T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

PEKİŞTİRMELİ ÖĞRENME YÖNTEMİ İLE OPTİMAL DC MOTOR HIZ KONTROLCÜSÜNÜN TASARLANMASI

YÜKSEK LİSANS TEZİ

Bekir Murat AYDIN

Enstitü Anabilim Dalı	**:**	**ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ**
Enstitü Bilim Dalı	**:**	**ELEKTRONİK**
Tez Danışmanı	**:**	**Dr. Öğr. Üyesi Burhan BARAKLI**

Haziran 2022

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

PEKİŞTİRMELİ ÖĞRENME YÖNTEMİ İLE OPTİMAL DC MOTOR HIZ KONTROLCÜSÜNÜN TASARLANMASI
I

YÜKSEK LİSANS TEZİ

Bekir Murat AYDIN

Enstitü Anabilim Dalı	**:**	**ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ**
Enstitü Bilim Dalı	**:**	**ELEKTRONİK**

Bu tez 16.06.2022 tarihinde aşağıdaki jüri tarafından oybirliği ile kabul edilmiştir.

Jüri Başkanı	**Üye**	**Üye**

BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim

Bekir Murat AYDIN
16.06.2022

TEŞEKKÜR

Yüksek lisans eğitimim boyunca değerli bilgi ve deneyimlerinden yararlandığım, her konuda bilgi ve desteğini almaktan çekinmediğim, araştırmanın planlanmasından yazılmasına kadar tüm aşamalarında yardımlarını esirgemeyen, teşvik eden, aynı titizlikte beni yönlendiren değerli danışman hocam Dr. Öğr. Üyesi Burhan BARAKLI'ya teşekkürlerimi sunarım.

Ayrıca beni koşulsuz şartsız seven ve desteğini esirgemeyen babam Remzi AYDIN'a, annem Binnaz AYDIN'a ve abim Özal Mesut AYDIN'a sevgilerimi ve teşekkürlerimi sunarım.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

$\mathcal{V}_\pi$: Değer fonksiyonu

τ_s : Sistemin zaman sabiti

A : Aksiyon sinyali

E : Hata sinyali

G : Kontrol edilen sistemin transfer fonksiyonu

G_c : Kontrolcünün transfer fonksiyonu

G_n : Beklenen ödül miktarı

K : Güç katı kazancı

K_b : Elektromotor kuvveti katsayısı

K_i : Motor tork katsayısı

K_s : Sistemin açık-çevrim kazancı

R : Ödül sinyali

$\mathcal{R}$: Sistemin alabileceği tüm ödülleri içeren küme

S : Sistem durum bilgisi

T : Örnekleme zamanı

T_e : Elektriksel tork

T_m : Mekanik tork

U : Kontrol işareti

Q : Aksiyon-Değer fonksiyonu

$\mathcal{A}$: Sistemin seçebileceği tüm aksiyonları içeren küme

$\mathcal{S}$: Sistemin alabileceği tüm durumları içeren küme

α : Öğrenme oranı

γ : Zayıflatma faktörü

π : Politika

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar Kelimeler: Pekiştirmeli Öğrenme, Adaptif PID, Optimal Kontrol

Çalışmamızda, pekiştirmeli öğrenme tabanlı adaptif PID kontrolör tasarlanmıştır. Literatürde pekiştirmeli öğrenme yöntemi kullanan kontrolcüler tasarlanmıştır ve başarılı sonuçlar verdiği görülmüştür. Çalışmamızda en yaygın kullanılan kontrolcü yapısı olan PID kontrolörün tasarımında pekiştirmeli öğrenme yöntemlerinden birisi olan Q-Öğrenme algoritması kullanılmıştır. Q-Öğrenme algoritması çalışmamızda üç farklı yolla uygulanmıştır. Birinci yöntemde bir ajan oluşturulmuş ve tüm PID katsayılarını arttırıp, azaltabilmektedir. İkinci yöntemde her PID katsayısı için bir ajan atanmış ve her ajan ilgili PID katsayısını arttırıp azaltabilmektedir. Üçüncü yöntemde ise derin öğrenme tabanlı Q-Öğrenme algoritması kullanan bir ajan oluşturulmuş ve tüm PID katsayılarını ayarlayabilmektedir. Q-Öğrenme yöntemi ile tasarlanan kontrolcüler, model tabanlı tasarlanan PID katsayıları kadar başarılı sonuçlar vermiştir. Her yapının avantajları ve dezavantajları değerlendirilmiştir.

OPTIMAL DC MOTOR SPEED CONTROLLER DESIGN WITH REINFORCEMENT LEARNING METHOD

SUMMARY

Keywords: Reinforcement Learning, Adaptive PID, Optimal Control

In this study, a reinforcement learning based adaptive PID contoller is designed. The reinforcement learning based controllers are designed in the literature and it has been seen that they give successful results. In our study, the Q-Learning algorithm, which is one of the reinforcement learning methods, was used in the design of the PID controller, which is the most widely used controller structure. Q-Learning algorithm was applied in three different methods in our study. In the first method, an agent is created and agent can increase or decrease all of the PID parameters. In the second method, an agent is assigned for each PID parameter and each agent can increase or decrease the revelant PID parameter. In the third method, an agent using deep learning-based Q-Learning algorithm is created and can adjust each PID parameter. Controllers designed with the Q-Learning method gave as successful results as model-based PID controllers. The advantages and disadvantages of each method are examined.

BÖLÜM 1. GİRİŞ

Makine öğrenmesi algoritmalarından birisi olan pekiştirmeli öğrenme, yapı itibariyle diğer makine öğrenmesi yöntemlerinden farklıdır. Literatürde kullanılan makine öğrenmesi yöntemleri gözlemcili ve gözlemcisiz olarak ikiye ayrılabilirken pekiştirmeli öğrenme bu iki sınıfa da ait değildir. Temel olarak canlıların öğrenme içgüdüsünü örnek alır. Çevresi ile etki-tepki ilişkisi içerisindeki bir canlı karmaşık davranışları çevresinden gelen tepkiler ile öğrenebilir ve bulunduğu durumlar ile seçeceği eylemleri ilişkilendirir [1].

Canlıların öğrenmesi alanında yapılan çalışmalar ve makine öğrenmesi alanındaki gelişmeler 1950'li yıllara dayanmaktadır. Minksy, geliştirdiği stokastik nöral analog pekiştirmeli bilgisayar (SNARC) ile bir farenin beyin modeli ile bir labirentten çıkma bulmacasının benzetim çalışmasını yapmıştır [2]. Bu çalışma ile pekiştirmeli öğrenmenin karmaşık problemlerin çözümünde kullanılabileceği gündeme gelmiştir ve birçok pekiştirmeli öğrenme yöntemi geliştirilmiştir. Watkins ve Dayan, Richard E. Bellman tarafından ortaya atılan dinamik programlama yöntemi [1], [3]–[5] ile zamansal fark öğrenmeyi bir araya getirerek Q-Öğrenme algoritmasını geliştirmişlerdir [6]. Q-Öğrenme algoritması az sayıda durum ve aksiyon içeren sistemlerde optimal noktaya başarılı bir şekilde yakınsayabilmektedir. Sistemin durum ve aksiyon sayısı arttıkça Q-Öğrenme algoritmasının optimal noktaya yakınsaması zorlaşmaktadır. 2015 yılında yapılan bir çalışma ile derin öğrenme ve Q-Öğrenme birleştirilerek Derin Q- Öğrenme Ağı (DQN – Deep Q-Network) algoritması geliştirilmiştir [7]. Makalede DQN kontrolcülerinin karmaşık sistemlerde oldukça başarılı sonuçlar verdiği görülmüştür ve DQN kontrolcülerin karmaşık sistemlerin kontrolünde kullanılmasının yolu açılmıştır. Karmaşık problemlerde pekiştirmeli öğrenmenin başarılı sonuçlar verdiği birçok çalışma yapılmıştır [7]–[12]. Reaktif güç kontrolü, trafik ışıklarının kontrolü, video oyunları için kontrol, adaptif PID

(Proportional-Integral-Derivative) tasarımı gibi çeşitli uygulamalarda Q-Öğrenme çalışması kullanılmıştır.

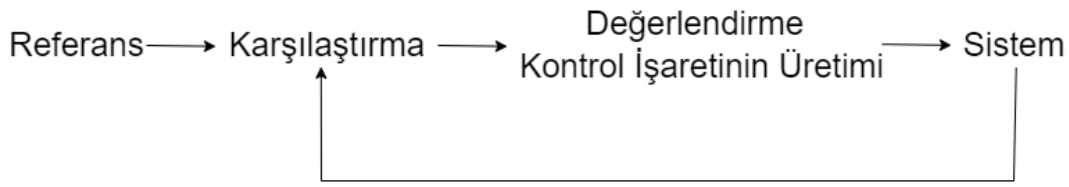
PID kontrolcüler uygulanabilirliği ve basitliği açısından endüstride en çok tercih edilen kontrolcülerden birisidir. PID kontrolcü üstüne yapılan çalışmalar endüstride hızla uygulamaya konması açısından akademi ve endüstri arasında önemli bir ortak çalışma alanıdır [13]. Uzun yıllardır optimal PID katsayılarının ayarlanması için yöntemler geliştirilmektedir. Pekiştirmeli öğrenme yöntemlerindeki gelişmeler PID kontrolcü tasarımı alanında da kullanılmaya başlamıştır. Fırçasız DC motorun hız kontrolü için yapılan bir çalışmada DDPG-PID kontrolcünün standart PID kontrolcüden daha iyi sonuçlar verdiği gözlemlenmiştir [14]. PID parametre optimizasyonu için parçacık sürü algoritması ve genetik algoritma ile pekiştirmeli öğrenme algoritması karşılaştırılmış, pekiştirmeli öğrenmenin daha hızlı yakınsadığı gözlemlenmiştir [15]. Bu konuda yapılan diğer bir çalışmada Deep Q-Öğrenme ve Advantage Actor Critic Yöntemi ile tasarlanan kontrolcülerin özellikle sistem parametresinin değiştiği durumlarda daha başarılı olduğu görülmüştür [16].

Bu çalışmada Q-Öğrenme yöntemi ile DC makine hız kontrolü gerçekleştirilmiştir. Çalışmamızda literatürdeki çalışmalara ek olarak Q-Öğrenme algoritması gerçek-zamanlı bir sistemin kontrolünde kullanılmıştır. Üç ajanlı Q-Öğrenme yapısı, tek ajanlı Q-Öğrenme yapısı ve Derin Q-Öğrenme yapısı ayrı ayrı sisteme uygulanarak sistemin kontrolü sağlanmıştır. Benzetim çalışmaları ve gerçek-zamanlı çalışmalara ait sonuçlar verilmiştir. Pekiştirmeli Öğrenme, Q-Öğrenme ve kontrol yöntemi ile alakalı bilgiler Materyal ve Yöntem başlığı altında verilmiştir. Bulgular ve Tartışma bölümünde sistemin benzetim ortamı ve gerçek-zamanlı çalışma için sonuçları verilmiştir.

BÖLÜM 2. KONTROL SİSTEMLERİ

2.1. Kontrol

Kontrol, bir sistemde fiziksel büyüklük veya büyüklüklerin istenen referans değeri belirlenmiş olan performansta takip etmesini sağlamak için geliştirilmiş kurallar olarak tanımlanabilir. Kontrolcünün işlem basamakları sırasıyla; ölçme, değerlendirme ve kontrol işareti üretimi şeklindedir. Kontrol edilecek olan sistemlerin çıkış işaretleri genelde fiziksel (sıcaklık, basınç, hız, konum, vb.) büyüklüklerdir. Kontrolcü ise elektriksel işaretlere göre işlem yapan sistemlerdir.



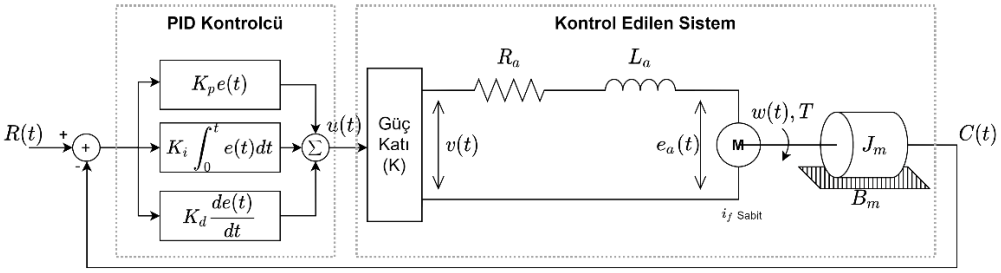
Şekil 2.1. Kontrol Akış Diyagramı

Çalışmamızda DC motor sistemi Ayrık zaman kontrolcü ile kontrol edilecektir. Ayrık zaman kontrolcü ile çalışılacağı için sürekli zaman işaretler, sistemin dinamik davranışına uygun olarak seçilecek olan örnekleme zamanı örneklenecektir.

2.2. PID Kontrol

PID kontrolör, Proportional – Integral – Derivative kelimelerinin baş harflerinden oluşmaktadır. Endüstride kullanılan en yaygın kontrol yöntemlerinden birisidir. PID kontrolcüler anlaşılabilirliği ve uygulanabilirliği sayesinde günümüzde endüstride kontrol denilince akla gelen ilk yöntemlerdendir. Günümüzde PID yapısı, analog PID yapısından oldukça farklıdır. PID tasarımı artık dijital tasarıma dayanmaktadır. Dijital tasarımda ise PID kontrolcünün performansını iyileştirecek birçok yöntem

geliştirilmiş ve geliştirilmeye devam edilmektedir. Endüstri ise bu gelişmeleri hızla bazen de şevkle uygulamaya koymaktadır. Bu sebeple PID kontrol, endüstri ve akademi arasındaki ortak çalışma alanı olmuştur [17]. PID kontrolcünün amacı; girişindeki işareti, işaretin türevini ve integralini belirli katsayılar ile çarparak kontrol işareti üretmektir. PID katsayıları K_p, K_i ve K_d dir. Kapalı çevrim kontrol sisteminde hata işaretini sadece K_p katsayısı ile çarpıldığı kontrolcülerde 0-tipi bir sistem için sürekli hal hatasını ve bozucu etkisini azaltır ancak sıfırlayamaz. Hata işaretinin integralini K_i katsayısı ile çarpılarak üretilen kontrol işareti ise sürekli zamanda $s = 0$'da ayrık zamanda ise $z = 1$'de bir adet kutup ekler. Bu sayede sistemin tipini arttırır. 0-tipi bir sistemde sürekli hal hatasını ve bozucu etkisini sıfırlar ancak sistemin kararsızlığını arttırır ve cevabını yavaşlatır. Hata işaretinin türevini K_d katsayısı ile çarparak ürettilen kontrol işareti ise hata işaretinin değişimine bağlı olarak çıkış ürettiği için geçici durumlarda etkilidir ve sistemin kararlılığını arttırır. Gürültülere karşı çok hassas olduğu için endüstride hassas olmayan uygulamalarda kullanımı tercih edilmemektedir. Sistem incelenip P, PI, PD veya PID kontrolcü yapılarından birisi tercih edilmelidir.



Şekil 2.2. PID Kontrol Blok Diyagramı

Bu durumda PID kontrolcünün transfer fonksiyonu Denklem (2.1)'de ifade edilmiştir.

$$\frac{U(s)}{E(s)} = G_c(s) = (K_p + K_i \frac{1}{s} + K_d s)E(s) \tag{2.1}$$

Ayrık-zaman PID kontrolcü transfer fonksiyonu Denklem (2.2) kullanılarak elde edilmiştir.

$$\frac{U(z)}{E(z)} = G_c(z) = K_p + K_i \frac{z}{z-1} + K_d \frac{z-1}{z} \quad (2.2)$$

PID katsayılarının hesabı için birçok yöntem geliştirilmiştir. En çok uygulanan yöntem Ziegler-Nichols yöntemidir. Sistemin basamak cevabından elde edilen ölü zaman, zaman sabiti ve açık-çevrim kazancı bilgilerine dayanarak kontrolcü katsayılarını belirler [18]. Diğer bir yöntem ise model-tabanlı PID tasarımıdır. Model-tabanlı PID tasarımında matematik modeli bilinen bir sisteme uygun kontrol kutupları atanarak davranışı istenilen performansa en yakın hale getirilir. Model-tabanlı parametrik denklemler yardımı ile analitik olarak PID katsayılarının hesabı yapılabilmektedir [19]. Günümüzde modelden bağımsız, kendi kendini ayarlayabilen PID kontrolcü yapıları üzerine de çalışılmaktadır [20]–[22]. Bu çalışmada modelden bağımsız bir yöntem olan Q-Öğrenme tabanlı adaptif PID kontrolcüsü tasarımı yapılacaktır.

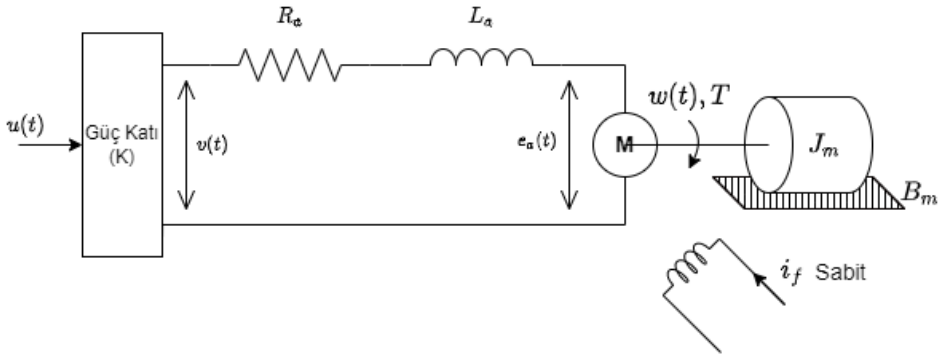
2.2.1. Optimal PID kontrol

Kontrol sistemlerinde lineer olmayan durumlar, değişken olan ve doğru kestirilemeyen olaylar, sistem parametrelerinin zamanla değişmesi ve sistemin dinamiğini etkileyen bir yükün sisteme bağlanması gibi zorluklardan dolayı PID kontrolcüler nadiren optimal olarak ayarlanır [23]. Örnek olarak bir helikopter'i ele alırsak, helikopterin uçacağı hava şartları tasarım esnasında kestirilemez. Hava basıncı, nemi, sıcaklığı her uçuş için farklı değerler alacaktır. Zamanla değişmez kabul edilen birçok kritik parametre zamanla değişip sistemin dinamiğini etkileyecektir. Verilen örnekteki gibi durumlar sebebiyle endüstride oldukça yaygın kullanılan PID kontrolcülerin optimal olarak ayarlanması önemli bir çalışma alanıdır. Optimal kontrolün gerçekleştirilmesi için birçok yöntem geliştirilmiştir, Lineer Quadratik Regülatör, PID performans indeksleri, sezgisel optimizasyon algoritmaları vb. Günümüzde sezgisel optimizasyon algoritmaları, geleneksel yöntemlere alternatif bir yaklaşım haline gelmiştir [24], [25]. Sezgisel optimizasyon algoritmalarının dışında, bulanık mantık da yaygın bir şekilde kullanılmaktadır. Bulanık mantık ile yapılan çalışmalarda tatmin edici sonuçlar elde edilmiştir [26]. Optimal PID tasarımı için her geçen gün yeni yöntemler keşfedilmektedir. Her yöntemin artıları ve eksileri vardır, tasarımcı en uygun yöntemi belirlemelidir.

Tasarlanacak olan kontrolcüden hatayı en uygun biçimde sıfırlanması beklenmektedir. Bu durumda kontrolcü tasarımında optimize edilmesi gereken büyüklük hatadır. Literatürde kullanılan birçok PID performans indeksi bulunmaktadır. Mutlak hatanın integrali (IAE), zamanla çarpılmış mutlak hatanın integrali (ITAE), Hatanın karesinin integrali (ISE), Hatanın karesinin zamanla çarpılmış integrali (ITSE) en çok kullanılan performans indeksleridir. Burada dikkat edilmesi gereken en önemli noktalardan birisi kontrol işaretinin genliğidir. Hatanın hızlı sıfırlanması için yüksek genlikli bir kontrol işareti gerekir. Optimizasyon algoritması ile hesaplanan kontrol katsayıları gerçek zamanlı uygulamada üretilebilecek maksimum işaret genliğini aşıp, sistemin hesaplanandan farklı davranmasına sebep olabilir. Bu sebeple kontrolcü optimize edilirken kontrol işaretinin genliği de belirlenecek maliyet fonksiyonuna eklenmelidir.

Çalışmamızda PID kontrolcünün optimize edilmesi için pekiştirmeli öğrenme yöntemlerinden birisi olan Q-Öğrenme algoritması kullanılmıştır. Sistemin giriş-çıkış ilişkisi incelenerek ödül fonksiyonunu maksimum yapacak olan kontrol katsayıları algoritma tarafından belirlenecektir.

2.3. DC Motor Eşdeğer Devresi



Şekil 2.3. DA Motoru Eşdeğer Devresi

Şekil 2.3.'te DC motora ait olan eşdeğer devre verilmiştir. Verilmiş olan devreye göre sisteme ait olan matematiksel denklemler çıkartılacaktır.

2.3.1. Sisteme ait diferansiyel denklemler

$$V(t) = K * u(t) \tag{2.3}$$

Denklem (2.3)'te verilen $u(t)$, giriş sinyali; K, güç kuvvetlendirici kazancı; $V(t)$ ise motorun girişine uygulanan gerilimdir.

$$V(t) = R_a * i(t) + L_a * \frac{d\, i(t)}{dt} + \ e_a(t) \tag{2.4}$$

Denklem (2.4)'te verilen $i(t)$, akım; R_a, direnç; L_a, indüktans; $e_a(t)$ ise indüklenen ters elektromotor kuvvetidir.

$$e_a(t) = K_b * \frac{d\theta(t)}{dt} \tag{2.5}$$

Denklem (2.5)'te θ, motor milinin açısal yer değiştirmesi; K_b ise ters elektromotor kuvveti katsayısıdır.

$$T_e(t) = K_i * i(t) \tag{2.6}$$

Denklem (2.6)'da K_i, tork katsayısı; T_e ise elektriksel momenttir.

$$T_m(t) = J_m \ * \frac{d^2\theta(t)}{dt^2} + B_m * \frac{d\theta(t)}{dt} + \ T_y(t) \tag{2.7}$$

Denklem (2.7)'de J_m, atalet momenti; B_m, sürtünme katsayısı; T_y, yük momenti; T_m ise mekanik momenttir.

$$T_e(t) = \ T_m(t) \tag{2.8}$$

$$w(t) = \frac{d\theta(t)}{dt} \tag{2.9}$$

Denklem (2.9)'da $w(t)$, motor milinin açısal hızıdır.

2.3.2. Kontrol blok diyagramı

Sistemin dinamik davranışını ifade eden matematiksel denklemler Bölüm 2.3.1.'de verildi. Sistemin transfer fonksiyonunu elde etmek için Bölüm 2.3.1.'de verilen denklemleri Laplace dönüşümü yardımı ile s-domeninde aşağıdaki gibi elde ederiz.

$$V(s) = K * U(s) \tag{2.10}$$

$$V(s) = R_a \ * \ I(s) \ + L_a * s * I(s) + E_a(s) \tag{2.11}$$

$$E_a(s) = K_b * \ s * \ \theta(s) \tag{2.12}$$

$$T_e(s) = K_i * I(s) \tag{2.13}$$

$$T_m(s) = J_m \ * \ s^2 \ * \ \theta(s) \ + \ B_m * s * \ \theta(s) \ + \ T_y(s) \tag{2.14}$$

$$T_e(s) = \ T_m(s) \tag{2.15}$$

$$\Omega(s) = \ s \ * \ \theta(s) \tag{2.16}$$

DA motoru hız kontrolü sisteminin girişi, aynı zamanda tasarlanacak olan kontrolcünün çıkışı $u(t)$ işaretidir. Sistemin çıkışı ise hız kontrolü sistemi için w(t) işaretidir. Kontrolcü tasarımında bozucu etkisi sıfır alınır, dolayısıyla $T_y(s) = 0$ alınmalıdır. Sistemin açık çevrim transfer fonksiyonu $G(s)$ olmak üzere,

$$G(s) = \frac{Çıkışın\ Laplace\ Dönüşümü}{Girişin\ Laplace\ Dönüşümü} \tag{2.17}$$

Denklem (2.10) ve Denklem (2.12) denklemlerini Denklem (2.11)'de yerine yazıp düzenlersek Denklem (2.18)'i elde ederiz.

$$K * U(s) \ - s * Kb * \ \theta(s) = (R_a + s \ * L_a) * I(s) \tag{2.18}$$

Denklem (2.15) ve Denklem (2.16) yardımı ile Denklem (2.13) ve Denklem (2.14) düzenlenirse Denklem (2.19) elde edilir.

$$K_i * I(s) = (s * J_m + B_m) * \Omega(s) \tag{2.19}$$

Denklem (2.18) ve Denklem (2.19) düzenlenirse giriş-çıkış arasındaki bağıntı Denklem (2.20)'deki gibi elde edilir.

$$K * K_i * U(s) = \Omega(s) * (s^2 * (J_m * L_a) + s * (J_m * R_a + B_m * L_a) + B_m * R_a + K_b * K_i) \tag{2.20}$$

Denklem (2.20) kullanılarak sistemin transfer fonksiyonu Denklem (2.21)'deki gibi elde edilir.

$$G(s) = \frac{K*K_i}{s^2*(J_m*L_a)+s*(J_m*R_a+B_m*L_a)+B_m*R_a+K_b*K_i} \tag{2.21}$$

Sistemin doğası gereği elektriksel zaman sabiti τ_e , mekanik zaman sabiti olan τ_m 'den çok küçüktür. Bu sebeple biz kontrolcü tasarlarken Armatür endüktansı olan L_a'yı sıfır alabiliriz. Bu durumda sistemin transfer fonksiyonu olan $G(s)$, Denklem (2.22)'deki gibi elde edilir.

$$G(s) = \frac{\Omega(s)}{U(s)} = \frac{K* K_i}{s*(J_m*R_a)+ B_m*R_a+K_b*K_i} \tag{2.22}$$

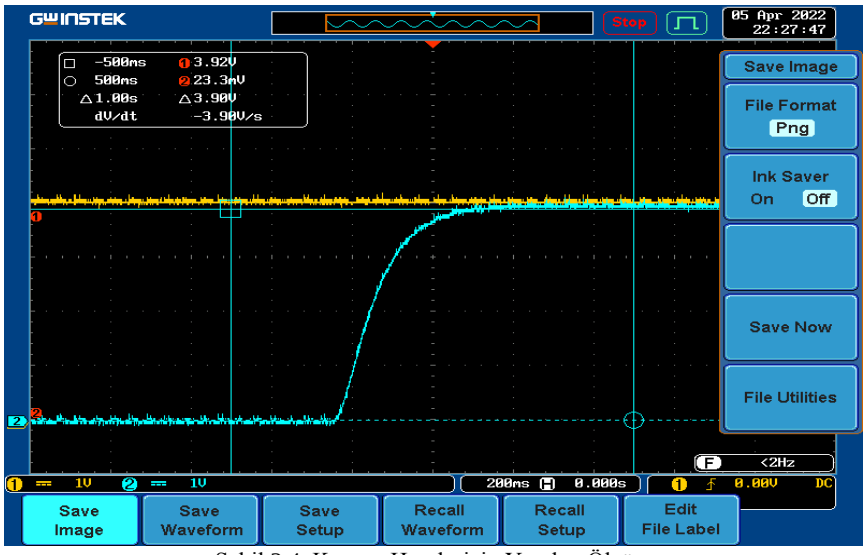
Birinci dereceden sistemin transfer fonksiyonu ise Denklem (2.23)'te verilmiştir.

$$G(s) = \frac{K}{\tau s+1} = \frac{\frac{K*K_i}{(B_m*R_a+K_b*K_i)}}{s*\frac{J_m*R_a}{(B_m*R_a+K_b*K_i)}+1} \tag{2.23}$$

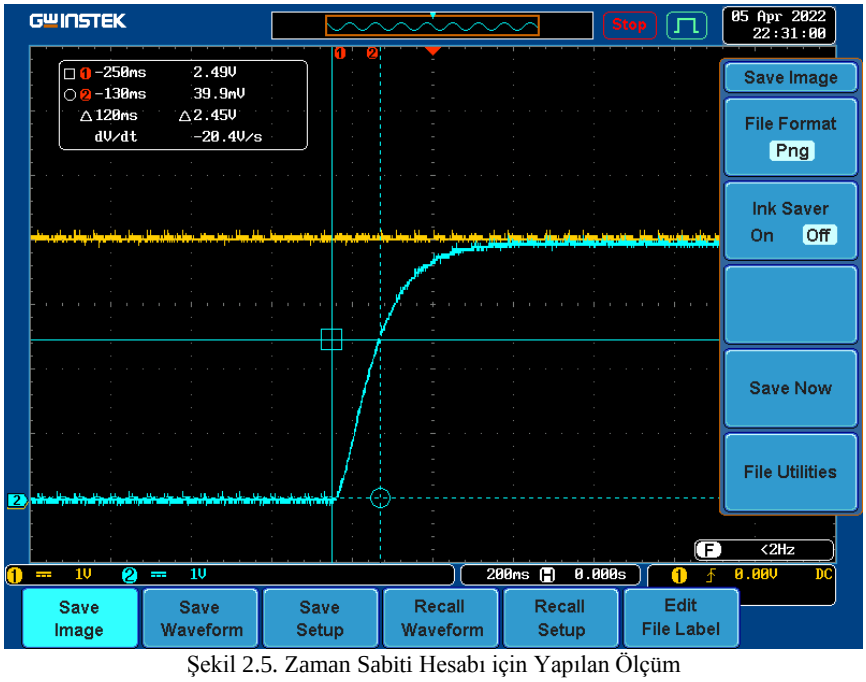
Açık çevrim transfer fonksiyonu için kazanç ve zaman sabiti Denklem (2.23)'teki gibi parametrik olarak elde edilir.

2.3.3. Sistem parametrelerinin elde edilmesi

Sistemi istenilen performansla kontrol edebilmek için sistemin matematik modeline tamamen hâkim olmamız gereklidir. Gerçek-zamanlı bir sistemin model parametrelerini net olarak elde edemeyebiliriz. Bu durumda sistem modelini elde edebilmek amacıyla deneysel çalışma yapılmalıdır. Bu çalışmada birinci dereceden bir sistem kullanılmaktadır, sistemin kazancı ve zaman sabiti deneysel olarak açık çevrim cevap incelenerek elde edilmiştir.



Şekil 2.4. Kazanç Hesabı için Yapılan Ölçüm



Şekil 2.5. Zaman Sabiti Hesabı için Yapılan Ölçüm

Osiloskop yardımı ile yapılan ölçümler incelendiğinde; Şekil 2.4.'te görüldüğü gibi giriş işaretinin genliği 4,05 Volt olduğu durumda çıkış işareti 3,98 Volt'a ulaşmaktadır. Bu durumda açık çevrim transfer fonksiyonunun kazancı;

$$K = \frac{3,90}{4,03} = 0,968 \tag{2.24}$$

Olarak hesap edilir. Şekil 2.5.'te görüldüğü gibi zaman sabitinin bulunması için çıkış işaretinin, son değerinin %63,2'sine ulaştığı süre bize, zaman sabiti τ'yu verir. Çıkışın %63,2'si ise;

$$0,632 * 3,9 = 2,45 \tag{2.25}$$

Denklem (2.25)'te hesaplanmıştır. Şekil 2.5.'te zaman sabiti hesabı için yapılan ölçümde çıkışın 2,45 Volt'a ulaştığı süre 120 milisaniye olarak ölçüldü.

$$\tau = 0,12 \tag{2.26}$$

Bu durumda sistemin açık-çevrim transfer fonksiyonu Denklem (2.27)'de elde edilmiştir.

$$G(s) = \frac{K}{\tau * s + 1} = \frac{0.968}{(0.12) * s + 1} \tag{2.27}$$

2.3.4. Örnekleme zamanı seçimi

Günümüzde ayrık zaman kontrolcü kullanımı standartlaşmış durumdadır. Sürekli zamanlı sistemleri ayrık zaman kontrolcüler ile kontrol edebilmek için sistemin giriş ve çıkışlarını belli frekanslarda örneklemek gerekir. Örneklenen işaretler ayrık zaman kontrolcü ile işlenip sistemin kontrolü gerçekleştirilir. Seçilecek olan örnekleme zamanı, ayrık zaman sistemin dinamiği üzerinde ciddi bir öneme sahiptir. Örnekleme zamanının değişimi sistemin bütün dinamiğini değiştirir.

Örnekleme zamanının seçiminde en önemli kriterlerden birisi sistemin zaman sabitidir. Hızlı dinamiğe sahip olan bir sistemi düşük frekanslarda örneklemek sistem hakkındaki birçok veriyi kaybetmemize sebep olur. Yavaş dinamiğe sahip olan bir sistemi yüksek frekanslarda örneklemek ise sistem hakkında kontrole etkisi olmayacak birçok veriyi gereksiz yere işlememize sebep olacaktır. Bir diğer önemli kriter ise kullanılacak olan ayrık zaman kontrolcünün işlem yapma hızıdır. Seçeceğimiz örnekleme periyodu içinde kontrolcünün veriyi okuyup, işleyip, kontrol işaretini üretmesi gerekmektedir. İşlemcinin hızı da örnekleme zamanı seçimindeki en önemli kriterlerden biridir.

Çalışmamızda, sistemin zaman sabitinin 120 milisaniye olduğu göz önünde bulundurularak örnekleme zamanı 10 milisaniye seçilmiştir. Bu durumda zaman sabiti içerisinde alınacak olan örnek sayısı 120 milisaniye boyunca 12 kere olacaktır. Deneysel çalışmada kullanılan mikroişlemci hızı da seçilen örnekleme zamanı için yeterlidir.

2.3.5. Ayrık zaman transfer fonksiyonu

Ayrık zaman kontrolcülerin kullanıldığı kontrol uygulamalarında, sistemin giriş ve çıkışları örneklenerek dijitalleştirilmelidir. Ayrık zaman işlemcilerin veri işleme hızları sınırlı olduğu için sürekli zaman işaretleri işlemesi söz konusu değildir. Günümüzde mikroişlemciler 480 MHz saat frekansı hız seviyelerine çıkmış durumdadır. Yüksek hızlı bir şekilde analog işaretleri ölçmek yapmak için özelleştirilmiş mikroişlemciler, yüksek hızlı ölçüm yapmak için geliştirilmiş ADC entegreleri sektörde oldukça yaygın kullanılmaktadır.

Fiziksel sistemlerin ayrık zaman kontrolcüler ile kontrol edildiği uygulamalarda sistemin ayrık zaman transfer fonksiyonunun elde edilmesi gerekir. Bunun için sürekli zaman sistemin matematiğine tamamen hâkim olmak gerekir. Sistemin sürekli zaman açık çevrim transfer fonksiyonunu elde ettikten sonra seçilecek olan örnekleme zamanına göre ayrık zaman transfer fonksiyonu elde edilir.

2.4. Kontrol Sistemlerinin Alt Bileşenleri

Fiziksel bir sistemi kontrol edebilmek için geliştirilmiş olan kontrol yöntemleri ancak sürekli veya ayrık işlemci mimarilerinde çalıştırılabilir. Matematiksel ifadelerin elektriksel işaretlere dönüştürülebileceği işlemciler en temel kontrol bileşenleridir. Sürekli zaman kontrolcüler işlemsel yükselteçler kullanılarak oluşturulurlar. Ayrık zaman kontrolcüler ise mikroişlemciler, DSP'ler, FPGA'ler, PLC'ler gibi matematiksel işlem yapabilme kabiliyetine sahip dijital devreler kullanılarak oluşturulurlar. Kontrolcülerin fiziksel sistemlerle uyumlu çalışabilmesi için birçok ara birime ihtiyacı vardır. Sensörler ve ölçme işlemine yardımcı devreler, analog-dijital ve dijital-analog dönüştürücüler, güç elektroniği devreleri mutlaka kullanılması gereken bileşenlerdendir.

2.4.1. Analog dijital dönüştürücü (ADC)

Sensörler yardımı ile elektriksel sinyallere dönüştürülmüş fiziksel büyüklükleri, dijital büyüklüklere dönüştüren elektronik devrelere analog dijital dönüştürücüler denir. Sürekli zaman işaretlerin ayrık zamana dönüştürülmesi için öncelikle işaretin periyodik olarak örneklenmesi gerekir. Örneklenen işaretlerin dijital büyüklüğe dönüştürülmesi için birçok yöntem geliştirilmiştir. Bütün yöntemlerde kabaca, girişteki gerilim seviyesinin ADC tarafından üretilen gerilim ile karşılaştırılarak dijital bir karşılık bulma yöntemine dayanır. Genelde ardışıl yaklaşım ve delta-sigma yöntemleri kullanır. ADC seçiminde en önemli faktörler çözünürlük, hız ve doğruluktur. Çözünürlük ve hız ters orantılıdır. Teknoloji gelişimi ile eş zamanlı olarak ADC'ler de gelişmeye devam etmektedirler.

Günümüzde pratikte kullanılan ADC'lerde delta-sigma tipi genelde hassas ölçüm yapmak için tercih edilirken ardışıl-yaklaşım ADC'ler genelde hızlı ölçüm yapmak için kullanılır. Periyodik olarak açılıp kapanan bir anahtar olarak modelleyebiliriz.

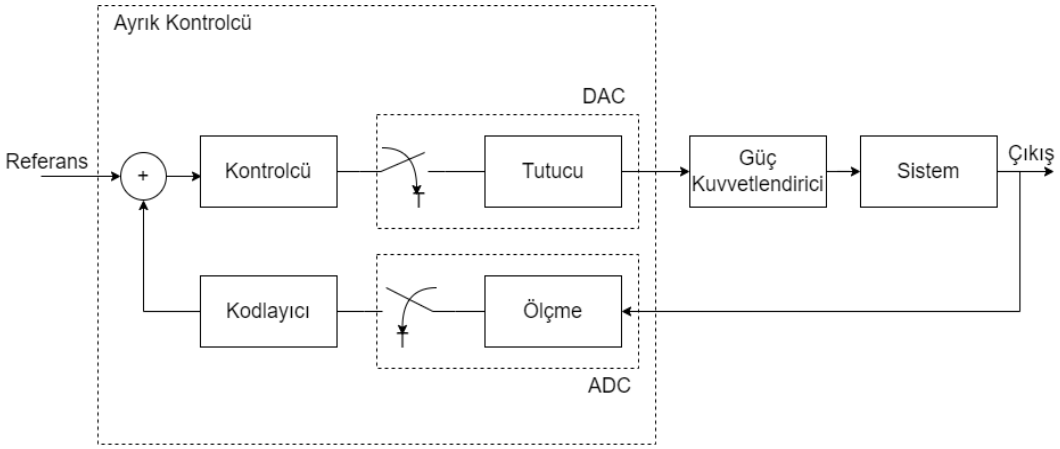
2.4.2. Dijital analog dönüştürücü (DAC)

Dijital işaretleri analog işaretlere dönüştüren elektronik devrelere dijital analog dönüştürücü denir. Çözünürlük ve hızları sınırlı olan bu devreler, ayrık kontrol işaretlerinin sürekli zaman sistemlere uygulanması için kullanılırlar. Bir adet anahtar ve sıfırıncı dereceden tutucu (ZOH - Zero Order Holder) ile modellenebilirler. Genelde sıfırıncı dereceden tutucular kullanılırken bazı uygulamalarda birinci dereceden tutucular (FOH) da kullanılmaktadır. DAC devreleri genelde ayrık zaman işlemcilerin mimarilerinde bulunurlar fakat uygulamaya göre entegre olarak da tercih edilebilirler. Ayrık zaman kontrol işlemlerinde en çok dikkat edilmesi gereken konulardan birisi de dijital analog dönüştürücünün transfer fonksiyonudur. Sürekli zaman bir sistemin ayrık zaman kontrolcü ile kontrol edildiği uygulamalarda, sürekli zaman sistemin transfer fonksiyonu ayrıklaştırılırken önüne bir de sıfırıncı dereceden tutucunun transfer fonksiyonu eklenir.

$$ZOH(s) = \frac{1 - e^{-sT}}{s} \tag{2.28}$$

Sıfırıncı dereceden tutucunun transfer fonksiyonu Denklem (2.28)'de verilmiştir.

2.4.3. Ayrık zaman kontrol blok diyagramı



Şekil 2.6. Ayrık-Zaman Kontrol Blok Diyagramı

Şekil 2.6.'da ayrık zaman kontrolcü ile sürekli zaman sistemin kontrolü uygulamasına ait blok diyagram verilmiştir. Şekilde görüldüğü üzere, kontrol uygulaması için referans ve sistemin geri beslemesini okumak için iki adet giriş, güç kuvvetlendirici üzerinden sisteme kontrol işareti uygulamak için bir adet çıkış bulunmaktadır

2.4.4. Ölçme sistemi

Sensörler yardımı ile fiziksel büyüklükler elektriksel büyüklüklere dönüştürülür. Elektriksel büyüklükler ise şartlandırıldıktan sonra ADC yardımı ile dijital büyüklüklere dönüştürülürler. Örnek olarak bir devrede akan akımı ölçmek istiyorsak onu gerilim cinsinden bir büyüklüğe dönüştürmemiz gerekir. Bu işlem için birçok yöntem geliştirilmiştir. Şönt direnç üzerinden ölçmek, akım trafosu yardımı ile ölçmek vb. Burada şartlandırma olarak belirtilen işlem, kuvvetlendirme, filtreleme, vb. işlemler olabilir. İşaret dijitalleştirildikten sonra da şartlandırma işlemleri yapılabilir. Ölçme yaparken tasarımcıların en büyük problemi işaret üzerine binen gürültülerdir. Hassas sistemlerde bu gürültülerin önüne geçmek çok pahalı ekipmanlar gerektirebilir. Gürültüler özellikle düşük genlikli işaretlerin ölçümünde çok ciddi hatalara sebep olabilir. Kimi zaman belirli frekanslarda olan gürültü kimi zaman da bütün frekans düzlemine yayılmış şekilde olabilir. Elektromanyetik etkiler, sıcaklık, topraklama hatası, donanımsal hatalar gürültülerin oluşmasının başlıca sebepleridir.

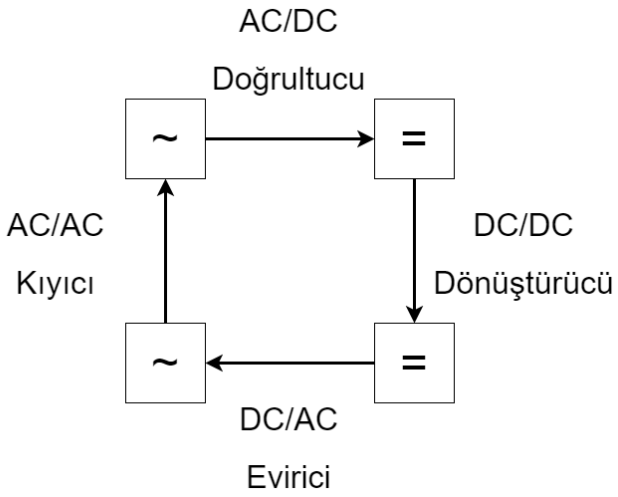
Ölçmenin düzgün gerçekleştirilmesi ve ölçme ile alakalı hiçbir problem kalmadığından her şeyden önce emin olunması gerekir. Kontrolcü tasarımı işlemine geçilmeden önce kullanılan yöntemler ile alakalı problemlerin giderilmesi, kontrolcü tasarlandıktan sonra oluşabilecek olan problemlerin sayısını ciddi şekilde azaltmamıza yardımcı olur.

2.4.5. Güç elektroniği devresi

Elektrik gücünün kontrol edilmesi veya dönüştürülmesi için geliştirilmiş uygulamalara denir. Kullandığımız kontrolcülerin güç seviyeleri fiziksel sistemleri kontrol edebilecek seviyede değildir. Özellikle ayrık zaman kontrolcülerde akım seviyesi

birkaç yüz mili amperden yüksek değildir. Kontrol edilecek olan fiziksel sistemler ise yüzlerce amper akım çekebilirler. Bu sebeple üretilen kontrol işaretlerinin güç elektroniği devreleri ile kuvvetlendirilip sisteme uygulanması gerekir.

Güç elektroniğinde kullanılan standart yapılar ve devre elemanları vardır. Güç elektroniği devreleri temel olarak 4 kategoride incelenir. Dönüştürücüler, doğrultucular, eviriciler ve kıyıcılar. Fiziksel sistemlerin kontrolünde bu yapıların bir veya daha fazlası güç katı olarak kullanılabileceği gibi; güç elektroniği devrelerinin de kontrolü ciddi bir çalışma alanıdır.



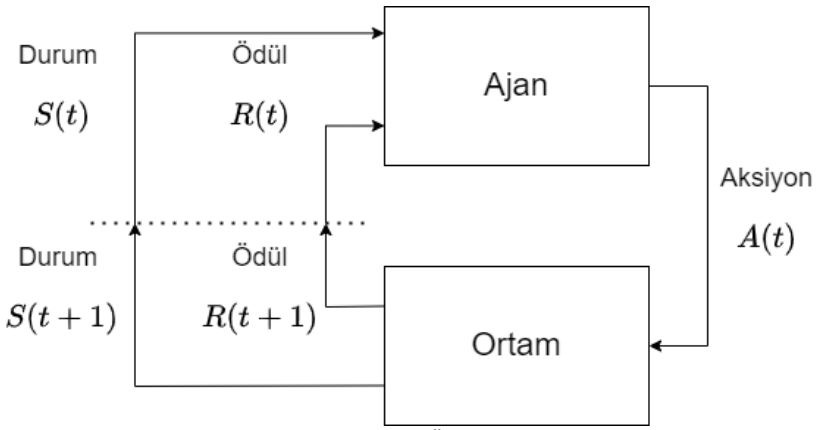
Şekil 2.7. Güç Elektroniği Yapıları

Güç elektroniği devrelerinde yapılacak işleme göre topoloji seçimi yapılır. Sistemin gücüne ve anahtarlama frekansına bağlı olarak kullanılacak olan devre elemanları seçilir. Transistör, mosfet, igbt, tristör, diyot, transformatör gibi devre elemanları sıkça kullanılmaktadır.

BÖLÜM 3. PEKİŞTİRMELİ ÖĞRENME

3.1. Pekiştirmeli Öğrenme

Günümüzün popüler araştırma konularından birisi olan pekiştirmeli öğrenme, makine öğrenmesi yöntemlerinden birisidir. Temel olarak, doğadaki canlıların öğrenme iç güdüsünü örnek alır. Çevresi ile direkt olarak sensörimotor bağı bulunan bir canlı herhangi bir öğretmene ihtiyaç duymadan, etki - tepki bağlantısı ile yürüme, koşma vb. karmaşık hareketleri öğrenebilmektedir, en basit tabirle bulunduğu durumlar ile seçeceği aksiyonları ilişkilendirmiştir [1]. Pekiştirmeli öğrenme yönteminde problemi öğrenecek olan ajana ortam hakkında bilgi verecek olan uygun sinyaller, gözlem sinyali olarak verilmelidir. Aynı zamanda alacağı aksiyonun uygunluğu ile ilgili bilgi verecek olan doğru ödül sinyalleri verilmelidir. Ajan, tasarımcı tarafından seçilecek olan algoritmayı kullanarak toplam ödül miktarını maksimize edecek olan aksiyonlar ile durumları ilişkilendirir.



Şekil 3.1. Pekiştirmeli Öğrenme Modeli

Pekiştirmeli öğrenme ajanı temel olarak;

- Ajanın hangi durumda hangi eylemi seçeceğini belirleyen politika fonksiyonu
- Bulunduğu durumun ne kadar iyi/kötü olduğunu belirleyen değer fonksiyonu
- Ajanın ortam tasviri, model.

Parçalarından en az birini içerir.

Pekiştirmeli öğrenme Markov Özelliğini sağlamaktadır. Ayrık zamanlı bir sistemde, sistemin $n+1$ anındaki durumu n anındaki durumuna bağlıdır. Sonlu durum ve sonlu aksiyon kümesine sahip olan bir ortam ile çalışılıyorsa uygulamaya Sonlu Markov Karar Verme Süreci denir [1]. Ajan, sistem ile üç temel işaret üzerinden etkileşime girer. Bunlar; ortamın durum sinyali, $s \in \mathcal{S}$; ortama uygulanan aksiyon sinyali, $a \in \mathcal{A}(\mathcal{S})$ ve ortamdan gelen skaler ödül sinyali, $r \in \mathcal{R}(s,a)$ olarak ifade edilir. Ödül sinyali ajanın n anındaki durumunun ve seçtiği aksiyonun ne kadar iyi olduğu bilgisini verir. Ajanın toplam ödül miktarı ise G_n ile ifade edilir. G_n ifadesi Denklem (3.1)'de verilmiştir.

$$G_n = R_{n+1} + \gamma R_{n+2} + \gamma^2 R_{n+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{n+k+1} \tag{3.1}$$

Denklem (3.1)'de verilen γ, $0 \leq \gamma \leq 1$ aralığında seçilen zayıflatma faktörüdür. Zayıflatma faktörü sıfıra ne kadar yakınsa ajan anlık ödüllere o kadar çok odaklanır. Genelde ajanların anlık ödüllere değil toplam ödüle odaklanması istenir. Ajan çalışma esnasında toplam beklenen ödül miktarını maksimize edecek olan aksiyonları durumlar ile eşleştirir. Bu eşleştirmeye politika, $\pi(a|s)$ denir. Ajan $\pi(a|s)$ politikası altında beklenen toplam ödül miktarına göre bir değer fonksiyonu $\mathcal{V}_\pi(s)$ oluşturur. Değer fonksiyonu, s durumunda ajanın beklediği toplam ödül miktarıdır. Ayrıca her durum için seçilebilecek olan aksiyonların değer fonksiyonuna aksiyon-değer fonksiyonu denir ve $Q(s\,,a)$ ile gösterilir.

$$\mathcal{V}_\pi(s) = E_\pi[G_n \mid S_n = s] \tag{3.2}$$

Denklem (3.2)'de verilen E_π stokastik sistemlerde beklenen değer anlamına gelmektedir. Denklem (3.2) ifadesinin eşiti Denklem (3.3)'tür.

$$\mathcal{V}_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r \mid s, a) \, [G_n \mid S_n = s] \tag{3.3}$$

Denklem (3.3)'te verilen p stokasik sistemlerde s durumundan s' durumuna geçiş olasılığıdır. Eğer Denklem (3.3)'ü düzenlersek Denklem (3.4)'ü elde ederiz.

$$\mathcal{V}_\pi(s) = E_\pi[\sum_{k=0}^{\infty} \gamma^k R_{n+k+1} \mid S_n = s] \tag{3.4}$$

Denklem (3.4)'te verilen ifadenin eşiti Denklem (3.5)'tir.

$$\mathcal{V}_\pi(s) = E_\pi[\, R_{n+1} + \gamma \sum_{k=0}^{\infty} \gamma^k R_{n+k+2} \mid S_n = s] \tag{3.5}$$

Denklem (3.5) düzenlenirse değer fonksiyonu, Denklem (3.6)'da verilmiştir.

$$\mathcal{V}_\pi(s) = E_\pi[R_{n+1} + \gamma \, \mathcal{V}_\pi(s') \mid S_n = s \,] \tag{3.6}$$

Denklem (3.6)'yı elde etmek için yapılan çıkarımların benzerleri yapılarak sistemin aksiyon-değer fonksiyonu, $Q(s, a)$ Denklem (3.7)'deki elde edilebilir.

$$Q(s, a) = E_\pi[R_{n+1} + \gamma Q(s', a') \mid S_n = s, \, A_n = a] \tag{3.7}$$

Ajanın görevi, toplam ödül miktarını maksimum yapacak optimal politika π_*'ı belirlemektir.

$$\mathcal{V}^*(s) = \max_\pi \mathcal{V}_\pi(s) \tag{3.8}$$

π_* politikası altında elde edilen optimal değer fonksiyonu, Denklem (3.8)'de verilmiştir.

$$Q^*(s) = \max_{\pi} Q(s,a) \tag{3.9}$$

Optimal aksiyon-değer fonksiyonu ise Denklem (3.9)'da verilmiştir. Pekiştirmeli öğrenme algoritmaları politika tabanlı, değer fonksiyonu tabanlı veya aktör-kritik tabanlı olabilir. Çalışmamızda kullanılan Q-Öğrenme algoritması değer fonksiyonu tabanlı bir yöntemdir. Değer fonksiyonunu kullanarak optimal aksiyonları seçer.

Pekiştirmeli öğrenme ajanlarının eğitim süreci bittikten sonra normal çalışma düzenine geçerler. Sistem tarafından ajana tamam sinyali gönderilene kadar ajan aksiyon seçmeye devam eder. Tamam sinyali gelene kadar geçen çalışmaya çevrim denir. Tamam sinyali sistem tarafından olumsuz bir durumu önlemek için gönderilebileceği gibi ajan görevini tamamladıktan sonra çalışmasını durdurması için de gönderilebilir.

Ajanlar sistemden belirlenen T_s, örnekleme periyotuyla bilgi almaktadırlar. Bu örnekleme periyotunun seçimi oldukça önemlidir. Ajan hem sistemin durumunu okumakta hem de sisteme aksiyonları uygulamaktadır. Sistemin zaman sabiti dikkate alınarak ve bilgisayar hesap hızı dikkate alınarak T_s belirlenmelidir.

3.1.1. Adaptif ε-Greedy

Ajan optimal bir politika belirleyebilmek için sistemin bütün durumlarını deneyimlemesi gerekir. Diğer yandan elde ettiği deneyimleri tekrardan tecrübe ederek değer fonksiyonlarını güncellemelidir. Bu sebeple ajan için bir keşif-istifade dengesi kurulmalıdır. Ajan eğitim sürecinin başında bütün durum ve aksiyonları keşfetmeye yönelik aksiyon seçerken; eğitim sürecinin sonuna doğru yüksek ödül verdiğini bildiği aksiyonları seçerek değer fonksiyonunu güncellemelidir. Bu keşif ve istifade dengesi sağlamak için Adaptif ε-Greedy yöntemi kullanılır. Bu algoritmaya göre seçilen rastgele bir sayı eğer ε sayısından büyükse rastgele bir aksiyon seçilir. Rastgele bir sayı eğer ε sayısından küçük ise maksimum ödülü verecek olan aksiyon seçilir.

$$\pi(a|s) = \begin{cases} Rastgele\ \ a \in \mathcal{A}(S), & eğer\ \xi < \varepsilon \\ \underset{a \in \mathcal{A}(s)}{\operatorname{argmax}}\, Q(s,a), & diğer \end{cases} \tag{3.10}$$

Adaptif ε-Greedy'nin matematiksel ifadesi Denklem (3.10)'da verilmiştir. Burada ε ve ξ, [0, 1] aralığındadır. ξ her örnekleme periyodunda rastgele sayı üreteci yardımı ile üretilir. ε değeri ise;

$$\varepsilon = \varepsilon * (1 - \varepsilon_{decay}) \tag{3.11}$$

Başlangıçta değeri 1'den başlatılır ve Denklem (3.11)'de verilen fonksiyon yardımı ile $\varepsilon_{min} = 0{,}005$ değerine kadar azaltılır. Ajanın seçeceği eylemlerin uzun vadeli etkilerinin olabileceği; ödül sinyalinin gecikmeli olabileceği, uzun vadedeki büyük ödül için kısa vadeli ödüllerden vazgeçilmesi gerekebileceği gibi aşılması gereken birtakım problemlerden dolayı keşif-istifade dengesinin doğru kurulmaması cevabın yakınsayamamasına sebep olabilir.

3.1.2. Markov karar süreci

Pekiştirmeli öğrenmede ajan, seçeceği aksiyonlara ortamdan gelen durum sinyallerinin bir fonksiyonu ile karar verir. Durum sinyali, sistem hakkında bilgi içeren bir sinyaldir. Sistemin tamamı hakkında bilgi içermesi gerekmediği gibi; sistem hakkında bilgi içeren herhangi bir sinyal, durum sinyali olabilir. Durum sinyali sistemin anlık ölçümlerinin yanında, sistemin geçmişteki tüm durumları hakkında bir bilgi içeriyorsa o sinyal Markov özelliğini sağlamaktadır. Markov özelliğini sağlayan sistemlerde sistemin bir sonraki durumu sadece o anki durumuna bağlıdır [1], [27].

Örnek olarak;

$$Pr_n\{R_{n+1} = r,\ S_n = s' \mid S_0,\ A_0,\ S_1,\ A_1, \ldots S_n,\ A_n\} \tag{3.12}$$

Denklem (3.12)'de verilen ifadenin eşiti Denklem (3.13)'te verilmiştir.

$$Pr_n\{R_{n+1} = r,\ S_{n+1} = s' \mid S_n,\ A_n\} \tag{3.13}$$

$Pr_n\{.\}$ ifadesi sonlu markov zinciri için S_n durumundan S_{n+1} durumuna geçiş olasılığıdır. Her durum için bir bu olasılık dinamik kontrolcü tarafından bir önceki durumun olasılığı $Pr_{n-1}\{.\}$ Baz alınarak hesaplanmaktadır [3]. Yani, t anındaki S_t ve R_t, Daha önceki tüm S ve R 'leri kapsamaktadır. Markov özelliğini sağlayan pekiştirmeli öğrenme işlemine markov karar verme süreci denir. Eğer ortamın durum ve eylem uzayı sonlu ise sürece sonlu Markov Karar Süreci denir. Tek ajanlı pekiştirmeli öğrenme için en uygun model Markov Karar Sürecidir [1], [3], [27], [28].

3.2. Optimal Değer Fonksiyonları

Pekiştirmeli öğrenmede iki çeşit değer fonksiyonu vardır. Bunlar, durum-değer fonksiyonu $\mathcal{V}_\pi(s)$ ve aksiyon değer fonksiyonu $Q(s,a)$ dır.

$$Q(s,a) = R(s,a) + \gamma Q(s',a') \tag{3.14}$$

Deterministik bir sistemde aksiyon-değer fonksiyonu Denklem (3.14)'te verilmiştir. Optimal Q fonksiyonu Q^*, π politikası altındaki en iyi aksiyon-değer fonksiyonu olarak ifade edilir ve Denklem (3.15)'te verilmiştir.

$$Q^*(s,a) = \max_\pi Q(s,a) \tag{3.15}$$

Benzer şekilde optimal durum-değer fonksiyonu Denklem (3.16)'da verilmiştir.

$$V_\pi^*(s,a) = \max_\pi V_\pi(s) \tag{3.16}$$

Optimal politika $\pi^*(s,a)$, her durum için en iyi $Q(s,a)$ 'yı seçer.

$$\pi^*(s,a) \in \operatorname*{argmax}_a Q^*(s,a) \tag{3.17}$$

Sonlu Markov karar süreci için her zaman $\max_\pi Q(s,a) \geq \max_{\pi'} Q(s,a)$ veya $\mathcal{V}_\pi(s) \geq \mathcal{V}_{\pi'}(s)$'i maksimum yapacak bir optimal politika mutlaka yazılabilir. Elimizde sisteme

ait model dahi olsa optimal politikayı elde etmek uygulamada ciddi hesap yükü getirecektir. Bu nedenle optimal bir politika bulmaktansa; iteratif yöntemler ile optimal politikaya yaklaşım yaparak sonuç elde edilebilir. İteratif yöntemler ile $\pi^*(s,a)$ Bellman Optimallik Fonksiyonu [29] çözülerek elde edilir.

$$Q^*(s,a) = R(s,a) + \gamma \max_{a'} Q^*(s',a') \tag{3.18}$$

Bellman Optimallik Fonksiyonu Denklem (3.18)'de verilmiştir. Başta keyfi seçilen bir aksiyon-değer fonksiyonu ile iterasyona başlanır. Ajan, her adımda değer fonksiyonunu günceller ve optimal politikaya yaklaşır [1], [27], [29], [30].

3.3. Dinamik Programlama

Dinamik programlama yöntemi ile modeli bütünüyle bilinen bir sistemin optimal değer fonksiyonu hesaplanabilir. Sistemin bütün durumları ve durumlar arasındaki geçiş matrisleri biliniyorsa her durum için toplam değer fonksiyonu hesaplanarak iteratif yöntemler ile optimal değer fonksiyonu veya politika elde edilebilir. Diğer bir ifade ile; herhangi bir n anında, S_n durumu için seçilebilecek aksiyonlar ve aksiyonlar arasındaki geçiş matrisleri $p(s',r|s,a)$ biliniyorsa optimal politika Denklem (3.19) ve Denklem (3.20) denklemleri çözülerek elde edilebilir.

$$V_\pi(s) = E_\pi[R_{n+1}, \gamma V_\pi(S_n)] \tag{3.19}$$

$$V_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma V_\pi(s')] \tag{3.20}$$

Dinamik programlamanın sisteme ait olan modeli bilinmesini gerektirdiği için gerçek-zamanlı bir sistemde optimal politika bulmakta teorik olarak güzel bir altyapı sağlasa da uygulamada iş göremeyebilir. Bu sebeple pekiştirmeli öğrenmede model kestirimi ve modelden bağımsız kontrol yöntemleri daha yaygın kullanılmaktadır [1], [3]–[5], [27].

3.4. Model Kestirimi

Pekiştirmeli öğrenmede modelden bağımsız kontrol yapabilen algoritmalar Monte-Carlo yöntemi ve Zamansal-Fark yöntemidir. İki yöntemde de algoritma, sistemin modelini bilmemektedir. Tecrübeler yardımı ile sistemin modelini öğrenmekte ve aynı zamanda optimal değer fonksiyonunu elde etmeye çalışmaktadır. Her iki yöntemde de başta toplam ödül miktarı sıfırdır. Keyfi olarak seçilen bir politika ile aksiyonlar alan ajan değer fonksiyonunu her iterasyonda güncelleyerek optimale yaklaşmaktadır. Monte-Carlo yönteminde bir çevrim boyunca çalışan ajan, çevrim bitince değer fonksiyonunu güncellemektedir. Zamansal-Fark yönteminde ise ajan, çevrim boyunca her durum geçişinde değer fonksiyonunu güncellemektedir. Sonlu Markov Özelliğini sağlayan sistemler için optimal politikaya her iki yöntem de ulaşabilmektedir. Zamansal-Fark yöntemi Monte-Carlo yöntemine göre daha hızlı yakınsamaktadır. Monte-Carlo yönteminde Denklem (3.21) ifadesi kullanılarak değer fonksiyonları güncellenir.

$$V(S_n) \leftarrow V(S_n) + \alpha(G_n - V(S_n)) \tag{3.21}$$

Zamansal-Fark yönteminde ise Denklem (3.22)'de verilen ifade kullanılır.

$$V(S_n) \leftarrow V(S_n) + \alpha(R_{n+1} + \gamma V(S_{n+1}) - V(S_n)) \tag{3.22}$$

Çalışmamızda bir Zamansal-Fark yöntemi olan Q-Öğrenme algoritması kullanılmaktadır.

3.5. Q-Öğrenme Algoritması

Q-Öğrenme, bir Zamansal-Fark yöntemidir. Modelden bağımsızdır. Q-Öğrenme algoritmasında optimal politika bir Q-tablosu yardımı ile oluşturulur. Sisteme ait olan her durum; $\forall s \in \mathcal{S}$ ve sistemin her durumunda seçilebilecek aksiyonları; $\forall a \in \mathcal{A}(\mathcal{S})$ içeren bir Q-tablosu oluşturmaktadır. Q-tablosu her iterasyonda güncellenerek optimal bir Q-tablosu oluşturulur. Optimal Q-tablosu oluştuktan sonra bu tablo yardımı ile her

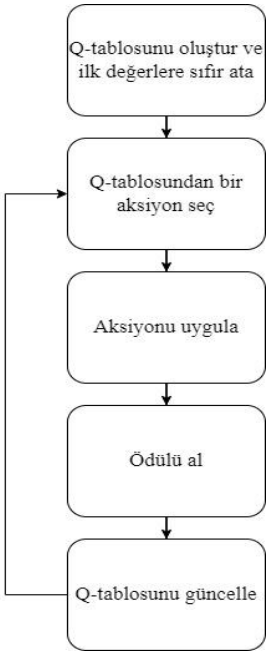
bir durum için kümülatif ödül miktarını en yüksek yapacak olan aksiyonların seçilmesini sağlamaktır.

$$Q(s,a) = [R_{n+1} + \gamma Q(s',a') \mid S_n = s,\ A_n = a] \tag{3.23}$$

Deterministik bir sistem için aksiyon değer fonksiyonu Denklem (3.23)'te verilmiştir. İlk değerleri sıfır olarak atanan $Q(s,a)$ tablosu, her iterasyonda s ve a çiftleri için güncellenerek optimal Q-tablosu oluşturulur. Optimal tabloyu oluşturmak için ajanın bütün durumları ve aksiyonlar deneyimlemesi; deneyimlediği her durumu ve aksiyonu da tekrar tekrar deneyerek değer fonksiyonunu güncellemesi gerekir.
Ajan Q-tablosunu Denklem (3.24)'te verilen fonksiyon ile güncellemektedir.

$$Q(s,a) \leftarrow Q(s,a) + \alpha[R + \gamma(Q(s',a') - Q(s,a))] \tag{3.24}$$

Denklem (3.24)'te verilmiş olan α öğrenme faktörüdür. Öğrenme faktörü sıfıra ne kadar yakınsa ajan o kadar uzun süre sonunda optimal politikaya ulaşır. Öğrenme faktörü eğer yüksek seçilirse ajan yerel bir maksimum noktasına yakınsayabilir. Bu sebeple α sıfıra yakın bir değer seçilmelidir. Şekil 3.2.'de Q-Öğrenme diyagramı verilmiştir.



Şekil 3.2. Q-Öğrenme Diyagramı

Literatürde Q-Öğrenme ile yapılmış kontrol uygulamaları bulunmaktadır. Bu uygulamalarda fiziksel sistemlerde Q-Öğrenme ile kontrol işlemi gerçekleştirilmiş ve başarılı sonuçlar alınmıştır [7]–[10]. Örnek Q-tablosu Tablo 3.1.'de verilmiştir.

Tablo 3.1. Örnek Q-Tablosu

	A_0	A_1	A_2	A_3	
S_0	$Q(S_0, A_0)$	$Q(S_0, A_1)$	$Q(S_0, A_2)$	$Q(S_0, A_3)$	
S_1	$Q(S_1, A_0)$	$Q(S_1, A_1)$	$Q(S_1, A_2)$	$Q(S_1, A_3)$	
S_2	$Q(S_2, A_0)$	$Q(S_2, A_1)$	$Q(S_2, A_2)$	$Q(S_2, A_3)$	…
S_3	$Q(S_3, A_0)$	$Q(S_3, A_1)$	$Q(S_3, A_2)$	$Q(S_3, A_3)$	
		…			

3.6. Derin Q-Öğrenme

Yüksek sayıda durum ve aksiyon içeren sistemlerin kontrolünde Q-Öğrenme algoritmasının optimal bir politikaya yakınsaması oldukça zordur. 2015 yılında yapılan bir çalışmada Derin öğrenme yöntemleri ve Q-Öğrenme birleştirilerek çok fazla durum ve aksiyon içeren sistemlerin kontrolü başarılı bir şekilde gerçekleştirilmiştir [7]. Yapılan çalışmada video oyunlarında; konvolüsyonal sinir ağları kullanılarak ekrandan alınan pikseller üzerinden durum bilgisi gözlemlenmiştir. Alınan durum bilgisi ile seçilecek olan aksiyonlar arasındaki politika, derin öğrenme yöntemi ile oluşturulmuş ve bir joystick ile aksiyonlar video oyununa iletilmiştir.

Birden çok sistem ve aksiyon içeren kontrol uygulamalarında derin Q-Öğrenmenin başarılı sonuçlar verecektir.

3.7. Q-Öğrenme Tabanlı Kontrol

Pekiştirmeli öğrenme yöntemleri günümüzde birçok alanda uygulanmaktadır. Fiziksel sistemleri optimal kontrol etmek için de pekiştirmeli öğrenme kullanılmaktadır. Fiziksel sistemleri kontrol edebilmek için sistemden durum bilgisi gözlemlenmeli, skaler ödül sinyali alınmalı ve sisteme bir aksiyon sinyali verilmelidir. Fiziksel sistemlerde durumlar ve aksiyonlar sürekli-zamandır ve çok fazla durum ve aksiyon olabilir. Sistemin durum ve aksiyon sayısı arttıkça Q-Öğrenmenin optimal bir

politikaya yakınsaması oldukça zorlaşacaktır. Bu sebeple ayrık gruplama veya nöral ağlar kullanılarak uygulama gerçekleştirilebilir. Pekiştirmeli öğrenme yöntemleri doğrudan kontrol işareti üreterek sistemleri kontrol edebileceği gibi klasik veya modern kontrol yöntemleri ile kullanılarak kontrolcü katsayılarını ayarlayabilir. Çalışmamızda PID kontrolcü parametreleri Q-tabloları ile arttırılıp azaltılarak optimal kontrolcü katsayılarına ulaşmak hedeflenmiştir. Her bir katsayı için bir ajan atanabileceği gibi; bir ajan ve bir Q-tablosu PID katsayılarını ayarlayabilir. Birden çok ajan kullanılan uygulamalarda ajanlar aynı ödül ve durum bilgisini alabilecekleri gibi farklı ödül ve durumlar için ajanlar benzer ortamı kontrol edebilir. PID katsayıları için aksiyonlar; arttırmak, azaltmak ve değiştirmemek şeklindedir. Ne kadar çok aksiyon varsa optimal politikaya yaklaşma süresi artacaktır.

Pekiştirmeli öğrenme kontrolcüsünün PID kontrolcü katsayılarını hangi değerde arttırıp azaltacağı oldukça önemlidir. Katsayıları büyük bir sayı ile toplayıp çıkartmak daha hızlı ayar yapılmasını sağlar fakat kararsız bir davranış oluşturabilir. Katsayıları küçük bir sayı ile toplayıp çıkartmak optimal kontrolcü katsayılarına yakınsamasını kolaylaştırır fakat sistemin davranışını yavaşlatır. Bir büyük, bir küçük katsayı ile toplanıp çıkarılırsa hem hız olarak hem de yakınsama olarak daha iyi bir davranış oluşur fakat bu seferde aksiyon sayısı artar ve eğitim süreci uzar. Her katsayı farklı bir sayı ile toplanıp çıkartılabilir. Çalışmamızda oransal katsayı için 0,1, integratör katsayısı için 0,01 , türevsel katsayı için 0,001 sayıları ile PID kontrolcü ayarı yapılmaktadır.

Pekiştirmeli öğrenme örnekleme zamanı çalışmamız için 10 milisaniye seçilmiştir. Bu çalışmada DA motoru için kullanılacak olan bir kontrolcü tasarlanacaktır. DA motoru zaman sabiti deneysel yöntem ile 120 milisaniye olarak elde edilmişti. Q-Öğrenme algoritmasının çalıştırılması için yeterli olan ve sistemin zaman sabitinin yaklaşık 10'da birine yakın bir değer olan 10 milisaniyelik periyot değeri çalışmamız için uygundur.

Çalışmamızda sistemin durumu hatanın değeri ve hatanın değişim yönü ile belirlenmektedir. Kontrol sistemlerinde hata; referans ile kontrol edilmek istenen

büyüklük arasındaki farktır. Kontrol edilecek olan sistemimizde hata 5 Volt ile -5 Volt arasında değişebilmektedir. $[-5,\ 5]$ Aralığı çalışmamızda ayrık gruplama ile 100 eşit parçaya bölünmüştür; bu sayı aynı zamanda kontrol edilecek olan sistemden gelen durum sayısı ile doğrudan ilişkilidir. Ayrık grup sayısının artması Q-tablosunun satır sayısının artmasına dolayısıyla optimal politikaya yakınsama süresinin uzamasına sebep olmaktadır. Daha iyi bir performans için düzgün ayarlanması gereken bir parametredir.

Diğer durum sinyali ise hatanın değişim yönüdür. Hatanın mutlak değerinin değişimi hangi yönde olduğu bilgisi de sistemin durumu hakkında kontrolcüye bilgi vermektedir. Denklem (3.25)'te durum fonksiyonunun ifadesi verilmiştir.

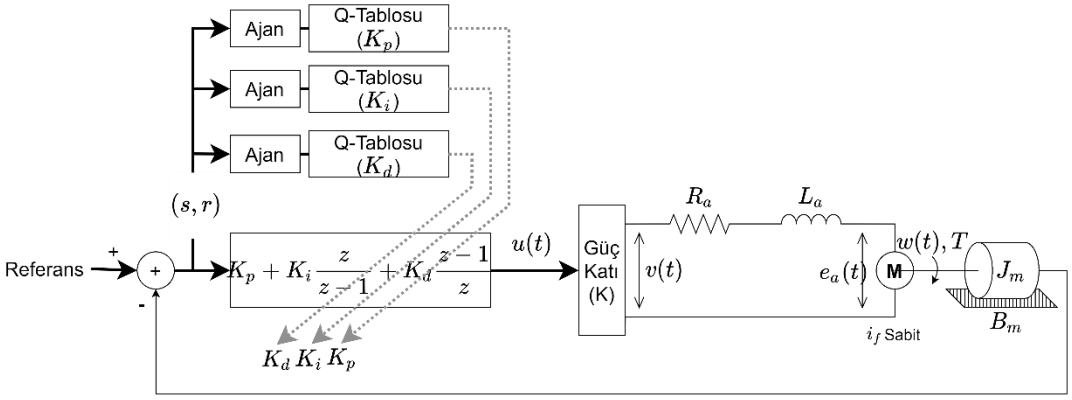
$$s \in \mathcal{S}_n = [f(e_n),\ f(|e_n - e_{n-1}|)] \tag{3.25}$$

Ajanın sistemden aldığı ödül ise yine hatanın bir fonksiyonudur. Her örnekleme periyodunda ajan sistemden aldığı ödül ile ne kadar iyi olduğu hakkında bilgi sahibi olur. Hatanın mutlak değeri azalma yönünde ise ajan +1 ödül almaktadır. Hatanın mutlak değeri artma yönünde ise -1 ödül almaktadır. Eğer hata 0'a eşitse ajan +1 ödül daha almaktadır. Ajan sistemden aldığı anlık skaler ödüller yardımı ile değer fonksiyonu oluşturmakta ve optimal politikayı aramaktadır. Denklem (3.26)'da ödül fonksiyonunun matematiksel ifadesi yer almaktadır.

$$R(s,a) = \begin{cases} 1, & e_n = 0 \\ 0.1, & sign(|e_n| - |e_{n-1}|) = -1 \\ -0.1, & sign(|e_n| - |e_{n-1}|) = 1 \end{cases} \tag{3.26}$$

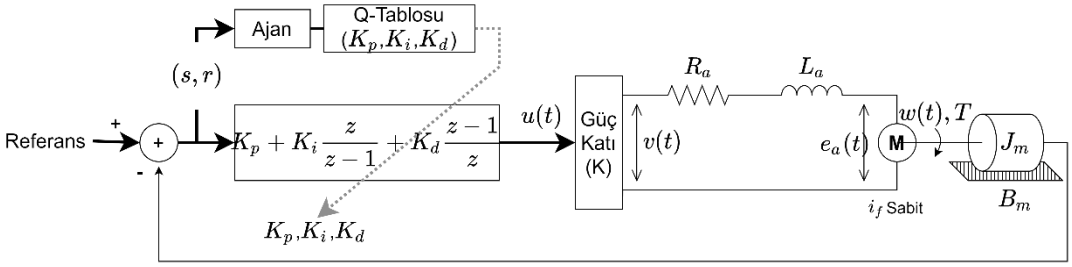
Optimal kontrolcü, tasarımcının belirlediği ödül fonksiyonuna göre ayarlanmaktadır. Fiziksel bir büyüklüğün kontrolü için yazılacak ödül sinyali; aşım, yerleşme zamanı, PID performans kriterleri (IAE, ISE, ITAE, vb.), hatanın değişim yönü vb. değerler olabilir. Fiziksel bir sistemden alınabilecek olan bilgiler ve fiziksel sistemlerin kısıtları göz önünde bulundurularak tasarımcı tarafından ödül sinyali seçilmelidir.

Çalışmamızda hem birden çok ajan ile hem de bir ajan ile Q-Öğrenme çalışması yapılmıştır. Birden çok ajan olduğu durumda her ajan bir katsayının kontrolünü yapar. Bir ajan oransal kontrolcü için ayarlama, diğer bir tanesi integratör katsayısı için ve diğeri de türevsel kontrolcü için ayarlama yapar. Bu ajanlar birbirlerinden bağımsızdır ancak ortamdan gelen durum ve ödül sinyali her ajan için de aynıdır. Bu yapının avantajı ajanların aksiyon sayısının 3'e inmesidir. Bu yapının dezavantajı ise ajan sayısı ve dolayısı ile işlem sayısı artmaktadır. Eğer eğitim süreci düzgün ayarlanmazsa ajanların bir tanesi bile optimal bir politika belirlememişse bu yapı global bir optimal bulmak yerinde yerel bir optimum bulmuş olur.



Şekil 3.3. Üç Ajanlı Q-Öğrenme Tabanlı PID

Tek bir ajanın bulunduğu uygulamada ise bir adet ajan ve bir adet Q-tablosu bulunmaktadır. Bu durumda ajanımızın aksiyon sayısı artmaktadır. PID katsayıları için toplam aksiyon sayısı 7'dir. Ajan aldığı durum bilgisine göre toplamda 7 aksiyona göre Q-tablosu oluşturmaktadır. Bu yöntemin avantajı tek bir ajan eğitildiği için global optimal noktasına yakınsama garantilidir fakat eğitim daha uzun sürmektedir.



Şekil 3.4. Tek Ajanlı Q-Öğrenme Tabanlı PID

Q-Öğrenme algoritması, endüstride sıkça kullanılan mikrodenetleyicilere yazılımsal olarak gömülebilir. Ayrık kontrolcü her örnekleme periyodunda algoritmayı çalıştırarak PID katsayılarını ayarlayabilir. Bu çalışmada Q-Öğrenme algoritması MATLAB & Simulink ortamında çalıştırılmış ve seri haberleşme ile mikrodenetleyici ile veri akışı gerçekleştirilmiştir. MATLAB & Simulink ortamında ajanlar oluşturulmuş ve hem eğitim sürecinde hem de çalışma sürecinde mikrodenetleyici ile paralel olarak çalıştırılmıştır. Mikrodenetleyici bilgisayara hata değerini her örnekleme periyodunda göndermiştir. Bilgisayar, gelen hata değerine göre ödül sinyalini ve sistemin durumunu kestirmiştir. MATLAB ise aksiyon sinyalini mikrodenetleyiciye ileterek sistemi kumanda etmiştir.

3.8. Ayrık Gruplama

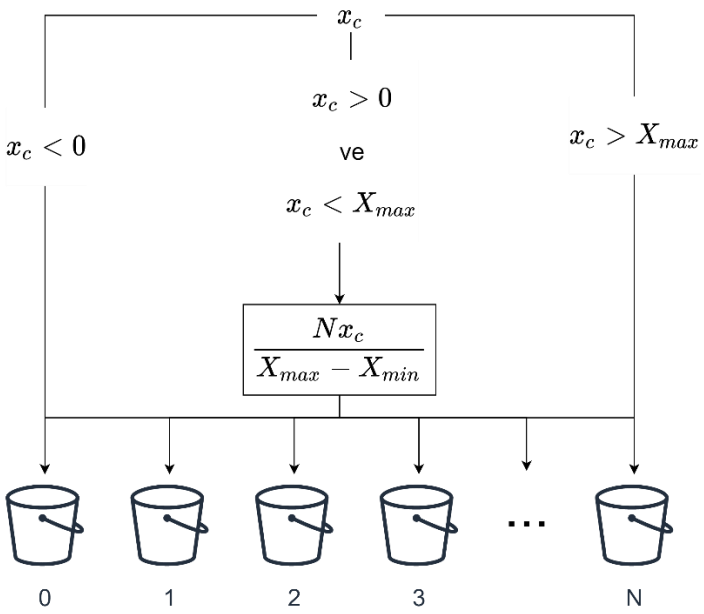
Eğitim süreci boyunca ajan, bütün durumları deneyimler ve her durumda seçebileceği bütün aksiyonları teker teker deneyerek değer fonksiyonunu günceller. Bu sebeple çok fazla durum ve aksiyon içeren sistemlerin kontrolünde Ayrık Gruplama yöntemi kullanılmalıdır. Fiziksel sistemlerde durumlar sürekli işaret formundadır. Sürekli işaret formundaki sinyalleri Q-Öğrenme ile işleyebilmek için ayrık gruplama yöntemi kullanılmaktadır. Sürekli-sinyallerin ayrık-sinyallere dönüştürüldüğü kuantalama ile benzer işlemdir fakat burada asıl amaç, en az ayrıklaştırma kadar da durum sayısını azaltmaktır.

Ayrık gruplama sistemden okunan verileri; belirli bir hata toleransı içinde gruplara ayırarak, Q-Öğrenme algoritmasının işleyebileceği forma dönüştürülmesini sağlar.

$$x_d = \begin{cases} 0 & x_c < X_{min} \\ N & x_c > X_{max} \\ \frac{x_c * N}{X_{max} - X_{min}} & X_{min} \leq x_c \leq X_{max} \end{cases} \tag{3.27}$$

Denklem (3.27)'de verilmiş olan N, grup sayısı; x_c, sürekli formdaki işaret; X_{max} ve X_{min}, grup sayısının maksimum ve minimum değeri; x_d ise x_c değişkeninin ayrık gruplanmış değeridir.

Bu teknik sayesinde sürekli sinyal olan durum bilgisi Q-Öğrenme ajanı tarafından kullanılabilir hale getirilir. Şekil 3.5.'te Ayrık Gruplama yönteminin şeması verilmiştir.



Şekil 3.5. Ayrık Gruplama Yöntemi

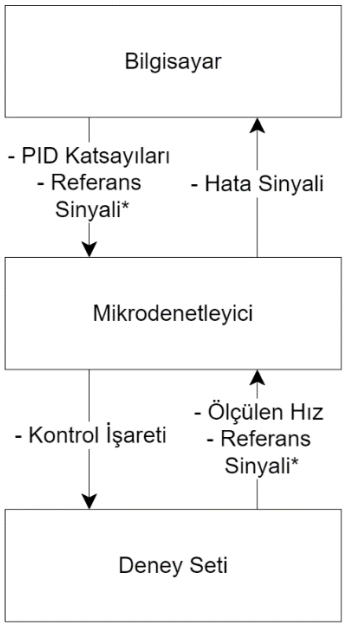
3.9. Deney Düzeneği

Çalışmamız hem benzetim ortamında hem de gerçek-zamanlı olarak çalıştırılarak elde edilen sonuçlar incelenmiştir. Gerçek-zamanlı sistemin çalıştırılması için Q-Öğrenme algoritması bilgisayar ortamında hazırlanmıştır. Kontrol edilecek olan sistem olarak DA motoru hız kontrolü deney seti seçilmiştir. Deney seti bir adet analog sürücü kartı ve bir adet kontrol kartı yardımı ile çalıştırılmaktadır. Kontrol kartı üzerinde ADUC841 mikrodenetleyicisi bulunmaktadır. ADUC841 mikrodenetleyicisi üzerindeki 2 adet 12 bit ADC yardımı ile deney setinden referans hız ve motor hızı ölçülebilmektedir. Kontrol kartında bulunan DAC yardımı ile analog sürücü kartına kontrol işareti uygulanmaktadır. Sürücü kartı, kontrol kartından gelen kontrol işaretini şartlandırarak motor sürücüsüne iletmektedir.

ADUC841 mikrodenetleyicisi seri kanaldan bilgisayar ile haberleşmektedir. Her örnekleme periyodunda bilgisayara hata değerini göndermektedir ve PID katsayılarını

bilgisayardan almaktadır. Bilgisayarda çalışmakta olan Simulink programı mikrodenetleyiciden gelen hata değerini almakta; Durum ve ödül fonksiyonlarını elde ederek Q-tablolarına göre PID katsayılarını güncellemekte ve seri kanaldan mikrodenetleyiciye iletmektedir. Şekil 3.6.'da Deney düzeneğinin blok diyagramı verilmiştir.

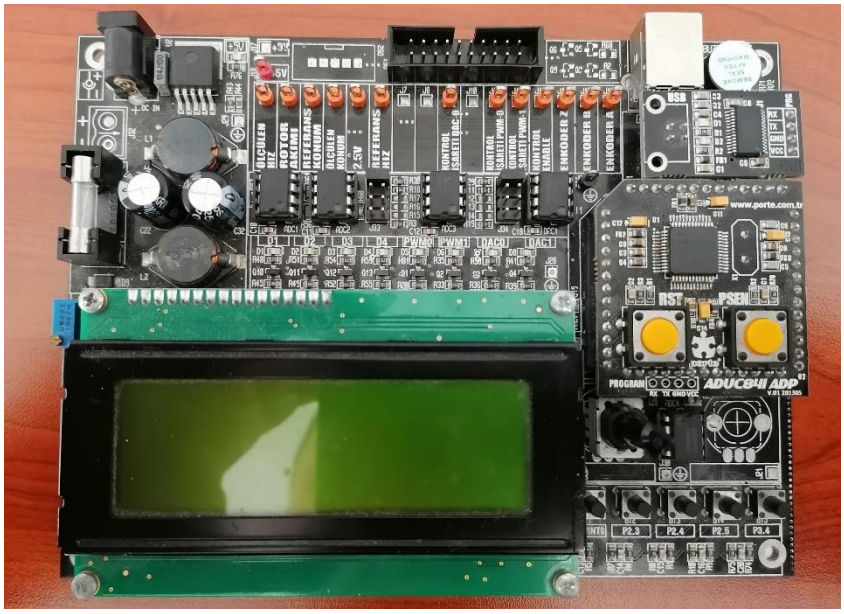
DA motoru hız kontrolü sistemine hız referansı bilgisayardan gönderilebileceği gibi deney seti üzerindeki referans hız potansiyometresi ile de verilebilmektedir. Ajan eğitim süreci boyunca farklı referanslar verilerek eğitilmektedir.



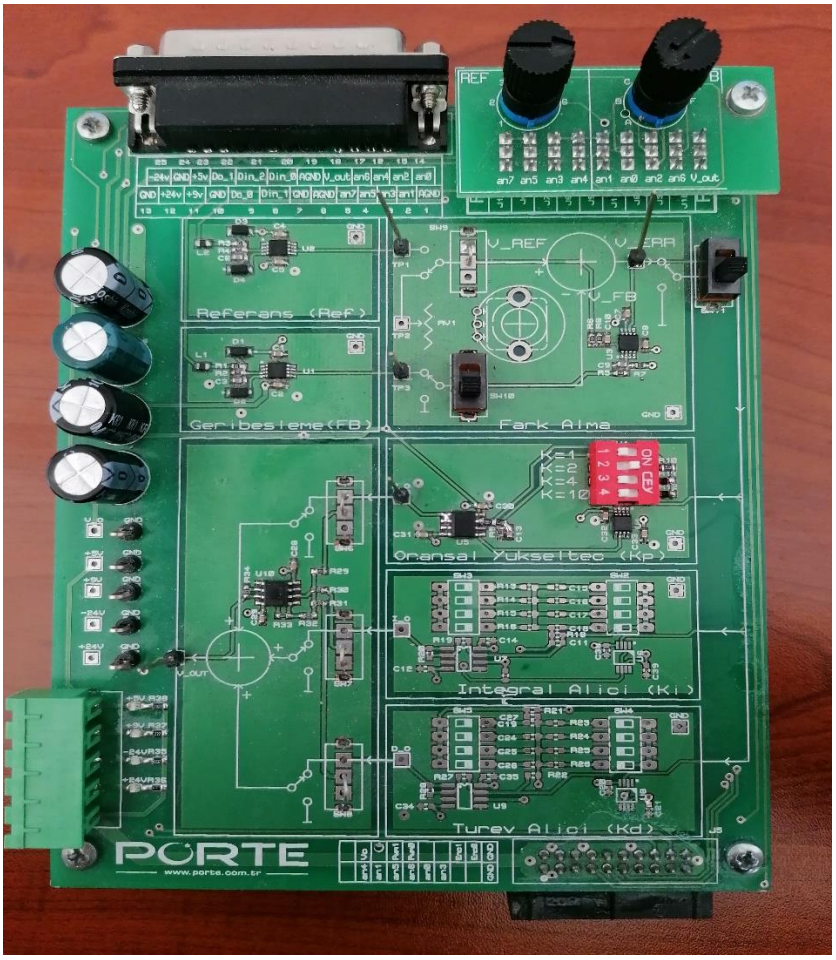
* Referans sinyali bilgisayardan da gönderilebilir, Deney seti üzerindeki potansiyometre ile de ayarlanabilir

Şekil 3.6. Deney Düzeneği Blok Diyagramı

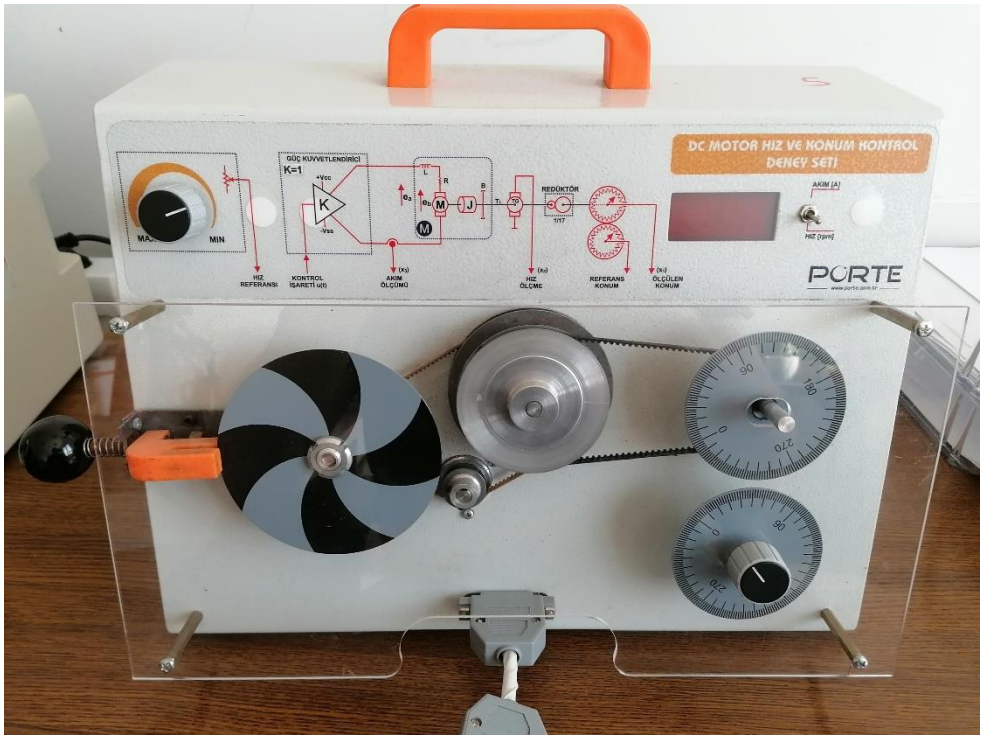
Eğitim süreci boyunca eğer sabit bir referans seçilirse ve seçilen referans değer düşük bir değerse ajanın deneyimleyebileceği durum miktarı kısıtlanır. Bu sebeple sabit referans verilirken verilebilecek maksimum referansa yakın bir değer seçilmelidir. Rastgele referans da verilse, sabit referans da verilse ajan kontrol işlemini yapabilmeyi yeterli sayıda iterasyon ile öğrenebilmektedir.



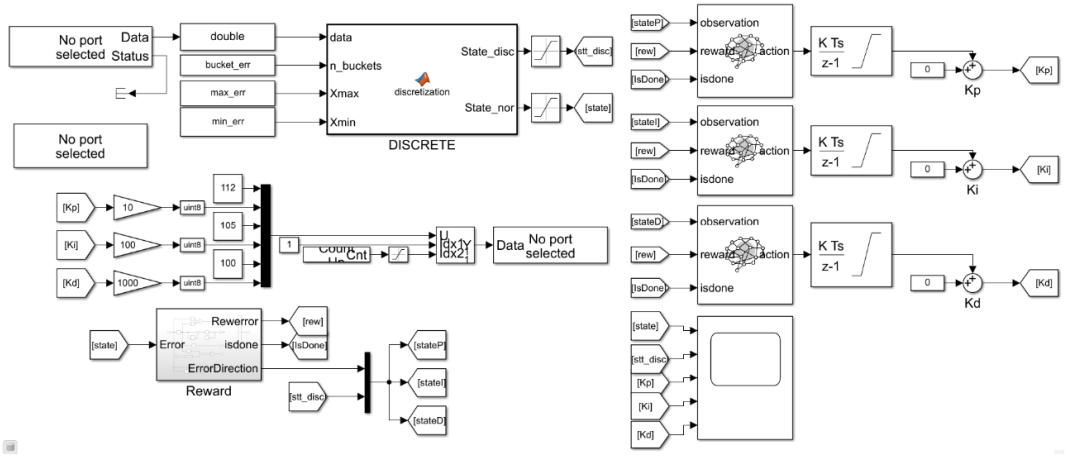
Şekil 3.7. Kontrol Kartı (ADUC841)



Şekil 3.8. Düzenek Sürücü Kartı



Şekil 3.9. DA Motoru Deney Seti

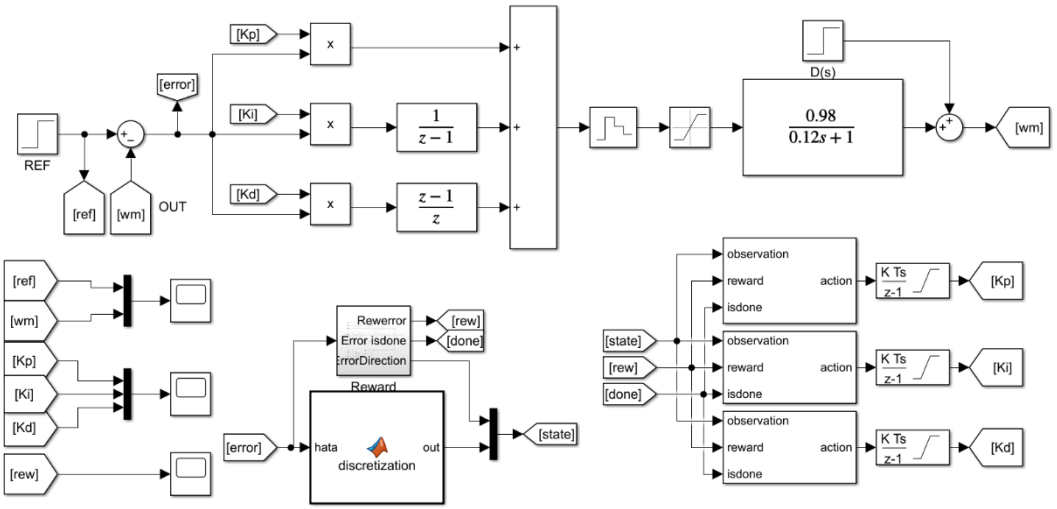


Şekil 3.10. Gerçek-Zamanlı Çalışma Simulink Ortamı

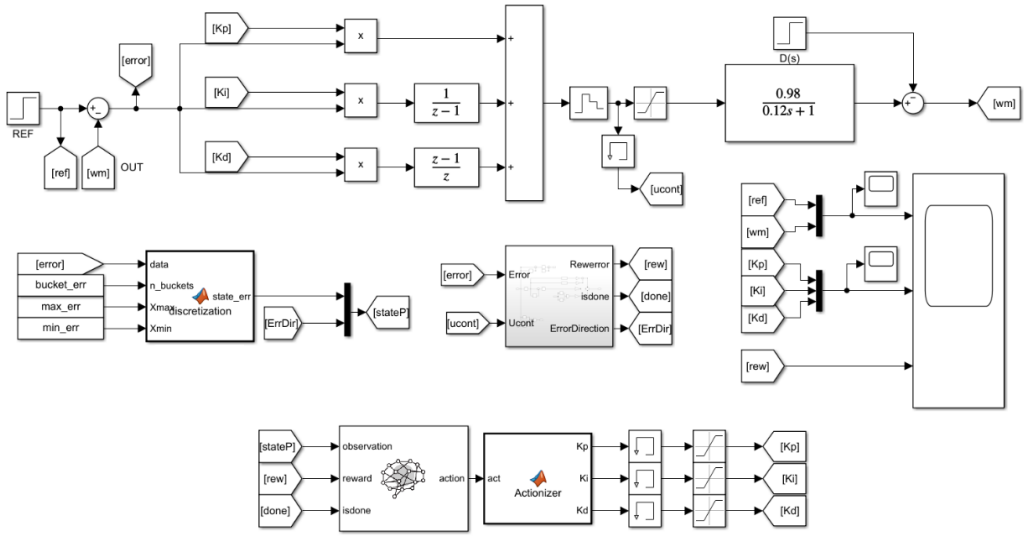
3.10. Benzetim Ortamı

Benzetim ortamı olarak MATLAB & Simulink tercih edilmiştir. MATLAB tarafından sunulan hazır pekiştirmeli öğrenme fonksiyonları ile çalışmamızın benzetimi gerçekleştirilmiştir [31]. Benzetim çalışmasında sistemin modeli oluşturulmuş, ajanlar ve ortam tanımlanmış ve eğitim süreci gerçekleştirilmiştir. Simulink ortamında kurulan benzetim çalışmasında hem eğitilmiş bir ajan elde edilmiştir; hem de sistemin çalışması doğrulanmıştır.

Benzetim ortamında elde edilmiş olan Q-tabloları gerçek zamanlı çalışmada da kullanılabilmektedir. Simulink ortamında model üzerinde kolayca değişiklikler yapılarak kontrolcü performansı incelenebilmektedir. Gerçek-zamanlı çalışma ile öğrenme işlemine göre daha hızlı ve daha kolay bir yöntem sağlamaktadır.



Şekil 3.11. Üç Ajanlı Çalışma Benzetimi

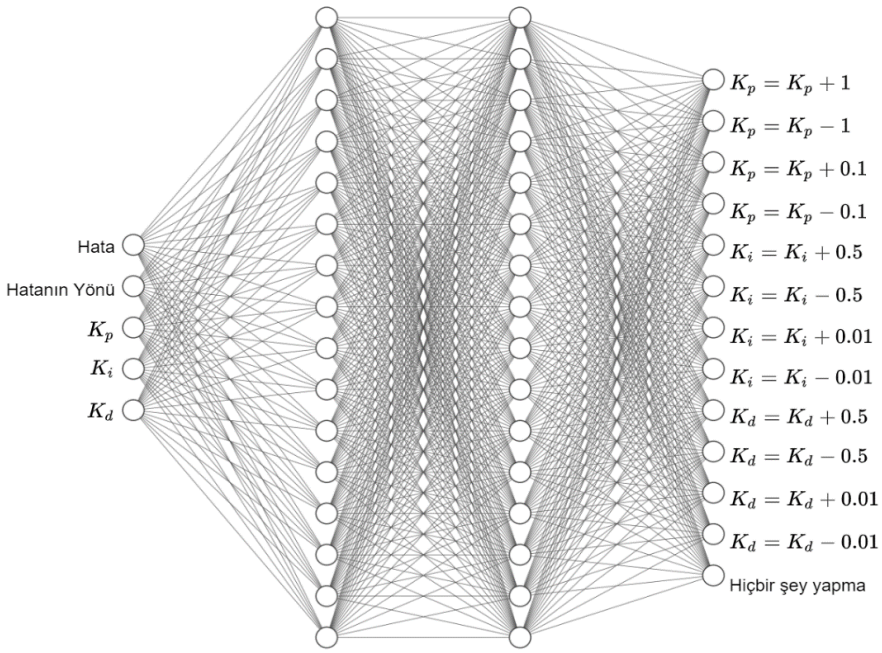


Şekil 3.12. Bir Ajanlı Çalışma Benzetimi

3.11. Derin Q-Öğrenme Tabanlı PID Kontrolör Tasarımı

Q-Öğrenme yöntemi çok fazla durum ve aksiyon içeren sistemlerde kullanılamaz. Ayrık gruplama vb. Yöntemler ile Q-Öğrenme yöntemi her sisteme uygulanabileceği gibi derin Q-Öğrenme yöntemi sayesinde sürekli durumlar ile de kullanılabilir. Derin

Q-Öğrenme yöntemi de uygulanmış ve sonuçlar değerlendirilmiştir. Derin öğrenme yapısı için bir nöral ağ tasarlanmıştır. Ajan optimal politikaya nöral ağ sayesinde yaklaşım yapmaktadır. Ayrıca derin öğrenme yöntemi sayesinde sürekli sinyalleri durum sinyali olarak değerlendirebiliriz.



Şekil 3.13. Nöral Ağ Yapısı

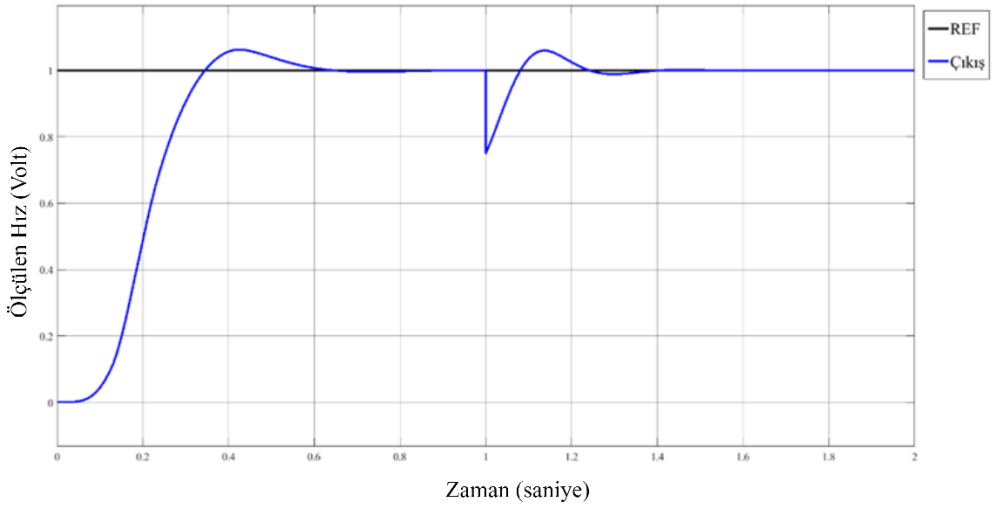
Çalışmada 1 adet Derin Q-Öğrenme ajanına 5 adet sinyal durum bilgisi olarak verilmektedir. Bunlar; hata, hatanın değişim yönü, oransal katsayı, integral katsayısı ve türevsel katsayıdır. Ajan, hata sıfıra eşitse 1; azalma yönünde ise 1; artma yönünde ise -1 ödül almaktadır. Ajana gelen ödül ve durum bilgileri ile ajan, K_p, K_i ve K_d katsayılarını arttırıp azaltmaktadır. Derin öğrenme yönteminde aksiyonlar PID katsayılarını hem ince ayar yapabilecek hem de kaba ayar yapabilecek şekilde seçilmiştir. Nöral ağ yapısı Şekil 3.13.'te verilmiştir.

BÖLÜM 4. BULGULAR VE TARTIŞMA

Sistemimiz için Q-Öğrenme algoritması tabanlı adaptif PID tasarlanmıştır. Bir ajanlı, üç ajanlı ve derin öğrenme destekli Q-Öğrenme yöntemleri uygulanmıştır.

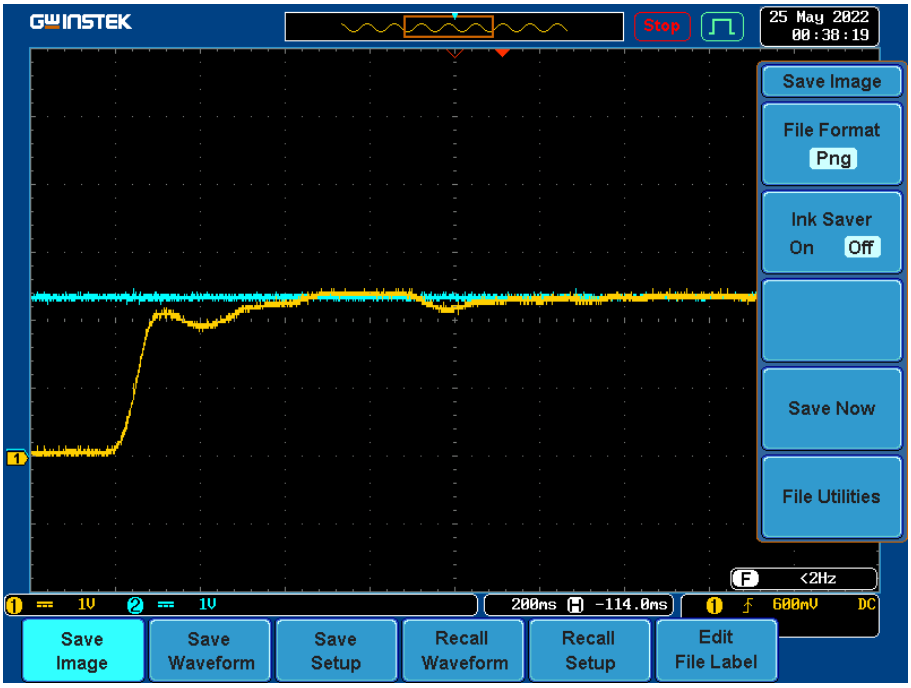
4.1. Bir Ajanlı Çalışma Bulguları

Ajanımız Şekil 3.12.'de verilen yapıdaki gibi benzetim ortamında eğitilmiş ardından gerçek-zamanlı sistem üzerinde test edilmiştir.



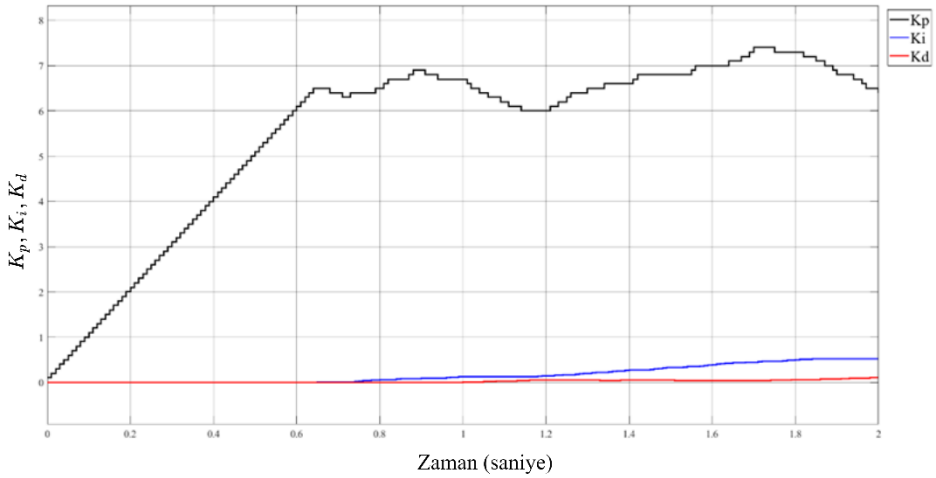
Şekil 4.1. Bir Ajanlı Çalışma Benzetim Çalışması Sonucu

Şekil 4.1.'de benzetim çalışması sonucunda elde edilen referans ve çıkış grafiği verilmiştir. Q-Öğrenme ajanı PID katsayılarını arttırıp azaltmak suretiyle hatayı fiziksel kısıtları altında mümkün olan en hızlı şekilde sıfırlamaya çalışmıştır. PID kontrolcü yapısı bozucu etkisini gidermektedir. Q-Öğrenme ajanı bozucu geldiği anda PID katsayılarını güncelleyerek hatanın daha kısa sürede giderilmesini sağlar. Elde edilen Q-tablosu ile gerçek-zamanlı sistemin kontrolü gerçekleştirilmiştir.



Şekil 4.2. Bir Ajanlı Q-PID Gerçek-Zaman Performansı

Elde edilen grafik incelendiğinde 500 milisaniyede çıkış referansa aşımsız olarak ulaşmıştır. 2. Saniyede verilen bozucu etki giderilmiştir. Gerçek-zamanlı çalışmada referans ve çıkış işaretleri üzerinde birçok gürültü olmasına rağmen Q-Öğrenme ajanı kontrol işlemini gerçekleştirebilmektedir.



Şekil 4.3. Bir Ajanlı Gerçek-Zamanlı Çalışma PID Katsayıları

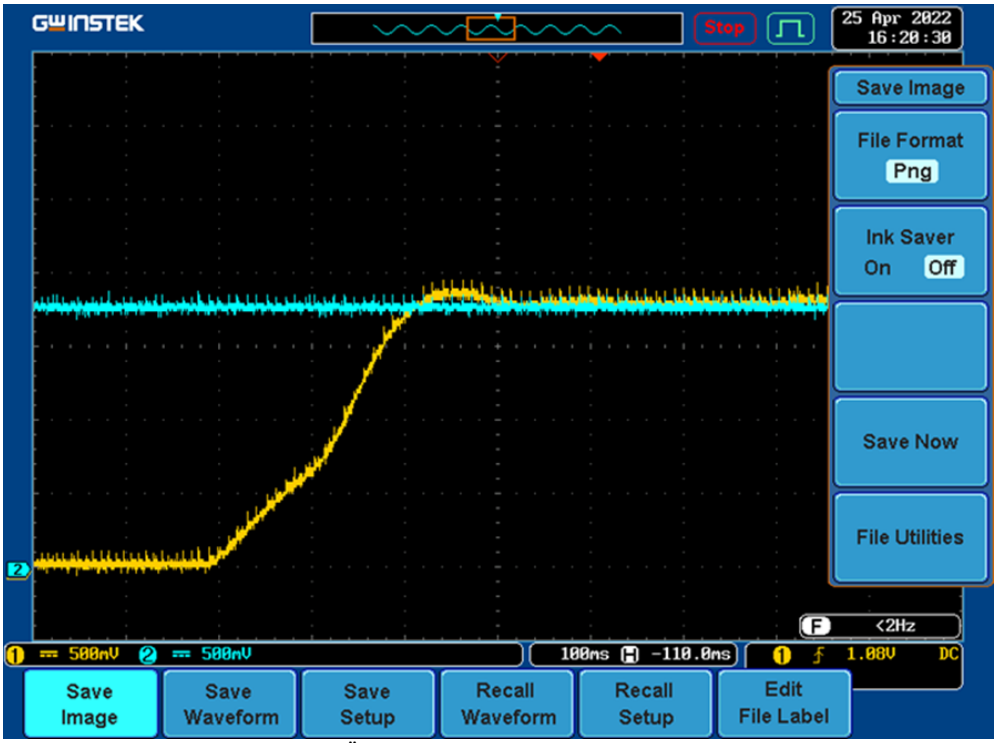
Şekil 4.3.'te verilen grafik incelendiğinde ajan ilk 1,5 saniye boyunca K_p ve K_i katsayılarını arttırmış. Ardından K_d katsayısını arttırmaya başlamış ve K_p ve K_i katsayılarını değiştirmemiştir. 2. saniyede gelen bozucudan sonra K_p'yi bir miktar

düşürmüş ardından yükseltmeye karar vermiştir. Nihayetinde $K_p, K_i \; ve \; K_d$ katsayılarını sırasıyla [6,5, 0,5, 0,3] seviyelerine getirmiştir.

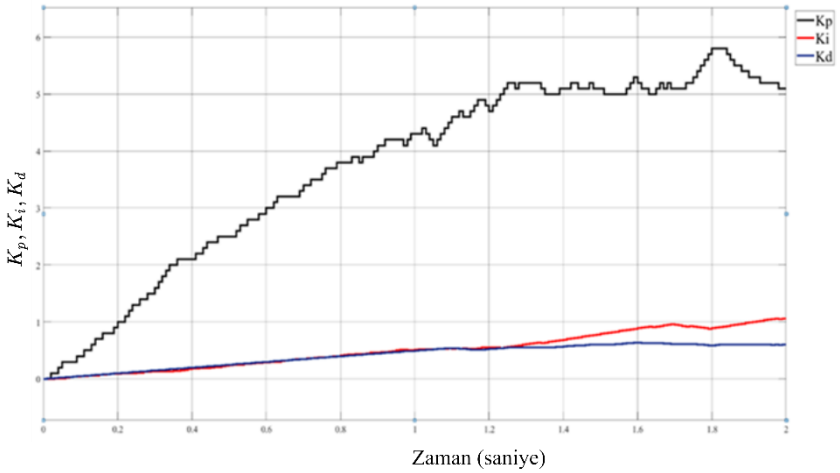
Ajanımız durum ve aksiyon sayısı yüksek olmasına rağmen kontrol işlemini gerçekleştirebilmiştir.

4.2. Üç Ajanlı Çalışma Bulguları

Ajanımız Şekil 3.11.'de verilen yapıdaki gibi benzetim ortamında eğitilmiş ve Q-tabloları oluşturulmuştur. Her PID katsayısı için bir adet ajan atanmıştır. Ajanlar ilgili katsayı için arttırma, azaltma veya değişiklik yapmama kararı almaktadır. $K_p = [-0{,}1 \;\; 0 \;\; 0{,}1]$, $K_i = [-0{,}01 \;\; 0 \;\; 0{,}01]$ ve $K_d = [-0{,}005 \;\; 0 \;\; 0{,}005]$ sayıları ile değişmektedir.



Şekil 4.4. Üç Ajanlı Çalışma Gerçek-Zamanlı Sonuç



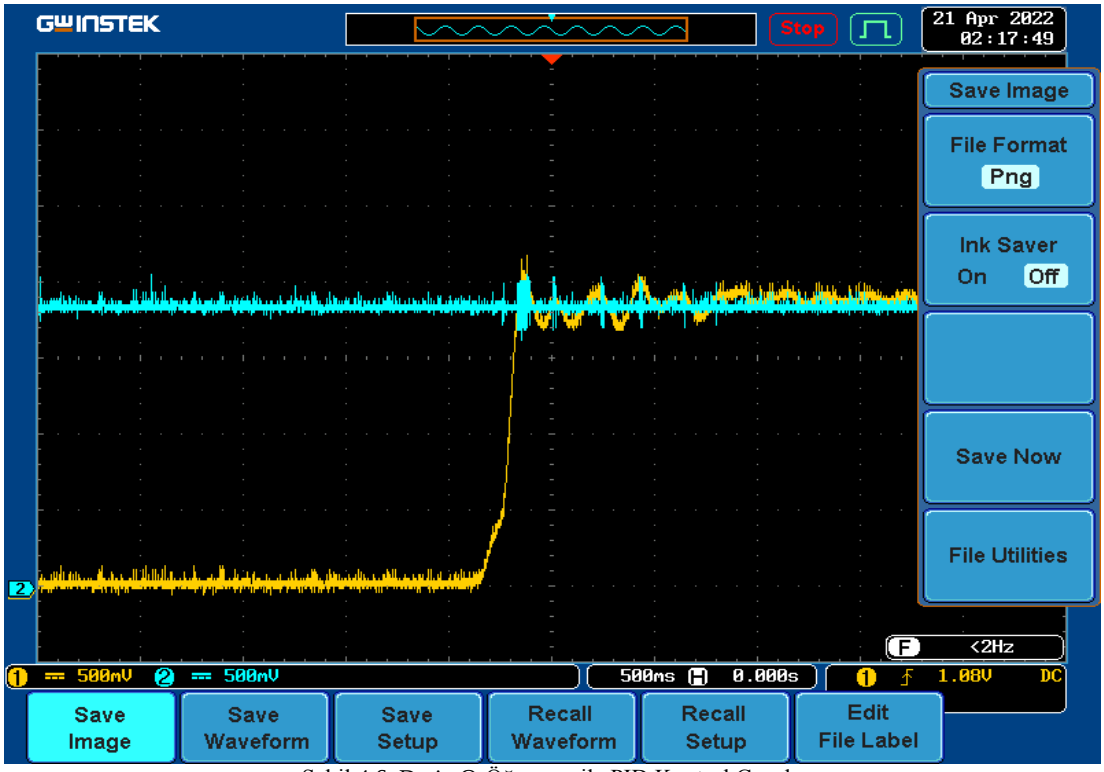
Şekil 4.5. Üç Ajanlı Çalışma PID parametreleri

Yapılan çalışma sonucunda 3 ajan kullanılarak gerçekleştirilen kontrol işleminde ajanlar sürekli hal hatasını %7'lik aşımla 450 milisaniyede sıfırlamıştır. Sistemin davranışı incelendiğinde kontrolcü optimal davranışa yakın bir şekilde sistemi kontrol etmeyi başarmıştır.

4.3. Derin Q-Öğrenme ile Kontrol

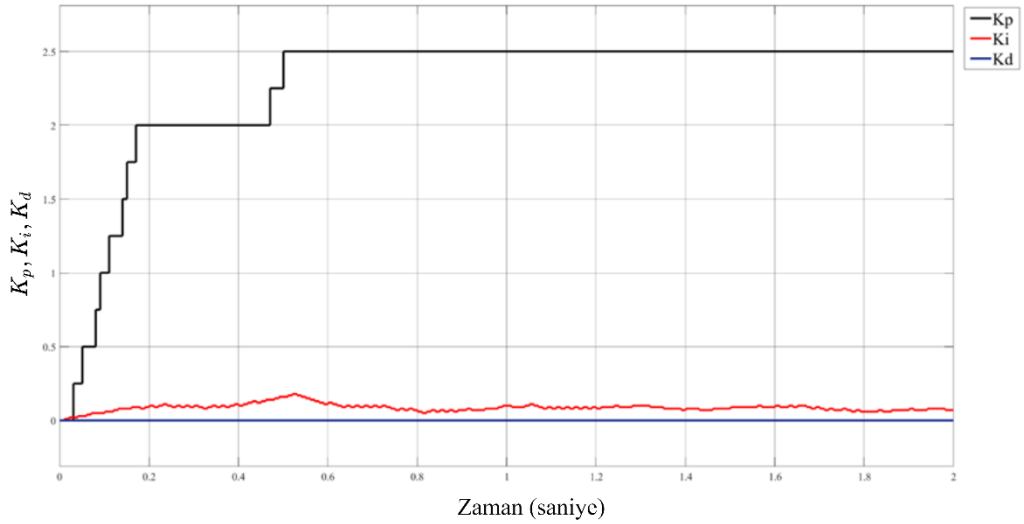
Tasarlanacak olan kontrolcünün PID katsayılarının optimal bir değere ayarlanabilmesi için derin öğrenme tabanlı Q-Öğrenme ajanı oluşturulmuştur ve eğitilmiştir. Sistemden gelen sürekli-zaman durum sinyali işlenerek her PID katsayısı için arttırma, azaltma ve değiştirmeme kararları verilmektedir. Şekil 3.13.'te verilen nöral ağ yapısı yardımı ile optimal değer fonksiyonu elde edilmektedir.

Derin Q-Öğrenme yöntemi ile eğitilen adaptif PID kontrolörün gerçek-zamanlı bir sistemdeki cevabı şekildeki gibidir.



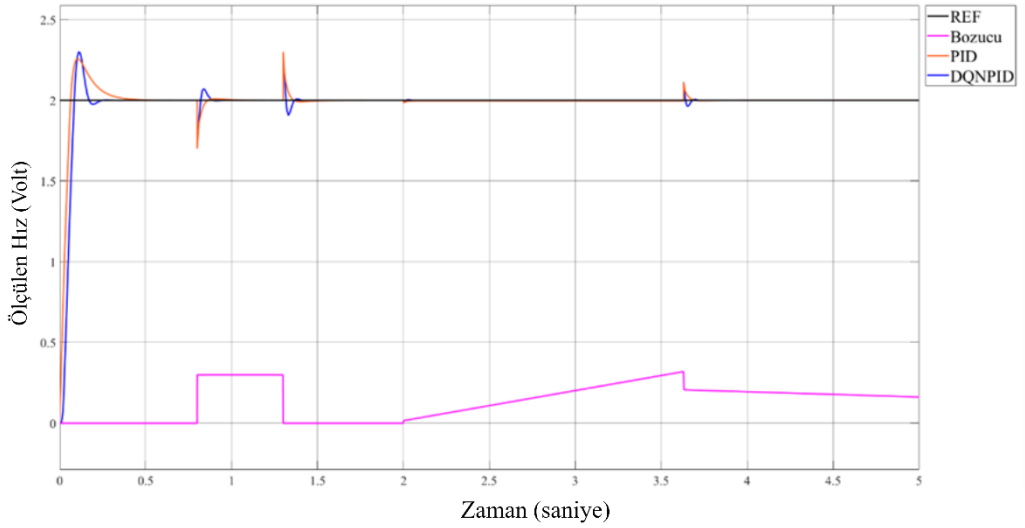
Şekil 4.6. Derin Q-Öğrenme ile PID Kontrol Cevabı

Sistemin yaklaşık 450 milisaniye yerleşme zamanı ve %5'lik bir aşımla referansı takip ettiği görülmektedir.



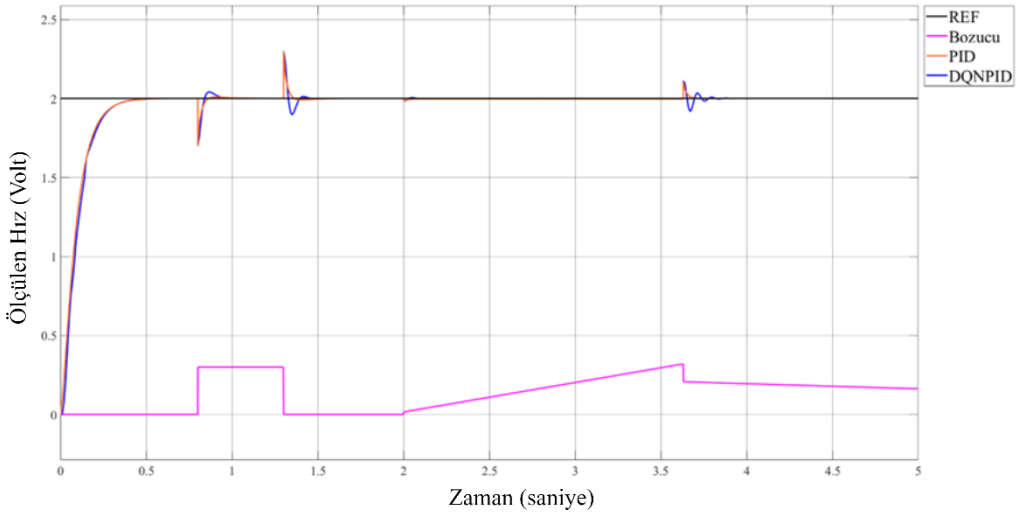
Şekil 4.7. Derin Q-Öğrenme PID Katsayıları

Tasarlanan kontrolcü ile model-tabanlı parametrik denklemler [19] ile elde edilmiş olan PID kontrolör cevabı karşılaştırılmıştır.



Şekil 4.8. Derin Q-Öğrenme ve Klasik PID Kontrolör Karşılaştırması

Tasarlanan kontrolcüler modifiye PID yapısıyla tekrar çalıştırılmıştır ve performansları tekrar gözlemlenmiştir.



Şekil 4.9. Derin Q-Öğrenme ve PID, Modifiye Kontrolör Karşılaştırması

Sonuçlar incelendiğinde elde edilen sistem dinamiği kısıtlar göze alındığında model tabanlı tasarlanan PID kontrolör kadar başarılı sonuçlar vermektedir.

BÖLÜM 5. SONUÇ

Pekiştirmeli öğrenme algoritması yöntemlerinden birisi olan Q-Öğrenme yöntemi ile Adaptif PID kontrolcü tasarlanmıştır. Bir ajanlı yapı, üç ajanlı yapı ve derin Q-Öğrenme yapısına ait olan sonuçlar elde edilmiştir. Adaptif PID kontrol sisteminde durum ve aksiyon sayısı fazla olduğundan dolayı ajanların optimal politikayı elde etmeleri için eğitim süreci uzun tutulmuştur. Ajanlar benzetim ortamında eğitilmiş ve gerçek-zamanlı sisteme uygulanmıştır. Gerçek-zamanlı sistemde elektriksel gürültüler ve modelde bulunan farklılıklara rağmen ajanlar görevlerini yerine getirmişlerdir. Elde edilen sonuçlar incelendiğinde hem Q-Öğrenme, hem de Derin Q-Öğrenme yönteminin oldukça başarılı olduğu ve model tabanlı olarak tasarlanan bir PID kontrolcüsüne oldukça yakın cevap verdiği görülmektedir. Karmaşık sistemler, yüksek dereceden sistemler ve doğrusal olmayan durumlar içeren sistemler gibi model-tabanlı kontrolcü tasarımının yetersiz kaldığı uygulamalarda kullanılabilir. Derin Q-Öğrenme tabanlı kontrolör aynı zamanda standart PID kontrolcüsüne bir destekleyici olarak çalışabilir. Kontrol işaretini destekleyerek, klasik PID'nin modeldeki doğrusal olmayan durumlarda daha kararlı çalışmasını sağlayabilir. Derin Q-Öğrenme yöntemi sürekli durum sinyali için ayrık aksiyonlar yerine getirebilmektedir. Sürekli aksiyonlar üretebilecek olan pekiştirmeli öğrenme yöntemleri ile adaptif kontrolcüler tasarlanabilir ve doğrusal olmayan sistemlerde daha başarılı sonuçlar elde edilebilir.

KAYNAKLAR

[1] Sutton, R. S. ve Barto, A. G., An introduction to reinforcement learning, Decis. Theory Model. Appl. Artif. Intell. Concepts Solut., 63–80, doi: 10.4018/978-1-60960-165-2.ch004, 2011.

[2] Minsky, M. L., Theory Of Neural-Analog Reinforcement Systems and Its Application To The Brain-Model Problem, Princeton University, Princeton, 1954.

[3] Bertsekas, D. P., Dynamic Programming and Stochastic Control, New York-London: Academic Press, 1976.

[4] Bellman, R., The Theory of Dynamic Programming, Bull. Am. Math. Soc., vol. 60, no. 6, 503–515, doi: 10.1090/S0002-9904-1954-09848-8, 1954.

[5] Bellman, R., Dynamic programming and stochastic control processes, Inf. Control, vol. 1, no. 3, 228–239, doi: 10.1016/S0019-9958(58)80003-0, 1958.

[6] Watkins, C. J. C. H. ve Dayan, P., Q-learning, Mach. Learn., vol. 8, no. 3, 279–292, doi: 10.1007/BF00992698, 1992.

[7] Mnih, V. ve ark., "Human-level control through deep reinforcement learning," Nature, vol. 518, no. 7540, pp. 529–533, 2015, doi: 10.1038/nature14236.

[8] Shi, Q., Lam, H. K., Xiao, B., Tsai, S. H., Adaptive PID controller based on Q-learning algorithm, CAAI Trans. Intell. Technol., vol. 3, no. 4, 235–244, doi: 10.1049/trit.2018.1007, 2018.

[9] Lewis, F. L. ve Vrabie, D., "Adaptive dynamic programming for feedback control," Proc. 2009 7th Asian Control Conf. ASCC 2009, pp. 1402–1409, 2009.

[10] Amiruddin, B. P. ve Kadir, R. E. A., "Ball and beam control using adaptive pid based on q-learning," Int. Conf. Electr. Eng. Comput. Sci. Informatics, vol. 2, no. October, 203–208, 2020, doi: 10.23919/EECSI50503.2020.9251898, 2020.

[11] Ali, M., Mujeeb, A., Ullah, H., Zeb, S. Reactive Power Optimization Using Feed Forward Neural Deep Reinforcement Learning Method : (Deep Reinforcement Learning DQN algorithm), 2020 Asia Energy Electr. Eng. Symp. AEEES 2020, 497–501, doi: 10.1109/AEEES48850.2020.9121492, 2020.

[12] Tan, T. Bao, F., Deng, Y. Jin, A., Dai, Q., Wang, J., Cooperative deep reinforcement learning for large-scale traffic grid signal control, IEEE Trans. Cybern., vol. 50, no. 6, 2687–2700, doi: 10.1109/TCYB.2019.2904742, 2020.

[13] Hung, J. C. ve Hewlett, J. D., PID control, Control Mechatronics, vol. 9, pp. 10.1-10.8, doi: 10.3156/jfuzzy.7.4_768, 2016.

[14] Lu, P., Huang, W., ve Xiao, J., Speed tracking of Brushless DC motor based on deep reinforcement learning and PID, 2021 7th Int. Conf. Cond. Monit. Mach. Non-Stationary Oper. C, 130–134, doi:10.1109/CMMNO53328.2021.9467649, 2021

[15] Shang, X. Y., Ji, T. Y., Li, M. S., Wu, P. Z. ve Wu, Q. H., Parameter optimization of PID controllers by reinforcement learning, 2013 5th Comput. Sci. Electron. Eng. Conf. CEEC 2013 - Conf. Proc., 77–81, doi: 10.1109/CEEC.2013.6659449, 2013.

[16] Mukhopadhyay, R., Bandyopadhyay, S., Sutradhar, A. ve Chattopadhyay, P., Performance Analysis of Deep Q Networks and Advantage Actor Critic Algorithms in Designing Reinforcement Learning-based Self-tuning PID Controllers, 2019 IEEE Bombay Sect. Signat. Conf. IBSSC 2019, vol. 2019 Januar, 1–6, doi: 10.1109/IBSSC47189.2019.8973068, 2019.

[17] Knospe, C., "PID control," IEEE Control Syst., vol. 26, no. 1, 30–31, doi: 10.1109/MCS.2006.1580151, 2006.

[18] Ziegler, J. G. ve Nichols, N. B., Optimum Settings for Automatic Controllers, J. Dyn. Syst. Meas. Control, vol. 115, no. 2B, 220–222, doi: 10.1115/1.2899060, 1993.

[19] Ozdemir, A. ve Erdem, Z., A new approach for calculation of PID parameters with model based compact form formulations, Elektron. ir Elektrotechnika, vol. 20, no. 3, 3–10, doi: 10.5755/J01.EEE.20.3.4415, 2014.

[20] Cetin, M. ve Iplikci, S., A novel auto-tuning PID control mechanism for nonlinear systems, ISA Trans., vol. 58, 292–308, doi: 10.1016/J.ISATRA.2015.05.017, 2015.

[21] Guan, Z. ve Yamamoto, T., Design of a reinforcement learning PID controller, IEEJ Trans. Electr. Electron. Eng., doi: 10.1002/tee.23430, 2021.

[22] Jung, J. W., Leu, V. Q., Do, T. D., Kim, E. K. ve Choi, H. H., Adaptive PID speed control design for permanent magnet synchronous motor drives, IEEE Trans. Power Electron., vol. 30, no. 2, 900–908, doi: 10.1109/TPEL.2014.2311462, 2015.

[23] Liu, G. P. ve Daley, S., Optimal-tuning PID control for industrial systems, Control Eng. Pract., vol. 9, no. 11, 1185–1194, doi: 10.1016/S0967-0661(01)00064-8, 2001.

[24] Ekinci, S. ve Hekimoğlu, B., Multi-machine power system stabilizer design via HPA algorithm, J. Fac. Eng. Archit. Gazi Univ., vol. 32, no. 4, 1271–1285, doi: 10.17341/gazimmfd.369716, 2017.

[25] Pareek, S., Kishnani, M. ve Gupta, R., Optimal tuning of PID controller using Meta heuristic algorithms, doi: 10.1109/ICAETR.2014.7012816, 2014.

[26] Guo, L., Hung, J. Y. ve Nelms, R. M., Evaluation of DSP-based PID and fuzzy controllers for DC-DC converters, IEEE Trans. Ind. Electron., vol. 56, no. 6, 2237–2248, doi: 10.1109/TIE.2009.2016955, 2009.

[27] Kaelbling, L. P., Littman, M. L., ve Moore, A. W., Reinforcement learning: A survey, J. Artif. Intell. Res., vol. 4, 237–285, doi: 10.1613/JAIR.301, 1996.

[28] Littman, M. L., Value-function reinforcement learning in Markov games, Cogn. Syst. Res., vol. 2, no. 1, 55–66, doi: 10.1016/S1389-0417(01)00015-8, 2001.

[29] Bellman, R., A Markovian Decision Process, J. Math. Mech., vol. Vol. 6, no. No. 5, 679–684, 1957.

[30] Yu, W., ve Perrusquía, A., Human-Robot Interaction Control Using Reinforcement Learning, 2021.

[31] Reinforcement Learning ToolboxTM User's Guide, https://www.mathworks.com/help/pdf_doc/reinforcement-learning/rl_ug.pdf., Erişim Tarihi: 29.03.2022.

ÖZGEÇMİŞ

Adı Soyadı **:** Bekir Murat AYDIN

ÖĞRENİM DURUMU

Derece	**Eğitim Birimi**	**Mezuniyet Yılı**
Yüksek Lisans	Sakarya Üniversitesi / Fen Bilimleri Enstitüsü / Elektrik Elektronik Mühendisliği	2022
Lisans	Sakarya Üniversitesi / Mühendislik Fakültesi / Elektrik Elektronik Mühendisliği	2019
Lise	Akyazı Anadolu Lisesi	2015

İŞ DENEYİMİ

Yıl	**Yer**	**Görev**
2021-Halen	Sakarya Üniversitesi	Araştırma Görevlisi
2019-2021	Kolarc Makine İmalat San. ve Tic. A.Ş.	ARGE Mühendisi

YABANCI DİL

İngilizce