GU J Sci, Part C, 9(2): 200-210 (2021)

Gazi University

Journal of Science

PART C: DESIGN AND TECHNOLOGY

http://dergipark.gov.tr/gujsc

Investigation of the Standard Deviation of Ornstein - Uhlenbeck Noise in the DDPG Algorithm

Mustafa Can BINGOL

Fırat University, Faculty of Technology, Department of Mechatronics Engineering, 23160, Merkez/ELAZIG

Article Info:

Research article
Received:01/02/2021
Revision:17/04/2021
Accepted:30/04/2021

Highlights

- *Reinforcement learning*
- *Hyperparameter choosing*

Keywords

Deep deterministic policy gradient (DDPG)
Ornstein–Uhlenbeck noise
Reinforcement learning
Inverse kinematics
Standard deviation range

Graphical/Tabular Abstract

In this study, the Standard Deviation of Ornstein - Uhlenbeck Noise Found in the deep deterministic policy gradient (DDPG) method, which is one of the reinforcement learning (RL), was investigated. For this purpose, system block diagram (seen Figure A) was used.

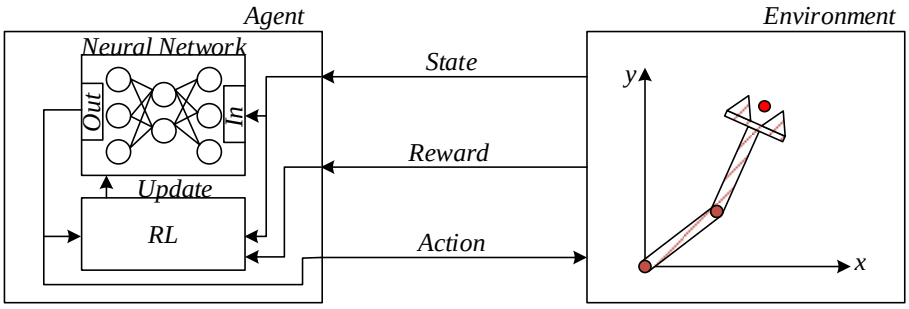


Figure A. RL algorithm block diagram

Purpose: In this study, the noise standard deviation of the deep deterministic policy gradient (DDPG) method, which is one of the policy learning algorithms, was examined.

Theory and Methods: Eight different functions were determined depending on the maximum value of the output of the action artificial neural network. Created artificial neural networks were trained by using these functions in 1000 iterations with 200 steps in each iteration. The best three groups were retrained 2500 iterations and 200 steps and tested for 100 different test scenarios after the training.

Results: After the training, the statistical difference between the groups was examined and it was found that there was no statistical difference between the three best groups. After testing, the inverse kinematic equation of a 2-DoF planar robot with minimal errors was obtained with the help of artificial neural networks.

Conclusion: In the light of the results, the importance of the choice of the standard deviation of noise and the correct range of selection was presented for researchers who will work in this field.

*Corresponding author, e-mail: mustafacanbingol@gmail.com
DOI: 10.29109/gujsc.872646

GU J Sci, Part C, 9(2): 200-210 (2021)

Gazi Üniversitesi

Fen Bilimleri Dergisi

PART C: TASARIM VE TEKNOLOJİ

http://dergipark.gov.tr/gujsc

Investigation of the Standard Deviation of Ornstein - Uhlenbeck Noise in the DDPG Algorithm

Mustafa Can BİNGÖL

Fırat Üniversitesi, Teknoloji Fakültesi, Mekatronik Mühendisliği Bölümü, 23160, Merkez/ELAZIG

Makale Bilgisi

Araştırma makalesi
Başvuru: 01/02/2021
Düzeltme: 17/04/2021
Kabul: 30/04/2021

Keywords

Deep deterministic policy gradient (DDPG)
Ornstein–Uhlenbeck noise
Reinforcement learning
Inverse kinematics
Standard deviation range

Abstract

Reinforcement learning is a learning method that many creatures often unwittingly use to gain abilities such as eating and walking. Inspired by this learning method, machine learning researchers have reduced this learning method to subheadings as value learning and policy learning. In this study, the noise standard deviation of the deep deterministic policy gradient (DDPG) method, which is one of the policy learning algorithms, was examined to solve inverse kinematics of a 2 degrees-of-freedom planar robot. For this examination, 8 different functions were determined depending on the maximum value of the output of the action artificial neural network. Created artificial neural networks were trained by using these functions in 1000 iterations with 200 steps in each iteration. After the training, the statistical difference between the groups was examined and it was found that there was no statistical difference between the three best groups. For this reason, the best three groups were retrained 2500 iterations and 200 steps and tested for 100 different test scenarios after the training. After testing, the inverse kinematic equation of the 2 degrees-of-freedom planar robot with minimal errors was obtained with the help of artificial neural networks. In the light of the results, the importance of the choice of the standard deviation of noise and the correct range of selection was presented for researchers who will work in this field.

Anahtar Kelimeler

Derin deterministtik politika gradyanı (DDPG)
Ornstein–Uhlenbeck gürültüsü
Pekiştirmeli öğrenme
Ters kinematik
Standart sapma aralığı

DDPG Algoritmasında Bulunan Ornstein–Uhlenbeck Gürültüsünün Standart Sapmasının Araştırılması

Öz

Pekiştirmeli öğrenme, birçok canlının yemek yeme ve yürüme gibi beceriler kazanmak için genellikle farkında olmadan kullandığı bir öğrenme yöntemidir. Bu öğrenme yönteminden ilham alan makine öğrenmesi araştırmacıları, değer öğrenme ve politika öğrenme olarak bu öğrenme yöntemini alt başlıklara indirgemişlerdir. Yapılan bu çalışmada politika öğrenme algoritmalarından biri olan derin deterministtik politika gradyanı (deep deterministic policy gradiend-DDPG) yönteminin gürültü standart sapması bir adet 2 serbestlik dereceli düzlemsel robotunun ters kinematik çözümü için incelenmiştir. Yapılan bu inceleme için eylem yapay sinir ağının çıkışının maksimum değerine bağlı olarak 8 farklı fonksiyon belirlenmiştir. Oluşturulan yapay sinir ağları, bu fonksiyonlar kullanılarak her bir iterasyonda 200 adım olacak şekilde 1000 iterasyon eğitilmiştir. Eğitim sonrasında gruplar arası istatistiksel fark bakılmış ve en iyi üç grup arasında istatistiksel fark olmadığı saptanmıştır. Bu nedenle en iyi üç grup 2500 iterasyon ve 200 adım yeniden eğitilmiş ve eğitim sonrasında 100 farklı test senaryosu için test edilmiştir. Test işleminden sonra minimal hatalar ile 2 serbestlik dereceli düzlemsel robotunun ters kinematik denklemi yapay sinir ağları yardımı ile elde edilmiştir. Sonuçlar ışığında, gürültünün standart sapması seçiminin önemi ve hangi aralıkta seçilmesinin daha doğru olacağı bu alanda çalışacak olan araştırmacılar için sunulmuştur.

1. GİRİŞ (INTRODUCTION)

Makine öğrenmesi, öğrenme yöntemlerine göre denetimli (supervised), denetimsiz (unsupervised) ve pekiştirmeli öğrenme (reinforcement learning-RL) olmak üzere üç ana başlık altında toplanabilir [1]. Denetimli öğrenme, sayısal veya sözel ifadeler ile etiketlenmiş veri kümeleri kullanılarak belirlenen öğrenebilir mimarilerin girişleri ile çıkışları arasındaki bağlantının kurulması işlemidir. Sınıflandırma

veya regresyon problemleri denetimli öğrenme yardımıyla çözülebilir. Denetimli öğrenmenin diğer öğrenme yöntemlerine göre en büyük dezavantajı veri kümesi oluşturma maliyetidir. Denetimsiz öğrenme, etiketlenmemiş verilerin belirlenen öğrenebilir mimariler tarafından analiz edilmesi işlemidir. Denetimsiz öğrenme yardımı ile kümeleme veya matris tamamlama problemleri çözülebilir. Denetimsiz öğrenme ile denetimli öğrenme karşılaştırıldığında, denetimsiz öğrenmede veri kümesinin etiket verisi olmadığından veri kümesi daha hızlı oluşturulabilir. RL, doğrudan uygulanacağı çevrede deneme yanılma yöntemi ile oluşan ödül veya ceza değerlerine göre öğrenebilir bir mimarinin eğitilmesi işlemidir. RL, robot kontrolü veya karmaşık kontrol problemlerini çözmek için kullanılabilir. RL, denetimli veya denetimsiz öğrenme gibi bir veri kümesine ihtiyaç duymadığı için diğer yöntemler ile kıyaslandığında daha maliyetsizdir.

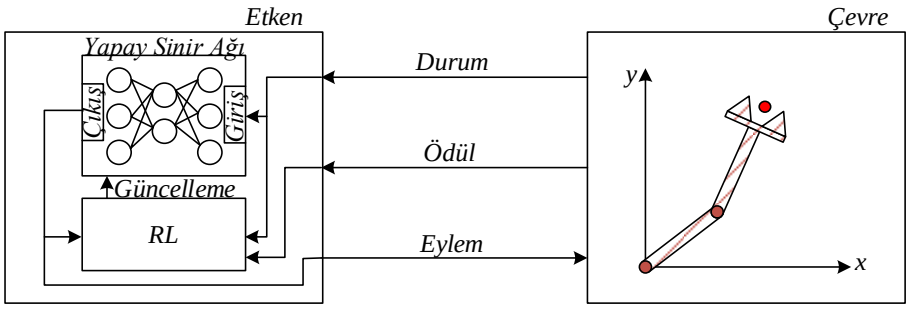
RL'nin temeli Markov karar sürecine dayanmaktadır. Markov karar süreci, ayrık zamanda gelişen skolastik olayları anlık duruma (s) göre muhtemel eylem seçeneklerinin (a) ödüllerini $(Q(s,a) = E[R(s,a)])$. dinamik olarak hesaplayarak en iyi bir sonraki durumu (s') tahmin etme yaklaşımıdır [2]. Derin RL algoritmaları, $Q(s,a)$ sonucunun tahmin edilerek $a = \text{argmax}_{a'} Q(s,a')$ işlemini gerçekleştiren değer öğrenme (value learning) ve $a \simeq \pi(s)$ denklemini gerçekleştirmeye çalışan politika öğrenme (policy learning) olmak üzere iki farklı grupta incelenebilir. Derin deterministtik politika gradyanı (deep deterministic policy gradiend-DDPG) literatürde sıklık ile kullanılan bir mimaridir [3–9]. Örnek olarak 27 serbestlik derecesine (degree of freedom-DoF) sahip bir insansı robotun uç işlevcisinin hareketi sırasında robotun dengesini sağlama işlemi verilebilir [3]. Diğer bir çalışmada, özelleşmiş bir DDPG mimarisi olan ikiz gecikmiş DDPG yapısı kullanılarak 5-DoF robot kolunun ters kinematik çözümü yapılmıştır [4]. 7-DoF bir robotun nesnenin çalışma uzayında olup olmadığını algılama ve çalışma uzayında olan bir nesneye dokunabilme işlemi başka bir çalışmada DDPG yapısı kullanılarak gerçekleştirilmiştir [8]. Sadece eklemli robotlarda değil aynı zamanda doğrusal olmayan bir sistem olan insansız hava aracı gibi sistemlerde irtifa kontrolü DDPG yardımı ile gerçekleştirilmiştir [9].

Ornstein–Uhlenbeck süreci (OUS), matematikten [10] enerjiye [11] kadar birçok alanda sıklıkla kullanılan skolastik türev işlemidir. Yapılan bir çalışmada, OUS kullanılarak rüzgâr gücünün tahmini çalışılmıştır [11]. Bir diğer çalışmada, uzun dönem tutarlı hedef tahmini OUS yardımı ile gerçekleştirilmiştir [12]. Robotik özelinde OUS incelendiğinde, RL'de bulunan etken (agent) yapısının keşfi için OUS kullanılmıştır [13]. Ayrıca, Çince kaligrafi robotu oluşturulması sırasında OUS bir gürültü terimi olarak kullanılmıştır [14].

Yürütülen bu çalışmada, DDPG algoritmasında yer alan gürültü terimi olarak OUS kullanılmıştır. Bu gürültü teriminin standart sapması 8 farklı fonksiyon ile kontrol edilmiştir. Oluşturulan bu sistem ile çok serbestlik dereceli bir robotun ters kinematik denkleminin çözümü sağlanmıştır.

2. MATERYAL VE METOT (MATERIAL AND METHODS)

RL algoritması, etken (agent), çevre (environment), ödül (reward- r), durum (state- s) ve eylem (action- a) olmak üzere beş temel yapı üzerine kurulmuştur ve bu yapıların bir birleri ile ilişkisi Şekil 1'de sunulmuştur.



Şekil 1. *Derin RL algoritmasının blok diyagramı*

Derin RL algoritmalarında, geleneksel RL algoritmalarında bulunan Q tabloları yerine yapay sinir ağı mimarileri kullanılmaktadır. Ayrık zamanda çalıştığı varsayılan RL sisteminin zaman indeksi t olmak üzere, gözlenen durum değişkenleri x_t, eylem a_t ve ödül r_t olur. Durum $s_t = (x_1, a_1, \ldots, a_{t-1}, x_t)$

şeklinde gözlem ve eylem çiftleri halinde veya $s_t = x_t$ gösterilebilir. Eylemin modele giriş olarak verildiği RL sistemlerine model tabanlı RL denilirken aksi durum için modelden bağımsız RL olarak adlandırılabilir.

Politika tabanlı bir RL algoritmasında, durum uzayı (S), eylem uzayı (A) ve olasılık dağılımı $(\pi: S \rightarrow P(A))$ olarak tanımlanabilir. Bu algoritmalarda Markov yaklaşımı kullanarak işlem dinamiği $p(s_{t+1}|s_t, a_t)$ ve ödül fonksiyonu $r(s_t, a_t)$ olarak ifade edilebilir. Her hangi bir anda:

$$R_t = \sum_{i=t}^{T} \gamma^{(i-t)} r(s_i, a_i) \quad (1)$$

eşitliği kullanılarak sadeleştirilmiş gelecek ödülü hesaplanabilir [15]. Sadeleştirme katsayısı $(\gamma \in [0,1])$ gelecek ödül değerinin (R) sonsuza gitmesini engellemek için kullanılmaktadır. RL algoritmasının amacı gelecek ödül değerini olabildiğince maksimum yapmaktır. Q değer fonksiyonu bu bilgiler ışığında:

$$Q(s_t, a_t) = E[R_t] \quad (2)$$

şeklinde ifade edilir. Denklem 2 ve Bellman denklemi kullanılarak:

$$Q^*(s_t, a_t) = E[r(s_t, a_t) + \gamma Q^*(s_{t+1}, a'_{t+1})] = E[r(s_t, a_t) + \gamma Q^*(s_{t+1}, \mu(s_{t+1}))] \quad (3)$$

optimal Q^* fonksiyonu elde edilir [16]. Hedef politika fonksiyonu $\mu: S \leftarrow A$ olarak tanımlanır.

2.1. DDPG Algoritması (DDPG Algorithm)

Değer öğrenme süreci ayrık zamanda çalışıp, sürekli zamanda yavaş çalıştığından dolayı geliştirilen DDPG algoritması Tablo 1'de sunulmuştur [16].

Tablo 1. *DDPG Algoritması*

Eleştirmen ağı $Q(s, a\|\theta^Q)$ ve eylem ağını $\mu(s\|\theta^\mu)$ ağırlıkları θ^Q ile θ^μ başlat.
Hedef ağları Q' ve μ' başlat ve $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$ ağırlıkları eşitle.
Tekrar oynatma hafızasını RB başlat.
for iterasyon=1:M
Keşif için gürültüyü η başlat.
İlk durum verileri s_1'i al.
for t=1:T
$a_t = \mu(s_t\|\theta^\mu) + \eta_t$
$r_t, s_{t+1} = Çevre(a_t)$
$(s_t, a_t, r_t, s_{t+1}) \rightarrow RB$
RB'den N tane örnek al.
$y_i = r_i + \gamma Q'(s_{t+1}, \mu'(s_{t+1}\|\theta^{\mu'})\|\theta^{Q'})$
$L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i\|\theta^Q))^2$ fonksiyonunu minimize ederek eleştirmen ağı güncelle
$\nabla_{\theta^\mu} J \simeq \frac{1}{N}\sum_i \nabla_a Q(s_i, \mu(s_i)) \nabla_{\theta^\mu} \mu(s_i\|\theta^\mu)$ gradyanı kullanarak eylem ağını güncelle
$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$ hedef ağların ağırlıklarını güncelle
$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$

RL algoritmalarının keşif ve sömürme olmak üzere iki temel özelliği vardır. Sömürme, çözüm uzayında bulunan en iyi sonuca yakın daha iyi bir sonuç olup olmadığını arama özelliğidir. Keşif özelliği ise çözüm uzayında bulunan en iyi sonuçtan bağımsız daha iyi bir sonucun olup olmadığının incelenmesidir. Keşif özelliği genetik algoritmadaki mutasyon evresine benzetilebilir. DDPG algoritmasının yayınlandığı makalede keşif özelliği için OUS kullanılmıştır [16].

2.2. Ornstein–Uhlenbeck Süreci (Ornstein–Uhlenbeck Process)

OUS, Denklem 4'te sunulan skolastik türev ifadesidir.

$$d\eta_t = -\theta\eta_t dt + \sigma dw_t \quad (4)$$

Denklem 4'te η_t durum değişkeni türevinin $(d\eta_t)$, hız katsayısı (θ), standart sapma (σ) ve Wiener işlemine (dw_t) bağlı olduğu görülmektedir. Kayma katsayısı (ν) ve $d\eta_t = \frac{\eta_{t+1}-\eta_t}{dt}$ ileri fark eşitliği kullanılarak:

$$\eta_{t+1} = \eta_t + \theta(\nu - \eta_t)dt^2 + \sigma w_t \quad (5)$$

Denklem 4 revize edilebilir. Wiener işlemi:

$$w_n(t) = \frac{1}{\sqrt{n}} \sum_{1\le k\le nt} \xi_k, t \in [0,1] \quad (6)$$

denklemi ile ifade edilir. Denklem 6'da kullanılan ξ_k ortalaması 0 varyansı 1 olan rasgele bir sayıdır. Yapılan çalışmada w_t yerine ortalaması 0 varyansı 1 olan Gauss dağılımlı beyaz gürültü kullanılmıştır. Ayrıca, θ, ν ve dt sırası ile sistemin maksimum çıkışı (upper bound-ub) $ub = \pi/18$, 0 ve 10^{-1} olarak seçilmiştir ve σ olarak:

$$SSF1 = \frac{ub}{10^5} \quad (7)$$

$$SSF2 = \frac{ub}{10^4} \quad (8)$$

$$SSF3 = \frac{ub}{10^3} \quad (9)$$

$$SSF4 = \frac{ub}{10^2} \quad (10)$$

$$SSF5 = \frac{ub}{10} \quad (11)$$

$$SSF6 = \begin{cases} \frac{ub}{10^2} & i < 100 \\ \frac{ub}{10(i/10)} & DD \end{cases}, i = 1,2,\dots,N \quad (12)$$

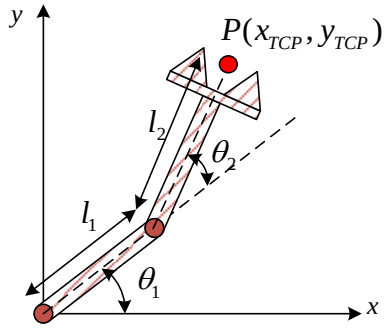
$$SSF7 = \begin{cases} \frac{ub}{10^2} & i < 100 \\ \frac{ub}{10\sqrt{i}} & DD \end{cases}, i = 1,2,\dots,N \quad (13)$$

$$SSF8 = \begin{cases} \frac{ub}{10^2} & i = 1 \\ \frac{SSF8}{10} & \mathrm{mod}(i,250) = 0 \\ SSF8 & DD \end{cases}, i = 1,2,\dots,N \quad (14)$$

standart sapma fonksiyonları (SSF) kullanılmıştır. Denklem 12-14'te *i*, *N* ve *DD* sırası ile anlık iterasyon numarasını, toplam iterasyon sayısını ve diğer durumları sembolize etmektedir.

2.3. Sistemin Kinematik Modeli (Kinematic Model of System)

Hem standart sapma fonksiyonlarının araştırılabilmesi hem de sistemin kolay anlaşılabilmesi için yürütülen bu çalışmada Şekil 2'de sunulan 2 DoF düzlemsel robottan yararlanılmıştır.



Şekil 2. *2 DoF düzlemsel robot*

$\theta = \{\theta_1, \theta_2\} \in \mathbb{R}$ ve $K: \mathbb{R}^{2\times1} \rightarrow \mathbb{R}^{2\times1}$ olmak üzere robot manipülatörünün araç-merkez noktası (tool-center point-TCP) $P = K(\theta)$ şeklinde ifade edilebilir. K fonksiyonu sistemin kinematik modelini sembolize etmektedir. Yapılan çalışmada bu fonksiyon DH yöntemi ile elde edilmiştir ve DH parametreleri Tablo 2'de sunulmuştur.

Tablo 2. *2 DoF Düzlemsel Robot DH Parametreleri*

Eksen-i	θ_i	d_i	a_i	α_i
1	θ_1	-	$l_1 = 0.1m$	-
2	θ_2	-	$l_2 = 0.1m$	-

Elde edilen Tablo 2 ve:

$$T_i^{i-1} = R_{Z_{i-1}}(\theta_i)T_{Z_{i-1}}(d_i)T_{X_i}(a_i)R_{X_i}(\alpha_i) \quad (15)$$

$$K(\omega) = T_1^0 T_2^1 \quad (16)$$

homojen dönüşüm matrisi (T_i^{i-1}) kullanılarak sistemin kinematik modeli oluşturulmuştur. Denklem 15'te bulunan R_{Eksen} ve T_{Eksen} sırası ile eksene ait dönme ve öteleme işlemlerini ifade etmektedir.

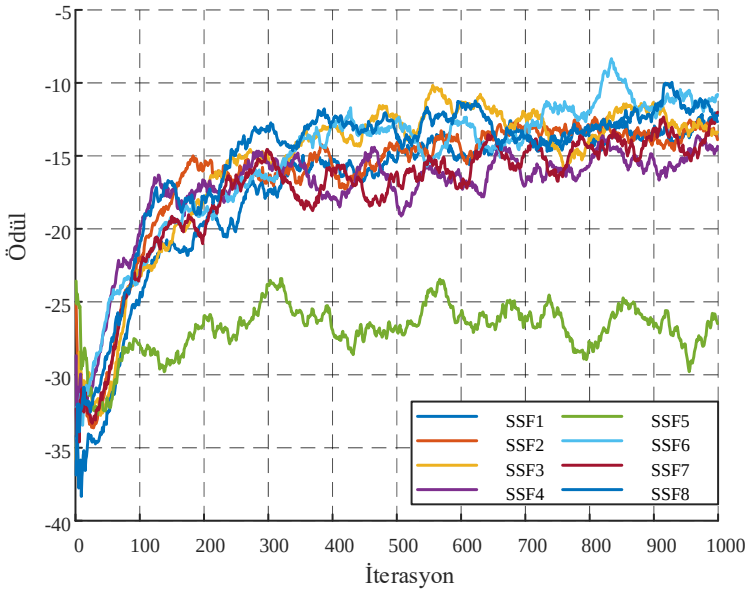
Yapılan bu çalışmada, DDPG algoritması ile sistemin ters kinematiği olan $\omega = K^{-1}(P)$ eşitliğini sağlayan K^{-1} fonksiyonun elde edilmesi amaçlanmıştır. Oluşturulan bu sistemde ödül denklemi:

$$r(x_{TCP}, y_{TCP}, x_{REF}, y_{REF}) = -\sqrt{(x_{TCP} - x_{REF})^2 + (y_{TCP} - y_{REF})^2} \quad (17)$$

kullanılmıştır. Denklem 17'de kullanılan (x_{TCP}, y_{TCP}) noktaları Şekil 2'de bahsedildiği üzere manipülatörün araç-merkez noktasının koordinatlarını ifade etmektedir. Bu koordinatlar RL'nin ürettiği eklem açıları sayesinde güncellenmektedir. (x_{REF}, y_{REF}) koordinatı ise referans noktasını temsil etmektedir ve oluşturulan ortam sayesinde robotun çalışma uzayında bulunan noktalardan rasgele birinin seçilmesi ile belirlenmektedir.

3. BULGULAR (RESULTS)

Yapılan çalışmada, ilk olarak 1000 iterasyonda robotun çalışma uzayına uygun olarak farklı rastlantısal (x_{REF}, y_{REF}) noktası belirlenmiştir. Her bir iterasyon 200 adım çalıştırılmıştır ve elde edilen ödüllerin son 40 adıma göre ortalamaları Şekil 3'te sunulmuştur.



Şekil 3. *Ortalama ödül değerleri*

Farklı standart sapma fonksiyonlarına göre elde edilen Şekil 3, istatistiksel olarak Tablo 3 ve 4'te incelenmiştir.

Tablo 3. *Ortalama Ödül Değerlerinin Ortalamaları ve Standart Sapmaları*

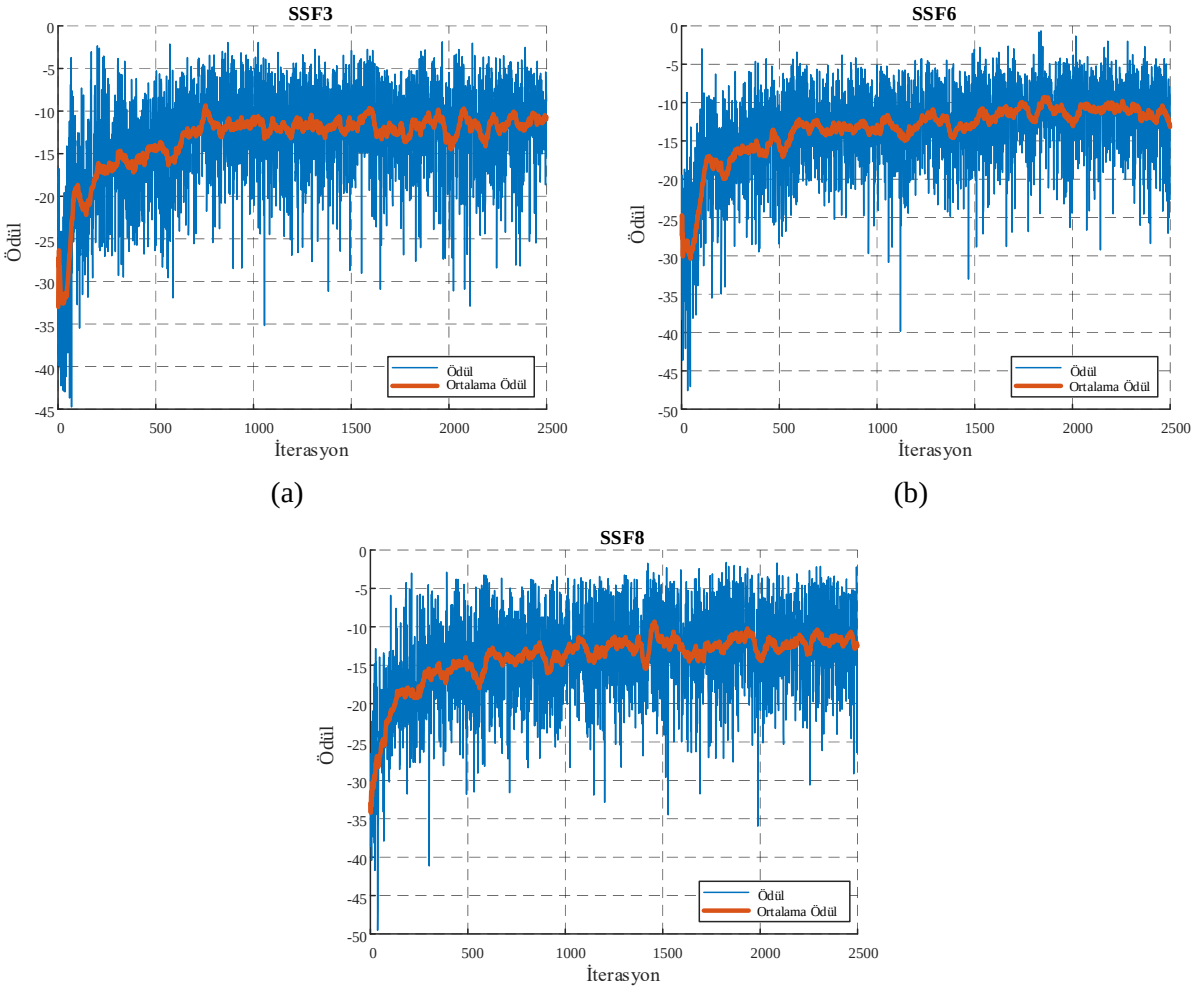
SSF (n)	*Ortalama ($\bar{X}$)*	*Standart Sapma (SS)*
SSF1 (n:1000)	*-17.289*	*5.520*
SSF2 (n:1000)	*-16.246*	*4.622*
SSF3 (n:1000)	*-15.600*	*5.413*
SSF4 (n:1000)	*-17.253*	*3.518*
SSF5 (n:1000)	*-26.864*	*1.695*
SSF6 (n:1000)	*-15.388*	*4.903*
SSF7 (n:1000)	*-17.489*	*4.549*
SSF8 (n:1000)	*-15.125*	*4.976*

Ödül değerlerinin genel ortalaması yükseldikçe sistemin başarısı artmaktadır. Bütün gruplar istatistiksel olarak Tukey çoklu karşılaştırma testiyle karşılaştırılmıştır. Bu testin kullanılma nedeni grup sayısının 2'den fazla olması ve gruplardaki örneklem sayısının eşit olmasıdır. Karşılaştırma sonrasında SSF3-6-8 gruplarının diğer gruplar ile aralarında istatistiksel bir fark olduğu sonucuna varılmıştır ($p<0.05$). Ödül değerlerinin genel ortalaması karşılaştırma testi ile yorumlandığında, SSF3-6-8 gruplarının belirlenen problemin çözümünde diğer gruplardan daha başarılı olduğu görülmektedir. SSF3-6-8 gruplarının diğer gruplar ile istatistiksel olarak karşılaştırılması Tablo 4'te sunulmuştur.

Tablo 4. *Standart Sapma Fonksiyonlarının Karşılaştırılması*

SSF-I1	*SSF-J1*	*p(I1-J1)*	*SSF-I2*	*SSF-J2*	*p(I2-J2)*	*SSF-I3*	*SSF-J3*	*p(I3-J3)*
	SSF1	***<0.05***		*SSF1*	***<0.05***		*SSF1*	***<0.05***
	SSF2	***<0.05***		*SSF2*	***<0.05***		*SSF2*	***<0.05***
	SSF4	***<0.05***		*SSF3*	*0.969*		*SSF3*	*0.278*
SSF3	*SSF5*	***<0.05***	*SSF6*	*SSF4*	***<0.05***	*SSF8*	*SSF4*	***<0.05***
	SSF6	*0.969*		*SSF5*	***<0.05***		*SSF5*	***<0.05***
	SSF7	***<0.05***		*SSF7*	***<0.05***		*SSF6*	*0.903*
	SSF8	*0.278*		*SSF8*	*0.903*		*SSF7*	***<0.05***

SSF3-6-8 diğer standart sapma fonksiyonlarına göre istatistiksel olarak farklı çıkarken ($p<0.05$) kendi aralarında bir istatistiksel fark bulunmamaktadır. Bu sebepten dolayı SSF3-6-8 2500 iterasyon ve 200 adım tekrar eğitilmiştir. Bu eğitim sırasında oluşan ödül ve ortalama ödül grafikleri Şekil 4'te verilmiştir.



Şekil 4*. 2500 iterasyon eğitim sonrası ödül ve ortalama ödül değerleri; (a) SSF3 ile eğitim sırasında oluşan ödül değerleri, (b) SSF6 ile eğitim sırasında oluşan ödül değerleri, (c) SSF8 ile eğitim sırasında oluşan ödül değerleri*

Elde edilen ortalama ödül değerleri Tablo 5 ve 6'da sunulduğu üzere istatistiksel olarak incelenmiştir.

Tablo 5. *Ortalama Ödül Değerlerinin Ortalamaları ve Standart Sapmaları*

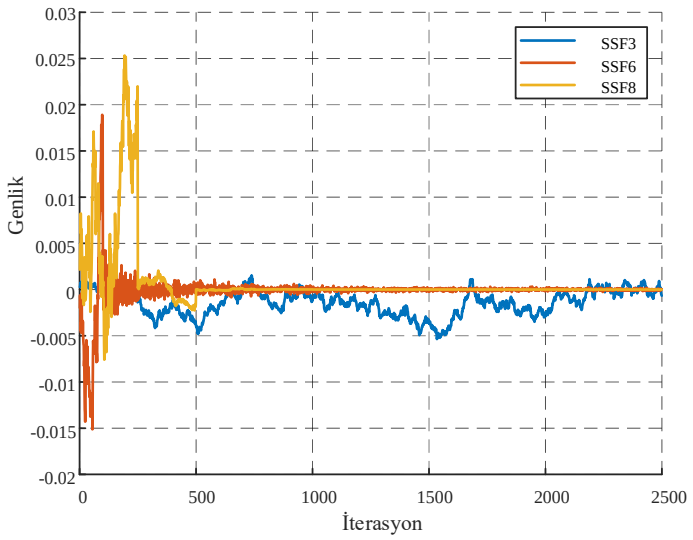
SSF (n)	*Ortalama ($\bar{X}$)*	*Standart Sapma (SS)*
SSF3 (n:2500)	*-13.332*	*3.693*
SSF6 (n: 2500)	*-13.605*	*3.618*
SSF8 (n: 2500)	*-14.075*	*3.362*

Ortalama ödül değerlerinin ortalamaları çok yakın değerlerlerdir ve üç grup kendi aralarında yeniden Tukey karşılaştırma testine Tablo 6'da sunulduğu gibi tabi tutulmuştur.

Tablo 6. *Belirlenen Standart Sapma Fonksiyonlarının Karşılaştırılması*

SSF-I1	*SSF-J1*	*p(I1-J1)*	*SSF-I2*	*SSF-J2*	*p(I2-J2)*	*SSF-I3*	*SSF-J3*	*p(I3-J3)*
SSF3	*SSF6*	***<0.05***	*SSF6*	*SSF3*	***<0.05***	*SSF8*	*SSF3*	***<0.05***
	SSF8	***<0.05***		*SSF8*	***<0.05***		*SSF6*	***<0.05***

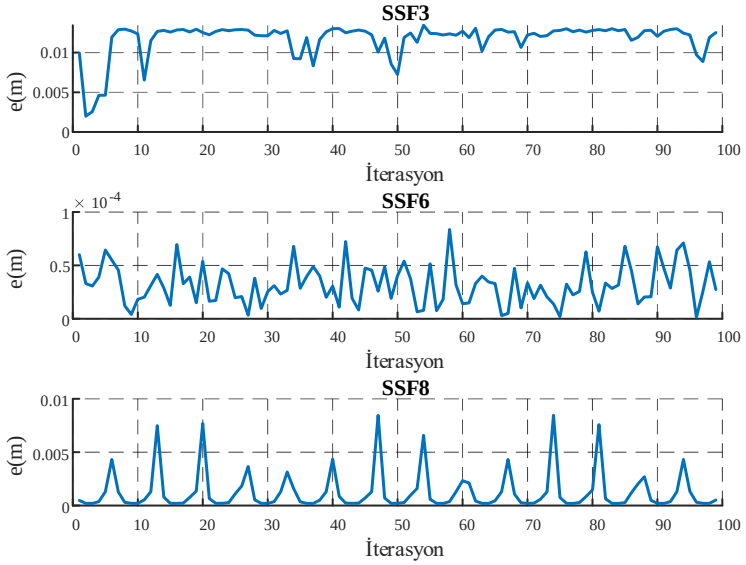
Karşılaştırma testinde üç grup birbirlerinden istatistiksel olarak farklı çıkmıştır ($p<0.05$). Bu deney sırasında oluşturulan gürültü Şekil 5'te sunulmuştur.



Şekil 5. *Oluşturulan gürültüler*

Yürütülen bu çalışmada bulunan yapay sinir ağları grafik işlem birimi (graphic processor unit-GPU) vasıtası ile eğitilmiştir. SSF3'ün eğitildiği PC özellikleri; 2 çekirdekli Intel Xeon CPU 2.00GHz, 12.72GB RAM, 147.15GB disk hafızası ve GPU olarak Nvidia Tesla V100-16GB-5120 Cuda- 640 Tensor çekirdeğidir. SSF6 ve 8'in eğitildiği PC özellikleri; 2 çekirdekli Intel Xeon CPU 2.00GHz, 12.72GB RAM, 147.15GB disk hafızası ve GPU olarak Nvidia Tesla P100-16GB-3584 Cuda çekirdeğidir. SSF3-6-8'in eğitim süreleri sırası ile 3335.451s, 2871.760s ve 2908.519s'dir. Bu üç SSF ile eğitilen yapılar eğitim aşamasına benzer şekilde 100 farklı referans noktası için test edilmiş, her bir test için oluşan hata Denklem 18 kullanılarak hesaplanmış ve bu testler sırasında oluşan minimum hatalar Şekil 6'da gösterilmiştir.

$$e(x_{TCP}, y_{TCP}, x_{REF}, y_{REF}) = \sqrt{(x_{TCP} - x_{REF})^2 + (y_{TCP} - y_{REF})^2} \quad (18)$$



Şekil 6. *Test hata sonuçları*

Şekil 6'da sunulduğu üzere çok serbestlik dereceli K^{-1} fonksiyonu üç farklı SSF fonksiyonu yardımıyla minimal hatalı olarak elde edilmiştir.

4. SONUÇ (CONCLUSION)

RL, doğada sıkılıkla kullanılan bir öğrenme yöntemidir. RL'ye doğal bir örnek olarak bir çitanın herhangi bir öğreticisi olmadan doğumundan saatler sonrasında yüksek hızla koşması verilebilir. Yapılan bu çalışmada yapay sinir ağlarının 2 DoF düzlemsel robotun ters kinematiğini öğrenmesi için bir çevre oluşturulmuştur. Oluşturulan bu sistem içerisinde bulunan yapay sinir ağları (eleştirmen-eylem) DDPG algoritması yardımı ile eğitilmiştir. Bu eğitimler sırasında DDPG algoritmasında bulunan gürültünün standart sapmasının araştırılması için 8 farklı standart sapma fonksiyonu belirlenmiştir. Belirlenen bu standart sapma fonksiyonları, 1000 iterasyon ve her iterasyonda 200 adım olacak şekilde eğitilmiştir. Eğitim sonrası gruplar arası farka bakılmış ve en iyi üç standart sapma fonksiyonu 2500 iterasyon-200 adım eğitilmiştir. Bu eğitimlerde SSF3 kullanılan eğitimde Tesla V100, SSF6-8'de Tesla P100 GPU kullanılmıştır. V100 GPU gerek işlemci saatleri gerekse hem fazla sayıda hem de farklı mimaride bulunan çekirdekleri ile P100'den 3 kat daha hızlıdır. Buna rağmen SSF3 3335.451s ile en çok vakit alan eğitim süresine sahiptir. Diğer iki fonksiyon ise bire bir aynı özelliklere sahip PC'lerde eğitilmiş ve SSF6, SSF8'den 36.759s daha hızlı eğitilmiştir. Hem oluşturulan gürültüler (Şekil 5) hem de Denklem 9-12-14 incelendiğinde SSF3 sürekli rejime sahip bir gürültü iken SSF6 ve 8 sıfıra yakınsayan gürültülerdir. Bu sebepten dolayı eğitim süresinde farklılıklar oluştuğu düşülmektedir. Eğitim performansı olarak Tablo 5 ve 6'da görüldüğü üzere üç grup istatistiksel olarak farklıdır. Test hata sonuçları (Şekil 6) incelendiğinde, SSF6 ile eğitilen yapay sinir ağlarının test sırasında en düşük hata üreten sinir ağı olduğu sonucu çıkarılabilir. Bu sebeplerden dolayı, çözülmesi planlanan 2 DoF düzlemsel robotun ters kinematiği için en uygun standart sapma fonksiyonu, SSF6'dır. Yapılan bu çalışma sayesinde, sistem dinamiği içermeyen benzer problemlerin çözümü için doğrudan belirtilen SSF3-6-8 kullanımı araştırmacıların çalışmalarına hız kazandırabilir. Gelecek çalışmalarda, farklı gürültü denklemlerinin farklı sistemler üzerine etkisi araştırılacaktır.

KAYNAKLAR (REFERENCES)

[1] Murphy, K.P. (2012). Machine learning: a probabilistic perspective. MIT press. https://doi.org/10.1109/pes.2005.1489456

[2] Geron, A. (2019). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow:

Concepts, tools, and techniques to build intelligent systems. O'Reilly Media.

[3] Phaniteja, S., Dewangan, P., Guhan, P., Sarkar, A. ve Krishna, K.M. (2017). A deep reinforcement learning approach for dynamically stable inverse kinematics of humanoid robots. 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), s. 1818–23.

[4] Shi, X., Guo, Z., Huang, J., Shen, Y. ve Xia, L. (2020). A Distributed Reward Algorithm for Inverse Kinematics of Arm Robot. Proceedings - 5th International Conference on Automation, Control and Robotics Engineering, CACRE 2020, s. 92–6. https://doi.org/10.1109/CACRE50138.2020.9230347

[5] Wang, Y., Tong, J., Song, T.Y. ve Wan, Z.H. (2018). Unmanned surface vehicle course tracking control based on neural network and deep deterministic policy gradient algorithm. 2018 OCEANS - MTS/IEEE Kobe Techno-Oceans, OCEANS - Kobe 2018, IEEE. s. 3–7. https://doi.org/10.1109/OCEANSKOBE.2018.8559329

[6] Tuyen, L.P. ve Chung, T.C. (2017). Controlling bicycle using deep deterministic policy gradient algorithm. 2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence, URAI 2017, s. 413–7. https://doi.org/10.1109/URAI.2017.7992765

[7] Wang, W.Y., Ma, F. ve Liu, J. (2019). Course tracking control for smart ships based on a deep deterministic policy gradient-based algorithm. ICTIS 2019 - 5th International Conference on Transportation Information and Safety, IEEE. s. 1400–4. https://doi.org/10.1109/ICTIS.2019.8883840

[8] De La Bourdonnaye, F., Teuliere, C., Chateau, T. ve Triesch, J. (2019). Within Reach? Learning to touch objects without prior models. 2019 Joint IEEE 9th International Conference on Development and Learning and Epigenetic Robotics, ICDL-EpiRob 2019, s. 93–8. https://doi.org/10.1109/DEVLRN.2019.8850702

[9] Ghouri, U.H., Zafar, M.U., Bari, S., Khan, H. ve Khan, M.U. (2019). Attitude Control of Quad-copter using Deterministic Policy Gradient Algorithms (DPGA). 2019 2nd International Conference on Communication, Computing and Digital Systems, C-CODE 2019, IEEE. s. 149–53. https://doi.org/10.1109/C-CODE.2019.8681003

[10] Ayoubi, T. ve Bao, H. (2020). Persistence and extinction in stochastic delay Logistic equation by incorporating Ornstein-Uhlenbeck process. Applied Mathematics and Computation, Elsevier Inc. C.386, 125465. https://doi.org/10.1016/j.amc.2020.125465

[11] Arenas-López, J.P. ve Badaoui, M. (2020). The Ornstein-Uhlenbeck process for estimating wind power under a memoryless transformation. Energy, C.213. https://doi.org/10.1016/j.energy.2020.118842

[12] Millefiori, L.M., Braca, P. ve Willett, P. (2016). Consistent Estimation of Randomly Sampled Ornstein-Uhlenbeck Process Long-Run Mean for Long-Term Target State Prediction. IEEE Signal Processing Letters, IEEE. C.23, Sayı 11, 1562–6. https://doi.org/10.1109/LSP.2016.2605705

[13] Nauta, J., Khaluf, Y. ve Simoens, P. (2019). Using the Ornstein-Uhlenbeck process for random exploration. COMPLEXIS 2019 - Proceedings of the 4th International Conference on Complexity, Future Information Systems and Risk, s. 59–66. https://doi.org/10.5220/0007724500590066

[14] Wu, R., Zhou, C., Chao, F., Yang, L., Lin, C.M. ve Shang, C. (2020). Integration of an actor-critic model and generative adversarial networks for a Chinese calligraphy robot. Neurocomputing, Elsevier B.V. C.388, 12–23. https://doi.org/10.1016/j.neucom.2020.01.043

[15] Hou, Y., Liu, L., Wei, Q., Xu, X. ve Chen, C. (2017). A novel DDPG method with prioritized experience replay. 2017 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2017, s. 316–21. https://doi.org/10.1109/SMC.2017.8122622

[16] Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y. vd. (2016). Continuous control with deep reinforcement learning. 4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings .