

İÇİNDEKİLER

	Sayfa
SİMGE LİSTESİ	iv
KISALTMA LİSTESİ	v
ŞEKİL LİSTESİ	vi
ÇİZELGE LİSTESİ	vii
ÖNSÖZ.....	viii
ÖZET	ix
1. GİRİŞ	1
1.1 Motivasyon	1
1.2 Literatür Taraması	1
1.3 Amaç	2
1.4 Vinç Çeşitleri	2
2. TEPE VİNCİNİ OLUŞTURAN SARKAÇ VE ARABA SİSTEMİNİN MODELLENMESİ.....	5
2.1 Lagrangian Dinamikleri Analizi	5
2.2 Çalışma Noktası Yakınında Lineerleştirme.....	9
2.3 Sistemin Transfer Fonksiyonu	10
2.4 Durum-Uzay (State-Space) Gösterimi.....	11
3. KONTROL SİSTEMLERİ	13
3.1 Matematiksel Modelleme	14
3.2 Kontrol Yöntemleri.....	14
3.2.1 Açık Çevrim (Open-Loop) Kontrol Yöntemleri.....	14
3.2.2 Kapalı Çevrim (Closed-Loop) Kontrol Yöntemleri	15
3.3 Kontrol Sistemlerinde Kararlılığın İncelenmesi	15
3.3.1 s-Domeni Yaklaşımı	16
3.3.2 Zaman Domeni Yaklaşımı	17
3.3.3 Frekans Domeni Yaklaşımı	20
4. PID KONTROLÖRLER.....	22
4.1 SISO PID Kontrolör Tasarım Metotları	25
4.1.1 Ziegler-Nichols Metodu.....	25
4.1.2 Cohen Coon (C-C) Metodu	27
4.1.3 İç Model Kontrolü	27
4.1.4 Ho-ang-Cao Metodu	28
4.2 MIMO PID Kontrolör Tasarım Metotları.....	29
4.2.1 Genel Ziegler-Nichols Metodu	29
4.2.2 Zhuang ve Atherton Optimizasyon Metodu	31

4.2.3	Seraji Metodu.....	31
5.	GENETİK ALGORİTMALAR	33
5.1	Genetik Algoritmanın Tarihçesi	33
5.2	Genetik Algoritma Nedir	33
5.3	Genetik Algoritmalar Sözlüğü.....	36
5.4	Genetik Algoritma Uygulama Alanları	37
5.5	Genetik Algoritma ile Geleneksel Yöntemler Arasındaki Farklar	38
5.6	Genetik Algoritmanın Avantajları	40
5.7	Kodlama (Encoding).....	41
5.7.1	İkili Kodlama (Binary Encoding)	41
5.7.2	Dizilim (Permütasyon) Kodlaması	41
5.7.3	Reel (Gerçel) Kodlama	42
5.7.4	Ağaç Kodlaması.....	42
5.8	İlk Popülasyonun Rasgele Oluşturulması.....	43
5.9	İlk Popülasyonun Uygunluk Değerinin Hesaplanması.....	43
5.10	Yeni Popülasyonun Oluşturulması	44
5.11	Seçim (Selection).....	44
5.11.1	Rulet Çarkı Sistemi.....	44
5.11.2	Rank Seçimi.....	46
5.11.3	Kararlı Hal Seçimi	48
5.11.4	Turnuva Seçimi.....	48
5.11.5	Elitistik Seçim.....	48
5.12	Çaprazlama	48
5.12.1	Tek noktalı çaprazlama.....	48
5.12.2	Çok Noktalı Çaprazlama.....	49
5.12.3	Düzgün (Uniform) Çaprazlama	50
5.12.4	Karışık (Shuffled) Çaprazlama	50
5.13	Mutasyon	51
5.14	Genetik Algoritmanın Kontrol Parametreleri	51
5.15	Genetik Algoritma Uygulamasına Bir Örnek	53
6.	KONTROL SİSTEMİ TASARLANMASI	56
6.1	Amaç.....	56
6.2	Sistemin PID Tasarımı.....	56
6.3	Sistemin Genetik Algoritmalarla Yararlanılarak Tasarlanması.....	63
7.	SONUÇLAR.....	71
	KAYNAKLAR	72
	EKLER	74
	ÖZGEÇMİŞ.....	83

SİMGE LİSTESİ

x	Arabanın yatay pozisyonu
x_l	Sarkacın yatay pozisyonu
θ	Salınım açısı
y	Sarkacın dikey pozisyonu
l	Sarkaç ipi uzunluğu
M	Arabanın kütlesi
m	Sarkaç yükünün kütlesi
g	Yerçekimi ivmesi
L	Kinetik ve potansiyel enerjiden oluşan Lagrangian sistemi
T	Sistemin toplam kinetik enerjisi (E_K)
U	Sistemin toplam potansiyel enerjisi (E_P)
D	Sürtünmede kaybolan enerji
i	$x(t)$ ve $\theta(t)$ serbestlik dereceleri
r	Referans giriş
q_i	Genelleştirilmiş koordinat setleri ; $i=1,2,3,...,n$
F_i	q serbestlik derecesinde oluşan net kuvvet
$\dot{x}$	Arabanın hızı
$\dot{x}_1$	Sarkacın yatay hızı
$\dot{\theta}$	Salınım hızı
N	Popülasyon büyüklüğü
P_c	Çaprazlama oranı
P_m	Mutasyon oranı
G	Nesil ayrılığı
OS	Maksimum üst aşım (Overshoot)
ζ	Sönüm oranı
t_p	Tepe zamanı
t_s	Yerleşme zamanı
t_r	Yükselme zamanı
t_d	Gecikme zamanı
T_i	İntegral kazancı
T_d	Türev kazancı
K_p	Oransal kazanç
K_i	İntegral kazancı
K_d	Türev kazancı

KISALTMA LİSTESİ

ISE	Integral of squared error
ITAE	Integral of time-multiplied by absolute error
ITSE	Integral of time squared error
IAE	Integral of absolute error
LQR	Linear Quadratic Regulator
GA	Genetik Algoritma

ŞEKİL LİSTESİ

Şekil 1.1 Rotary crane (Al-Mousa 2000)	3
Şekil 1.2 Bomlu vinç (Al-Mousa 2000)	3
Şekil 1.3 Gantry crane (Al-Mousa 2000)	4
Şekil 2.1 Tepe vinci sisteminin şematik gösterimi	5
Şekil 2.2 Kaskat halde transfer fonksiyonu gösterimi	10
Şekil 3.1 Temel bir kontrol sistemi	13
Şekil 3.2 Kontrol sistem mimarisi	13
Şekil 3.3 Açık çevrim kontrol sistemi	14
Şekil 3.4 Açık çevrim sistemle oda sıcaklık kontrolünün yapılması	14
Şekil 3.5 Kapalı çevrim kontrol sistemi	15
Şekil 3.6 Kapalı çevrim sistemle oda sıcaklık kontrolünün yapılması	15
Şekil 3.7 Köklerin s-düzlemindeki yerlerine bağlı olarak değişen ani darbe cevap eğrileri	16
Şekil 3.8 Zaman alanı cevabı parametreleri	17
Şekil 4.1 Adaptif kontrol edicinin basit bir şematik gösterimi (Gündoğdu,2005)	22
Şekil 4.2 Geri beslemeli kontrol sistemi	23
Şekil 4.3 Açık çevrim basamak cevabı	26
Şekil 5.1 İkili kodlama yapısındaki birey	34
Şekil 5.2 Genetik algoritma programı blok şeması	35
Şekil 5.3 $f(x) = x^2$ amaç fonksiyonu	39
Şekil 5.4 Kapalı kutu örneği	39
Şekil 5.5 Rulet çarkı sistemi	44
Şekil 5.6 Üretilen sayıların (deneme i) uygun düştüğü birey aralıkları	45
Şekil 5.7 Sıralama öncesi durum (uygunluk değerlerine göre grafik)	46
Şekil 5.8 Sıralama sonrası durum	46
Şekil 5.9 Lineer ve lineer olmayan rank seçimlerinin karşılaştırılması	47
Şekil 5.10 Tek noktalı çaprazlama	48
Şekil 5.11 Çok noktalı çaprazlama	49
Şekil 6.1 $K_p = 90$, $K_d = 28,7$ ve $K_i = 1,18$ için sistemin birim basamak cevabı	59
Şekil 6.2 $K_p = 90$, $K_d = 65$ ve $K_i = 3$ için sistemin birim basamak cevabı	60
Şekil 6.3 Sistemin PID kontrol simulink diyagramı	61
Şekil 6.4 Satürasyondan geçmiş giriş sinyali	62
Şekil 6.5 Çıkış sinyali	62
Şekil 6.6 Genetik algoritma ile PID optimizasyon genel şeması	63
Şekil 6.7 Genetik algoritmayla hesaplanan sistemin birim basamak cevabı	65
Şekil 6.8 Her iterasyon sonunda ortalama karesel hatanın değişimi	66
Şekil 6.9 K_p 'nin iterasyonlar boyunca değişimi	67
Şekil 6.10 K_d 'nin iterasyonlar boyunca değişimi	67
Şekil 6.11 K_i 'nin iterasyonlar boyunca değişimi	68
Şekil 6.12 Optimal çözümü sağlayan kromozoma ait uygunluk değeri değişimi	68
Şekil 6.13 Genetik algoritma boyunca kromozomların uygunluk değerleri değişimi	69
Şekil 6.14 Her iterasyonda en iyi olan kromozom (birey)	70

ÇİZELGE LİSTESİ

Çizelge 4.1 P, I ve D'nin karakteristikleri	25
Çizelge 4.2 Z-N ayar kuralları (sürekli döngü için)	26
Çizelge 4.3 Z-N ayar kuralları (süreç eğrisi metodu için)	27
Çizelge 4.4 IMC tabanlı PID kontrol edici parametreleri	28
Çizelge 4.5 Genel Z-N kural tablosu	30
Çizelge 5.1 Genetik algoritmanın akış tablosu	35
Çizelge 5.2 İkili kodlama kullanılarak oluşturulan bireyler	41
Çizelge 5.3 Dizilim kodlaması kullanılarak oluşturulan bireyler	42
Çizelge 5.4 Gerçel kodlama kullanılarak oluşturulan bireyler	42
Çizelge 5.5 Ağaç yapı kodlaması kullanılarak oluşturulan bireyler	43
Çizelge 5.6 Seçilme olasılıkları ve uygunluk değerleri	45
Çizelge 5.7 Uygunluk değerinin seçilme baskısı ile değişimi	47
Çizelge 5.8.a Genetik Algoritmalar	54
Çizelge 5.8.b (devam)	54
Çizelge 6.1 Routh Tablosu	56
Çizelge 6.2 Yeni Routh Tablosu	57
Çizelge 6.3 Hesaplanan farklı PID parametreleri için üst aşım, yükselme ve oturma zamanı karşılaştırması	65

ÖNSÖZ

Öncelikle, hem tezin hazırlanmasında hem de moral olarak hiçbir zaman yardımlarını esirgemeyen değerli danışman hocam Sayın Yrd. Doç. Dr. Şeref Naci ENGİN' e minnetlerimi sunarım. Ayrıca, sadece tez döneminde değil, hayatım boyunca sonsuz desteklerini benden esirgemeyen aileme ve bu tezde, az veya çok, emeği geçen tüm arkadaşlarıma, dostlarıma şükranlarımı sunarım.

ÖZET

Ağır nesnelerin taşınmasında en çok kullanılan yapılardan biri olan vinçler genellikle operatörler tarafından manuel olarak kontrol edilmektedirler. İnsana dayalı bu kontrol yönteminde, halatlara bağlı cisimler üzerinde istenilmeyen salınımlar oluşmakta, bu da etraftaki diğer cisimler hatta çalışanların hayatları için büyük bir risk teşkil etmektedir.

Bu çalışmada iki farklı kontrol yöntemi ile kontrolcü tasarlanarak, minimum salınımlı pozisyon kontrolü amaçlanmıştır. Örnek olarak tepe vinç (overhead crane) sistemi alınıp sistem modelleri ve karakteristikleri çıkarılmış ve sistemin transfer fonksiyonu elde edilmiştir. s domeni, zaman ve frekans domenlerinde kontrol sistemleri kısaca anlatılarak PID kontrolörler ve Genetik Algoritmalar hakkında ayrıntılı bilgiler verilmiştir.

Kontrolcü olarak klasik PID kontrolörler ve Genetik Algoritmalar seçilerek her 2 kontrolcü teker teker sisteme uygulanmış ve sistem cevapları elde edilmiştir. Öncelikle PID kontrolörlerden Ziegler-Nichols metodu ile katsayılar (K_c , T_i , T_d - K_p , K_i , K_d) bulunmuş ve sisteme uygulanarak çıktılar elde edilmiştir. Bu işlemler önce Matlab komut satırlarıyla çıkarılmış daha sonra bir simulasyon ile sonuçlar elde edilmiştir.

En iyinin hayatta kalması prensibine dayanan Genetik Algoritmalar, ISE (toplam karesel hatanın integrali) performans indeksi kullanılarak sisteme uygulanmış ve hatayı minimum yapan katsayılar bulunarak çıkış grafikleri elde edilmiştir. Son olarak her iki yöntemle yapılan Matlab çalışmaları ve simulasyonlar sonucunda elde edilen grafiklerden sonuçlar çıkarılmış, sonuçlar ve karşılaştırması tezin bitiminde ele alınmıştır. Matlab komut satırları açıklanarak teze eklenmiştir.

Anahtar kelimeler: Tepe vinci, kontrol sistemleri, PID kontrolörler, Genetik Algoritmalar, Matlab, Simulink

ABSTRACT

The cranes, which are the mostly used tools for transporting heavy loads, are generally operated manually. This causes unwanted oscillatory movement of the load, so constitutes danger to surroundings and workers.

In this work, we intended to get the desired position with minimum oscillation of the load by using two different control methods. We deal with widely used crane construction: the overhead crane. We obtained system models, system characteristics and the transfer function. Control designs in s-domain, time domain and frequency domain are presented. PID controllers and Genetic Algorithms are explained in detail.

PID controllers and genetic algorithms are chosen as controllers and by taking into consideration of those two controllers; the system responses obtained for each. First we found coefficients (K_c , T_i , T_d - K_p , K_i , K_d) by using Ziegler-Nichols method. The computed coefficients were used for system and obtained outputs. These processes are obtained from Matlab command lines and simulation.

Genetic algorithms, whose main principle is “only the best survive”, are applied to the system by using ISE performance index and the coefficients, which produce minimum error, are obtained. After all simulations were done; conclusions were drawn based on responses of the system to PID and genetic algorithms methods. Conclusions and comparisons are dealt with at the end of the thesis. The Matlab command lines added to the thesis with explanations.

Keywords: Overhead crane, control systems, PID controllers, Genetic Algorithms, Matlab, Simulink

1. GİRİŞ

1.1 Motivasyon

Günümüzde sanayide ağır nesnelerin bir yerden başka bir yere taşınması işlemlerinde en çok vinçlerden faydalanılmaktadır. Fakat yükün salınım hareketi genellikle operasyon verimi ve güvenliğini sınırlamaktadır. Yükün salınım açısını düşürmek için, vinç özel bir kontrol yöntemine ihtiyaç duyar.

Bu problem popüler olarak vinç sistemlerinin salınım (anti-swing) kontrolü olarak bilinir ve kontrol mühendislerinin uzun zamandır ilgisini çekmektedir. Kontrolör daha önceden bir ön bilgi olmadan geniş bir aralıktaki yükleri taşıyabilecek kapasitede olmalıdır, bu da endüstriyel bir vincin tipik bir özelliğidir. Bu salınım açısını gidermek için öngördüğümüz tasarımda sistem modeli, klasik kontrol yöntemleri olan PD – PID Kontrolörler ve Genetik Algoritmalar ile kontrol edilecektir.

Özellikle sanayide kullanılan vinçler, bir operatör tarafından manuel olarak kullanılmaktadırlar. Burada operatörün en küçük bir ani hareketi bile sistemde büyük salınımlar oluşturarak birçok hataya neden olmaktadır.

1.2 Literatür Taraması

Bir tepe vincinin (overhead crane) modellenmesi ve kontrolü uzun zamandır araştırma konusu olmaktadır. Birçok araştırmacı vinçlerin salınım kontrolü konusu üzerinde çalışmıştır. Onların çalışmaları genel olarak iki grupta toplanabilir:

- Açık çevrim kontrol
- Kapalı Çevrim kontrol

Açık çevrim metotlar; alçak geçiren (low pass) filtre, simetrik ivmelenme profili (symetric acceleration profile) ve giriş şekillendirme (input shaping) metotları salınım hareketini harekete geçirmemeye çalışmaktadır. Uygulaması basittir fakat dış distürbanslardan kaynaklanan salınım hareketini bastıramazlar. Aynı zamanda geniş bir yük yelpazesinde tatmin edici performans verememektedirler.

Diğer taraftan salınım hareketinin kapalı çevrim metodu, var olan salınım hareketini elemine etmeyi sağlar ve eklenen sensörlerin yardımıyla distürbanslardan kurtulur.

Ho-Hoon Lee, kök yer eğrisi temelli kapalı çevrim kontrolör geliştirmiştir. Aynı zamanda üç

boyutlu dinamik tepe vinci modelini 2 serbestlik dereceli salınım açısına dayanarak yaklaşık olarak elde etmiştir. Ama ayırık (decoupled) kontrol yasasını bulmak için dinamik eşitliğini lineerleştirmiştir. Yeni ayırık kontrol yasası değişen yüklerde çalışmamakta, yük hakkında ekstra bilgiye ihtiyaç duymaktadır.

Wen, LQR metoduna dayalı bir kapalı çevrim kontrolör tasarlamıştır. Bu yaklaşımda da aynı şekilde yük hakkında önceden bilgi sahibi olmak gerekmektedir.

Chi-Chen Cheng ve Cheng-Yi Chen geribesleme lineerleştirme tekniği kullanan kontrol yasası elde etmiştir. Bu metotta değişen yük konusu işlenmiştir.

J.A. Mendez self-tuning kontrolüne dayalı daha kompleks yapay sinir ağı kontrolör hazırlamıştır. M. Mahfouf ve A. Benhidjeb bulanık mantık tabanlı kontrolör geliştirmişlerdir. Yukarıda bahsedilen tüm kontrolörler bir dereceli lineer tepe vinci modelleriyle sınırlıdır.

Ji Sup Yoon kapalı ve açık çevrim yaklaşımları kullanan salınımı giderici birçok kontrolör dizayn etmiştir. Daha verimli ve robust kontrolör geliştirmek için, önceden programlanmış geri besleme kontrolör ve fuzzy kontrolör geliştirmiştir. Tüm kontrolörlerini 2 serbestlik dereceli lineer vinç sistemine uygulamıştır.

Genelde analizler sadece 2 serbestlik dereceli tepe vinçlerinde sınırlandırılmıştır. Aynı zamanda vinç kriterleri genellikle ihmal edilmiş ve sadece bir hız kaynağı olarak kontrol edilmiştir. Yapılan çalışmalardan görülebildiği gibi tüm sistemin kesin modelini gerektirmekte ve geniş bir yük yelpazesinde salınım kontrolünü tam olarak sağlayamamaktadır.

1.3 Amaç

Bu bahsedilen hataları önlemek üzere bir kontrolcü tasarlanması ve iki serbestlik derecesinde; salınım açısını minimize ederek pozisyon kontrolü sağlayacak operatörsüz çalışabilen bir tepe vinci sistemi simüle etmektir. Yapılan çalışma; sistem dinamiklerinin teorik analizlerini ve kontrol metotlarını içermektedir. Matlab yazılımı ve Simulink blokları yardımı ile sistemin simülasyonu elde edilerek sonuçların alınmasında kolaylık sağlanmış ayrıca görsel olarak sistemin daha kolay anlaşılmasına yardımcı olunmuştur.

1.4 Vinç Çeşitleri

Vinçler mekanik ve dinamik yapılarına göre genel olarak 3 çeşide ayrılmıştır.

- 1 eksen etrafında dönen vinç (Rotary crane)

İnşaat sektöründe en çok kullanılan vinç tipidir. Sabit bir nokta etrafında rotasyonel hareket yaparak bir konumdan istenilen başka bir konuma geçiş yapabilmektedir.



Şekil 1.1 Rotary crane (Al-Mousa 2000)

- Bomlu vinci

Gemi ve limanlarda kullanılan çok aksamli vinç türüdür. Hem radyal hem de yatay hareket sağlayabilmektedir.



Şekil 1.2 Bomlu vinç (Al-Mousa 2000)

- Tepe vinci (Gantry- Overhead crane)

X-Y-Z düzlemlerinde doğrusal olarak hareket ederek daha hassas konum kontrolü sağlayan bir

vin türüdür. Kullanım sahası en geniş olan vintir. Açık ve kapalı alanlarda kullanıma uygundur. Tez alışmasında üzerinde alışılan vin tipidir.



Şekil 1.3 Gantry crane (Al-Mousa 2000)

Şekil 2.1 de basit bir tepe vinci sisteminin genel diyagramı gösterilmiş olup analizler için bu model kullanılmıştır. Burada araba ağırlığı M , sarkaç ağırlığı m olarak adlandırılmış; kullanılan ipin ağırlıksız ve rijit olduğu kabul edilmiştir. Ayrıca sistem sürtünmesiz kabul

edildiğinden sürtünme bileşenleri modelde gösterilmemiştir.

x : arabanın yatay pozisyonu (m)

x_I : sarkacın yatay pozisyonu (m)

y : Sarkacın dikey pozisyonu (m)

θ : salınım açısı (rd)

l : Sarkaç ipi uzunluğu (m)

İki serbestlik derecesine sahip bu sistemin modeli Lagrangian dinamikleri kullanarak elde edilmiştir. Lagrangian eşitlikleri hareket denklemlerini, sürtünmeli ve de sürtünmesi ihmal edilmiş sistemler için elde etmemizi sağlar. Sürtünmesiz yaklaşımda sistemin enerji kaybının olmadığı kabul edilir. Lagrangian eşitlikleri kullanan bir sistemin dinamiklerini elde etmek için gerekli matematiksel eşitlikler, aşağıdaki şekilde verilmiştir:

Kinetik ve potansiyel enerji denklemlerinden

$$L = T - U = E_k - E_p \quad (2.1)$$

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{q}_i} \right) - \frac{\partial L}{\partial q_i} + \frac{\partial D}{\partial \dot{q}_i} = F_i \quad (2.2)$$

(Williams, James 1996)

L : Kinetik ve potansiyel enerjiden oluşan Lagrangian sistemi

T : Sistemin toplam kinetik enerjisi (E_K)

U : Sistemin toplam potansiyel enerjisi (E_P)

D : Sürtünmede kaybolan enerji

F_i : q serbestlik derecesinde oluşan net kuvvet

i : $x(t)$ ve $\theta(t)$ serbestlik dereceleri (number of degrees of freedom)

q_i : genelleştirilmiş koordinat setleri ; $i=1,2,3,...,n$

Lagrangian fonksiyonu oluşturmak için toplam kinetik enerjiden sistemin toplam potansiyel

enerjisinin çıkartılması gerekmektedir.

Hareket Halindeki Sistemin Kinetik Enerji Modeli:

Arabanın Kinetik Enerjisi:

$$T_1 = \frac{1}{2} M \left(\frac{dx}{dt} \right)^2 = \frac{1}{2} M (\dot{x})^2 \quad (2.3)$$

Sarkaç Yükünün Kinetik Enerjisi:

$$T_2 = \frac{1}{2} m (\dot{x}_1^2 + \dot{y}^2) \quad (2.4)$$

Burada

$$x_1 = x + l \sin \theta, \quad y = l \cos \theta \quad (2.5)$$

$$\dot{x}_1 = \dot{x} + l \dot{\theta} \cos \theta, \quad \dot{y} = -l \dot{\theta} \sin \theta \quad (2.6)$$

araba + sarkaç kinetik enerjisi toplamı

$$E_K = T_1 + T_2 = \frac{1}{2} M (\dot{x})^2 + \frac{1}{2} m \left[(\dot{x} + l \dot{\theta} \cos \theta)^2 + (-l \dot{\theta} \sin \theta)^2 \right] \quad (2.7)$$

olarak bulunur. Buradan:

$$T = E_K = \frac{1}{2} (M + m) \dot{x}^2 + m x l \dot{\theta} \cos \theta + \frac{1}{2} m \left[(l \dot{\theta})^2 (\cos^2 \theta + \sin^2 \theta) \right] \quad (2.8)$$

elde edilir.

Sistemin Potansiyel Enerjisi:

Araba sadece yatay ekseninde hareket ettiği için, sistemin potansiyel enerjisi sadece sarkaç açısından kaynaklanmaktadır.

$$U = E_p = mgl - mgl \cos \theta \quad (2.9)$$

Lagrangien Denklemleri:

Lagrangian eşitliğinde bulduğumuz enerji denklemlerini yerlerine yazarsak denklem aşağıdaki forma kavuşacaktır:

$$L = T - U = \frac{1}{2}(M + m)\dot{x}^2 + m\dot{x}l\dot{\theta}\cos\theta + \frac{1}{2}m\left[l\dot{\theta}\right]^2(\cos^2\theta + \sin^2\theta) - mgl(1 - \cos\theta) \quad (2.10)$$

Sistemde arabaya ve salınım dinamiklerine etki eden sadece bir adet dış kuvvet bulunmaktadır. Bu da sisteme yatay ekseninde hareket kazandıran F_x kuvvetidir. θ koordinat sistemine herhangi bir dış kuvvet direkt olarak etki etmemektedir.

Ayrıca simülasyon çalışmasında 2 serbestlik derecesi için de sürtünme kayıpları göz ardı edildiğinden $D=0$ alınmıştır.

O halde (2.2) eşitliği aşağıdaki şekli alır:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x} = F \quad (2.11)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} = F_{\theta} = 0 \quad (2.12)$$

yazılabilir.

q(t)=θ(t) Serbestlik Derecesi için Lagrange Denklemi:

$$\frac{\partial L}{\partial \dot{\theta}} = ml^2\dot{\theta} + m\dot{x}l\cos\theta \quad (2.13)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) = -ml^2\ddot{\theta} + m\ddot{x}l\cos\theta - m\dot{x}l\dot{\theta}\sin\theta \quad (2.14)$$

$$\frac{\partial L}{\partial \theta} = -m\dot{x}l\dot{\theta}\sin\theta - mgl\sin\theta \quad (2.15)$$

Lagrange denkleminde yerine yerleştirildiğinde:

$$l\ddot{\theta} + \ddot{x}\cos\theta + g\sin\theta = 0 \quad (2.16)$$

elde edilir.

q(t)= x(t) Serbestlik Derecesi için Lagrange Denklemi:

Arabaya uygulanan kuvvet $F_x = F(t)$;

$$\frac{\partial L}{\partial \dot{x}} = (M + m)\dot{x} + ml\dot{\theta}\cos\theta \quad (2.17)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) = (M + m)\ddot{x} + ml\ddot{\theta}\cos\theta - ml(\dot{\theta})^2 \sin\theta \quad (2.18)$$

$$\frac{\partial L}{\partial x} = 0 \quad (2.19)$$

Lagrange denkleminde yerine yerleştirildiğinde:

$$(M + m)\ddot{x} + ml\ddot{\theta}\cos\theta - ml(\dot{\theta})^2 \sin\theta = F(t) \quad (2.20)$$

elde edilir.

2.2 Çalışma Noktası Yakınında Lineerleştirme

Elde edilen model non-lineer dir. Sistem üzerinde daha fazla çalışma yapmadan önce sistemin lineerleştirilmesi gerekmektedir.

Sarkacın denge noktası θ_0 etrafında küçük bir değişim gösterdiğini kabul edelim. Yeni değerine θ adını verelim:

$$\theta = \theta_0 + \varepsilon \quad (2.21)$$

$$\dot{\theta} = \dot{\varepsilon} \quad (2.22)$$

Taylor Serisi açılımından, θ 'nın herhangi bir fonksiyonunun birinci derece yaklaşımı

$$f(\theta) \approx f(\theta_0) + \varepsilon \left. \frac{df}{d\theta} \right|_{\theta_0} \quad (2.23)$$

olup, aynı zamanda daha büyük dereceli terimler ihmal edildiğinden

$$\dot{\varepsilon}^2 \approx 0 \quad (2.24)$$

$\dot{\varepsilon}^2 \approx 0, \theta=0$ için :

$$\begin{aligned} \cos\theta &\approx \cos(0) + \theta[-\sin(0)] = 1 \\ \sin\theta &\approx \sin(0) + \theta[\cos(0)] = \theta \end{aligned} \quad (2.25)$$

yazılabilir.

Sonuçları elde ettiğimiz Lagrange eşitliklerinde yerlerine koyarsak:

$$\left. \begin{aligned} (M+m)\ddot{x} + ml\ddot{\theta} &= F(t) \\ \ddot{x} + I\ddot{\theta} + g\theta &= 0 \end{aligned} \right\} \text{Lineer Diferansiyel Denklemleri elde edilir.} \quad (2.26)$$

Düzenlenirse:

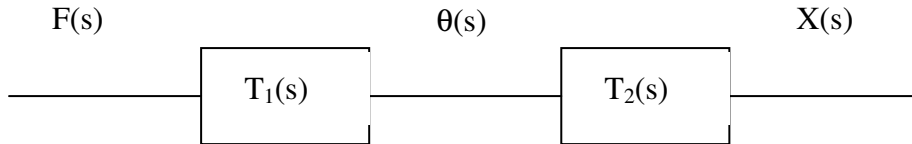
$$\left. \begin{aligned} \ddot{\theta}(t) &= -g \frac{(M+m)}{Ml} \theta(t) - \frac{1}{Ml} F(t) \\ \ddot{x}(t) &= g \frac{m}{M} \theta(t) + \frac{1}{M} F(t) \end{aligned} \right\} \text{bulunur.} \quad (2.27)$$

2.3 Sistemin Transfer Fonksiyonu

Diferansiyel denklemlerden, $\theta(t)$ nin $F(t)$ ile; $x(t)$ nin ise hem $\theta(t)$ hem de $F(t)$ ile bağlantılı olduğunu görüyoruz.

Bu prosesi kaskat bağlı iki transfer fonksiyonu ile göstermek mümkündür:

$$T_1 = \frac{\theta(s)}{F(s)} \text{ ve } T_2 = \frac{X(s)}{\theta(s)} \text{ olarak aldığımızda} \quad (2.28)$$



Şekil 2.2 Kaskat halde transfer fonksiyonu gösterimi

(2.27) denkleminin Laplace Transformu alındığında;

$$s^2 \theta(s) + g \frac{(M+m)}{Ml} \theta(s) = -\frac{1}{Ml} F(s) \quad s^2 X(s) = g \frac{m}{M} \theta(s) + \frac{1}{M} F(s) \quad (2.29)$$

Buradan

$$T_1(s) = \frac{\theta(s)}{F(s)} = \frac{-1}{Mls^2 + g(M+m)} \quad (2.30)$$

$$T_2(s) = \frac{X(s)}{\theta(s)} = -\frac{g + ls^2}{s^2} \quad (2.31)$$

elde edilir.

2.4 Durum-Uzay (State-Space) Gösterimi

Vinç sistemini durum uzay modeli şeklinde göstermemizin sebebi; sistemi matrisel forma sokmak istememizdir.

Durum uzay gösterimi

$$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu} \quad \mathbf{y} = \mathbf{Cx} + \mathbf{Du} \quad (2.32)$$

biçimindedir.

Burada durum vektörü $\mathbf{X}$ aşağıdaki gibi seçilebilir:

$$x_1 = x, \quad x_2 = \dot{x}, \quad x_3 = \theta, \quad x_4 = \dot{\theta}$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} \quad (2.33)$$

Elde ettiğimiz düzenlenmiş lineer diferansiyel denklemler kullanılarak sistemin durum denklemleri elde edilir:

$$\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & gm/M & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -g(M+m)/Ml & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/M \\ 0 \\ -1/Ml \end{bmatrix} F(t) \quad (2.34)$$

$$\begin{bmatrix} x \\ \theta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} F(t) \quad (2.35)$$

Simülasyon için $M = 10$ kg, $m = 5$ kg, $l = 1$ m, ve de $g = 10$ g/cm³ alınmış olup elde edilen matrisler;

$\dot{x} = Ax + Bu$ ve $y = Cx + Du$ formunda düzenlenerek,

$$\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -15 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0.1 \\ 0 \\ -0.1 \end{bmatrix} u \quad (2.36)$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u \quad (2.37)$$

halini almıştır. Matlab ile bu sistem aşağıdaki komut satırı ile elde edilebilmektedir:

% Açık çevrim durum-uzay sürtünmesiz vinç sistemi

M = 10; m = 5

l = 1; g = 10;

k1 = m*g/M;

k2 = -(M+m)*g/(M*l);

k3 = 1/M;

k4 = -1/(M*l);

% Durum-uzay matris gösterimi

A = [0 1 0 0 ; 0 0 k1 0 ; 0 0 0 1 ; 0 0 k2 0]

B = [0; k3; 0; k4]

C = [1 0 0 0; 0 0 1 0]

D = 0;

G = ss(A, B, C, D)

Böylece sistem G ile temsil edilerek durum uzayında ifade edilmiş olur. Sistemi darbe ve basamak cevapları ile test etmek için impulse(G), step(G) gibi komutlardan yararlanılabilir.

3. KONTROL SİSTEMLERİ

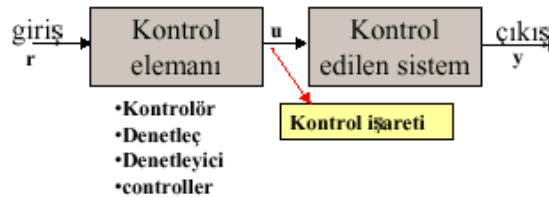
Bir sistemin davranışının istenilen bir iş akışı doğrultusunda çalışmasını sağlamak amacı ile yapılan çalışmalara "kontrol" denir. Davranışları kontrol eden sistemler ise "Kontrol Sistemleri" olarak adlandırılırlar. Dinamik sistemlerin kontrolünün temelini oluşturan geri besleme kavramının bilincine 19. yüzyılın sonunda varılmış olmasına rağmen eski çağlardan beri bilindiği ve başarı ile uygulandığı anlaşılmaktadır (Kuo,1992).

Bir kontrol sistemi genel anlamda üç temel öğeden oluşur;

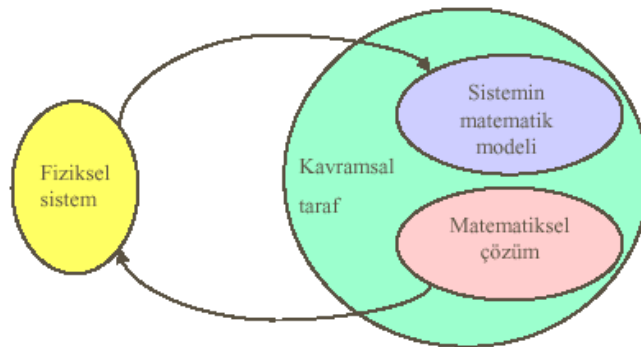
- Giriş Birimleri (amacı anlatanlar)
- Kontrol Sistemi Elemanları (amacı uygulayacak birimler)
- Çıkış (sonuç)

Kontrol Sistemlerinin açıklanışı;

- **Kontrol edilen sistem:** Çıkışları kontrol edilecek sistem
- **Kontrol Elemanı:** Kontrol edilen sistemin verilen amaca yönelik çıkışları üretmesi için gerekli olan kontrol işaretlerini üreten elemanlar topluluğu (genelde elektronik bir devre ve/veya bu devre üzerinde döngüsel çalışan programlardır).



Şekil 3.1 Temel bir kontrol sistemi



Şekil 3.2 Kontrol sistem mimarisi

3.1 Matematiksel Modelleme

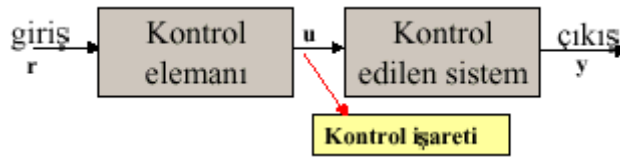
Sistemi oluşturan elemanlar, bu elemanların birbiri ile nasıl bağlı olduğu, her bir elemana ilişkin denklemler sistemin matematiksel modelini oluşturur. Bilinen fizik, kimya vb. kanunlar kullanılarak sistem içerisindeki işaretlerin birbiri ile ilişkileri matematiksel olarak ifade edilir. Özellikle sistemin giriş işareti ile çıkış işareti arasındaki bağıntı bizim için özel anlam ifade etmektedir (Transfer Fonksiyonu).

3.2 Kontrol Yöntemleri

Matematik modelini belirlediğimiz bir sistemi kontrol etmek için kullanılabilecek kontrol yöntemleri açık çevrim ve kapalı çevrim olarak iki ana gruba ayrılır.

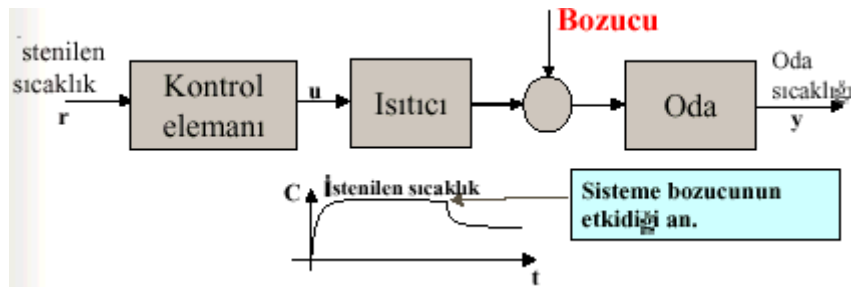
3.2.1 Açık Çevrim (Open-Loop) Kontrol Yöntemleri

Kontrol işareti çıkış işaretinden etkilenmeyen sistemlere açık çevrim kontrol sistemleri denir. Örnek olarak klasik çamaşır makinesi, trafik lambaları verilebilir.



Şekil 3.3 Açık çevrim kontrol sistemi

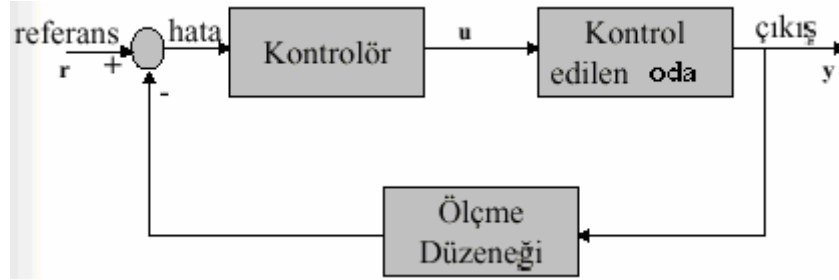
Açık çevrim kontrol sistemi; sistemde oluşan değişiklikler veya sisteme etkiyen bozucu faktörler nedeni ile sistemin çıkışında oluşabilecek etkileri düzeltemez. Örneğin; aşağıda temsil edilen oda sıcaklık kontrolü;



Şekil 3.4 Açık çevrim sistemle oda sıcaklık kontrolünün yapılması

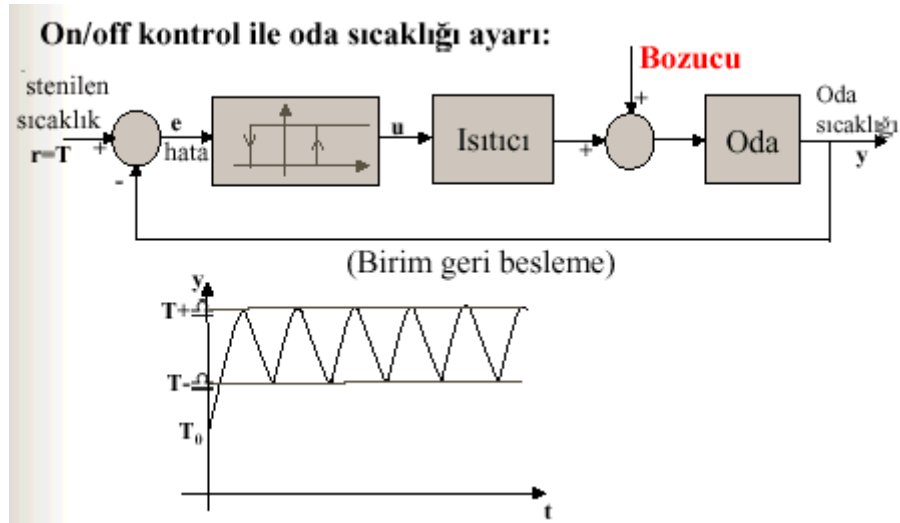
3.2.2 Kapalı Çevrim (Closed-Loop) Kontrol Yöntemleri

Sisteme etkiyen kontrol işaretinin sistem çıkışının da göz önüne alınarak üretildiği kontrol sistemlerine kapalı çevrim(geri beslemeli) kontrol sistemleri denir. Çıkışı ise aşağıda görüldüğü gibi sistem girişini etkilemesine ise geri besleme (feedback) denir.



Şekil 3.5 Kapalı çevrim kontrol sistemi

Şimdi aynı oda sıcaklığı kontrol örneğini kapalı çevrim ile yapalım;



Şekil 3.6 Kapalı çevrim sistemle oda sıcaklık kontrolünün yapılması

3.3 Kontrol Sistemlerinde Kararlılığın İncelenmesi

Kurulan modellere göre transfer fonksiyonu elde edilen bir sistemin dinamik davranışı geçici-durum cevabından saptanır. Denetim sistemlerinin, kararlılık, hızlı cevap ve küçük kalıcı hal hatası şeklinde asgari üç özelliği sağlaması gerekir. Bir denetim sisteminin dinamik davranışının en önemli özelliği kararlılığıdır. Kararlı sistem, sınırlı girişe sınırlı bir cevap veren sistem olarak tanımlanır. Diğer bir deyişle, eğer sistem sınırlı bir referans girişi ve

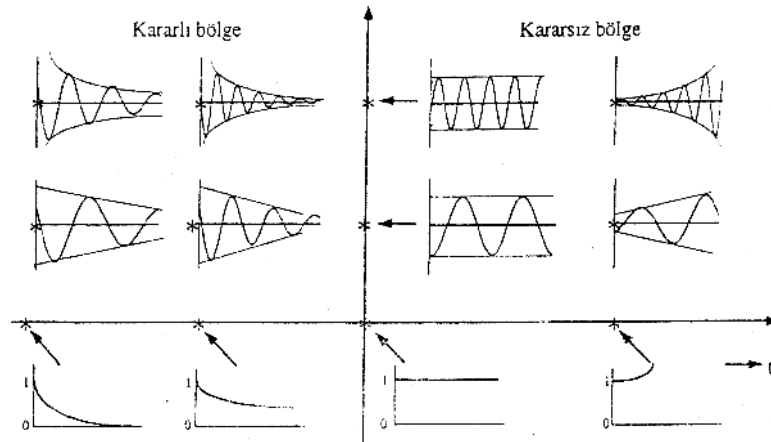
bozucu giriş karşısında sınırlı büyüklükte bir cevap veriyorsa kararlıdır denir. Kararlı bir sistem, bir bozucu giriş karşısında geçici-durum davranışı gösterdikten sonra tekrar denge konumuna dönen sistemdir.

3.3.1 s-Domeni Yaklaşımı

Kapalı döngü bir sistemin kararlılığı transfer fonksiyonu kutuplarının karmaşık düzlemdeki (s -alanı) yerlerine göre saptanabilir. Sistemin kutupları ise transfer fonksiyonun paydası olan karakteristik denklemin kökleri olarak bulunur. Bir geri beslemeli sistemin kararlı olabilmesi için gerek ve yeter şart sistem transfer fonksiyonu kutuplarının negatif gerçel kısımlara sahip olmasıdır. Transfer fonksiyonu kutuplarının karmaşık düzlemdeki yeri sistemin kararlılığı yanında sistemin dinamik davranışını da tanımlar.

Arzu edilen kararlı bir cevaptır. Bu nedenle bir sistemin mutlak olarak kararlı olabilmesi için tüm kutuplarının sol yarı s -düzlemi içinde yer alması gerekir. Kutuplardan bir tanesi dahi sağ yarı düzlemde yer alırsa sistem kararsız olur.

Bir geri beslemeli denetim sisteminin kararlılığının belirlenebilmesi için transfer fonksiyonunun paydası olan karakteristik denklemin köklerinin belirlenmesi gerekir. 's' in bir polinomu olan karakteristik denklemin derecesi küçük ise kökler kolaylıkla belirlenir. Buna karşılık polinomun derecesi büyüdükçe köklerin bulunması zorlaşır. Diğer taraftan sadece sistemin kararlı olup olmadığının bulunması gerektiğinde karakteristik denklemin köklerinin işaretinin belirlenmesi gereği kadar bilgi sağlayabilir. Bu durumda köklerin bulunmasına gerek kalmadan kararlılığın incelenmesi yeterli olabilir. Karmaşık sayı düzleminde köklerin bulunmasına gerek kalmadan kararlılığın incelenmesi Routh-Hurwitz kararlılık ölçütü ile yapılabilmektedir.

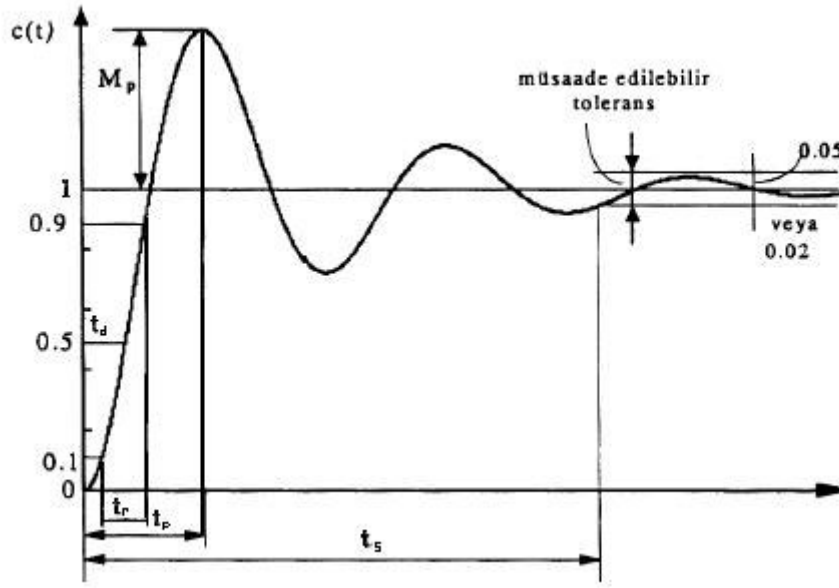


Şekil 3.7 Köklerin s -düzlemindeki yerlerine bağlı olarak değişen ani darbe cevap eğrileri

3.3.2 Zaman Domeni Yaklaşımı

Sistemlerin test edilebilmesi için bazı önemli giriş fonksiyonları tanımlanmıştır. Bu giriş fonksiyonları yardımıyla matematiksel çözümler gerçekleştirilebilir. Zaman tanım bölgesi davranışı birim basamak cevabına göre incelenebilir. İkinci mertebeden birim geri beslemeli bir sistemin zaman tanım bölgesi cevabının tipik bir ifadesi ve grafiği aşağıda verilmiştir.

$$c(t) = 1 - \frac{1}{1 - \zeta^2} e^{-\zeta \omega_n t} \sin(\sqrt{1 - \zeta^2} \omega_n t + \cos^{-1} \zeta) \quad (3.1)$$



Şekil 3.8 Zaman alanı cevabı parametreleri

Zaman cevabının incelenmesinde kullanılan parametreler grafik üzerinde gösterilmiştir. Bunların tanımları aşağıdaki gibidir.

Gecikme Zamanı, t_d : Gecikme zamanı, cevabın nihai değerinin yarısına ilk defa ulaşması için geçen zamandır. Birinci dereceden gecikmeli sistemlerde gecikme zamanı, t_c zaman sabiti T ya eşit olarak alınabilir.

Yükselme Zamanı, t_r : Yükselme zamanı, cevabın nihai değerinin %10 dan %90 nına, %5 den %95 ine veya %0 dan %100 üne kadar ulaşması için geçen zamandır. Aşırı sönümlü birinci dereceden sistemler için %0-100 yükselme zamanı kullanılır. Salınlı sönümlü sistemlerde ise genel olarak % 10-90 yükselme zamanı kullanılır.

Tepe Zamanı, t_p : Tepe zamanı cevabın nihai değerini ilk defa aşarak bir tepe yaptığı noktaya erişmesi için gerekli zamandır.

Maksimum Aşma, M_p : Cevap eğrisinin $t = t_p$ iken aldığı genlik değeridir. Eğer cevabın nihai kalıcı-durum değeri birim değerden farklı ise, aşağıdaki formül ile tanımlanan maksimum aşmanın yüzde değeri kullanılır.

$$\text{Maksimum Aşım Yüzdesi} = \% \frac{C(t_p) - C(\infty)}{C(\infty)} \times 100 \quad (3.2)$$

Maksimum aşmanın miktarı doğrudan doğruya sistemin bağıl kararlılığını belirlenir. Birinci dereceden gecikmeli sistemlerde cevap eğrisi hiçbir zaman olması gerektiği nihai değeri aşmadığından maksimum aşma tanımlanmaz, sıfırdır.

Oturma Zamanı, t_s : Oturma zamanı, cevap eğrisindeki salınım genliklerinin müsaade edilebilir tolerans değeri sınırlarına erişmesi için geçen zamandır. Müsaade edilebilir tolerans değerleri ise genellikle nihai değer %5 veya %2 lik aşma değerleri olarak tanımlanır. Oturma zamanı denetim sisteminde tanımlanan en büyük zaman sabitidir. Birinci dereceden gecikmeli sistemlerde oturma zamanının yükselme zamanına eşit olduğu kabul edilebilir.

Sönüm Oranı (ζ): Sisteme herhangi bir giriş uygulandıktan sonra meydana gelen geçici hal sönür ve geriye kararlı hal hataları kalır, sisteme genellikle basamak, rampa ve parabolik test fonksiyonları kullanılır. Sönüm oranı (ζ), kararlı sistemler için $0 < \zeta < 1$ arasındadır. ζ küçüldükçe aşım artar.

Yukarıda tanımlanan zaman alanı-cevabı ile ilgili özellikler sistemlerin uygun cevap hızlarına göre tasarımlarında büyük önem arz ederler. Eğer t_d , t_r , t_p , M_p ve t_s değerleri belirlenebilirse sistemin cevap eğrisinin biçimi saptanabilir.

Burada tanımlanan tüm özelliklerin verilen herhangi bir duruma uygulanması gerekli değildir. Örneğin aşırı sönümlü ikinci derece ve birinci derece sistemler için tepe zamanı ve maksimum aşma tanımları geçerli değildir.

Çok düşük genlikli titreşimlerin dahi müsaade edilmediği belli uygulamalar dışında, sistemin geçici durum cevabının yeteri kadar hızlı ve yeteri kadar sönümlü olması gerekir. Buna göre ikinci dereceden bir sistemden arzu edilen bir geçici durum cevabı elde edilebilmesi için sönüm oranının 0.4 ile 0.8 arasında olması gerekir. 0.4 den küçük sönüm oranı değerlerinde geçici durum cevabında aşırı büyüklükte bir aşma değeri ve buna karşılık sönüm oranının değerinin 0.8 den büyük olduğu durumlarda ise sistem cevabı çok yavaştır.

Gerçekte ise maksimum aşma değeri ile oturma zamanı değeri birbirine göre zıtlık teşkil eder.

Diğer bir deyişle sistemin aynı anda hem maksimum aşma ve hem de oturma zamanı değerleri küçük tutulamaz. Eğer bunlardan birisi küçük tutulacak olursa diğerinin büyük tutulması gerekir. Bir kontrol sisteminde, zaman domeninde, $c(t)$ çıkışının; yükselme süresi, gecikme zamanı, yerleşme zamanı ve maksimum aşım değerlerinin küçük olması istenir.

Bunların dışında kalıcı hal hatasının küçük kalması da kontrol sisteminden beklenen en büyük özelliklerden bir tanesidir. Giriş ile kalıcı hal çıkışı arasındaki bu küçük farka kalıcı veya sürekli hal hatası denir ve e_{ss} ile gösterilir. Girişi $r(t)$ çıkışı $y(t)$ ile tanımlayacak olursak ;

$$e(t) = r(t) - y(t) \quad (3.3)$$

şeklinde ifade edilir. Sürekli hal hatası son değer teoreminden yararlanılarak;

$$e_{ss} = \lim_{s \rightarrow \infty} E(s) \text{ şeklinde tanımlanır (Kuo,1992, Ogata,1994, Nise,2000).}$$

Bir kontrol sisteminin optimum kontrol sistemi olarak tanımlanabilmesi için sistem parametreleri ve de performans indeksi en uygun değeri almalıdır. En uygun değer genelde minimum olduğu değerlerdir. İndeks her zaman pozitif veya sıfır bir sayıdır (Dorf and Bishop,2001).

Kesin bir çalışma kriteri olmamakla birlikte, hata fonksiyonu $e(t)$ ' nin minimum yapılmasının, sürekli hal hatasını ve oturma zamanını da küçültebileceği ya da minimum yapacağı söylenebilir. Ancak hata fonksiyonunu minimum yapmak için literatürde birçok önerilerde bulunulmuştur.

Bunlardan ilk akla gelen hatanın integralinin minimum yapılmasıdır. Ancak hata fonksiyonu sadece pozitif değil negatif değerler de alabileceği için hatanın mutlak değerinin integralinin minimum yapılması gerekir.

Burada minimum sistem parametrelerine göre yapılır. Parametrelere göre I' nın minimum yapılmasına parametre optimizasyonu denir (Sarıoğlu,1999).

ISE; hatanın karesinin integralidir ve genelde analitik olarak hesaplanır ve hatanın performansı hakkında bilgi verir.

ISE genelde büyük hataların yüksek derecede önemli olduğu süreçlerde kullanılır. Bu algorithmada büyük hatalar daha büyük ISE değerlerine götürür ve hataların küçülmesini sağlar. ISE (3.4) formülüyle hesaplanır:

$$ISE := \int_0^T e(t)^2 dt \quad (3.4)$$

IAE; hatanın mutlak değerinin integralidir ve bilgisayar simülasyonları için uygun bir kriterdir. Sistem için küçük hatalar çok önemliyse bu durumda IAE kullanılmalıdır; çünkü IAE küçük hata-büyük hata ayırımı yapmaz. IAE şu şekilde bulunur:

$$IAE := \int_0^T |e(t)| dt \quad (3.5)$$

ITAE; hatanın mutlak değeri ile zamanın çarpımının integralidir. Girişteki büyük başlangıç hatalarının etkisini azaltmak ve sistem cevabına etkisini geciktirmek için ITAE kullanılmalıdır. İntegralin içindeki t çarpanı tüm sapmaların etkisini genişletir. ITAE şekildeki gibi hesaplanır:

$$ITAE := \int_0^T t |e(t)| dt \quad (3.6)$$

Hatanın karesi ile zaman çarpımının integrali olan ITSE' nin kullanım amaçları ITAE ile hemen hemen aynıdır ve aşağıdaki gibi hesaplanır:

$$ITSE := \int_0^T t e(t)^2 dt \quad (3.7)$$

Bu 4 performans indeksini karşılaştıracak olursak IAE, ISE' ye göre daha geniş sapmayı yani daha küçük maksimum aşımı sağladığını, ISE' nin yerleşme zamanının daha kısa olduğunu ve ITAE' de hatanın daha ağır bir biçimde meydana geldiğini söyleyebiliriz.

3.3.3 Frekans Domeni Yaklaşımı

Bir sistemin frekans cevabı, sistemin sinüzoidal giriş sinyaline gösterdiği sürekli hal cevabı olarak tanımlanır. Sistemin çıkış cevabı, geçici hali ortadan kalktıktan sonra, sadece genlik ve faz bakımından yine sinüzoidal sinyal olmak üzere girişten farklıdır. Frekans domeninde sistem cevapları Bode ve Nyquist diyagramları ve Nichols grafikleri ile bulunabilir. Bu yöntemde transfer fonksiyonunda 's' yerine 'jω' koyarak, sürekli hal transfer fonksiyonu elde edilebilir. Elde edilen transfer fonksiyonunun hem büyüklük hem de faz açısı olmak üzere iki unsuru vardır. Frekans cevabının hem zaman hem de s-domeni yaklaşımına üstünlükleri vardır. Sistem parametre değerlerinin değişim etkilerini belirlemek zaman domeninde çok

zorken frekans domeninde kolay olması, Nyquist kararlılık yoluyla bağıl kararlılığında incelenebilmesi üstünlüklerinden birkaçıdır. Geçici durum cevabı ile frekans cevabının bazı parametreleri arasındaki dolaylı ilişki biliniyorsa frekans cevabı tercih edilebilir (Muzoğlu,2005).

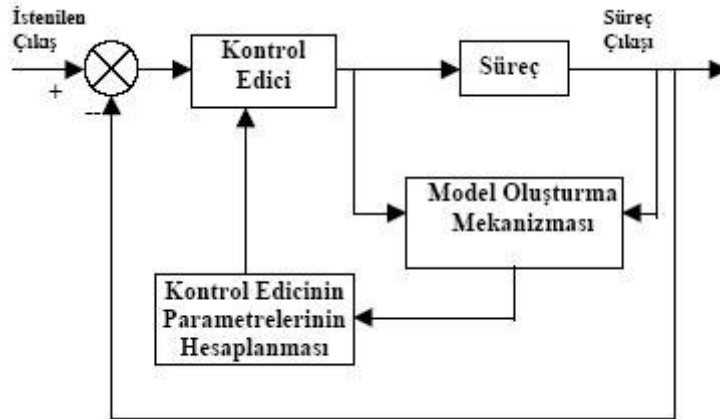
4. PID KONTROLÖRLER

Hali hazırda üç terimli kontrol edici olan PID kontrol ediciler şu an için endüstride kullanılan otomatik geri beslemeli kontrol cihazlarının %90' nında kullanılmaktadır (Sarioğlu,1999). Geçmişte bunlar frekans analiz metotları yardımı ile ayarlanır iken şu an modern yaklaşımda PID kontrol edicinin ayarları süreç modeline bağlı olarak oluşturulmaktadır.

Kolayca gösterilebileceği gibi PI (oransal + integral) kontrol edici birinci dereceden doğrusal zamanla değişmeyen sistemler için optimum bir çözüm sunmaktadır. Benzer olarak da PID (oransal + integral + türevsel) kontrol edici de ikinci dereceden doğrusal zaman gecikmesi içermeyen sistemler için optimum çözüm sunmaktadır.

Fakat gerçek hayattaki sistemler çoğu zaman doğrusal değildir ve zamanla süreç karakteristikleri değişiklik göstermektedir. Dolayısıyla doğrusal bir model için seçilmiş başlangıç kontrol edici sistemin zaman ile değişerek farklı bir bölgede işlemeye başladığı durumlarda uygulanabilirlikten çıkacaktır. Buna bir çözüm birden fazla kontrol edici tanımlamalarının daha önceden hafızaya alınması veya bir yerde tutulması ve süreç çalışma bölgesini değiştirdikçe buna uygun kontrol edicinin devreye girmesinin sağlanmasıdır. Bu strateji parametrik- veya kazanç- tarifli kontrol olarak adlandırılır ve çalışma bölgesinin değiştiği süreçlerde oldukça yaygın bir kullanım alanı bulmaktadır.

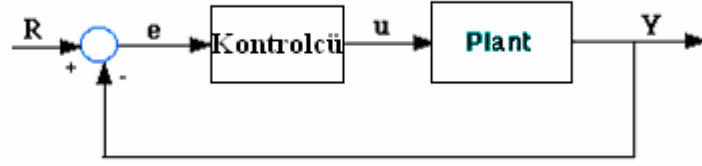
Daha zarif bir teknik ise kontrol ediciyi adaptif olacak şekilde ayarlamaktır. Bu şekilde doğrusal modelin parametreleri o anki sürecin karakteristiklerine bağlı olarak sürekli güncellenecek, yenilenecektir. Buna ilişkin olarak, parametrelerin nasıl kontrol edicinin ayarlarının hesaplanmasında kullanılabileceği Şekil 4.1' de görülmektedir.



Şekil 4.1 Adaptif kontrol edicinin basit bir şematik gösterimi (Gündoğdu,2005)

Bu yöntemden kontrol edicinin ayarları sürekli olarak süreç modelindeki değişimlere bağlı olarak güncellenmektedir. Bu gibi yapılar genelde otomatik-ayarlı / adaptif / kendinden-ayarlı gibi tanımlamalar ile nitelendirilir.

PID (*Proportional-Integral-Derivative*: Oransal-Integral-Türev) kontrol ediciler geri beslemeli kontrol edicilerde en çok kullanılan algoritmalardır. Aşağıda temel geri beslemeli bir sistem görülmektedir.



Şekil 4.2 Geri beslemeli kontrol sistemi

Burada plant; kontrol edilecek sistemdir ve denetleyici bu sistem için uyarıyı sağlayıp sistemi denetlemek için tasarlanmıştır.

PID Kontrol edicinin transfer fonksiyonu denklem (4.1) deki formdadır.

$$G(s) = \frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s \quad (4.1)$$

Burada K_p , K_i , K_d , sırası ile oransal, integral ve türev kazançlarıdır. $U(s)$ kontrol edicinin çıkışı, $E(s)$ de hata sinyali ve kontrol edicinin girişidir. Değişken (e) izleme hatasını, istenilen giriş değeri (R) ile gerçek çıkış değeri (Y) arasındaki farkı gösterir. Bu hata sinyali (e), PID denetleyiciye gönderilir ve denetleyici bu hata sinyalinin hem türevini hem de integralini hesaplar. Denetleyicinin ürettiği (u) sinyali, oransal kazanç (K_p) ile hata değeri, integral kazancı (K_i) ile hatanın integrali, türevsel kazanç (K_d) ile hatanın türevi çarpımlarının toplamına eşittir. Bu (u) sinyali denetlenen sisteme (plant) gönderilir ve yeni çıkış (y) elde edilmiş olur. Bu (y) çıkış sinyali algılayıcıya geri gönderilerek yeni hata sinyali (e) bulunur. Denetleyici yeni hata sinyaline aynı işlemleri uygular ve bu işlem böyle devam eder. Yukarıdaki eşitlik bir başka formda;

$$\frac{U(s)}{E(s)} = K_p \cdot \left[1 + sT_d + \frac{T_i}{s} \right] \quad (4.2)$$

olarak ifade edilebilir, burada $T_i = K_p / K_i$ ve $T_d = K_d / K_p$ dir. T_i ve T_d integral ve türev

zaman sabitleri olarak isimlendirilirler ve K_p de kontrol edici kazancıdır. Bazı kontrol ediciler kontrol edici kazancı yerine oransal band ayarını kullanmaktadır, $K_p = 100 / PB$ gibi, burada PB oransal ayardır ve 'kontrol edici çıkışı %100 değiştiği zamanki yüzde hata miktarı' olarak tanımlanır. Türevsel terimden dolayı frekansın artımı ile kontrol edici kazancı da artmaktadır. Bu yüzden pratik PID kontrol ediciler bu yüksek frekanslardan kaynaklanan kazançları birinci dereceden bir ön eleman (alçak geçiren süzgeç) kullanarak limitler. Bu filtrenin zaman sabiti τ genelde türev terimi kazancının onda biri mertebesinde seçilir. Pratikte uygulanan PID kontrol edicinin transfer fonksiyonu da;

$$\frac{U(s)}{E(s)} = \frac{K_p s^2 + K_d s + 1}{s(\tau K_d s + 1)} \quad (4.3)$$

veya

$$\frac{U(s)}{E(s)} = K_p \left[1 + \frac{T_i}{s} \right] \left[\frac{1 + T_d s}{1 + \tau s} \right] \quad (4.4)$$

ile ifade edilir.

PID kontrol edici tasarımının temel hedefi, K_p , K_d , K_i veya K_p , T_i , T_d katsayılarının bulunması ve verilen kapalı döngü sistemin performans koşullarını sağlayacak şekilde ayarlanmasıdır. Bu kazançlar sistemin geçici başlangıç tepkisi, gürültü yok edimi ve kalıcı hal hata özellikleri yardımı ile bulunur. Pratikte bu üç kavramın hepsini aynı anda kullanmak çok mümkün değildir. Örnek olarak eğer kontrol edici ayar noktası değişiminde daha iyi bir geçici sistem tepkisi vermek üzere ayarlanmış ise (minimum üst aşım ve minimum oturma zamanı için) genelde gürültü değişimlerine daha ağır tepki verir.

PID Kontrol edici tasarımında izlenecek temel basamaklar:

- Sistemin açık döngü transfer fonksiyonu elde edilmesi ve nelerin geliştirilmesi gerektiğine karar verilmesi.
- Sisteme yükselme zamanını iyileştirmek için oransal denetleyici eklenmesi.
- Sistemde ani tepkinin iyileştirilmesi için türevsel denetleyicinin eklenmesi.
- Kararlı durum hatasını yok etmek için integral denetleyicisinin sisteme eklenmesi.
- Bütün K_p , K_i , K_d ' lerin istenilen genel tepkiyi verene kadar ayarlanması.

Son olarak unutulmamalıdır ki; her üç denetleyiciyi de her seferinde kullanmak zorunluluğumuz yoktur. Eğer PI denetleyici yeterince iyi sonuç veriyor ise türevsel denetleyiciyi kullanmaya lüzum yoktur.

Oransal denetleyicilerin (K_p), yükselme zamanını azaltmada etkisi vardır ve azaltır, ama kararlı hal hatasını tamamen yok etmez. İntegral denetleyicinin (K_i) kararlı hal hatasının azaltılmasında doğrudan etkisi vardır ancak bu, geçici tepkinin daha kötü olmasına sebep olabilir. Türevsel denetleyicinin (K_d) sistemin kararlılığının artmasında etkisi vardır, aşımı azaltır ve geçici tepkiyi düzeltir. Kapalı döngülü bir sistemde, K_p , K_d ve K_i 'ye göre her bir denetleyicinin etkisi aşağıdaki çizelgede özet olarak gösterilmiştir.

Çizelge 4.1 P, I ve D'nin karakteristikleri

Denetleyici	Yükselme zamanı	Sistem ani tepkisi	Oturma zamanı	Kararlı durum hatası
K_p	Azalır	Artar	Ufak Değişim Gösterir	Azalır
K_i	Azalır	Artar	Artar	Yok eder
K_d	Ufak Değişim Gösterir	Azalır	Azalır	Ufak Değişim Gösterir

Unutmamalı ki bu düzeltmeler tam olarak geçerli değildir. Çünkü K_p , K_i ve K_d birbirlerine bağımlıdır. Yani değişkenlerden birinin değişimi diğer ikisinin etkisini değiştirebilir. Bu yüzden çizelge K_i , K_p ve K_d değerlerinin belirlenmesinde sadece bir referanstır.

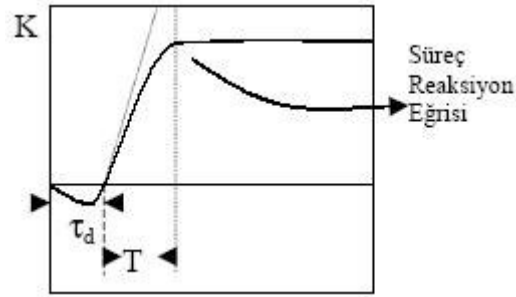
4.1 SISO PID Kontrolör Tasarım Metotları

4.1.1 Ziegler-Nichols Metodu

Ziegler-Nichols (Z-N) tasarım metodu PID kontrol edicilerin parametrelerinin belirlenmesinde kullanılan en yaygın metotlardan birisidir. Bu metotlardan birincisi sürekli döngü metodu olarak adlandırılır ve bu metot da kontrol edici kazancı, tam salınım sağlanana kadar arttırılır. Bu kazanç değeri K_u , salınımın periyodu T_u olarak kaydedilir ve PID kontrol edicinin parametreleri Z-N metoduna göre çizelge 4.2' deki gibi bunlara göre seçilir.

Çizelge 4.2 Z-N ayar kuralları (sürekli döngü için)

Kontrol Edici	Parametreler		
	K_p	T_i	T_d
P	$0.5 K_u$	-	-
PI	$0.45 K_u$	$(1/1.2) T_u$	-
PID	$0.6 K_u$	$0.5 T_u$	$(1/8) T_u$



Şekil 4.3 Açık çevrim basamak cevabı

Yine Z-N' nin önermiş olduğu ikinci bir yaklaşım, genelde süreç reaksiyon eğrisi metodu olarak bilinir. Bu metotta sistemin açık döngü birim basamak tepkisi incelenir ve çizelge 4.2' ye benzer bir tabloya göre K_p , T_i ve T_d değerleri seçilir.

Sistem tepkisinin yaklaşık olarak düzgün bir doğru ile ve τ_d , T ve K katsayıları ile ifade edilir. Bu tepki birinci dereceden zaman gecikmeli (FOPTD, *First Order Plus Time Delay*) denklem (4.5)' deki gibi bir sistemin basamak tepkisini vermektedir.

$$G(s) = \frac{K e^{-\tau_d s}}{Ts + 1} \quad (4.5)$$

Bu eğriye göre Z-N kuralları çizelge 4.3' de verilmiştir, kontrol edicinin parametreleri bu çizelgeye uygun bir şekilde seçilebilir.

Çizelge 4.3 Z-N ayar kuralları (süreç eğrisi metodu için)

Kontrol edici	Parametreler		
	K_p	T_i	T_d
P	$\frac{T}{K\tau_d}$	-	-
PI	$\frac{0.9T}{K\tau_d}$	$\frac{\tau_d}{0.3}$	-
PID	$\frac{1.2T}{K\tau_d}$	$2\tau_d$	$0.5\tau_d$

Z-N metodunun iki temel dezavantajı vardır. Sonuç kapalı döngü sistemi genelde istenenden daha çok salınıma sahiptir. Bir diğer dezavantajı ise dengelenmemiş sistemin kazanç ve frekansında bağlı olarak, oransal geri besleme döngüsünde salınıma sebep olmaktadır.

4.1.2 Cohen Coon (C-C) Metodu

Cohen ve Coon denklem (4.5)' deki gibi bir FOPTD (first-order plus time delay) sistem için kapalı döngü tepkisinin 1/4 oranında bir düşüş oranını sağlaması için geliştirdikleri tasarım metodudur. Bu metodun temel avantajı basitliğidir fakat en belirgin dezavantajı ise çıkış tepkisi için çok salınımlı yargısının konulmasıdır. Bu metot için PID kontrol edicinin parametreleri;

$$K_c = \frac{T}{K\tau_d} \left[\frac{16T + 3\tau}{12T} \right]$$

$$T_i = \tau_d \left[\frac{32 + 6\left(\frac{\tau_d}{T}\right)}{13 + 8\left(\frac{\tau_d}{T}\right)} \right] \quad T_d = \left(\frac{4\tau_d}{11 + 2\left(\frac{\tau_d}{T}\right)} \right) \quad (4.6)$$

denklem seti ile verilir (Kuo,1995).

4.1.3 İç Model Kontrolü

İç model kontrolü (IMC: *Internal Model Control*) diğer tasarım metotlarına nazaran daha yeni olarak 1980' lerin başlarında ortaya konulmuştur.

Morari, Rivera, Chien ve Chia tarafından IMC' de kullanılacak olan PID parametreleri seçimlerinin nasıl yapılacağı ortaya konulmuştur. Morari tarafından ulaşılan bazı sonuçlar çizelge 4.4' de verilmiştir buradaki η tasarım parametresidir. Bu model tabanlı yaklaşım, değişik modellere karşılık sunduğu çözümler ile daha iyi kontrol performansı sağlar.

Çizelge 4.4 IMC tabanlı PID kontrol edici parametreleri

Model	K_c	T_i	T_d
$\frac{K}{s}$	$\frac{1}{K\eta}$	-----	-----
$\frac{K}{Ts+1}$	$\frac{1}{K\eta}$	T	-----
$\frac{K}{s(Ts+1)}$	$\frac{1}{K\eta}$	-----	T
$\frac{K e^{-\tau_d s}}{Ts+1}$	$\frac{2\left(\frac{T}{\tau_d}\right)+1}{K\left[2\left(\frac{T}{\tau_d}\right)\right]+1}$	$\frac{\tau_d}{2}+1$	$\frac{1}{2\left(\frac{T}{\tau_d}\right)+1}$
$\frac{K}{(T_1s+1)(T_2s+1)}$	$\frac{T_1+T_2}{K\eta}$	T_1+T_2	$\frac{1}{2\left(\frac{T}{\tau_d}\right)+1}$
$\frac{K}{T^2s^2+2\zeta Ts+1}$	$\frac{2\zeta T}{K\eta}$	$2\zeta T$	$\frac{T}{2\zeta}$

4.1.4 Ho-Hang-Cao Metodu

W. K. Ho, C.C. Hang ve L.S. Cao [1995] in geliştirmiş olduğu kullanışlı PI / PID kontrol edicileri genlik kazancı ve faz aralığı özelliklerine dayanmaktadır. Bu metotda bazı basit yaklaşımlar ile grafiksel veya sayısal yöntemler kullanılmaktadır. Bahsi geçen yaklaşımda PI kontrol edicide FOPTD için denklem [4.5] deki model seçilmekte ve ikinci dereceden zaman gecikmeli sistemler için ise denklem [4.7] deki model seçilmektedir.

$$G(s) = \frac{K e^{\tau_d s}}{(T_1 s + 1)(T_2 s + 1)} \quad (4.7)$$

FOPTD modeli için yazarlar (W. K. Ho, C.C. Hang ve L.S. Cao) aşağıdaki PI ayarlarını genlik kazancı A ve faz aralığı cinsinden sunmuşlardır.

$$K_c = \frac{w_p T}{AK} \quad (4.8)$$

$$T_i = \left(2w_p - \frac{4w_p^2 \tau_d}{\pi} + \frac{1}{T} \right)^{-1} \quad (4.9)$$

ki burada;

$$w_p = \frac{A\theta + \frac{1}{2}\pi A(A-1)}{(A^2-1)\tau_d} \quad (4.10)$$

ikinci dereceden zaman gecikmeli sistem için model olarak denklem (4.7) yi kullanmaktayız ve bu sistem için PID kontrol edici ayarları;

$$K_c = \frac{w_p T_1}{AK}$$

$$T_i = \left(2w_p - \frac{4w_p^2 \tau_d}{\pi} + \frac{1}{T_1} \right)^{-1} \quad (4.11)$$

$$T_d = T_2$$

olarak verilmektedir. Burada sağlanan performans IMC yaklaşımı ile elde edilen performans ile kıyaslanabilecek düzeydedir (Ogata,2002).

4.2 MIMO PID Kontrolör Tasarım Metotları

MIMO (*Multi Input Multi Output*: Çok Girişli Çok Çıkışlı) sistemler için PID kontrol edici tasarımı bu tarz sistemlerin karmaşık yapısı yüzünden daha zordur. MIMO PID kontrol edici tasarımı sistem içinde tekil döngüler üzerine uygulanarak veya tüm sistemi bir arada düşünerek gerçekleştirilebilir.

4.2.1 Genel Ziegler-Nichols Metodu

Daha önce bahsi geçmiş olan (Bölüm 4.1.1) sürekli çevrim metodu MIMO sistemler için genişletilebilir ve bu metot Genel Ziegler-Nichols Metodu olarak isimlendirilir. Bu metot kısaca;

1. **n** tane kontrol değişkeni göreceli kontrol kalitesi için **n** tane **c_i** (i=1,2,...n) ağırlık faktörü seçilsin.
2. En iyi giriş-çıkış çifti tanımlaması kullanılarak P kontrol edicili sistemin kararlılığını

döngüler arasında aşağıdaki limitleri sağlaması temin edilmeli.

$$\frac{K_{c,j} G_{i,j}(0)}{K_{c,j+1} G_{i+1,j+1}(0)} = \frac{c_i}{c_{i+1}}, \quad i = 1, \dots, n-1 \quad (4.12)$$

ki burada $K_{c,i}$ P kontrol edicinin i . döngüdeki kazancı ve G_{ii} de diyagonal süreç kazancınıdır.

3. Verilen sistemin salınıma başladığı noktadan, salınım periyodu T_u dan kritik frekans olan w_c ve kritik kontrol edici kazancı olan $K_{u,i}$ bulunmalı.
4. Genel **Z-N** formüllerinin bulunduğu çizelge **4.5** den faydalananarak kontrol edicinin parametreleri α_i ye bağlı olarak;

$$\alpha_i = \frac{w_c}{w_{i,c}} \quad (4.13)$$

$w_{i,c}$ G_{ii} nin kritik frekansı olma üzere seçilir.

5. Kontrol kalitesinin istenilen düzeyde olup olmadığının kontrol edilmesi, eğer değil ise α_i değiştirilerek 2. adıma geri dönülmesi.

şeklinde özetlenebilir.

Çizelge 4.5 Genel Z-N kural tablosu

Kontrol Edici	Parametreler		
	K_c	T_i	T_d
P	$\alpha_1 K_{u,i}$	-	-
PI	$\alpha_2 K_{u,i}$	$0.8 T_u$	-
PID	$\alpha_3 K_{u,i}$	$0.5 T_u$	$0.12 T_u$
ki burada;	$0.5 \leq \alpha_1 \leq \sqrt{0.5}$ $0.45 \leq \alpha_2 \leq \sqrt{0.45}$ $0.6 \leq \alpha_3 \leq \sqrt{0.6}$		

4.2.2 Zhuang ve Atherton Optimizasyon Metodu

Bu metot ani geit tepkilerine genel Z-N metoduna, karakteristik kk ve diğerk metotlara oranla daha iyi cevap vermektedir. Bu metot daha ziyade TITO (*Two Input Two Output*: İki Girişli İki Çıkışlı) sistemler için uygulanmaktadır. Fakat diğerk MIMO sistemler için de genellenebilir.

TITO sistemlerde hata sinyali;

$$E(s) = R(s) - C(s) = [I + G(s)G_c(s)]^{-1} R(s) \quad (4.14)$$

şekinde yazılabilir, burada $R(s)$ ve $C(s)$ ler doğal olarak 2x1 lik giriş ve çıkış vektörleridir, $G(s)$ 2x2 lik sistem transfer matrisi ve $G_c(s)$ de kontrol edici sistemin transfer matrisidir. Yukarıdaki tanımlamadan hareketle $E(s)$ için;

$$E(s) = \begin{bmatrix} 1 + G_{11}(s)G_{c1}(s) & G_{12}(s)G_{c2}(s) \\ G_{21}(s)G_{c1}(s) & 1 + G_{22}(s)G_{c2}(s) \end{bmatrix}^{-1} R(s) \quad (4.15)$$

yazılır. Optimum PID ayarları, denklem (4.15) deki elemanlardan birisinin integral performans kriteri fonksiyonunun minimize edilmesi ile seçilir. Yani $E(s)$ hata sinyalini minimum edecek kontrol edici seçimi yapılır.

4.2.3 Seraji Metodu

Bu yöntem kutup atamasına dayalı olarak Seraji ve Tarkoh tarafından teklif edilmiştir. Bu metodun temel basamakları sırası ile;

1. Kontrol edilecek sistemin durum uzay gösterimi denklem (4.16)'da verilmiştir.

$$\begin{aligned} \dot{x} &= Ax + Bu + Ed \\ y &= Cx \end{aligned} \quad (4.16)$$

ki burada x ; $n \times 1$ lik durum vektörü, u ; $m \times 1$ lik kontrol vektörü, d ; $v \times 1$ lik bozucu etki (gürültü, hata) vektörü, y de $L \times 1$ lik çıkış vektörü olarak tanımlanır.

2. Aşağıdaki gibi yeni bir sistem için (AA BB CC DD) matris tanımlamaları yapılır ve bu sistemin durum değişkenlerini içeren matris de z olsun.

$$AA = \begin{bmatrix} A & 0 \\ 0 & 1 \end{bmatrix}, \quad BB = \begin{bmatrix} B \\ 0 \end{bmatrix}, \quad CC = \begin{bmatrix} C & 0 \\ 0 & 1 \end{bmatrix}$$

ki bu yeni tanımlanmış sistem gözlemlenebilir ve kontrol edilebilir olmalı.

3. Bu yeni sistemi, kontrolün başlangıç kurallarının uygulaması olan $u = u_c + Kz$ ile d6ng6sel hale getirelim ki burada K $m \times L$ ' lik tam rank L matristir ve u_c de $m \times 1$ ' lik tasarlanan kontrol sinyali olsun. Bu kořullarda tanımlanan;

$$AAI = \begin{bmatrix} A & BK \\ -C & 0 \end{bmatrix}$$

sistemi de d6ng6sel olmalıdır.

4. $m \times L$ lik kontrol transfer fonksiyonun birim-rank yapısında olacak řekilde;

$$k(s) = \left(k_c + \frac{t_i}{s} + t_d s \right) M \quad (4.17)$$

seçilmeli, burada M ; $1 \times L$ ' lik n6merik tasarımcı tarafından belirlenen bir vekt6rd6r. Daha sonra kapalı d6ng6 sistemin karakteristik denklemi bulunur. Tasarlanan kapalı d6ng6 karakteristik denklemi ile bunu karřılařtırarak kontrol edici olan $k(s)$ in parametreleri belirlenir.

5. İstenilen PID kontrol edicinin parametreleri de orijinal sistemden;

$$G_c(s) = \frac{K}{s} + k(s) \quad (4.18)$$

ile bulunur.

Fakat bu metot iinde iki 6nemli sınırlayıcı etken vardır, bunlar kontrol edicinin birim rank yapısında olması gereklilięi, bir dięeri de, seilen K ve M matrislerinin kontrol edici tasarımında olduka 6nemli bir yer tutuyor olmasıdır.

Sonuç olarak PID kontrol ediciler end6striyel s6relerin %90 nından fazlasında kullanılmaktadır 1940 lardan bu yana kontrol edicilerin tasarımı ayarlanması ile ilgili olarak olduka fazla metot uygulamaya konulmuř ve geliřtirilmiřtir.

5. GENETİK ALGORİTMALAR

5.1 Genetik Algoritmanın Tarihçesi

Michigan Üniversitesinde psikoloji ve bilgisayar bilimi uzmanı olan John Holland bu konuda ilk çalışmaları yapan kişidir. Makine öğrenmesi (machine learning) konusunda çalışan Holland, Darwin'in evrim kuramından etkilenerek canlılarda yaşanan genetik süreci bilgisayar ortamında gerçekleştirmeyi düşündü. Tek bir mekanik yapının öğrenme yeteneğini geliştirmek yerine böyle yapılarda oluşan bir topluluğun çoğalma, çiftleşme, mutasyon vb. genetik süreçlerden geçerek başarılı (öğrenebilen) yeni bireyler oluşturabildiğini gördü. Çalışmalarının sonucunu açıkladığı kitabın 1975'te yayınlanmasından sonra geliştirdiği yöntemin adı Genetik Algoritmalar (ya da kısaca GA) olarak yerleşti. Ancak 1985 yılında Holland'ın öğrencisi olarak doktorasını veren David E. Goldberg adlı inşaat mühendisi 1989 da konusunda bir klasik sayılan kitabını yayınlayana dek GA' ların pek pratik yararı olmayan bir araştırma konusu olduğu düşünülüyordu. Halbuki Goldberg'in gaz boru hatlarının denetimi üzerine yaptığı doktora tezi ona sadece 1985 National Science Foundation Genç Araştırmacı ödülünü kazandırmakla kalmadı, GA' ya dayalı tam 83 uygulamaya yer vererek GA' nın dünyanın her yerinde çeşitli konularda kullanılmakta olduğunu gösterdi (Goldberg, D.E., 1989).

5.2 Genetik Algoritma Nedir

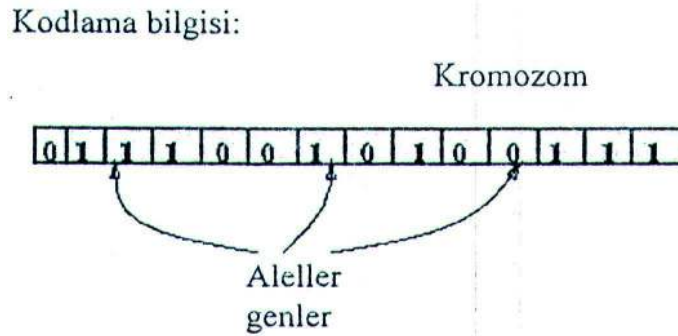
Doğadaki evrim kuramının yapay sistemlere uygulanması sürecinde evrimsel hesaplamalar başlığı altında genetik algoritmalar (GA), evrimsel programlama (EP) ve evrimsel stratejiler (ES) olmak üzere 3 yöntem ortaya çıkmıştır. Bunlar arasında günümüzde en çok bilinen ve en çok kullanılan yöntem genetik algoritmalarlardır. Az önceki açıklamalardan da anlaşılabileceği gibi genetik algoritmalar doğal seçim ve doğal kalıtım kurallarına dayanır. Doğal seçim, çevre koşullarına uyum sağlamış olan canlıların yaşamlarını sürdürmesi, uyum sağlayamamış olanların ise elenmesi yani ölmesi anlamına gelir. Doğal kalıtım ise, seçilmiş olan bireylerin birbirleri ile çiftleşerek yeni yavrular meydana getirmesi olarak kısaca özetlenebilir.

Genetik algoritmalarla problem çözümünde ilk topluluğu oluşturacak n adet birey tamamen rastlantısal belirlenir. Daha sonraki aşamalar ise iki çatı altında toplanabilir.

a) Genetik işlemler: çaprazlama ve mutasyon

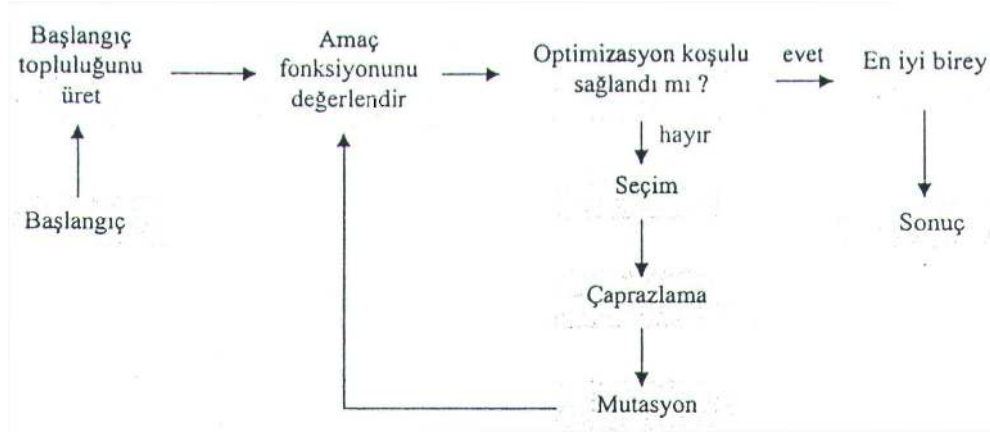
b) Evrimsel işlem: seçim

Problem çözümüne noktalar topluluğundan başlayan genetik algoritmalar öncelikle bu noktaları kodlarlar. Farklı tipte kodlama sistemleri olmakla birlikte genelde ikili kodlama kullanılır. Daha sonraki bölümlerde (bölüm 5.7) kodlama sistemleri ayrıntılı olarak ele alınacaktır. Çözüm kümesini (topluluğu) oluşturan her bir birey “kromozom” veya “birey” olarak adlandırılır. Kromozomlar gen adı verilen sembol katarlarından oluşur. Örneğin Şekil 5.1’ de ikili kodlama yapısına sahip 0 ve 1 sembollerinden oluşan “birey” görülüyor.



Şekil 5.1 İkili kodlama yapısındaki birey

Topluluğu oluşturan her bir bireyin temsil ettiği değer “amaç fonksiyonu”nda (objective function) yerine konarak bireylerin “uygunluk değerleri” (fitness value) yani bir sonraki nesilde yaşama olasılıkları bulunur. Bir sonraki adım, uygunluk değerleri göz önünde bulundurularak bir sonraki topluluğu meydana getirecek “ebeveynler” seçilmesidir. Önceden belirtilen oranda ebeveyn çaprazlamaya girerek birbirleri ile gen alışverişinde bulunurlar ve ortaya bir sonraki toplumun bireylerini oluşturacak olan “yavrular” çıkar. Yine önceden tanımlanmış mutasyon oranları nedeniyle ortaya çıkan yavrularda başkalaşımalar meydana gelir. Bu aşamadan sonra yeni topluluğu meydana getirecek bireyler hazır. Bu döngü istenilen koşullar sağlanıncaya kadar devam eder ve her bir yenileme işlemi “nesil” adını alır. Bu döngü Şekil 5.2’ de ve Çizelge 5.1’ de gösterilmiştir. Probleme başlarken kullanılacak topluluk rastlantısal olarak üretilir. Genetik algoritmalar rastlantısal kuralları kullanarak gittikçe daha iyi bir çözüme yönelirler.



Şekil 5.2 Genetik algoritma programı blok şeması

Çizelge 5.1 Genetik algoritmanın akış tablosu

1. [Başlatma] n bireyli bir topluluğu rastlantısal olarak üret. (problem için uygun çözüm)
2. [Uygunluk] Topluluktaki her bir bireyin uygunluk değerini kaydet.
3. [Yeni Topluluk] Yeni topluluk tamamlanana kadar aşağıdaki döngüye bağlı olarak yeni topluluğu yarat.
 - a) [Seçim] Uygunluk değerlerine göre topluluk içinden 2 ebeveyn birey seç.
 - b) [Çaprazlama] Çaprazlama olasılıklarına bağlı olarak yeni yavru oluşturacak ebeveynleri çaprazlamaya sok. Eğer çaprazlama olmayacaksa yavru ebeveynlerinin tamamen aynısıdır.
 - c) [Mutasyon] Mutasyon olasılığına bağlı olarak mutasyonunu gerçekleştir.
- 4) [Kabul] Oluşan yavruları yeni topluluğa yerleştir.
- 5) [Yerleşim] Yeni üretilen topluluğu algoritmanın tekrar işlemesi için kullanıma hazırla.
- 6) [Sınama] Eğer istenilen koşullar sağlanmışsa, dur, ve o anki topluluğun en iyi çözümünü döndür.
- 7) [Döngü] Adım 2' ye dön.

5.3 Genetik Algoritmalar Sözlüğü

Algoritma Kavramı: Verilen bir problemin çözümü için izlediğimiz sistemli yöntemle algoritma denir. Verilebilecek en basit örnek Euklides' in (Oklit) iki sayının ortak bölenini bulma algoritmasıdır.

Biyolojik Temeller: Biyolojinin kalıtımla ilgilenen dalına genetik denir. Kalıtım, bazı genetik özelliklerin bir kuşaktan diğer kuşağa aktarılma sürecidir. Modern genetik bilimin temelleri Gregor Mendel (1822-1884) tarafından gerçekleştirilen deneylere dayanmaktadır. Bu deneyler sonucunda, bir sonraki kuşağa aktarılacak özelliklerin bir takım kurallara uyduğu ortaya çıkmıştır. Genetik biliminde kullanılan ve tezde sözü edilen bazı kavram ve terimler aşağıda açıklanmıştır.

Allel (Allele): Bir özelliği temsil eden bir genin alabileceği değişik değerlerdir.

Çaprazlama (Cross-over): İki kromozomun bir araya gelerek genetik bilgi değişimi yapması
Örnek: 100011101 ve 010110001 kromozomları üzerinde 4. ayırımdan başlayarak tek noktalı çaprazlama yapıldığında 100110001 ve 010011101 kromozomları elde edilir.

Evrimsel Algoritma (Evolutionary Algorithm EA): Genetik algoritmalarını da içine alan bir algoritmik yöntem.

Evrilmek (Evolve): Bir evrim sürecinden geçmek

Evrim (Evolution): Genetik bilgi taşıyan bir topluluk üzerinde genetik işlemlerin uygulanması süreci.

Gen (Gene): Kendi başına anlamlı genetik bilgi taşıyan en küçük genetik yapı
Örnek: 101 bit dizisi bir noktanın x- koordinatının ikilik düzende kodlandığı bir gen olabilir.

Locus: Kromozom üzerindeki her bitin yerine verilen isimdir.

Genetik Programlama (Genetic Programming GP): Genlerinde program parçacıklarının kodlandığı kromozomlar üzerinde çalışan, bir genetik algoritma yolu ile istenilen işi yapan bir programın oluşturulması.

Tersinme (Inversion): Bir kromozomu oluşturan genlerden ardışık bir grubun kendi içerisinde birbirleri ile yer değiştirerek ters dizilmesi

Örnek: 011110101 kromozomu (her genin bir konum olduğu varsayımı ile) 5. ve 8. Gen konumları arasında tersindiğinde ortaya 011101011 kromozomu çıkar.

Eşleme (Matching): İki kromozomun çaprazlanma amacıyla seçilmesi

Mutasyon (Mutation): Bir kromozomun taşıdığı genetik bilgide bir nedene bağlı olmaksızın (rastgele) değişme olması.

Örnek: 100110000 kromozomunun 3. konumunun değişmeye uğraması sonucunda 1011100000 kromozomu oluşur.

Çoğalma, Kopyalama (Reproduction): Bir kromozomun kendisi ile aynı genetik bilgiyi taşıyan bir kopyasının oluşturulması

Örnek: 100111011 kromozomu çoğaltıldığında 100111011 ve 100111011 kromozomları elde edilir.

Seçme (selection): Bir kromozom havuzundaki kromozomlardan hangilerinin yeni yaratılacak havuza aktarılacağını, kromozomların başarı değerlerine bağlı olarak belirlenmesi.

En iyileme (Optimizasyon): Matematiksel bir terim olarak optimizasyon en iyinin tanımlanması ve çözülmesi anlamına gelir. Optimizasyon teorisi, optimalın, yani en uygunun, sayısal olarak elde edilmesi için kullanılan tüm yöntemleri içerir. Matematiksel olarak ifade etmek gerekirse, optimizasyon bilinen bir fonksiyonun (amacın) belirli kısıtlar altında ya da kısıtlar olmaksızın optimumunun bulunması demektir.

5.4 Genetik Algoritma Uygulama Alanları

Genetik Algoritma uygulamaları genel olarak üç gruba ayrılır: (1) optimizasyon, (2) sınıflandırılmış sistemlerin öğrenimi kapsamındaki makine öğretisi, (3) genetik programlama. Amaç fonksiyonu ile ifade edilen optimizasyon uygulamaları, arzu edilen herhangi bir sonuca ulaşmada yol gösterici konumundadırlar. Bu tip uygulamalara örnek olarak kâr payının en yüksek seviyeye çekilmesi veya herhangi bir hata teriminin minimuma getirilmesi verilebilir. İkinci gruba giren uygulamalar ise genetik algoritmalar, etkin çözüme ulaşmak için, makinelerin işleyiş kurallarını düzenleyen yapıyı oluşturur. Genetik programlama yapılan bir işte kullanılan programın geliştirilmesi ve daha verimli bir hale getirilmesine olanak sağlar. Bu işlemi yaparken de uygunluk fonksiyonu kullanılır.

Genetik Algoritma yaklaşımının endüstrideki uygulama alanları ise kısaca şöyledir:

Mühendislik: Elektrik enerji sistemleri, reaktif güç kompanzasyonu, bina tasarımı (kirişlerin büyüklükleri), çeşitli elektriksel tasarımlar, mekanik tasarımlar (ağırlık, hacim ve fiyat),

proses kontrolü, ağ tasarımları (bilgisayar ağları), hücresele otomatların ve sıralama ağlarının tasarımı.

Yönetim: Dağıtım, planlama, proje yönetimi, kurye yönlendirme (courier routing), paketleme, görev atama, zaman çizelgeleri.

Finans: Portfolyo dengelemesi, bütçeleme, tahmin, yatırım analizi ve ödeme takvimi, yaratıcı girişimin modellenmesi, pazarlık ve açık artırma stratejilerinin geliştirilmesi, ekonomik pazarların doğumunun modellenmesi.

Araştırma-Geliştirme (R&D): Eğri ve yüzey uydurma, yapay sinir ağları, bağlantı matrisleri, fonksiyon optimizasyonu, bulanık mantık, topluluk modellemesi, moleküler modelleme ve ilaç hazırlama.

5.5 Genetik Algoritma ile Geleneksel Yöntemler Arasındaki Farklar

Genetik algoritmaların temel özellikleri, aşağıdaki gibi dört maddede toplanabilir;

- GA parametrelerin kendileri ile değil, parametrelerin kodlanmış halleriyle çalışırlar.
- GA tek bir nokta yerine noktalar topluluğundan hareket eder.
- GA türev ve daha farklı bilgileri değil sadece amaç fonksiyonu değerlerini kullanır.
- GA belirgin (deterministik) değil, olasılık kurallara dayanır.

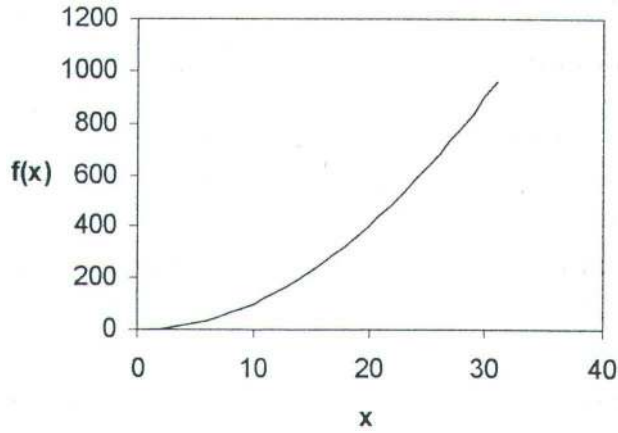
Aramaya tek bir noktadan değil de birçok noktadan başlamanın en büyük yararı yerel optimuma yakalanma olasılığını ortadan kaldırmasıdır. Geleneksel yöntemler aramaya tek bir noktadan başladıkları için ilk buldukları yerel optimum noktasında aramayı sonlandırırlar. Dolayısıyla birden çok optimuma sahip fonksiyonlar için kullanışlı değildir. Genetik algoritmalar ise tüm optimum noktaları bularak bunları karşılaştırır ve fonksiyonun evrensel optimum noktasına ulaşır.

Genetik algoritmalarda diğer yöntemlerde olduğu için gibi türev bilgilerine ihtiyaç yoktur, yalnızca amaç fonksiyonu ve uygunluk değerleri yaklaşımın hangi yönde ilerleyeceğini belirler.

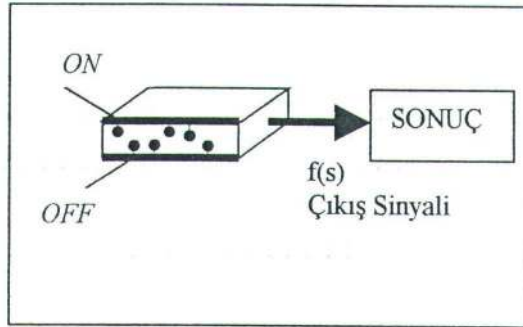
Rastlantısal geçiş kurallarına bağlı olarak çalıştığından model değiştirilmese bile her çalıştığında farklı çözümler elde edilir. Geleneksel yöntemler ise aynı değerler ile her çalıştığında aynı çözüme ulaşır.

Genetik algoritmalarda parametreler sonlu uzunlukta diziler ile gösterilirler. Örnek olarak şekil 5.3’ te gösterilen $[0,31]$ aralığındaki optimizasyon problemi ele alınsın. İlk olarak x parametresi açık kapalı biçimde iki farklı değer alan bir değişkenle kodlanır. Daha sonra açık kapalı yerine “bir” ve “sıfır” lar kullanılır (Üstün, 2001).

Sistem şekil 5.4’ te gösterildiği gibi bir kapalı kutu olarak düşünülebilir.



Şekil 5.3 $f(x) = x^2$ amaç fonksiyonu



Şekil 5.4 Kapalı kutu örneği

Kapalı kutu örneğinde beş adet giriş anahtarı bir adet f çıkış işareti bulunmaktadır. Amaç fonksiyonu, matematiksel olarak s belli bir anahtar kombinasyonu olmak üzere $f = f(s)$ (örneğin $f = x^2$) şeklinde yazılabilir. Problemin amacı, bu beş anahtarın mümkün olan en yüksek f değerini elde edecek uygun kombinasyonunu bulmaktır. Diğer arama yöntemleri anahtarların durumları ve bir durumdan diğerine nasıl geçileceğini araştırırken, genetik algoritmalarda önce anahtarların durumları kodlanır. Bu kodlama beş bitlik sayı dizileriyle yapılabilir.

Tek bir nokta ele alıp, bu noktadan diğer bir noktaya belli bir kural kullanarak geçmek yanlış

bir yerel en iyi noktaya yerleşmeye sebep olabilir. GA tek bir noktalar topluluğundan yararlandığı için, yapılacak ikinci adım bu noktalar topluluğunu belirlemektir. Böylece global en iyi noktaya yakınsama şansı artacaktır. GA'da nesil adı verilen bu noktalar topluluğundan, yeni bir nesil yani yeni bir noktalar topluluğu elde edilir. Örnekte birey sayısı $n = 4$ alınmıştır ki bu sayı GA standartları için küçük bir sayıdır.

01101

11000

01000

10011

Bu dört noktadan hareketle GA kullanılarak yeni nesiller bulunmaktadır. Bir nokta topluluğu üzerinde paralel çalışmanın, gürbüzlüğü sağlama yolunda olumlu etkileri olmaktadır.

5.6 Genetik Algoritmanın Avantajları

Genetik algoritmaların temel avantajı optimize etmeye çalıştıkları problemle ilgili herhangi bir bilgiye ihtiyaç duymamalarıdır. Sistemin matematik model bilgilerini gerektirmez. Karmaşık matematik hesaplamaları yerine yalnızca giriş-çıkış bilgilerine ihtiyaç duyar. Bu avantajından dolayı karmaşık problemlerin optimizasyonu için tercih edilebilir bir yöntemdir. Büyük parametre ve yapısal belirsizlikleri tolere edebilir. Yine uygun performans endeksi seçimi ile sistemin arzu edilen dinamik davranışları ve kararlılığı elde edilebilir. (Zuo,1995)

Oluşturulan başlangıç popülasyonu girdiği genetik algoritma döngüsü sonucunda en uygun çözümü kolayca bulabilir. Genetik algoritma evrim teorisinin adımlarını kullanarak çalışır. Bir veri grubundaki özel bir veriyi bulmak için kullanılır. Genetik algoritma klasik yöntemlerin çok uzun zamanda bulabileceği çözümü yeterli doğrulukla çok kısa zamanda bulabilir. Optimal değeri bulmayı garanti etmez, fakat uzun nesiller sonucunda en uygun değere çok yakın çözümler bulunmasını sağlar. Genetik algoritmalar özellikle araştırmacının kesin, konu uzmanı olmadığı zamanlarda araştırmacıya o alanda bilgi edindirmede başarılıdır (Valanzuela,1995) (Muzoğlu,2005).

Temel başlıklar halinde genetik algoritmaların avantajlarını şu şekilde sıralayabiliriz:

Sürekli ve ayrık parametreleri optimize eder.

Türevsel bilgiler gerekmez.

Amaç fonksiyonunu geniş bir yelpazede araştırır.

Çok sayıda parametrelerle çalışma imkânı vardır.

Paralel PC'ler kullanılarak çalıştırılabilir.

Karmaşık amaç fonksiyon parametrelerini, yerel ekstremlara takılmadan optimize edebilir.

Birden fazla parametrenin optimum çözümlerini elde edebilir.

5.7 Kodlama (Encoding)

Genetik algoritmaları kullanarak problem çözümüne başlanıldığında, bireylerin kodlanması sorunu karşımıza çıkar. Kodlama türleri, çalışılan problemin türüne bağlı olarak çeşitlilik gösterir.

5.7.1 İkili Kodlama (Binary Encoding)

En yaygın olarak kullanılan kodlama biçimidir. Bunun en önemli sebebi, genetik algoritmalar ile ilgili ilk çalışmalarda kullanılmış olmasıdır. Bu tür kodlamada her bir birey 0 ve 1 bitlerinden oluşmuş bir katar olarak ifade edilir, (Çizelge 5.2).

Çizelge 5.2 İkili kodlama kullanılarak oluşturulan bireyler

Kromozom A	101100101100101011100101
Kromozom B	111111100000110000011111

İkili kodlamada çok küçük değerli alellere rağmen çok çeşitli bireyler yaratmak mümkündür. Diğer bir yandan da bazı problemler için kodlama türü yeterince uygun (doğal) olmadığı için çaprazlama ve/veya mutasyon aşamasından sonra düzeltme yapılması gerekebilir. Knapsack problemlerinde olduğu gibi, değeri ve büyüklüğü sabit cisimlerin hacmi ve boyutları belli olan uzay parçasına maksimum sayıda yerleştirilmesi problemi için bu kodlama tipi kullanılabilir.

5.7.2 Dizilim (Permütasyon) Kodlaması

Dizilim (permütasyon) kodlaması seyyar satıcı (travelling salesman) problemleri ve görev sıralama problemleri gibi sıralamayı gerektiren problemlerde kullanılır. Bu tür kodlamada her bir birey, sıralama gösteren sayılardan oluşan katar ile ifade edilir, (Çizelge 5.3).

Çizelge 5.3 Dizilim kodlaması kullanılarak oluşturulan bireyler

Kromozom A	1 5 3 2 6 4 7 9 8
Kromozom B	8 5 6 7 2 3 1 4 9

Dizilim kodlaması yalnızca sıralama problemlerinde kullanışlıdır. Bu kodlaya seyyar satıcı problemi örnek olarak verilebilir. Birçok şehre gitmesi gereken bir seyyar satıcının, kat edeceği mesafenin en kısa olması için izlemesi gereken yol üzerindeki şehirlerin sırası ne olmalıdır? Bu problem için bir birey kodu, satıcının ziyaret edeceği şehirlerin sırasını gösterir.

5.7.3 Reel (Gerçel) Kodlama

Gerçel kodlama karışık değerlerin (örneğin reel sayıların) bulunduğu problemlerde kullanılır. Bu tür problemlerde ikili kodlama kullanılması problemin daha zor bir biçim almasına neden olur. Gerçel kodlamada her bir birey problemle bağlantılı olan form numaralarının, reel sayılar veya karakterlerin oluşturduğu katarla ifade edilir. (Çizelge 5.4)

Çizelge 5.4 Gerçel kodlama kullanılarak oluşturulan bireyler

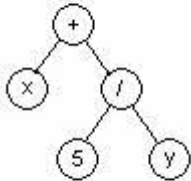
Kromozom A	1.2324 5.3243 0.4556 2.3293 2.4545
Kromozom B	ABDJEIFJDHDIERJFDLDFLFEGT
Kromozom C	(back), (back), (right), (forward), (left)

Gerçel kodlama bazı özel problemler için çok iyi sonuçlar vermekle birlikte, bu tür kodlama kullanıldığında, yeni çaprazlama ve mutasyon tanımlamaları yapılması gereklidir. Örneğin, yapay sinir ağlarında istenilen çıkışların elde edilmesi için girişlerin ağırlıklarının bulunması. Bu problemde bireyi temsil eden katarı oluşturan reel sayılar giriş ağırlıklarını temsil eder.

5.7.4 Ağaç Kodlaması

Ağaç kodlaması genellikle genetik programlama için gerekli ifadelerin daha kolay geliştirilmesini sağlamak amacıyla kullanılan bir yöntemdir. Bu tip kodlamada her bir birey, programlama dilinde kullanılan fonksiyonlar veya komutlar, nesnelerin birbirileri ile bağlantılarını gösteren ağaç yapıdan oluşmaktadırlar, (Çizelge 5.5).

Çizelge 5.5 Ağaç yapı kodlaması kullanılarak oluşturulan bireyler

Kromozom A	Kromozom B
	
$(+ x (/ 5 y))$	$(do_until step wall)$

Daha önceden de sözü edildiği gibi ağaç yapı kodlaması programlarının değiştirilmesi ve geliştirilmesi için oldukça elverişlidir. Ağaç yapı nedeniyle çaprazlama ve mutasyon olacağı zaman, kullanılan programı oluşturan bölümler kolayca parçalanabilir, bu yüzden LISP programlama dili sık sık bu kodlama biçimini kullanmaktadır. Giriş ve çıkış değerleri verilen bir fonksiyonun bulunması örnek olarak verilebilir. Burada amaç tüm giriş değerlerine en yakın sonucu sağlayacak fonksiyonun hesaplanmasıdır.

5.8 İlk Popülasyonun Rasgele Oluşturulması

Başlangıç popülasyonunu oluştururken rasgele sayı üretici kullanılır. Problem için önerilen belirli çözüm aralığı varsa başlangıç popülasyonu bu aralıktan oluşturulabilir. Bu yöntem optimal sonuca giderken sistemin daha hızlı davranmasını sağlayacaktır. Başlangıç popülasyonundaki kromozomların kaç bitele temsil edileceğini ve başlangıç popülasyonundaki birey sayısı da sistemin hızını etkileyeceğinden dikkat edilmesi gereken parametrelerdir.

5.9 İlk Popülasyonun Uygunluk Değerinin Hesaplanması

Uygunluk değeri, yeni popülasyona taşınacak kromozomların belirlenmesinde kullanılan bir araçtır. Bu nedenle, algoritmanın her iterasyonunda popülasyondaki kromozomların uygunluk değeri hesaplanır. (Dengiz,1998). Yüksek uygunluk değerine sahip bireyler daha büyük olasılıkla seçilmelidir.

Her bir çözümün kalitesinin belirlenmesinde kullanılan uygunluk değeri hesaplaması genetik algoritmanın çalışma performansında önemlidir. Ancak genetik algoritmanın ilerleyen iterasyonlarında bireylerin uygunluk değerleri birbirlerine yakınsamaya başlar. Bu da bir sonraki aşamada uygunluk değerlerine göre seçim işlemini olanaksız kılar. Bu durumda genetik algoritmanın optimal sonuca ulaşması zordur. Bu sorunu çözmek için uygunluk değeri

işlemini normalizasyon işlemi takip etmelidir. Normalizasyon işleminde en yaygın olarak kullanılan teknik ölçeklendirme işlemidir (De Jong,1975).

5.10 Yeni Popülâsyonun Oluşturulması

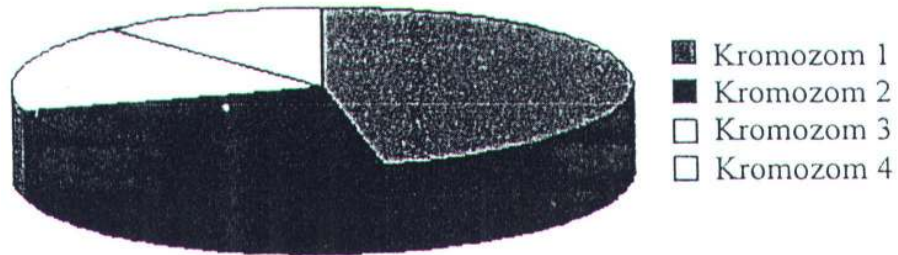
Rasgele oluşturulan ilk popülâsyondan ya da bir önceki iterasyonda oluşturulan popülâsyondan yeni popülasyona bireylerin seçilmesidir. Bir sonraki popülâsyona taşınan bireyler en güçlü genetik yapıya sahip dizilerdir. Bireyler uygunluk değerlerinin yüksekliğiyle orantılı olarak bir sonraki popülâsyonda yer alırlar. Kalitesi yüksek olan kromozomların bir sonraki iterasyonda yer alma şansı yükselirken daha zayıf bireylerin popülâsyondan kaybolması mantığına dayanır. Bu yapı döngü içerisinde değerlendirme fonksiyonu ve seçim mekanizması ile temsil edilmektedir.

5.11 Seçim (Selection)

Bilindiği gibi çaprazlamaya girecek olan ebeveynler topluluğu (popülâsyonu) oluşturan bireyler arasından seçilir. Buradaki temel problem çaprazlamaya girecek olan bireylerin seçiminin ne şekilde olacağıdır. Darwin' in evrim teorisine göre, daha uygun (fitness değeri yüksek) olan bireyler yaşamaya devam edip yavru oluşturabilirler. Bu ilke doğrultusunda, ebeveyn seçiminde rulet çarkı seçimi (roulette wheel selection), Boltzman seçimi, turnuva seçimi (tournament selection), rank seçimi, kararlı hal seçimi (steady state selection) gibi birçok yöntem kullanılabilir. Bu yöntemlerden bir kısmı aşağıda açıklanmıştır.

5.11.1 Rulet Çarkı Sistemi

Bu yöntemde ebeveynler uygunluk değerlerine göre seçilirler, dolayısıyla uygunluk değeri yüksek olan bireylerin seçilme olasılığı daha fazla olacaktır. Şekil 5.5' te dört bireyden oluşan topluluğa ilişkin rulet çarkı görülmektedir. Uygunluk değeri yüksek olan bireylerin çarkta kaplayacağı alan daha geniş olacağından çark rasgele çevrildiğinde bu bireylerin seçilme olasılığı diğerlerinden yüksek olacaktır.



Şekil 5.5 Rulet çarkı sistemi

Bu seçim aşağıdaki algoritma ile özetlenebilir;

- [Uygunluk] Topluluğu oluşturan tüm bireylerin uygunluk değerleri bulunur. - U_i
- [Toplam] Uygunluk değerleri toplanır ve bu değer toplam uygunluk değeri olarak adlandırılır. - $\sum U$
- [Olasılık] her bir bireyin seçilme olasılığı bulunur. - $p_i = U_i / \sum U$
- [Birikimli Olasılık] Her bir bireyin birikimli olasılıkları bulunur. P_{bi}

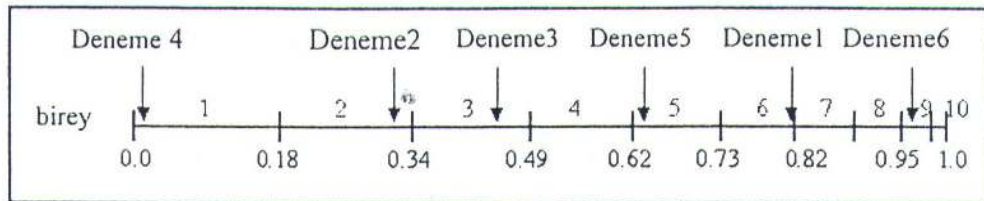
$$P_{bi} = \sum_{j=1}^i p_j \quad i = 1, 2, \dots, \text{pop_size}, j = 1, 2, \dots, i$$
- [Seçim] (0,1) aralığında rasgele pop_size adet sayı üretilir - r
- [Döngü] Eğer $r \leq p_{b1}$ ise birinci ebeveyn olarak seç; diğer durumlarda üretilmiş olan r sayısı hangi birikimli olasılık aralığında ise ($p_{i-1} < r \leq p_i$) i . bireyi seç. ($2 \leq i \leq \text{pop_size}$).

Örneğin Çizelge 5.6' da uygunluk değerleri ve seçilme olasılıkları verilen on bir birey ele alınırsa;

Çizelge 5.6 Seçilme olasılıkları ve uygunluk değerleri

Birey Numarası	1	2	3	4	5	6	7	8	9	10	11
Uygunluk Değeri	2.0	1.8	1.6	1.4	1.2	1.0	0,8	0.6	0.4	0.2	0.0
Seçilme Olasılığı	0.18	0.16	0.15	0.13	0.11	0.09	0.07	0.06	0.03	0.02	0.0

Altı adet rasgele sayı üretildiğinde: 0.81, 0.32, 0.96, 0.01, 0.65, 0.42

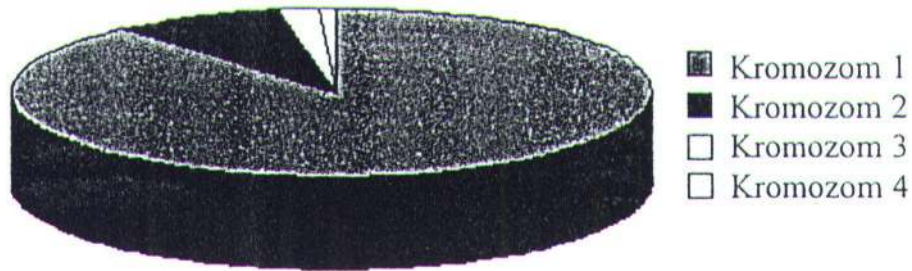


Şekil 5.6 Üretilen sayıların (deneme i) uygun düştüğü birey aralıkları

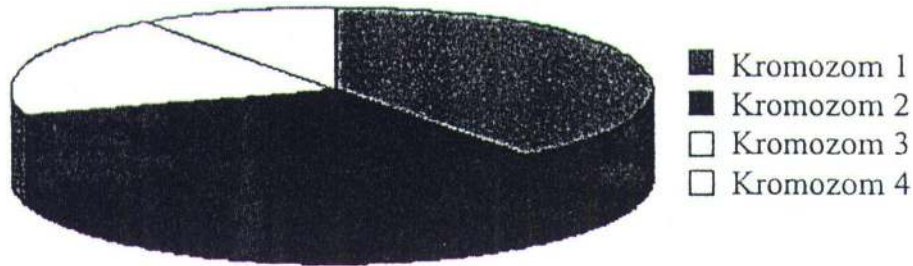
Seçim sonraki topluluğu oluşturan bireylerin 1, 2, 3, 5, 6, 9 numaralı bireyler olacağı Şekil 5.6' da görülmektedir.

5.11.2 Rank Seçimi

Uygunluk değerleri birbirinden çok farklı olan durumlar rulet çarkı seçimi için problem yaratır. Örneğin, en iyi bireyin uygunluk değeri %90 ise çark döndürüldüğünde diğer bireyler çok az oranda seçilecek ve algoritmanın kısır döngüye girme ihtimali doğacaktır. Rank seçimi öncelikle topluluğu sıralar ve diğer bireylerin uygunluk değerlerini bu sıralamaya göre düzenler. En kötü bireyin uygunluk değeri 1, daha az kötü olanın 2 ... ve en iyi bireyin uygunluk değeri n_{pop} olacaktır. Burada n_{pop} topluluğu oluşturan birey sayısını gösterir. Sıralama öncesi ve sonrası durumlar Şekil 5.7 ve Şekil 5.8’ de görülmektedir.



Şekil 5.7 Sıralama öncesi durum (uygunluk değerlerine göre grafik)



Şekil 5.8 Sıralama sonrası durum

Topluluktaki birey sayısı $npop$, bireyin topluluktaki pozisyonu Pos (en az uygun bireyi için $Pos = 1$, en uygun birey için $Pos = npop$), seçilme baskısı parametresi SP olmak üzere her bir birey için uygunluk değeri aşağıdaki formüllerle hesaplanır.

Lineer rank seçimi için;

$$Uygunluk(Pos) = 2 - SP + 2.(SP - 1).(Pos - 1) / (npop - 1)$$

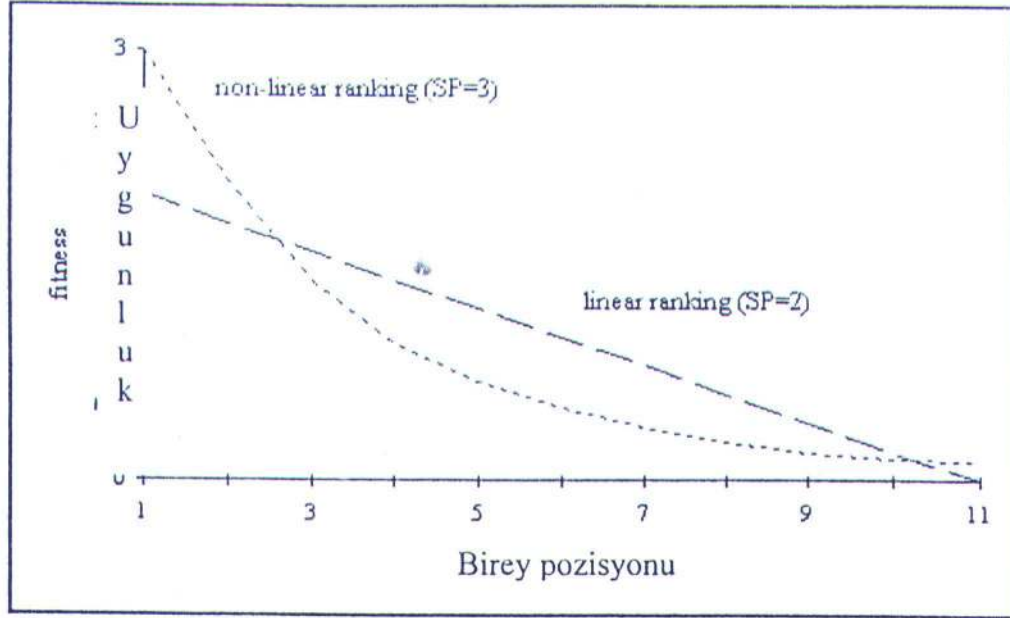
Lineer olmayan rank seçimi için;

$$Uygunluk(Pos) = npop \cdot X^{(Pos - 1)} / \sum(X^{(i - 1)}); i = 1:npop$$

Buradaki X ise aşağıdaki polinomun köküdür.

$$0 = (SP - 1) \cdot X^{(npop - 1)} + SP \cdot X^{(npop - 2)} + \dots + SP \cdot X + SP$$

Lineer rank seçimi için SP, [1.0, 2.0] aralığında iken lineer olmayan rank seçimi için bu aralık [1.0, npop - 2.0]'dır.



Şekil 5.9 Lineer ve lineer olmayan rank seçimlerinin karşılaştırılması

Rulet çarkı sisteminde verilen örneğin aynısını rank seçimi için tekrar ele alırsak Çizelge 5.7 yardımı ile seçilme baskısı parametresinin değeri azaldıkça bireylerin uygunluk değerlerinin birbirine yaklaştığı görülebilir.

Çizelge 5.7 Uygunluk değerinin seçilme baskısı ile değişimi

Amaç Fonksiyonu	Uygunluk değerleri (parametre: seçilme baskısı)				
	Lineer rank		Rank yok	Lineer olmayan rank	
	2,0	1,1	1,0	3,0	2,0
1	2,0	1,10	1,0	3,00	2,00
3	1,8	1,08	1,0	2,21	1,69
4	1,6	1,06	1,0	1,62	1,43
7	1,4	1,04	1,0	1,99	1,21
8	1,2	1,02	1,0	0,88	1,03
9	1,0	1,00	1,0	0,65	0,87
10	0,8	0,98	1,0	0,48	0,74
15	0,6	0,96	1,0	0,35	0,62
20	0,4	0,94	1,0	0,26	0,53
30	0,2	0,92	1,0	0,19	0,45
95	0,0	0,90	1,0	0,14	0,38

Rank seçiminde tüm bireylerin seçilme olasılığı vardır fakat en iyi birey diğer bireylerden çok fazla farklılık göstermediğinden bu yöntem çok yavaş bir yaklaşımdır.

5.11.3 Kararlı Hal Seçimi

Bu yöntem ebeveyn seçimi için partikular değildir. Kararlı hal seçiminde uygunluk derecesi yüksek olan bireyler belirlenerek yavru oluşturmak üzere çaprazlamaya girerken, uygunluk derecesi düşük olan bireyler atılarak yerlerine yeni oluşan yavrular konulur.

5.11.4 Turnuva Seçimi

Bu yöntemde bir grup birey rastlantısal olarak seçilir ve bu grup içerisindeki en iyi birey çiftleşmeye gönderilir. En genel turnuva büyüklüğü 2' dir ve bu ikili turnuva olarak adlandırılır.

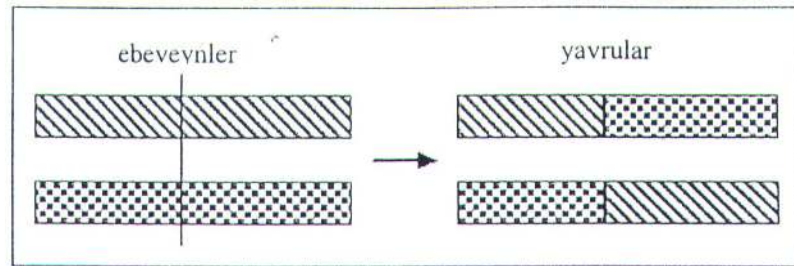
5.11.5 Elitistik Seçim

Çaprazlama sonucunda oluşan yavruların meydana getirdiği yeni toplulukta, geçmiş toplulukta bulunan ve uygunluk derecesi yüksek olan bireyler bulunmayabilir. Bu durumlarda seçkinlik (elitizm) sıkça kullanılmaktadır. Bunun anlamı uygunluk derecesi en yüksek olan bireyin değişikliğe uğramadan, yani çaprazlama ve mutasyona uğramadan, yeni topluluğa doğrudan kopyalanmasıdır. Bu yöntem kullanıldığında genetik algoritmanın performansı daha yüksek olur ve sonuca daha hızlı yaklaşılr.

5.12 Çaprazlama

5.12.1 Tek noktalı çaprazlama

Tek noktalı çaprazlamada, çaprazlamanın oluşacağı tek bir konum vardır ve bu konum rasgele belirlenir. Çaprazlamaya giren bireylerin bu noktaya kadar ve bu noktadan sonraki bit'leri yer değiştirerek iki yeni yavru oluşturulur. Tüm bu söylenenlerin daha iyi anlaşılabilmesi için Şekil 5.10 etkili olacaktır.



Şekil 5.10 Tek noktalı çaprazlama

Aşağıda ikili kodlama ile verilmiş bireyler için çaprazlama konumu 5 kabul edilerek tek noktalı çaprazlama yapılırsa;

birey 1 0 1 1 1 0 0 1 1 0 1 0

birey 2 1 0 1 0 1 1 0 0 1 0 1

Çaprazlama sonrası oluşan yeni bireyler (yavrular);

yavru 1 0 1 1 1 0 | 1 0 0 1 0 1

yavru 2 1 0 1 0 1 | 0 1 1 0 1 0

5.12.2 Çok Noktalı Çaprazlama

Çok noktalı çaprazlamada, çaprazlamanın yapılacağı m adet konum rasgele belirlenir ve çiftleşen bireylerin bu konumlar arasındaki bit'leri karşılıklı yer değiştirilerek yeni bireyler elde edilir. Şekil 5.11 bu işlemi göstermektedir.

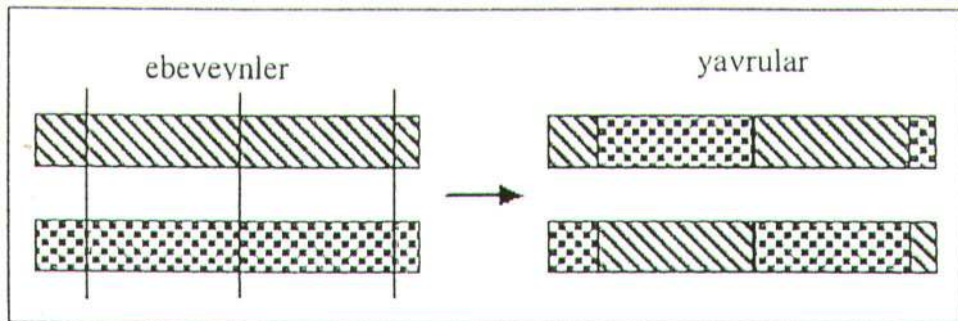
Çaprazlama pozisyonu ($m = 3$) 2 6 10 olmak üzere çaprazla yapıldığında;

birey 1 0 1 1 1 0 0 1 1 0 1 0

birey 2 1 0 1 0 1 1 0 0 1 0 1

yavru 1 0 1 | 1 0 1 1 | 1 1 0 1 | 1

yavru 2 1 0 | 1 1 0 0 | 0 0 1 0 | 0



Şekil 5.11 Çok noktalı çaprazlama

Çok noktalı çaprazlamanın ana fikri, ebeveynleri birçok parçaya ayırarak çaprazlamaya sokup, ebeveynlerden oldukça farklı olan yavrular meydana getirmektir. Çok noktalı çaprazlamanın sözü edilen bozucu etkisi nedeniyle araştırma alanı genişler ve başlangıçta uygun olan bireylere yakınsama ihtimal azalır. Dolayısıyla bu süreç araştırmayı daha dayanıklı (robust) hale getirir.

5.12.3 Düzgün (Uniform) Çaprazlama

Tek ve çok noktalı çaprazlamalarda, çaprazlamanın yapılacağı konum(lar) belirtiyor ve bu noktalar arasındaki bit'ler yer değiştirilerek yavrular oluşturuluyordu. Düzgün çaprazlamada ise, bireylerle aynı uzunlukta olan ve bit'leri rasgele üretilen bir maske kullanılır. Kullanılan bu maske, çaprazlama sonucunda oluşacak yavrunun bit'lerinin hangi ebeveynden geleceğini gösterir.

birey 1 0 1 1 1 0 0 1 1 0 1 0

birey 2 1 0 1 0 1 1 0 0 1 0 1

Örneğin maskede bulunan 1, birey 1' den gelmesi gereken bit'i, maskede bulunan 0 ise birey 2' den gelmesi gereken bit'i gösterdiğini kabul edersek;

maske 1 0 1 1 0 0 0 1 1 0 1 0

maske 2 1 0 0 1 1 1 0 0 1 0 1

Çaprazlama sonucunda oluşan yeni bireyler;

yavru 1 1 1 1 0 1 1 1 1 1 1 1

yavru 2 0 0 1 1 0 0 0 0 0 0 0

Düzgün çaprazlama algoritması, ayrık çaprazlama ile özdeştir.

5.12.4 Karışık (Shuffled) Çaprazlama

Karışık çaprazlamada düzgün çaprazlamayla yakından alakalıdır. Tek noktalı çaprazlamada olduğu gibi, çaprazlamanın olacağı konum belirlenir. Fakat değişkenler yer değiştirmeden önce iki ebeveyn arasında rasgele olarak karıştırılır. Çaprazlama sonucunda oluşan yavrulardaki değişkenler karıştırılmamıştır.

5.13 Mutasyon

Çaprazlama aşamasından sonra mutasyon olma olasılığına bağlı olarak mutasyon işlemi meydana gelir. Mutasyonun oynadığı rolün önemi hala bir tartışma konusudur. Bazı araştırmacılar mutasyonun ikincil derecede önemli olduğuna inansa da, doğa baz alınarak geliştirilen bu yöntem mutasyonsuz düşünülemez.

Mutasyon sonucunda asıl (orijinal) toplulukta olmayan özellikler kazanılabilir, dolayısıyla bu genetik algoritmanın hızlıca bir sonuca ulaşmasını (yakınsamasını) önler. Mutasyon olasılığı arttıkça algoritma öngörülen araştırma uzayından uzaklaşacaktır.

Birden çok optimum noktasına sahip fonksiyonlarda, yerel optimum noktasına yakınsayan ve kısır döngüye giren genetik algoritma mutasyon sayesinde bu nokta etrafından uzaklaşabilir. Böylece evrimsel optimum noktasına ulaşma olasılığı artar.

En iyi çözümün yitirilmemesi için, son yineleme işlemi sırasında mutasyon meydana gelmez.

5.14 Genetik Algoritmanın Kontrol Parametreleri

Genetik algoritmaların işleyişi, arama uzayından daha önce örnekleşmiş olan bölgelerin işletilmesi ve bu uzaydan yeni bölgelerin araştırılmasının dengeli birleşimi olarak düşünülebilir. Genetik algoritmanın performansını kontrol eden bu denge, kontrol parametrelerinin doğru seçilmesi ile belirlenmektedir(Yeniay,1999). Genetik algoritmanın işleyişini kontrol etmek için kullanılan parametreler aşağıda açıklanmıştır.

Popülasyon Büyüklüğü (N)

Popülasyon genişliğinin küçük olması çözüm uzayını etkilediği için genetik algoritmanın performansını azaltır. Çözüm uzayının küçük olması zamansız yakınsamaya neden olur. Popülasyon genişliğinin büyük olması ise çözüm uzayında yeterli aramaya olanak sağladığı halde, her iterasyonda daha geniş bir popülasyon üzerinde işlem yapıldığı için çalışma zamanını artırır.

Çaprazlama Oranı(p_c)

Seçim işlemi ile oluşan popülasyondaki kromozomlara her iterasyonda kaç tanesinin çaprazlamaya gireceğini belirler. Her yeni popülasyonda $p_c \times N$ tane kromozoma çaprazlama uygulanır. Düşük çaprazlama oranı aramanın çok yavaş, büyük çaprazlama oranı ise hızlı popülasyon değişikliğine neden olur.

Mutasyon Oran(p_m)

Seçim işlemi sonrasında elde edilen yeni popülasyondaki her kromozomun her elemanı p_m mutasyon oranında mutasyona uğrar. Her iterasyon sonucunda $p_m \times N \times L$ (L , her bir kromozomda mutasyona uğrayan bit adeti) adet bit mutasyona uğrar.

Seçim Mekanizmaları

Rastasal olarak oluşturulan ilk popülasyondan sonra seçim mekanizmaları devreye girerek yeni popülasyonu oluşturur. Bu konu daha önce açıklandığı için tekrar üzerinde durulmamıştır.

Ölçeklendirme Fonksiyonu Faktörü

Son iterasyona kadar popülasyondaki değişkenliğin devamı ölçeklendirme fonksiyonları ile sağlanır. Kontrol parametrelerinin en iyi kombinasyonları için yapılan çalışmalar vardır. İlk çalışma DeJong(1975) tarafından yapılmıştır (Goldberg,1989). DeJong(1975), popülasyon genişliğinin, çaprazlama ve mutasyon oranlarının etkilerini 5 farklı özellikteki problem üzerinde incelenmiştir. Popülasyon 50 ile 100 arasında olduğunda, çaprazlama oranı 0.60 ve mutasyon oranı 0.001 alındığında GA performansının iyi olduğu görülmüştür (Altıparmak,1996).

Nesil Ayrılığı(G)

Her iterasyonda popülasyonda meydana gelecek değişimin yüzdesini kontrol eder. Böylece popülasyonun iterasyonlar boyunca değişimi nesil ayrılığı katsayısı ile belirlenmiş olur. Bu durumda, t . iterasyondaki popülasyonun $N \times (1-G)$ adet kromozomu, $(t+1)$. iterasyonda yaşayacaktır, (Muzoğlu,2005).

Bitirme Koşulu

GA da üretim-değerlendirme-seçim süreci, kullanıcı tarafından belirlenen iterasyon sayısına ulaşıncaya ya da popülasyonun ortalama uygunluk değeri popülasyondaki en iyi dizinin uygunluk değerine önceden belirlenen oranda yaklaşıncaya kadar devam eder, (Goldberg,1989).

Ölçeklendirme Fonksiyonu

Genetik algoritmanın ilk birkaç iterasyonundan sonra elde edilen popülasyonda uygunluk değeri yüksek kromozom bulunmaktadır. Bu diziler yüksek uygunluk değerlerinden dolayı bu uygunluk değerleri etrafında zamansız bir yakınsama olabilmektedir. Son iterasyonlarda da

kromozomların uygunluk değerleri birbirine çok yakınlaştığı için iyi kromozomların korunması zorlaşmaktadır. Orantılı seçim mekanizmalarında bu sorunu ortadan kaldırmak için ölçeklendirme fonksiyonu kullanılmalıdır. Literatürde 3 farklı ölçeklendirme fonksiyonu kullanılmaktadır. Kullanılan ölçeklendirme fonksiyonlarından biri doğrusal ölçeklendirme fonksiyonudur. Bu metotta kromozomların uygunluk değerleri doğrusal bir fonksiyon kullanılarak yeniden düzenlenir.

$$f'_i = a \times f_i + b \quad i=1,2,3,\dots,N \quad (5.1)$$

Burada f ham uyum f' ise ölçeklendirilmiş uyumdur. Fonksiyonda a ve b katsayıları f'_{ort} ortalama ölçeklendirilmiş uyumu, f_{ort} ortalama ham uyuma eşit olacak şekilde seçilmelidir. Böylece mevcut yığında ortalama uygunluk değerine sahip dizilerin yeni yığında birer kopyalarının olmasını sağlanmış olur.

Ayrıca mevcut yığında maksimum uygunluk değerine sahip dizinin kopya sayısı kontrol edilmelidir. Bunun için ölçeklendirilmiş maksimum uygunluk değeri

$$f'_{max} = c \times f_{ort} \quad (5.2)$$

eşitliğinden hesaplanabilir. Goldberg(1989) c değerini 1.2 ile 2 arasında seçilebileceğini belirtmiştir. Doğrusal fonksiyon kullanıldığında negatif değerlerden kurtulabilmek için Forrest(1985), yığının varyant bilgisinin kullanılmasını önermiştir. Sigma kesim planı olarak adlandırılan ölçeklendirme modelinde hesaplamalar için aşağıdaki fonksiyon kullanılır.

$$f'_i = f_i - (f_{ort} - c \times \sigma) \quad i=1,2,3,\dots,N \quad (5.3)$$

Bu eşitlik de c sabiti 1 ile 3 arasında seçilmesi önerilmektedir. Forrest(1985) bu yaklaşımda elde edilen negatif değerlerin 0' a eşitlenmesini önermektedir. Negatif değerlerin üretilmediği ölçeklendirme fonksiyonu ise üs yaklaşımı modelidir. Aşağıdaki eşitlikten yararlanılır;

$$f'_i = f_i^k \quad i=1,2,\dots,N \quad (5.4)$$

k sabiti problemin yapısına bağlı olup Gillies çalışmalarında k sabitini 1.005 olarak almıştır(Goldberg 1989; Michalewicz 1992).

5.15 Genetik Algoritma Uygulamasına Bir Örnek

Burada, örnek amaç fonksiyonu olarak Şekil 5.3' te gösterilen $f(x) = x^2$ fonksiyonu ele alınmıştır. x değişkeni 0 ile 31 arasında değerler almakta ve ikilik düzende 5 bit ile ifade edilmektedir. Sayılar işaretsiz ve tam sayı olsun. Bu durumda $00000_2 = 0_{10}$ ve $11111_2 = 31_{10}$

olur. Başlangıç olarak rasgele oluşturulmuş bir dizi topluluğu ele alınsın. Bunun için daha önceki kapalı kutu örneğinde ele alınmış topluluk düşünölsün. Bu topluluğun elemanları çizelge 5.8(a)'nın sol tarafında gösterilmiştir. Çizelgede soldan sağa doğru x 'in onluk tabanındaki değeri, amaç fonksiyon değeri, seçilme olasılığı, buna göre bir sonraki nesildeki tahmin sayısı ve rulet çarkı döndüröldükten sonraki sayı bulunmaktadır.

Çizelge 5.8.a Genetik Algoritmalar

Dizi No	İlk Nesil	x değeri	$f(x) = x^2$	$\frac{f_1}{\sum f_1}$	$\frac{f_1}{f}$	Sayı
1	0 1 1 0 1	13	169	0,14	0,58	1
2	1 1 0 0 0	24	576	0,49	1,97	2
3	0 1 0 0 0	8	64	0,06	0,22	0
4	1 0 0 1 1	19	361	0,31	1,23	1
Toplam			1170	1,00	4,00	4,0
Ortalama			293	0,25	1,00	1,0
Maksimum			576	0,49	1,97	2,0

İlk nesildeki dizilerin ortalama uygunluk değeri 293 ve en iyi uygunluk değeri ise 576'dır. Bir sonraki adım bu dizilerden işlem havuzunun oluşturulması ve çaprazlama işleminin uygulanmasıdır. Bunlar Çizelge 5.8.b' de gösterilmiştir.

Çizelge 5.8.b (devam)

İşlem	Eşi	Yer	Yeni Nesil	X	$f(x)$
0 1 1 0 - 1	2	4	0 1 1 0 0	12	144
1 1 0 0 - 0	1	4	1 1 0 0 1	25	625
0 1 - 0 0 0	4	2	1 1 0 1 1	27	729
1 0 - 0 1 1	3	2	1 0 0 0 0	16	256
Toplam					1754
Ortalama					439
Maksimum					729

Çizelge 5.2.b' de üreme ve çaprazlama işlemleri uygulanmıştır. Mutasyon olasılığı 0.001 olarak alındığından dolayı ve küçük bir değeri olduğundan yukarıdaki dizilerin $5 \times 4 = 20$ bitinde görölmemiştir. Sonuçta hiçbir bit mutasyon sonucu değışmemiştir.

Çizelge 5.2.a ve çizelge 5.2.b' deki diziler (ve dolayısıyla temsil ettikleri onluk düzendeki

sayılar) karşılaştırıldığında en iyi sonuca yaklaşıldığı görülmektedir. Ortalama ve en yüksek uygunluk değeri yükselmiştir. Örneğin ortalama 293'ten 439'a yükselmiştir. Ancak genetik algoritmalar olasılığa da dayalı olduğu için bir nesilden diğerine geçerken iyileşme bu kadar hızlı olmayabilir. Hatta yeni neslin ortalaması bir öncekinden daha düşük olabilir. Fakat uzun vadede bulunacak sayılar mutlaka ilk nesli oluşturanlardan daha iyi amaç fonksiyon değerleri vereceklerdir.

Genetik algoritmalarındaki en son adım, bulunan yeni nesildeki elemanları, ilk nesil gibi ele alarak işlemleri bunların üzerinde yeniden uygulamaktır. Bu işlemler tekrar ve tekrar belli bir durma kriteri sağlanana kadar yürütülür. En sonra bulunan sayı tekrar onluk düzene çevrilerek kullanılır (Üstün, 2001).

6. KONTROL SİSTEMİ TASARLANMASI

6.1 Amaç

Sistemin amacı, minimum sarkaç osilasyonu ile birlikte arabayı istenilen x pozisyonuna getirmektir.

6.2 Sistemin PID Tasarımı

Sistemde $M = 10$ kg, $m = 5$ kg, $l = 1$ m, ve de $g = 10$ g/cm³ alınmış olup (2.30), (2.31) denklemleriyle tanımlanan $T_1(s)$ ve $T_2(s)$ çarpılırsa;

$$T_1(s) * T_2(s) = \frac{X(s)}{F(s)} = \frac{g + ls^2}{Mls^4 + g(M + m)s^2} \quad \text{formülünden,}$$

$$T_3(s) = T_1(s) * T_2(s) = \frac{X(s)}{F(s)} = \frac{1s^2 + 10}{10s^4 + 150s^2} \quad \text{elde edilir, (F girişine karşılık X çıkışı).}$$

Kapalı çevrim transfer fonksiyonu ise;

$$G(s) = \frac{K(s^2 + 10)}{10s^4 + (150 + K)s^2 + 10K} \quad \text{şeklindedir.}$$

Ziegler-Nichols yöntemi ile katsayıları bulmak için routh tablosunu oluşturursak:

Çizelge 6.1 Routh Tablosu

s^4	10	$150 + K$	$10K$
s^3	0	0	0
s^2	$(K + 150) / 2$	$10K$	
s^1	x	0	
s^0	$10K$		

2. satırda tüm elemanlar sıfır olduğu için bir üst satıra çıkılarak bu satırın türevi alınır ve katsayıları 2. satıra yerleştirilir. Buna göre;

$$P(s) = 10s^4 + (150 + K)s^2 + 10K$$

$P'(s) = 40s^3 + (300 + 2K)s$ olur. Böylece yeni routh tablosu şu şekilde elde edilir.

Çizelge 6.2 Yeni Routh Tablosu

s^4	10	$150 + K$	$10K$
s^3	40	$300 + 2K$	0
s^2	$(K + 150) / 2$	$10K$	
s^1	x	0	
s^0	$10K$		

2. satır sıfıra eşitlenerek $K = 150$ bulunur.

T_i ve T_d sabitlerini elde etmek için öncelikle s^1 satırındaki x'i bulmamız gerekmektedir.

$$x = \frac{(300 + 2K) \left(\frac{K + 150}{2} \right) - 400K}{\frac{K + 150}{2}} = \frac{2(K^2 - 100K + 22500)}{K + 150}$$

daha sonra sıfır olması mümkün olan bir satır sıfıra eşitlenip ω değeri bulunur.

2. satırdan;

$$40s^3 = (300 + 2K) = 0$$

$$s^3 = 15 \rightarrow \omega = 2,464$$

$$T = \frac{2\pi}{\omega} = 2,55 \text{ elde edilir.}$$

Elde edilen K ve T değerlerinin yardımıyla;

$$K_c = 0,6 \cdot K = 0,6 \cdot 150 \rightarrow K_c = 90$$

$$T_i = 0,5 \cdot T = 0,5 \cdot 2,55 \rightarrow T_i = 1,275$$

$$T_d = 0,125 \cdot T = 0,125 \cdot 2,55 \rightarrow T_d = 0,319 \text{ elde edilir.}$$

K_c , T_i ve T_d değerlerini K_p , K_i ve K_d değerlerine dönüştürsek;

$$K_c = K_p = 90$$

$$K_d = K_p * T_d = 28,7$$

$$K_i = (K_p / (60 * T_i)) = 1,18 \text{ elde edilir.}$$

Matlab'de bu değerleri vererek sistem tasarımını gerçekleştirirsek şu sonuçlar elde edilir;

```
>> Kp=90; Ki=1.18; Kd=28.7; numc=[Kd Kp Ki]; denc=[1 0]; Gc=tf(numc,denc)
```

Transfer function:

$$\frac{28.7 s^2 + 90 s + 1.18}{s}$$

```
>> num=[1 0 10]; den=[10 0 150 0 0]; G=tf(num,den)
```

Transfer function:

$$\frac{s^2 + 10}{10 s^4 + 150 s^2}$$

```
>> Gp=series(Gc,G)
```

Transfer function:

$$\frac{28.7 s^4 + 90 s^3 + 288.2 s^2 + 900 s + 11.8}{10 s^5 + 150 s^3}$$

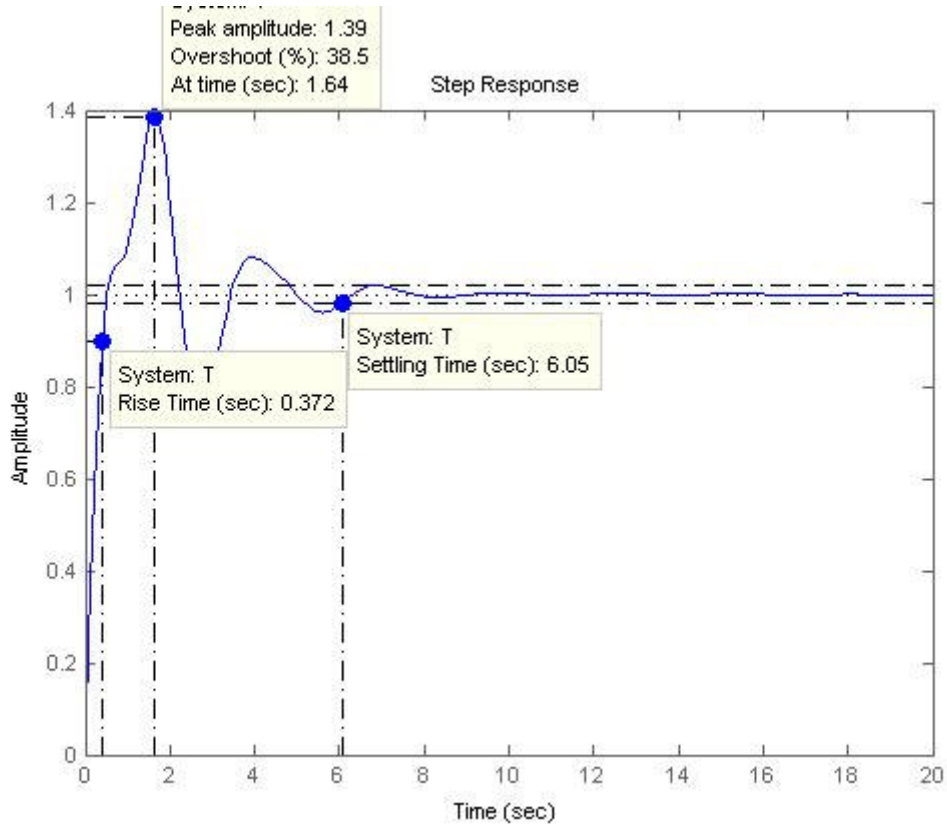
Birim geri beslemeli sistemin transfer fonksiyonu ve basamak cevabı,

```
>> T=feedback(Gp,1)
```

Transfer function:

$$\frac{28.7 s^4 + 90 s^3 + 288.2 s^2 + 900 s + 11.8}{10 s^5 + 28.7 s^4 + 240 s^3 + 288.2 s^2 + 900 s + 11.8}$$

```
>> step(T,20)
```



Şekil 6.1 $K_p = 90$, $K_d = 28,7$ ve $K_i = 1,18$ için sistemin birim basamak cevabı

Görüldüğü gibi sistem cevabı istenilenden çok uzaktadır. Oturma zamanı kabul edilebilir sınırlar içerisinde olmasına rağmen üst aşım (overshoot) %38,5 gibi bir değerdedir. Bu üst aşımı azaltabilmek için K_d ve K_i değerlerini arttırmak gerekmektedir. Sistem cevabı optimuma yakın olacak şekilde $K_d = 65$ ve $K_i = 3$ seçersek Matlab komutları ve sistem cevabı şu şekilde olacaktır;

```
>> Kp=90; Ki=3; Kd=65; numc=[Kd Kp Ki]; denc=[1 0]; Gc=tf(numc,denc)
```

Transfer function:

$$\frac{65s^2 + 90s + 3}{s}$$

```
>> num=[1 0 10]; den=[10 0 150 0 0]; G=tf(num,den)
```

Transfer function:

$$\frac{s^2 + 10}{10s^4 + 150s^2}$$

```
>> Gp=series(Gc,G)
```

Transfer function:

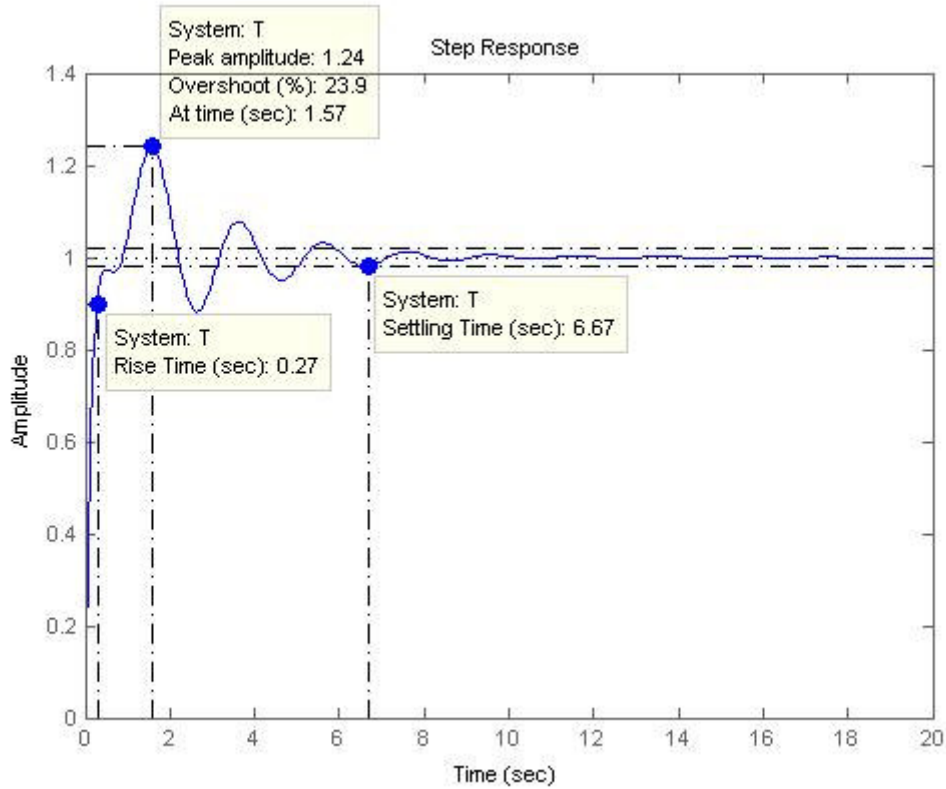
$$\frac{65s^4 + 90s^3 + 653s^2 + 900s + 30}{10s^5 + 150s^3}$$

```
>> T=feedback(Gp,1)
```

Transfer function:

$$\frac{65s^4 + 90s^3 + 653s^2 + 900s + 30}{10s^5 + 65s^4 + 240s^3 + 653s^2 + 900s + 30}$$

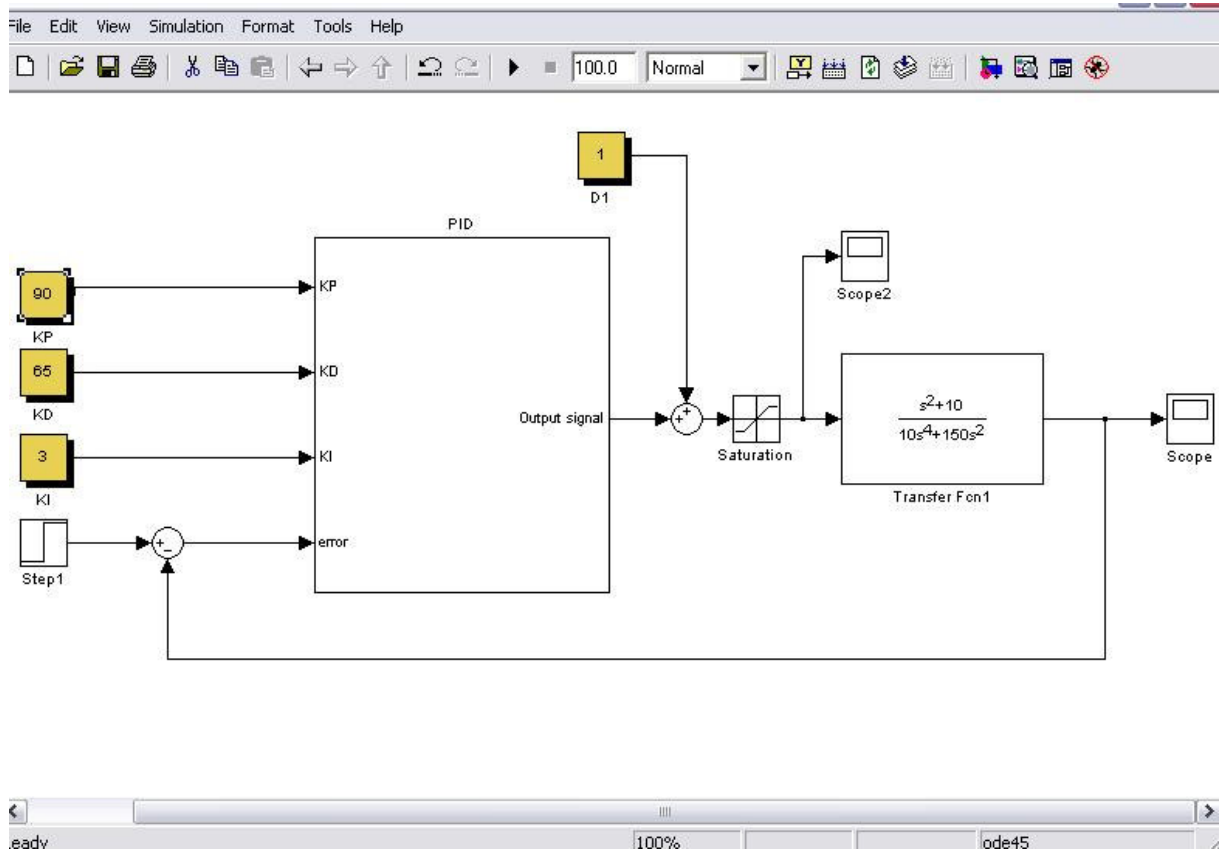
```
>> step(T,20)
```



Şekil 6.2 $K_p = 90$, $K_d = 65$ ve $K_i = 3$ için sistemin birim basamak cevabı

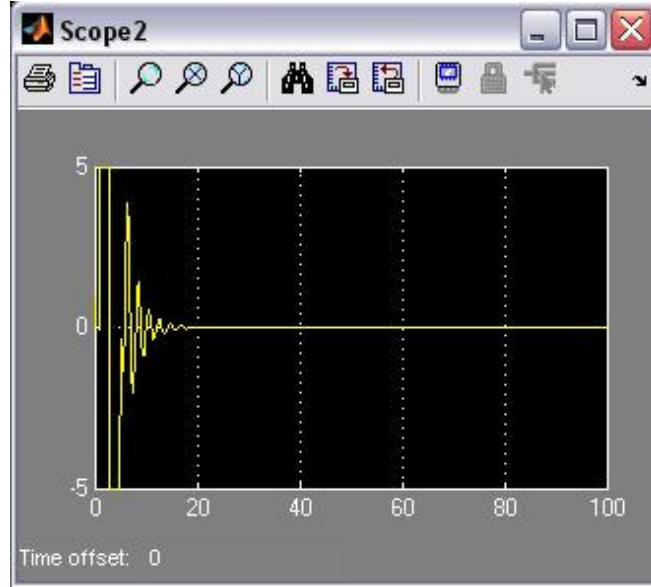
Şekil 6.2’de görüldüğü gibi yeni K_d ve K_i değerlerinin ardından üst aşım (overshoot) %23,9’a kadar gerilerken yükselme zamanı (rise time) 0,372’den 0,27’e gerilemiş, dolayısıyla sistem hem daha hızlı hala gelmiş hem de daha az üst aşım vererek kararlılığa yaklaşmıştır.

Bu değerleri Matlab Simulink'te bir PID bloğuyla beraber, disturbance katarak gerçeklersek;

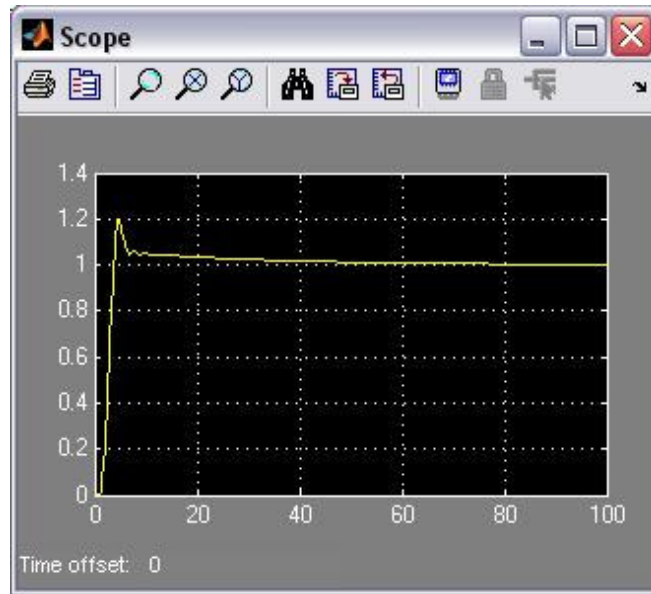


Şekil 6.3 Sistemin PID kontrol simulink diyagramı

Birim basamak girişinin uygulanan sistemde bir PID bloğun çıkışına bir birim disturbance uygulanmış ve daha sonra saturation'dan geçirilen sinyal sisteme uygulanmıştır. Sistemin çıkış sinyali Scope'tan, giriş sinyali ise Scope2'den izlenmektedir. Pratikte giriş sinyali 10^{14} gibi çok yüksek değerlere ulaştığı için buraya saturation konmuş ve girişin +5,-5 aralığında olması sağlanmıştır. Çıkışa da birim geri besleme alınarak girişte istenilen hata değeri kontrol edilmiştir. Sistemin çalıştırılması sonucunda scopelardan alınan değerler şu şekildedir;



Şekil 6.4 Satürasyondan geçmiş giriş sinyali

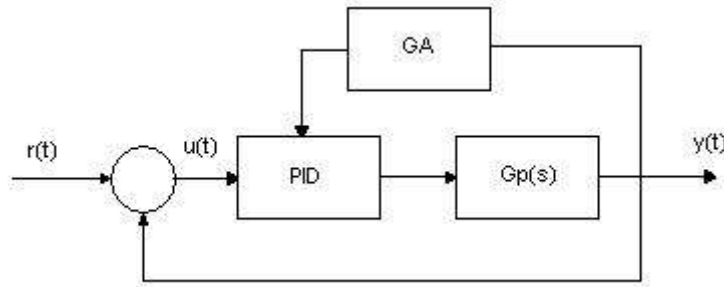


Şekil 6.5 Çıkış sinyali

Şekil 6.4'te satürasyon sonucu +5,-5 aralığında kalmış olan giriş sinyali ve Şekil 6.5'te ise sistem cevabı gözükmemektedir. Üst aşımın %20 ve oturma zamanının 40 saniye civarında olduğu gözükmemektedir. Üst aşım istenilen düzeye inmekle beraber bu sefer de oturma zamanı (settling time) yüksek çıkmıştır.

6.3 Sistemin Genetik Algoritmalarla Yaratılarak Tasarlanması

Genetik Algoritmalar yardımıyla, özellikle PID katsayıları optimize edilerek sistem için daha uygun sonuçlar elde edilebilir. Bu optimizasyon kriterleri ISE, ITAE, ITSE ve IAE gibi kriterler olup burada ISE, yani çıkıştaki hatanın karesel toplamının minimum olması kriteri üzerine program hazırlanmıştır. Böylelikle üst aşım, yükselme ve oturma zamanlarının daha iyi sonuç vermesi hedeflenmiştir. Genetik algoritmaların PID optimizasyonuna etkisi aşağıdaki şekil ile gösterilebilir.



Şekil 6.6 Genetik algoritma ile PID optimizasyon genel şeması

Yapılan Matlab çalışmasında, popülasyon genişliği 120 birey, çaprazlama oranı %70 ve mutasyon oranı %10 seçilerek optimum seçim sağlanmaya çalışılmıştır. Bu durumda her bir iterasyonda $120 \times 0,7 = 84$ birey çaprazlanmaya girmiş, $120 \times 0,1 = 12$ birey ise mutasyona uğramıştır. Cmax, sistemde denenerek, mümkün olan en küçük değer olarak 0,14 tespit edilmiş ve GA'yı durdurmak için hata eşiği olarak 0,002 seçilirken, K_p , K_i , K_d alt ve üst limitleri bir önceki bölümde belirlenen değerler arasında alınmıştır. 25 iterasyon sürecektir olan bu programın her iterasyonunda her bir birey için üretilen K_p , K_i , K_d değerlerinden birim basamak cevabına bakılarak ISE performans kriteri hesaplanır. 25 iterasyon bitmeden veya 25 iterasyonun sonucunda minimum toplam karesel hatayı veren bireye ait K_p , K_i , K_d değerleri program tarafından seçilir ve bu değerler sistemin değerleri olarak atanır (25 iterasyon bitmeden 0,002'lik hata eşiğinin altına inilirse operasyon duracaktır). Bu genetik algoritma programının seçim işleminde, en çok kullanılan yöntem olan Rulet Çarkı sistemi kullanılmış ve çaprazlama yöntemi olarak da tek noktalı çaprazlama metodu uygulanmıştır.

Genetik algoritma optimizasyonu için yazılan programda GenetikPID.m dosyası çalıştırılmasıyla programın komut çıktıları ve birim basamak cevabı şu şekilde elde edilmiştir;

```
>> GenetikPID
```

```
OrnekPeriyot = 0.0020
```

$T_p = 40$

Genetik Algoritma Optimizasyon işlemi sürüyor.....

(İşlemi herhangi bir anda durdurmak için

Ctrl+c basabilirsiniz)

!!İterasyon sayısı maksimum iterasyon sayısı 25 ye ulaştı!!

PID katsayı optimizasyonu tamamlandı

***** GENETİK OPTİMİZASYON RAPORU *****

*****Genetik Algoritma Parametreleri:

Populasyondaki kromozom sayısı=120

Her kromozom 39 bit ile temsil edilmiştir

Caprazlama Olasılığı=0.700000

Mutasyon Olasılığı=0.100000

PID katsayıları değişim aralığı=(3.000000 90.000000)

Maksimum İterasyon Sınırı=25

İterasyonu durduran hata eşiği=0.002000

$C_{max}=1.400000e-001$

*****Elde Edilen PID Katsayıları:

25. iterasyon sonunda elde edildi:

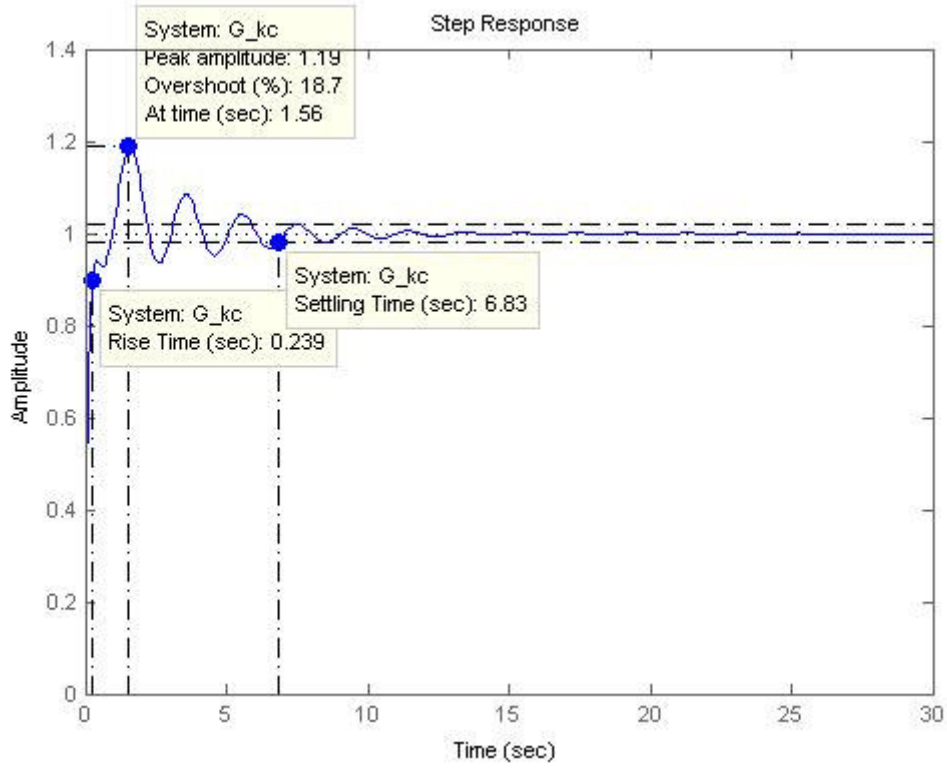
Optimum $k_p=60.511355$

Optimum $k_i=9.267399$

Optimum $k_d=89.235165$

Toplam Karesel Ortalama 0.002190

* Birim basamak giriş için pid cevabı figure No. 1 de çizilmiştir (Şekil 6.7)



Şekil 6.7 Genetik algoritmayla hesaplanan sistemin birim basamak cevabı

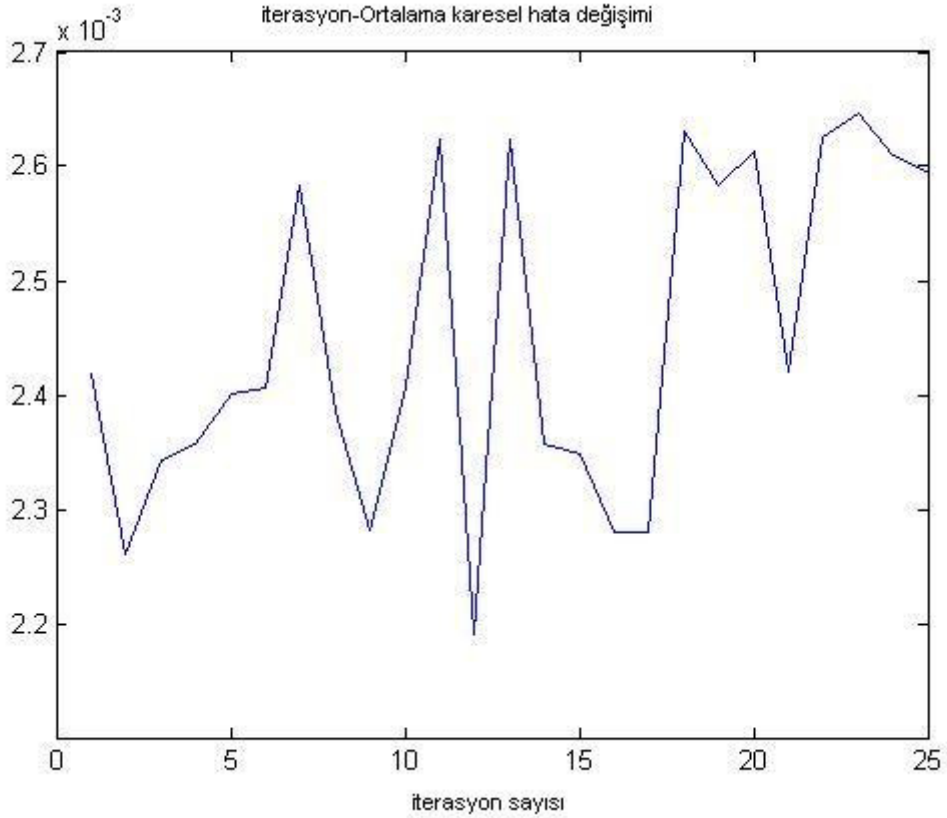
Şekil 6.7’de görüldüğü gibi genetik algoritma kullanılarak yapılan PID optimizasyonu ile sistemin üst aşımı %18,7, oturma zamanı 6,83 saniye ve yükselme zamanı 0,239 saniye olarak belirlenmiştir.

Daha önce aynı sistem için Ziegler-Nichols yardımıyla bulduğumuz parametreler ile GA kullanarak elde edilen parametrelerin karşılaştırması aşağıdaki çizelgededir;

Çizelge 6.3 Hesaplanan farklı PID parametreleri için üst aşım, yükselme ve oturma zamanı karşılaştırması

Yöntem/Sonuç	K_p	K_i	K_d	üstaşım(%os)	t_s	t_r
Ziegler-Nichols	90	1,18	28,7	38,5	6,05	0,372
Optimize edilmiş Ziegler-Nichols	90	3	65	23,9	6,67	0,27
Genetik Algoritma Sonuçları	60,5	9,27	89,2	18,7	6,83	0,239

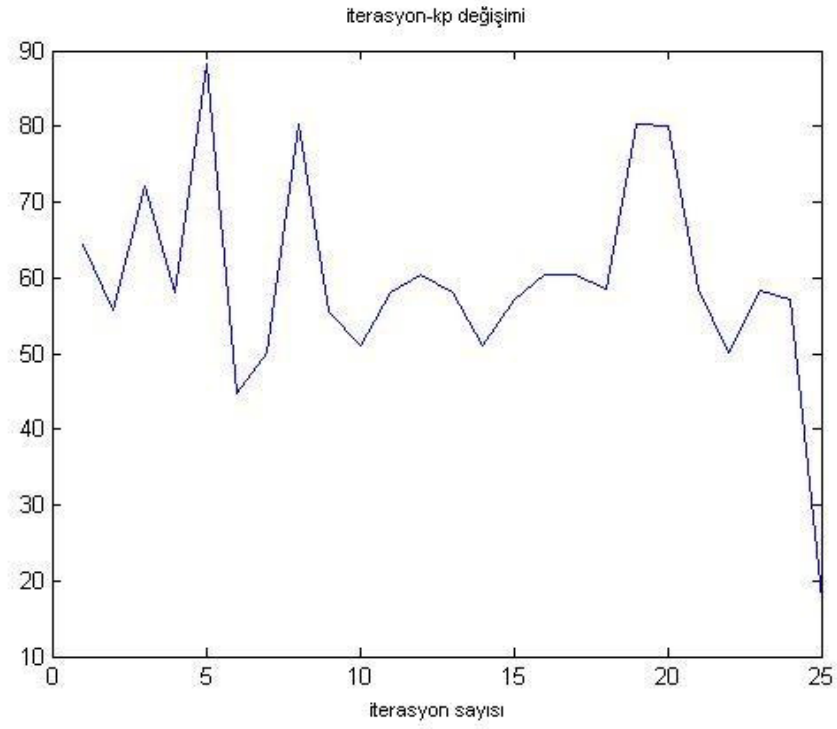
Çıkışta maksimum toplam karesel hatayı minimum yapan K_p , K_i , K_d katsayılarının hesaplanmasıyla beraber, üst aşım, yükselme ve oturma zamanlarında daha optimum sonuçlar elde edilmiştir. Popülasyondaki 120 bireyin 36'şar biti bulunmaktadır ve bunların 12 biti K_p 'yi, 12 biti K_i 'yi ve 12 biti de K_d 'yi temsil etmektedir. 25 iterasyon boyunca, önceden belirlenen minimum hata limiti 0,002'ye ulaşmadığı için genetik süreç 25 iterasyon bitimine kadar devam etmiştir. Sürecin sonundaki hata ise 0,00219 olarak ölçülmüştür. ISE kriteri kullanılarak bulunan, ortalama karesel hatanın değişim grafiğinin 25 iterasyon için değişimi Şekil 6.8' de verilmiştir.



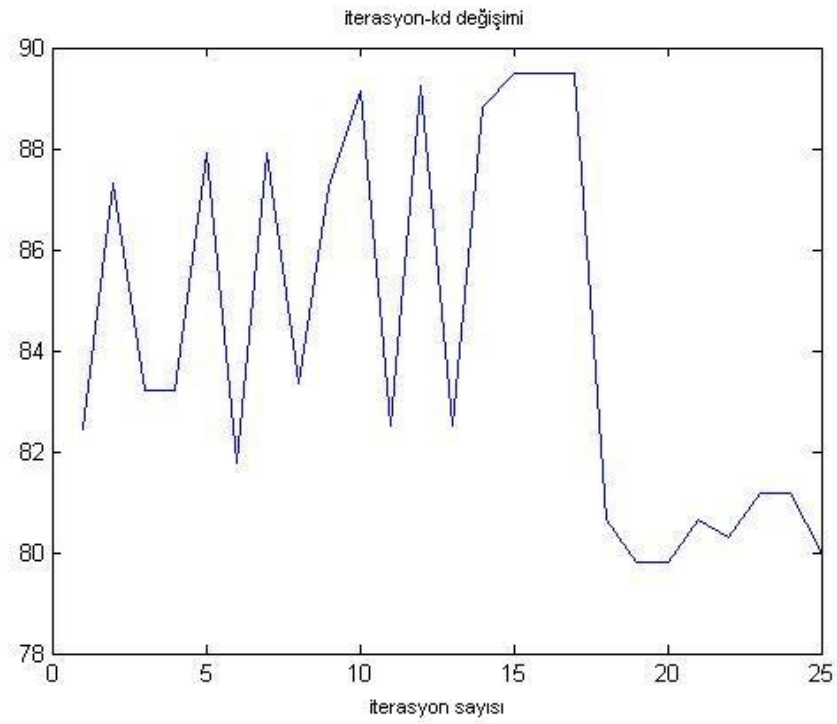
Şekil 6.8 Her iterasyon sonunda ortalama karesel hatanın değişimi

Genetik algoritma için en düşük çıkış hatasının 12. iterasyonda gerçekleştiği Şekil 6.8' de görülmektedir. Bu iterasyonda ortalama karesel hatanın değeri 0,00219 olmuş ve daha sonra başta belirlenen 0,002'nin altına düşmediği için 25 iterasyon tamamlanmıştır.

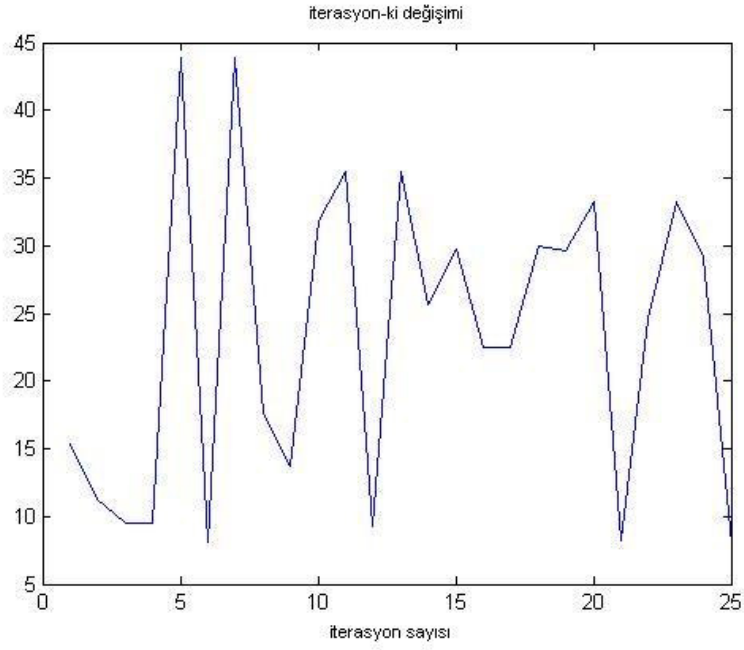
Genetik Algoritma boyunca K_p , K_d ve K_i 'nin aldığı değerlerin her bir iterasyondaki değişimleri aşağıdaki şekillerde verilmiştir;



Şekil 6.9 K_p 'nin iterasyonlar boyunca deęiřimi

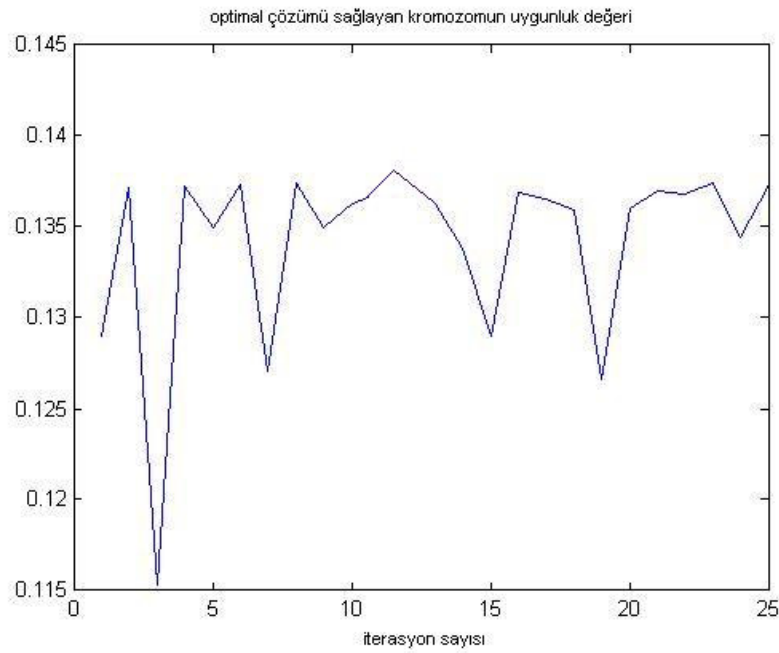


Şekil 6.10 K_d 'nin iterasyonlar boyunca deęiřimi



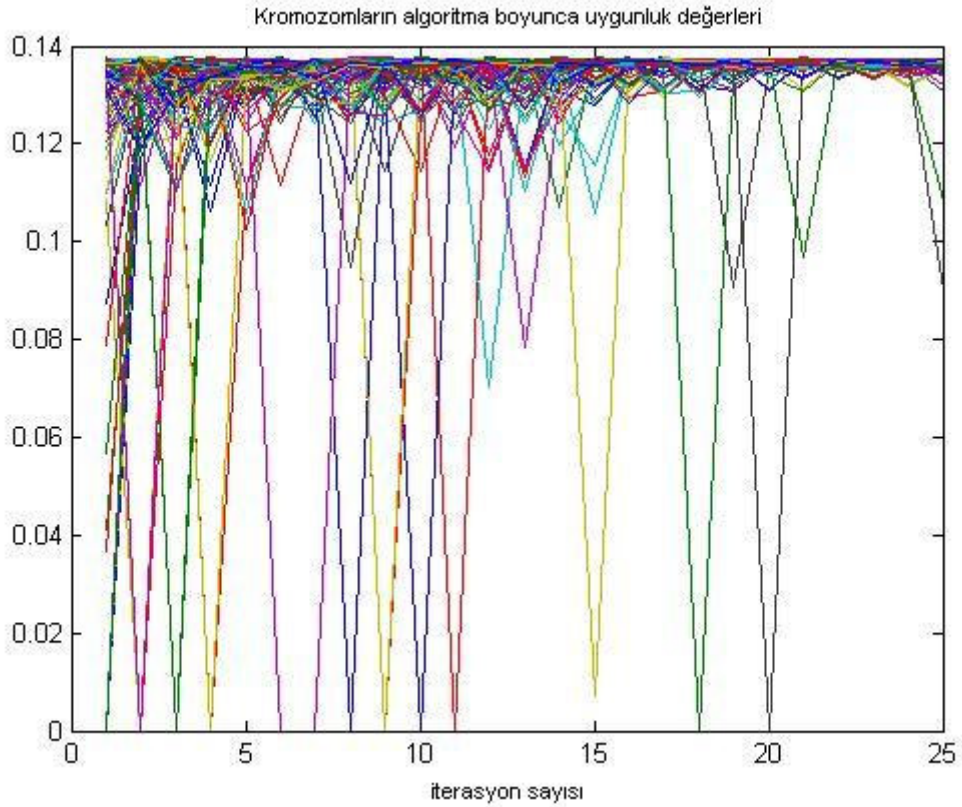
Şekil 6.11 K_i 'nin iterasyonlar boyunca değişimi

12. iterasyonda toplam ortalama karesel hatanın minimum olmasını sağlayan $K_p = 60,5$, $K_d = 89,2$ ve $K_i = 9,27$ değerleri Şekil 9-10 ve 11'den görülmektedir. Şekil 6.12' de, optimal çözümü (optimal ISE) sağlayan kromozomun uygunluk değerinin iterasyonlar boyunca değişimi incelendiğinde, ortalama karesel hatanın minimum olduğu 12. iterasyonda maksimum uygunluk değerine ulaştığı görülmektedir.



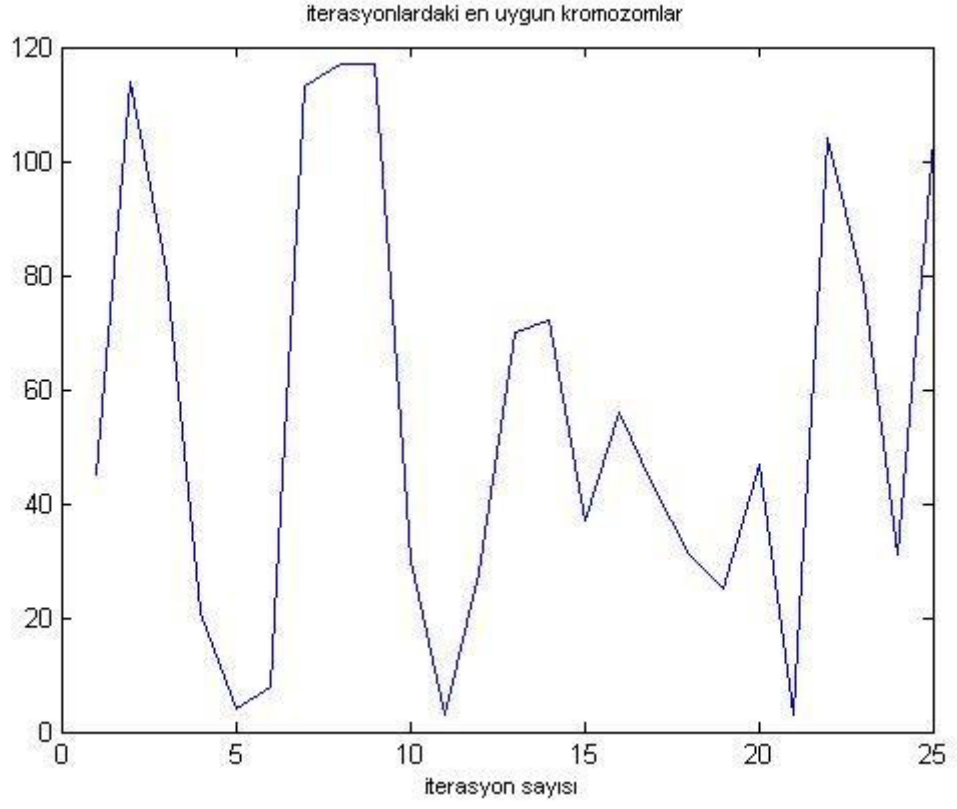
Şekil 6.12 Optimal çözümü sağlayan kromozoma ait uygunluk değeri değişimi

Toplam 120 kromozom bulunup, 25 iterasyon boyunca her bir kromozom için uygunluk değeri sisteme gönderildiğinden toplam $120 \times 25 = 3000$ farklı uygunluk değeri elde edilmiştir. Bu 3000 uygunluk değerini gösteren grafik Şekil 6.13'teki gibidir. Genetik algorithma oluşturulan ilk popülasyon rastlantısal olarak oluşturulur ve bu yüzden uygunluk değerlerinin ilk iterasyonda sıfıra daha yakın olduğu gözükmemektedir. İterasyon sayısı arttıkça uygunluk değerleri de artmış ve son iterasyonlarda ortalama olarak en yüksek uygunluk değerlerine ulaşılmıştır.



Şekil 6.13 Genetik algoritma boyunca kromozomların uygunluk değerleri değişimi

Şekil 6.14' te, 25 iterasyon boyunca her bir iterasyonda 120 birey içerisinde en yüksek uygunluk değeri gösteren birey gösterilmiştir. Genetik algoritmanın Matlab'de hazırlanan programı, sonuçları ve program bitiminde grafiklerin elde edilmesinin sağlayan komutlar Ekler'de bulunabilir.



Şekil 6.14 Her iterasyonda en iyi olan kromozom (birey)

7. SONUÇLAR

PID kontrolörler eski ve klasik bir kontrol yöntemi olmalarına rağmen günümüzde endüstride halen çok yüksek oranda kullanılmaktadırlar. Bunun nedeni kazanç, türev ve integral parametreleri kullanılarak sistemin kontrolünün iyi bir şekilde sağlanması, kalıcı hal hatasının sıfırlanmasıdır. Ayrıca maliyet ve uygulama yönünden de PID kontrolörler uygundur. Fakat yukarıda belirtilen kazanç, türev ve integral parametreleri iyi ayarlanmazsa bu kontrolün verimi ve uygunluğu düşer.

Bu çalışmada önce Ziegler-Nichols metodu yardımıyla K_c , T_i ve T_d değerleri hesaplanmış, daha sonra bu değerlerden K_p , K_d ve K_i 'ye geçilerek Matlab'de sistem çıkışı elde edilmiştir. Simulink'te de aynı şekilde PID bloğu ve disturbans konularak sistemin simülasyonu yapılmış ve oradan da bazı sonuçlar elde edilmiştir. Daha sonra Genetik Algoritma kullanılarak PID katsayılarının optimizasyonu yapılmıştır. Burada ISE kriteri baz alınarak, minimum karesel hatanın integral değerinin üst aşım ve yerleşme süresini minimum yapması beklenmiştir.

Elde edilen sonuçlara ve grafiklere bakıldığı zaman Genetik Algoritma kullanıldığında elde edilen sonuçların, sadece PID kontrolör kullanılarak elde edilen sonuçlardan daha iyi olduğu (aşımında %21,8 iyileşme, yükselme zamanında %11,5 iyileşme) net bir şekilde gözükmemektedir. Böylelikle ISE performans kriterinin çıkış hatasını minimize ederken maksimum üst aşım yükselme ve yerleşme süresini de azalttığı gözlemlenmiştir. Genetik Algoritmalar, sistemin tam olarak bilinmediği, parametrik belirsizliğin yüksek olduğu sistemlerde PID kontrolörlere göre çok daha iyi sonuç vermektedirler. Ele alınan sistemin belirsizliği çok yüksek olmasa da Genetik Algoritmalar bize daha iyi bir sonuç vermiştir. Genetik Algoritma ile optimizasyon yapılırken dikkat edilmesi gereken kriterlerin olduğu görülmüştür. Özellikle sistemin K_p , K_i ve K_d değerlerinin aralığının belirlenmesi süreci mutlak olarak etkilemektedir. Genetik Algoritmanın tüm kontrol parametreleri etkili kullanılırsa elde edilen sonuç optimuma yakınsayacaktır.

Bu tezde katsayıları uzun denemeler neticesi en uygun şekilde seçilen PID kontrolörle, katsayıları GA ile seçilmiş PID kontrolörün karşılaştırılması amacıyla, matematiksel modeli lineerleştirilmiş bir sistemin kontrolü gerçekleştirilmiştir. GA'nın bu uygulamalardaki belirgin başarısını gösterdikten sonra, klasik PID'yle kontrolün yetersiz kaldığı nonlineer kontrol uygulamaları yapılmalıdır. Özellikle bu tezde ele alınan tepe vincinin nonlineer modeli, bir takım parametre belirsizlikleriyle birlikte GA ile ayarlanmış PID temelli kontrol denenmelidir. GA'nın esas gücü bu tür uygulamalarda daha iyi ortaya çıkacaktır.

KAYNAKLAR

- Altıparmak,F., (1996), “Genetik Algoritmalar ile Haberleşme Şebekelerinin Topolojik Optimizasyonu” , Gazi Üniversitesi Fen Bilimleri Enstitüsü
- Astrom, K.J. ve Hagglund, T., (1984), “Automatic Tuning of Simple Regulators with Specification on Phase and Amplitude Margins”, Automatica, Vol: 20,No:5
- Astrom, K.J., (1998), “Automatic Tuning of PID Controllers”, Research Triangle Park, N.C. , Instrument Society of America
- Baker, B. M. ve Ayechev, M. A. (2002), “A Genetic Algorithm For The Vehicle Routing Problem”, Computers & Operations Research, s. 1-14
- De Jong, K.A., (1998), “An analysis of the behaviour genetic adaptive systems, PhD Dissertation”, University of Michigan
- Dengiz, B. ve Altıparmak F., (1998), “Genetik Algoritmalar Genel Bakış” , Gazi Üniversitesi Mühendislik Mimarlık Fakültesi, Endüstri Mühendisliği
- Emel, Gül Gökay ve Taşkın, Çağatay, (2002), “Genetik algoritmalar ve uygulama alanları” Uludağ üniversitesi
- Goldberg, D. E., (1989), “Genetic Algorithms in Search, Optimization, and Machine Learning Reading”, Addison-Wisley
- Griffin, Ian, (2003), “On-line PID Controller Tuning using Genetic Algorithms”, DCU
- Gündoğdu, Ömer, (2005), “Optimal-Tuning of PID controller gains using genetic algorithms”, Pamukkale Üniversitesi
- Halder, Bibhrajit, (2002), “Anti-swing Control Of a Suspended Varying Load With a Robotic Crane”, College Of Engineering and Technology of Ohio University
- Houck, C.R., Joines, J. ve Kay, M., (1996), “A genetic algorithm for function optimisation: A Matlab implementation”, ACM Transactions on Mathematical Software
- Kandemir, Kaan, (2006), “Bir operatörsüz vinç için kutup yerleşimi ve LQR temelli minimum salınımlı konum kontrolü”, Yüksel Lisans Tezi, YTÜ
- Klaudiusz, Fatla, (1999), “Using Genetic Algorithms for Optimaization of PIDController” DLX4 Project, CIT
- Koivo, H.N. ve Tantt, J.T., (1991), “Tuning of PID Controllers" in Proceedings of Intelligent Tuning and Adaptive Control”, Singapore
- Korukoğlu, Serdar, (2003), “Genetik Algoritma Yaklaşım ve Yöneylem Araştırmasında Bir Uygulama”, Celal Bayar Üniversitesi
- Kuo, Benjamin C., (1995), “Otomatik Kontrol Sistemleri” Yedinci Baskı Prentice Hall
- Lee, H.H., (1998), “Modeling and Control of a Three-Dimensional Overhead Crane Journal of Dynamic Systems”, Measurement and Control
- Mathworks, “Genetic Algorithm and Direct Search Toolbox For Use with Matlab”
- Matsukuri, Yamamoto, (1997), “A Genetic Tuning algorithm of PID parameters”, Kaneda IEEE Transactions

Michalewicz,Z., (1992), “Genetic Algorithm + Data Structures = Evolution Programs”, Springer-Verlag

Muzoğlu, Nedim, (2005), “Genetik Algoritmalar Kullanılarak PID Katsayılarının Optimizasyonu”, Yüksek Lisans Tezi, YTÜ Fen Bilimleri Enstitüsü

Ogata, Katsuhiko, (2002), “Modern Control Engineering” Fourth Edition Prentice Hall

Ogata, Katsuhiko, (1994), “Designing Linear Control Systems With Matlab” Prentice Hall

Paula, Deasy, (2001), “PID Parameter optimisation using Genetic Algorithms – The Astrom Benchmark Test” DLX4 Project, CIT

Sarıoğlu, Kemal, (1999), “Otomatik Kontrol 1, 2” Birsen Yayınevi, İstanbul

Sivritepe, Murat, (2003), “The Stabilization of the Inverted Pendulum By Matlab Simulink” Yüksek Lisans Tezi, İTÜ Fen Bilimleri Enstitüsü

Tavakoli Saeed ve Mahdi, (2003), “Optimal Tuning of PID Controllers for first order plus time delay models using dimensional analysis”, ICCA

Valenzuela,C.L., (1995), “Evolution Divide and Conquer;a novel genetic approach to the TSP”, Imperial College

Yeniay, M.Ö.,(1999), “Taguchi Deney Tasarımı Problemlerine Genetik Algoritma Yaklaşımı”, Doktora Tezi , Hacettepe Üniversitesi Fen Bilimleri Enstitüsü

http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol1/tcw2/article1.html Genetic Algorithms

<http://www.engin.umich.edu/group/ctm/> Control Tutorials For Matlab

<http://www.mathworks.com> Matlab Tutorials

http://www.mmo.org.tr/muhendismakina/arsiv/2001/ekim/Genetik_Algoritma.htm Genetik algoritma ve Uygulama Alanları

EKLER**PIDSISTEM3.M**

```
% PIDSistem kapalı çevrim pid kontrol sistemini simule eder.
% In: Sisteme uygulanan kontrol işareti.(Birim basamak giriş işareti)
% Out: Sistemin birim basamağa cevabı
% OrnekPeriyot: Sistem ornekleme periyodu
% Tp test suresi
%***** PID kontrol sistemi parametreleri*****
% H(s) kontrol sistemi kutupları p1,p2; sıfırları z1,z2

b5=0;b4=0;b3=1;b2=0;b1=10;
a5=10;a4=0;a3=150;a2=0;a1=0;

x6=kd*b5; x5=(kd*b4+kp*b5); x4=(kd*b3+kp*b4+ki*b5); x3=(kd*b2+kp*b3+ki*b4);
x2=(kd*b1+kp*b2+ki*b3); x1=(kp*b1+ki*b2); x0=ki*b1;
y6=kd*b5; y5=(kd*b4+a5); y4=(kd*b3+kp*b4+a4); y3=(kd*b2+kp*b3+ki*b4+a3);
y2=(kd*b1+kp*b2+ki*b3+a2); y1=(kp*b1+ki*b2+a1);y0=ki*b1;
Pay=[x6 x5 x4 x3 x2 x1 x0];
Payda=[y6 y5 y4 y3 y2 y1 y0];
%%HOCANINORNEGIPay=[0 0 0.7*kd 0.7*(kp+0.5*kd) 0.7*(ki+0.5*kp) 0.5*0.7*ki];
%%HNINORNEGIPayda=[1 2.6 1.98 (0.565+0.35*kd) (0.05+0.35*kp) 0.35*ki];

t=0:OrnekPeriyot:Tp;
[Out]=step(Pay,Payda,t);
```

GENETIKPID.M

```
%***PID KATSAYILARI OPTİMİZASYONU İÇİN ÜÇ PARAMETRELİ GENETİK
OPTİMASYON***
```

```
% 3 boyutlu uzayda tanımlı bir fonksiyonun max noktasına giden genetik
% algoritma gerçekleştirilmiş ve pid kontrolor optimizasyonuna uygulanmıştır.
```

```
clear
%Genetik Algoritma parametreleri
popsize=120;%18; %Populasyon genişliği
Size=12; % Kp,Ki,Kd nin kaç bit ile temsil edileceği.
BitSize=3*(Size+1);%12; %Her bir kromozomun kaç bit ile temsil edileceği
pc=0.7;%0.2 %(pc*popsize 2ye bölünebilmeli. ayrıca popsize 3 bölünebilmeli)
pm=0.1;%Mutasyon olasılığı
Alt=3;%Parametreler için alt sınır.
Ust=90;%Parametreler için üst sınır.
itSay=25; % iterasyon adım sayısı
Cmax=0.14;% büyük bir tam sayı.
Elimit=0.002;%genetik optimizasyonu durdurmak için hata esigi

%PID Kontrol Sistemi Parametreleri
OrnekPeriyot=0.002%0.05%0.2;%Kontrol Sistemi ornekleme periyodu
Tp=40%3%12;%Hatanın inceleneceği üst sınır
```

```
N=(Tp/OrnekPeriyot);%Toplam ornek adedi
```

```
%Parametre ayarları kontrolu
```

```
if ne((pc*popsiz-round(pc*popsiz)),0)
```

```
    error('pc*popsiz tamsayı olacak şekilde seçilmeli');
```

```
end
```

```
if ne(((pc*popsiz)/2-round((pc*popsiz)/2)),0)
```

```
    error('pc*popsiz çarpımı 2 ile bölünebilir seçilmeli');
```

```
end
```

```
if ne((pm*popsiz-round(pm*popsiz)),0)
```

```
    error('pm*popsiz tamsayı olacak şekilde seçilmeli');
```

```
end
```

```
fprintf('Genetik Algoritma Optimizasyon işlemi sürüyor.....\n');
```

```
fprintf('(İşlemi herhangi bir anda durdurmak için\n');
```

```
fprintf('Ctrl+c basabilirsiniz)\n');
```

```
for it=1:itSay
```

```
%Random kromozom matrisi olusturur
```

```
if it==1
```

```
for i=1:popsiz
```

```
    for j=1:BitSize
```

```
        f=rand;
```

```
        if f>=0.5
```

```
            Kromozom(i,j)=1;
```

```
        else
```

```
            Kromozom(i,j)=0;
```

```
        end
```

```
    end
```

```
end
```

```
end
```

```
%Kromozom'i reel KromozomMatR dönüştürür.
```

```
%KromozomMatR 3xpopsiz boyutundadır. 1. stün kp, 2. stün ki, 3.stün kd
```

```
for i=1:popsiz
```

```
    for j=1:BitSize
```

```
        Krom(j)=Kromozom(i,j);
```

```
    end
```

```
    [KromozomMatR(i,1) KromozomMatR(i,2)
```

```
        KromozomMatR(i,3)]=B2r3(Krom,Size,Alt,Ust);
```

```
end
```

```
% UYGUN FONKSİYONU HESAPLANACAK
```

```
%*****
```

```
for i=1:popsiz
```

```
kpn=KromozomMatR(i,1);kin=KromozomMatR(i,2);kdn=KromozomMatR(i,3);
```

```
[O]=PIDSistem3(kpn,kin,kdn,OrnekPeriyot,Tp);
```

```
I=ones(N+1,1);
```

```
top=0;
```

```
for m=1:(N+1)
```

```

    e=(I(m)-O(m))^2;
    top=top+e;
end

```

```

E(i)=(1/N)*top;

```

```

%Ekr(i)=E;
%pause
if E(i)>Cmax
    Ekr(i)=0;
else
    Ekr(i)=Cmax-E(i);
    HistEkr(it,i)=Ekr(i);
end
end

```

```

%*****

```

```

%Genetik Algoritma random başlangıç kromozomunu KromozomMatRilk matrisinde
%saklar.Başlangıç uygunluk değeri Ekrilk da saklanır.

```

```

%Eort kromozomların ortalama hatası
[minE minI]=min(E);
HistE(it)=minE;
Histkp(it)=KromozomMatR(minI,1);
Histki(it)=KromozomMatR(minI,2);
Histkd(it)=KromozomMatR(minI,3);
HistI(it)=minI;

```

```

if minE< Elimit
    fprintf('!!Hata Eşik Değerinin Altına Ulaştı!!\n');
    break;
end

```

```

if ne(it,itSay)

```

```

%*****GENETİK ALGORİTMA*****

```

```

%*****(Seçim,Çaprazlama,Mutasyon)*****

```

```

%SEÇİM İŞLEMİ

```

```

%Rulet tekeri kullanılır.

```

```

Sk=Secim(Ekr,popsize); % Seçim yapılır ve seçilen kromozomların indexi Sk de tutulur

```

```

%Seçilen kromozomlar kromozom matrisine yazılır

```

```

for i=1:popsize
    for j=1:BitSize
        KromozomS(i,j)=Kromozom(Sk(i),j);
    end
end

```

```

Kromozom=KromozomS;
%pause

```

```

%ÇAPRAZLAMA İŞLEMİ

```

```

%popsize.pc tane kromozom rasgele seçilir.
for i=1:round(popsize*pc)
    S=round(popsize*rand);
    if S==0
        Secilen(i)=1;
    else
        Secilen(i)=S;
    end
end

end

%(popsize.pc)/2 tane kromozom çifti için çaprazmanın hangi bitten başlayacağı
%rasgele belirlenir
for i=1:round((popsize*pc)/2)
    h=round(BitSize*rand);
    if h==0
        SecBit(i)=1;
    else
        SecBit(i)=h;
    end
end

end

%Kromozom matrisi üzerinde, seçilen kromozom çiftleri için, seçilen
%çaprazlama noktasından itibaren bitlerin değiştirilmesi.

for i=1:round((popsize*pc)/2)
    for j=SecBit(i)+1:BitSize
        g=Kromozom(Secilen(2*i-1),j);
        Kromozom(Secilen(2*i-1),j)=Kromozom(Secilen(2*i),j);
        Kromozom(Secilen(2*i),j)=g;
    end
end

Kromozomc=Kromozom;

%MUTASYON İŞLEMİ
%popsize.pm tane kromozom rasgele seçilir.
for i=1:round(popsize*pm)
    SM=round(popsize*rand);
    if SM==0
        SecilenM(i)=1;
    else
        SecilenM(i)=SM;
    end
end

end

%(popsize.pm) tane kromozomun hangi bitinde mutasyon uygulanacağı belirlenir.
for i=1:round(popsize*pm)
    h=round(BitSize*rand);
    if h==0
        %SecBitM(i)=1;

```

```

        Kromozom(SecilenM(i),1)=not(Kromozom(SecilenM(i),1));
    else
        Kromozom(SecilenM(i),h)=not(Kromozom(SecilenM(i),h));
    end
end
end
end

[y t]=min(HistE);
Kp=Histkp(t);
Ki=Histki(t);
Kd=Histkd(t);
if it==itSay
    fprintf('!!Iterasyon sayısı maksimum iterasyon sayısı %d ye ulaştı!!\n',itSay);
end
fprintf('PID katsayı optimizasyonu tamamlandı\n');
fprintf('***** GENETİK OPTİMİZASYON RAPORU ***** \n');
fprintf('*****Genetik Algoritma Parametreleri: \n');
fprintf(' Populasyondaki kromozom sayısı=%d \n',popsiz);
fprintf(' Her kromozom %d bit ile temsil edilmiştir \n',BitSize);
fprintf(' Çaprazlama Olasılığı=%f \n',pc);
fprintf(' Mutasyon Olasılığı=%f \n',pm);
fprintf(' PID katsayıları değişim aralığı=(%f %f) \n',Alt,Ust);
fprintf(' Maksimum Iterasyon Sınırı=%d \n',itSay);
fprintf(' Iterasyonu durduran hata eşiği=%f \n',Elimit);
fprintf(' Cmax=%d \n',Cmax);
fprintf('\n');
fprintf('*****Elde Edilen PID Katsayıları:\n');
fprintf('%d. iterasyon sonunda elde edildi:\n',it);
fprintf(' Optimum kp=%f \n',Kp);
fprintf(' Optimum ki=%f \n',Ki);
fprintf(' Optimum kd=%f \n',Kd);
fprintf(' Toplam Karesel Ortalama %f\n',y);
fprintf('* Birim basamak giriş için pid cevabı figure No. 1 de çizilmiştir \n');
fprintf('Aşım: %f \n',max(O)-1);
[O]=PIDSistem3(Kp,Ki,Kd,OrnekPeriyot,Tp);
n=0:OrnekPeriyot:Tp;
plot(n,I,n,O)
title('PID Birim Basamak Giriş Çıkış Karakteristigi')
xlabel('t Sec')
ylabel('Output')
[b1 b2]=size(HistEkr);
for k=1:b1
    Krom1(k)=HistEkr(t,k);
end
HistI(t)

CAPRAZ.M
%Kromozom(i,j): i.kromozom j. biti
%Popsiz : kromozom sayısı
%pc : çaprazlama olasılığı

```

```

%pm : mutasyon olasılığı
clear
Kromozom=[ 1 0 0 1 0 1 0 1 1 0 0 0 0 1;
            0 0 1 0 0 1 1 0 0 1 1 0 1 1;
            1 1 1 1 1 0 0 0 0 0 0 0 0 1;
            0 1 0 0 0 0 0 0 1 1 1 1 1 0;
            1 0 1 0 1 0 0 0 0 0 1 1 0 1;
            0 1 1 1 0 0 0 0 1 0 0 1 0 0;
            1 1 1 1 0 0 1 1 0 0 1 0 0 1;
            0 0 1 0 1 1 0 1 1 0 1 0 1 1;
            1 0 1 0 1 1 1 0 0 0 1 1 1 1;
            1 0 0 0 0 0 1 0 0 1 0 0 1 0;
            0 0 0 1 0 1 0 0 0 1 0 0 1 0;
            1 0 1 0 0 0 0 0 0 1 1 1 1 1];

popsize=12;
BitSize=14;
pc=0.5;
pm=0.2;
%popsize.pc tane kromozom rasgele seçilir.
for i=1:round(popsize*pc)
    S=round(popsize*rand);
    if S==0
        Secilen(i)=1;
    else
        Secilen(i)=S;
    end
end

end

%(popsize.pc)/2 tane kromozom çifti için çaprazmanın hangi bitten başlayacağı
%rasgele belirlenir
for i=1:round((popsize*pc)/2)
    h=round(BitSize*rand);
    if h==0
        SecBit(i)=1;
    else
        SecBit(i)=h;
    end
end

end

%Kromozom matrisi üzerinde, seçilen kromozom çiftleri için, seçilen
%çaprazlama noktasından itibaren bitlerin değiştirilmesi.

for i=1:round((popsize*pc)/2)
    for j=SecBit(i)+1:BitSize
        g=Kromozom(Secilen(2*i-1),j);
        Kromozom(Secilen(2*i-1),j)=Kromozom(Secilen(2*i),j);
        Kromozom(Secilen(2*i),j)=g;
    end
end
end

```

```

Kromozomc=Kromozom;
%popsize.pm tane kromozom rasgele seçilir.
for i=1:round(popsize*pm)
    SM=round(popsize*rand);
    if SM==0
        SecilenM(i)=1;
    else
        SecilenM(i)=SM;
    end
end

end

%(popsize.pm) tane kromozomun hangi bitinde mutasyon uygulanacağı belirlenir.
for i=1:round(popsize*pm)
    h=round(BitSize*rand)
    if h==0
        %SecBitM(i)=1;
        Kromozom(SecilenM(i),1)=not(Kromozom(SecilenM(i),1));
    else
        Kromozom(SecilenM(i),h)=not(Kromozom(SecilenM(i),h));
    end
end
end

SECİM.M
function [S]=Secim(Ekr,PopSize)
% Ekr ile verilen kromozom uygunluk değerlerini
% kullanarak rulet-tahtası türü seçim yapar.
% S : Seçilen kromozomların indeksi
% PopSize : Populasyon genişliği

F=0;
for i=1:PopSize
    F=F+Ekr(i); %Toplam uygunluk değeri F hesaplanır.
end
if F==0
    fprintf('***** UYARI *****\n')
    fprintf('Uygunluk değerleri toplamı 0 dır. Bunu önlemek için:\n');
    fprintf(' 1.Genetik algoritmayı bir kaç kez tekrar çalıştırın. Sürekli bu durumla\n');
    fprintf('    karşılaşırsanız aşağıdaki 2. maddeyi uygulayın\n');
    fprintf(' 2.Cmax parametresini artırın! Unutmayın ÇOK büyük seçilen Cmax genetik\n');
    fprintf('    algoritmanın performansını olumsuz etkiler');
    error('Yukardaki uyarıyı okuyun')
end
for i=1:PopSize
    p(i)=Ekr(i)/F; %Her bir kromozomun seçilme olasılığı pi hesaplanır
end

for i=1:PopSize
    t=0;
    for j=1:i
        t=t+p(j);
    end
end

```

```

end
q(i)=t; %Rulet tekeri işlemi için qi değerleri
end

for i=1:PopSize
    r=rand; %Sıfırla Bir arasında random sayı üretilir
    for j=1:PopSize
        if r<q(1)
            S(i)=1;
        elseif r>q(j)
            S(i)=j+1; % Random sayıya göre seçim yapılır.
        end
    end
end
end

```

Matlabde GenetikPID.m çalıştırılarak sonuçlar alındıktan sonra aşağıdaki komutlar ve verdiği sonuçlar şu şekildedir;

```

t = en iyi kromozomun bulunduğu iterasyonu verir
HistI(t) = en iyi iterasyondaki kromozomun bulunduğu indeks (sütun numarası)
HistEkr = tüm iterasyonlar boyunca sisteme gönderilen her kromozoma karşı düşen uygunluk
           değerleri
HistE = en iyi uygunluk değerine sahip kromozomun hata değişim değerleri
y = en iyi kromozomun hatasının minimum olduğu değer ki bu noktada uygunluk değeri
    maksimum olur.
plot(HistEkr) = tüm iterasyonlar boyunca sisteme gönderilen her kromozoma karşı düşen
               uygunluk değerlerinin değişim grafiği (Şekil 6.13)
plot(HistE) = en iyi uygunluk değerine sahip kromozomun hata değişim değerlerinin değişim
              grafiği (Şekil 6.8)
plot(Histkd) = en iyi kromozom için elde edilen kd nin değişim grafiği (Şekil 6.10)
plot(Histkp) = en iyi kromozom için elde edilen kp nin değişim grafiği (Şekil 6.9)
plot(Histki) = en iyi kromozom için elde edilen ki nin değişim grafiği (Şekil 6.11)

```

```
>> HistI(t)
```

```
ans =
```

```
28
```

```
>> y
```

```
y =
```

```
0.0022
```

```
>> t
```

```
t =
```

```
12
```

```
>> HistE
```

```
HistE =
```

```
Columns 1 through 6
```

```
0.0024 0.0023 0.0023 0.0024 0.0024 0.0024
```

```
Columns 7 through 12
```

```
0.0026 0.0024 0.0023 0.0024 0.0026 0.0022
```

```
Columns 13 through 18
```

```
0.0026 0.0024 0.0023 0.0023 0.0023 0.0026
```

```
Columns 19 through 24
```

```
0.0026 0.0026 0.0024 0.0026 0.0026 0.0026
```

```
Column 25
```

```
0.0026
```

```
>> HistI
```

```
HistI =
```

```
Columns 1 through 10
```

```
45 114 81 21 4 8 113 117 117 30
```

```
Columns 11 through 20
```

```
3 28 70 72 37 56 43 31 25 47
```

```
Columns 21 through 25
```

```
3 104 78 31 104
```

DEN1.M (grafik çıktılarında üst aşım, yükselme-oturma zamanlarını veren program)

```
clear, clc
```

```
s=tf('s');
```

```
Gp=(s^2+10)/(10*s^4+150*s^2)
```

```
K=60.511; Ti=6.529; Td=1.475;
```

```
Gc=K*(1+Td*s+(1/(Ti*s)))
```

```
G_ac=Gc*Gp;
```

```
G_kc=feedback(G_ac,1)
```

```
figure
```

```
step(G_kc,30)
```

ÖZGEÇMİŞ

Adı Soyadı	Onur YAĞSAN	
Doğum tarihi	10.05.1980	
Doğum yeri	İstanbul	
İlkokul	1986-1990	Mehmet Akif Ersoy İlkokulu
	1990-1991	Halil Vedat Fıratlı Pansiyonlu İlkokulu
Ortaokul	1991-1994	Bahçelievler Ortaokulu
Lise	1994-1998	Ahmet Rıfat Canayakın Lisesi
Lisans	1998-2003	Yıldız Üniversitesi Elektrik-Elektronik Fakültesi Elektrik Mühendisliği Bölümü

Çalıştığı kurum(lar)

2000	TELEKOM - Türk Telekomünikasyon A.Ş.(Staj)
2001	ASEL Mimarlık-Mühendislik Ltd Aş. (Staj)
2005-Devam ediyor	MTM Yapı Denetim

