KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

YILAN ROBOTUN ÖĞRENME TABANLI ADAPTİF HAREKET KONTROLÜ

DOKTORA TEZİ

Elektrik Yük. Müh. Yeşim Aysel BAYSAL

TEMMUZ 2022
TRABZON

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

YILAN ROBOTUN ÖĞRENME TABANLI ADAPTİF HAREKET KONTROLÜ

Elektrik Yük. Müh. Yeşim Aysel BAYSAL
ORCID : - - -

Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsünce
"DOKTOR (ELEKTRİK MÜHENDİSLİĞİ)"
Unvanı Verilmesi İçin Kabul Edilen Tezdir.

Tezin Enstitüye Verildiği Tarih : 04 / 07 /2022
Tezin Savunma Tarihi : 26 / 07 /2022

Tez Danışmanı : Prof. Dr. İsmail Hakkı ALTAŞ
ORCID : - - -

Trabzon 2022

ÖNSÖZ

Yılan robotlar esnek gövdeleri sayesinde bilinmeyen düzensiz ortamlara adaptif hareket kabiliyetleriyle öne çıkmaktadır. Buna karşın bu robotların artan serbestlik dereceleri bilinmeyen ortamlardaki kontrolünü zorlaştırmaktadır. Ayrıca sürtünme bağımlı hareketlerinden dolayı düşük hareket verimliliğine sahip olmaları, potansiyel uygulamaları için enerji verimli hareketlerini zorunlu kılmaktadır. Bu nedenle bu tez çalışmasında yılan robotun enerji verimli adaptif hareketi için optimizasyon ve öğrenme tabanlı denetleyiciler ele alınarak yılan robotun otonom hareketinin iyileştirilmesi amaçlanmıştır.

Bilimsel desteği ve yönlendirici fikirleriyle yol gösteren, kendisiyle çalışmaktan mutluluk duyduğum danışmanım Prof. Dr. İsmail Hakkı ALTAŞ'a teşekkürlerimi saygıyla sunarım. Tez izleme komitesinde yer alarak değerli görüşleri ile çalışmanın şekillenmesine katkı sağlayan Doç. Dr. Emre ÖZKOP ve Doç. Dr. Mehmet İTİK hocalarıma da gönülden teşekkür ederim. Ayrıca tecrübelerini ve değerli zamanını esirgemeyerek bana her fırsatta yardımcı olan kıymetli hocam Doç. Dr. Mustafa Şinasi AYAS'a ve güler yüzüyle motivasyon sağlayan Dr. Öğr. Üyesi Beste ÜSTÜBİOĞLU'na da teşekkürlerimi sunarım.

Tez çalışmam süresince BİDEB 2211-C Yurt İçi Öncelikli Alanlar Doktora Burs Programı kapsamında maddi olarak beni destekleyen Türkiye Bilimsel ve Teknolojik Araştırma Kurumu (TÜBİTAK)'a teşekkür etmeyi bir borç bilirim.

Çalışmalarım süresince sabır, destek ve sevgileriyle yanımda olan değerli dostlarım Seniha KETENCİ'ye, Reyhan ÖRNEK'e, Büşra ÖZGENÇ'e, Sibel BEKTAŞ'a ve Yunus ASLANHAN'a çok teşekkür ederim. Ayrıca Farklı Boyut Akademisi ailesinin her bir üyesine bana kattıkları güzel değerlerle sağladıkları motivasyon için şükranlarımı sunarım. Son olarak hayatım boyunca varlıkları ve destekleriyle bana güç veren ve her zaman yanımda olan sevgili anneme ve babama sonsuz teşekkürlerimi ve sevgilerimi sunarım.

Bu tez çalışmasını, okuyup öğrenmeye, öğrenene ve öğretene büyük değer veren canım dedem rahmetli Hanifi TABAR'a ithaf ediyorum.

Yunus Emre'nin "İlim, ilim bilmektir; İlim kendin bilmektir. Sen kendin bilmezsen; Ya nice okumaktır" mısralarını rehber edinerek ilerlediğim bu süreçte ortaya çıkan bu tezin bundan sonraki çalışmalara ve insanlığa fayda sağlaması temennisiyle.

Yeşim Aysel BAYSAL

Trabzon 2022

TEZ ETİK BEYANNAMESİ

Doktora Tezi olarak sunduğum "Yılan Robotun Öğrenme Tabanlı Adaptif Hareket Kontrolü" başlıklı bu çalışmayı baştan sona kadar danışmanım Prof. Dr. İsmail Hakkı ALTAŞ'ın sorumluluğunda tamamladığımı, verileri/örnekleri kendim topladığımı, deneyleri/analizleri ilgili laboratuvarlarda yaptığımı/yaptırdığımı, 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. 26/07/2022

Yeşim Aysel BAYSAL

İÇİNDEKİLER

Sayfa No

Doktora Tezi

ÖZET

YILAN ROBOTUN ÖĞRENME TABANLI ADAPTİF HAREKET KONTROLÜ

Yeşim Aysel BAYSAL

Karadeniz Teknik Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik-Elektronik Mühendisliği Anabilim Dalı
Danışman: Prof. Dr. İsmail Hakkı ALTAŞ
2022, 126 Sayfa

Yılan robotun gerçek dünyadaki potansiyel uygulamaları için oldukça önemli olan enerji verimli adaptif hareketi bu tez çalışmasında ele alınmıştır. Tekerleksiz bir yılan robot mekanizmasının göz önüne alındığı bu çalışmada simbiyotik organizma arama algoritmasına dayanan iki farklı çok amaçlı optimizasyon yöntemi kullanılarak yılan robotun enerji verimli hareketi için optimum yürüyüş parametreleri araştırılmıştır. Önerilen her iki yöntem ile tatmin edici kararlı sonuçlar elde edilirken hızlı baskın olmayan sıralamalı çok amaçlı simbiyotik organizma arama algoritmasının tek bir çalışmada daha çok çözüm üreterek Pareto cephesinde daha düzgün bir dağılım sağladığı görülmüştür. Önerilen yöntemin yılan robotunun farklı çevre koşullarındaki enerji verimli adaptif hareketi için optimum yürüyüş parametrelerini bulmadaki performansı, robotun hareket ettiği yüzeyin camdan ahşaba kadar oldukça geniş bir aralıkta değiştiği dokuz farklı ortamda test edilerek gösterilmiştir. Yılan robotun eklem yörünge kontrolü için geleneksel kontrol yöntemlerinden farklı olarak değişen çevre koşullarına uyum sağlama yetenekleriyle öne çıkan derin pekiştirmeli öğrenme tabanlı dört farklı denetleyici geliştirilerek yörünge takip performansları birbirleriyle ve PD denetleyici ile karşılaştırılmıştır. Elde edilen sonuçlardan transfer öğrenmenin, ajanın yakınsama hızını ve genelleştirme özelliğini iyileştirerek daha üstün bir performans sunduğu görülmüştür. Son olarak, yılan robotun enerji verimli hareketi için ikiz gecikmeli derin deterministik politika gradyanı algoritmasının kullanıldığı bir denetleyici ile hız kontrolü yapılarak biyolojik yılanların yürüyüş biçimlerine benzer adaptif yörüngeler üretilmiştir.

Anahtar Kelimeler: Tekerleksiz Yılan Robot, Enerji Verimli Hareket, Adaptif Hareket Kontrolü, Yörünge Kontrolü, Öğrenme Tabanlı Kontrol, Çok Amaçlı Optimizasyon, Derin Pekiştirmeli Öğrenme, Transfer Öğrenme

PhD. Thesis

SUMMARY

LEARNING-BASED ADAPTIVE MOTION CONTROL OF A SNAKE ROBOT

Yeşim Aysel BAYSAL

Karadeniz Technical University
The Graduate School of Natural and Applied Sciences
Electrical-Electronics Engineering Graduate Program
Supervisor: Prof. Dr. İsmail Hakkı ALTAŞ
2022, 126 Pages

Energy efficient locomotion of a wheel-less snake robot is very crucial for potential applications of untethered snake robots. For this reason, the optimal gait parameters for the energy efficient locomotion of the snake robot are investigated with two different multi-objective optimization methods based on symbiotic organism search algorithm in this study. The obtained results demonstrate that both proposed methods achieve satisfying stable results regarding power consumption reduction with optimal forward velocity for lateral undulation motion. However, it is seen that fast non-dominated sorting multi-objective symbiotic organism search algorithm provides an advantage on obtaining a uniformly distributed solution set with a good diversity only in a single run. The effectiveness of the proposed method in finding the optimal gait parameters for adaptive locomotion of the snake robot is proved by testing at nine different environments changing in a fairly wide friction range from glass to wood. Moreover, four different controllers based on deep reinforcement learning algorithms which stand out with their ability to adapt to changing environmental conditions unlike traditional control methods, are developed for trajectory tracking control of the snake robot joints in this study. The obtained results indicate that TD3 algorithm combined with transfer learning presents best performance in terms of the convergence speed and generalization capability of the controller. Finally, for the energy efficient locomotion of the snake robot, the movement patterns similar to biological snake gaits are achieved by generating adaptive joint trajectories with a speed controller trained using TD3 algorithm.

Key Words: Wheel-less Snake Robot, Energy Efficient Locomotion, Adaptive Motion Control, Trajectory Control, Learning Based Control, Multi-objective Optimization, Deep Reinforcement Learning, Transfer Learning

ŞEKİLLER DİZİNİ

Sayfa No

TABLOLAR DİZİNİ

Sayfa No

SEMBOLLER DİZİNİ

a	:	Aksiyon
A	:	Aksiyon uzayı
A	:	Bir vektörün peş peşe gelen eleman çiftlerini toplamak için kullanılan matris
AC	:	Aktör- Kritik (Actor-Critic)
Adam	:	Adaptif Moment Tahmini (Adaptive Moment Estimation)
AIRL	:	Çekişmeli Ters Pekiştirmeli Öğrenme (Adversarial Inverse Reinforcement Learning)
A_r	:	Aksiyon uzay aralığı
B	:	Bias vektörü
BF	:	Fayda faktörü
CD	:	Kalabalık Mesafe (Crowding Distance)
CNN	:	Evrişimli Sinir Ağları (Convolutional Neural Network)
COT	:	Yer Değiştirme Maliyeti (Cost of Transportation)
CPG	:	Merkezi Desen Üreteci (Central Pattern Generetor)
CS	:	Guguk Kuşu Arama (Cuckoo Search)
$(c_{t,i}, c_{n,i})$	:	i. linkin teğet ve normal yöndeki sürtünme katsayıları
d	:	TD3 politika ve hedef ağlarının güncelleme sayısı
D	:	Tekrarlama belleği
D	:	Bir vektörün peş peşe gelen eleman çiftlerini çıkarmak için kullanılan matris
DDPG	:	Derin Deterministik Politika Gradyanı (Deep Deterministic Policy Gradient)
DDQN	:	Çift Derin Q Ağı (Double Deep Q Network)
DL	:	Derin Öğrenme (Deep Learning)
DOF	:	Serbestlik Derecesi (Degrees of Freedom)
DPG	:	Deterministik Politika Gradyanı (Deterministic Policy Gradient)
DRL	:	Derin Pekiştirmeli Öğrenme (Deep Reinforcement Learning)
DNN	:	Derin sinir ağı (Deep Neural Network)
DQN	:	Derin Q Ağı (Deep Q Network)
e	:	Bir vektörün bütün elemanlarını toplamak için kullanılan vektör
e_v	:	Hız hatası

e_ϕ	:	Referans yörünge takip hatası
EC	:	Evrimsel Hesaplama (Evolutionary Computation)
F_k	:	k. cephe
FNS	:	Hızlı Baskın Olmayan Sıralama (Fast Non-dominated Sorting)
FNSMOSOS	:	Hızlı Baskın Olmayan Sıralamalı Çok Amaçlı Simbiyotik Organizma Arama (Fast Non-dominated Sorting Multi-objective Symbiotic Organism Search)
$(f_{R,x,i},f_{R,y,i})$	:	i. linke etki eden sürtünme kuvvetleri
GA	:	Genetik Algoritma (Genetic Algorithm)
GD	:	Gradyan Azaltma (Gradient Descent)
$(h_{x,i},h_{y,i})$	:	i+1. linkten i. linke etki eden kısıtlama kuvveti
$(h_{x,i-1},h_{y,i-1})$	:	i-1. linkten i. linke etki eden kısıtlama kuvveti
ISE	:	Hatanın karesinin integrali (Integral square error)
J	:	Amaç fonksiyonu
J_i	:	i. linkin eylemsizlik momenti
k_D	:	Türev kazancı
k_P	:	Oransal kazanç
L	:	Kayıp fonksiyonu
m	:	Linklerin kütleleri toplamı
M	:	Mini-paket boyutu
MDP	:	Markov Karar Süreci (Markov Decision Process)
m_i	:	i. linkin kütlesi
m_n	:	Amaç sayısı
MLP	:	Çok Katmanlı Algılayıcı (Multi Layer Perceptron)
MO	:	Çok Amaçlı Optimizasyon (Multi-objective Optimization)
MOSOS	:	Çok Amaçlı Simbiyotik Organizma Arama (Multi-objective Symbiotic Organism Search)
MV	:	Ortak vektör
n	:	Link sayısı
N	:	DDPG aksiyon gürültüsü
n_d	:	Organizma boyutu
N_{max}	:	Maksimum iterasyon sayısı
n_o	:	Organizma sayısı
n_p	:	p organizmasına baskın olan organizmaların sayısı

n_q	:	S_p kümesindeki q organizmasının baskınlık sayısı
NSGA-II	:	Baskın Olmayan Sıralamalı Genetik Algoritma II (Non-dominated Sorting Genetic Algorithm II)
l_i	:	i. linkin yarı uzunluğu
OU	:	Ornstein-Uhlenbeck
$\hat{P}$	:	Normalize edilmiş güç tüketimi
P_{avg}	:	Ortalama güç tüketimi
$(P_{avg})_{sc}$	:	Normalize edilmiş ortalama güç tüketimi
PD	:	Oransal-türevsel (Proportional Derivative)
PID	:	Oransal-integral-türevsel (Proportional Integral Derivative)
PG	:	Politika gradyanı (Policy Gradient)
PPO	:	Yakınsal Politika Optimizasyonu (Proximal Policy Optimization)
PSO	:	Parçacık Sürü Optimizasyonu (Particle Swarm Optimization)
(p_x, p_y)	:	Robotun ağırlık merkezinin global koordinatları
$(\dot{p}_x, \dot{p}_y)$	:	Robotun hızı
$(\ddot{p}_x, \ddot{p}_y)$	:	Robotun ivmesi
$(Q_{\theta_1}, Q_{\theta_2})$	:	TD3 kritik ağları
Q^*	:	Optimal aksiyon değer fonksiyonu
r	:	Ödül
R	:	Rotasyon matrisi
ReLU	:	Doğrultulmuş Lineer Birim (Rectified Linear Unit)
RL	:	Pekiştirmeli Öğrenme (Reinforcement Learning)
RMSProp	:	Ortalama Karesel Yayılımın Karakökü (Root Mean Square Propogation)
RNN	:	Tekrarlamalı Sinir Ağları (Recurrent Neural Network)
r_P	:	Güç tüketimi ödül fonksiyonu
r_v	:	Hız ödül fonksiyonu
s	:	Durum
s'	:	Gelecek durum
S	:	Durum uzayı
SARSA	:	State Action Reward State Action
SGD	:	Stokastik Gradyan Azaltma (Stochastic Gradient Descent)
SOS	:	Simbiyotik Organizma Arama (Symbiotic Organism Search)
S_p	:	p organizmasının baskın olduğu organizmalardan oluşan küme

TD	:	Zamansal Fark (Temporal difference)
TD3	:	İkiz Gecikmeli Derin Deterministik Politika Gradyanı (Twin Delayed Deep Deterministic Policy Gradient)
T_f	:	Simülasyon zamanı
TRPO	:	Güven Bölgesi Politika Optimizasyonu (Trust Region Policy Optimization)
T_s	:	Örnekleme zamanı
u_i	:	i+1. linkten i. linke etki eden eyleyici torku
u_{i-1}	:	i-1. linkten i. linke etki eden eyleyici torku
v_f	:	Robotun ileri yöndeki hızı
$(v_f)_{sc}$	:	Normalize edilmiş ileri hız
v_{ref}	:	Referans hız değeri
$\overline{v_t}$	:	Yılan robotun teğet yöndeki hızı
V^*	:	Optimal durum değer fonksiyonu
w	:	Amaç ağırlık katsayısı
$\mathbf{W}$	:	Ağ ağırlık katsayısı matrisi
X_{best}	:	En yüksek uygunluk değerine sahip organizma
(x_i, y_i)	:	i. linkin ağırlık merkezinin global koordinatları
X_i	:	Ekosistemdeki i. organizma
X_j	:	X_i ile etkileşime giren organizma
x_{ref}	:	Filtrelenmiş referans işareti
α	:	Referans işaretinin genliği
α_n	:	Öğrenme hızı
α_a	:	Aktör için öğrenme hızı
α_c	:	Kritik için öğrenme hızı
α_σ	:	Standart sapma azalma oranı
β	:	Art arda bağlı iki modül arasındaki faz farkı
γ	:	Azaltma faktörü
γ_c	:	Yön kontrol parametresi
ε	:	TD3 keşif aksiyon gürültüsü
ε̃	:	TD3 hedef aksiyon gürültüsü
θ	:	Ortalamaya çekim sabiti
θ_i	:	i. link ile global x ekseni arasındaki açı

$\dot{\theta}_i$	:	*i.* linkin hızı
$\ddot{\theta}_i$	:	*i.* linkin ivmesi
θ^{Q}	:	DDPG kritik ağının parametreleri
$\theta^{Q'}$	:	DDPG kritik hedef ağının parametreleri
θ^{μ}	:	DDPG aktör ağının parametreleri
$\theta^{\mu'}$	:	DDPG aktör hedef ağının parametreleri
θ'	:	TD3 hedef kritik ağının parametreleri
$\bar{\theta}$	:	Yönelme açısı
θ^{-}	:	DQN hedef ağ parametreleri
μ	:	Gürültü modelinin ortalaması
ξ	:	Sönüm oranı
π	:	Politika
π^*	:	Optimal politika
$(\pi_{\phi_1},\pi_{\phi_2})$	:	TD3 aktör ağları
ρ^{β}	:	Durum ziyaret dağılımı
σ	:	Gürültü modelinin standart sapmasını
τ	:	Yumuşak hedef güncelleme katsayısı
$\boldsymbol{\tau}_R$	:	Linkin ağırlık merkezine etki eden sürtünme tork matrisi
ϕ_i	:	*i.* eklemin açısı
$\dot{\phi}_i$	:	*i.* eklemin hızı
ϕ_{ref}	:	Referans eklem yörünge işareti
$\dot{\phi}_{ref}$	:	Referans eklem hızı
ϕ'	:	TD3 hedef aktör ağının parametreleri
ω	:	Hareketin açısal frekansı
ω_n	:	Doğal frekans

1. GENEL BİLGİLER

1.1. Giriş

Yılan robotların ince-uzun yapıları, yüksek serbestlik derecesine sahip olmaları ve ağırlık merkezlerinin yere yakın olmasının sağladığı hareket kararlılığı, bu mekanizmaların hemen hemen her türlü arazide hareket edebilmesine olanak sağlamaktadır. İnsanların veya diğer mobil robotların ulaşmasının mümkün olmadığı dar veya tehlikeli ortamlara ulaşarak çalışabilen yılan robotların gerçek ve potansiyel uygulamaları; yangınla mücadele (Liljebäck vd., 2006), nükleer tesisler (Bogue, 2014, Huang vd., 2018), boru hatları (Virgala vd., 2020), keşif (Hou vd., 2020), askeri, medikal (Seetohul ve Shafiee, 2022), denizaltı operasyonları (Kelasidi vd., 2019), afet yönetimi ve arama kurtarma (Wolf vd., 2005; Kamegawa vd., 2020) olmak üzere oldukça geniştir (Liu vd., 2021; Yang vd., 2022). Bu özelliklerinden dolayı araştırmacılar tarafından ilgi çekici hale gelen yılan robotlarla ilgili çalışmalar özellikle son yıllarda artarak devam etmekle birlikte sürtünme bağımlı hareketlerinden dolayı düşük hareket verimliliği ve sahip oldukları yüksek serbestlik derecelerinden dolayı kontrol edilebilmelerinin zorlaşması çözülmesi gereken konuların başında gelmektedir.

Biyolojik yılanlar, üzerinde hareket ettikleri yüzeye ve hareket hızlarına bağlı olarak dört temel hareket biçimini kullanırlar. Bu yürüyüş biçimlerinden yanal dalgalanma hareketi, hemen hemen bütün biyolojik yılanlarda görülen en yaygın ve en hızlı hareket biçimidir. Yılanın bu hareketi gerçekleştirebilmesi için yerdeki taş, dal gibi doğal çıkıntılara ve destek noktalarına ihtiyacı vardır. Dolayısıyla bu hareket biçimi ile yılanın kaygan ve pürüzsüz ortamlarda hareket etmesi mümkün değildir. İkinci yürüyüş biçimi, genellikle çöl yılanları tarafından kumlu ve kaygan zeminlerde kullanılan yanal kayma hareketidir. Bu hareketin yanal dalgalanma hareketine göre daha verimli olduğu belirtilmektedir (Sarrafan vd., 2011). Yılanın kendisini arka kısmıyla iterken ön kısmı ile çekmesi şeklinde gerçekleştirdiği akordeon tipi (concertina) hareket biçimi ise dar alanlarda ilerlemek ve tırmanmak için kullanılmaktadır. Yılanın akordeonun açılıp kapanmasını andıran bu hareket biçimi, vücudun önce uzayıp sonra kısalması şeklinde sıralı bir eylemi gerektirdiğinden en yavaş ve en verimsiz hareket biçimlerinden biridir

(Walton vd., 1990). Dördüncü yürüyüş biçimi olan doğrusal hareket ise büyük ve ağır gövdeli yılanlar tarafından kullanılan daha az enerji gerektiren yavaş bir harekettir.

Biyolojik yılanların eşsiz hareket yeteneklerinden esinlenerek geliştirilen yılan robotların var olan potansiyellerini kullanabilmeleri için gerçek partnerleri gibi çevre koşullarına göre hareketlerini uyarlamaları gerekmektedir. Bunun için ya hareket biçimini değiştirmeleri ya da aynı hareket biçimini koruyarak sadece hareketin yürüyüş parametrelerini modüle etmeleri gerekir.

Yılan robotların kontrol yöntemleri sinüs tabanlı, model tabanlı ve merkezi desen üreteci (Central Pattern Generetor, CPG) tabanlı olmak üzere kategorize edilebilir.

Sinüs tabanlı yaklaşımlar, hareket dalgalarını üretmek için sinüs tabanlı fonksiyonları kullanırlar. Bu yaklaşımın avantajı basit bir yapıya sahip olması ve faz, frekans ve genlik gibi önemli yürüyüş parametrelerinin açık bir şekilde tanımlanmasıdır. Dezavantajı ise parametre değişimlerinin istenen değerlere geçişlerde salınım hareketleri meydana getirmesidir. Bir diğer dezavantajı ise sensör geri besleme sinyalleri ile kullanılmasının kolay olmamasıdır (Ijspeert ve Crespi, 2008). Bu nedenle özelikle robotun bilinmeyen çevrelerde etkili hareket sağlayabilmesi için hareketini değiştirmesi gereken durumlarda bu kontrol yaklaşımının uygun olmadığı görülmektedir.

Model tabanlı yaklaşımlar, yürüyüş üreten kontrol kurallarını tasarlamak için robotun kinematik veya dinamik modellerini kullanırlar. Kontrol kuralları bazen sinüs tabanlı fonksiyonlara dayanır fakat model tabanlı yaklaşımlar kinematik kısıtları veya yaklaşık hareket denklemlerini kullanarak belirli bir robot için en hızlı yürüyüşü belirlemeyi sağlar. Böylece denetleyici tasarlamak için oldukça faydalı olan model tabanlı yaklaşımlar, iki tane önemli dezavantaja sahiptir. Bunlardan ilki, denetleyicinin her zaman operatör ile etkileşime uygun olmaması, ikinci ise model doğru olmadığında denetleyici performansının kötüleşmesidir. Bu durumda özellikle bilinmeyen karmaşık ortamlarda modele dâhil edilmeyen etki kuvvetlerinden dolayı denetleyici performansı hızlı bir şekilde bozulacaktır (Ijspeert ve Crespi, 2008). Böylece model tabanlı kontrol yönteminin de bilinmeyen çevreler için tek başına yeterli olmadığı açıktır.

CPG tabanlı yaklaşım, omurgalı hayvanlarda nefes, çiğneme veya hareket biçimi gibi periyodik davranışları üretmekten sorumlu sinir ağından esinlenerek ortaya konulmuş bir yaklaşım olup çeşitli robotik uygulamalarındaki başarısı kanıtlanmıştır (Yu vd., 2014). CPG, basit ve düşük boyutlu giriş sinyallerini alarak yüksek boyutlu ritmik çıkış sinyallerini üretebilmektedir. Ayrıca, bu ritmik hareketleri üretmek için sensör geri

beslemesine ihtiyaç duyulmamaktadır. Yılan robotların değişen çevre koşullarına göre adaptif hareketi için literatürde genellikle sağladıkları avantajlar sebebiyle CPG tabanlı kontrol yöntemleri kullanılmaktadır (Crespi ve Ijspeert, 2006; Wu ve Ma, 2010a; Tang ve Ma, 2011; Matsuo vd., 2012; Nor ve Ma, 2014a; 2014b; 2014c; Bing vd., 2017a; 2017b; Wang vd., 2017). Adaptif yörüngeler üretebilmek için CPG kontrol parametrelerinin ayarlanması gerekmektedir. Bu parametrelerin ayarlanması için literatürde genellikle deneme-yanılma yöntemi kullanılmıştır. CPG parametrelerinin deneme-yanılma yöntemiyle ayarlanmasının dezavantajı, geniş bir arama uzayında istenilen hareket biçimini elde edebilmek için ayarlanması gereken pek çok parametrenin bulunması ve bu işin operatör tarafından yapılmasının oldukça zor olmasıdır. Bu durum robotun adaptasyonunu kısıtlamaktadır.

Yılan robotun hareket parametrelerini optimize ederek enerji verimli adaptif yürüyüşler üretme üzerine yapılan çalışmalar da mevcuttur. Bu çalışmalar bölüm 1.4'te detaylandırılmıştır. Elde edilen optimize edilmiş yürüyüşler manuel ayarlanan yürüyüşlerden daha iyi performans gösterse de biyolojik yılanların doğal hareketleriyle karşılaştırıldığında hala çok verimsiz kalmaktadırlar (Bing vd., 2020a).

Sistemin doğru modeline ve ortam hakkında ön bilgiye ihtiyaç duymamasıyla robot kontrolünde öne çıkan derin pekiştirmeli öğrenme (Deep Reinforcement Learning, DRL), algoritmalardaki gelişmelerle birlikte yürüyüş üretme, otonom sürüş sistemleri gibi zorlu kontrol görevlerinde de kullanılmaya başlamıştır (Rubí vd., 2021; Li vd., 2021, Ji vd., 2022). Bölüm 1.5'te bahsedilen çalışmalarda DRL'nin, özellikle biyolojik canlılardan esinlenerek geliştirilmiş yüksek serbestlik derecesine sahip robotlar için enerji verimli adaptif yörüngeler üretmede oldukça faydalı olduğu görülmektedir.

1.2. Tezin Amacı

Bu tez çalışmasında yılan robotun enerji verimli adaptif hareketi için optimizasyon ve öğrenme tabanlı denetleyiciler ile yılan robotun otonom hareketinin iyileştirilmesi amaçlanmıştır. Bunun için;

- Gerçek uygulamalar için uygun tekerleksiz bir yılan robot mekanizmasının dinamik modellemesi yapılarak yılan robot simülatörünün elde edilmesi,
- Yılan robotun eklem yörünge kontrolü için bilinmeyen düzensiz çevrelerde adaptif optimal kontrol sağlayan DRL tabanlı denetleyicilerin geliştirilmesi,

- Yılan robotun sabit ve değişen çevre koşullarındaki enerji verimli hareketi için optimal yürüyüş parametrelerinin analizi,
- DRL tabanlı bir denetleyici ile enerji verimli adaptif referans yörüngelerin üretilmesi ele alınmıştır.

1.3. Tezin Kapsamı

Yılan robotlarla ilgili çalışmaların büyük çoğunluğunda uzuvların holonomik olmayan kısıtlara bağlı olduğu hareketi kullanan tekerlekli robotlara odaklanılmıştır. Bu mekanizmalar, sahip oldukları pasif tekerleklerin dağınık ortamlarda çok iyi hareket edememelerinden dolayı genellikle sadece düz yüzeylerde hareket edebilmektedirler ve bu nedenle yılan robotların daha zorlu ortamlar içeren pratik uygulamaları için uygun değillerdir. Fakat yılan robotun gerçek uygulamaları bu mekanizmaların bilinmeyen karmaşık ortamlarda hareket etmesini içermektedir. Bu nedenle bu tezde gerçek çevre koşulları için uygun olan tekerleksiz bir yılan robot ele alınarak gerçek uygulamalara yönelik yaklaşımlara yer verilmeye çalışılmıştır. Bölüm 1.3'te literatürdeki yılan robotun modellemesi üzerine yapılan çalışmalara kısaca değinildikten sonra, tekerleksiz yılan robotun yürüyüş biçimlerinden yanal dalgalanma hareketini gerçekleştirebilmesi için robot ile zemin arasındaki sürtünme özelliklerinin dikkate alındığı bir matematiksel model yaklaşımı sunulmuştur. Bu modele dayanan yılan robot simülatörünün elde edilmesi ve geleneksel kontrol yöntemi olan oransal-türevsel (Proportional Derivative, PD) denetleyici ile kontrolü Bölüm 2.2'de anlatılarak elde edilen sonuçlar raporlanmıştır.

Enerji verimli yürüyüş üretme, sürtünme bağımlı hareketlerinden dolayı zaten düşük hareket verimliliğine sahip olan yılan robotların gerçek dünya uygulamalarında sahip oldukları kısıtlı kaynakları verimli kullanabilmeleri için oldukça önemlidir. Yılan robotun enerji verimli optimal hareketi ile ilgili literatür Bölüm 1.4'te sunulmuştur. Bu çalışmaların çoğunda sadece yılan robotun hızının göz önüne alındığı görülmektedir. Oysaki enerji verimli hareket için robotun hızı maksimize edilirken enerji tüketimi minimumda tutulmalıdır. Bu iki amacın göz önüne alınarak ağırlıklı toplam yöntemiyle aynı amaç fonksiyonunda birleştirildiği çalışmalar da mevcuttur. Fakat bu yöntem Bölüm 1.4'te bahsedilen bazı dezavantajlara sahiptir. Bu nedenle bu tezde yılan robotun enerji verimli hareket problemi, tek bir çözüm yerine amaçlar arasındaki Pareto optimal çözüm setini veren çok amaçlı optimizasyon (Multi-objective Optimization, MO) problemi olarak ele

alınmıştır. Ayrıca literatürde evrimsel optimizasyon algoritmalarının hareket optimizasyonu için daha uygun olduğu belirtilmesine rağmen bu yöntemlerin yılan robotların enerji verimli hareketi için çok az çalışmada ele alındığı görülmüştür. Mevcut çalışmalarda kullanılan geleneksel optimizasyon yöntemlerinin de lokal optimuma takılma, yüksek hesaplama karmaşası ve yavaş yakınsama gibi problemleri bulunmaktadır. Bu nedenle bu tezde yılan robotun optimal hareket parametrelerini bulmak için Bölüm 1.4'te sunulan pek çok avantajından dolayı hızlı baskın olmayan sıralamalı çok amaçlı simbiyotik organizma arama (Fast Non-dominated Sorting Multi-objective Symbiotic Organism Search, FNSMOSOS) algoritması önerilmiştir. Ayrıca ağırlıklı toplam yöntemine dayanan MOSOS algoritması da yılan robotun optimal hareketi için ele alınarak elde edilen sonuçlar FNSMOSOS algoritmasıyla karşılaştırılmıştır. SOS algoritmasına dayanan çok amaçlı bu iki algoritmanın teorik altyapısı Bölüm 1.4'te, yılan robotun enerji verimli hareket problemine uygulanması ve elde edilen sonuçlar ise Bölüm 2.3'te sunulmuştur.

Sürtünme kuvvetlerindeki değişiklikler yılanın hızını doğrudan etkilediğinden, farklı zemin koşullarında etkin hareket kabiliyetlerini korumak için hareketlerini çevreye göre uyarlarlar. Benzer şekilde yılan robotun farklı çevre koşullarındaki enerji verimli adaptif hareketi için optimal hareket parametreleri robotun hareket ettiği yüzeyin camdan ahşaba kadar oldukça geniş bir aralıkta değiştiği düşünülerek belirlenen dokuz farklı ortamda FNSMOSOS algoritması kullanılarak incelenmiştir. Elde edilen sonuçlar Bölüm 2.3'te sunulmuştur.

Yılan robotun enerji verimli hareket edebilmesi için sadece uygun referans yörüngelerin üretilmesi yeterli olmamaktadır. Üretilen bu yörüngelerin takibi de oldukça önemlidir. Fakat literatürde bu konu üzerinde pek durulmamakla birlikte sinüs tabanlı fonksiyonlar veya CPG ile üretilen referans yörüngelerin takibi için çoğunlukla basitliğinden ötürü PD veya PID (Proportional Integral Derivative) denetleyiciler kullanılmıştır. Fakat bu denetleyicilerin doğrusal olmayan sistemler için yeterli olmadığı bilinmektedir. Bu nedenle bu tezde yılan robotun eklem yörünge kontrolü, klasik denetleyicilerden farklı olarak yeni bir ortam veya yeni bir amaç için otonom öğrenme esnekliğine ve genelleme yeteneğine sahip olan DRL tabanlı algoritmalar kullanılarak ele alınmıştır. Teorik altyapıları Bölüm 1.5'te sunulan sürekli aksiyon uzayı için uygun derin deterministik politika gradyanı (Deep Deterministic Policy Gradient, DDPG) ve ikiz gecikmeli derin deterministik politika gradyanı (Twin Delayed Deep Deterministic Policy Gradient, TD3) algoritmaları ile dört farklı denetleyici geliştirilerek yörünge takip

performansları birbirileriyle ve PD denetleyici ile karşılaştırılmıştır. Elde edilen sonuçlar Bölüm 2.4'te verilmiştir.

Literatürde yılan robotların farklı hareket biçimleri arasında geçişini ele alan az sayıda çalışma bulunmakla birlikte bu çalışmalarda kontrol parametrelerinin deneme-yanılma yöntemiyle operatör tarafından ayarlandığı görülmektedir. Bu çalışmaların dezavantajı kontrol edilmesi gereken pek çok kontrol parametresinin operatör tarafından ayarlanmasının zorluğudur. Bu durum düşünüldüğü zaman bu tür yaklaşımların robotun bilinmeyen çevrelere adaptasyonu için yetersiz kaldığı ve bu kontrol parametrelerinin otonom olarak ayarlanmasının robotun bilinmeyen çevrelere adaptasyonu için önemli olduğu görülmektedir. Bu nedenle bu tezde, operatör bilgisine ve parametrelerin deneme yanılmayla ayarlanmasına gerek kalmadan referans hareket hızının değişimi ile enerji verimli hareketi sağlayabileceği adaptif yörüngeleri kendisinin üretip öğrenebilmesini sağlayan TD3 algoritmasının kullanıldığı DRL tabanlı bir denetleyici önerilmiştir. Elde edilen sonuçlar Bölüm 2.5'te sunulmuştur.

Bölüm 3'te elde edilen sonuçlar özetlenerek tartışılırken, Bölüm 4'te ise çalışmanın kısıtları ve gelecek çalışmalar için önerilere yer verilmiştir.

1.4. Yılan Robotun Matematiksel Modeli

Yılan robot hareketinin modellenmesi ve analizi ile ilgili yapılan çalışmalar robotun hareket ettiği yüzeyin düz veya düzensiz olmasına göre kategorize edilebilirler (Liljebäck vd., 2012a). Düz yüzeylerdeki modeller de kendi aralarında yana kayma kısıdının olup olmamasına göre ikiye ayrılabilir. Bu çalışmalara aşağıda değinilirken, düzensiz ortamlarda hareketin modellenmesi bu tez kapsamının dışında olduğu için bu çalışmalara değinilmemiştir.

Yılan robotların yana doğru hareket edemeyeceği varsayımı yılan robotun modellenmesinde kullanılan en yaygın yaklaşımdır. Bu varsayım, robotun hareket denklemlerine holonomik olmayan kısıtların dahil edilmesiyle sağlanırken (Bloch vd., 2003), yılan robot prototiplerinde ise genellikle robotun gövdesi boyunca pasif tekerleklerin yerleştirilmesiyle elde edilmektedir. Bu çalışmalardan bazıları aşağıdaki gibi özetlenebilir.

Krishnaprasad ve Tsakiris (1994) ile Kelly ve Murray'ın (1995) çalışmalarında tekerlekli yılan robotun vücut şeklinin değişimi ile buna bağlı olarak yer değiştirmesi

arasındaki ilişki analiz edilerek bu mekanizmaların kinematik modeli ele alınmış ve kontrol edilebilirlikleri incelenmiştir. Benzer yaklaşımların ele alındığı bazı çalışmalarda tekerlekli yılan robotun dinamiği de göz önüne alınmıştır (Ostrowski, 1996; Ostrowski ve Burdick, 1998). Bu çalışmalarda robotun indirgenmiş modeli sistem simetrilerinden yararlanılarak elde edilmiştir. Bir başka çalışmada ise tekerlekli bir yılan robotun 2D dinamik modeli Lagrange hareket denklemleri kullanılarak geliştirilmiştir (Prautsch ve Mita, 1999).

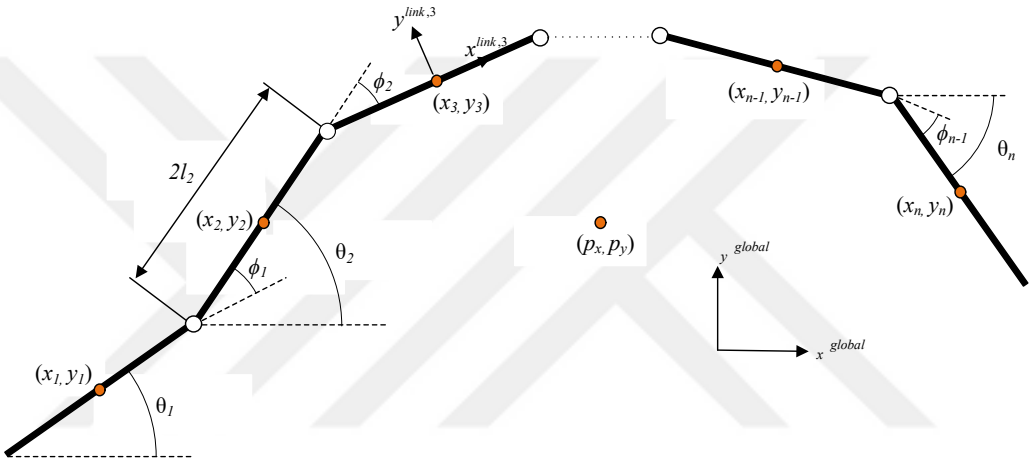
Yana kayma kısıdının göz önüne alındığı çalışmaların yanı sıra yılan robotun linklerinin biyolojik yılanlardaki gibi izotropik olmayan sürtünme özelliğini sergilediği varsayımını yapan pek çok model de bulunmaktadır. Bu sürtünme özelliğine dayanan modellerde vücut şeklinin değişimi ile robotun yer değiştirmesi arasında doğrudan bir ilişki olmadığı için, bu modeller genellikle yana kayma kısıtlı modellerden daha karmaşıktırlar (Liljebäck vd., 2012a). Bu çalışmalardan bazıları aşağıdaki gibi özetlenebilir.

Statik ve dinamik Coulomb sürtünme kuvvetlerini içeren izotropik olmayan sürtünme özellikleri ile yılan robotun 2D dinamik modelini ele alan bir çalışmada Newton-Euler denklemleri kullanılmıştır. (Ma, 2001). Bir diğer çalışmada ise tekerleksiz düzlemsel bir yılan robotun dinamik modeli Newton'un ikinci hareket yasası kullanılarak geliştirilmiştir (Saito, 2002). İzotropik olmayan viskoz zemin sürtünmesi ile yılan robotun düzlemsel dinamik modelleri Kane ve Lecison (2000) ve Hicks (2003) tarafından yapılan çalışmalarda sunulurken hem viskoz hem de Coulomb sürtünme kuvvetlerini içeren bir sürtünme modeli ise Mehta vd. (2008) tarafından önerilmiştir. İzotropik zemin sürtünme kuvvetlerini göz önünde bulunduran bir model ise Nilsson (2004) ve Chernousko (2005) tarafından ele alınmıştır. Newton-Euler denklemlerini kullanarak yılan robotun düz yüzeylerdeki hareketinin 3D dinamik modelini elen alan çalışmalar da mevcuttur. (Ma vd., 2004; Transeth vd., 2008). Statik ve dinamik Coulomb sürtünme kuvvetlerini içeren Ma vd. (2004)'nin geliştirdiği bu 3D model yanal dalgalanma hareketi sırasında sinüs kaldırma hareketi için kullanılmıştır. Transeth vd. (2008) ise set değerli sürtünme yasalarını kullanan düzgün olmayan mekanik modelleme yaklaşımını kullanmışlardır. Malayjerdi vd. (2019), tekerlekli bir yılan robotun 2D dinamik analizi için Vossoughi vd. (2008)'nin çalışmasında ilk kez kullanılan Gibbs-Appell yöntemini kullanarak tekerleksiz bir yılan robotun 3D dinamik analizini gerçekleştirmişlerdir.

Bu tezde sunulan tekerleksiz yılan robotun 2D modeli, Saito vd. (2002)'deki kinematik ve dinamik eşitliklere benzer bir yaklaşımın kullanılmasıyla elde edilmiştir.

1.4.1. Yılan Robotun Kinematik Modeli

Yılan robot, *2l* uzunluğundaki *n* tane linkin *n-1* tane eklemle birbirine bağlanmasıyla oluşan modüler robotlardır. Yatay düzlemde 2D hareket edebilen *n* linkli bir yılan robot, *n+2* serbestlik derecesine (Degrees of Freedom, DOF) sahiptir. Şekil 1.1'de *n* linkten oluşan bir yılan robotun kinematik parametrelerini gösteren diyagram görülmektedir. Kinematik ve hareket denklemlerinde kullanılan yılan robotu karakterize eden bu parametreler ve açıklamaları ise Tablo 1.1'de verilmiştir.



Şekil 1.1. *n* linkten oluşan bir yılan robotun kinematik parametrelerini gösteren diyagram

Tablo 1.1. Yılan robotu karakterize eden parametreler

Parametre	**Açıklama**
n	Link sayısı
l_i	*i*. linkin yarı uzunluğu
m_i	*i*. linkin kütlesi
J_i	*i*. linkin eylemsizlik momenti
θ_i	*i*. link ile global *x* ekseni arasındaki açı
ϕ_i	*i*. eklemin açısı
(x_i, y_i)	*i*. linkin ağırlık merkezinin global koordinatları
(p_x, p_y)	Robotun ağırlık merkezinin global koordinatları
u_i	*i*+1. linkten *i*. linke etki eden eyleyici torku
u_{i-1}	*i*-1. linkten *i*. linke etki eden eyleyici torku
$(f_{R,x,i}, f_{R,y,i})$	*i*. linke etki eden sürtünme kuvveti
$(c_{t,i}, c_{n,i})$	*i*. linkin teğet ve normal yöndeki sürtünme katsayıları
$(h_{x,i}, h_{y,i})$	*i*+1. linkten *i*. linke etki eden kısıtlama kuvveti
$-(h_{x,i-1}, h_{y,i-1})$	*i*-1. linkten *i*. linke etki eden kısıtlama kuvveti

Yılan robotun her bir linki dönme ekseni bir ucundan geçen m_i kütleli $2l_i$ uzunluklu bir çubuk olarak düşünülebilir. Buna göre robotun *i*. linkinin eylemsizlik momenti (J_i) eşitlik (1.1)'deki gibi hesaplanır.

$$J_i = \frac{1}{3} m_i l_i^2 \tag{1.1}$$

Şekil 1.1'de görüldüğü gibi link açıları, her bir linkin global *x* ekseniyle saat yönünün tersi yönde yaptığı açı olarak tanımlanırken eklem açısı ϕ_i ise komşu iki linkin link açıları arasındaki farktır ve eşitlik (1.2)'deki gibi ifade edilebilir.

$$\phi_i = \theta_i - \theta_{i+1} \tag{1.2}$$

Yılan robotun kinematik ve hareket denklemleri türetilirken link ve eklem açıları sırasıyla $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_n]^{\mathrm{T}}$ ve $\boldsymbol{\phi} = [\phi_1, \ldots, \phi_{n-1}]^{\mathrm{T}}$ şeklinde vektör formda ifade edilecektir. Ayrıca model geliştirilirken kullanılacak olan bazı matris ve vektörler aşağıda tanımlanmıştır.

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & & \\ & \ddots & \ddots & \\ & & 1 & 1 \end{bmatrix}_{(n-1)xn}, \quad \mathbf{D} = \begin{bmatrix} 1 & -1 & & \\ & \ddots & \ddots & \\ & & 1 & -1 \end{bmatrix}_{(n-1)xn}$$

$$\mathbf{e} = [1,\ldots,1]^T, \quad \mathbf{E} = \begin{bmatrix} \mathbf{e} & \mathbf{0}_{nx1} \\ \mathbf{0}_{nx1} & \mathbf{e} \end{bmatrix}$$

$$\sin\boldsymbol{\theta} = [\sin\theta_1,\ldots,\sin\theta_n]^{\mathrm{T}}, \quad \cos\boldsymbol{\theta} = [\cos\theta_1,\ldots,\cos\theta_n]^{\mathrm{T}}, \quad \dot{\boldsymbol{\theta}}^2 = [\dot{\theta}_1^2,\ldots,\dot{\theta}_n^2]^{\mathrm{T}}$$

$$\mathbf{S}_{\boldsymbol{\theta}} = diag(\sin\boldsymbol{\theta}), \quad \mathbf{C}_{\boldsymbol{\theta}} = diag(\cos\boldsymbol{\theta}), \quad \mathrm{sgn}\,\boldsymbol{\theta} = [\sin\theta_1,\ldots,\sin\theta_n]^{\mathrm{T}}$$

$$\mathbf{M} = diag(m_1,\ldots,m_n), \quad \mathbf{L} = diag(l_1,\ldots,l_n), \quad \mathbf{J} = diag(J_1,\ldots,J_n)$$

A ve **D** matrisleri, bir vektörün peş peşe gelen eleman çiftlerini toplamak veya çıkarmak için kullanılırken **e** vektörü ise *n* boyutlu bir vektörün bütün elemanlarını toplamak için kullanılır.

Yılan robot için lokal ve global olmak üzere farklı koordinat eksenleri tanımlanır. Her linkin lokal koordinat ekseni (x_i, y_i) ilgili linkin ağırlık merkezine atanırken, global koordinat ekseni (p_x, p_y) ise yılan robotun ağırlık merkezine atanır. İki koordinat ekseni arasındaki rotasyon matrisi ise eşitlik (1.3)'teki gibi tanımlanır.

$$\mathbf{R}_{link,i}^{global} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix} \tag{1.3}$$

Buna göre robotun ağırlık merkezinin global pozisyonu **p** eşitlik (4)'teki gibi tanımlanabilir ve bu eşitlikten robotun global ağırlık merkezinin, linklerin ağırlık merkezlerinin pozisyonlarına bağlı olduğu görülmektedir.

$$\mathbf{p} = \begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} \frac{1}{m}\sum_{i=1}^{n} m_i x_i \\ \frac{1}{m}\sum_{i=1}^{n} m_i y_i \end{bmatrix} = \frac{1}{m}\begin{bmatrix} \mathbf{e}^T\mathbf{MX} \\ \mathbf{e}^T\mathbf{MY} \end{bmatrix} \tag{1.4}$$

Bu eşitlikteki $\mathbf{X} = [x_1, \ldots, x_n]^T$ ve $\mathbf{Y} = [y_1, \ldots, y_n]^T$ her bir linkin ağırlık merkezinin global koordinatlarını gösteren vektörleri ifade ederken, m ise bütün linklerin kütlelerinin toplamıdır.

Yılan robotun teğet (ileri) yöndeki hızı $\bar{v}_t$, robotun ağırlık merkezi hızının $\dot{\boldsymbol{p}}$, eşitlik (1.5)'te tanımlanan ve link açılarının ortalaması alınarak hesaplanan yılan robotun yönelme açısı $\bar{\theta}$ doğrultusundaki bileşenlerinden oluşur ve eşitlik (1.6)'daki gibi hesaplanabilir.

$$\bar{\theta} = \frac{1}{n}\sum_{i=1}^{n}\theta_i \tag{1.5}$$

$$\bar{v}_t = \dot{p}_x \cos\bar{\theta} + \dot{p}_y \sin\bar{\theta} \tag{1.6}$$

Linklerin bağlandığı eklem noktalarında birbirlerine uyguladıkları kısıtlamalar söz konusudur. *i.* eklemdeki, *i.* link ve *i+1.* link arasındaki bağlantı için bu holonomik kısıtlamalar eşitlik (1.7)'deki gibi yazılabilir.

$$x_{i+1} - x_i = l_i \cos\theta_i + l_{i+1} \cos\theta_{i+1}$$
$$y_{i+1} - y_i = l_i \sin\theta_i + l_{i+1} \sin\theta_{i+1} \tag{1.7}$$

Bu kısıtlar, daha önceden tanımlanan matris notasyonları kullanılarak robotun bütün eklemleri için eşitlik (1.8)'deki gibi matris formunda yazılabilir.

$$\mathbf{DX} + \mathbf{AL}\cos\boldsymbol{\theta} = \mathbf{0}$$
$$\mathbf{DY} + \mathbf{AL}\sin\boldsymbol{\theta} = \mathbf{0} \tag{1.8}$$

Robotun link pozisyonları eşitlik (1.4) ve eşitlik (1.8) kullanılarak robotun link açıları ve ağırlık merkezinin pozisyonları cinsinden eşitlik (1.9)'daki gibi yazılabilir.

$$\mathbf{X} = -\mathbf{N}\cos\boldsymbol{\theta} + \mathbf{e}p_x$$
$$\mathbf{Y} = -\mathbf{N}\sin\boldsymbol{\theta} + \mathbf{e}p_y \tag{1.9}$$

Bu eşitlikteki $\mathbf{N} = \mathbf{M}^{-1}\mathbf{D}^T(\mathbf{D}\mathbf{M}^{-1}\mathbf{D}^T)^{-1}\mathbf{AL}$ olarak ifade edilmektedir. Linklerin lineer hızları ise eşitlik (1.9)'un zamana göre türevi alınarak eşitlik (1.10)'daki gibi ifade edilebilir.

$$\dot{\mathbf{X}} = \mathbf{N}\mathbf{S}_\theta\dot{\boldsymbol{\theta}} + \mathbf{e}\dot{p}_x$$
$$\dot{\mathbf{Y}} = -\mathbf{N}\mathbf{C}_\theta\dot{\boldsymbol{\theta}} + \mathbf{e}\dot{p}_y \tag{1.10}$$

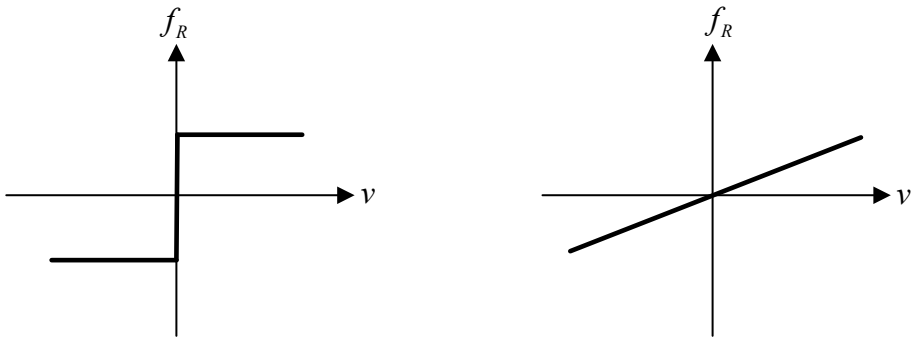
1.4.2. Zemin Sürtünme Modeli

Yılan robotlarla ilgili yapılan ilk çalışmalar biyolojik bir yılanın hareketini anlamaya yönelik analitik ve deneysel çalışmalardır. Yılan hareketi ilk kez biyolog Gray (1946) tarafından ele alınarak bir yılana etki eden kuvvetlerin matematiksel ifadeleri geliştirilmiş ve yılan hareketinin özellikleri ortaya konulmuştur. Bu çalışmanın önemli bir sonucu, yılanın ileri yönde düzlemsel hareketi için gövdesine normal yönde dış kuvvetlerin etki etmesinin gerekliliğidir. Diğer bir önemli çalışmada, Hu vd. (2009), yılan derisinin sürtünme özelliklerini deneysel olarak inceleyerek yılanın normal yöndeki sürtünme

katsayısının, teğet yöndeki sürtünme katsayısından daha büyük olduğunu belirtmişlerdir. İzotropik olmayan sürtünme olarak adlandırılan bu özellik, ileri yönde etkili bir hareket sağlayabilmek için yılan robot modellerinde kullanılmaktadır. Pratikte bu tür koşullar, genellikle yılan robotun gövdesi boyunca pasif tekerleklerin yerleştirilmesiyle elde edilir (Hirose, 1993; Miller, 2002; Mori ve Hirose, 2002; Yamada vd., 2005). Fakat her linkin aktif tekerleklerle donatıldığı (Onho ve Hirose, 2001; Saito vd., 2002) veya gövde boyunca bant, palet benzeri yapılar kullanılarak tüm vücudun tahrik edildiği (Kimura ve Hirose, 2002; Wright vd., 2007; Yamada ve Hirose, 2006; 2008; 2009) robot prototipleri de mevcuttur. Tekerlekli yılan robotlarda, tekerlekler sürtünmeyi azalttığı için robot ile zemin arasındaki etkileşimin dinamikleri genellikle göz ardı edilmektedir ve robotun sadece kinematik modeli çoğu durumda yeterli olmaktadır. Fakat tekerleksiz yılan robotlarda, robot ile zemin arasındaki sürtünme, yılan robotun hareketini önemli bir miktarda etkilemektedir. Bu nedenle bu robotlarda, özellikle yanal dalgalanma gibi yürüyüş biçimleri için sürtünme özelliklerinin göz önüne alındığı dinamik modele ihtiyaç duyulmaktadır.

Zemin sürtünme modeli olarak literatürde genellikle Coulomb ve viskoz sürtünme modelleri kullanılmaktadır. Coulomb sürtünme modelinde linke uygulanan sürtünme kuvvetinin linkin ağırlığıyla orantılı olduğu varsayılırken viskoz sürtünme modelinde ise sürtünme kuvveti linkin hızıyla orantılıdır. Bu modellerden Coulomb sürtünme modeli fiziksel açıdan daha doğru bir yaklaşım sunmasına rağmen viskoz sürtünme modeline göre daha karmaşık olması sebebiyle denetleyici tasarımı ve analiz amaçları için viskoz sürtünme modelinin kullanılmasını daha uygundur (Liljebäck vd., 2012b). Coulomb ve viskoz sürtünme modellerinin karakteristikleri sırasıyla Şekil 1.2 (a)'da ve Şekil 1.2 (b)'de görülmektedir.

Tekerlekli yılan robotlarda linkin zemin ile temasının tek noktadan olduğu varsayımı yapılarak sürtünme kuvvetinin her bir linkin sadece ağırlık merkezine etki ettiği varsayılır. Tekerleksiz yılan robotun sürtünme modelinde ise dağıtık temas modelinin kullanılması daha doğru bir yaklaşımdır (Mehta vd., 2008).



(a) (b)

Şekil 1.2. Sürtünme modellerinin grafikleri (a) Coulomb sürtünmesi (b)Viskoz sürtünmesi

Bu çalışmada, dağıtık temas modeline dayanan izotropik olmayan viskoz sürtünme modeli kullanılmıştır. Buna göre, *i*. linke etki eden viskoz sürtünme kuvveti lokal koordinat ekseninde eşitlik (1.11)'deki gibi tanımlanabilir.

$$\mathbf{f}_{R,i}^{link,i} = -m_i \begin{bmatrix} c_{t,i} & 0 \\ 0 & c_{n,i} \end{bmatrix} \mathbf{v}_i^{link,i} \tag{1.11}$$

Bu eşitlikteki, $c_{t,i}$ ve $c_{n,i}$ terimleri sırasıyla *i*. linke teğet ve normal yönde etki eden sürtünme katsayılarını ifade ederken $v_i^{link,i}$ ise *i*. linkin lokal koordinat eksenindeki hızını ifade etmektedir.

Eşitlik (1.11)'de lokal koordinat ekseninde ifade edilen *i*. linke etki eden viskoz sürtünme kuvveti, global koordinat ekseninde eşitlik (1.3)'te verilen rotasyon matrisi kullanılarak eşitlik (1.12)'deki gibi ifade edilebilir.

$$\begin{aligned} \mathbf{f}_{R,i} &= \mathbf{f}_{R,i}^{global} = \mathbf{R}_{link,i}^{global} \mathbf{f}_{R,i}^{link,i} \\ &= -\mathbf{R}_{link,i}^{global} \, m_i \begin{bmatrix} c_{t,i} & 0 \\ 0 & c_{n,i} \end{bmatrix} (\mathbf{R}_{link,i}^{global})^T \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \end{bmatrix} \end{aligned} \tag{1.12}$$

Global koordinat ekseninde bütün linklere etki eden viskoz sürtünme kuvvetinin matris formundaki ifadesi eşitlik (1.13)'te verilmiştir.

$$\mathbf{f}_R = \begin{bmatrix} \mathbf{f}_{R,x} \\ \mathbf{f}_{R,\mathrm{y}} \end{bmatrix} = -\begin{bmatrix} \mathbf{C}_\theta & -\mathbf{S}_\theta \\ \mathbf{S}_\theta & \mathbf{C}_\theta \end{bmatrix} \begin{bmatrix} \mathbf{C}_t\mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_n\mathbf{M} \end{bmatrix} \begin{bmatrix} \mathbf{C}_\theta & \mathbf{S}_\theta \\ -\mathbf{S}_\theta & \mathbf{C}_\theta \end{bmatrix} \begin{bmatrix} \dot{\mathbf{X}} \\ \dot{\mathbf{Y}} \end{bmatrix} \tag{1.13}$$

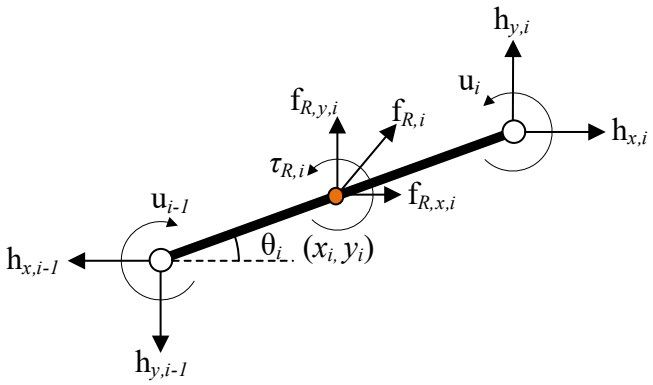
Bu eşitlikte $\mathbf{f}_{R,x} = [f_{\mathrm{R,x},1}, \ldots, f_{\mathrm{R,x,n}}]^{\mathrm{T}}$ ve $\mathbf{f}_{R,y} = [f_{\mathrm{R,y},1}, \ldots, f_{\mathrm{R,y,n}}]^{\mathrm{T}}$ global x ve y yönlerinde linklere etki eden sürtünme kuvvetlerini göstermektedir. $\mathbf{C}_t = \mathrm{diag}(c_{t,1},\ldots, c_{t,n})$ ve $\mathbf{C}_n = \mathrm{diag}(c_{n,1},\ldots, c_{n,n})$ ise her bir linkin teğet ve normal yönündeki sürtünme katsayılarından oluşan köşegen matrislerdir.

Tekerlekli yılan robotlardan farklı olarak tekerleksiz yılan robotlarda, robot ile zemin arasındaki temas kayma şeklinde olduğu için sürtünme torku önemlidir ve dinamik denkleme katılmalıdır (Mehta vd., 2008). Eşitlik (1.14)'te linkin ağırlık merkezine etki eden sürtünme tork matrisi verilmiştir.

$$\boldsymbol{\tau}_R = -\mathbf{C}_n\mathbf{J}\dot{\boldsymbol{\theta}} \tag{1.14}$$

1.4.3. Yılan Robotun Dinamik Modeli

Yılan robotun her bir linkine etkiyen kuvvet ve torklar Şekil 1.3'te gösterilmiştir. Bu şekilden görüldüğü gibi *i.* linke etki eden kuvvetler, linkin ağırlık merkezine etki eden sürtünme kuvveti ve *i*-1. ve *i*+1. linkten etki eden kısıtlama kuvvetleridir. Buna göre global koordinat ekseninde *i*. link için kuvvet denge eşitliği Newton'un ikinci hareket yasası kullanılarak eşitlik (1.15)'teki gibi yazılabilir.



Şekil 1.3. Yılan robotun her bir linkine etkiyen kuvvet ve torkların gösterimi

$$\begin{aligned} m_i \ddot{x}_i &= f_{R,\mathrm{x},i} + h_{\mathrm{x},i} - h_{\mathrm{x},i-1} \\ m_i \ddot{y}_i &= f_{R,y,i} + h_{y,i} - h_{y,i-1} \end{aligned} \tag{1.15}$$

Bütün linkler için kuvvet denge eşitliği matris formunda eşitlik (1.16)'da verilmiştir.

$$\begin{aligned} \mathbf{M}\ddot{\mathbf{X}} &= \mathbf{f}_{R,\mathrm{x}} + \mathbf{D}^T \mathbf{h}_x \\ \mathbf{M}\ddot{\mathbf{Y}} &= \mathbf{f}_{R,\mathrm{y}} + \mathbf{D}^T \mathbf{h}_y \end{aligned} \tag{1.16}$$

Bu eşitlikte $\mathbf{h}_\mathrm{x} = [h_{\mathrm{x},1}, \ldots, h_{\mathrm{x},\mathrm{n}}]^\mathrm{T}$ ve $\mathbf{h}_\mathrm{y} = [h_{\mathrm{y},1}, \ldots, h_{y,\mathrm{n}}]^\mathrm{T}$ olarak ifade edilmektedir. Bu eşitlikteki link ivmeleri ayrıca eşitlik (1.8)'de verilen link pozisyonlarının zamana göre ikinci türevinin alınmasıyla eşitlik (1.17)'deki gibi yazılabilir.

$$\begin{aligned} \mathbf{D}\ddot{\mathbf{X}} &= \mathbf{AL}(\mathbf{C}_\theta \dot{\boldsymbol{\theta}}^2 + \mathbf{S}_\theta \ddot{\boldsymbol{\theta}}) \\ \mathbf{D}\ddot{\mathbf{Y}} &= \mathbf{AL}(\mathbf{S}_\theta \dot{\boldsymbol{\theta}}^2 - \mathbf{C}_\theta \ddot{\boldsymbol{\theta}}) \end{aligned} \tag{1.17}$$

Robotun ivmesi ise eşitlik (1.4)'te verilen robotun ağırlık merkezi pozisyonunun zamana göre ikinci türevinin alınmasıyla eşitlik (1.18)'deki gibi elde edilir.

$$\begin{bmatrix} \ddot{p}_x \\ \ddot{p}_y \end{bmatrix} = \frac{1}{m} \begin{bmatrix} \mathbf{e}^T \mathbf{M}\ddot{\mathbf{X}} \\ \mathbf{e}^T \mathbf{M}\ddot{\mathbf{Y}} \end{bmatrix} \tag{1.18}$$

Bu eşitlikteki link ivmelerinin yerine eşitlik (1.16)'daki ifade konulduğu zaman link ivmelerinin toplanması ($\mathbf{e}^T \mathbf{D}^T = 0$) kısıtlama kuvvetlerinin etkisini ortadan kaldırır. Böylece yılan robotun ivmesi, robota etki eden sürtünme kuvvetleri toplamının robotun kütlesine oranı olarak eşitlik (1.19)'daki gibi ifade edilebilir.

$$\begin{bmatrix} \ddot{p}_x \\ \ddot{p}_y \end{bmatrix} = \frac{1}{m} \begin{bmatrix} \mathbf{e}^T \mathbf{f}_{R,x} \\ \mathbf{e}^T \mathbf{f}_{R,x} \end{bmatrix} = \frac{1}{m} \mathbf{E}^T \mathbf{f}_R \tag{1.19}$$

Şekil 1.3'te *i*. linke etkiyen torklar göz önüne alındığı zaman tork denge eşitliği *i*. link için eşitlik (1.20)'deki gibi yazılabilir. Her bir link için matris formdaki ifadesi ise eşitlik (1.21)'de verilmiştir.

$$J_i\ddot{\theta}_i = u_i - u_{i-1} + \tau_{R,i} - l_i \sin\theta_i(h_{\mathrm{x},i} + h_{\mathrm{x},i-1}) + l_i \cos\theta_i(h_{y,i} + h_{y,i-1}) \tag{1.20}$$

$$\mathbf{J}\ddot{\boldsymbol{\theta}} = \mathbf{D}^T\mathbf{u} + \boldsymbol{\tau}_R - \mathbf{S}_\theta\mathbf{L}\mathbf{A}^T\mathbf{h}_x + \mathbf{C}_\theta\mathbf{L}\mathbf{A}^T\mathbf{h}_y \tag{1.21}$$

Bu eşitlikteki $\mathbf{h}_x$ ve $\mathbf{h}_y$ kısıtlama kuvvetleri yerine eşitlik (1.16)'dan elde edilen eşitlik (1.22)'deki ifadeleri yazılırsa eşitlik (1.23) elde edilir.

$$\begin{aligned}\mathbf{h}_{\mathrm{x}} &= (\mathbf{D}\mathbf{M}^{-1}\mathbf{D}^T)^{-1}(\mathbf{A}\mathbf{L}(\mathbf{C}_\theta\dot{\boldsymbol{\theta}}^2 + \mathbf{S}_\theta\ddot{\boldsymbol{\theta}}) - \mathbf{D}\mathbf{M}^{-1}\mathbf{f}_{R,\mathrm{x}}) \\ \mathbf{h}_y &= (\mathbf{D}\mathbf{M}^{-1}\mathbf{D}^T)^{-1}(\mathbf{A}\mathbf{L}(\mathbf{S}_\theta\dot{\boldsymbol{\theta}}^2 - \mathbf{C}_\theta\ddot{\boldsymbol{\theta}}) - \mathbf{D}\mathbf{M}^{-1}\mathbf{f}_{R,y})\end{aligned} \tag{1.22}$$

$$\mathbf{M}_\theta\ddot{\boldsymbol{\theta}} + \mathbf{W}\dot{\boldsymbol{\theta}}^2 - \mathbf{S}_\theta\mathbf{N}^T\mathbf{f}_{R,x} + \mathbf{C}_\theta\mathbf{N}^T\mathbf{f}_{R,y} - \boldsymbol{\tau}_R = \mathbf{D}^T\mathbf{u} \tag{1.23}$$

Bu eşitlikte;

$$\mathbf{M}_\theta = \mathbf{J} + \mathbf{S}_\theta\mathbf{H}\mathbf{S}_\theta + \mathbf{C}_\theta\mathbf{H}\mathbf{C}_\theta$$

$$\mathbf{W} = \mathbf{S}_\theta\mathbf{H}\mathbf{C}_\theta - \mathbf{C}_\theta\mathbf{H}\mathbf{S}_\theta$$

$$\mathbf{H} = \mathbf{L}\mathbf{A}^T(\mathbf{D}\mathbf{M}^{-1}\mathbf{D}^T)^{-1}\mathbf{A}\mathbf{L}$$

Yılan robotun dinamik modeli, eşitlik (1.19) ve eşitlik (1.23)'ün birleştirilmesiyle matris formunda eşitlik (1.24)'teki gibi elde edilmektedir.

$$\begin{bmatrix}\mathbf{M}_\theta & \mathbf{0} \\ \mathbf{0} & m\mathbf{I}\end{bmatrix}\begin{bmatrix}\ddot{\boldsymbol{\theta}} \\ \ddot{\mathbf{p}}\end{bmatrix} + \begin{bmatrix}\mathbf{W}\dot{\boldsymbol{\theta}}^2 \\ \mathbf{0}\end{bmatrix} - \mathbf{C}\begin{bmatrix}\dot{\boldsymbol{\theta}} \\ \dot{\mathbf{p}}\end{bmatrix} = \begin{bmatrix}\mathbf{D}^T \\ \mathbf{0}\end{bmatrix}\mathbf{u} \tag{1.24}$$

Bu eşitlikte;

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_n\mathbf{J} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathcal{I}^T \\ \mathbf{E}^T \end{bmatrix} \mathbf{R} \begin{bmatrix} \mathbf{C}_t\mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_n\mathbf{M} \end{bmatrix} \mathbf{R}^T \begin{bmatrix} \mathcal{I} & \mathbf{E} \end{bmatrix}$$

$$\mathcal{I} = \begin{bmatrix} \mathbf{S}_\theta\mathbf{N}^{\mathrm{T}} & -\mathbf{C}_\theta\mathbf{N}^{\mathrm{T}} \end{bmatrix}^T$$

1.5. Yılan Robotun Enerji Verimli Hareketi

Biyolojik yılanlar çim, ahşap, kum, bataklık, cam veya su gibi çok çeşitli ortamlarda eşsiz bir şekilde hareket edebilme kabiliyetine sahiptirler. Sürtünme kuvvetlerindeki değişiklikler yılanın hızını doğrudan etkilediğinden, farklı zemin koşullarında etkin hareket kabiliyetlerini korumak için ya hareket biçimini değiştirirler ya da aynı hareket biçimini koruyarak sadece hareketin kıvrım açısını modüle ederler (Ryu vd., 2010). Benzer şekilde yılan robotların farklı sürtünmelere sahip yüzeylerde enerji verimli hareketleri için yürüyüş parametreleri ayarlanmalıdır.

Yılan robotlar, bilinmeyen, düzensiz ortamlarda tekerlekli, paletli veya bacaklı diğer mobil robotlara göre daha üstün hareket kabiliyeti sağlamalarına rağmen sürtünme bağımlı hareketlerinden dolayı bu mobil robotlara kıyasla daha az hareket verimliliğine sahiptirler. Bu robotların hareket kabiliyetlerini etkili bir şekilde kullanabilmeleri için hareket verimliliği çözülmesi gereken önemli bir sorundur. Ayrıca bu mekanizmaların gerçek dünyadaki uygulamaların çoğunda kullanılabilme potansiyeline sahip olması yılan robotun özellikle bu tür uygulamalar için enerji tasarruflu hareketinin önemini arttırmaktadır. Buna rağmen, literatürde sürtünme katsayılarının optimal parametreler üzerindeki etkisini ele alan çalışmalar sınırlıdır.

Crespi ve Ijspeert (2008), türev içermeyen klasik bir yöntem olan Powell optimizasyon yöntemini, CPG'nin parametrelerini optimize ederek belirli bir ortamda robotun ileri hızını maksimize etmek için kullanmışlardır. Bu çalışmada sadece yürüyüş biçiminin genlik ve faz farkı parametreleri ele alınmış olup frekans parametresinin hareket üzerine etkisi incelenmemiştir. Deneysel çalışmalarda robotun yüzme hareketi ve yatay düzlemdeki sürünme hareketi ele alınmış olup simülasyon çalışmalarında ise yatay ve eğimli düzlemlerdeki sürünme hareketi incelenmiştir. Bu çalışmanın sonuçlarına göre

verimli bir hareket için eğimli bir yüzeyden yukarı doğru çıkarken genliğin büyük, aşağı doğru inerken ise çok daha küçük olması gerekmektedir. Ayrıca yavaş bir yüzme hareketinin yavaş bir sürünme hareketinden daha küçük bir faz gecikmesi gerektirdiği sonucuna varılmıştır. Bu çalışmanın önemli bir dezavantajı kullanılan optimizasyon yönteminin lokal optimizasyona takılma potansiyelinin olmasıdır.

Inoue vd. (2007)'nin çalışmasında ise yılan robotunun adaptif hareketi için, CPG tabanlı bir denetleyicinin parametreleri değişen zemin sürtünme durumuna göre genetik algoritma (Genetic Algorithm, GA) kullanılarak elde edilmiştir. Bu çalışmada önerilen yöntemin geçerliliği yanal dalgalanma hareketi için sadece teğet sürtünme katsayıları aynı olan üç farklı ortamda test edilmiştir.

CPG parametrelerinin GA ile ayarlandığı bir başka çalışmada pasif tekerlekli bir yılan robotun farklı sürtünme katsayılı ortamlara adaptasyonu aynı yürüyüş biçimi içinde vücut şeklinin genliği değiştirilerek ele alınmıştır (Kousuke vd., 2007). Amaç fonksiyonu, tüketilen enerji ve robotun hızından oluşan tek bir fonksiyondur.

GA ayrıca Hasanzadeh ve Tootoonchi (2008)'de CPG parametrelerini ve bağlantı ağırlıklarını yılan robotunun hızı açısından optimize etmek için kullanılmıştır. Ayrıca bu çalışmada çevre koşulları ile optimal CPG parametreleri arasında elde edilen ilişkiye göre tasarlanan bulanık mantık denetleyici kullanılarak CPG parametreleri değişen çevreye göre ayarlanmıştır. Aynı yazarların (2010)'daki diğer bir çalışmasında ise yılan robotun potansiyel uygulamaları için yeni bir hareket önerilerek optimal parametreleri GA ile bulunmuş ve bu hareket parametreleri ile zemin sürtünme katsayıları arasındaki ilişki incelenmiştir. Bu çalışmada yılan robotun hızı optimize edilirken kafa modülünün hareket boyunca kabul edilebilir açılar arasında kalması göz önünde bulundurulmuş fakat enerji tüketimi dikkate alınmamıştır.

Wu ve Ma (2010b)'nın çalışmasında CPG parametreleri ve hareket verimliliği arasındaki ilişki farklı sürtünme ve farklı eğime sahip çevrelerde incelenmiştir. Robotun ileri yöndeki yer değiştirmesi ile enerji tüketimi arasındaki oran optimal hareket için kriter olarak kullanılmıştır.

Ariizumi ve Matsuno (2017) yılan robotun yanal dalgalanma, yanal kayma ve sinüs kaldırma hareketlerini içeren üç farklı yürüyüşünü sekiz farklı ortamda robotun hareket hızı ve enerji verimliliği arasındaki denge açısından analiz etmişlerdir. Farklı sürtünme katsayıları ve hareket biçimleri için hareket hızı ile verimliliği arasındaki Pareto eğrileri belirli bir değer aralığındaki genlik ve frekans parametrelerinin 1500 rastgele

kombinasyonu ile elde edilmiştir. Simülasyon çalışmaları, yanal kayma ve sinüs kaldırma hareketlerinin genellikle yanal dalgalanma hareketine göre daha verimli olduğunu ortaya koymuştur. Normal yöndeki sürtünme katsayısının teğet yöndeki sürtünme katsayısından büyük olmasıyla elde edilen izotropik olmayan sürtünme yeterince büyük ise sinüs kaldırma hareketi bu üç hareketten en verimlisiyken, eğer bu sürtünme yeterince büyük değilse yanal kayma hareketi en verimli harekettir. Her iki yöndeki sürtünmenin küçük olması durumunda ise robotun hareket hızı arttıkça en verimli hareket yanal dalgalanma hareketi olmuştur.

Kelasidi vd. (2015), sualtı yılan robotları için hareketi tanımlayan genlik, frekans ve faz farkı parametrelerinden ikisini sabit tutup diğerini değiştirerek tüketilen enerjiyi ve robotun hareket hızını her bir parametre için incelenmişlerdir. Ayrıca bu çalışmada belirli bir kütleye sahip bir sistemi belirli bir mesafeye hareket ettirmek için gerekli enerjinin miktarını gösteren yer değiştirme maliyeti (Cost of Transportation, COT) olarak tanımlanan indeks, robotun enerji verimli hareketini tanımlamak için kullanılmıştır. Aynı yazarlar tarafından bu problem (2016)'daki bir diğer çalışmada parçacık sürü optimizasyonu (Particle Swarm Optimization, PSO) algoritması kullanılarak ele alınmıştır. Bu çalışmada güç tüketimi ve robotun hızı aynı amaç fonksiyonunda ağırlıklı toplam metoduyla birleştirilmiş ve her bir ağırlık için PSO algoritması tekrar tekrar çalıştırılarak Pareto optimal çözümler elde edilmiştir. Sualtı yılan robotunun yanal dalgalanma ve yılan balığı benzeri hareketi olmak üzere iki farklı yürüyüş hareketi için optimal yürüyüş parametreleri elde edilmiştir. Önerilen bu yöntemin genişletilmiş bir hali kara ve sualtı yılan robotları için (Kelasidi vd., 2018)'de sunulmuştur.

Cao vd. (2019), hareket verimliliğini optimize etmek için CPG denetleyici tarafından kontrol edilen bir yılan robotun optimum hareket parametrelerini robotun hızını ve enerji tüketimini dikkate alarak araştırmışlardır. Farklı hareket genişliğine ve teğet ile normal yöndeki sürtünme katsayılarının farklı oranlarına sahip ortamlar için robotun hareket parametreleri guguk kuşu arama (Cuckoo Search, CS) algoritması ile optimize edilmiştir. Optimizasyonun amacı robotun yer değiştirmesinin tükettiği enerjiye bölünmesiyle elde edilen hareket verimliliğinin maksimize edilmesidir.

Yılan robotun enerji verimli hareketindeki amaç, robotun hızını maksimize ederken enerji tüketimini minimumda tutmaktır. Literatürdeki çalışmalarda yılan robotun optimal hareketi genellikle sadece robotun hızı göz önüne alınarak ele alınmış veya robotun hızı ve enerji tüketimi tek bir amaç fonksiyonunda birleştirilerek formülize edilmiştir. Fakat

robotun enerji tüketimi, robotun hızı ile arttığı için bu iki amaç birbiriyle çelişmektedir. Bu nedenle de amaçlar arasındaki Pareto optimal çözüm setini elde etmek için yılan robotun enerji verimli hareket probleminin MO problemi olarak ele alınması gerekmektedir.

MO problemlerini çözmek için aynı anda bir dizi çözüm üretebilen popülasyon tabanlı evrimsel hesaplama (Evolutionary Computation, EC) teknikleri daha elverişlidir (Xue et al., 2016). Farklı optimizasyon problemleri için birçok farklı EC tekniği vardır. Bu teknikler arasında, bir ekosistemdeki organizmalar arasındaki simbiyotik etkileşim stratejilerini simüle eden SOS algoritması, güç sistemlerinin optimizasyonu (Duman, 2016), ulaşım (Vincent et al., 2016), makine öğrenmesi (Nanda and Jonwal, 2017) ve optimal denetleyici tasarımı (Zhou et al., 2019) gibi pek çok farklı alandaki optimizasyon problemleri için daha hızlı yakınsama hızı ile daha kararlı sonuçlar sunmasıyla öne çıkmaktadır (Abdullahi vd., 2019). SOS algoritmasının diğer EC algoritmalarına göre üstünlüğü, çözümlerin güncellenmesi için kullanılan üç aşamalı stratejisinden (mutualizm, kommensalizm ve parazitizm) kaynaklanmaktadır. Bu aşamalardan mutualizm ve kommensalizm, keşfe odaklanarak algoritmanın yeni çözüm üretme yeteneğine katkıda bulunur. İki aşamalı bu keşif, algoritmanın en iyi çözümün bulunduğu bölgeye odaklanmasını sağlar. SOS algoritmasının üçüncü aşaması olan parazitizm ise rastgele boyutlu mutasyon işlevini kullanarak algoritmanın lokal optimuma takılmasını önler (Han vd., 2019). SOS algoritmasının bir diğer avantajı ise diğer meta-sezgisel algoritmaların çoğundan farklı olarak herhangi bir özel algoritma parametresine sahip olmamasıdır. Böylece bu parametrelerin düzgün ayarlanamamasından kaynaklanan lokal optimal çözümlere takılma sorunu ortadan kalkar. Algoritmanın bu özelliği aynı zamanda hesaplama süresinin azaltılmasını da sağlar (Tran vd., 2016). Literatürdeki çalışmalardan SOS algoritmasının ayrıca MO problemlerine başarılı bir şekilde uygulandığı diğer çok amaçlı optimizasyon algoritmalarıyla karşılaştırılarak ortaya konulmuştur (Tran vd., 2016; Dosoglu vd., 2018)

Hareket optimizasyonu için türev tabanlı algoritmalar daha hızlı olmalarına rağmen lokal optimizasyona takılma problemlerinden dolayı türev içermeyen ve stokastik yöntemlerin daha uygun bir seçim olduğu belirtilmektedir (Wiens ve Nahon, 2012). Bu nedenle bu çalışmada yılan robotun optimal hareket parametrelerini bulmak için FNSMOSOS algoritması önerilmiştir. Bu algoritma, SOS algoritmasının yukarda bahsedilen bütün avantajlarını, çözümü Pareto optimal çözüme mümkün olduğunca yakın üreten hızlı baskın olmayan sıralama (Fast Non-dominated Sorting, FNS) tekniği ve

çözümde çeşitlilik sağlayan kalabalık mesafe (Crowding Distance, CD) tekniği gibi iki önemli teknik ile birleştirdiği için hedefler arasında optimal dengeyi bulmada oldukça etkili bir optimizasyon algoritmasıdır. Bu çalışmada ayrıca ağırlıklı toplam yöntemine dayanan MOSOS algoritmasının yılan robotun optimal hareketi problemine uygulanması ele alınmış olup elde edilen sonuçlar FNSMOSOS algoritmasıyla karşılaştırılmıştır. SOS algoritmasına dayanan çok amaçlı bu iki algoritmanın detayları aşağıda açıklanmıştır.

1.5.1. SOS Algoritması

SOS algoritması, ekosistemdeki herhangi iki farklı tür organizma arasındaki simbiyotik ilişkilerden esinlenerek geliştirilen meta-sezgisel bir algoritmadır (Cheng ve Prayago, 2014). Bu algoritma, bir organizmanın ilişki içinde olabileceği en uygun organizmayı aramak için iki organizma arasındaki simbiyotik ilişkileri simüle eder. Diğer popülasyon tabanlı algoritmalarda olduğu gibi, SOS algoritması da optimal global çözümü ararken arama uzayını oluşturan aday çözümleri iteratif olarak kullanır ve ekosistem olarak adlandırılan rastgele üretilen başlangıç bir popülasyon ile başlar. Her organizma bir aday çözümü temsil eder ve istenen amaca uyum derecesini yansıtan belirli bir uygunluk değerine sahiptir. SOS algoritmasında yeni çözümler, gerçek dünyada en çok kullanılan biyolojik etkileşimler olan mutualizm, kommensalizm ve parazitzm simbiyotik ilişkilerindeki etkileşimin türü göz önüne alınarak üretilir. Ekosistemdeki her organizma sırasıyla bu üç aşamada rastgele olarak seçilmiş bir organizma ile etkileşime girer ve bu etkileşim sonucunda üretilen yeni organizmaların uygunluk değerleri önceki organizmaların uygunluk değerleriyle karşılaştırılarak organizmalar güncellenir. Bu süreç durdurma kriteri (maksimum iterasyona ulaşma) sağlanana kadar devam eder.

SOS algoritması, GA'daki çaprazlama, mutasyon oranı, PSO'daki atalet ağırlığı, bilişsel ve sosyal faktör gibi algoritmaya özgü herhangi bir parametreye sahip olmayıp sadece popülasyon büyüklüğü ve maksimum iterasyon sayısı gibi her algoritmada olan iki tane genel parametreye sahiptir.

1.5.1.1. Mutualizm Aşaması

Mutualizm, iki farklı türün karşılıklı olarak fayda sağladığı simbiyotik ilişkidir. Bu aşamada, X_i ekosistemdeki *i.* organizma, X_j ise X_i ile etkileşime girecek olan ekosistemden rastgele olarak seçilmiş diğer bir organizma olmak üzere bu iki organizma ekosistemdeki canlılıklarını sürdürme olasılıklarını karşılıklı olarak arttırmak amacıyla etkileşime girer. X_i ve X_j için yeni aday çözümler, bu iki organizma arasındaki mutualizm ilişkisine göre modellenen eşitlik (1.25) ve eşitlik (1.26) kullanılarak hesaplanır.

$$X_i^{'} = X_i + rand(0,1) * (X_{best} - MV * BF_1) \tag{1.25}$$

$$X_j^{'} = X_j + rand(0,1) * (X_{best} - MV * BF_2) \tag{1.26}$$

Bu eşitliklerdeki BF_1 ve BF_2 terimleri, aralarında mutualizm ilişkisi bulunan organizmaların edindikleri faydanın ölçüsünü belirlemek için kullanılırlar ve 1 veya 2 olarak rastgele olarak belirlenirler. X_{best} ekosistemdeki en yüksek uygunluk değerine sahip organizmayı temsil ederken eşitlik (1.27)'deki gibi hesaplanan MV ortak (mutual) vektörü ise X_i ve X_j organizmaları arasındaki ilişkinin özelliğini temsil eden bir vektördür.

$$MV = \frac{X_i + X_j}{2} \tag{1.27}$$

1.5.1.2. Kommensalizm Aşaması

SOS algoritmasının bu aşamasında, birlikte yaşayan iki türden birinin diğerine zarar vermeden ondan yararlandığı simbiyotik ilişki olan kommensalizme göre modelleme yapılmıştır. Mutualizm aşamasındaki gibi X_j, X_i ile etkileşime girecek olan ekosistemden rastgele olarak seçilmiş bir organizma olmak üzere bu iki organizmanın beraberliğinden sadece X_i fayda görür. X_j ise etkilenmez. Buna göre, sadece X_i için yeni aday çözümler, eşitlik (1.28) kullanılarak hesaplanır.

$$X_i^{'} = X_i + rand(-1,1) * (X_{best} - X_j) \tag{1.28}$$

1.5.1.3. Parazitizm Aşaması

Parazitizm iki türden birinin diğerine zarar vererek yarar sağladığı simbiyotik ilişki biçimidir. Bu aşamada, X_i organizması parazit vektör olarak adlandırılan yapay parazitin oluşturulmasında kullanılır. Parazit vektör oluşturulurken öncelikle X_i çoğaltılır ve daha sonra arama uzayının sınırları içerisinde rastgele olarak modifiye edilir. X_j organizması ise önceki aşamalarda olduğu gibi ekosistemden rastgele olarak seçilir ve parazitin konağı olarak kullanılır. Uygunluk değerleri belirlenen bu iki organizmadan parazit vektör daha iyi ise X_j'yi ortadan kaldırarak ekosistemdeki yerini alır. Aksi taktirde, X_j bu parazite karşı bağışıklık kazanır ve parazit vektör ekosistemde daha fazla yaşayamaz.

1.5.2. Ağırlıklı Toplam Tabanlı MOSOS Algoritması

Bu yöntemde, problemin her bir amacı tek bir amaç fonksiyonunda birleştirilerek çok amaçlı optimizasyon problemi tek amaçlı bir probleme dönüştürülür (Coello, 1999). Her bir amaç önceden belirlenen bir ağırlık katsayısı ile çarpılarak toplanır. Amaçların ağırlık katsayılarının toplamı 1 olmalıdır. m_n tane amaca sahip bir optimizasyon problemi için ağırlıklı toplam yöntemi eşitlik (1.29) ve eşitlik (1.30)'daki gibi formülize edilebilir.

$$f(x) = w_1 f_1(x) + w_2 f_2(x) ... + w_{m_n} f_{m_n}(x) \tag{1.29}$$

$$\sum_{i=1}^{m_n} w_i = 1 \text{ ve } w_i > 0 \tag{1.30}$$

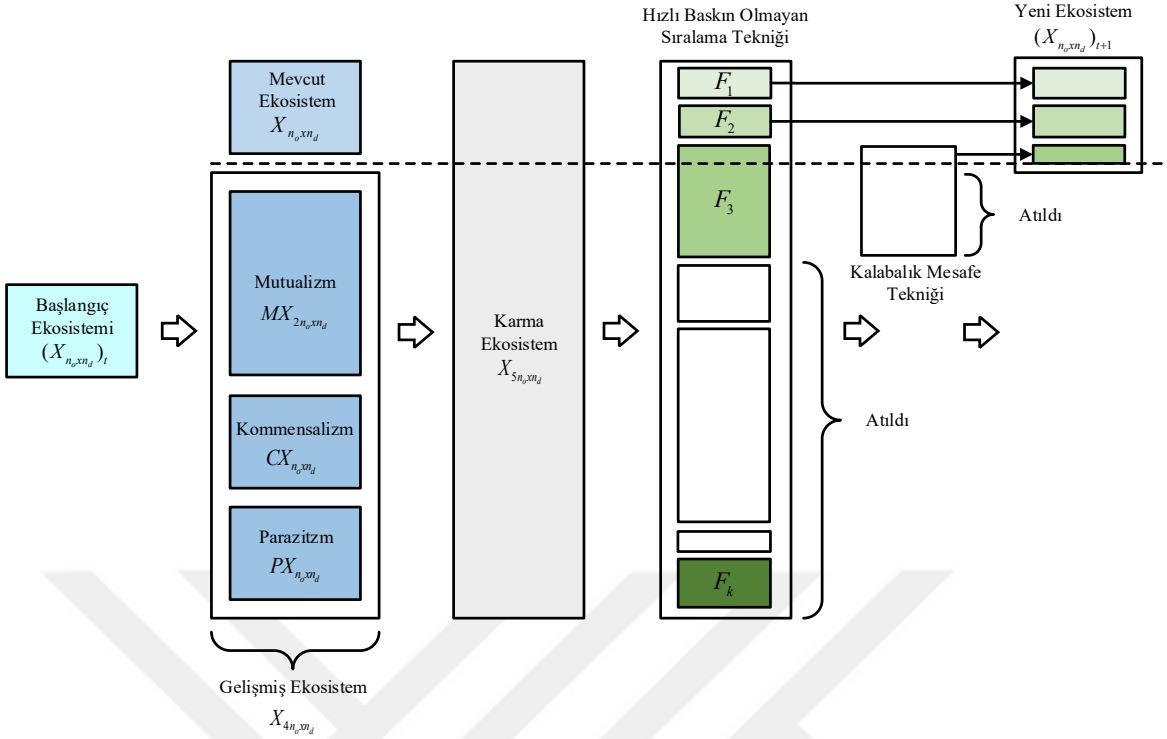
Bu yaklaşımın uygulanması basit olmasına rağmen başarısı amaçlara verilen ağırlık katsayılarının doğru ayarlanmasıyla oldukça ilişkilidir ve bu katsayıların düzgün dağılımı her zaman düzgün dağılımlı bir Pareto cephesi elde etmeyi garanti etmemektedir. Ayrıca bu yaklaşım ile Pareto optimal çözümlerin bulunabilmesi için algoritmanın farklı ağırlıklarla birden çok kez çalıştırılması gerekir. Bu da zaman alıcı ve zahmetli bir süreçtir. Bu yaklaşımın diğer bir önemli dezavantajı ise karma optimizasyon (min-max) problemlerinde amaçların aynı tipe dönüştürülmesi gerekliliğidir.

Ağırlıklı toplam tabanlı MOSOS algoritmasının amaç fonksiyonu oluşturma dışında geri kalan aşamaları klasik SOS algoritmasıyla aynıdır.

1.5.3. Hızlı Baskın Olmayan Sıralamalı MOSOS (FNSMOSOS) Algoritması

MO problemlerinde kullanılan diğer bir yaklaşım birbirine baskın olmayan çözümleri içeren Pareto optimal çözüm kümesini bulmaktır. FNSMOSOS algoritması, klasik SOS algoritmasının aşamalarına ek olarak seçkinliği korumak ve çeşitliliği sürdürmek için FNS ve CD tekniklerini kullanan çok amaçlı meta-sezgisel bir optimizasyon algoritmasıdır. Önerilen bu yöntem, genel hatlarıyla Tran vd. (2016)'nin önerdiği MOSOS algoritmasına dayanmaktadır. Farklı olarak sadece kalabalık entropi mesafe tekniği yerine Deb vd. (2002)'nin baskın olmayan sıralamalı genetik algoritma II (Non-dominated Sorting Genetic Algorithm II, NSGA-II) algoritmasında kullanmış oldukları CD tekniği kullanılmıştır.

Klasik SOS algoritmasında olduğu gibi FNSMOSOS algoritmasında da ilk olarak başlangıç ekosistemi (n_o) oluşturulur ve n_d boyutlu her organizmanın her bir amaç için uygunluk değerleri belirlenerek FNS ve CD teknikleri başlangıç ekosistemi için uygulanır. Bu tekniklerin uygulanmasından sonra oluşan birinci cepheden, en iyi organizma X_{best} olarak rastgele bir organizma seçilir ve klasik SOS algoritmasının üç aşaması sırayla uygulanarak yeni organizmalar üretilir. Bu aşamaların uygulanmasındaki tek farklılık, yeni organizmaların uygunluk değerinin önceki organizmalarla karşılaştırılması sonucunda daha iyi olan organizmaların mevcut ekosistemde tutularak diğerlerinin gelişmiş ekosistem olarak adlandırılan ayrı bir ekosisteme taşınmasıdır. Elde edilen mevcut ekosistem (n_o) ile gelişmiş ekosistem ($4n_o$) birleştirilerek karma ekosistem ($5n_o$) oluşturulur. Optimizasyon işlemi boyunca ekosistem büyüklüğünün başlangıç ekosistemine eşit kalması gerektiğinden yeni ekosistemi oluşturmak için en iyi n_o organizma karma ekosistemden seçilir. Bu seçim için öncelikle karma ekosisteme FNS tekniği uygulanarak baskılanmayan çözüm kümeleri (F_1, F_2,… F_k) belirlenir. Daha sonra en iyi organizmaların bulunduğu birinci cepheden başlanarak ekosistem büyüklüğüne ulaşılıncaya kadar her cephe sırayla yeni ekosisteme eklenir. Herhangi bir cephenin bütün olarak eklenmesi ekosistem boyutunun aşılmasına neden olursa, bu cephedeki organizmalar CD tekniği kullanılarak sıralanır ve ekosistemi doldurmak için gereken sayıda en iyi organizma seçilerek yeni ekosisteme eklenir. Böylece, bir sonraki iterasyonda kullanılmak üzere n_o büyüklüğünde yeni ekosistem elde edilir. Ekosistem seçim prosedürü Şekil 1.4'te gösterilmektedir.



Şekil 1.4. Ekosistem seçim prosedürü

1.5.3.1. Hızlı Baskın Olmayan Sıralama Tekniği

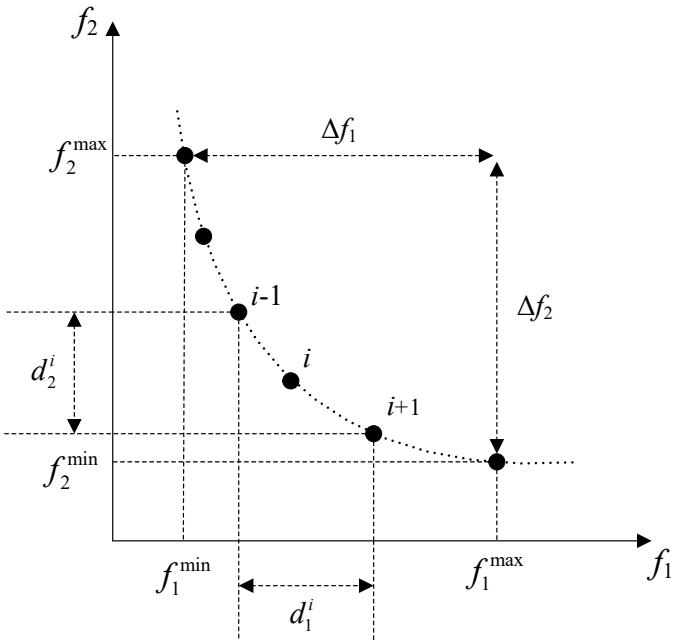
FNS tekniği ile ekosistemdeki organizmaların (çözümlerin) cephesi belirlenir. Bunun için öncelikle her organizmanın uygunluk değerleri ekosistemdeki diğer organizmalarla karşılaştırılarak her organizma için baskınlık sayısı (n_p) ve baskın olduğu çözüm kümesi (S_p) bulunur. n_p, p organizmasına baskın olan organizmaların sayısını belirtirken, S_p ise p organizmasının baskın olduğu organizmalardan oluşan kümeyi tanımlar. $n_p = 0$ olan organizmalar Pareto cephesi olarak bilinen birinci cepheye (F_1) atanır. Bu cephedeki organizmalar ekosistemdeki diğer organizmalar tarafından baskılanmayan, yani en azından bir amaç için diğerlerinden daha iyi olan organizmalardır. Diğer cephelerdeki organizmaları bulmak için Pareto cephesine ait her organizmanın S_p kümesindeki her bir q organizmasının baskınlık sayısı bir azaltılır ve $n_q = 0$ olan organizmalar ikinci cepheye (F_2) atanır. Bu işlem, ekosistemdeki her organizma bir cepheye atanıncaya kadar devam eder. FNS tekniği hakkında daha detaylı bilgi edinmek için Deb vd. (2002)'nin çalışmasına bakınız.

1.5.3.2. Kalabalık Mesafe Tekniği

Bir cephedeki organizmaların CD değerlerini hesaplamak için, öncelikle bütün organizmalar her bir amaç fonksiyonuna göre küçükten büyüğe doğru sıralanır. Daha sonra cephenin sınırlarını oluşturan en küçük (i=1) ve en büyük (i=l) organizmaların CD değerlerine sonsuz bir değer atanır ve arada kalan diğer organizmaların (i= 2,..,l-1) CD değerleri ise kendisine komşu iki organizmanın (i-1 ve i+1) uygunluk değerleri arasındaki farkın normalize edilmesiyle eşitlik (1.31)'deki gibi hesaplanır. Bu hesaplama her amaç fonksiyonu j (j=1,2,…,m) için ayrı ayrı yapılır ve i. organizma için toplam CD değeri, her bir amaç fonksiyonu için bulunan bireysel CD değerlerinin toplanmasıyla eşitlik (1.32)'deki gibi elde edilir. İki amaç fonksiyonuna sahip bir problem için, i. organizmanın CD değerinin hesaplanması Şekil 1.5'te gösterilmiştir.

$$\Delta_j^i = \frac{d_j^i}{\Delta f_j} = \frac{f_j^{i+1} - f_j^{i-1}}{f_j^{\max} - f_j^{\min}} \tag{1.31}$$

$$\Delta_i = \Delta_1^i + \Delta_2^i + \ldots + \Delta_m^i \tag{1.32}$$



Şekil 1.5. i. organizma için CD değerinin hesaplanması

1.6. Öğrenme Tabanlı Kontrol

Akıllı sistemlerin her geçen gün yaygın hale gelmesiyle, bu sistemlerin kontrolü ve çevreden haberdar olmaları önem kazanmıştır. Çevreyle etkileşimli olarak peş peşe birbirini etkileyen kararlar alabilen sistemler geliştirebilmek için makine öğrenmesi yöntemleri arasında pekiştirmeli öğrenme (Reinforcement Learning, RL) oldukça önemli bir yere sahiptir. Fakat gerçek dünya problemlerinin çoğunun yüksek boyutlu durum uzayına ve genellikle sürekli bir aksiyon uzayına sahip olması RL algoritmalarının bu tür karmaşık problemler için yetersiz kalmasına sebep olmaktadır. Bu nedenle zaman serileri, resim veya video gibi yüksek boyutlu sensör girişleri için oldukça uygun olan ve özellikle bilgisayar görüsü (Krizhevsky vd., 2012) ve doğal dil işleme (Dahl vd., 2011) problemlerinde gösterdiği başarıdan dolayı ilgi çeken derin öğrenme (Deep Learning, DL) stratejileri, RL ile birleştirilerek kullanılmaya başlamıştır. DRL olarak adlandırılan bu yöntem, daha önce bir makinenin yapamayacağı çok çeşitli karmaşık karar verme süreçlerinde, yani yüksek boyutlu durum uzayına sahip problemlerde daha başarılı olabilmektedir. Üstelik bunu yaparken veriden yapılan çıkarımla öğrenme kabiliyetlerinden dolayı geçmiş bilginin daha az bir miktarına ihtiyaç duymaktadır (François-Lavet vd., 2018). Böylece DRL, sadece sanal oyunlarda değil, robotik (Levine vd., 2016; Gandhi vd., 2017; Pinto vd., 2017; Bing vd, 2020b; Dooraki ve Lee, 2021, Zhu ve Rosendo, 2022), otonom araçlar (Pan vd., 2017), akıllı şebekeler (François-Lavet, 2017) ve finans (Deng vd., 2017) gibi birçok gerçek hayat uygulamalarında da potansiyel başarı sunmaktadır. Son zamanlardaki bu hızlı gelişmenin temel sebebi özellikle GPU ve dağıtık hesaplamanın kullanımı ile hesaplama gücündeki exponansiyel artış (Krizhevsky vd., 2012), DL yöntemlerindeki ilerleme (Srivastava vd., 2014; Szegedy vd., 2016), ve Tensorflow gibi yazılım ve ImageNet gibi veri setlerinin çeşitliliğinin artmasıdır.

Başlangıçta genellikle yüksek boyutlu sensör girişlerine ve sürekli aksiyon uzayına sahip olan robotik alanında yaygın olarak kullanılmayan DRL, algoritmalardaki ilerlemelerle birlikte yürüyüş biçimi üretme (Peng vd., 2017) ve manipülatörlerin beceri isteyen karmaşık işlerde kullanılması (Rajeswaran vd., 2017) gibi zorlu kontrol görevlerinde ele alınmaya başlamıştır. Schulman vd. (2017), DRL algoritmalarından yakınsal politika optimizasyonu (Proximal Policy Optimization, PPO) algoritmasını 2D yüzen robottan 3D insansı robota kadar 7 farklı robotik hareket problemine uygulamışlardır. DRL ile referans yörünge üretme dışında enerji verimli hareket

sağlayabilmek için de çalışmalar yapılmıştır. İki bacaklı bir robot için enerji verimli yürüme hareketi Kormushev vd. (2019) tarafından ele alınırken, Yu vd. (2018) iki ve altı bacaklı robotun yürümesi ile dört bacaklı ve insansı bir robotun koşması olmak üzere dört farklı yürüyüş biçimi için düşük enerjili ve uygun hızlı hareketleri oluşturabilmek için DRL tabanlı bir yaklaşım önermişlerdir. Bir başka çalışmada DDPG algoritması ile CPG parametrelerinin optimizasyonu yapılarak altı bacaklı bir robotun çevresel adaptasyonu için yörüngeler üretilmiştir (Ouyang vd., 2021). Bing vd. (2019)'nin çalışmasında ise tekerlekli bir yılan robot için enerji verimli yürüyüşlerin üretilebilmesi için PPO algoritması kullanılmıştır. Bu çalışmanın genişletilmiş bir versiyonu, çekişmeli ters pekiştirmeli öğrenme (Adversarial Inverse Reinforcement Learning, AIRL) algoritması kullanılarak sunulmuştur (Bing vd., 2020a). Yılan robot için enerji verimli yürüyüş üretmeyi ele alan bir başka çalışmada ise PPO algoritması ve güven bölgesi politika optimizasyonu (Trust Region Policy Optimization, TRPO) algoritması kullanılarak elde edilen sonuçlar karşılaştırılmıştır (Liu ve Farimani, 2021).

1.6.1. Pekiştirmeli Öğrenme

RL modeli canlıların biyolojik öğrenme sistemlerinden esinlenilerek geliştirilen ve önceden hazırlanmış bir eğitim verisi olmadan öğrenen makinenin (ajanın) çevreyle etkileşimi sonucunda aldığı tepkileri kullanarak eğitilmesini amaçlayan bir makine öğrenmesi türüdür. Bu öğrenme yönteminde ajan, çevresinde karşılaştığı durumlara verdiği tepki karşılığında sayısal bir ödül alır ve bu ödülü maksimuma çıkarmaya çalışarak istenilen davranışı deneme-yanılma yoluyla öğrenir.

RL'de, öğrenme sisteminin aksiyonları sistemin gelecek girişlerini etkilediği için bu öğrenme yöntemi kapalı çevrim problemler için uygundur. Ayrıca, RL'de diğer çoğu öğrenme yöntemlerinde olduğu gibi ajana hangi aksiyonun gerçekleştirileceği söylenmez. Bunun yerine, öğrenen sistemden hangi aksiyonu gerçekleştirdiğinde en fazla ödülü aldığını keşfetmesi beklenir. RL'nin bir diğer önemli özelliği ise gecikmeli ödüllerdir. Yani ajanın o anlık yaptığı aksiyon, anlık ödüle katkı sağlamazken gelecekteki toplam ödülün artmasına katkı sağlamış olabilir. Bu üç özellik RL probleminin en önemli ayırt edici özelliklerindendir (Sutton ve Barto, 2018).

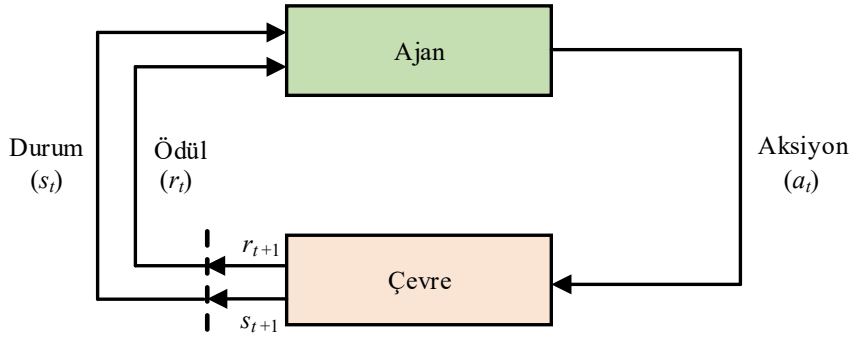
RL yaklaşımı, danışmalı öğrenme ve danışmansız öğrenmeden oluşan diğer makine öğrenme yaklaşımlarından farklıdır. Danışmanlı öğrenmede, bir uzman tarafından

oluşturulan etiketli bir veri kümesi kullanılır. Ağın girişi ve çıkışı önceden bellidir. Ağ, verilen bu girişlere göre istenen çıkışları oluşturabilmek için kendi çıkışlarını günceller. Danışmanlı öğrenmede eğitim setinde olmayan bir durum için doğru kararı verebilmek amacıyla genelleştirme yapılabilir ancak bu tür bir öğrenme çevreyle interaktif bir etkileşimin olduğu problemlerde öğrenme için yetersiz kalmaktadır. Çünkü böyle bir problemde, bütün durumları temsil eden ajanın yapması gereken doğru aksiyona ait örnekler elde etmek genellikle pratik bir yol değildir (Sutton ve Barto, 2018). Çoğu zaman doğru aksiyonun ne olması gerektiği önceden kestirilemez. Bu nedenle özellikle bilinmeyen etkileşimli ortamlarda RL kullanılarak ajanın kendi tecrübelerinden öğrenmesi beklenilir.

Danışmansız öğrenmede ise, ağa herhangi bir etiketli veri verilmez. Sadece giriş verileri ağa verilerek sistemin bu verileri birbirine en çok benzeyenleri bir araya getirerek kümelemesi beklenir. RL çoğu zaman danışmansız öğrenmenin bir alt dalı olarak görünse de RL'de ajan bir aksiyon seçtiği zaman ona bunun doğru ya da yanlış olduğu söylenmediği ve sadece ödül sinyalini maksimize etmeye dayalı bir yapısı olduğu için danışmansız öğrenmeden ayrılarak üçüncü bir makine öğrenme yöntemi olarak ortaya çıkmaktadır.

1.6.1.1. Pekiştirmeli Öğrenme Modelinin Bileşenleri

Bir RL sisteminde, ajan ve çevre bileşenleri dışında politika, ödül, değer fonksiyonu ve opsiyonel olan çevrenin modeli olmak üzere dört temel bileşen bulunmaktadır (Sutton ve Barto, 2018). Ajan çevreden aldığı duruma (state) göre bir aksiyon üretir ve çevreden bu aksiyona karşı oluşan tepki önceden belirlenmiş ödül sistemine tabi tutularak ajan eğitilir. RL modelinin genel yapısı Şekil 1.6'da verilmiştir.



Şekil 1.6. RL modelinin genel yapısı

Politika (π), bir aksiyonu belirlemek için ajanın şu andaki durumuna bağlı olarak uyguladığı strateji olarak tanımlanır. Yani algılanan çevre durumlarından aksiyonlara haritalama yapan $\pi : S \to A$ fonksiyonu olarak düşünülebilir. Politika basit bir fonksiyon veya arama tablosundan oluşabileceği gibi kapsamlı bir hesaplama içeren arama sürecinden de oluşabilir. Genel olarak iki çeşit politika fonksiyonu bulunmaktadır.

- Stokastik politika fonksiyonu, ajanın s durumundayken a aksiyonunu seçme olasılığıdır ve eşitlik (1.33)'teki gibi ifade edilir.

$$\pi(a / s) = \mathrm{P}\left[a_t = a \mid s_t = s\right] \tag{1.33}$$

- Deterministik politika fonksiyonu, ajanın s durumundayken a aksiyonunu seçmesini ifade eder ve eşitlik (1.34)'deki gibi tanımlanır.

$$a = \pi(s) \tag{1.34}$$

Ödül (r), ajanın aldığı aksiyonu değerlendirmek için her zaman adımında çevreden alınan sayısal bir değer olup RL probleminin amacını tanımlar. Ajanın tek amacı uzun vadede aldığı toplam ödülü maksimuma çıkarmak olduğundan çevreden alınan bu ödüller ajanın izlediği politikayı değiştirmesi için baz alınan temel kontrol mekanizmasını oluşturur. Yani ajan tarafından karar verilen bir aksiyon sonucunda düşük bir ödül kazanılmışsa, bu durumla tekrar karşılaşıldığı zaman farklı bir aksiyon seçmek için politika değiştirilebilir.

Değer fonksiyonu, ajanın içinde bulunduğu durumdan başlayarak gelecekte alabileceği beklenilen ödüllerin toplamı olarak tanımlanır. Bir durumun anlık almış olduğu

ödülün yüksek veya düşük olması bu durumun iyi ya da kötü olduğunu göstermez. Çünkü herhangi bir durum düşük bir ödüle sahip olmasına rağmen kendinden sonraki durumların yüksek ödüle sahip olmasından dolayı yüksek bir değere sahip olabilir. Aynı şekilde yüksek ödüle sahip bir durumu takip eden düşük ödüllü diğer durumlar yüksek ödüllü bu durumun düşük bir değere sahip olmasına neden olabilir. Bu nedenle uzun vadede toplam ödülü maksimuma çıkarmak için aksiyon seçiminde durumun değeri göz önüne alınır. Yani en yüksek ödüllü durumları meydana getiren aksiyonlar değil en yüksek değerli durumları meydana getiren aksiyonlar aranır. Değerlerin tahmini ajanın bütün yaşamı boyunca yapacağı gözlemlere dayandığı için çevreden doğrudan elde edilebilen ödüle göre çok daha zordur. Bu nedenle değerlerin tahmin edilmesi, karar verme ve planlama için RL algoritmalarının hemen hemen hepsinde rol oynayan en önemli faktördür (Sutton ve Barto, 2018).

Değer fonksiyonu eşitlik (1.35)'teki gibi tanımlanabilir. Beklenen gelecek ödül spesifik bir durumdan alındığı için bu değer fonksiyonu, durum değer fonksiyonu adını alır.

$$\begin{aligned} V^{\pi}(s) &= \mathrm{E}\left[R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \ldots \mid s_0 = s, \pi \right] \\ &= \mathrm{E}_{\pi}\left\{ R_t \mid s_t = s \right\} = \mathrm{E}_{\pi}\left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right\} \end{aligned} \tag{1.35}$$

Bu eşitlikteki γ azaltma faktörü olup [0,1] aralığında değer alır ve gelecekteki ödüllerin önemini belirler. t+1 zaman adımından sonra alınan ödülün değeri $\gamma^t R$ olur. γ'nin 0'a yakın olması ajanın sadece mevcut ödülleri göz önünde bulundurarak miyop bir değerlendirme yapmasına sebep olurken 1'e yakın olması ise ajanın uzun vadeli yüksek bir ödül için çaba sarf etmesini sağlayacaktır.

Ajanın s durumundayken spesifik bir a aksiyonu seçmesi durumda beklenen gelecek ödülü ifade eden değer fonksiyonu ise aksiyon değer fonksiyonu olarak adlandırılır ve eşitlik (1.36)'daki gibi ifade edilir.

$$\begin{aligned} Q^{\pi}(s,a) &= \mathrm{E}_{\pi}\left\{ R_t \mid s_t = s,\ a_t = a \right\} = \mathrm{E}_{\pi}\left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s,\ a_t = a \right\} \\ &= \mathrm{E}_{\pi}\left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s,\ a_t = a \right\} \end{aligned} \tag{1.36}$$

Model, bazı RL yöntemlerinde çevrenin davranışını simüle etmek için kullanılan dördüncü bileşendir. Ajanın belirli bir durumda bir aksiyonu gerçekleştirmesi ile gelecekte oluşacak durumun ve alınacak ödülün tahmin edilmesini sağlar.

1.6.1.2. Markov Karar Süreci

Markov özelliği, gelecekte oluşması muhtemel durumların gerçekleşme olasılıklarının geçmiş durumlardan bağımsız olarak yalnızca şimdiki duruma bağlı olmasıdır. Çevredeki bütün durumlar Markov özelliğine sahip ise çevre de Markov özelliği taşır. Markov çevresi olarak adlandırılan böyle bir çevrede tanımlanan RL problemi ise Markov karar süreci (Markov Decision Process, MDP) olarak adlandırılır. Hemen hemen bütün RL problemleri MDP'ler olarak formülize edilir. Bunun için çevrenin tam gözlemlenebilir olması gerekmektedir. Kısmi gözlemlenebilir problemler ise MDP'lere dönüştürülebilir.

MDP'ler ilk kez Bellman (1957) tarafından tanımlanmıştır. Durum ve aksiyonların sonlu olduğu kabul edilirse MDP'yi oluşturan öğelerden biri olan durum geçiş olasılığı eşitlik (1.37)'deki gibi ifade edilebilir.

$$p(s' \mid s, a) = \mathrm{P}\left[s_{t+1} = s' \mid s_t = s, a_t = a\right] \tag{1.37}$$

Bu eşitlik şu andaki durumun *s* ve seçilen aksiyonun *a* olması durumunda gelecek durumum *s'* olma olasılığını ifade eder.

Eşitlik (1.38)'deki ifade ise şu andaki durumun *s*, gelecek durumun *s'* ve seçilen aksiyonun *a* olması durumunda gelecek ödülün beklenilen değerini ifade eder.

$$r(s, a, s') = \mathrm{E}\left[r_{t+1} \mid s_t = s, a_t = a, s_{t+1} = s'\right] \tag{1.38}$$

Ajana verilen tek bilgi şu andaki durum bilgisi olduğundan ve bir aksiyonu seçip öğrenmesi bu bilgiye bağlı olduğundan Markov özelliği RL'de oldukça önemlidir. Durum geçmişle ilgili bütün bilgiye sahiptir ve böylece geçmiş bilgisinin artık tutulmasına gerek kalmamaktadır. Geleceği tahmin etmek için şu andaki durumu bilmek yeterlidir.

1.6.1.3. Optimal Politika

Optimal politika (π^*), uzun vadede alınan toplam ödülü maksimum yapmak için her durumda alınabilecek en iyi aksiyonu belirleyen politikadır ve eşitlik (1.39) ve (1.40)'ta sırasıyla verilen optimal durum değer fonksiyonu V^* ile optimal aksiyon değer fonksiyonu Q^* kullanılarak tanımlanabilir.

$$V^*(s) = \max_{\pi} V^{\pi}(s) \tag{1.39}$$

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a) \tag{1.40}$$

Optimal durum değer fonksiyonu, optimal politika altındaki bir durumun değerinin, en iyi aksiyon için o durumdan beklenen değere eşit olması gerektiğini ifade ettiği için aksiyon değer fonksiyonunun bir fonksiyonu olarak eşitlik (1.41)'deki gibi ifade edilebilir (Sutton ve Barto, 2018).

$$V^*(s) = \max_{a \in A(s)} Q^*(s,a) \tag{1.41}$$

$\pi^*(s)$'nin en yüksek aksiyon değerini veren a olduğu düşünülürse $\pi^*(s)$, $Q^*(s,a)$'nın bir fonksiyonu olarak eşitlik (1.42)'deki gibi tanımlanabilir.

$$\pi^*(s) = \arg\max_{a \in A(s)} Q^*(s,a) \tag{1.42}$$

Bu eşitlik $Q^*(s,a)$'yı içerdiğinden dolayı $\pi^*(s)$'yi hesaplamak için tek seçenek, π için bütün olası değerleri denemektir. Bu da genellikle olası değildir (Nissen, 2007). Bu nedenle optimal politikayı bulabilmek için daha kolay hesaplanabilecek bir optimal durum değeri fonksiyonunun bulunması gerekmektedir. Bunun için Bellman eşitlikleri, eşitlik (1.37) ve (1.38)'deki $p(s'|s,a)$ ve $r(s,a,s')$ fonksiyonları cinsinden eşitlik (1.43)'teki gibi yazılabilir.

$$
\begin{aligned}
V^*(s) &= V^{\pi^*}(s) \\
&= \max_{a \in A(s)} Q^{\pi^*}(s,a) \\
&= \max_{a \in A(s)} \mathrm{E}_{\pi} \left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s,\ a_t = a \right\} \\
&= \max_{a \in A(s)} \mathrm{E} \left\{ r_{t+1} + \gamma V^*(s_{t+1}) \mid s_t = s,\ a_t = a \right\} \\
&= \max_{a \in A(s)} \sum_{s' \in S} p(s' \mid s,a) \left[r(s,a,s') + \gamma V^*(s') \right]
\end{aligned}
\tag{1.43}
$$

Aynı şekilde aksiyon değer fonksiyonu için Bellman eşitlikleri eşitlik (1.44)'teki gibi yazılabilir.

$$
\begin{aligned}
Q^*(s,a) &= \mathrm{E}_{\pi^*} \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s,\ a_t = a \right\} \\
&= \mathrm{E}_{\pi^*} \left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s,\ a_t = a \right\} \\
&= \mathrm{E} \left\{ r_{t+1} + \gamma V^*(s_{t+1}) \mid s_t = s,\ a_t = a \right\} \\
&= \mathrm{E} \left\{ r_{t+1} + \gamma \max_{a' \in A(s_{t+1})} Q^*(s_{t+1}, a') \mid s_t = s,\ a_t = a \right\} \\
&= \sum_{s' \in S} p(s' \mid s,a) \left[r(s,a,s') + \gamma \max_{a' \in A(s')} Q^*(s',a') \right]
\end{aligned}
\tag{1.44}
$$

Eşitlik (1.43) ve (1.44)'te elde edilen Bellman eşitlikleri, eşitlik (1.42)'de yerine konulursa optimal politika, V^* cinsinden eşitlik (1.45)'te ve $Q^*(s,a)$ cinsinden eşitlik (1.46)'da verildiği gibi elde edilir.

$$
\pi^*(s) = \arg\max_{a \in A(s)} \sum_{s' \in S} p(s' \mid s,a) \left[r(s,a,s') + \gamma V^*(s') \right] \tag{1.45}
$$

$$
\pi^*(s) = \arg\max_{a \in A(s)} \sum_{s' \in S} p(s' \mid s,a) \left[r(s,a,s') + \gamma \max_{a' \in A(s')} Q^*(s',a') \right] \tag{1.46}
$$

$p(s'|s,a)$ ve $r(s,a,s')$ bilindiği zaman, optimal politikayı hızlı bir şekilde bulabilmek için yukardaki eşitlikleri iteratif olarak çözen iki farklı dinamik programla yöntemi vardır. Bunlar değer iterasyonu ve politika iterasyonu algoritmalarıdır. Değer iterasyonu yönteminde optimal deterministik politikayı bulmak için öncelikle eşitlik (1.43)

kullanılarak optimal değer fonksiyonu $V^*(s)$ hesaplanır, daha sonra optimal politika eşitlik (1.45)'ten bulunur. Politika iterasyonu yönteminde de eşitlik (1.43) ve (1.45) kullanılmaktadır. Fakat değer iterasyonu optimal değer fonksiyonunu bulduktan sonra politikayı güncellerken, politika iterasyonu ise her iterasyonda politikayı günceller ve bir sonraki iterasyonda güncellenmiş politikayı kullanır.

1.6.1.4. Pekiştirmeli Öğrenme Algoritmaları

RL algoritmaları model tabanlı öğrenme algoritmaları ve modelsiz öğrenme algoritmaları olmak üzere iki gruba ayrılabilir.

1.6.1.4.1. Model Tabanlı Öğrenme Algoritmaları

Model, çevre dinamiklerinin simülasyonunu temsil eder. Model tabanlı öğrenme algoritmaları, optimal politikayı bulmak için dinamik programlama yöntemi kullanmadan önce çevreyi keşfederek $p(s'|s,a)$ ve $r(s,a,s')$ fonksiyonlarını tahmin etmeye çalışır. Bu da bütün durum-aksiyon çiftlerinin keşfedilmesi anlamına geldiği için bu algoritmalar ile öğrenme fazla zaman almaktadır. Bu sebepten dolayı model tabanlı algoritmaların durum-aksiyon uzayı büyük olan problemlerde kullanılması pratik değildir. Ayrıca bu yöntemde algoritma yalnızca çevreyi keşfetmeye odaklanır ve keşif sırasında optimal politika hakkında edindiği bilgilerin bir kısmını kullanmaz. Bu da keşif sırasında çok az bir ödülün elde edilmesine sebep olur. Değer iterasyon algoritması ve politika iterasyon algoritması model tabanlı öğrenme algoritmalarıdır.

1.6.1.4.2. Modelsiz Öğrenme Algoritmaları

Modelsiz öğrenme algoritmaları, en yüksek beklenen ödüle sahip aksiyonu seçerek mümkün olduğunca fazla ödül kazanmaya çalışan aç gözlü yaklaşımlar olarak bilinirler. Çevre ile etkileşimden kazanılan bilgi derhal kullanılarak doğrudan optimal politikanın bulunması hedeflenir. Böylece bu algoritmalar ile daha hızlı bir öğrenme gerçekleştirilebilmektedir. Fakat aç gözlü bu yaklaşım sıklıkla yerel optimal çözüme takılarak global optimal politikayı bulma ihtimalini azaltır. Modelsiz öğrenme

algoritmaları değer tabanlı ve politika tabanlı algoritmalar olmak üzere iki başlık altında toplanabilir.

Değer tabanlı algoritmalarda ajan öncelikle durum değer fonksiyonu $V^{\pi}(s)$'yi tahmin ederek politikayı öğrenmeye çalışır. Zamansal fark (Temporal difference, TD) öğrenimi (Sutton, 1988), Q öğrenme (Watkins, 1989) ve SARSA (State Action Reward State Action) (Rummery ve Niranjan, 1994) değer tabanlı algoritmalardır. Değer tabanlı algoritmalar zor problemleri çözmede başarılıdırlar.

Politika tabanlı algoritmalarda ise ajan, her durumda gelecekte maksimum ödül kazanabileceği aksiyonları uygulayabileceği politikayı öğrenmeye çalışır. Yani bu algoritmalarda doğrudan optimal politika öğrenilmeye çalışılır ve değer fonksiyonunu içermezler. Herhangi bir durumdaki gelecek aksiyon, politika tarafından belirlenir. Politika gradyanı (Policy Gradient, PG) (Williams, 1992), TRPO (Schulman vd., 2015) ve PPO (Schulman vd., 2017) politika tabanlı algoritmalardır. Politika tabanlı algoritmaların sürekli aksiyon uzayında kullanımı uygun olup danışmalı öğrenme algoritmaları ile uyumludur. Değer tabanlı algoritmalara göre daha kolay problemlerde başarılı olurlar.

1.6.1.4.3. Aktif Politika ve Pasif Politika

Aktif politikalı öğrenmede ajan halihazırda takip edilen politikadan türetilen mevcut aksiyona bağlı olarak öğrenme işlemini gerçekleştirir. Yani bir politikanın güncellenmesi kendisi tarafından toplanan veriler kullanılarak yapılır. Pasif politikalı öğrenmede ise, takip edilen politikadan farklı bir politikaya göre öğrenme yapılır. Yani her ajanın topladığı veriler tampon belleğe eklenir ve politika güncellenmesi bu bellekten örneklenen veriler ile yapılır. Pasif politikalı öğrenme eski politikalar tarafından toplanan geçmiş örneklerin kullanılmasına olanak verir.

1.6.2. Derin Öğrenme

DL, çok katmanlı yapay sinir ağlarını kullanarak giriş verisinden ayırt edici özellikleri otomatik olarak öğrenebilen bir makine öğrenmesi yöntemidir. DL'yi geleneksel makine öğrenmesinden ayıran yönü, ayırt edici özellikleri doğrudan veriden öğrenebilmesidir (Deng ve Yu, 2014; Sarker, 2021a). Geleneksel makine öğrenmesi

algoritmalarında ayırt edici bu özellikler eğitim aşamasından önce uzman biri tarafından belirlenerek hesaplanır ve öğrenme işlemi hesaplanan bu özelliklere göre yapılır. Veriden çıkarılacak özelliklerin uygun belirlenememesi durumunda ise öğrenme işleminin başarısı düşer. Bu nedenle insan müdahalesine gerek kalmadan girdiden temel özellikleri otomatik olarak keşfetme yeteneği DL yaklaşımlarının başarısında oldukça önemli bir etkendir (Sarker, 2021b). Katmanlardan oluşan özellik öğrenme işleminin başarılı bir şekilde yapılabilmesi için sistemin yeterince eğitilmesi gerekir. Ağın alt katmanlarındaki özellikler daha az ayırt ediciliğe sahip iken daha anlamlı özellikler için temel oluştururlar. Üst katmanlar ise daha fazla ayırt ediciliğe sahiptirler (Alzubaidi, 2021).

Derin sinir ağı (Deep Neural Network, DNN)'nın temel mimarisi, ileri beslemeli yapay sinir ağının bir çeşidi olan çok katmanlı algılayıcı (Multi Layer Perceptron, MLP) ağ modelinden oluşmaktadır. Şekil 1.7'de geleneksel bir tam bağlı MLP mimarisi verilmiştir. İlk katman giriş katmanı olup n_f boyutlu giriş verisini ($x = x_1, x_{2, \ldots,} x_f$) alarak birinci gizli katmanın girişine iletir ve bu değerler eşitlik (1.33)'teki gibi doğrusal olmayan parametrik bir fonksiyon ile birinci gizli katmanın çıkış değerlerine dönüştürülür.

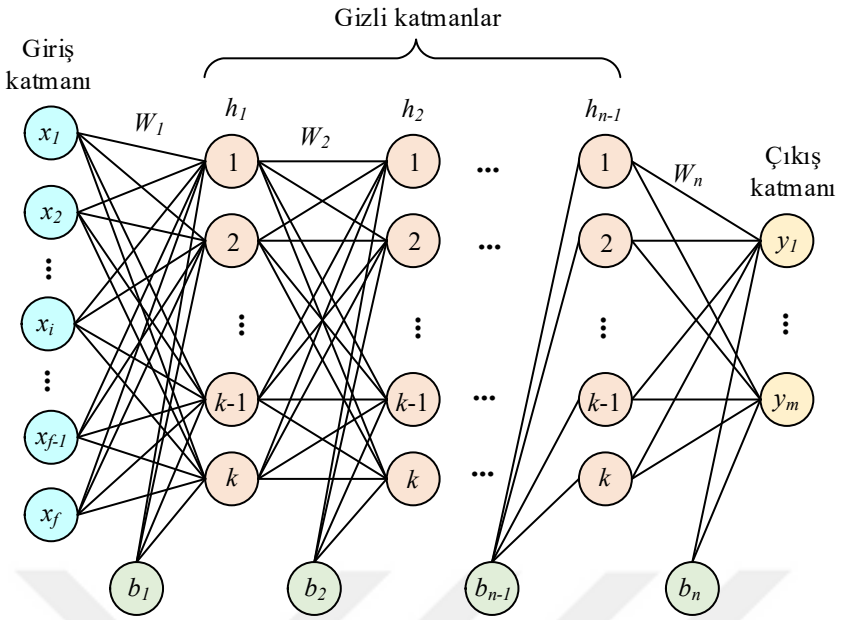
$$\mathbf{h}_1 = f(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1) \tag{1.33}$$

Her bir katmanda bulunan nöron sayısı k olmak üzere $\mathbf{W}_1$, n_k x n_f boyutlu ağırlık katsayısı matrisini ifade ederken, $\mathbf{b}_1$ terimi n_k boyutlu bias vektörüdür. f ise doğrusal olmayan dönüşümü yapan aktivasyon fonksiyonudur.

Her katmanın çıkışı kendinden sonraki katmanın girişini oluşturur ve giriş katmanından çıkış katmanına ulaşılıncaya kadar her katmanda eşitlik (1.33)'teki gibi doğrusal olmayan bir dönüşüm yapılır. Buna göre n_m boyutlu ağın çıkış değeri ($y = y_1, y_{2, \ldots,} y_m$) eşitlik (1.34)'teki gibi ifade edilebilir.

$$\mathbf{y} = O(\mathbf{W}_n \mathbf{h}_{n-1} + \mathbf{b}_n) \tag{1.34}$$

Bu eşitlikteki O, çıkış katmanında kullanılan aktivasyon fonksiyonunu ifade etmekte olup bu fonksiyonun seçimi ele alınan probleme bağlıdır. Eşitlikteki $\mathbf{W}_n$ ağırlık matrisi n_m x n_k boyutlu iken $\mathbf{b}_n$ bias vektörü ise n_m boyutludur.



Şekil 1.7. Tam bağlı DNN'nin temel mimarisi

Ağın eğitilmesindeki temel amaç ağın hata fonksiyonu $E(X,\theta)$ değerini minimize etmektir. Bu fonksiyon, ağın parametreleri θ iken istenilen çıkış değeriyle elde edilen çıkış değeri arasında oluşan hatayı tanımlar. Hatayı azaltmak için sinir ağı parametrelerini optimize etmek gerekir. Bunun için en çok kullanılan yöntemlerden biri gradyan azaltma (Gradient Descent, GD) algoritmasına dayanan geriye yayılım algoritmasıdır (Rumelhart vd., 1988). Gradyan azaltma algoritması ile ağın eğitilmesi için hata fonksiyonunun θ'ya göre gradyanının hesaplanması gerekir. Daha sonra öğrenme hızı α_n'ya göre her iterasyonda ağırlık ve bias değerlerinden oluşan parametre vektörü θ eşitlik (1.35)'teki gibi güncellenir.

$$\theta^{t+1} = \theta^t - \alpha_n \frac{\partial E(X,\theta^t)}{\partial \theta} \tag{1.35}$$

Literatürde GD tabanlı çeşitli optimizasyon algoritmaları kullanılmaktadır. Bunlar arasında stokastik gradyan azaltma (Stochastic Gradient Descent, SGD) (Bottou, 2010), Momentum (Qian, 1999), AdaGrad (Duchi vd., 2011), Adadelta (Zeiler, 2012), ortalama karesel yayılımın karakökü (Root Mean Square Propogation, RMSProp) (Hinton vd., 2012) ve adaptif moment tahmini (Adaptive Moment Estimation, Adam) (Kingma ve Ba, 2014) en popüler olanlarıdır. Bu tez çalışmasında Momentum ve RMSProp yöntemlerinin

avantajlı yönlerinin birleştirilmesi ile elde edilen Adam optimizasyon algoritması kullanılmıştır.

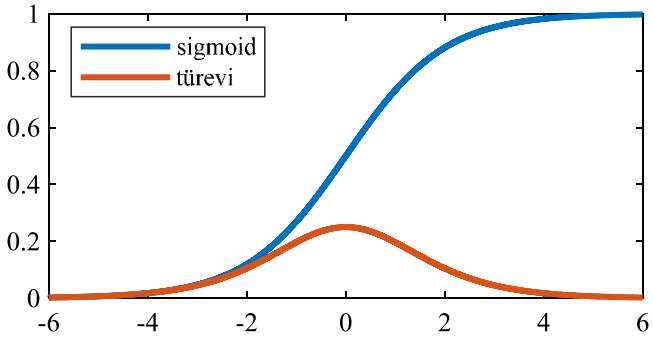
Basit ileri beslemeli sinir ağları dışında sinir ağı katmanlarının pek çok çeşidi bulunmaktadır. Bunlardan her biri uygulamaya bağlı olarak belirli avantajlar sağlar. Örneğin; veriyi çeşitli katmanlarla işleyen evrişimli sinir ağları (Convolutional Neural Network, CNN) özellikle resim ve video verileri için oldukça uygun iken çıkışlarını bir sonraki işlemde giriş olarak kullanarak şimdiki ve önceki bilgilerin birleştirilmesi ile çıkış üreten tekrarlamalı sinir ağları (Recurrent Neural Network, RNN) sıra bilgisinin önemli olduğu yazı, konuşma, zamana bağlı çeşitli sensör veya istatiksel verilerin yapısını anlamada kullanılır.

1.6.2.1. Aktivasyon Fonksiyonları

Aktivasyon fonksiyonu, yapay sinir ağlarına görüntü, video, yazı ve ses gibi doğrusal olmayan karmaşık gerçek dünya özelliklerini tanıtmak için kullanılır. Aktivasyon fonksiyonu içermeyen bir yapay sinir ağı sadece doğrusal ilişkileri öğrenebilen sınırlı öğrenme kabiliyetine sahip basit bir lineer regresyon gibi davranacaktır. Türev alan bir sistemden oluşan geriye yayılım algoritması ile öğrenme işlemi gerçekleştirildiğinden aktivasyon fonksiyonunun türevlenebilir olması öğrenmenin gerçekleştirilebilmesi için gereklidir. Literatürde aktivasyon fonksiyonu olarak kullanılan birçok fonksiyon olmasına rağmen son zamanlarda özellikle DL'de kullanılan aktivasyon fonksiyonlarına aşağıda yer verilmiştir.

Sigmoid fonksiyonu en sık kullanılan aktivasyon fonksiyonlarından biridir. Bu fonksiyonun iyi bir sınıflayıcı olarak kullanılmasının sebebi fonksiyonun sıfıra yakın bölgesinde girişin küçük bir değişimin çıkışta büyük bir değişime neden olmasıdır. Ayrıca [0,1] aralığında değerler üretmesi sigmoid fonksiyonun başka bir avantajıdır. Bu fonksiyonun dezavantajı ise fonksiyonun uçlarına doğru girişin değişimine karşı çıkışta çok az bir değişimin olmasıdır. Bu da bu bölgelerde türev değerlerinin çok küçük olmasına ve sıfıra yakınsamasına neden olur. Bu durum gradyanların ölmesi olarak adlandırılır ve öğrenme olayının minimum düzeyde gerçekleştirildiği anlamına gelir. Öğrenme olayının yavaşlaması hatayı minimize eden optimizasyon algoritmasının lokal minimuma takılmasına neden olabilir. Sigmoid fonksiyonu ve türevi Şekil 1.8'de görülmektedir. Fonksiyonun matematiksel ifadesi ise eşitlik (1.36)'da verilmiştir.

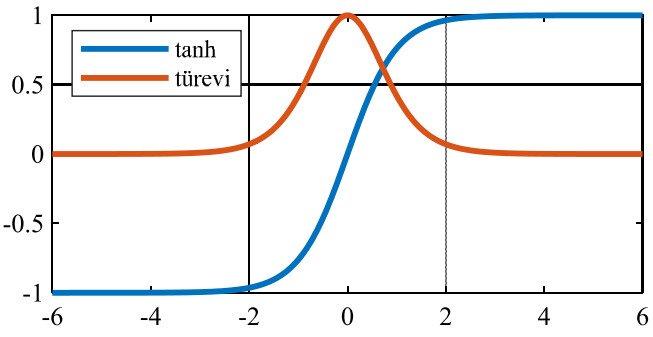
$$f(x)=\frac{1}{1+e^{-x}} \tag{1.36}$$



Şekil 1.8. Sigmoid fonksiyonu ve türevi

Şekil 1.9'de türeviyle birlikte verilen hiperbolik tanjant fonksiyonu [-1,+1] aralığında çıkış üretmektedir. Sigmoid fonksiyonuna benzer bir yapısı olmasına rağmen daha geniş aralığa sahip olması sınıflandırma işlemlerinde avantaj sağlamaktadır. Ayrıca türevinin daha büyük değerler alması öğrenme işleminin daha hızlı gerçekleşmesini sağlar. Sigmoid fonksiyonundaki gibi fonksiyonun uçlarına doğru gradyanların ölmesi problemi mevcuttur. Fonksiyonun matematiksel ifadesi ise eşitlik (1.37)'de verilmiştir.

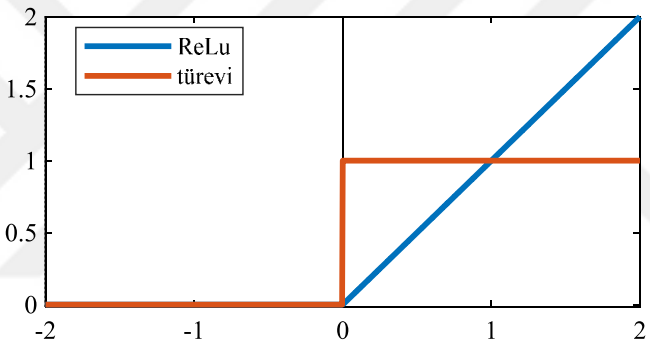
$$f(x)=\tanh(x)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{1.37}$$



Şekil 1.9. Hiperbolik tanjant fonksiyonu ve türevi

Doğrultulmuş lineer birim (Rectified Linear Unit, ReLU), [0,∞) aralığında değer alan bir aktivasyon fonksiyonudur. Bu fonksiyon ve türevi Şekil 1.10'da görülmektedir. Fonksiyonun negatif eksende sıfır değerini alması ağın daha hızlı çalışmasını sağlarken türevin bu bölgede sıfır olmasına yani öğrenmenin gerçekleşmemesine sebep olur. Hesaplama açısından sigmoid ve hiperbolik tanjant fonksiyonlarına göre çok daha verimli olması sebebiyle çok katmanlı derin ağlarda yaygın olarak tercih edilmektedir. ReLU fonksiyonu çıkış katmanından ziyade genellikle ara katmanlarda kullanılır. ReLU'nun matematiksel ifadesi eşitlik (1.38)'de verilmiştir.

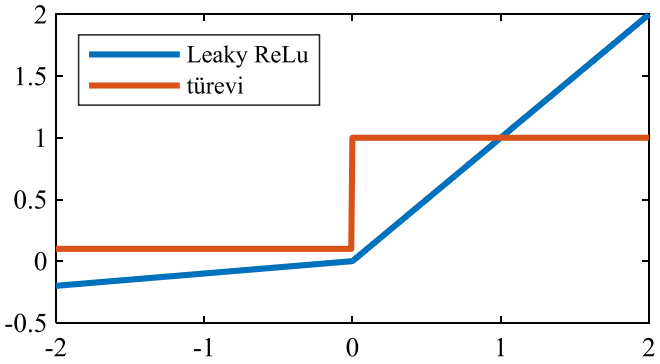
$$f(x)=\begin{cases}0 \text{ için} & x<0\\ x \text{ için} & x\geq 0\end{cases} \tag{1.38}$$



Şekil 1.10. ReLU fonksiyonu ve türevi

ReLU'nun negatif bölgedeki eksikliğini gidermek için (-∞,∞) aralığında değer alan Leaky ReLU fonksiyonu önerilmiştir. Negatif bölgedeki sızıntı değeri 0.01 olan bu fonksiyon ile negatif bölgede de öğrenme sağlanmış olur. Fonksiyonun matematiksel ifadesi eşitlik (1.39)'da verilmiştir. Leaky ReLU fonksiyonu ve türevi Şekil 1.11'de görülmektedir.

$$f(x)=\begin{cases}0.01x \text{ için} & x<0\\ x \text{ için} & x\geq 0\end{cases} \tag{1.39}$$



Şekil 1.11. Leaky ReLu fonksiyonu ve türevi

Genellikle çok sınıflı DL problemlerinin çıkış katmanında sınıflayıcı olarak kullanılan softmax aktivasyon fonksiyonu, verilen bir girdinin belirli bir sınıfa ait olma olasılığını [0,1] aralığında değerler üreterek belirler. Tüm sınıflara ait bu olasılık değerlerinin toplamı 1'e eşittir. Softmax fonksiyonunun matematiksel ifadesi eşitlik (1.40)'da verildiği gibidir.

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=0}^{k} e^{x_j}} \tag{1.40}$$

Bu eşitlikteki k sınıf sayısını göstermektedir.

1.6.3. Derin Pekiştirmeli Öğrenme

DRL'de ajan, politikayı öğrenebilmek için derin sinir ağlarını kullanır ve bu politikaya göre bir aksiyon üretir. Bunun karşılığında çevreden alınan ödül, daha iyi bir politika üretebilmek için sinir ağına geri besleme olarak verilir. Bu yöntem Mnih vd. (2013) tarafından geliştirilip ajanın Atari oyunu oynamayı doğrudan piksellerden öğrenmesi için kullanılmıştır. Daha sonra AlhpaGo ekibi, CNN ve RNN mimarilerini kullanarak Go oyununu oynamayı öğrenebilecek, uzmanları izleyerek ve kendine karşı oynayarak eğitilen bir DRL sistemi geliştirmişlerdir (Silver vd., 2016). Zamanla pek çok farklı alana uygulanan DRL, görsel dünyayı daha yüksek düzeyde anlayan otonom sistemlerin geliştirilmesinde önemli bir rol oynamaktadır. Bununla birlikte, RL'nin DL ile

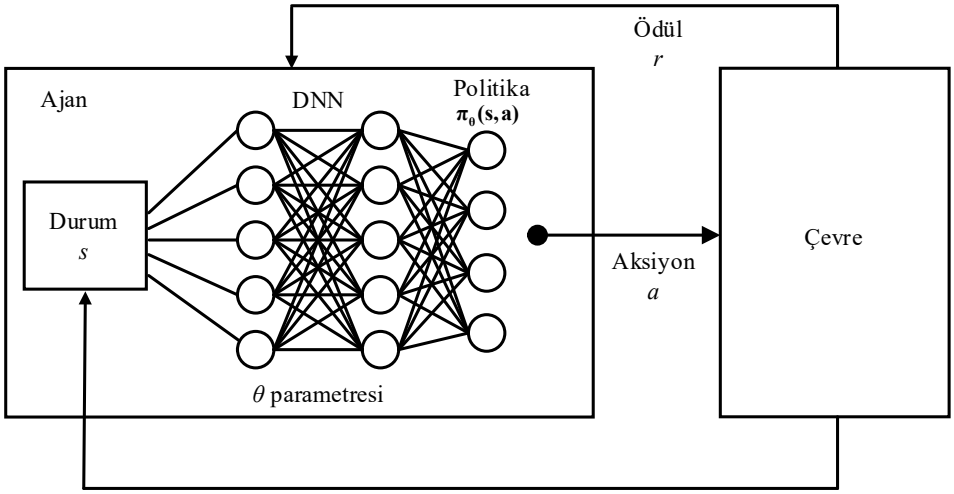
kullanılmasında bazı zorluklar da mevcuttur (Mnih vd., 2013). DL uygulamaları genellikle büyük miktarda etiketli veri gerektirir ve girdiyle hedef arasında doğrudan bir ilişki söz konusudur. RL algoritmaları ise genellikle gecikmeli olan gürültülü ve seyrek bir sayısal ödül sinyalinden öğrenirler. Bunun sonucu olarak, RL'deki aksiyon ve buna bağlı ortaya çıkan ödül sinyalleri arasında oluşan zaman gecikmesi mevcut problemlerden ilkidir. Diğer bir problem ise çoğu DL algoritmasının veri örneklerinin birbirinden bağımsız olduğunu göz önüne almasıdır. Bunun aksine RL algoritmalarında ise genellikle birbiriyle yüksek korelasyona sahip durumlar (veriler) ile karşılaşılmaktadır. Ayrıca RL algoritmaları yeni bir davranış öğrendikçe veri dağılımları değişir. Bu durum veri dağılımın sabit olduğunu varsayan DL algoritmaları için problem teşkil etmektedir.

1.6.3.1. Derin Q Ağı

Derin sinir ağları ile değer tabanlı bir RL yöntemi olan Q öğrenmenin birleştirilmesiyle elde edilen derin Q ağı (Deep Q Network, DQN), ilk kez Atari oyunu oynamayı öğrenmek için kullanılmıştır (Mnih vd., 2015). Bu algoritmada ajanın politikayı öğrenebilmesi için derin sinir ağları kullanılarak durum ve aksiyonlar arasında lineer olmayan bir haritalama gerçekleştirilir.

Eğitim süreci boyunca, çevre ile etkileşimde olan ajanın çevreden edindiği veriler (gözlem ve ödül) Q ağının eğitilmesi için kullanılır ve ağın çıkışı ajanın uygulayabileceği her olası aksiyonun Q değerini verir. Maksimum Q değerine sahip olan aksiyon bir sonraki aksiyon olarak seçilir. Şekil 1.12'de politikası DNN'e dayanan RL'ye ait blok diyagramı verilmiştir.

RL'deki değişken veri dağılımı ve veriler arasındaki korelasyondan oluşan problemin üstesinden gelebilmek için DQN algoritmasında önceki tecrübeleri rastgele örnekleyen ve böylece çoğu geçmiş davranış sayesinde veri dağılımının değişimini yumuşatan bir deneyim tekrarlama mekanizması (Lin, 1993) kullanılmıştır. Deneyim tekrarlama mekanizmasını, yabancı bir dilde yeni kelime öğrenme sürecindeki bir kişinin kelime tekrarını yaparken ki durumu göz önüne alınarak açıklamaya çalışılırsa, tekrarın kelime listesinden mi yoksa kelime kartlarından mı yapıldığı öğrenme süreci için önemlidir. Çünkü, eğer kelime listesinden tekrar yapılırsa kelimeler arasındaki korelasyon öğrenmeyi değil ezberi beraberinde getirirken, kelime kartlarından rastgele bir kart seçerek kelimeyi tekrar etmek ise gerçek öğrenmeye daha fazla katkıda bulunacaktır.



Şekil 1.12. Politikası DNN'e dayanan RL

Deneyim tekrarlama mekanizmasına göre ajanın her *t* anındaki deneyimleri $e_t = (s_t, a_t, r_t, s_{t+1})$, her bölüm yani çevrenin simülasyonu boyunca toplanarak tekrarlama belleğindeki $D=e_1,...,e_N$ veri setinde depolanır ve öğrenme süresince depolanan bu verilerden rastgele seçilen deneyim örneklerine Q öğrenme güncellemeleri uygulanılır. Deneyim tekrarlama işlemi gerçekleştirildikten sonra ajan bir aksiyon seçerek uygular. Ajan başlangıçta aksiyonlara rastgele karar verirken çevreyi keşfettikçe Q ağının çıkışına göre nasıl aksiyon alacağına karar verir. Yani bu algoritma ϵ- Greedy seçimini kullanır.

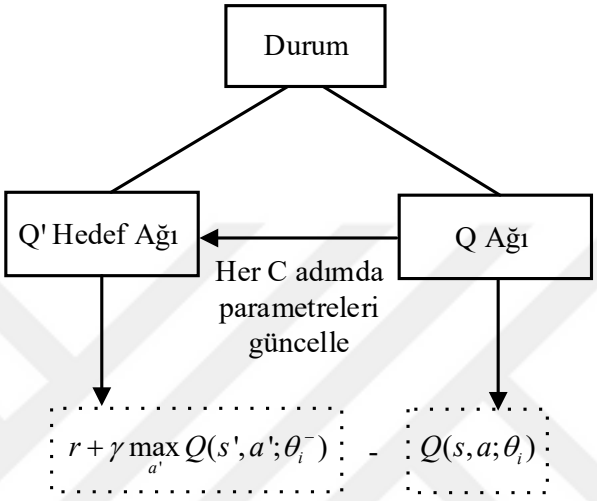
Bu algoritmada ağın parametrelerini güncelleyerek öğrenme sağlayabilmesi için minimize edilmesi gereken kayıp fonksiyonu, eşitlik (1.41)'de verilen Bellman eşitliğinden türetilen *Q* değerini güncelleme fonksiyonudur. Hedef değeri ile gerçek değerler arasındaki farka dayanan bu fonksiyonu optimize etmek için GD algoritmasına dayanan geriye yayılım algoritması kullanılır.

$$L_i(\theta_i) = \left(r + \gamma \max_{a'} Q(s', a'; \theta_i^-) - Q(s, a; \theta_i)\right)^2 \tag{1.41}$$

Bu eşitlikteki θ_i Q ağının *i.* iterasyondaki parametreleriyken, θ_i^- *i.* iterasyondaki hedefi hesaplamak için kullanılan hedef ağının parametreleridir.

DL'de hedef değerler sabit etiket değerleri olduğu için öğrenme süreci kararlıdır. Fakat RL'de etiket değeri olmadığı için doğru hedef değerini bulana kadar hedef değerler her iterasyonda değişir. Bu da karasızlığa sebep olur. Bunu önlemek için öngörülen ve

hedef değeri aynı ağ ile hesaplamak yerine iki tane mimarisi ve parametreleri aynı olan ağ kullanılır. Böylece bu iki değer arasında çok farklılık olmaz. Hedef ağın parametreleri θ_i^- her iterasyonda güncellenmeyerek sadece her C adımda bir Q ağının parametreleri kopyalanarak güncellenir. Bu da hedef fonksiyonu bir süreliğine sabit tutulduğu için öğrenme sürecini daha kararlı hale getirir. Bu işlem Şekil 1.13'te gösterilmiştir.



Şekil 1.13. DQN'de hedef ve aksiyon Q değerinin belirlenmesi

1.6.3.2. Derin Deterministik Politika Gradyanı

DQN, yüksek boyutlu durum uzayına sahip problemleri çözebilirken yalnızca ayrık ve düşük boyutlu aksiyon uzayında kullanılabilir. Fakat özellikle birçok fiziksel kontrol uygulaması sürekli ve yüksek boyutlu aksiyon uzayı içermektedir. DQN, sürekli bir uzayda her adımda iteratif bir optimizasyon süreci ile aksiyon değer fonksiyonunu maksimize eden aksiyonu bulmaya çalıştığı için sürekli uzaya sahip kontrol problemleri için doğrudan uygulanamaz. Bu nedenle Lilicrap vd. (2016) tarafından derin deterministik politika gradyanı olarak adlandırılan, yüksek boyutlu, sürekli aksiyon uzayında politikaları öğrenebilen derin fonksiyon belirleyicileri kullanılarak modelsiz, pasif politikalı bir aktör-kritik (Actor-Critic, AC) algoritması geliştirilmiştir. Bu algoritma, deterministik politika gradyanı (Deterministic Policy Gradient, DPG) algoritmasının (Silver vd., 2014) DQN'nin sunduğu faydaları kullanarak modifiye edilmesiyle elde edilmiştir.

Şekil 1.14'te verilen AC mimarisine dayanan bu algoritmada aktör fonksiyonu $\mu(s|\theta^{\mu})$, durumları belirli bir aksiyona deterministik olarak haritalayan politikayı belirler.

Amaç, beklenen ödülü maksimize eden optimal politikayı bulmak olduğu için aktör politikası amaç fonksiyonunun (J) aktör parametrelerine göre türevinin alınması ile eşitlik (1.42)'deki gibi güncellenir (Silver vd., 2014).

$$\begin{aligned}\nabla_{\theta^\mu} J &\approx \mathrm{E}_{s_t\sim\rho^\beta}\left[\nabla_{\theta^\mu} Q(s,a\mid\theta^Q)\big|_{s=s_t,a=\mu(s_t\mid\theta^\mu)}\right] \\ &= \mathrm{E}_{s_t\sim\rho^\beta}\left[\nabla_a Q(s,a\mid\theta^Q)\big|_{s=s_t,a=\mu(s_t)}\,\nabla_{\theta^\mu}\mu(s\mid\theta^\mu)\big|_{s=s_t}\right]\end{aligned} \tag{1.42}$$

Bu eşitlikte θ^μ ve θ^Q sırasıyla gerçek aktör ve kritik ağlarının parametreleridir. ρ^β ise durum ziyaret dağılımını ifade eder.

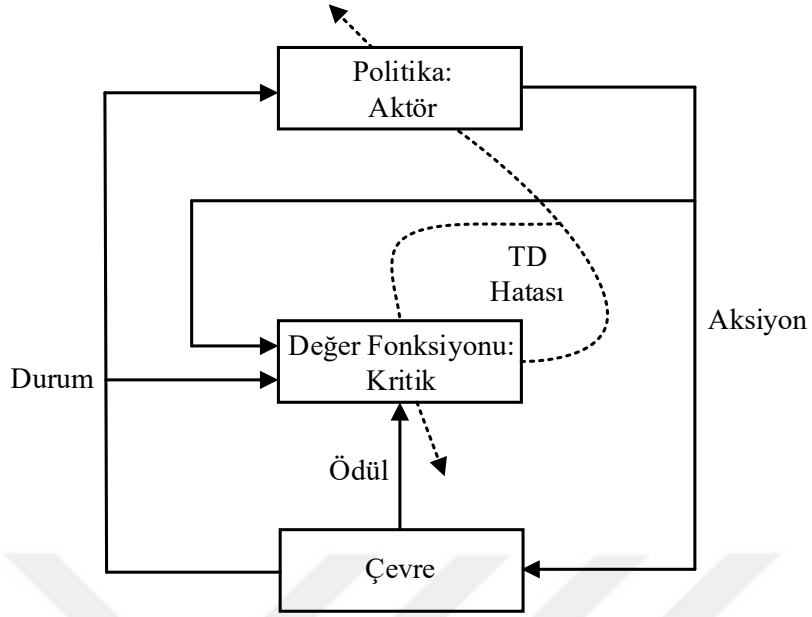
Kritik fonksiyonu $Q(s,a)$, aktör tarafından belirlenen politikayı değerlendirmek için Q öğrenmedeki gibi Bellman eşitliklerini kullanır. Eşitlik (1.43)'teki TD hatası olarak da adlandırılan kayıp fonksiyonu (L) minimize edilerek güncellenir.

$$\begin{aligned}L &= \left(y_t - Q(s_t,a_t\mid\theta^Q)\right)^2 \\ y_t &= r(s_t,a_t) + \gamma Q'(s_{t+1},\mu'(s_{t+1}\mid\theta^{\mu'})\mid\theta^{Q'})\end{aligned} \tag{1.43}$$

Bu eşitlikteki $\mu'(s|\theta^{\mu'})$ ve $Q'(s,a|\theta^{Q'})$ sırasıyla aktör ve kritiğin hedef değerlerini hesaplamak için kullanılan hedef ağları iken $\theta^{\mu'}$ ve $\theta^{Q'}$ ise sırasıyla aktör ve kritik hedef ağlarının parametreleridir. Hedef ağlar, DQN algoritmasındaki gibi Q öğrenmenin RL'ye doğrudan uygulanması sonucu oluşan öğrenme karasızlığını ortadan kaldırmak için kullanılırlar ve parametrelerini güncellemek için DQN'deki gibi gerçek ağın katsayılarının doğrudan kopyalanması yerine eşitlik (1.44)'te verildiği gibi daha yumuşak bir güncelleme yapılır. Böylece hedef değerlerin daha yavaş değişmesi sağlanarak öğrenme kararlılığı büyük ölçüde geliştirilir. Hedef ağ, değer tahminlerinin yayılmasını geciktirdiği için öğrenmeyi bir miktar yavaşlatabilir. Fakat buna rağmen öğrenme kararlığına katkısı daha ağır basmaktadır (Lilicrap vd., 2016).

$$\begin{aligned}\theta^{Q'} &\leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'} \\ \theta^{\mu'} &\leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}\end{aligned} \tag{1.44}$$

Bu eşitlikte, $\tau \ll 1$ olmak üzere yumuşak hedef güncelleme katsayısıdır.



Şekil 1.14. DDPG için aktör-kritik mimarisi

DDPG algoritmasında da RL'deki değişken veri dağılımı ve veriler arasındaki korelasyon problemini çözebilmek için DQN'deki gibi tekrarlama belleği kullanılır. Keşif politikasına göre çevreden örneklenen değişken grubu (s_t, a_t, r_t, s_{t+1}) sonlu büyüklükte olan bu bellekte depolanır. Bu bellek dolduğu zaman öncelikle en eski örnekler çıkarılır. Eşitlik (1.42) ve (1.43)'teki aktör ve kritik güncellemeleri bu bellekten her zaman aralığında rastgele örneklenen küçük bir grup (minibatch) veri kullanılarak eşitlik (1.45) ve (1.46)'daki gibi yapılır.

$$\nabla_{\theta^\mu} J \approx \frac{1}{M} \sum_i \nabla_a Q(s,a \mid \theta^Q)\big|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu} \mu(s \mid \theta^\mu)\big|_{s=s_i} \tag{1.45}$$

$$L = \frac{1}{M} \sum_i \left(y_i - Q(s_i, a_i \mid \theta^Q) \right)^2 \tag{1.46}$$

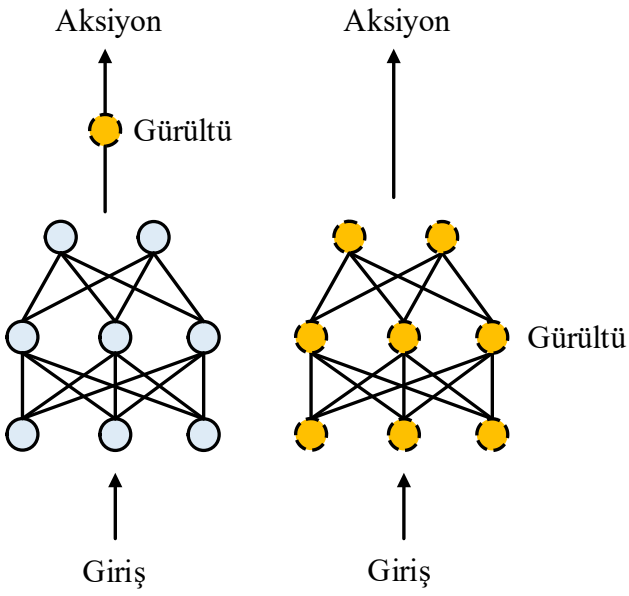
Bu eşitlikteki *M,* bellekten rastgele seçilen deneyim örneklerinin sayısını ifade eder ve mini-paket boyutu olarak adlandırılır.

Düşük boyutlu özellik vektör gözlemlerinden öğrenirken, eğer gözlemler farklı birimlere sahip fiziksel özellikler ise (örneğin, hız ve pozisyon gibi) değer aralıkları çevreye göre değişeceği için ağın etkili bir şekilde öğrenmesi zorlaşabilir. Bu problemi

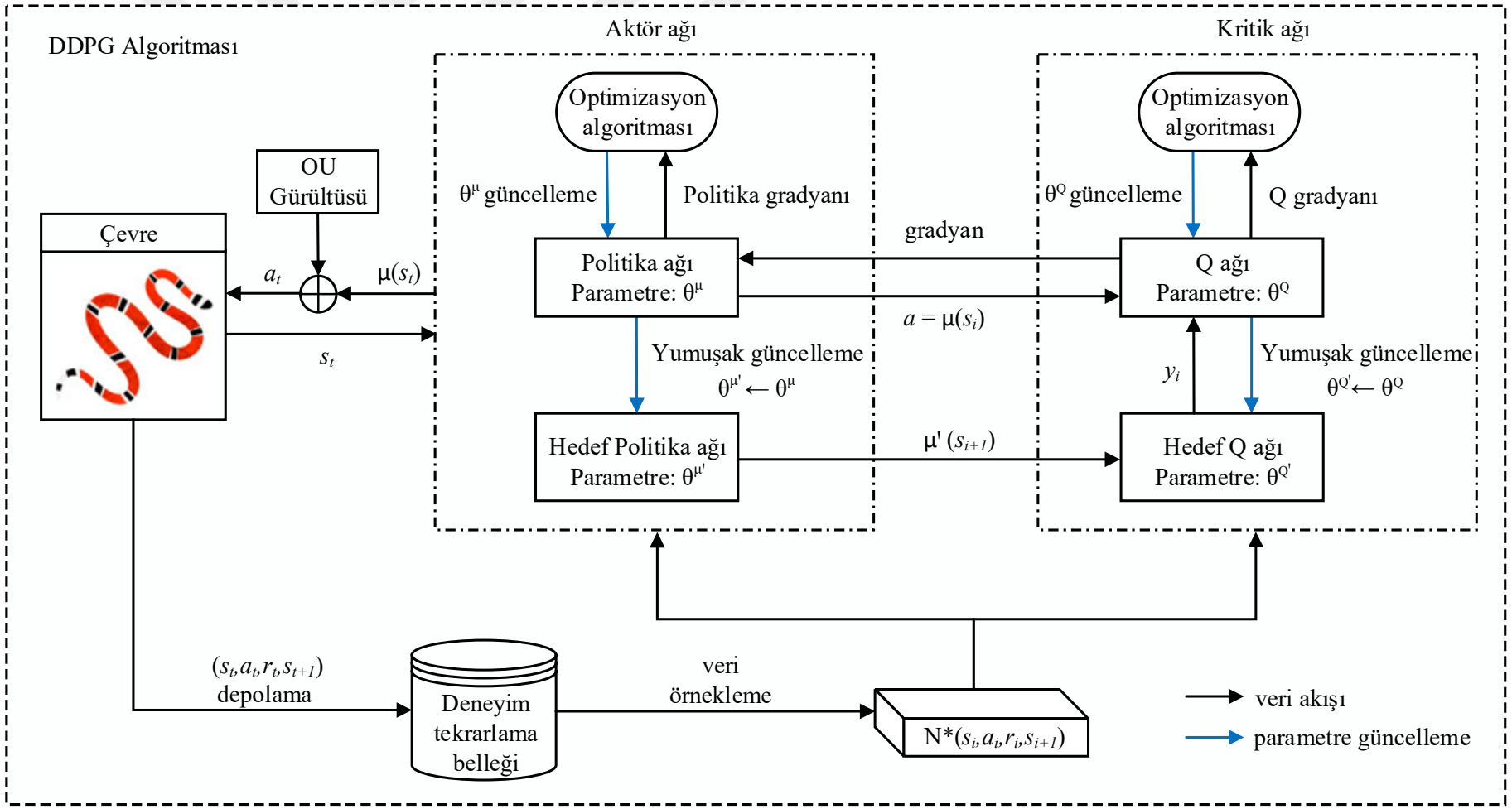
çözmek için paket normalizasyonu olarak adlandırılan yeni bir DL tekniği bu algoritmada benimsenmiştir (Ioffe ve Szegedy, 2015). Bu teknik, küçük bir gruptaki örneklerin her boyutunu birim ortalama ve varyansa sahip olacak şekilde normalize eder. Paket normalizasyonu ile birimleri belirli bir aralıkta manuel olarak ayarlamaya gerek kalmadan, farklı birimlerle birçok farklı görevde etkili bir şekilde öğrenme sağlanabilmektedir.

Sürekli aksiyon uzayında öğrenmenin en büyük zorluğu keşiftir (Lilicrap vd., 2016). DDPG gibi pasif politikalı algoritmaların bir avantajı, keşif probleminin öğrenme algoritmasından bağımsız olarak ele alınabilmesidir. Bu algoritmada keşif problemine çözüm olarak aktör politikasına gürültü (N) eklenerek elde edilen ve eşitlik (1.47)'de verilen keşif politikası önerilmiştir. Bu gürültü Şekil 1.15'te görüldüğü gibi aksiyon veya parametre uzayına eklenebilir. Fakat gürültünün parametre uzayına eklenmesinin aksiyon uzayına eklenmesinden daha iyi olduğu görüşü mevcuttur. Bunun nedeni, politikanın parametrelerine kasıtlı bir şekilde gürültü eklenmesi, ajanın keşfini farklı zaman aralıklarında tutarlı hale getirirken, aksiyon uzayına gürültü eklenmesi ise ajanın parametrelerine özgü herhangi bir şeyle ilişkili olmayan daha öngörülemez bir keşfe yol açmasıdır (Plappert vd., 2017). DDPG algoritmasının yapısı Şekil 1.16'da verilmiştir.

$$\mu'(s_t) = \mu(s_t \mid \theta_t^{\mu}) + N \tag{1.47}$$



Şekil 1.15. Gürültünün aksiyon ve parametre uzayına eklenmesi



Şekil 1.16. DDPG algoritmasının yapısı

1.6.3.3. İkiz Gecikmeli Derin Deterministik Politika Gradyanı Algoritması

DDPG algoritması, robotik ve otonom araçlar gibi sürekli aksiyon uzayına sahip kontrol problemleri için en çok kullanılan ve iyi sonuçlar üreten algoritmalardan biri olmasına rağmen, değer tabanlı diğer RL yöntemlerindeki gibi fonksiyon yaklaşım hatalarından kaynaklanan dezavantajlara sahiptir. TD3, DDPG algoritmasının dezavantajlarını gidermek için önerilmiş modelsiz, pasif politikalı bir RL yöntemidir (Fujimoto vd., 2018). TD3 algoritması, fonksiyon yaklaşım hatasının sonucu olarak ortaya çıkan aşırı tahmin yanlılığı ve yüksek varyans problemlerine odaklanılarak önerilen üç ana özelliğin DDPG algoritmasına eklenmesiyle elde edilmiştir. Bu problemler ve önerilen çözümler aşağıda detaylı olarak açıklanmıştır.

1.6.3.3.1. Aşırı Tahmin Yanlılığı

Fonksiyon yaklaşım hatası kritik ağının Q değerinin aşırı tahminine sebep olur. Bu aşırı tahmin hatası başlangıçta her güncelleme için minimum düzeyde olsa da kontrol edilemediği zaman birçok güncelleme üzerinden Bellman eşitliği ile yayılarak daha önemli bir hale gelebilir. Ayrıca hatalı bir değer tahmini, bir sonraki politika güncellemesinde optimal olmayan aksiyonların optimal olmayan kritik tarafından yüksek oranda derecelendirilerek güçlendirileceği bir geri besleme döngüsü oluşturarak zayıf politika güncellemelerine de yol açabilir. Böyle bir durum, ajanın lokal optimal çözüme takılmasına neden olur.

TD3 algoritmasında aşırı tahmin yanlılığı problemi için çift Q-öğrenmenin (Double Q-learning) (Van Hasselt, 2010) yeni bir versiyonu olan kırpılmış çift Q-öğrenme (Clipped Double Q-learning) önerilmiştir. Çift Q-öğrenmede, her birinin diğeri tarafından güncellendiği iki ayrı Q değer fonksiyonu kullanılarak aksiyon seçimi aksiyon değerlendirmesinden ayrılır. Böylece bir Q değer fonksiyonu tarafından seçilen aksiyonun değerlendirilmesi diğerinin değer tahmini kullanılarak yansız bir tahmin yapılabilir. Aynı yazarlar tarafından önerilen çift DQN (Double Deep Q Network, DDQN) yönteminde ise DQN ağı ve hedef ağı olarak adlandırılan iki tane özdeş ağ kullanılır (Van Hasselt vd., 2016). Bunlardan DQN ağı, bir sonraki durum için maksimum Q değerine sahip aksiyonu seçerken, hedef ağ ise seçilen bu aksiyonun Q değerini hesaplar. Böylece tahmin yanlılığı azalır. Bu tekniğin AC yöntemlerinde kullanılmasının yansız tahmini azaltma açısından

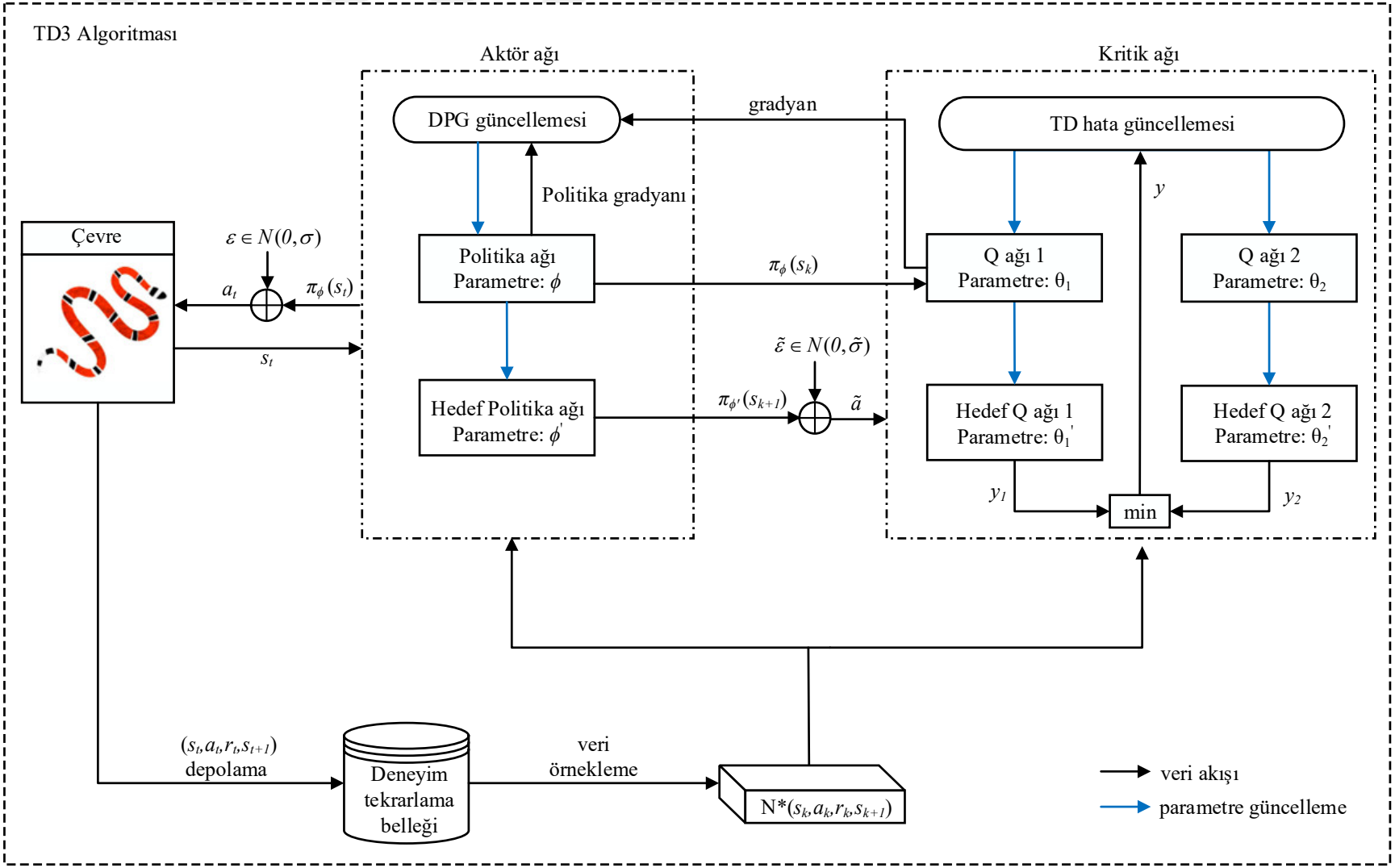
sadece küçük bir iyileşme sağladığı ve AC çift DQN yönteminin, DDPG ile benzer bir sonuç ürettiği görülmüştür (Fujimoto vd., 2018). Bunun nedeni politika ve hedef ağlarının güncellenmesinin bağımsız bir tahmin yapabilmek için çok benzer görünecek kadar yavaş olmasıdır. Buna karşın 2010'da önerilen orijinal çift Q-öğrenme yönteminin AC yöntemlerinde kullanılmasının yanlı tahmini azaltma konusunda daha etkili olduğu görülmüştür. Orijinal çift Q-öğrenme yöntemi iki aktör ağı (π_{ϕ_1},π_{ϕ_2}) ve iki kritik ağı (Q_{θ_1},Q_{θ_2}) ile eşitlik (1.48)'deki gibi kullanılabilir.

$$\begin{aligned} y_1 &= r + \gamma Q_{\theta_2'}(s', \pi_{\phi_1}(s')) \\ y_2 &= r + \gamma Q_{\theta_1'}(s', \pi_{\phi_2}(s')) \end{aligned} \tag{1.48}$$

Bu eşitlikteki π_{ϕ_1}aktör ağı Q_{θ_1}'e göre güncellenirken, π_{ϕ_2}aktör ağı Q_{θ_2}'e göre güncellenir. Eşitlik (1.48)'de görüldüğü gibi kritik ağlarının hedef güncellemesinde diğer hedef kritik ağın değerinin kullanılması bağımsız bir tahmin yapılmasını sağlar ve böylece politika güncellemesinden kaynaklanan tahmin yanlılığı önlenir. Fakat kritik ağlarının güncellemesinde diğer hedef kritik ağın ve aynı tecrübe belleğinin kullanılmasından dolayı kritik tamamen bağımsız değildir. Örneğin; $Q_{\theta_2} < Q_{\theta_1}$ olması durumu Q_{θ_1}'nin aşırı tahmin yapmasına sebep olacak ve arama uzayının belirli bölgelerinde giderek büyüyecektir. Bu problemi ortadan kaldırmak için TD3 algoritmasında önerilen kırpılmış çift Q-öğrenme yönteminde, eşitlik (1.49)'da görüldüğü gibi hedef güncellemesi yapılırken iki kritik ağının Q değerlerinden küçük olanı kullanılmaktadır.

$$y_1 = r + \gamma \min_{i=1,2} Q_{\theta_i'}(s', \pi_{\phi_1}(s')) \tag{1.49}$$

Bu yöntem aşırı tahmini önlerken Q değerlerinin eksik tahminine sebep olabilir. Fakat aşırı tahminin aksine, eksik tahmin politika güncellemesi boyunca yayılmadığı için bir problem oluşturmaz. Ayrıca fonksiyon yaklaşım hatasının rastgele bir değişken olduğu göz önüne alındığında, rastgele değişkenler kümesinin beklenen minimumu, varyans arttıkça azaldığı için eşitlik (1.49)'daki minimum alma işlemi, düşük varyanslı tahmin hatasına sahip durumların yüksek değer almasını ve dolayısıyla bu durumların tercih edilmelerini sağlar. Böylece daha kararlı bir öğrenme süreci elde edilir. TD3 algoritmasının yapısı Şekil 1.17'de verilmiştir.



Şekil 1.17. TD3 algoritmasının yapısı

1.6.3.3.2. Yüksek Varyans Problemi

Fonksiyon yaklaşım hatasının sonucu olarak ortaya çıkan bir diğer problem ise yüksek varyanslı tahmindir. Bu problem politika güncellemeleri için gürültülü bir gradyana sebep olarak öğrenme hızını düşürmektedir. TD3 algoritmasında yüksek varyans problemi için iki iyileştirme yapılmıştır.

1.6.3.3.2.1. Gecikmeli Politika Güncellemesi

DRL'de eğitimin kararlığı için hedef ağlarının önemli olduğu bilinmektedir. Fakat AC yöntemlerinde politika ve değer ağlarındaki güncellemelerin birbirlerini etkilemesinden dolayı istenilen bu kararlılık her zaman sağlanamayabilir. Çünkü zayıf bir politika, değer tahmininin aşırı tahmin yüzünden yakınsayamamasına sebep olurken değer tahmininin yanlış olması durumu da politikanın daha da zayıflamasına neden olacaktır.

Bu problemi çözmek için TD3 algoritmasında gecikmeli politika güncellenmesi önerilmiştir. Bu yaklaşım politika ağının kritik ağından daha az sıklıkla güncellenmesini sağlayarak, politika güncellemesinden önce kritik ağının değer tahmini hatasını mümkün olduğunca azaltmasına olanak sağlar. Sonuç olarak değer ağı her zaman adımından sonra güncellenirken, politika ve hedef ağları kritik güncellemesinden belirlenen sabit d güncelleme sayısı kadar sonra güncellenir. Politikanın daha az sıklıkla güncellenmesi daha düşük varyanslı değer tahminlerinin kullanılmasını sağlayarak daha iyi politikaların üretilmesini sağlar. Bu da daha kararlı ve etkili bir eğitim ile sonuçlanır.

TD hatasını düşük tutmak için eşitlik (1.50)'de verilen hedef ağların güncellenmesi DDPG'deki gibi daha yumuşak bir şekilde yapılır.

$$\dot{\theta} \leftarrow \tau\theta + (1-\tau)\dot{\theta} \tag{1.50}$$

1.6.3.3.2.2. Hedef politika yumuşatması

Deterministik politikalı algoritmalardaki öğrenme hedefi, kritik güncellenirken fonksiyon yaklaşım hatasından kaynaklanan yanlışlıklara karşı oldukça hassastır ve değer tahminindeki ani yükselmelere fazla uyum göstererek yüksek varyanslı hedef değerler

üretme eğilimindedir. Bu varyansı azaltmak için TD3 algoritmasında hedef politika yumuşatması olarak bilinen bir strateji kullanılır. Buna göre hedef değer güncellemesi, hedef aksiyona az miktarda rastgele gürültü eklenerek ($\tilde{\varepsilon}$) ve mini paketler üzerinden ortalama alınarak eşitlik (1.51)'deki gibi modifiye edilir. Hedefi orijinal aksiyona yakın tutmak için eklenen gürültü kırpılır. Hedef politikaya eklenen bu gürültü keşif politikasından bağımsız olarak seçilir.

$$y = r + \gamma Q_{\theta'}(s', \pi_{\phi'}(s') + \tilde{\varepsilon})$$
$$\tilde{\varepsilon} \sim clip(N(0, \tilde{\sigma}), -c, c) \tag{1.51}$$

Bu eşitlikteki $N(0,\tilde{\sigma})$ Gauss gürültüsünü temsil eder ve [$-c$, c] aralığında kırpıldıktan sonra hedef aksiyona eklenir.

Değer tahmininin gürültülü bir politikaya göre öğrenilmesi, hedef değerin gürültülere karşı dayanıklı aksiyonlar için yüksek bir değer vermesini sağlar. Bu da daha kararlı politikaların üretilmesine yardımcı olur.

1.6.4. Aksiyon Gürültü Modelleri

Diğer öğrenme yaklaşımlarında olmayıp da RL sürecinde ortaya çıkan en önemli problemlerden biri keşif ve sömürü kavramlarının uygulamaya geçirilmesindeki ikilemdir. Bu ikilem, ajan daha fazla ödül alabilmek için geçmiş tecrübelerinden faydalanarak iyi olduğu bilinen aksiyonları mı seçmeli yoksa gelecekte daha büyük ödüller alıp alamayacağını keşfetmek için şimdi daha düşük ödüle sahip bir aksiyonu mu tercih etmeli probleminden ortaya çıkmaktadır. Bu nedenle optimal politikayı bulmak için keşif-sömürü arasındaki dengeyi sağlayan bir aksiyon seçme stratejisine ihtiyaç duyulur. Literatürde pek çok farklı aksiyon seçme stratejisi mevcuttur. Bunlardan sıklıkla kullanılan ve çoğu zaman (1- ϵ zamanda) beklenen en yüksek ödülü veren aksiyonu seçip geri kalan zamanda da rastgele aksiyonları seçen bir yaklaşımı benimseyen ϵ-greedy algoritmasının derin RL algoritmalarından DQN ve DDQN gibi ayrık aksiyon uzayına sahip problemlerde kullanıldığı görülmektedir. Keşif problemi, problem sürekli aksiyon uzayına sahip olduğu zaman daha da karmaşıklaşmaktadır (Lilicrap vd., 2016). Sürekli aksiyon uzayı için uygun olan DDPG, TD3 gibi algoritmalarda yönlendirilmemiş keşif modellerinin kullanıldığı

görülmektedir. Ayrıca pasif politikalı olan bu algoritmaların keşif problemini öğrenme algoritmasından bağımsız olarak ele alabilmesi bu algoritmalara avantaj sağlamaktadır. DDPG algoritmasında keşif problemi, Ornstein-Uhlenbeck (OU) dağılımına göre rastgele örneklenen gürültünün aktör politikasına eklenmesiyle ele alınırken, TD3 algoritmasında ise keşif problemi için klasik Gauss gürültüsü önerilmiştir. Bu aksiyon gürültü modelleri aşağıda açıklanmıştır.

1.6.4.1. Ornstein-Uhlenbeck Aksiyon Gürültüsü

Ornstein-Uhlenbeck süreci x_t, sürtünme etkisi altındaki Brown hareketi yapan büyük bir parçacığın hızını modellemek için önerilmiştir (Uhlenbeck ve Ornstein, 1930). Bu süreç, sürekli zamanda rastgele yürüyüş (random walk) veya Wiener sürecinin özelliklerinin değiştirilmesiyle elde edilmiştir. Bu sürece göre yürüyüşün merkezi bir konuma geri dönme eğilimi vardır ve bu geri dönüş merkezden uzaklaştıkça daha büyük bir çekimle gerçekleşir. OU süreci eşitlik (1.52)'deki gibi tanımlanmaktadır.

$$dx_t = \theta(\mu - x_t)dt + \sigma dW_t \tag{1.52}$$

Bu eşitlikteki $\theta > 0$ ve $\sigma > 0$ olmak üzere dW_t Wiener sürecini ifade etmektedir.

Mathworks R2022a kullanım kılavuzunda (URL-1, 2022) OU aksiyon gürültüsünün her k adımında, eşitlik (1.53)'deki gibi güncellendiği belirtilmiştir.

$$x(k+1) = x(k) + \theta(\mu - x(k)).*\mathrm{T_s} + \sigma(k).*(\mathrm{rand}(\mathrm{size}(\mu))).*\sqrt{T_s} \tag{1.53}$$

Bu eşitlikteki μ gürültü modelinin ortalamasını, σ gürültü modelinin standart sapmasını, θ (mean attraction constraint) ise gürültü model çıkışının ortalamaya ne kadar hızlı döneceğini belirleyen bir sabittir. T_s ise ajanın örnekleme zamanıdır. Standart sapma her k adımda belirlenen azalma oranına (decay rate, α_σ) bağlı olarak eşitlik (1.54)'te gösterildiği güncellenir. $x(1)$ başlangıç aksiyon değeri tarafından belirlenir. Bu süreç bölümün sona ermesiyle sıfırlanarak her bölümün başında yeniden başlar.

$$\sigma_d = \sigma(k).*(1-\alpha_\sigma)$$
$$\sigma(k+1) = \max(\sigma_d, \sigma_{\min}) \tag{1.54}$$

Sürekli aksiyon sinyalleri için, ajanın keşfini sağlayabilmek amacıyla gürültü standart sapmasının uygun bir şekilde belirlenmesi önemlidir. Bunun için standart sapma ve örnekleme sayısına göre eşitlik (1.55)'te verilen ifadenin, problemin aksiyon uzay aralığının (A_r) %1'i ile %10'u arasında olması önerilmektedir.

$$0.01*A_r \le \sigma\sqrt{T_s} \le 0.1*A_r \tag{1.55}$$

Belirlenen standart sapmanın yarıya düşmesi için gereken örnek sayısı ise eşitlik (1.56)'daki gibi hesaplanabilir.

$$\sigma_{0.5} = \frac{\log(0.5)}{\log(1-\alpha_\sigma)} \tag{1.56}$$

1.6.4.2. Gauss Aksiyon Gürültüsü

Gauss aksiyon modeli, olasılık yoğunluk fonksiyonu gauss dağılımı olarak bilinen normal dağılıma eşit gürültü üretir. Gauss aksiyon gürültüsü her k adımında, eşitlik (1.57)'deki gibi örneklenir.

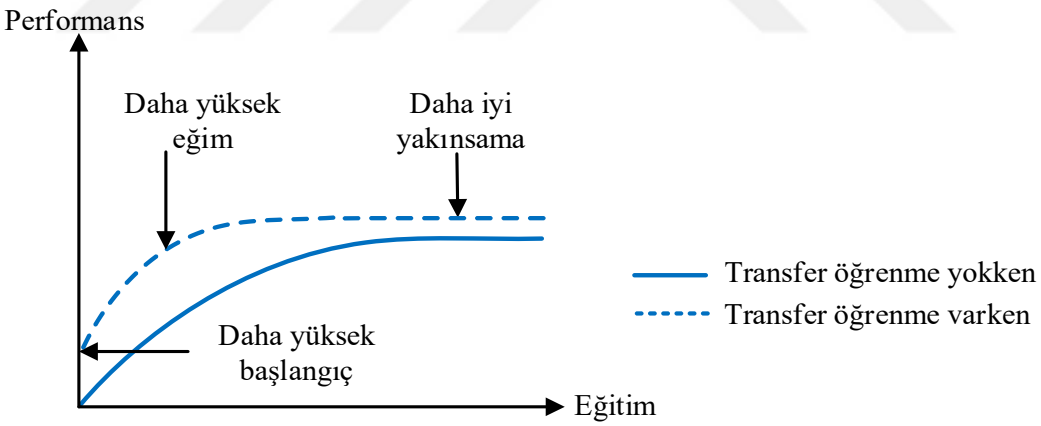
$$w = \mu + \text{rand}(a_n).*\sigma(k)$$
$$x(k+1) = min(max(w, -c), c) \tag{1.57}$$

Bu eşitlikteki a_n aksiyon boyutu, $-c$ ve c ise gürültü örneğinin alt ve üst sınırlarını ifade etmektedir. Standart sapma, OU aksiyon gürültü modelindeki gibi her k adımda belirlenen azalma oranına bağlı olarak eşitlik (1.54)'teki gibi güncellenir.

Ornstein-Uhlenbeck süreci zamansal olarak ilişkili sonuçlar sunarken, Gauss gürültü modeli ise zamandan bağımsız değişikliklere neden olur. Ornstein-Uhlenbeck sürecinin eylemsizliğe sahip fiziksel kontrol problemlerinde keşfi modellemek için daha elverişli olduğu Lilicrap vd. (2016)'nin çalışmasında belirtilmiştir.

1.6.5. Transfer Öğrenme

Transfer öğrenme, daha önceden öğrenilen bir modelin farklı ama ilgili yeni bir problemin çözümünde kullanılmasıyla genelleştirme yeteneğini eğitime katmayı amaçlayan bir öğrenme yöntemidir. Transfer öğrenme daha çok görüntü işleme ve veri analizi gibi çok büyük ve zorlu veri kümelerinin eğitiminde ihtiyaç duyulan güçlü kaynaklar yüzünden özellikle DL'de popüler olarak kullanılmaktadır. Transfer öğrenmenin RL'de kullanımı çok yaygın olmamakla birlikte var olan bazı çalışmalar (Konidaris vd., 2012; Koga vd. 2015; Glatt vd., 2016; Martinsen vd. 2018), daha az veri kullanarak eğitimi hızlandırma, daha karmaşık zorlu çevrelerde öğrenmeyi mümkün hale getirme ve genelleştirme yeteneğini arttırarak farklı yetenekleri öğrenebilme yönünden tatmin edici sonuçlar sunmaktadır. Başarılı bir transfer öğrenme uygulamasında Şekil 1.18'de görüldüğü gibi eğitim eğrisinde daha yüksek bir başlangıç, daha dik bir eğim ve daha yüksek bir değere yakınsamayı içeren üç faydanın görülmesi beklenir (Torrey ve Shavlik, 2010).

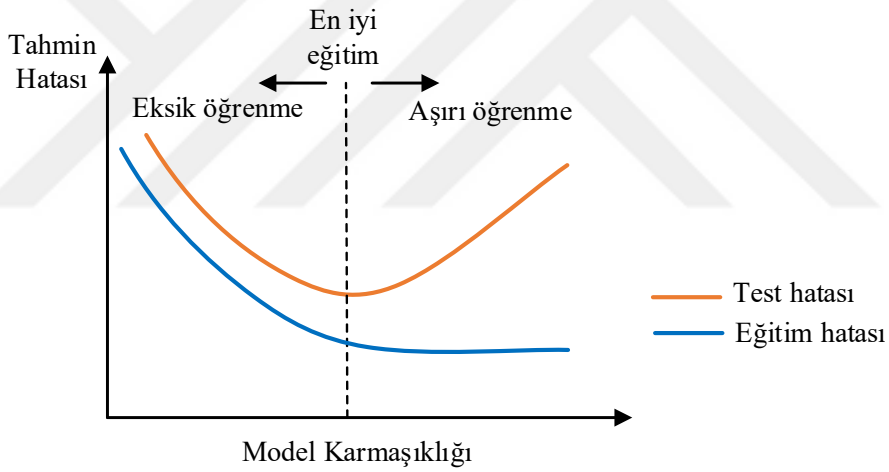


Şekil 1.18. Transfer öğrenmenin eğitim sürecine olan üç etkisi

1.6.6. Yanlılık-Varyans İkilemi

Makine öğrenmesindeki amaç, eğitim verilerinden elde edilen örüntülere göre oluşturulan modeli kullanarak yeni veriler için doğru tahmin yapabilmektir. Bu işlem sonucunda Şekil 1.18'de gösterilen istenmeyen iki durum meydana gelebilir. Bunlardan biri ezberleme dediğimiz modelin aşırı öğrenmesi durumu (yüksek varyans hatası) diğeri

ise modelin eksik öğrenmesi (yüksek yanlılık hatası) durumudur (Geman vd., 1992). Aşırı öğrenme, veri seti üzerinde gereğinden fazla çalışılması ya da yeterince çeşitli olmayan tek tip verilere sahip olan veri setlerinin kullanılması durumunda meydana gelir. Bu durumda model, eğitim verisini çok iyi öğrenirken, test verisi için yeterli öngörüde bulunamaz ve yüksek varyans hatası oluşur. Şekil 1.19'da görüldüğü gibi test hatasının eğitim hatasından ayrıldığı noktada aşırı öğrenme meydana gelmektedir. Aşırı öğrenmenin aksine eksik öğrenme doğrusal olmayan bir fonksiyonun daha basit doğrusal bir model kullanılarak modellenmesi sonucu modelin girdiler ve çıktılar arasındaki temel ilişkiyi öğrenememesi durumunda meydana gelir ve böyle bir modelin yanlılık hatası yüksek olur. Yüksek yanlılık hatasına sahip bir modelde eğitim ve test verilerinin sonuçları tutarlı olmakla birlikte iki veri setinde de hata oranı yüksektir. Sonuç olarak iyi bir model, yanlılık ve varyans arasında denge kurabilen ne aşırı basit ne de aşırı karmaşık olan modeldir.



Şekil 1.19. Yanlılık ve varyans hatalarının gösterimi

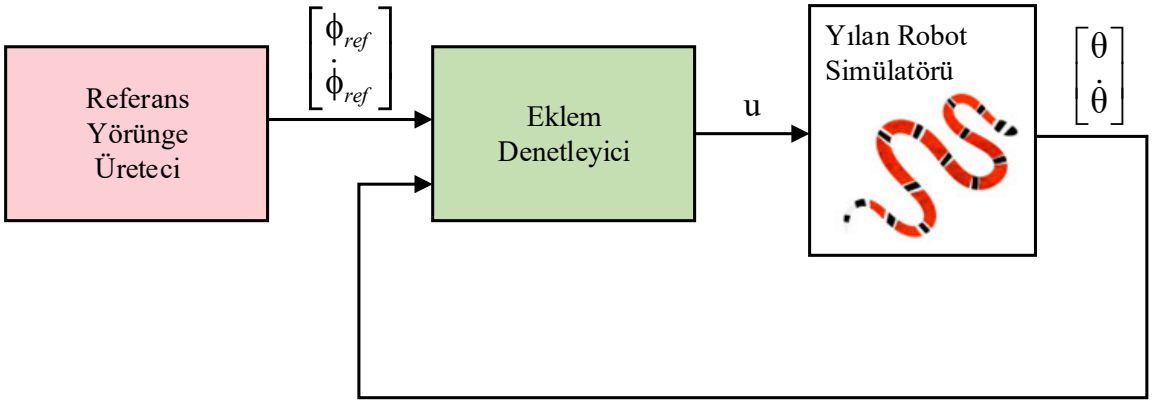
2. YAPILAN ÇALIŞMALAR, BULGULAR VE İRDELEME

2.1. Giriş

Yılan robotların benzersiz yeteneklerine rağmen var olan potansiyellerini tam olarak elde edemedikleri görülmektedir. Biyolojik benzerleri gibi ortama göre adaptif hareket edebilmeleri bu potansiyellerini elde edebilmeleri için önem taşımaktadır. Bu nedenle bu tez çalışmasında yılan robotun farklı çevrelerde enerji verimli adaptif hareket problemi için optimizasyon ve öğrenme tabanlı denetleyiciler geliştirilmiştir. Bu amaçla öncelikle tekerleksiz bir yılan robot mekanizmasının dinamik modellemesi Matlab/Simulink ortamında yapılarak yılan robot simülatörü elde edilmiştir. Yılan robotun enerji verimli hareket problemi için optimal hareket parametreleri öncelikle sabit sürtünme katsayısına sahip bir ortamda daha sonra ise değişen çevre koşullarında araştırılmıştır. Bunun için robotun hızı ve güç tüketimi arasındaki dengeyi göz önüne alan SOS algoritmasına dayanan iki tane çok amaçlı sezgisel optimizasyon yöntemi kullanılmış ve elde edilen sonuçlar karşılaştırmalı olarak sunulmuştur. Ardından yılan robotun üretilen referans yörüngeleri takip edebilmesi için DRL tabanlı denetleyiciler tasarlanmıştır. DDPG ve TD3 algoritmaları kullanılarak geliştirilen dört farklı denetleyici kendi aralarında ve klasik kontrol yöntemi olan PD denetleyici ile karşılaştırılarak yörünge takip performansları analiz edilmiştir. Son olarak yılan robotun enerji verimli hareketi için robota uygulanan referans hız değerleri değiştirilerek DRL tabanlı bir denetleyici ile biyolojik yılanların yürüyüş biçimlerine benzer doğal adaptif yörüngeler üretilmiştir.

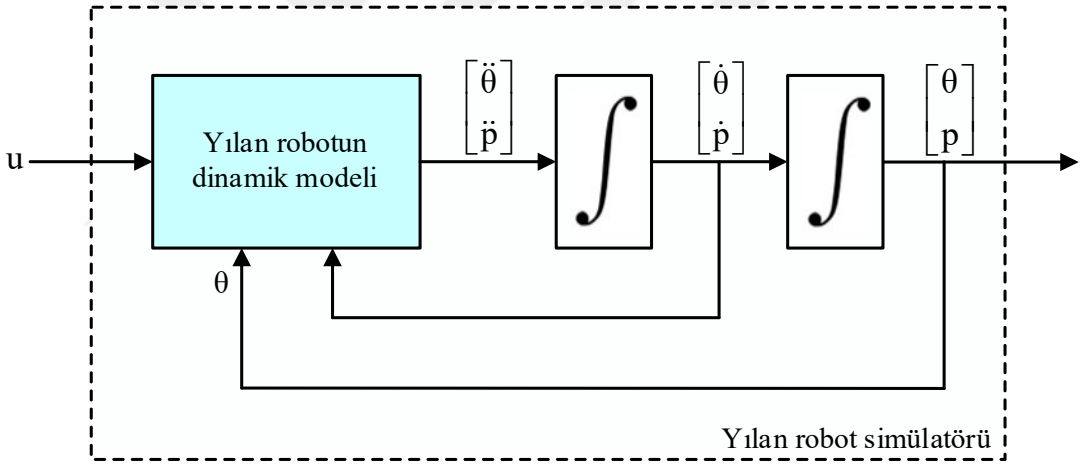
2.2. Yılan Robotun Modellenmesi ve Simülasyonu

Yılan robotun eşitlik (1.24)'teki dinamik modeli kullanılarak yılan robot simülatörü elde edilmiş ve üretilen referans yörüngelerin takibi için PD denetleyici kullanılmıştır. Bahsedilen sistemin blok diyagramı Şekil 2.1'de verilmiştir.



Şekil 2.1. Sistemin blok diyagramı

Şekil 2.1'deki blok diyagramında görülen yılan robot simülatörünün iç yapısı Şekil 2.2'de verilmiştir.



Şekil 2.2. Yılan robot simülatörünün iç yapısı

2.2.1. Referans Yörüngelerin Üretilmesi

Yılan robotun çoğu hareket biçimi biyolojik yılanlardan esinlenerek geliştirilmiştir. Biyolojik yılanlar çevresel koşullara bağlı olarak yanal dalgalanma, yanal kayma, doğrusal ve akordeon tipi hareket olmak üzere dört ana hareket türü gerçekleştirebilirler (Hirose, 1993). Bu hareket biçimlerinden yanal dalgalanma hareketi çoğu biyolojik yılan türünde görülen en yayın harekettir. Bu nedenle de yılan robot araştırmalarında en çok incelenen yürüyüş biçimidir (Transeth, 2007). Şekil 2.3'te biyolojik bir yılanın yanal dalgalanma hareketi görülmektedir.

Şekil 2.3. Yanal dalgalanma hareketi (Mattison, 2002).

Yanal dalgalanma yürüyüş biçimi Hirose (1993) tarafından yılanın gövdesinin sürekli bir eğri olarak modellenmesiyle geliştirilmiştir. Bu hareket biçimini elde edebilmek için yılan robotun her bir eklemine eşitlik (2.1)'de verilen sinüzoidal işaret uygulanır.

$$\phi_{i,ref} = \alpha \sin(\omega t + (i-1)\beta) + \gamma_c \tag{2.1}$$

Bu eşitlikteki $i = 1,\ldots,n$ olmak üzere α, β *ve* γ_c parametreleri yılan robot tarafından gerçekleştirilecek serpenoid eğrinin şeklini tanımlar. α referans işaretinin genliğini gösterirken, ω hareketin açısal frekansını ifade etmektedir. β art arda bağlı iki modül arasındaki faz farkını gösterirken, γ_c kontrol parametresi ise yılan robotun yönünü belirlemek için kullanılmaktadır.

β parametresinin eklemler arasındaki oluşturduğu faz farkı eklem 2 ve eklem 3'ün referans işaretlerinde basamak etkisi oluşturarak türev denetim elemanının kullanılmasıyla kontrol işaretinde ani bir darbeye neden olmaktadır. Bu etkiyi önlemek için referans işaret eşitlik (2.2)'de transfer fonksiyonu verilen ikinci dereceden alçak geçiren bir filtreden geçirilerek robota uygulanmıştır. Böylece daha yumuşak bir referans işaret elde etmeye ek olarak denetleyici tasarımında ihtiyaç duyulan referans işaretin türevi de filtrenin çıkışından elde edilebilmektedir.

$$\frac{x_{ref}}{r} = \frac{\omega_n^{\,2}}{s^2 + 2\xi\omega_n s + \omega_n^{\,2}} \tag{2.2}$$

Eşitlik (2.2)'deki r eşitlik (2.1)'den elde edilen eklemlere uygulanan referans işareti ifade ederken, x_{ref} ise filtrelenmiş referans işarettir. Bu filtre ile ϕ_{ref} ve $\dot{\phi}_{ref}$ için yumuşak referans işaretleri elde edilir. Kontrol sistemin çıkışını motorun fiziksel limitlerinde tutacak referans işaretlerini elde edebilmek için doğal frekans ve sönüm oranı $\omega_n = 25$ ve $\xi=1$ olarak belirlenmiştir.

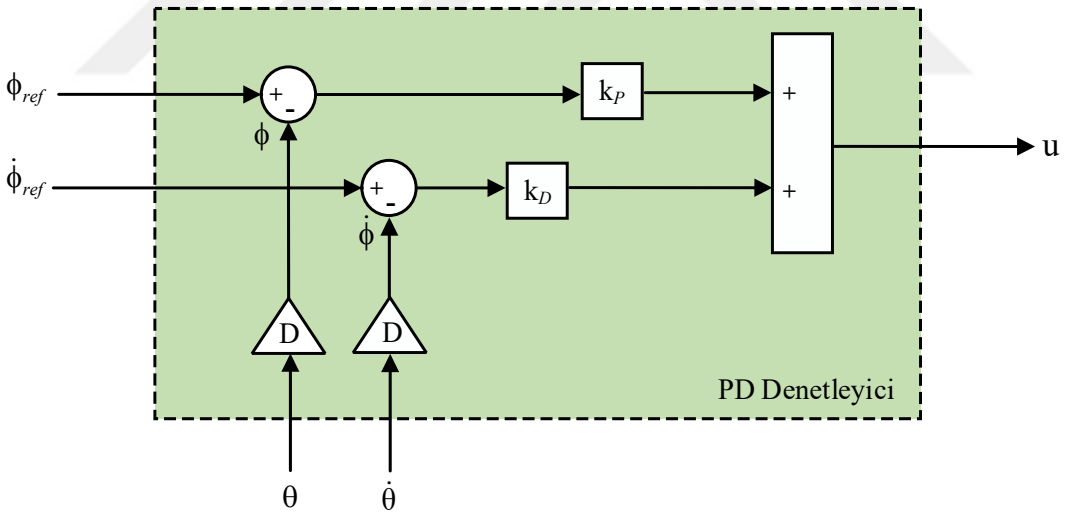
2.2.2. Eklem Yörünge Takip Denetleyicisi

Yılan robotun eşitlik (2.2)'de üretilen filtrelenmiş referans eklem açılarını takip edebilmesi için PD denetleyici kullanılmıştır. Buna göre *i*. eklem için kontrol kuralı eşitlik (2.3)'deki gibi ifade edilebilir.

$$u_i = k_P(\phi_{i,ref} - \phi_i) + k_D(\dot{\phi}_{i,ref} - \dot{\phi}_i) \tag{2.3}$$

Bu eşitlikteki $k_P > 0$ ve $k_D > 0$ denetleyici parametreleridir. $\dot{\phi}_{i,ref}$ ise eşitlik (2.2)'deki filtrenin çıkışından elde edilmektedir.

Şekil 2.1'deki 'Eklem Denetleyici' bloğunun iç yapısı Şekil 2.4'te görülmektedir. Bu şekildeki D kazanç blokları link açılarından eklem açılarının elde edilebilmesi için kullanılmış olup daha önceden verilen, bir vektörün peş peşe gelen elemanlarını çıkarmak için kullanılan, **D** matrisini ifade etmektedir.



Şekil 2.4. Eklem denetleyicisinin iç yapısı

2.2.3. Sistem ve Simülasyon Parametreleri

Bu çalışmada 5 özdeş linkli tekerleksiz bir yılan robot kullanılmıştır. Modellenen robotun parametreleri Tablo 2.1'de verilmiştir. Modelleme yapılırken aşağıdaki varsayımlar yapılmıştır:

- Linklerin genişliği modellemeye katılmamıştır.
- Her bir linkin kütlesi ve eylemsizlik momentinin aynı olduğu kabul edilmiştir.
- Her bir linkin kütlesinin düzgün dağılımlı olduğu ve bu nedenle de ağırlık merkezinin linkin tam ortasında olduğu kabul edilmiştir.
- Yanal dalgalanma hareketini sağlayabilmek için modellemede dağıtık temas modeline dayanan izotropik olmayan viskoz sürtünme modeli kullanılmış olup teğet ve normal yöndeki sürtünme katsayıları Tablo 2.1'de verilmiştir ve bu katsayıların her bir link için aynı olduğu kabul edilmiştir.

Tablo 2.1. Yılan robot model parametreleri

Parametreler	**Değerleri**
Link sayısı	$n = 5$
Link uzunluğu	$2l = 0.18$ m
Link kütlesi	$m = 0.8$ kg
Eylemsizlik momenti	$J = 0.00216$ kgm^2
Sürtünme katsayıları	$c_t = 0.1$; $c_n = 10$
PD denetleyici parametreleri	$k_P = 20$; $k_D = 5$

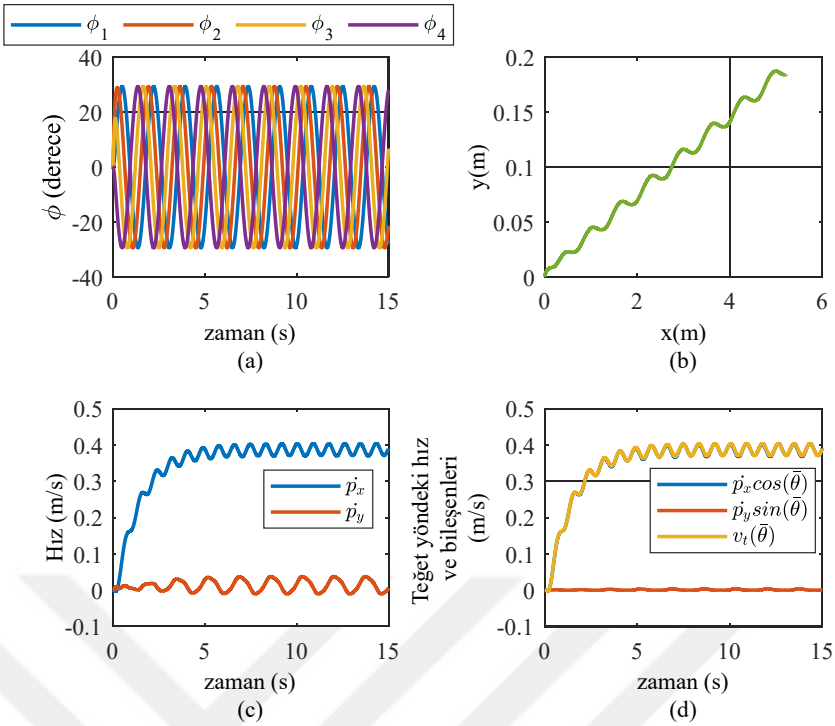
2.2.4. Simülasyon Sonuçları

Yanal dalgalanma hareketi için serpenoid eğrilerin üretilmesi Şekil 2.1'de görülen 'Referans Yörünge Üreteci' matlab fonksiyon bloğunda eşitlik (2.1) kullanılarak gerçekleştirilmektedir. Yörüngeleri tanımlayan parametreler Tablo 2.2'de verilmiştir.

Tablo 2.2. Referans yörünge parametreleri

Parametreler	**Değerleri**
Genlik	$\alpha = 30^\circ$
Açısal frekans	$\omega = 210$ °/s
Faz farkı	$\beta = 60^\circ$
Yönelme açısı	$\gamma_c = 0^\circ$

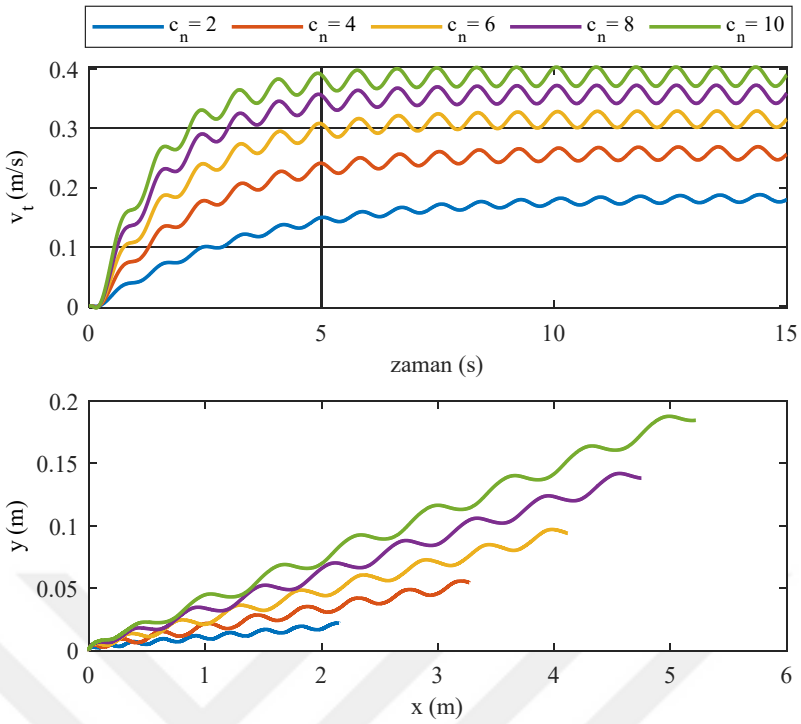
Tablo 2.1'deki parametrelere göre üretilen filtrelenmiş referans yörüngeler, yörüngelerin modellenen yılan robota uygulanmasıyla elde edilen yılan robotun ağırlık merkezinin hızı, yılan robotun teğet yöndeki hızı ve yılan robotun katettiği yol (ağırlık merkezinin pozisyonları) Şekil 2.5'te verilmiştir.



Şekil 2.5. (a) Referans eklem yörüngeleri (b) yılan robotun ağırlık merkezinin pozisyonları (c) yılan robotun ağırlık merkezinin hızı, (d) yılan robotun teğet yöndeki hız ve bileşenleri

Yılan robotun ağırlık merkezi hızının x ekseni boyunca artarken y ekseni boyunca sıfır civarında kaldığı Şekil 2.5 (c)'de görülmektedir. Robotun teğet (ileri) yöndeki hızı ise Şekil 2.5 (d)'de görüldüğü gibi sadece x yönündeki hız bileşeninden oluşmaktadır. Bu grafiklerden yılan robotun yaklaşık 0,34 m/s hıza ulaştığı ve x yönünde yaklaşık 5 m ilerlediği Şekil 2.5 (b)'de görülmektedir.

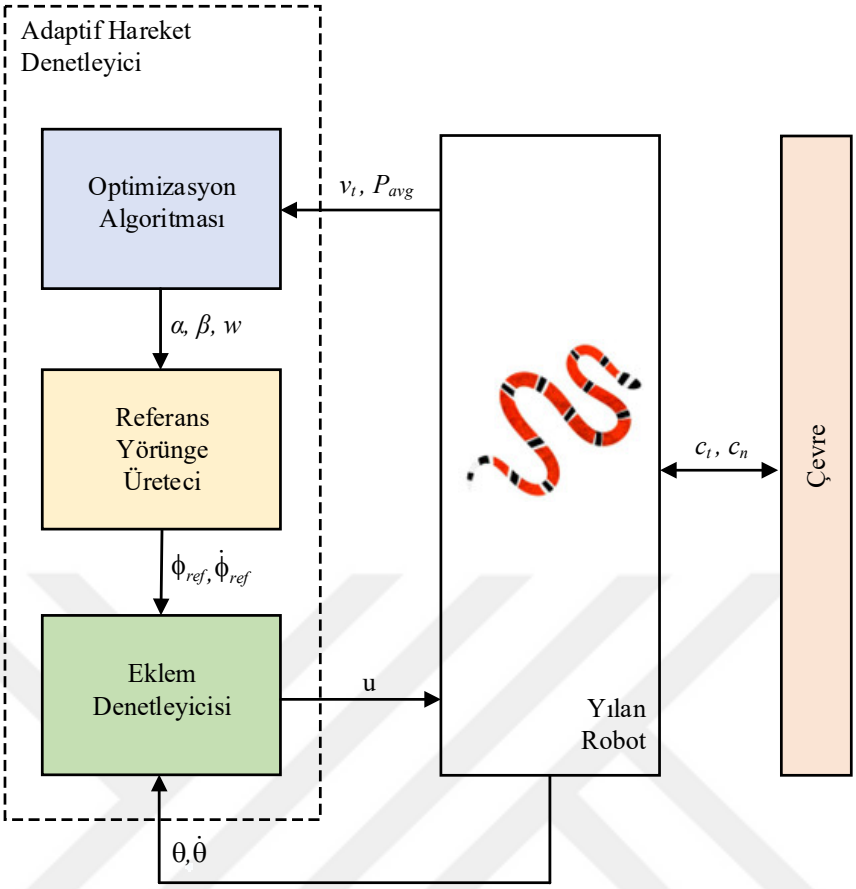
Yılan robot modelinde teğet yöndeki sürtünme katsayısı $c_t = 0.1$ olacak şekilde sabit tutularak, normal yöndeki sürtünme katsayısının (c_n) değiştirilmesinin robotun ileri yöndeki hızına ve dolayısıyla katettiği mesafeye etkisi incelenmiştir. Elde edilen sonuçlar Şekil 2.6'da verilmiştir. Bu şekilden görüldüğü gibi normal ve teğet yöndeki sürtünme katsayıları arasındaki fark arttıkça yılan robot daha hızlı hareket ederek aynı süre içerisinde daha fazla yol alabilmiştir. Bu sonuç Hu vd.'nin (2009) çalışmasında da belirtildiği gibi ileri yönde etkili bir hareket sağlayabilmek için izotropik olmayan sürtünme özelliğinin gerekliliği ile uyuşmaktadır. Ayrıca Ariizumi ve Matsuno'nın (2017) çalışmasında belirtilen izotropik olmayan sürtünmenin yeterince büyük olduğu ortamlarda yılan robotun daha hızlı hareket edebileceği sonucunu desteklemektedir.



Şekil 2.6. Normal yöndeki sürtünme katsayısı değişiminin robotun ileri yöndeki hızına ve katettiği mesafeye etkisi

2.3. Yılan Robotun Optimal Hareket Kontrolü

Yılan robotun enerji verimli hareketi için önerilen sistemin blok diyagramı Şekil 2.7'de verilmiştir. Sistem, çevreyle etkileşimi olan bir yılan robot modeli, referans yörünge üreteci, eklem denetleyici ve optimizasyon algoritmasından oluşmaktadır. Yılan robot simülatörü eşitlik (1.24)'teki dinamik modelin kullanılmasıyla elde edilmiş olup, referans yörünge üreteci ve eklem denetleyici 2.2.1 ve 2.2.2 bölümlerinde açıklandığı gibi uygulanmıştır. Şekil 2.7'de görüldüğü gibi optimizasyon algoritması belli bir çevrede hareket eden yılan robot simülatöründen elde edilen robotun hareket hızını ve tükettiği ortalama gücü kullanarak referans yörünge üreteci için optimum yürüyüş parametrelerini üretir. Bu çalışmada ağırlıklı toplam yöntemine dayanan MOSOS algoritması ve FNSMOSOS algoritması yılan robotun optimal hareket problemi için uygulanmıştır. Bu iki çok amaçlı optimizasyon algoritması ile öncelikle yılan robotun sabit sürtünme katsayılarına sahip bir ortamda optimal hareket parametreleri araştırılmış daha sonra ise değişen çevre koşullarında yılan robotun adaptif hareketi için optimal yürüyüş parametreleri incelenmiştir.



Şekil 2.7. Önerilen yöntem ile yılan robotun adaptif hareket kontrolü

2.3.1. Ağırlıklı Toplam Tabanlı MOSOS Algoritmasının Yılan Robotun Enerji Verimli Hareket Problemine Uygulanması

Yılan robotun enerji verimli hareket probleminin ağırlıklı toplam tabanlı MOSOS algoritması kullanılarak çözümünde öncelikle Tablo 2.3'te verilen üst ve alt sınırlar arasında n_d boyutuna sahip rastgele n_o tane organizma üretilir. Böylece, arama uzayı eşitlik (2.4)'teki gibi n_oxn_d boyutundaki ekosistem matrisinden oluşur. Bu matrisin her satırı bir organizmayı (çözümü) temsil ederken, satır vektörünün her elemanı ise yılan robota uygulanacak yürüyüşün parametrelerini temsil eder. Bu problemde, optimize edilecek yürüyüş parametreleri α, β ve ω olduğu için organizmaların boyutu n_d = 3 olarak belirlenmiştir.

$$ekosistem = \begin{bmatrix} organizma_1 \\ \vdots \\ organizma_{n_o} \end{bmatrix} = \begin{bmatrix} \alpha_{1,1} & \beta_{1,2} & \omega_{1,3} \\ \vdots & \ddots & \vdots \\ \alpha_{n_o,1} & \beta_{n_o,2} & \omega_{n_o,3} \end{bmatrix}_{n_o x 3} \tag{2.4}$$

Üretilen parametre seti ile eşitlik (2.1) kullanılarak referans eklem yörüngeleri üretilir ve buna göre hesaplanan eklem torklarının u_i yılan robota uygulanmasıyla yılan robotun ortalama güç tüketimi P_{avg} ve ileri yöndeki hızı v_f sırasıyla eşitlik (2.5) ve (2.6)'daki gibi hesaplanır.

$$P_{avg} = \frac{1}{T}\int_0^T \left(\sum_{i=1}^{n-1} \left| u_i(\mathrm{t})\, \dot{\phi}_i(\mathrm{t}) \right| \right) dt \tag{2.5}$$

Bu eşitlikteki T simülasyon süresidir ve i. eklem için açısal hız $\dot{\phi}_i = \dot{\theta}_i - \dot{\theta}_{i+1}$ şeklinde hesaplanır.

$$v_f = \frac{\sqrt{(p_x(\mathrm{T}) - p_x(0))^2 + (p_y(\mathrm{T}) - p_y(0))^2}}{T} \tag{2.6}$$

Bu eşitlikteki $(p_x(0),\ p_y(0))$ ve $(p_x(\mathrm{T}),\ p_y(\mathrm{T}))$ yılan robotun ağırlık merkezinin başlangıç ve son pozisyonlarını göstermektedir.

Arama uzayındaki her organizmanın uygunluk değeri eşitlik (2.7)'te verilen amaç fonksiyonuna göre belirlenir. Eşitlik (2.5) ve (2.6)'daki her bir amaç ağırlıklı toplam metoduna göre tek bir amaç fonksiyonunda birleştirilmiştir.

$$J = (1 - w)\,(P_{avg})_{sc} - w\,(v_f)_{sc} \tag{2.7}$$

Bu eşitlikteki w, amaç fonksiyonundaki P_{avg} ve v_f'nin önceliğini belirler ve her iki amacın ağırlıkları toplamı 1 olmalıdır. $(P_{avg})_{sc}$ ve $(v_f)_{sc}$ robotun ortalama güç tüketimi ve ileri yöndeki hızının normalize edilmiş değerleridir ve eşitlik (2.8)'deki gibi hesaplanır. Bu eşitliklerdeki P_{avg} ve v_f'nin maksimum değerlerini bulmak için optimizasyon algoritması w=1 için çalıştırılmıştır.

$$(P_{avg})_{sc} = \frac{P_{avg}}{(P_{avg})_{\max}}, \quad (v_f)_{sc} = \frac{v_f}{(v_f)_{\max}} \tag{2.8}$$

Böylece J amaç fonksiyonunun verilen kısıtlara göre minimize edilmesine dayanan bu optimizasyon problemi matematiksel olarak eşitlik (2.9)'daki gibi ifade edilebilir. Bu problemin kısıtlarını, servo motordan kaynaklanan eklemlerin fiziksel sınırları ve buna bağlı olarak sinüzoidal hareket parametrelerinin alabileceği değer aralığı oluşturur. Eğer bu değerler herhangi bir organizma için sınırları aşarsa, bu organizmanın uygunluk değerine yüksek bir değer atanarak organizmanın ekosistemden atılması sağlanır.

$$\begin{aligned} &\min_{\alpha,\beta,\omega} \ \mathrm{J} = \left[P_{avg}, -v_f \right] \\ &\text{s.t} : \left| \phi_{\mathrm{i,ref}} \right| \le \phi_{\mathrm{i}}^{\max}, \left| \dot{\phi}_{\mathrm{i,ref}} \right| \le \dot{\phi}_{\mathrm{i}}^{\max}, \left| \mathrm{u_i} \right| \le \mathrm{u_i}^{\max} \\ &\quad 0 \le \alpha \le \alpha_{\max}, 0 \le \beta \le \beta_{\max}, 0 \le \omega \le \omega_{\max} \end{aligned} \tag{2.9}$$

2.3.1.1. Ağırlıklı Toplam Tabanlı MOSOS Algoritmasıyla Elde Edilen Sonuçlar

Yılan robotun enerji verimli hareket problemine ağırlıklı toplam tabanlı MOSOS algoritmasının uygulanmasıyla elde edilen sonuçlar bu bölümde sunulmuştur. Önerilen bu yöntem, parametreleri Tablo 2.1'de verilen 5 özdeş linkli tekerleksiz yılan robot için test edilmiştir.

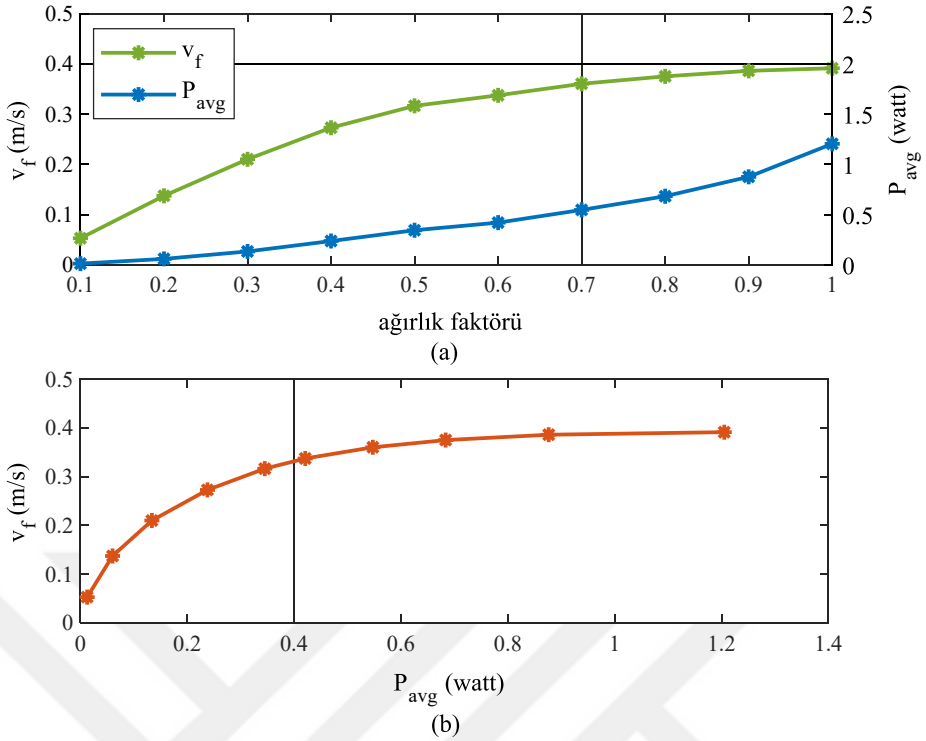
Algoritmanın başlangıç parametreleri Tablo 2.3'teki gibi ayarlanmıştır. Problemin boyutu, optimize edilen parametre sayısına eşit olması gerektiğinden organizmanın boyutu 3 olarak belirlenmiştir. Ekosistemdeki organizmaların sayısını belirlemede herhangi bir kural olmamakla birlikte bazı kaynaklarda (Parsopoulos ve Vrahatis, 2002) karar vektör boyutunun 5 katı olması önerildiği için n_o = 15 olarak ayarlanmıştır. Maksimum iterasyon sayısı da çözüme etkisi göz önünde bulundurularak N_{max} = 20 olacak şekilde deneysel olarak seçilmiştir. Eklemlerin fiziksel kısıtları HSR-5990TG servo motoruna göre belirlenirken, arama uzayının alt ve üst sınırları sinüzoidal hareket parametrelerinin alabileceği minimum ve maksimum değerlere göre belirlenmiştir.

Tablo 2.3. Optimizasyon parametreleri

Parametreler	**Değerleri**
Organizma boyutu	n_d= 3
Organizma sayısı	n_o = 15
Maksimum iterasyon sayısı	N_{max} = 20
Yürüyüş parametrelerinin üst sınırları	$\alpha_{max} = 90^\circ$, $\beta_{max} = 90^\circ$ $\omega_{max} = 210\ ^\circ/s$
Eklemlerin fiziksel sınırları	$\phi_i^{max} = 90^\circ$, $\dot{\phi}_i^{max} = 429^\circ/s$, u_i^{max}=2.3 Nm

Öncelikle zemin sürtünme katsayıları c_t = 0.1; c_n = 10 iken amaç fonksiyonundaki w ağırlık faktörü 0.1 ile 1 arasında adım sayısı 0.1 olacak şekilde değiştirilerek robotun ortalama güç tüketimi ve ileri hareket hızı elde edilmiştir. Normalizasyon işleminde kullanılacak maksimum değerler algoritmanın daha önceden w = 1 için koşturulmasıyla v_f = 0.39 m/s ve P_{avg} = 1.20 W olarak elde edilmiştir. Ağırlık faktörünün robotun ortalama güç tüketimi ve ileri hızına etkisi Şekil 2.8 (a)'da verilmiştir. Bu şekle göre w değeri arttıkça robotun ileri hızının arttığı ve artan hız nedeniyle güç tüketiminin de arttığı görülmektedir. Elde edilen Pareto optimal çözümler ise Şekil 2(b)'de verilmiştir. Bu şekilden ağırlıklı toplam tabanlı MOSOS algoritmasının yakınsadığı Pareto cephenin iyi bir yayılama sahip olduğu fakat çözümlerin çok düzgün dağılmadığını görülmektedir. Bu bulgu, eşit olarak dağıtılan ağırlıklar ile her zaman düzgün dağılımlı Pareto optimal çözümlerin elde edilemeyeceğini kanıtlamaktadır.

Her bir ağırlık faktörü için elde edilen optimal yürüyüş parametrelerine göre elde edilen optimal sonuçlar Tablo 2.4'te verilmiştir. Bu sonuçlara göre, yılan robot 1.20 W ortalama güç tüketimiyle maksimum 0.39 m/s'lik bir hıza ulaşabilmektedir. Beklenildiği gibi robotun maksimum hızına w = 1 iken ulaşılmıştır. Ağırlık faktörü w =1'den w = 0.6'ya azaltıldığı zaman, yılan robotun ortalama güç tüketimi P_{avg} = 1.20 W'dan P_{avg} = 0.42 W'a düşmektedir. Buna rağmen hızı v_f = 0.39 m/s'den v_f = 0.34 m/s'ye düşmüştür. Böylece robotun hızındaki sadece %13.77'lik küçük bir azalma ile ortalama güç tüketiminde %65.08'lik önemli bir azalma elde edilebilir. Eğer robotun hızında daha az bir düşüş istenirse, w =0.8'deki optimum parametreler kullanılarak robotun hızındaki %4.11'lik bir azalmaya karşı güç tüketiminden %43.27'lik bir kazanç sağlanabilir. Bu tablo, kontrol hedefleri ve yılan robotun mevcut gücü gibi sistem gereksinimleri göz önünde bulundurularak yılan robot için en uygun çalışma stratejisini belirlemek amacıyla kullanılabilir.



Şekil 2.8. (a) Ağırlık faktörünün robotun ortalama güç tüketimi ve ileri hızına etkisi, (b) Pareto optimal çözümler

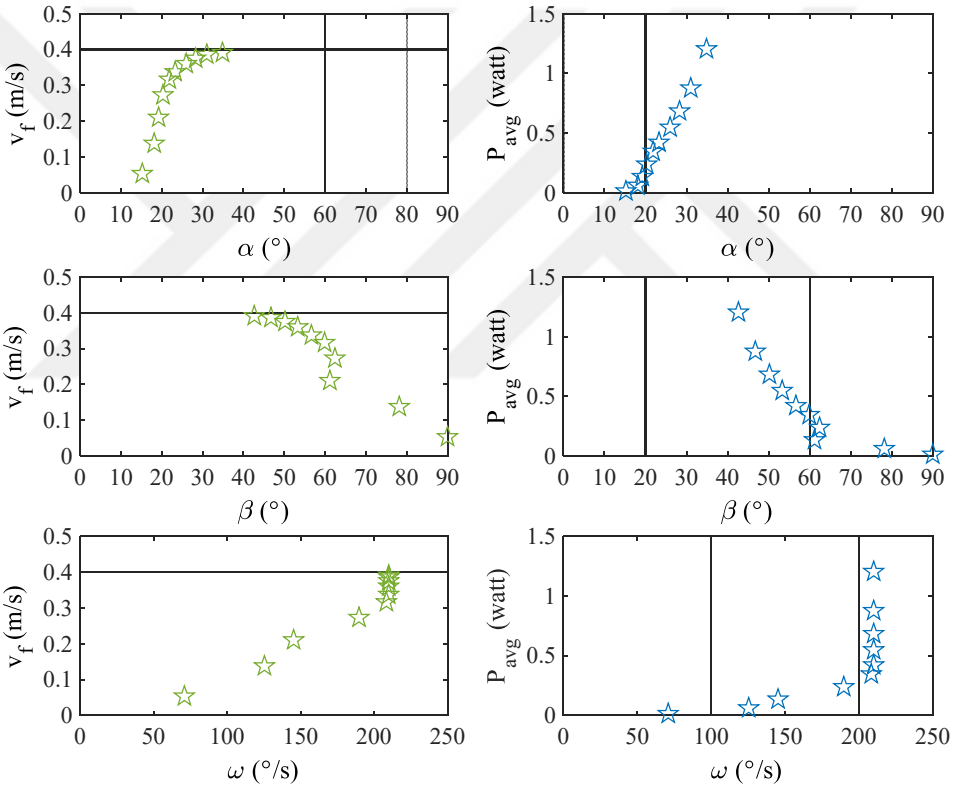
Tablo 2.4. Optimum yürüyüş parametrelerine göre elde edilen sonuçlar

w	α (°)	ω (° /s)	β (°)	v_f (m/s)	P_{avg} (W)
0.1000	15.2200	70.9344	90.0000	0.0531	0.0125
0.2000	18.1374	125.4008	78.1864	0.1375	0.0601
0.3000	19.2169	145.2789	61.1747	0.2102	0.1337
0.4000	20.2889	189.7583	62.4138	0.2730	0.2377
0.5000	21.8587	208.3734	59.8795	0.3166	0.3450
0.6000	23.2913	210.0000	56.6467	0.3375	0.4207
0.7000	25.9909	210.0000	53.3036	0.3605	0.5473
0.8000	28.2939	210.0000	50.1877	0.3753	0.6835
0.9000	31.0264	210.0000	46.7528	0.3863	0.8764
1.0000	34.8682	210.0000	42.6450	0.3914	1.2049

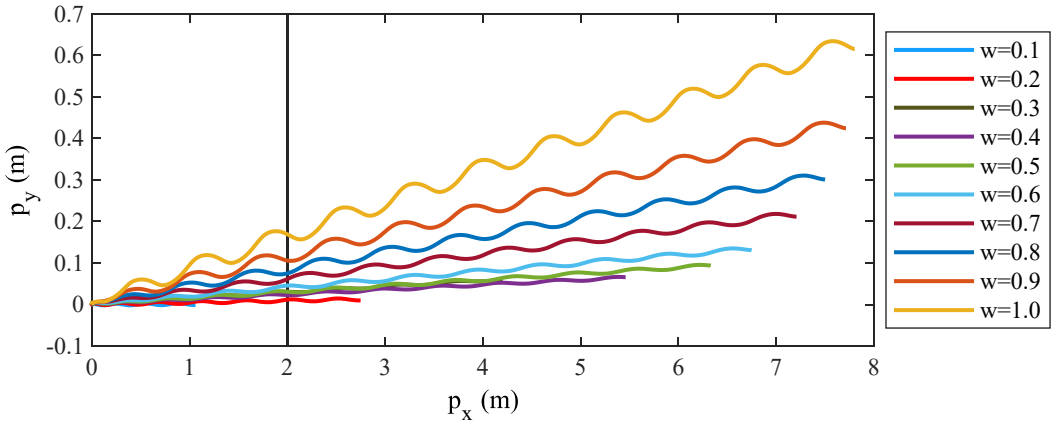
Yılan robotun hızının ve ortalama güç tüketiminin optimum yürüyüş parametrelerine göre değişimi Şekil 2.9'da gösterilmiştir. Bu şekle göre, α parametresinin artması robotun ileri hızında ve ortalama güç tüketiminde de artışa neden olmaktadır. Öte yandan, β parametresi α parametresinin tam aksine bir etki yapar. Yani β parametresi arttıkça robotun ileri hızında ve ortalama güç tüketiminde azalma olur. Şekil 2.9'dan elde edilen bir diğer önemli bulgu ise, α parametresinin optimum değerinin 15º ile 45º

aralığında iken, β parametresinin optimum değerinin 40º ile 65º aralığında olduğudur. ω parametresinin yılan robotunun ileri hızı üzerindeki etkisini incelediğimiz zaman ise çoğu ağırlık faktörü için maksimum değeri olan 210 º/s olarak bulunduğu görülmektedir.

Bu gözlemlere göre α ve β parametrelerinin alt ve üst sınırları daha dar bir aralıkta belirlenebilir ve ω'yı maksimum değerine ayarlayarak arama uzayının boyutu 3'ten 2'ye azaltılabilir. Böylece, yılan robotun optimum yürüyüş parametrelerini bulmak için uygulanan optimizasyon süreci daha verimli ve daha az maliyetli bir hesaplama haline gelebilir. Son olarak, her bir ağırlık faktörü için yılan robotun ağırlık merkezinin pozisyonu Şekil 2.10'da verilmiştir.



Şekil 2.9. Yılan robotun hızının ve ortalama güç tüketiminin optimum yürüyüş parametrelerine göre değişimi



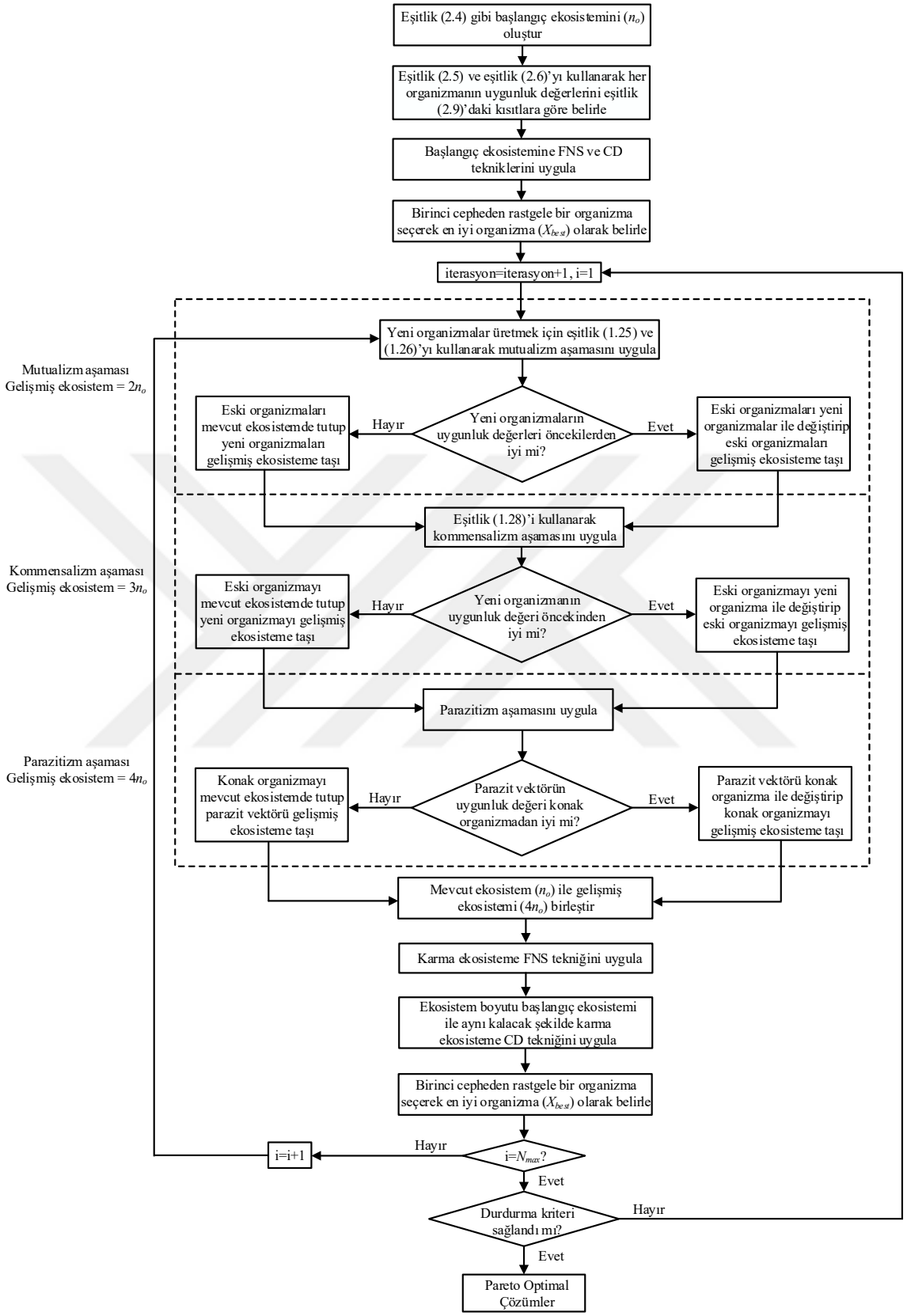
Şekil 2.10. Her bir ağırlık faktörüne göre robotun ağırlık merkezinin pozisyonu

2.3.2. FNSMOSOS Algoritmasının Yılan Robotun Enerji Verimli Hareket Problemine Uygulanması

FNSMOSOS algoritması, ağırlıklı toplam tabanlı MOSOS algoritmasında olduğu gibi aynı şekilde üretilmiş başlangıç ekosistemi ile başlar. Yılan robotun ortalama güç tüketimi P_{avg} ve ileri yöndeki hızı v_f için eşitlik (2.5) ve (2.6)'da verilen iki ayrı amaç fonksiyonu her organizma için değerlendirilir ve eşitlik (2.9)'da tanımlanan sınırları aşan organizmanın uygunluk değerine yüksek bir değer atanır. Yılan robotun optimal hareket problemini çözmek için önerilen FNSMOSOS algoritmasının akış diyagramı Şekil 2.11'de verilmiştir.

2.3.2.1. FNSMOSOS Algoritmasıyla Elde Edilen Sonuçlar

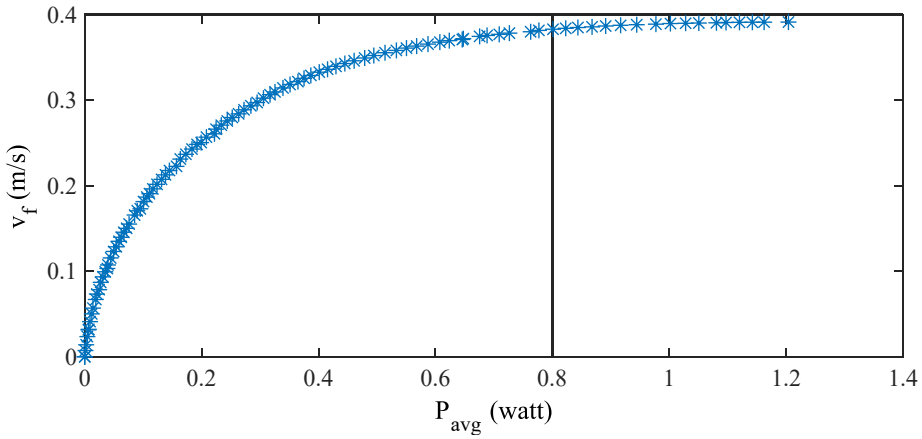
Yılan robotun enerji verimli hareket problemine FNSMOSOS algoritmasının uygulanmasıyla elde edilen sonuçlar bu bölümde sunulmuştur. Önerilen bu yöntem, parametreleri Tablo 2.1'de verilen 5 özdeş linkli tekerleksiz bir yılan robot için zemin sürtünme katsayıları $c_t = 0.1$ ve $c_n = 10$ olan bir ortamda test edilmiştir. Bu algoritmada, ekosistemdeki organizma sayısı 100, maksimum iterasyon sayısı 20 olarak seçilmiştir. Eklemlerin fiziksel kısıtları ve arama uzayının alt ve üst sınırları ise Tablo 2.3'teki gibi ayarlanmıştır. Buna göre elde edilen Pareto optimal çözüm kümesi Şekil 2.12'de verilmiştir. Bu şekil, FNSMOSOS algoritması ile yakınsanan Pareto cephesinin iyi bir çözüm çeşitliliğine sahip olduğunu göstermektedir.



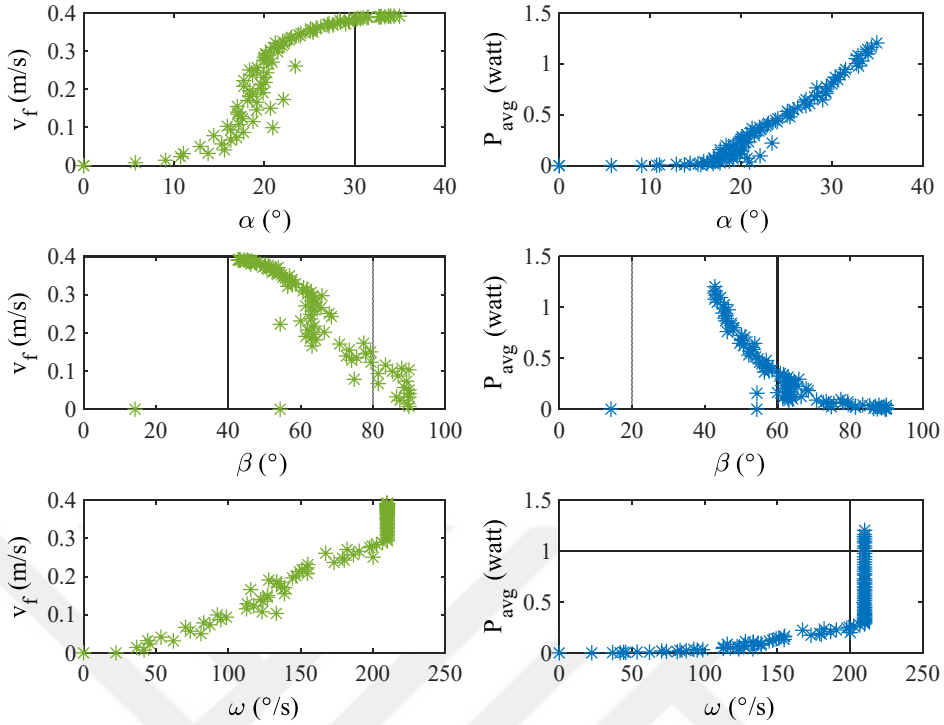
Şekil 2.11. Yılan robotun optimal hareket problemi için önerilen FNSMOSOS algoritmasının akış diyagramı

Bu algoritmayla da ağırlıklı toplam tabanlı MOSOS algoritmasında olduğu gibi yılan robotun maksimum hızı v_f =0.39 m/s olarak elde edilmiştir. Maksimum hıza karşılık gelen maksimum ortalama güç tüketimi P_{avg} = 1.20 W'tır. 100 tane Pareto optimal yürüyüş parametresinden 30 tanesi için elde edilen sonuçlar Tablo 2.5'te verilmiştir. Tablodan da görüldüğü gibi, FNSMOSOS algoritması daha çok farklı çözüm sunmaktadır. Böylece karar verici, sistem gereksinimlerine göre daha fazla seçenek arasından en uygun yürüyüş parametrelerini seçerek yılan robotun ileri hızı ile ortalama güç tüketimi arasında denge kurabilir. Örneğin; çözüm 30'a kıyasla çözüm 22, robotun hızındaki %6.56'lık küçük bir azalmaya karşın ortalama güç tüketimindeki %51.23'lük bir kazanç ile daha iyi bir seçenek olabilir. Sistemin mevcut gücü daha sınırlı ise çözüm 17'deki yürüyüş parametreleri seçilerek ortalama güç tüketimi %74.60 azaltılabilir. Bu durumda ileri hız sadece %22.74 azalır.

Pareto optimal yürüyüş parametrelerinin ileri hız ve ortalama güç tüketimi üzerindeki etkisi Şekil 2.13'te sunulmuştur. Bu eğriler, ağırlıklı toplam tabanlı MOSOS algoritması ile elde edilen Şekil 2.9'daki eğrilere benzerdir. Bu bulgu, MOSOS tabanlı iki farklı algoritmanın, yılan robotun optimal hareketi için kararlı sonuçlar ürettiğini göstermektedir.



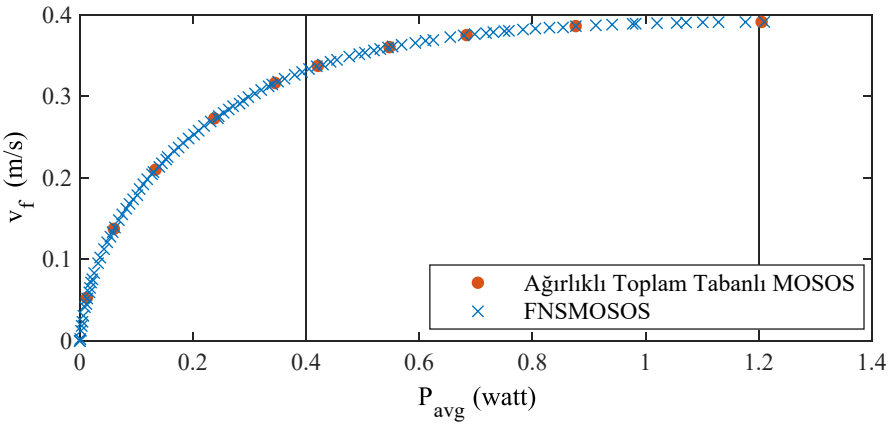
Şekil 2.12. FNSMOSOS algoritması ile elde edilen Pareto optimal çözümler



Şekil 2.13. Yılan robotun hızının ve ortalama güç tüketiminin FNSMOSOS ile elde edilen optimum yürüyüş parametrelerine göre değişimi

2.3.3. Önerilen Yöntemlerin Karşılaştırılması

Şekil 2.14'te, yılan robotun optimal hareket problemi için önerilen ağırlıklı toplam tabanlı MOSOS ve FNSMOSOS algoritmaları ile elde edilen Pareto optimal eğriler karşılaştırmalı olarak verilmiştir. İki eğrinin benzer bir yayılıma sahip olması algoritmaların kararlılıklarını gösterir. Fakat FNSMOSOS algoritması ile tek bir çalışmada daha çok çözüm noktası üretilerek Pareto cephesinde daha düzgün bir dağılımının elde edildiği görülmektedir.



Şekil 2.14. Önerilen yöntemler ile elde edilen Pareto optimal eğrilerin karşılaştırılması

Tablo 2.5. Pareto Optimal yürüyüş parametrelerine göre elde edilen sonuçlar

Çözüm	***α*** **(˚)**	***ω*** **(˚ /s)**	***β*** **(˚)**	v_f **(m/s)**	P_{avg} **(W)**
1	5.7348	41.8266	90.0000	0.0073	0.0012
2	13.7767	44.2929	86.5326	0.0321	0.0068
3	16.2800	70.4723	81.4882	0.0670	0.0189
4	17.5858	117.6275	79.4923	0.1231	0.0494
5	17.0468	124.0257	75.5258	0.1350	0.0585
6	17.0036	138.8863	73.4650	0.1552	0.0758
7	18.1607	138.9609	70.7857	0.1707	0.0898
8	19.6000	146.8214	66.5524	0.2016	0.1236
9	19.5987	149.1822	63.4268	0.2127	0.1381
10	19.8998	155.0219	60.0336	0.2312	0.1634
11	20.2750	180.2280	68.5126	0.2429	0.1834
12	19.1930	186.5953	63.2537	0.2568	0.2082
13	20.4790	190.5852	65.6480	0.2661	0.2246
14	19.9386	197.8035	63.8226	0.2757	0.2438
15	20.1859	206.7052	63.8109	0.2888	0.2725
16	20.1264	210.0000	63.3527	0.2934	0.2842
17	20.6437	210.0000	61.9566	0.3023	0.3058
18	21.9738	210.0000	61.6950	0.3143	0.3391
19	22.7583	210.0000	58.5487	0.3289	0.3868
20	23.7820	210.0000	55.7554	0.3428	0.4449
21	25.3091	210.0000	54.0794	0.3555	0.5137
22	27.1476	210.0000	53.3289	0.3656	0.5872
23	28.9861	210.0000	53.7887	0.3709	0.6459
24	29.3437	210.0000	50.1804	0.3784	0.7265
25	30.1980	210.0000	47.3687	0.3840	0.8229
26	31.3692	210.0000	45.7991	0.3876	0.9169
27	33.3200	210.0000	45.4232	0.3900	1.0281
28	33.9895	210.0000	44.3541	0.3909	1.0971
29	33.9215	210.0000	42.8199	0.3912	1.1423
30	34.9466	210.0000	42.8074	0.3913	1.2040

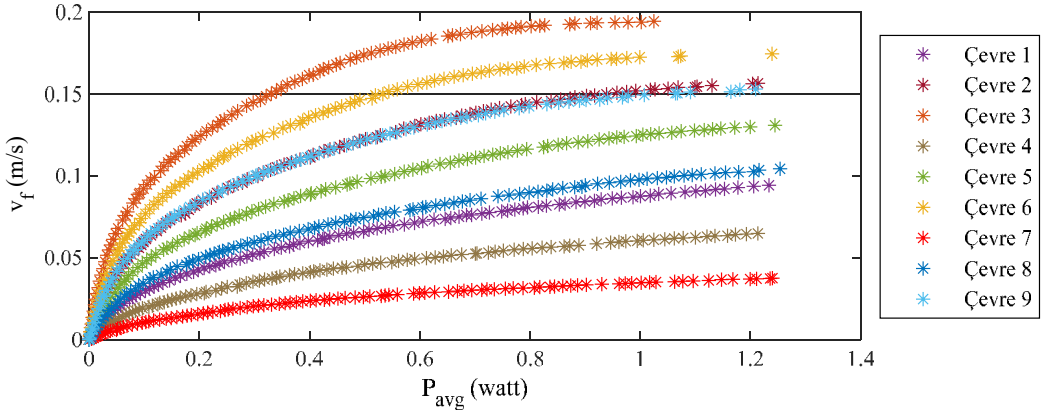
2.3.4. Yılan Robotun Farklı Çevre Koşullarındaki Enerji Verimli Adaptif Hareketi ve Elde Edilen Sonuçlar

Buraya kadar sabit sürtünme katsayılarına sahip bir ortamda yılan robotun enerji verimli hareketi için optimum yürüyüş parametreleri bulundu. Bu bölümde ise yılan robotunun farklı çevre koşullarındaki enerji verimli adaptif hareketi dokuz farklı ortamda FNSMOSOS algoritması kullanılarak incelenmiştir. Bu ortamların sürtünme katsayıları, robotun hareket ettiği yüzeyin camdan ahşaba kadar oldukça geniş bir aralıkta değiştiği düşünülerek Tablo 2.6'daki gibi belirlenmiştir.

Tablo 2.6. Adaptif hareket için kullanılan farklı çevreler

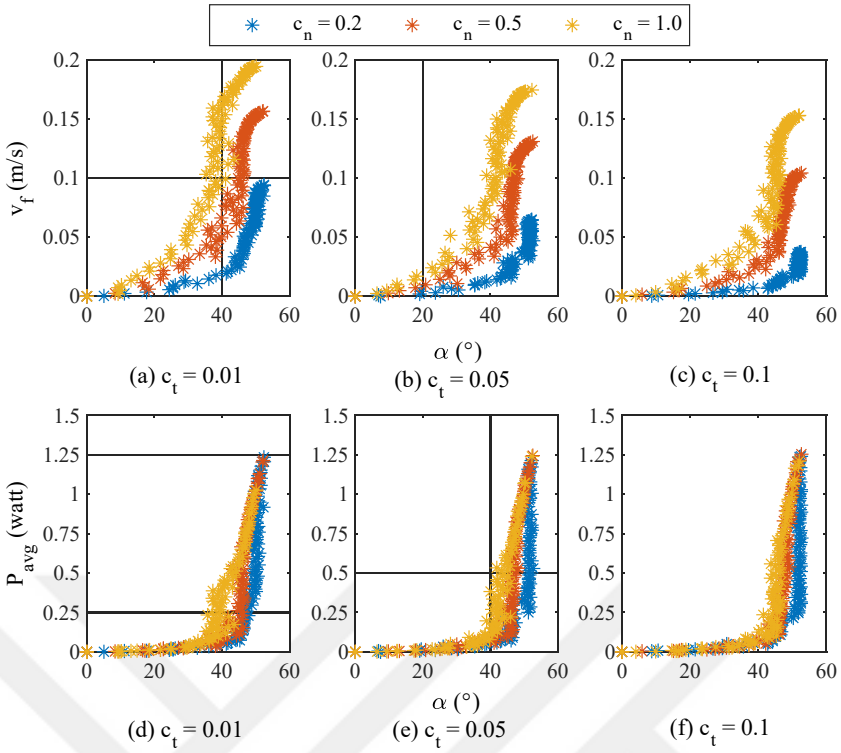
Çevre	c_t	c_n
1	0.01	0.2
2	0.01	0.5
3	0.01	1.0
4	0.05	0.2
5	0.05	0.5
6	0.05	1.0
7	0.1	0.2
8	0.1	0.5
9	0.1	1.0

Algoritma parametreleri ve optimizasyon süreci, çevre sabit iken yapılan analiz ile aynı şekilde uygulanmıştır. Buna göre her çevre için elde edilen Pareto optimal çözümler Şekil 2.15'te verilmiştir. Bu şekilden, FNSMOSOS algoritmasının bütün çevreler için Pareto cephesine yakınsayan optimal çözümlere ulaşmada genel olarak iyi bir performans sunduğu görülmektedir. Beklenildiği gibi yılan robot, c_n'nin yüksek olduğu çevre 3, çevre 6 ve çevre 9 gibi ortamlarda daha yüksek hızlara ulaşmıştır. Ayrıca, c_n sabit iken çevre 3 gibi sürtünme katsayıları arasındaki oranın c_n/c_t daha büyük olduğu ortamlarda yılan robotun daha hızlı hareket edebildiği görülmektedir.

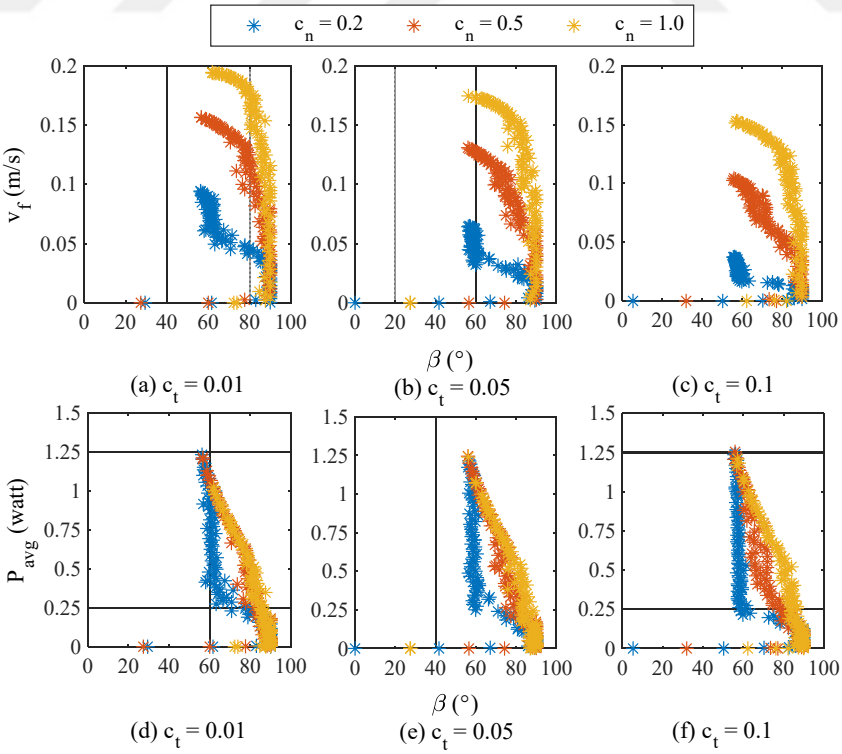


Şekil 2.15. Her çevre için FNSMOSOS algoritması ile elde edilen Pareto optimal çözümler

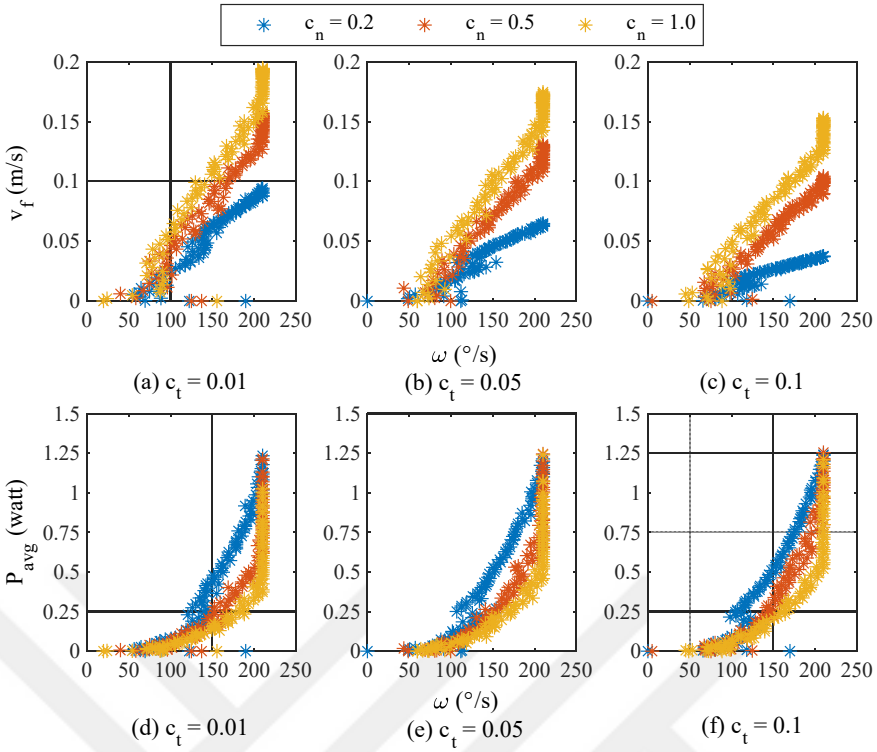
Yılan robota normal ve teğet yönde etki eden farklı sürtünme katsayılarının, optimal yürüme parametrelerine göre robotun hızı ve ortalama güç tüketimi üzerindeki etkisi Şekil 2.16-Şekil 2.21'de sunulmuştur. Bu şekillerden, α parametresinin optimal değerinin dokuz farklı çevre için 20º ile 50º arasında değişirken, β parametresinin optimal değerinin 55º ile 90º arasında değiştiği görülmektedir. ω parametresinin optimal değeri ise 50º ile 210º arasında daha geniş bir aralıkta değişmiştir. Buna rağmen, yılan robotun bütün çevrelerde maksimum hızına ulaşması için ω parametresinin, maksimum değeri olan 210º/s değerinde olması gerektiği görülmektedir. Bu şekiller, yılan robotun farklı zemin koşullarında enerji verimli hareketini sürdürebilmesi için ortama göre yürüyüş parametrelerini değiştirmesi gerektiğini kanıtlamaktadır. c_t sabitken c_n'nin değiştiği ortamlardaki robotun hızı ve ortalama güç tüketimi, sırasıyla optimal α, β ve ω parametrelerine göre Şekil 2.16, Şekil 2.17 ve Şekil 2.18'de verilmiştir. Yılan robotun c_n'nin küçük olduğu kaygan ortamlardaki hareketi daha fazla sürtünme kuvveti gerektirdiğinden Şekil 2.16 ve Şekil 2.18'de görüldüğü gibi c_n azaldıkça optimal α ve ω parametreleri artmaktadır. Bu bulgunun önemli bir sonucu, yılan robotun bu tür ortamlarda ileri yönde daha fazla sürtünme kuvveti kazanmak için daha büyük genlik ve frekansla hareket etmesi gerektiğidir. Öte yandan Şekil 2.17, aynı ortam koşullarında optimal β parametresinin azaldığını göstermektedir. Benzer şekilde, Şekil 2.19, Şekil 2.20 ve Şekil 2.21, sırasıyla optimal α, β ve ω parametrelerine göre c_n sabitken c_t'nin değiştiği ortamlarda robotun hızını ve ortalama güç tüketimini göstermektedir. c_t'nin azalması, optimal yürüme parametrelerini c_n'nin tersi yönünde etkiler. Yani, c_t azaldıkça optimal α ve ω parametreleri azalırken optimal β parametresi artar. Ancak, c_t'nin optimal yürüme parametreleri üzerindeki etkisinin c_n'ye göre daha az olduğu görülmektedir.



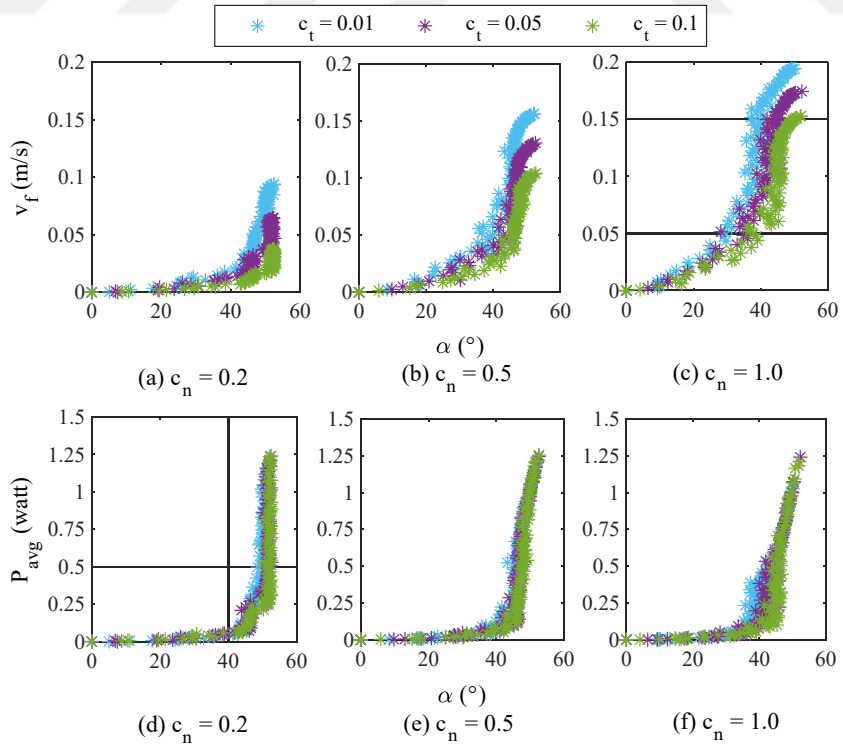
Şekil 2.16. Yılan robota normal yönde etki eden farklı sürtünme katsayılarının optimal α parametresine göre robotun hızı ve ortalama güç tüketimi üzerindeki etkisi



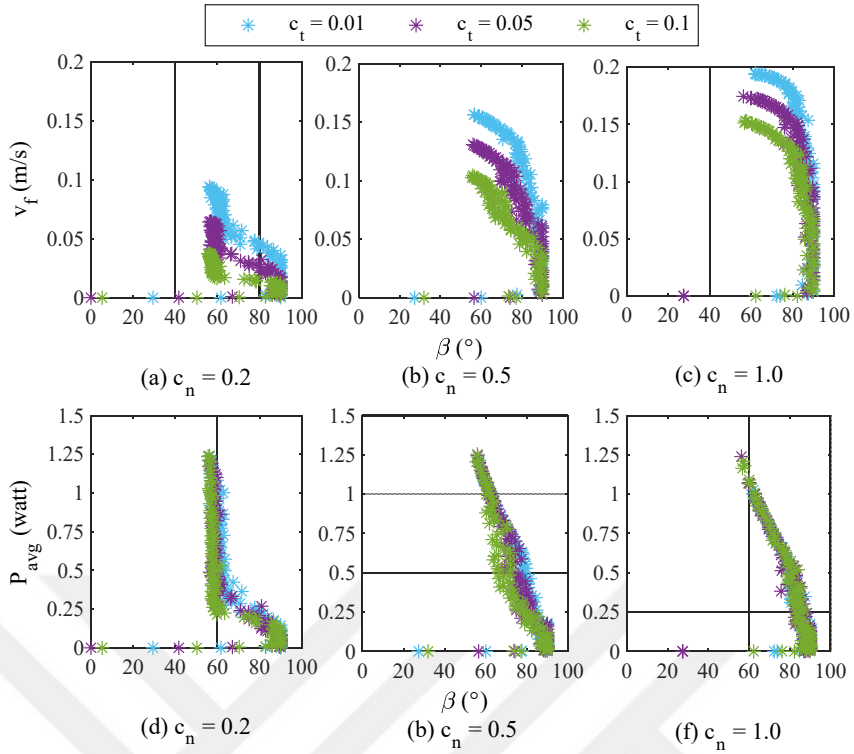
Şekil 2.17. Yılan robota normal yönde etki eden farklı sürtünme katsayılarının optimal β parametresine göre robotun hızı ve ortalama güç tüketimi üzerindeki etkisi



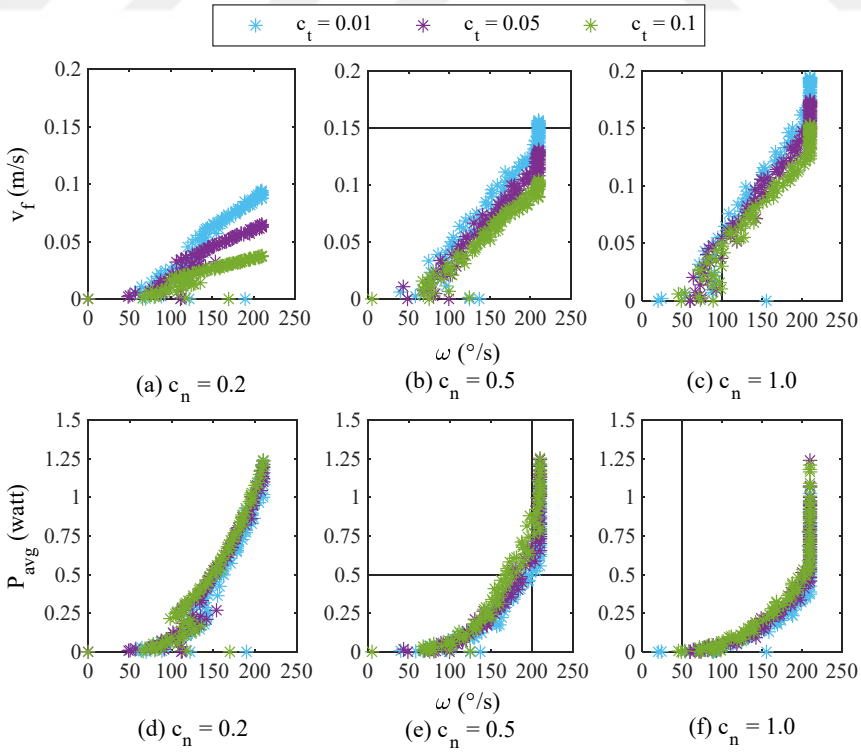
Şekil 2.18. Yılan robota normal yönde etki eden farklı sürtünme katsayılarının optimal ω parametresine göre robotun hızı ve ortalama güç tüketimi üzerindeki etkisi



Şekil 2.19. Yılan robota teğet yönde etki eden farklı sürtünme katsayılarının optimal α parametresine göre robotun hızı ve ortalama güç tüketimi üzerindeki etkisi



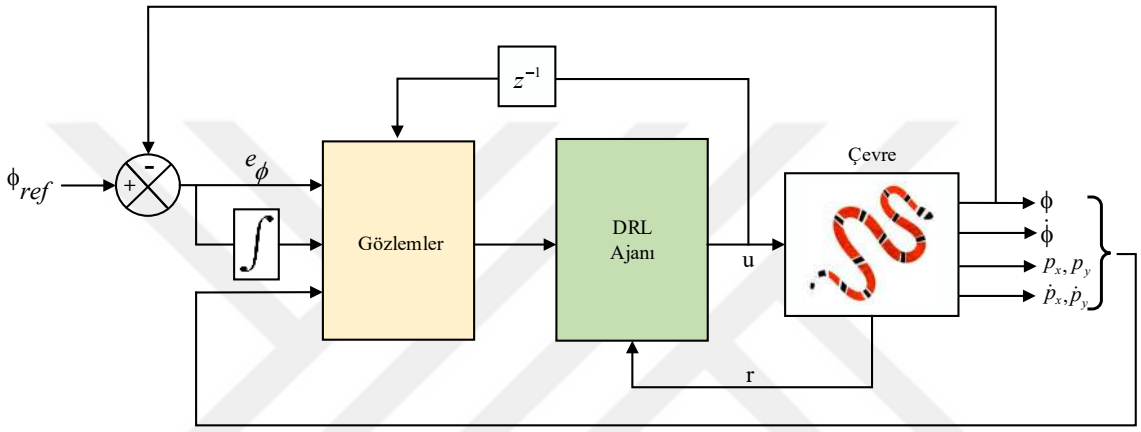
Şekil 2.20. Yılan robota teğet yönde etki eden farklı sürtünme katsayılarının optimal β parametresine göre robotun hızı ve ortalama güç tüketimi üzerindeki etkisi



Şekil 2.21. Yılan robota teğet yönde etki eden farklı sürtünme katsayılarının optimal ω parametresine göre robotun hızı ve ortalama güç tüketimi üzerindeki etkisi

2.4. Eklem Yörünge Kontrolü

Bu bölümde tekerleksiz yılan robotun yanal dalgalanma hareketi için eklem yörünge kontrolü DRL tabanlı denetleyiciler kullanılarak ele alınmıştır. Bunun için DDPG, TD3 ve Tranfer-TD3 algoritmaları kullanılmıştır. Parametreleri Tablo 2.1'de verilen 5 özdeş linkli tekerleksiz yılan robotun modeli çevre olarak kullanılmıştır. Şekil 2.22'de sistemin genel blok diyagramı görülmektedir.



Şekil 2.22. DRL ile yılan robot eklem yörünge kontrolü

Yılan robotun her bir eklemine eşitlik (2.2)'de verilen filtrelenmiş sinüzoidal referans işareti uygulanmaktadır. Filtre parametreleri $\omega_n = 25$ ve $\xi=1$ olarak belirlenmiştir. Eğitilen ajanların genelleme yeteneklerinin arttırılması için referans işareti tanımlayan parametreler $[\alpha, \beta, \omega]$'den oluşan farklı referans dizileri tanımlanmış ve eğitim boyunca her bölümde bu dizilerden biri rastgele seçilerek ajanın eğitilmesi gerçekleştirilmiştir. Bu amaçla eğitimde 10 tane farklı referans işareti kullanılmıştır.

2.4.1. Gözlem Uzayı

Ajan, her bölümün her adımında gözlemler aracılığıyla çevreden bilgiler toplar. Ajanın eylemlerine göre alacağı ödüller arasında ilişki kurması için doğru bir veri setine ihtiyacı vardır. Bu nedenle gözlem uzayını oluşturan büyüklüklerin uygun seçimi RL'de oldukça önemlidir.

Bu çalışmada yılan robotun ajana sağladığı 12 tane gözlem değeri bulunmaktadır. Bu gözlemler robotun hareketi öğrenmesi için gereklidir. Robotun sağladığı bu gözlemlere ek olarak referans yörüngeyi takip hatası, bu hatanın integrali ve bir önceki adımdaki aksiyonlar ajana gözlem olarak verilmiştir. Toplamda 24 tane gözlem kullanılmıştır. Gözlem uzayını oluşturan büyüklükler Tablo 2.7'de verilmiştir.

Tablo 2.7. DRL denetleyicinin gözlem uzayı

Sembol	Açıklama	Boyut
ϕ	Robotun eklem pozisyonları	4
$\dot{\phi}$	Robotun eklem hızları	4
p_x, p_y	Robotun pozisyonu	2
$\dot{p}_x$, $\dot{p}_y$	Robotun hızı	2
e_ϕ	Referans yörünge takip hataları	4
$\int e_\phi$	Referans yörünge takip hatalarının integralleri	4
u_{t-1}	Bir önceki adımdaki tork girişi	4

2.4.2. Aksiyon Uzayı

DRL ajanı, 24 gözlemi yılan robotun 4 tane 1 DOF eklemine uygulanacak tork aksiyon değerlerine haritalamaktadır. Tork aksiyon sinyali [-2.3,2.3] Nm arasında bir değer alır.

2.4.3. Ödül Fonksiyonu

Robotun öğrenmesi için her adımda geri döndürülen ödül fonksiyonu eşitlik (2.10)'da verilmiştir.

$$r_t = -0.1\sum_i (e_\phi)_t^{i\,2} - 0.02\sum_i u_{t-1}^{i\,2} + 2H_t - 10F_t \tag{2.10}$$

Bu eşitlikteki;

- $(e_\phi)_t^i$ *i.* ekleme ait yörünge hatası olup, referans yörüngeden sapma durumunda ödül fonksiyonunda ceza oluşturmak için kullanılır.
- u_{t-1}^iönceki zaman aralığından *i.* ekleme uygulanan tork değeridir. Aşırı kontrol çabası kullanımı durumunda ödül fonksiyonunda ceza oluşturmak için kullanılır.

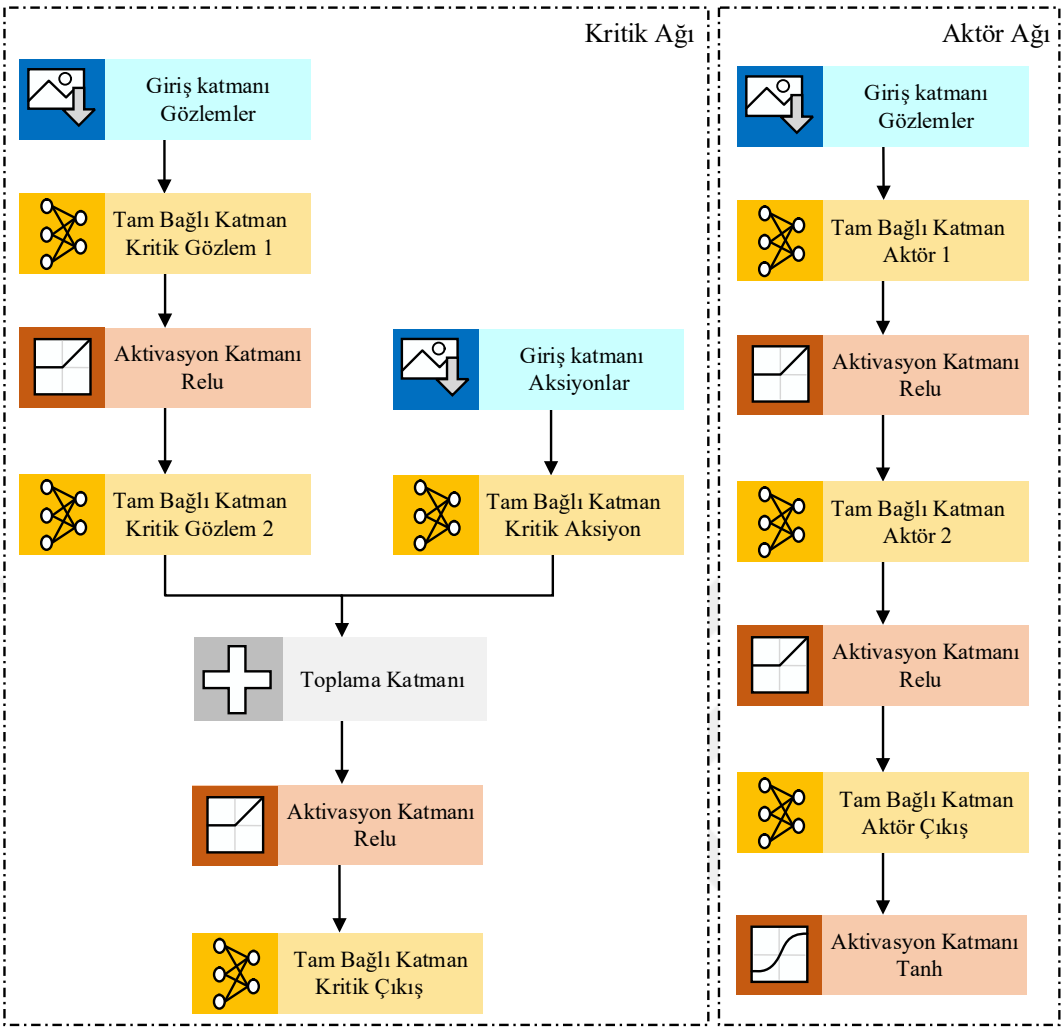
- H_t referans yörünge hatasının azaltılmasını teşvik etmek için ödül fonksiyonunda kullanılan lojik bir terimdir. $e_\phi^2 < 0.001$ ise $H_t = 1$, değilse de $H_t = 0$ değerini alır.
- F_t simülasyonun erken durması durumunda ajana ceza oluşturmak için ödül fonksiyonunda kullanılan lojik bir değerdir. Eğer simülasyon maksimum adım sayısına ulaşmadan durursa $F_t = 1$, aksi durumda ise $F_t = 0$ değerini alır. Simülasyonu durdurma koşulu $e_\phi > 5$ olarak belirlenmiştir.

2.4.4. Ağ Mimarisi

Bu çalışmada kullanılan aktör ve kritik ağlarının mimarisi Şekil 2.23'te verilmiştir. Aktör ağı toplamda 7 katmandan oluşurken, kritik ağı ise gözlem ve aksiyon için iki farklı girişten oluşan toplamda 9 katmanlı bir ağdır. Her iki mimaride de sırasıyla 400 ve 300 nörondan oluşan tam bağlı iki gizli katman kullanılmıştır. Aktör ağının çıkışı robota uygulanacak tork değerlerini verirken, kritik ağının çıkışı ise robottan alınan ödül ile karşılaştırılarak elde edilen hata değeriyle hem aktör hem de kritik ağlarının güncellemesinde kullanılır. Her iki ağda da tam bağlı gizli katmanlardan sonra ReLU aktivasyon fonksiyonu kullanılmış olup, aktör ağının çıkışı için ise tanh aktivasyon fonksiyonu kullanılmıştır. Aktör ve kritik ağlarındaki katmanların giriş, çıkış ve ağ parametrelerine ait vektör boyutları Tablo 2.8'de özetlenmiştir.

Tablo 2.8. Aktör ve kritik ağ katmanlarına ait boyutlar

Aktör ağı			Kritik ağı		
Katmanlar	Boyutlar		Katmanlar	Boyutlar	
	Ağırlıklar	Bias		Ağırlıklar	Bias
Gözlem	24x1		Gözlem	24x1	
Aktör 1	400x24	400x1	Kritik gözlem 1	400x24	400x1
Aktör 2	300x400	300x1	Kritik gözlem 2	300x400	300x1
Aktör çıkış	4x300	4x1	Aksiyon	4x1	
			Kritik aksiyon	300x4	300x1
			Kritik çıkış	1x300	1x1



Şekil 2.23. Aktör ve kritik ağ mimarileri

2.4.5. Eğitim Parametreleri

Bu tez çalışmasında yapılan bütün eğitimler i7-6700 CPU @ 3.40 GHz, 8 GB RAM Nvidia GTX 1080 GPU ekran kartına sahip bir bilgisayarda gerçekleştirilmiştir. Ajanın eğitilmesi için gerekli parametreler aşağıda açıklanmış ve yılan robotun eklem yörünge kontrolü için bu çalışmada kullanılan değerler Tablo 2.9'da verilmiştir.

Maksimum bölüm sayısı: Ajanı eğitmek için kullanılan bölümlerin maksimum sayısını ifade eder. Eğitim durdurma kriteri sağlanmadıkça, eğitim maksimum bölüm sayısına ulaşınca durur.

Maksimum adım sayısı: Simülasyon ve örnekleme zamanlarına bağlı olarak belirlenir. Simülasyon durdurma kriteri sağlanmadıkça, simülasyon maksimum adım sayısına ulaşana kadar devam eder.

Ortalama pencere uzunluğu: Eğitim için kullanılan parametrelerin ortalama cinsinden olması durumunda ortalama hesaplanırken kaç bölümün kullanılacağını belirleyen parametredir.

Eğitim durdurma kriteri: Ortalama adım sayısı, ortalama ödül, bölüm ödülü, toplam adım sayısı, bölüm sayısı gibi parametreler eğitim durdurma kriteri olarak kullanılabilir.

Eğitim durdurma değeri: Belirlenen eğitim durdurma kriterinin bu değere eşit ya da büyük olması durumunda eğitim sonlanır.

Ajan kaydetme kriteri: Ortalama adım sayısı, ortalama ödül, bölüm ödülü, toplam adım sayısı, bölüm sayısı gibi parametreler ajan kaydetme kriteri olarak kullanılabilir. Hiçbir kriter belirlenmezse sadece en son bölümdeki ajan kaydedilir.

Ajan kaydetme değeri: Belirlenen ajan kaydetme kriterinin bu değere eşit ya da büyük olması durumunda ajan belirlenen klasöre kaydedilir. Bu değer eğer küçük seçilirse çok fazla ajanın kaydedilerek hem belleğin dolmasına hem de kaydedilen bu ajanlardan gerçekten iyi olan ajanın belirlenememesine sebep olabilir. Büyük seçildiği zaman ise de eğitim o değere ulaşmadan durabilir. Böylece hiçbir ajan kaydedilemediği için eğitim süreci boşa gidebilir. Bu nedenle ajan kaydetme değeri kritik bir öneme sahiptir.

Tablo 2.9. Eğitime ait parametreler

Parametreler	Değerleri
Örnekleme zamanı	T_s= 0.002 s
Simülasyon zamanı	T_f= 10 s
Maksimum adım sayısı	T_f/T_s = 5000
Maksimum bölüm sayısı	20000
Ortalama pencere uzunluğu	250
Eğitim durdurma kriteri	Maksimum bölüm sayısı
Eğitim durdurma değeri	20000
Ajan kaydetme kriteri	Bölüm ödülü
Ajan kaydetme değeri	39500

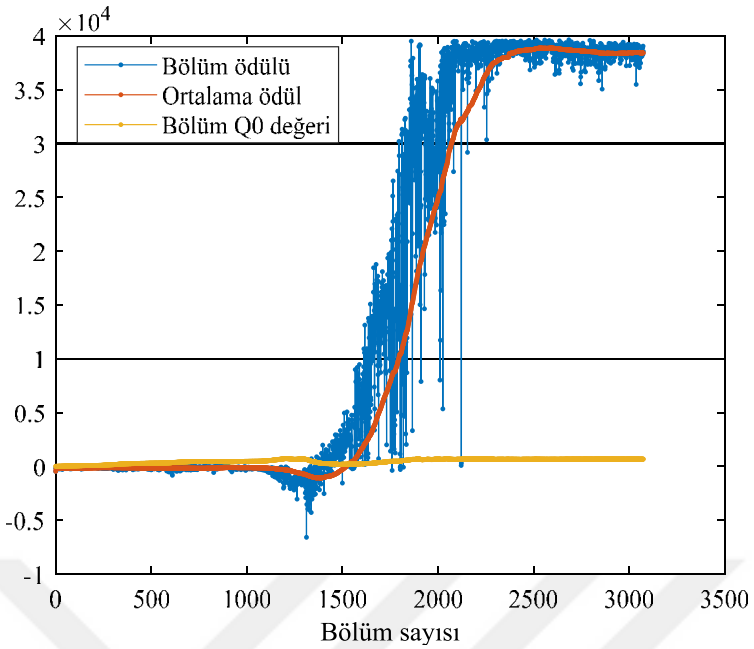
2.4.6. DDPG ile Elde Edilen Sonuçlar

Tablo 2.9'daki eğitim parametreleri ve Tablo 2.10'da verilen ajan parametreleri kullanılarak yılan robotun eklem yörünge kontrolü için DDPG ajanı eğitilmiş ve eğitim eğrisi Şekil 2.24'te verilmiştir. Bu şekilde her bölüm için bölüm ödülü, ortalama ödül ve bölüm Q0 değeri görülmektedir. Bölüm ödülü bir bölümdeki her adımdan alınan ödüllerin

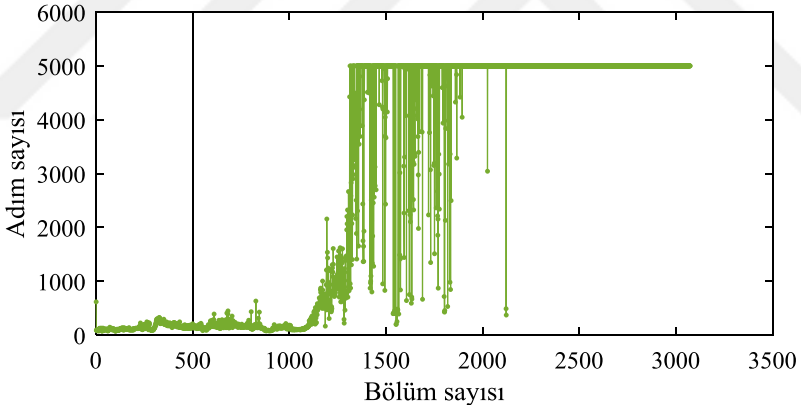
toplamını ifade eder. Şekil 2.24'te görüldüğü gibi bölüm ödülü yaklaşık ilk 1500 bölüm boyunca düşük değerler almıştır. Bunun sebebi eğitimin başlarında e_{ϕ}>5 simülasyon durdurma kriteri sağlandığı için ilk bölümlerin oldukça kısa sürmüş olmasıdır. Her bölüm için adım sayısının değişimini gösteren Şekil 2.5'te görüldüğü gibi adım sayısının maksimum adım sayısı olan 5000'e ulaşmasıyla birlikte bölüm ödülünün giderek arttığı görülmektedir. Fakat bölüm ödülündeki bu artış düzenli bir şekilde olmayıp bölümler arasındaki farkın büyük olduğu görülmektedir. Ortalama ödül, ortalama pencere uzunluğu kadar bölümün bölüm ödülünün ortalaması alınarak elde edilir ve bu nedenle değişim eğrisi bölüm ödülüne göre daha kararlıdır. Bölüm Q0 değeri ise her bölümün başındaki başlangıç koşullarına göre o bölümün alabileceği beklenilen ödülün tahmini değerini ifade eder. Eşitlik 2.10'daki ödül fonksiyonu göz önüne alındığı zaman ajanın her bir adım için alabileceği maksimum ödül değeri yaklaşık olarak 8'dir. Eğitimin istenilen koşullara sahip olması için simülasyonun maksimum adım sayısına ulaşması gerektiği düşünüldüğünde bir bölümden alınabilecek maksimum ödül 8x5000=40000 olmaktadır. Eğitim sonucunda elde edilen maksimum bölüm ödülü ise 39631'dir. Eğitim, durdurma kriteri olan maksimum bölüm sayısı 20000 değerine ulaşmadan ortalama ödülün yaklaşık 38400 değerine yakınsamasıyla birlikte aşırı öğrenmeyi önlemek amacıyla 3088. bölümde manuel olarak durdurulmuştur. Bu eğitim süreci boyunca toplam 8750257 veri kullanılmış ve eğitim yaklaşık 140 saat sürmüştür. Ajan kaydetme kriterini sağlayan ajanlar test edilerek en iyi ajanın denetleyici performansı Bölüm 2.4.10'da diğer algoritmalardan elde edilen ajanların sonuçlarıyla karşılaştırmalı olarak verilmiştir.

Tablo 2.10. DDPG ajanına ait parametreler

Parametreler	Değerleri
Aktör için öğrenme hızı	α_a =1e^{-4}
Kritik için öğrenme hızı	α_c =1e^{-3}
Azaltma faktörü	γ = 0.99
Yumuşak hedef güncelleme katsayısı	τ= 1e^{-3}
Hedef güncelleme frekansı	1
Bellek boyutu	1e^{6}
Mini-paket boyutu	512
Aksiyon gürültü modeli	OU
Gürültü standart sapması	0.1
Minimum standart sapma	0
Standart sapma bozulma oranı	0
Ortalamaya çekim sabiti	1



Şekil 2.24. DDPG ajanına ait eğitim eğrisi



Şekil 2.25. DDPG ajanı ile her bölüm için adım sayısının değişimi

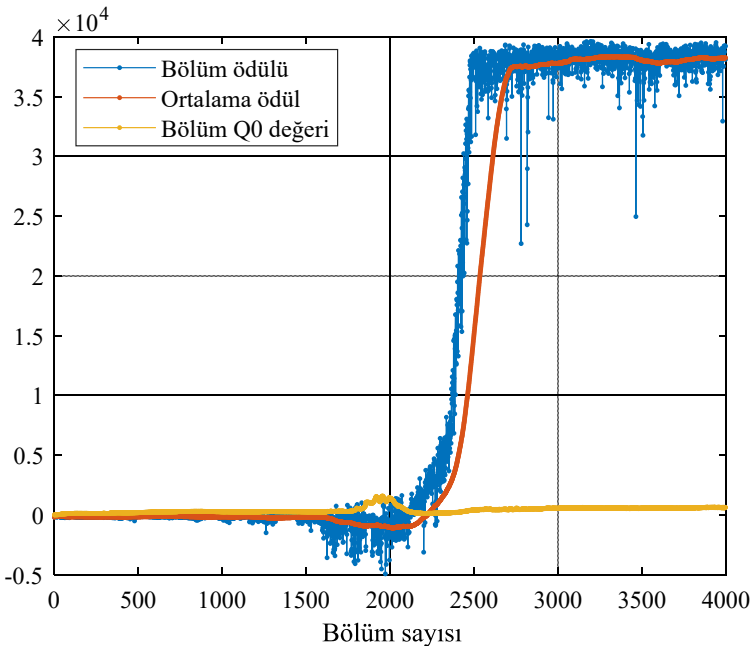
2.4.7. DDPG-*ė* ile Elde Edilen Sonuçlar

DDPG ajanıyla yapılan bu eğitimde gözlem uzayına referans yörünge takip hatalarının türevleri eklenerek gözlem uzayı 28 boyuta çıkarılmıştır. Hatanın türevi filtre çıkışından elde edilen referansın türevi kullanılarak $\dot{\phi}_{ref}$ elde edilmiştir. Eğitim parametreleri Tablo 2.9'daki gibi olup ajan parametreleri Tablo 2.11'deki gibi ayarlanmıştır. Bu parametreler kullanılarak yapılan eğitimin eğrisi ve her bölüm için adım sayısı değişimi sırasıyla Şekil 2.26 ve Şekil 2.27'de verilmiştir. Eğitim eğrisinden bölüm

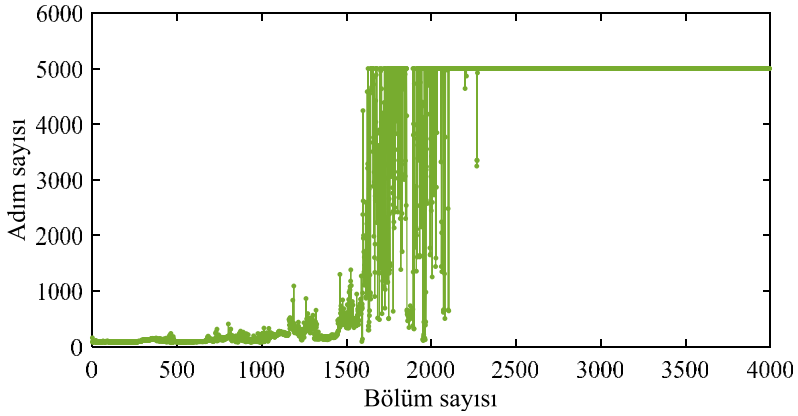
ödülünün yaklaşık 2200. bölümden sonra kararlı bir şekilde artmaya başladığı ve ortalama ödülün yaklaşık 38150 değerine yakınsadığı görülmektedir. Ortalama ödülün yakınsamasıyla birlikte eğitim 3995. bölümde manuel olarak durdurulmuştur. Eğitim sonucunda elde edilen maksimum bölüm ödülü 39619 olup eğitim süreci boyunca toplam 11402395 veri kullanılmıştır. Bu eğitim yaklaşık 200 saat sürmüştür. Ajan kaydetme kriterini sağlayan ajanlar test edilerek en iyi ajanın denetleyici performansı Bölüm 2.4.10'da diğer algoritmalardan elde edilen ajanların sonuçlarıyla karşılaştırılmıştır.

Tablo 2.11. DDPG-*ė* ajanına ait parametreler

Parametreler	Değerleri
Aktör için öğrenme hızı	α_a =1e^{-4}
Kritik için öğrenme hızı	α_c =1e^{-3}
Azaltma faktörü	γ = 0.99
Yumuşak hedef güncelleme katsayısı	τ= 1e^{-3}
Hedef güncelleme frekansı	1
Bellek boyutu	1e^{6}
Mini-paket boyutu	256
Aksiyon gürültü modeli	OU
Gürültü standart sapması	0.8
Minimum standart sapma	1e^{-3}
Standart sapma bozulma oranı	2e^{-4}
Ortalamaya çekim sabiti	1



Şekil 2.26. DDPG-*ė* ajanına ait eğitim eğrisi



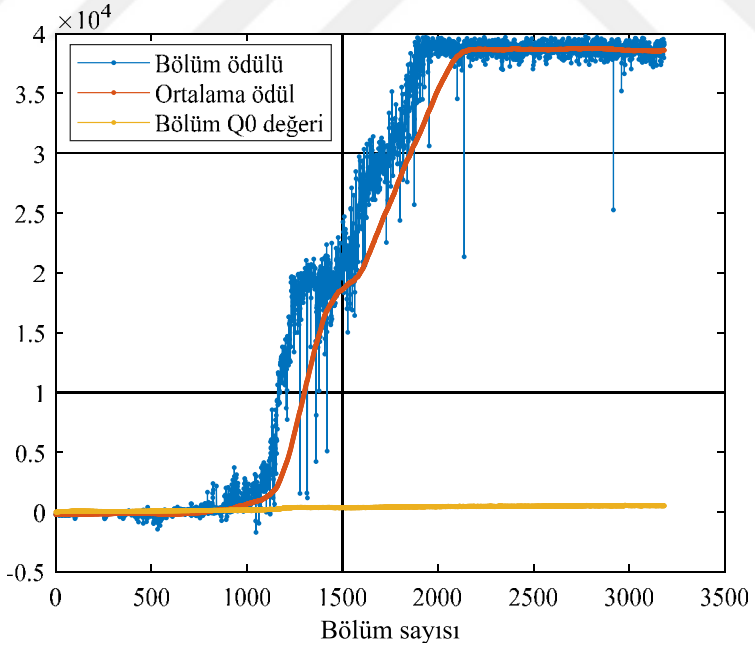
Şekil 2.27. DDPG-*ė* ajanı ile her bölüm için adım sayısının değişimi

2.4.8. TD3 ile Elde Edilen Sonuçlar

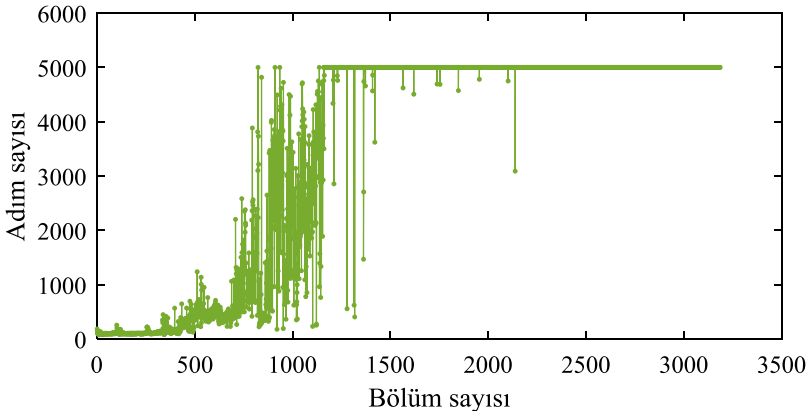
Bu eğitimde yılan robotun eklem yörünge kontrolü için TD3 ajanı kullanılmıştır. Gözlem ve aksiyon uzayı, ödül fonksiyonu ve kullanılan AC ağ mimarisi DDPG algoritması ile aynıdır. Ajan kaydetme kriteri dışındaki bütün eğitim parametreleri de Tablo 2.9'daki gibidir. Bölüm adım sayısı olarak belirlenen ajan kaydetme kriterinin değeri maksimum adım sayısı olan 5000'e ayarlanmıştır. Bu eğitime ait TD3 ajan parametreleri de Tablo 2.12'de verilmiştir. Şekil 2.28'deki TD3 ajanıyla elde edilen öğrenme eğrisinden bölüm ödülünün yaklaşık 1900 bölüm sonra kararlı yüksek bir ödüle ulaştığı görülmektedir. Elde edilen maksimum bölüm ödülü 39765 olup ortalama ödül yaklaşık 38600 değerine yakınsamıştır. Diğer ajanlarda olduğu gibi ortalama ödülün yakınsamasıyla birlikte eğitim 3184. bölümde manuel olarak durdurulmuştur. Bu eğitim süreci boyunca toplam 11175785 veri kullanılmış ve her bölüm için adım sayısı değişimi Şekil 2.29'da verilmiştir. 189 saat süren eğitim sonucunda elde edilen en iyi ajanın denetleyici performansı Bölüm 2.4.10'da diğer algoritmalardan elde edilen ajanların sonuçlarıyla karşılaştırmalı olarak verilmiştir.

Tablo 2.12. TD3 ajanına ait parametreler

Parametreler	Değerleri
Aktör için öğrenme hızı	α_a =1e^{-3}
Kritik için öğrenme hızı	α_c =1e^{-3}
Azaltma faktörü	γ = 0.99
Bellek boyutu	1e6
Mini-paket boyutu	256
Keşif aksiyon gürültü modeli	OU
Keşif aksiyon gürültüsünün standart sapması	0.8
Keşif aksiyon gürültüsünün minimum standart sapması	1e^{-3}
Keşif aksiyon gürültüsünün standart sapma bozulma oranı	2e^{-4}
Ortalamaya çekim sabiti	1
Politika güncelleme frekansı	2
Hedef aksiyon gürültü modeli	Gauss
Hedef aksiyon gürültüsünün standart sapması	0.32
Hedef aksiyon gürültü standart sapması alt ve üst sınırları	-0.3, 0.3
Hedef aksiyon gürültüsünün standart sapma bozulma oranı	0
Hedef aksiyon gürültüsünün minimum standart sapması	0.1
Yumuşak hedef güncelleme katsayısı	τ= 5e^{-3}
Hedef güncelleme frekansı	2



Şekil 2.28. TD3 ajanına ait eğitim eğrisi

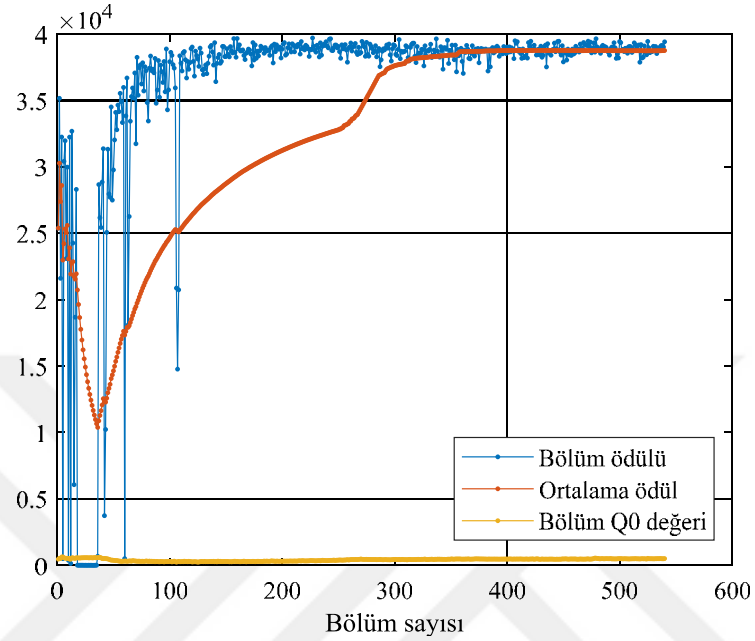


Şekil 2.29. TD3 ajanı ile her bölüm için adım sayısının değişimi

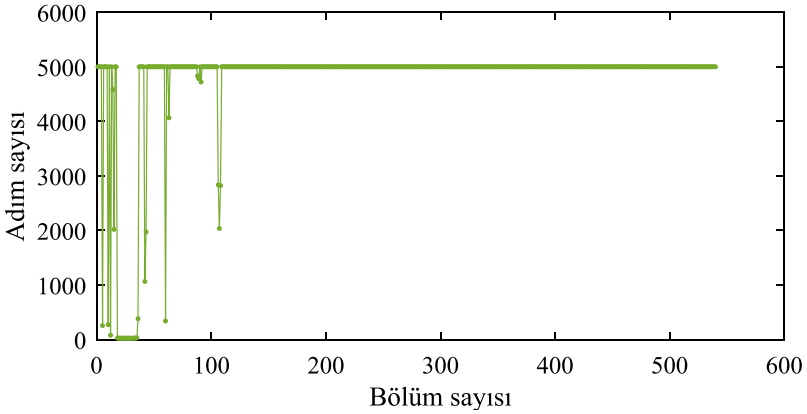
2.4.9. Transfer-TD3 ile Elde Edilen Sonuçlar

Eğitimin başlarında referans hata değerleri büyük olduğu için şimdiye kadar yapılan bütün denemelerde simülasyon durdurma kriteri e_ϕ >5 olarak belirlenmiştir. Bu koşulun daha küçük seçilmesi durumunda yapılan eğitimlerde bütün bölümler için adım sayıları küçük değerlerde kalarak eğitilen ajanın simülasyonun sadece ilk saniyelerini öğrenmesine neden olmaktadır. Fakat e_ϕ >5 seçilmesi durumunda da özellikle motorun limitlerine yaklaşan referanslar için eklem 2 ve eklem 3 gibi robotun orta eklemlerinde geçici durum yörünge takip hataları daha küçük olabilecekken eğitimi başlatma probleminden dolayı girilen koşul dolayısıyla büyümektedir. Bu problemi çözmek için transfer öğrenmeden yararlanılarak eğitilmiş TD3 ajanı eğitimin başlangıç noktası olarak kullanılmıştır. Böylece durdurma kriteri e_ϕ >1.5 olarak belirlendiğinde eğitime rastgele ağ parametreleriyle değil de önceden eğitilmiş ağ parametreleriyle başlandığı için ajan simülasyondan çıkmadan maksimum adım sayısına ulaşabildiği için simülasyonun tamamını öğrenebilmiştir. Bu denemede e_ϕ >5 için önceden eğitilen TD3 ajanının sadece parametreleri transfer edilmiş olup ajanın deneyim belleği aktarılmamıştır. Tablo 2.9'daki eğitim parametreleri ve Tablo 2.12'deki ajan parametreleri kullanılarak yapılan eğitim sonucunda elde edilen eğitim eğrisi Şekil 2.30'da, her bölüm için adım sayısı değişimi de Şekil 2.31'de verilmiştir. Şekil 2.30'dan bölüm ödülünün küçük bir dalgalanmadan sonra hızlı bir şekilde kararlı hale geldiği görülmektedir. Yani Transfer- TD3 ajanı ile sadece 100 bölüm sonra yılan robotun eklem yörünge kontrolü için strateji üretilebilmiştir. Bölüm ödülünün aldığı maksimum değer 39705 olup ortalama ödül yaklaşık olarak 38750 değerine yakınsamaktadır. 541. bölümde manuel olarak durdurulan eğitim süreci yaklaşık 45 saat sürmüş ve toplamda

2567562 veri kullanılmıştır. Eğitim sonucunda elde edilen en iyi ajanın denetleyici performansı Bölüm 2.4.10'da diğer algoritmalardan elde edilen ajanların sonuçlarıyla karşılaştırmalı olarak verilmiştir.



Şekil 2.30. Transfer-TD3 ajanına ait eğitim eğrisi

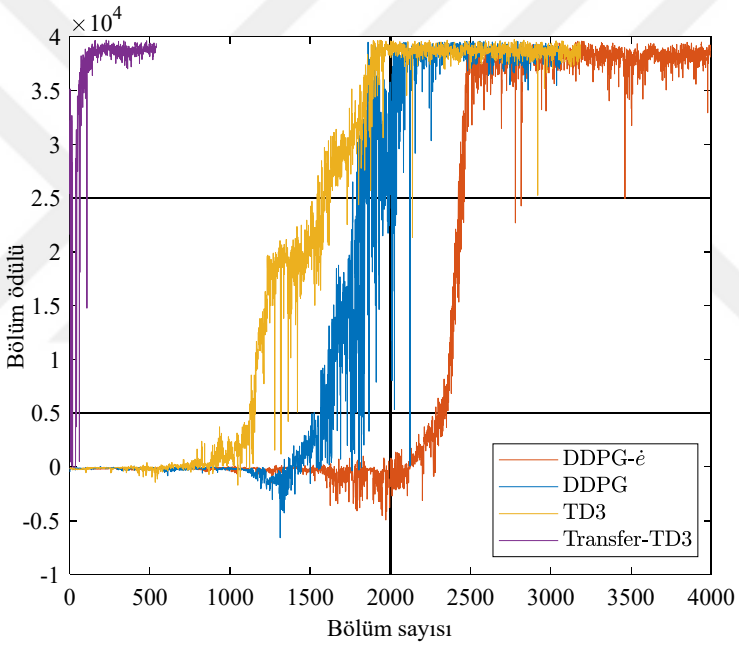


Şekil 2.31. Transfer-TD3 ajanı ile her bölüm için adım sayısının değişimi

2.4.10. Sonuçların Karşılaştırılması

DDPG, DDPG-*ė*, TD3 ve Transfer-TD3 ajanları ile yapılan eğitimlere ait öğrenme eğrileri Şekil 2.32'de verilmiştir. Bu şekle göre her ajanın öğrenme eğrisinin kararlı bir

değere yakınsayabildiği fakat TD3 ajanlarının DDPG ajanlarına göre daha erken yakınsadıkları yani daha hızlı öğrenebildikleri görülmektedir. DDPG ajanları kendi aralarında karşılaştırıldığında hatanın türevini gözlem olarak içeren eğitimin daha geç yakınsamasına rağmen öğrenme eğrisinin çok daha kararlı olduğu görülmektedir. TD3 ajanları arasında yapılan karşılaştırmada ise transfer öğrenimi ile TD3 ajanının yakınsama hızının arttığı görülmektedir. Bu da TD3 ile karşılaştırıldığında eğitim süresinde neredeyse %76'lık bir azalma ile Transfer-TD3 ajanının çok daha hızlı bir öğrenme gerçekleştirdiği anlamına gelir. Ayrıca Transfer-TD3 ile elde edilen öğrenme eğrisinin, transfer öğrenme ile kazanılması beklenen daha yüksek bir başlangıç, daha dik bir eğim ve daha yüksek bir değere yakınsamayı içeren üç faydanın hepsini sağladığı görülmektedir.



Şekil 2.32. Farklı DRL ajanlarına ait eğitim eğrilerinin karşılaştırılması

Eğitim ve test verileri için robotun eklemlerine göre DRL tabanlı denetleyicilerin hatanın karesinin integrali (Integral Square Error, ISE) performans ölçüt değerleri sırasıyla Tablo 2.13 ve Tablo 2.14'te verilmiştir. Bu tablolarda ayrıca geleneksel kontrol yöntemi olan PD denetleyicinin yılan robotun eklem yörünge kontrolü için uygulanmasıyla elde edilen sonuçlar da sunulmuştur. PD denetleyici Bölüm 2.2.2'de anlatıldığı gibi uygulanmış olup denetleyici parametreleri Tablo 2.1'deki gibidir.

Tablo 2.13 ve Tablo 2.14'e göre hem eğitim hem de testte kullanılan referansların hepsi için eklem 1'in en düşük hata performans değerlerinin PD ile elde edildiği

görülmüştür. Eklem 2 için ISE değerlerine bakıldığı zaman ise PD ve DDPG-*ė*'nin daha iyi olduğu birkaç referans dışında DDPG'nin üstünlük sağladığı görülmüştür. Eklem 3 için referansların çoğunda en iyi performansı DDPG-*ė* sunarken bazı referanslar için ise PD başarılı olmuştur. Eklem 4 için ise birkaç referans için TD3 ile daha düşük hata değerleri elde edilse de çoğu referans için PD'nin daha iyi sonuçlar ürettiği görülmüştür. Bu sonuçlardan anlaşılacağı gibi eklem ya da referans bazlı bakıldığında tek bir denetleyicinin bütün durumlarda daha iyi performans sunduğu söylenememektedir. Bu nedenle bu tablolarda ayrıca her referans için dört eklemin ortalama ISE değerleri verilmiştir. Bu sonuçlara bakıldığı zaman ise eklemlerdeki hatayı genelleme yapabilmesi açısından en iyi performansı sunan denetleyicinin Transfer-TD3 olduğu görülmektedir. Transfer-TD3'ten sonra en iyi genelleme PD denetleyici ile elde edilmiştir.

Tablo 2.13. Eğitim verileri için denetleyicilerin ISE performans ölçüt değerleri

Referans		**Denetleyici**	**Eklem 1**	**Eklem 2**	**Eklem 3**	**Eklem 4**	**Ort. ISE**
			ISE				
1	$\alpha = 30^\circ$ $\omega = 210^\circ/s$ $\beta = 60^\circ$	PD	**0.0001**	**0.0015**	**0.0008**	**0.0001**	**0.0006**
		DDPG	0.0068	0.0083	0.0088	0.0043	0.0071
		DDPG-*ė*	0.0040	0.0024	0.0025	0.0237	0.0082
		TD3	0.0018	0.0023	0.0021	0.0037	0.0025
		Transfer-TD3	0.0020	0.0024	0.0030	0.0016	0.0023
2	$\alpha = 60^\circ$ $\omega = 210^\circ/s$ $\beta = 40^\circ$	PD	**0.0005**	0.0180	0.0929	**0.0009**	0.0280
		DDPG	0.0070	**0.0070**	0.1401	0.0055	0.0399
		DDPG-*ė*	0.0048	0.0274	**0.0042**	0.2211	0.0644
		TD3	0.0015	0.0224	0.0567	0.0032	0.0210
		Transfer-TD3	0.0015	0.0242	0.0502	0.0021	**0.0195**
3	$\alpha = 80^\circ$ $\omega = 210^\circ/s$ $\beta = 50^\circ$	PD	**0.0002**	0.0923	0.1054	**0.0009**	0.0497
		DDPG	0.0066	**0.0057**	0.3250	0.0046	0.0855
		DDPG-*ė*	0.0053	0.1152	**0.0033**	0.1398	0.0659
		TD3	0.0008	0.0986	0.0764	0.0028	0.0447
		Transfer-TD3	0.0009	0.0994	0.0618	0.0018	**0.0410**
4	$\alpha = 90^\circ$ $\omega = 210^\circ/s$ $\beta = 40^\circ$	PD	**0.0003**	0.1204	0.2816	0.0019	0.1010
		DDPG	0.0058	**0.0038**	0.7309	0.0046	0.1863
		DDPG-*ė*	0.0060	0.1572	**0.0037**	0.5442	0.1778
		TD3	0.0022	0.1531	0.1985	**0.0016**	0.0889
		Transfer-TD3	0.0025	0.1590	0.1778	0.0029	**0.0855**
5	$\alpha = 30^\circ$ $\omega = 210^\circ/s$ $\beta = 40$	PD	**0.0002**	**0.0013**	0.0077	**0.0003**	**0.0023**
		DDPG	0.0070	0.0079	0.0137	0.0035	0.0080
		DDPG-*ė*	0.0041	0.0030	**0.0029**	0.0360	0.0115
		TD3	0.0014	0.0020	0.0030	0.0031	0.0024
		Transfer-TD3	0.0007	0.0021	0.0046	0.0025	0.0025

Tablo 2.13'ün devamı

Referans		Denetleyici	Eklem 1	Eklem 2	Eklem 3	Eklem 4	Ort. ISE
			ISE				
6	$\alpha = 70^\circ$ $\omega = 210^\circ/s$ $\beta = 40^\circ$	PD	**0.0004**	0.0384	0.1435	**0.0012**	0.0458
		DDPG	0.0067	**0.0061**	0.2697	0.0052	0.0719
		DDPG-$\dot{e}$	0.0051	0.0581	**0.0039**	0.3167	0.0959
		TD3	0.0017	0.0470	0.0952	0.0030	0.0367
		Transfer-TD3	0.0016	0.0466	0.0874	0.0014	**0.0342**
7	$\alpha = 80^\circ$ $\omega = 210^\circ/s$ $\beta = 40^\circ$	PD	**0.0003**	0.0713	0.2071	**0.0016**	0.0700
		DDPG	0.0061	**0.0044**	0.4549	0.0055	0.1178
		DDPG-$\dot{e}$	0.0057	0.1013	**0.0030**	0.4202	0.1326
		TD3	0.0018	0.0911	0.1417	0.0021	0.0592
		Transfer-TD3	0.0017	0.0909	0.1294	0.0019	**0.0560**
8	$\alpha = 90^\circ$ $\omega = 210^\circ/s$ $\beta = 60^\circ$	PD	**0.0001**	0.1522	0.0311	**0.0007**	0.0460
		DDPG	0.0061	**0.0066**	0.2761	0.0046	0.0733
		DDPG-$\dot{e}$	0.0043	0.1669	**0.0036**	0.0217	0.0491
		TD3	0.0015	0.1648	0.0198	0.0027	0.0472
		Transfer-TD3	0.0014	0.1252	0.0183	0.0015	**0.0366**
9	$\alpha = 70^\circ$ $\omega = 210^\circ/s$ $\beta = 50^\circ$	PD	**0.0003**	0.0535	0.0715	**0.0007**	0.0315
		DDPG	0.0068	**0.0068**	0.2001	0.0052	0.0547
		DDPG-$\dot{e}$	0.0055	0.0711	**0.0035**	0.0963	0.0441
		TD3	0.0009	0.0544	0.0494	0.0039	0.0272
		Transfer-TD3	0.0012	0.0542	0.0415	0.0013	**0.0245**
10	$\alpha = 70^\circ$ $\omega = 210^\circ/s$ $\beta = 60^\circ$	PD	**0.0002**	0.0616	0.0142	**0.0004**	0.0191
		DDPG	0.0064	**0.0073**	0.1273	0.0046	0.0364
		DDPG-$\dot{e}$	0.0047	0.0582	**0.0057**	0.0187	0.0218
		TD3	0.0014	0.0603	0.0102	0.0039	0.0189
		Transfer-TD3	0.0005	0.0441	0.0092	0.0018	**0.0139**

Tablo 2.14. Test verileri için denetleyicilerin ISE performans ölçüt değerleri

Referans		Denetleyici	Eklem 1	Eklem 2	Eklem 3	Eklem 4	Ort. ISE
			ISE				
1	$\alpha = 85^\circ$ $\omega = 210^\circ/s$ $\beta = 65^\circ$	PD	**0.0001**	0.1150	**0.0066**	**0.0005**	0.0305
		DDPG	0.0060	**0.0084**	0.1560	0.0049	0.0438
		DDPG-$\dot{e}$	0.0040	0.1220	0.0046	0.0049	0.0339
		TD3	0.0016	0.1304	0.0102	0.0038	0.0365
		Transfer-TD3	0.0021	0.0767	0.0075	0.0016	**0.0220**
2	$\alpha = 50^\circ$ $\omega = 210^\circ/s$ $\beta = 70^\circ$	PD	**0.0001**	0.0105	**0.0006**	**0.0001**	**0.0028**
		DDPG	0.0066	**0.0094**	0.0135	0.0047	0.0086
		DDPG-$\dot{e}$	0.0037	0.0104	0.0060	0.0039	0.0060
		TD3	0.0026	0.0164	0.0043	0.0040	0.0068
		Transfer-TD3	0.0026	0.0107	0.0032	0.0028	0.0048

Tablo 2.14'ün devamı

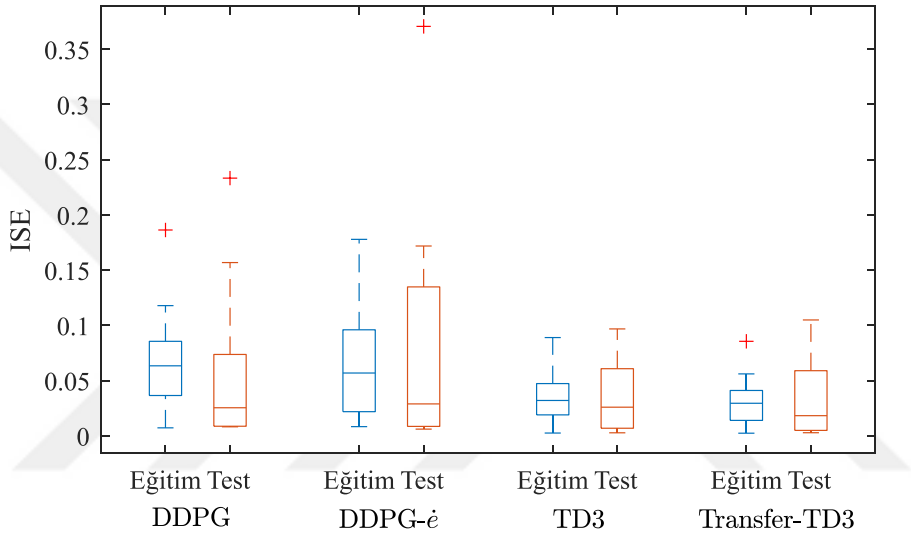
Referans		Denetleyici	Eklem 1	Eklem 2	Eklem 3	Eklem 4	Ort. ISE
			ISE				
3	$\alpha = 30^\circ$ $\omega = 180^\circ/s$ $\beta = 50^\circ$	PD	**0.0001**	**0.0010**	0.0038	**0.0001**	**0.0012**
		DDPG	0.0069	0.0090	0.0119	0.0048	0.0081
		DDPG-$\dot{e}$	0.0034	0.0023	**0.0010**	0.0239	0.0076
		TD3	0.0015	0.0020	0.0034	0.0033	0.0026
		Transfer-TD3	0.0012	0.0027	0.0053	0.0018	0.0027
4	$\alpha = 90^\circ$ $\omega = 210^\circ/s$ $\beta = 30^\circ$	PD	**0.0005**	0.0906	0.3151	0.0036	0.1024
		DDPG	0.0060	**0.0050**	0.9173	0.0047	0.2333
		DDPG-$\dot{e}$	0.0047	0.1025	**0.0071**	1.3680	0.3707
		TD3	0.0128	0.1267	0.2446	**0.0029**	**0.0967**
		Transfer-TD3	0.0112	0.1741	0.2151	0.0188	0.1048
5	$\alpha = 90^\circ$ $\omega = 210^\circ/s$ $\beta = 45^\circ$	PD	**0.0002**	0.1331	0.2222	**0.0014**	0.0892
		DDPG	0.0065	**0.0040**	0.6123	0.0044	0.1568
		DDPG-$\dot{e}$	0.0061	0.1661	**0.0036**	0.3635	0.1348
		TD3	0.0015	0.1623	0.1550	0.0019	0.0802
		Transfer-TD3	0.0020	0.1681	0.1360	0.0019	**0.0770**
6	$\alpha = 40^\circ$ $\omega = 210^\circ/s$ $\beta = 60^\circ$	PD	**0.0001**	0.0059	**0.0020**	**0.0002**	**0.0020**
		DDPG	0.0068	0.0081	0.0138	0.0037	0.0081
		DDPG-$\dot{e}$	0.0050	**0.0044**	0.0030	0.0210	0.0084
		TD3	0.0019	0.0059	0.0038	0.0043	0.0040
		Transfer-TD3	0.0017	0.0057	0.0044	0.0017	0.0034
7	$\alpha = 60^\circ$ $\omega = 210^\circ/s$ $\beta = 60^\circ$	PD	**0.0002**	0.0338	0.0084	**0.0003**	0.0106
		DDPG	0.0064	**0.0080**	0.0651	0.0042	0.0209
		DDPG-$\dot{e}$	0.0048	0.0321	**0.0051**	0.0127	0.0137
		TD3	0.0018	0.0314	0.0070	0.0043	0.0111
		Transfer-TD3	0.0004	0.0231	0.0068	0.0018	**0.0080**
8	$\alpha = 90^\circ$ $\omega = 210^\circ/s$ $\beta = 80^\circ$	PD	**0.0001**	0.0387	**0.0007**	**0.0002**	**0.0099**
		DDPG	0.0110	**0.0206**	0.0435	0.0110	0.0215
		DDPG-$\dot{e}$	0.0066	0.0450	0.0104	0.0329	0.0237
		TD3	0.0095	0.1274	0.0948	0.0111	0.0607
		Transfer-TD3	0.0190	0.1713	0.0344	0.0112	0.0589
9	$\alpha = 70^\circ$ $\omega = 210^\circ/s$ $\beta = 30^\circ$	PD	**0.0008**	0.0257	0.1681	0.0022	0.0492
		DDPG	0.0069	**0.0084**	0.2736	0.0055	0.0736
		DDPG-$\dot{e}$	0.0056	0.0336	**0.0064**	0.6416	0.1718
		TD3	0.0049	0.0379	0.1311	**0.0020**	0.0440
		Transfer-TD3	0.0087	0.0511	0.0999	0.0036	**0.0408**
10	$\alpha = 55^\circ$ $\omega = 210^\circ/s$ $\beta = 40^\circ$	PD	**0.0004**	0.0116	0.0714	**0.0008**	0.0210
		DDPG	0.0071	**0.0074**	0.0975	0.0049	0.0292
		DDPG-$\dot{e}$	0.0050	0.0180	**0.0045**	0.1791	0.0516
		TD3	0.0015	0.0164	0.0395	0.0032	0.0152
		Transfer-TD3	0.0013	0.0153	0.0383	0.0024	**0.0143**

Referanslara göre genellemede performans kıyaslaması yapabilmek için denetleyicilerin bütün referanslar için ortalama eklem ISE değerlerinin ortalamaları alınarak eğitim ve test hatası olarak Tablo 2.15'te verilmiştir. Bu tabloda eğitim ve testte kullanılan referanslara göre ortalama eklem ISE değerlerinin maksimum ve minimum değerleri de verilmiştir. Bu sonuçlara göre eğitim ve testte kullanılan referansların ortalama eklem ISE değerlerinin ortalamasının daha düşük olduğu denetleyicilerin sırasıyla Transfer-TD3, TD3 ve PD olduğu görülmektedir. Transfer-TD3'ün hiçbir referans ve eklem için en iyi sonucu vermemesine rağmen ortalamaya bakıldığı zaman iyi bir performans sunmasının sebebi robotun bütün eklemlerindeki hata değerlerinin birbirine daha yakın ve dengeli olmasıdır. Diğer denetleyicilere bakıldığı zaman genellikle eklem 1 ve eklem 4 gibi robotun kontrolü daha kolay olan uç eklemlerinde çok küçük ISE değerleri sunarken eklem 2 ve eklem 3 gibi kontrolü daha zor olan orta eklemlerde daha büyük bir hata performans değeri sunmaları ortalamalarını yükseltmiştir.

Şekil 2.33'te DRL tabanlı denetleyicilerin eğitim ve test referans ortalama eklem ISE değerlerinin kutu diyagramı karşılaştırılmalı olarak verilmiştir. Bu şekilden eğitim ve test referanslarının genellemesinde en düşük standart sapmaya sahip dağılımının Transfer-TD3 denetleyicisi ile elde edildiği görülmektedir. Transfer-TD3'ten sonra en iyi dağılım TD3 tarafından benzer şekilde elde edilmiştir. DDPG ve DDPG-$\dot{e}$ ise daha büyük standart sapmaya sahip bir dağılım sergilemişlerdir. TD3 ajanları ile elde edilen eğitim ve test referanslarının ortalama eklem ISE değerlerinin DDPG ajanlarına göre daha küçük standart sapmaya sahip bir dağılım sergilemeleri TD3 ajanlarının daha iyi bir genelleme performansına sahip olduklarını gösterir. Bu şekilden ayrıca DRL tabanlı denetleyicilerin eğitim referans ortalama eklem ISE değerlerinin standart sapmasının test referans ortalama eklem ISE değerlerine göre daha küçük olduğu fakat aynı zamanda da birbirine yakın oldukları ve hata değerlerinin de düşük olduğu görülmektedir. Böylece eğitilmiş ajanların hiçbirinde aşırı öğrenme veya eksik öğrenmenin meydana gelmediği söylenebilir.

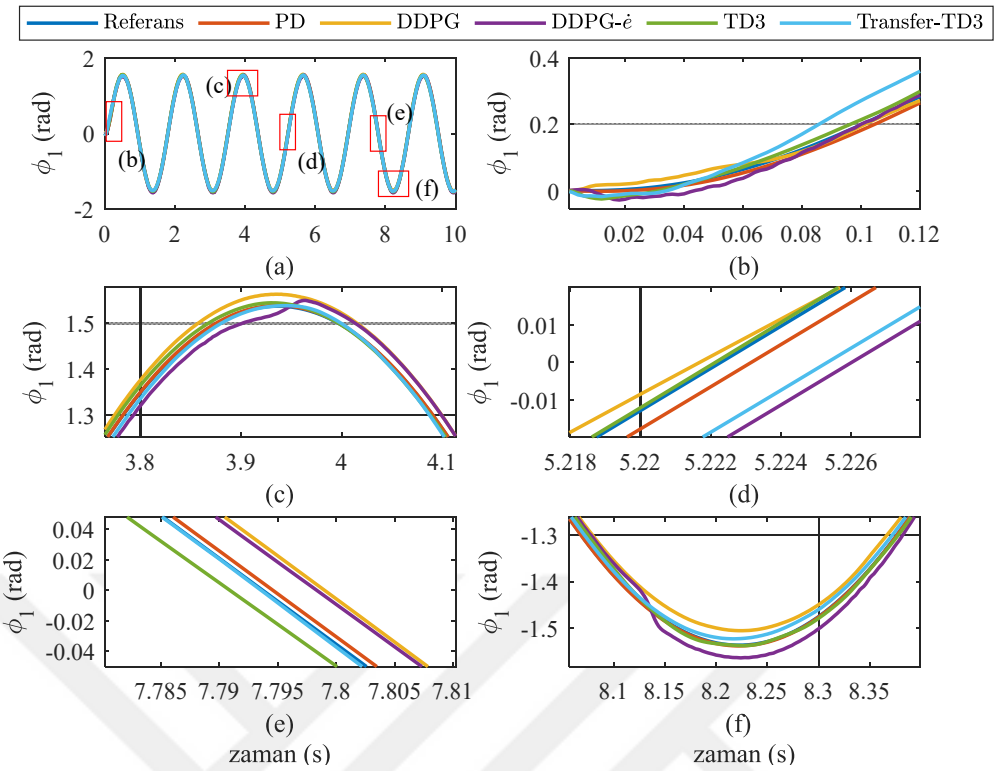
Tablo 2.15. Denetleyicilerin eğitim ve test referans genelleme hataları

Denetleyici	Eğitim			Test		
	Min	Max	Ort.	Min	Max	Ort.
PD	0.0006	0.1010	0.0394	0.0012	0.1024	0.0319
DDPG	0.0071	0.1863	0.0681	0.0081	0.2333	0.0604
DDPG-*ė*	0.0082	0.1778	0.0671	0.0060	0.3707	0.0822
TD3	0.0024	0.0889	0.0349	0.0026	0.0967	0.0358
Transfer-TD3	0.0023	0.0855	0.0316	0.0027	0.1048	0.0337

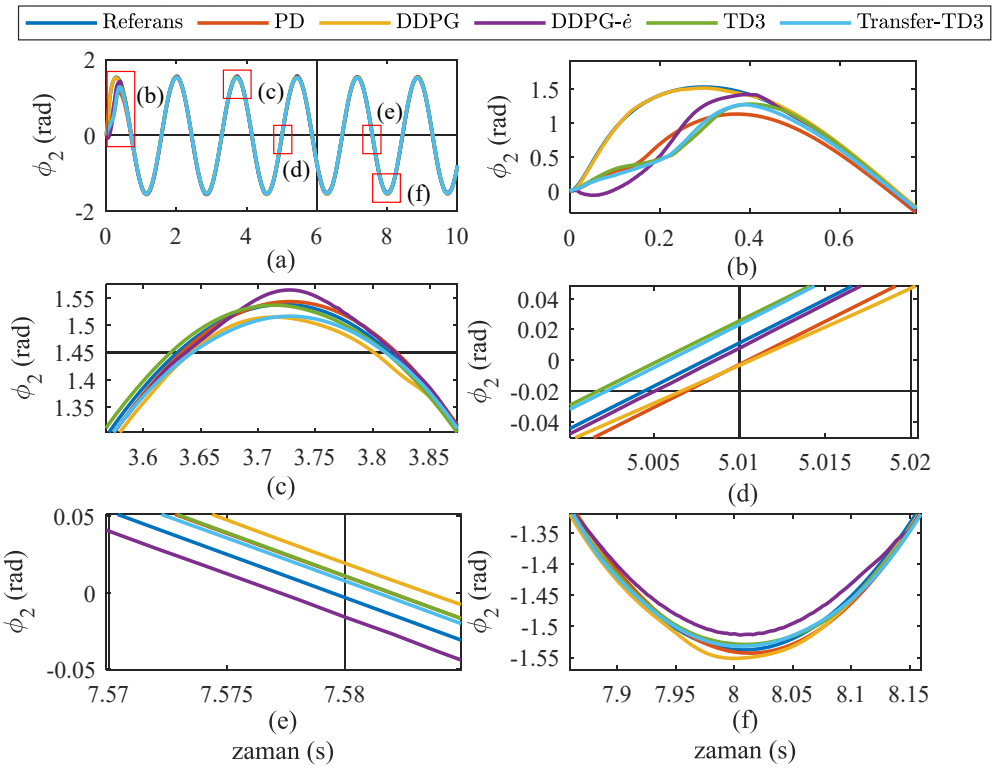


Şekil 2.33. Denetleyicilerin kendi aralarında eğitim ve test ortalama eklem ISE değerlerinin karşılaştırılması

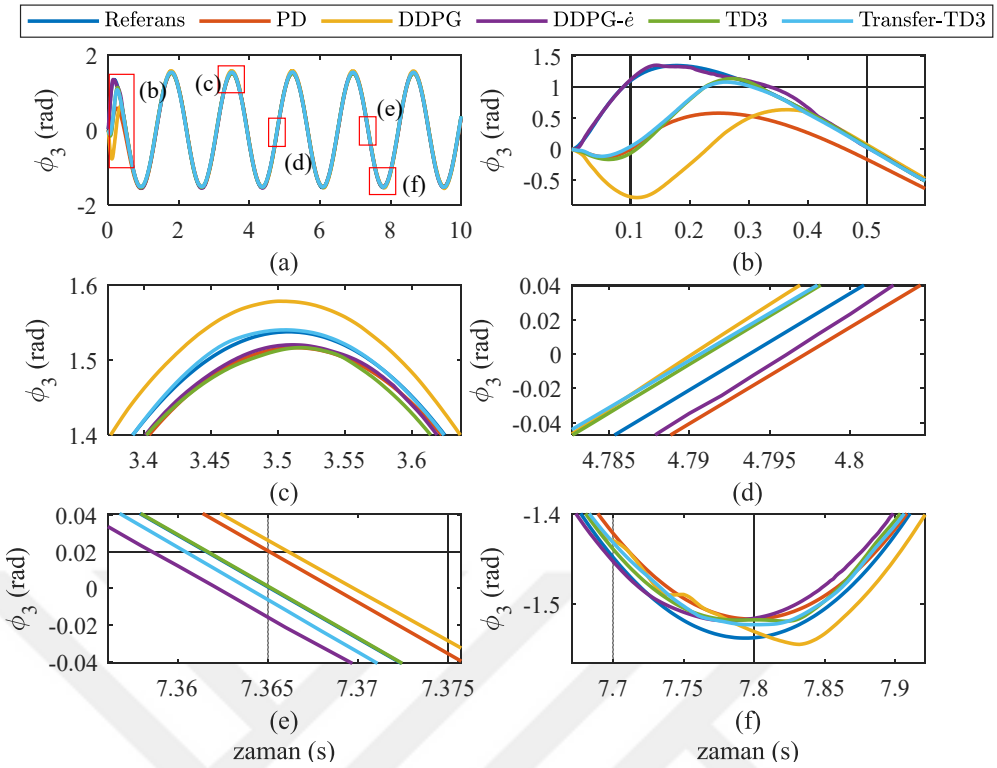
Denetim zorluğu açısından iki farklı test referansı için bu denetleyiciler ile elde edilen eklem yörüngeleri her bir eklem için Şekil 2.34- Şekil 2.41'de sunulmuştur. Denetleyicileri zorlayan test referans 5 ve göreceli olarak daha kolay olan test referans 6 bu gösterim için tercih edilmiştir. Şekillerin (a) şıkkında denetleyiciler ile elde edilen yörüngeler, (b)-(f) şıklarında ise şekil (a) üzerindeki seçili bölgelerin büyütülmüş halleri gösterilmiştir. Referansların etkisini daha doğru görebilmek için her iki referans için de şekillerin aynı bölgeleri büyütülmüştür. Bu bölgeler geçici durum anı, referans işaretin maksimum ve minimum tepe noktaları ve sinyalin işaret değiştirdiği x-eksenini içeren doğrusal bölgelerden seçilmiştir.



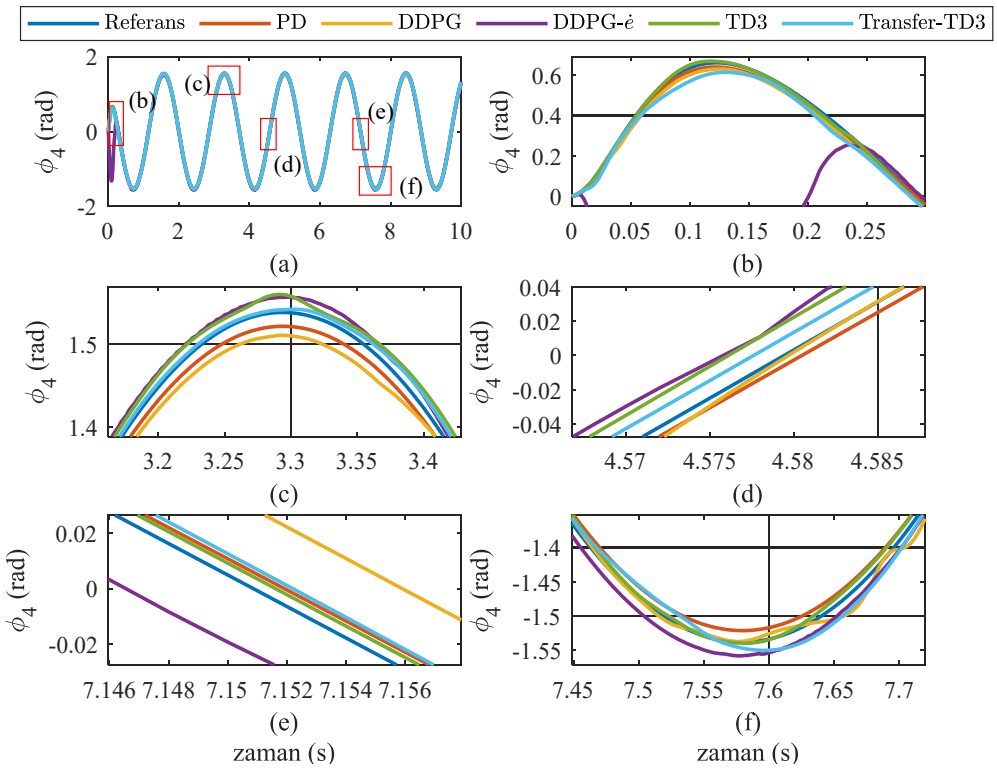
Şekil 2.34. (a) Test referans 5 için eklem 1 yörüngeleri (b)-(f) şekil (a) üzerindeki seçili bölgelerin büyütülmüş hali



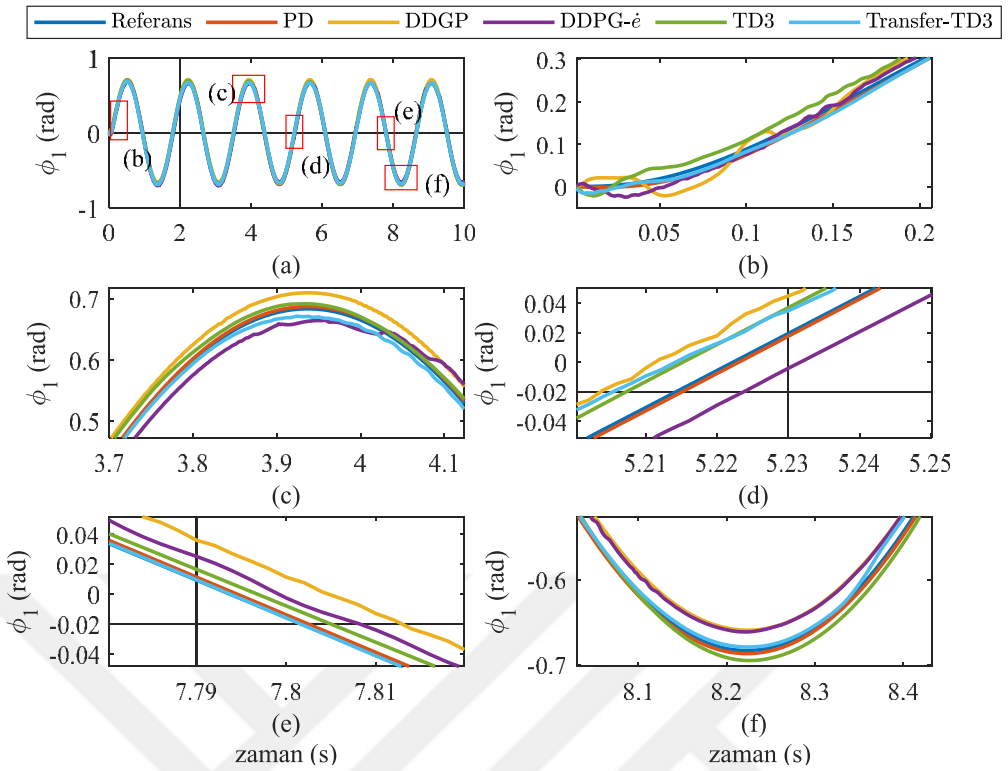
Şekil 2.35. (a) Test referans 5 için eklem 2 yörüngeleri (b)-(f) şekil (a) üzerindeki seçili bölgelerin büyütülmüş hali



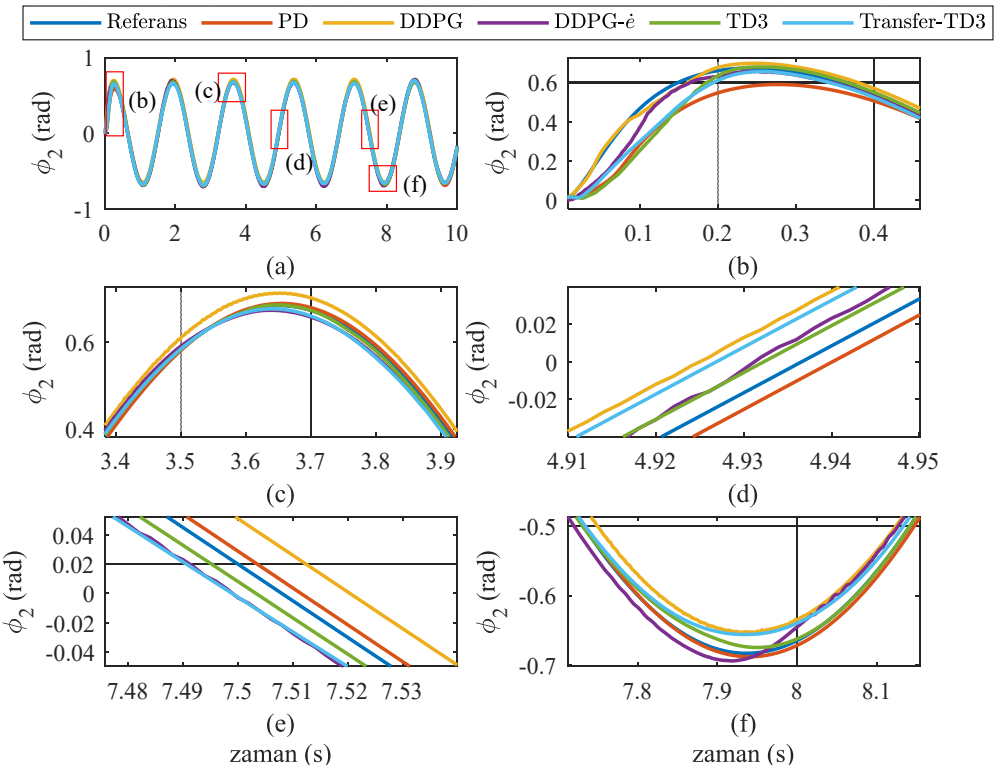
Şekil 2.36. (a) Test referans 5 için eklem 3 yörüngeleri (b)-(f) şekil (a) üzerindeki seçili bölgelerin büyütülmüş hali



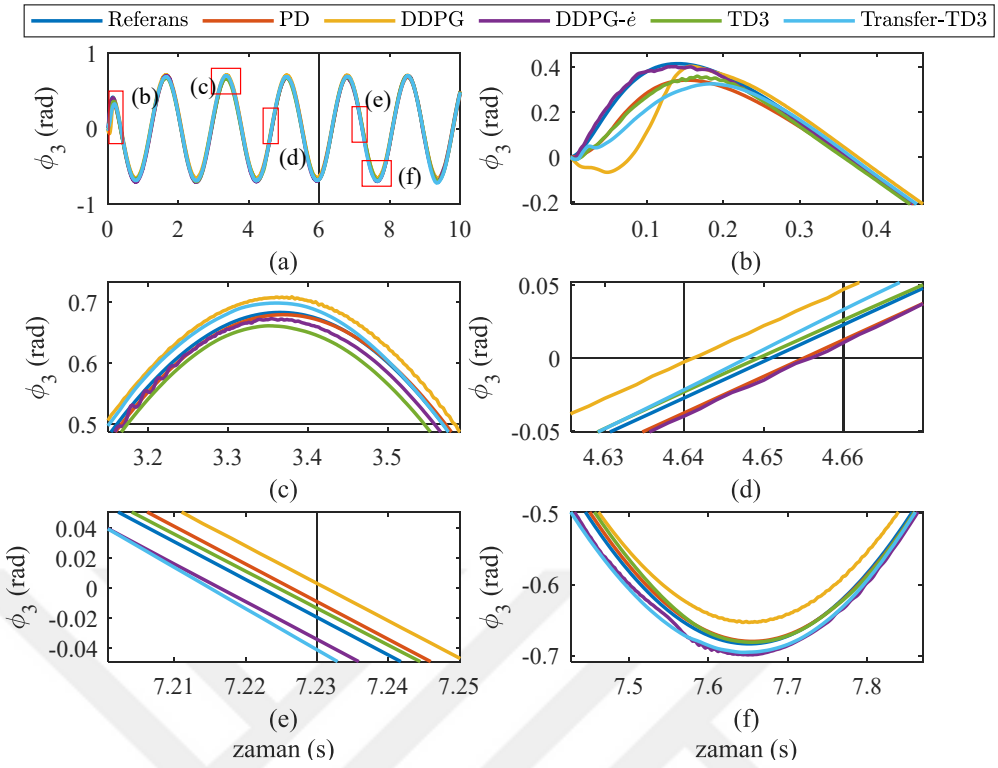
Şekil 2.37. (a) Test referans 5 için eklem 4 yörüngeleri (b)-(f) şekil (a) üzerindeki seçili bölgelerin büyütülmüş hali



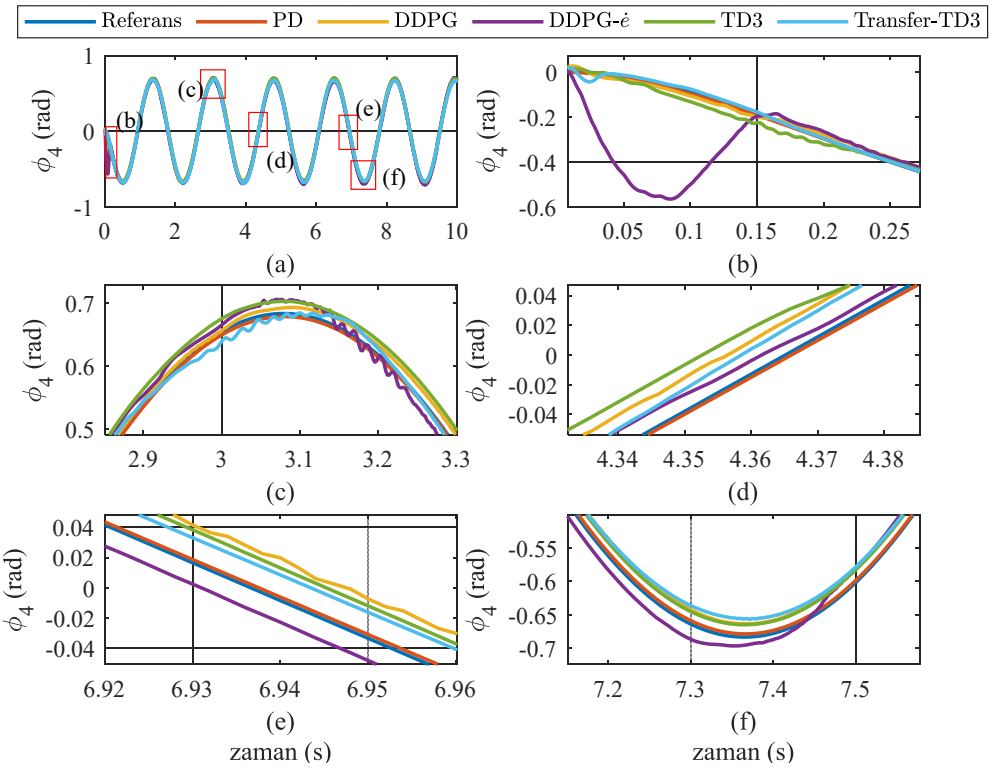
Şekil 2.38. (a) Test referans 6 için eklem 1 yörüngeleri (b)-(f) şekil (a) üzerindeki seçili bölgelerin büyütülmüş hali



Şekil 2.39. (a) Test referans 6 için eklem 2 yörüngeleri (b)-(f) şekil (a) üzerindeki seçili bölgelerin büyütülmüş hali



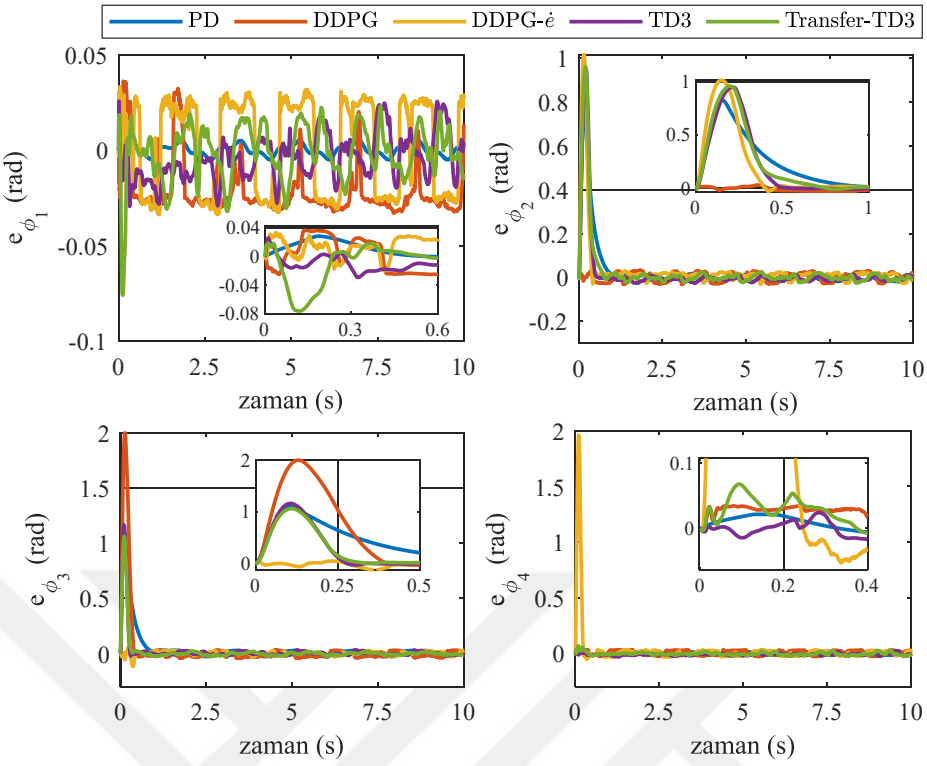
Şekil 2.40. (a) Test referans 6 için eklem 3 yörüngeleri (b)-(f) şekil (a) üzerindeki seçili bölgelerin büyütülmüş hali



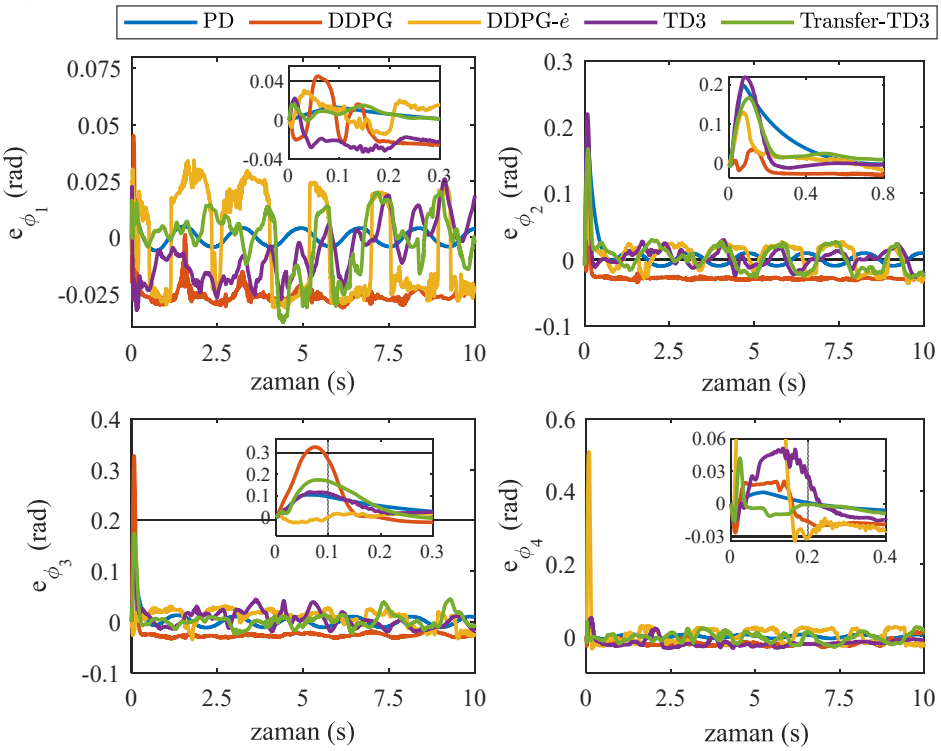
Şekil 2.41. (a) Test referans 6 için eklem 4 yörüngeleri (b)-(f) şekil (a) üzerindeki seçili bölgelerin büyütülmüş hali

Bu şekiller incelenecek olursa zor bir referans olan referans 5 için geçici durum anında eklem 1 için PD, eklem 2 için DDPG, eklem 3 için DDPG-*ė*, eklem 4 için ise TD3'nin yörüngeyi daha iyi takip edebildiği görülmektedir. Referans 5'in maksimum ve minimum tepe noktalarında ise genel olarak TD3 ve Transfer-TD3 daha üstün performans sunarken, lineer bölgelerde ise TD3 ve DDPG-*ė*'nin daha iyi olduğu söylenebilir. Referans 6 için ise bütün eklemler için maksimum ve minimum tepe noktalarında PD'nin çok iyi bir performans sunduğu, aynı şekilde eklem 1 ve eklem 4 için geçici durum anında ve lineer bölgelerde başarılı olduğu görülmektedir. Eklem 2 ve eklem 3 için ise geçici durum anında DDPG ve DDPG-*ė*, lineer bölgelerde ise TD3 daha iyi bir takip performansı sergilemiştir. Bu sonuçlardan anlaşıldığı gibi herhangi bir referansın herhangi bir bölgesi için herhangi bir denetleyicinin daha iyi sonuç verdiği şeklinde bir genelleme yapılamamaktadır.

Şekil 2.42 ve Şekil 2.43'te sırasıyla referans 5 ve referans 6 için denetleyicilere göre yılan robotun eklemlerine ait yörünge takip hataları verilmiştir. Bu şekillerde ayrıca geçici durum anının yakınlaştırılmış hali sunulmuştur. Referans 5 için özellikle eklem 3'te geçici durum anındaki eklem hatalarının eğitimdeki simülasyon durdurma koşulunda Transfer-TD3 ile sınırlandırılmasının etkisi gözlemlenebilmektedir. Referans 6 göreceli daha kolay bir yörünge olduğu için eklem yörünge hatalarının referans 5'e göre daha küçük olduğu görülmektedir. Eklemler arasında karşılaştırılma yapıldığı zaman ise eklem 2 ve eklem 3'teki yörünge takip hatalarının genel olarak daha büyük olduğu görülmektedir. Sadece DDPG-*ė* ile diğer denetleyicilerden farklı olarak eklem 4 için daha büyük yörünge hata değerleri elde edilmiştir.



Şekil 2.42. Referans 5 için denetleyicilere göre yılan robotun eklemlerine ait yörünge takip hataları



Şekil 2.43. Referans 6 için denetleyicilere göre yılan robotun eklemlerine ait yörünge takip hataları

2.5. Yılan Robotun Enerji Verimli Hareketi İçin Adaptif Yörünge Üretme

TD3 algoritmasının kullanıldığı DRL tabanlı denetleyici ile yılan robotun enerji verimli hareketi için adaptif yörüngeler üretilerek robotun hız kontrolü ele alınmıştır. Bunun için 0.05, 0.10, 0.15, 0.20 ve 0.25 m/s referans hız değerlerini içeren 5 farklı referans hız değeri kullanılmış ve eğitim boyunca her bölümde rastgele bir referans seçilerek ajanın eğitilmesi gerçekleştirilmiştir. Referans hız değerleri dışında referans eklem yörüngeleri gibi robotun hareketini tanımlayacak hiçbir referans işaret kullanılmamıştır. Buradaki amaç yılan robotun referans hızı değiştiği zaman enerji verimli hareketi sağlayabileceği yörüngeleri kendisinin üretip öğrenebilmesidir. Bu çalışma aslında yılan robotun farklı sürtünme katsayılarına sahip çevrelerde adaptif hareket edebilmesinin alt yapısını oluşturmaktadır. Çünkü literatürden ve daha önceden yaptığımız optimizasyon çalışmalarından bildiğimize göre sürtünme katsayısı değiştiği zaman robotun hızı değişecek ve enerji verimli hareket için yörüngesini değiştirmek zorunda kalacaktır. Daha önceden bu problemi sezgisel optimizasyon yöntemleriyle eklem referans yörüngelerinin parametrelerinin ayarlanması şeklinde ele almıştık. Bu yaklaşımda ise daha doğal ve biyolojik yılanın hareketine daha gerçekçi bir şekilde ulaşabilmek için yörüngeden bağımsız öğrenme gerçekleştirilmiştir.

Bu amaç için 14 boyutlu gözlem uzayı oluşturulmuştur. Bu büyüklükler Tablo 2.16'da verilmiştir. Aksiyon uzayı ise yine yılan robotun 4 tane 1 DOF eklemine uygulanacak tork sinyallerinden oluşur. Tork aksiyon sinyali [-2.3,2.3] Nm arasında bir değer alır. Kullanılan AC ağ mimarisi eklem yörünge kontrolü için kullanılan mimariyle aynıdır. Ödül fonksiyonu ise Şekil 2.44'te verilen hız hatasına ve güç tüketimine bağlı olarak iki temel bölümü içerir ve eşitlik (2.11)'deki gibi oluşturulmuştur. Ödül fonksiyonunun tasarımında Bing vd. (2020a)'nin çalışması referans olarak alınmıştır. Bu çalışmada AIRL algoritması kullanılarak tekerlekli bir yılan robotun enerji verimli hareketi ele alınmıştır. Bu çalışmadan farklı olarak gerçek motor limitleri göz önüne alınarak ödül fonksiyonu sınırlandırılmıştır. Aynı sınırlandırma simülasyonu durdurma koşulu için de kullanılmıştır.

$$r_t = \begin{cases} |\phi| < 90° \; ve \; |\dot{\phi}| < 429°/\text{s} & \left(1 - \dfrac{|v_{ref} - v_t|}{a_1}\right)^{1/a_2} |1 - \hat{P}|^{b_1^{-2}} \\ değilse & 0 \end{cases} \tag{2.11}$$

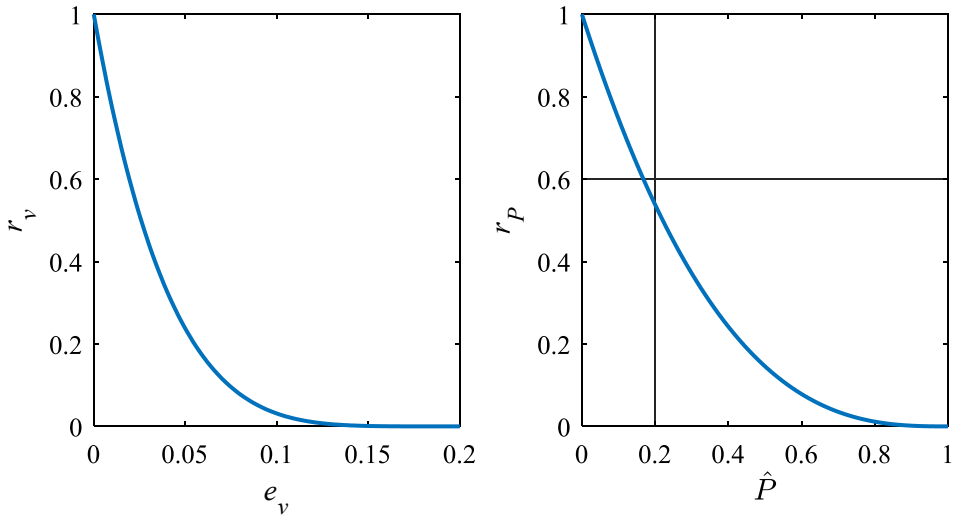
Tablo 2.16. Enerji verimli hareket için TD3 denetleyicinin gözlem uzayı

Sembol	Açıklama	Boyut
ϕ	Robotun eklem pozisyonları	4
$\dot{\phi}$	Robotun eklem hızları	4
u	Eklemlere uygulanan torklar	4
v_t	Robotun ileri yöndeki hızı	1
v_{ref}	Referans hız değeri	1

Bu eşitlikteki hız ödülü için tanımlanan r_v eğrisinin x eksenini kestiği noktayı belirleyen a_1 parametresi ile eğrinin eğimini tanımlayan a_2 parametresi 0.2 olarak belirlenmiştir. Toplam ödülün hız ödül kısmını oluşturan bölümü, hız hatası ($e_v = |v_{ref} - v_t|$) 0 olduğu zaman maksimum 1 değerini alacaktır. Güç verimliliğini oluşturan ödülün ikinci kısmındaki b_1 parametresi ise 0.6 olarak seçilmiş olup ödülün ikinci kısmını oluşturan r_P ödül eğrisinin eğimini belirler. Ödül fonksiyonundaki $\hat{P}$ ise normalize edilmiş güç tüketimini ifade eder ve eşitlik (2.12)'deki gibi hesaplanır.

$$\hat{P} = \frac{1}{n_e - 1} \sum_{i=1}^{n_e - 1} \frac{\left| u_i \dot{\phi}_i \right|}{u_{\max} \dot{\phi}_{\max}} \tag{2.12}$$

Bu eşitlikteki n_e eklem sayısını ifade ederken eklemlerin fiziksel sınırları temsil eden u_{max} ve $\dot{\phi}_{max}$ Tablo 2.3'teki gibi alınmıştır.



Şekil 2.44. Ödül fonksiyonunun hız hatasına ve güç tüketimine bağlı iki parçası

2.5.1. TD3 Denetleyici ile Elde Edilen Sonuçlar

Yılan robotun enerji verimli adaptif hareket problemine TD3 denetleyicisinin uygulanmasıyla elde edilen sonuçlar bu bölümde sunulmuştur. Önerilen bu yöntem, parametreleri Tablo 2.1'de verilen 5 özdeş linkli tekerleksiz bir yılan robot için zemin sürtünme katsayıları $c_t = 0.1$ ve $c_n = 10$ olan bir ortamda test edilmiştir. TD3 ajanına ve eğitime ait parametreler sırasıyla Tablo 2.17 ve Tablo 2.18'de verilmiştir. Elde edilen eğitim eğrisi ve her bölüm için adım sayısı değişimi Şekil 2.45'te verilmiştir.

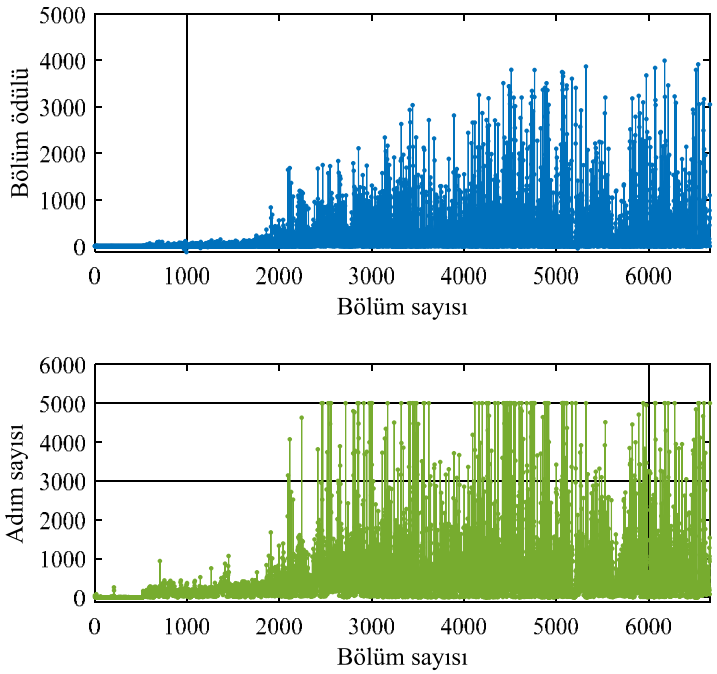
Tablo 2.17. TD3 ajanına ait parametreler

Parametreler	Değerleri
Aktör için öğrenme hızı	$\alpha_a = 1e^{-3}$
Kritik için öğrenme hızı	$\alpha_c = 1e^{-3}$
Azaltma faktörü	$\gamma = 0.99$
Bellek boyutu	1e6
Mini-paket boyutu	512
Keşif aksiyon gürültü modeli	OU
Keşif aksiyon gürültüsünün standart sapması	0.5
Keşif aksiyon gürültüsünün minimum standart sapması	$1e^{-3}$
Keşif aksiyon gürültüsünün standart sapma bozulma oranı	$2e^{-4}$
Ortalamaya çekim sabiti	1
Politika güncelleme frekansı	2
Hedef aksiyon gürültü modeli	Gauss
Hedef aksiyon gürültüsünün standart sapması	0.7
Hedef aksiyon gürültü standart sapması alt ve üst sınırları	-0.5, 0.5
Hedef aksiyon gürültüsünün standart sapma bozulma oranı	$1e^{-4}$
Hedef aksiyon gürültüsünün minimum standart sapması	0.1
Yumuşak hedef güncelleme katsayısı	$\tau = 5e^{-3}$
Hedef güncelleme frekansı	5

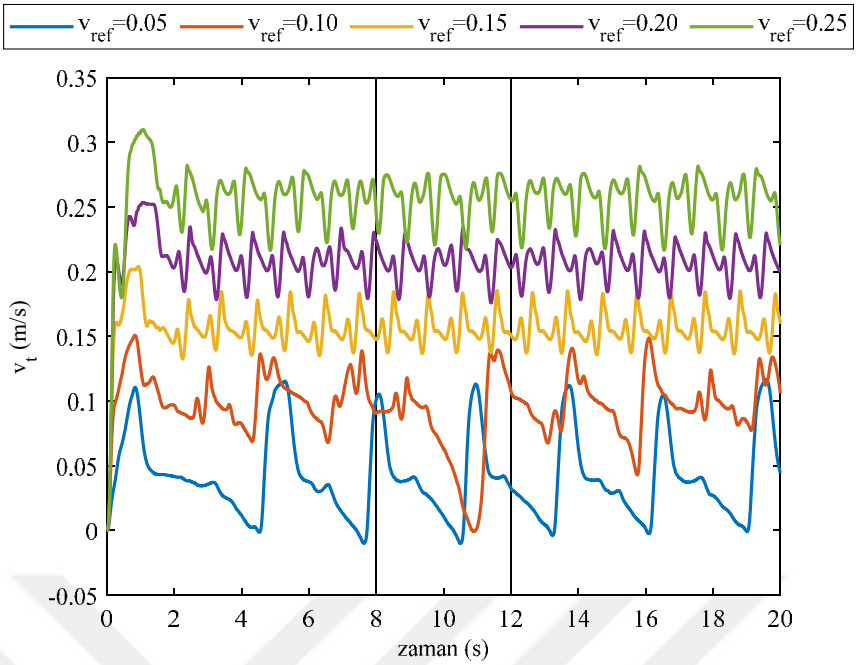
Tablo 2.18. Yılan robotun enerji verimli adaptif hareketi için kullanılan eğitim parametreleri

Parametreler	Değerleri
Örnekleme zamanı	$T_s = 0.002$s
Simülasyon zamanı	$T_f = 10$s
Maksimum adım sayısı	$T_f/T_s = 5000$
Maksimum bölüm sayısı	20000
Ortalama pencere uzunluğu	250
Eğitim durdurma kriteri	Maksimum bölüm sayısı
Ajan kaydetme kriteri	Bölüm adım sayısı
Ajan kaydetme değeri	5000

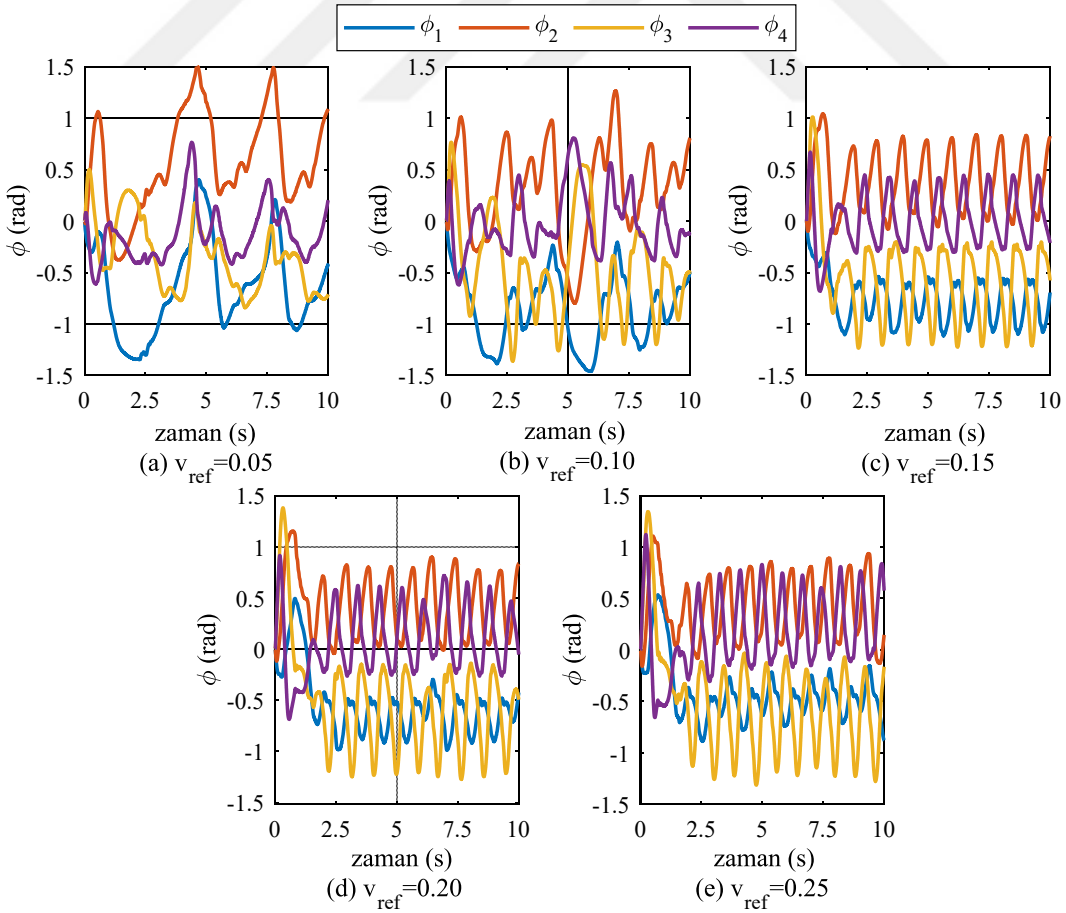
Öğrenme eğrisine bakıldığı zaman bölüm ödülünün bazı bölümlerde idealde maksimum 5000 olan değere yaklaşırken bazı bölümlerde ise daha düşük ödül değerleri aldığı ve 5000 olan maksimum adım sayısına yaklaşık 2500. bölümden sonra ulaşılabildiği görülmektedir. Bu sonuçtan ajanın bazı referans hız değerleri için ödül fonksiyonuna konulan motorun fiziksel sınırlamalarına uyacak tork değerlerini üretmede zorlandığı anlaşılmaktadır. Eğitim 6664. bölümde manuel olarak durdurulmuş ve yaklaşık 51 saat sürmüştür. Eğitim süreci boyunca toplam 2372695 veri kullanılmıştır. Eğitim sonucunda elde edilen ajanın yılan robotun hız kontrolündeki performansı Şekil 2.46'da, ürettiği yörüngeler ise Şekil 2.47'de verilmiştir. Şekil 2.46'ya bakıldığı zaman ajanın bütün referans hız değerlerini sanki referans eklem sinyali uygulanmışçasına periyodik bir profil üreterek öğrenebildiği görülmektedir. Aynı periyodik davranış Şekil 2.47'de üretilen yörüngelerde de görülmektedir. Zaten hız profili yörüngeye bağlı olarak değişmektedir. Şekil 2.48'de ise yılan robotun ağırlık merkezinin pozisyonu verilmiştir. Bu şekle göre v_{ref} =0.05 m/s iken robotun hareketi daha çok akordeon tipi yürüyüş biçimine benzerken hız arttıkça hareket de yanal dalgalanma yürüyüş biçimine benzemeye başlamıştır. Biyolojik yılanlarda en hızlı hareket biçiminin yanal dalgalanma ve en yavaş hareket biçiminin de akordeon tip yürüyüş olduğu göz önüne alındığı zaman elde edilen sonucun biyolojik yılanların hareket biçimiyle uyumlu olduğu görülmektedir.



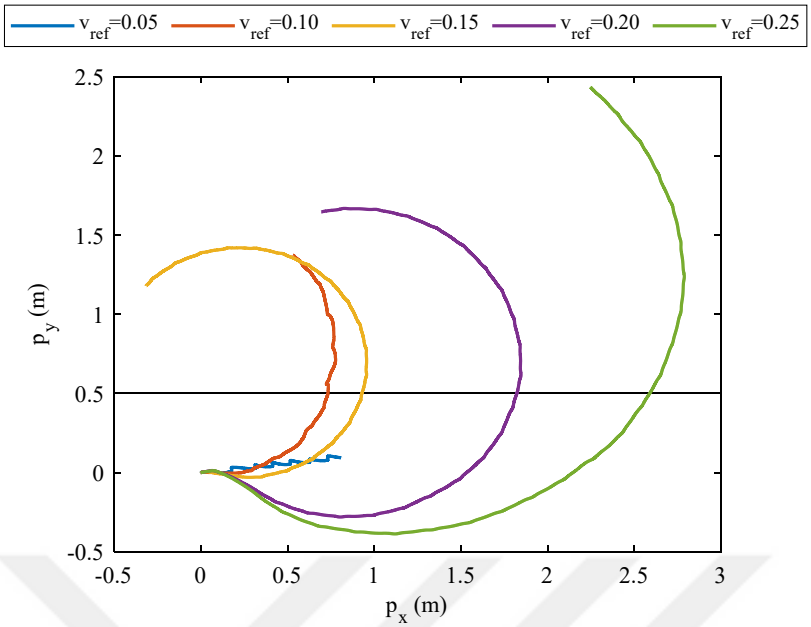
Şekil 2.45. TD3 ajanı ile yılan robotun enerji verimli hareketi için eğitim eğrisi



Şekil 2.46. TD3 denetleyici ile yılan robotun hız kontrolü

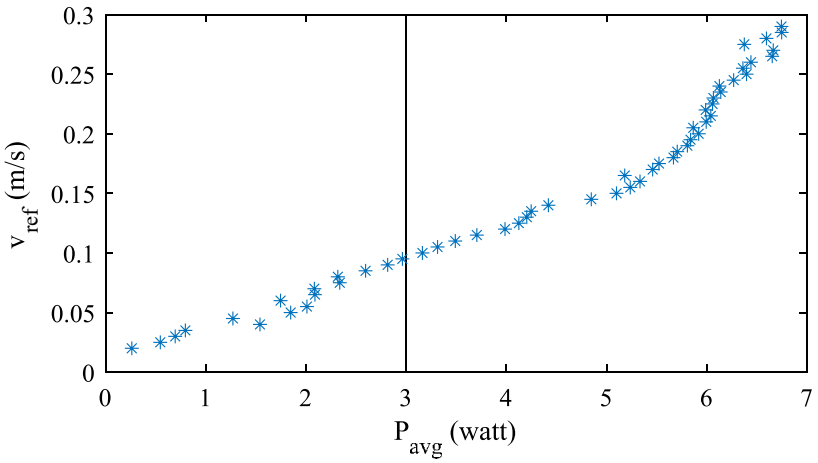


Şekil 2.47. TD3 denetleyici ile üretilen adaptif yörüngeler



Şekil 2.48. TD3 denetleyici ile yılan robotun ağırlık merkezinin pozisyonu

Eğitilen TD3 ajanının performansını değerlendirmek için [0.02, 0.29] m/s aralığında 0.005 adım büyüklüğü ile 54 tane referans hız değeri kullanılmıştır. Öğrenilen yörüngeler ile elde edilen hareket sonucunda yılan robotun referans hızı ve ortalama güç tüketimi arasındaki Pareto eğrisi Şekil 2.49'daki gibi elde edilmiştir. Bu şekilden birkaç hız değeri dışında referans hız değeri arttıkça güç tüketiminin de arttığı, yani beklenildiği gibi hız ve güç tüketimi arasında doğrusala yakın bir ilişkinin olduğu görülmektedir. Bu eğri aynı zamanda TD3 denetleyicinin farklı hızlarda adaptif yörüngeler oluşturabilmek için genelleme yapabilme yeteneğini de ortaya koymaktadır.



Şekil 2.49. TD3 denetleyici ile elde edilen Pareto optimal eğrisi

3. SONUÇLAR VE TARTIŞMA

Bu tez çalışmasında çeşitli uygulamalarda ve sektörlerde rol oynama potansiyeline sahip yılan robotların bu potansiyellerini kullanabilmelerine yönelik farklı çevre koşullarına göre enerji verimli adaptif hareket kontrolü için optimizasyon ve öğrenme tabanlı denetleyiciler geliştirilmiştir. Bu amaçla öncelikle gerçek potansiyel uygulamalar için uygun olan tekerleksiz bir yılan robot mekanizması ele alınarak dinamik modellemesi ve analizi yapılmıştır.

Yılan robotun enerji verimli hareket problemi öncelikle sabit sürtünme katsayılarına sahip bir ortamda ele alınarak optimal hareket parametreleri kararlı ve hızlı yakınsama sunan SOS algoritmasına dayanan çok amaçlı iki farklı sezgisel optimizasyon yöntemi ile elde edilmiştir. Elde edilen sonuçlar, önerilen her iki yöntemin, yılan robotun yanal dalgalanma hareketi için optimal ileri hız ile güç tüketiminin azaltılmasına ilişkin tatmin edici kararlı sonuçlar elde edebildiğini göstermektedir. Bununla birlikte, ağırlıklı toplam yöntemine dayanan MOSOS algoritması ile karşılaştırıldığında FNSMOSOS algoritmasının defalarca çalıştırılmasına gerek kalmadan tek bir çalışmada iyi bir çeşitlilik ile düzgün dağılımlı bir çözüm seti elde etmede avantaj sağladığı görülmektedir. Böylece yılan robot için en uygun çalışma stratejisi, kontrol hedefleri ve yılan robotun mevcut gücü gibi sistem gereksinimleri göz önünde bulundurularak kolayca belirlenebilir.

Yılan robotun değişen çevre koşullarındaki adaptif hareketi için ise optimal yürüyüş parametreleri sadece FNSMOSOS algoritması ile ele alınmıştır. Önerilen yöntemin performansı oldukça geniş bir sürtünme aralığına sahip farklı ortamlarda test edilmiştir. Elde edilen sonuçlardan, yılan robotun yürüyüş parametrelerini değiştirerek farklı ortamlar üzerinde enerji verimli hareketini sürdürebildiği görülmektedir. Bu çalışma, yılan robotların diğer mobil robotlara kıyasla daha az hareket verimliliğine sahip oldukları göz önüne alındığında çevresel adaptasyonunun ve enerji verimli hareketinin geliştirilmesine katkıda bulunma açısından önemlidir.

Yılan robotun enerji verimli hareketi için optimal yörünge üretmek kadar bu yörüngelerin takibi de oldukça önemlidir. Bunun için bilinmeyen düzensiz çevrelerde adaptif optimal kontrol sağlayan DRL tabanlı algoritmalar kullanılarak yılan robotun eklem yörünge kontrolü ele alınmıştır. DDPG ve TD3 algoritmaları ile bu tez çalışmasında DDPG, DDPG-$\dot{e}$, TD3 ve Transfer-TD3 olarak adlandırılan dört farklı denetleyici

geliştirilerek yörünge takip performansları birbirileriyle ve PD denetleyici ile karşılaştırılmıştır. Elde edilen sonuçlardan TD3 ajanlarının DDPG ajanlarına göre daha erken yakınsayarak daha üstün performans sundukları görülmektedir. TD3 ajanları arasındaki karşılaştırma ise benzer performanslar elde edilmesine rağmen transfer öğrenmenin orijinal TD3 ajanının yakınsama hızını ve farklı referanslar için genelleme yeteneğini iyileştirdiğini açıkça göstermektedir. Bu sonuç, DRL tabanlı denetleyicileri simülasyon ortamından gerçek dünyaya geçirmeye çalışırken transfer öğrenmenin pratik bir yol olabileceğini gösterir.

DRL tabanlı denetleyici sonuçlarıyla PD denetleyicinin sonuçları karşılaştırıldığında ise Tranfer-TD3'nin daha üstün genelleme yeteneği sunmasına karşın sonuçların birbirine yakın olduğu görülmektedir. Basit yapısı nedeniyle yılan robotların kontrolü için birçok çalışmada hala PD denetleyiciler sıklıkla tercih edilmektedir. Bu geleneksel kontrol yöntemi, temel olarak yılan robotun kontrol gereksinimlerini karşılayabilmesine rağmen, gerçek zamanlı olarak öngörülemeyen çevresel değişikliklere uyum sağlayamaz ve dış ortamın belirsizliği ile baş edemezler. Bu nedenle DRL tabanlı denetleyiciler ile yılan robotun eklem yörünge kontrolünün ele alınması robotun otonom hareketi için oldukça önemlidir.

Bu çalışmada ayrıca TD3 algoritmasının kullanıldığı DRL tabanlı bir denetleyici ile yılan robotun enerji verimli hareketi için adaptif yörüngeler üretilerek robotun hız kontrolü ele alınmıştır. Elde edilen sonuçlardan yılan robotun referans hareket hızı değiştiği zaman enerji verimli hareketi sağlayabileceği yörüngeleri kendisinin üretip öğrenebildiği görülmüştür. Bu şekilde referans yörüngeden bağımsız olarak öğrenme, yılan robotun daha doğal ve biyolojik yılanların hareket biçimine benzer bir hareket elde etmesini sağlamıştır. Ayrıca eğitilen ajanın farklı hızlar için performansı, TD3 tabanlı DRL denetleyicinin genelleştirilmiş kontrol stratejilerini keşfetmedeki kararlılığını gösterir.

4. ÖNERİLER

Tekerleksiz bir yılan robotun adaptif hareket kontrolü bu tez kapsamında ele alınmıştır. DRL tabanlı bir denetleyici ile yılan robotun enerji verimli hareketinin ele alındığı bölümde elde edilen öğrenme eğrisinden ajanın bazı referans hız değerleri için ödül fonksiyonuna konulan motorun fiziksel kısıtlarına uyacak tork değerlerini üretmede zorlandığı görülmüştür. Bu nedenle gelecek çalışmalarda kısıtlamalı RL yöntemi kullanılarak daha kararlı ve verimli bir öğrenme gerçekleştirilebilir. Bunun için öncelikle rastgele harici olarak üretilen aksiyonlar DRL ajanı üzerinden çevreye uygulanarak gözlem, aksiyon ve ödül eğitim verileri toplanır. Daha sonra toplanan bu veriler ile sistemin durum değişkenlerine bağlı olan kısıt fonksiyonunun katsayıları DNN gibi lineer olmayan bir kestirim yöntemiyle öğrenildikten sonra karesel programlama tekniği ile kısıtları sağlayan optimal kontrol işareti üretilebilir.

Bu çalışmada yapılan transfer öğrenmede önceden eğitilen ajanın sadece parametreleri aktarılmıştır. Ajanın parametreleriyle birlikte deneyim belleğinin de transfer edilerek öğrenme eğrisine ve denetleyicilerin performansına etkisi analiz edilebilir. Bunun için kaydedilen ajanların deneyim belleklerinin de kaydedilmesi gerekir ki bu işlem büyük miktarda bellek kaynağı gerektirir.

Sürekli aksiyon uzayında öğrenmenin en büyük zorluğu keşiftir. Yılan robotun eğitimi süresince referans ve çevre değiştikçe ajan yeni stratejiler keşfetmeye ihtiyaç duyar. Bu nedenle DRL algoritmalarında kullanılan OU ve Gauss gürültü aksiyon modellerine alternatif yeni bir gürültü aksiyon modeli olarak bazı hayvanların yiyecek aramak için izledikleri stratejilerden biri olan Lévy uçuşunun etkinliği incelenebilir. Bu yöntemde adım büyüklükleri sürekli bir olasılık dağılımı olan Lévy dağılımına göre belirlenmektedir. Bu stratejiye göre arama uzayının küçük bir alanında keşif yapıldıktan sonra gelişigüzel belirlenen bir yön seçilerek büyük ölçekli hareketlerle başka bir alanda keşif yapılır. Guguk kuşu optimizasyon algoritmasında kullanılan bu dağılım algoritmanın global optimum değere daha hızlı yakınsamasını sağladığı için DRL algoritmalarında da aksiyon keşfini verimli kılarak eğitim hızına ve yakınsamasına olumlu etki edeceği düşünülmektedir.

Eğitim hızını ve öğrenme kararlılığını arttırmak için klasik kontrol yöntemleriyle eğitimin başlangıcında ajanın daha iyi veriler üretmesini sağlayarak öğrenmesinin yönlendirilmesi ve elde edindiği bilgiyi daha sonra kendisinin geliştirilip genelleştirilmesi ele alınabilir.

5. KAYNAKLAR

Abdullahi, M., Ngadi, M.A., Dishing, S. I. ve Usman, M. J., 2019 A survey of symbiotic organisms search algorithms and applications, Neural Computing and Applications, 1-20.

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaria, J., Fadhel, M.A., Al-Amidie, M., ve Farhan, L., 2021. Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions, Journal of big Data, 8, 1, 1-74.

Ariizumi, R. ve Matsuno, F., 2017. Dynamic analysis of three snake robot gaits, IEEE Transaction on Robotics, 33, 5, 1075-1087.

Bellman, R. A., 1957. Markovian decision process, Journal of mathematics and mechanics, 679-684.

Bing, Z., Cheng, L., Huang, K., Zhou, M. ve Knoll, A., 2017. CPG-based control of smooth transition for body shape and locomotion speed of a snake-like robot, IEEE International Conference on Robotics and Automation, 4146-4153.

Bing, Z., Cheng, L., Chen, G., Röhrbein, F., Huang, K. ve Knoll, A., 2017. Towards autonomous locomotion: CPG-based control of smooth 3D slithering gait transition of a snake-like robot, Bioinspiration and biomimetics, 12, 3, 035001.

Bing, Z., Lemke, C., Jiang, Z., Huang, K. ve Knoll, A., 2019. Energy-efficient slithering gait exploration for a snake-like robot based on reinforcement learning, Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence.

Bing, Z., Lemke, C., Jiang, Z., Huang, K., ve Knoll, A., 2020a. Energy-efficient and damage-recovery slithering gait design for a snake-like robot based on reinforcement learning and inverse reinforcement learning, Neural Networks, 129, 323-333.

Bing, Z., Lemke, C., Morin, F. O., Jiang, Z., Cheng, L., Huang, K. ve Knoll, A., 2020b. Perception-action coupling target tracking control for a snake robot via reinforcement learning, Frontiers in Neurorobotics, 14, 591128.

Bloch, A.M. Baillieul, J., Crouch, P. ve Marsden, J., 2003. Nonholonomic Mechanics and Control, Springer-Verlag.

Bogue, R., 2014. Snake robots: a review of research, products and applications. Industrial Robot: An International Journal, 41, 3, 253-258.

Bottou, L., 2010. Large-scale machine learning with stochastic gradient descent, Proceedings of COMPSTAT'2010, 177-186, Physica-Verlag HD.

Cao, Z., Zhang, D., Hu, B. ve Liu, J., 2019. Adaptive path following and locomotion optimization of snake-like robot controlled by the central pattern generator, Complexity.

Cheng, M.Y. ve Prayogo, D., 2014. Symbiotic Organisms Search: A new metaheuristic optimization algorithm, Computers and Structures, 139, 98-112.

Chernousko, F., 2005. Modelling of snake-like locomotion, Applied Mathematics and Computation, 164, 2, 415-434.

Coello, C. A. C., 1999. A comprehensive survey of evolutionary-based multiobjective optimization techniques, Knowledge and Information systems, 1, 3, 269-308.

Crespi, A. ve Ijspeert, A.J., 2006. Amphibot II: an amphibious snake robot that crawls and swims using a central pattern generator, International Conference on Climbing and Walking Robots, 19-27.

Crespi, A., ve Ijspeert, A.J., 2008. Online optimization of swimming and crawling in an amphibious snake robot, IEEE Transactions on Robotics, 24, 1, 75-87.

Dahl, G. E., Yu, D., Deng, L., ve Acero, A., 2011. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition, IEEE Transactions on audio, speech, and language processing, 20, 1, 30-42.

Deb, K., Pratap, A., Agarwal, S. ve Meyarivan, T., 2002. A fast and elitist multi objective genetic algorithm: NSGA-II, IEEE Transaction on Evolutionary Computation, 6, 2, 182-197.

Deng, L. ve Yu, D., 2014. Deep learning: methods and applications. Foundations and trends in signal processing, 7, 3-4, 197-387.

Deng, Y., Bao, F., Kong, Y., Ren, Z., ve Dai, Q.. 2017. Deep direct reinforcement learning for financial signal representation and trading, IEEE transactions on neural networks and learning systems, 28, 3, 653-664.

Dooraki, A. R. ve Lee, D. J., 2021. An innovative bio-inspired flight controller for quad-rotor drones: Quad-rotor drone learning to fly using reinforcement learning, Robotics and Autonomous Systems, 135, 103671.

Dosoglu, M. K., Guvenc, U., Duman, S., Sonmez, Y. ve Kahraman, H. T. 2018. Symbiotic organisms search optimization algorithm for economic/emission dispatch problem in power systems, Neural Computing and Applications, 29, 3, 721-737

Duchi, J., Hazan, E. ve Singer, Y., 2011. Adaptive subgradient methods for online learning and stochastic optimization, Journal of machine learning research, 12, 7, 2121-2159.

Duman, S., 2016. Symbiotic organisms search algorithm for optimal power flow problem based on valve-point effect and prohibited zones, Neural Computing and Applications, 28, 11, 3571-3585.

François-Lavet, V., 2017. Contributions to deep reinforcement learning and its applications in smartgrids, Ph.D thesis, University of Liege, Belgium.

François-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G., ve Pineau, J., 2018. An introduction to deep reinforcement learning, Foundations and Trends in Machine Learning, 11, 3-4, 219-354.

Fujimoto, S., Hoof, H., and Meger, D., 2018. Addressing function approximation error in actor-critic methods, In International Conference on Machine Learning, 1587-1596.

Gandhi, D., Pinto, L. ve Gupta, A., 2017. Learning to Fly by Crashing, arXiv preprint arXiv:1704.05588.

Geman, S., Bienenstock, E. Doursat, R., 1992. Neural networks and the bias/variance dilemma, Neural Computing, 4, 1-58.

Glatt, R., d. Silva F.L. ve Costa, A.H.R., 2016. Towards knowledge transfer in deep reinforcement learning, 5th Brazilian Conference on Intelligent Systems (BRACIS), 91-96.

Gray, J., 1946. The mechanism of locomotion in snakes. Journal of experimental biology, 23, 2, 101-120.

Han, C., Zhou, G. ve Zhou, Y., 2019. Binary Symbiotic Organism Search Algorithm for Feature Selection and Analysis, IEEE Access, 7, 166833-166859.

Hasanzadeh, S. ve Tootoonchi, A. A., 2008. Adaptive optimal locomotion of snake robot based on CPG-network using fuzzy logic tuner, IEEE Conference on Robotics, Automation and Mechatronics, 187-192.

Hasanzadeh, S. ve Tootoonchi, A.A., 2010. Ground adaptive and optimized locomotion of snake robot moving with a novel gait, Autonomous Robots, 28, 4, 457-470.

Hicks, G.P., 2003. Modeling and control of a snake-like serial-link structure, Ph.D. Dissertation, North Carolina State University.

Hinton, G., Srivastava, N. ve Swersky, K., 2012. Neural networks for machine learning, Coursera, video lectures, 264, 1, 2146-2153.

Hirose, S., 1993. Biologically Inspired Robots: Snake-Like Locomotors and Manipulators, Oxford University Press, Oxford.

Hou, X., Shi, Y., Li, L., Tian, Y., Su, Y., Ding, T. ve Deng, Z., 2020. Revealing the mechanical characteristics via kinematic wave model for snake-like robot executing exploration of lunar craters, IEEE Access, 8, 38368-38379.

Hu, D., Nirody, J., Scott, T. ve Shelley, M., 2009. The mechanics of slithering locomotion. Proceedings of the National Academy of Sciences, 106, 25, 10081-10085.

Huang, C. W., Huang, C. H., Hung, Y. H., ve Chang, C. Y., 2018. Sensing pipes of a nuclear power mechanism using low-cost snake robot, Advances in Mechanical Engineering, 10, 6, 1687814018781286.

Inoue, K., Sumi, T. ve Ma, S., 2007. CPG-based control of a simulated snake-like robot adaptable to changing ground friction, IEEE/RSJ International Conference on Intelligent Robots and Systems, 1957-1962.

Ioffe, S. ve Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.03167.

Ji, Q., Fu, S., Tan, K., Muralidharan, S. T., Lagrelius, K., Danelia, D., Andrikopoulos, G., Wang, X.V., Wang, L. ve Feng, L., 2022. Synthesizing the optimal gait of a quadruped robot with soft actuators using deep reinforcement learning, Robotics and Computer-Integrated Manufacturing, 78, 102382.

Kamegawa, T., Akiyama, T., Sakai, S., Fujii, K., Une, K., Ou, E., Matsumura, Y., Kishutani, T., Nose, E., Yoshizaki, Y. ve Gofuku, A., 2020. Development of a separable search-and-rescue robot composed of a mobile robot and a snake robot, Advanced Robotics, 34, 2, 132-139.

Kane, T. ve Lecison, D., 2000. Locomotion of snakes: a mechanical 'explanation', International Journal of Solids and Structures, 37, 41, 5829-5837.

Kelly, S., ve Murray, R.M., 1995. Geometric phases and robotic locomotion, Journal of Robotic Systems,12, 6, 417-431.

Kelasidi, E., Pettersen, K.Y. ve Gravdahl, J.T., 2015. Energy efficiency of underwater snake robot locomotion, IEEE 23rd Mediterranean Conference on Control and Automation (MED), 1124-1131.

Kelasidi, E., Jesmani, M., Pettersen, K.Y. ve Gravdahl, J.T., 2016. Multi-objective optimization for efficient motion of underwater snake robots, Artificial Life and Robotics, 21, 4, 411-422.

Kelasidi, E., Jesmani, M., Pettersen, K.Y. ve Gravdahl, J.T., 2018. Locomotion efficiency optimization of biologically inspired snake robots, Applied Sciences, 18, 1, 80.

Kelasidi, E., Moe, S., Pettersen, K.Y., Kohl, A.M., Liljeback, P. ve Gravdahl, J.T., 2019. Path following, obstacle detection and obstacle avoidance for thrusted underwater snake robots, Frontiers in Robotics and AI, 6, 57.

Kimura, H. ve Hirose, S., 2002. Development of genbu: active wheel passive joint articulated mobile robot, IEEE/RSJ International Conference Intelligent Robots and Systems, 823-828.

Kingma, D. P. ve Ba, J., 2014. Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

Koga, M.L., Freire, V. ve Costa, A. H., 2015. Stochastic Abstract Policies: Generalizing knowledge to improve Reinforcement Learning, IEEE Transactions on Cybernetics, 45, 1, 77-88.

Konidaris, G., Scheidwasser, I., ve Barto, A. G., 2012. Transfer in Reinforcement Learning via shared features, Journal of Machine Learning Research (JMLR), 13, 1, 1333-1371.

Kormushev, P., Ugurlu, B., Caldwell, D. G. ve Tsagarakis, N. G., 2019. Learning to exploit passive compliance for energy-efficient gait generation on a compliant humanoid, Autonomous Robots, 43, 1, 79-95.

Kousuke, I., Takaaki, S. ve Ma, S, 2007. CPG-based control of a simulated snake-like robot adaptable to changing ground friction, IEEE/RSJ International Conference on Intelligent Robots and Systems, 1957-1962.

Krishnaprasad, P. ve Tsakiris, D., 1994. G-snakes: nonholonomic kinematic chains on lie groups, 33rd IEEE Conf. Decision and Control, Lake Buena Vista, FL USA, 2955-2960.

Krizhevsky, A., Sutskever, I., ve Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks, Advances in neural information processing systems, 1097-1105.

Levine, S., Finn, C., Darrell, T. ve Abbeel, P., 2016. End-to-end training of deep visuomotor policies, Journal of Machine Learning Research, 17, 39, 1-40.

Li, B., Yang, Z. P., Chen, D. Q., Liang, S. Y. ve Ma, H., 2021. Maneuvering target tracking of UAV based on MN-DDPG and transfer learning, Defence Technology, 17, 2, 457-466.

Liljeback, P., Stavdahl, O. ve Beitnes, A., 2006. SnakeFighter-development of a water hydraulic firefighting snake robot, 9th IEEE International Conference on Control, Automation, Robotics and Vision, 1-6.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Siver, D. ve Wierstra, D., 2016. Continuous control with deep reinforcement learning, arXiv preprint arXiv:1509.02971.

Liljebäck, P., Pettersen, K. Y., Stavdahl, Ø. ve Gravdahl, J. T., 2012a. A review on modelling, implementation, and control of snake robots, Robotics and Autonomous systems, 60, 1, 29-40.

Liljebäck, P., Pettersen, K. Y., Stavdahl, Ø., ve Gravdahl, J. T., 2012b. Snake robots: Modelling, Mechatronics, and Control, Springer Science & Business Media.

Lin, L.J., 1993. Reinforcement learning for robots using neural networks, Technical report, DTIC Document.

Liu, J., Tong, Y. ve Liu, J., 2021. Review of snake robots in constrained environments. Robotics and Autonomous Systems, 141, 103785.

Liu, Y., ve Farimani, A. B., 2021. An Energy-Saving Snake Locomotion Gait Policy Obtained Using Deep Reinforcement Learning, arXiv preprint arXiv:2103.04511.

Ma, S., 2001. Analysis of creeping locomotion of a snake-like robot, Advanced Robotics, 15, 2, 205-224.

Ma, S., 2004. Ohmameuda, Y. and Inoue, K., Dynamic analysis of 3-dimensional snake robots, IEEE/RSJ International Conference Intelligent Robots and Systems, 767-772.

Malayjerdi, M. ve Akbarzadeh Tootoonchi, A., 2019. Analytical modeling of a 3-D snake robot based on sidewinding, International Journal of Dynamics and Control, 7, 83-93.

Martinsen, A. B., ve Lekkas, A. M., 2018. Curved path following with deep reinforcement learning: Results from three vessel models, In OCEANS 2018 MTS/IEEE Charleston, 1-8.

MathWorks R2022a. MATLAB: rlDDPGAgentOptions. URL-1: https://www.mathworks.com/help/reinforcement-learning/ref/rlddpgagentoptions.html 4 Temmuz 2022

Matsuo, T., Sonoda, T. ve Ishii, K., 2012. A Design Method of CPG Network Using Energy Efficiency to Control a Snake-Like Robot, Fifth International Conference on In Emerging Trends in Engineering and Technology (ICETET), 287-292.

Mattison, C., 2002. The Encyclopaedia of Snakes. Cassell Paperbacks, London.

Mehta, V., Brennan, S. ve Gandhi, F., 2008. Experimentally verified optimal serpentine gait and hyperredundancy of a rigid-link snake robot, IEEE Transactions on Robotics, 24, 2, 348-360.

Miller, G., 2002. Snake robots for search and rescue, in: Neurotechnology for Biomimetic Robots, MIT Press, Cambridge, MA, USA, London, 271-284.

Mori, M. ve Hirose, S., 2002. Three-dimensional serpentine motion and lateral rolling by active cord mechanism ACM-R3, IEEE/RSJ International Conference Intelligent Robots and Systems, 829-831.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., ve Riedmiller, M., 2013. Playing atari with deep reinforcement learning, arXiv preprint arXiv:1312.5602.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... ve Petersen, S., 2015. Human-level control through deep reinforcement learning, Nature, 518,7540, 529-533.

Nanda, S. J. ve Jonwal, N., 2017. Robust nonlinear channel equalization using WNN trained by symbiotic organism search algorithm. Applied Soft Computing, 57, 197-209.

Nilsson, M., 2004. Serpentine locomotion on surfaces with uniform friction, IEEE/RSJ International Conference Intelligent Robots and Systems, 1751-1755.

Nissen, S., 2007. Large scale reinforcement learning using Q-sarsa (λ) and cascading neural networks. M.Sc. Thesis, Department of Computer Science, University of Copenhagen, København, Denmark.

Nor, N. M. ve Ma, S. 2014a. A simplified CPGs network with phase oscillator model for locomotion control of a snake-like robot, Journal of Intelligent and Robotic Systems, 75,1, 71-86.

Nor, N. M. ve Ma, S. 2014b. CPG-based locomotion control of a snake-like robot for obstacle avoidance, IEEE International Conference on Robotics and Automation, 347-352.

Nor, N. M. ve Ma, S. 2014c. CPG-based straight path control of a snake-like robot for moving in various space width, World Congress on Intelligent Control and Automation, 2233-2238.

Ohno, H., ve Hirose, S., 2001, November. Design of slim slime robot and its gait of locomotion, IEEE/RSJ International Conference Intelligent Robots and Systems, 707-715.

Ostrowski, J.P., 1996. The mechanics and control of undulatory robotic locomotion, Ph.D. Dissertation, California Institute of Technology.

Ostrowski, J. ve Burdick, J., 1998. The geometric mechanics of undulatory robotic locomotion, International Journal of Robotics Research, 17, 7, 683-701.

Ouyang, W., Chi, H., Pang, J., Liang, W. ve Ren, Q., 2021. Adaptive locomotion control of a hexapod robot via bio-inspired learning, Frontiers in Neurorobotics, 15, 627157.

Pan, X., You, Y., Wang, Z. ve Lu., C., 2017. Virtual to Real Reinforcement Learning for Autonomous Driving, arXiv preprint arXiv:1704.03952.

Parsopoulos, K. ve Vrahatis, M., 2002. Recent approaches to global optimization problems through particle swarm optimization, Natural Computing, 1, 2, 235-306.

Peng, X. B., Berseth, G., Yin, K. ve Van De Panne, M., 2017. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning, ACM Transactions on Graphics (TOG), 36, 4, 41.

Pinto, L., Andrychowicz, M., Welinder, P., Zaremba, W. ve Abbeel., P., 2017. Asymmetric Actor Critic for Image-Based Robot Learning, arXiv preprint arXiv:1710.06542.

Plappert, M., Houthooft, R., Dhariwal, P., Sidor, S., Chen, R. Y., Chen, X., ... ve Andrychowicz, M., 2017. Parameter space noise for exploration, arXiv preprint arXiv:1706.01905.

Prautsch, P. ve Mita, T., 1999. Control and analysis of the gait of snake robots, IEEE International Conference Control Applications, Kohala Coast, HI USA, 502-507.

Rajeswaran, A., Kumar, V., Gupta, A., Vezzani, G., Schulman, J., Todorov, E., ve Levine, S., 2017. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations, arXiv preprint arXiv:1709.10087.

Rubí, B., Morcego, B. ve Pérez, R., 2021. Deep reinforcement learning for quadrotor path following with adaptive velocity, Autonomous Robots, 45, 1, 119-134.

Rummery, G. A. ve Niranjan, M., 1994. On-line Q-learning using connectionist systems. Technical Report CUED/F-INFENG/TR 166, Engineering Department, Cambridge University.

Ryu, J.K., Chong, N.Y., You, B.J. ve Christensen, H.I., 2010. Locomotion of snake-like robots using adaptive neural oscillators, Intelligent Service Robotics, 3,1, 1-10.

Saito, M., Fukaya, M. ve Iwasaki, T., 2002. Serpentine locomotion with robotic snakes, IEEE Control Systems Magazine, 22, 1, 64-81.

Sarker, I.H., 2021a. Machine learning: Algorithms, real-world applications and research directions, SN Computer Science, 2, 3, 1-21.

Sarker, I.H., 2021b. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions, SN Computer Science, 2, 6, 1-20.

Sarrafan, S., Akbarzadeh, A., Molavipoor, S. ve Arhami M., 2011. Kinematics and motion analysis of a three-dimensional sidewinding snake-like robot, National Conference on Mechanical Engineering, 1-8.

Schulman, J., Levine, S., Abbeel, P., Jordan, M. ve Moritz, P., 2015. Trust region policy optimization, In International conference on machine learning, 1889-1897.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A. ve Klimov, O., 2017. Proximal policy optimization algorithms, arXiv preprint arXiv:1707.06347.

Seetohul, J. ve Shafiee, M., 2022. Snake Robots for Surgical Applications: A Review, Robotics, 11, 3, 57.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D. ve Riedmiller, M., 2014. Deterministic policy gradient algorithms. In International conference on machine learning, 387-395.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... ve Dieleman, S, 2016. Mastering the game of Go with deep neural networks and tree search, Nature, 529, 7587, 484-489.

Srivastava, N., G. E. Hinton, A. Krizhevsky, I. Sutskever, ve R. Salakhutdinov, 2014. Dropout: a simple way to prevent neural networks from overfitting, Journal of Machine Learning Research, 15, 1, 1929-1958.

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences, Machine learning, 3, 1, 9-44.

Sutton, R. S. ve Barto, A. G., 2018. Reinforcement learning: An introduction, MIT press.

Szegedy, C., Ioffe, S., Vanhoucke, V. ve Alemi., A., 2016. Inception-v4, inception-resnet and the impact of residual connections on learning, arXiv preprint arXiv:1602.07261.

Qian, N., 1999. On the momentum term in gradient descent learning algorithms, Neural networks, 12, 1, 145-151.

Tang, C. ve Ma, S., 2011. A self-tuning multi-phase CPG enabling the snake robot to adapt to environments, IEEE/RSJ International Conference on Intelligent Robots and Systems, 1869-1874.

Torrey, L. ve Shavlik, J., 2010. Transfer learning, Handbook of research on machine learning applications and trends: Algorithms, methods, and techniques: Algorithms, methods, and techniques, IGI global, 242-264.

Tran, D.H., Cheng, M.Y., Prayogo, D., 2016. A novel Multiple Objective Symbiotic Organisms Search (MOSOS) for time–cost–labor utilization tradeoff problem, Knowledge-Based Systems, 94, 132-145.

Transeth, A. A., 2007. Modelling and control of snake robots. Ph.D. Dissertation, Norwegian University of Science and Technology.

Transeth, A. A., Leine, R. I., Glocker, C., ve Pettersen, K. Y., 2008. 3-D snake robot motion: nonsmooth modeling, simulations, and experiments, IEEE Transactions on Robotics, 24, 2, 361-376.

Uhlenbeck, G. E. ve Ornstein, L.S, 1930. On the theory of the Brownian motion, Physical review, 36, 5,823.

Van Hasselt, H., 2010. Double q-learning, Advances in Neural Information Processing Systems, 2613-2621.

Van Hasselt, H., Guez, A., ve Silver, D., 2016. Deep reinforcement learning with double q-learning, In Proceedings of the AAAI conference on artificial intelligence, 30,1, 2094-2100.

Vincent, F.Y., Redi, A.P., Yang, C.L., Ruskartina, E. ve Santosa, B., 2016. Symbiotic organism search and two solution representations for solving the capacitated vehicle routing problem, Applied Soft Computing, 52, 657-672.

Virgala, I., Kelemen, M., Božek, P., Bobovský, Z., Hagara, M., Prada, E., Oščádal, P. ve Varga, M., 2020. Investigation of snake robot locomotion possibilities in a pipe, Symmetry, 12, 6, 939.

Vossoughi, G., Pendar, H., Heidari, Z., ve Mohammadi, S., 2008. Assisted passive snake-like robots: conception and dynamic modeling using Gibbs-Appell method, Robotica, 26, 3, 267-276.

Walton, M., Jayne, B. C., ve Bennet, A. F., 1990. The energetic cost of limbless locomotion. Science, 249, 496, 524-527.

Wang, W., Gu, D. ve Xie, G., 2017. Autonomous Optimization of Swimming Gait in a Fish Robot With Multiple Onboard Sensors, IEEE Transactions on Systems, Man, and Cybernetics Systems, 1-13.

Watkins, C.J.C.H., 1989. Learning from Delayed Rewards, Ph.D. thesis, Cambridge University.

Wiens, A. ve Nahon, M., 2012. Optimally efficient swimming in hyper-redundant mechanisms: Control, design, and energy recovery, Bioinspiration and Biomimetics, 7, 4, 046016.

Williams, R. J., 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning, Machine learning, 8, 3, 229-256.

Wolf, A., Choset, H. H., Brown, B. H., ve Casciola, R. W., 2005. Design and control of a mobile hyper-redundant urban search and rescue robot. Advanced Robotics, 19, 3, 221-248.

Wright, C., Johnson, A., Peck, A., McCord, Z., Naaktgeboren, A., Gianfortoni, P., Gonzalez-Rivero, M., Hatton, R. ve Choset, H., 2007. Design of a modular

snake robot, IEEE/RSJ International Conference Intelligent Robots and Systems, 2609-2614.

Wu, X. ve Ma, S., 2010a. CPG-based control of serpentine locomotion of a snake-like robot, Mechatronics, 20, 2 326-334.

Wu, X. ve Ma, S., 2010b. Adaptive creeping locomotion of a CPG-controlled snake-like robot to environment change, Autonomous Robots, 28, 3, 283-294.

Yang, X., Zheng, L., Lü, D., Wang, J., Wang, S., Su, H., Wang, Z, Ren, L., 2022. The snake-inspired robots: a review, Assembly Automation, 42, 4, 567-583.

Xue, B., Zhang, M., Browne, W. N. ve Yao, X., 2016. A survey on evolutionary computation approaches to feature selection, IEEE Transactions on Evolutionary Computation, 20 ,4, 606-626.

Yamada, H., Chigisaki, S., Mori, M., Takita, K., Ogami, K. ve Hirose, S., 2005. Development of amphibious snake-like robot ACM-R5, Proceedings of the 36th International Symposium on Robotics.

Yamada, H. ve Hirose, S., 2006. Development of practical 3-dimensional active cord mechanism ACM-R4, Journal of Robotics and Mechatronics, 18, 3, 1-7.

Yamada, H. ve Hirose, S., 2008. Study of active cord mechanism-generalized basic equations of the locomotive dynamics of ACM and analysis of sinus lifting, Journal of the Robotics Society of Japan, 26, 7, 801-811.

Yamada, H. ve Hirose, S., 2009. Study of a 2-DOF joint for the small active cord mechanism, IEEE International Conference Robotics and Automation, 3827-3832.

Yu, J., Tan, M., Chen, J. ve Zhang, J., 2014. A survey on CPG-inspired control models and system implementation, IEEE transactions on neural networks and learning systems, 25, 3, 441-456.

Yu, W., Turk, G., ve Liu, C. K., 2018. Learning symmetric and low-energy locomotion, ACM Transactions on Graphics (TOG), 37, 4, 144.

Zeiler, M. D., 2012. Adadelta: an adaptive learning rate method, arXiv preprint arXiv:1212.5701.

Zhou, Y., Miao, F. ve Luo, Q., 2019. Symbiotic organisms search algorithm for optimalevolutionary controller tuning of fractional fuzzy controllers. Applied Soft Computing, 77, 497-508.

Zhu, W. ve Rosendo, A., 2022. PSTO: Learning Energy-Efficient Locomotion for Quadruped Robots, Machines, 10, 3, 185.

ÖZGEÇMİŞ

Yeşim Aysel BAYSAL, İlkokul, ortaokul ve lise öğrenimini Diyarbakır'da tamamladı. 2007 yılında başladığı Karadeniz Teknik Üniversitesi, Mühendislik Fakültesi, Elektrik–Elektronik Mühendisliği Bölümü'nden 2012 yılında Elektrik-Elektronik Mühendisi unvanı ile bölüm ikincisi olarak mezun oldu. Aynı sene Karadeniz Teknik Üniversitesi, Elektrik-Elektronik Mühendisliği Anabilim Dalı'nda başladığı yüksek lisans eğitimini 2016 yılında tamamlayarak aynı anabilim dalında doktora eğitimine başladı. Doktora çalışmaları sırasında 2017-2018 Bahar yarıyılında İstanbul Teknik Üniversitesi, Kontrol ve Otomasyon Mühendisliği'nde, 2019 yılında ise Aalborg Üniversitesi, Elektronik Sistemler Bölümü, Otomasyon ve Kontrol Anabilim Dalı'nda bulundu. 2012 yılında Karadeniz Teknik Üniversitesi, Elektrik-Elektronik Mühendisliği, Kontrol ve Kumanda Anabilim Dalı'nda Araştırma Görevlisi olarak başladığı görevine halen devam etmektedir. Yabancı dil olarak İngilizce ve Arapça bilmektedir.

SCI/SCI-E indekslerine giren dergilerde yayınlanan makaleler

Baysal, Y.A. ve Altas, I. H., A, 2022. Fast Non-dominated Sorting Multi-objective Symbiotic Organism Search Algorithm for Energy Efficient Locomotion of Snake Robot, Computer Science and Information Systems, 19, 1, 353-378.

Baysal, Y.A., Ketenci, S., Altas, I. H. ve Kayıkçıoğlu, T, 2021. Multi-objective Symbiotic Organism Search Algorithm for Optimal Feature Selection in Brain Computer Interfaces, Expert Systems with Applications, 165, 113907.

Diğer dergilerde yayınlanan makaleler

Baysal, Y.A. ve Altas, I. H., A, 2017. Power Quality Improvement via Optimal Capacitor Placement in Electrical Distribution Systems using Symbiotic Organisms Search Algorithm, Mugla Journal of Science and Technology, 3, 1, 64-68.

Hakemli konferans/sempozyumların bildiri kitaplarında yer alan yayınlar

Baysal, Y.A. ve Altas, I. H., 2020, October. Adaptive Snake Robot Locomotion in Different Environments, International Conference on Control, Automation and Diagnosis (ICCAD), 1-6, IEEE.

Akpolat, A. N., Baysal, Y. A., Yang, Y. ve Blaabjerg, F., 2020, October. Optimal PV generation using symbiotic organisms search optimization algorithm-based MPPT, In IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society, 2850-2855, IEEE.

Baysal, Y.A. ve Altas, I. H., A, 2020, August. Optimally Efficient Locomotion of Snake Robot, International Conference on INnovations in Intelligent SysTems and Applications (INISTA), 1-6, IEEE.

Baysal, Y. A. ve Altas, I. H., 2020, April. Modelling and simulation of a wheel-less snake robot. 7th International Conference on Electrical and Electronics Engineering (ICEEE), 285-289, IEEE.

Baysal Y. A., Altaş İ. H. ve Gedikli E., 2016, Ekim. Simbiyotik Organizmalar Arama Algoritması ile Dağıtım Şebekelerinde Güç Kayıplarının Azaltılması, Akıllı Sistemlerde Yenilikler ve Uygulamaları (ASYU) Sempozyumu, Bildiriler Kitabı: 295-299.

Baysal, Y. A. ve Altas, I. H., 2016, August. Cuckoo Search Algorithm for Power Loss Minimization by Optimal Capacitor Allocation in Radial Power Systems. International Symposium on INnovations in Intelligent SysTems and Applications (INISTA), 1-5, IEEE.

Baysal, Y. A. ve Altas, I. H., 2015, September. A Fuzzy Reasoning Approach for Optimal Location and Sizing of Shunt Capacitors in Radial Power Systems. IEEE Energy Conversion Congress and Exposition (ECCE), 5838-5842, IEEE.

Ödüller

- TÜBİTAK BİDEB 2211-C Yurt İçi Öncelikli Alanlar Doktora Bursu, 2018-2021.
- TÜBİTAK BİDEB 2224-A Yurt Dışı Bilimsel Etkinliklere Katılım Desteği, 2016.