T.C.
NECMETTİN ERBAKAN NİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

QUADCOPTERLER İÇİN LQR KONTROLCÜ PARAMETRELERİNİN OPTİMİZE EDİLMESİ

Yasin BÜYÜKER

YÜKSEK LİSANS TEZİ

Mekatronik Mühendisliği Anabilim Dalı

Temmuz-2021
KONYA

TEZ KABUL VE ONAYI

Yasin BÜYÜKER tarafından hazırlanan "QUADCOPTERLER İÇİN LQR KONTROLCÜ PARAMETRELERİNİN OPTİMİZE EDİLMESİ" adlı tez çalışması 09/07/2021 tarihinde aşağıdaki jüri tarafından oy birliği ile Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü Mekatronik Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS Tezi olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Prof. Dr. Gülay TEZEL	……..
Danışman Doç. Dr. İlhan İLHAN	……..
Üye Dr. Öğr. Üyesi Ümit ÖNEN	……..

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun …./…/20.. gün ve …….. sayılı kararıyla onaylanmıştır.

Prof. Dr. İbrahim KALAYCI
FBE Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Yasin BÜYÜKER

Tarih: 09/07/2021

ÖZET

YÜKSEK LİSANS TEZİ

QUADCOPTERLER İÇİN LQR KONTROLCÜ PARAMETRELERİNİN OPTİMİZE EDİLMESİ

Yasin BÜYÜKER

Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü
Mekatronik Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. İlhan İLHAN

2021, 121 Sayfa

Jüri
Doç. Dr. İlhan İLHAN
Prof. Dr. Gülay TEZEL
Dr. Öğr. Üyesi Ümit ÖNEN

Quadcopterler üzerinde yapılan çalışmaların önemi özellikle son yirmi yılda oldukça artmıştır. Teknolojik gelişmeler, quadcopterlerin başarılı uçuşlar gerçekleştirmesi için yeterli donanım ve tasarım olanakları sunmaktadır. Sunulan donanım ve tasarım imkanlarıyla daha performanslı bir uçuş gerçekleştirmek için kontrol sistemleri kullanılmaktadır. Bu amaçla araştırmacılar yeni kontrol yöntemleri geliştirmeye ya da var olanları quadcopterlere uyarlamaya çalışmaktadır. Bu kontrol sistemlerinden biri de LQR kontroldür.

LQR hem lineer hem de lineer olmayan sistemler için geliştirilmiş temel olarak minimum maliyetle maksimum verim sağlamayı hedefleyen bir kontrol yöntemidir. İlerleyen bölümlerde LQR kontrol yönteminin temel dayanağı ve matematiksel ifadesi anlatılmıştır. Ayrıca yöntemin avantajlarından ve dezavantajlarından bahsedilmiştir.

Son yıllarda yapay zekâ optimizasyon algoritmaları kullanılarak LQR kontrolcü parametrelerini belirlemek için çeşitli çalışmalar yapılmıştır. Bu çalışmada ise literatürde kabul görmüş, Quanser firmasına ait, quadcopter sistemi modeli üzerinde LQR kontrolcü uygulanmış, yeni bir amaç fonksiyonu belirlenmiş ve LQR kontrolcü parametreleri yedi farklı yapay zekâ optimizasyon algoritması (yusufçuk, genetik, parçacık sürü optimizasyon, benzetilmiş tavlama, yapay arı kolonisi, diferansiyel gelişim, gri kurt

optimizasyon) ile optimize edilmiştir. Ayrıca benzetilmiş tavlama ve gri kurt optimizasyon algoritmalarını içeren yeni hiyerarşik SAA-GWO algoritması önerilmiştir. Hiyerarşik SAA-GWO algoritması ve diğer yedi algoritmanın optimize ettiği LQR kontrolcü parametrelerinin sisteme etkisi, belirlenen yeni amaç fonksiyonu ile ayrı ayrı incelenmiş ve düşük çalışma koşullarına göre verimlilikleri karşılaştırılmıştır. Roll açısının oturma ve yükselme zamanı açısından, Hiyerarşik SAA-GWO algoritması hem Quanser firmasına hem de diğer yedi algoritmaya kıyasla daha iyi sonuçlara ulaşmıştır.

Anahtar Kelimeler: LQR Kontrol, Quadcopter, Yapay Zekâ Optimizasyon Algoritmaları, Simülasyon.

ABSTRACT

MS THESIS

OPTIMIZING OF LQR CONTROLLER PARAMETERS FOR QUADCOPTERS

Yasin BÜYÜKER

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF NECMETTİN ERBAKAN UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE IN MECHATRONICS ENGINEERING

Advisor: Assoc. Prof. Dr. İlhan İLHAN

2021, 121 Pages

Jury
Assoc. Prof. Dr. İlhan İLHAN
Prof. Dr. Gülay TEZEL
Asst. Prof. Dr. Ümit ÖNEN

The importance of studies on quadcopters has increased considerably, especially in the last two decades. Technological developments provide sufficient equipment and design possibilities for quadcopters to perform successful flights. Control systems are used to achieve a more efficient flight with the hardware and design possibilities offered. For this purpose, researchers are trying to develop new control methods or adapt existing ones to quadcopters. One of these control systems is LQR control.

LQR is a control method developed for both linear and non-linear systems, basically aiming to provide maximum efficiency with minimum cost. In the following sections, the basics and mathematics of the LQR control method are explained, the advantages and disadvantages of the method are mentioned.

In recent years, various studies have been carried out to determine LQR controller parameters using artificial intelligence optimization algorithms. In this study, an LQR controller was applied to the quadcopter system model of Quanser company, which is accepted in the literature, a new objective function was determined, and the LQR controller parameters were optimized by seven different artificial intelligence optimization algorithms (dragonfly, genetics, particle swarm optimization, simulated annealing, artificial bee colony, differential evolution, gray wolf optimization). In

addition, a new hierarchical SAA-GWO algorithm including simulated annealing and gray wolf optimization algorithms is proposed. The effect of LQR controller parameters optimized by hierarchical SAA-GWO algorithm and seven other algorithms on the system was examined separately with the determined new objective function and their efficiency was compared according to low operating conditions. In terms of settling and rising time of the roll angle, the Hierarchical SAA-GWO algorithm achieved better results compared to both Quanser and the other seven algorithms.

Keywords: LQR Controller, Quadcopter, Artificial Intelligence Optimization Algorithms, Simulation

TEŞEKKÜR

Yüksek lisans sürecimde değerli yönlendirmeleriyle yoluma ışık tutan çalışmaktan keyif aldığım danışmanım Doç. Dr. İlhan İLHAN'a teşekkür ederim.

Yardıma ihtiyaç duyduğumda yardımcı olacaklarına inandığım bölüm hocalarıma teşekkür ederim.

Sadece lisansüstü eğitimimde değil hayatımın her alanında beni destekleyen sevgili aileme sonsuz teşekkürlerimi sunuyorum.

Bu süreçte yanımda olan arkadaşlarıma teşekkür ederim.

Yasin BÜYÜKER
KONYA-2021

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

SİMGELER VE KISALTMALAR

Simgeler

$\vec{X}_\alpha, \vec{X}_\beta, \vec{X}_\delta$	:	Alfa, beta ve delta kurtların konumları
$\dot{\theta}_r, \dot{\theta}_p, \dot{\theta}_y$	:	Roll, pitch, yaw açısal hızları
$\tau_f, \tau_b, \tau_r, \tau_l$	:	Motorların oluşturduğu moment
$\vec{A}, \vec{C}, \vec{D}$	:	Gri kurt optimizasyon algoritmasında kullanılan vektörler
F_f, F_b, F_r, F_l	:	Motorların oluşturduğu kaldırma kuvveti
J_r, J_p, J_y	:	Eylemsizlik momenti
K_f	:	İtki kuvveti sabitini
K_p, K_i, K_d	:	Oransal-İntegral-Türevsel Denetleyicinin kontrolcü parametresi
K_t	:	Motorun itme momenti sabiti
P_i	:	Yapay arı algoritması rulet tekerleği işleminde hesaplanan oran
S_i, A_i, C_i	:	Yusufçuk algoritmasının işlemleri (Ayrışma, Sıralanma, Uyum)
V_f, V_b, V_r, V_l	:	Motorlara verilen gerilim
V_j, F_i, E_i	:	Yusufçuk algoritmasına ait ifadeler (Hız,Yemek,Düşman)
X^+, X^-	:	Yusufçuk algoritması (yemek konumu/düşman konumu)
$X_{üst}(i)$	:	Parametre sınırlarını belirten bir ifade
$X_{alt}(i)$	:	Parametre sınırlarını belirten bir ifade
$X_j, \Delta X_i$	:	Komşu yusufçuk, Yusufçuğun yer değiştirme isteğini
$\vec{a}$	:	Gri kurt optimizasyon algoritmasının parametresi
g_{best}, p_{best}	:	Parçacık sürü optimizasyon algoritması (global en iyi/lokal en iyi)
q_{ii}, r_{jj}	:	Q ve R matrislerinin diegonal elemanları (i=1,2,3,4,5,6 j=1,2,3,4)
t_{start}, t_{end}, t	:	Benzetilmiş tavlama algoritması parametreleri (başlangıç sıcaklığı, bitiş sıcaklığı, mevcut sıcaklık)
u_{ij}	:	Algoritmanın hesapladığı yeni çözüm kümesinin elemanı
x_j^{min}, x_j^{max}	:	Parametre sınırlarını belirten bir ifade
$\ddot{\theta}, \ddot{\theta}_r, \ddot{\theta}_p, \ddot{\theta}_y$	:	Roll, pitch, yaw açısal ivme
$\theta_p, \theta_r, \theta_y$	:	Roll, pitch, yaw açıları
ϕ_{ij}	:	Yapay arı algoritmasında üretilen rasgele sayı
X, Y, Z	:	Eksen takımları
$\Delta F, \Delta\tau$	:	Kuvvet farkı, Moment farkı
A, B, C, D	:	Durum uzay matrisleri

J	:	LQR kontrolün maliyet fonksiyonu
Je	:	Sistem için oluşturulan amaç fonksiyonu
K	:	Ricatti Denkleminin kökü
L	:	Motorların ağırlık merkezine olan uzaklığı
N	:	Kuvvet birimi
P	:	Ricatti denkleminde kullanılan bir matris
Q, R	:	LQR kontrolcü parametreleri
V	:	Gerilim
e	:	e sayısı (Benzetilmiş tavlama algoritması)
e, w, f	:	Yusufçuk algoritmasının parametreleri (Düşman katsayısı, Eylemsizlik katsayısı, Yemek katsayısı)
m	:	Uzunluk birimi
$s, a, c,$	:	Yusufçuk algoritmasının parametreleri (Ayrışma katsayısı, Sıralanma katsayısı, Uyum katsayısı)
u	:	3-DOF Hover deney setinin kontrol vektörü
v	:	Parçacık sürü optimizasyon algoritması parçacığın hızı
w, c_1, c_2	:	Parçacık sürü optimizasyon algoritması parametreleri (atalet ağırlığı, öğrenme faktörleri)
$x, \dot{x}$	:	3-DOF Hover deney setinin durum değişkenleri/türevi
y	:	3-DOF Hover deney setinin çıkış değişkenleri
$\alpha, \beta, \delta, \omega$	:	Gri kurtların grupları
σ, Γ, β	:	Lèvy Uçuş Mekanizmasını hesaplanırken kullanılan simgeler (sigma, gamma fonksiyonu, beta katsayısı)

Kısaltmalar

ABC	:	Yapay arı koloni algoritması
AGL	:	Yer Seviyesinin Üstünde
BLOS	:	Beyond-Line-Of-Sight (Görüş Hattının Ötesinde)
CO	:	Çaprazlama oranı
CR	:	Çaprazlama oranı
DA	:	Yusufçuk algoritması
DC	:	Doğru akım
DEA	:	Diferansiyel gelişim algoritması
deg	:	Derece
DOF	:	Serbestlik derecesi
F	:	Ölçekleme faktörü
GA	:	Genetik algoritma
GN	:	Maksimum iterasyon sayısı
GPS	:	Küresel Konumlama Sistemi
GWO	:	Gri kurt optimizasyon algoritması
HALE	:	Yüksek İrtifa, Uzun Süreli Dayanıklılık
Hiy.	:	Hiyerarşik çalışma yöntemi
İHA	:	İnsansız hava aracı
kg	:	Kilogram
KKK	:	Kayan kipli kontrol
km	:	Kilometre
LOS	:	Line Of Sight (Görüş Hattı)
LQG	:	Doğrusal karesel Gaussian kontrol
LQR	:	Doğrusal karesel düzenleyici
MALE	:	Medium-Altitude Long-Endurance(Orta İrtifa Uzun Dayanıklık)
MO	:	Mutasyon oranı
MSL	:	Mean Sea Level (Ortalama Deniz Seviyesi)
NATO	:	Kuzey Atlantik Antlaşması Örgütü
PD	:	Oransal-Türevsel Denetleyici
PI	:	Oransal-İntegral Denetleyici
PID	:	Oransal-İntegral-Türevsel Denetleyici
PN	:	Popülayon büyüklüğü sayısı

PSO	:	Parçacık sürü optimizasyonu algoritması
SAA	:	Benzetilmiş tavlama algoritması
sn	:	Saniye
U.d	:	Uygunluk değeri
UAV	:	Unmanned aerial vehicle

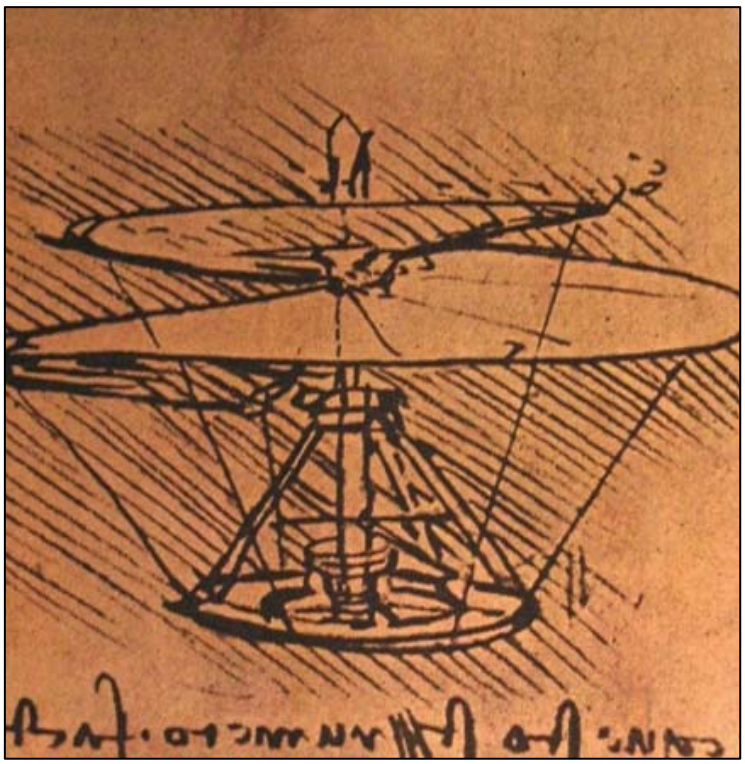
1. GİRİŞ

Uçmak, geçmişten günümüze kadar insanlık için merak uyandıran bir konu olmuş ve akılların bir köşesinde gün geçtikçe karşı konulamaz bir arzuya dönüşmüştür. Uçma arzusu öyle bir noktaya gelmiştir ki tarihte insanların kendilerini, yüksek bir noktadan kanat benzeri yapılarla serbest bırakarak uçma girişiminde bulundukları yazılmaktadır.

Tarihçilere göre bilinen başarılı, en eski insanlı uçuş 9. yüzyılda Abbas İbn Firnas tarafından gerçekleştirilmiştir. Abbas İbn Firnas uçuşunu, geliştirdiği kanat benzeri yapıyla ilk modern uçaktan yaklaşık 1000 yıl öncesinde gerçekleştirmiştir.

Bu tarihten sonra da insanların bu tarz uçma girişimleri olmuştur. Özellikle Türk tarihinde Hezârfen Ahmed Çelebi'nin, Galata Kulesi'nden kanat benzeri bir yapı yardımıyla atlayarak uçuş gerçekleştirmesi en bilinen örnektir. Her ne kadar roketler bu tezin konusu olmasa da Lâgari Hasan Çelebi'nin sırtına bağladığı roket ile uçma girişiminde bulunması da bu olaylara örnektir.

Sadece Türk tarihinde değil dünya tarihinde de benzer örnekler bulunmaktadır. Örneğin, Şekil 1.1'de dünyaca ünlü bilim insanı Leonardo da Vinci'nin 15. ya da 16. yüzyılda tasarladığı düşünülen bir helikopter tasarımı görülmektedir.



Şekil 1.1. Leonardo da Vinci'nin tasarladığı helikopter

Tüm bu kaydedilen, bugüne ulaşmış ve kaydedilmemiş, henüz bilinmeyen nice olaylar, insanlığın havacılığa olan ilgisinin çok eskiye dayandığını göstermektedir. Havacılığa duyulan geçmişten gelen ilgi bugün hala insanlığa ilham kaynağı olmaktadır. Yüzyıllardır süre gelen teknolojik gelişmeler, havacılığa duyulan ilginin makinelere dönüşmesini sağlamıştır. Sonuç olarak, havacılık tutkusu ve gelişen teknolojik imkânların katkıları ile insanlar, güvenli ve kontrollü uçuşu gerçekleştirebilen makineleri yapabilmişlerdir.

1.1. Havacılığın Gelişim Süreci

Zaman ilerledikçe bu tarz ilkel denemeler artık yerini daha uzun süreli uçuşlara bırakmaya başlamıştır. 18. yüzyıla gelindiğinde modern havacılığın temellerinin atıldığı görülmektedir. Bu yüzyıl içerisinde sıcak hava balonları icat edilmiştir. Bilinen ilk insanlı sıcak hava balonu ile uçuş, Fransa'da 1783 yılında gerçekleştirilmiştir. İlk üretilen sıcak hava balonlarının bir kontrol mekanizması bulunmamaktadır. Çünkü o tarihlerde insanlık uçuş kontrol mekanizması geliştirebilecek kadar uçma deneyimine sahip değildir. Kontrol mekanizması olmamasından dolayı uçuşlar hava şartlarına göre gerçekleşmektedir. Rüzgâr ne yönde eserse balon oraya sürüklenmektedir. Gerçekleşen uçuş, her ne kadar kontrolsüz olsa da insanlara hava şartlarını öğrenmesini sağlayacak kadar gökyüzünde kalabilme imkânı sunmuştur. İlerleyen süreçte daha kontrol edilebilir sıcak hava balonları üretilebilmiştir.

Sıcak hava balonlarıyla kazanılan uçuş deneyimleri ve teknolojinin olağan ilerleme süreci sonucunda, havacılık yavaş yavaş seviye atlamıştır. Bir sonraki yüzyılın sonlarına gelindiğinde, insanlar artık zeplin adını verdikleri ve kontrol edebildikleri hava taşıtlarını geliştirmişlerdir. Zeplinler, havacılık için henüz yolun başı sayılırdı ve zeplinlerin uçabilme kabiliyetleri havadan daha hafif olmalarından kaynaklanıyordu. Bu yüzden zeplinler, kısa ömürlü ve kırılgan bir yapıya sahipti. Havadan daha ağır ve dayanıklı cisimlerin uçabilmeleri için içten yanmalı motorların bulunması gerekiyordu.

Bir taraftan balonla yapılan çalışmalar devam ederken diğer taraftan da havadan daha ağır bir makineyi uçurabilme hayalleri devam ediyordu, insanlık gökyüzünde kalma konusunda gün geçtikçe daha fazla tecrübe kazanıyordu. Kazanılan bu yeni tecrübeler sayesinde aerodinamik yasalarının temelleri oluşturulmaya başlanmıştır.

Havacılık alanında çalışma yapan insanlar, öğrendikleri yöntemleri ve bilgileri dergilerde yayınlayarak havacılığın daha sistematik bir şekilde ilerlemesine fayda

sağlamışlardır. Eş zamanlı olarak dünyada bilgi paylaşımının artmasını sağlayacak teknolojik gelişmeler de yaşanmaya devam etmiştir. Sonuç olarak bu sistematik ilerleme ve haberleşmenin çoğalması yapılan hataların başka çalışmalarda tekrarlanmasının önüne geçmiş ve havacılık çalışmalarına ivme kazandırmıştır. Zaman ilerledikçe yapılan çalışmalar, deneme yanılma yönteminden kurtulup daha bilimsel bir hâl kazanmaya başlamıştır.

Havadan daha ağır bir cismi uçurma konusu üzerine yapılan çalışmalar yıllar ilerledikçe sonuçlarını vermeye başlamıştır. Albatros II adı verilen ve Fransız araştırmacı Jean-Marie Le Bris tarafından geliştirilen bir planör, 1856 yılında, atlar tarafından 200 metre çekilerek yerden 100 metre yüksekliğe ulaştırılmıştır. Bu uçuş havadan daha ağır bir makinenin ilk konumundan daha yüksek bir konuma çıkması açısından önemlidir.

Havacılık alanında bilimsel çalışmalara hız kazandıracak bir gelişme de aynı yıllarda İngiltere'de yaşanmıştır. Hava araçlarında kullanılan kanatların testleri için 1871 yılında İngiltere'de ilk rüzgâr tüneli inşa edilmiştir. Bu rüzgâr tüneli sayesinde bir kanadın uçabilmesi için sahip olması gereken tasarım hakkında birçok bilgi edinilmiştir.

İnsanların havada kalma tecrübelerine, kanatlar ile ilgili buldukları keşiflerin de eklenmesiyle modern anlamda ilk planörlerin inşası mümkün olmuştur. Havacılık alanındaki öncü çalışmalarıyla tanınan Alman mucit Otto Lilienthal yaptığı planörlerle 2500'den fazla uçuş gerçekleştirmiştir. Son uçuşunda rüzgâr nedeniyle planörünün kanatlarının kırılması sonucu 17 metre yükseklikten düşüp omurgasını kırmıştır. Ertesi gün ölüm döşeğinde bile son sözü "Küçük fedakarlıklar yapılmalıdır." olan bilim insanının havacılığa yaptığı katkılar unutulmamalıdır. Ayrıca Otto Lilienthal yaptığı bütün çalışmaları yazılı ve görsel olarak kaydetmiş, kendisinden sonra gelen havacılara ışık tutmuştur. Şekil 1.2'de Otto Lilienthal'in gerçekleştirmiş olduğu bir uçuşa ait fotoğraf görülmektedir.

Şekil 1.2. Otto Lilienthal'ın gerçekleştirdiği bir uçuş

Bu çalışmada ismi geçen ya da geçmeyen nice bilim insanının katkılarıyla insanlık, yeni bir devrin kapılarını aralamıştır. 1903 yılına gelindiğinde modern anlamdaki ilk uçak uçurularak bahsedilen yeni devrin kapıları ardına kadar açılmıştır. Bu uçuşu gerçekleştiren ve adları havacılık tarihine altın harflerle yazılan Wright kardeşler, sayısız başarısız denemenin ardından bu zafere ulaşmışlardır. Onların bu azim ve kararlılıkları kendilerinden sonraki nesillere örnek olmuştur.

Donanımsal ve tasarımsal gelişmeler bugün bir cismin kolaylıkla havalanmasına yetmektedir. Günümüze kadar gelen ve bugünden sonra da devam edecek olan havacılık alanındaki bütün çalışmaların en büyük hedefi, daha kontrollü uçan sistemler geliştirmektir. Bunu, özellikle, üzerinde insan bulundurmuyorken başarmaktır.

1.2. İnsansız Hava Araçları

Hava araçlarının önemi her geçen gün daha da iyi anlaşılmaktadır. Bundan dolayı havacılığa olan ilgi ve yatırımlar katlanarak artmaya devam etmektedir. Ancak insanlı hava araçlarını yapmak ve onları kullanabilecek kalifiye personel yetiştirmek oldukça maliyetlidir. Günümüzde bazı görevler için hava araçlarının üzerinde insan bulundurmasına gerek yoktur. Teknolojik imkânlar, bu görevler için hava araçlarının insansız da çalışmalarını sağlayabilecek çözümleri sunmaktadır. Geliştirilen bu çözümler kaynaklarda kısaca İHA (İnsansız Hava Aracı) ya da UAV (Unmanned Aerial Vehicle) olarak adlandırılmaktadır.

İnsansız hava araçları, üzerinde insan bulundurmayan, bir yer istasyonu yardımıyla veya otonom uçuş gerçekleştiren hava araçlarının genel tanımıdır. Farklı amaçlara göre tasarlanmış birçok insansız hava aracı bulunmaktadır.

1.2.1. İnsansız hava araçlarının kullanım alanları

İnsansız hava araçları, icat edildikleri zamandan itibaren askeri alanlarda kullanılmaya başlanmıştır. Askeri alanda keşif, gözetleme, istihbarat, saldırı, savunma gibi görevler için silahlı ya da silahsız insansız hava aracı kullanılmaktadır. Türkiye'nin de aralarında bulunduğu birçok ülke, özellikle son yirmi yılda insansız hava araçlarının askeri alanda kullanımına yönelik büyük yatırımlar yapmaktadır. Ayrıca Türkiye, bahsi geçen hava araçlarını aktif bir şekilde terörle mücadelede kullanmaktadır.

Askeri alanların yanı sıra insansız hava araçlarının en önemli kullanım alanlarından bir diğeri de afetlerdir. Deprem, çığ, sel, heyelan, göçük, yangın gibi afetlere ilk müdahalede bulunabilmek, durum tespiti yapabilmek ve arama kurtarma çalışmalarını koordineli bir şekilde gerçekleştirmek için kullanılmaktadır.

Diğer başlıca kullanım alanları ise kısaca şu şekildedir: Baraj, akarsu, maden yataklarının ve doğal güzelliklerin korunması, posta ve kargo teslimatları, tarımda verimliliği kontrol etme, sulama ve ilaçlama, haritalama ve eğlence sektörü.

Örneklerden de görüldüğü üzere insansız hava araçları günümüzde pek çok alanda kullanılmaktadır. İnsansız hava araçlarının bu denli yaygınlaşmasında, maliyetlerinin ulaşılabilir seviyelere inmesinin yanında kullanıcısına sunduğu başka birçok avantajı da etkili olmaktadır.

1.2.2. İnsansız hava araçlarının avantajları

İnsansız hava araçlarının önceki bölümlerde anlatıldığı gibi en önemli avantajı, maliyetlerinin uygun olmasıdır. Maliyet denilince sadece hava aracının üretim maliyeti düşünülmemelidir. Bahsedilen maliyet hesaplanırken üretilen hava aracını kullanacak personelin eğitimi için yapılacak yatırımları, bu hava araçlarının güvenli iniş kalkış yapabilmeleri için ihtiyaç duyulan pistlerin maliyetleri, kullanılacak yakıtların maliyetleri gibi konular göz önünde bulundurularak bir hesap yapılmalıdır.

İnsansız hava araçları, maliyetinin dışında da kullanıldığı alana göre pek çok avantajı kullanıcısına sunmaktadır. Örneğin son yıllarda tarım ilaçlamalarının insansız hava araçlarıyla yapıldığı görülmektedir. Böylesi bir ilaçlama klasik tarım araçları kullanılarak yapılan ilaçlamadan daha iyi sonuç vermektedir. Çünkü klasik tarım araçları ekili alanlara zarar vermektedir.

Başka bir kullanım alanı olan eğlence sektörü içinde birçok avantajı vardır. Örneğin filmlerde, havadan çekilmesi gereken sahneler için eskiden helikopterler kullanılmıştır. Bu yöntem hem çok riskli hem de aşırı maliyetli bir yöntemdir. Şimdilerde insansız hava araçlarının bu alanda kullanılmasıyla bu sorun son derece basit bir şekilde halledilmektedir.

1.2.3. İnsansız hava aracı çeşitleri

İnsansız hava araçları literatürde iki özellik esas alınarak sınıflandırılmaktadır: Kanat yapılarına ve ağırlıklarına göre. Kanat yapılarına göre; sabit kanatlı, döner kanatlı ve hem döner hem de sabit kanatlı olmak üzere üç gruba ayrılmaktadır. İnsansız hava aracının sabit kanatlı olup olmamasının en belirgin sonucu enerji tüketimi ile ilgilidir. Sabit kanatlı bir insansız hava aracı daha az enerji tüketerek daha uzun uçuş süresi elde etmektedir. Aynı uçuş süresi için ağırlık bakımından birbirine yakın döner kanatlı insansız hava araçları, daha fazla enerji tüketmektedir. Bunun yanında döner kanatlı insansız hava aracının avantajı ise yüksek manevra kabiliyetidir. Sabit kanatlı insansız hava araçları, manevra kabiliyeti bakımından döner kanatlı insansız hava araçlarının gerisinde kalmaktadır. Anlatılan özelliklere ek olarak sabit kanatlı insansız hava araçları genellikle kalkış ve iniş için piste ihtiyaç duyar. Döner kanatlı insansız hava araçları ise kalkış ve iniş için piste ihtiyaç duymamaktadır. Şekil 1.3'te döner kanatlı, sabit kanatlı ve hem döner hem de sabit kanatlı insansız hava araçlarının örnekleri verilmiştir.

Şekil 1.3. İnsansız hava aracı örnekleri

İnsansız hava araçları ağırlıklarına göre ise NATO tarafından Çizelge 1.1'de görüldüğü gibi sınıflandırılmıştır. Temel olarak insansız hava araçları bu şekilde sınıflandırılsalar da kullandıkları yakıt çeşidine, otonom olup olmamasına, silahlı ya da silahsız olmasına ve kullanım amacına göre farklı kriterler göz önünde bulundurularak da sınıflandırılmaktadırlar.

Çizelge 1.1. İnsansız hava araçlarının ağırlıklarına göre NATO tarafından yapılan sınıflandırılma [1]

Sınıfı	Kategorisi	Görev Yüksekliği (ft)	Görev Yarıçapı (km)	Sivil Kategori
Sınıf 1 (150 kg ve altı)	Mikro (<2 kg)	<200 (AGL)	5 (LOS)	Ağırlık Sınıfı Grup 1 Küçük İHA
	Mini (2 kg – 20 kg)	<3.000 (AGL)	25 (LOS)	
	Midi (>20 kg)	<5.000 (AGL)	50 (LOS)	Ağırlık Sınıfı Grup 2 Hafif İHA
Sınıf 2 (150 kg- 600 kg arası)	Taktik	<10.000 (AGL)	200 (LOS)	Ağırlık Sınıfı Grup 3 İHA
Sınıf 3 (600 kg ve üstü)	Orta İrtifa Uzun Havada Kalış (MALE)	<45.000 (MSL)	Limitsiz (BLOS)	
	Yüksek İrtifa Uzun Havada kalış (HALE)	<65.000	Limitsiz (BLOS)	
	Darbe/Muharebe	<65.000	Limitsiz (BLOS)	

1.3. Quadcopterler

Quadcopterler bilinen diğer adıyla quadrotorlar, dört döner kanata sahip insansız hava araçlarıdır. Quad Latincede dörtlü anlamına gelmektedir. Bu yüzden Türkçeye dört rotorlu olarak çevrilmektedir. Quad ismini, üzerinde bulunan dört adet motordan ve copter ismini ise helicopterin kısaltmasından almaktadır. Benzer şekilde üç motorlu olanlara tricopter, altı motorlu olanlara hexacopter ve sekiz motorlulara ise octocopter isimleri verilmektedir. Açıklanan insansız hava araçlarına genel olarak multicopter denilmektedir.

Quadcopterler uçması için gereken kuvveti, dört motoruna bağlı pervanelerden sağlamaktadır. Uçmalarına yardımcı herhangi bir sabit kanat bulunmadığı için sabit kanatlılara göre daha hafif yapılmak zorundadır. Bu bağlamda quadcopterler, kanat

yapılarına göre döner kanatlı insansız hava aracı sınıfına girmektedir. Ağırlığı göz önüne alındığında ise genelde 25 kilogram altı grupta yer almaktadır.

1.3.1. Quadcopterlerin kullanım alanları

Quadcopterler, diğer insansız hava araçlarına göre boyut olarak küçük oldukları için daha dar alanlarda da kullanılabilmektedir. Ayrıca kalkış için piste ihtiyaç duymadıkları için kullanım alanları artmaktadır. Quadcopterlerin belli başlı kullanım alanları ve uygulamaları şu şekildedir:

Quadcopterler sağlık sektöründe, örneğin insanların vücut sıcaklıklarını ölçmek için, kullanılmaktadır. Quadcopterler kullanılarak haritalama işlemleri gerçekleştirilebilir. Quadcopterler ile geniş arazilerde hızlı ve ekonomik bir şekilde haritalama yapılabilir. Çeşitli sensörlerin yardımıyla haritalama üç boyutlu gerçekleştirilebilir. Tarım uygulamalarında quadcopterler kullanılarak çeşitli tarımsal faaliyetler gerçekleştirilebilmektedir. Quadcopterlerin tarım alanında kullanımına ilaçlama, verim analizi, gübreleme gibi uygulamalar örnek gösterilebilir. Quadcopterler, güvenlik güçleri tarafından sıkça kullanılmaktadır. Örneğin trafik denetlemelerinde trafik polisleri tarafından kullanılmaktadır.

Quadcopterlerin örneklerde anlatıldığı üzere oldukça geniş bir kullanım alanı bulunmaktadır. Ancak kullanım alanları sadece bu örnekler ile sınırlı değildir. Daha birçok kullanım alanları bulunmaktadır ve gün geçtikçe bu alanlar artmaktadır.

1.3.2. Diğer insansız hava araçlarına göre avantajları ve dezavantajları

Sabit kanadı bulunmayan hava araçlarının kontrolleri sabit kanadı bulunanlara göre çok daha zordur. Çünkü sabit kanatlar, hava aracının gökyüzünde kalmasına yardımcı olur. Döner kanatlı hava araçlarında böyle bir yardım söz konusu değildir. Quadcopter gibi döner kanatlı hava araçlarının havada asılı kalabilmeleri için gerekli kuvvet sadece dönen pervaneleri sayesinde elde edilmektedir. Uçuş esnasında sürekli olarak dönen pervaneler, birçok bozucu etkiye maruz kalmaktadır. Bozucu etkilere; kullanılan motor, gövde malzemesi gibi donanımsal faktörler ve rüzgâr gibi doğal faktörler de etki etmektedir. Dolayısıyla sistem sürekli olarak titreşime maruz kalmaktadır. Yaşanan titreşimler, quadcopterin kontrolünün zorlaşmasına sebep olmaktadır.

Kontrollerindeki zorluklara rağmen sıkça kullanılma sebeplerinden bir tanesi de kalkış ve iniş için piste ihtiyaç duymamalarıdır. Bu durum quadcopterlerin en büyük avantajıdır ve onların dar alanlarda kullanılabilmelerine imkân tanımaktadır. Diğer bir avantajı ise yüksek manevra kabiliyetleridir. Quadcopterlerin anlatılan iki avantajı da döner kanata sahip olmalarından kaynaklanmaktadır. Kontrolünü zorlaştıran sabit kanatlara sahip olmama durumu bu konuda bu hava araçlarına avantaj sağlanmaktadır. Quadcoprerlerin anlatılan avantajlarını sağlıklı bir şekilde kullanabilmek ve dezavantajlarından oldukça az etkilenmek için kontrol yöntemlerinden faydalanılmaktadır.

1.4. Quadcopterlerde Kullanılan Kontrol Yöntemleri

Quadcopterlerin kontrolleri için birçok yöntem kullanılmaktadır. Bu yöntemler arasında doğrusal ve doğrusal olmayan yöntemler bulunmaktadır. Doğrusal kontrol yöntemlerine örnek olarak PID (Oransal-İntegral-Türevsel Denetleyici), PD (Oransal-Türevsel Denetleyici) ve PI (Oransal-İntegral Denetleyici) kontrol yöntemleri verilebilir. Doğrusal olmayan kontrol yöntemlerine ise LQR (Lineer-Karesel Düzenleyici), LQG (Lineer-Karasel Gaussian Kontrol), KKK (Kayan Kipli Kontrol) ve bulanık mantık örnek gösterilebilir.

Kullanılan kontrol yöntemlerinin uygun bir şekilde çalışabilmesi için kontrolcü parametrelerinin doğru ayarlanması gerekmektedir. Örneğin PID kontrol, K_p, K_i ve K_d adı verilen kontrolcü parametrelerine sahiptir. Eğer bir sistem, PID kontrol yöntemi ile kontrol edilmek isteniyorsa bu kontrolcü parametrelerinin sistem için olabildiğince iyi hesaplanması gerekir. Aynı şekilde LQR kontrol için de durum böyledir.

Kontrolcü parametrelerinin hesaplanması için literatürde birçok yöntem bulunmaktadır. Örneğin deneme yanılma yöntemi bunlardan bir tanesidir. Deneme yanılma yöntemi, kullanılan en eski kontrolcü parametresi hesaplama yöntemidir. Oldukça zahmetli olan bu yöntemde kontrolcü parametreleri için farklı değerler denenir ve gözlem yapılır. Yapılan gözleme göre tekrar tekrar farklı değerler denenerek doğru sonuca ulaşılmaya çalışılır.

Deneme yanılma yönteminin anlatılan zorluklarından dolayı araştırmacılar, yeni yöntemler geliştirmektedirler. Örneğin PID kontrolcünün parametrelerini hesaplamak

için çeşitli numerik yöntemler geliştirilmiştir. Bu yöntemlere Cohen-Coon, Yuwana-Seborg ve Ziegler-Nichols yöntemleri örnek olarak verilebilir.

Numerik yöntemlerin zorluğu, kontrol edilen sistemin serbestlik derecesi arttıkça hesap yükünün de artmasıyla ortaya çıkmaktadır. Gelişen bilgisayar teknolojileriyle bu hesap yükü bilgisayarlara devredilmiştir. Ancak PID gibi kontrol yöntemleri, quadcopterler gibi doğrusal olmayan sistemlerde çok da istenilen sonucu verememektedir. Bundan dolayı son dönemde lineer olmayan kontrol yöntemleri kullanılmaya başlanmıştır. Kullanılan bu yöntemlerin başında bulanık mantık, kayan kipli kontrol gibi kontrol yöntemleri gelmektedir. Son dönemlerde ise LQR kontrol yönteminin kullanıldığı çalışmalar görülmektedir [2], [4], [8]-[14].

LQR kontrolcü parametreleri, deneme yanılma yöntemi kullanılarak belirlenebildiği gibi yapay zekâ optimizasyon algoritmaları kullanılarak da belirlenebilmektedir. İlk yöntemin zorluğu nedeniyle günümüzde, problem için oluşturulacak amaç fonksiyonu ve yapay zekâ optimizasyon algoritmaları ile mantıklı ve kabul edilebilir sonuçlar üretilebilmektedir.

1.5. Kaynak Araştırması

Kaynaklarda birçok araştırmacı quadcopterlerin kontrol yöntemleri ile ilgilenmişlerdir. Yapılan çalışmalardan bazıları şunlardır:

Reizenstein [2], yüksek lisans tezinde GPS verilerinin quadcopterin konumunu ve yörüngesini belirlemede yetersiz kalmasından dolayı filtreleme işlemi uygulamıştır. GPS verilerini, PID ve LQ kontrol kullanarak filtrelemiştir. Çalışmasını hem simülasyon ortamında hem de gerçek sistem kullanarak yapmıştır.

Oktay ve Köse [3], çalışmalarında farklı uçuş koşulları için quadcopterin dinamik modelini oluşturarak simülasyonunu gerçekleştirmiştir.

Bayrakçeken [4], doktora tezinde sabit bir quadcopter test düzeneği üretmiştir. Ürettiği test düzeneği üç serbestlik derecesine sahiptir. Quadcopter test düzeneğini kullanarak roll, pitch ve yaw açıları için kontrol uygulamaları gerçekleştirmiştir. PD kontrol, kayan kipli kontrol ve bulanık mantık olmak üzere üç farklı kontrol yöntemi kullanmıştır. Test düzeneğine bilgisayar yazılımları ile bağlanmak mümkündür. Bu sayede veriler hem bilgisayar ortamında okunurken hem de yapılan gözlem neticesinde, analizdeki verilerin gerçek yaşamda ne anlama geldiği kolayca görülebilmektedir.

Tran ve Nguyen [5], çalışmasında bir quadcoptere yörünge takibi yaptırmıştır. PID kontrol kullanarak quadcopteri kontrol etmiştir. Kullanılan kontrol yönteminin parametrelerini genetik algoritma ile hesaplamıştır. Çalışmasını hem simülasyon ortamında hem de gerçek bir quadccopter kullanarak gerçekleştirmiştir.

Beard [6], çalışmasında bir quadcopterin matematiksel modelini çıkararak onun davranışlarını kontrol etmiştir. Kontrol yöntemi olarak PD ve PID kontrol kullanmıştır.

Bayraktar ve Güldaş [7], insansız hava araçlarının yörünge takibindeki hataları en aza indirmek için bir çalışma gerçekleştirmiştir. Çalışmalarında PID kontrol kullanmışlardır. Bu çalışmayı bilgisayar ortamında gerçekleştirmişlerdir.

Ömürlü ve arkadaşları [8], üç serbestlik derecesine sahip bir quadcopter deney düzeneği geliştirmişlerdir. Sistemin kontrolü için bulanık mantık ve PD kontrolü birlikte kullanmışlardır.

Ateş [9], tez çalışmasında prototip helikopter modeli kullanmıştır. Bu sistemin matematiksel modelini deneysel yöntemlerle elde etmiştir. Elde etiği matematiksel model üzerinde kontrolcü geliştirmeye çalışmıştır. Ulaştığı veriler sonucunda PID kontrolü önermektedir. PID kontrolün parametrelerini geliştirdiği yapay sinir ağını kullanarak hesaplamıştır.

Demiryürek [10], yaptığı çalışmada bir quadcopterin matematiksel modelini çıkarmıştır. Çıkardığı model üzerinde doğrusal ve doğrusal olmayan kontrol yöntemlerini denemiştir. Bu yöntemlerin sistemde denenmesi sonucunda ulaşılan verileri karşılaştırmıştır. Quadcopterin hangi kontrol yöntemi kullanılarak kontrol edilmesinin daha iyi olacağını tespit etmeye çalışmıştır.

Matlab firmasının [11] geliştirmiş olduğu PARROT Minidrone isimli proje, kontrol mühendisleri için kontrol algoritmalarını gerçek bir drone üzerinde deneme imkânı sunmaktadır. PARROT Minidrone altı serbestlik derecesine sahiptir. Bu sayede roll, pitch ve yaw açılarının dışında ileri-geri yönlü hareket, aşağı-yukarı yönlü hareket, sağa-sola doğru ilerleme hareketleri de incelenebilmektedir. Ayrıca sadece yazılım kullanılarak, girilen fizik kanunlarını göz önünde bulundurarak, bilgisayarın ürettiği sanal veriler ile simülasyon gerçekleştirilebilmektedir. Son olarak sistem varsayılan olarak LQR kontrol ile kontrol edilebilmektedir.

Quanser firması [12] tarafından geliştirilen 3-DOF Hover isimli deney seti birçok araştırmacı tarafından kullanılmaktadır. Deney seti üç serbestlik derecesine sahiptir. Matlab ve simulink programları ile eş zamanlı çalışabilmektedir. Bu sayede deney setine sahipken gerçek dünya verileriyle çalışılabilmektedir. Deney seti olmadığında ise

bilgisayar ortamında bilgisayarın fizik kanunlarını göz önünde bulundurarak ürettiği sanal veriler kullanılarak simülasyon gerçekleştirilebilmektedir. Deney seti varsayılan olarak LQR kontrol ile kontrol edilmektedir.

Mohanty ve Misra [13], yaptıkları çalışmada üç serbestlik derecesine sahip 3-DOF Hover deney setini kullanmışlardır. Kullanılan sistemin kontrol edilebilmesi için LQR kontrolcü kullanmışlardır. LQR kontrolcü parametrelerini, yapay sinir ağı kullanarak hesaplamışlardır.

İçel ve arkadaşları [14], yaptıkları çalışmayı 3-DOF Hover deney setini kullanarak gerçekleştirmiştir. Bu sistem için kullanılan kontrolcü parametrelerini, parçacık sürü optimizasyonu ve temel optimizasyon algoritmalarını kullanarak hesaplamışlardır. Ulaştığı sonuçları gerçek sistem üzerinde denemişlerdir.

2. MATERYAL VE YÖNTEM

2.1. Materyal

Bu tez çalışmasında 3-DOF (Degrees Of Freedom) Hover eğitim setinin simülasyon dosyaları kullanılmıştır. Bütün uygulamalar bilgisayar ortamında gerçekleştirilmiştir. Kullanılan yazılımlar Matlab ve Simulink'tir.

2.1.1. 3-DOF Hover Simülasyonu

3-DOF Hover deney seti Quanser firması tarafından geliştirilmiş bir eğitim materyalidir. Geliştirilen deney seti gerçek zamanlı olarak Matlab/Simulink programı ile çalışabilmektedir. Bu sayede araştırmacılar kontrol yöntemlerini rahatlıkla gözlemleyebilmektedir. Şekil 2.1'de 3-DOF Hover deney sistemin bir görseli bulunmaktadır. Görselden anlaşılacağı üzere sistem sabit bir platformdur. Dolayısıyla 3 serbestlik derecesine sahiptir. 3-DOF Hover deney seti, bir quadcopterin uçuş esnasındaki davranışlarını çok dar bir alanda inceleyebilme imkânı sunmaktadır.

Şekil 2.1. 3-DOF Hover deney seti [12]

3-DOF Hover deney seti iki farklı şekilde kullanılabilmektedir. Bunlardan birincisi için 3-DOF Hover deney setine sahip olmak gerekmektedir. Birinci yöntem gerçek zamanlı bir şekilde deney düzeneği ile haberleşme imkânı sunmaktadır. Bahsedilen haberleşme Quarc adı verilen bir yazılım ile gerçekleşmektedir. Kurulan gerçek zamanlı iletişim sayesinde deney setinde bulunan sensörler tarafından okunan gerçek dünya verilerini kullanmak mümkün olmaktadır.

İkinci kullanım yönteminde ise 3-DOF Hover deney setinin sadece simülasyon dosyasına sahip olmak gerekmektedir. Deney düzeneği olmadan gerçekleştirilen simülasyonda ise gerçek veriler ile çalışılamamaktadır. Gerçek veriler yerine sisteme girilen fizik kanunları gözetilerek bilgisayar tarafından üretilen sanal veriler kullanılmaktadır. Bu tez çalışmasında bu yöntem kullanılmış, deneyler simülasyon üzerinde gerçekleştirilmiştir.

2.1.1.1. Deney setinin avantajları

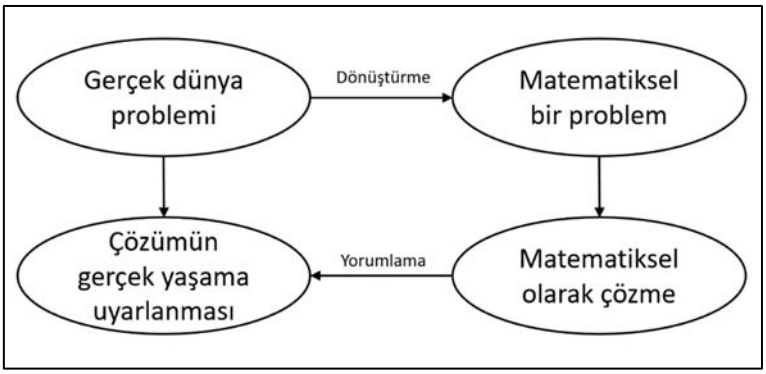
3-DOF Hover deney seti kullanıcısına birçok imkân sunmaktadır. Bu deney setinin sunduğu en büyük avantaj, dar bir alanda quadcopterlerin uçuş dinamikleri hakkında birçok bilgi elde edilebiliyor olmasıdır. Ayrıca bilgisayar programları ile gerçek zamanlı iletişim kurulabildiği için bilgisayarda görülen veriler gerçek hayatta kolayca anlamlandırılabilmektedir. Sahip olduğu en büyük dezavantaj ise maliyetidir. Bu dezavantajına karşılık simülasyon dosyaları gerçek sistem olmadan da çalıştırılabilir ve bu dosyalara ücretsiz olarak erişilebilmektedir.

2.1.1.2. Deney setinin matematiksel modeli

Deneme yanılma yöntemiyle üretim yapmak oldukça zordur. Günümüzde bir sistem inşa edilmeden önce çeşitli bilgisayar yazılımlarından faydalanılır. Sistemin inşasında faydalanılan bilgisayar yazılımları ile bu deneme yanılma yöntemleri bilgisayar ortamına taşınır. Bilgisayar ortamında gerçekleştirilen simülasyonlar sonucunda en iyi sonuç veren sistem üretilir. Bu işlemler sayesinde, geliştirilmesi planlanan sistemin dünya şartlarından ya da tasarım tercihlerinden dolayı yaşayabileceği problemler, henüz işin başındayken tespit edilir. Yaşanması muhtemel problemler üretim sürecine başlamadan öngörülebilir. Sonuç olarak yapılacak olan çalışmanın temeli sağlam bir şekilde atılmış olur.

Başarılı bir simülasyon için oluşturulan sistemin doğru bir şekilde bilgisayar ortamına aktarılması gerekmektedir. Sistemin başarılı bir şekilde bilgisayar ortamına aktarılabilmesi için sistemin matematiksel modeli gerçeğe en yakın şekilde çıkarılmalıdır.

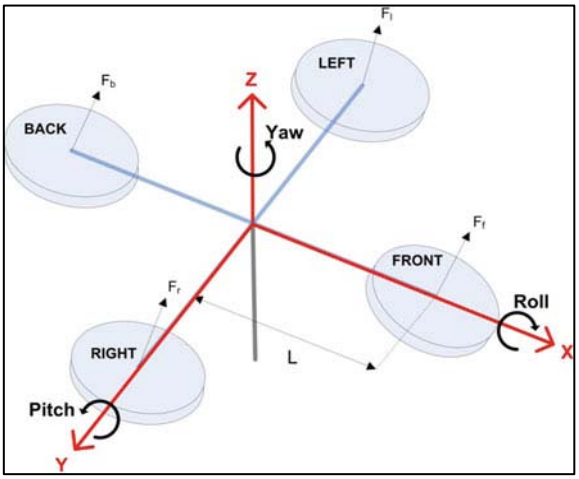
Matematiksel model, en basit tanımıyla oluşturulması planlanan ya da oluşturulan gerçek bir sistemin sadece matematiksel ifadeler ile betimlenmesine denir. Şekil 2.2'de matematiksel modelleme süreci görülmektedir. Matematiksel modelleme sürecinde gerçek dünya problemini, matematiksel modele dönüştürürken temel fizik yasalarından faydalanılır.



Şekil 2.2. Matematiksel modelleme süreci [15]

İncelenen sistemler, quadcopterler gibi doğrusal olmadığında yüzde yüz gerçeklikle matematiksel model çıkarmak oldukça zordur. İster istemez matematiksel model çıkarılırken birtakım varsayımlar kabul edilir. Sistemde doğrusal olmayan her şey doğrusal kabul edilir. Sistem olabildiğince doğrusallaştırılmaya çalışılır.

Bu tez çalışmasında simülasyon dosyaları kullanılan 3-DOF Hover deney setinin izometrik görünümünün serbest cisim diyagramı Şekil 2.3'te görülmektedir.



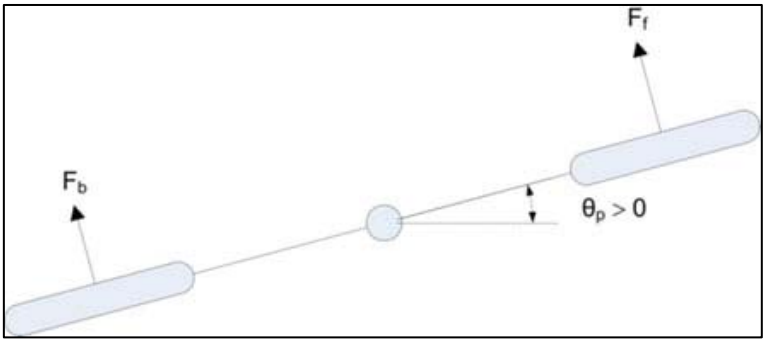
Şekil 2.3. 3-DOF Hover deney setinin izometrik görünümünün serbest cisim diyagramı [13]

3-DOF Hover deney setinin matematiksel modeli çıkarılırken şu bilgilerin sağlandığı kabul edilmektedir.

- 3-DOF Hover deney seti rijit ve simetriktir.
- 3-DOF Hover deney setinin ağırlık merkezi X, Y ve Z eksenlerinin çakıştığı noktadadır.
- 3-DOF Hover deney setinin motorları rijit bir şekilde gövdeye sabitlenmiştir.
- 3-DOF Hover deney seti yataydır. Yani yere paraleldir. ($\theta_p = \theta_r = 0$)
- 3-DOF Hover deney setinin gövdesi saat yönünün tersine döndüğünde yaw açısı artmaktadır.
- 3-DOF Hover deney seti Y-ekseni etrafında saat yönünde döndürüldüğünde, pitch açısı artmaktadır.
- 3-DOF Hover deney seti X-ekseni etrafında saat yönünde döndürüldüğünde, roll açısı artmaktadır.

3-DOF Hover deney setinin dört adet DC motoru bulunmaktadır. DC motorlar Şekil 2.3'te görüldüğü üzere ön(front), arka(back), sağ(right) ve sol(left) olarak isimlendirilmiştir. Bu motorlara pozitif DC gerilim uygulandığında motorlar sistemi harekete geçirecek itme kuvvetini sağlamaktadır. Bahsedilen motorların oluşturduğu itme kuvveti sırasıyla F_f, F_b, F_r ve F_l olarak isimlendirilmektedir.

Pitch açısının kontrolünden daha çok ön ve arka motorlar sorumludur. Öndeki motorun oluşturduğu itme kuvveti, arkadaki motorun oluşturduğu itme kuvvetinden daha büyük olduğu durumda pitch açısı artmaktadır. Aksi durumda ise pitch açısı azalmaktadır. Şekil 2.4'te öndeki motorun oluşturduğu kuvvetin, arkadaki motorun oluşturduğu kuvvetten büyük olduğu durum için 3-DOF Hover deney setinin sağ taraftan görünümünün serbest cisim diyagramı verilmiştir.



Şekil 2.4. 3-DOF Hover deney setinin sağ taraftan görünümünün serbest cisim diyagramı [13]

Roll açısının kontrolünden ise daha çok sağ ve sol motorlar sorumludur. Sağdaki motorun oluşturduğu itme kuvveti, soldaki motorun oluşturduğu itme kuvvetinden daha büyük olduğu durumda roll açısı artmaktadır. Aksi durumda ise roll açısı azalmaktadır. Görüldüğü üzere pitch ve roll açılarının değişimleri benzerlik göstermektedir. Bu benzerlikten dolayı Denklem (2.1) her iki açı içinde geçerlidir.

$$J\ddot{\theta} = \Delta FL \tag{2.1}$$

Burada J, eksene göre eylemsizlik momentini; $\ddot{\theta}$ açısal ivmeyi; ΔF, kuvvetler arasındaki farkı ve L, motorun merkeze olan uzaklığını ifade etmektedir. Denklem (2.1), Şekil 2.4'teki sistemin sağ taraftan görünümünün serbest cisim diyagramına uygulanırsa Denklem (2.2) elde edilir.

$$J_p\ddot{\theta_p} = K_f \times (V_f - V_b) \tag{2.2}$$

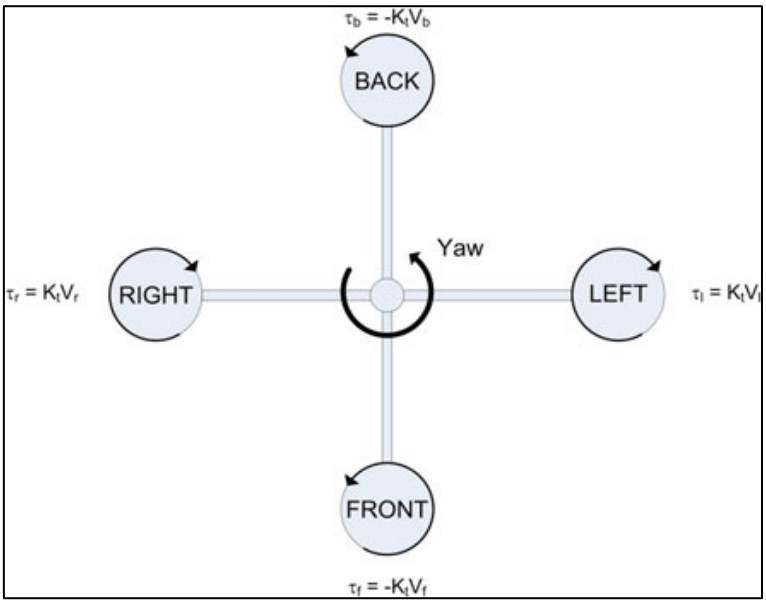
Burada K_f, itki kuvveti sabitini; V_f, öndeki motora verilen gerilimi ve V_b, arkadaki motora verilen gerilimi ifade etmektedir. Benzer şekilde Denklem (2.1) sistemin önden görünümü için sağ ve sol motorlara uygulanırsa Denklem (2.3) elde edilir.

$$J_r\ddot{\theta_r} = K_f \times (V_r - V_l) \tag{2.3}$$

Burada K_f, itki kuvveti sabitini; V_r, sağdaki motora verilen gerilimi ve V_l, soldaki motora verilen gerilimi ifade etmektedir. Sonuç olarak roll ve pitch eksenindeki hareketleri ifade etmek için denklemler bu şekilde oluşturulur. Yaw ekseni ise önceki iki eksenden farklıdır.

Şekil 2.5'te 3-DOF Hover deney setinin üstten görünümünün serbest cisim diyagramı verilmiştir. Serbest cisim diyagramında görüleceği üzere karşılıklı motorlar aynı yönde dönmektedir. Ayrıca ardışık motorların zıt yönde döndükleri görülmektedir. Motorların merkeze olan uzaklıkları eşittir. Bu tasarımsal simetri sayesinde kuvvet ve moment dengesi sağlanmaktadır. Yaw ekseninde hareket sağlamak için bu dengeyi bozmak gerekmektedir ve yaw eksenindeki hareket matematiksel olarak Denklem (2.4)'te görüldüğü gibi ifade edilir.

$$J_y\ddot{\theta_y} = \Delta\tau = \tau_l + \tau_r - \tau_b - \tau_f \tag{2.4}$$



Şekil 2.5. 3-DOF Hover deney setinin üstten görünümünün serbest cisim diyagramı [13]

Burada τ_l, soldaki motorun oluşturduğu momenti; τ_r, sağdaki motorun oluşturduğu momenti ifade etmektedir. Sol ve sağ motorlar saat yönünde döndükleri için ürettikleri momentler pozitif alınmıştır. τ_b, arkadaki motorun oluşturduğu momenti; τ_f, öndeki motorun oluşturduğu momenti ifade etmektedir. Ön ve arka motorlar saat yönünün tersinde döndükleri için ürettikleri momentler negatif alınmıştır.

Denklem (2.4) momenti oluşturan değerler motorlara verilen gerilim ve motorun itme momenti sabiti şeklinde açıkça yazılırsa Denklem (2.5) elde edilir.

$$J_y\ddot{\theta_y} = K_t(V_r + V_l) - K_t(V_f + V_b) \tag{2.5}$$

Denklem (2.5)'te K_t, motorun itme momenti sabitini ifade etmektedir. Denklem (2.1) ve Denklem (2.5) arasındaki denklemlerde kullanılan sisteme ait parametrelerin tamamı Çizelge 2.1'de verilmiştir.

Çizelge 2.1. 3-DOF Hover deney setine ait parametreler [13]

Sembol	Tanımlama	Değer	Birim
$K_{t,n}$	Ters yönlü motorun itme momenti sabiti	0.0036	$N \cdot m/V$
$K_{t,c}$	Normal yönlü motorun itme momenti sabiti	0.0036	$N \cdot m/V$
K_f	İtki kuvveti sabitini	0.1188	N/V
l	Motorların ağırlık merkezine olan uzaklığı	0.197	m
J_r	Roll ekseni etrafında eşdeğer atalet momenti	0.0552	$kg \cdot m^2$
J_p	Pitch ekseni etrafında eşdeğer atalet momenti	0.0552	$kg \cdot m^2$
J_y	Yaw ekseni etrafında eşdeğer atalet momenti	0.110	$kg \cdot m^2$

Hareket denklemleri bu şekilde çıkarılan sisteme LQR kontrol uygulanabilmesi için sistemin durum uzay modelinin çıkarılması gerekmektedir. Durum uzay modeli, durum değişkeninin zamana göre türevinin, durum değişkenleri cinsinden ifade edilebilmesi demektir. Sistemin durum uzay modeli çıkarıldığında sistem Denklem (2.6) ve Denklem (2.7)'deki gibi ifade edilebilmektedir.

$$\dot{x} = Ax + Bu \tag{2.6}$$

$$y = Cx + Du \tag{2.7}$$

3-DOF Hover deney setinin durum değişkenleri Denklem (2.8)'deki gibidir.

$$x^T = [\theta_y\ \theta_p\ \theta_r \dot{\theta}_y\ \dot{\theta}_p\ \dot{\theta}_r] \tag{2.8}$$

3-DOF Hover deney setinin çıkış değişkenleri Denklem (2.9)'daki gibidir.

$$y^T = [\theta_y\ \theta_p\ \theta_r] \tag{2.9}$$

3-DOF Hover deney setinin kontrol vektörü Denklem (2.10)'daki gibidir.

$$u^T = [V_f\ V_b V_r\ V_l] \tag{2.10}$$

Oluşturulan hareket denklemleri ve Çizelge 2.1'deki veriler kullanılarak sistemin durum uzay matrisleri Denklem (2.11) - (2.14)'deki gibi bulunur. Ulaşılan durum uzay modeli ve matrisler, kullanılan simülasyon dosyalarında Quanser firması tarafından hazır olarak sunulmuştur.

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.11}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{K_t}{J_y} & -\frac{K_t}{J_y} & \frac{K_t}{J_y} & \frac{K_t}{J_y} \\ \frac{L \times K_f}{J_p} & \frac{L \times K_f}{J_p} & 0 & 0 \\ 0 & 0 & \frac{L \times K_f}{J_r} & \frac{L \times K_f}{J_r} \end{bmatrix} \tag{2.12}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad (2.13)$$

$$D = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (2.14)$$

2.1.2. Kullanılan Diğer Materyaller

Bu tez çalışması için Matlab yazılımı kullanılmıştır. Matlab, günümüzde kontrol mühendisliği, sayısal analiz, lineer cebir, görüntü işleme gibi birçok bilim dalında kullanılmaktadır ve literatürde kabul görmüş bir yazılımdır. Kullanılan bir diğer yazılım ise Simulink'tir. Simulink dinamik sistemlerin modellenmesinde ve simüle edilmesinde kullanılan bir yazılımdır.

Bu tez çalışması kapsamında kullanılan yazımlar Lenovo Thinkpad Yoga 12 isimli bilgisayar üzerinde çalıştırılmıştır. Tez kapsamında gerçekleştirilen bütün testler bahsedilen bilgisayar üzerinde yapılmıştır. Bilgisayar İntel Core i7 4510u işlemciye ve DDR3 8GBytes rame sahiptir.

2.2. Yöntem

3-DOF Hover sistemi üç serbestlik derecesine sahip bir eğitim setidir. Üç serbestlik derecesine sahip olan sistemin kontrolünün gerçekleştirilebilmesi için LQR kontrol yöntemi kullanılmıştır. LQR kontrolcü parametrelerini deneme yanılma yöntemini ile belirlemek oldukça zordur. Bu nedenle bu parametreler, yapay zekâ optimizasyon algoritmaları kullanılarak hesaplanmıştır.

2.2.1. LQR Kontrol

Kısaca LQR olarak kısaltılmış olan Doğrusal-Karesel Düzenleyici (Linear-Quadratic Regulator), dinamik bir sistemin minimum maliyet ile çalışması için geliştirilmiş bir kontrol yöntemidir. LQR kontrolün uygulanabilmesi için oluşturulan sistemin Denklem (2.15)'teki gibi durum değişkeninin zamana göre türevinin, durum

değişkenleri cinsinden ifade edilebilmesi gerekmektedir. Bu yapılan işleme sistemin durum uzay modelini çıkarma denir. Maliyeti en aza indirgemek için ise Denklem (2.15)'teki durum uzak modelinin A ve B matrislerinden faydalanılır. Burada u sisteme verilen girişleri temsil etmektedir ve Denklem (2.16)'da nasıl hesaplandığı açıklanmıştır.

$$\dot{x} = Ax + Bu \tag{2.15}$$

$$u = -Kx \tag{2.16}$$

Maliyet fonksiyonu ise Denklem (2.17)'de görüldüğü gibidir. Burada Q ve R, LQR kontrolcü parametrelerini temsil etmektedir.

$$J = \int_0^\infty (x^T Q x + u^T R u) dt \tag{2.17}$$

$$x[k+1] = Ax[k] + Bu[k] \tag{2.18}$$

$$J = \sum_{k=0}^{\infty} (x^T Q x + u^T R u) \tag{2.19}$$

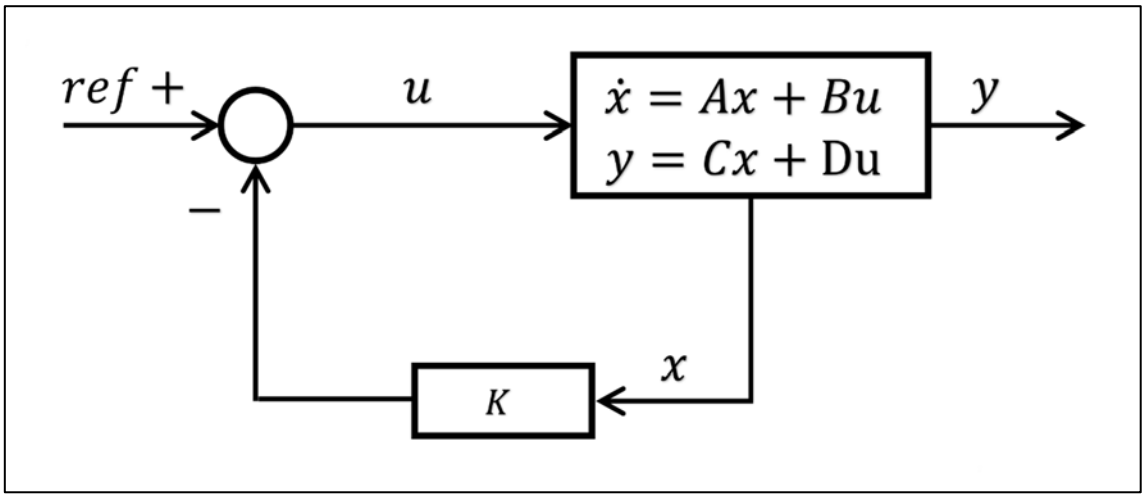
Bu maliyet fonksiyonunun minimuma indirilebilmesi için Denklem (2.22)'de ulaşılan Riccati denkleminden faydalanılmaktadır. Burada K denklemin kökünü ifade etmektedir.

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{2.20}$$

$$A^T PA - PA^T PB(B^T PB + R)^{-1}B^T PA + Q = 0 \tag{2.21}$$

$$K = (B^T PB + R)^{-1}B^T PA \tag{2.22}$$

LQR kontrolün sisteme uygulanabilmesi için oluşturulan örnek bir blok diyagramı Şekil 2.6'da verilmiştir. LQR kontrolü diğer kontrol yöntemlerinden ayıran en büyük özelliği kontrol yönteminin enerji dönüşümlerinden faydalanılarak ortaya çıkmasıdır. Bu durum da gerçek hayatta mümkün olmayan çözümlerin kendiliğinden elenmesini sağlamaktadır. Sonuç olarak LQR kontrol gerçek dünyada uygulanabilir çözümler sunmaktadır. Diğer kontrol yöntemlerinde ise örneğin PID kontrolde bu işlem satürasyon denilen bloklar kullanılarak yapay olarak sağlanmaya çalışılmaktadır. Oysa LQR kontrol, örneğin kinetik enerjinin potansiyel enerjiye dönüştürüldüğü bir çalışmada, bu dönüşümün kütleden dolayı doğacak sınırlarla gerçekleşeceği göz önünde bulundurulur. LQR kontrol bu sınırlar dahilinde çalışmalar yapılmasına olanak sağlar.



Şekil 2.6. LQR kontrolün blok diyagramı

2.2.2. Yapay Zekâ Optimizasyon Algoritmaları

Optimizasyon, bir sistemde belirlenen sınırlar içerisinde en uygun sonuca ulaşabilmek için yapılan çalışmalara verilen isimdir. En uygun sonuç probleme göre değişebilmektedir. Örneğin problem verimlilik ile alakalı ise en fazla verimi sağlayan çözüm aranmaktadır. Dolayısıyla konu verim olduğunda en büyük değeri veren çözüm aranır ve bu işlem maksimizasyon olarak adlandırılır. Başka bir örnekte ise problem, maliyetin düşürülmesidir ve en küçük değeri veren çözüm aranır. Bu durumda ise yapılan işlem minimizasyon olarak adlandırılır.

Klasik yöntemler ile çözülmek istenen problemin parametre sayısı arttıkça doğru çözüme ulaşmak zorlaşmaktadır. Eğer çözülmek istenen probleme ve problemi etkileyen parametrelere elektronik bir form kazandırılabilirse bilgisayarlar mükemmel bir optimizasyon aracı olur. Birtakım bilgiler bilgisayara girilir ve bir çözüm elde edilir. Bilgisayarın girilen verilere karşın uygulanabilir bir sürede, kapsamlı bir arama gerçekleştirip çözüm bulabilmesi için yapay zekâ optimizasyon algoritmaları kullanılmaktadır.

Optimizasyon problemlerinin çözülmesi için geliştirilen yapay zekâ optimizasyon algoritmaları meta sezgisel arama yöntemleri olarak bilinmektedir. Meta sezgisel arama yöntemleri mantıklı ihtimalleri kullanarak doğru çözüme yaklaşmayı hedefler. Meta sezgisel bir yaklaşım ile gerçekleştirilen arama işleminde, belki yüzde yüz doğru sonuca ulaşılamasa da uygulanabilir bir süre içerisinde doğruya yani global en iyiye yakın bir

cevap bulunur. Dolayısıyla meta sezgisel arama yöntemlerinin lokal minimumlardan ya da lokal maksimumlardan kurtulmasını sağlayan yapılar bulunmaktadır.

Bu tez çalışması kapsamında dinamik bir sistemin kontrol edilmesi için kullanılan LQR kontrol yönteminin parametreleri yapay zekâ optimizasyon algoritmaları ile belirlenmeye çalışılmıştır. LQR kontrolün kontrolcü parametreleri Q ve R matrisleridir. Üzerinde çalışılan sistem için Denklem (2.23) ve Denklem (2.24)'teki gibi Q ve R matrisleri aranmaktadır. Bu amaç ile matrislerin diegonal elemanlarında optimizasyon işlemi gerçekleştirilecektir. Dolayısıyla 10 boyutlu bir problem, yapay zekâ optimizasyon algoritmaları ile çözülmeye çalışılacaktır.

$$Q = \begin{bmatrix} 500 & 0 & 0 & 0 & 0 & 0 \\ 0 & 350 & 0 & 0 & 0 & 0 \\ 0 & 0 & 350 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 20 & 0 \\ 0 & 0 & 0 & 0 & 0 & 20 \end{bmatrix} \tag{2.23}$$

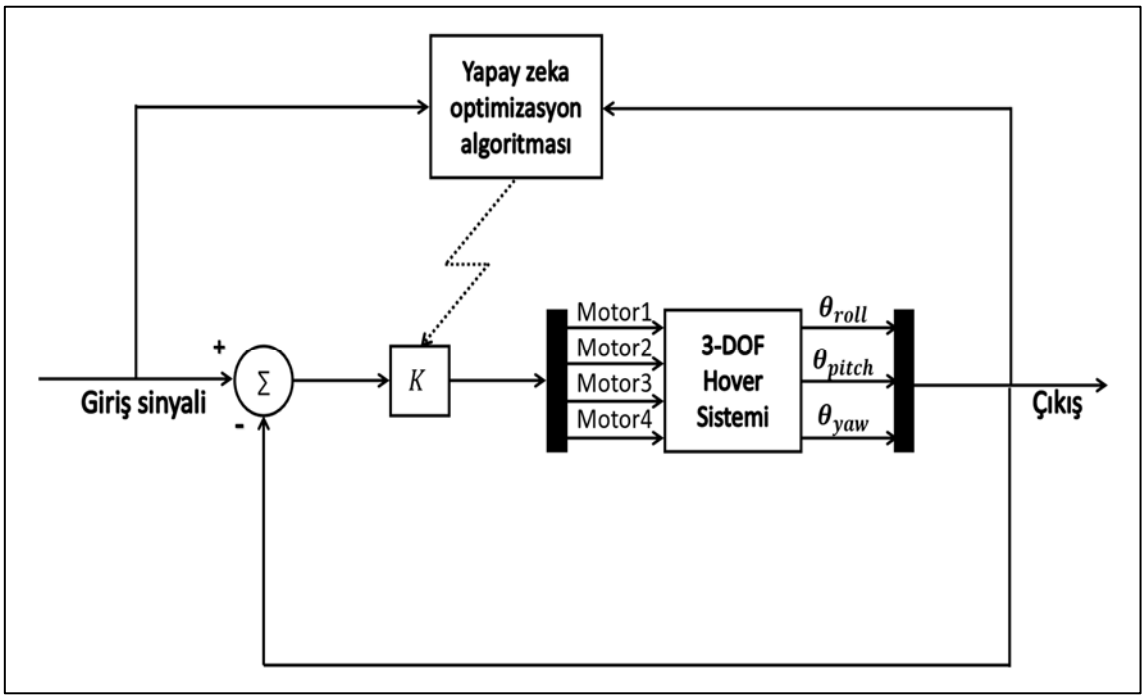
$$R = \begin{bmatrix} 0{,}01 & 0 & 0 & 0 \\ 0 & 0{,}01 & 0 & 0 \\ 0 & 0 & 0{,}01 & 0 \\ 0 & 0 & 0 & 0{,}01 \end{bmatrix} \tag{2.24}$$

Bu çalışmada kullanılan yapay zekâ optimizasyon algoritmaları ise aşağıda listelenmiştir:

- Yusufçuk algoritması
- Genetik algoritma
- Parçacık sürü optimizasyonu algoritması
- Benzetilmiş tavlama algoritması
- Yapay arı kolonisi algoritması
- Diferansiyel gelişim algoritması
- Gri kurt optimizasyon algoritması
- Hiyerarşik Benzetilmiş tavlama – Gri kurt optimizasyon algoritması

3. UYGULAMALAR

Önceki bölümlerde 3-DOF Hover simülasyonundan ve LQR kontrol yönteminden bahsedildi. Kullanılan yapay zekâ optimizasyon algoritmaları listelendi. Bu bölümde ise bahsedilen tüm konular bir çatı altında birleştirilmektedir. Şekil 3.1'de çatı ifadesini özetleyen bir kontrol diyagramı görülmektedir.



Şekil 3.1. Algoritmaların sisteme uygulanması

Kontrol diyagramında kontrolü yapılmak istenen sistem 3-DOF Hover deney setidir. Bu çalışmada, doğrusal olmayan yapısıyla bilinen quadcopterlerin davranışlarının incelenebilmesi için geliştirilmiş bir eğitim seti olan 3-DOF Hover deney setinin simülasyon dosyaları kullanılmıştır. Üç serbestlik derecesine sahip olan bu sistemin verilen giriş sinyallerine karşı mümkün olan en iyi sistem cevabını vermesi istenmektedir. Quadcopterler gibi bu deney seti de doğrusal değildir. Bu tür sistemler doğrusal olmadıklarından dolayı doğrusal kontrol yöntemleriyle kontrol edilmeleri bir hayli güçtür. Dolayısıyla bahsedilen sistemin kontrolü için doğrusal kontrol yöntemlerinden farklı bir yöntem tercih edilmiştir. Tercih edilen kontrol yöntemi Şekil 3.1'deki diyagramda Ricatti denkleminin kökü olan K matrisinden anlaşılacağı üzere LQR kontrol yöntemidir.

LQR kontrol yöntemi hem doğrusal hem de doğrusal olmayan sistemlerin kontrolü için geliştirilmiş yenilikçi bir kontrol yöntemidir. Ancak bu kontrol yönteminin

kontrolcü parametresi olan Q ve R matrislerinin numerik metotlar ile hesaplanabilmesi oldukça zordur. İfade edilen hesaplama zorluğunu aşmak için Q ve R kontrolcü parametreleri, diyagramda görüldüğü üzere yapay zekâ optimizasyon algoritmaları ile belirlenmiştir.

3.1. Uygulamalar Öncesinde Yapılan Hazırlıklar

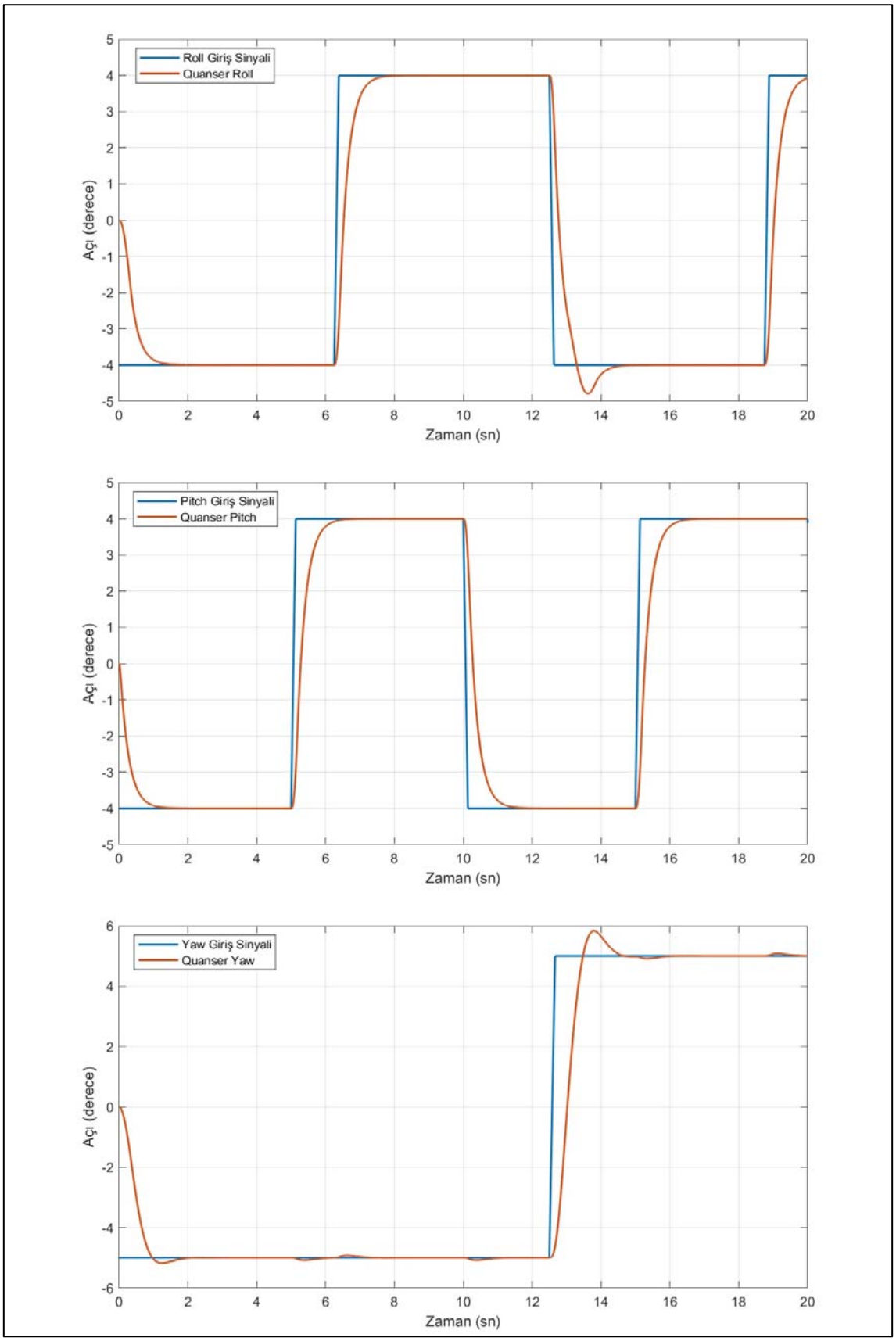
Amaç fonksiyonu, maliyet fonksiyonu ya da kazanç fonksiyonu olarak adlandırılan fonksiyonlar problemi temsil eden matematiksel ifadelerdir. Algoritmalar, problem için özel hazırlanmış amaç fonksiyonunu kullanarak belirlenen problem için çözüm üretirler ve bu çözümün uygunluk değerini hesaplarlar. İteratif olarak uygunluk değerini daha iyi bir noktaya taşımaya çalışırlar. Algoritmalar sadece amaç fonksiyonunun iyileştirilmesinden sorumludurlar. Bu nedenle, amaç fonksiyonunun çözülmesi istenen problemi en iyi bir şekilde temsil etmesi gerekmektedir.

Yapay zekâ optimizasyon algoritmalarının çalışabilmeleri için bir diğer önemli konu ise arama uzayının belirlenmesidir. Arama uzayı problemin çözümü için üretilecek değerlerin sınırlarını belirlemektedir. Belirlenen sınır ne çok küçük ne de çok büyük olmalıdır. Çok küçük olduğunda arama kısıtlı bir uzayda gerçekleşir ve problem için yeterli sonuca ulaşılamayabilir. Çok büyük olduğunda ise çözüm için gerçekte mantıklı olmayan değerler üzerinde arama yapılacağı için zaman kaybı olabilir.

Yapay zekâ optimizasyon algoritmaları kendilerine özgü birtakım parametrelere sahiptir. Bu parametreler ne kadar uygun seçilirse algoritmanın performansı da o oranda artar. Yapay zekâ optimizasyon algoritmalarının kendilerine özgü parametrelerinin, üzerinde çalışılan problem için, en uygun şekilde belirlenmesi gerekir.

3.1.1. Amaç Fonksiyonunun Belirlenmesi

Şekil 3.2'de 3-DOF Hover sisteminin Quanser firması tarafından önerilen Q ve R matrisiyle simülasyonundan elde edilen sistem cevabı görülmektedir. Şekilde mavi ile gösterilen sinyal, giriş sinyalidir. Kırmızı ile gösterilen sinyal ise, kontrolcüye verilen Q ve R matrislerinin kullanılmasıyla elde edilen sistem cevabı, çıkış sinyalidir. Sistemin sağlıklı bir şekilde çalışabilmesi için çıkış sinyalinin giriş sinyaline uyumlu olması gerekmektedir. Yani, giriş sinyali ile çıkış sinyali arasındaki hatanın mümkün olan en az seviyede olması gerekir.



Şekil 3.2. Quanser firması tarafından varsayılan olarak sunulan Q ve R matrisleriyle sistemin simülasyonu sonucunda elde edilen sistem cevabı [12]

Şekil 3.2'deki giriş sinyalleri incelendiğinde roll açısı için üretilen giriş sinyalinin frekansı 0.08 Hz ve genliği 4 derecedir. Roll açısı için üretilen sinyalin bir periyodu 12.5 saniye sürmektedir. Pitch açısı için üretilen giriş sinyali incelendiğinde sinyalin 0.1 Hz frekansa ve 4 derece genliğe sahip olduğu görülmektedir. Pitch açısı için üretilen giriş sinyalinin bir periyodu için geçen zaman 10 saniyedir. Son olarak yaw açısı için üretilen giriş sinyaline bakıldığında sinyalin 0.04 Hz frekansa sahip olduğu ve genliğinin 5 derece olduğu görülmektedir. Yaw sinyali için üretilen giriş sinyalinin bir periyodu için geçmesi gereken süre 25 saniyedir. Bu bahsedilen sinyaller simülasyon dosyalarında varsayılan olarak gelen sinyallerdir. Literatürde bahsedilen giriş sinyallerine göre sistem kontrol edilmeye çalışıldığı için bu tez çalışmasında da giriş sinyalleri değiştirilmemiştir.

Çizelge 3.1'de Şekil 3.2'deki giriş sinyalleri için üretilen çıkış sinyallerinin yorumlanabilmesi için sistem cevaplarının oturma zamanı, yükselme zamanı ve üst aşım değeri verilmiştir. Burada roll giriş sinyalinin son dalgasının 18.75. saniyede, pitch giriş sinyalinin son dalgasının 15.00. saniyede ve yaw giriş sinyalinin son dalgasının 12.50. saniyede başladığına dikkat edilmelidir. Çünkü Çizelge 3.1'deki oturma zamanı verileri, sinyallerin son dalgalarına yerleştiği zamanı göstermektedir.

Çizelge 3.1. Şekil 3.2'deki sistem cevabı hakkında bazı bilgiler

Oturma Zamanı (sn)			Yükselme Zamanı (sn)			Üst Aşım (deg)		
Roll	Pitch	Yaw	Roll	Pitch	Yaw	Roll	Pitch	Yaw
19.7107	16.0741	14.3349	0.5042	0.5440	0.3475	0.7890	0.0003	0.8323

Çizelge 3.1 ve Şekil 3.2'de roll açısı için üretilen karesel giriş sinyaline karşı sistemin vermiş olduğu cevap görülmektedir. İlk iki dalga için sistem giriş sinyaline uygun cevaplar üretmektedir. Üçüncü dalgada ise bir üst aşım söz konusudur. Üst aşım, sistem cevabının ulaştığı maksimum değer ile sistemin kararlı halde olduğu durum arasındaki fark olarak tanımlanmaktadır. Gerçekleşen üst aşımın sayısal değeri Çizelge 3.1'de 0.7890 derece olarak verilmiştir. Uçuş güvenliği açısından üst aşımlar tehlikeli olabilmektedir. Bu denli küçük bir üst aşım simülasyon için sorun olmasa da gerçek sistemde bu üst aşım çok daha büyük olacaktır. Dolayısıyla kontrolcü parametrelerinin optimize edilmesiyle bu gibi üst aşımların giderilmesi gerekmektedir.

Sistem cevabının ulaşılmak istenen referans değere %2 tolerans ile yerleşmesi olarak tanımlanan oturma zamanı, Çizelge 3.1'de görülmektedir. Son roll giriş dalgasının 18.75. saniyede başladığı bilinmektedir. Bu açıdan bakıldığında sistem roll için istenilen

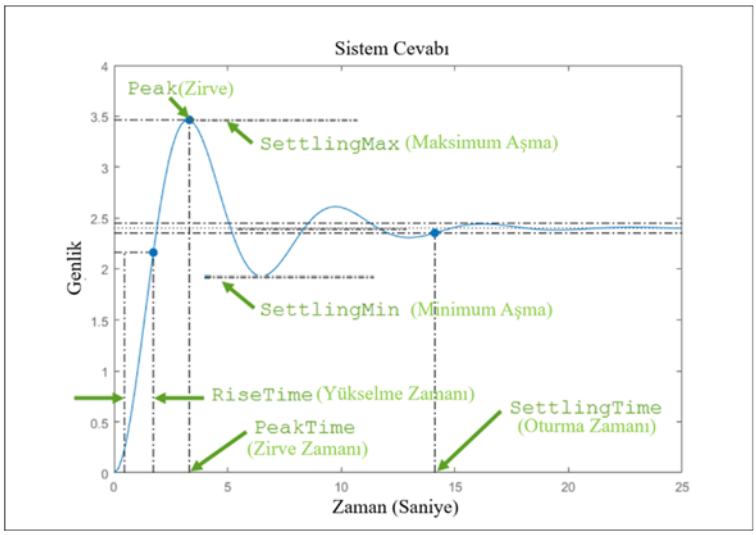
açı değerine 0.9607 saniyede yerleşmektedir. Oturma zamanının kısa sürmesi sistem cevabının sağlıklı olduğunu göstermektedir. Oluşturulacak amaç fonksiyonu, oturma zamanının mümkün olan en kısa zamanda gerçekleşmesini sağlayacak nitelikte olmalıdır.

Yaw açısı için sisteme uygulanan karesel giriş sinyalleri ve sistemin bu sinyale verdiği yanıt incelendiğinde üst aşımlar dikkat çekmektedir. Ancak önceki bölümlerde anlatıldığı üzere yaw açısı Z-ekseninde dönmeyi ifade eder ve Z-ekseni, yer çekimi ivmesinin yönüyle çakışıktır. Dolayısıyla Z-ekseninde gerçekleşecek üst aşımlar, büyük olmadığı sürece uçuş güvenliği açısından sorun oluşturmayacaktır. Z-ekseninin bu özel durumundan dolayı diğer açıların sistem cevaplarını düzeltmek için yaw açısının sistem cevabına, gerektiğinde, toleranslı yaklaşılabilir. Anlaşılacağı üzere oluşturulacak amaç fonksiyonunun öncelikli hedefi roll ve pitch açılarının sistem cevaplarını düzeltmek olacaktır.

Sistemin pitch açısı için verilen karesel giriş sinyaline cevabı neredeyse hatasızdır. Üç çıkış sinyali arasında en iyi üst aşım değerine sahiptir. Neredeyse hiç üst aşım (0.0003 derece) yapmamaktadır. Ancak roll ve yaw sistem cevaplarının üst aşımları incelendiğinde pitch açısının üst aşımından çok daha yüksek üst aşımlar yaptıkları görülmektedir. Oluşturulacak amaç fonksiyonu, gerekirse pitch açısının üst aşımını biraz yükselterek diğer iki açının üst aşımını azaltabilmelidir. Bu sayede üç açıda meydana gelecek üst aşım eşit seviyede olabilir.

Çizelge 3.1'de görülen bir diğer veri yükselme zamanıdır. Yükselme zamanı sistem cevabının ilk üst aşıma ulaştığı andır. Sistemin kararlı bir şekilde çalışabilmesi için yükselme zamanının mümkün olan en kısa zamanda gerçekleşmesi istenir. Burada da öncelik roll ve pitch açısının yükselme zamanlarına verilmelidir. Eğer roll ve pitch açısının yükselme zamanı azaltılamazsa, Z-ekseninin özel durumundan dolayı yaw açısının yükselme zamanı biraz uzatılarak roll ve pitch açısının yükselme zamanı azaltılmaya çalışılmalıdır. Oluşturulacak amaç fonksiyonu bu imkânı sağlamalıdır.

Şekil 3.3'te yukarıda bahsedilen kavramlara ek olarak amaç fonksiyonun oluşturulmasında kullanılan birçok kavram bulunmaktadır. Amaç fonksiyonu, örneklerde bir kısmı anlatılan koşulların gerçekleşmesini sağlamada yardımcı olmalıdır. Bu sayede optimizasyon sonucunda daha düzgün bir sistem cevabına ulaşılabilir ve sistem daha kararlı hale getirilebilir.



Şekil 3.3. Sistem cevabı hakkında bazı kavramlar [16]

Dinamik bir sistem için kararlılık, sistem cevabının mümkün olan en kısa zamanda, en az üst aşımı yaparak istenilen değere yerleşmesi ve konumunu koruması olarak tanımlanabilir. Kararlılık için kaynaklardaki bazı tanımlamalar şu şekildedir.

- Bir sistemin sınırlı her girişe cevabı sınırlı ise o sistem kararlıdır.
- Kararlı bir sistem, bir bozucu giriş karşısında geçici durum davranışını gösterdikten sonra tekrar denge konumuna dönen sistemdir.
- Bir sistemin impuls cevabı zaman sonsuza giderken sıfıra yaklaşırsa, o sistem kararlıdır.

Şekil 3.3'te sistemin kararlılık tanımlamalarına uygun çıkış sinyali görülmektedir. Sistemin ulaşmaya çalıştığı değer 2.5'tir ve sistem birkaç salınım yaptıktan sonra istenilen konuma yerleşmektedir. Amaç fonksiyonu (Je), örnekteki gibi, 3-DOF Hover deney setinin sistem cevabında salınımları minimuma indirmek için Denklem (3.1)'deki gibi oluşturulmuştur. Genel hatları ile Denklem (3.1)'deki gibi oluşturulan amaç fonksiyonunun Matlab'de kullanılan tam hali EK-1'de verilmiştir.

$$Je = (\theta_{Rrt} + \theta_{Prt} + \theta_{Yrt} + \theta_{Rst} + \theta_{Pst} + \theta_{Yst} + \theta_{Rpt} + \theta_{Ppt} + \theta_{Ypt} + \theta_{Ros} + \theta_{Pos} + \theta_{Yos} + \theta_{Rp} + \theta_{Pp} + \theta_{Yp} + \theta_{Rn} + \theta_{Pn} + \theta_{Yn}) \quad (3.1)$$

Denklem (3.1)'deki ifadelerin anlamları Çizelge 3.2'de verilmiştir.

Çizelge 3.2. Denklem (3.1)'deki ifadelerin anlamları

Roll			Pitch			Yaw		
Yükselme Zamanı (Rise Time)	Otuma Zamanı (Settling Time)	Zirve Zamanı (Peak Time)	Yükselme Zamanı (Rise Time)	Otuma Zamanı (Settling Time)	Zirve Zamanı (Peak Time)	Yükselme Zamanı (Rise Time)	Otuma Zamanı (Settling Time)	Zirve Zamanı (Peak Time)
θ_{Rrt}	θ_{Rst}	θ_{Rpt}	θ_{Prt}	θ_{Pst}	θ_{Ppt}	θ_{Yrt}	θ_{Yst}	θ_{Ypt}
Üst Aşım (Over Shot)	Zirve (Peak)	Norm	Üst Aşım (Over Shot)	Zirve (Peak)	Norm	Üst Aşım (Over Shot)	Zirve (Peak)	Norm
θ_{Ros}	θ_{Rp}	θ_{Rn}	θ_{Pos}	θ_{Pp}	θ_{Pn}	θ_{Yos}	θ_{Yp}	θ_{Yn}

Çizelge 3.2'de verilen ifadelerin neredeyse tamamı yukarıda açıklanmaktadır. Açıklanmayan ya da Şekil 3.3'te gösterilmeyen tek ifade normdur. Norm, matematiksel bir işlemdir. Sistem cevabını oluşturan çizgiler aslında bir vektördür. Norm, bu vektörün elemanlarının toplamının karekökünü ifade etmektedir. Yapılan testler sonucunda norm işleminin uygulanması ve norm değerinin minimumda tutulması çıkış sinyaline olumlu etki ettiği için amaç fonksiyonuna eklenmiştir.

Özetle oluşturulan amaç fonksiyonu için sistemin cevabına bakılmaktadır. Amaç fonksiyonu sistem cevabını, en kısa yükselme zamanında, maksimum aşma ve kalıcı durum hatası olmadan, istenilen giriş sinyaline yerleştirmeye çalışmaktadır. Bunun için sistem cevabındaki maksimum aşma, yükselme zamanı, oturma zamanı gibi parametreler kullanılarak bir amaç fonksiyonu oluşturulmuştur.

3.1.2. Arama Uzayının Belirlenmesi

Yapay zekâ optimizasyon algoritmaları kullanılarak bir çözüm kümesi bulunmaya çalışılmaktadır. Bulunan çözüm kümesinin elemanları, Denklem (3.2) ve Denklem (3.3)'te görüleceği üzere, LQR kontrolcünün Q ve R matrislerinin diegonal elemanlarından oluşmaktadır. İncelenen sistem için üretici firma tarafından önerilen çözüm kümesi Çizelge 3.3'te Quanser satırında verildiği gibidir. Firmanın sunmuş olduğu çözüm kümesinden yola çıkarak bu çalışma için arama uzayının maksimum ve minimum değerleri Çizelge 3.3'te belirtildiği gibi belirlenmiştir.

$$Q = \begin{bmatrix} q_{11} & 0 & 0 & 0 & 0 & 0 \\ 0 & q_{22} & 0 & 0 & 0 & 0 \\ 0 & 0 & q_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & q_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & q_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & q_{66} \end{bmatrix} \quad (3.2)$$

$$R = \begin{bmatrix} r_{11} & 0 & 0 & 0 \\ 0 & r_{22} & 0 & 0 \\ 0 & 0 & r_{33} & 0 \\ 0 & 0 & 0 & r_{44} \end{bmatrix} \quad (3.3)$$

Eğer herhangi bir optimizasyon algoritmasında ulaşılan yeni çözümler, arama uzayının dışına çıkarsa bu çözümler Denklem (3.4) kullanılarak yakın olan sınır değerlerine çekilir. Burada u_{ij}, algoritmanın hesapladığı yeni çözüm kümesinin elemanını; x_j^{min}, çözüm elemanı için belirlenen alt sınır değerini ve x_j^{max}, çözüm elemanı için belirlenen üst sınır değerini temsil etmektedir.

$$u_{ij} = \begin{cases} x_j^{min} & x_j^{min} > u_{ij} \\ u_{ij} & x_j^{min} \leq u_{ij} \leq x_j^{max} \\ x_j^{max} & x_j^{max} \leq u_{ij} \end{cases} \quad (3.4)$$

Çizelge 3.3. Arama uzayının belirlenmesi

	q_{11}	q_{22}	q_{33}	q_{44}	q_{55}	q_{66}	r_{11}	r_{22}	r_{33}	r_{44}
Quanser [12]	500	350	350	0	20	20	0.01	0.01	0.01	0.01
Üst sınır	700	500	500	20	50	50	2	2	2	2
Alt sınır	1	1	1	0	0.01	0.01	0.01	0.01	0.01	0.01

LQR kontrolcünün parametrelerinin oranlanabilir olmasından dolayı belirlenen alt ve üst sınır yeterlidir. Parametrelerin oranlanabilir olması şu anlama gelmektedir: Örneğin firmanın sunmuş olduğu çözüm kümesinin bütün elemanları sabit bir sayı ile, 0 ve 0'dan küçük olmamak şartıyla, çarpılır ve yeni oluşan çözüm kümesi sisteme uygulanırsa ilk durum ile aynı sistem cevabı elde edilecektir.

3.1.3. Algoritmalar İçin Parametre Optimizasyonu

Optimizasyon algoritmalarının kendine özgü, ulaşılacak en iyi sonucu etkileyen parametreleri bulunmaktadır. Örneğin genetik algoritma için iterasyon sayısı, popülasyon

büyüklüğü, çaprazlama oranı ve mutasyon oranı birer parametredir. Bahsedilen parametrelerden iterasyon sayısı ve popülasyon büyüklüğü deneme sayısı ile ilgilidir. Deneme sayısının arttırılması algoritmanın en iyi sonuca ulaşması için yeterli değildir. Arama kalitesini iyileştirecek diğer iki parametrenin, çaprazlama ve mutasyon oranları, doğru seçilmesi oldukça önemlidir. Ayrıca, deneme sayısı arttırıldığında algoritmanın çalışma süresi de artmaktadır. Dolayısıyla algoritmaların uygulanabilir bir sürede kabul edilebilir bir sonuca ulaşmaları gerekir. Bu nedenle, optimizasyon algoritmaları için parametre optimizasyonu öncelikle uygulanmalıdır.

Bu çalışmada kullanılan yapay zekâ optimizasyon algoritmaları için parametre optimizasyonu yapılırken aşağıdaki kriterlere bağlı kalınmıştır:

1. Algoritmalarda iterasyon sayısı ve popülasyon büyüklüğüne denk gelen bütün parametreler eşit tutulmuştur (İterasyon sayısı = 100, Popülasyon büyüklüğü = 50).
2. Algoritmaya özgü diğer parametreler belirlenmiş, bunlar için olası değerler seçilmiş, bu değerlerin kombinasyonlarıyla çeşitli varyasyonlar oluşturulmuştur.
3. Algoritmalar, oluşturulan her varyasyon için üçer kez çalıştırılmış ve elde edilen uygunluk değerleri kaydedilmiştir.
4. Kaydedilen uygunluk değerlerinin her varyasyon için aritmetik ortalaması alınmış ve en iyi sonucu veren parametreler belirlenmiştir.

3.2. Yapay Zekâ Optimizasyon Algoritmalarının Sisteme Uygulanması

Bu çalışmada, önerilen hiyerarşik SAA-GWO algoritması ile birlikte yedi farklı yapay zekâ optimizasyon algoritması kullanılmıştır. Tüm yapay zekâ optimizasyon algoritmaları için amaç fonksiyonu ve arama uzayı önceki bölümlerde belirlendiği şekilde kullanılmıştır. Bu bölümde verilen yapay zekâ optimizasyon algoritmaları elde ettikleri sonuca göre sıralanmıştır. İlk sırada, ortalama uygunluk değeri açısından, en kötü sonucu üreten algoritma yer almaktadır.

3.2.1. Yusufçuk Algoritması

Yusufçuk algoritması (Dragonfly Algorithm - DA) Seyedali Mirjalili tarafından 2015 yılında önerilmiştir. Algoritma sürü zekasına dayanmaktadır. Yusufçukların doğal

yaşamdaki yiyecek arama ve düşmandan kaçma davranışları modellenerek bu algoritma geliştirilmiştir [17], [18].

3.2.1.1. Algoritmanın İşleyişi

Yusufçuk algoritması rasgele oluşturulan başlangıç popülasyonu ve ilk yer değiştirme istekleri oluşturularak başlamaktadır. Başlangıçta yusufçukların yer değiştirme istekleri sıfır olarak kabul edilmektedir. Başlangıç popülasyonunu oluşturan her bir yusufçuk için uygunluk değeri hesaplanır. En iyi uygunluk değerini veren çözüm yemek, en kötü uygunluk değerini veren çözüm ise düşman olarak adlandırılır. Algoritma, temel olarak yusufçuk adı verilen çözümleri yemeğe yaklaştırmaya ve düşmandan uzaklaştırmaya çalışır.

Algoritmada komşuluk adı altında bir ilişki bulunmaktadır. Komşuluk, komşuluk yarı çapı adı verilen bir uzaklık ile sağlanır. Algoritmada komşuluğun olup olmamasına göre iki çeşit konum güncelleme işlemi gerçekleştirilir. Komşuluk yarıçapı başlangıçta küçük seçilir ve iterasyonlar arttıkça komşuluk yarıçapı da artırılır. Komşuluk yarıçapı iterasyonlara göre Denklem (3.5)'teki gibi güncellenmektedir.

$$Komşu_r = iter \times \frac{(X_{\text{üst}}(i) - X_{alt}(i))}{iter_{max}} \tag{3.5}$$

Denklem (3.5)'te $Komşu_r$, komşuluk yarıçapını; $iter$, o anki iterasyonu; $X_{\text{üst}}$, yusufçukların bulunabileceği maksimum konum değerlerini; X_{alt}, yusufçukların bulunabileceği minimum konum değerini; i, 1'den problem boyutuna (parametre sayısı) kadar olan sayıyı ve $iter_{max}$, maksimum iterasyon sayısını ifade etmektedir. Komşuluk olduğu durumlarda algoritma şu üç durumu sağlamaya çalışır.

- Komşu yusufçuklar birbirine yakın olabilirler. Ancak birbirlerine çarpmamaları gerekir (ayrışma).
- Komşu yusufçukların hızları birbirine yakın olmalıdır (sıralanma).
- Komşu yusufçukların pozisyonları birbirine yakın olmalıdır (uyum).

Ayrışma işlemi Denklem (3.6)'da formüle edildiği gibi gerçekleşir.

$$S_i = -\sum_{j=1}^{N} X - X_j \tag{3.6}$$

Denklem (3.6)'da S_i, sürüde komşuluğu bulunan i. yusufçuk için gerçekleştirilen ayrışma işlemini; N, komşuluk sayısını; X, ayrışma işleminin uygulandığı yusufçuğun konumunu ve X_j, ayrışma işlemi uygulanan yusufçuğun j. komşusunu ifade etmektedir.

Sıralanma işlemi Denklem (3.7)'de formüle edildiği şekilde gerçekleşir.

$$A_i = \frac{\sum_{j=1}^{N} V_j}{N} \tag{3.7}$$

Denklem (3.7)'de bir önceki denkleme ek olarak A_i, sürüde komşuluğu bulunan i. yusufçuk için gerçekleştirilen sıralanma işlemini ve V_j, j. komşunun hızını temsil etmektedir. Denklem (3.7)'de yapılan sıralanma işlemi komşu yusufçukların hızlarının ortalamasını bulur.

Uyum işlemi Denklem (3.8)'de formüle edildiği şekilde gerçekleşir.

$$C_i = \frac{\sum_{j=1}^{N} X_j}{N} \tag{3.8}$$

Denklem (3.8)'de önceki denklemlere ek olarak C_i, sürüde komşuluğu bulunan i. yusufçuk için gerçekleştirilen uyum işlemini ve X_j, j. komşunun konumunu temsil etmektedir. Denklemde görüldüğü üzere komşu yusufçukların konumlarının ortalaması alınmaktadır. Bu denklemlerden yola çıkarak komşuluğu bulunan yusufçuklar için hız değeri Denklem (3.9)'daki gibi hesaplanır.

$$V_i = s \times S_i + a \times A_i + c \times C_i \tag{3.9}$$

Burada s, ayrışma katsayısını; a, sıralanma katsayısını ve c, uyum sayısını ifade etmektedir. Algoritmada yusufçukların yemeğe yaklaşmalarını ve düşmandan kaçmalarını sağlamak için yemek yönünün ve düşman yönünün belirlenmesi gerekmektedir. Yemek yönü ve düşman yönü Denklem (3.10) ve Denklem (3.11)'de gösterildiği gibi hesaplanmıştır.

$$F_i = X^+ - X \tag{3.10}$$

$$E_i = X^- - X \tag{3.11}$$

Burada F_i, yemek yönünü; E_i, düşman yönünü; X^+, yemeğin konumunu; X^-, düşmanın konumunu ve X konumu güncellenecek yusufçuğu temsil etmektedir. Sonuç olarak komşuluk ilişkisi olan bir yusufçuğun yeni yer değiştirme isteği ve yeni konumu Denklem (3.12)'de ve Denklem (3.13)'te gösterildiği gibi hesaplanır.

$$\Delta \mathrm{X}_i = \mathrm{V}_\mathrm{i} + f \times F_i + e \times E_i + w \times \Delta X_{i-1} \tag{3.12}$$

$$X_i = X_{i-1} + \Delta X_i \tag{3.13}$$

Burada $\Delta \mathrm{X}_i$, i. komşuluk ilişkisi olan yusufçuğun yer değiştirme isteğini; $\Delta \mathrm{X}_{i-1}$, i. komşuluk ilişkisi olan yusufçuğun bir önceki yer değiştirme isteğini; X_i, i. komşuluk ilişkisi olan yusufçuğun hesaplanan yeni konumunu; X_{i-1}, i. komşuluk ilişkisi olan yusufçuğun eski konumunu; w, eylemsizlik katsayısını; f, yiyecek yönü katsayısını ve e, düşman yönü katsayısını ifade etmektedir. Denklem (3.9) ve Denklem (3.12)'de toplamda altı adet katsayı kullanılmıştır. Bu katsayılar, iterasyonlara göre değişiklik gösterecek şekilde Denklem (3.14) - (3.20) arasındaki denklemler kullanılarak hesaplanmaktadır.

$$\mathrm{w} = 0.9 - \mathrm{iter} \times \left(\frac{(0.9 - 0.4)}{\mathrm{iter}_{\max}} \right) \tag{3.14}$$

$$e = 0.1 - iter \times (\frac{0.1 - 0}{\frac{\mathrm{iter}_{\max}}{2}}) \tag{3.15}$$

$$e < 0 \; ise, e = 0 \tag{3.16}$$

$$s = 2 \times rand \times e \tag{3.17}$$

$$a = 2 \times rand \times e \tag{3.18}$$

$$c = 2 \times rand \times e \tag{3.19}$$

$$f = 2 \times rand \tag{3.20}$$

Yusufçuklar arasında komşuluk ilişkisi yoksa yeni yusufçuğun konumu, Lèvy uçuş mekanizması kullanılarak rasgele oluşturulur. Literatürde başka rasgele hareket yöntemleri kullanılarak yusufçuk algoritması geliştirilmeye çalışılmıştır. Örneğin Brownian Hareketi bunlardan biridir [19].

Lèvy uçuş mekanizmasını Paul Lèvy adında bir matematikçi geliştirmiştir ve Denklem (3.21) ve Denklem (3.22)'de gösterildiği gibi hesaplanmaktadır [19].

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi \times \beta}{2}\right)}{\Gamma\left(\frac{(1+\beta)}{2}\right) \times \beta \times 2^{\frac{\beta-1}{2}}} \right)^{\frac{1}{\beta}} \tag{3.21}$$

$$Lèvy(x) = 0.01 \times \frac{rand_1 \times \sigma}{rand_2} \quad (3.22)$$

Burada β, bir sabit sayıdır ve temel yusufçuk algoritmasında β'nın değeri 1.5 olarak alınmıştır [17]. Ayrıca Γ, gamma fonksiyonunu temsil etmektedir ve Denklem (3.23)'deki gibi hesaplanmaktadır.

$$\Gamma(Sayi) = (Sayi - 1)! \quad (3.23)$$

Şekil 3.4'te yusufçuk algoritmasının çalışması basit kod şeklinde verilmiştir.

```
Rasgele başlangıç yusufçuk sürüsünü oluştur
Yusufçukların başlangıç yer değiştirme isteklerini oluştur (ΔX)
while (durdurma koşulu sağlanıncaya kadar)
    Yusufçukların uygunluk değerlerini hesapla
    Yemek ve düşmanı güncelle
    w, s, a, c, f ve e katsayılarını güncelle
    S, A, C, F ve E değerlerini hesapla
    Komşuluk yarıçapını güncelle
    if (Komşu sayısı > 1)
        Hızı S, A ve C değerlerini kullanarak güncelle
        Pozisyonu S, A, C, F ve E değerlerini kullanarak güncelle
    else
        Lèvy uçuş mekanizmasını kullanarak pozisyonları güncelle
    end if
Parametre sınırlarını kontrol et
end while
```

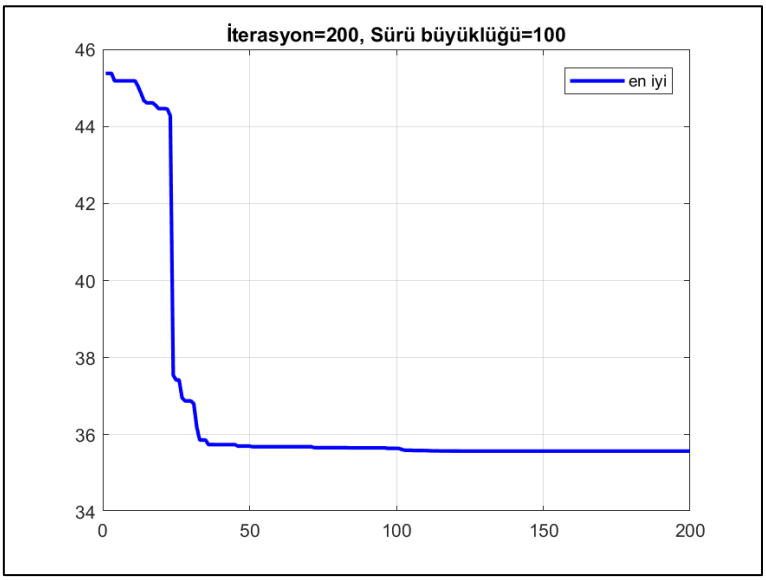
Şekil 3.4. Yusufçuk algoritmasının basit kodu [18]

3.2.1.2. Sonuçlar

İterasyon sayısı 200 ve popülasyon büyüklüğü 100 alınarak yusufçuk algoritması 10 defa çalıştırılmıştır. Bu çalıştırmalar sonucunda elde edilen uygunluk değerlerine ait özet bilgiler Çizelge 3.4'te verilmiştir. Çizelgeden de görülebildiği gibi bu algoritma tarafından elde edilen en iyi uygunluk değeri 35.5737'dir. Bu uygunluk değerinin elde edildiği çalışmaya ait yakınsama eğrisi Şekil 3.5'te verilmiştir.

Çizelge 3.4. İterasyon sayısı = 200, popülasyon büyüklüğü = 100 değerleri için 10 kez çalıştırılan yusufçuk algoritmasının sonuçları

En iyi uygunluk değeri	En kötü uygunluk değeri	Ortalama uygunluk değeri	Aralık	Standart Sapma	Ortalama Geçen Süre (sn)
35.5737	42.8027	39.6322	7.2289	2.3939	2794.2376



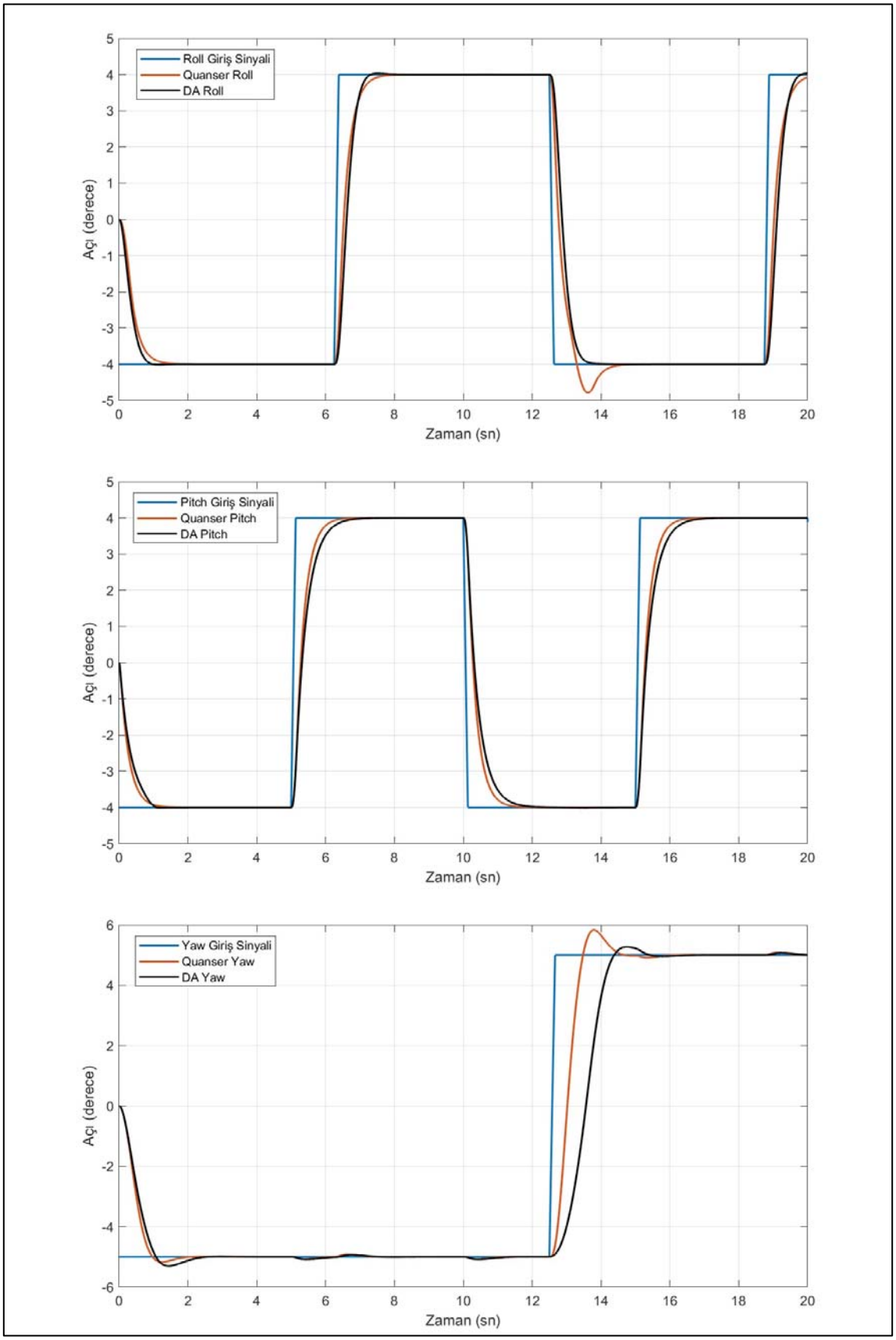
Şekil 3.5. Yusufçuk algoritmasının yakınsama eğrisi

Bu tez çalışması kapsamında ulaşılan en iyi uygunluk değeri 27.5239'dur. Yusufçuk algoritması elde ettiği 35.5737 uygunluk değeri ile bu sonucun oldukça uzağındadır. Şekil 3.5'teki yakınsama eğrisinden yusufçuk algoritmasının yaklaşık 50. iterasyondan sonra lokal minimum noktaya takıldığı görülebilmektedir.

Çizelge 3.5'te yusufçuk algoritmasının ulaşmış olduğu en iyi uygunluk değeri olan 35.5737 değerine ait çözüm kümesi verilmiştir. Çizelge 3.5'teki veriler Q ve R matrislerini oluşturmaktadır. Oluşturulan matrisler Ricatti denkleminde kullanılarak denklemin kökü olan K matrisi hesaplanmıştır. K matrisi 3-DOF Hover sistemine uygulandığında sistemin verdiği cevap Şekil 3.6'da görülmektedir.

Çizelge 3.5. Yusufçuk algoritması ile ulaşılan en iyi çözüm

q_{11}	q_{22}	q_{33}	q_{44}	q_{55}	q_{66}	r_{11}	r_{22}	r_{33}	r_{44}
332.5799	500	500	0.1961	50	14.9544	0.01	0.01	0.1992	0.3860



Şekil 3.6. Giriş sinyalleri, Quanser parametreleri ile hesaplanan çıkış sinyalleri ve yusufçuk algoritması ile hesaplanan en iyi çıkış sinyalleri

Şekil 3.6’da mavi, kırmızı ve siyah olmak üzere üç farklı çizgi ile belirtilen sinyaller bulunmaktadır. Mavi çizgi ile gösterilen sinyaller giriş sinyaline aittir. Kırmızı çizgi ile gösterilen sinyaller, Quanser firmasının önerdiği kontrolcü parametrelerinin kullanılmasıyla elde edilen çıkış sinyallerine aittir. Son olarak siyah çizgi ile gösterilen sinyaller, yusufçuk algoritması kullanılarak hesaplanan kontrolcü parametrelerinin, sistemde uygulanmasıyla elde edilen sistem cevaplarına ait çıkış sinyalleridir. Çizelge 3.6’da, Şekil 3.6’da verilen çıkış sinyallerinin yorumlanabilmesi ve karşılaştırılabilmesi için bazı bilgiler yer almaktadır.

Çizelge 3.6. Şekil 3.6'daki Quanser çıkış sinyalleri ile yusufçuk algoritması ile hesaplanan çıkış sinyallerinin karşılaştırılması

	Oturma Zamanı (sn)			Yükselme Zamanı (sn)			Üst Aşım (deg)		
	Roll	Pitch	Yaw	Roll	Pitch	Yaw	Roll	Pitch	Yaw
Qua	19.7107	16.0741	14.3349	0.5042	0.5440	0.3475	0.7890	0.0003	0.8323
DA	19.6582	16.3588	15.0229	0.4040	0.7144	0.5771	0.0313	0.0097	0.3072
Fark	0.0525	-0.2847	-0.688	0.1002	-0.1704	-0.2296	0.7577	-0.0094	0.5251

Çizelge 3.6’da yusufçuk algoritması kullanılarak ulaşılan parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları ile Quanser firmasının parametrelerinin sistemde uygulanmasıyla elde edilen sistem cevapları karşılaştırılmıştır. İlk satırda Quanser firmasının önerdiği parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları hakkında bilgiler yer almaktadır. İkinci satırda yusufçuk algoritmasının kullanılmasıyla elde edilen parametrelerin sistemde uygulanmasıyla ulaşılan sistem cevabı hakkında bilgiler yer almaktadır. Son satırda ise bahsedilen bu bilgilerin farkları yer almaktadır. Hatırlanacağı üzere amaç fonksiyonun görevi oturma zamanını, yükselme zamanını ve üst aşım değerini azaltmaktı. Dolayısıyla yusufçuk algoritmasının başarısız olduğu sütunlar fark satırında negatif değere sahip olan sütunlardır.

Çizelge 3.6’da verilerin çoğunluğunun kötüleştiği görülmektedir. Pitch açısının üst aşımındaki kötüleşme, Amaç Fonksiyonunun Belirlenmesi bölümünde anlatıldığı gibi beklenen bir durumdur. Pitch açısında yaşanan 0.0094 derecelik kötüleşme sayesinde roll açısının üst aşımı 0.7577 derece azaltılabilmiştir. Yaw açısının oturma zamanında yaşanan kötüleşme, Amaç Fonksiyonunun Belirlenmesi bölümünde anlatıldığı gibi beklenen bir durumdur. Yaw açısının oturma zamanında yaşanan kötüleşme her ne kadar

beklenen bir durum olsa da karşılığında roll ve pitch açılarında belli bir oranda iyileşme olması gerekirdi. Ancak, çizelgeden de görülebildiği gibi, sadece roll açısının oturma zamanında 0.0525 saniyelik bir iyileşme meydana gelmiştir.

3.2.2. Genetik Algoritma

Genetik algoritmanın (Genetic Algorithms - GA) temelleri 1975 yılında John Holland tarafından atılmıştır. Evrimsel süreçte meydana gelen doğal seçilim, çaprazlama ve mutasyon olayları modellenerek genetik algoritma oluşturulmuştur. Algoritma popülasyon tabanlı bir algoritmadır. Popülasyonu kromozom adı verilen çözümler oluşturmaktadır. Popülasyon temelli bir algoritma olmasından dolayı çözüm, birçok farklı noktadan eş zamanlı aranmaya başlanır ve böylece daha kapsamlı bir arama gerçekleştirilmiş olur [20].

3.2.2.1. Algoritmanın İşleyişi

İkili ve gerçek kodlu olmak üzere iki tür genetik algoritma bulunmaktadır. Bu tez çalışmasında gerçek kodlu genetik algoritma kullanılmıştır. Genetik algoritma üç kısımdan oluşur: doğal seçilim, çaprazlama ve mutasyon. Doğal seçilim, en iyi sonucu veren kromozomların daha çok üremesine olanak verir. Çaprazlama, genetik çeşitlilik sağlar ve lokal arama sonucunu geliştirir. Mutasyon, daha önce hiç rastlanmamış çözümler oluşmasını sağlar ve bu sayede global aramanın gerçekleşmesine imkân tanır.

Doğal seçilim işleminde çaprazlama işlemine girecek bireyler belirlenir. Uygunluk değeri yüksek olan kromozomların daha fazla oranda çaprazlama işlemine girebilmesi için bu çalışmada rulet tekerliği yöntemi kullanılır. Rulet tekerleği yönteminde bireylerin ne kadarının çaprazlamaya girebileceği, çaprazlama oranı sayesinde belirlenir. Örneğin 10 adet kromozomu olan bir popülasyonda çaprazlama oranı 0.70 olarak verilmişse burada en iyi sonucu veren 7 tane birey çaprazlama işlemine girecek demektir. Ayrıca çaprazlamaya giren birey rulet tekerleği yönteminde birden fazla sayıda seçilebilir ve tekrar çaprazlamaya girebilir.

Çaprazlama işlemi için kaynaklarda birçok yöntem önerilmektedir. Bu çalışmada, gen sayısı 10 olduğu için tek noktalı ve çift noktalı çaprazlama yöntemleri kromozomlar arasındaki bilgi alışverişi için yetersiz kalmaktadır. Bundan dolayı çok noktalı

çaprazlama yöntemi kullanılmıştır. Kullanılan yöntem Denklem (3.24) ve (3.25)'te formüle edildiği gibi gerçekleşir.

$$kromozom1_{yeni} = kromozom_1 \times \alpha + (1 - \alpha) \times kromozom_2 \quad (3.24)$$

$$kromozom2_{yeni} = kromozom_2 \times \alpha + (1 - \alpha) \times kromozom_1 \quad (3.25)$$

Burada α ile gösterilen değer [-0.05, 0.05] arasında üretilecek rastgele bir sayıyı ifade etmektedir.

Mutasyon işlemi ise klasik yöntem ile gerçekleştirilmiştir. Öncelikle, popülasyondaki toplam gen sayısı mutasyon oranı ile çarpılarak mutasyona uğrayacak gen sayısı tespit edilir. Daha sonra, tespit edilen sayı kadar rastgele genler seçilir. Bu genler çözüm kümesinin alt ve üst sınırlarının izin verdiği ölçüde rastgele sayılar ile değiştirilir.

Çizelge 3.7'de örnek bir popülasyon ve bunlara ait uygunluk değerleri verilmektedir. Ebeveyn bireyler olarak ilk iki bireyin seçildiği ve çaprazlama için üretilen rastgele sayının da α = 0.05 olduğu kabul edilsin. Denklem (3.24) ve Denklem (3.25) kullanılarak ulaşılan yeni bireyler Çizelge 3.8'deki gibi olacaktır.

Çizelge 3.7. Genetik algoritma için örnek bir popülasyon (UD = Uygunluk Değeri)

UD	q_{11}	q_{22}	q_{33}	q_{44}	q_{55}	q_{66}	r_{11}	r_{22}	r_{33}	r_{44}
36.24	370.75	275.38	294.36	6.27	36.84	4.85	0.01	0.01	0.54	1.12
38.16	416.42	477.93	356.02	18.91	3.54	1.12	0.05	0.31	0.13	1.06
38.21	247.90	318.73	357.69	3.97	10.54	8.96	0.01	0.22	0.12	0.87
38.57	459.56	449.65	425.27	15.18	0.41	8.43	0.06	0.68	0.16	0.79
39.05	596.20	398.42	470.69	1.73	7.35	32.11	0.02	0.34	0.45	1.35
39.13	474.13	410.82	345.48	15.39	12.96	6.25	0.01	0.59	0.17	0.92
39.35	578.52	403.62	107.34	17.02	13.48	17.20	0.01	0.61	0.24	0.85
39.76	391.44	400.41	91.29	2.58	14.13	16.00	0.01	0.89	0.45	1.52
40.12	162.82	311.34	257.57	16.35	8.71	18.20	1.98	0.02	0.75	1.96
40.35	68.77	41.54	323.58	13.47	0.51	21.71	0.70	0.01	0.14	1.06

Bu şekilde diğer kromozomlar için de çaprazlama işlemi gerçekleştirilir. Son olarak mutasyon işleminin bir örneğini incelemek için mutasyona uğrayacak genlerden

bir tanesinin Y1 kromozomunun ilk genine, Çizelge 3.8'de koyu ile gösterilen gen, isabet ettiği kabul edilsin. Bu durumda mutasyon işlemi şu şekilde gerçekleşir:

Çizelge 3.8. Genetik algoritma için örnek çaprazlama işlemi (K1 = Ebeveyn Kromozom1, K2 = Ebeveyn Kromozom2, Y1 = Çocuk Kromozom1, Y2 = Çocuk Kromozom2)

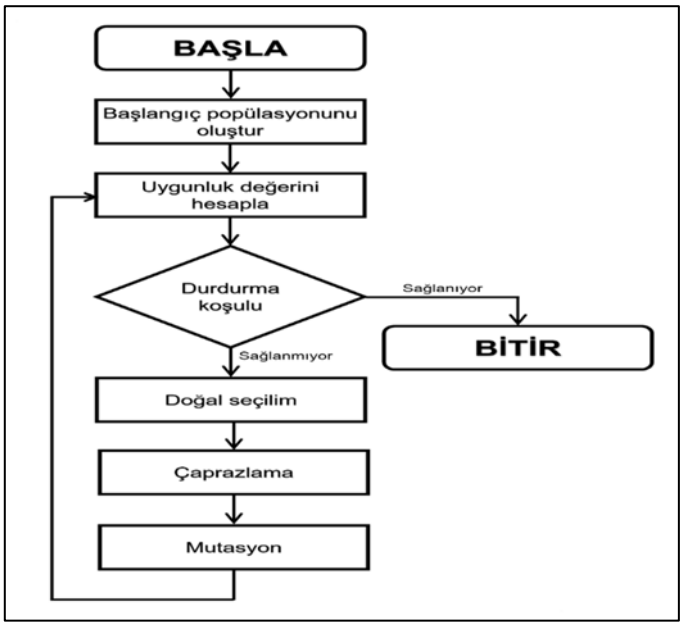
	q_{11}	q_{22}	q_{33}	q_{44}	q_{55}	q_{66}	r_{11}	r_{22}	r_{33}	r_{44}
$\boldsymbol{K1}$	370.75	275.38	294.36	6.27	36.84	4.85	0.01	0.01	0.54	1.12
$\boldsymbol{K2}$	416.42	477.93	356.02	18.91	3.54	1.12	0.05	0.31	0.13	1.06
$\boldsymbol{K1 \times \alpha}$	14.83	11.02	11.77	0.25	1.47	0.19	0.00	0.00	0.02	0.04
$\boldsymbol{K1 \times (1-\alpha)}$	355.92	264.37	282.59	6.02	35.37	4.66	0.01	0.01	0.51	1.08
$\boldsymbol{K2 \times \alpha}$	16.66	19.12	14.24	0.76	0.14	0.04	0.00	0.01	0.01	0.04
$\boldsymbol{K2 \times (1-\alpha)}$	399.77	458.82	341.78	18.15	3.40	1.08	0.05	0.30	0.13	1.02
$\boldsymbol{Y1}$	**414.60**	469.83	353.56	18.40	4.87	1.27	0.05	0.30	0.15	1.06
$\boldsymbol{Y2}$	372.58	283.48	296.83	6.77	35.51	4.70	0.01	0.02	0.52	1.12

Arama Uzayının Belirlenmesi Bölümünden hatırlanacağı üzere q_{11} Denklem (3.26)'da belirtilen değerler arasında olabilir. Mutasyon işlemiyle elde edilecek yeni gen ise Denklem (3.27)'de formüle edildiği gibi elde edilir.

$$1 \leq q_{11} \leq 700 \tag{3.26}$$

$$Gen_{mutant} = q_{11}(min) + rand \times (q_{11}(max) - q_{11}(min)) \tag{3.27}$$

Örneklerden de görüldüğü gibi doğal seçilim, çaprazlama ve mutasyon işlemi durdurma koşulu sağlanıncaya kadar devam eder. Durdurma koşulu belirlenen iterasyon sayısıdır. Gerçekleşen işlemler esnasında çözüm kümesi için belirlenen sınır değerleri kontrol edilir. Şekil 3.7'de genetik algoritmanın işleyişi özetleyen bir akış diyagramı görülmektedir.



Şekil 3.7. Genetik algoritma akış diyagramı [20]

3.2.2.2. Algoritma İçin Parametre Optimizasyonu

Genetik algoritmanın arama kalitesini iyileştiren iki tane parametresi vardır. Bunlar çaprazlama ve mutasyon oranlarıdır. Teoride her iki parametreyi de 0 ile 1 arasında seçmek mümkündür. Ancak yapılan araştırmalarda çaprazlama oranının 0.75'ten büyük mutasyon oranının ise 0.10'dan küçük seçildiği görülmektedir [21]. Çaprazlama ve mutasyon oranlarının tespit edilmesi için Algoritmalar İçin Parametre Optimizasyonu Bölümünde anlatılanlara ek olarak,

- Çaprazlama oranı 0.75, 0.85 ve 0.95 seçilmiştir
- Mutasyon oranı 0.03, 0.05 ve 0.07 seçilmiştir

Bu parametrelerin ikili kombinasyonları oluşturularak parametre grupları oluşturulmuştur. Oluşturulan parametre gruplarıyla gerçekleştirilen parametre optimizasyonu sonucunda her parametre grubu için elde edilen en iyi değerlerin ortalaması alınarak Çizelge 3.9'da sunulmuştur. Örnek olarak çaprazlama oranı (CO) 0.75 ve mutasyon oranı (MO) 0.03 seçilip üçer kez çalıştırılmış ve ulaşılan üç uygunluk değerinin aritmetik ortalaması 43.1684894 olarak bulunmuştur.

Çizelge 3.9. Genetik algoritma için parametre optimizasyonu (İterasyon Sayısı = 100, Popülasyon büyüklüğü = 50)

CO / MO	0.75	0.85	0.95
0.03	43.1684894	42.46313816	41.59500529
0.05	43.14596861	40.76813323	41.06998698
0.07	40.59167826	**40.47416078**	41.38556809

Yapılan parametre optimizasyonu deneylerinden genetik algoritma için Çizelge 3.9'da görüleceği üzere çaprazlama oranının 0.85 ve mutasyon oranının 0.07 seçilmesine karar verilmiştir.

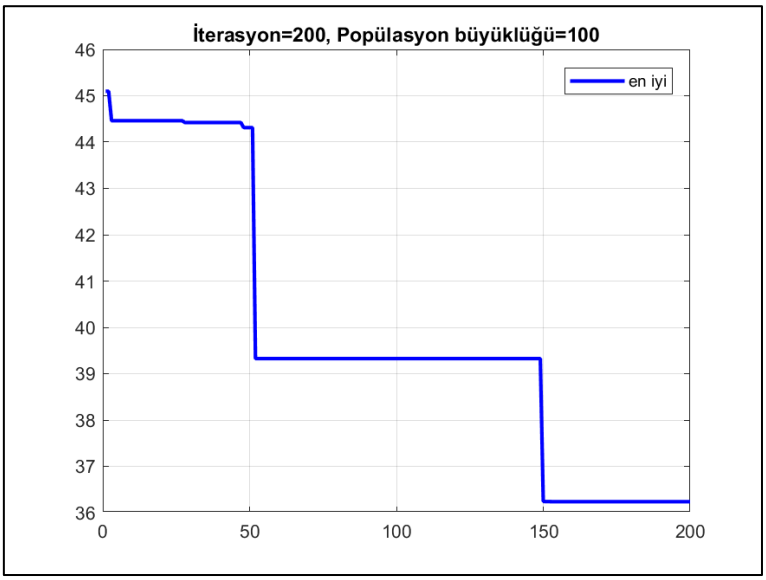
3.2.2.3. Sonuçlar

Parametre optimizasyonu sonucunda belirlenen çaprazlama ve mutasyon oranına ek olarak iterasyon sayısı 200 ve popülasyon büyüklüğü 100 olarak seçilip genetik algoritma 10 defa çalıştırılmıştır. Belirtilen şartlarda gerçekleşen on çalışmanın sonucunda elde edilen veriler özetlenerek Çizelge 3.10'da sunulmuştur.

Çizelge 3.10. İterasyon Sayısı = 200, Popülasyon Büyüklüğü = 100, CO = 0.85 ve MO = 0.07 değerleri için 10 kez çalıştırılan genetik algoritmanın sonuçları

En iyi uygunluk değeri	En kötü uygunluk değeri	Ortalama uygunluk değeri	Aralık	Standart Sapma	Ortalama Geçen süre
36.2370	40.3512	38.8935	4.1141	1.1331	2922.9061

Genetik algoritmanın anlatılan şartlarda 10 defa çalıştırılması sonucunda on farklı uygunluk değerine ulaşılmıştır. Ulaşılan uygunluk değerlerinin en iyisi Çizelge 3.10'da 36.2370 olarak görülmektedir. Bu uygunluk değerinin elde edildiği çalışmaya ait yakınsama eğrisi Şekil 3.8Şekil 3.5'de verilmiştir.



Şekil 3.8. Genetik algoritmanın yakınsama eğrisi

Şekil 3.8'den de görüleceği üzere ulaşılan sonucun daha iyiye taşınması için uzun iterasyonlar gerekmektedir. Genetik algoritmanın nerdeyse 50 iterasyon gelişim göstermeden çalıştığı görülmektedir. Bu durum algoritmanın lokal minimumlarda takılı kaldığını göstermektedir. Lokal minimumlara takılma işlemi daha sonraki iterasyonlarda da devam etmiştir. Genetik algoritma ile ulaşılan en iyi uygunluk değeri, bu tez çalışması kapsamında ulaşılan en iyi uygunluk değerinin (27.5239) oldukça uzağındadır.

Çizelge 3.11'de genetik algoritmanın ulaşmış olduğu en iyi uygunluk değeri olan 36.2370 değerine ait çözüm kümesi verilmiştir. Çizelgedeki veriler Q ve R matrislerini oluşturmaktadır. Oluşturulan matrisler Ricatti Denkleminde kullanılarak denklemin kökü olan K matrisi hesaplanmıştır. K matrisi 3-DOF Hover sistemine uygulandığında sistemin verdiği cevap Şekil 3.9'da görülmektedir.

Çizelge 3.11. Genetik algoritma ile ulaşılan en iyi çözüm

q_{11}	q_{22}	q_{33}	q_{44}	q_{55}	q_{66}	r_{11}	r_{22}	r_{33}	r_{44}
370.7508	275.3803	294.3642	6.2674	36.8406	4.8529	0.01	0.01	0.5361	1.1211

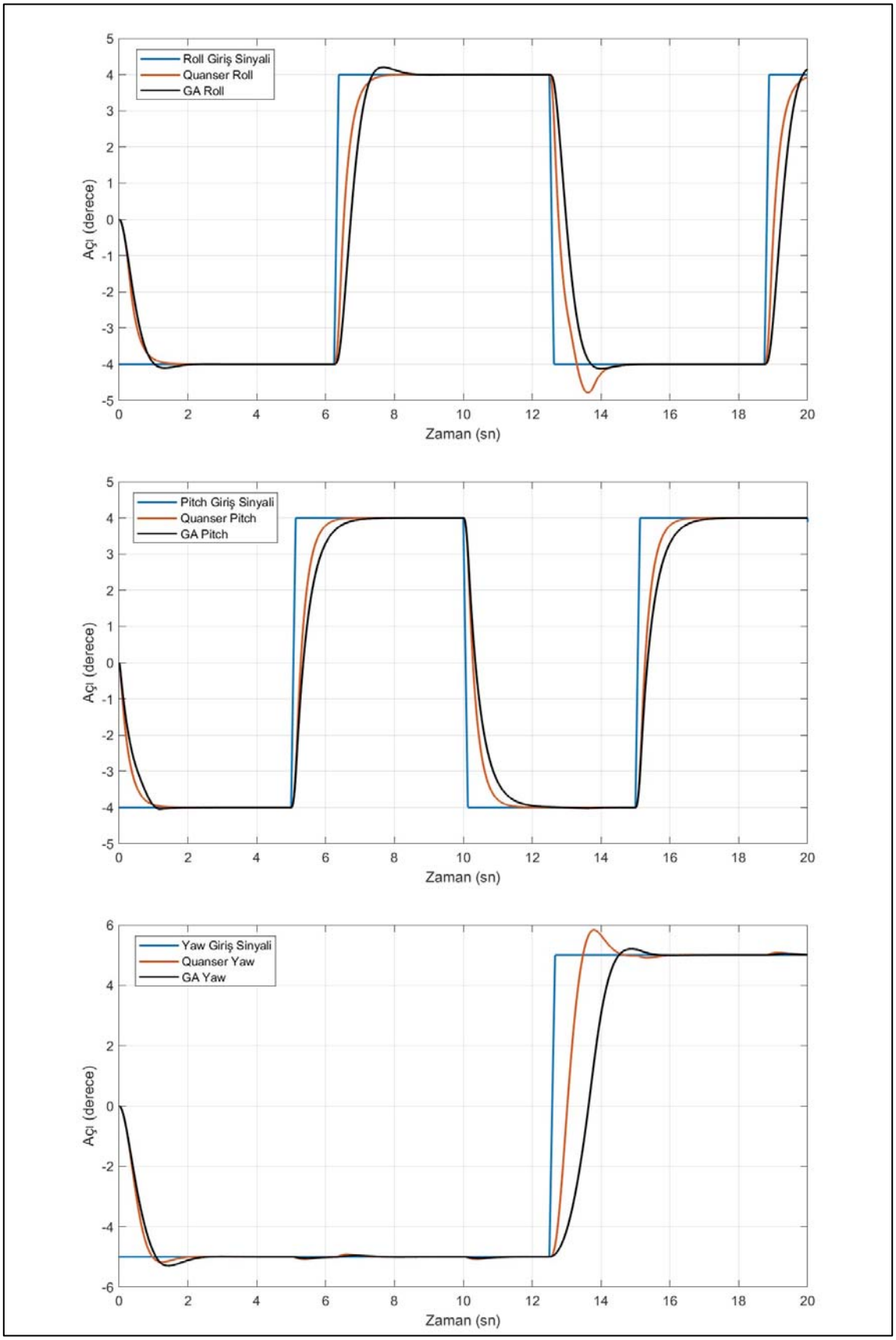
Şekil 3.9'da mavi, kırmızı ve siyah olmak üzere üç farklı çizgi ile belirtilen sinyaller bulunmaktadır. Mavi çizgi ile gösterilen sinyaller giriş sinyaline aittir. Kırmızı çizgi ile gösterilen sinyaller, Quanser firmasının önerdiği kontrolcü parametrelerinin kullanılmasıyla elde edilen çıkış sinyallerine aittir. Son olarak siyah çizgi ile gösterilen sinyaller, genetik algoritma kullanılarak hesaplanan kontrolcü parametrelerinin, sistemde

uygulanmasıyla elde edilen sistem cevaplarına ait çıkış sinyalleridir. Çizelge 3.12'de Şekil 3.9'da verilen çıkış sinyallerinin yorumlanabilmesi ve karşılaştırılabilmesi için bazı bilgiler yer almaktadır.

Çizelge 3.12. Şekil 3.9'daki Quanser çıkış sinyalleri ile genetik algoritma ile hesaplanan çıkış sinyallerinin karşılaştırılması

	Oturma Zamanı (sn)			Yükselme Zamanı (sn)			Üst Aşım (deg)		
	Roll	**Pitch**	**Yaw**	**Roll**	**Pitch**	**Yaw**	**Roll**	**Pitch**	**Yaw**
Qua	19.7107	16.0741	14.3349	0.5042	0.5440	0.3475	0.7890	0.0003	0.8323
GA	19.8586	16.5520	14.4112	0.4837	0.8230	0.6087	0.2034	0.0437	0.3002
Fark	-0.1479	-0.4779	-0.0763	0.0205	-0.279	-0.2612	0.5856	-0.0434	0.5321

Çizelge 3.12'de verilen genetik algoritma kullanılarak ulaşılan parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları ile Quanser firmasının parametrelerinin sistemde uygulanmasıyla elde edilen sistem cevapları karşılaştırılmıştır. İlk satırda Quanser firmasının önerdiği parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları hakkında bilgiler yer almaktadır. İkinci satırda genetik algoritmanın kullanılmasıyla elde edilen parametrelerin sistemde uygulanmasıyla ulaşılan sistem cevabı hakkında bilgiler yer almaktadır. Son satırda ise bahsedilen bu bilgilerin farkları yer almaktadır. Genetik algoritmanın başarısız olduğu sütunlar fark satırında negatif değere sahip olan sütunlardır.



Şekil 3.9. Giriş sinyalleri, Quanser parametreleri ile hesaplanan çıkış sinyalleri ve genetik algoritma ile hesaplanan en iyi çıkış sinyalleri

Çizelge 3.12'de verilerin çoğunluğunun kötüleştiği görülmektedir. Ancak sistemin roll ve yaw cevabındaki üst aşımlar azaltılabilmiştir. Oturma zamanlarında da kötüleşmeler gerçekleşmiştir. Oturma zamanında gerçekleşen en kötü sonuç 0.4779 saniye fark ile pitch cevabında yaşanmıştır. Oturma zamanında yaşanan kötüleşmeleri 0.1479 saniye fark ile roll cevabı ve 0.0763 saniye fark ile yaw cevabı takip etmektedir. Yükselme zamanlarına bakıldığında roll cevabında iyileşme görülürken pitch ve yaw cevaplarında kötüleşmeler vardır.

3.2.3. Parçacık Sürü Optimizasyon Algoritması

Parçacık sürü optimizasyon algoritması (Particle Swarm Optimization - PSO) 1995 yılında Dr. Kennedy ve Dr. Eberhart tarafından önerilmiştir. Algoritma sürü zekasına dayanan bir arama yöntemi kullanır ve popülasyon temellidir. Parçacık sürü optimizasyon algoritması, kuş ve balık gibi sürü halinde yaşayan hayvanların sosyal davranışları incelenerek geliştirilmiştir [22].

3.2.3.1. Algoritmanın İşleyişi

Algoritma rastgele oluşturulan başlangıç popülasyonu (sürü) ile başlamaktadır. Popülasyonu oluşturan her bir çözüme parçacık adı verilmektedir. Parçacıkların uygunluk değerleri her iterasyonda hesaplanarak o anki iterasyona kadar ulaşılan en iyi çözüm (g_{best}) ve ilgili parçacığın ulaştığı en iyi çözümü (p_{best}) belirlenir.

Denklem (3.28)'e göre her parçacık için bir hız değeri hesaplanır. Hesaplanan hız Denklem (3.29)'da görüldüğü gibi mevcut parçacıklara eklenerek bir sonraki iterayonun sürüsü elde edilir.

$$v_{i+1} = w \times v_i + c_1 \times r_1 \times (p_{best} - x_i) + c_2 \times r_2 \times (g_{best} - x_i) \quad (3.28)$$

$$x_{i+1} = x_i + v_{i+1} \quad (3.29)$$

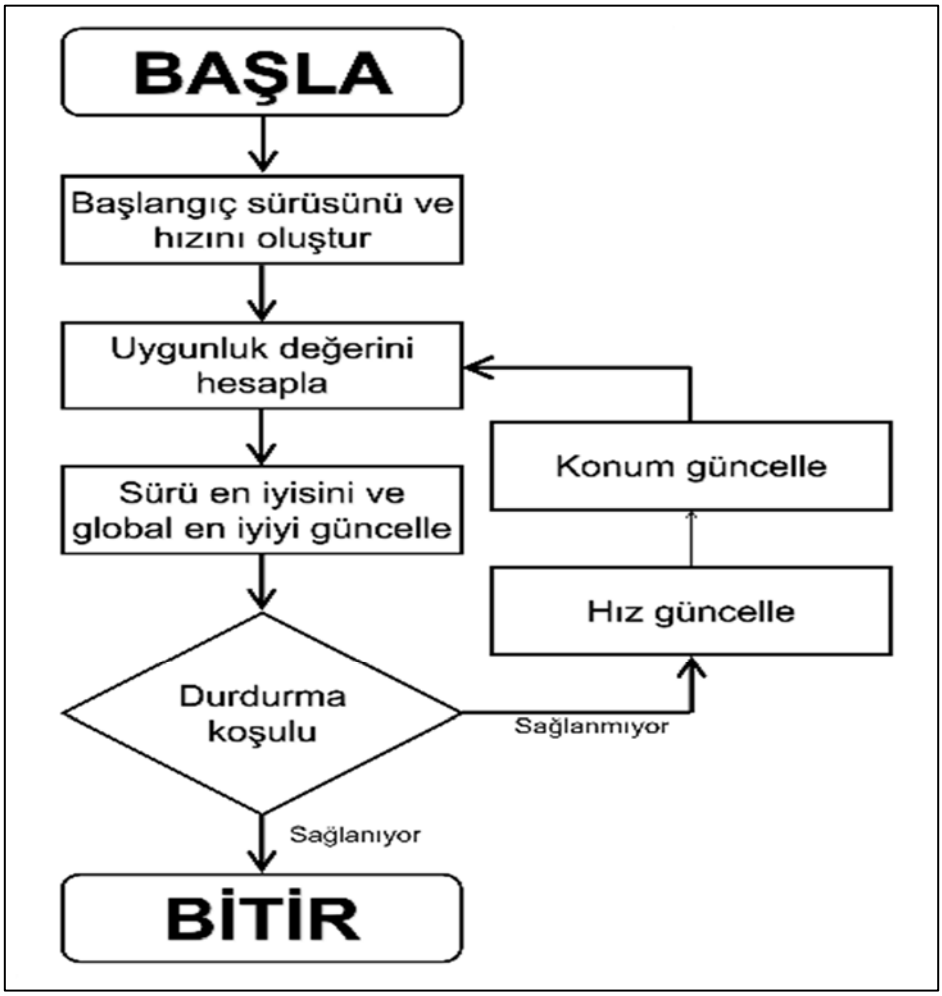
Hız değerinin çok yüksek olması algoritmanın ilerleyişini olumsuz etkilediği için hız değerleri sınırlandırılmaktadır. Bu çalışmada hız değeri parçacıkların maksimum alabileceği değerlerin $-\%20$'si ile $+\%20$'si arasında sınırlandırılmıştır. Örneğin çözüm kümesinin q_{11} elemanının alabileceği maksimum değer Arama Uzayının Belirlenmesi Bölümünden hatırlanacağı üzere 700'dür. Dolayısıyla q_{11} parçacığı için hesaplanabilecek maksimum ve minimum hız değerleri $[-140, +140]$ aralığında olacaktır. Eğer q_{11}

elemanının hız değeri, bu aralıkların dışında bir değer olarak hesaplanırsa, yakın olan sınır değerine çekilir. Hız değerlerinin diğer çözüm elemanlarına göre belirlenen sınırları Çizelge 3.13'te verilmiştir.

Çizelge 3.13. Parçacık sürü optimizasyonu algoritması için belirlenen hız limitleri

	v_{q11}	v_{q22}	v_{q33}	v_{q44}	v_{q55}	v_{q66}	v_{r11}	v_{r22}	v_{r33}	v_{r44}
Üst sınır	140	100	100	4	10	10	0.4	0.4	0.4	0.4
Alt sınır	-140	-100	-100	-4	-10	-10	-0.4	-0.4	-0.4	-0.4

Durdurma koşulu sağlanıncaya kadar sürünün önce hızı sonra konumu güncellenerek iteratif olarak algoritma çalışmaya devam eder. Durdurma koşulu belirlenen iterasyon sayısıdır. Her iterasyonda çözüm kümesi ve hız değerleri için belirlenen sınır değerleri kontrol edilir. Parçacık sürü optimizasyon algoritmasının akış diyagramı Şekil 3.10'da verilmiştir.



Şekil 3.10. Parçacık sürü optimizasyon algoritmasının akış diyagramı

3.2.3.2. Algoritma İçin Parametre Optimizasyonu

Parçacık sürü optimizasyon algoritmasının arama kalitesini etkileyen Denklem (3.28)'den de görüleceği üzere üç tane parametresi vardır. Bunlar atalet ağırlığı (w) ve öğrenme katsayılarıdır (c_1 ve c_2). Üç parametre bilindiği üzere parçacığın yeni hızını hesaplarken, dolayısıyla bir sonraki parçacığı belirlerken kullanılmaktadır.

Atalet ağırlığı, algoritmada doğrudan parçacığın güncel hızıyla çarpılmaktadır. Atalet ağırlığı $[0, 1]$ aralığında seçilebilmektedir. Ancak yapılan çalışmalar, atalet ağırlığının sabit alınması yerine iterasyonlar arttıkça azaltılmasının daha iyi sonuç verdiğini göstermektedir [23]. Bu şekilde başlarda büyük hızlarla lokal minimumlara takılmadan global arama gerçekleştirilebilirken, iterasyonlar arttıkça hız değerinin küçülmesiyle, daha lokal bölgelerde arama yapılarak, global sonuca ulaşmak daha da mümkün olmaktadır. Bu çalışmada geometrik azaltma yöntemi kullanılarak atalet ağırlığı azaltılmıştır. Bunun için iterasyonlara göre değişen bir azaltma katsayısı hesaplanır. Azaltma katsayısına Denklem (3.30) kullanılarak ulaşılır.

$$ak = \left(\frac{W_{son}}{W_{ilk}}\right)^{\frac{1}{Max_{it}-1}} \tag{3.30}$$

$$w = w \times ak \tag{3.31}$$

Burada ak, azaltma katsayısını; w_{son}, son atalet ağırlığı değerini; w_{ilk}, başlangıç atalet ağırlığı değerini ve max_{it}, belirlenen maksimum iterasyon sayısını ifade etmektedir. Denklem (3.30) kullanılarak hesaplanan azaltma katsayısı Denklem (3.31)'de iterasyonlar boyunca atalet ağırlığı ile çarpılarak atalet ağırlığı güncellenir.

Öğrenme katsayısının ilki, oluşacak yeni parçacığı daha önce ulaşmış olduğu en iyi değere yaklaştırmak, ikincisi ise yeni parçacığı global en iyi değere yaklaştırmak için kullanılır. Öğrenme faktörleri $[0, 2]$ aralığında seçilebilir. Kaynaklarda öğrenme katsayısının değeri 2 ya da 2'ye yakın seçilmektedir [24].

Algoritmalar İçin Parametre Optimizasyonu bölümünde açıklandığı üzere, üç parametrenin optimizasyon işleminin yapılması gerekmektedir. Optimize edilecek parametreler için şu değerler belirlenmiştir;

- Öğrenme katsayıları eşit kabul edilerek sırasıyla 1.50, 1.75 ve 2.00 seçilmiştir.
- Atalet ağırlığı ise sırasıyla 1'den 0.5'e, 0.9'dan 0.4'e ve 0.8'den 0.3'e doğru geometrik olarak azalan seçilmiştir.

- Parçacık sürü optimizasyon algoritmasında sürü büyüklüğü, popülasyon büyüklüğüne karşılık gelmektedir.

Parametreler için yukarıda belirlenen değerler kullanılarak dokuz farklı parametre çifti oluşturulmuştur. Oluşturulan her bir parametre çifti için algoritma üçer defa çalıştırılmıştır. Her parametre çifti için üç farklı uygunluk değeri elde edilmiştir. Elde edilen uygunluk değerlerinin aritmetik ortalaması alınarak Çizelge 3.14 oluşturulmuştur. Çizelge 3.14 görüleceği üzere öğrenme faktörlerinin 1.50 ve atalet ağırlığının 0.8'den 0.3'e doğru azalacak şekilde seçilmesine karar verilmiştir.

Çizelge 3.14. Parçacık sürü optimizasyon algoritması için parametre optimizasyonu (İterasyon Sayısı = 100, Popülasyon büyüklüğü = 50)

w \ c_1, c_2	**1.50**	**1.75**	**2.00**
(0.8-0.3)	**27.89701363**	28.16664001	28.32962629
(0.9-0.4)	36.35311879	32.9109729	32.53968408
(1.0-0.5)	34.15991858	30.41222003	30.32481318

3.2.3.3. Sonuçlar

Parametre optimizasyonu sonucunda ulaşılan öğrenme katsayılarına ve atalet ağırlığına ek olarak iterasyon sayısı 200 ve sürü büyüklüğü 100 olarak seçilerek parçacık sürü optimizasyon algoritması on defa çalıştırılmıştır. Belirtilen şartlarda gerçekleşen 10 çalışmanın sonucunda elde edilen veriler özetlenerek Çizelge 3.15'te sunulmuştur.

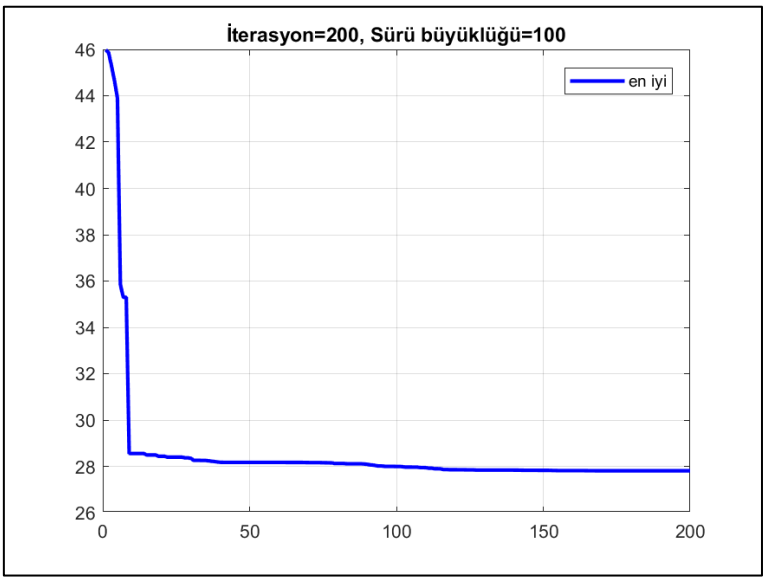
Çizelge 3.15. İterasyon Sayısı = 200, Popülasyon Büyüklüğü = 100, c1 = c2 = 1.50, w = (0.8'den 0.3'e geometrik azalan) değerleri için 10 kez çalıştırılan parçacık sürü optimizasyon algoritmasının sonuçları

En iyi uygunluk değeri	En kötü uygunluk değeri	Ortalama uygunluk değeri	Aralık	Standart Sapma	Ortalama Geçen süre
27.8077	38.7970	33.5334	10.9892	4.5893	2914.2415

Uygunluk değerlerinin en iyisi Çizelge 3.15'te 27.8077 olarak verilmiştir. Bu değer, bu tez çalışması kapsamında ulaşılan en iyi uygunluk değerine (27.5239) yakın bir

değerdir. Ancak parçacık sürü optimizasyon algoritması için standart sapma değerinin de yüksek olduğu görülmektedir.

Şekil 3.11'de parçacık sürü optimizasyon algoritması tarafından ulaşılan en iyi uygunluk değerine ait çalışmanın yakınsama eğrisi verilmiştir. Şekil 3.11'den de görüldüğü gibi parçacık sürü optimizasyon algoritması henüz 20. iterasyona gelmeden amaç fonksiyonu için bulunabilecek en iyi değere yaklaşmıştır. Sonraki 180 iterayonda ise 1 puandan az bir gelişme yaşanmıştır.

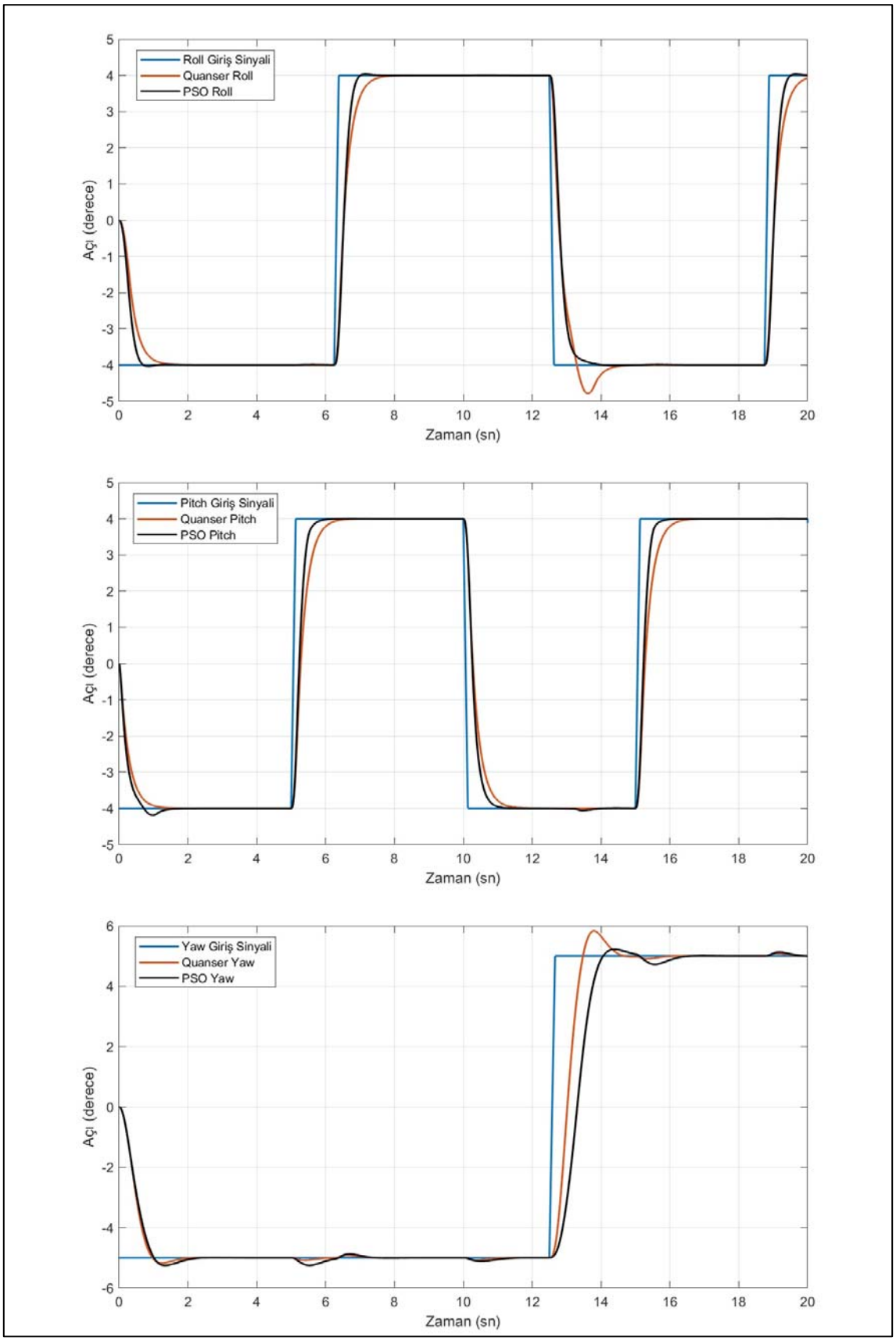


Şekil 3.11. Parçacık sürü optimizasyon algoritmasının yakınsama eğrisi

Çizelge 3.16'da parçacık sürü optimizasyon algoritmasının ulaştığı en iyi uygunluk değeri olan 27.8077 değerini veren çözüm kümesi verilmiştir. Çizelgedeki veriler Q ve R matrislerini oluşturmaktadır. Oluşturulan matrisler Ricatti Denkleminde kullanılarak denklemin kökü olan K matrisi hesaplanmıştır. K matrisi 3-DOF Hover sistemine uygulandığında sistemin verdiği cevap Şekil 3.12'de görülmektedir.

Çizelge 3.16. Parçacık sürü optimizasyon algoritmasının 10 kez çalıştırılması sonucunda ulaşılan en iyi çözüm

q_{11}	q_{22}	q_{33}	q_{44}	q_{55}	q_{66}	r_{11}	r_{22}	r_{33}	r_{44}
676.1471	499.9487	476.2601	10.5375	12.9570	6.4731	0.01	0.0191	0.0465	0.1237



Şekil 3.12. Giriş sinyalleri, Quanser parametreleri ile hesaplanan çıkış sinyalleri ve parçacık sürü optimizasyon algoritması ile hesaplanan en iyi çıkış sinyalleri

Şekil 3.12'de mavi, kırmızı ve siyah olmak üzere üç farklı çizgi ile belirtilen sinyaller bulunmaktadır. Mavi çizgi ile gösterilen sinyaller giriş sinyaline aittir. Kırmızı çizgi ile gösterilen sinyaller, Quanser firmasının önerdiği kontrolcü parametrelerinin kullanılmasıyla elde edilen çıkış sinyallerine aittir. Son olarak siyah çizgi ile gösterilen sinyaller, parçacık sürü optimizasyon algoritması kullanılarak hesaplanan kontrolcü parametrelerinin, sistemde uygulanmasıyla elde edilen sistem cevaplarına ait çıkış sinyalleridir. Çizelge 3.17'de Şekil 3.12'de verilen çıkış sinyallerinin yorumlanabilmesi ve karşılaştırılabilmesi için bazı bilgiler yer almaktadır.

Çizelge 3.17. Şekil 3.12'deki Quanser çıkış sinyalleri ile parçacık sürü optimizasyon algoritması ile hesaplanan çıkış sinyallerinin karşılaştırılması

	Oturma Zamanı (sn)			Yükselme Zamanı (sn)			Üst Aşım (deg)		
	Roll	Pitch	Yaw	Roll	Pitch	Yaw	Roll	Pitch	Yaw
Qua	19.7107	16.0741	14.3349	0.5042	0.5440	0.3475	0.7890	0.0003	0.8323
PSO	19.3961	15.6035	15.8231	0.2763	0.2677	0.5182	0.0337	0.1882	0.2557
Fark	0.3146	0.4706	-1.4882	0.2279	0.2763	-0.1707	0.7553	-0.1879	0.5766

Çizelge 3.17'de verilen parçacık sürü optimizasyon algoritması kullanılarak ulaşılan parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları ile Quanser firmasının parametrelerinin sistemde uygulanmasıyla elde edilen sistem cevapları karşılaştırılmıştır. İlk satırda Quanser firmasının önerdiği parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları hakkında bilgiler yer almaktadır. İkinci satırda parçacık sürü optimizasyon algoritmasının kullanılmasıyla elde edilen parametrelerin sistemde uygulanmasıyla ulaşılan sistem cevabı hakkında bilgiler yer almaktadır. Son satırda ise bahsedilen bu bilgilerin farkları yer almaktadır. Parçacık sürü optimizasyon algoritmasının başarısız olduğu sütunlar fark satırında negatif değere sahip olan sütunlardır.

Çizelge 3.17'de verilerin çoğunluğunun iyileştiği görülmektedir. Üst aşım değerlerine bakıldığında sadece pitch açısının değerinin kötüleştiği anlaşılmaktadır. Bunun nedeni Quanser firmasının sunduğu parametrelerin pitch açısının üst aşımı için mükemmele yakın (0.0003 derece) sonuç vermesinden dolayıdır. Amaç Fonksiyonunun Belirlenmesi Bölümünde gerektiğinde pitch açısında bozulmaların yaşanabileceği belirtilmiştir. Pitch açısında yaşanan 0.1879 derecelik bozulma roll ve yaw cevabındaki üst aşımların giderilmesine imkân sağlamıştır.

Oturma zamanlarında gerçekleşen kötüleşme 1.4882 saniye ile yaw cevabında yaşanmıştır. Ancak bilindiği üzere yaw açısı, Z-ekseninde dönmeyi ifade eder ve Z-ekseni, yer çekimi ivmesinin yönüyle çakışıktır. Yaw açısında bozulmaya izin verilerek uçuş için daha önemli olan roll ve pitch açılarının oturma zamanları iyileştirilebilmiştir.

3.2.4. Benzetilmiş Tavlama Algoritması

Benzetilmiş tavlama algoritması (Simulated Annealing Algorithms - SAA) 1983 yılında S. Kirkpatrick ve arkadaşları tarafından önerilmiştir. Algoritma doğrusal olmayan problemlerin çözümü için geliştirilmiştir. Algoritma, metallerin tavlama sürecini taklit etmektedir. Metallerin önce ısıtılıp daha sonra moleküllerinin daha kararlı durumlara yayılmalarına imkân verecek şekilde soğutulması ile sağlanan işlem, tavlama olarak adlandırılmaktadır. Benzetilmiş tavlama algoritmasında bu olay problemler için uygulanmaktadır. Benzetilmiş tavlama algoritmasının en önemli avantajı yerel minimuma takılmayı önleyen bir yapıya sahip olmasıdır [25].

3.2.4.1. Algoritmanın İşleyişi

Benzetilmiş tavlama algoritmasına başlangıç sıcaklık değeri, bitiş sıcaklık değeri ve başlangıç konumu belirlenerek başlanır. Başlangıç konumu için 100 adet rasgele çözüm oluşturulmuş ve bunların içinden en iyi uygunluk değerini veren çözüm başlangıç konumu olarak seçilmiştir. Aday başlangıç konumları Denklem (3.32)'de formüle edildiği gibi oluşturulmuştur.

$$X_{\mathrm{ij}} = X_j^{max} \times rand + X_j^{min} \tag{3.32}$$

Burada X, oluşturulan i. çözüm kümesini; j, 1'den problemin boyutuna kadar olan sayıyı ve rand, 0 ile 1 arasında oluşturulan rasgele sayıları temsil etmektedir. Başlangıç konumu rasgele bir noktaya taşınarak algoritma çalışmaya devam eder. Yeni gelinen konumun uygunluk değeri hesaplanır. Yeni uygunluk değeri daha iyi ise yeni konum, en iyi çözüm olarak kabul edilir ve saklanır. Yeni çözüm daha kötü ise uygunluk değerleri arasındaki fark Denklem (3.33) ve Denklem (3.34) kullanılarak kötü çözümü kabul etme olasılığı hesaplanır. Hesaplanan değer ile rasgele üretilecek bir sayı karşılaştırılarak kötü çözüm kabul edilir ya da reddedilir.

$$\Delta f = Je_{yeni} - Je_{kabul} \tag{3.33}$$

$$w = e^{-\left(\frac{\Delta f}{t}\right)} \tag{3.34}$$

Burada Je_{yeni}, yeni konumun uygunluk değerini; Je_{kabul}, mevcut konumun uygunluk değerini ve t, o anki sıcaklığı ifade etmektedir.

Bu çalışmada, temel benzetilmiş tavlama algoritmasında biraz değişiklik yapılmış ve yeni çözüm Denklem (3.35) kullanılarak üretilmiştir.

$$X_{yeni}(i) = X_{kabul}(i) + rand \times X_{eniyi}(i) - \frac{X_{eniyi}(i)}{2} \tag{3.35}$$

Burada X_{yeni}, üretilecek yeni konumu; X_{kabul}, mevcut konumu; X_{eniyi}, o zamana kadar bulunmuş en iyi konumu; $rand$, 0 ile 1 arasında üretilen rasgele sayıyı ve i, konumların i. elemanını temsil etmektedir.

Benzetilmiş tavlama algoritması bilindiği üzere popülasyon temelli bir algoritma değildir. Benzetilmiş tavlamada yeni çözümü kabul etmek için denemeler yapılmaktadır. Algoritma bir sonraki iterasyonun rasgele oluşturulacak konumuna, girilen deneme sayısı kadar deneme yaparak ulaşmaktadır. Belirlenen deneme sayısı kadar deneme yapıldıktan sonra sıcaklık bir miktar azaltılarak iterasyonlara t_{end} sıcaklığına ulaşıncaya kadar devam edilir.

Literatürde başlangıç ve bitiş sıcaklıklarının belirlenmesi konusunda farklı görüşler vardır. Seçilmesi muhtemel değerler arasında çok yüksek farklar olduğu için bu parametreler için uygun bir parametre optimizasyonu gerçekleştirilememiştir. Ancak ortak fikir başlangıç sıcaklık değerinin yüksek, bitiş sıcaklık değerinin düşük bir değer seçilmesi gerektiğidir [26].

Başlangıç sıcaklığını bitiş sıcaklığı değerine kadar soğutmanın çeşitli yöntemleri vardır [27]. Bu tez çalışmasında kullanılan soğutma yöntemine geometrik soğutma yöntemi denilmektedir. Bahsedilen yöntem için soğutma katsayısı Denklem (3.36)'daki gibi hesaplanır.

$$frac1 = \left(\frac{t_{end}}{t_{start}}\right)^{\frac{1}{GN-1}} \tag{3.36}$$

Denklemde GN, girilen iterasyon sayısını ve DN, girilen deneme sayısını ifade etmektedir. Benzetilmiş tavlama algoritmasında durdurma koşulu sağlanıncaya kadar bu

işlemlere devam edilir. Şekil 3.13'te benzetilmiş tavlama algoritmasın basit kodu verilmiştir.

```
GN, DN, t_start, t_end, X_Start değerlerini belirle
sayaç1 = 1
frac = (t_end / t_start)^(1/(GN−1))
while t_start > t_end
    X_kabul, X_eniyi = X_start
    sayaç2 = 1
    while sayaç2 < DN
        X_yeni konumunu hesapla
        Je_k(X_kabul), Je_e(X_eniyi), Je_y(X_yeni) uygunluk değerlerini hesapla
        Delta'yı hesapla
        if Je_y < Je_e
            X_eniyi = X_yeni
            Je_e = Je_y
        end
        if Delta < 0
            X_kabul = X_yeni
        else
            w kabul olasılığını hesapla
            if rand < w
                X_kabul = X_yeni
            end
        end
        sayaç2 = sayaç2 + 1
    end
    Xyeni = Xeniyi
    tStart = tStart × frac
    sayaç1 = sayaç1 + 1
end
Sonuçları yazdır.
```

Şekil 3.13. Benzetilmiş tavlama algoritmasının basit kodu

3.2.4.2. Sonuçlar

Benzetilmiş tavlama algoritması iterasyon sayısı 200, deneme sayısı 100, başlangıç sıcaklık değeri 100 ve bitiş sıcaklık değeri 0.01 seçilerek 10 defa çalıştırılmıştır. Belirtilen şartlarda gerçekleşen on çalışmanın sonucunda elde edilen veriler özetlenerek Çizelge 3.18'de sunulmuştur.

Çizelge 3.18. İterasyon Sayısı = 200, Deneme Sayısı = 100, Başlangıç Sıcaklığı = 100, Bitiş Sıcaklığı = 0.01 değerleri ve geometrik soğutma yöntemi kullanılarak benzetilmiş tavlama algoritmasının 10 kez çalıştırılması neticesinde elde edilen sonuçlar

En iyi uygunluk değeri	En kötü uygunluk değeri	Ortalama uygunluk değeri	Aralık	Standart Sapma	Ortalama Geçen süre
27.6713	41.9617	33.2717	14.2904	5.3609	3184.2986

Ulaşılan uygunluk değerlerinin en iyisi Çizelge 3.18Çizelge 3.15'de 27.6713 olarak verilmiştir. Bu değer, bu tez çalışması kapsamında ulaşılan en iyi uygunluk değerine (27.5239) yakın bir değerdir. Ancak benzetilmiş tavlama algoritması için standart sapma değerinin de oldukça yüksek olduğu görülmektedir. Dolayısıyla ulaşılan sonuçların tekrarlanabilirliği açısından sorunlar olduğu anlaşılmaktadır.

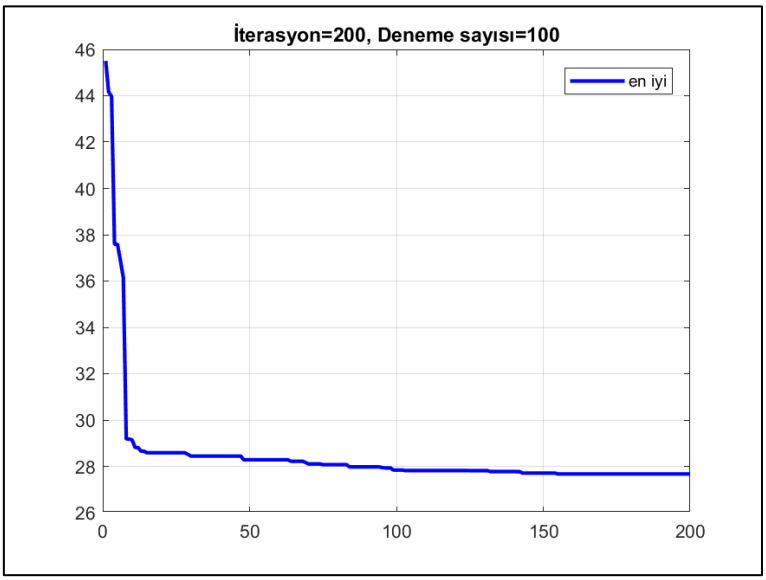
Benzetilmiş tavlama algoritmasının çok farklı sonuçlar elde etmesinin nedeni rasgele oluşturulan başlangıç konumudur. Çizelge 3.19'da benzetilmiş tavlama algoritmasının 10 defa çalıştırılması esnasında bulunan başlangıç konumları ve bitiş konumları verilmiştir.

Çizelge 3.19. Benzetilmiş tavlama algoritmasının 200 iterasyon ve 100 deneme sayısı ile başlangıç konumlarına göre ulaşılan ile en iyi konumları (UD = Uygunluk Değeri)

Çalışma	Konum	UD	q_{11}	q_{22}	q_{33}	q_{44}	q_{55}	q_{66}	r_{11}	r_{22}	r_{33}	r_{44}
1	Başlangıç	44.46	**1.32**	**1.00**	202.82	18.69	31.47	30.77	0.69	1.29	0.60	1.61
	Çözüm	41.86	**1.00**	**1.00**	464.64	0.00	31.77	12.83	0.74	2.00	0.05	2.00
2	Başlangıç	45.06	**1.00**	**19.25**	260.10	0.50	13.01	20.63	1.53	0.49	0.19	2.00
	Çözüm	41.96	**1.00**	**1.00**	65.78	0.00	50.00	2.01	0.05	2.00	0.01	2.00
3	Başlangıç	42.52	592.09	500.00	220.85	3.21	21.10	0.73	0.63	0.41	0.05	0.13
	Çözüm	37.45	65.25	482.20	500.00	1.60	0.03	0.02	0.06	0.13	0.06	0.03
4	Başlangıç	44.19	697.48	462.76	205.71	0.06	10.37	14.97	1.03	0.08	0.24	0.54
	Çözüm	27.79	700.00	338.72	84.73	0.11	4.19	0.03	0.01	0.02	0.01	0.07
5	Başlangıç	43.22	700.00	326.87	178.13	2.33	10.37	6.29	0.16	0.13	0.08	1.61
	Çözüm	33.21	249.44	323.81	500.00	20.00	3.36	7.45	0.05	0.03	0.23	0.01
6	Başlangıç	43.40	30.17	231.43	500.00	2.59	0.01	50.00	0.02	0.04	0.29	0.10
	Çözüm	27.89	700.00	157.04	58.51	12.32	0.19	0.01	0.01	0.01	0.01	0.05
7	Başlangıç	46.76	556.10	435.41	89.85	10.60	32.60	9.19	2.00	0.97	0.69	2.00
	Çözüm	27.79	619.69	395.79	238.05	0.00	6.87	3.61	0.01	0.02	0.02	0.10
8	Başlangıç	45.50	695.49	419.94	458.01	17.84	36.13	17.12	1.87	0.23	0.77	1.30
	Çözüm	27.67	700.00	500.00	59.95	0.00	8.74	0.14	0.01	0.01	0.01	0.05
9	Başlangıç	43.30	140.11	500.00	257.80	5.96	11.14	3.01	0.31	0.08	0.25	0.01
	Çözüm	33.32	700.00	500.00	1.01	2.95	7.00	14.41	0.01	0.01	0.04	0.01
10	Başlangıç	44.67	**3.50**	322.72	348.63	0.05	9.18	12.61	1.36	0.02	0.02	1.09
	Çözüm	33.77	11.57	421.03	51.80	0.03	6.24	0.93	0.04	0.03	0.01	0.03

Çizelge 3.19'da birinci ve ikinci satırda koyu ile belirtilmiş parametre değerleri üzerinde çalışılan problem için uygun olmayan değerlerdir. Diğer bir örnek için 10. satır verilebilir. Ancak tek bir parametre kötü olduğu için benzetilmiş tavlama algoritması gelişim sağlamış ve uygunluk değerini çok daha iyi bir noktaya taşımıştır. Görüldüğü üzere başlangıç konumu olması gerekenden çok kötü seçildiğinde benzetilmiş tavlama algoritması uygun bir çözüm üretememektedir.

Benzetilmiş tavlama algoritmasının ulaştığı en iyi sonuca ait çalışmanın yakınsama eğrisi Şekil 3.14'te verilmiştir. Şekilden algoritma henüz 20. iterasyonda bu tez çalışmasında ulaşılan en iyi uygunluk değerine yaklaştığı görülmektedir.

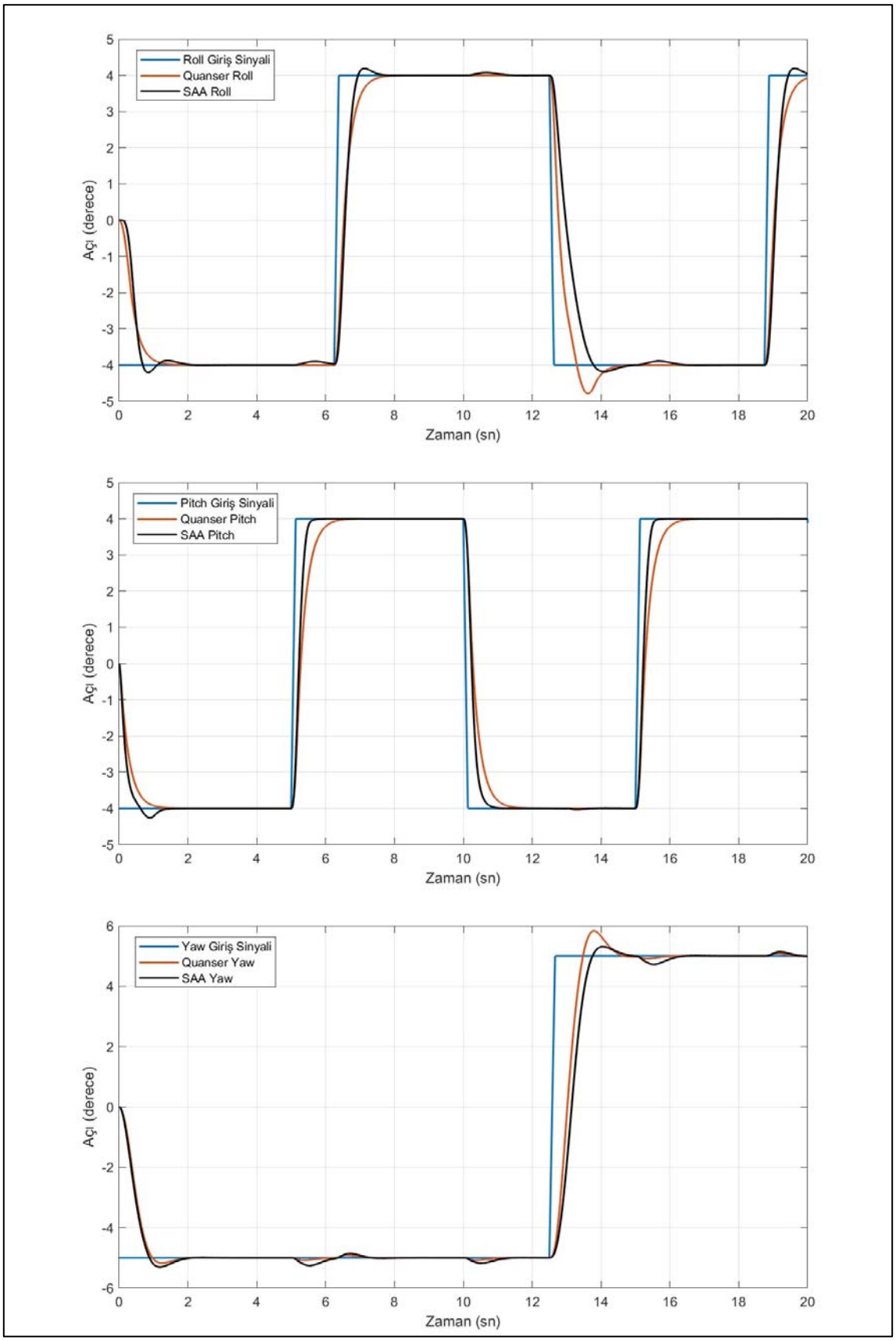


Şekil 3.14. Benzetilmiş tavlama algoritmasının yakınsama eğrisi

Çizelge 3.20'de en iyi uygunluk değeri olan 27.6713 değerini veren çözüm kümesi verilmiştir. Çizelge 3.20'deki veriler Q ve R matrislerinin diegonal elemanlarını oluşturmaktadır. Oluşturulan matrisler Ricatti Denkleminde kullanılarak denklemin kökü olan K matrisi hesaplanmıştır. K matrisi 3-DOF Hover sistemine uygulandığında sistemin verdiği cevap Şekil 3.15'te görülmektedir.

Çizelge 3.20. Benzetilmiş tavlama algoritmasının 10 kez çalıştırılması sonucunda ulaşılan en iyi çözüm

q_{11}	q_{22}	q_{33}	q_{44}	q_{55}	q_{66}	r_{11}	r_{22}	r_{33}	r_{44}
700.0000	500.0000	59.9486	0.0000	8.7361	0.1376	0.0100	0.0137	0.0119	0.0513



Şekil 3.15. Giriş sinyalleri, Quanser parametreleri ile hesaplanan çıkış sinyalleri ve benzetilmiş tavlama algoritması ile hesaplanan en iyi çıkış sinyalleri

Şekil 3.15'te mavi, kırmızı ve siyah olmak üzere üç farklı çizgi ile belirtilen sinyaller bulunmaktadır. Mavi çizgi ile gösterilen sinyaller giriş sinyaline aittir. Kırmızı çizgi ile gösterilen sinyaller, Quanser firmasının önerdiği kontrolcü parametrelerinin kullanılmasıyla elde edilen çıkış sinyallerine aittir. Son olarak siyah çizgi ile gösterilen sinyaller, Benzetilmiş tavlama algoritması kullanılarak hesaplanan kontrolcü parametrelerinin, sistemde uygulanmasıyla elde edilen sistem cevaplarına ait çıkış sinyalleridir. Çizelge 3.21'de Şekil 3.15'te görülen çıkış sinyallerinin yorumlanabilmesi ve karşılaştırılabilmesi için bazı bilgiler yer almaktadır.

Çizelge 3.21. Şekil 3.15'teki Quanser çıkış sinyalleri ile benzetilmiş tavlama algoritması ile hesaplanan çıkış sinyallerinin karşılaştırılması

	Oturma Zamanı (sn)			Yükselme Zamanı (sn)			Üst Aşım (deg)		
	Roll	Pitch	Yaw	Roll	Pitch	Yaw	Roll	Pitch	Yaw
Qua	19.7107	16.0741	14.3349	0.5042	0.5440	0.3475	0.7890	0.0003	0.8323
SAA	19.4051	15.5028	15.7505	0.2712	0.2122	0.4276	0.2052	0.2613	0.3122
Fark	0.3055	0.5712	-1.4155	0.2329	0.3317	-0.0800	0.5837	-0.2610	0.5201

Çizelge 3.21'de verilen benzetilmiş tavlama algoritması kullanılarak ulaşılan parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları ile Quanser firmasının parametrelerinin sistemde uygulanmasıyla elde edilen sistem cevapları karşılaştırılmıştır. İlk satırda Quanser firmasının önerdiği parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları hakkında bilgiler yer almaktadır. İkinci satırda benzetilmiş tavlama algoritmasının kullanılmasıyla elde edilen parametrelerin, sistemde uygulanmasıyla ulaşılan sistem cevabı hakkında bilgiler yer almaktadır. Son satırda ise bahsedilen bu bilgilerin farkları yer almaktadır. Benzetilmiş tavlama algoritmasının başarısız olduğu sütunlar fark satırında negatif değere sahip olan sütunlardır.

Çizelge 3.21'de verilerin çoğunluğunun iyileştiği görülmektedir. Üst aşım değerlerine bakıldığında sadece pitch açısının değerinin kötüleştiği anlaşılmaktadır. Bunun nedeni daha öncede anlatıldığı gibi Quanser firmasının sunduğu parametrelerin pitch açısının üst aşımı için mükemmele yakın (0.0003 derece) sonuç vermesinden dolayıdır. Sistemin roll ve yaw cevabındaki üst aşımlar azaltılmıştır.

Oturma zamanlarında gerçekleşen en kötü sonuç yaw cevabında yaşanmıştır. Ancak daha öncede anlatıldığı üzere yaw açısı, Z-ekseninde dönmeyi ifade eder ve Z-ekseni, yer çekimi ivmesinin yönüyle çakışıktır. Bundan dolayı burada yaşanan

kötüleşme diğer verilerin iyileşmesinde etkili olmaktadır. Yükselme zamanlarına bakıldığında yine yaw açısında kötüleşme yaşanmıştır. Bu sayede hem roll hem de pitch cevabında iyileşmeler gerçekleşmiştir.

3.2.5. Yapay Arı Kolonisi Algoritması

Yapay arı koloni (Artificial Bee Colony - ABC) algoritması Derviş Karaboğa tarafından önerilmiş bir optimizasyon algoritmasıdır. Algoritma, arıların besin ararken sergiledikleri sosyal davranışlar modellenerek geliştirilmiştir. Arıların modellendiği diğer algoritmaların aksine çok boyutlu problemler ile de çalışabilmektedir. Algoritma popülasyon temellidir ve sürü zekâsını kullanmaktadır [28].

3.2.5.1. Algoritmanın İşleyişi

Yapay arı kolonisi algoritması işçi arı, gözcü arı ve kâşif arı fazı olmak üzere üç fazdan oluşmaktadır. Algoritma rastgele oluşturulan başlangıç çözüm kümesi ile başlar. Başlangıç çözüm kümesine besin kaynakları denilmektedir ve Denklem (3.37)'deki gibi oluşturulur.

$$X_{ij} = X_j^{min} + rand \times (X_j^{max} - X_j^{min}) \tag{3.37}$$

Burada X_{ij}, oluşturulacak besin kaynağını; i, 1'den besin kaynağı sayısına kadar olan sayıyı; j, problemin boyutunu; X_j^{min}, j. elemanın alabileceği minimum değeri; X_j^{max}, j. elemanın alabileceği maksimum değeri ve $rand$, 0 ile 1 arasında rasgele üretilecek sayıyı temsil etmektedir. Başlangıç besin kaynaklarının nektar miktarı hesaplanır. Yapay arı kolonisi algoritmasında uygunluk değeri ilgili kaynağın nektar miktarına karşılık gelir.

Yapay arı kolonisi algoritması işçi arı, gözcü arı ve kâşif arı fazlarından geçerek en iyi çözüme ulaşmaya çalışmaktadır. İşçi arı fazı, oluşturulan besin kaynaklarına işçi arıların gönderilmesiyle başlar. Her bir besin kaynağından sorumlu bir işçi arı vardır. İşçi arılar için Denklem (3.38) yardımıyla komşu yeni besin kaynakları üretilir.

$$V_{ij} = X_{ij} + \phi_{ij} \times (X_{ij} - X_{kj}) \tag{3.38}$$

Burada V_{ij}, X_{ij} besin kaynağına gönderilen işçi arı için üretilen yeni komşu besin kaynağını; X_{kj}, mevcut besin kaynakları arasından rasgele seçilen bir besin kaynağını ve

ϕ_{ij}, 0 ile 1 arasında problemin üretilen rasgele sayıları ifade etmektedir. Üretilen komşu besin kaynağının parametre sınırları kontrol edilir ve sınır değerlerini aşan parametreler yakın olan sınır değerine çekilir. Üretilen komşu besin kaynağının nektar miktarı hesaplanır. Eğer komşu besin kaynağının nektar miktarı, mevcut besin kaynağının nektar miktarından daha iyi ise eski kaynak ile yeni kaynak yer değiştirilir ve her besin kaynağı için ayrı ayrı oluşturulan geliştirememe sayacı adı verilen sayaç sıfırlanır. Aksi durumda geliştirilememe sayacını bir arttırılır ve mevcut besin kaynağı korunur.

Denklem (3.39) kullanılarak gözcü arıların seçim işleminde kullanacakları uygunluk değerlerine dayalı olasılık değerleri hesaplanır.

$$P_i = \frac{Nektar_i}{\sum_{i=1}^{N} Nektar_i} \tag{3.39}$$

Rulet tekerleğine göre seçim işleminde her bir kaynak için [0, 1] aralığında rastgele bir sayı üretilir. Rastgele üretilen sayı Denklemde (3.39)'da hesaplanan P_i oranından daha küçük ise gözcü arı için Denklem (3.38) kullanılarak yeni kaynak üretilir. Eğer yeni üretilen kaynak daha iyi ise eski kaynak terk edilerek yeni kaynağa geçilir ve geliştirilememe sayacı sıfırlanır. Şayet eski kaynak daha iyi ise sadece geliştirilememe sayacı bir arttırılır. Bu adım tüm gözcü arılar yiyecek kaynağı bölgelerine dağılıncaya kadar tekrarlanır.

Yapay arı kolonisi algoritmasının son fazı olan kâşif arı fazı ise geliştirilememe sayaçları, limit değerini aşan besin kaynakları için gerçekleşir. Bahsedilen nektar miktarı tükenen kaynaklar için tamamen rasgele yeni bir kaynak üretilir. Kâşif arı için üretilen yeni kaynak Denklem (3.37) kullanılarak hesaplanır. Kâşif arı fazı algoritmaya global aramada katkı sağlamaktadır. Şekil 3.16'da yapay arı koloni algoritmasının işleyişini anlatan basit kod görülmektedir.

```
Rasgele başlangıç besin kaynaklarını oluştur
while (durdurma koşulu sağlanıncaya kadar)
    İşçi arıları kaynaklara gönder ve nektar miktarlarını hesapla
    Gözcü arıları kaynaklara gönder ve nektar miktarlarını hesapla
    Rasgele yeni kaynaklar bulmaları için kaşif arıları gönder
    O ana kadarki en iyi kaynağı hafızada tut
end while
```

Şekil 3.16. Yapay arı kolonisi algoritması basit kodu [28]

3.2.5.2. Algoritma İçin Parametre Optimizasyonu

Yapay arı kolonisi algoritmasının lokal minimumdan kurtulmasını sağlayan limit adında bir parametresi vardır. Limit parametresi, işçi arının gittiği besin kaynağının geliştirilememesi sonucunda o besin kaynağının terk edilmesini sağlayan bir parametredir. İşçi arılar, besin kaynaklarına limit sayısı kadar gitmesine rağmen besin kaynağı geliştirilememiş ise seçilen bir işçi arı kâşif arıya dönüşür ve kaynak terk edilir. Limit parametresi olması gerekenden büyük seçildiğinde algoritma gereğinden uzun çalışacaktır. Küçük seçildiğinde ise iyi sonuçları gözden kaçıracaktır. Bunun için limit parametresinin optimizasyonunun yapılması gerekmektedir. Yapay arı kolonisi algoritması için parametre optimizasyonu şu şartlarda gerçekleştirilmiştir:

- Limit parametresinin değeri sırasıyla 20, 30, 40 ve 50 seçilmiştir.
- Yapay arı kolonisi algoritmasında besin kaynağı sayısı diğer algoritmalardaki popülasyon büyüklüğüne denk gelmektedir ve 50 olarak belirlenmiştir.

Yapay arı kolonisi algoritmasının limitten başka bir parametresi daha vardır. Bu parametre toplam işçi ve gözcü arı sayısını ifade etmektedir. Ancak bahsedilen bu parametre literatürde önerildiği gibi besin kaynağının 2 katı olarak alınmıştır.

Belirlenen limit değerleri için algoritma üçer defa çalıştırılmıştır. Limit değerleri için elde edilen uygunluk değerlerinin aritmetik ortalaması alınmıştır ve Çizelge 3.22'de listelenmiştir. Çizelge 3.22'den anlaşılacağı üzere limit sayısı 50 iken algoritma en iyi sonuçları üretmiştir. Bu nedenle deneylerde bu değer kullanılmıştır.

Çizelge 3.22. Yapay arı kolonisi algoritması için parametre optimizasyonu (İterasyon Sayısı = 100, Besin Kaynağı Sayısını = 50)

Limit=20	33.88347095
Limit=30	32.25558793
Limit=40	33.20713235
Limit=50	31.1199672

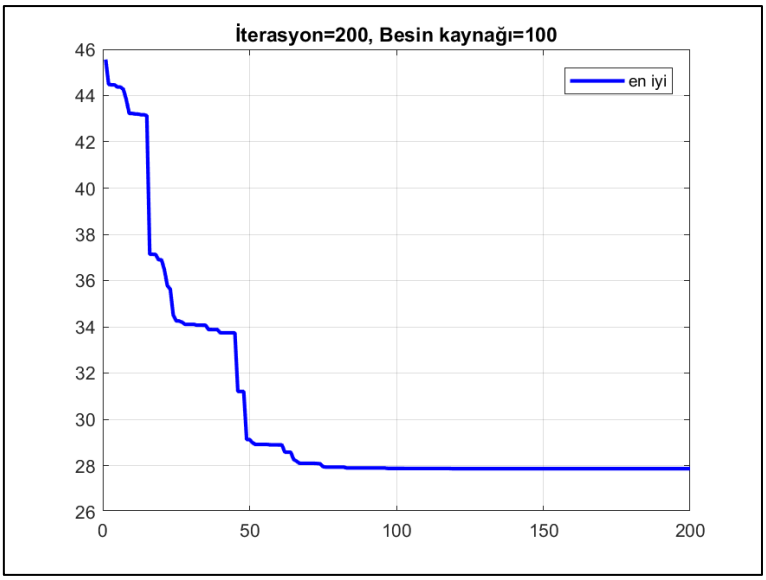
3.2.5.3. Sonuçlar

Parametre optimizasyonu sonucunda ulaşılan limit değerine ek olarak iterasyon sayısı 200 ve besin kaynağı sayısı 100 seçilerek yapay arı kolonisi algoritması 10 defa çalıştırılmıştır. Belirtilen şartlarda gerçekleşen on çalışmanın sonucunda elde edilen veriler özetlenerek Çizelge 3.23'te sunulmuştur.

Çizelge 3.23. İterasyon Sayısı = 200, Besin Kaynağı Sayısı = 100, Limit = 50 değerleri için 10 kez çalıştırılan yapay arı kolonisi algoritmasının sonuçları

En iyi uygunluk değeri	En kötü uygunluk değeri	Ortalama uygunluk değeri	Aralık	Standart Sapma	Ortalama Geçen süre
27.8650	31.9778	29.5503	4.1127	1.3140	5694.5955

Yapay arı kolonisi algoritmasının ulaştığı en iyi uygunluk değerinin Çizelge 3.23'te 27.8650 olduğu görülmektedir. Bu uygunluk değerinin elde edildiği çalışmaya ait yakınsama eğrisi Şekil 3.17'de verilmiştir.



Şekil 3.17. Yapay arı kolonisi algoritmasının yakınsama eğrisi

Şekilden de görüleceği üzere yaklaşık 75. iterasyondan sonra algoritma bu tez çalışmasında ulaşılan en iyi uygunluk değerine çok yaklaşmıştır. Bu noktadan sonra uygunluk değerinde kayda değer bir gelişme yaşanmamıştır.

Çizelge 3.24'te yapay arı kolonisi algoritması kullanılarak ulaşılan en iyi uygunluk değeri olan 27.8650 değerini veren çözüm kümesi verilmiştir. Çizelge 3.24'teki

veriler Q ve R matrislerinin diegonal elemanlarını oluşturmaktadır. Oluşturulan matrisler Ricatti denkleminde kullanılarak denklemin kökü olan K matrisi hesaplanmıştır. K matrisi 3-DOF Hover sistemine uygulandığında sistemin verdiği cevap Şekil 3.18'de görülmektedir.

Çizelge 3.24. Yapay arı kolonisi algoritmasının 10 kez çalıştırılması sonucunda ulaşılan en iyi çözüm

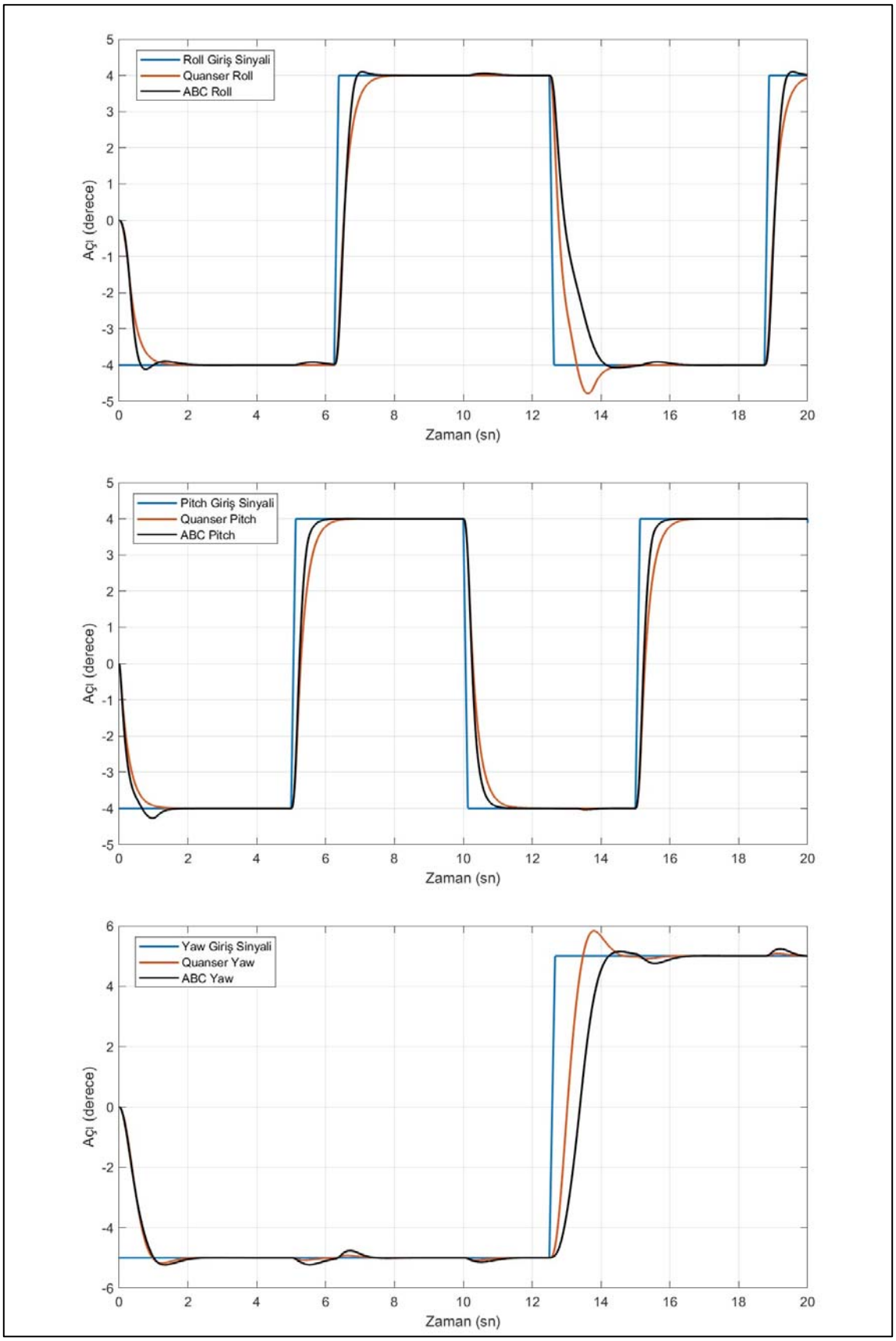
q_{11}	q_{22}	q_{33}	q_{44}	q_{55}	q_{66}	r_{11}	r_{22}	r_{33}	r_{44}
670.1780	466.6217	119.1117	20	12.0749	1.9794	0.01	0.01484	0.01	0.2215

Şekil 3.18'de mavi, kırmızı ve siyah olmak üzere üç farklı çizginin belirttiği sinyaller bulunmaktadır. Mavi çizgi ile gösterilen sinyaller giriş sinyaline aittir. Kırmızı çizgi ile gösterilen sinyaller, Quanser firmasının önerdiği kontrolcü parametrelerinin kullanılmasıyla elde edilen çıkış sinyallerine aittir. Son olarak siyah çizgi ile gösterilen sinyaller, yapay arı kolonisi algoritması kullanılarak hesaplanan kontrolcü parametrelerinin, sistemde uygulanmasıyla elde edilen sistem cevaplarına ait çıkış sinyalleridir. Çizelge 3.25'te Şekil 3.18'de verilen çıkış sinyallerinin yorumlanabilmesi ve karşılaştırılabilmesi için bazı bilgiler yer almaktadır.

Çizelge 3.25. Şekil 3.18'deki Quanser çıkış sinyalleri ile yapay arı kolonisi algoritması ile hesaplanan çıkış sinyallerinin karşılaştırılması

	Oturma Zamanı (sn)			Yükselme Zamanı (sn)			Üst Aşım (deg)		
	Roll	Pitch	Yaw	Roll	Pitch	Yaw	Roll	Pitch	Yaw
Qua	19.7107	16.0741	14.3349	0.5042	0.5440	0.3475	0.7890	0.0003	0.8323
ABC	19.3626	15.6683	19.3429	0.2554	0.2956	0.5559	0.1120	0.2723	0.2359
Fark	0.3481	0.4058	-5.008	0.2488	0.2484	-0.2084	0.677	-0.272	0.5964

Çizelge 3.25'te verilen yapay arı kolonisi algoritması kullanılarak ulaşılan parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları ile Quanser firmasının parametrelerinin sistemde uygulanmasıyla elde edilen sistem cevapları karşılaştırılmıştır. İlk satırda Quanser firmasının önerdiği parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları hakkında bilgiler yer almaktadır. İkinci satırda yapay arı kolonisı algoritmasının kullanılmasıyla elde edilen parametrelerin sistemde uygulanmasıyla ulaşılan sistem cevabı hakkında bilgiler yer almaktadır. Son satırda ise bahsedilen bu bilgilerin farkları yer almaktadır. Yapay arı kolonisi algoritmanın başarısız olduğu sütunlar fark satırında negatif değere sahip olan sütunlardır.



Şekil 3.18. Giriş sinyalleri, Quanser parametreleri ile hesaplanan çıkış sinyalleri ve yapay arı kolonisi algoritması ile hesaplanan en iyi çıkış sinyalleri

Çizelge 3.25'te verilerin çoğunluğunun iyileştiği görülmektedir. Üst aşım değerlerine bakıldığında sadece pitch açısının değerinin kötüleştiği görülmektedir. Sistemin roll ve yaw cevabındaki üst aşımlar azaltılabilmiştir. Oturma zamanlarında gerçekleşen en kötü sonuç 5.008 saniye ile yaw cevabında yaşanmıştır. Bu çok kötü bir sonuçtur. Ancak Şekil 3.18'de yaw açısının sistem cevabı incelendiğinde çıkış sinyalinin, giriş sinyalini çok kötü takip ettiği söylenemez. Sonucun bu kadar kötü çıkmasının nedeni simülasyon programının bu oturma zamanının hesaplarken kullandığı fonksiyonun %2'lik bir toleransla yaklaşmasından kaynaklanmaktadır. Şekil 3.18'de görüleceği üzere yaw açısı 19.202. saniyede 5.2299 dereceye ulaşarak üst aşım gerçekleştirmektedir. Buradaki ulaşılan değer, %2'lik toleransı aştığı için yaw açısının oturma zamanı bu kadar geç sürmüştür. Yaşanan üst aşım 5.2299 derece yerine 5.1999 derece olsaydı oturma zamanı daha iyi bir değer olacaktı.

3.2.6. Diferansiyel Gelişim Algoritması

Diferansiyel gelişim algoritması (Differential Evolution Algorithm - DEA) 1995 yılında Price ve Storn tarafından önerilmiştir. Genetik algoritma ile birçok ortak noktası bulunmaktadır. DEA, genetik algoritma gibi evrimsel sürece dayandırılmaktadır ve popülasyon temelli bir algoritmadır. Genetik algoritmada olduğu gibi popülasyon kromozom adı verilen çözümleri içeren bir matristen oluşmaktadır. Genetik algoritma ile çoğu özelliği benzerlik göstermesine rağmen çaprazlama, mutasyon ve doğal seçilim işlemleri farklı bir şekilde uygulanmaktadır [29].

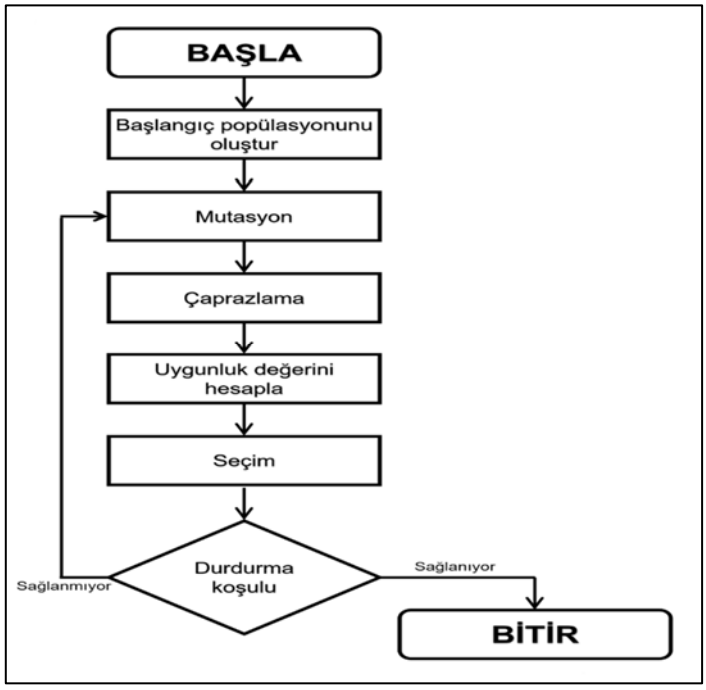
3.2.6.1. Algoritmanın İşleyişi

Algoritma rasgele oluşturulmuş başlangıç popülasyonu ile aramaya başlar. Popülasyonu kromozom adı verilen çözüm kümeleri oluşturur. Çözüm kümesinin her bir elemanı gen olarak adlandırılır. Diferansiyel gelişim algoritmasında popülasyon büyüklüğü üçten büyük olmak zorundadır. Çünkü yeni neslin oluşturulabilmesi için en az dört kromozom gerekmektedir. Her ne kadar ifadeler genetik algoritma ile benzerlik gösterse de işleyişte büyük farklılıklar bulunmaktadır. Bu farklardan en büyüğü mutasyon işleminin, çaprazlama işleminden önce yapılmasıdır. Mutasyon işleminin uygulanmasında da farklılıklar bulunmaktadır.

Mutasyon işlemi şu şekilde gerçekleşmektedir: Mutasyona uğrayacak kromozom dışında üç farklı kromozom seçilir. Seçilen üç kromozomdan ilk ikisinin farkı alınarak fark kromozomu elde edilir. Fark kromozomu, ölçekleme faktörü (F) adı verilen bir katsayı ile çarpılıp ağırlıklandırılmış fark kromozomuna ulaşılır. Ağırlıklandırılmış fark kromozomu başta seçilen üçüncü kromozom ile toplanarak çaprazlamada kullanılacak kromozom elde edilir.

Çaprazlama işlemi için mutasyon sonucu elde edilen fark kromozomu ile başlangıçta seçilen kromozom kullanılır. Kullanılan iki kromozom ile yeni jenerasyon için aday kromozom üretilir. Aday kromozomun genleri belirlenirken çaprazlama oranı (CR) adı verilen bir parametre kullanılır. Aday kromozomun genleri CR olasılık ile mutasyon sonucu elde edilen fark kromozomundan, 1-CR olasılıkla başlangıçta belirlenen kromozomdan seçilir. Yani 0 ile 1 arasında rasgele sayılar üretilir. Üretilen sayı çaprazlama oranından küçükse gen, mutasyon sonucu elde edilen fark kromozomundan seçilir. Aksi durumda gen, başlangıçta belirlenen kromozomdan seçilir.

Aday kromozomlar bu şekilde belirlendikten sonra uygunluk değerleri hesaplanır. Yeni jenerasyonun parçası olabilmek için mevcut kromozomlar ile ulaşılan yeni kromozomların uygunluk değerleri karşılaştırılır. Uygunluk değeri büyük olan kromozomlar yeni jenerasyonun bir parçası olurlar. Anlatılan işlemler, parametre sınırlarına dikkat edilerek durdurma koşulu sağlanıncaya kadar devam eder. Şekil 3.19'da diferansiyel gelişim algoritmanın akış diyagramı verilmiştir.



Şekil 3.19. Diferansiyel gelişim algoritmasının akış diyagramı

3.2.6.2. Algoritma İçin Parametre Optimizasyonu

Diferansiyel gelişim algoritmasının arama kalitesini etkileyen iki parametresi vardır. Bunlardan ilki ölçekleme faktörüdür (F) ve [0, 2] aralığında seçilebilir. Ölçekleme faktörü, mutasyon işleminde kullanılır. Yapılan çalışmalarda, ölçekleme faktörünün 2'ye yakın değerlerinin daha iyi sonuçlar ürettiği ifade edilmiştir. Bundan dolayı ölçekleme faktörü (F), sırasıyla 1.50, 1.75 ve 2.00 olarak seçilmiştir.

Algoritmanın performansına etki eden diğer bir parametre ise çaprazlama oranıdır. Çaprazlama oranı (CR) [0, 1] arasında seçilebilmektedir. Çaprazlama oranı, yeni jenerasyon için aday kromozom seçiminde kullanılır. Çaprazlama oranının düşük olması yeni oluşacak genin daha az genetik çeşitliliğe sahip olmasına sebebiyet vermektedir. Bundan dolayı çaprazlama oranı sırasıyla 0.75, 0.85 ve 0.95 seçilmiştir.

Bunların dışında kalan parametreler Algoritmalar İçin Parametre Optimizasyonu bölümünde anlatıldığı gibi seçilip parametre optimizasyonu gerçekleştirilmiştir. Her parametre grubu için algoritma üçer defa çalıştırılmıştır. Parametre grupları için ulaşılan uygunluk değerlerinin aritmetik ortalamaları alınarak Çizelge 3.26'da listelenmiştir.

Çizelge 3.26. Diferansiyel gelişim algoritması için parametre optimizasyonu (İterasyon Sayısı = 100, Popülasyon Büyüklüğü = 50)

CR \ F	1.50	1.75	2.00
0.75	29.87812565	**29.24082836**	29.69569077
0.85	29.65271325	30.31355312	30.02325692
0.95	30.57099422	30.88063905	30.84187561

Çizelge 3.26'da görüleceği üzere çaprazlama oranının 0.75 ve ölçekleme faktörünün 1.75 seçilmesinin daha iyi sonuçlar verdiği görülmektedir. Çalışmanın devamında bu parametreler kullanılmıştır.

3.2.6.3. Sonuçlar

Parametre optimizasyonu sonucunda ulaşılan ölçekleme faktörü ve çaprazlama oranına ek olarak iterasyon sayısı 200 ve popülasyon büyüklüğü 100 olarak seçilip

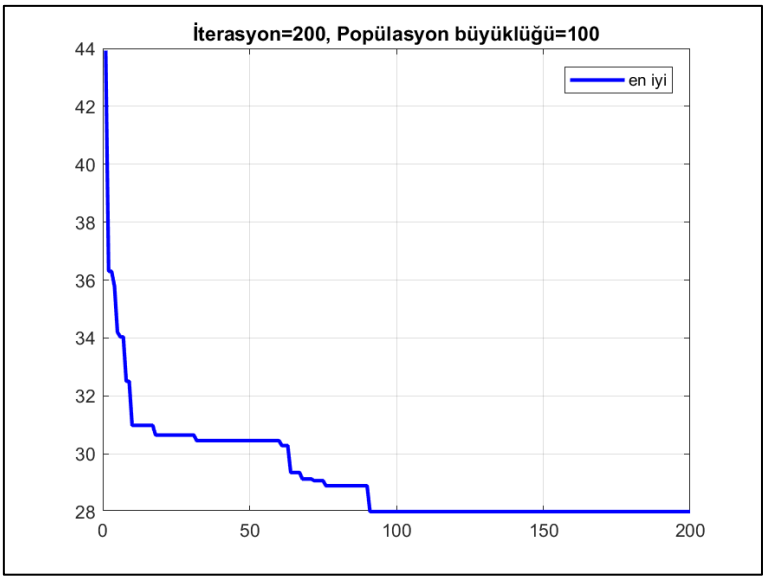
diferansiyel gelişim algoritması 10 defa çalıştırılmıştır. Belirtilen şartlarda gerçekleşen 10 çalışmanın sonucunda elde edilen veriler özetlenerek Çizelge 3.27'de sunulmuştur.

Çizelge 3.27. İterasyon Sayısı = 200, Popülasyon Büyüklüğü = 100, F = 1.75, CR = 0.75 değerleri için 10 kez çalıştırılan diferansiyel gelişim algoritmasının sonuçları

En iyi uygunluk değeri	En kötü uygunluk değeri	Ortalama uygunluk değeri	Aralık	Standart Sapma	Ortalama Geçen süre
28.0015	29.1910	28.6060	1.1895	0.4192	2676.3085

Diferansiyel gelişimi algoritması ile elde edilen en kötü uygunluk değerinin bile bu tez çalışmasının genelinde elde edilen en iyi uygunluk değerine (27.5239) yakın olduğu görülmektedir. Bu durum çizelgedeki standart sapma değerinden de anlaşılmaktadır.

Diferansiyel gelişim algoritması kullanılarak elde edilen en iyi uygunluk değeri 28.0015'dir. En iyi uygunluk değerini veren çalışmanın yakınsama eğrisi Şekil 3.20'de verilmiştir.



Şekil 3.20. Diferansiyel gelişim algoritmasının yakınsama eğrisi

Şekil 3.20'deki yakınsama eğrisinden 100. iterasyondan sonra uygunluk değeri açısından bir gelişme olmadığı görülebilmektedir. Bu noktadan sonra diferansiyel gelişim algoritmasının lokal minimum noktaya takıldığı söylenebilir.

Çizelge 3.28'de diferansiyel gelişim algoritmasının ulaştığı en iyi uygunluk değeri olan 28.0015 değerini veren çözüm kümesi verilmiştir. Çizelge 3.28 veriler Q ve R

matrislerini oluşturmaktadır. Oluşturulan matrisler Ricatti denkleminde kullanılarak denklemin kökü olan K matrisi hesaplanmıştır. K matrisi 3-DOF Hover sistemine uygulandığında sistemin verdiği cevap Şekil 3.21'de görülmektedir.

Çizelge 3.28. Diferansiyel gelişim algoritmasının 10 kez çalıştırılması sonucunda ulaşılan en iyi çözüm

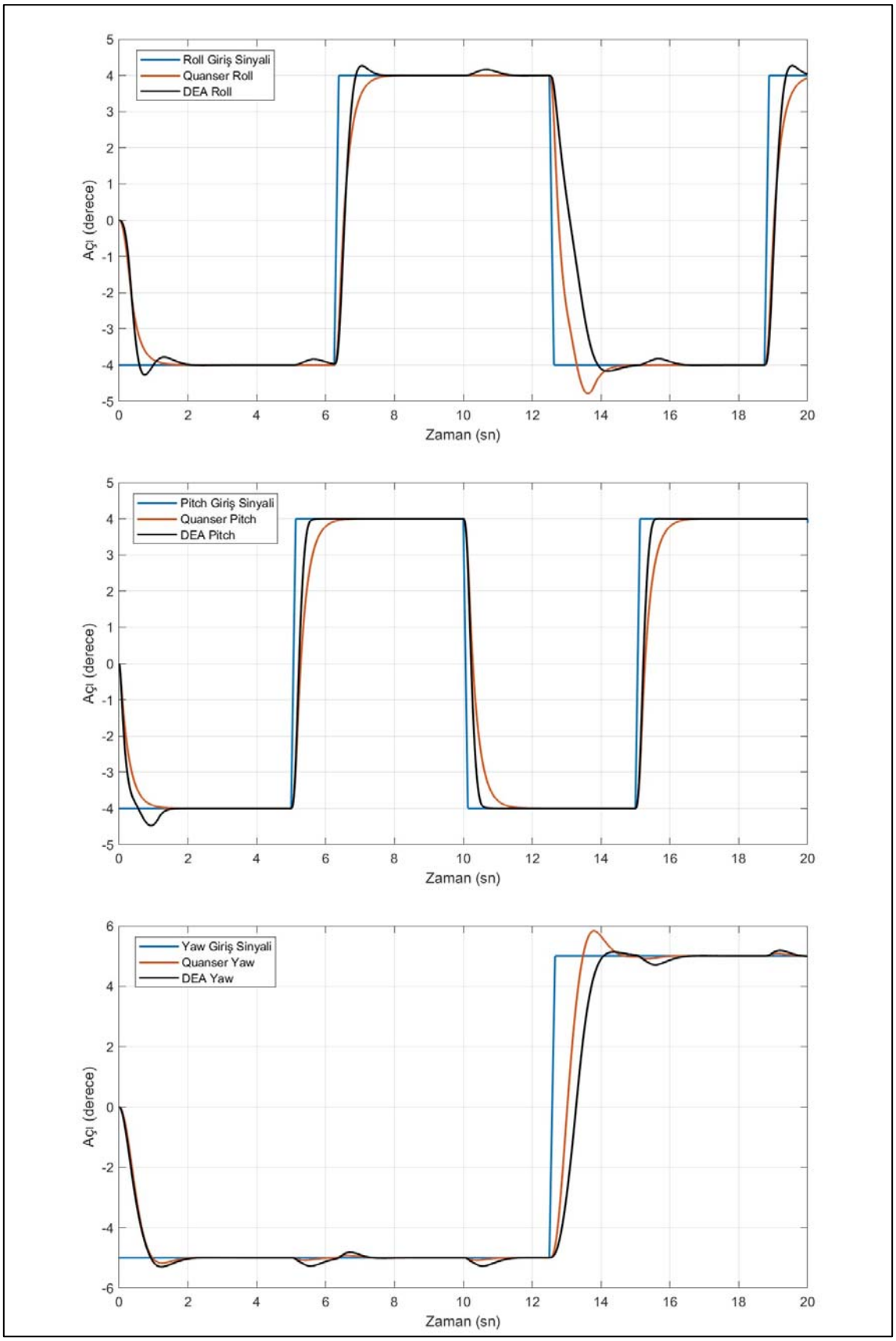
q_{11}	q_{22}	q_{33}	q_{44}	q_{55}	q_{66}	r_{11}	r_{22}	r_{33}	r_{44}
700	500	69.9276	20	8.1113	0.01	0.01	0.01	0.01	0.1113

Şekil 3.21'de mavi, kırmızı ve siyah olmak üzere üç farklı çizgi ile belirtilen sinyaller bulunmaktadır. Mavi çizgi ile gösterilen sinyaller giriş sinyaline aittir. Kırmızı çizgi ile gösterilen sinyaller, Quanser firmasının önerdiği kontrolcü parametrelerinin kullanılmasıyla elde edilen çıkış sinyallerine aittir. Son olarak siyah çizgi ile gösterilen sinyaller, diferansiyel gelişim algoritması kullanılarak hesaplanan kontrolcü parametrelerinin, sistemde uygulanmasıyla elde edilen sistem cevaplarına ait çıkış sinyalleridir. Çizelge 3.29'da Şekil 3.21'de verilen çıkış sinyallerinin yorumlanabilmesi ve karşılaştırılabilmesi için bazı bilgiler yer almaktadır.

Çizelge 3.29. Şekil 3.21'deki Quanser çıkış sinyalleri ile diferansiyel gelişim algoritması ile hesaplanan çıkış sinyallerinin karşılaştırılması

	Oturma Zamanı (sn)			Yükselme Zamanı (sn)			Üst Aşım (deg)		
	Roll	Pitch	Yaw	Roll	Pitch	Yaw	Roll	Pitch	Yaw
Qua	19.7107	16.0741	14.3349	0.5042	0.5440	0.3475	0.7890	0.0003	0.8323
DEA	19.6945	15.4931	15.8278	0.2399	0.2084	0.5195	0.2731	0.4739	0.3043
Fark	0.0162	0.581	-1.4929	0.2643	0.3356	-0.172	0.5159	-0.4736	0.528

Çizelge 3.29'da verilen diferansiyel gelişim algoritması kullanılarak ulaşılan parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları ile Quanser firmasının parametrelerinin sistemde uygulanmasıyla elde edilen sistem cevapları karşılaştırılmıştır. İlk satırda Quanser firmasının önerdiği parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları hakkında bilgiler yer almaktadır. İkinci satırda diferansiyel gelişim algoritmasının kullanılmasıyla elde edilen parametrelerin sistemde uygulanmasıyla ulaşılan sistem cevabı hakkında bilgiler yer almaktadır. Son satırda ise bahsedilen bu bilgilerin farkları yer almaktadır.



Şekil 3.21. Giriş sinyalleri, Quanser parametreleri ile hesaplanan çıkış sinyalleri ve diferansiyel gelişim algoritması ile hesaplanan en iyi çıkış sinyalleri

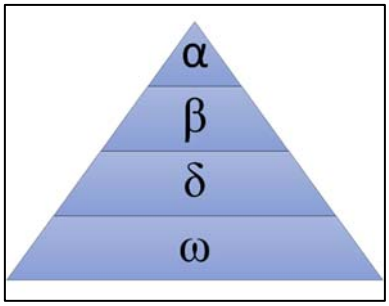
Diferansiyel gelişim algoritmasının başarısız olduğu sütunlar fark satırında negatif değere sahip olan sütunlardır. Çizelge 3.29'da verilerin çoğunluğunun iyileştiği görülmektedir. Üst aşım değerlerine bakıldığında sadece pitch açısının değerinin kötüleştiği anlaşılmaktadır. Sistemin roll ve yaw cevabındaki üst aşımlar azaltılabilmiştir. Oturma zamanlarında gerçekleşen en kötü sonuç beklenildiği gibi yaw cevabında yaşanmıştır. Yaw açısının oturma zamanında yaşanan kötüleşmeye karşı pitch ve roll oturma zamanları azaltılabilmiştir. Yükselme zamanlarına bakıldığında yaw açısında bozulma görülmektedir. Ancak roll ve pitch açılarının yükselme zamanları iyileşmiştir.

3.2.7. Gri Kurt Optimizasyon Algoritması

Gri kurt optimizasyon (Grey Wolf Optimization - GWO) algoritması, 2014 yılında Seyedali Mirjalili ve Andrew Lewis tarafından önerilmiştir. Araştırmacılar, bu algoritmaya adından da anlaşılacağı üzere gri kurtların doğadaki davranışlarından yola çıkarak ulaşmışlardır [30].

Gri kurtlar doğada sürü halinde yaşadıklarından bir hiyerarşik düzene göre hareket etmektedirler. Kurtlar, araştırmacılar tarafından alfa, beta, delta ve omega olmak üzere dört gruba ayrılmıştır. Kurt sürüsündeki her bir grubun, görevi vardır. Bu gruplardan bir tanesi dahi yok olsa bütün sürü iç karışıklığa sürüklenir. Bahsedilen hiyerarşik ilişki kurtların birçok sosyal davranışına etki etmektedir. Örneğin avlanma yöntemlerinde bu hiyerarşik düzenin çok büyük önemi vardır. Aynı şekilde yavru kurtların ve yaşlı kurtların bakımı, sürü güvenliği gibi pek çok konuda kurtlar hiyerarşik düzene göre hareket ederler.

Şekil 3.22'den anlaşılacağı üzere sürünün lideri alfa grubu kurtlardır. Bu grup biri dişi, biri erkek kurt olmak üzere 2 kurttan oluşur. Sürünün avlanmasından konaklayacağı yere kadar her konuda karar alma yetkisine sahiptir. Kısaca sürünün karar alma organıdır. Alfa kurtların biyolojik açıdan diğer kurtlardan iri olmasına gerek yoktur. Önemli olan aldığı kararları uygulatabilmeleridir.



Şekil 3.22. Gri kurtlar arasındaki hiyerarşik ilişki [30]

Beta grubu kurt, alfa kurtların karar almadan önce görüşüne başvurduğu kurtlardır. Alfa kurdun aldığı kararlar beta kurtların yardımıyla uygulanabilir. Bunun yanında alfa kurtların liderliği devam ettiremeyeceği anlaşıldığında beta kurt liderlik için en güçlü adaydır.

En düşük rütbeli gri kurt omegadır. Omega kurtları her zaman baskın olan diğer bütün kurtlara boyun eğmek zorundadır. Onlar avı yemek için izin verilen son kurtlardır. Omega sürü içinde önemli olmayan bir birey gibi görünebilir. Ancak omeganın kaybı durumunda sürünün tamamının bir iç savaşa maruz kaldığı ve sorunlar yaşadığı görülür. Bu, tüm sürünün baskınlık yapısını korumaya yardımcı olur. Bazı durumlarda omega sürüdeki bebek bakıcılığı rolünü de üstlenir.

Eğer bir kurt alfa, beta veya omega değilse, delta olarak isimlendirilir. Delta kurtlar alfa ve betalara boyun eğmek zorundadır, ancak omegaya baskındırlar. İzciler, nöbetçiler, yaşlılar, avcılar ve bekçiler bu kategoriye girer. İzciler, bölge sınırlarını izlemek ve herhangi bir tehlike durumunda sürüyü uyarmaktan sorumludurlar. Nöbetçiler sürünün güvenliğini sağlar ve garanti altına alırlar. Yaşlılar, daha önce alfa veya beta olan deneyimli kurtlardır. Avcılar av avlarken ve sürü için yiyecek sağlarken alfa ve betalara yardım ederler. Son olarak, bakıcılar sürüdeki zayıf, hasta ve yaralı kurtların korunmasından sorumludurlar.

3.2.7.1. Algoritmanın İşleyişi

Doğadaki gri kurtların avlanmaları dört bölümden meydana gelmektedir. Bunlar; av arama, av yorulana kadar takip, avın etrafını sarma ve saldırı bölümleridir. Denklem (3.40) ve Denklem (3.41)'de avın etrafının kurtlar tarafından çevrelenmesi matematiksel olarak ifade edilmektedir.

$$\vec{D} = \left|\vec{C} \times \vec{X}_p(t) - \vec{X}(t)\right| \tag{3.40}$$

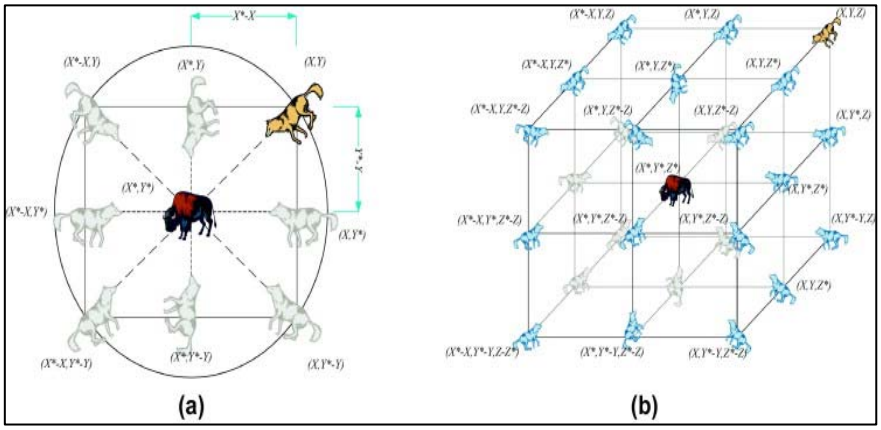
$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \times \vec{D} \tag{3.41}$$

Burada t, geçerli iterasyonu ifade etmektedir. $\vec{A}$ ile $\vec{C}$, katsayı vektörlerini temsil etmektedir. $\vec{A}$ ile $\vec{C}$ katsayı vektörleri Denklem (3.42) ve Denklem (3.43)'te formüle edildiği gibi hesaplanır. $\vec{X}_p$, avın konum vektörüdür. $\vec{X}$ ise bir gri kurdun pozisyon vektörüdür.

$$\vec{A} = 2 \times \vec{a} \times \vec{r}_1 - \vec{a} \tag{3.42}$$

$$\vec{C} = 2 \times \vec{r}_2 \tag{3.43}$$

Denklem (3.42)'deki $\vec{a}$ değeri iterasyonlar boyunca 2'den 0'a doğru azalır ve $\vec{r}_1$ ile $\vec{r}_2$ [0, 1] aralığında rastgele vektörlerdir. Şekil 3.23'te gri kurtların iki ve üç boyutta avların etrafını sarmalarını gösteren bir görsel verilmiştir.



Şekil 3.23. Gri kurtların, 2 ve 3 boyutlu uzayda, avın etrafını sarmaları [30]

Alfa sürüdeki en iyi uygunluk değerine sahip olan kurttur. Alfayı beta ve delta takip etmektedir. Bu üç en iyi çözüme göre diğer kurtların pozisyonları güncellenmektedir. Pozisyon güncelleme işlemi Denklem (3.44) ile Denklem (3.50) arasında verilen denklemler kullanılarak gerçekleştirilir.

$$\vec{D}_\alpha = \left|\vec{C}_1 \times \vec{X}_\alpha - \vec{X}\right| \tag{3.44}$$

$$\vec{D}_\beta = \left|\vec{C}_2 \times \vec{X}_\beta - \vec{X}\right| \tag{3.45}$$

$$\vec{D}_\delta = \left|\vec{C}_3 \times \vec{X}_\delta - \vec{X}\right| \tag{3.46}$$

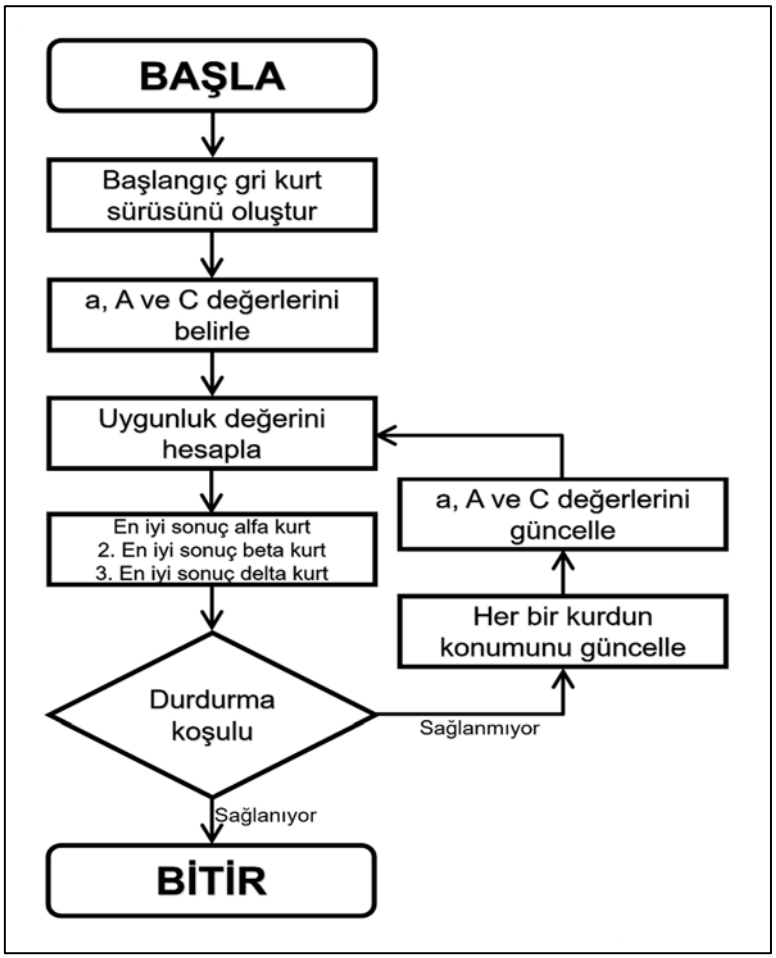
$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \times \vec{D}_\alpha \tag{3.47}$$

$$\vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \times \vec{D}_\beta \tag{3.48}$$

$$\vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \times \vec{D}_\delta \tag{3.49}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{3.50}$$

Şekil 3.24'te gri kurt optimizasyon algoritmasının akış diyagramı verilmiştir.



Şekil 3.24. Gri kurt optimizasyon algoritmasının akış diyagramı

3.2.7.2. Sonuçlar

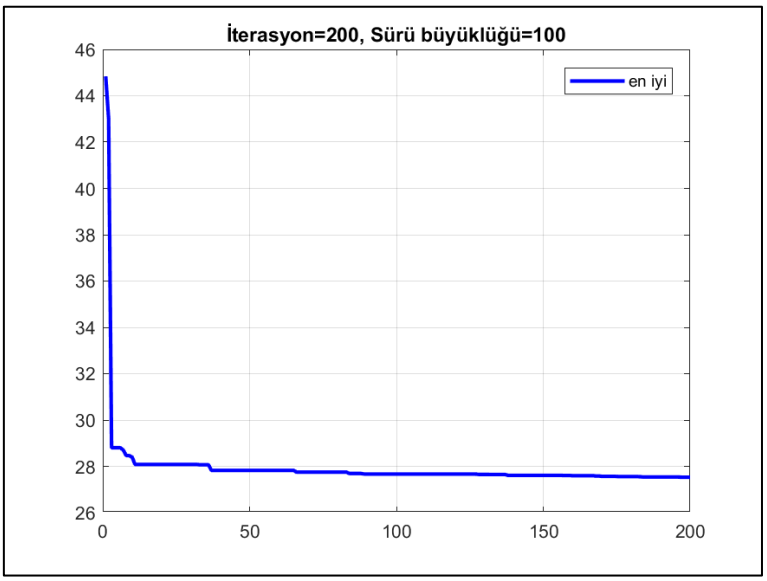
Gri kurt optimizasyon algoritmasında algoritmanın arama kalitesini etkileyen birtakım parametreler vardır. Ancak bunlar algoritma içerisinde rastgele sayılar kullanılarak hesaplatılmaktadır. Dolayısıyla herhangi bir parametre optimizasyonu işlemi yapılmamıştır. Algoritma 200 iterasyon ve 100 popülasyon ile 10 defa çalıştırılıp elde edilen sonuçlar kaydedilmiştir. Ulaşılan sonuçların özeti Çizelge 3.30'da paylaşılmaktadır.

Çizelge 3.30. İterasyon Sayısı = 200, Popülasyon Büyüklüğü = 100 değerleri için 10 kez çalıştırılan gri kurt optimizasyon algoritmasının sonuçları

En iyi uygunluk değeri	En kötü uygunluk değeri	Ortalama uygunluk değeri	Aralık	Standart Sapma	Ortalama Geçen süre
27.5295	29.7794	28.3077	2.2498	0.7456	3058.0901

Gri kurt optimizasyon algoritması ile elde edilen en iyi uygunluk değeri 27.5295'tir. Bu değer aynı zamanda bu tez kapsamında kullanılan bütün algoritmalar içerisinde 100 popülasyon büyüklüğü ve 200 iterasyon kullanılarak ulaşılan en iyi değerdir.

Şekil 3.25'te gri kurt optimizasyon algoritması kullanılarak ulaşılan en iyi uygunluk değerine ait çalışmanın yakınsama eğrisi verilmiştir. Şekil 3.25 gri kurt optimizasyon algoritmasının oldukça sağlıklı çalıştığı söylenebilir.

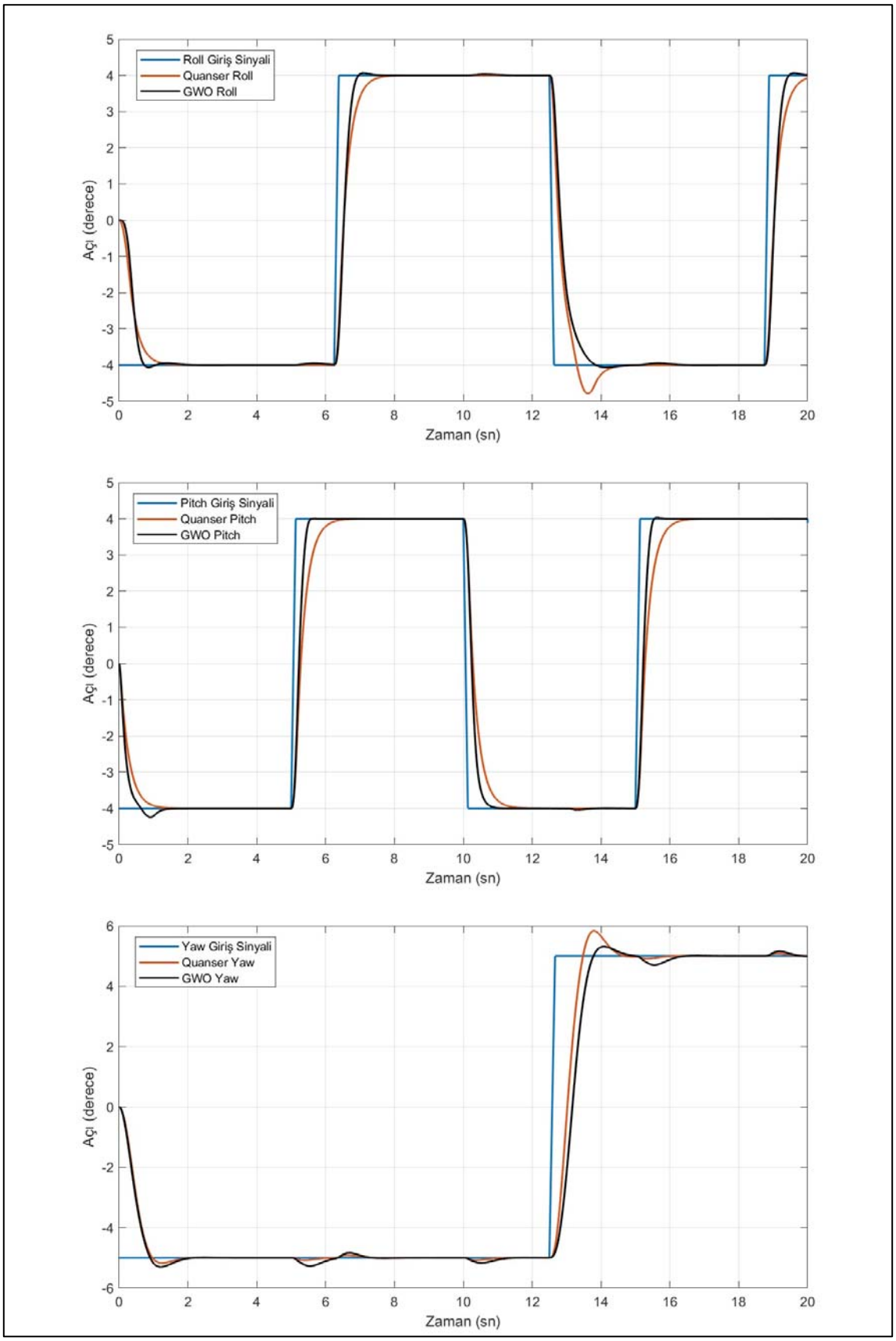


Şekil 3.25. Gri kurt optimizasyon algoritmasının yakınsama eğrisi

Çizelge 3.31'de gri kurt optimizasyon algoritmasının ulaştığı en iyi uygunluk değeri olan 27.5295 değerini veren çözüm kümesi verilmiştir. Çizelge 3.31'deki çözüm kümesi, Q ve R matrislerini oluşturmaktadır. Oluşturulan matrisler Ricatti Denkleminde kullanılarak denklemin kökü olan K matrisi hesaplanmıştır. K matrisi 3-DOF Hover sistemine uygulandığında sistemin verdiği cevap Şekil 3.26'da görülmektedir.

Çizelge 3.31. Gri kurt optimizasyon algoritmasının 10 kez çalıştırılması sonucunda ulaşılan en iyi çözüm

q_{11}	q_{22}	q_{33}	q_{44}	q_{55}	q_{66}	r_{11}	r_{22}	r_{33}	r_{44}
700	499.2160	142.3161	0.0083	8.5509	1.7155	0.01	0.0148	0.0143	0.0571



Şekil 3.26. Giriş sinyalleri, Quanser parametreleri ile hesaplanan çıkış sinyalleri ve gri kurt optimizasyon algoritması ile hesaplanan en iyi çıkış sinyalleri

Şekil 3.26'dan da görüldüğü gibi mavi, kırmızı ve siyah olmak üzere üç farklı çizgi ile belirtilen sinyaller bulunmaktadır. Mavi çizgi ile gösterilen sinyaller giriş sinyaline aittir. Kırmızı çizgi ile gösterilen sinyaller, Quanser firmasının önerdiği kontrolcü parametrelerinin kullanılmasıyla elde edilen çıkış sinyallerine aittir. Son olarak siyah çizgi ile gösterilen sinyaller, gri kurt optimizasyon algoritması kullanılarak hesaplanan kontrolcü parametrelerinin, sistemde uygulanmasıyla elde edilen sistem cevaplarına ait çıkış sinyalleridir.

Çizelge 3.32'de verilen gri kurt optimizasyon algoritması kullanılarak ulaşılan parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları ile Quanser firmasının parametrelerinin sistemde uygulanmasıyla elde edilen sistem cevapları karşılaştırılmıştır. İlk satırda Quanser firmasının önerdiği parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları hakkında bilgiler yer almaktadır. İkinci satırda gri kurt optimizasyon algoritmasının kullanılmasıyla elde edilen parametrelerin sistemde uygulanmasıyla ulaşılan sistem cevabı hakkında bilgiler yer almaktadır. Son satırda ise bahsedilen bu bilgilerin farkları yer almaktadır. Gri kurt optimizasyon algoritmasının başarısız olduğu sütunlar fark satırında negatif değere sahip olan sütunlardır.

Çizelge 3.32. Şekil 3.26'daki Quanser çıkış sinyalleri ile gri kurt optimizasyon algoritması ile hesaplanan çıkış sinyallerinin karşılaştırılması

	Oturma Zamanı (sn)			Yükselme Zamanı (sn)			Üst Aşım (deg)		
	Roll	Pitch	Yaw	Roll	Pitch	Yaw	Roll	Pitch	Yaw
Qua	19.7107	16.0741	14.3349	0.5042	0.5440	0.3475	0.7890	0.0003	0.8323
GWO	19.3775	15.4815	15.8019	0.2622	0.2027	0.4326	0.0629	0.2418	0.3078
Fark	0.3332	0.5926	-1.467	0.242	0.3413	-0.0851	0.7261	-0.2415	0.5245

Şekil 3.26 ve Çizelge 3.32 incelendiğinde gri kurt optimizasyon algoritması kullanılarak elde edilen sistem cevaplarının, Quanser firmasının varsayılan olarak önerdiği parametrelerin sisteme uygulanmasıyla elde edilen sistem cevaplarından çoğu konu da daha iyi olduğu görülmektedir.

3.2.8. Hiyerarşik Benzetilmiş Tavlama - Gri Kurt Optimizasyon Algoritması

Bu çalışmada, yedi farklı yapay zekâ optimizasyon algoritması tek bir probleme uygulanmış ve çeşitli sonuçlar elde edilmiştir. Elde edilen sonuçlar, gri kurt optimizasyon algoritmasının hızlı bir şekilde yakınsağını ve problem için en iyi çözümleri ürettiğini

göstermiştir. Sonuçlardan, ayrıca, benzetilmiş tavlama algoritmasının, lokal minimumlara takılmadan, yakınsama özelliğinin tüm iterasyonlar boyunca devam ettiği gözlenmiştir. Bu gözlemlerden hareketle gri kurt optimizasyon algoritmasının hızlı yakınsama özelliği ile benzetilmiş tavlama algoritmasının lokal minimumları aşma özelliğinden faydalanılmış ve hiyerarşik bir algoritma (Hiyerarşik SAA-GWO) önerilmiştir.

Hiyerarşik SAA-GWO algoritması şu şekilde çalışmaktadır: Başlangıçta 0 ile 1 arasında bir sayı belirlenir. Belirlenen sayı maksimum iterasyon sayısı ile çarpılarak gri kurt optimizasyon algoritmasının kaç iterasyon çalışacağı tespit edilir. Gri kurt optimizasyon algoritması problem üzerinde tespit edilen iterasyon sayınca çalıştırılır. Elde edilen en iyi çözüm benzetilmiş tavlama algoritmasının başlangıç çözümü olarak alınır. Maksimum iterasyon sayısı gri kurt optimizasyon algoritması için belirlenen iterasyon sayısından çıkarılır. Hesaplanan iterasyon sayısı kadar benzetilmiş tavlama algoritması çalıştırılır.

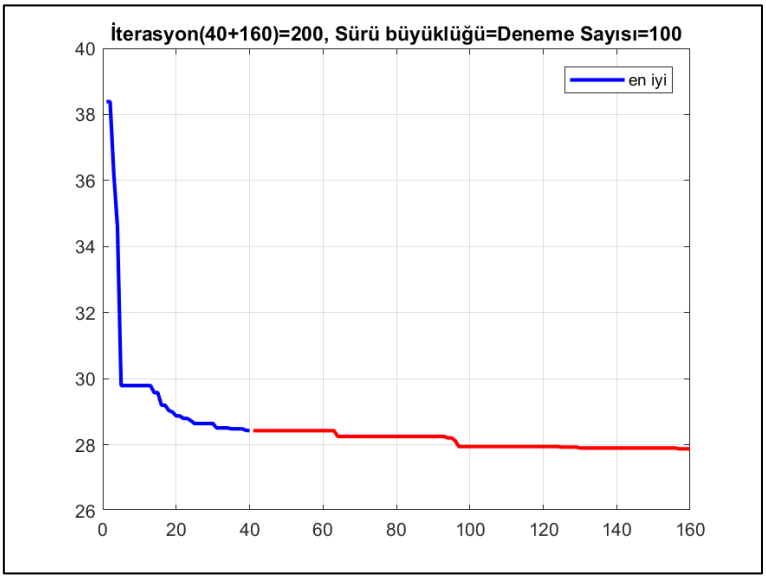
3.2.8.1. Sonuçlar

Hiyerarşik SAA-GWO algoritması için gri kurt optimizasyon algoritmasının 0.2 oranında çalıştırılmasına karar verilmiştir. Dolayısıyla 40 iterasyon boyunca gri kurt optimizasyon algoritması, 160 iterasyon boyunca benzetilmiş tavlama algoritması çalıştırılmıştır. Diğer algoritmalar ile adil bir karşılaştırma olması adına gri kurt optimizasyon algoritmasının popülasyon büyüklüğü ve benzetilmiş tavlama algoritmasının deneme sayısı 100 olarak alınmıştır. Belirtilen şartlarda Hiyerarşik SAA-GWO yöntemi 10 defa çalıştırılmıştır. Elde edilen sonuçlar özetlenerek Çizelge 3.33'te listelenmiştir.

Çizelge 3.33. İterasyon Sayısı = (40+160), Pürü Büyüklüğü = Deneme Sayısı = 100 değerleri için 10 kez çalıştırılan Hiyerarşik SAA-GWO yönteminin sonuçları

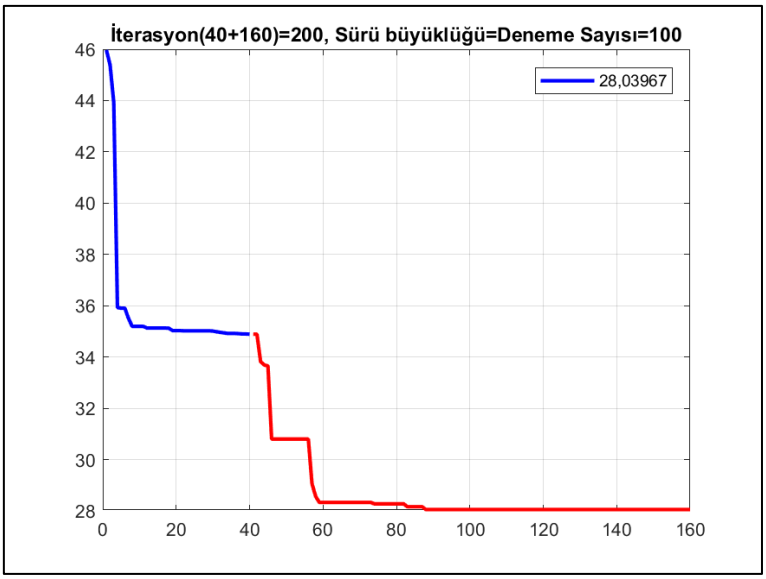
En iyi uygunluk değeri	En kötü uygunluk değeri	Ortalama uygunluk değeri	Aralık	Standart Sapma	Ortalama Geçen süre
27.8759	29.9020	28.6594	2.0261	0.6538	3368.5950

Çizelge 3.33'te hiyerarşik çalıştırma yöntemi kullanılarak ulaşılan en iyi uygunluk değerinin 27.8759 olduğu görülmektedir. Bu değeri veren çalışmanın yakınsama eğrisi Şekil 3.27'de verilmiştir.



Şekil 3.27. Hiyerarşik SAA-GWO yakınsama eğrisi

Önerilen Hiyerarşik SAA-GWO algoritmasına ait başka bir yakınsama eğrisi Şekil 3.28'de verilmiştir. Hızlı yakınsama işlemi Şekil 3.27'de gri kurt optimizasyon algoritması tarafından sağlanırken Şekil 3.28'de benzetilmiş tavlama algoritması tarafından sağlanmıştır. Bu iki çalıştırma için elde edilen uygunluk değerleri birbirine yakındır.



Şekil 3.28. Hiyerarşik SAA-GWO yönteminde benzetilmiş tavlama algoritmasının etkisini gösteren yakınsama eğrisi

Çizelge 3.33'te hiyerarşik çalışma yönteminin ulaştığı en iyi uygunluk değerinin 27.8759 olduğu görülmektedir. Bu uygunluk değerini veren çözüm kümesi Çizelge 3.34'te verilmiştir.

Çizelge 3.34. Hiyerarşik SAA-GWO yönteminin 10 kez çalıştırılması sonucunda ulaşılan en iyi çözüm

q_{11}	q_{22}	q_{33}	q_{44}	q_{55}	q_{66}	r_{11}	r_{22}	r_{33}	r_{44}
563.8831	488.5267	500.0000	0.0408	10.5214	0.0132	0.0103	0.0124	0.0535	0.0100

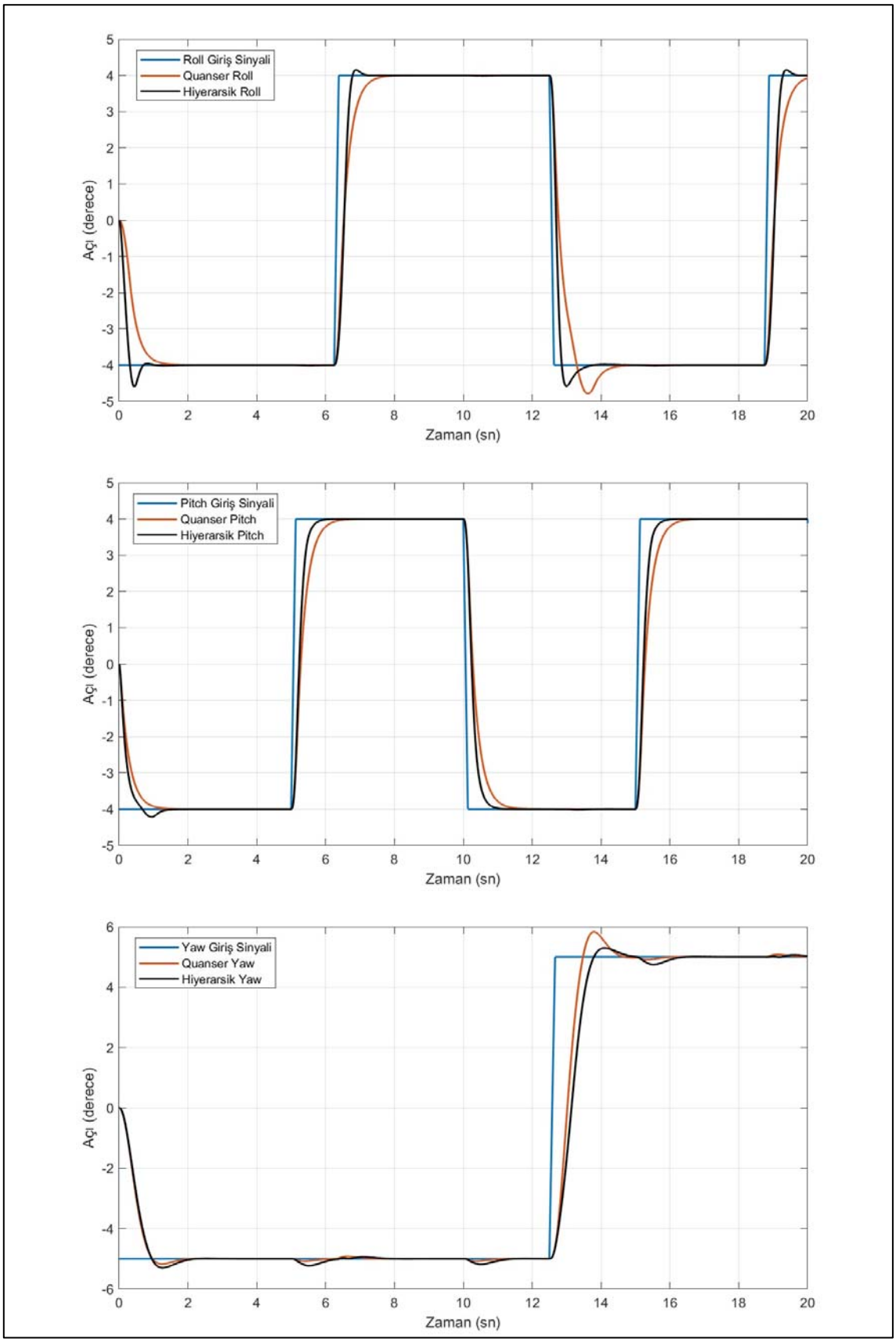
Çizelge 3.34'te verilen çözüm kümesi Q ve R matrislerinin diegonal elemanlarını oluşturmaktadır. Oluşturulan matrisler Ricatti Denkleminde kullanılarak denklemin kökü olan K matrisi hesaplanmıştır. K matrisi 3-DOF Hover sistemine uygulandığında sistemin verdiği cevap Şekil 3.29'da görülmektedir.

Şekil 3.29'dan da görüldüğü gibi mavi, kırmızı ve siyah olmak üzere üç farklı çizgi ile belirtilen sinyaller bulunmaktadır. Mavi çizgi ile gösterilen sinyaller giriş sinyaline aittir. Kırmızı çizgi ile gösterilen sinyaller, Quanser firmasının önerdiği kontrolcü parametrelerinin kullanılmasıyla elde edilen çıkış sinyallerine aittir. Son olarak siyah çizgi ile gösterilen sinyaller, Hiyerarşik SAA-GWO yöntemi kullanılarak hesaplanan kontrolcü parametrelerinin, sistemde uygulanmasıyla elde edilen sistem cevaplarına ait çıkış sinyalleridir.

Çizelge 3.35'te verilen Hiyerarşik SAA-GWO yöntemi kullanılarak ulaşılan parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları ile Quanser firmasının parametrelerinin sistemde uygulanmasıyla elde edilen sistem cevapları karşılaştırılmıştır. İlk satırda Quanser firmasının önerdiği parametrelerin sistemde uygulanmasıyla elde edilen sistem cevapları hakkında bilgiler yer almaktadır. İkinci satırda Hiyerarşik SAA-GWO yönteminin kullanılmasıyla elde edilen parametrelerin sistemde uygulanmasıyla ulaşılan sistem cevabı hakkında bilgiler yer almaktadır. Son satırda ise bahsedilen bu bilgilerin farkları yer almaktadır.

Çizelge 3.35. Şekil 3.29'daki Quanser çıkış sinyalleri ile Hiyerarşik SAA-GWO yöntemi kullanılarak ulaşılan çıkış sinyallerinin karşılaştırılması

	Oturma Zamanı (sn)			Yükselme Zamanı (sn)			Üst Aşım (deg)		
	Roll	Pitch	Yaw	Roll	Pitch	Yaw	Roll	Pitch	Yaw
Qua	19.7107	16.0741	14.3349	0.5042	0.5440	0.3475	0.7890	0.0003	0.8323
Hiy	19.2470	15.6549	15.8021	0.1678	0.2826	0.4666	0.6000	0.2175	0.3023
Fark	0.4637	0.4192	-1.4672	0.3364	0.2614	-0.1191	0.1890	-0.2172	0.5300



Şekil 3.29. Giriş sinyalleri, Quanser parametreleri ile hesaplanan çıkış sinyalleri ve Hiyerarşik SAA-GWO yöntemi kullanılarak elde edilen en iyi çıkış sinyalleri

Hiyerarşik SAA-GWO algoritmasının başarısız olduğu sütunlar fark satırında negatif değere sahip olan sütunlardır. Çizelge 3.35 ve Şekil 3.29 incelendiğinde hiyerarşik çalışma yönteminin bu tez çalışmasında üzerinde çalışılan problem için oldukça uygun çözümler ürettiği görülmektedir.

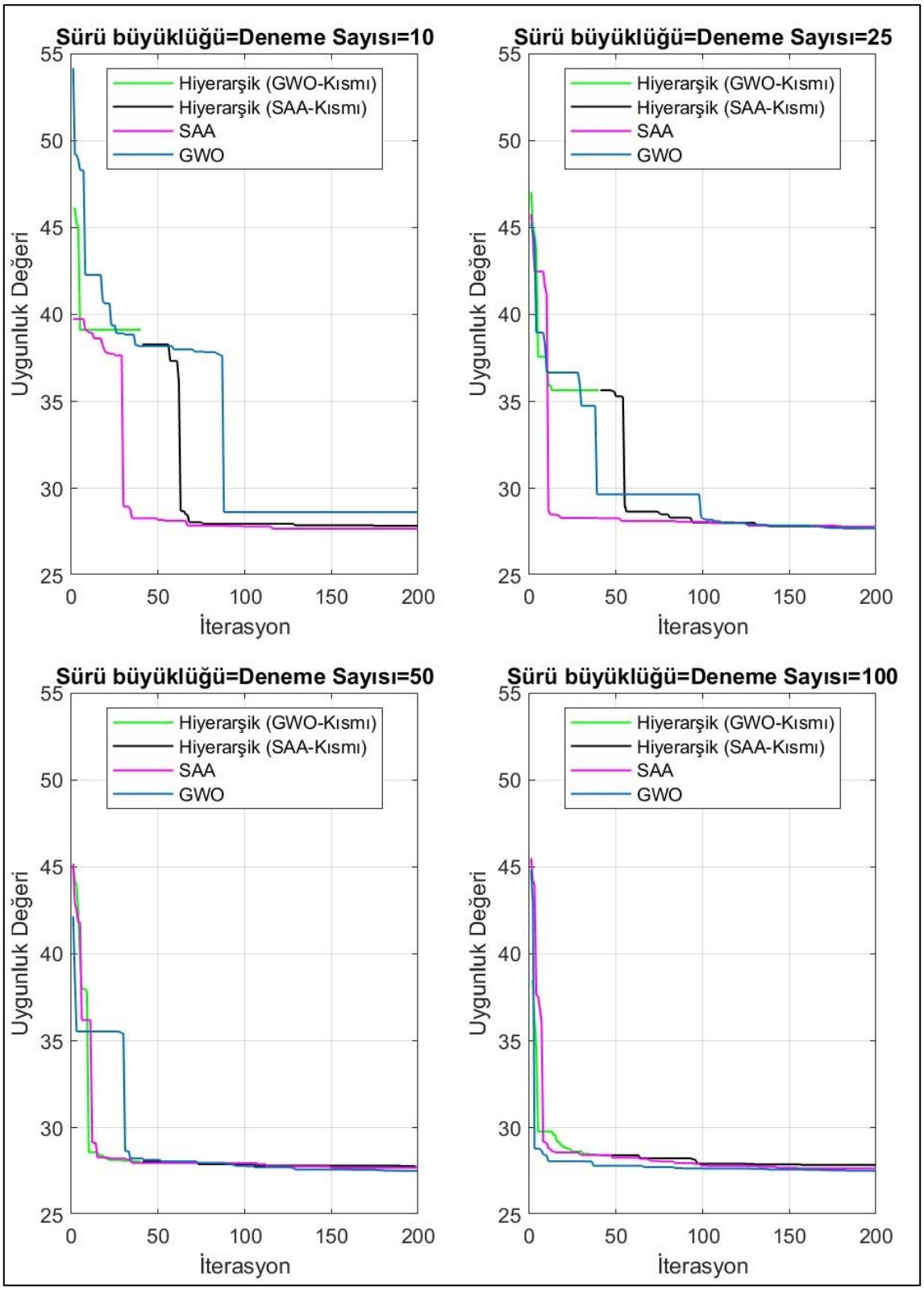
3.2.8.2. Düşük deneme sayıları için hiyerarşik çalışmanın sonuçları

Hiyerarşik SAA-GWO yöntemi 10, 25, 50 ve 100 deneme sayısı (popülasyon büyüklüğü) ile çalıştırılmıştır. Elde edilen sonuçlar Çizelge 3.36'da özetlenerek listelenmiştir.

Çizelge 3.36. Hiyerarşik SAA-GWO yöntemi ile 10'ar çalıştırma için elde edilen sonuçlar

	Popülasyon Büyüklüğü			
	10	**20**	**50**	**100**
Ortalama uygunluk değeri	33.7497	30.1291	30.4335	29.5504
Standart sapma	2.7465	1.6421	2.2243	1.3141
En iyi uygunluk değeri	28.2986	27.8592	27.8558	27.8650
En kötü uygunluk değeri	37.2843	33.7023	35.2462	31.9778
Aralık	8.9857	5.8431	7.3903	4.1128
Geçen ortalama zaman (sn)	658.7605	1512.55	3155.9721	5694.5955

Deneme sayıları geliştirilen hiyerarşik yöntem ile eşit tutularak benzetilmiş tavlama algoritması ve gri kurt optimizasyon algoritması 10'ar defa çalıştırılmıştır. Bu çalışmalar sonucunda elde edilen en iyi uygunluk değerlerine ait yakınsama eğrileri Şekil 3.30'da verilmiştir. Özellikle deneme sayısının 10 ve 25 olduğu durumlarda hiyerarşik çalışma yönteminin faydaları görülmektedir.



Şekil 3.30. Farklı deneme sayıları için Hiyerarşik SAA-GWO yönteminin yakınsama eğrileri

4. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

Bu tez çalışmasında 3-DOF Hover deney setinin simülasyon ortamında kontrolü gerçekleştirilmiştir. Sistemin kontrol edilebilmesi için hem doğrusal hem de doğrusal olmayan sistemlerde kullanılabilen LQR kontrol yöntemi kullanılmıştır. Kullanılan kontrol yönteminin, deneme yanılma yöntemleriyle hesaplanması zor olan, kontrolcü parametreleri yapay zekâ optimizasyon algoritmaları kullanılarak belirlenmiştir.

LQR kontrolcünün en uygun parametre değerlerini bulmak için kullanılan yapay zekâ optimizasyon algoritmalarının tamamında iterasyon sayısı 200 ve popülasyon büyüklüğü (deneme sayısı) 100 olarak alınmıştır. Bütün algoritmalar 10'ar kez çalıştırılmıştır. Her bir algoritma için elde edilen sonuçlar algoritmaların anlatıldığı bölümlerde verilmiş ve yorumlanmıştır. Bu bölümde ise daha düşük popülasyon büyüklüğü yani deneme sayısıı için yapay zekâ optimizasyon algoritmalarının ulaştıkları sonuçlar karşılaştırılmıştır. Daha önceki sonuçlar (popülasyon büyüklüğü 100 için) ile birlikte elde edilen sonuçlar Çizelge 4.1'de listelenmiştir. Listelenen sonuçlar 200 iterasyon sayısı ve 10'ar çalıştırma için elde edilmiştir.

Ulaşılan en iyi değer baz alınarak elde edilen yakınsama eğrileri ise Şekil 4.1'de verilmiştir.

Çizelge 4.1 incelendiğinde yapay zekâ optimizasyon algoritmalarının farklı alanlarda birbirlerine göre üstünlükleri olduğu görülebilmektedir. Ortalama uygunluk değerlerine göre, 10 ve 50 popülasyon büyüklüğü için diferansiyel gelişim algoritmasının en iyi sonucu elde ettiği görülmektedir. Popülasyon büyüklüğü 25 olduğunda en iyi sonucu yapay arı algoritması ve 100 olduğunda ise en iyi sonucu gri kurt optimizasyon algoritması elde etmiştir.

En kötü uygunluk değerleri açısından, 10 popülasyon büyüklüğü için, genetik algoritmanın ulaştığı en iyi uygunluk değeri Çizelge 4.1'de 45.3840 olarak verilmiştir. Halbuki aynı çizelgeden diferansiyel gelişim algoritmasının 10 popülasyon için ulaştığı en kötü uygunluk değerinin 34.4117 olduğu görülmektedir. Bu alanda, tüm popülasyon büyüklükleri için en iyi sonucu veren yapay zekâ optimizasyon algoritması diferansiyel gelişim algoritması olmuştur.

Yapay zekâ optimizasyon algoritmalarının tamamı aynı özelliklere sahip bilgisayar üzerinde çalıştırılmıştır. Çalışma süreleri incelendiğinde en çarpıcı sonuç

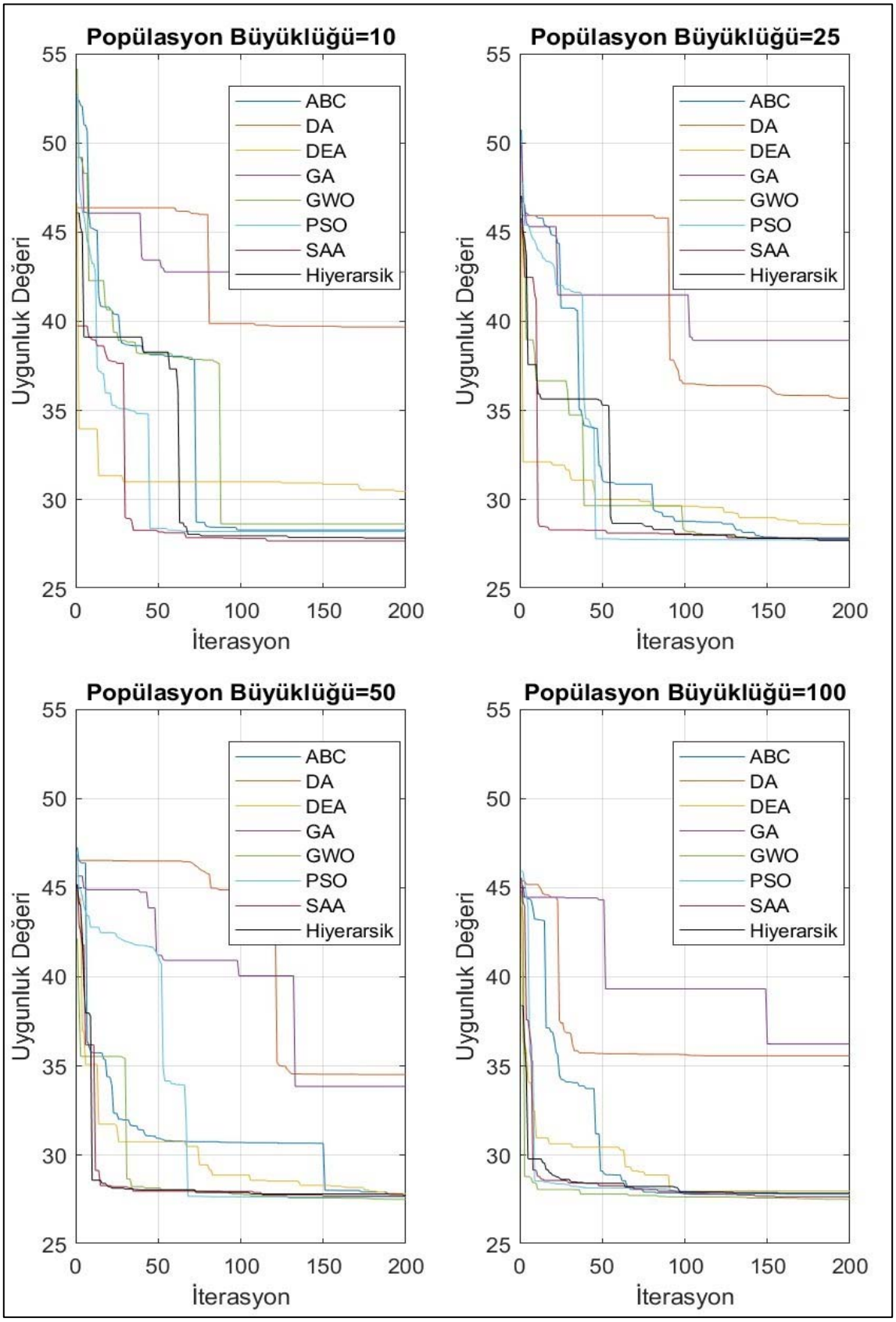
yapay arı kolonisi algoritmasına aittir. Yapay arı kolonisi algoritması bütün popülasyon büyüklükleri için diğer algoritmalardan yaklaşık iki kat daha yavaş çalışmıştır.

Çizelge 4.1. Bütün algoritmaların farklı popülasyon büyüklüklerine göre elde edilen sonuçları (İterasyon Sayısı = 200)

		Popülasyon Büyüklüğü			
		10	**25**	**50**	**100**
ABC	Ortalama uygunluk değeri	33.7497	**30.1291**	30.4335	29.5504
	Standart sapma	2.7465	1.6421	2.2243	1.3141
	En iyi uygunluk değeri	28.2986	27.8592	27.8558	27.8650
	En kötü uygunluk değeri	37.2843	33.7023	35.2462	31.9778
	Aralık	8.9857	5.8431	7.3903	4.1128
	Geçen ortalama zaman (sn)	658.7605	1512.55	3155.9721	5694.5955
DA	Ortalama uygunluk değeri	44.1887	42.0068	39.6959	39.6322
	Standart sapma	2.2750	2.9013	3.4477	2.3939
	En iyi uygunluk değeri	39.6810	35.6858	34.5224	35.5738
	En kötü uygunluk değeri	47.4990	45.0150	44.3873	42.8027
	Aralık	7.8180	9.3292	9.8649	7.2289
	Geçen ortalama zaman (sn)	371.1625	946.6877	1676.5219	2794.2376
DEA	Ortalama uygunluk değeri	**31.5470**	30.2976	**29.0818**	28.6060
	Standart sapma	1.1229	**0.7754**	**0.9843**	**0.4193**
	En iyi uygunluk değeri	30.4756	28.5780	27.8563	28.0015
	En kötü uygunluk değeri	**34.4117**	**31.4077**	**30.5048**	**29.1910**
	Aralık	3.9361	**2.8297**	**2.6484**	**1.1895**
	Geçen ortalama zaman (sn)	358.0735	839.1341	1652.0999	**2676.3085**
GA	Ortalama uygunluk değeri	44.0272	40.7933	39.3906	38.8936
	Standart sapma	**0.8960**	1.0255	2.1138	1.1332
	En iyi uygunluk değeri	42.6499	38.9304	33.8631	36.2370
	En kötü uygunluk değeri	45.3840	42.3657	41.8453	40.3512
	Aralık	**2.7342**	3.4353	7.9822	4.1142
	Geçen ortalama zaman (sn)	344.5165	810.0436	1682.0768	2922.9062

Çizelge 4.1. (Devamı) Bütün algoritmaların farklı popülasyon büyüklüklerine göre elde edilen sonuçları (İterasyon Sayısı = 200)

		Popülasyon Büyüklüğü			
		10	**25**	**50**	**100**
GWO	Ortalama uygunluk değeri	35.5548	32.1099	29.5936	28.3077
	Standart sapma	3.9295	4.3666	3.6638	0.7456
	En iyi uygunluk değeri	28.6439	27.7217	**27.5239**	**27.5296**
	En kötü uygunluk değeri	40.1800	38.8996	37.9892	29.7795
	Aralık	11.5362	11.1779	10.4653	2.2499
	Geçen ortalama zaman (sn)	**311.4497**	855.8895	**1583.6175**	3058.0902
PSO	Ortalama uygunluk değeri	37.7184	33.9600	33.2920	33.5334
	Standart sapma	4.2882	5.2773	3.9054	4.5894
	En iyi uygunluk değeri	28.2237	27.7563	27.6717	27.8078
	En kötü uygunluk değeri	42.1586	42.7989	38.0265	38.7970
	Aralık	13.9350	15.0426	10.3548	10.9893
	Geçen ortalama zaman (sn)	360.2703	828.4859	1651.0739	2914.2415
SAA	Ortalama uygunluk değeri	33.6560	31.5616	31.2066	33.2717
	Standart sapma	6.1962	5.3027	4.9310	5.3609
	En iyi uygunluk değeri	**27.6914**	27.8042	27.7106	27.6713
	En kötü uygunluk değeri	45.5927	43.6808	40.9759	41.9617
	Aralık	17.9013	15.8766	13.2653	14.2904
	Geçen ortalama zaman (sn)	361.5455	**801.9039**	1699.7117	3184.2987
Hiyerarşik	Ortalama uygunluk değeri	32.8796	30.3486	29.2232	28.3704
	Standart sapma	4.1781	3.9550	2.5799	0.6538
	En iyi uygunluk değeri	27.8545	**27.7148**	27.7300	27.8759
	En kötü uygunluk değeri	37.3466	40.2730	36.7152	29.9020
	Aralık	9.4921	12.5582	8.9853	2.0261
	Geçen ortalama zaman (sn)	347.1286	934.5381	1761.1909	3368.5950



Şekil 4.1. Farklı deneme sayılarına göre en iyi uygunluk değerlerini veren çalışmaların yakınsama eğrileri

Çizelgelerde, ayrıca, standart sapma değerlerine de yer verilmiştir. Ancak standart sapma verisinin tek başına değil, ortalama uygunluk değeri ve en iyi uygunluk değeri alanları ile birlikte dikkate alınması gerekir. Bu duruma en güzel örnek genetik algoritmanın 10 popülasyon büyüklüğü için elde ettiği sonuçtur. İlgili popülasyon büyüklüğü için standart sapmanın elde ettiği değer en küçük iken ortalama uygunluk değeri ve en iyi uygunluk değeri oldukça kötüdür.

Bu çalışmada önerilen Hiyerarşik SAA-GWO yönteminin bazı alanlarda hem gri kurt optimizasyon algoritmasından hem de benzetilmiş tavlama algoritmasından daha iyi sonuçlar üretebildiği görülebilmektedir. Örneğin 10 popülasyon büyüklüğü için Hiyerarşik SAA-GWO yöntemi en iyi ikinci ortalama uygunluk değerini elde etmiştir. Benzetilmiş tavlama algoritması bu hususta üçüncü sıradadır. Gri kurt optimizasyon algoritması ise ilk üçte yer almamaktadır. 25 popülasyon büyüklüğü için ise Hiyerarşik SAA-GWO yöntemi en iyi uygunluk değerini elde etmiştir.

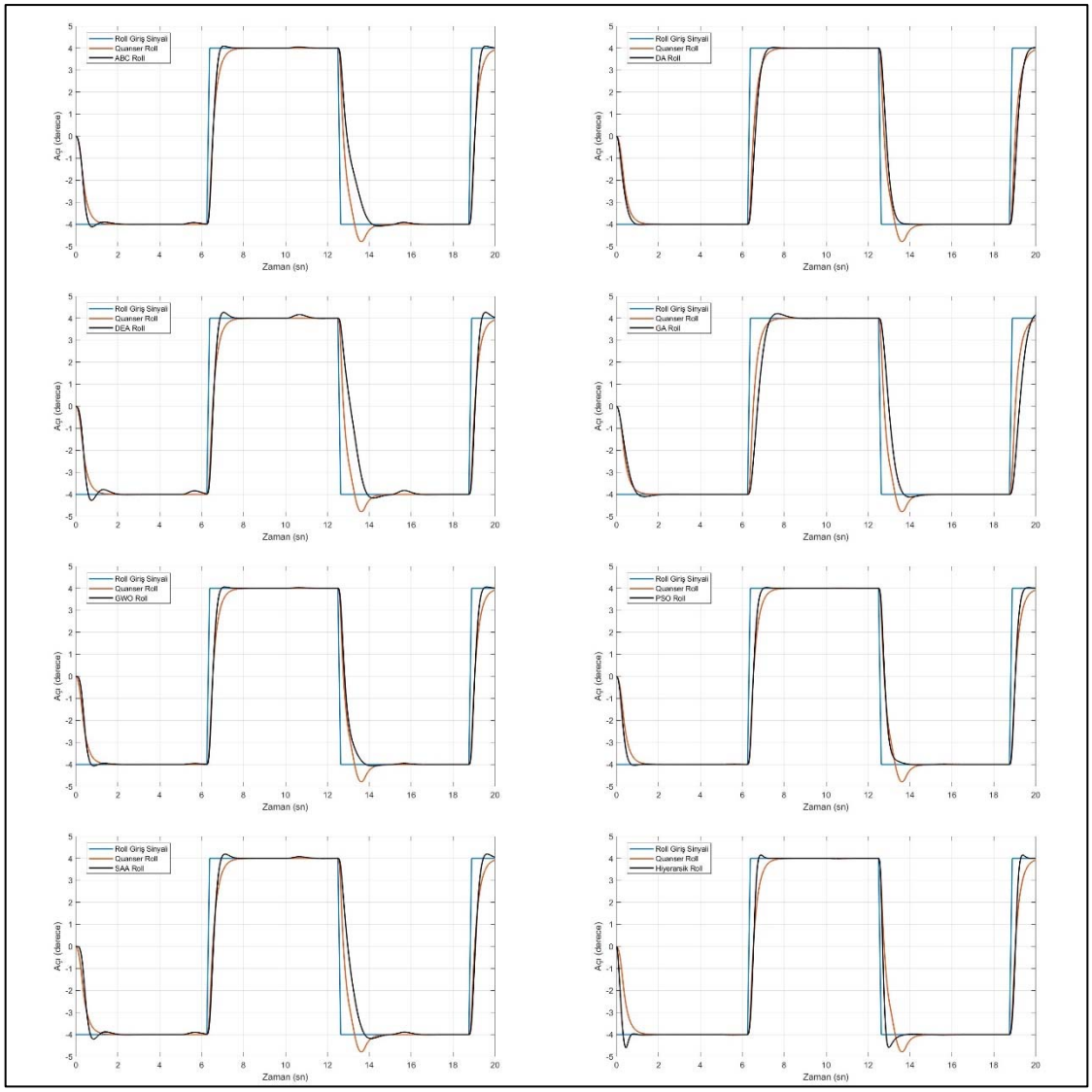
Çizelge 4.1'deki veriler ile yapılan bu karşılaştırmalar sonucunda en iyi ilk 3 yapay zekâ optimizasyon algoritması özetlenerek Çizelge 4.2'te verilmiştir.

Çizelge 4.2. Çizelge 4.1'deki veriler için en iyi üç sonucu veren yapay zekâ optimizasyon algoritmaları (P.b = Popülasyon büyüklüğü, Ort.u = Ortalama uygunluk değeri, Std.s = Standart sapma, En i. = En iyi, En k. = En kötü, Ara. = Aralık, G.s = Geçen süre)

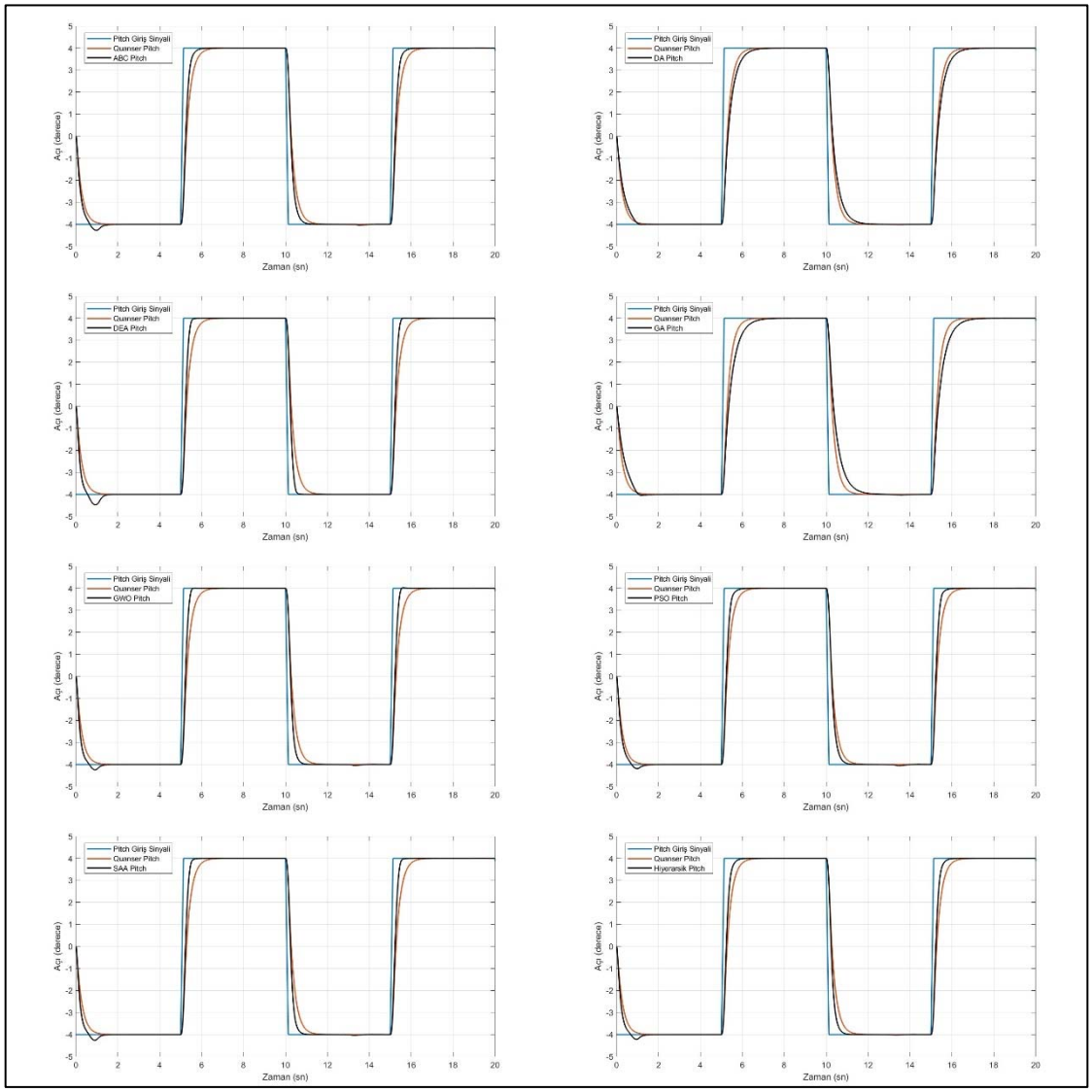
	1.				2.				3.			
P.b	**10**	**25**	**50**	**100**	**10**	**25**	**50**	**100**	**10**	**25**	**50**	**100**
Ort.u	DEA	ABC	DEA	GWO	Hiy.	DEA	Hiy.	Hiy.	SAA	Hiy.	GWO	DEA
Std.s	GA	DEA	DEA	DEA	DEA	GA	GA	Hiy.	DA	ABC	ABC	GWO
En i.	SAA	Hiy.	GWO	GWO	Hiy.	GWO	PSO	PSO	PSO	PSO	SAA	SAA
En k.	DEA	DEA	DEA	DEA	ABC	ABC	ABC	GWO	Hiy.	GWO	Hiy.	Hiy.
Ara.	GA	DEA	DEA	DEA	DEA	GA	ABC	Hiy.	DA	ABC	GA	GWO
G.s	GWO	SAA	GWO	DEA	GA	GA	PSO	DA	Hiy.	PSO	DEA	PSO

Yapay zekâ optimizasyon algoritmalarının 100 popülasyon büyüklüğü (deneme sayısı) için ulaştıkları en iyi çözümlerin 3-DOF Hover deney seti simülasyonuna uygulanmasıyla elde edilen sistem cevapları tek bir şekil üzerinde birleştirilmiştir. Birleştirilen şekiller roll, pitch ve yaw açıları için Şekil 4.2, Şekil 4.3 ve Şekil 4.4'te verilmiştir. Şekillerde görüldüğü gibi mavi, kırmızı ve siyah olmak üzere üç farklı çizgi ile belirtilen sinyaller bulunmaktadır. Mavi çizgi ile gösterilen sinyaller giriş sinyaline aittir. Kırmızı çizgi ile gösterilen sinyaller, Quanser firmasının önerdiği kontrolcü parametrelerinin kullanılmasıyla elde edilen çıkış sinyallerine aittir. Son olarak siyah

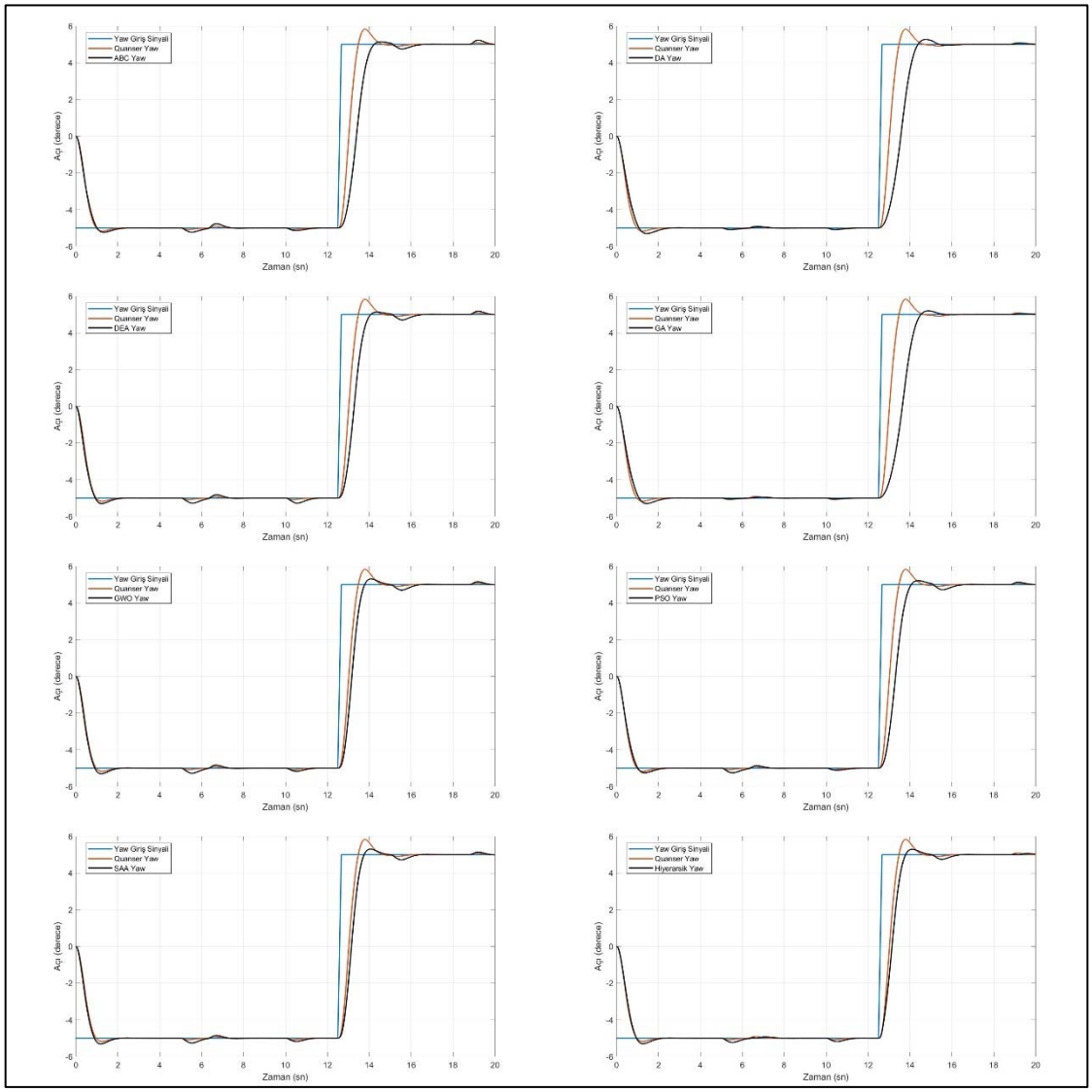
çizgi ile gösterilen sinyaller, yapay zekâ optimizasyon algoritmaları kullanılarak hesaplanan kontrolcü parametrelerinin, sistemde uygulanmasıyla elde edilen sistem cevaplarına ait çıkış sinyalleridir.



Şekil 4.2. Roll açısı için algoritmaların ulaştığı sistem cevapları ile Quanser firmasının parametreleri kullanılarak ulaşılan sistem cevabının karşılaştırılması (İterasyon Sayısı = 200, Popülasyon Büyüklüğü = 100)



Şekil 4.3. Pitch açısı için algoritmaların ulaştığı sistem cevapları ile Quanser firmasının parametreleri kullanılarak ulaşılan sistem cevabının karşılaştırılması (İterasyon Sayısı = 200, Popülasyon Büyüklüğü = 100)



Şekil 4.4. Yaw açısı için algoritmaların ulaştığı sistem cevapları ile Quanser firmasının parametreleri kullanılarak ulaşılan sistem cevabının karşılaştırılması (İterasyon Sayısı = 200, Popülasyon Büyüklüğü = 100)

Çizelge 4.3'te yapay zekâ optimizasyon algoritmaları kullanılarak ulaşılan Şekil 4.2, Şekil 4.3 ve Şekil 4.4'teki sistem cevaplarına ait değerler verilmiştir. Elde edilen en iyi değerler koyu ile gösterilmiştir. Çizelge 4.3'den görüleceği üzere kullanılan algoritmalar ile yaw açısının oturma ve yükselme zamanları ile pitch açısının üst aşım değeri dışında kalan veriler için gelişim sağlanmıştır. Bu çalışmada önerilen Hiyerarşik SAA-GWO yöntemi ile roll açısının oturma ve yükselme zamanlarında en iyi değerler elde edilmiştir.

Çizelge 4.3. Şekil 4.2, Şekil 4.3 ve Şekil 4.4'te verilen çıkış sinyallerine ait değerler

ALG	Oturma Zamanı (sn)			Yükselme Zamanı (sn)			Üst Aşım (deg)		
	Roll	Pitch	Yaw	Roll	Pitch	Yaw	Roll	Pitch	Yaw
Qua	19.7107	16.0741	**14.3349**	0.5042	0.5440	**0.3475**	0.7890	**0.0003**	0.8323
DA	19.6582	16.3588	15.0229	0.4040	0.7144	0.5771	**0.0313**	0.0097	0.3072
GA	19.8586	16.5520	14.4112	0.4837	0.8230	0.6087	0.2034	0.0437	0.3002
PSO	19.3961	15.6035	15.8231	0.2763	0.2677	0.5182	0.0337	0.1882	0.2557
SAA	19.4051	15.5028	15.7505	0.2712	0.2122	0.4276	0.2052	0.2613	0.3122
ABC	19.3626	15.6683	19.3429	0.2554	0.2956	0.5559	0.1120	0.2723	**0.2359**
DEA	19.6945	15.4931	15.8278	0.2399	0.2084	0.5195	0.2731	0.4739	0.3043
GWO	19.3775	**15.4815**	15.8019	0.2622	**0.2027**	0.4326	0.0629	0.2418	0.3078
Hiy.	**19.2470**	15.6549	15.8021	**0.1678**	0.2826	0.4666	0.6000	0.2175	0.3023

5. SONUÇLAR VE ÖNERİLER

Bu tez çalışması kapsamında quadcopterlerin uçuş esnasındaki davranışlarının incelenebilmesi için kullanılan bir deney setinin Simulink ortamında kontrolü gerçekleştirilmiştir. Sistemin kontrolü için LQR kontrol yöntemi kullanılmıştır. Kullanılan kontrol yönteminin parametreleri yapay zekâ optimizasyon algoritmaları ile belirlenmiştir. Önerilen Hiyerarşik SAA-GWO algoritması ile birlikte sekiz farklı yapay zekâ optimizasyon algoritması kullanılmıştır. Algoritmalar ile belirlenen yeni parametrelerin sisteme uygulanmasıyla yeni sistem cevapları elde edilmiştir. Ulaşılan yeni sistem cevapları ile Quanser firması tarafından varsayılan olarak kullanılan parametrelerin sisteme uygulanmasıyla elde edilen sistem cevapları karşılaştırılmıştır.

Yapay zekâ optimizasyon algoritmalarının farklı alanlarda birbirlerine göre üstünlükleri bulunmaktadır. Yapılan karşılaştırma Çizelge 4.1'den görülebilmektedir. Çizelgelerde genetik ve yusufçuk algoritmalarının ulaştıkları en iyi uygunluk değerlerinin diğer algoritmaların ulaştıkları en iyi uygunluk değerlerine göre yüksek olduğu görülmektedir. Bu duruma paralel olarak genetik ve yusufçuk algoritmalarının ulaştıkları sistem cevapları da diğer algoritmaların ulaştığı sistem cevaplarına göre kötüdür. Ayrıca bahsedilen iki algoritmanın ulaştıkları sistem cevapları, Quanser firması tarafından varsayılan olarak kullanılan parametrelerin sisteme uygulanmasıyla elde edilen sistem cevaplarına karşı üstünlük sağlayamamıştır. Bu durum Çizelge 4.3'ten görülebilmektedir.

Yapılan çalışmaların tamamı simülasyon ortamında gerçekleştirilmiştir. İlerleyen çalışmalarda yapay zekâ optimizasyon algoritmaları kullanılarak belirlenen kontrolcü parametreleri gerçek sistem üzerinde denenebilir. Bu sayede parametrelerin gerçek sistem üzerindeki etkisi gözlemlenebilir.

3-DOF Hover test düzeneği oldukça pahalı bir sistemdir. Bu yüzden, bu ve benzeri çalışmalarda kullanılmak üzere yeni bir test platformu geliştirilebilir. Kaynak araştırması esnasında daha hesaplı sistemlerin geliştirildiğine dair çalışmalarla karşılaşılmıştır. Benzer bir test platformunu geliştirmek için ilgili çalışmalar referans kaynak olarak alınabilir.

KAYNAKLAR

[1] M. Dikmen, «İnsansız Hava Aracı (İHA) Sistemlerinin Hava Hukuku Bakımından İncelenmesi,» *The Journal of Defense Sciences,* cilt 14, no. 1, pp. 145-176, 2015.

[2] A. Reizenstein, "Position and Trajectory Control of a Quadcopter Using PID and LQ Controllers", Master's thesis, Linköping: Linköping University, 2017.

[3] T. Oktay ve O. Köse, «Dynamic Modeling and Simulation of Quadrotor for Different Flight Conditions,» *Avrupa Bilim ve Teknoloji Dergisi,* no. 15, pp. 132-142, 2019.

[4] M. K. Bayrakçeken, "Dikine İniş Kalkış Yapabilen Dört Rotorlu Hava Aracının (Quadrotor) Uçuş Kontrolü", Doktora tezi, Eskişehir: Eskişehir Osmangazi Üniversitesi Fen Bilimleri Enstitüsü, 2013.

[5] H. K. Tran ve T. N. Nguyen, «Flight Motion Controller Design using Genetic Algorithm for a Quadcopter,» *Measurement and Control,* no. 51, pp. 59-64, 2018.

[6] R. W. Beard, «"Quadrotor Dynamics and Control Rev 0.1",» Brigham Young University, Brigham, 2008.

[7] Ö. Bayraktar ve A. Güldaş, «Optimization of Quadrotor's thrust and torque coefficients and simulation with Matlab/Simulink,» *Journal of Polytechnic,* no. 23, pp. 1197-1204, 2020.

[8] V. E. Ömürlü, U. Büyükşahin, R. Artar, A. Kırlı ve M. N. Turgut, «An experimental stationary quadrotor with variable DOF,» *Sādhanā,* no. 38, pp. 247-264, 2013.

[9] A. Ateş, "Prototip Helikopter Sisteminin Matematiksel Modelinin Deneysel Belirlenmesi Ve Denetçi Tasarımı", Yüksek lisans tezi, Malatya: İnönü Üniversitesi Fen Bilimleri Enstitüsü, 2013.

[10] A. Demiryürek, "Modeling And Control Of A Quadrotor", Yüksek lisans tezi, Ankara: Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, 2018.

[11] Anonim, «www.mathworks.com,» MathWorks, [Çevrimiçi]. Available: https://www.mathworks.com/hardware-support/parrot-drone-matlab.html. [Erişildi: 14 Haziran 2021].

[12] Anonim, «www.quanser.com,» Quanser, [Çevrimiçi]. Available: https://www.quanser.com/products/3-dof-hover/. [Erişildi: 14 Haziran 2021].

[13] S. Mohanty ve A. Misra, «3 DOF Autonomous Control Analysis,» *Modern Approaches in Machine Learning and Cognitive Science: A Walkthrough,* no. 885, pp. 39-57, 2020.

[14] M. İçen, A. Ateş ve C. Yeroğlu, «Optimization of LQR Weight Matrix to Control Three Degree of Freedom Quadcopter,» %1 içinde *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)*, Malatya, 2017.

[15] K. A. Cheng, «Teaching Mathematical Modelling in Singapore Schools,» *The Mathematics Educator,* no. 6, pp. 62-74, 2001.

[16] Anonim, «www.mathworks.com,» MathWorks, [Çevrimiçi]. Available: https://www.mathworks.com/help/control/ref/lti.stepinfo.html. [Erişildi: 16 Haziran 2021].

[17] S. Mirjalili, «seyedalimirjalili.com/da,» [Çevrimiçi]. Available: https://seyedalimirjalili.com/da. [Erişildi: 14 Haziran 2021].

[18] S. Mirjalili, «Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems,» *Neural Comput & Application,* 2015.

[19] H. Gülcan, Yusufçuk Algoritmasının Brownian Hareket İle İyileştirilmesi, Yüksek lisans tezi, Mersin: Mersin Üniversitesi Fen Bilimleri Enstitüsü, 2018.

[20] D. E. Goldberg ve J. H. Hollanf, «Genetic Algorithms and Machine Learning,» *Machine Learning,* no. 3, pp. 95-99, 1988.

[21] A. Hassananat, K. Almohammadi, E. Alkafaween, E. Abunawas, A. Hammouri ve . S. Prasath, «Choosing Mutation and Crossover Ratios for Genetic Algorithms—A Review with a New Dynamic Approach,» *İnformation,* no. 10, 2019.

[22] J. Kenedy ve R. Eberhart, «Particle Swarm Optimization,» %1 içinde *In Proceedings of IEEE international conference on neural networks*, Perth, Australia, 1995.

[23] J. C. Bansal, P. K. Singh, M. Saraswat, A. Verma, S. S. Jasdon ve A. Abraham, «Inertia Weight Strategies in Particle Swarm Optimization,» %1 içinde *Third World Congress on Nature and Biologically Inspired Computing*, Salamanca, Spain, 2011.

[24] D. Wang, D. Tan ve L. Liu, «Particle swarm optimization algorithm: an overview,» *Soft Computing,* cilt 2, no. 22, pp. 387-408, 2018.

[25] S. Kirkpatrick, C. D. Gelatt ve J. M. P. Vecchi, «Optimization by Simulated Annealing,» *Science,* no. 220, pp. 671-680, 1983.

[26] H. Shakouri, K. Shojaee ve M. Behnam, «Investigation on the choice of the initial temperature in the Simulated Annealing: A Mushy State SA for TSP,» %1 içinde *17th Mediterranean Conference on Control & Automation*, Thessaloniki, Greece, 2009.

[27] A. K. Peprah, S. K. Appiah ve S. K. Amponsah, «An Optimal Cooling Schedule Using a Simulated Annealing Based Approach,» *Mathematics,* no. 8, pp. 1195-1210, 2017.

[28] D. Karaboğa, «An Idea Based On Honey Bee Swarm For Numerical Optimization,» Erciyes University, Engineering Faculty Computer Engineering Department, Kayseri, 2005.

[29] R. Storn ve K. Price, «Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces,» *Journal of Global Optimization,* no. 11, pp. 341-359, 1997.

[30] S. Mirjalili, S. M. Mirjalili ve A. Lewis, «Grey Wolf Optimizer,» *Advances in Engineering Software,* no. 69, pp. 46-61, 2014.

EKLER

EK-1 Uygulamalar gerçekleştirilirken kullanılan amaç fonksiyonunun Matlab kodu

```
S1 = stepinfo(Response_Yaw, tout);
S2 = stepinfo(Response_Pitch, tout);
S3 = stepinfo(Response_Roll, tout);

a1 = ((S1.SettlingTime) / (S1.RiseTime));
a2 = ((S2.SettlingTime) / (S2.RiseTime));
a3 = ((S3.SettlingTime) / (S3.RiseTime));
peak1 = abs(5 - rad2deg(S1.Peak));
peak2 = abs(4 - rad2deg(S2.Peak));
peak3 = abs(4 - rad2deg(S3.Peak));
s = (...
    + (S1.RiseTime + S2.RiseTime + S3.RiseTime)...
    + (S1.SettlingTime/a1 + S2.SettlingTime/a2 + S3.SettlingTime/a3) ...
    + (S1.PeakTime + S2.PeakTime + S3.PeakTime)...
    + (2*S1.Overshoot + S2.Overshoot + S3.Overshoot) / 100 ...
    + (peak1 + peak2 + peak3)...
    + (norm(Response_Roll) + norm(Response_Pitch) + norm(Response_Yaw))...
    );
```