sürücülere bağlı olan motorlar, EtherCAT üzerinden gönderilen hız, konum veya tork referanslarına göre çalışmakta; motorlardan elde edilen geri besleme bilgileri aynı ağ üzerinden hedef bilgisayara iletilmektedir. Bu sistemde ana bilgisayar yalnızca hedef bilgisayarda çalışan simülasyonu tasarlamak, verileri izlemek ve görüntü işlemek amacıyla kullanılmaktadır.

Motor kontrolü için referans konum değerleri, KSPR sistemine özgü ters kinematik modelin çözümü temel alınarak oluşturulmuştur. Bu modelde, uç işlemcinin istenen konumu $\vec{p} = [x_p, y_p, z_p]^T$ olarak verildiğinde ters kinematik denkleme dayanarak her bir kablonun uzunluğu $\|l_i\|$ hesaplanır. Tambur çapı D ve çevresi $C = \pi \cdot D$ olmak üzere kablo uzunluğuna karşılık gelen motor dönüş miktarı θ_i açısal olarak şu şekilde bulunur:

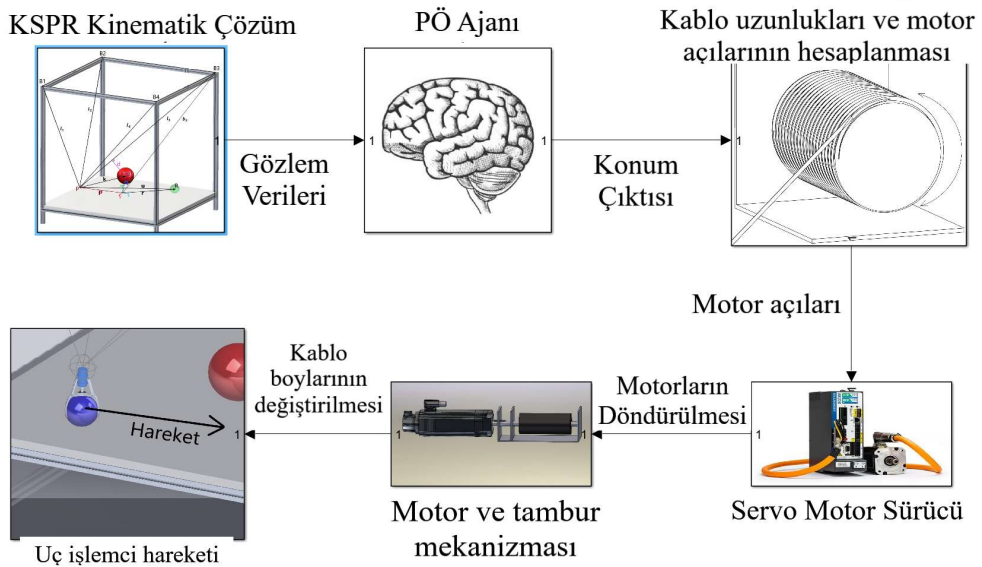
$$\theta_i = \frac{\|l_i\|}{2\pi r_{tambur}} \cdot 360^\circ = \frac{\|l_i\|}{\pi D_{tambur}} \cdot 360^\circ$$

Bu açı değeri servo motor kontrolcüsüne referans konum değeri gönderilerek her bir kablonun uygun uzunlukta sarılması sağlanır. Tüm motorlar senkronize biçimde çalıştığından uç işlemci istenen pozisyona düzgün şekilde yönlendirilir. Motor kontrolleri Şekil 16'da görüldüğü gibi Simulink/Realtime üzerinden hedef bilgisayar aracılığı ile gerçekleştirilir.



Şekil 16. Simulink/Realtime servo motor kontrolü

Sistemin otomatik karar verme yeteneği kazanabilmesi amacıyla, uç işlemci konum verilerinin PÖ ajanı tarafından üretilmelidir. Bu doğrultuda, benzetim ortamında eğitilecek ve konum tabanlı aksiyonlar üretecek şekilde yapılandırılacak olan PÖ ajanının, gerçek zamanlı kontrol uygulamaları için Simulink/Realtime modeline doğrudan entegre edilebilmesine yönelik altyapı hazırlanmıştır. Tasarlanan yapıda, simülasyon ve gerçek zamanlı sistem arasında tutarlılığı sağlamak amacıyla, aynı kinematik denklemler ve gözlem uzayı kullanılmalıdır. PÖ ajanı, çevreden aldığı gözlemler doğrultusunda uç işlemci için yeni bir hedef konum üretmekte; bu hedef konum, sistemdeki konum tabanlı motor kontrol modülüne iletilmektedir. Motor kontrol modülü ise bu hedef konumu kullanarak ters kinematik çözüm yapmakta ve servo motorlara gönderilecek açı komutlarını üretmektedir. Bu yapı Şekil 17’de gösterilmiştir.



Şekil 17. Pekiştirmeli öğrenme ajanının entegrasyonu

Böylece uç işlemci, PÖ ajanının kararına göre adım adım hareket ettirilmiş olur. Sonuç olarak benzetim ortamında eğitilerek optimal politika öğrenen PÖ ajanı, deney sistemiyle entegre edilerek fiziksel KSPR üzerinde gerçek zamanlı çalışan bir kontrol bileşenine dönüştürülmüş olur.

Bu tez çalışmasında kullanılan kamera ise Microsoft tarafından geliştirilen Kinect V1 RGB-D kamera, 640×480 piksel RGB çözünürlüğe ve 320×240 piksel derinlik çözünürlüğüne sahiptir. Kamera, ana bilgisayara USB 2.0 protokolü üzerinden bağlanmakta olup, eş zamanlı olarak renk (RGB) ve derinlik (D) verileri alınarak işlenmektedir. Kinect kamerası, 43° dikey ve 57° yatay görüş açısına sahip olup, derinlik sensörü 0.4 m ile 4.0 m arasında güvenilir mesafede çalışmaktadır. Derinlik algılama işlemi, kızılötesi (IR) ışık yayan bir infrared projektör ve bu projeksiyonu algılayan bir IR kamera aracılığıyla gerçekleştirilir. Bu yöntem, dış ortamda bulunan güneş ışığının yüksek IR bileşeni nedeniyle bozulmalara yol açabileceğinden, sistem yalnızca kapalı alan uygulamalarında kullanılmıştır.

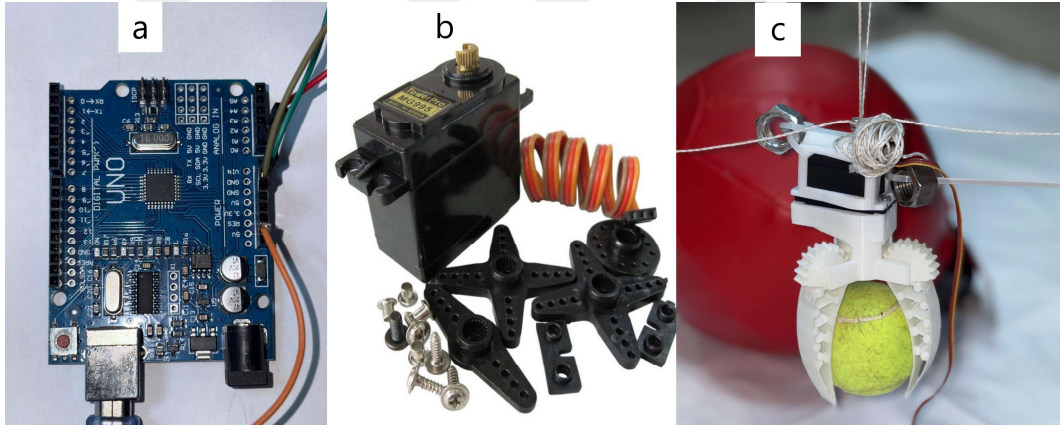
2.1.1. Uç İşlemci Bağlantısı, Tasarımı ve Kontrolü

KSPR'lerde sistemin kinematik ve dinamik karakteristikleri doğrudan kablo bağlantı noktalarının geometrik yerleşimine bağlıdır. Uç işlemci serbestlik derecelerinin kontrolü açısından, kabloların bağlantı şekilleri hem hareket doğruluğunu hem de sistemin denge durumunu belirleyen önemli bir parametredir. Özellikle altı serbestlik derecesine sahip uç işlemcilerde, kabloların sabitlendiği noktalarda oluşan momentler sistemde istenmeyen dönme hareketlerine ve kontrol dışı salınımlara sebep olabilmektedir.

Bu çalışmada kullanılan uç işlemci; x, y ve z eksenlerinde yalnızca doğrusal hareketleri gerçekleştirecek bir yapıda tasarlanmıştır. Sistemde açısal serbestlik derecelerinin aktif olarak kontrol edilmesi hedeflenmediğinden gerçekleştirilen uygulama yalnızca öteleme tabanlı bir kontrol stratejisi içermekte ve uç işlemci tarafından taşınan yükler hem düşük kütleli hem de simetrik yapıda olduğundan ve sistem yüksek hız gerektirmeyen görevler için kullanıldığından önemli bir atalet etkisi oluşturmamaktadır. Bu koşullar göz önüne alındığında Şekil 18'de görüldüğü gibi uç işlemcinin yalnızca tek bir noktadan dört kablo ile bağlanması yapısal sadeleştirme sağlayarak çalışma alanını artırmış, kabloların uç işlemci üzerinde üreteceği momenti minimize ederek kinematik denge

kadar öteleme yapılmıştır. Böylece tutucu ucun tam olarak istenen pozisyona ulaşması sağlanmıştır. Ayrıca tutucu ucun da engele temasını engellemek amacıyla tutucu uç merkez noktasından (Q) tutucu ucun tamamını kapsayacak şekilde sanal bir küre tanımlanmış olup bir güvenli alanı oluşturulmuştur. Oluşturulan kürenin yarıçapı ise c ile gösterilmiştir. Belirtilen tüm geometrik büyüklükler, gerekli denklemler ile ifade edilerek kinematik modelde gösterilmiştir.

Uç işlemci olarak küçük bir servo motorla tahrik edilen ve beyzbol topu gibi objeleri kavrayabilecek boyutta bir tutucu uç tasarlanmıştır. Tasarlanan tutucu uç, objeyi sarmalayarak kavrayabilen elastik yapıda parmaklara sahiptir. Parmakların hareketi, tutucunun merkezine entegre edilen bir servo motor aracılığıyla yapılmakta ve motor kontrolü Arduino geliştirici kartı tarafından sağlanmaktadır. Arduino kart, analog çıkış modülünden gelen sinyalleri değerlendirerek servo motoru hareket ettirmektedir. Servo motorun döndürülmesiyle parmaklar açılıp kapanmakta ve objeleri tutup bırakabilmektedir. Tutucu uç üretimi 3B yazıcı kullanılarak gerçekleştirilmiştir. Kullanılan bileşenler ve tutucu uç nihai görseli Şekil 19’da gösterilmiştir.



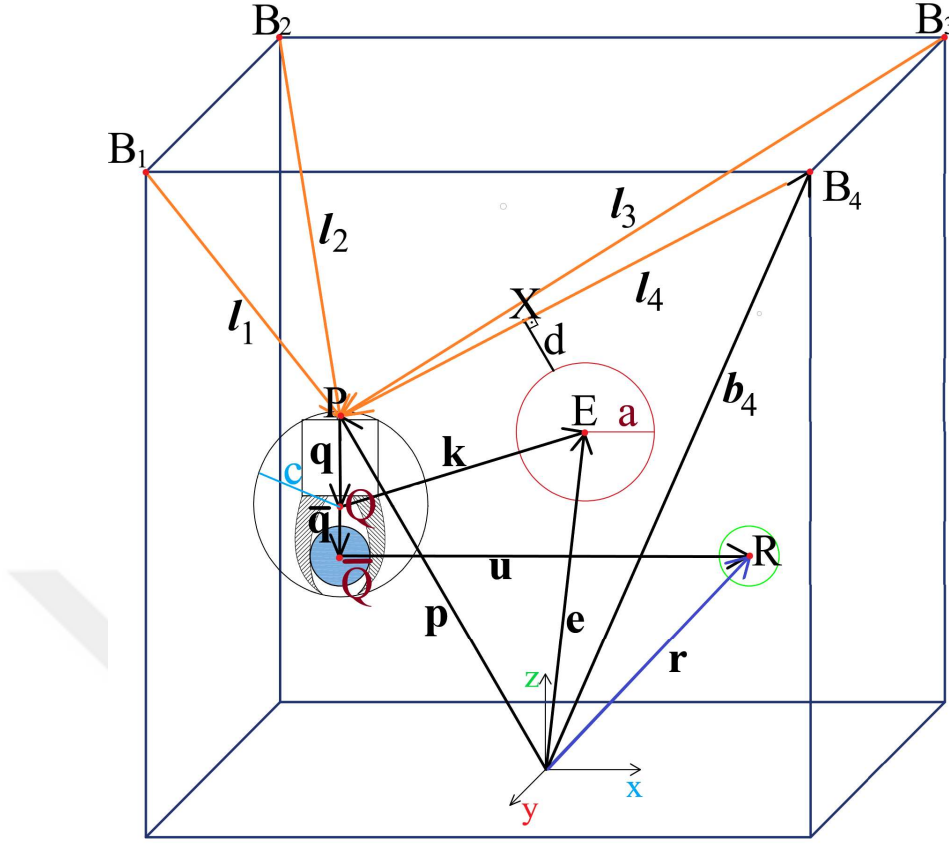
Şekil 19. a) Arduino geliştirici kart, b) mini servo motor (URL-11, 2025), c) uç işlemci

2.1.2. Ortamsal KSPR’nin Kinematik Modeli

Bu çalışmada kullanılan KSPR sisteminin benzetim ortamında, PÖ ajanı ile eğitilebilmesi için sistemin ters kinematik modelinin elde edilmesi gerekmektedir. Ters kinematik model, uç işlemcinin konumu bilindiğinde bu konuma karşılık gelen kablo uzunluklarının hesaplanmasını içerir ve açık çevrim konum kontrolü açısından kritik öneme

sahiptir. KSPR sisteminin hassas ve kararlı bir şekilde çalışabilmesi için kablo çıkış noktalarının uzaydaki konumlarının yüksek doğrulukla belirlenmesi ve yerleştirilmesi büyük önem taşımaktadır. Bu amaçla sistemin taşıyıcı yapısını oluşturan alüminyum sigma profillerden oluşan çerçevenin üst dört köşesine yerleştirilen servo motor ve tambur mekanizmasının kablo çıkış noktaları, hem düşey (z-ekseni) hem de yatay (x-y düzlemi) konumları bakımından dikkatle hizalanmıştır. Robot çerçevesinin kurulmuş olduğu yüzeyin mutlak düz olmaması sebebiyle düşey eksenindeki yüksekliklerin eşitliği, Dewalt Dw088cg lazer terazisi yardımıyla ölçülerek ± 0.5 mm doğrulukla ayarlanmış ve böylece kabloların eğiminden kaynaklanabilecek sistematik kinematik hataların önüne geçilmiştir. Aynı şekilde, kablo çıkış noktalarının x ve y düzlemi üzerindeki konumları da hassas ölçümlerle belirlenmiş ve simetrik yerleşimleri sağlanmıştır. Bu yerleşim doğruluğu sayesinde dört kablonun da uç işlemciye eşit aralıklarla bağlanması mümkün olmuş ve böylece başlangıç konumundaki denge durumu korunmuştur. Sistem çerçevesi, nominal olarak $1.5 \text{ m} \times 1.5 \text{ m} \times 1.5 \text{ m}$ boyutlarında kübik bir hacim oluşturacak şekilde inşa edilmiştir. Motor yerleşim konumları ve mekanik montaj gereksinimleri nedeniyle kablo çıkış noktaları üstten $1.45 \text{ m} \times 1.45 \text{ m}$ boyutlarında bir küpün köşelerine denk gelecek şekilde yerleştirilmiştir. Bu nedenle uç işlemcinin teorik olarak erişebileceği çalışma hacmi, kenar uzunluğu 1.45 m olan bir küp hacmi ile sınırlandırılmıştır. Bu hassas konumlandırmalar, sistemdeki oluşabilecek hataları minimize etmektedir.

Fiziksel kurulum sonrası KSPR sisteminin kinematik modeli oluşturulmuştur. Kinematik hesaplamalarda kullanılan temel geometrik büyüklükler Şekil 20’de gösterilmiştir.



Şekil 20. Dört kablo ile sürülen KSPR ve kinematik modeli

Bu çalışmada tasarlanan robotun çalışma alanı, 1,45 m kenar uzunluklarına sahip bir küpten oluşmaktadır. Çalışma alanı boyutlarına göre kablo çıkış noktaları üst köşelere konumlandırılarak mümkün olan en geniş çalışma alanı elde edilmiştir. Belirtilen çalışma alanı boyutları dikkate alındığında robotun global eksen takımı (x, y, z) çalışma alanının tabanında orta noktaya konumlandırılmıştır. Şekil 20’de kabloların sabit bağlantı noktaları B_1 , B_2 , B_3 , B_4 , kabloların uç işlemciye bağlantı noktası P, uç işlemcinin merkezi Q, uç işlemcinin objeyi kavrama merkezi $\bar{Q}$, uç işlemci için belirlenen kürenin yarıçapı c, engelin konumu E, engeli temsil eden kürenin yarıçapı a, hedef olarak tanımlanan referans noktası R ve kablolar üzerinde engele en yakın nokta X ile ifade edilmiştir.

Engelin küresel olarak modellenmesi hem geometrik büyüklüklerin hesaplanmasını kolaylaştırmakta hem de çarpışma olmayan güvenli bir alan oluşturmaktadır. Ayrıca bu engel yaklaşımı sayesinde uç işlemci ya da kabloların engelin en yakın noktasına olan dik uzaklığı kolaylıkla hesaplanabilmektedir. Bu noktalar arasında tanımlanan vektörler, sistemin geometrik ve fiziksel özelliklerinin hesaplanmasında kullanılmıştır.

Sabit bağlantı noktalarının her birinden uç işlemcinin bağlantı noktasına uzanan vektörler, her bir kablonun yönünü ve anlık uzunluğunu temsil etmekte olup şekilde turuncu renkle gösterilmiş ve $\mathbf{l}_1, \mathbf{l}_1, \mathbf{l}_1, \mathbf{l}_1$ olarak adlandırılmışlardır. Benzer şekilde engel konumu ile uç işlemci arasındaki vektör ($\mathbf{k}$), çarpışma kontrolü açısından kritik bir parametredir. Engel ile çarpışma olabilecek en yakın kablo arasındaki dik mesafe ise d ile temsil edilmiştir. Uç işlemci ile hedef noktası arasındaki vektör ($\mathbf{u}$) ise ajan için hedef konuma olan uzaklığın bir ölçüsü olarak kullanılmıştır. Vektörlerden elde edilen büyüklükler; mesafeler, yönler, açı ilişkileri ve doğrusal hizalanma gibi çeşitli gözlem verilerine dönüştürülerek PÖ ajanına gözlem verisi olarak sağlanmıştır. Bu durumda kablo uzunlukları ve PÖ ajanının eğitiminde kullanılacak olan büyüklükler Şekil 20 dikkate alınarak aşağıdaki gibi hesaplanır:

-İlk olarak ters kinematik modelin elde edilmesi için kabloların global eksen takımına göre bağlı oldukları noktalar aşağıdaki şekilde tanımlanmıştır:

$$\begin{aligned} B_1 &= (x_{B_1}, y_{B_1}, z_{B_1}), & B_2 &= (x_{B_2}, y_{B_2}, z_{B_2}), & B_3 &= (x_{B_3}, y_{B_3}, z_{B_3}), \\ B_4 &= (x_{B_4}, y_{B_4}, z_{B_4}) \end{aligned} \quad (33)$$

Burada KSPR için uç işlemcinin kablolarla bağlantı konumunun ($\vec{p} = [x_p, y_p, z_p]^T$) bilindiği durumdaki kablo uzunlukları aşağıda verilen kinematik denklem kullanılarak elde edilir.

$$\|\mathbf{l}_i\| = \|\mathbf{p} - B_i\| = \sqrt{((p_x - B_{ix})^2 + (p_y - B_{iy})^2 + (p_z - B_{iz})^2)} \quad i = 1, 2, 3, 4 \quad (34)$$

Kablo üzerindeki engele en yakın nokta ise şu şekilde hesap edilir:

$$X = \left(\vec{B}_i + \frac{(\vec{e} - \vec{b}_i) \cdot (\vec{p} - \vec{b}_i)}{|\vec{p} - \vec{b}_i|^2} \cdot (\vec{p} - \vec{b}_i) \right) \quad i = 1, 2, 3, 4 \quad (35)$$

Küresel engelin kablolarla olan dik uzaklığı ise Denklem (36) ile hesaplanır.

$$d_i = \left(\left\| \vec{e} - \left(\vec{B}_i + \frac{(\vec{e} - \vec{b}_i) \cdot (\vec{p} - \vec{b}_i)}{|\vec{p} - \vec{b}_i|^2} \cdot (\vec{p} - \vec{b}_i) \right) \right\| - a \right) \quad i = 1,2,3,4 \quad (36)$$

Uç işlemci için belirlenen sanal küre yüzeyinden engel yüzeyine olan dik uzaklık Denklem (37) ile hesaplanır:

$$(\|\vec{k}\| - (a + c)) = (\|\vec{e} - (\vec{p} + \vec{q})\| - (a + c)) \quad (37)$$

Uç işlemcinin objeyi tutma noktasından hedefe doğru dik uzaklık aşağıdaki denklem ile hesaplanır:

$$\|\vec{u}\| = \|\vec{r} - (\vec{p} + \vec{q} + \vec{q})\| \quad (38)$$

2.1.3. Kamera Kalibrasyonu

RGB-D kamera, KSPR sisteminin üst çerçevesine rijit bir taşıyıcı yapı ile sabitlendikten sonra, hem yatay düzleme paralel olacak şekilde hizalanmış hem de görüş merkezinin robotun global koordinat sisteminin orijini ile çakışacak biçimde konumlandırılmıştır. Bu işlem sırasında kameradan elde edilen nokta bulutu verileri kullanarak z-eksenindeki düzlemsellik kontrol edilmiş ve gerekli mekanik ayarlamalar fiziksel olarak yapılmıştır. Kamera, MATLAB yazılımında Image Acquisition Toolbox kullanılarak çalıştırılmış, *pcfromkinect* fonksiyonu aracılığıyla RGB ve derinlik verilerinden otomatik olarak üç boyutlu nokta bulutu oluşturulmuştur. MATLAB tarafından sağlanan nokta bulutu verilerinin konum doğruluğu, deneysel ölçümlerle test edilmiştir. Kalibrasyon öncesi yapılan ölçümlerde uzunluğu bilinen bir obje çalışma alanına yerleştirilerek MATLAB üzerinden ölçülen uzunluk değeri birbirinden farklı ölçülmüştür. Bunun sonucu yapılan ölçümde 1 metrelik bir çubuk yaklaşık 92cm olarak ölçülmüştür. Yarım metrelik bir çubuksa yaklaşık 46cm olarak ölçülmüştür. Sonrasında çalışma alanı içerisinde sistematik olarak yapılan ölçümlerde ölçüm noktaları belirlenmiş ve bu noktalara göre ölçümler yapılmıştır. Yapılan ölçümlerin hata oranları hesaplanarak ortalama bir hata değeri

bulunmuştur. Ölçülen mesafelerin beklenen değerden anlamlı şekilde sapma gösterdiği gözlemlenmiştir. Bu hata, yalnızca bir noktaya özgü olmayıp sistematik olarak tüm uzaya yayıldığı için merkeze yakın bölgelerde daha az, uzaklaştıkça daha fazla olmak üzere mesafeye bağlı oransal bir hata olduğu belirlenmiştir. Yapılan çoklu ölçümler sonucunda bu sapmanın yaklaşık olarak sabit bir yüzde ile arttığı saptanmış ve elde edilen konum verilerine aşağıdaki gibi bir düzeltme faktörü uygulanmıştır.

$$P_{\text{düzeltmiş}} = P_{\text{ölçülen}} * (1 + \delta)$$

Burada $P_{\text{ölçülen}}$, MATLAB tarafından verilen 3B konum vektörünü, δ ise tespit edilen yüzde hata oranını temsil etmektedir. Bulunan hata değeri yaklaşık %8 olarak ölçülmüş ve her noktadaki hatanın %8'e çok yakın olduğu görülmüştür. Bulunan hata değeri kamera konum verilerine öteleme olarak eklenmiştir. Ekleme sonrası yapılan ölçümlerde hatanın +/- %0.5'e kadar gerilediği görülmüştür. Kalan mevcut hatanın nedeni ise kullanılan kameranın nokta bulutu verilerindeki gürültü olduğu gözlemlenmiştir. Bu düzeltmenin ardından sistem genelinde elde edilen konum verileri, gerçek mesafelerle tutarlı hâle gelmiş ve kamera deneysel düzeneğe entegre edilmeye hazır hale getirilmiştir.

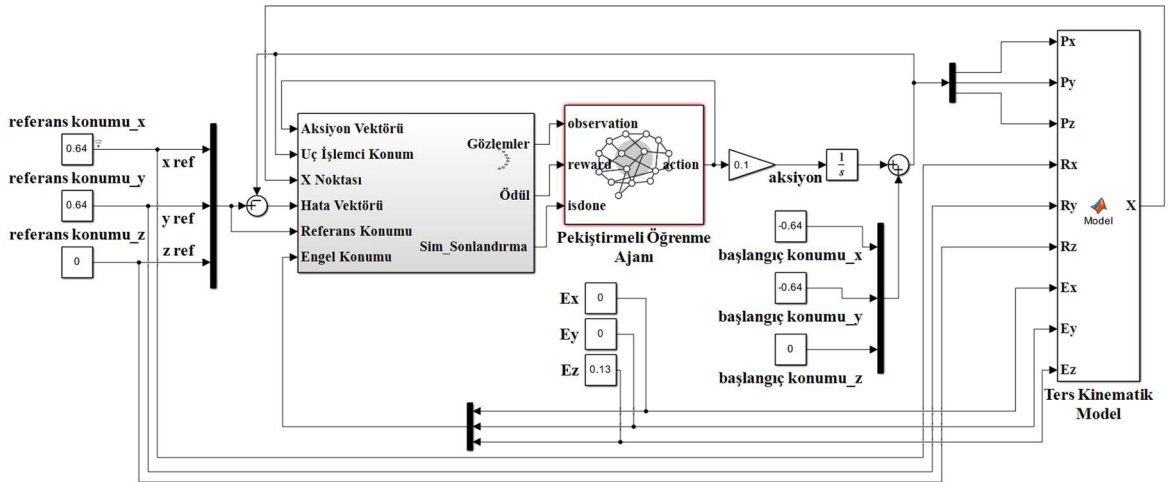
2.2. Pekiştirmeli Öğrenme Ortamının Oluşturulması

Bu çalışmada kullanılan KSPR'nin, engele çarpmadan hedefe ulaşabilmesi için PÖ ajanının bu doğrultuda eğitilmesi gerekmektedir. Eğitimin doğrudan gerçek sistem üzerinde gerçekleştirilmek istendiğinde bazı önemli zorluklar ortaya çıkmaktadır. Eğitim sürecinde sistemin doğrudan kullanılması, özellikle yalnızca konum temelli bir kontrol yapısı söz konusu olduğunda hem verimlilik hem de güvenlik açısından çeşitli sorunlara yol açabilmektedir. Eğitim sürecinin başlarında, ajan henüz yeterli politika öğrenimini tamamlamadığından, uç işlemciyi çalışma alanı dışına yönlendirebilir. Bu durumda bazı kablolar gevşeyebilir, birbirine dolanabilir ve kontrolsüz hareketler sonucunda istenmeyen gerilimler oluşturarak mekanik bileşenlere zarar verebilir. Ayrıca fiziksel sistemin çalıştırılması sırasında oluşan elektrik sarfiyatı da dikkate alındığında sadece konum üreten bir PÖ ajanının deney sistemi üzerinde eğitilmesi birçok açıdan rasyonel değildir. Eğer kontrol süreci yalnızca kinematik değil, aynı zamanda dinamik etkileşimleri de içeren bir yapıda olsaydı; örneğin, ani yön değişimlerine bağlı olarak kabloların gevşemesi gibi

durumların hesaba katılması gerekseydi, bu durumda simülasyon ortamında eğitilen ajanın deney sistemi üzerinde ek eğitimlerle uyarlanması anlamlı bir yaklaşım olabilirdi. Böylece model belirsizlikleri ortadan kaldırılabilir ve ajan, gerçek robot üzerinde daha etkili bir performans sergileyebilirdi. Ancak bu çalışmada ters kinematik denklemler kullanılarak açık çevrim konum kontrolü yapıldığından simülasyon ortamında gerçekleştirilen eğitim hem yeterli hem de tercih edilebilir bir yaklaşımdır. Ayrıca gerçek ortamında her etkileşim için gereken gözlem verilerinin toplanması oldukça işlem gücü gerektiren, zaman alıcı, gürültülü ve hata barındırmaya açık bir süreçtir. Buna ek olarak, robotun eğitimin başlangıç konumlarında hazır olarak bulunması durumunda gerçek dünyada bir eğitim bölümünün tamamlanması yaklaşık 40 saniye gibi bir süre gerektirebilirken her bölüm sonu yeni başlangıç koşullarına giderek hazırda beklemesi çok daha fazla zaman alacaktır. Fakat aynı eğitim bölümü benzetim ortamında sadece 4-5 saniye içinde tamamlanabilmektedir. Bu nedenle zaman ve kaynak verimliliği açısından benzetim ortamı tercih edilmiştir.

Bu çalışmada, benzetim ortamı olarak MATLAB/Simulink yazılımı tercih edilmiş ve ajan eğitimi, MATLAB'ın Reinforcement Learning Toolbox'ında yer alan DQA algoritması kullanılarak gerçekleştirilmiştir. Bu seçim, hem Simulink'in gerçek zamanlı simülasyon kabiliyetlerinden yararlanmayı hem de MATLAB'ın sağladığı hazır DQA altyapısı ile algoritmanın kolayca özelleştirilmesini mümkün kılmıştır.

Şekil 21'de görüldüğü üzere KSPR sisteminin kinematik modelinden elde edilen büyüklükler, gözlem vektörü olarak DQA ajanına beslenmiş ve bu veriler hem gözlem hem de ödül fonksiyonlarının hesaplanmasında kullanılmıştır.



Şekil 21. Matlab/Simulink PÖ ortam kurulumu

Tablo 1’de Matlab DQA algoritmasının sözde kodu verilmiştir.

Tablo 1. Matlab DQA algoritması sözde kodu

Ana Q-ağı $Q(s,a | \theta)$ ve hedef Q-ağı $Q'(s,a | \theta^-)$, rastgele başlangıç parametreleri θ ve θ^- ile oluşturulur.

Tekrarlama belleği kapasitesi N olacak şekilde başlatılır.

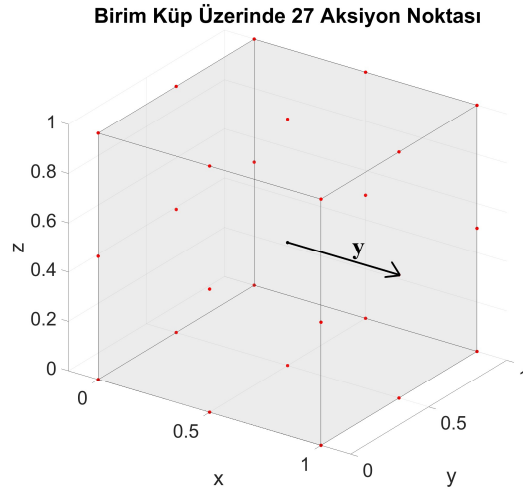
Keşif parametresi ϵ , başlangıç değeri ϵ_0 olacak şekilde ayarlanır.

Her bölüm için:

- Durum gözlemi s_0 alınır.
- Her zaman adımında:
 - Olasılık ϵ ile rastgele bir eylem seçilir, aksi halde: $a_t = \operatorname{argmax}_a Q(s_t, a | \theta)$ ile en yüksek Q değerine sahip aksiyon seçilir.
 - Eylem a_t uygulanır, ödül r_t ve sonraki durum s_{t+1} gözlemlenir.
 - Deneyim (s_t, a_t, r_t, s_{t+1}) arabelleğe eklenir.
 - Eğer arabellekte en az M adet deneyim varsa:
 - n adet örnek (s_i, a_i, r_i, s'_i) rastgele seçilir.
 - Her örnek için hedef değer aşağıdaki gibi hesaplanır:
 $y_i = \{ r_i, \text{eğer terminal durum ise}; r_i + \gamma \max_{a'} Q'(s'_i, a' | \theta^-), \text{aksi halde} \}$
 - Ortalama karesel hata (MSE) kaybı hesaplanır:
 $L(\theta) = (1/n) \sum_{i=1}^n (y_i - Q(s_i, a_i | \theta))^2$
 - Ana ağ parametreleri, $L(\theta)$ 'ı en aza indirilecek şekilde güncellenir.
 - Hedef ağ güncellemesi:
 - Periyodik güncelleme: $\theta^- \leftarrow \theta$; her C adımda hedef ağ ana ağ parametreleriyle bire bir güncellenir.
 - Yumuşak güncelleme: $\theta^- \leftarrow \tau \cdot \theta + (1 - \tau) \cdot \theta^-$; her C adımda hedef ağ ana ağ parametreleriyle belirli bir oranda güncellenir.
 - Keşif oranı azaltılır: $\epsilon \leftarrow \max(\epsilon_{\min}, \epsilon - \Delta\epsilon)$
- Bölüm sonlandırma koşulu kontrol edilir.

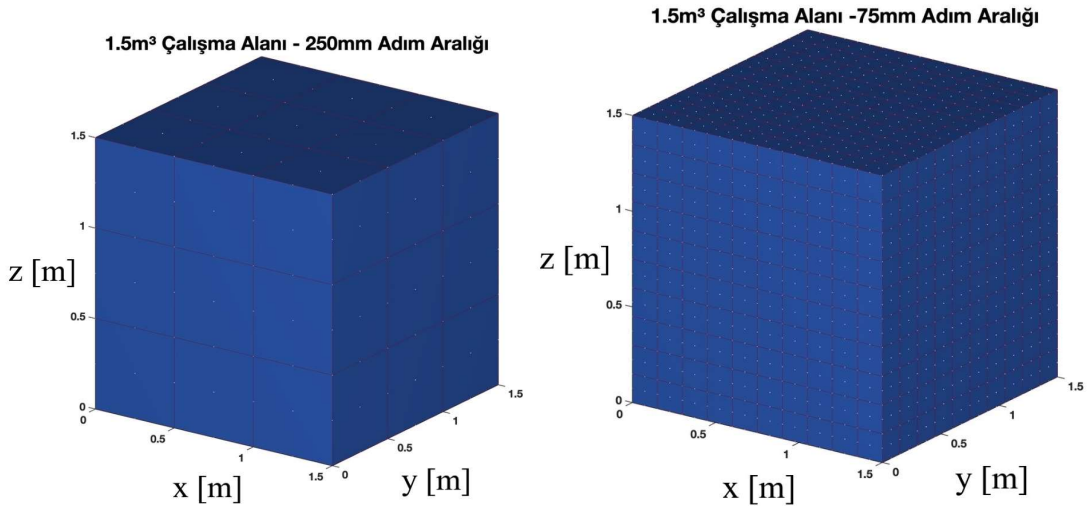
Eğitim sonlandırma koşulu kontrol edilir.

Bu çalışmada kullanılan PÖ algoritması, ayrık aksiyon uzayında çalışan DQA yöntemidir. Aksiyon uzayı, üç boyutlu bir küpün merkezinden köşe, kenar ve yüzey orta noktalarına uzanan yönleri temsil edecek şekilde belirlenmiştir. Şekil 22’de görüldüğü gibi ajan, her bir konumda 26 farklı yön doğrultusunda hareket edebilmekte veya mevcut konumunda kalabilmektedir. Toplamda 27 adet ayrık aksiyon tanımlanmış olup bunlar sırasıyla 26 hareket yönü ve 1 hareketsiz aksiyon içermektedir. Alınan aksiyon vektörel olarak y vektörü ile ifade edilmiştir.



Şekil 22. Birim küp üzerindeki aksiyon yönleri ve aksiyon vektörü

Bu çalışmada, ajan tarafından üretilen aksiyon vektörlerine uygulanan kazanç (gain) parametresi hem birim adım boyunu hem de çalışma uzayının bölünme hassasiyetini belirlemektedir. Şekil 23'te görüldüğü gibi aksiyon adım aralığı ne kadar küçük seçilirse çalışma alanı o kadar fazla sayıda küçük parçaya ayrılmış olur. Dolayısıyla çalışma ortamının çözünürlüğü doğrudan aksiyon kazancı ile ilişkilidir.



Şekil 23. Aksiyon adım aralığına bağlı çalışma uzayı çözünürlüğü

PÖ ajanının aksiyon çıktıları, her yönde 1 cm'lik adım mesafeleri ile tanımlanmıştır. Bu tanım doğrultusunda çalışma ortamı, her bir kenarı 2 cm olan kübik hücrelere ayrılmış olur. Böylelikle ajan, her eğitim bölümünde daha az sayıda aksiyon olarak hedefe

ulaşılabilmektedir. Bu durum eğitim süresini ve kullanılan kaynak miktarını önemli ölçüde azaltmaktadır. Tanımlanan aksiyonlar, sistem tarafından doğrudan konum değişimi olarak uygulanmakta ve çalışma ortamı içerisinde her bir noktaya erişim bu şekilde sağlanmaktadır. Eğitim süresi her bir bölüm için 25 saniyelik simülasyon zamanı olarak belirlenmiş, zaman adımı ise 0,1 saniye olarak tanımlanmıştır. Bu yapı altında her bir bölüm en fazla 250 zaman adımı içermekte ve ajan bu adım sayısında ortalama eğitim sürecini tamamlamaktadır.

2.3. Engelsiz Benzetim Ortamında Yol Planlama

Kinematik modelin oluşturulması ve PÖ ortamının kurulumu sonrası ilk yol planlama çalışması, engelsiz bir üç boyutlu benzetim ortamında gerçekleştirilmiştir. Bu ortamda, yalnızca uç işlemcinin konumu dikkate alınarak sistemin başlangıç noktasından hedef konuma ulaşması hedeflenmiştir. Kabloların ya da uç işlemcinin herhangi bir engelle temas etmesi söz konusu olmadığından bu aşamada çarpışma kontrolü mevcut değildir. Bu deneysel kurulumda PÖ ajanının görevi KSPR'nin çalışma alanı sınırları içinde kalarak hedefe ulaşacak bir yol oluşturmaktır. Bu nedenle amaç en kısa mesafeyi kat eden yolun öğrenilmesidir. Bu aşama, hem eğitim sürecinde temel politika öğreniminin sağlanması hem de sonraki aşamalarda engel içeren ortamlara geçmeden önce ajan davranışlarının kontrolü açısından önemli bir ön hazırlık aşaması olarak değerlendirilmiştir. PÖ eğitimlerinde 3 katmandan oluşan 256 nöronlu bir sinir ağı kullanılmış olup aktivasyon fonksiyonu olarak *LeakyRelu* fonksiyonu kullanılmıştır.

2.3.1. Kullanılan Gözlemler, Ödül Fonksiyonu ve Sınırlamalar

Bu çalışmada, her eğitim bölümü başlangıcında uç işlemci başlangıç ve hedef konumları çalışma ortamı içerisinde rastgele olarak belirlenmiştir. Herhangi bir engelin bulunmadığı bu çalışma senaryosunda, gözlem verisi olarak yalnızca uç işlemcinin anlık konum vektörü ($\mathbf{p}+\mathbf{q}+\bar{\mathbf{q}}$) ile hedef konumu arasındaki fark, yani $\mathbf{u}$ vektörü kullanılmıştır. Kullanılan ödül fonksiyonu ise üç ayrı bileşenden oluşmaktadır ve Denklem (39) ile ifade edilmiştir.

$$r(t) = r_1(t) + r_2(t) + r_3(t) \quad (39)$$

Uç işlemcinin hedef konuma yaklaşıp yaklaşmadığına göre belirlenen ödül/ceza bileşeni $r_1(t)$ Denklem (40) ile ifade edilmiş ve hesaplanırken hata normunun türevi alınarak değerlendirme yapılmıştır. Eğer ajan hedefe yaklaşıyorsa yani hata normu türevi negatifse ödül, uzaklaşıyorsa ceza uygulanmıştır.

$$r_1(t) = \begin{cases} a, \left(\frac{d}{dt}|u_i| < 0\right) \\ b, \left(\frac{d}{dt}|u_i| > 0\right) \\ c, \left(\frac{d}{dt}|u_i| = 0\right) \end{cases} \quad (40)$$

Ajanın hedef konuma ulaştıktan sonra bu konumda sabit kalmasını teşvik eden ek ödül bileşeni $r_2(t)$ olarak Denklem (41) ile ifade edilmiştir. Ajan hedefe ulaştığında sabit kaldığı her adım için d kadar ödül verilmiştir.

$$r_2(t) = \{d, (|\mathbf{u}| = 0)\} \quad (41)$$

Hata normu belirli bir eşik değerin altına düştüğünde bu bileşen devreye girerek ajanın hedef konumunda sabit kalmasını sağlar. Uç işlemcinin çalışma ortamı sınırları dışına çıkması durumunda uygulanan yüksek ceza ise $r_3(t)$ ile ifade edilmiştir.

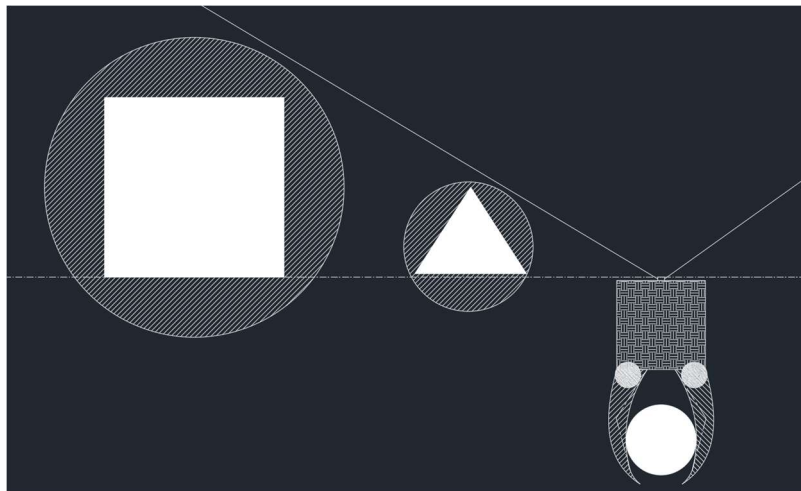
$$r_3(t) = \begin{cases} e, & (|p_x| > 0.725) \\ e, & (|p_y| > 0.725) \\ e, & (0 < |p_z| < 1.5) \end{cases} \quad (42)$$

Bu sayede ajan, fiziksel sınırları ihlal etmemeyi öğrenir. Bu yapının amacı; ajanı hedefe doğru yönlendirmek, hedefe ulaştığında konumda kalmasını sağlamak ve sistem sınırlarını öğrenmesini teşvik etmektir. Kullanılan katsayılar; $a=0.125$, $b=-0.5$, $c=-0.1$, $d=5$, $e=-2$ olarak belirlenmiştir.

2.4. Küresel Engel Yaklaşımı

Bu tez çalışmasında kullanılan KSPR sisteminde hem uç işlemci hem de robotu yönlendiren kabloların engelle çarpışması kritik olduğundan temasların önüne geçebilmek için çalışma ortamındaki objelerin geometrik yapılarının ve boyutlarının doğru şekilde modellenmesi gerekir. Ancak fiziksel ortamdaki objelerin şekilleri genellikle düzensiz ve değişkendir. Bu nedenle hem obje tespiti hem de uç işlemci ile engel arasındaki minimum mesafenin hassas bir şekilde hesaplanması zorlaşır. Özellikle farklı geometriye sahip objelerin boyut ve uzaklık hesaplamaları için farklı matematiksel denklemler gerekir ve bu durum sistemin karmaşıklığını artırır. Bu durumda küresel engel yaklaşımı kullanımı önemli avantajlar sunar. Engellerin kürelerle modellenmesi, mesafe hesaplarını önemli ölçüde basitleştirir. Çünkü bir noktanın bir küre yüzeyine olan dik uzaklığı Denklem (36) ve Denklem (37) ile tanımlanabilir. Böylece hesaplamalar sadeleşerek sistemin gerçek zamanlı performansı artar.

Kullanılan KSPR sisteminde tüm kablolar, uç işlemciyle tek bir noktada birleştiğinden sistem geometrisi itibariyle kabloların doğal uzanım yönü, engelin çevresinde bir tampon bölge oluşturur. Örneğin Şekil 24'te görüldüğü gibi engel, silindirik bir cisme sahip olsa bile bu cismin içine alan küre modeli, kabloların ulaşamayacağı bir hacmi temsil eder. Bu nedenle küresel modelleme ile oluşturulan güvenli alan, pratikte zaten kullanılmayacak bir alan olduğu için sistemin verimliliğini önemli ölçüde etkilemez.



Şekil 24. Küresel engel yaklaşımı ve güvenli alan

Ayrıca benzetim ortamında değişken engel boyutları ve konumu ile yapılan çalışmalarda uç işlemci başlangıç konumu, referans konumu, obje konumu ve obje çapı rastgele tanımlanmış olup bu yaklaşım ajanın genelleme yapabilmesi için önemlidir.

2.4.1. Sabit Çaplı Küresel Engel İçeren Benzetim Ortamında Yol Planlama

Bu tez çalışmasının asıl amacı, engel bulunan çalışma ortamında, PÖ yöntemiyle engele temas olmadan yol planlamadır. Bu nedenle çalışma ortamı içerisine sabit çaplı noktasal bir engel yerleştirilmiştir. Bu durumda ajan hem uç işlemciyi hem de kabloları engele temas ettirmeden yol planlamayı öğrenmelidir. Paralel robotların yapısı gereği uç işlemci hareket ederken tüm uzuvlar da hareket ettiğinden bu problem noktadan noktaya engelsiz yol planlamadan çok daha zor bir problemdir. Bu bölümde gerçekleştirilen yol planlama süreci, tezin temel hedeflerinden biri olan engel içeren ortamlarda güvenli ve etkin hareket stratejisi öğrenimi açısından kritik öneme sahip olduğundan gözlem uzayı ve ödül fonksiyonu bu sınırlamaları dikkate alacak şekilde yapılandırılmıştır. PÖ minimum epsilon değeri 0.1, epsilon azalma oranı $1e-5$ ve minibatch boyutu 512 olarak sabit tutulmuştur. Sinir ağı mimarisi ise üç katmanlı ve her katmanda 200 nöron olacak şekilde yapılandırılmış, aktivasyon fonksiyonu olarak LeakyReLU tercih edilmiştir. Sabit tutulan bu parametrelerin seçimi, ön denemeler sonucunda eğitim sürecinde istikrarlı bir öğrenme eğrisi sağladıkları için yapılmıştır. Diğer hiperparametreler ise, literatürde ve ön denemelerde modelin öğrenme başarısı üzerinde en yüksek etkiye sahip oldukları gözlemlendiğinden optimizasyon sürecine dahil edilmiştir. Özellikle öğrenme oranı, azaltma faktörü, hedef ağ güncelleme sıklığı gibi hiperparametreler üzerinde yoğunlaşmıştır.

Gerçekleştirilen yol planlama çalışmasında kullanılan gözlem değişkenleri, sistematik denemeler ve performans analizleri sonucunda belirlenmiştir. Gözlem sayısı ne kadar az olursa, öğrenme süreci o kadar az kaynak tüketir; ancak gözlem eksikliği, ajanın öğrenememesine veya yanlış politika geliştirmesine sebep olabilir. Bu nedenle gözlem uzayı, öğrenme başarısını ve işlem verimliliğini dengeleyecek şekilde yapılandırılmıştır.

Kinematik modelleme bölümünde hesaplanan büyüklükler, bu aşamada doğrudan gözlem vektörü olarak kullanılmıştır. Tanımlanan gözlem uzayı Denklem (43)'teki değişkenlerden oluşmaktadır.

$$[\mathbf{u}, \mathbf{X} - \mathbf{e}] \quad (43)$$

Engel konum vektörü $\mathbf{e}$ ile kablo üzerindeki objeye en yakın nokta konumu $\mathbf{X}$ arasındaki hata, uç işlemci ve hedef arası hata vektörü $\mathbf{u}$, gözlem verisi olarak tanımlanmıştır. Bu veriler her zaman adımında ajan tarafından gözlemlenerek aksiyon seçiminde kullanılmıştır. Ödül fonksiyonu Denklem (44)'teki gibi dört temel bileşene ayrılmıştır.

$$r(t) = r_1(t) + r_2(t) + r_3(t) + r_4(t) \quad (44)$$

Ödülün ilk kısmında Denklem (45)'de görüldüğü gibi $\mathbf{X} - \mathbf{e}$ vektör normu azaldıkça üstel olarak artan bir ceza olarak tanımlanmıştır. Böylece PÖ ajanını kabloların engelden uzak durması yönünde teşvik etmektedir.

$$r_1(t) = -1 * e^{-\|\mathbf{x}_L - \mathbf{q}\|} \quad (45)$$

Benzer biçimde Denklem (46)'de $\mathbf{u}$ vektörünün normu azaldıkça üstel olarak artan bir ödül tanımlanmıştır. Böylece PÖ ajanının uç işlemciyle referans arasındaki dik uzaklığı azaltacak yönde aksiyonlar seçmesi teşvik edilmiştir.

$$r_2(t) = -1 * e^{-\|\mathbf{u}\|} \quad (46)$$

Denklem (47)'de $r_3(t)$, $\mathbf{u}$ vektörü ile seçilen aksiyon vektörü $\mathbf{y}$ arasındaki açının kosinüsü hesaplanarak ödül fonksiyonuna eklenmiştir.

$$r_3(t) = \frac{\vec{u} \cdot \vec{y}}{|\vec{u}| \cdot |\vec{y}|} \quad (47)$$

Denklem (48)'de $r_4(t)$, uç işlemci hedefe ulaştığı durumda +100 , kablolar ya da uç işlemci engele çarptığında ya da çalışma alanı dışına çıktığında -10 olarak belirlenmiştir ve buradaki tüm durumlarda eğitim sonlandırılmıştır.

$$r_4(t) = \begin{cases} -10 & , & (|p_x| > 0.725) \\ -10 & , & (|p_y| > 0.725) \\ -10 & , & (0 < |p_z| < 1.5) \\ -10 & , & (\|k\| - (c + a)) = 0 \\ -10 & , & (d = 0) \\ 100 & , & (\|u\| = 0) \end{cases} \quad (48)$$

Çarpma kriterinin belirlenmesinde, engel çapı a , sabit olarak 20 cm alınmış ve çarpışma kontrolü bu değere göre gerçekleştirilmiştir. Bu yapı sayesinde ajan hedefe yönelmeyi, engelden kaçınmayı ve sistem sınırları içinde kalmayı aynı anda öğrenmektedir.

2.5. Pekiştirmeli Öğrenme Ajanı ile Deneysel Konumlama

2.5.1. Değişken Çaplı Küresel Engel İçeren Benzetim Ortamında Yol Planlama

Gerçek koşullarda çalışma ortamında bulunan engel boyutlarının değişken olması ve robotun gerçek koşullarda çalışabilmesi için engel çapının her eğitim bölümünde rastgele olarak değiştiği bir simülasyon ortamı tasarlanmış ve PÖ ajanı bu yeni ortamda tekrar eğitilerek gerçek deney düzeneğinde kullanılabilir hale getirilmiştir.. Eğitimlerde minimum epsilon değeri 0.05, epsilon azalma oranı 10^{-4} olarak sabit tutulmuştur. Sinir ağı mimarisi ise üç katmanlı olup ilk iki katman 256 nöron, üçüncü katman ise 128 nöron olacak şekilde yapılandırılmıştır; aktivasyon fonksiyonu olarak LeakyReLU tercih edilmiştir.Yapılan eğitimlerde gözlem olarak Denklem (49)'daki veriler kullanılmıştır.

$$[u, e - X, k, a] \quad (49)$$

Burada gözlem verisi olarak Şekil 20'de kinematik modelde açıklandığı üzere $\bar{Q}$ konumu ile referans arasındaki hata vektörü u , engel konum vektörü e ile kablo üzerindeki en yakın nokta konumu X arasındaki hata $e - X$, obje konum vektörü e ile uç işlemci merkez konumu Q arasındaki hata k ve obje yarıçapı a seçilmiştir. Yalnızca hata vektörleri ve

yarıçap olmak üzere dört gözlem verisi kullanılmıştır. Ödül fonksiyonu ise engelsiz ortamda yol planlamaya benzer olarak Denklem (50)'deki gibi tanımlanmıştır.

$$r(t) = r_1(t) + r_2(t) + r_3(t) + r_4(t) \quad (50)$$

Uç işlemcinin hedef konuma yaklaşıp yaklaşmadığına göre belirlenen ödül/ceza bileşeni $r_1(t)$ Denklem (51) ile ifade edilmiştir. Hata normunun türevi alınarak değerlendirme yapılmıştır. Eğer ajan hedefe yaklaşıyorsa yani hata normu türevi negatifse ödül, sabit duruyorsa ve uzaklaşıyorsa ceza uygulanır.

$$r_1(t) = \begin{cases} a, \left(\frac{d}{dt}|u_i| < 0\right) \\ b, \left(\frac{d}{dt}|u_i| > 0\right) \\ c, \left(\frac{d}{dt}|u_i| = 0\right) \end{cases} \quad (51)$$

Ajanın hedef konuma ulaştıktan sonra bu konumda sabit kalmasını teşvik eden ek ödül bileşeni $r_2(t)$ Denklem (52) ile ifade edilmiştir. Ajan hedefe ulaştığında sabit kaldığı her adım için d kadar ödül verilmiştir.

$$r_2(t) = \{d, (|u_i| = 0)\} \quad (52)$$

Hata normu belirli bir eşik değer altına düştüğünde bu bileşen devreye girerek ajanın hedef konumunda sabit kalmasını sağlar. Uç işlemcinin çalışma ortamı sınırları dışına çıkması durumunda uygulanan yüksek ceza $r_3(t)$ Denklem (53) ile ifade edilmiştir.

$$r_3(t) = \begin{cases} e, & (|p_x| > 0.725) \\ e, & (|p_y| > 0.725) \\ e, & (0 < |p_z| < 1.5) \end{cases} \quad (53)$$

Uç işlemci küresel olarak modellendiğinden engele çarpma durumu uç işlemci merkezinden engel merkezine doğru çizilen $\mathbf{k}$ vektörünün normundan hem uç işlemcinin hem de engelin yarıçapı çıkarılması ile hesap edilir. Böylece uç işlemci için belirlenen sanal

küre yüzeyinden engeli oluşturan sanal küre yüzeyine olan dik uzaklık hesap edilir. Benzer şekilde kablo üzerinde engele en yakın noktadan engel yüzeyine olan dik uzaklık (d_i) hesap edilerek kabloların engele çarpması durumu sorgulanır. Uç işlemci veya kabloların engele çarpması durumunda ise uygulanan ödül fonksiyonu $r_4(t)$ Denklem (54) ile ifade edilmiştir.

$$r_4(t) = \begin{cases} e & , \quad (|k| - a - c = 0) \\ e & , \quad (d_i = 0) \quad i = 1,2,3,4 \end{cases} \quad (54)$$

Ödül için kullanılan katsayılar; $a=0.01$, $b=-0.04$, $c=-0.035$, $d=2$, $e=-5$ olarak belirlenmiştir. Böylece hem uç işlemcinin hem de kabloların engele temas etmeden belirlenen referansa ulaşması amaçlanmıştır.

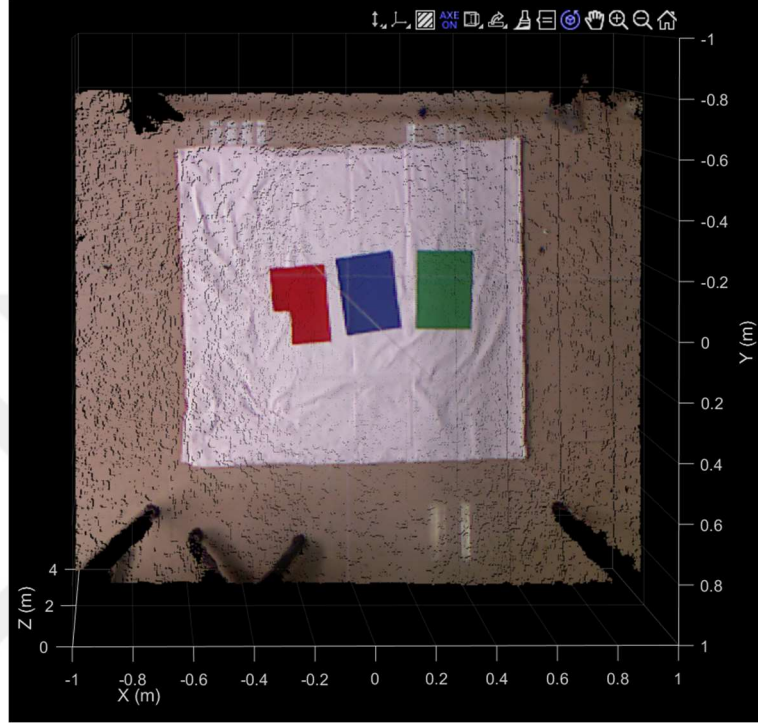
2.5.2. Görüntü İşleme ile Öznitelik Çıkarma

Yapılan çalışmada benzetim ortamında eğitilen PÖ ajanının gözlem uzayının büyük bir bölümü, doğrudan kinematik denklemler aracılığıyla elde edilen büyüklüklerden oluşmaktadır. Ancak benzetim ortamında engelin konumu ve boyutu sanal olarak tanımlanmaktadır. Fiziksel sistemde, konum ve boyut gibi gerekli büyüklüklerin doğrudan sensörler aracılığıyla tespit edilmesi gerekmektedir. Bu amaçla, KSPR sistemine entegre edilerek ana bilgisayara bağlanan RGB-D kameradan RGB ve derinlik verileri, MATLAB yazılımı üzerinde Image Acquisition Toolbox yardımıyla *colorDevice* ve *depthDevice* fonksiyonları aracılığıyla elde edilmiştir. Ardından bu verilerden üç boyutlu nokta bulutu oluşturulmuştur. Bu nokta bulutu üzerinden, çevredeki engellerin konum ve boyut bilgileri aşağıdaki gibi hesaplanmıştır.

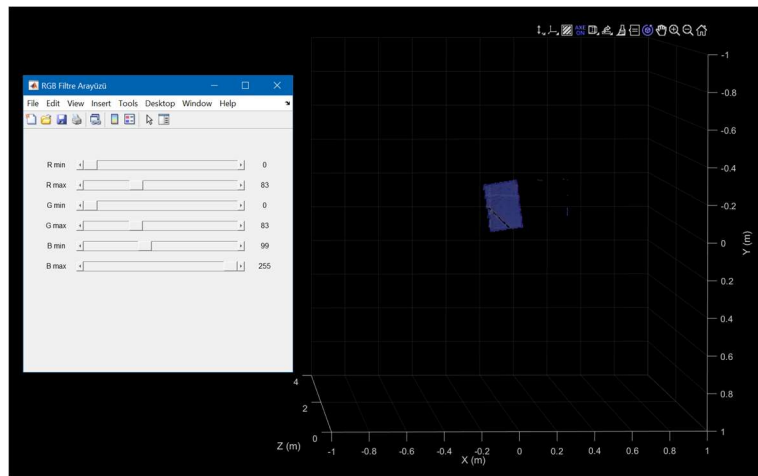
Uygulamada tespit edilecek objeler kırmızı, mavi ve yeşil gibi tek renkli olarak seçilmiştir. Bu sayede görüntü işleme süreci sadeleştirilmiş ve daha kararlı hale getirilmiştir. Şekil 25'te görülen filtre uygulanmamış nokta bulutu verisine, Denklem (55)'deki gibi öncelikle RGB görüntü üzerinden belirli bir renk aralığına karşılık gelen renk filtresi uygulanır.

$$\begin{aligned} \{Maske\} = & (R_{\{min\}} \leq R_i \leq R_{\{max\}}) \wedge (G_{\{min\}} \leq G_i \\ & \leq G_{\{max\}}) \wedge (B_{\{min\}} \leq B_i \leq B_{\{max\}}) \end{aligned} \quad (55)$$

Şekil 26, Şekil 27 ve Şekil 28’de görüldüğü gibi kırmızı, yeşil ve mavi renklerin alt ve üst eşikleri belirlenerek filtreleme yapılır. Bu filtreleme işlemi, yalnızca hedeflenen renge örneğin kırmızı renge sahip pikselleri izole ederken diğer renkleri görmezden gelir. Böylece renk tabanlı segmentasyon gerçekleştirilmiş olur.



Şekil 25. Renk filtresi uygulanmamış nokta bulutu



Şekil 26. Mavi renkteki objeye göre uygulanmış filtre

Mavi renge göre yapılan filtredeki kırmızı, yeşil ve mavi alt ve üst eşikleri, görüntüde sadece istenen obje kalacak şekilde belirlenmelidir. Bu objede:

Kırmızı ve yeşil için;

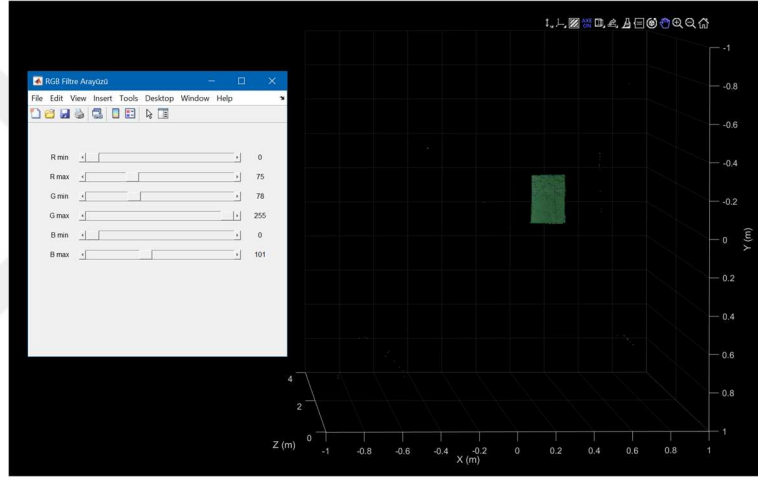
Alt eşik değeri= 0

Üst eşik değeri=83

Mavi için;

Alt eşik değeri=99

Üst eşik değeri=255 olarak belirlenmiştir. Böylece nokta bulutunda sadece filtrelenen objeyi oluşturan noktalar kalmaktadır. Bu noktalar daha sonra objeleri saracak olan kürelerin merkez ve çap ölçülerinin hesabı için kullanılmıştır.



Şekil 27. Yeşil renge göre uygulanmış filtre

Yeşil renge göre yapılan filtredeki kırmızı, yeşil ve mavi alt ve üst eşikleri, görüntüde sadece istenen obje kalacak şekilde belirlenmiştir. Bu objede:

Kırmızı için;

Alt eşik değeri= 0,

Üst eşik değeri=75,

Yeşil için;

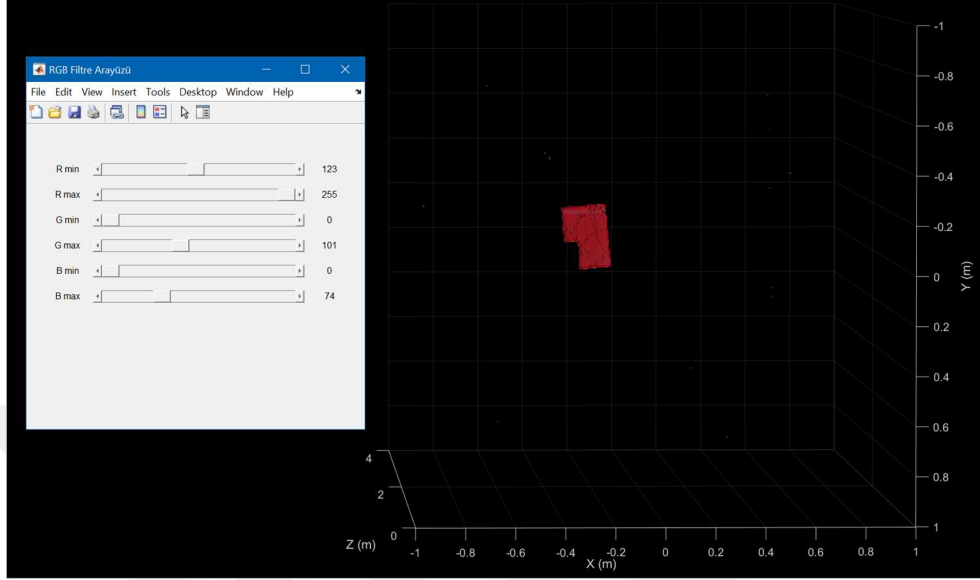
Alt eşik değeri=78,

Üst eşik değeri=255,

Mavi için;

Alt eşik değeri=0,

Üst eşik değeri=101 olarak belirlenmiştir. Böylece nokta bulutunda sadece filtrelenen objeyi oluşturan noktalar kalmaktadır.



Şekil 28. Kırmızı renge göre uygulanmış filtre

Kırmızı renge göre yapılan filtredeki kırmızı, yeşil ve mavi alt ve üst eşikleri, görüntüde sadece istenen obje kalacak şekilde belirlenmiştir. Bu obje:

Kırmızı için;

Alt eşik değeri= 123,

Üst eşik değeri=255,

Yeşil için;

Alt eşik değeri=0,

Üst eşik değeri=101,

Mavi için;

Alt eşik değeri=0,

Üst eşik değeri=74 olarak belirlenmiştir. Böylece nokta bulutunda sadece filtrelenen objeyi oluşturan noktalar kalmaktadır. Bu noktalar daha sonra objeleri saracak olan kürelerin merkez ve çap ölçülerinin hesabı için kullanılmıştır.

Cismin şekli ayırt edilmeksizin elde edilen renk maskesi üzerinden, ilgili renk kümesine ait piksellerin yoğunluk merkezi j Denklem (56) kullanılarak hesaplanmış ve kürenin yerleştirileceği merkez konumu üç boyutlu uzayda belirlenmiştir.

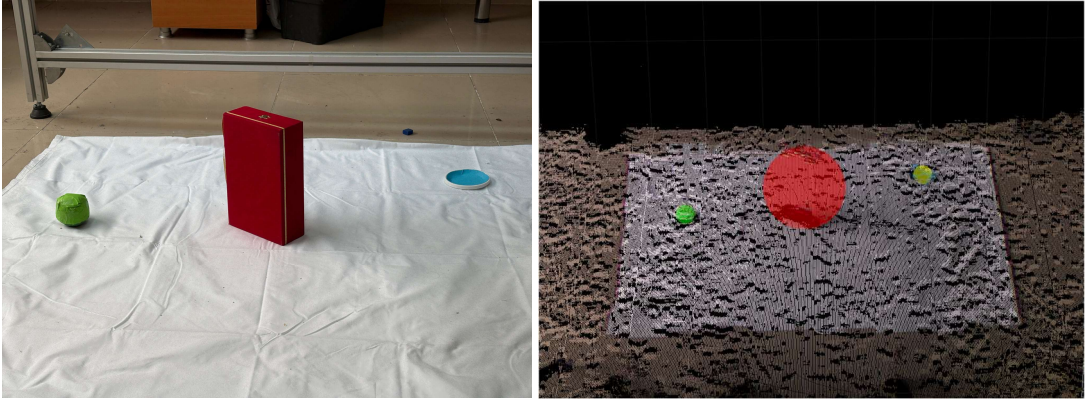
$$j = (\text{median}(x_i), \text{median}(y_i), \text{median}(z_i)) \quad (56)$$

Merkez konumu belirlendikten sonra aynı kümeye ait tüm piksellerin merkeze olan uzaklıkları Denklem (57) ile hesaplanmış ve bu uzaklıklar arasında en büyük olanı, ilgili küreye ait yarıçap a olarak atanmıştır.

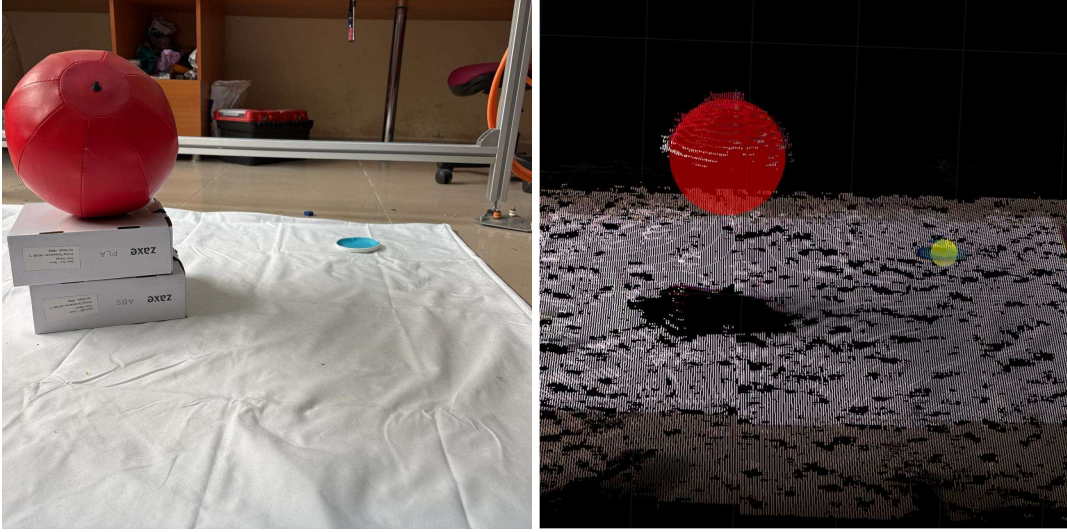
$$a_i = \sqrt{(x_i - j_x)^2 + (y_i - j_y)^2 + (z_i - j_z)^2} \quad (57)$$

Böylece tespit edilen engel, şekli ne olursa olsun uzayda konumu ve çapı belirlenmiş olan bir küre ile temsil edilmiştir. Bu işlem, hem sistemin geometri temelli hesaplamalarıyla uyumluluk sağlamış hem de PÖ ajanı tarafından kullanılabilir bir gözlem verisi sunmuştur.

Sonuç olarak Şekil 29 ve Şekil 30’da görüldüğü gibi fiziksel objeler, RGB-D kamera verileri işlenerek sahnedeki engel küresel bir obje olarak tanımlanmış; merkez konumu (x , y , z) ve çap (r) bilgisi çıkarılmıştır.

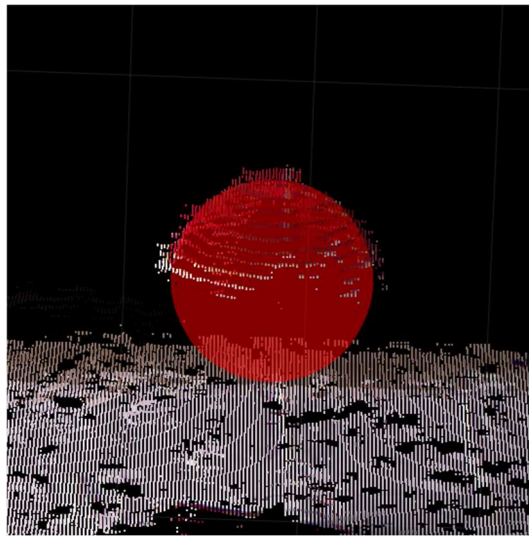


Şekil 29. Dikdörtgen prizma engelin, yeşil dairesel objenin ve düzlemsel referans noktasının küre içine alınması



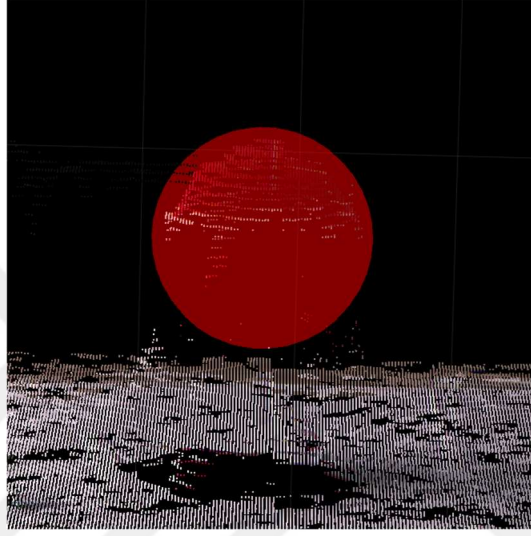
Şekil 30. Dairesel engelin, yeşil dairesel objenin ve düzlemsel referans noktasının küre içine alınması

Ayrıca, kamera görüntüsünden segmentasyon ile elde edilen engel konum ve boyut verilerinde belirgin sapmalar gözlemlenmiştir. Şekil 31’de görüldüğü gibi engel olarak kullanılan objenin çapı, Kinect kamerası tarafından hesaplandığında asıl çap ölçüsünden daha küçük ölçülmüştür. Bu durumun temel nedeni, değişken çevresel ışık koşulları altında renk tabanlı filtreleme yönteminin bazı yüzey piksellerini doğru şekilde ayırt edememesidir. Aydınlatma farklılıklarına rağmen sabit RGB eşik değerleri kullanılması, objenin bazı bölgelerinde piksellerin filtre dışında kalmasına neden olmuş; böylece küre şeklinde kapsanan obje daha küçük bir hacimle temsil edilmiştir.



Şekil 31. Sanal kürenin engelin tamamını kapsayamaması

Bu eksik temsil, engelin tamamının dikkate alınmamasına ve potansiyel çarpışma riskine yol açabileceğinden sistemde güvenlik payı içeren bir düzeltme uygulanmıştır. Şekil 32’de görüldüğü gibi kamera tarafından tahmin edilen küre çapı, her obje için %25 oranında artırılarak genişletilmiş ve bu sayede hem objenin tamamı küre içine alınmış hem de çevresinde güvenli bir tampon bölge tanımlanmıştır.



Şekil 32. Sanal kürenin engelin tamamını kapsamı

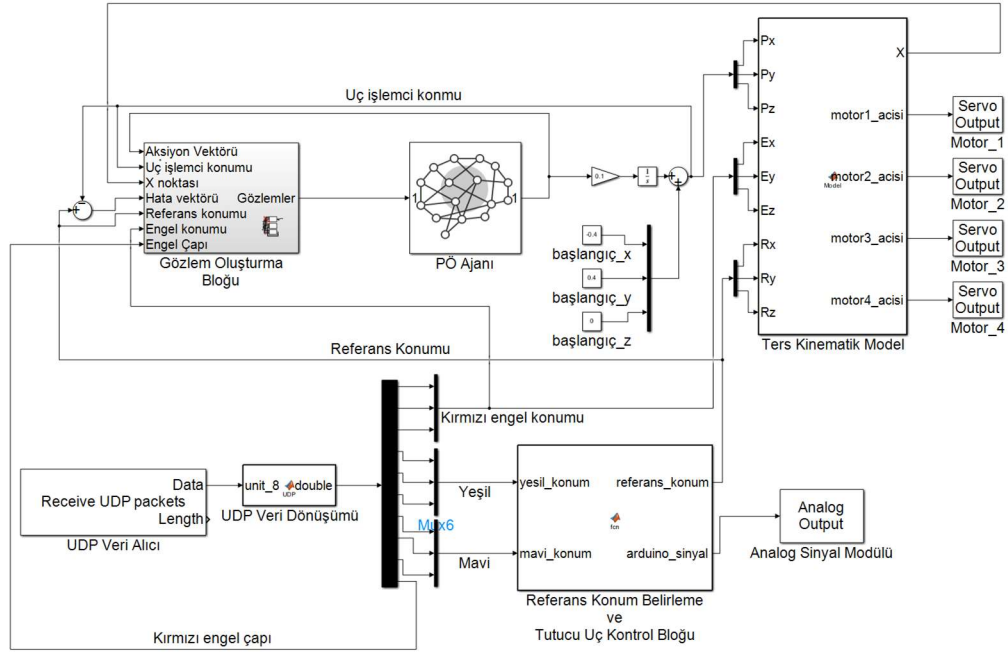
Bu genişletilmiş küresel temsil, uç işlemcinin engelle temasını önleyerek engele çarpmayan bir hareketin sağlanmasına katkı ve PÖ ajanının ortamı doğru şekilde gözlemleyebilmesi için gözlem uzayına dahil edilmiştir.

2.6. KSPR ile Tut-Yerleştir Uygulaması

Eğitim sürecini başarıyla tamamlamış olan PÖ ajanı, RGB-D kameradan elde edilen çevresel verileri kullanarak uç işlemcinin engele çarpmadan yörünge planlamasını mümkün hale gelmiştir. Tam anlamıyla otonom bir tut-yerleştir uygulaması için hedeflerin sistem tarafından otomatik olarak tanımlanması gerekmektedir. Bu amaç doğrultusunda daha önce engel tespiti amacıyla geliştirilen renk tabanlı görüntü işleme ve öznitelik çıkarma yapısı, sahadaki diğer objeleri de algılayabilecek şekilde genişletilmiştir. Geliştirilen algoritma, RGB görüntüde belirli renk aralıklarına ait pikselleri maskeleyerek yalnızca istenen renk özelliğine sahip objeleri ayırt edebilmektedir. Bu yapı, çalışma ortamına yerleştirilen farklı

renklerdeki objelerin ayırt edilebilmesine olanak tanımaktadır. Bu kapsamda, engel olarak kullanılan kırmızı objenin konumu daha önce tanımlanmış renk eşikleri ile tespit edilirken taşınacak obje için yeşil ve bırakılacağı hedef konum için mavi renkler atanmış ve aynı segmentasyon yöntemi bu renkler için de uygulanmıştır. Böylece tüm objeler için konum bilgileri hesaplanmıştır. Sonuç olarak sistem; engeli algılayan, obje ve referans konumlarını tespit eden ve bu verileri kullanarak uç işlemciyi otomatik olarak yönlendiren tam entegre, sensör destekli bir otonom tut-yerleştir uygulamasına hazır hale getirilmiştir.

Geliştirilen sistemin nihai hedefi; uç işlemcinin çalışma ortamındaki engele çarpmadan taşınacak objeye yönelmesi, objeyi güvenli biçimde kavraması ve ardından belirlenen hedef konuma ulaştırarak bırakmasıdır. Bu sürecin tamamen otonom olarak çalışabilmesi için PÖ ajanı ile gerçek zamanlı çalışan bir kontrol algoritması geliştirilmiş ve Simulink/Real-Time ortamı içerisinde bütünleşmiş bir yapıda uygulanmıştır. Bu kontrol akışı, sistemin tüm bileşenlerini bir araya getiren merkezi bir yapı olarak çalışmaktadır ve Şekil 33’te detaylı olarak gösterilmiştir.



Şekil 33. Otonom tut-yerleştir uygulamasında kullanılan Simulink/Realtime kontrol yapısı

Sürecin ilk aşamasında, taşınacak objenin konumu referans konum belirleme bloğu tarafından ajan için hedef olarak belirlenir. PÖ ajanı, eğitim sürecinde öğrendiği çarpışma olmayan hareket politikası doğrultusunda uç işlemciyi bu konuma yönlendirir. Uç işlemci

objeye ulaştığında ajan hedef konumda sabit kalır. Uç işlemci sabit pozisyondayken kontrol bloğu, tutucu mekanizmaya belirli bir süre boyunca kapanma sinyali gönderir ve objenin kavranması sağlanır. Ardından aynı kontrol bloğu, PÖ ajanına bu kez referans konumunu yeni hedef olarak gönderir. Ajan, yine engelden kaçınarak uç işlemciyi bu hedefe yönlendirir ve burada da sabit kalmayı sürdürür. Bu konumda, kontrol algoritması uç işlemciye açılma komutu göndererek objenin hedefe bırakılmasını sağlar. Bu işlem döngüsel bir yapıya sahiptir. Eğer kullanıcı tarafından obje tekrar alınıp başka bir konuma yerleştirilirse sistem bu değişikliği algılar ve aynı tut-yerleştir döngüsünü yeniden başlatır. Böylece sistem yalnızca bir defalık değil, sürekli çalışan otonom bir obje taşıma mekanizması olarak görev yapar.

2.7. Hiperparametre Optimizasyonu

PÖ eğitim çalışmalarında, ödül fonksiyonu tasarımı ve gözlem kümesinin içeriği, ajan performansını belirleyen en temel unsurlar arasında yer almaktadır. Ancak bu çalışmada, performans değişikliklerinin yalnızca hiperparametre ayarlarından kaynaklandığını izole bir biçimde inceleyebilmek amacıyla, ödül fonksiyonu ve gözlem seti sabit tutulmuştur. Böylece, farklı hiperparametre kombinasyonlarının eğitim sürecine ve öğrenme dinamiklerine etkisi, diğer faktörlerden bağımsız olarak değerlendirilebilmiştir.

DQA algoritmasında öğrenme süreci, hem ana ağın hem de hedef ağın parametre güncellemelerine bağlıdır. Bu nedenle, öğrenme mekanizmasını doğrudan etkileyen ve literatürde performans üzerinde kritik rol oynadığı belirtilen üç temel hiperparametreye odaklanılmıştır: öğrenme oranı, hedef ağ güncelleme parametreleri ve azaltma faktörü. Öğrenme oranı, ağırlık güncellemelerinin büyüklüğünü belirleyerek eğitim hızını ve kararlılığını etkilerken; hedef ağ güncelleme parametreleri, Q-değerlerinin kararlılığını ve politika optimizasyonunun istikrarını sağlar. Azaltma faktörü ise, ajanın gelecekteki ödüllere ne ölçüde önem vereceğini belirleyerek, kısa vadeli ve uzun vadeli stratejiler arasındaki dengeyi doğrudan etkiler.

Çalışma kapsamında, bu hiperparametreler sistematik olarak farklı değerlerle test edilmiş ve her kombinasyonun eğitim süreci üzerindeki etkisi ayrıntılı şekilde analiz edilmiştir. Denemeler, yalnızca nihai performans artışını hedeflemekle kalmamış, aynı zamanda farklı ayarların öğrenme dinamikleri, kararlılık, yakınsama hızı ve nihai başarımları üzerindeki etkilerini de gözlemlemeye odaklanmıştır. Elde edilen bulgular, eğitim sürecinin

hangi hiperparametre aralıklarında daha istikrarlı ve verimli çalıştığını ortaya koyarak, gelecekteki DQA tabanlı PÖ uygulamaları için deneysel bir referans sağlamaktadır.

2.7.1. Sabit Çaplı Küresel Engelden Kaçınma Uygulaması İçin Hiperparametre Optimizasyonu

Bu çalışmada; PÖ ajanının kararlı, verimli ve genellenebilir bir politika öğrenebilmesi amacıyla öğrenme oranı, hedef ağ güncelleme sıklığı ve azaltma faktörü parametreleri üzerinde sistematik bir hiperparametre arayışı gerçekleştirilmiştir. Bu süreçte denemeler yapılmıştır ve her bir parametre için elde edilen eğitim sonuçları değerlendirilmiştir. Yapılan bu optimizasyon çalışması, öğrenme sürecinin hem hızını hem de elde edilen nihai başarıyı önemli ölçüde etkilemiştir. Yapılan tüm deneylerde hedef ağ güncellenirken ana ağ katsayıları $1 \cdot 10^{-3}$ yumuşatma oranı ile çarpılarak güncelleme gerçekleştirilmiştir. Tablo 2’de kullanılan hiperparametreler gösterilmiş olup optimizasyon öncesi standart olarak kullanılan parametreler 16 numaralı deneyde kullanılan parametrelerdir. En uygun hiperparametre kombinasyonları, bulgular ve tartışma bölümünde ayrıntılı olarak sunulmuştur.

Tablo 2. Sabit engel çaplı eğitim için uygulanan hiperparametreler

Deney Numarası	Öğrenme Oranı	Hedef Ağ Güncelleme Sıklığı	Azaltma Faktörü
1	$2 * 10^{-4}$	5	0.99
2	$3 * 10^{-4}$	5	0.99
3	$4 * 10^{-4}$	5	0.99
4	$5 * 10^{-4}$	5	0.99
5	$6 * 10^{-4}$	5	0.99
6	$1 * 10^{-3}$	5	0.99
7	$1 * 10^{-4}$	1	0.99
8	$1 * 10^{-4}$	3	0.99
9	$1 * 10^{-4}$	5	0.99
10	$1 * 10^{-4}$	10	0.99
11	$1 * 10^{-4}$	20	0.99
12	$1 * 10^{-4}$	50	0.99
13	$1 * 10^{-4}$	5	0.999
14	$1 * 10^{-4}$	5	0.996
15	$1 * 10^{-4}$	5	0.993
16	$1 * 10^{-4}$	5	0.99
17	$1 * 10^{-4}$	5	0.987

2.7.2. Değişken Çaplı Küresel Engelden Kaçınma Uygulaması İçin Hiperparametre Optimizasyonu

Bu bölümde, bir önceki bölümde olduğu gibi PÖ ajanının kararlı, verimli ve genellenebilir bir politika öğrenebilmesi amacıyla öğrenme oranı, hedef ağ güncelleme sıklığı ve azaltma faktörü parametreleri üzerinde sistematik bir hiperparametre arayışı gerçekleştirilmiştir. Bu süreçte gözlem ve ödül fonksiyonları sabit tutularak denemeler yapılmıştır ve her bir parametre için elde edilen eğitim sonuçları değerlendirilmiştir. Yapılan bu optimizasyon çalışması, öğrenme sürecinin hem hızını hem de elde edilen nihai başarıyı önemli ölçüde etkilemiştir. Yapılan tüm deneylerde hedef ağ güncellenirken ana ağ

katsayıları yumuşatma oranı kullanılmadan güncelleme gerçekleştirilmiştir. Tablo 3’de kullanılan hiperparametreler gösterilmiş olup optimizasyon öncesi standart olarak kullanılan parametreler 2 numaralı deneyde kullanılan parametrelerdir. En uygun hiperparametre kombinasyonları, bulgular ve tartışma bölümünde ayrıntılı olarak sunulmuştur.

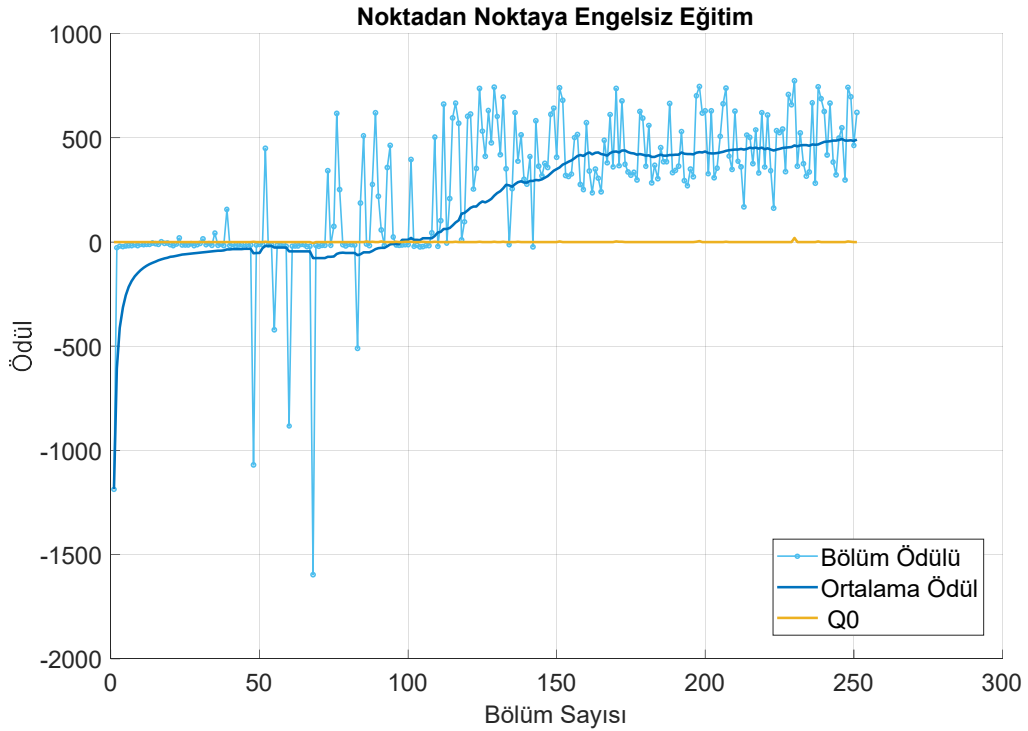
Tablo 3. Değişken çaplı eğitim için uygulanan hiperparametreler

Deney Numarası	Öğrenme Oranı	Hedef Ağ Güncelleme Sıklığı	Azaltma Faktörü
1	$1 * 10^{-4}$	640	0.99
2	$1 * 10^{-4}$	640	0.95
3	$1 * 10^{-4}$	640	0.90
4	$1 * 10^{-4}$	640	0.85
5	$1 * 10^{-4}$	640	0.80
6	$1 * 10^{-4}$	640	0.75
7	$1 * 10^{-3}$	640	0.95
8	$5 * 10^{-3}$	640	0.95
9	$5 * 10^{-4}$	640	0.95
10	$1 * 10^{-5}$	640	0.95
11	$1 * 10^{-4}$	320	0.95
12	$1 * 10^{-4}$	690	0.95
13	$1 * 10^{-4}$	1280	0.95

3. BULGULAR

3.1. Engelsiz Benzetim Ortamında Pekiştirmeli Öğrenme Eğitimi Bulguları

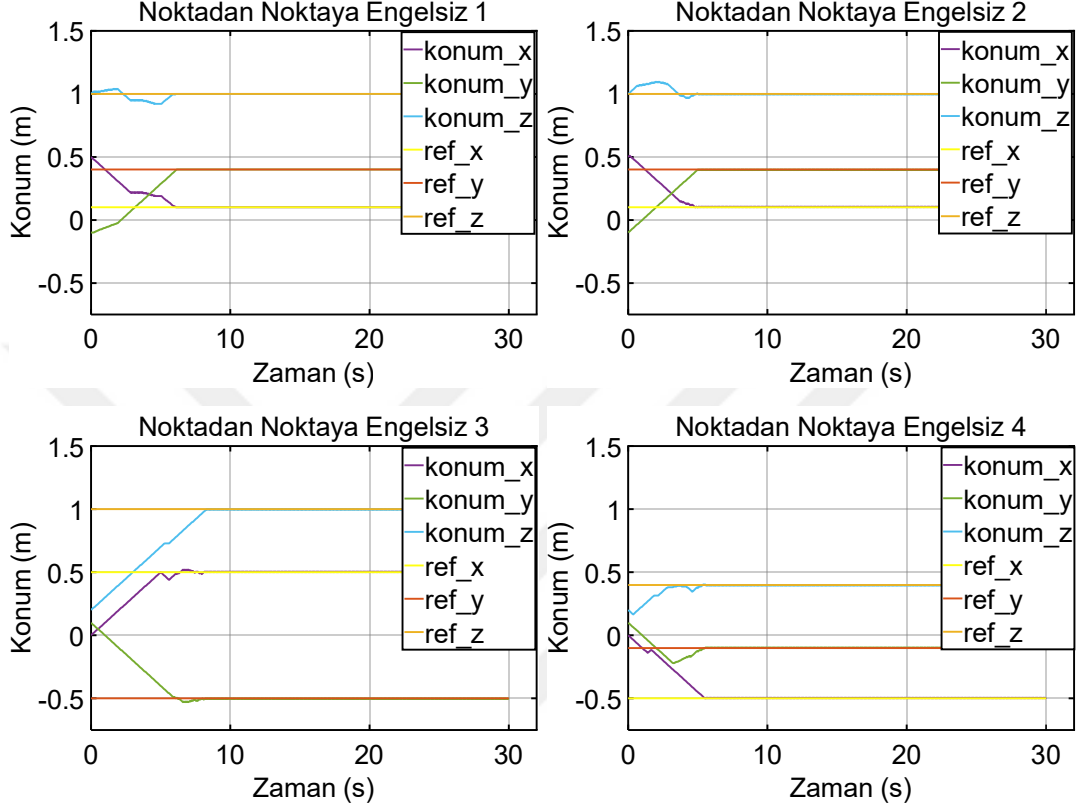
Engelsiz benzetim ortamında benzetim çalışmaları MATLAB programında DQA algoritması kullanarak yapılmıştır. Yapılan eğitimler kaynak kullanımı ve zaman verimliliği açısından aksiyon adımları 1 cm olarak belirlenmiştir. Yapılan çalışmada Şekil 34'te görüldüğü gibi ortalama ödül yaklaşık 160.bölümden sonra sabit ve kararlı devam etmiştir ve bu durum eğitimin doygunluğa ulaştığını göstermektedir. Eğitim süreci 250 bölümde sonlandırılmıştır. Eğitim yaklaşık 30 dakika sürmüş ve ajan yüksek ödül alacağı politikayı kısa sürede öğrenmiştir.



Şekil 34. Noktadan noktaya engelsiz PÖ ajan eğitimi

Eğitim sonuçlarında bölüm ödülleri gözlenen iniş çıkışların temel nedeni, her bölümün rastgele seçilen başlangıç koşullarında başlatılmasıdır. Bazı başlangıç durumlarında uç işlemci veya kablolar, simülasyonun ilk adımında engele çarptığı için

bölüm hemen sona ermiş ve bu nedenle sıfır ödül kaydedilmiştir. Ajan Şekil 35’te görüldüğü gibi rastgele üretilen başlangıç ve referans konumlarıyla yapılan 100 denemenin 98’inde başarıyla hedef konuma ulaşmıştır.



Şekil 35. Rastgele başlangıç ve referans konumlarıyla yapılan denemeler

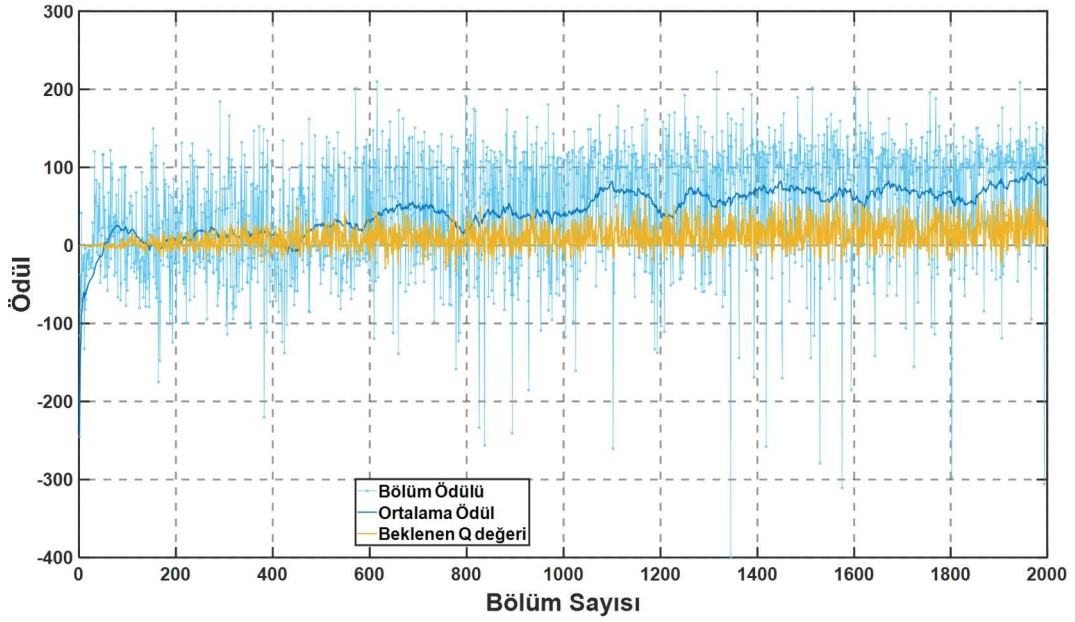
Test sonuçlarına bakıldığında ajanın sergilediği aksiyon seçiminde her üç eksendeki başlangıç konumları ve eksenlerin referans konuma olan uzaklıkları farklı olsa da tüm eksenlerde aynı anda referansa ulaşacak bir davranış sergilemiştir. Bu sebeple yakın olan eksen referans değerine ulaşsa bile diğer eksenler henüz ulaşmadığından o eksenin tekrar referans konumundan saptığı gözlemlenmiştir. Sergilenen bu davranışın ödül fonksiyonu ile büyük oranda ilgili olduğu değerlendirilmiştir. Bu çalışmada kullanılan ödül fonksiyonuna göre üç eksen de referansa ulaştığında en yüksek ödül alındığından ve iki eksendeki yaklaşma hareketinin tek bir eksenin referanstan uzaklaşma cezasını tolere etmesinden dolayı ajanın bu hareketi sergilediği sonucuna varılmıştır.

3.2. Sabit Çaplı Küresel Engel İçin Hiperparametre Optimizasyonun Eğitim Sonuçlarına Etkisi

Yapılan optimizasyon çalışmaları sonucunda, eğitim performansını maksimize eden hiperparametre kombinasyonu belirlenmiş ve bu optimum değerlerle eğitilen ajan, test ortamında değerlendirilmiştir. Elde edilen sonuçlar, hiperparametre seçiminin eğitim başarısı üzerinde belirleyici bir rol oynadığını açıkça göstermektedir.

3.2.1. Öğrenme Oranı Optimizasyonu

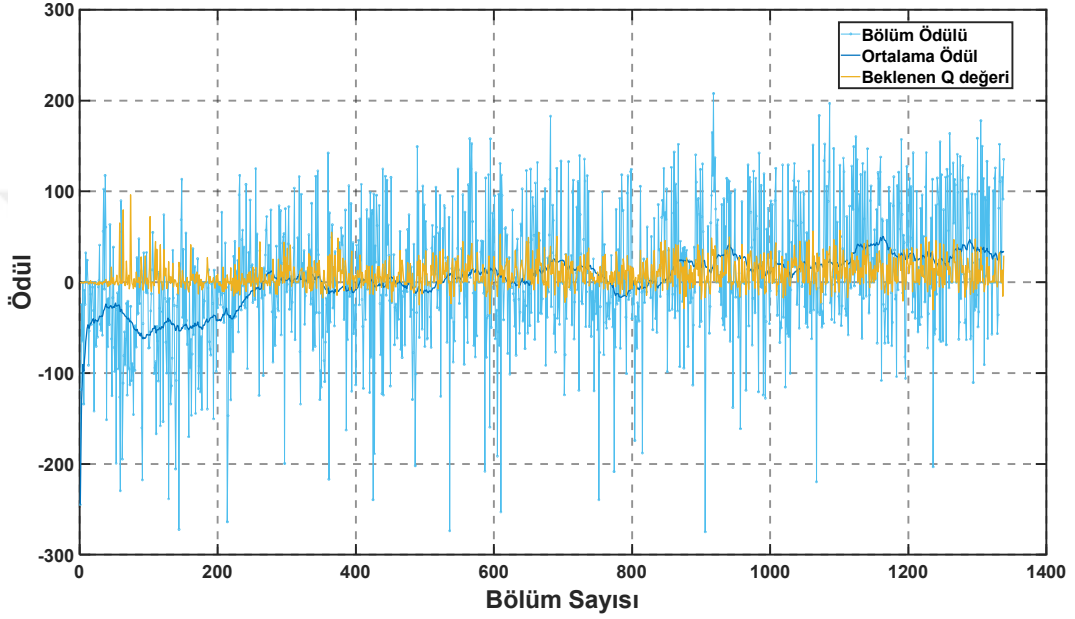
Farklı öğrenme oranı ile yapılan 1 ile 6 numaralı arasındaki eğitimlerin sonuçları aşağıdaki şekillerde görüldüğü gibi gözlemlenmiştir.



Şekil 36. Deney 1, Öğrenme Oranı: $2 * 10^{-4}$

Şekil 36'da öğrenme oranı olarak $\alpha = 2 \times 10^{-4}$ seçildiğinde elde edilen eğitim verileri görülmektedir. Eğitimin ilerleyen bölümlerinde, hem tahmin edilen Q değerlerinde istikrarlı bir yakınsama eğilimi hem de ortalama ödül değerlerinde belirgin bir artış gözlemlenmiştir. Bu durum, ajan tarafından öğrenilen politikanın zamanla geliştiğini ve daha iyi kararlar verdiğini göstermektedir. Düşük öğrenme oranı sayesinde sınır ağı parametreleri, daha küçük adımlarla güncellenmiş ve bu da öğrenme sürecinde ani yükselip

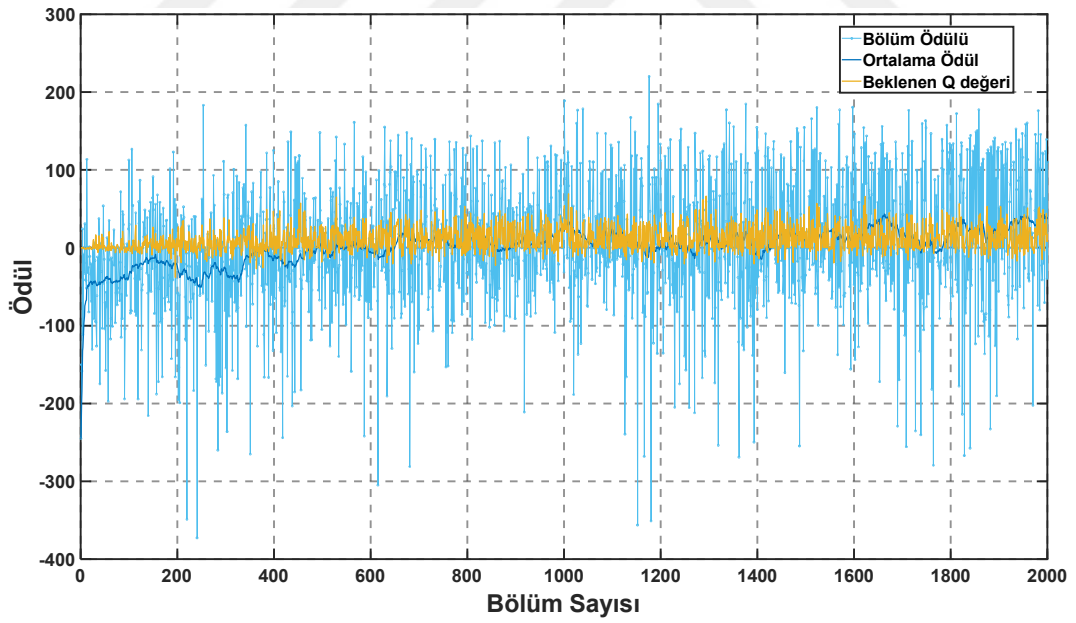
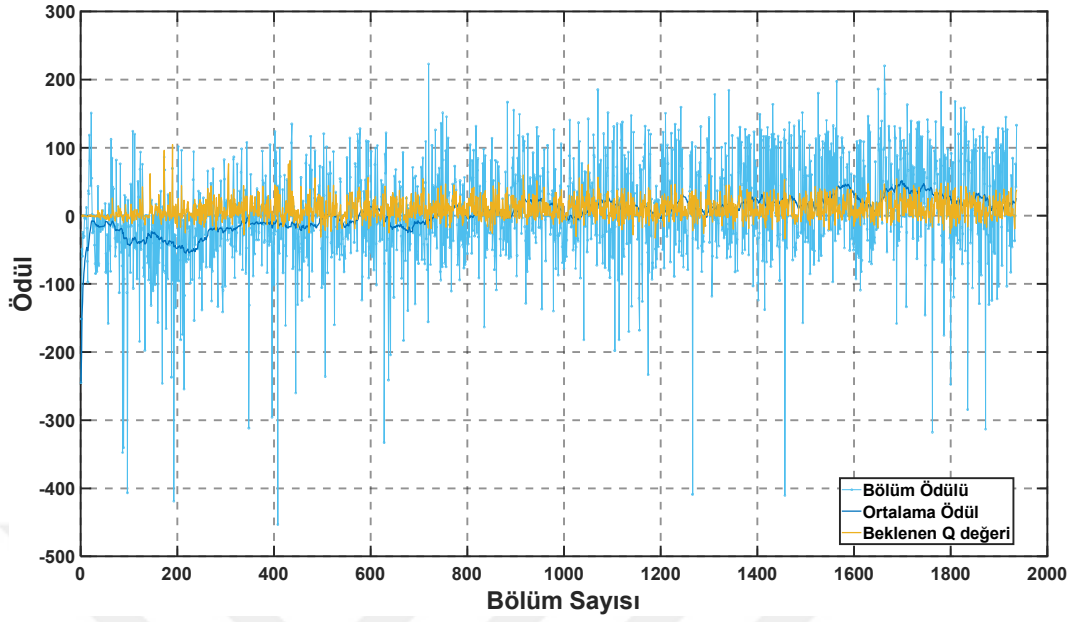
alçalmaların önüne geçerek Q değerlerinin daha kararlı ve doğru bir şekilde yakınsanmasına imkân tanımıştır. Seçilen bu düşük öğrenme oranı, özellikle karmaşık ve yüksek boyutlu gözlem uzayına sahip KSPR sistemlerinde, engele çarpmadan yol planlanması için kararlı ve güvenilir bir öğrenme süreci sağlamıştır. Böylece elde edilen politika yalnızca ödül bazında değil, aynı zamanda tahmin edilen Q değerlerinin stabilitesi açısından da başarılı bir performans ortaya koymuştur.

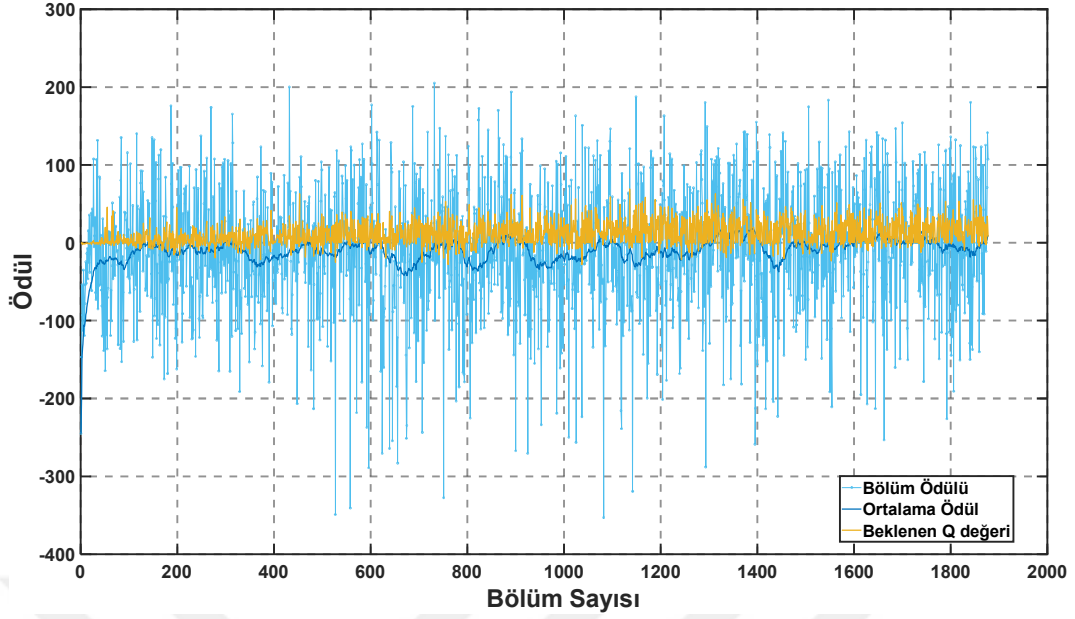


Şekil 37. Deney 2, Öğrenme Oranı: $3 * 10^{-4}$

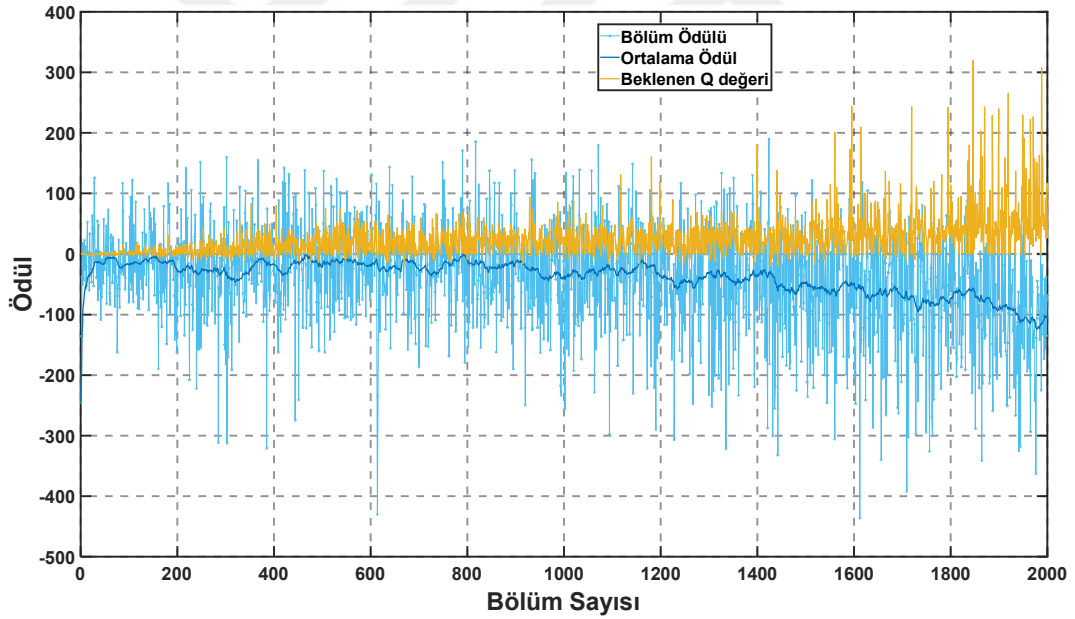
Şekil 37’de öğrenme oranının $3 * 10^{-4}$ olarak belirlendiği durumda elde edilen eğitim verileri gösterilmektedir. Görüldüğü üzere, öğrenme oranında yapılan küçük bir artış, eğitim sürecini ciddi şekilde etkilemiştir. Genel beklenti, öğrenme oranı arttıkça ağı daha hızlı yakınsayarak ödül fonksiyonunu iyileştirmesidir. Ancak burada ağı parametreleri daha büyük adımlarla güncellendiği için tahmin edilen Q değerleri başta oldukça tutarsız hale gelmiştir.

Eğriler incelendiğinde ortalama ödül değerlerinin sınırlı bir artış gösterdiği, buna karşılık hedef Q değerlerinin sürekli yükseldiği gözlemlenmektedir. Bu durum, ağı kısa vadeli getirilerle aşırı iyimser hedefler üretmeye başladığını ve politikasını sağlıklı biçimde güncelleyemediğini göstermektedir. Ayrıca bazı bölümlerde yüksek ödüller alınmasına rağmen bazı diğer bölümlerde yüksek cezalar elde edilmiştir. Bu da öğrenme sürecinin dengesiz ve kararsız olduğunu ortaya koymaktadır.





Şekil 40. Deney 5, Öğrenme Oranı: $6 * 10^{-4}$



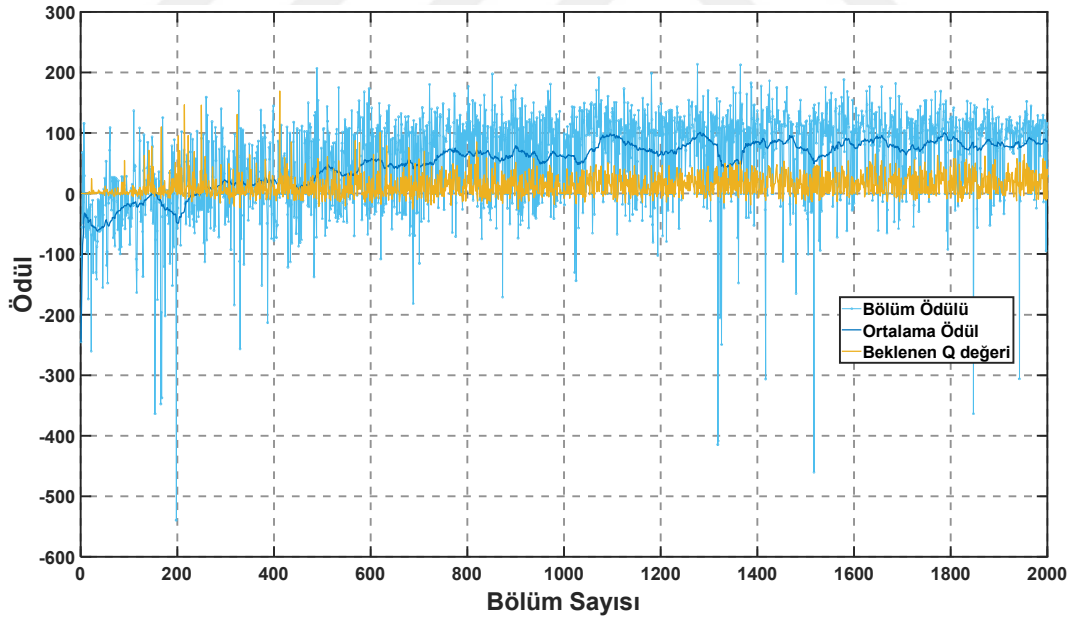
Şekil 41. Deney 6, Öğrenme Oranı: $1 * 10^{-3}$

Eğitim sonuçlarına bakıldığında öğrenme oranı büyüdükçe PÖ ajanı performansının belirgin şekilde azaldığı gözlemlenmiştir. Bu durumun temel sebebi öğrenme oranının yapay sinir ağı ağırlıklarının güncellemeleri üzerindeki etkisidir. Düşük bir öğrenme oranı ağı daha küçük adımlarla güncellenerek istikrarlı bir politika öğrenmesini sağlar. Bu sayede Q değerlerinin doğru yönde güncellenir. Fakat öğrenme oranı büyüdükçe sinir ağının ağırlıkları daha büyük adımlarla güncellendiğinden optimizasyon süreci daha kararsız hale

gelerek Q değerlerinde sapmaya ve ajanın doğru bir politika öğrenememesine sebep olmuştur. Ayrıca DQA algoritmasında hedef Q değerlerini tahmin eden hedef ağ gecikmeli olarak güncellendiğinden, yüksek öğrenme oranı gecikmeli güncellenmenin avantajlarını da ortadan kaldırmıştır. Grafiklerde görüldüğü gibi hedef ödül değerleri ile hesaplanan ödül değerleri arasında tutarsızlıklar görülmektedir. Bu durumda öğrenilen politika, yüksek öğrenme oranının neden olduğu dengesiz ağırlık güncellemeleri nedeniyle kararsız hale gelmiştir. Bu da ajanın çevreden aldığı ödül sinyallerini sağlıklı şekilde genelleştirememesine ve dolayısıyla tutarsız aksiyon seçimleri yapmasına yol açmıştır. Bu sebeple öğrenme oranının mümkün olduğunca küçük tutulması, kararlı öğrenme açısından oldukça önemli bir rol oynamaktadır.

3.2.2. Hedef Ağ Güncelleme Sıklığı Optimizasyonu

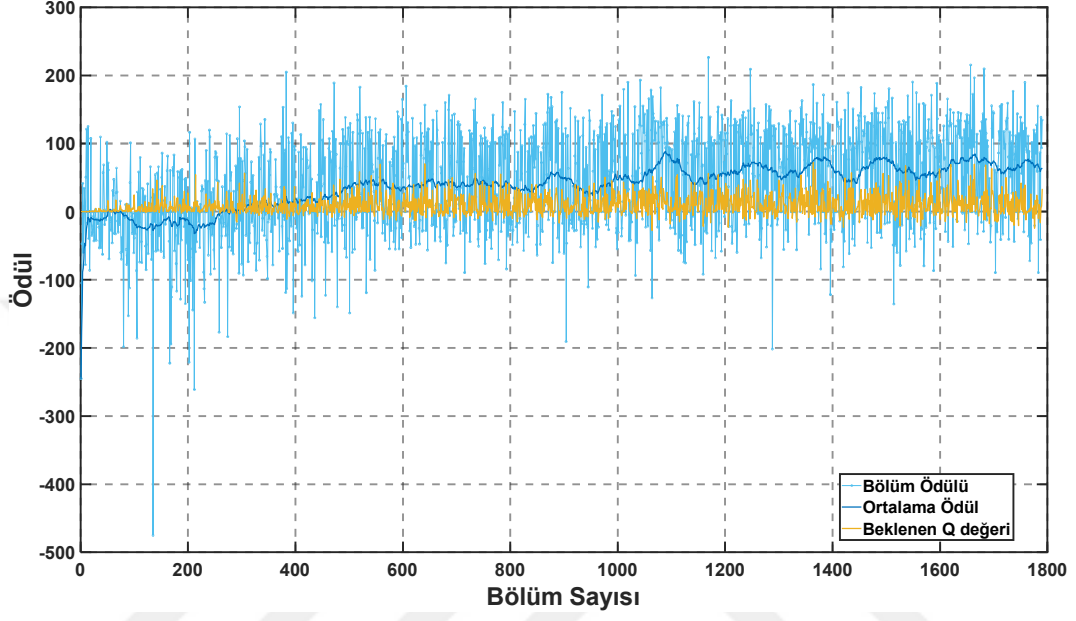
Farklı hedef ağ güncelleme sıklığı ile yapılan 7 ile 12 numara arasındaki eğitimlerin sonuçları aşağıdaki şekillerde görüldüğü gibi gözlemlenmiştir.



Şekil 42. Deney 7, Hedef Ağ Güncelleme Sıklığı: Her adımda (1)

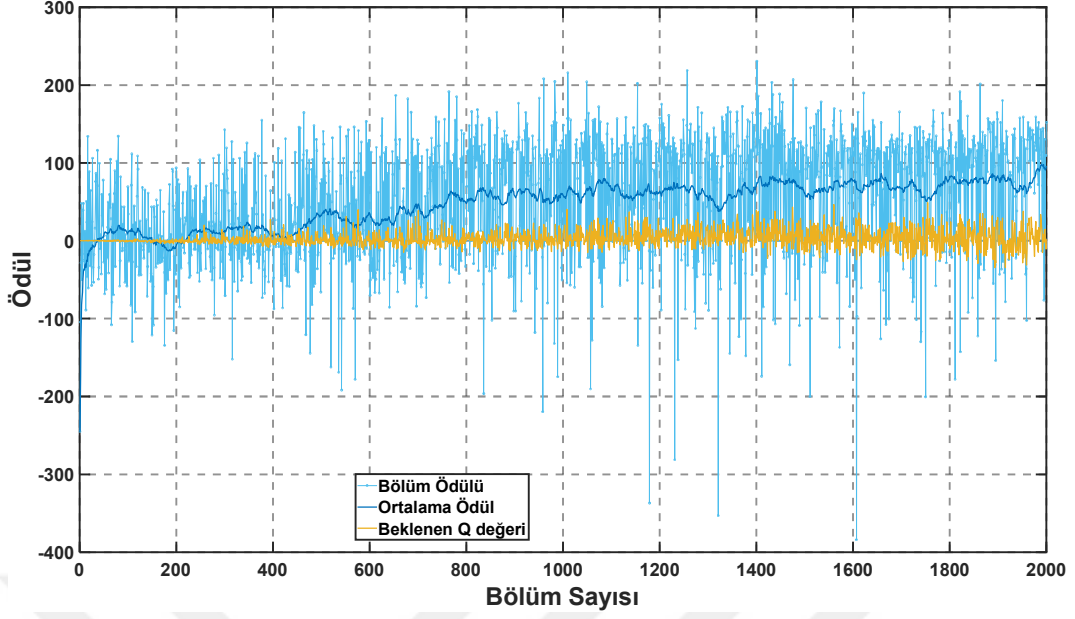
Şekil 42'de, hedef ağın her adımda güncellendiği durumda elde edilen eğitim sonuçları yer almaktadır. Eğitimin ilk evrelerinde, ajan henüz yeterli deneyim kazanmadan önce, hedef ödül değerlerinde belirgin bir aşırı tahmin gözlemlenmektedir. Bu durumun temel nedeni, hedef ağın her adımda güncellenmesidir. Ana ağın henüz doğru tahmin yapacak bilgiye sahip

olmaması, hedef ağı da hatalı yönlendirmiştir. Zamanla deneyim arttıkça ortalama ödül ve hedef ödül değeri birbirine yaklaşmış, böylece sınırlı da olsa bir öğrenme gerçekleşmiştir. Ancak yine de bölüm bazında görülen ani yükselme ve düşmeler, sistemin hâlâ kararsız çalıştığını ve öğrenmenin istikrarsız ilerlediğini ortaya koymaktadır.



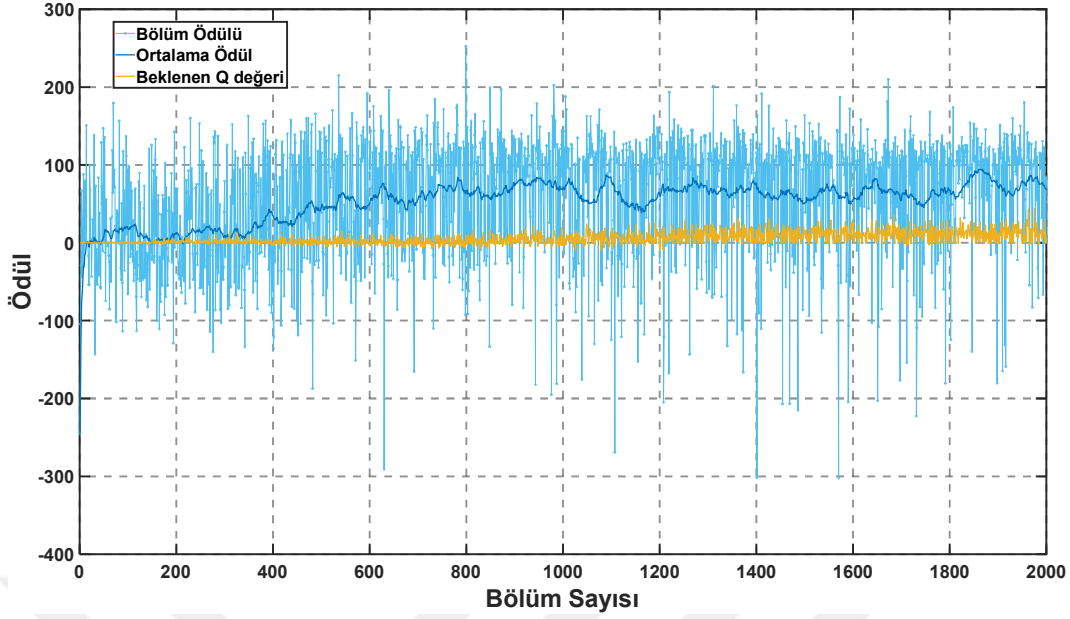
Şekil 43. Deney 8, Hedef Ağ Güncelleme Sıklığı: 3 adımda 1 defa

Şekil 43'te, hedef ağı her 3 adımda bir güncellenmesi ile elde edilen öğrenme eğrisi sunulmaktadır. Bu ayarlama, önceki durumda görülen sert iniş ve çıkışlara kıyasla daha kararlı bir ortalama ödül artışı sağlamıştır. Eğitim sürecinin başında yine hedef ödül değerlerinde belirgin bir aşırı tahmin görülmekle birlikte, bu fark ilerleyen bölümlerde azalarak dengeli bir yakınsama göstermiştir. Ayrıca hedef ödül değerlerinin, bölüm ödülleriyle daha az iniş çıkış yaptığını ve eğitimin sonunda daha istikrarlı bir seviyeye oturduğu gözlemlenmiştir. Bu durum, hedef ağı güncelleme frekansının düşürülmesinin, öğrenme sürecinde ağı kararlılığını artırdığına işaret etmektedir.



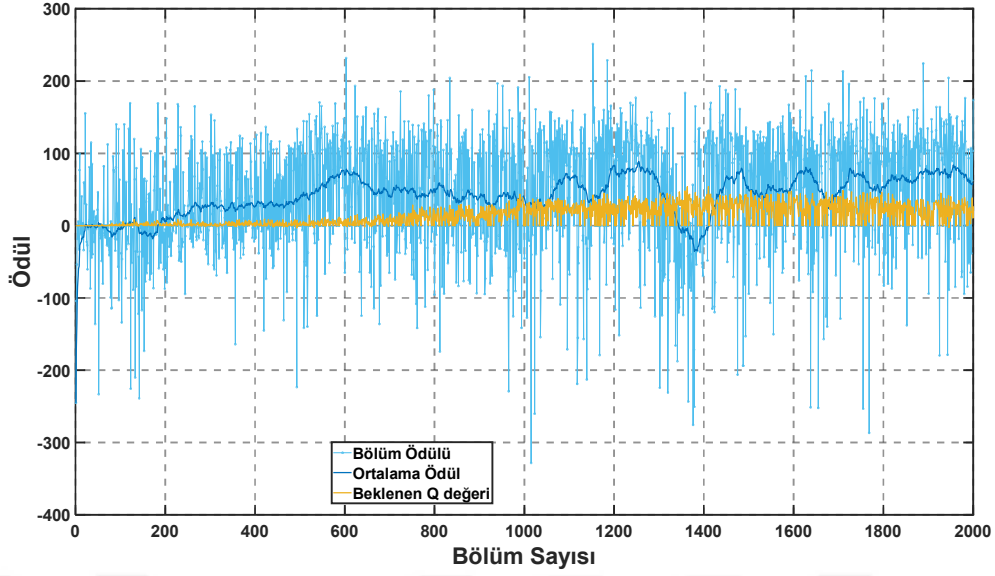
Şekil 44. Deney 9, Hedef Ağ Güncelleme Sıklığı: 5 adımda 1 defa

Şekil 44'te hedef ağın her 5 adımda bir güncellendiği durumda elde edilen sonuçlar görülmektedir. Bu ayarda, eğitim sürecinin başlarında gözlemlenen aşırı hedef ödül değeri tahminleri önemli ölçüde azaltılmıştır. Hedef ağın ana ağdan bir miktar gecikmeli güncellenmesi sayesinde yanıltıcı değer tahminlerinin önüne geçilmiş, böylece daha istikrarlı bir öğrenme süreci sağlanmıştır. Grafik üzerinde, ortalama ödül ve hedef ödül değerinin paralel bir şekilde arttığı, ayrıca eğitim ilerledikçe daha düşük varyanslı ve daha tutarlı bir ödül eğrisi elde edildiği görülmektedir. Bu durum, 5 adımda bir güncellemenin, hedef ağın güvenilirliğini koruyarak öğrenme sürecine olumlu katkı sağladığını göstermektedir.



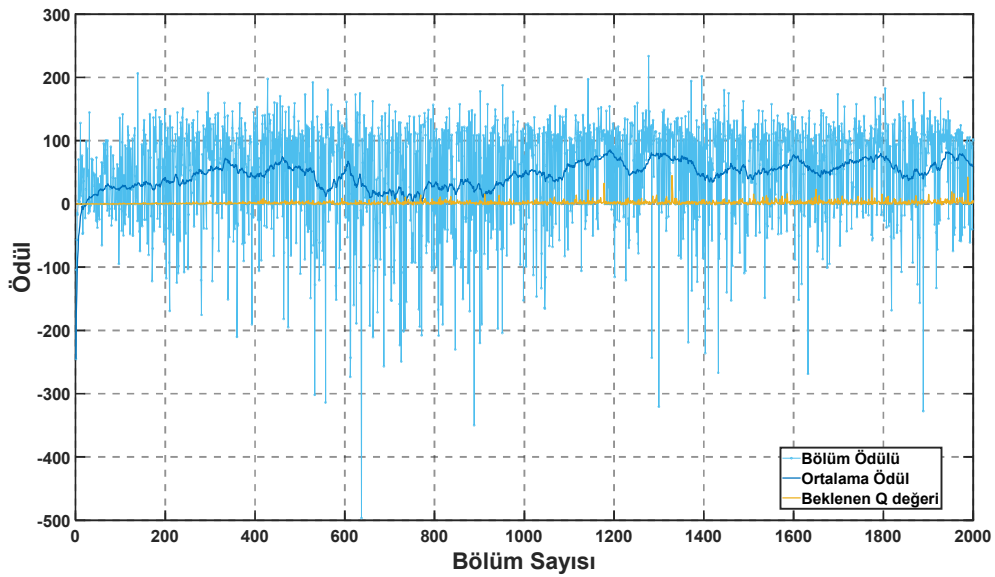
Şekil 45. Deney 10, Hedef Ağ Güncelleme Sıklığı:10

Şekil 45'te hedef ağın her 10 adımda bir güncellendiği durumda elde edilen sonuçlar gösterilmektedir. Eğitimin ilk aşamalarında hem çok yüksek hem de çok düşük ödüllerin alındığı görülmekte, bu da sistemin başlangıçta oldukça kararsız olduğunu ortaya koymaktadır. Zaman ilerledikçe ortalama ödül genel olarak artış eğilimi gösterse de bu artışın düzenli ve istikrarlı olmadığı eğrinin iniş çıkışlı yapısından anlaşılmaktadır. Ayrıca, beklenen ödül değeri ile elde edilen ortalama ödül arasında sürekli bir fark bulunmakta, bu durum da hedef ödül değerlerinin alınan ödül değerleriyle tutarlı şekilde yakınsamadığını göstermektedir. Sonuç olarak, güncelleme sıklığının seyrekleşmesi öğrenmenin verimini düşürmüş ve kararsızlığı artırmıştır.



Şekil 46. Deney 11, Hedef Ağ Güncelleme Sıklığı: 20 adımda 1 defa

Şekil 46’da görüldüğü üzere, eğitim süreci boyunca ortalama ödül değerleri ciddi bir kararsızlık sergilemiştir. Beklenen ödül değerinin zamanla daha düzenli ve artan bir yapı göstermesine rağmen ajanın elde ettiği ortalama ödül, bu artışı takip edememiş ve ciddi iniş çıkışlarla negatif eğilimler göstermiştir. Bu durum, ajan tarafından tahmin edilen hedef ödül değerlerinin, ortamdan alınan asıl ödülleri yeterince temsil edemediğini göstermektedir. Böylece güncelleme sıklığının yetersizliği sebebiyle ajanın optimal politikayı öğrenemediği ve kararsız davranışlar sergilediği görülmektedir.



Şekil 47. Deney 12, Hedef Ağ Güncelleme Sıklığı: 50 adımda 1 defa

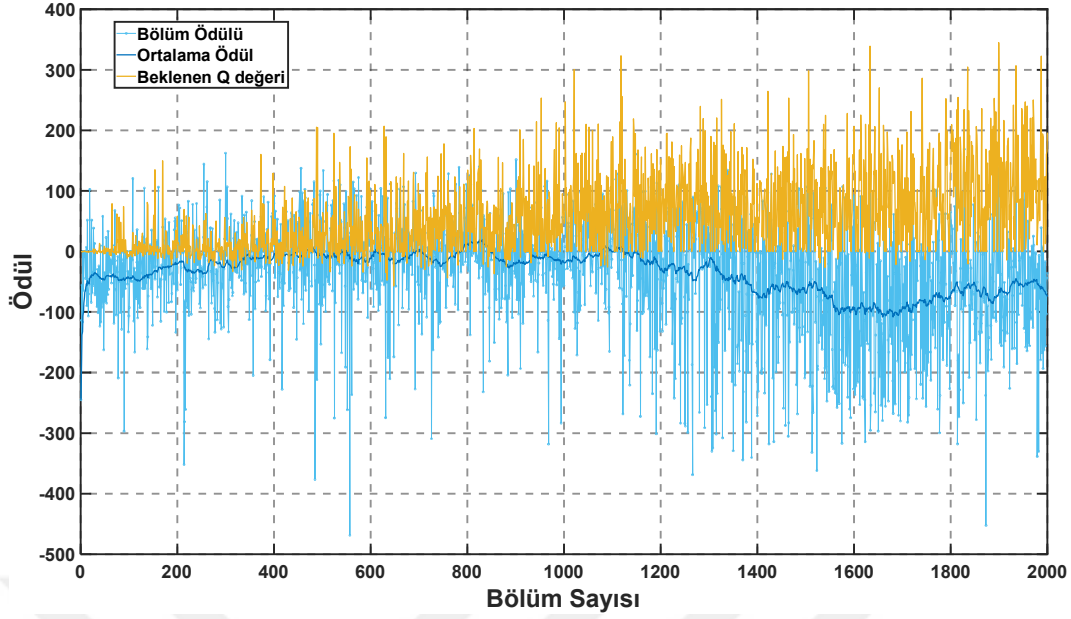
Şekil 47’de beklenen ödül değeri, eğitimin tamamında ortalama ödülün belirgin şekilde altında kalmıştır. Bu durum, hedef ve ana ağ arasında ciddi bir tutarsızlık olduğunu göstermektedir. Ayrıca, ortalama ödüldeki düzensiz iniş çıkışlar ve zamanla azalan değerler, öğrenmenin kararsızlaştığını ve ajan performansının sürdürülebilir olmadığını ortaya koymaktadır.

Yapılan deneysel analizler, hedef ağ güncelleme sıklığının PÖ ajanının öğrenme başarımı üzerinde doğrudan etkili olduğunu göstermiştir. Her adımda yapılan güncellemeler aşırı tahmin hatasına neden olurken, bu durum hedef ödül değeri ile elde edilen ödül arasında dengesizlik yaratmıştır. Güncelleme sıklığı üçe çıkarıldığında öğrenme daha kararlı hale gelmiş, ortalama ödül ve beklenen ödül değeri daha uyumlu seyretmiştir. Beş adımda bir güncelleme yapılan durumda ise aşırı tahmin hatası büyük ölçüde ortadan kalkmış ve ortalama ödül istikrarlı biçimde artmıştır. Ancak güncelleme sıklığının daha da artırılması öğrenmenin dengesizleşmesine yol açmıştır. Beklenen ödül değerleri artarken ajan bu değerleri asıl ödüle yansıtamamış ve öğrenme tutarsız bir hale gelmiştir. Özellikle 20 adımda bir yapılan güncellemede hedef ağın, ana ağın gerisinde kaldığı gözlemlenmiş, bu da hedef ağın yetersiz güncellenmesine bağlı olarak politikada bozulmalara neden olmuştur.

Sonuç olarak, güncelleme sıklığı belirli bir noktaya kadar artırıldığında öğrenmeyi dengeleyip aşırı tahmini azaltırken, bu sınır aşıldığında öğrenme kapasitesi düşmekte ve performans olumsuz etkilenmektedir. Bu çalışmada n uygun sonuç 5 adımda bir güncelleme yapılan senaryoda elde edilmiştir.

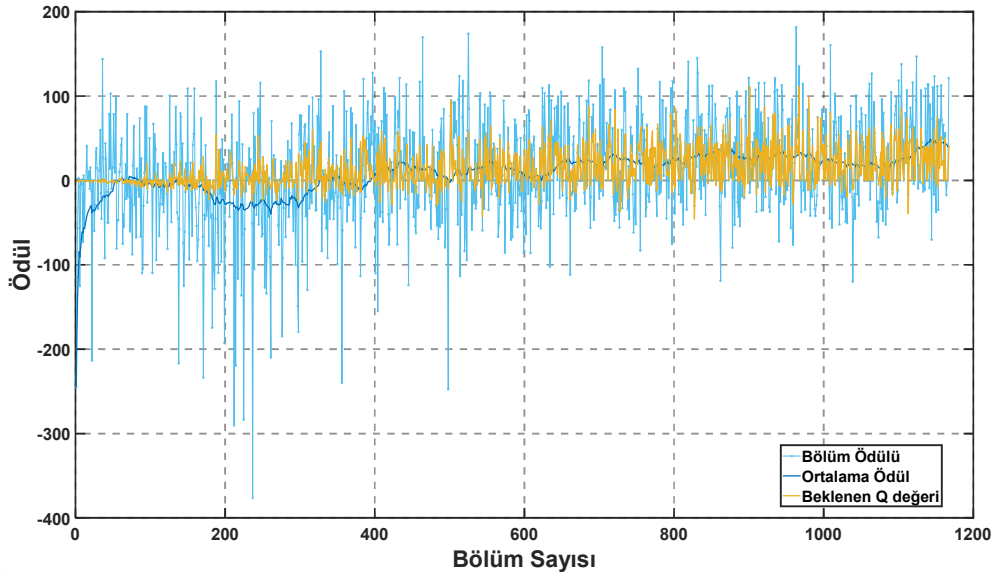
3.2.3. Azaltma Faktörü Optimizasyonu

Farklı azaltma faktörü ile yapılan 13 ile 17 numara arasındaki eğitimlerin sonuçları aşağıdaki şekillerde görüldüğü gibi gözlemlenmiştir.



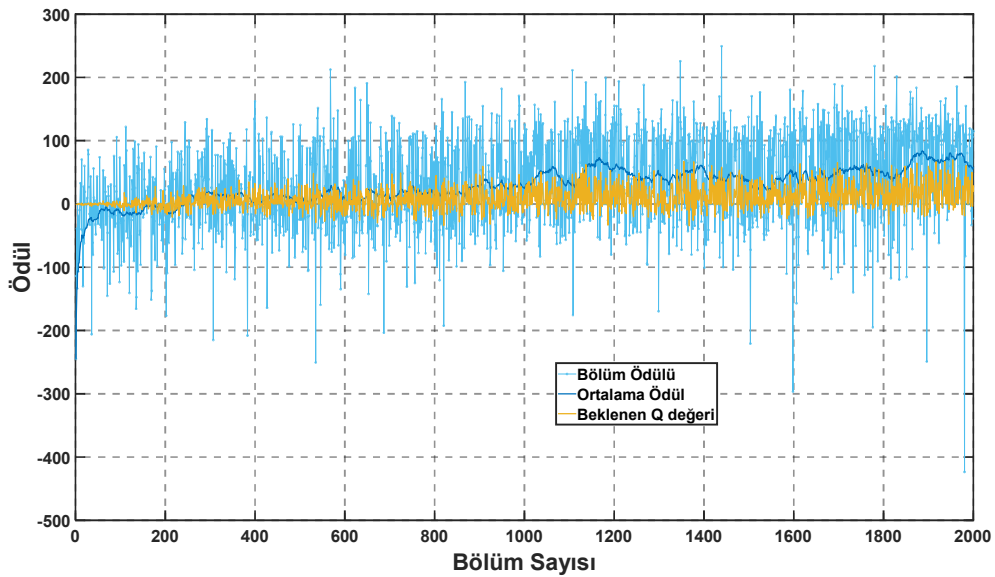
Şekil 48. Deney 13, Azaltma faktörü $\gamma=0.999$

Azaltma faktörü 0.999 gibi çok yüksek bir değere ayarlandığında, ajan uzun vadeli ödüllere oldukça fazla önem vermeye başlamıştır. Böyle bir durumda, anlık ödüller ile gelecekte alınacak ödüllerin ağırlığı neredeyse eşitlenmiş olur ve ajan, aksiyonlarını seçerken ilerideki ödülleri daha çok ön planda tutar. Bu kadar yüksek bir azaltma faktörü kullanıldığında ajan, kısa vadede hata yapmaktan çekinmemiş, uzun vadeli ödüllere çok fazla odaklandığından doğru davranışı öğrenmekte zorlanmıştır. Bu nedenle, bölüm ödülleri artmak yerine zamanla azalmış ve eğitim süreci istikrarsız bir hâl almıştır. Sonuç olarak, Q değerleri birikerek gerçekçi olmayan şekilde çok yüksek seviyelere ulaşmıştır.



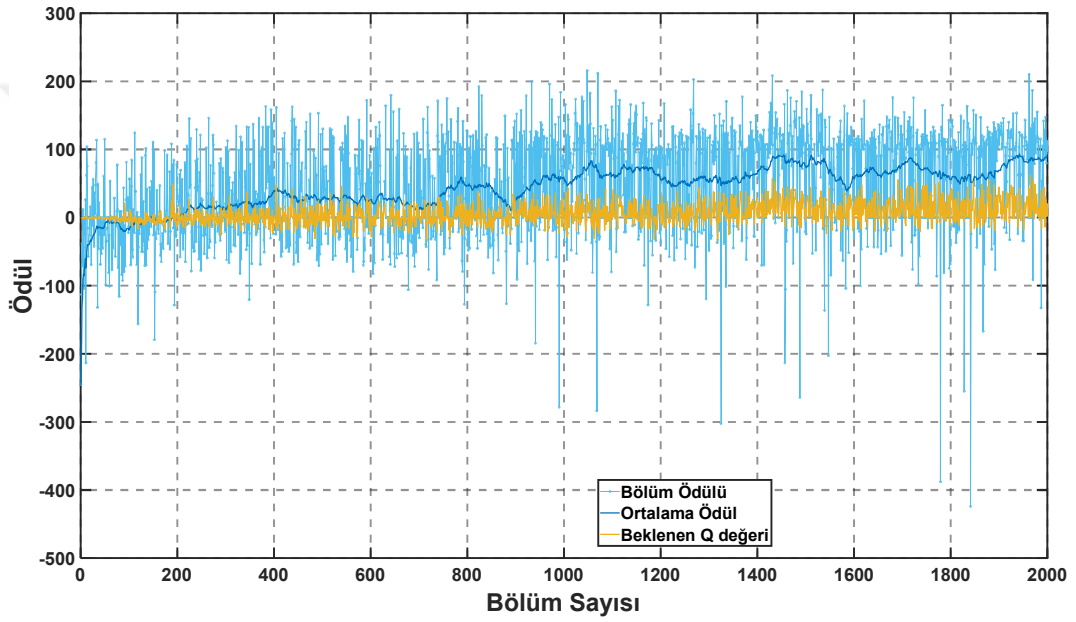
Şekil 49. Deney 14, Azaltma faktörü $\gamma=0.996$

Azaltma faktörünün 0.996 gibi biraz daha düşük bir değere ayarlanması, ajanın anlık ödüllere verdiği önemi bir miktar artırmış olsa da yapılan hataların fazlalığı nedeniyle ajan yine de optimum politikayı öğrenememiştir. Bu durumda da Q değerlerinde anlamsız bir yükselme görülmüş ve eğitim süreci kararsız bir şekilde devam etmiştir. Grafikte ortalama ödül artış eğiliminde olsa da, bölüm bazında elde edilen ödüllerde aşırı uç değerler görülmüş; bu durum, öğrenme sürecindeki tutarsızlığın ve politikanın istikrarsız doğasının bir göstergesi olmuştur.

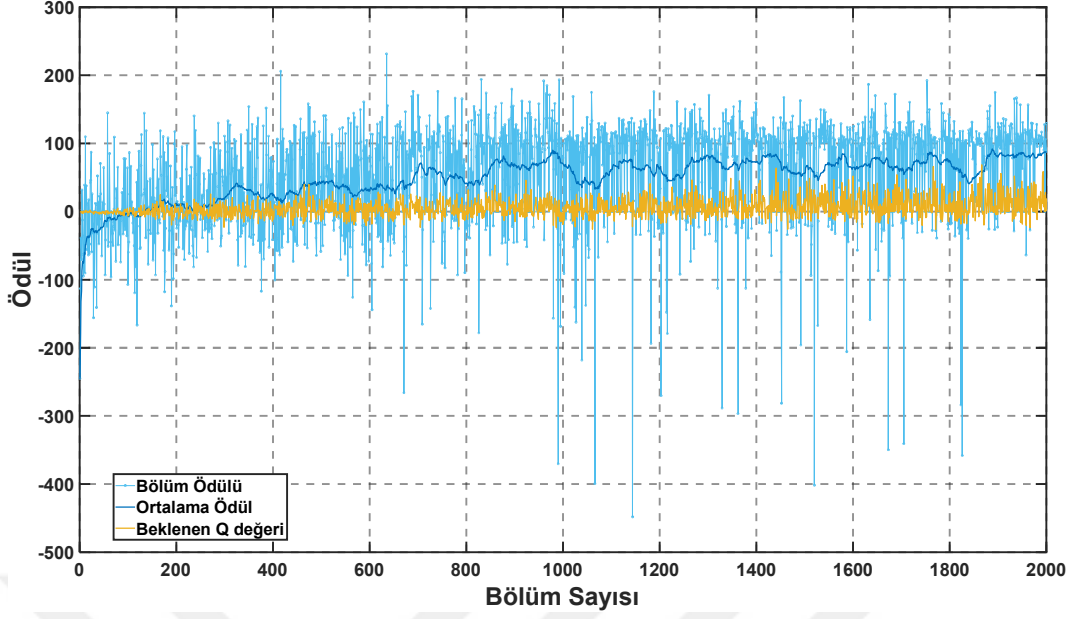


Şekil 50. Deney 15, Azaltma faktörü $\gamma=0.993$

Azaltma faktörünün 0.993 gibi daha küçük bir değer seçilmesi, ajanın gelecek ödülleri daha az dikkate almasını ve anlık hatalardan daha fazla kaçınmasını sağlamış, böylece önceki denemelere göre daha istikrarlı bir öğrenme süreci elde edilmiştir. Ancak, eğitimin erken bölümlerinde tahmin edilen Q değerlerinin gerçek ödül değerlerinden daha yüksek olması, ajanın başta çevreyi tam anlamıyla öğrenememesine ve geçici olarak iyimser tahminler üretmesine neden olmuştur. Bu durum, ilerleyen bölümlerde ajan çevreyi daha iyi öğrendikçe Q değerlerinin gerçek değerlere yakınsamaya başlamasıyla birlikte azalmıştır.



Şekil 51. Deney 16, Azaltma faktörü $\gamma=0.990$

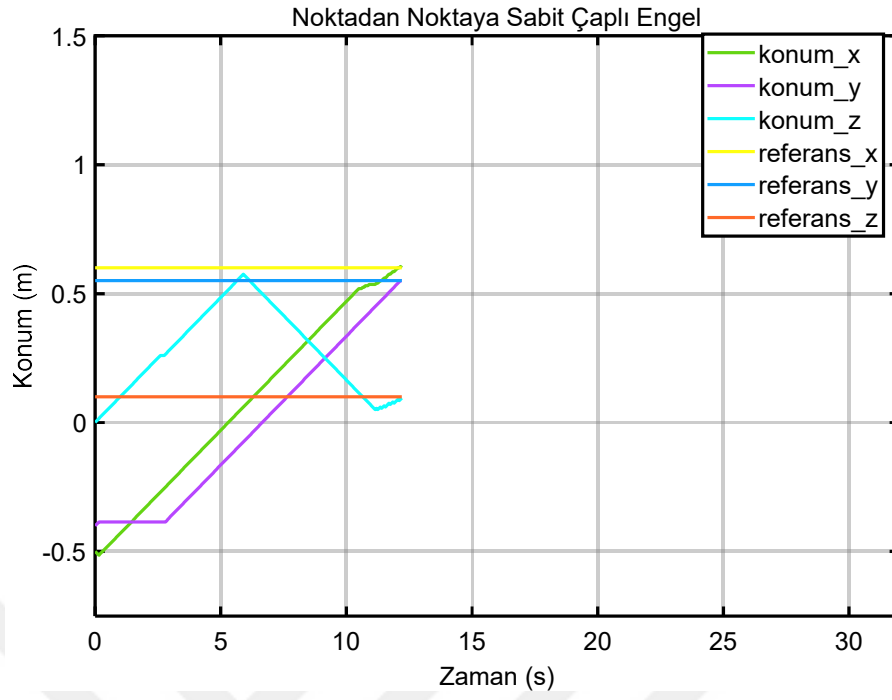


Şekil 52. Deney 17, Azaltma faktörü $\gamma=0.987$

Azaltma faktörünün 0.990 ve 0.987 gibi daha düşük değerler seçilmesi, hedef Q değerlerinin daha doğru ve hızlı bir şekilde yakınsamasını sağlamıştır. Bu durumda, ajan eğitimin erken evrelerinde ortalama ödül miktarını artırmış ve daha kısa sürede kararlı bir öğrenme sürecine ulaşmıştır.

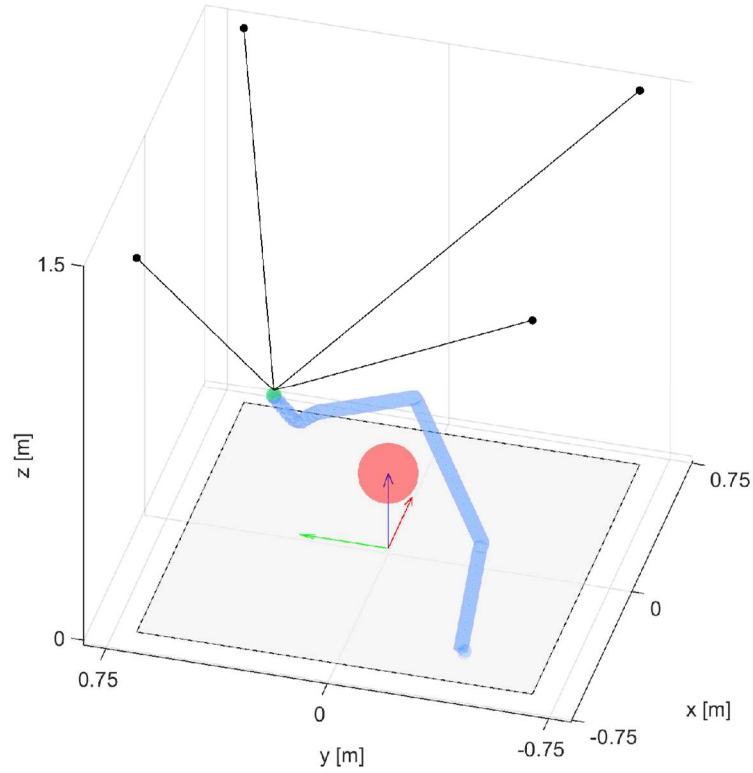
3.2.4. Hiperparametre Optimizasyonunun Başarımı

Yapılan hiperparametre optimizasyonu sonrasında, her parametre için elde edilen eğitim sonuçları incelenmiş ve en iyi sonucu veren parametre kombinasyonu öğrenme oranı $1e-4$, hedef ağ güncelleme sıklığı 5, ve azaltma faktörü 0.99 olarak tespit edilmiştir. Bu parametrelerle eğitilen ajan, eğitimde kullanılan başlangıç koşullarına benzer şekilde oluşturulan yaklaşık 100 adet rastgele test senaryosunun 86'sında başarılı olmuştur; yani, ajanın sabit çaplı engele çarpmadan referans konumuna ulaşabildiği senaryoların oranı %86'dır. Şekil 53'te belirlenen bir senaryo için başlangıç ve referans konumları arasında ajanın hareketleri grafiksel olarak gösterilmiş,



Şekil 53. Noktadan noktaya sabit çaplı engel benzetimi konum sonuçları

Şekil 54'te ise ajanın oluşturduğu yol üç boyutlu olacak şekilde görselleştirilmiştir.



Şekil 54. Noktadan noktaya sabit çaplı engel benzetiminin 3B görselleştirilmesi

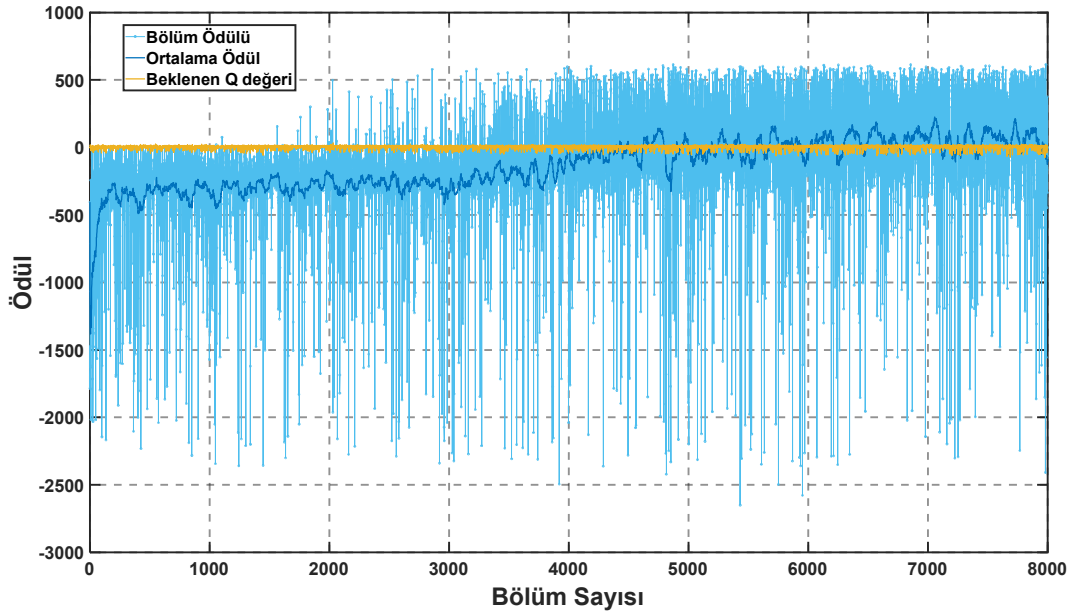
Kırmızı küre engeli, yeşil küre referans konumu, mavi çizgi ise yörüngeyi ifade etmektedir. Böylece gösterilen senaryoda ajan engelle temas olmadan başarılı bir şekilde referansa ulaşmıştır.

3.3. Değişken Çaplı Küresel Engel İçin Hiperparametre Optimizasyonun Eğitim Sonuçlarına Etkisi

Yapılan optimizasyon çalışmaları sonucunda, eğitim performansını maksimize eden hiperparametre kombinasyonu belirlenmiş ve bu optimum değerlerle eğitilen ajan, test ortamında değerlendirilmiştir. Test süreci, eğitimde olduğu gibi rastgele başlangıç koşullarına sahip yaklaşık 100 farklı senaryodan oluşmuştur. Elde edilen sonuçlar, hiperparametre seçiminin eğitim başarısı üzerinde belirleyici bir rol oynadığını açıkça göstermektedir.

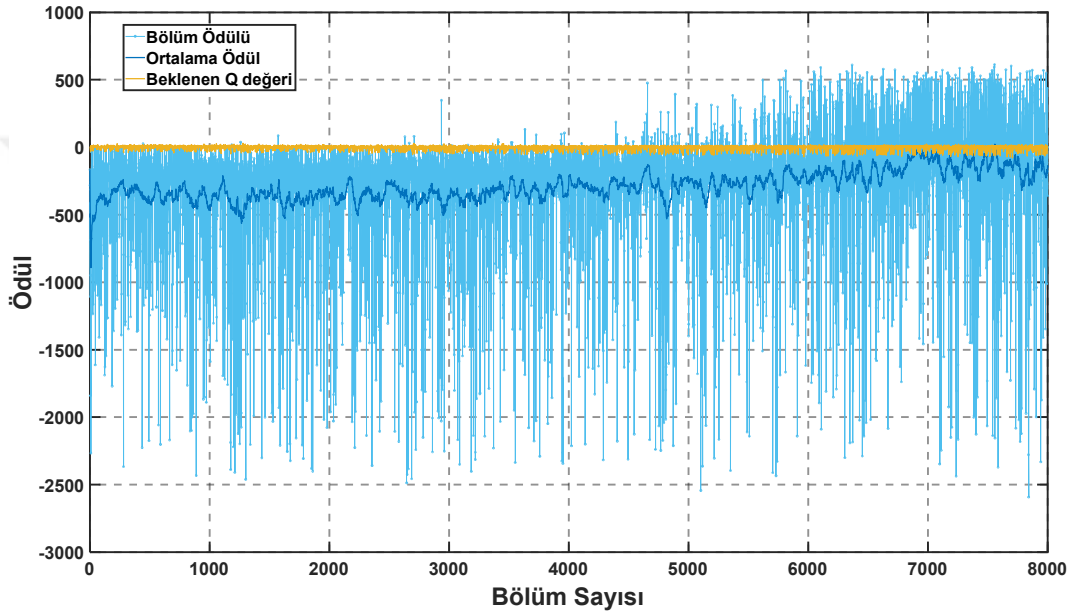
3.3.1. Öğrenme Oranı Optimizasyonu

Farklı öğrenme oranı ile yapılan 7 ile 10 numara arasındaki eğitimlerin sonuçları aşağıdaki şekillerde görüldüğü gibi gözlemlenmiştir.



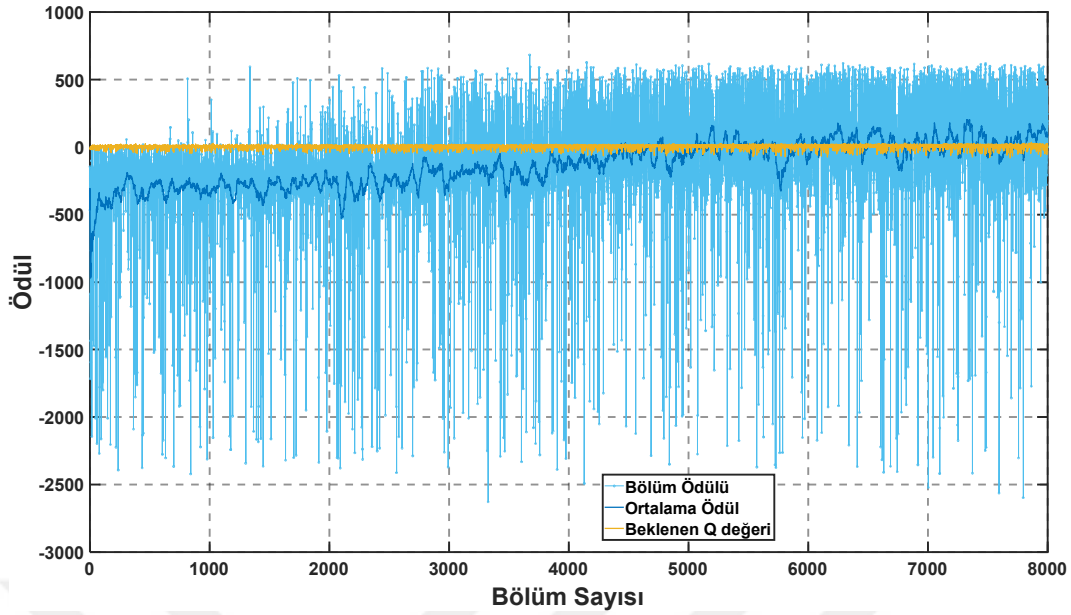
Şekil 55. Deney 7, Öğrenme oranı $1 * 10^{-3}$

Öğrenme oranı $1 * 10^{-3}$ olarak yapılan eğitimde Şekil 55'te görüldüğü gibi ilerleyen bölümler ortalama ödülde düzenli olarak artış görülmüştür. Fakat yapılan başarımlar testlerinde rastgele oluşturulan yaklaşık 100 adet başlangıç koşulunda sadece %44 oranla engele temas etmeden hedefe ulaşabilmiş, belirlenen yolların %12'sinde engele temas etmiş, %43'ünde ise hedefe ulaşamamıştır. Öğrenme oranının yüksek olması, ana ağ ağırlıklarının her güncellemede büyük değişikliklere uğramasına yol açmakta; bu durum, ajanın çevreyi genelleyememesine ve dolayısıyla düşük bir genel başarı elde etmesine neden olmaktadır.



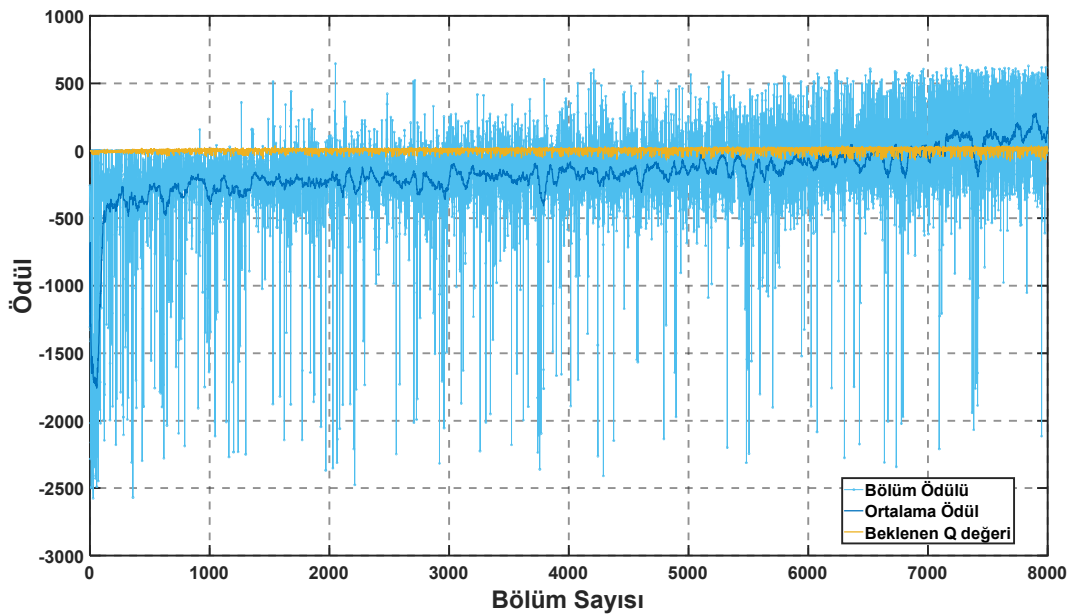
Şekil 56. Deney 8, Öğrenme Oranı $5 * 10^{-3}$

Öğrenme oranının $5 * 10^{-3}$ olarak seçildiği durumda ise genel eğitimin bir önceki parametreye göre daha yavaş ilerlediği ve ortalama ödülün ise sıfırın üzerine çıkamadığı, ayrıca çok yüksek miktarda çarpışma gerçekleştiği görülmektedir. Yapılan başarımlar testlerinde ise rastgele oluşturulan yaklaşık 100 adet başlangıç koşulunda %33 oranla engele temas etmeden hedefe ulaşabilmiş, %15 oranla engele çarpmış ve %51 oranla hedefe ulaşamamıştır. Bir önceki duruma benzer şekilde öğrenme oranının yüksek olması ajanın genelleme yeteneğinin kısıtlanmasına sebep olarak başarımları oldukça düşürmüştür.



Şekil 57. Deney 9, Öğrenme oranı $5 * 10^{-4}$

Öğrenme oranı $5 * 10^{-4}$ olarak seçildiği durumda eğitim sonuçlarına bakıldığında ortalama ödülün düzenli bir şekilde arttığı görülmekte. Bir önceki seçilen değere göre daha az çarpışma görülmektedir. Rastgele oluşturulan yaklaşık 100 adet başlangıç koşulunda yapılan başarımlar testlerinde oluşturulan yolların %59'unda engele temas olmadan hedefe ulaşmış, %6'sında engele temas etmiş, %34'ünde ise hiç ulaşamamıştır. Başarımlar oranlarına bakıldığında öğrenme oranı azaldıkça başarımın arttığı görülmektedir.

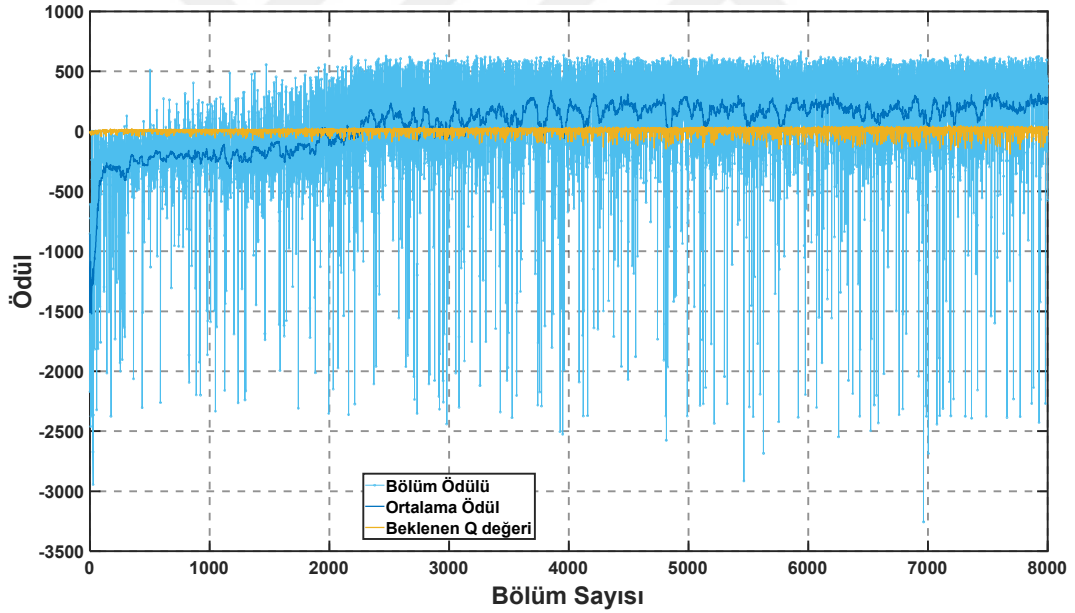


Şekil 58. Deney 10, Öğrenme oranı $1 * 10^{-5}$

Öğrenme oranının $1 * 10^{-5}$ olarak seçildiği durumda eğitim verilerine bakıldığında çok daha az çarpışma olduğu, öğrenmenin yavaş ve kararlı arttığı görülmektedir. Yapılan başarımlar testlerinde oluşturulan yolların %80'inde engele temas olmadan hedefe ulaşılmış, sadece %6'sında çarpışma gerçekleşmiş, %14'ünde ise hedefe ulaşılamamıştır. Öğrenme oranının düşük seçilmesi ana ağ parametrelerinin güncellenmesinin daha küçük oranla gerçekleşmesini sağlamaktadır. Böylece elde edilen deneyimle daha yavaş işlenerek ajanın genelleme yeteneği kazanması sağlanmıştır.

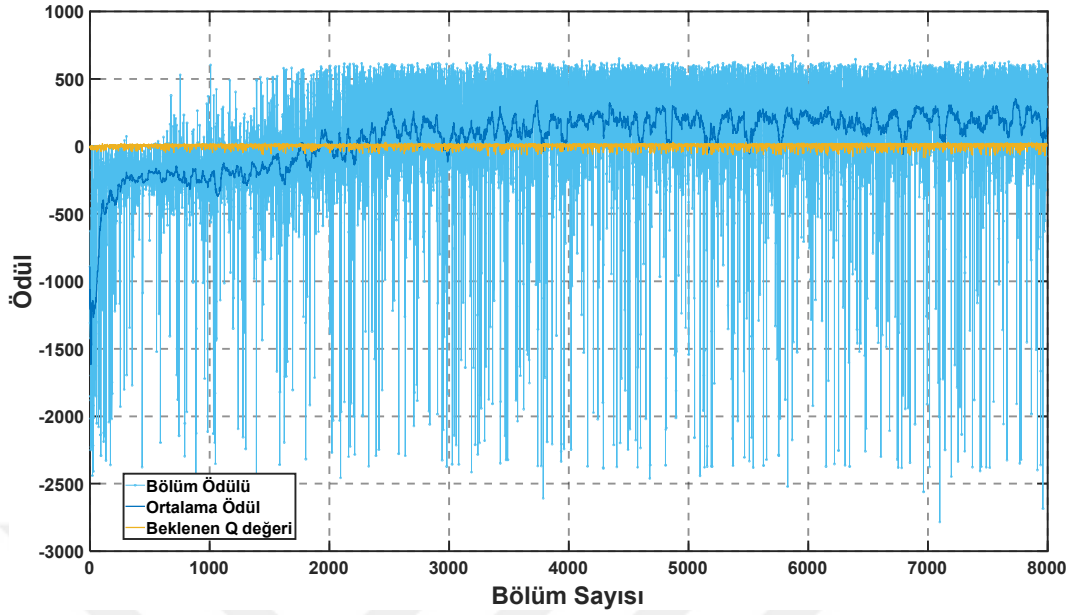
3.3.2. Hedef Ağ Güncelleme Sıklığı Optimizasyonu

Farklı hedef ağ güncelleme sıklığı parametreleri kullanılarak yapılan 11 ile 13 numara arasındaki eğitimlerin sonuçları aşağıdaki şekillerde görüldüğü gibi gözlemlenmiştir.



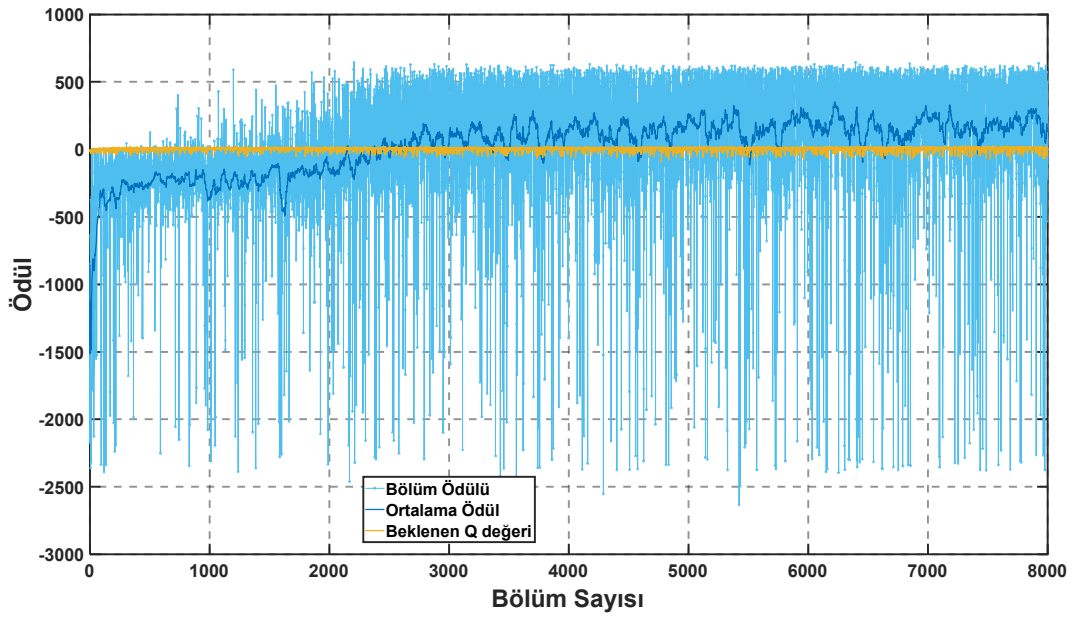
Şekil 59. Deney 11, Hedef ağın 320 adımda bir güncellenmesi

Bu çalışmada hedef ağ her 320 adımda bir güncellenmektedir. 320 adım yaklaşık olarak 1 bölüme karşılık gelmektedir. Hedef ağ güncellemesi yumuşatma faktörü kullanılmadan gerçekleştirilmiştir. Yani her 320 adımda bir defa ana ağın parametreleri tamamen hedef ağa kopyalanır. Rastgele oluşturulan başlangıç koşullarında yapılan başarımlar testlerinde oluşturulan yolların %70'inde engele temas etmeden hedefe ulaşılmış, %12'sinde temas gerçekleşmiş, %18'inde ise hedefe ulaşılamamıştır.



Şekil 60. Deney 12, Hedef ağın 960 adımda bir güncellenmesi

Hedef ağın 960 adımda bir güncellenmesi yaklaşık üç bölüme tekabül etmektedir. Rastgele oluşturulan başlangıç koşullarında yapılan başarımlar testlerinde oluşturulan yolların %56'sında engelle temas etmeden hedefe ulaşılmış, %14'ünde temas gerçekleşmiş, %30'unda ise hedefe ulaşamamıştır.

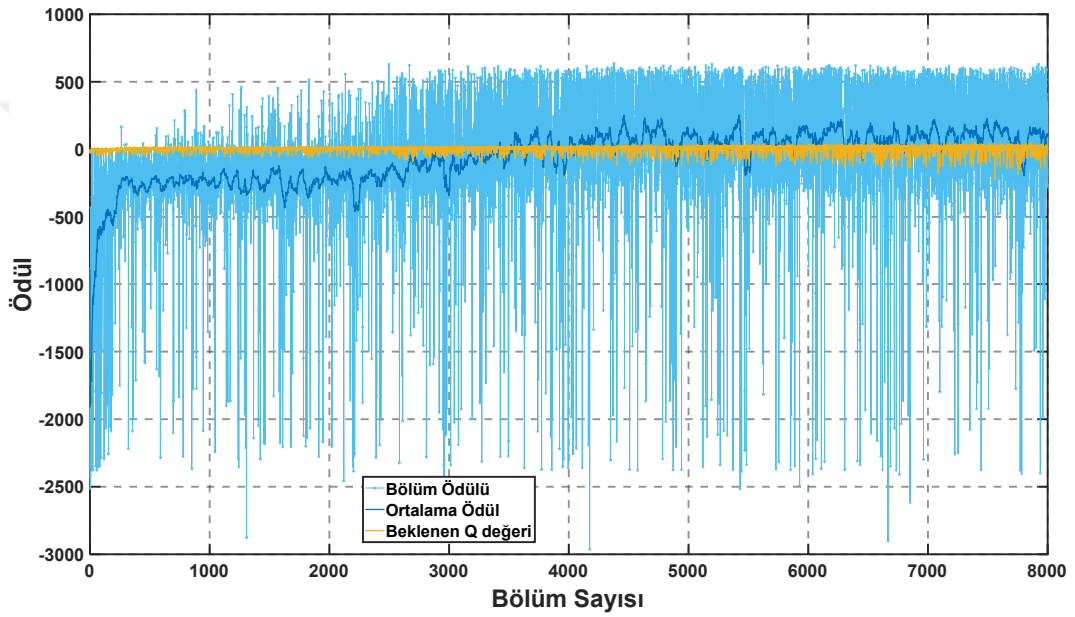


Şekil 61. Deney 13, Hedef ağın 1280 adımda bir güncellenmesi

Rastgele oluşturulan başlangıç koşullarında yapılan başarımlar testlerinde oluşturulan yolların %67'sinde engele temas etmeden hedefe ulaşılmış, %9'unda temas gerçekleşmiş, %24'ünde ise hedefe ulaşılamamıştır.

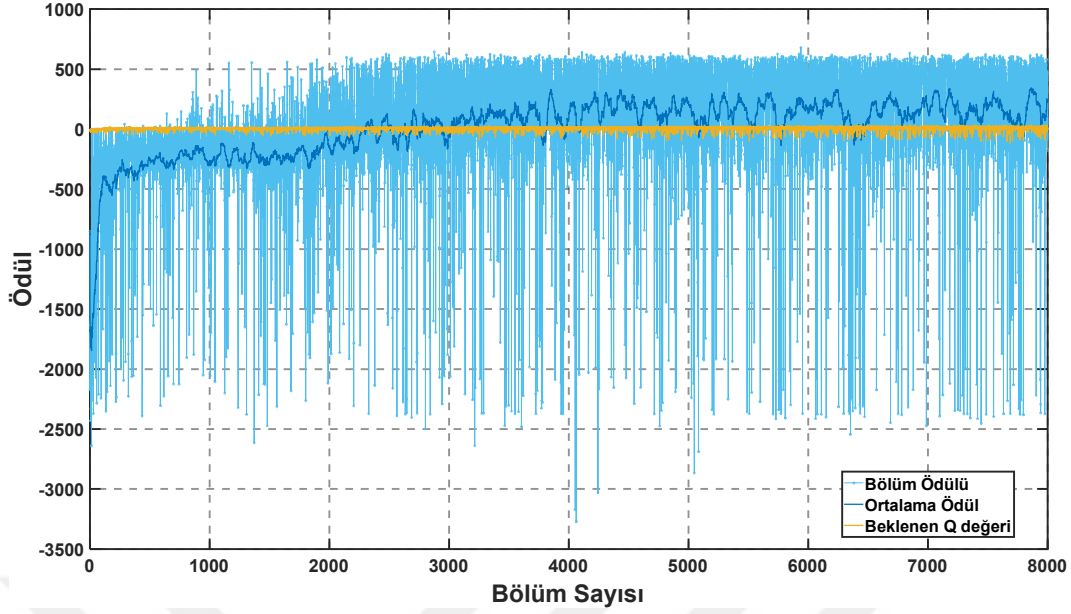
3.3.3. Azaltma Faktörü Optimizasyonu

Farklı azaltma faktörü ile yapılan 1 ile 6 numara arasındaki eğitimlerin sonuçları aşağıdaki şekillerde görüldüğü gibi gözlemlenmiştir



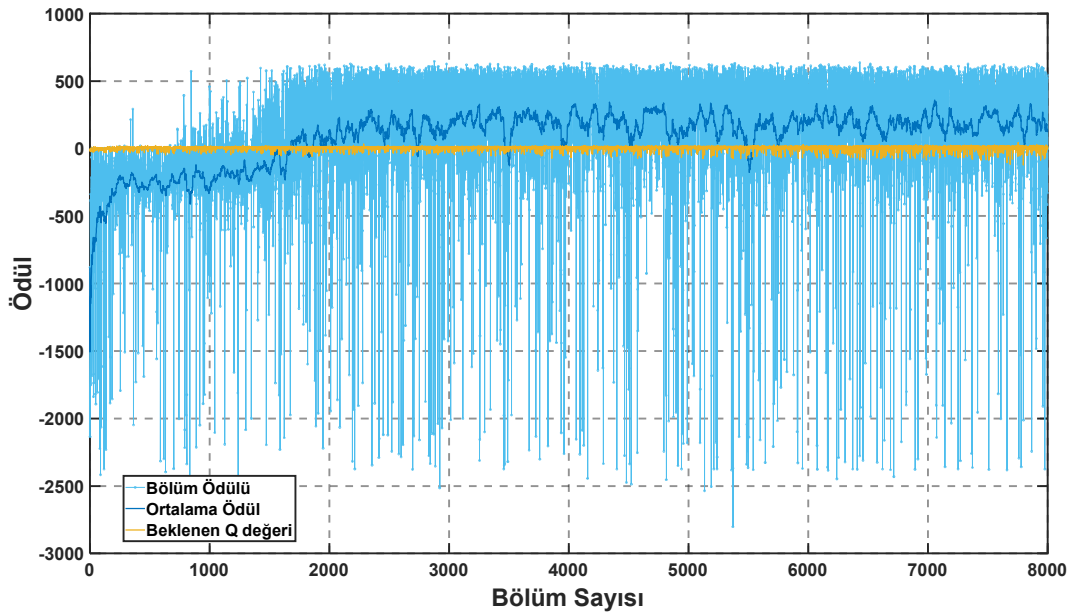
Şekil 62. Deney 1, Azaltma faktörü $\gamma=0.99$

Azaltma faktörünün 0.99 olarak seçildiği durumda gerçekleştirilen eğitimde, ortalama ödül değeri yaklaşık 4000 bölümden sonra sıfırın üzerine çıkmış; ancak, eğitim ilerledikçe tekrar sıfıra yakınsamıştır. Bu süreçte, tahmin edilen Q değerleri, yüksek azaltma faktörü nedeniyle gelecekteki ödüllere daha fazla ağırlık vermiş ve bu durum, ağırlık genellikle kötümser tahminler üretmesine yol açmıştır. Sonuç olarak, ana ağırlık çıktıları da aşağıya çekilmiş ve öğrenme süreci stabil hale gelememiştir. Rastgele başlangıç koşullarıyla yapılan başarımlar testlerinde ise, ajanın oluşturduğu yolların yalnızca %56'sında engele temas etmeden hedefe ulaşılabilmiş; %13'ünde engele temas gerçekleşmiş, %31'inde ise hedefe ulaşılamamıştır.



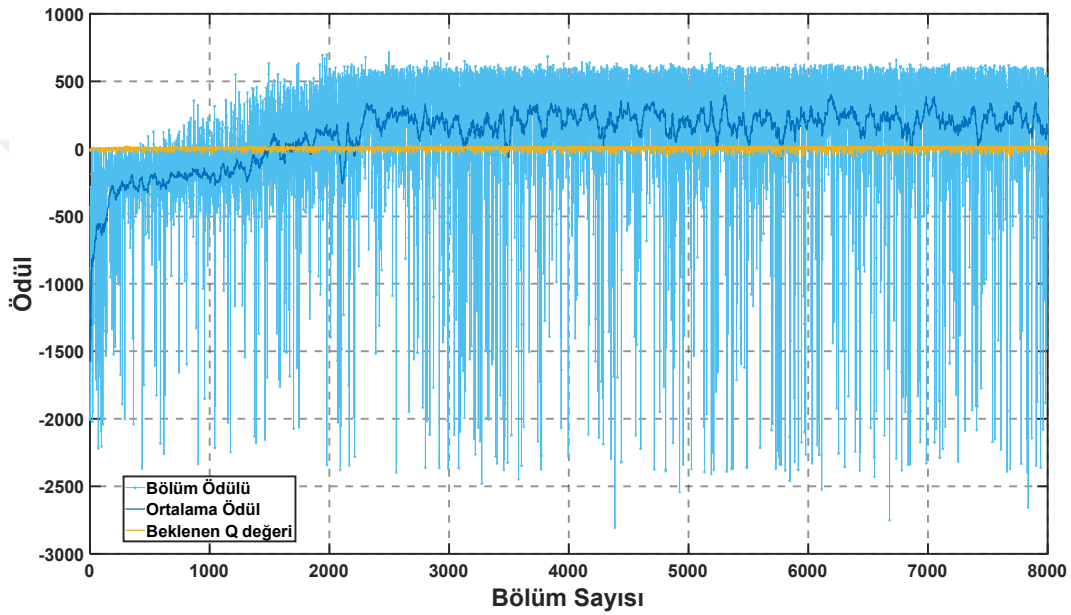
Şekil 63. Deney 2, Azaltma faktörü $\gamma=0.95$

Azaltma faktörünün bir miktar küçültülerek 0.95'e ayarlanması, ortalama ödül değerinin yaklaşık 3000. bölümden itibaren sıfırın üzerine çıkmasına olanak sağlamıştır. Ancak, eğitim ilerledikçe ortalama ödül tekrar sıfıra yakınsamış ve istikrarlı bir öğrenme elde edilememiştir. Rastgele başlangıç koşullarıyla yapılan başarımlar testlerinde, oluşturulan yolların %69'unda ajan engele temas etmeden hedefe ulaşabilmiş; %10'unda engele temas gerçekleşmiş, %21'inde ise hedefe ulaşılammıştır.



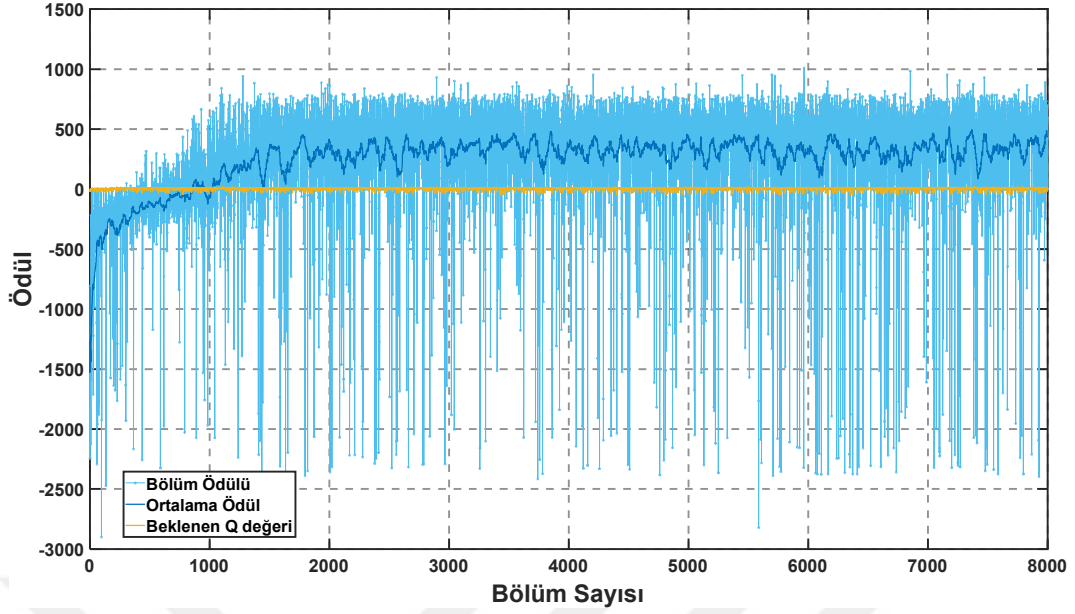
Şekil 64. Deney 3, Azaltma faktörü $\gamma=0.90$

Azaltma faktörünün 0.9 olarak seçildiği durumda, ortalama ödül değeri çok daha erken, yaklaşık 2000. bölümde sıfırın üzerine çıkabilmiş; ilerleyen bölümlerde tekrar sıfıra yakınsamalar görülse de bu iniş çıkışların sıklığı önceki denemelere göre daha seyrek. Rastgele başlangıç koşullarında yapılan başarımlar testlerinde, oluşturulan yolların %59'unda engele temas olmadan hedefe ulaşılmış, %6'sında engele temas gerçekleşmiş ve %35'inde ise hedefe ulaşamamıştır.



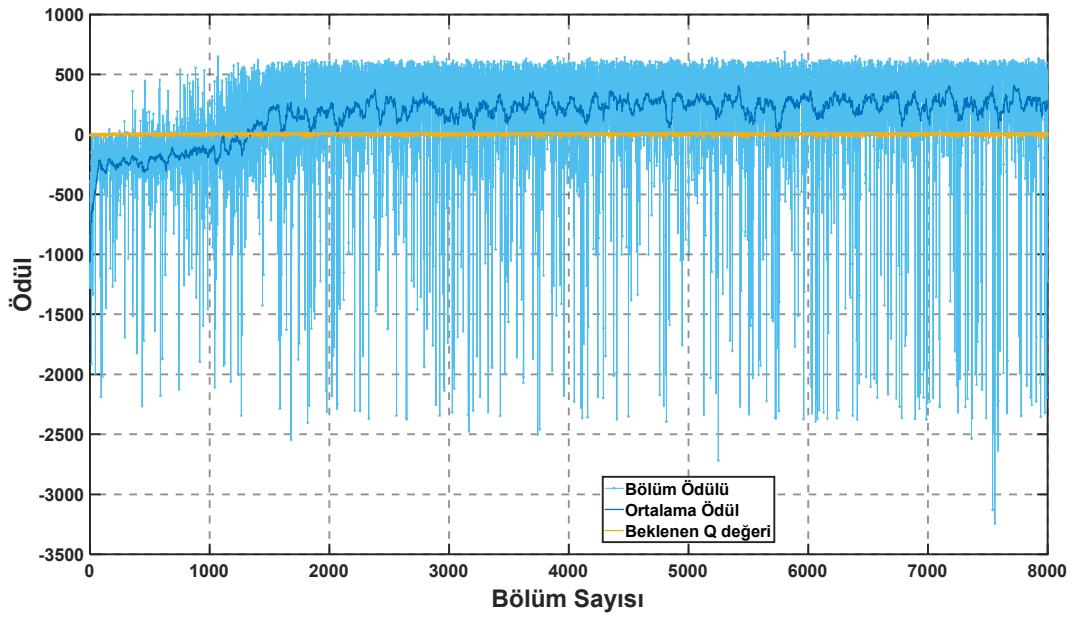
Şekil 65. Deney 4, Azaltma faktörü $\gamma=0.85$

Azaltma faktörünün 0.85 olarak seçildiği durumda, eğitimin erken aşamalarında ortalama ödül değeri sıfıra yakın seyretmiş; ilerleyen bölümlerde ise artış göstermiştir. Rastgele başlangıç koşulları altında yapılan başarımlar testlerinde, ajanın oluşturduğu yolların %74'ünde engele temas olmadan hedefe ulaşılmış, %7'sinde engele temas gerçekleşmiş ve %19'unda hedefe ulaşamamıştır.



Şekil 66. Deney 5, Azaltma faktörü $\gamma=0.80$

Azaltma faktörünün 0.80 olarak seçildiği durumda, ortalama ödül değeri yaklaşık 1000. bölümde sifıra yakınsamış; ancak sonraki bölümlerde neredeyse hiç sıfırın altına düşmeden istikrarlı bir şekilde artış göstermiştir. Rastgele başlangıç koşullarıyla yapılan başarımlar testlerinde ise, ajanın oluşturduğu yolların %79’unda engelle temas olmadan hedefe ulaşılmış, %8’inde engelle temas gerçekleşmiş ve %13’ünde ise hedefe ulaşamamıştır.



Şekil 67. Deney 6, Azaltma faktörü $\gamma=0.75$

Azaltma faktörünün 0.75 olarak seçildiği durumda ise, ortalama ödül değeri yine yaklaşık 1000. bölümde sıfırın üzerine çıkmış; ancak sonraki bölümlerde ödül eğrisinde tekrar iniş çıkışlar gözlemlenmiş ve zaman zaman sıfıra yakınsamalar meydana gelmiştir. Yapılan başarımlar testlerinde, ajanın %76 oranında engele temas etmeden hedefe ulaştığı, %10 oranında engele temas ettiği ve %14 oranında ise hedefe ulaşamadığı belirlenmiştir.

Yapılan kapsamlı hiperparametre optimizasyonu sonucunda, farklı azaltma faktörü değerlerinin PÖ ajanının hem eğitim süreci hem de başarımı üzerinde belirgin ve doğrudan etkileri olduğu gözlemlenmiştir.

Yüksek γ değerleri ajanın uzun vadeli ödülleri daha fazla dikkate almasına yol açmış, ancak bu durum eğitim eğrisinde sıklıkla iniş çıkışlar ve ortalama ödülün tekrar sıfıra yakınsaması ile sonuçlanmıştır. Bu değerlerde ortalama ödülün yükselmesi daha geç gerçekleşmiş ve başarı oranları düşük kalmıştır. Özellikle $\gamma=0.99$ ve $\gamma=0.95$ için hem Q değerlerinde kötümser tahminler hem de öğrenme stabilitesinde eksiklikler gözlemlenmiştir. Bunun sonucu olarak, testlerde başarılı yol planlama oranları sırasıyla %56 ve %69'da kalmıştır. Azaltma faktörünün orta seviyelere indirilmesiyle birlikte, ajanın anlık ödülleri daha fazla dikkate almaya başladığı ve hatalardan daha hızlı ders çıkarabildiği görülmüştür. Özellikle $\gamma=0.80$ ve $\gamma=0.85$ değerlerinde, ortalama ödül daha erken yükselmiş, eğitim daha istikrarlı ilerlemiş ve başarımlar oranlarında da ciddi bir artış kaydedilmiştir. Bu değerlerde ajanın engele temas etmeden hedefe ulaşma oranı %74–79 aralığına çıkmıştır. $\gamma=0.75$ gibi daha düşük azaltma faktörlerinde ise, eğitim eğrisinin başında ödül hızlıca yükselmiş fakat sonraki bölümlerde tekrar iniş çıkışlar ve sıfıra yakınsamalar meydana gelmiştir. Bu durum, ajanın kısa vadeli ödüllere gereğinden fazla odaklanarak bazı senaryolarda optimum olmayan politikalar geliştirmesine yol açmıştır. Buna rağmen, başarı oranı yine de makul seviyelerde seyretmiş, ancak eğitim sürecinin kararlılığı bir miktar azalmıştır.

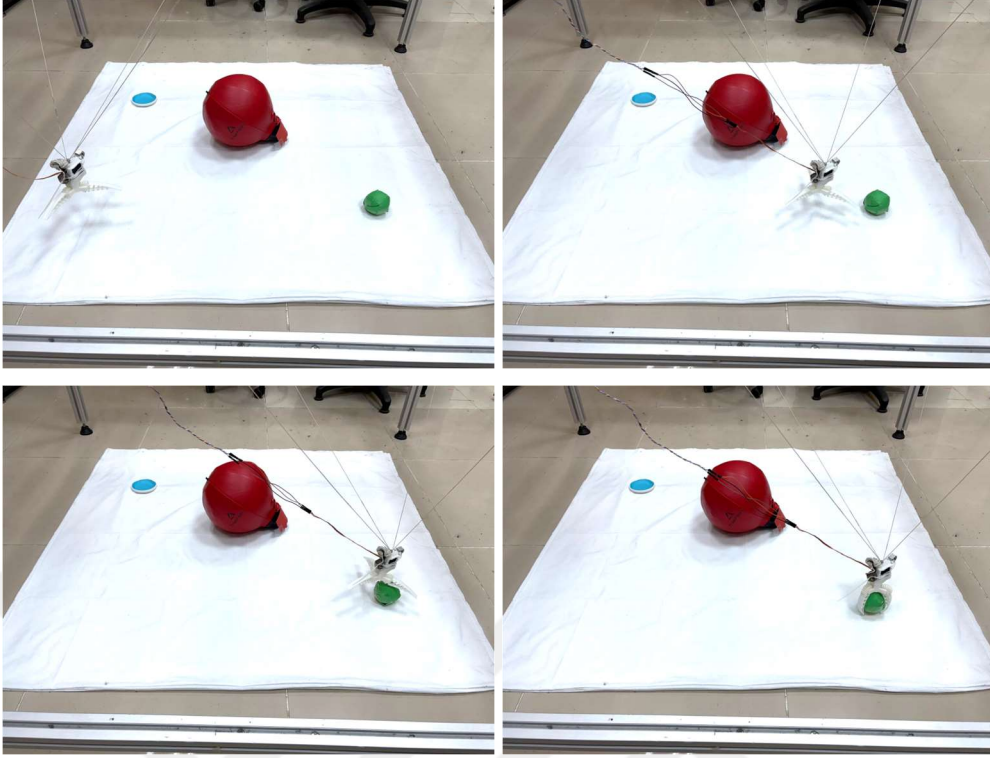
Genel olarak değerlendirildiğinde, azaltma faktörü için çok yüksek değerlerin eğitimde istikrarsızlık ve düşük başarımla, çok düşük değerlerin ise kısa vadeli ödüllere aşırı duyarlılığa neden olduğu görülmektedir. Optimum başarı ve istikrar, bu çalışmada $\gamma=0.80$ değeriyle elde edilmiştir. Elde edilen sonuçlarda, düşük azaltma faktörlerinin daha iyi performans göstermesinin temel sebebi doğrudan ödül fonksiyonunun yapısıyla ilişkilidir. Her adımda üretilen sabit miktardaki ödüller nedeniyle, ajan anlık ödüllere daha fazla odaklandığında, doğru yola ulaşması ve engelden kaçınması daha kolay olmaktadır. Buna karşılık, ajan çok uzun vadeli ödüllere öncelik verdiğinde ise yakın çevresindeki riskleri göz ardı ederek engele daha fazla temas etmiş ve başarımlar oranı düşmüştür. Bu nedenle, optimum

bir azaltma faktörü seçilerek, ajanın hem kısa vadeli riskleri gözetmesi hem de uzun vadeli hedefleri dikkate alması, yani bir denge kurması gerekmektedir. Böylece, ajan gereksiz risk almadan hem güvenli hem de etkili bir yol planlaması gerçekleştirebilir.

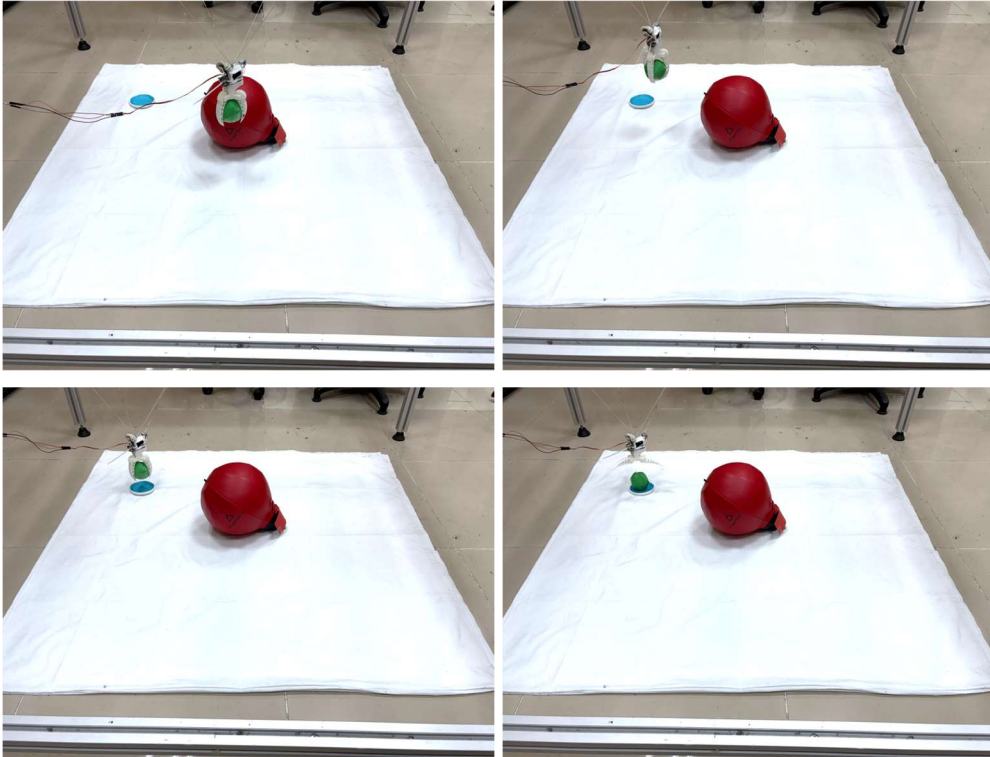
3.4. Tut-Yerleştir Uygulaması Bulguları

Benzetim ortamında eğitilen PÖ ajanı, 2.6 bölümünde ayrıntılı olarak açıklanan yöntem izlenerek deney düzeneğine entegre edilmiş ve önerilen kontrol döngüsü uygulanmıştır. Gerçekleştirilen deneylerde, kurulan sistemin hem pozisyon hem de engel çapı ölçümlerini doğru şekilde tespit edebildiği ve kontrol döngüsünün sorunsuz şekilde çalıştığı gözlemlenmiştir. Deneylerde taşınacak obje olarak 65 mm çapında bir tenis topu, engel olarak ise 200 mm çapında bir küresel cisim kullanılmıştır. Referans konum, deney düzeneğinde yüzeyde tanımlanan mavi renkli alan olarak belirlenmiştir.

Şekil 68 ve Şekil 69’da görüldüğü gibi elde edilen bulgular, geliştirilen ajanın uç işlemciyi başarıyla yeşil topunun bulunduğu konuma yönlendirdiğini ve topa ulaşıldığında beklemiş ve kavrama işlemini gerçekleştirmiş. Sonrasında uç işlemci engele çarpmadan referans konuma doğru hareket etmiş ve burada topun bırakılması için sabit şekilde beklemiş ve topu bırakmıştır.



Şekil 68. Uç işlemcinin yeşil topa yönelmesi ve tutması



Şekil 69. Uç işlemcinin engele çarpmadan topu referansa götürmesi ve bırakması

Tüm bu işlemler sırasında uç işlemci ve kablolar, herhangi bir şekilde engele temas etmeden güvenli bir biçimde hareket etmiş ve görev planlandığı şekilde tamamlanmıştır.



4. SONUÇLAR

Bu çalışmanın temel amacı, eksik sınırlandırılmış bir KSPR sisteminin, çalışma ortamı içerisinde engellere temas etmeden, PÖ tabanlı yöntemlerle gerçek zamanlı yörünge planlamasının gerçekleştirilmesidir. Bu hedef doğrultusunda hem benzetim ortamında hem de gerçek sistem üzerinde kapsamlı çalışmalar yürütülmüştür.

İlk aşamada deney sistemi fiziksel olarak kurulmuş, mekanik ve elektriksel bağlantılar hassas bir şekilde gerçekleştirilmiştir. Kinematik hataları önlemek amacıyla bağlantılar dikkatle ayarlanmış ve sistemin kinematik modeli oluşturulmuştur. Bu model kullanılarak PÖ tabanlı benzetim ortamı hazırlanmış, gerçek koşulları temsil edecek şekilde rastgele başlangıç noktaları ile eğitim senaryoları tasarlanmıştır.

Eğitim süreci, basitten karmaşığa aşamalı olarak ilerlemiştir. Önce engelsiz ortamda çalışmalar yürütülmüş, ardından sabit çaplı engel bulunan ortamda ve son olarak değişken çaplı engellerin bulunduğu ortamda eğitimler gerçekleştirilmiştir. Bu süreçte farklı gözlem kümeleri ve ödül fonksiyonları denenerek, sistem dinamiği ve ajanın öğrenme davranışları gözlemlenmiş ve analiz edilmiştir. Değişken çaplı engel durum, sabit çaplı senaryoya kıyasla daha karmaşık bir problem oluşturmuş, bu nedenle başlangıçta başarı oranı belirgin şekilde düşük kalmıştır.

Bu noktada, hiperparametre optimizasyonunun performansı artırıp artırmayacağı araştırılmış ve hem sabit hem de değişken çaplı engel senaryolarında testler yapılmıştır. Elde edilen sonuçlar, hiperparametre seçiminin doğrudan başarı oranı üzerinde etkili olduğunu açıkça ortaya koymuştur. Engelsiz ortamda noktadan noktaya yapılan testlerde yaklaşık %98 başarı oranı elde edilirken, sabit çaplı engel senaryosunda bu oran başlangıçta yaklaşık %80 iken optimizasyon sonucu %86'ya kadar yükselmiştir. Değişken çaplı engellerin bulunduğu durumda ise hem küre çapındaki değişkenlik hem de rastgele başlangıç koşulları nedeniyle başarı oranı %70 seviyelerine kadar gerilemiştir. Ancak hiperparametre optimizasyonu ile eğitim kararlılığı artmış ve başarı oranı %80'e kadar yükselmiştir.

Benzetim ortamında tamamlanan PÖ eğitimleri, gerçek KSPR sistemine entegre edilmiştir. Ortam verileri, geliştirilen görüntü işleme yöntemleri ile kamera üzerinden algılanmış; küresel engellerin ve uç işlemci konumunun belirlenmesine yönelik görüntü işleme algoritmaları geliştirilmiştir. Bu sayede sistemin gerçek koşullara uyumu artırılmıştır. Entegrasyonun ardından sistem, tut-yerleştir uygulaması ile test edilmiş, bu amaçla özel bir

kontrol döngüsü tasarlanmıştır. Sonuç olarak, DQA tabanlı bu yörünge planlama yaklaşımı, yüksek zorluk derecesine rağmen başarılı bir şekilde uygulanmış ve deneysel ortamda doğrulanmıştır. Eğitilen PÖ ajanının yalnızca yapay sinir ağı matrislerinden oluşması sayesinde farklı platformlara kolayca entegre edilebilmesi ve işlem gücü açısından yüksek verimlilik sağlaması da önemli bir avantaj olarak öne çıkmıştır.

Literatürdeki benzer çalışmalarla karşılaştırıldığında, geliştirilen yöntemin yörünge planlama esnasında düşük kaynak kullanımı, farklı donanım ve platformlara hızlı adaptasyonu ve gerçek zamanlı çalışmaya elverişli yüksek işlem hızı gibi özellikleri, endüstriyel ve otonom uygulamalarda pratik bir çözüm potansiyeline sahip olduğunu açıkça ortaya koymaktadır.



5. ÖNERİLER

Bu çalışmada, dört kabloyla sürülen paralel robotun pekiştirmeli öğrenme ile yol planlaması ve otonom tut–yerleştir görevlerinde hem benzetim hem de deney ortamlarında başarılı bir şekilde kullanılabileceği gösterilmiştir. Ancak, elde edilen bulgular ışığında, sistem performansını artırmaya ve yöntemi daha genellenebilir hale getirmeye yönelik aşağıdaki öneriler sunulmaktadır.

Eğitimlerde başlangıç konumlarının rastgele belirlenmesi, farklı senaryolar arasında ödül fonksiyonunun etkisini doğrudan karşılaştırmayı zorlaştırmaktadır. Bu nedenle, her eğitim oturumunun başında ulaşılabilecek maksimum ödülün belirlenmesi ve elde edilen anlık ödüllerin uygun bir katsayı ile ölçeklendirilmesi önerilmektedir. Bu yaklaşım, ödül sinyali her bölümde daha tutarlı hale getirerek öğrenme sürecinin kararlılığını artırabilir.

Gözlem uzayındaki değişkenlerin ölçek farklılıklarının giderilmesi, ajanın durum–eylem ilişkisini daha tutarlı öğrenmesine olanak tanır. Tüm gözlem bileşenlerinin uygun şekilde normalize edilmesi, öğrenme hızını artırabilir ve eğitimdeki iniş çıkışları azaltabilir. Mevcut çalışmada kullanılan gözlem–ödül yapılarına ek olarak, farklı kombinasyonların test edilmesi, eğitim sürecinin daha stabil olmasına ve başarı oranının yükselmesine katkı sağlayabilir.

Gelecekteki PÖ eğitimlerinde yalnızca engelden kaçınma değil, aynı zamanda enerji tüketiminin ve kablo gerginliklerinin minimize edilmesi hedeflenebilir. Bu şekilde hem sistem ömrü uzatılabilir hem de yörünge planlama daha verimli hale getirilebilir. Uç işlemci konumunun yalnızca kinematik denklemlerden değil, aynı zamanda görsel tabanlı yöntemlerden de doğrulanması, konum hassasiyetini artırabilir. Böylece kinematik modelleme hataları azaltılarak daha güvenilir hareket planlaması yapılabilir.

Sistemin gerçek dünya uygulamalarına uyumunu artırmak amacıyla, test senaryolarında tek bir engel yerine farklı boyut ve şekillere sahip birden fazla engel kullanılabilir. Bu engeller, yalnızca sabit konumlarda değil, belirli hız ve yörüngelerle hareket eden dinamik engeller olarak da modellenebilir. Böyle bir yaklaşım, robotun hem statik hem de dinamik engeller karşısındaki karar verme ve yol planlama yeteneklerini geliştirir. Ayrıca, hareketli engellerin öngörülemeyen davranışlar sergilediği senaryolar tasarlanarak, ajanın anlık karar verme kabiliyeti, adaptasyon hızı ve çarpışma önleme başarımı ölçülebilir. Bu tür karmaşık ve değişken ortamlarda yapılan eğitimler, sistemin

otonom araçlar, endüstriyel robotlar veya arama-kurtarma robotları gibi gerçek dünya uygulamalarına daha yüksek doğruluk ve güvenilirlikle uyum sağlamasını mümkün kılabilir.

Sistemin yalnızca konum tabanlı kontrol yerine doğrudan kontrol tabanlı bir yaklaşımla modellenmesi hem performans hem de farklı senaryolara genellenebilirlik açısından önemli avantajlar sağlayabilir. Bu tür bir yaklaşımda, sensör girdilerinden doğrudan motor komutlarını üreten bir PÖ mimarisi kurgulanarak, robotun dinamik olarak kontrol edilmesi mümkün olur. Böylece yüksek hızlara ulaşmak, titreşim ve salınımları minimize etmek, farklı ağırlık ve boyutlardaki nesneler için özel kontrol stratejileri geliştirmek mümkün hale gelir.

Geliştirilebilecek yöntemler, iyileştirme çalışmaları ve uygulanabilecek senaryolar açısından herhangi bir sınır bulunmamakta olup sistem mimarisi, kullanılan algoritmalar ve test koşulları, hedeflenen amaçlara ve uygulama gereksinimlerine göre uyarlanabilir, genişletilebilir ve daha ileri düzeyde geliştirilebilir. Bu çalışmada elde edilen bulgular ve oluşturulan deney altyapısı, gelecekte gerçekleştirilecek araştırmalar için sağlam bir temel oluşturmaktadır. Böylelikle literatüre anlamlı bir katkı sunulmuş, bundan sonraki çalışmalar için hem yöntemsel hem de donanımsal açıdan kullanılacak bir araştırma zemini hazırlanmıştır. Ayrıca kurulan deney düzeneği sayesinde, ilerleyen süreçte gerçek dünya senaryolarında kapsamlı ve güvenilir testlerin gerçekleştirilmesi de mümkün hale gelmiştir.

6. KAYNAKLAR

- Alp, A. B., & Agrawall, S. K. (2002). Cable Suspend Robots: Feedback controllers with positive inputs. *Proceedings of the 2002 American Control Conference* (s. 815-820). Anchorage, Alaska, USA: IEEE.
- Bak, J.-H., Sung, W. H., Jonghyun, Y., Jong, H. P., & Jong-Oh, P. (2019). Collision-free path planning of cable-driven parallel robots in cluttered environments. *Intelligent Service Robotics*, 243-253.
- Barnett, E., & Gosselin, C. (2015). Large-scale 3D printing with a cable-suspended robot. *Additive Manufacturing*, 27-44.
- Choset, H., Lynch, K. M., Hutchinson, S., Kantor, G. A., & Burgard, W. (2005). *Principles of robot motion: theory, algorithms, and implementations*. MIT press.
- Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numer. Math. I*, 269-271.
- Duan, B. Y. (1999). A new design project of the line feed structure for large spherical radio telescope and its nonlinear dynamic analysis. *Mechatronics*, 53-64.
- François-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G., & Pineau, J. (2018). *An introduction to deep reinforcement learning* (Cilt 11). Foundations and Trends® in Machine Learning.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
- Grimshaw, A., & Oyekan, J. (2021). Applying deep reinforcement learning to cable driven parallel robots for balancing unstable loads: A ball case study. *Frontiers in Robotics and AI*, 611203.
- Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 100-107.
- James, A., Roger, B., & Nicholas, D. (1993). The NIST robocrane. *Journal of Robotic Systems*, 709-724.
- Karur, K., Sharma, N., Dharmatti, C., & Siegel, J. E. (2021). A survey of path planning algorithms for mobile robots. *Vehicles*, 448-468.
- Korayem, M. H., Zehfroosh, A., Tourajizadeh, H., & Manteghi, S. (2014). Optimal motion planning of non-linear dynamic systems in the presence of obstacles and moving boundaries using SDRE: application on cable-suspended robot. *Nonlinear Dynamics*, 1423-1441.

- Krishnaswamy, G., & Stentz, A. (1995). *Resolution Independent Grid-Based Path Planning* (No. CMURITR9508).
- Kurfess, T. (2005). *Robotics and automation handbook* (Vol. 414). Boca Raton, FL: CRC press.
- Lahouar, S., Ottaviano, E., Zeghoul, S., Romdhane, L., & Ceccarelli, M. (2009). Collision free path-planning for cable-driven parallel robots. *Robotics and Autonomous Systems*, 1083-1093.
- Laribi, M. A., Romdhane, L., & Zeghloul, S. (2007). Analysis and dimensional synthesis of the DELTA robot for a prescribed workspace. *Mechanism and machine theory*, 42(7), 859-870.
- LaValle, S. M., & Kuffner Jr, J. J. (2001). LaValle, S. M., & Kuffner Jr, J. J. (2001). Randomized kinodynamic planning. *the international journal of robotics research*, 378-400.
- Lecun, Y., Bengio, Y., & Hinton, G. (2016). Deep Learning. *Nature*, 521(7553), 436-444.
- Liu, Y., Cao, Z., Xiong, H., Du, J., Cao, H., & Zhang, L. (2023). Dynamic Obstacle Avoidance for Cable-Driven Parallel Robots With Mobile Bases via Sim-to-Real Reinforcement Learning. *Robotics and Automation Letters*, 1683-1690.
- Mao, Y., Jin, X., Dutta, G. G., Scholz, J. P., & Agrawal, S. K. (2014). Human movement training with a cable driven arm exoskeleton (CAREX). *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 84-92.
- Merlet, J. P. (2005). *Parallel robots* (Cilt 128). Springer Science & Business Media.
- Miermeister, P., Lachele, M., & Boss, R. (2016). The CableRobot simulator large scale motion platform based on Cable Robot technology. *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems* (s. 3024-3029). Daejeon: 2016.
- Mishra, U. A., Metillon, M., & Caro, S. (2021). Kinematic Stability based AFG-RRT* Path Planning for Cable-Driven Parallel Robots. *International Conference on Robotics and Automation (ICRA)* (s. 6369-6969). IEEE.
- Nierenberger, M., Poncelet, M., Pattofatto, S., Hamouche, A., Raka, B., & Virely, J. M. (2014). Multiaxial testing of materials using a stewart platform: Case study of the nooru-mohamed test. *Experimental Techniques*, 38(2), 74-83.
- Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint*, arXiv:1811.03378.
- Pinto, A. M., Moreira, E., Lima, J., Sousa, J. P., & Costa, P. (2017). A cable-driven robot for architectural constructions: a visual-guided approach for motion control and path-planning. *Autonomous Robots*, 1487-1499.
- Plaat, A. (2022). *Deep reinforcement learning*. Singapore: Springer.

- Qian, S., Zi, B., Shang, W. W., & Xu, Q. S. (2018). A review on cable-driven parallel robots. *Chinese Journal of Mechanical Engineering*, 31(1), 1-11.
- Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), 533-536.
- Sancak, C., Yamac, F., & Itik, M. (2022). Position control of a planar cable-driven parallel robot using reinforcement learning. *Robotica*, 40(10), 3378-3395.
- Stentz, A. (1994). Optimal and efficient path planning for partially-known environments. *In Proceedings of the 1994 IEEE international conference on robotics and automation* (s. 3310-3317). IEEE.
- Tanaka, M., Seguchi, Y., & Shimada, S. (1988). Kineto-statics of skycam-type wire transport system. *Proceedings of USA-Japan symposium on flexible automation, Crossing bridges: advances in flexible automation and robotics*.
- Tsai, L. W. (1999). *Robot analysis: the mechanics of serial and parallel manipulators*. John Wiley & Sons.
- URL-1. (2025, Haziran 1). Lufthansa Aviation Training. <https://www.lufthansa-aviation-training.com/dry-training> adresinden alınmıştır.
- URL-2. (2025, Haziran 1). Motion Systems. <https://motionsystems.eu/portfolio/sim-rst-6dof-racing-simulator/> adresinden alınmıştır.
- URL-3. (2025, Haziran 1). Academia. https://www.academia.edu/22183942/Multiaxial_Testing_of_Materials_Using_a_Stewart_Platform_Case_Study_of_the_Nooru_Mohamed_Test adresinden alınmıştır.
- URL-4. (2025, Haziran 1). Motion Systems. <https://motionsystems.eu/portfolio/earthquake-simulator> adresinden alınmıştır.
- URL-5. (2025, Haziran 2). OMRON. https://assets.omron.com/m/4cadd908564c4a55/Landscape_L-Washdown_Delta_robot-jpg.jpg adresinden alınmıştır.
- URL-6. (2025, Haziran 2). OMRON. https://www.ia.omron.com/Images/3512_dm_113-257445.gif adresinden alınmıştır.
- URL-7. (2025, Haziran 29). AI ML – Artificial Intelligence and Machine Learning. <https://ai-artificial-intelligence.webyes.com.br/most-used-activation-functions-in-neural-networks/> adresinden alınmıştır.
- URL-8. (2025, Haziran 29). YouTube. KINEMATICS | Serial robot vs. Parallel robot (This is not CGI). <https://youtu.be/3fbmguBgVPA?si=ZQCwPwaOTkCVSv2G&t=16> adresinden alınmıştır.
- URL-9. (2025, Haziran 30). Learn to Navigate: Cooperative Path Planning for Unmanned Surface Vehicles Using Deep Reinforcement.

https://www.researchgate.net/figure/Schematic-of-Deep-Q-network_fig7_337231033 adresinden alınmıştır.

- URL-10. (2025, Haziran 30). Medium. https://miro.medium.com/v2/resize:fit:1100/format:webp/1*J-v2B6T9RKxdvwThtQ1NVg.png adresinden alınmıştır.
- URL-11. (2025, Ağustos 4). <https://static.cytron.io/image/cache/catalog/products/MG955-360/servo-mg955-800x800.jpg> adresinden alınmıştır.
- Warren, C. W. (1989, January). Global path planning using artificial potential fields. In *1989 IEEE International Conference on Robotics and Automation* (pp. 316-317). IEEE Computer Society.
- Xiong, H., Ma, T., Zhang, L., & Diao, X. (2020). Comparison of end-to-end and hybrid deep reinforcement learning strategies for controlling cable-driven parallel robots. *Neurocomputing*, (377), 73-84.
- Xu, J., & Park, K. S. (2021). Moving obstacle avoidance for cable-driven parallel robots using improved RRT. *Microsystem Technologies*, 27(6), 2281-2292.
- Yang, L., Qi, J., Song, D., Xiao, J., Han, J., & Xia, Y. (2016). Survey of robot 3D path planning algorithms. *Journal of Control Science and Engineering*(1).
- Zi, B., Lin, J., & Qian, S. (2015). Localization, obstacle avoidance planning and control of a cooperative cable parallel robot for multiple mobile cranes. *Robotics and Computer-Integrated Manufacturing*, 34, 105-123.

ÖZGEÇMİŞ

Kadir İbrahim Ertürk, 2020 yılında Avrasya Üniversitesi Makine Mühendisliği Bölümünden mezun olmuştur. Mezuniyetinin ardından yaklaşık bir yıl boyunca imalat sektöründe tasarım ve imalat sorumlusu olarak görev yapmıştır. Daha sonra, 2022 yılında Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü Makine Mühendisliği Bölümünde yüksek lisans eğitime başlamış, 2024 yılında ise aynı bölümün Makine Teorisi ve Dinamiği Anabilim Dalı'nda Araştırma Görevlisi olarak göreve başlamıştır ve görevine hâlen devam etmektedir. Çalışmalarını mekatronik, robotik sistemler ve sistem dinamiği-kontrol alanlarında sürdürmektedir.