İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

KONTROL SİSTEMLERİ İÇİN BULANIK PID KONTROLÖRLERİN GENETİK ALGORİTMALAR YARDIMIYLA AYARLANMASI

Tezin Enstitüye Verildiği Tarih: 20 Mart 2007

Tezin Savunulduğu Tarih: 26 Nisan 2007

YÜKSEK LİSANS TEZİ
Müh. Ömer Güvenç KARAOĞLAN
504031132

Tez Danışmanı: Doç. Dr. Fuat GÜRLEYEN

Diğer Jüri Üyeleri: Prof. Dr. İbrahim EKSİN (İ.T.Ü)

Yrd. Doç. Dr. Osman Kaan EROL (İ.T.Ü)

MAYIS 2007

ÖNSÖZ

Bu tez çalışması boyunca yardımlarını ve ilgisini esirgemeyen danışman hocam Sayın Doç. Dr. Fuat Gürleyen' e, katkılarından ötürü Sayın Prof. Dr. İbrahim Eksin'e ve hayatımın her aşamasında olduğu gibi bu çalışma boyunca da bana olan desteklerinden ötürü aileme teşekkürü bir borç bilirim.

MART 2007 Ömer Güvenç Karaoğlan

İÇİNDEKİLER

KISALTMALAR v
TABLO LİSTESİ vı
ŞEKİL LİSTESİ vıı
SEMBOL LİSTESİ ıx
ÖZET x
SUMMARY xı

1. GİRİŞ 1

2. GENETİK ALGORİTMALAR 7
2.1. Genetik Algoritmalar Nedir? 7
2.2. Geleneksel Algoritma Yöntemleri 8
2.2.1. Genetik Algoritmaların Geleneksel Arama Yöntemleriyle Karşılaştırılması 9
2.3. Genetik Algoritmaların Kullanılma Nedenleri 9
2.4. GA' da Kullanılan Biyolojik Terimler 10
2.5. Basit Genetik Algoritma 11
2.6. Genetik Algoritmaların İşleyişi 14
2.6.1. Başlangıç Popülasyonunun Oluşturulması 17
2.6.2. Amaç ve Uygunluk Fonksiyonunun Seçimi 18
2.7. Seçim 19
2.7.1. Orantılı Seçim Yöntemi 20
2.7.1.1. Yerine Koyarak Stokastik Örnekleme 20
2.7.1.2. Yerine Koymadan Stokastik Örnekleme 21
2.7.1.3. Kalanı Stokastik Örnekleme 21
2.7.1.4. Stokastik Evrensel Örnekleme 21
2.7.2. Turnuva Seçim Yöntemi 22
2.7.3. Kararlı Durum Seçim Yöntemi 22
2.8. Çaprazlama 23
2.8.1. Tek Kesim Noktalı Çaprazlama 23
2.8.2. Çok Kesim Noktalı Çaprazlama 24
2.8.3. Düzgün Çaprazlama 24
2.9. Mutasyon 25

3. BULANIK MANTIK 27
3.1. Bulanık Kümeler 27
3.1.1. Üyelik Fonksiyonları 28
3.1.2. Bulanık Kümelerle İlgili Temel Kavramlar 29

3.1.3. Bulanık Kümelerde İşlemler 31
3.1.3.1. Bulanık Tümleme 31
3.1.3.2. Bulanık Birleşme Özelliği 32
3.1.3.3. Bulanık Kesişme Özelliği 32
3.1.3.4. Standart Bulanık Küme İşlemciler 32
3.2. Bulanık Gerektirme 35
3.3. Bulanık Sistemler 38
3.3.1. Bulanık Kural Tabanı 39
3.3.1.1 Mamdani Tipi Bulanık Kurallar 39
3.3.1.2 Tekli (Singleton) Tipi Bulanık Kurallar 40
3.3.1.3 Takagi-Sugeno Tipi Bulanık Kurallar 40
3.3.2. Bulanık Çıkarım Mekanizması 41
3.3.3. Bulandırıcı 41
3.3.3.1 Tekli Bulandırıcılar 42
3.3.3.2 Tekli Olmayan Bulandırıcılar 42
3.3.4. Durulayıcı 42
3.4. Bulanık Kontrolörler 45
3.4.1. Doğrudan Harekete Geçen (Direct – Action) Bulanık Kontrolörler 45
3.4.1.1. Bulanık PD Tipi Kontrolör 45
3.4.1.2. Bulanık PI Tipi Kontrolör 46
3.4.1.3. Bulanık PID Tipi Kontrolör 46
3.4.2. PID Kontrolör Kazancını Hesaplayanlar 47
3.4.3. Karma Kontrolörler 48

4. SİMÜLASYONLAR 50
4.1. Çalıştırılan program 50
4.2. Simülasyon Yapılan Sistemler 57

5. SONUÇLAR VE TARTIŞMA 77

KAYNAKLAR 79

EKLER 82

ÖZGEÇMİŞ 103

KISALTMALAR

EA	Evrim Algoritmaları
EP	Evrimsel Programlama
ES	Evrim Stratejileri
FIS	Fuzzy Interference System
FLC	Fuzzy Logic Controller
GA	Genetic Algoritmalar
GP	Genetik Programlama
HGA	Hiyerarşik Genetik Algoritma
ITAE	Integral Time Absolute Error
ITSE	Integral Time Square Error
HGA	Hiyerarşik Genetik Algoritma
KDGA	Kararlı Durum Genetik Algoritma
µGA	Mikro Genetik Algoritma
mGA	Dağınık Genetik Algoritma
PD	Proportional and Derivative Controller
PI	Proportional and Integral Controller
PID	Proportional, Integral and Derivative Controller
SGA	Simple Genetic Algorithm

TABLO LİSTESİ

		Sayfa No
Tablo 2.1:	GA'da çözüm aşamaları……..	15
Tablo 3.1:	Klasik gerektirme ile bulanık gerektirme bulanık gerektirme bağlacının çeşitli ifade biçimleri…….	35
Tablo 4.1:	Başlangıç açısına sahip sarkaç için kontrolörlerin ITSE değerleri……..	76

ŞEKİL LİSTESİ

		Sayfa No
Şekil 2.1:	GA'nın işleyişi……..	14
Şekil 2.2:	Rulet çemberi gösterimi……..	20
Şekil 2.3:	Çok kesim noktalı çaprazlama…….	24
Şekil 3.1:	$\mu_1(x)$ üyelik fonksiyonu……...	29
Şekil 3.2:	"Genç" bulanık kümesi…….	33
Şekil 3.3:	"Orta yaşlı" bulanık kümesi…….	33
Şekil 3.4:	"Yaşlı" bulanık kümesi…….	33
Şekil 3.5:	A_1 bulanık kümesinin tümleyeni……..	34
Şekil 3.6:	A_1 ve A_2 bulanık kümelerinin standart kesişim işlemi……	34
Şekil 3.7:	A_2 ve A_3 bulanık kümelerinin standart kesişim işlemi……	34
Şekil 3.8:	B ve C bulanık kümelerinin standart birleşim işlemi……..	34
Şekil 3.9:	Bulanık sistem yapısı……..	39
Şekil 3.10:	Singleton tipi bulandırıcı…….	42
Şekil 3.11:	Ağırlık merkezi durulama yöntemi……..	43
Şekil 3.12:	Alanların toplamının merkezi durulama yöntemi……	43
Şekil 3.13:	Maksimum durulayıcılar durulama yöntemi……	44
Şekil 3.14:	Yüksekliklerin ortalaması durulama yöntemi……..	44
Şekil 3.15:	Bulanık PD tipi kontrolör……	45
Şekil 3.16:	Bulanık PI tipi kontrolör……..	46
Şekil 3.17:	Bulanık PID tipi kontrolör(1)……..	47
Şekil 3.18:	Bulanık PID tipi kontrolör(2)……..	47
Şekil 3.19:	PID kontrolör kazancını hesaplayan bulanık kontrolör yapısı.	48
Şekil 3.20:	Karma kontrolör……...	48
Şekil 4.1:	c_m değerinin üyelik fonksiyonlarına etkisi……...	54
Şekil 4.2:	Bulanık kuralların oluşturulması…….	55
Şekil 4.3:	Birinci sistem için simulink modeli…….	58
Şekil 4.4:	Birinci sistemin çıkışı (y)……	59
Şekil 4.5:	Birinci giriş için üyelik fonksiyonları……..	59
Şekil 4.6:	İkinci giriş için üyelik fonksiyonları……	60
Şekil 4.7:	Çıkış üyelik fonksiyonları……	60
Şekil 4.8:	Kural tabanının oluşturduğu yüzey……..	61
Şekil 4.9:	Köşegen kural tablosu……..	62
Şekil 4.10:	Zaman gecikmeli sistem için çıkışlar……...	62
Şekil 4.11:	Zaman gecikmeli sistem için kontrol işaretleri……	63
Şekil 4.12:	Ters sarkaç sistemi…….	64
Şekil 4.13:	İkinci sistem için simulink modeli……...	64
Şekil 4.14:	Bozucu etki……..	65
Şekil 4.15:	İkinci sistemin çıkışı (y)……..	66
Şekil 4.16:	Birinci giriş için üyelik fonksiyonları……..	66
Şekil 4.17:	İkinci giriş için üyelik fonksiyonları……	67

Şekil 4.18: Çıkış üyelik fonksiyonları…… 67
Şekil 4.19: Kural tabanının oluşturduğu yüzey…….. 68
Şekil 4.20: Ters sarkaç sistemi için çıkışlar……... 69
Şekil 4.21: Kontrol işaretleri (0 – 0.4 sn)……... 69
Şekil 4.22: Kontrol işaretleri (1.8 – 2.2 sn)…… 70
Şekil 4.23: Kontrol işaretleri (3.8 – 4.2 sn)…..……... 70
Şekil 4.24: 0.25 radyan başlangıç açısı için sistem çıkışları…….. 71
Şekil 4.25: 0.25 radyan başlangıç açısı için kontrol işaretleri…….. 72
Şekil 4.26: -0.25 radyan başlangıç açısı için sistem çıkışları…….. 72
Şekil 4.27: -0.25 radyan başlangıç açısı için kontrol işaretleri…….. 73
Şekil 4.28: 1.5 radyan başlangıç açısı için sistem çıkışları…..…….. 73
Şekil 4.29: 1.5 radyan başlangıç açısı için kontrol işaretleri…….. 74
Şekil 4.30: -1.5 radyan başlangıç açısı için sistem çıkışları……. 74
Şekil 4.31: -1.5 radyan başlangıç açısı için kontrol işaretleri…… 75
Şekil E.1: 0.14 radyan başlangıç açısı için sistem çıkışları…….. 83
Şekil E.2: 0.14 radyan başlangıç açısı için kontrol işaretleri..…….. 84
Şekil E.3: -0.14 radyan başlangıç açısı için sistem çıkışları……. 85
Şekil E.4: -0.14 radyan başlangıç açısı için kontrol işaretleri…….. 86
Şekil E.5: 0.25 radyan başlangıç açısı için sistem çıkışları…….. 87
Şekil E.6: 0.25 radyan başlangıç açısı için kontrol işaretleri…… 88
Şekil E.7: -0.25 radyan başlangıç açısı için sistem çıkışları……. 89
Şekil E.8: -0.25 radyan başlangıç açısı için kontrol işaretleri…….. 90
Şekil E.9: 0.55 radyan başlangıç açısı için sistem çıkışları…….. 91
Şekil E.10: 0.55 radyan başlangıç açısı için kontrol işaretleri…… 92
Şekil E.11: -0.55 radyan başlangıç açısı için sistem çıkışları……. 93
Şekil E.12: -0.55 radyan başlangıç açısı için kontrol işaretleri…….. 94
Şekil E.13: 0.9 radyan başlangıç açısı için sistem çıkışları…… 95
Şekil E.14: 0.9 radyan başlangıç açısı için kontrol işaretleri…….. 96
Şekil E.15: -0.9 radyan başlangıç açısı için sistem çıkışları…… 97
Şekil E.16: -0.9 radyan başlangıç açısı için kontrol işaretleri…… 98
Şekil E.17: 1.5 radyan başlangıç açısı için sistem çıkışları…… 99
Şekil E.18: 1.5 radyan başlangıç açısı için kontrol işaretleri…….. 100
Şekil E.19: -1.5 radyan başlangıç açısı için sistem çıkışları…… 101
Şekil E.20: -1.5 radyan başlangıç açısı için kontrol işaretleri…….. 102

SEMBOL LİSTESİ

$\mu_A(x)$	Üyelik Fonksiyonu
G	Generation Gap
q	Turnuva büyüklüğü
P_c	Çaprazlama Oluşma Olasılığı
P_m	Mutasyon Oluşma Olasılığı
c	Bulanık Tümleme
i, *	Bulanık Kesişim
u, $\oplus$	Bulanık Birleşim
supp (.)	Bir Bulanık Kümenin Dayanağı
hgt (.)	Bir Bulanık Kümenin Yüksekliği
$^{\alpha}A$	α kesim
$^{\alpha+}A$	Güçlü α kesim
$\Rightarrow$	Gerektirme
τ	Bulanık Gerektirme
u	Kontrol işareti
e	Hata İşareti
$\dot{e}$	Hatanın Türevi
K_p	PID'ye ait oransal katsayı
K_d	PID'ye ait türev katsayısı
K_i	PID'ye ait integral katsayısı

ÖZET

KONTROL SİSTEMLERİ İÇİN BULANIK PID KONTROLÖRLERİN GENETİK ALGORİTMALAR YARDIMIYLA AYARLANMASI

Genetik algoritmalar, bugüne kadar birçok uygulamada etkinliği kanıtlanmış bir rastgele arama (random search) metodudur. Genetik algoritmalar, özellikle analitik çözümü olmayan veya zor yapılabilen, çözüm uzayının taranmasını gerektiren ve problem hakkında önbilgiye sahip olunmadığı durumlarda kullanılmaktadır. Bu algoritma problemin çözümüne ulaşmak için önbilgi olarak temelde sadece amaç fonksiyonunun seçimine ihtiyaç duyar.

Klasik mantıktaki kesin ifadelerin yerini bulanık mantıkta 'büyük, orta, küçük' gibi kesin olmayan dilsel ifadeler almaktadır. Bulanık mantık kullanılarak sistemlerin kontrolü ise son yıllarda artan bir ilgiyle kullanılmaktadır. Bulanık mantık kontrolün en iyi olduğu uygulama alanları lineer olmayan, iyi tanımlanmamış, zamanla değişen sistemlerdir. Sistemlerin karmaşıklığı arttıkça bulanık kontrol, klasik kontrolün yerini almaktadır.

Genetik algoritmaların bulanık mantık kontrolörü (FLC) tasarımında kullanılması konusunda literatürde birçok değişik yöntem mevcuttur. Bu tez çalışmasında kullanılan yöntemde genetik algoritmalar, kullanılan bulanık mantık kontrolöründeki ölçekleme katsayılarının ayarlanmasında, kullanılan üçgen tipi üyelik fonksiyonlarının sayısının ve tepe noktalarının belirlenmesinde ayrıca kural tabanının oluşturulması aşamalarında kullanılmaktadır.

Kullanılan bu yöntemin başarımını görmek için Matlab m-dosyaları (m-files) ve Simulink modelleri oluşturularak simülasyonlar yapılmış, sonuçlar katsayıları genetik algoritmalar tarafından ayarlanan klasik-PID kontrolör ve sabit FIS yapısına sahip, ölçekleme çarpanları genetik algoritmalar tarafından ayarlanan bulanık mantık kontrolörü ile karşılaştırılmıştır.

SUMMARY

TUNING OF FUZZY PID CONTROLLERS BY GENETIC ALGORITHMS FOR CONTROL SYSTEMS

Genetic algorithms are random search methods whose effectiveness have been proved by many applications up until now. Genetic algorithms are especially used when analytical solution is difficult or impossible, when it is required to search the solution space and there is no advance information about the problem. Nonetheless, these algorithms mainly need only the selection of objective function as a prerequisite to solve a problem.

Crisp definitions in classical logic are displaced by non-crisp linguistic terms as "big, medium, small" in fuzzy logic. Control of systems using fuzzy logic has been used popularly in the recent years. The best application areas of fuzzy logic control are non-linear, not well defined and time-variant systems. As the complexity of system increases, fuzzy control supersedes the conventional control methods.

There are so many methods in the literature regarding the use of genetic algorithms for the design of fuzzy logic controllers (FLCs). In this thesis, genetic algorithms are used to tune the controller's scaling coefficients, finding the number and peak points of triangular membership functions and composing the rule base stages.

To evaluate the performance of this method simulations have been made by using Matlab m-files and Simulink models, the results have been compared with classical-PID controller whose coefficients have been tuned by genetic algorithms and fuzzy logic controller, which has a constant FIS, has its scaling factors been tuned by genetic algorithms.

1. GİRİŞ

Esnek programlamayı (SC) oluşturan yapay sinir ağları, bulanık mantık ve evrim algoritmalarından bir veya birkaçının birlikte kullanılması ile oluşan yöntemlerin uygulanması son yıllarda gittikçe önem kazanmıştır. Bu yöntemlerden evrim algoritmaları doğal genetiği, yapay sinir ağları insanın sinir ve öğrenme yapısını, bulanık mantık ise insanın düşünme biçimini modelleyerek çözüm arar.

Evrim algoritmaları, doğal genetik yapıyı modelleyerek çözüm arayan bir arama algoritmasıdır. Evrim algoritması ilk olarak 1960 yılında I. Rechenberg'in "Evolutionsstrategie" (evrimsel stratejiler) çalışmasıyla ortaya atılmıştır [1]. Bir diğer evrim algoritması çeşidi olan Evrimsel Programlama (Evolutionary Programming) günümüzde kullanılan genetik algoritmalara benzeyen bir yöntem olup Fogel, Owens ve Walsh tarafından geliştirilmiştir.

Makine öğrenmesi (machine learning) konusunda çalışan John Holland 1975 yılında " Adaptation in Natural and Artificial Systems " adlı kitabını yayımlamıştır [3]. Bu çalışmada Holland tek bir mekanik yapının öğrenme yeteneğini geliştirmek yerine böyle yapılardan oluşan bir topluluğun çoğalma, çiftleşme, mutasyon vb. genetik süreçlerden geçerek başarılı yeni bireyler oluşturduğunu gözlemlemiş, doğadaki evrimsel işlemin benzeri bir yapı elde etmiştir. Holland'ın çalışmasından sonra bu yöntemin adı "Genetik Algoritmalar" (GA) olarak yerleşmiştir. Genetik algoritmaların popülerlik kazanması D. E. Goldberg'in 1983 yılında hazırlamış olduğu " Computer-aided gas pipeline operation using genetic algorithms and rule learning " konulu doktora tezinden sonra olmuştur [2]. D.E. Goldberg, K.A. DeJong, L.Davis, Bäck, T.Z. Michalewicz, ve J.J. Grefenstette genetik algoritmanın yapısı ve uygulaması konusunda oldukça önemli çalışmalar yapmıştır [4-7].

Evrim Algoritmaları, doğal evrim döngüsünü bir problem çözme biçimi olarak kullanan birden çok yöntemin genel adıdır ve dört temel çeşidi vardır. Bunlar Evrimsel Programlama (EP), Evrim Stratejisi (ES), Genetik Programlama (GP) ve Genetik Algoritmalardır (GA).

Evrimsel Programlama (EP) , Lawrence Fogel'in tanımladığı stokastik optimizasyon tekniğidir. Eşeysiz üreme yöntemine dayanır, dolayısıyla çaprazlama operatörü yoktur ve mutasyon operatörü üzerine odaklanmıştır. Evrimsel Programlamanın çalışması üç temel adımda açıklanabilir. Bu adımlar:

i) Başlangıç popülasyonu rastgele oluşturulur. Burada, popülasyon büyüklüğü oldukça önemlidir ve optimizasyonun performansını belirler. Fakat, bu büyüklüğün ne olacağına dair kesin bir çözüm yoktur.

ii) Popülasyondaki her birey mutasyona uğratılır. Mutasyon operatörün tipi ve olasılığı çok önemlidir.

iii) Popülasyondaki her bireyin uygunluk değeri bulunur ve stokastik bir seçim yöntemi ile yeni nesil elde edilir.

Evrim Stratejisi (ES), 1963 yılında I. Rechenberg ve H. Schwefel'in rüzgar tünelinde uygun kanat şeklini bulmak için yaptıkları deneyler sonucunda geliştirilmiştir [4]. Genel yapı olarak evrimsel programlamaya benzer fakat evrimsel programlamadan ayrı olarak geliştirilmiştir. Evrim stratejileri, evrimsel programlamadan farklı olarak eşeyli çözüm arayan algoritmadır. Evrim stratejisi, evrimsel programlama gibi bir problemin çözümü için problemin kodlanmış şekli yerine problemin kendisi üzerinde arama yapar. Evrim stratejileri ile evrimsel programlama arasındaki en önemli farklar şunlardır:

- Evrimsel programlama stokastik seçim yöntemlerini kullanırken, evrim stratejileri deterministik seçim yöntemlerini kullanır.

- Evrimsel programlamada tekrar üretim operatörü yoktur. Evrim stratejilerinde tekrar üretim operatörü vardır.

- Evrimsel programlama daha büyük popülasyon değerleri için geliştirilmiştir, evrim stratejilerinde popülasyon büyüklüğü çok küçüktür.

Genetik Programlama (GP) kavramı J. Koza tarafından tanımlanmıştır. [8,9]. Genetik programlamayı kısaca genetik algoritma operatörlerini kullanarak bilgisayar programı oluşturma yöntemi olarak tanımlanabilir. Genetik programlama, genetik algoritmanın bütün operatörlerini kullanır. Genetik programlamanın temel özellikleri şunlardır:

- Genetik programlamanın temel amacı, istenilen amaca uygun bilgisayar programını genetik operatörlerle oluşturmaktır.

- Genetik programlama, ağaç yapılı bir kodlama yöntemi kullanır.

-Genetik programlama, genetik algoritmanın bütün operatörlerini kullanır.

Genetik Algoritmalar (GA) adından da anlaşılabileceği gibi doğadaki evrim mekanizmasını örnek alarak oluşturulmuş çözüm tekniğidir. Genetik algoritmalar, ilk defa Charles Darwin tarafından ortaya atılan, "doğal sistemde güçlülerin hayatta kalması" diye özetlenebilecek doğal seçim ve doğal genetik mekanizmasına dayanır. Amacı, nesiller boyunca edinilmiş iyi özelliklerin taşınmasını sağlamak ve ilerleyen nesillerde daha da iyi bireylerin oluşmasını sağlamaktır. Genetik algoritma bu işlevi sağlamak için üreme, çaprazlama ve mutasyon operatörlerini kullanır.

Klasik mantığın dayandığı temel varsayım "her önerme doğru veya yanlıştır". Bu Aristo'dan beri tartışma konusu olmuştur. Aristo "Temel Varsayım" adlı çalışmasında gelecek şartlara bağlı olarak olayların şüpheli doğruluk durumlarından bahseder. Bahsettiği gelecek olaylar hakkındaki önermeler aslında ne doğru ne de yanlıştır. Fakat iki durumun da olması imkan dahilindedir. Eski Yunan düşünürün ortaya attığı aşağıdaki soruya klasik hatta çok değerli mantıkla yanıt bulmak olanaksızdır.

" … bir tek buğday yığın oluşturmaz; iki tane de, üç tane de… Diğer yandan 100 milyon buğdayın yığın oluşturacağını herkes kabul eder. Öyleyse, kesin sınır nedir? 325647 buğdayın bir yığın oluşturmazken 325648 buğdayın bir yığın oluşturacağını söyleyebilir miyiz? "

Yukarıda sözü edilen ve günlük hayatta sıkça karşılaşılan 'kesin olmama' ya da 'tam olarak tanımlanamama problemi', 1965 yılında California Üniversitesinden Lotfi A. Zadeh tarafından bir makaleyle ortaya atılan bulanık küme teorisinin uzantısı olarak gelişen bulanık mantık ile daha etkin bir şekilde çözülebilmiştir. Bulanık mantığın amacı, insanların tam ve kesin olmayan bilgiler ışığında, tutarlı ve doğru kararlar vermelerini sağlayan düşünme ve karar verme mekanizmalarının modellenebilmesidir.

Bulanık mantık kavramının ortaya atılmasıyla beraber araştırmacılar arasında bir tartışma ortamı doğdu. Bazı araştırmacılar bulanık mantığın temel bilimsel prensiplere aykırı olduğunu göstermeye çalıştılar. Özellikle olasılık ve istatistikle

uğraşan matematikçiler, olasılık teorisiyle belirsizlik problemlerinin çözülebileceğini savundular [10].

Zadeh, 1965'te bulanık küme kavramını ortaya attıktan sonra 1968'te bulanık algoritmalar, 1970'te ise bulanık karar verme yöntemlerini açıklayan bir makale yayımlamıştır. 1973'te Zadeh "Kompleks sistemlerin ve karar verme işlemlerinin analizine yeni bir yaklaşım" adıyla bulanık kontrolün temelini atan bir makale yayımladı. Zadeh bu makalesinde dilsel değişkenleri ve insan bilgisini formüle etmek için "EĞER-O HALDE" kurallarını kullanmayı önerdi [10].

1970'lerde gerçek sistemler için bulanık kontrolörler kullanılmaya başlandı. Bulanık mantık kullanılarak yapılan ilk çalışma 1975 yılında Mamdani ve Assilian tarafından bir buhar makinesinin denetimi için yapıldı. Bu çalışmayla bulanık kontrolör yapımının kolay olduğunu ve iyi sonuçlar verdiğini gösterdiler.

1978'te, ilk kez olarak bir çimento fırını için bulanık kontrolör uygulaması yapılmıştır. 1980'de Sugeno, Japonların ilk bulanık mantık uygulaması olan Fuji Electric su arıtma tesisinin kontrolünü gerçekleştirdi. 1983'te komutların söylenmesiyle kendi kendine park eden bir araba olan bulanık mantık kontrollü bir robot çalışmasına başlandı. 1980'lerin başında Hitachi'den Yasunobu ve Miyamoto, Sendai metrosu için bulanık kontrol sistemi geliştirmeye başladı. 1987'de bu proje bitirildi ve dünyadaki en gelişmiş metro yapıldı. Metroda hızlanma, yavaşlama ve durma bulanık sistemlerle kontrol edilmekteydi. 1987'de Tokyo'da ikinci uluslararası bulanık sistemler konferansı yapıldı. Bu konferansta katılımcılara Sendai metrosu ve ayrıca iki boyutlu masa tenisi oynayabilen bir bulanık robot kolu da tanıtıldı [10]. Sendai metrosunun uygulanmasından sonra bulanık mantığa olan ilgi artmış ve bu alanda yapılan çalışmaların koordinasyonu amacıyla Japonya'da 1989 yılında LIFE (Laboratory for International Fuzzy Engineering) adlı bir laboratuar aralarında SGS-Thomson, Omron, Hitachi, NCR, IBM, Toshiba ve Matsushita gibi büyük firmaların da aralarında bulunduğu 51 firma tarafından kurulmuştur.

Şubat 1992'de San Diego'da birinci IEEE bulanık sistemler konferansı yapıldı. Bu olay bulanık mantık teorisinin en büyük mühendislik kuruluşu olan IEEE tarafından kabul edilmesi anlamına geliyordu. Bulanık mantığın gelişmiş kontrol sistemlerinde başarı ile kullanılmasının nedeni, bu sistemlerin insanların kesin olmayan bilgiler ışığında doğru karar vermelerini sağlayan karar verme mekanizmalarını

modellemesidir. Böylece bulanık kontrol, sistemin kesin matematik modeline ihtiyaç duyan klasik kontrolün yerini alır. Klasik modelde gerçek sistem davranışları kesin matematiksel ifadelerle modellenmeye çalışılırken, bulanık kontrolde insan bilgisi, deneyimi ve mantığı ön plandadır [11].

Literatürde, bulanık kontrolör parametrelerinin genetik algoritmalar ile ayarlanmasına yönelik birçok çalışma vardır. Bu yöndeki çalışmaların genel özeti D. A. Linkens ve H. O. Nyongesa'nın makalelerinde bulunabilir [12,13]. İlk makalede çevrimdışı (offline) çalışan bir sistemde bulanık kontrolörün genetik algoritma ile ayarlanması açıklanırken, ikinci makalede çevrimiçi (online) çalışan bir sistem için bulanık kontrolörün genetik algoritma ile ayarlanmasında kullanılan yöntemler açıklanmıştır.

Herrera yaptığı çalışmada [14] üyelik fonksiyonları önceden belirlenmiş bir bulanık kontrolörün kural tabanının oluşturulmasında genetik algoritmaları kullanmıştır. Bu çalışmada parametreler gerçek değerleriyle kullanılmıştır.

Lee yaptığı çalışmada [15] bulanık kontrolör için üyelik fonksiyonlarının sayısının belirlenmesinde, bulanık kuralların ve kural tabanının oluşturulmasında genetik algoritmaları kullanmıştır.

Belarbi yaptığı çalışmada [16], kromozomları ikili tabanda kodlanmış olarak kullanmış ve bulanık mantık kontrolörü tasarımında yapay sinir ağlarını kullanmış, genetik algoritmaları ağırlık değerlerinin eğitilmesinde kullanmıştır.

Park yaptığı çalışmada [17], üyelik fonksiyonlarının sayısının, tepe noktalarının belirlenmesinde ve kural tabanının oluşturulmasında bazı parametreler kullanmış, bu sayede esnek bir tasarım elde edilmiştir. Buna benzer bir çalışma Cheong [18] tarafından da yapılmıştır.

A. Homaifar ve E. McCormick ise mikroGA yapısı kullanarak bulanık mantık kontrolörünün üyelik fonksiyonlarının ve kural tabanının tasarlanması üzerinde durmuştur. Çalışmada, bulanık kurallar ve üyelik fonksiyonları tek bir kromozomda kodlanmıştır. Geliştirilen yöntem bir araba park kontrol probleminde denenmiş ve oldukça başarılı sonuçlar verdiği görülmüştür [19].

Y. Shi ve R. Eberhart ise kullandığı kromozom yapısı içerisinde bulanık üyelik fonksiyonu ve kural yapısına ilave olarak üyelik fonksiyonlarının türlerini de katmış ve farklı bir kromozom yapısı oluşturmuştur [20].

Bu konuda yapılan diğer çalışmalar ise, B. Huveve, K.I. Mann tarafından gerçekleştirilen Genetik-PID kontrolör ile K.C. Ng ve Y. Li tarafından gerçekleştirilen, genetik algoritma tarafından ayarlanan bulanık kayan kipli Kontrolör uygulaması sayılabilir [21].

Genetik algoritma ile bulanık kontrolör tasarım yönteminin proses kontrolünde kullanılması ilk olarak 1992 yılında C. L. Karr tarafından gerçekleştirilmiştir [22]. Çalışmada 'Basit Genetik Algoritma' yapısındaki bir genetik algoritma ile ikili tabanda kodlama yöntemi kullanılarak yamuk şeklindeki bulanık üyelik fonksiyonları, seçim yöntemi olarak ise turnuva seçim yöntemi kullanılmıştır.

K. Tang ve K. Mann ise hiyerarşik genetik algoritma (HGA) yapısına dayanan bir genetik algoritma ile bulanık kontrolör üyelik değerlerini ve kurallarını optimize etmeye çalışmıştır. Yapılan bu çalışmada, proses olarak, bir tanktaki sıvı seviyesinin kontrolü ele alınmıştır. Burada bir genetik algoritma ile üçgen şeklindeki bulanık üyelik fonksiyonlarının sınır değerleri ayarlanırken diğer bir genetik algoritma ile de bulanık kuralların minimum sayısı bulunmaya çalışılmıştır. Burada genetik kodlama için ikili ve gerçek değerli kodlama yöntemleri kullanılmış ve minimum kural sayısı ile sıvı seviyesi başarılı bir şekilde kontrol edilmiştir [23].

2. GENETİK ALGORİTMALAR

2.1 Genetik Algoritmalar Nedir?

Genetik algoritmalar doğal biyolojik evrim mekanizmasını örnek alarak oluşturulan stokastik bir arama metodudur. Genetik algoritmalar, ilk defa Charles Darwin tarafından ortaya atılan, "doğal sistemde güçlülerin hayatta kalması" diye özetlenebilecek doğal seçme ve doğal genetik mekanizmasına (genetiğin evrim tezine dayandırılan) dayanır.

Genetik algoritmalar, problemin olası çözümlerinin oluşturduğu bir popülasyon üzerinde işlem yapar ve çözüm için en uygun bireyleri bir sonraki nesle taşıyarak bu bireylerin daha ileriki nesillerde çözüme daha da yakın bireyleri oluşturacağı ilkesini esas alır. Genetik algoritmalarda olasılığın rolü olmasına karşın gelişigüzel bir arama metodu değildir, etkinliği tabiattaki canlıların evriminden de görülebilir.

Genetik algoritmalar konusunda ilk çalışmayı Michigan Üniversitesi'nden psikoloji ve bilgisayar bilimi uzmanı John Holland yapmıştır. 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 yapılardan oluşan bir topluluğun çoğalma, çiftleşme, mutasyon vb. genetik süreçlerden geçerek başarılı yeni bireyler oluşturabildiğini gördü. Çalışmalarının sonucunu açıkladığı "Adaptation in Natural and Artificial Systems" adlı çalışmasını 1975'te yayımlamasından sonra, geliştirdiği yöntemin adı Genetik Algoritmalar (GA) olarak yerleşti. Ancak Holland'ın öğrencisi David E. Goldberg adlı inşaat mühendisi 1989'da konusunda bir klasik sayılan "Genetic algorithms in search, optimization, and machine learning " adlı kitabını yayımlayana dek genetik algoritmaların kullanımı yaygın değildi. Goldberg'in gaz boru hatlarının denetimi üzerine 1983 yılında yaptığı doktora tezi ona 1985 "National Science Foundation" Genç Araştırmacı ödülünü kazandırdı ve genetik algoritmaların pratik kullanımının da olabilirliğini kanıtladı [2].

Genetik algoritmalar, doğal seçme ve doğal genetik kurallara dayanan bir arama türüdür. Doğal seçme, doğa koşullarına en fazla uyum sağlamış olan canlının neslini devam ettirmesi, uyum sağlayamamış olan türlerin ise elenmesidir. Canlılar nesilden nesile genlerini aktarırken, bu genler de doğal genetik kurallara göre başka genlerle çaprazlanır, değişime uğrar ve yeni genleri oluştururlar. Genetik algoritmalar da tabaattaki bu iki oluşumu birleştirerek en iyiyi (optimal noktayı) arar. Bir önceki neslin en uyumlu fertleri kullanılarak yeni neslin üyeleri oluşturulur. Bu yöntem sayesinde, klasik yöntemlerle çözülmesi çok zor kimi zaman da imkansız olan problemler çözülebilmektedir.

2.2 Geleneksel Arama Yöntemleri

Geleneksel arama yöntemleri üç ana başlık altında toplanabilir. Bunlar:

1) Hesaba dayalı yöntemler

2) Nümerik yöntemler

3) Olasılığa dayalı yöntemlerdir

Hesaba dayalı yöntemlerden ilki amaç fonksiyonunun gradyanının sıfıra eşit alınması sonucu çıkan nonlineer denklem takımının çözülmesi esasına dayanır. Arama her yöndeki türevi sıfır olan tepe veya çukur olması muhtemel noktalar civarında yapılır. Diğer bir yöntem ise tepe tırmanma (hill climbing) denilen, maksimum nokta hakkında bir tahminde bulunduktan sonra buna yakın herhangi bir noktanın gradyanı veya tersi yönde hareket edilerek arama yapılan yöntemdir. Yerel en iyi noktayı bulmak için mümkün olan en dik eğimli yönde yürünür. Ancak çok sayıda dönme noktası içeren bir fonksiyonda çok sayıda tepe oluşur. Hangi tepenin en iyi çözüm olduğu bilinemez.

Hesaba dayalı yöntemlerin en önemli eksikliği aramaya başlanacak noktanın çözüm için çok önemli olmasıdır. Birden çok tepe noktasına sahip arama uzaylarında çözüm lokal tepe noktasında kalabilir. Diğer bir eksiklik ise amaç fonksiyonun türevlerinin var olmasının gerekliliğidir. Teorik olarak hesabı kolay kuadratik amaç fonksiyonları bulunabilmesine karşın gerçek hayatta karşılaşılanlar teoriden uzaktır. Parça parça süreksizlikler içerebileceği gibi, gürültülü ortamlardan elde edildikleri için pek çok tepe noktalı olabilirler. Genel olarak hesaba dayalı yöntemler ancak sınırlı sayıda uygulamalar için kullanışlıdır.

Nümerik yöntemlerde ise amaç, sürekli olan gerçel sayı aralıklarını belirli sayıda parçalara bölerek parçaların amaç fonksiyon değerlerine bakıp yeni parçaların denenmesidir. Bu tip arama yöntemlerinin eksikliği ise arama uzayının geniş olması ve noktaların birer birer ele alınmasının zorunluluğudur.

Rastgele arama (Random Search) yöntemleri, diğer arama yöntemlerinin bu zayıf yönlerinden dolayı artan bir ilgiyle incelenmektedir. Bununla beraber, rastgele bir yönde tepe tırmanma veya rastgele noktalar seçerek bunların amaç fonksiyonlarını bulma gibi düşünceler rastgele arama yöntemlerinin dışında tutulmalıdır, çünkü etkin değildirler. Genetik algoritmalar, bu gibi rastgele arama yöntemlerinden farklı olarak, olasılığı, kodlanmış parametre uzayında işlem yapmada bir araç olarak kullanırlar [25].

2.2.1 Genetik Algoritmaların (GA) Geleneksel Arama Yöntemleriyle Karşılaştırılması

1) GA çözümü ararken, tek bir nokta değil, bir çözüm grubu kullanır.

2) GA parametrelerin kendisiyle değil kodlanmış haliyle çalışır (parametrelerin GA da gerçek değerleriyle kullanımları hariç).

3) GA deterministik değil, olasılığa dayalı kurallar kullanır.

4) GA çözümü ararken türev veya diğer yardımcı herhangi bir bilgiye ihtiyaç duymaz. Amaç fonksiyonunun seçimi ve buna bağlı uygunluk değerleri (fitness) aramanın seyrini belirler [24].

2.3 Genetik Algoritmaların Kullanılma Nedenleri

1) GA, doğadaki evrim mekanizmasını örnek alan bir arama metodudur ve bir veri grubundan özel bir veriyi bulmak için kullanılır [27].

2) GA, özellikle araştırmacının kesin bilgiye sahip olmadığı ve konunun uzmanı olmadığı durumlarda çok yardımcıdır, çünkü GA önbilgi olarak sadece amaç fonksiyonuna ihtiyaç duyar.

3) GA, değerlendirme için yeni ve daha iyi sonuçlar üretmenin yanı sıra varolan potansiyel sonuçları değerlendirmek için de tasarlandığından dolayı başka alternatifler için büyük yardım sağlar.

4) GA, klasik yöntemlerin çok uzun zamanda yapacakları işlemleri kısa bir zamanda çok net olmasa da yeterli bir doğrulukta yapabilir.

5) GA, doğadaki evrim yöntemlerini kullanan bir arama türüdür. Bu yöntem sayesinde, klasik yöntemlerle çözülmesi çok zor, kimi zaman da imkansız problemler çözülebilmektedir [2].

6) GA, çeşitli mühendislik problemlerini çözmek için kullanılmış ve optimizasyon problemleri için oldukça etkili olmuştur.

7) GA, gerçek optimal değeri bulmayı garanti etmez fakat uzun nesiller sonunda optimal değere çok yakın çözümler bulunmasını sağlar [27].

9) Çok amaçlı problemleri çözerken GA, hedefler açısından birçok doyurucu çözümler sunar ve karar verecek kişinin en uygun sonucu seçmesine müsaade eder.
10) GA, çok amaçlı problemlerde karar vermenin taslak aşamasında yardımcı olur.

2.4 GA'da Kullanılan Biyolojik Terimler

Yaşayan bütün organizmalarda bilginin esas ünitesi gendir. Genlerin oluşturduğu diziye 'kromozom' adı verilir. GA'da problemin çözümüne, bir başlangıç çözüm uzayının oluşturulmasıyla başlanır. Bu çözüm kümesine 'popülasyon', bu çözümlerden her birine ise 'birey' adı verilir. Kromozom organizmanın bütün genetik bilgisini taşıdığından, her bir kromozom bir bireyi ifade eder. Kromozomu oluşturan dizinin kodlanmış şekline 'genotype', bu dizinin anlamı ise 'phenotype' olarak ifade edilir. Gen ise kromozomun tek bir karaktere etki eden ve onu belirleyen (örneğin göz rengi) parçası olarak tanımlanır. Belirli karakterlerin genleri kromozomun belirli bölgelerinde yerleştirilir. Bu bölgeler 'loci' olarak adlandırılır. Bu bölgelere yerleşen genlerin olası değerlerine o genin 'alleses' i denir. Örnek olarak farklı göz renkleri genin 'alleses' leridir. Göz renginin kromozom içindeki yeri, genin 'loci' sidir.

Örnek olarak x_1 ve x_2 gibi iki genden oluşan bir kromozom aşağıdaki gibi ikilik tabanda kodlanmış olsun.

1011010011 | 010111010100101

x_1 x_2

x_1 10 bit x_2 ise 15 bitle ifade edilmiştir. Bu gösterimde değişkenler kodlanmış halde bulundukları için doğrudan bir şey ifade etmezler. Bunu sağlayabilmek için bu

dizinin phenotype'ı yani tekrar onluk tabandaki değeri hesaplanmalıdır. GA'da arama işlemi değişkenlerin kendileriyle değil kodlanmış biçimleriyle yürütülür (değişkenlerin gerçek değerleriyle kullanıldığı GA hariç). Popülasyon içindeki bireylerin uygunluk değerlerinin hesaplanabilmesi içinse kodlanmış dizilerin içindeki her bir değişkenin kapladığı alan göz önüne alınarak gerçek değerlerinin bulunması gerekir.

Kromozomların her biri probleme bir çözüm sunar. Her çözüme, popülasyon içindeki diğer çözümler ile karşılaştırıldığında ne kadar başarılı bir çözüm olduğunu gösteren bir uygunluk değeri (fitness value) atanır. Bu değer, doğadaki bir organizmanın yaşayabilmek için mücadele ederken ne kadar başarılı olduğunu belirlenmesine karşılık gelir. Bu değer amaç fonksiyonu (objective function) vasıtasıyla hesaplanır. Bir bireyin uygunluk değeri ne kadar büyükse üreme için seçilme şansı o kadar fazla olur. Aynı şekilde uygunluk değeri diğerlerine göre az olan bireylerin seçilme şansları daha azdır. Mevcut kuşağın en iyi bireylerini seçip, yeni bireyler elde etmek için bunları eşleyerek, olası çözümlerden oluşan yeni bir popülasyon yaratılır. Yeni nesil bireyler, önceki neslin iyi bireylerinin sahip olduğu özelliklerin ileriki nesillere dağılmasını sağlar. Bu işlem belirli bir kriter sağlanana kadar veya önceden belirlenen bir nesil sayısı boyunca devam ettirilir [24].

2.5 Basit Genetik Algoritma (Simple Genetic Algorithm - SGA)

başla

t=0;

P(t) ' yi oluştur;

P(t) 'nin performansını değerlendir;

eğer şart sağlanmadıysa devam et;

başla

    t=t+1;

    P(t) 'yi P(t-1) den seç;

    P(t) deki elemanlara çaprazlama ve mutasyon operatörlerini uygula;

    P(t) 'nin performansını değerlendir;

bitir

bitir

Yukarıdaki tabloda P(0) başlangıç popülasyonunu, P(t) ise, t anındaki popülasyonu ifade etmektedir.

Literatürde bu tür bir yapı, Basit Genetik Algoritma (Simple Genetic Algorithm-SGA) olarak adlandırılır. Pek çok genetik algoritma uygulamasında bu model kullanılır. Bu model Goldberg tarafından tanımlanmış ve diğer ardışıl genetik algoritma türleri için referans oluşturmuş bir algoritmadır. SGA'da ikili kodlama biçimi, rulet çemberi (roulette wheel) seçim yöntemi, tek kesim noktalı çaprazlama ve bit mutasyon operatörleri kullanılmıştır. Literatürde ardışıl genetik algoritma tekniği içerisinde, SGA'dan başka mikro genetik algoritma (μGA), kararlı durum genetik algoritma (KDGA), hiyerarşik genetik algoritma (HGA) ve dağınık genetik algoritma (mGA) gibi farklı modeller ve paralel çalışan genetik algoritma modelleri tanımlanmıştır.

Pek çok uygulamada iyi sonuçlar veren basit genetik algoritmada üç operatör bulunur. Bunlar:

a) Üreme

b) Çaprazlama

c) Mutasyon

operatörleridir.

Üreme (seçim) işlemi, popülasyondaki kromozomları amaç fonksiyonu değerlerine bağlı olarak yeni oluşturulacak popülasyon içine taşıma işlemidir. Amaç fonksiyon değeri ise maksimize veya minimize edilecek fonksiyonun o noktadaki değeridir. Bir diziyi amaç fonksiyonuna göre kopyalamak demek, maksimizasyon problemlerinde amaç fonksiyonu değeri yüksek ve yüksek uygunluk değerli bireyleri sonraki nesillere kopyalamak demektir. Minimizasyon problemlerinde ise yüksek uygunluk değerine sahip bireyler, amaç fonksiyon değeri düşük olan bireyler olduğundan seçim işleminde bu bireyler sonraki nesillere daha yüksek bir oranda kopyalanır. Seçim işleminden çıkan bireyler işlem havuzunu oluşturur. İşlem havuzunda bu bireylere çaprazlama ve mutasyon işlemleri uygulanır.

Temel olarak çaprazlama operatörü bireyler arasında genetik bilginin değiş tokuş edilmesini sağlar. En basit çaprazlama operatörü tek kesim noktalı çaprazlamadır (single-point crossover).

Aşağıdaki gibi iki komşu bireyi göz önüne alalım

P_1=**10010110**

P_2=10111000

olsun. Tek kesim noktalı çaprazlamada, bir ile dizilerin uzunluğunun bir eksiği arasında bir tamsayı [1, l-1] rastgele olarak seçilir ve bu noktanın öncesi ve sonrasındaki genetik bilgi oluşan yeni bireylere dağıtılır.

Örneğin kesim noktası i=5 olarak seçilirse, çaprazlamadan sonra

O_1=**10010**000

O_2=10111**110**

bireyleri meydana gelir. Oluşan birinci birey i=5 noktasına kadar olan genetik bilgiyi P_1 den i=5 ten sonraki bilgiyi ise P_2 den alır, aynı şekilde oluşan ikinci birey de kesim noktasına kadar olan bilgiyi ikinci bireyden, kesim noktasından sonraki bilgiyi ise birinci bireyden alır. Bu sayede P_1 ve P_2 bireyleri arasında genetik bilgi alışverişi sağlanmış olur.

Çaprazlama operatörünün popülasyondaki bütün bireylere uygulanması zorunlu değildir. Bu işlem kullanıcı tarafından değeri belirlenen P_c (çaprazlama oluşma olasılığı) oranında gerçekleşir.

Bir diğer genetik operatör ise mutasyondur. Bu operatörün de her bireye uygulanma zorunluluğu yoktur, genelde bu operatör de bireylere P_m (mutasyon oluşma olasılığı) oranında uygulanır. En basit mutasyon işlemi ikilik tabandaki gösterimde bit bazında gerçekleşir. Bu işlem sonunda bireyin herhangi bir biti durum değiştirir(0→1 veya 1→0).

Örneğin O_1 dizisinde dördüncü bit mutasyona uğrayacak olursa

O_1= 100 **1** 0000

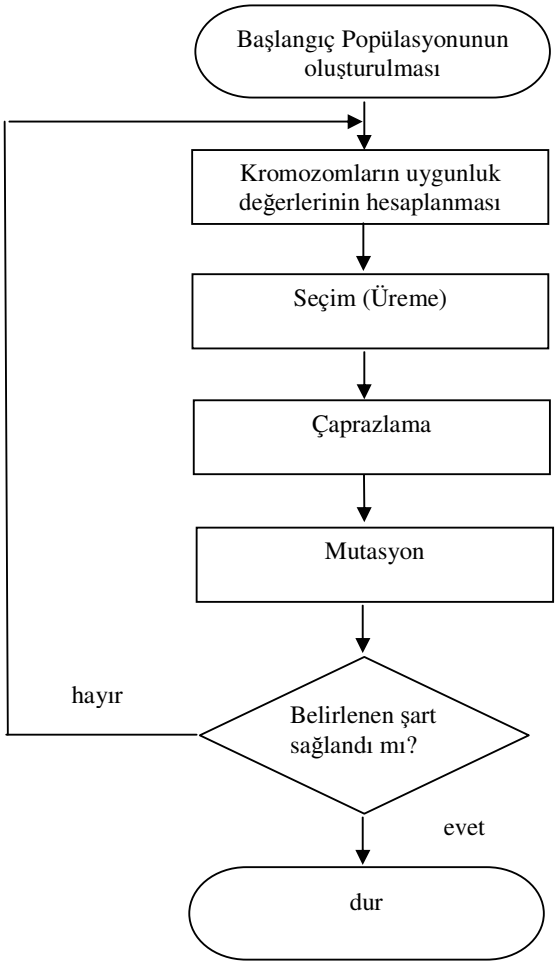
O_{1M}=100 **0** 0000

şeklini alır.

Mutasyon operatörü, çözüm arama uzayının belirli bir alt uzayının aranma olasılığının asla sıfır olmamasını sağlar. Bu sayede çözümün, global optimum yerine lokal optimum bir noktaya takılması engellenmiş olur.

2.6 Genetik Algoritmaların İşleyişi

Genetik Algoritmaların işleyişi Şekil 2.1'deki akış diyagramı ile gösterilebilir.



Şekil 2.1: GA'nın işleyişi

Akış diyagramından da görüldüğü üzere başlangıç popülasyonunun oluşturulmasından sonra genetik algoritma operatörlerinin oluşturduğu döngü belirlenen şart sağlanana kadar devam eder. Belirlenen şart sağlandığında ise genetik algoritma durur.

Döngünün işleyişi sırasındaki geçen aşamalar Tablo 2.1'deki gibi gösterilebilir.

Tablo 2.1: GA'da çözüm aşamaları

Adım1	Olası çözümlerin kodlandığı bir çözüm grubu oluşturulur.Çözüm grubu, biyolojideki benzerliği nedeniyle popülasyon,çözümlerin kodları da kromozom olarak adlandırılır.
Adım2	Her kromozomun uygunluk değeri hesaplanır.
Adım3	Bu kromozomlar eşlenerek (matching), yeniden kopyalama (reproduction),çaprazlama(crossover) ve mutasyon (mutation) işlemleri uygulanır.
Adım4	Yeni kromozomlar eski kromozomların yerini alarak yeni popülasyon elde edilir. (Reinsertion)
Adım5	Tüm kromozomların uygunluk değerleri hesaplanır
Adım6	Çalışmayı sonlandırma şartı sağlanmamış ise üçüncü adıma gidilir.
Adım7	O ana kadar bulunmuş en iyi kromozom, sonuçtur

Adım1: Bir çözüm grubu oluşturulur. Bu grubun oluşturulması tamamen rastgele olabileceği gibi probleme göre özelleşmiş de olabilir. Tamamen rastgele bir çözüm grubu genetik algoritmaya tüm problem uzayını arama şansı verir; probleme göre özelleşme ise işlemi önemli oranda hızlandırabilir. Fakat, özel bir çözüm grubunun oluşturulabilmesi için problem hakkında daha önceden bir önbilgiye sahip olunması gerekir ki bu her zaman mümkün olmayabilir. Çözüm grubunun büyüklüğü de önemli bir faktördür. Büyük çözüm grupları işlem süresini uzatır, küçük çözüm grupları ise yerel maksimum (veya minimum) değerlerine takılmaya neden olabilir. Başarılı bir çözümün, küçük bir grupta daha ilk nesillerde baskın hale gelmesi çok daha kolaydır, bu da istenmeyen bir durumdur. Popülasyon büyüklüğünün seçiminde standart bir yöntem olmamasına karşın yapılacak işlemin karmaşıklığı ve aramanın derinliğine göre 30-100 arasında bir sayı seçilebilir.

Adım2: Kromozomların ne kadar başarılı olduğunu hesaplayan fonksiyona uygunluk fonksiyonu denir. Bu fonksiyon uygunluk değerini, problemin minimizasyon veya

maksimizasyon problemi olmasına göre amaç fonksiyonu değerini kullanarak hesaplar. Amaç fonksiyonu ve uygunluk fonksiyonun seçimi genetik algoritmanın başarısını etkileyen en önemli unsurlardır. Genetik algoritmada probleme özel çalışan tek kısım burasıdır.

Adım3: Eldeki çözüm grubunun içinden başarılı çözümler diğerlerine göre daha yüksek bir oranda diğer nesillere aktarılır. Seçim işleminin temel mantığı, doğadaki gibi başarılı olan bireylere üstel çoğalma imkanı vermekle açıklanabilir.

Bilgisayar ortamında bellek ve işlem zamanı sınırlı olduğu için, popülasyon büyüklüğü GA'da genellikle bütün nesillerde sabit tutulur. Bu yüzden başarılı olan bireyleri çoğaltmak demek başarısız olanları azaltmak anlamına gelir. GA'da kullanılan bir çok seçim yöntemi vardır. Bu yöntemlere yardımcı olarak elitist seçim (ellitist strategy) gibi teknikler de kullanılmaktadır. Elitist seçim, nesil içinde en başarılı olan belirli sayıda bireyin diğer nesle herhangi bir elemeye maruz kalmadan doğrudan aktarılması esasına dayanır.

Çaprazlama, iki ebeveyn kromozom arasında belirlenen parçaların takas edilmesidir. Çaprazlama popülasyon içinde çeşitliliği sağlar. İyi özelliklerin bir araya gelmesini kolaylaştırarak en iyiye yaklaşmayı sağlar. GA'da bireylere genellikle 0.5 ile 1.0 arasında değişen olasılıklarda çaprazlama uygulanır.

Mutasyon, kromozomun bir parçasının dışarıdan değiştirilmesi şeklinde tanımlanır. İkilik tabanda gösterimde değiştirme, rastgele seçilen bir bitin durumunun değiştirilmesiyle sağlanır. Çok düşük mutasyon oranı aramanın yerel minimum veya maksimum noktaya takılmasına neden olurken, yüksek mutasyon oranı da eldeki iyi çözümleri bozar ve aramayı gelişigüzel bir hale getirebilir. GA'da genel olarak 0.01 ile 0.001 arasında değişen sabit bir sayı kullanılır veya her bireyin uzunluğunun (L_{ind}) çarpmaya göre tersi ($1/L_{ind}$) oranında mutasyon uygulanarak ortalama olarak her bireydeki tek bir bit mutasyona uğratılır.

Adım4: Eski kromozomlar oluşan yeni kromozomlarla yer değiştirir. Bu yer değiştirme değişik yöntemlerle yapılabilir. Uygulamalarda eski bireylerin sırayla yeni bireylerle yer değiştirmesi, yer değiştirmeye eski popülasyondaki uygunluk değeri düşük olan bireylerden başlanılması, en fazla nesil boyunca popülasyonda kalmış bireylerden başlanılması gibi teknikler kullanılmaktadır.

Adım5: Oluşan yeni nesildeki bireylerin uygunluk değerleri hesaplanır.

Adım6: Genetik algoritmanın durma şartı, önceden belirlenen bir sayıda nesil oluşturma veya belirlenen bir kriterin sağlanması şeklinde düzenlenebilir.

Adım7: O ana kadar bulunan en iyi kromozom, uygunluk fonksiyonu değeri en yüksek olan bireydir.

2.6.1 Başlangıç Popülasyonunun Oluşturulması

GA'nın uygulanmasında ilk adım, bir başlangıç popülasyonunun oluşturulmasıdır. Kısıtlı optimizasyon (Constrained Optimization) problemleri hariç popülasyon genelde rastgele olarak oluşturulur.

GA'nın uygulanmasında popülasyon genişliğinin (N) belirlenmesi gerekir. Büyük popülasyonlarda çözüm uzayı iyi örneklendiği için aramanın etkinliği artar ancak makul bir sürede yüksek kalitede bir çözüme ulaşılamayabilir. Küçük popülasyonlar ise çözüm uzayını yeterli bir biçimde örnekleyememe riskini taşır ve erken yakınsama (premature convergence) görülebilir. Popülasyon genişliği genelde 30-100 arasında seçilir, yalnız değişik bir GA uygulaması olan mikroGA' da bu sayı 10 civarında seçilir.

Bazı genetik algoritmalar parametrelerin gerçek değerleriyle işlem yürütmesine karşın çoğu GA'da parametreler kodlanmış halde bulunur. Literatürde en çok kullanılan yöntem ise ikilik tabanda kodlamadır. Bu yöntemde kromozomlar 0 ve 1 lerin oluşturduğu bit dizilerinden oluşur. Her parametre için n tane ($n \in Z^+$) bit kullanılır. Bu sayı her değişken için farklı seçilebilir. Bazı uygulamalarda Gray kodlaması standart ikilik tabanda kodlama yerine tercih edilmektedir. Bunun sebebi standart ikilik tabanda kodlamada komşu değerler arasında Hamming mesafesinin sabit olmamasıdır [28,29]. Bu da arama işleminin global minimum noktasına etkin bir şekilde ulaşamamasına neden olabilmektedir [28,29]. İkilik tabanda kodlamaya alternatif olarak tamsayı ve gerçek değerli kodlamalar da kullanılmaktadır. GA'da değişkenleri gerçek değerleriyle kullanmanın avantajı, kromozomun her uygunluk değerinin hesaplanmasından önce gerçek değerine çevrilmesine gerek kalmayacağı için işlem zamanını kısaltması ve parametrelerin kodlanması sırasında parametrelerin oluşabilecek hassasiyet kaybı problemlerini ortadan kaldırmasıdır [5].

2.6.2 Amaç ve uygunluk fonksiyonunun seçimi

Uygunluk değeri, yeni yığına taşınacak dizilerin belirlenmesinde kullanılan bir araçtır. Bu nedenle, algoritmanın her çevriminde, yığındaki dizilerin bir değerlendirme fonksiyonu yardımıyla uygunluk değeri hesaplanır. GA'da kullanılan değerlendirme fonksiyonu problemin amaç fonksiyonudur [30].

Amaç fonksiyonu, bireylerin problemin çözümü konusunda gösterdikleri başarımı ifade eden bir göstergedir. Minimizasyon problemlerinde en uygun bireyler amaç fonksiyonu değeri en küçük olan bireylerdir. Uygunluk fonksiyonu bu değeri kullanarak bireylerin popülasyon içindeki başarımlarını hesaplar.

$$F(x)=g(f(x)) \tag{2.1}$$

f=amaç fonksiyonu

g=amaç fonksiyonu değerlerini negatif olmayan sayılara dönüştüren fonksiyon

F=bağıl uygunluk (relative fitness) değerleri

Oransal uygunluk değeri atanması metodunda her bir bireyin uygunluk değeri $F(x_i)$, bireylerin amaç fonksiyonu değerlerinin ($f(x_i)$), amaç fonksiyonu değerlerinin toplamına bölünmesiyle bulunur.

$$F(x_i) = \frac{f(x_i)}{\sum_{i=1}^{N_{ind}} f(x_i)} \tag{2.2}$$

N_{ind}=Popülasyonun büyüklüğü

Minimizasyon problemlerinde ise bu yöntem, çıkan uygunluk değerlerinin çarpmaya göre tersi alınarak kullanılabilir.

Bu yöntem, bu haliyle amaç fonksiyonu değeri negatif sayılar çıkan bireylerin bulunduğu popülasyonlarda kullanılamaz. Bunun için F(x)=af(x)+b türünde lineer bir dönüşüm uygulanır. a değeri problem minimizasyon problemi ise negatif seçilmelidir, b sayısı ise sonucun pozitif bir sayı olmasını sağlamaya yöneliktir. Bu yöntemin dezavantajı yüksek uygunluk değerine sahip birkaç bireyin popülasyonda baskın hale gelip çözümün lokal minimum noktaya takılma ihtimalinin olmasıdır. Bu sorundan kaçınmak için sıralı (ranking) uygunluk ataması yöntemi kullanılmaktadır. Bu yöntemde her bireyin uygunluk değerine belirli bir sınır getirilmektedir. Bireyler

öncelikle amaç fonksiyon değerlerine göre sıralandıktan sonra sahip oldukları sıra numarasına göre bir uygunluk değeri atanır. Girilen 'max' değeri en uygun bireye atanacak değerdir ve seçici baskı (selective pressure) olarak isimlendirilir [24].

min=2-max (2.3)

Inc=2*(max-1)/Nind (2.4)

Low=Inc/2 (2.5)

min=alt sınır

Inc=komşu iki birey arasındaki uygunluk değeri farkı

Low=uygunluğu en az olan bireyin beklenen seçilme sayısı

max genelde [1.1, 2] arasında seçilir. Örnek olarak 40 bireyden oluşan ve 'max ' değeri 1.1 olan bir popülasyonda 'min' =0.9 'Inc' =0.005 ve 'Low' =0.0025 tir. Popülasyondaki her bir bireyin uygunluk değeri (2.6)'daki gibi hesaplanır.

$$F(x_i) = 2 - MAX + 2(MAX - 1)\frac{x_i - 1}{N_{ind} - 1} \quad (2.6)$$

x_i=i. bireyin sıralı popülasyondaki yeri

2.7 Seçim (Selection)

Kodlama yöntemi belirlenip başlangıç popülasyonu oluşturulduktan sonra, yeni popülasyona geçebilmek için seçimin nasıl yapılacağı belirlenmelidir. Seçim, bireylerin üreme için seçildiği bir işlemdir. Seçim operatörünün yüksek uyuma sahip bireylerin sonraki kuşağa kopyalanma olasılığını artırması istenir. Böylece seçim, arama uzayının başarılı olunacağı umulan bölgelerinde yoğunlaşır. Sonraki kuşak için üremeye katılacak bireylerin popülasyondan nasıl seçileceği ve her bir bireyin kaçar tane yavruya sahip olacağı belirlenmelidir. Seçim işleminden sonra işlem havuzu hazırlanmış olur ve bu bireylere çaprazlama ve mutasyon uygulanır. Bu işlemlerden sonra yeni nesil elde edilir.

2.7.1 Orantılı Seçim Yöntemi (Proportional Selection)

Popülasyondaki her bireyin seçilme olasılıkları belirlenir ve bu olasılıklar kullanılarak seçim işlemi gerçekleşir. Bu grupta bulunan yöntemler rulet çemberi, stokastik artan ve stokastik evrensel seçim yöntemidir.

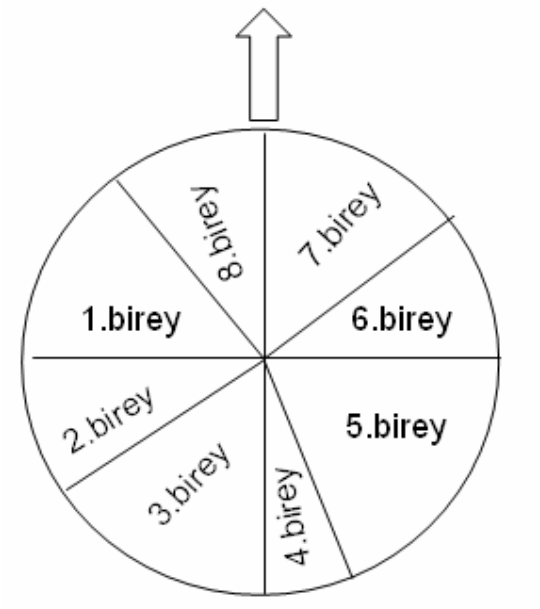
2.7.1.1 Yerine koyarak Stokastik Örnekleme (Roulette Wheel Selection –Rulet Çemberi Seçimi)

Bu seçim yönteminde, bireyin beklenen seçim olasılığını, bireyin uygunluk değerinin bütün bireylerin uygunluk değerlerinin toplamına oranı belirler. Bu yöntemde bazen en iyi bireyin kaybolma ihtimali oluşur. Çünkü rulet çemberi her çevrilişinde rassal bir noktada durur. Bunu önlemek için elitist model kullanılabilir. Yöntem genel olarak rulet oyununa benzetilir ve Şekil 2.2'deki gibi gösterilir. Popülasyondaki bireylerin toplam uygunluk değeri rulet çemberinin büyüklüğünü verir ve her birey uygunluk değeri kadar yer kaplar. Bu yöntemde her bir bireyin beklenen seçilme olasılığı (2.7)'deki gibi verilir.

$$p_i = \frac{f_i}{\sum_{j=1}^{N} f_j} \tag{2.7}$$

p_i = i. bireyin seçilme olasılığı

f_i = i. bireyin uygunluk değeri



Şekil 2.2: Rulet çemberi gösterimi

Bu yöntemde:

-Rulet çemberi popülasyon büyüklüğü kadar çevrilir.

-Her çevirme işleminden sonra ibrenin bulunduğu yerde olan birey işlem havuzuna dahil edilir.

2.7.1.2 Yerine Koymadan Stokastik Örnekleme (Stochastic Sampling Without Replacement)

Yerine koymadan stokastik örnekleme, De Jong'un beklenen değer modelinin diğer adıdır. Bu yöntem, rulet çemberi seçim yönteminin hatalarını azaltmaktadır.

f_k , k. bireyin uyum değerini $\sum_{k=1}^{N} f_k$, popülasyonun toplam uyumunu ve $\bar{f} = \frac{\sum_{k=1}^{N} f_k}{N}$, popülasyonun ortalama uyum değerini göstermek üzere her birey için $\frac{f_k}{\bar{f}}$ beklenen yavru sayısı hesaplanır. Bu sayı, birey çaprazlamaya her seçildiğinde 0.5, çaprazlama olmadan üremeye seçildiğinde ise 1 azaltılır [30].

2.7.1.3 Kalanı Stokastik Örnekleme (Stochastic Remainder Selection- SRS)

Kalanı stokastik örnekleme yöntemi, yerine koyarak veya koymadan uygulanabilir. Yöntem bireylerin beklenen üreme sayılarını hesaplar ve tamsayı kısmı kadar her bireyin deterministik olarak çoğalması sağlanır. Popülasyon büyüklüğüne ulaşılmadıysa, kesirli kısımlar kullanılarak ağırlıklı para atışları yapılır. Örnek olarak beklenen kopya sayısı 2.25 olan bir birey iki kopyayı kesin olarak, diğer kopyayı ise 0.25 olasılıkla elde eder.

2.7.1.4 Stokastik Evrensel Örnekleme (Stochastic Universal Sampling-SUS)

James Baker, bireye atanan gerçek yavru sayısı ile beklenen yavru sayısı arasındaki farkı minimize eden bir yöntem olarak bu yöntemi önermiştir. Bu yöntemde temel düşünce, rulet çemberini bir kez çevirerek seçilen tüm bireyleri belirlemektir. Bunu gerçekleştirmek için tek bir rulet işaretçisi yerine seçim çemberi denilen bir parça kullanılır. Bu parça eşit aralıklı N (popülasyon büyüklüğü) tane göstergeye sahiptir. Çember bir kez çevrilir ve durduğunda göstergelerin bulunduğu yerler seçilen bireyleri gösterir. Böylece N tane birey, tek adımda seçilmiş olur. Göstergeler eşit

aralıklı olduğu için, bir bireyin küçük popülasyonlarda bile tüm popülasyona hakim olma tehlikesi yoktur. Bu algoritma, seçim planının belirli bir işleyişinin sonucu, beklenen davranışa mümkün olduğunca yakındır yani ortalama değişim minimumdur [30].

2.7.2 Turnuva Seçim Yöntemi (Tournament Selection)

Popülasyondan rassal olarak q birey yerine koyarak veya koymadan seçilir ve grup içindeki en iyi uygunluk değerine sahip birey ara popülasyona kopyalanır. q değeri turnuva büyüklüğünü ifade eder. Bu işleme popülasyon büyüklüğüne (N) ulaşılıncaya dek devam edilir. Bu seçim yönteminde turnuva büyüklüğü önemlidir ve seçim yönteminin performansını önemli ölçüde etkiler. Çoğu uygulamalarda turnuva büyüklüğü 2 olarak seçilir. Turnuva seçim yöntemi popülasyon büyüklüğü küçük olan uygulamalarda orantılı seçim yöntemlerinden daha iyi sonuçlar verir. Bu seçim yönteminde bireyin seçilme olasılığı (2.8)'deki gibi verilir [31].

$$p_i = \frac{1}{N^q}((N - i + 1)^q - (N - i)^q) \tag{2.8}$$

N=popülasyon büyüklüğü

p_i =i. bireyin seçilme olasılığı

q= turnuva büyüklüğü

2.7.3 Kararlı Durum Seçim Yöntemi (Steady State Selection)

GA'nın her iterasyonunda mevcut popülasyondan bütünüyle yeni bir popülasyon oluşturulmaktadır. Bütün popülasyonu değiştiren GA'ya nesilsel (generational) GA denir. Bazı seçim planlarında (elitist model gibi), birbirini izleyen nesiller bir dereceye kadar örtüşmekte, yani önceki neslin bir bölümü yeni popülasyonda korunmaktadır. Her nesilde popülasyonun değiştirilen oranına nesil boşluğu (generation gap) denir ve G ile gösterilir. Böylece t. neslin N(1-G) bireyi (t+1). nesilde de aynen yaşamaya devam eder. Nesilsel genetik algoritmalarda G=1 dir.

Kararlı Durum (Steady State) seçimde, her kuşakta sadece birkaç birey değiştirilir. Bu seçimde, ebeveyn olarak iki bireyin nasıl seçileceğine ve gelecek yavrulara yer açmak için hangi iki bireyin silineceğine karar verilir. Örneğin, Whitler'in GENITOR

algoritması ebeveynleri sıralanan uyum skorlarına göre seçmekte ve yavrular en kötü iki bireyin yerini almaktadır.

2.8 Çaprazlama (Crossover- Recombination)

GA'da yeni kromozomlar oluşturmak için kullanılan temel operatör çaprazlamadır. Çaprazlama, farklı çözümler arasında bilgi değişimini sağlayarak, arama uzayının benzer ancak araştırılmamış bölgelerine ulaşılmasını sağlayan bir arama operatörüdür. Çaprazlamanın bireylerdeki iyi özellikleri birleştirerek daha iyi çözümler üretmesi beklenir. Çaprazlama bireyler arasında P_c olasılığıyla uygulanır.

GA'da en çok kullanılan çaprazlama operatörleri şunlardır:

a) Tek kesim noktalı çaprazlama (Singe-point crossover)

b) Çok kesim noktalı çaprazlama (Multi-point crossover)

c) Düzgün çaprazlama (Uniform crossover)

2.8.1 Tek kesim noktalı çaprazlama

GA'da kullanılan en yaygın çaprazlama türüdür. Öncelikle popülasyon rastgele olarak veya komşu ebeveynler arasında ikişerli olarak N/2 kümeye ayrılır. Popülasyon büyüklüğü (N) tek sayı ise, bir eleman bu gruplamanın dışında kalır. P_c çaprazlama olasılığı oranında çaprazlama uygulanacak çözüm çiftleri belirlenir. Algoritma uygulanırken [0, 1] aralığında rastgele bir sayı seçilir ve bu seçilen sayı çaprazlama olasılığından küçük ise çaprazlama uygulanır.

Örnek:

P_1=**00100 11010**

P_2=11100 10001

seçilir ve kesim noktası (n) 5 olarak seçilirse yeni bireyler

O_1=**00100** 10001

O_2=11100 **11010**

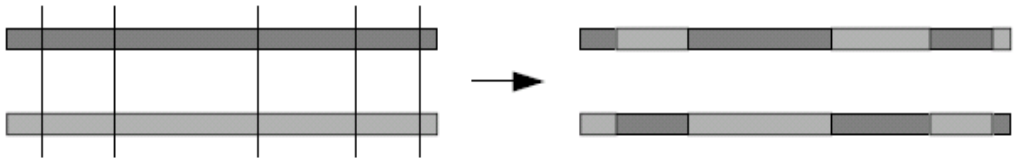
olarak oluşur.

2.8.2 Çok kesim noktalı çaprazlama

Çok kesim noktalı çaprazlamada [1, l-1] arasında rastgele olarak m tane kesim noktası belirlenir (m≥2).

Bu yöntemde kesim noktaları arasındaki bilgi bireyler arasında değiş tokuş edilir.

Bu yönteme örnek m=5 seçilerek Şekil 2.3'de gösterilmiştir.



Şekil 2.3: Çok kesim noktalı çaprazlama

Çok kesim noktalı ve diğer bir çok çaprazlama operatöründeki temel düşünce, bir problemin çözümüne olumlu etki yapan bilginin, bir bireyi ifade eden dizinin içindeki birbirini takip eden alt-dizilerin içinde sürekli bir biçimde olmayabileceğidir. Bunun için çaprazlama işlemi uygulanarak farklı bireyler arasında alt-dizilerin değişimi sağlanarak ilerleme sağlanmasına çalışılır. Çok kesim noktalı çaprazlama arama uzayının genişlemesine yol açarak, uygunluk değeri yüksek olan bireylerin erkenden baskın hale gelmesini engeller ve daha gürbüz bir arama yapılmasını sağlar [24].

2.8.3 Düzgün Çaprazlama (Uniform Crossover)

Düzgün çaprazlamada, kromozom üzerindeki her bir nokta olası bir çaprazlama noktasıdır. Bu yöntemde kromozom uzunluğuyla aynı uzunlukta bir çaprazlama maskesi (crossover mask) oluşturularak ve bu maskedeki bitlerin 0 veya 1 olmasına göre ebeveyn kromozomlar oluşacak yeni kromozomlara bilgiyi aktarırlar.

Örnek olarak:

P_1=**1011000111**

P_2=0001111000

M=0011001100 (çaprazlama maskesi)

olursa

O_1=00**11**11**01**00

O_2=**10**01**00**10**11**

bireyleri oluşur.

Örnekten de görüldüğü üzere, birinci yavru birey maske bitlerinin 1 olduğu kısımlarda bilgiyi birinci ebeveyn bireyden, 0 olduğu kısımlarda ise bilgiyi ikinci ebeveyn bireyden almaktadır. İkinci yavru birey ise maske bitlerinin 0 olduğu kısımlarda bilgiyi birinci ebeveyn bireyden, 1 olduğu kısımlarda ise bilgiyi ikinci ebeveyn bireyden almaktadır.

Düzgün çaprazlamada kesim noktası pozisyona bağlı olmadığından her bit çaprazlama operatörüyle değiştirilip etkinliği araştırılabilir fakat bu yöntem başarılı olan bireylerin fazla bir şekilde bozulmasına yol açarak aramanın etkinliğini azaltabilir.

Deneysel sonuçlar, düzgün çaprazlamanın küçük popülasyonlar için, tek kesim noktalı veya iki kesim noktalı çaprazlamanın ise büyük popülasyonlar için daha iyi sonuçlar verdiğini göstermektedir. Düzgün çaprazlamanın bozuculuğu küçük popülasyonlarda oldukça kesitsel bir arama yapmaya yardımcı olmaktadır. Büyük popülasyonlarda var olan çeşitlilik kesit gereksinimi azaltmakta ve tek veya iki kesim noktalı çaprazlamayı uygun kılmaktadır [30].

2.9 Mutasyon

Mutasyon doğada olduğu gibi GA uygulamalarında da geri planda kalan bir operatördür. GA'nın çalışmasında ikinci dereceden rol oynar. GA'da mutasyon operatörü küçük bir olasılıkla bir kromozom içindeki bir veya birkaç değeri rastgele olarak değiştirerek popülasyonda yeni kromozomların (arama uzayında yeni çözüm noktalarının) elde edilmesini sağlar. İkili tabanda mutasyon operatörü rastgele seçilen değeri 1 olan bitin 0 olmasını, değeri 0 olan bitin ise 1 olmasına neden olur. Mutasyon olasılığı genelde 0.01 ve 0.001 arasında seçilir. Muhleinbein ve Bach yaptıkları çalışmalar sonucunda mutasyon olasılığının (P_m) ideal değerinin, kromozomların uzunluğu ve problemin çözüm uzayının büyüklüğü ile orantılı olduğunu ileri sürmektedir.

İkilik tabanda gösterimde mutasyona uğrayacak nokta n=3 seçilirse

P= 00 **0** 1100010

P_M= 00 **1** 1100010

şeklinde olur.

Bazı çalışmalarda mutasyon her bireye eşit olarak değil uygunluk değeri daha düşük olan bireylere daha yüksek oranda uygulanmaktadır [24].

Bazı çalışmalarda ise dinamik mutasyon ve çaprazlama uygulanmaktadır. Bu çalışmalarda nesiller ilerledikçe mutasyon olasılığı artırılırken çaprazlama olasılığı azaltılmaktadır. Dinamik çaprazlama ve mutasyon olasılıkları (2.9) ve (2.10)'daki gibi uygulanabilir [32].

$$P_c = e^{(-Tg/M)} \quad (2.9)$$

$$P_m = e^{(0.005*Tg/M)} \quad (2.10)$$

Tg=O anda bulunulan nesil numarası

M=maksimum nesil sayısı

Özellikle GA'nın son çevrimlerinde mutasyon etkinliği artmaktadır. Çünkü son çevrimlerde popülasyon iyi çözümlere yakınsadığından kromozomlar birbirine çok benzemektedir.Bu durum, çaprazlama operatörünün farklı yeni bireyleri oluşturmasını engelleyerek aramanın etkinliğini kısıtlar. Bu aşamada mutasyon, yığındaki değişkenliği gerçekleştirerek arama uzayında yeni çözüm noktalarının elde edilmesini sağlamaktadır.

3. BULANIK MANTIK

Dünyadaki bazı olayları açıklamak için kesin tanımlamalarda bulunabilmek imkansızdır ve olaylar çoğu kere belirsizlikler ve doğrusal olmama özellikleri taşır. Terimler ya da ölçüler kesin olarak tanımlanıp ölçülemediğinden dolayı insanlar çoğu zaman belirsiz (kesin olmayan) ifadeler kullanırlar. İşte bulanık mantık bazı sorulara basitçe evet-hayır cevabı verilemeyen durumları kapsar. Bulanıklığın ve bulanık mantığın temeli de budur. Bulanık mantığın gücü basit şeyleri basit tutmaktır. Klasik mantık bizleri çok katı sınırlar çizmeye zorlar. Mesela batı edebiyatında "novel" denilen roman 90 veya daha fazla sayfadan , "novella" ise 90'dan daha az sayfadan oluşur. Bu standarda göre 91 sayfalık bir eser roman olurken, 89 sayfalık bir çalışma "novella" (uzun hikaye) olur. Eğer bir bilgisayarda kelimelerin puntosu büyütülürse uzun hikaye, roman haline gelebilir. Bulanık mantık bu tip mantıksızlıkları önler.

3.1 Bulanık kümeler

Matematikte küme, ayırt edilebilen belirli özellikleri olan nesnelerin bütünüyle kavranmış topluluğu olarak tarif edilir. Bir kümeden bahsedildiği zaman onun bütün olarak kavranmış veya anlaşılmış olması gerekir. Bir başka ifadeyle, küme kendisine neyin ait olup olmadığı konusunda herhangi bir şüphenin bulunmadığı topluluktur. İşte bahsedilen bu topluluğa veya nesnelere bu kümenin elemanları denir.

Klasik küme teorisinde herhangi bir A kümesinde üyelerin kümeye ait olup olmaması aşağıdaki gibi bir karakteristik fonksiyonla ifade edilir.

$$\mu_A(x) = \begin{cases} 1 \; x \in A \\ 0 \; x \notin A \end{cases} \qquad \forall x \in X$$

X=Evrensel küme

Bulanık küme teorisinde ise elemanların bir kümeye aitliği sadece 0 ve 1 ile değil [0,1] aralığındaki herhangi bir sayıyla gösterilebilir.

Tanım: Bir bulanık küme [0,1] arasında değerler alan üyelik fonksiyonu $\mu_A(x)$ tarafından tanımlanır. Bir klasik kümenin üyelik fonksiyonu sadece iki değer yani 0 ve 1 değerlerini alabilirken bir bulanık kümenin üyelik fonksiyonu 0 ile 1 arasında herhangi bir sayı değeri alabilir.

Üyelik fonksiyonu ile her reel sayıya karşı düşen bir üyelik derecesi (degree of membership) bulunabilir.

- Üyelik derecesi, A bulanık kümesinin belirttiği kategoriye bir x elemanının aitlik mertebesini belirtir.

- Aitlik derecesinin olasılık ile hiçbir ilintisi yoktur.

Rastlantısallık (veya) olasılık bir nesnenin veya olayın bulanık olmayan (keskin-crisp) bir kümeye ait olup olmamasının belirsiz durumu ile ilgilidir.

Bir X sonlu evrensel olay kümesi ele alalım

$\mu_A(x)$=a belirli bir x∈ X elemanının A kümesinin üyelik fonksiyonunun değeridir.

P(x∈ A)=a x∈ X elemanının A kümesine ait olma olasılığıdır.

3.1.1 Üyelik fonksiyonları

Reel sayılar kümesindeki '2'ye yakın sayılar' bulanık kümesi aşağıdaki gibi çeşitli üyelik fonksiyonları ile ifade edilebilir.

$$\mu_1(x) = \begin{cases} 0 & x < 0 \text{ ve } x > 4 \\ \dfrac{x}{2} & 0 \le x \le 2 \\ \dfrac{4-x}{2} & 2 < x \le 4 \end{cases}$$

$$\mu_2(x) = \frac{1}{1+10(x-2)^2}$$

$$\mu_3(x) = e^{-|5(x-2)|}$$

$$\mu_4(x) = \begin{cases} 0 & x \le 1 \text{ ve } x \ge 3 \\ \dfrac{1+\cos[2\pi(x-2)]}{2} & 1 < x < 3 \end{cases}$$

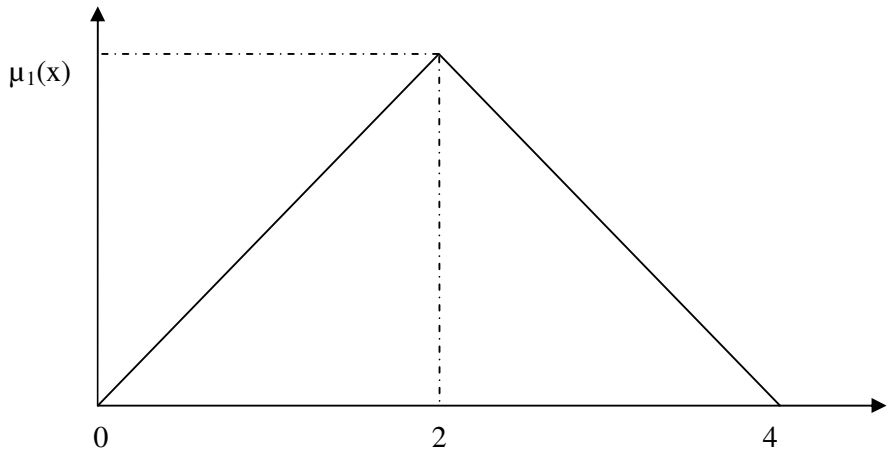
Farklı fonksiyonlar olmalarına rağmen dört fonksiyonun ortak özellikleri aşağıdaki gibidir.

i) $\mu_i(2)=1$ ve $\mu_i(x)<1 \quad \forall \quad x \neq 2$

ii) $\mu_i(x)$, x=2 noktasına göre simetriktir.

iii) $\mu_i(x)$ monoton olarak artar ve azalır.

Şekil 3.1'de gösterilen birinci üyelik fonksiyonunu ele alırsak, kümenin elemanı olma veya olmama hali belirli üyelik dereceleri ile ifade edilmektedir. Buna göre X evrensel kümesi tamsayılar kümesi olmak üzere '2' ye yakın sayılar' bulanık kümesi A={ 0/0, 0.5/1, 1/2, 0.5/3, 0/4} şeklinde liste biçiminde gösterilebilir. Bu gösterimde paydadaki sayılar X evrensel kümesine ait elemanları, paydaki sayılar ise elemanların üyelik derecesini göstermektedir. Liste gösteriminden anlaşıldığı gibi klasik küme kavramında kümenin elemanı olarak kabul edilmeyecek sayılar belirli bir üyelik derecesi ile A bulanık kümesine dahil olmaktadır.



Şekil 3.1: $\mu_1(x)$ üyelik fonksiyonu

3.1.2 Bulanık kümelerle ilgili temel kavramlar

Tanım: A ve B bulanık kümelerinin eşitlik, kapsama, tümleyen, birleşme ve kesişme özellikleri şu şekildedir.

- A ve B kümeleri ancak ve ancak $\forall x \in X$ için $\mu_A(x)=\mu_B(x)$ ise eşittir.

- Eğer her $x \in X$ için $\mu_A(x) \leq \mu_B(x)$ ise B kümesi A kümesini kapsar ve $A \subset B$ şeklinde gösterilir.

- Bir bulanık kümenin standart tümleyeni de bir bulanık kümedir ve üyelik fonksiyonu $c(\mu_A(x)) = 1 - \mu_A(x)$ şeklinde gösterilir.

- A ve B bulanık kümelerinin standart birleşimi AUB ile gösterilir ve üyelik fonksiyonu $\mu_{AUB}(x)=maks[\mu_A(x),\mu_B(x)]$ şeklinde gösterilir.

-A ve B kümelerinin standart kesişimi A∩B ile gösterilir ve üyelik fonksiyonu $\mu_{A\cap B}(x)=min[\mu_A(x),\mu_B(x)]$ şeklinde gösterilir.

Tanım: Üyelik dereceleri belli bir $\alpha \in [0,1]$ değerinden büyük veya eşit olan tüm elemanların oluşturduğu kümeye 'α-kesim' denir ve ${}^{\alpha}A$ biçiminde gösterilir.

$${}^{\alpha}A=\{x \in X \mid \mu_A(x) \geq \alpha\} \qquad (3.1)$$

Tanım: Üyelik dereceleri belli bir $\alpha \in [0,1]$ değerinden büyük olan tüm elemanların oluşturduğu kümeye ise 'güçlü α-kesim' denir ve ${}^{\alpha+}A=\{x \in X \mid \mu_A(x) > \alpha\}$ şeklinde ifade edilir.

Tanım: A bulanık kümesinin içindeki $x_1<x_2<x_3$ elemanları için $\mu(x_2) \geq min[\mu(x_1),\mu(x_3)]$ özelliği sağlanıyorsa, bu bulanık küme 'konvekstir' denir.

Teorem: R üzerinde tanımlı A bulanık kümesi ancak ve ancak

$\mu(\lambda x_1+(1-\lambda)x_2) \geq min(\mu(x_1),\mu(x_2)), \quad \forall x_1, x_2 \in R, \forall \lambda \in [0,1]$ koşulu sağlanıyorsa konvekstir.

Tanım: Bir A bulanık kümesinin üyelik fonksiyonunun değerinin sıfırdan büyük bütün bölgesine 'dayanak' adı verilir.

Tanım: Bir A bulanık kümesinin yüksekliği hgt(A) ile gösterilir ve

$$hgt(A)= \sup_{x\in X} \mu_A(x) \text{ şeklinde tanımlanır.} \qquad (3.2)$$

Tanım: 'Normal' (Olağan) bulanık küme, üyelik değerinin en az bir x elemanı için bire eşit olduğu kümedir.

Bir A bulanık kümesi için hgt(A)<1 ise bu kümeye 'olağanaltı' (subnormal) bulanık küme denir. Bir olağanaltı bulanık küme

$A^*(x) = \frac{A(x)}{hgt(A)}$ ifadesiyle normalize edilebilir.

Tanım: Eğer bir bulanık kümede bir ve yalnızca bir elemanın üyelik derecesi bire eşitse bu eleman kümenin 'prototipi' olarak adlandırılır.

Tanım: Bir bulanık kümeye ilişkin üyelik fonksiyonunun maksimum değerini aldığı noktaların ortalama değeri sonlu bir sayı ise bu sayıya o bulanık kümenin 'merkezi' denir. Eğer bu değer artı (eksi) sonsuz ise bu durumda üyelik fonksiyonunun maksimuma ulaştığı noktalar arasında en küçük (en büyük) olanı o bulanık kümenin 'merkezi' adını alır.

Tanım: Bulanık kümeye ait üyelik fonksiyonunun $\mu_A(x)=1$ olan bütün x elemanlarını kapsayan kümesi 'göbek ' olarak adlandırılır.

Tanım: Bir üyelik fonksiyonunda, üyelik derecelerinin 1'e veya 0'a eşit olmayan öğelerinin oluşturduğu kısımlara üyelik fonksiyonunun 'sınır' veya 'geçiş bölgeleri' denir($0<\mu_A(x)<1$).

3.1.3 Bulanık kümelerde işlemler

3.1.3.1 Bulanık tümleme

Bir bulanık kümenin tümleyeni c: $[0,1]\rightarrow[0,1]$ biçiminde bir fonksiyonla ifade edilir. $x_1,x_2\in X$ için $a=\mu_A(x_1)$ ve $b=\mu_A(x_2)$ olmak üzere c bulanık tümleme fonksiyonu şu aksiyomları sağlamalıdır.

1) $c(0)=1,\ c(1)=0$

2) $\forall\ a,b \in [0,1]$ için eğer $a \leq b$ ise $c(a) \geq c(b)$ olmalıdır

Bulanık tümlemenin iskeletini oluşturan bu iki özellik yanında c'nin sürekli bir fonksiyon olması ve $\forall\ a\in [0,1]$ için $c[c(a)]=a$ şartını sağlaması beklenebilir.

Bulanık tümleme fonksiyonlarına örnek olarak Sugeno tipi bulanık tümleme fonksiyonu verilebilir. Bu fonksiyon

$$c_\lambda(a)=\frac{1-a}{1+\lambda a},\quad \lambda\in(-1,\infty)$$

şeklinde ifade edilir. $\lambda=0$ durumu için Sugeno tipi bulanık tümleme fonksiyonu $c_\lambda(a)=1-a$ şeklinde tanımlanan standart bulanık tümleyene dönüşmektedir. Başka bir örnek olarak w parametre olmak üzere $c_w(a)=(1-a^w)^{1/w}$ $w\in(0,\infty)$ biçiminde

tanımlanan Yager tipi bulanık tümleme fonksiyonu verilebilir. Bu fonksiyon da w=1 için standart bulanık tümleyene dönüşür [26].

3.1.3.2 Bulanık birleşme özelliği (S normu , t-konormu)

İki bulanık kümenin birleşimi u:[0,1]x[0,1]→[0,1] biçiminde bir fonksiyonla ifade edilir. A ve B iki bulanık küme olmak üzere $a=\mu_A(x)$, $b=\mu_B(x)$ olsun. Bir fonksiyonun bulanık birleşim fonksiyonu olabilmesi için

1) u(0,0)=0, u(0,1)=u(1,0)=u(1,1)=1

2) u(a,b)=u(b,a)

3) u(u(a,b),d)=u(a,u(b,d))

4) eğer $b \leq d$ ise $u(a,b) \leq u(a,d)$ aksiyomlarını sağlaması gerekir. Ayrıca u'nun sürekli bir fonksiyon olması ve u(a,a)>a ve $a_1<a_2$, $b_1<b_2$ için $u(a_1,b_1)<u(a_2,b_2)$ koşullarını yerine getirmesi beklenebilir.

3.1.3.3 Bulanık kesişme özelliği (T normu)

İki bulanık kümenin kesişimi i:[0,1]x[0,1]→[0,1] biçiminde bir fonksiyonla ifade edilir. A ve B iki bulanık küme olmak üzere $a=\mu_A(x)$, $b=\mu_B(x)$ olsun. Bir fonksiyonun bulanık kesişim fonksiyonu olabilmesi için

1) i(1,1)=1, i(0,0)=i(0,1)=i(1,0)=0 (sınır koşulu)

2) i(a,b)=i(b,a) (değişim özelliği)

3) i(i(a,b),d)=i(a,i(b,d)) (dağılım)

4) $b \leq d$ için $i(a,b) \leq i(a,d)$ (monotonluk)

aksiyomlarını sağlamalıdır. Bu özelliklerin yanında bulanık kesişim fonksiyonunun sürekli bir fonksiyon olması ve i(a,a)<a ve $a_1<a_2$, $b_1<b_2$ için $i(a_1,b_1)<i(a_2,b_2)$ koşullarını yerine getirmesi beklenebilir.

3.1.3.4 Standart bulanık küme işlemciler

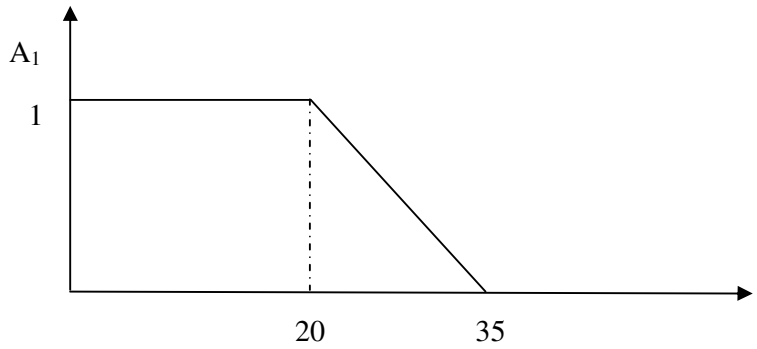
Bulanık tümleme, birleşim ve kesişim işlemlerinde yukarıda anlatılan aksiyomları sağlayan bir çok fonksiyon tanımlanabilir. Bunlar arasında aşağıdaki işlemler standart tümleme, birleşim ve kesişim olarak tanımlanır.

Standart tümleme $c(\mu_A(x))=1-\mu(x)$ (3.3)

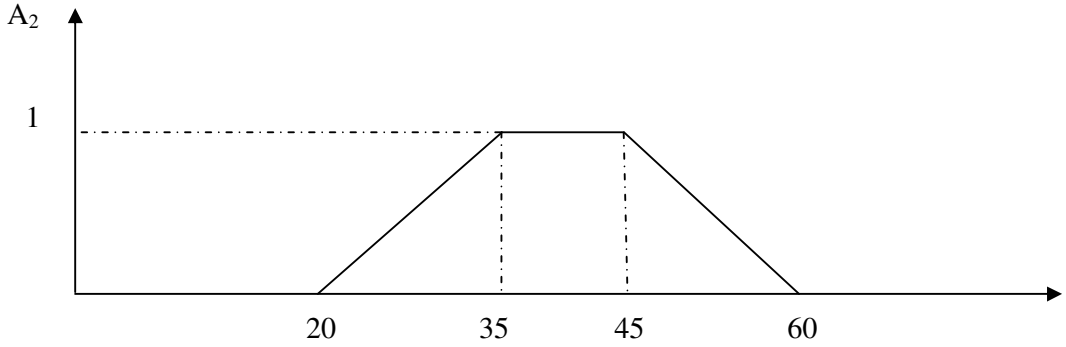
Standart birleşim $u(\mu_A(x),\mu_B(x))=maks(\mu_A(x),\mu_B(x))$ (3.4)

Standart kesişim $i(\mu_A(x),\mu_B(x))=min(\mu_A(x),\mu_B(x))$ (3.5)

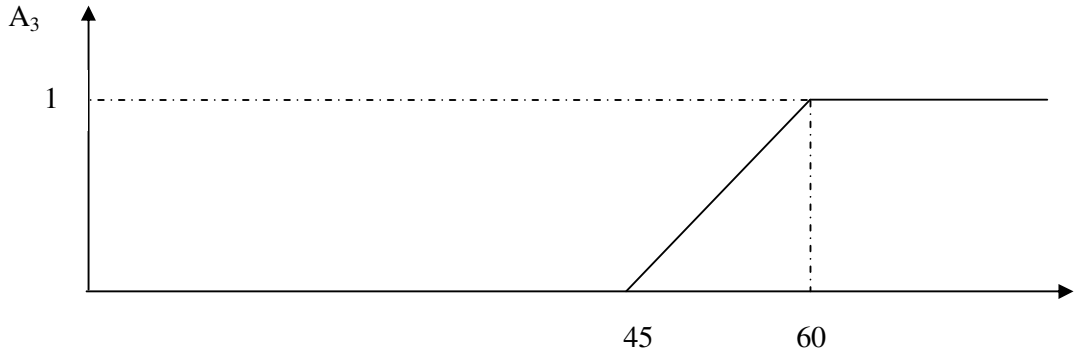
Aşağıdaki örneklerde standart tümleme, birleşim ve kesişim işlemleri gösterilmektedir. Şekil 3.2, 3.3 ve 3.4 sırasıyla "genç", "orta yaşlı" ve "yaşlı" bulanık kümelerini göstermektedir. Şekil 3.5'te A_1 kümesinin standart tümleyeni, Şekil 3.6'da $A_1 \cap A_2$ kümesi Şekil 3.7'de $A_2 \cap A_3$ Şekil 3.8'de ise BUC bulanık kümeleri gösterilmiştir.



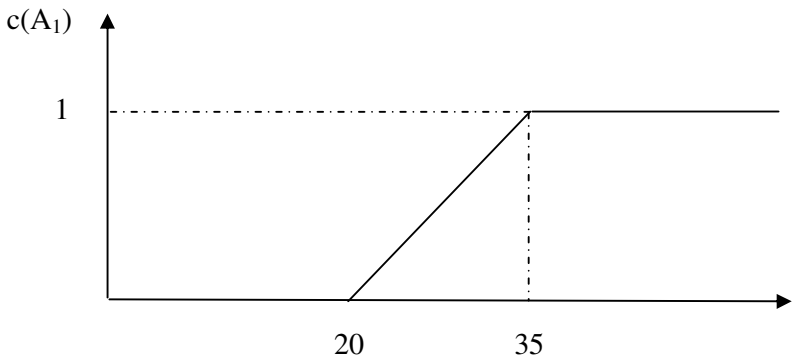
Şekil 3.2: "Genç" bulanık kümesi



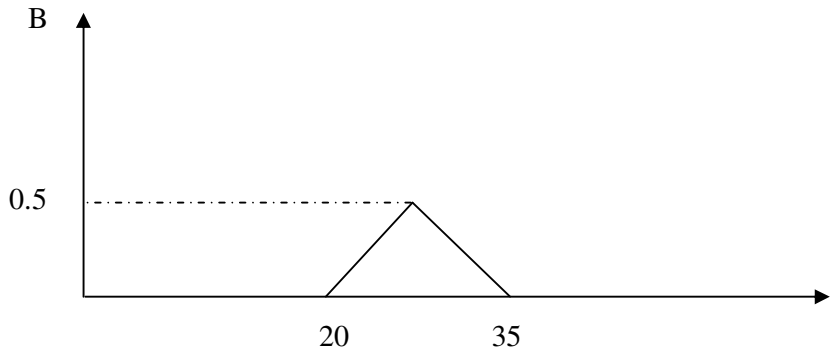
Şekil 3.3: "Orta yaşlı" bulanık kümesi



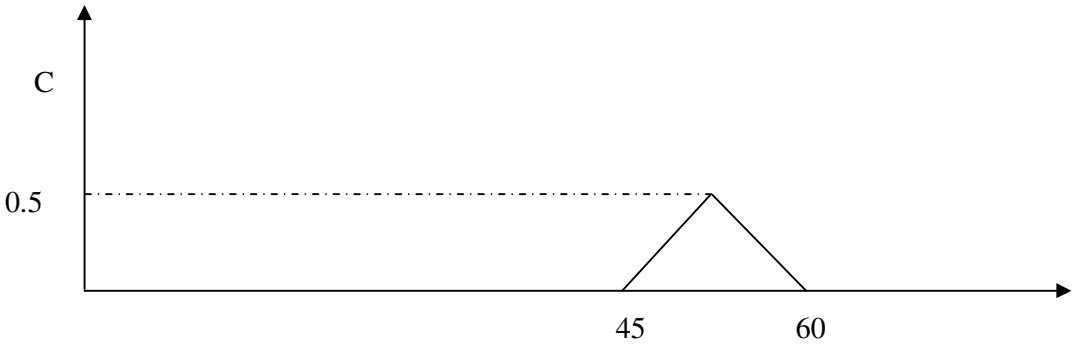
Şekil 3.4: "Yaşlı" bulanık kümesi



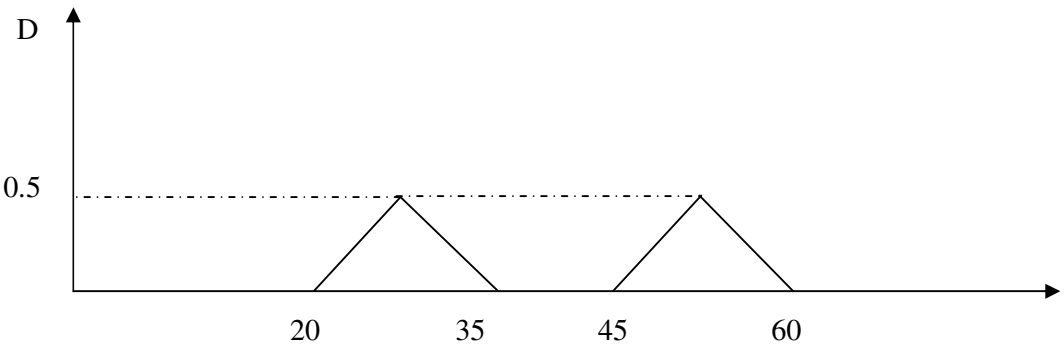
Şekil 3.5: A_1 bulanık kümesinin tümleyeni



Şekil 3.6: $B=A_1 \cap A_2$



Şekil 3.7: $C=A_2 \cap A_3$



Şekil 3.8: D=BUC

3.2 Bulanık Gerektirme (τ)

Tanım: Bir önermenin doğru (1) ya da yanlış (0) olmasına bu önermenin doğruluk değeri denir.

Tanım: Bulanık önermelerin doğruluk değerleri 0 ve 1 olmayıp, [0,1] aralığında değerler alır. Bu nedenle, doğruluk değeri kavramının yerini doğruluk derecesi kavramı alır.

Bulanık önerme τ: [0,1]x[0,1]$\rightarrow$[0,1]

Tanım: p ve q bulanık önermelerinin doğruluk dereceleri sırasıyla a ve b olmak üzere "Eğer p(öncül) o halde q(sonuç)" şeklinde koşullar önermenin doğruluk derecesi $\tau(a,b)$ olarak tanımlanır.

Tablo 3.1: Klasik Gerektirme ile Bulanık Gerektirme Bağlacının çeşitli ifade biçimleri

Klasik gerektirme($\Rightarrow$)	Bulanık gerektirme(τ)
$\bar{a} \cup b$	u(c(a),b)
maks $\{x \in \{0,1\}: a \cap x \leq b\}$	sup$\{x \in [0,1]: i(a,x) \leq b\}$
$\bar{a} \cup (a \cap b)$	u(c(a),i(a,b))

Yukarıdaki tablodaki ifadelerden yola çıkılarak ve tümleme, birleşim ve kesişim fonksiyonlarının çeşitliliği anımsanarak birçok bulanık gerektirme bağlacı üretilebilir.

Bulanık gerektirmenin sağlaması gereken aksiyomlar:

a) $a \leq b$ için $\tau(a,x) \geq \tau(b,x)$

b) $a \leq b$ için $\tau(x,a) \leq \tau(x,b)$

c) $\tau(0,a)=1$

d) $\tau(1,b)=b$

e) $\tau(a,a)=1$

f) $\tau(a,\tau(b,x))=\tau(b,\tau(a,x))$

g) $\tau(a,b)=1 \Leftrightarrow a \leq b$

h) $\tau(a,b)=\tau(c(b),c(a))$

i) τ sürekli bir fonksiyondur.

Bu özelliklere sahip bulanık gerektirmeler izleyen teoremle karakterize edilir.

Teorem: (Smets ve Magrez) Bir τ: $[0,1]^2 \rightarrow [0,1]$ gerektirme fonksiyonu ancak ve ancak $f:[0,1] \rightarrow [0,\infty]$ ve $f(0)=0$ biçiminde, artan ve sürekli bir f fonksiyonu varsa bu dokuz özelliği de sağlar, öyle ki her $a,b \in [0,1]$ için

$$\tau(a,b) = f^{-1}[f(1)-f(a)+f(b)] \tag{3.6}$$

$$c(a)=f^{-1}[f(1)-f(a)] \tag{3.7}$$

ifadeleri geçerlidir.

Tanım: u(c(a),b) gerektirme bağlacı S-gerektirmesi adını alır. c(a)=1-a (standart tümleme) seçilerek aşağıda tanınmış dört S-gerektirme bağlacı verilmiştir.

1) Kleene-Dienes gerektirme bağlacı

u(a,b)=maks(a,b) (standart birleşim) seçilerek

$\tau_b(a,b)=$maks(1-a, b)

2) Reichenbach gerektirme bağlacı

u(a,b)=a+b-ab (cebirsel toplam) seçilerek

$\tau_r(a,b)=1-a+ab$

3) Lukasiewicz gerektirme bağlacı

u(a,b)=min(1,a+b) (sınırlı toplam) seçilerek

$\tau_a(a,b)=$min(1,1-a+b)

4) En geniş S-gerektirmesi

$$u(a,b)=\begin{cases} a & b=0 \text{ ise} \\ b & a=0 \text{ ise} \\ 1 & \text{aksi halde} \end{cases}$$ (şiddetli birleşim) seçilirse

$$\tau_{LS}=\begin{cases} b & a=1 \\ 1-a & b=0 \\ 1 & \text{aksi halde} \end{cases}$$

biçiminde tanımlanır.

Tanım : $\sup\{x\in[0,1]: i(a,x)\le b\}$ şeklinde verilen bağlaç R-gerektirmesi adını alır.

1) Gödel gerektirme bağlacı

i(a,b)=min(a,b) seçilirse

$$\tau_{g}=\begin{cases} 1 & a\le b \\ b & a>b \end{cases}$$

2) Goguen gerektirme bağlacı

i(a,b)=a.b (cebirsel çarpım) seçilirse

$$\tau_{\Delta}=\begin{cases} 1 & a\le b \\ b/a & a>b \end{cases}$$

3) Lukasiewicz gerektirme bağlacı

i(a,b) sınırlı fark seçilirse

$\tau_a(a,b)=\min(1,1-a+b)$

4) En geniş R-gerektirmesi

$$\tau_{LR}(a,b)=\begin{cases} b & a=1 \\ 1 & \text{aksi halde} \end{cases}$$

biçiminde tanımlanır.

Tanım: u(c(a),i(a,b)) biçiminde verilen bulanık gerektirme bağlacı QL-gerektirme bağlacı adını alır.

1) Zadeh gerektirme bağlacı

kesişim ve birleşim için standart minimum ve maksimum operatörleri seçilirse

$\tau_m(a,b)=\text{maks}[1-a,\min(a,b)]$

2) i cebirsel çarpım ve u cebirsel toplam seçilirse $\tau_p(a,b)=1-a+a^2b$ ifadesi elde edilir.

3) i sınırlı fark ve u sınırlı toplam ise τ_b Kleene-Dienes gerektirme bağlacı elde edilir.

4) i bulanık kesişim fonksiyonu eğer

$$i(a,b)=\begin{cases} a & b=1 \\ b & a=1 \\ 0 & \text{aksi halde} \end{cases}$$

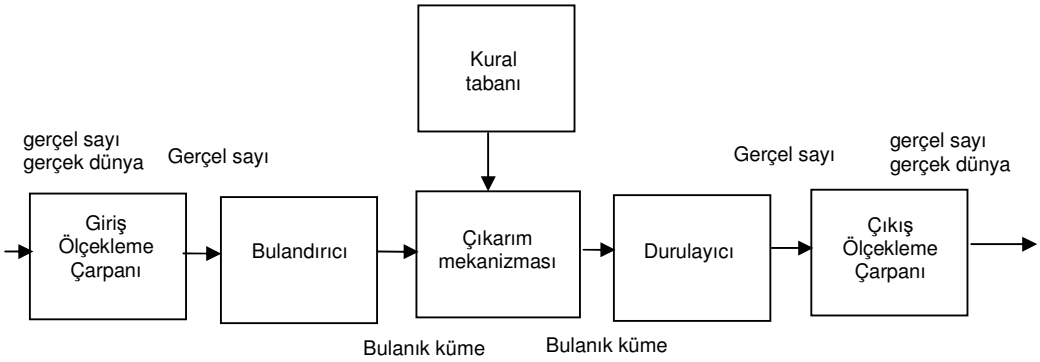
şeklinde tanımlanan şiddetli kesişim, u ise şiddetli birleşim olarak seçilirse

$$\tau_q(a,b)=\begin{cases} b & a=1 \\ 1-a & a\neq 1, b\neq 1 \\ 1 & a\neq 1, b=1 \end{cases}$$

biçiminde gösterilen bir başka bulanık gerektirme bağlacı elde edilir.

3.3 Bulanık Sistemler

Bulanık kümeler ve bunlarla ilişkili olan matematiksel yapıyı içeren, statik veya dinamik sistemlere genel olarak bulanık sistem denir. Bulanık sistemler temelini bulanık "EĞER-O HALDE" kurallarının oluşturduğu yapılardır. Temel bir bulanık sistem bulanık "EĞER-O HALDE" kurallarının bulunduğu kural tabanı ve bu kurallar aracılığı ile sistem giriş bulanık kümelerini sistem çıkış bulanık kümelerine dönüştüren çıkarım mekanizmasından oluşur. Sistemin giriş ve çıkışlarının bulanık kümeler oluşu onun mühendislik problemlerinde kullanımını kısıtladığından bir problemi çözmek amacıyla kural tabanı ve çıkarım mekanizmasından oluşan temel yapıya bulandırıcı ve durulayıcı adı verilen iki birim eklenir. Bulandırıcı reel değerli sistem girişini bulanık kümelere dönüştürürken, durulayıcı çıkarım mekanizmasının ürettiği bulanık kümeleri reel değerli sistem çıkışına dönüştürür. Ayrıca giriş ve çıkış değerlerini istediğimiz aralığa dönüştürmek için bulandırıcıdan önce ve durulayıcıdan sonra olmak üzere giriş ve çıkış ölçekleme çarpanları olarak adlandırılan iki birim kullanılmaktadır. Bu tür bulanık sistemlere ilişkin blok şema Şekil 3.9'da verilmiştir.



Şekil 3.9: Bulanık sistem yapısı

Bulandırıcı: gerçel değerli sistem girişini bulanık kümelere dönüştürür.

Durulayıcı: çıkarım mekanizmasının ürettiği bulanık kümeleri gerçel değerli sistem çıkışına dönüştürür.

Giriş ölçekleme çarpanı: fiziksel dünyadan elde edilen girişleri bulanık kurallarda yer alan giriş değişkenlerinin tanımlandığı aralığa dönüştürür.

Çıkış ölçekleme çarpanı: durulayıcıdan elde edilen gerçel sayıları fiziksel dünyadaki karşılığına dönüştürür.

3.3.1 Bulanık Kural Tabanı

Eğer öncül (antecedent) önerme O HALDE sonuç (consequent) önerme şeklindeki bulanık kurallarla ifade edilir.

Bulanık önermedeki sonuç ifadesinin yapısına göre bulanık kural tabanı üç farklı yapı oluşturur. Bunlar:

a) Mamdani tipi bulanık kurallar

b) Tekli(Singleton) tipi bulanık kurallar

c) Takagi-Sugeno(T-S) tipi bulanık kurallar

3.3.1.1 Mamdani tipi bulanık kurallar

K_i= Eğer $\tilde{x}$ A_i ise O HALDE $\tilde{y}$ B_i dir.

K_i= i. kural

$\tilde{x}$ =giriş(öncül) dilsel değişken

$\tilde{y}$ =çıkış(sonuç) dilsel değişken

A_i=öncül dilsel terim

B_i=sonuç dilsel terim

A_i ve B_i dilsel değişkenleri genellikle daha önceden tanımlanmış ' küçük, orta, büyük, negatif, pozitif' gibi terimlerden seçilir.

Mamdani tipi bulanık kurallara örnek olarak

"EĞER hava sıcaklığı yüksek ise O-HALDE yakıt tüketimi az dır" verilebilir.

3.3.1.2 Tekli (Singleton) tipi bulanık kurallar

Tanım $\mu_A(x)=\begin{cases}1 & x = x^* \\ 0 & \text{aksi halde}\end{cases}$

K_i= Eğer $\tilde{x}$ A_i ise O HALDE $\tilde{y}=b_i$ dir.(b_i skalardır)

Bu tip kurallara örnek olarak 'Eğer basınç fazla ise hacim 20 cm^3 tür' verilebilir.

3.3.1.3 Takagi-Sugeno tipi bulanık kurallar

Takagi-Sugeno tipi modelin sonuç kısmında bir keskin (belirgin) fonksiyon vardır.

K_i= Eğer $\tilde{x}$ A_i ise O HALDE $\tilde{y}=f_i(x)$

Genelde, her kuraldaki $f_i(x)$ fonksiyonu aynı yapıdadır, yalnızca parametreler değişiktir. Basitlik ve pratiklik açısından genelde $f_i(x)= a_i^T x + b_i$ şeklinde fonksiyonlar seçilir.

Tanım: Herhangi x girişi için bulanık kural tabanında en az bir kural bulunabilirse ki buna R_K diyelim

$\mu_{A_i}^{k}(x_i) \neq 0 \quad \forall i = 1,2,.....n$ olsun, bu durumda bulanık EĞER-O HALDE kural kümesi 'tamdır' denir.

Tanım: Bulanık kurallardan aynı öncül önerme için farklı sonuç önermeler yok ise bu kural tabanı 'Tutarlıdır' denir.

Tanım: Bulanık kümelerdeki herhangi bir kuralın öncül önermesinde bulunan bulanık kümenin diğer kurallardaki öncül önermelerdeki bulanık kümelerle kesişimi boş küme değil ise bulanık kural tabanı 'Süreklidir' denir.

3.3.2 Bulanık Çıkarım Mekanizması

Bulanık çıkarım mekanizması bulanık kural tabanında bulunan bulanık 'EĞER O HALDE' kuralları yardımıyla X giriş uzayından seçilen bir A bulanık kümesini Y çıkış uzayında B bulanık kümesine dönüştüren birimdir. $R^{(k)}$ XxY de bulanık bağıntı olmak üzere $R^{(k)}$: Eğer x_1, A_1^k ve ……ve x_n, A_n^k ise O Halde y, B^k dır. k=1,2,….M ile temsil edilsin yani $R^{(k)}= A_1^k$ x……..$A_n^k \rightarrow B^k$ olsun. Burada A_1^kx …….. x $A_n^k(x_1,x_2,…..x_n)=A_1^k(x_1)*…….A_n^k(x_n)$, '*' işlemi bir bulanık kesişimi belirtmek üzere $X=X_1$ x ….. X_n de bir bulanık bağıntı olarak ifade edilir. '→' sembolü ise bir bulanık gerektirme bağlacını ifade etmektedir. M tane kuralın (bağıntının) varolduğunu varsayalım ve bunları tek bir bağıntı ile ifade etmeye çalışalım. Eğer bileşke çıkarım kuralı $Q_M= \bigcup_{k=1}^{M} R^{(k)}$ olarak tanımlanacak olursa $Q_M=R^{(1)}(x,y) \oplus …….. \oplus R^{(M)}(x,y)$ olarak yazılabilir. Burada '$\oplus$' sembolü bir bulanık birleşimi ifade etmektedir. $Q_M= \bigcup_{k=1}^{M} R^{(k)}$ ifadesi Mamdani kombinasyonu olarak bilinir. Eğer bileşke çıkarım kuralı $Q_G= \bigcap_{k=1}^{M} R^{(k)}$ olarak tanımlanacak olursa $Q_G=R^{(1)}(x,y)*……*R^{(M)}(x,y)$ olarak yazılabilir, burada '*' sembolü bir bulanık kesişimi göstermektedir. $Q_G= \bigcap_{k=1}^{M} R^{(k)}$ ifadesi Gödel kombinasyonu olarak adlandırılır.

A, X den seçilen bir bulanık küme ve bir bulanık çıkarım mekanizmasının girişi olsun. Q_M ve Q_G 'yi tek bir bulanık "EĞER-O HALDE" kuralı olarak görerek bulanık çıkarım mekanizmasının çıkışı Mamdani kombinasyonu kullanıldığı durumda

$B(y)= \sup_{x \in X} i(A(x), Q_M(x,y))$ olarak, Gödel kombinasyonu kullanıldığı durumda ise

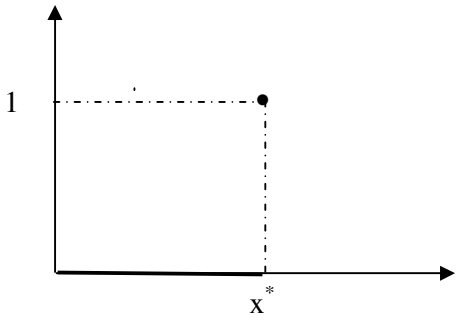
$B(y)= \sup_{x \in X} i(A(x), Q_G(x,y))$ olarak elde edilir [11].

3.3.3 Bulandırıcı

Bulanık sistemlerin yapılarındaki ilk blok bulandırıcılardır ve sisteme giren sayısal değeri $\mu_A(x)$ gibi bir bulanık kümeye çevirir. Bu işleme bulandırma denir.

3.3.3.1 Tekli (Singleton) bulandırıcılar

Bulanık sisteme giren bir sayısal değeri tekli bulanık kümeye çevirir.



Şekil 3.10: Singleton tipi bulandırıcı

3.3.3.2 Tekli olmayan bulandırıcılar

Bu tip bulandırıcılar ise sayısal değeri üçgen, gauss, çan gibi bulanık kümelere çevirir.

3.3.4 Durulayıcı

Durulayıcı çıkarım mekanizmasının ürettiği bulanık kümeleri gerçel değerli sistem çıkışına dönüştürür. Durulayıcı seçiminde göz önüne alınması gereken hususlar aşağıdaki gibidir.

- Akla yatkınlık: Durulayıcıdan elde edilen keskin çıkış değeri bulanık kümeyi sezgisel ve mantıksal açıdan iyi ifade etmelidir.

- Hesap basitliği: Algoritma kısa sürede hesaplanabilecek kadar basit olmalıdır.

- Süreklilik: Bulanık kümedeki küçük bir değişiklik, elde edilen keskin çıkışta çok büyük değişikliklere neden olmamalıdır.

Durulama yöntemleri:

a) Ağırlık merkezi

b) Alanlar toplamının merkezi

c) Maksimum durulayıcılar

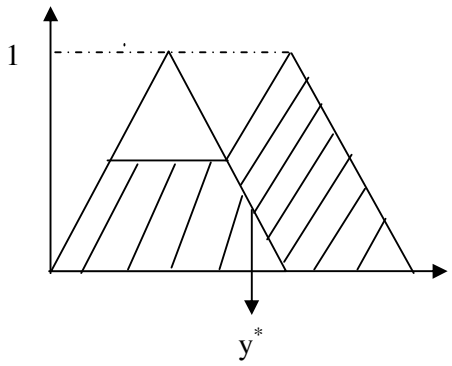
d) Yüksekliklerin ortalaması (Ağırlıklı ortalama)

a) Ağırlık merkezi (centroid) yöntemi

En yaygın kullanılan durulama yöntemidir. Birleşik çıkarım şeklinin ağırlık merkezinde yer alan üyelik derecesine karşılık gelen sayının bulunması ilkesine dayanır.

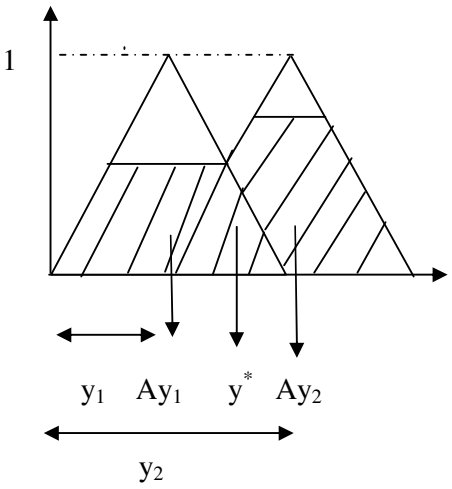
$$y^* = \frac{\int y\mu_{B'}(y)dy}{\int \mu_{B'}(y)dy} \qquad \text{sürekli} \qquad (3.8)$$

$$y^* = \frac{\sum y\mu_{B'}(y)}{\sum \mu_{B'}(y)} \qquad \text{kesikli} \qquad (3.9)$$



Şekil 3.11: Ağırlık merkezi durulama yöntemi

b) Alanların toplamının merkezi



Şekil 3.12: Alanların toplamının merkezi durulama yöntemi

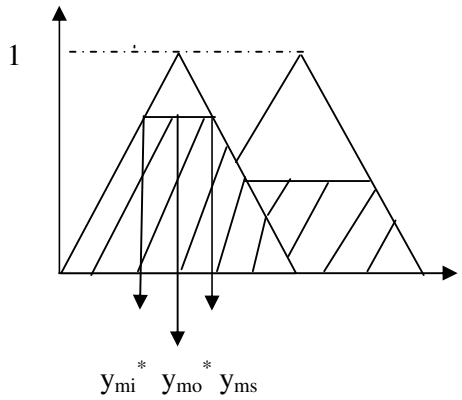
Bu yöntemde, bulanık kümelerin kesişim bölgesi iki kez hesaba katılmaktadır.

c) Maksimum Durulayıcılar

i) Maksimumun ilki (M_i)

ii) Maksimumun ortalaması (M_o)

iii) Maksimumun sonu (M_s)



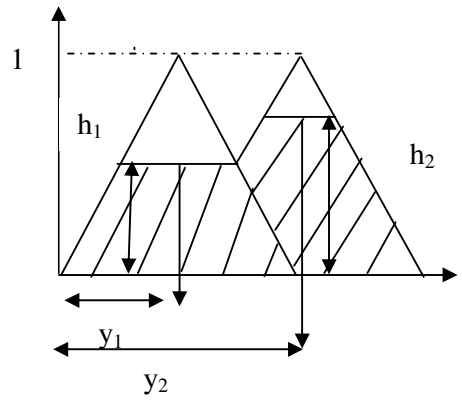
Şekil 3.13: Maksimum durulayıcılar durulama yöntemi

$$y_{mi}^* = \inf_{y \in Y} \{y \in Y \mid \mu_y(y) = hgt(y)\} \tag{3.10}$$

$$y_{ms}^* = \sup_{y \in Y} \{y \in Y \mid \mu_y(y) = hgt(y)\} \tag{3.11}$$

$$y_{mo}^* = \frac{y_{mi}^* + y_{ms}^*}{2} \tag{3.12}$$

d) Yüksekliklerin Ortalaması (Ağırlıklı ortalama)



Şekil 3.14 : Yüksekliklerin ortalaması durulama yöntemi

$$y^* = \frac{h_1 y_1 + h_2 y_2}{h_1 + h_2} \tag{3.13}$$

3.4 Bulanık Kontrolörler

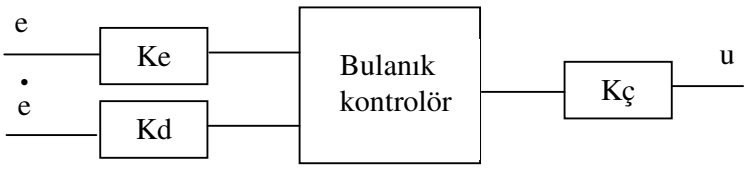
Bulanık kontrolörler, klasik kontrolörlere göre çok daha yeni olmalarına karşın giderek artan sayıda uygulamalarda kullanılmaktadır. Bulanık kontrolörler, klasik kontrolörlerden esinlenerek bulanık PD, bulanık PI, bulanık PID ve karma kontrolörler şeklinde kullanılmaktadır.

Bulanık PI tipi kontrol, bulanık PD tipi kontrole nazaran daha kullanışlı olarak bilinir çünkü bulanık PD tipi kontrol için sürekli hal hatasını ortadan kaldırmak, sistem tipi 0 ise imkansızdır. Ancak integrasyon nedeniyle PI tipi kontrolörün de yüksek mertebeli prosesler için geçici hal cevabında düşük başarım gösterdiği bilinmektedir. Bulanık PID tipi kontrolörün tasarımı için gerçekte üç boyutlu kural tabanı gerekmektedir ancak uzman operatörün algılaması açısından bu oldukça güçtür. Bu sebepten ötürü oran, integral ve türev kontrol elemanlarını bir araya getirerek PI ve PD etkilerini birleştirmek yoluyla bulanık PID tipi kontrolör tasarlama yoluna gidilmektedir [37-39]. Bulanık PID tipi kontrolörü, bulanık PI ve bulanık PD tipi kontrolörleri iki ayrı kural tabanı ile birleştirerek ya da bir bulanık PD tipi kontrolörün çıkışına integral ve toplam elemanı koyarak elde etmek mümkündür [39].

3.4.1 Doğrudan harekete geçen (direct-action) bulanık kontrolörler

3.4.1.1 Bulanık PD tipi kontrolör

Bulanık PD kontrolöründe hata (e) ve hatanın türevine ($\dot{e}$) göre bir kontrol işareti üretilir. Şekil 3.15'de böyle bir yapı gösterilmektedir.

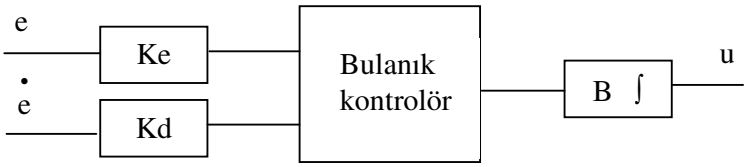


Şekil 3.15: Bulanık PD tipi kontrolör

Bulanık PD tipi kontrolör, yaklaşık olarak bir PD kontrolörü gibi davranacağından, eğer sistem tipi sıfır ise bulanık PD tipi kontrolörün sistem yanıtında sürekli hal hatası oluşacaktır. Böyle durumlarda kontrolöre integral kontrolü de dahil edilir.

3.4.1.2 Bulanık PI tipi kontrolör

Hata (e) ve hatanın türevi ($\dot{e}$) girdiler olmak üzere bir işaret üretilir. Buradan sonra integral alma işlemi uygulanmaktadır. Böyle bir yapı şekil 3.16'da gösterilmiştir.

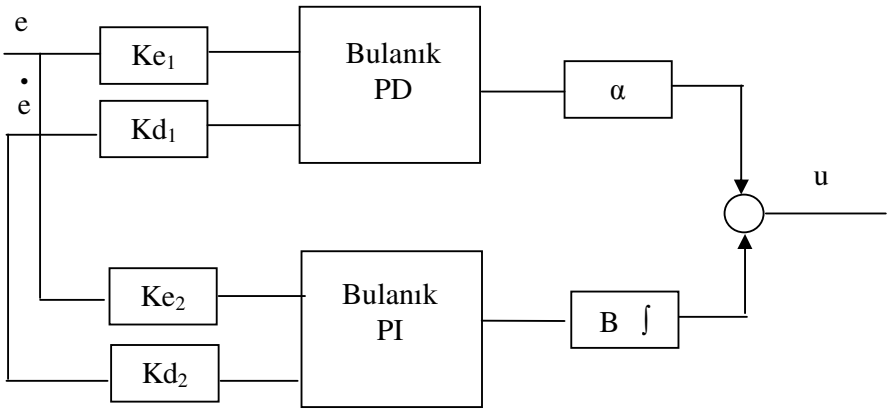


Şekil 3.16: Bulanık PI tipi kontrolör

3.4.1.3 Bulanık PID tipi kontrolör

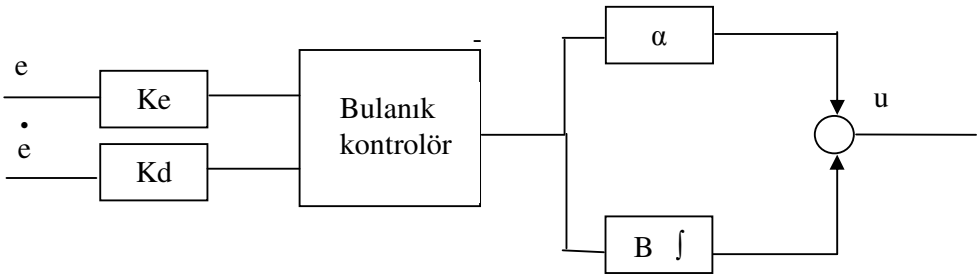
Bir PID tipi kontrolörün başarımı, kontrolöre ait Kp oransal parametresi, Ki integral parametresi ve Kd türev parametresi tarafından belirlenir. Oransal kontrol sistem cevabını hızlandırır fakat yüksek değerleri sistemi kararsızlığa götürür. İntegral kontrolü sistem yanıtındaki sürekli hal ortadan kaldırır fakat yüksek değerleri aşımı ve osilasyonu artırır. Türev kontrolü ise sistem yanıtının sönümünü artırarak, aşım ve osilasyonu azaltır bu da geçici hal cevabında iyileşme sağlar. Dolayısıyla PID kontrolör parametreleri doğru şekilde seçilirse yükselme ve yerleşme zamanı düşük, az aşımlı ve kalıcı hal hatası olmayan ideal cevaplar elde edilir.

Bulanık PID kontrolöründe sıkça kullanılan iki yapı aşağıda verilmiştir. Şekil 3.17 de verilen birinci kontrolörde bulanık PD ve bulanık PI kontrolörlerinin birleştirilmesi sonucu yeni bir kontrolör elde edilmektedir. Bu yapı hata, hatanın türevi ve hatanın integralini kullanarak oluşturulan bulanık PID kontrolüne göre daha kullanışlıdır. Çünkü o yapıda örnek olarak her giriş için beş adet dilsel değişken kullanıldığını düşünürsek, toplamda 125 tane kural varken, bu yapıda ise her bir kural tablosu için 25 olmak üzere toplam 50 kural mevcuttur.



Şekil 3.17: Bulanık PID tipi kontrolör (1)

Yukarıdaki yapıya göre daha kullanışlı bir bulanık PID kontrolörü yapısı Şekil 3.18 de gösterilmiştir. Bu yapıda kural tabanı tek olduğundan her girişe ait 5 dilsel değişken seçilmesi durumunda toplamda sadece 25 adet kural yazılması gerekir. Yalnız bu kontrolör bir önceki kontrolöre göre daha az sayıda ölçekleme faktörüne sahip olduğundan dolayı ölçekleme faktörlerinin doğru ayarlanması daha önemlidir.



Şekil 3.18: Bulanık PID tipi kontrolör (2)

Literatürde, iki girişli bulanık PID kontrolörü yapıları haricinde tek girişli ve üç girişli yapılar da olmasına karşın kullanımları pek yaygın değildir.

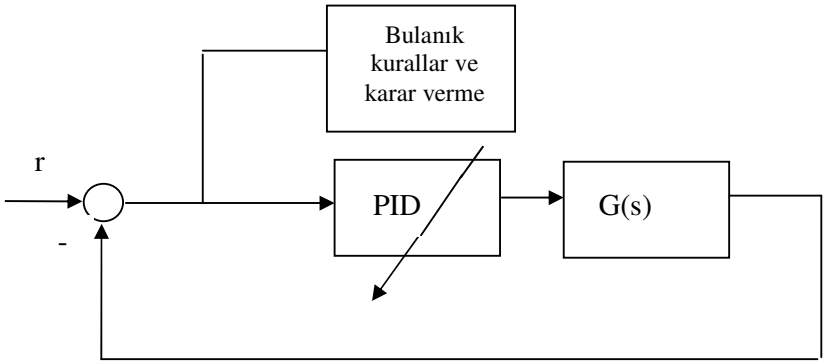
3.4.2 PID kontrolör kazancını hesaplayanlar

Şekil 3.19'daki gibi gösterilen kontrolör için tasarım parametreleri iki ayrı grupta toplanabilir [40].

i) Yapısal parametreler

ii) Ayarlama parametreleri

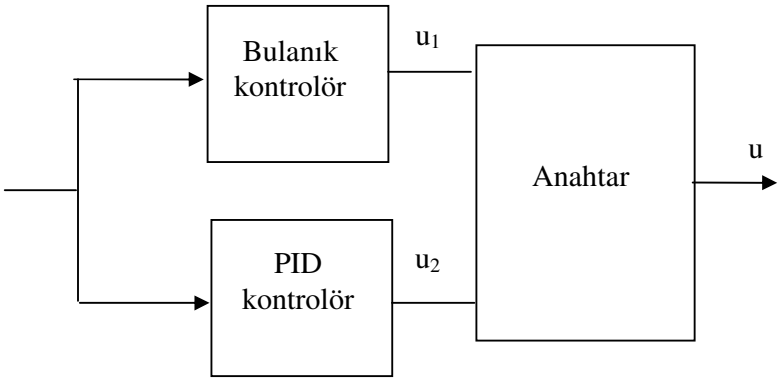
Temel olarak yapısal parametreler, bulanık çıkarım için giriş/çıkış (I/O) değişkenlerini, bulanık dilsel kümeleri, üyelik fonksiyonlarını, bulanık kuralları, çıkarım mekanizmasını ve durulama mekanizmasını içerir. Ayarlama parametreleri ise giriş/çıkış ölçekleme faktörlerini ve üyelik fonksiyon parametrelerini içerir.



Şekil 3.19: PID kontrolör kazancını hesaplayan bulanık kontrolör yapısı

3.4.3 Karma Kontrolörler

Kompleks yapıdaki gerçek sistemler için tek kontrolör hedeflerine ulaşmada yetersiz kalabilir ya da bunun için çok karmaşık bir kontrolör tasarımı gerekir. Bu nedenle pratikte iki seviyeli kontrolörler tasarlanır. Genellikle alt seviyeli kontrolörler yüksek hızlı bölgede, yüksek seviyeli kontrolörler ise düşük hız bölgesinde etkin olur.



Şekil 3.20: Karma kontrolör

Şekil 3.20'de görülen karma kontrolörde PID ve bulanık kontrolör beraber kullanılmıştır. Bu yapıda bulanık algoritmanın oluşturulmasında daha az üyelik fonksiyonuna ve daha küçük bir kural tabanına ihtiyaç duyulur. Tek başına bulanık algoritma kullanan bir kontrolöre göre daha küçük bir hafıza ve daha basit işlemci kullanma avantajı vardır. Burada anahtar elemanın görevi u işaretini, u_1 ve u_2 den

faydalanarak oluşturmaktır. Bulanık kontrolör nonlineer bir PD kontrolör gibi görev yaparken PID'nin integral terimi sürekli hal hatasını sıfırlar. Sistem referans değere yakınken PID kontrolörün, uzakken ise bulanık kontrolörün etkisindedir.

4. SİMÜLASYONLAR

Genetik algoritmaların bulanık mantık kontrolörü tasarımında etkinliğini araştırmak için Matlab 7.0 (R14) ve Simulink 6.0 (R14) kullanılarak simülasyonlar yapılmıştır. Bulanık mantık kontrolörü ve genetik algoritma işlevleri için ayrı ayrı fonksiyonlar kullanılmış ve bu fonksiyonlar "omurga" isimli ana m-dosyanın (m-file) altında çalıştırılmıştır. Populasyon içindeki bireylerin uygunluk değerlerinin hesaplanabilmesi içinse, Simulink'te oluşturulan model, program tarafından her birey için çalıştırılmıştır.

Programın işleyişi sırasında denenen her bireyi oluşturan parametrelerin ikilik ve onluk tabandaki değerleri, her bireyin amaç ve uygunluk fonksiyonu değerleri ve de her birey için kullanılan bulanık kontrolör, program içinde tanımlanan değişkenlere atandığından; program çalışmasını bitirdiğinde herhangi bir nesilde denenen herhangi bir birey hakkındaki bütün bilgilere ulaşılabilir. Çalışma ortamına (Workspace) atılan veriler programın son satırındaki 'save' komutuyla .mat uzantılı bir dosya olarak diske kaydedilmektedir. Ayrıca, program çalışırken denenen her kromozom için sistem çıkışı (y_{out}) ekrandan izlenebilmektedir.

Simülasyonlarda kullanılan tüm fonksiyon m-dosyaları, script (düzyazı) m-dosyaları ve Simulink'te oluşturulan modeller tezle birlikte verilen bir disket içinde bulunmaktadır. Bu kısımda sadece 'omurga' isimli ana m-dosya incelenecektir.

4.1 Çalıştırılan Program :Omurga

```
maxgen=100;

GGAP=0.9;

global FUZZY_SCALING

global FUZZY_SYSTEM
```

```
FieldD = [2 2 2 7   7   7   7   7   7   1  1  1  1  1  1 10 10 17     11;
          3 3 3 0.1 0.1 0.1 0.1 0.1 0.1 -1 -1 -1 -1 -1 -1  0  0  0      0;
          9 9 9 1   1   1   1   1   1   1  1  1  1  1  1  1 20 10 10000 2*pi;
          0 0 0 0   0   0   0   0   0   0  0  0  0  0  0  0  0  0  0      0;];

Chrom=CRTBP(50,102);

gen=0;

[ObjV]=run_fuzzysimmodel(BS2RV(Chrom, FieldD));

while gen<maxgen

FitnV=Obj_fun1(ObjV);

SelCh=SELECT('SUS', Chrom, FitnV, GGAP);

SelCh=RECOMBIN('XOVSP', SelCh, 0.7);

SelCh=MUT(SelCh, 0.01);

[ObjVSel]=run_fuzzysimmodel(BS2RV(SelCh,FieldD));

[Chrom,ObjV]=REINS(Chrom, SelCh, 1, 1, ObjV, ObjVSel);

gen=gen+1;

[kk,ll]=deger(BS2RV(Chrom, FieldD));

katsayilar(:, :, :, gen)=kk;

bulanik_sistem(:, :, :, gen)=ll;

marek(gen, :, :)=BS2RV(Chrom, FieldD);

karek(:, :, gen)=BS2RV(Chrom, FieldD);

ObjaV(:, gen)=ObjV;

FitnaV(:, gen)=FitnV;

end

Chrom

ObjV

FitnV=Obj_fun1(ObjV)

save degerlerim.mat
```

maxgen: durma şartı belirlenen bir nesil sayısı boyunca programın çalışmasıdır. maxgen değeri programın 100 nesil boyunca çalıştırılacağını gösterir. Bu durma şartı kullanıcı tarafından istenilen şekilde belirlenir.

GGAP: Programda elitist model kullanılmıştır ve bu örnek için değeri 0.9 dur yani populasyon büyüklüğünün 0.1'i miktarda uygunluk değeri en yüksek olan bireyler diğer nesle değişmeden aktarılır.

global FUZZY_SCALING, global FUZZY_SYSTEM: Matlab'da tanımlanan her fonksiyonun ayrı hafıza alanı vardır. Bu yüzden bir fonksiyonda kullanılan bir değişkenin diğer fonksiyonlarda da kullanabilmesi için bu değişkenler global değişken olmalıdır. FUZZY_SCALING değişkeni kullanılan kontrolördeki ölçekleme katsayılarını FUZZY_SYSTEM ise FIS (Fuzzy Interference System) yapısını ifade eder.

FieldD: Bu matrisin ilk satırı bir kromozom zincirini oluşturan değişkenlerin kaçar bitlik yer kapladıklarını, ikinci satır bu değişkenlerin alt sınırlarını, üçüncü satır üst sınırlarını, son satır ise ikilik tabanda yapılan bu kodlamada standart ikilik ya da Gray kodlamadan hangisinin kullanılacağını gösterir. (0'lar standart ikilik kodlamayı, 1'ler ise Gray kodlamayı ifade eder). Kromozomu oluşturan bu değişkenlere bulanık kontrolör açıklanırken tekrar değinilecektir. FieldD'nin ilk satırı aşağıdaki gibidir.

2	2	2	7	7	7	7	7	7	1	1	1	1	1	1	10	10	17	11
n			c_{m1}			p_{m1}			c_{m2}			p_{m2}			scale			tan

CRTBP (Create Binary Population-İkilik Sayı Tabanında Populasyon Oluşturma) Bu fonksiyon ikilik tabanda rastgele bir populasyon oluşturmayı sağlar. Bu örnek için oluşturulan 'Chrom' değişkeni 50x102 boyutunda bir matristir. 50 populasyon büyüklüğünü yani birey sayısını, 102 ise her bireyin 102 bitlik yer kapladığını gösterir. Bu değer FieldD matrisinin ilk satırındaki değişkenlerin kapladığı yerin toplamıdır.

BS2RV (Binary String to Real Vector-İkilik tabandan gerçek sayı değerlerine dönüşüm) Bu fonksiyon girilen ikilik tabandaki 'Chrom' populasyonundaki değişkenlerin onluk tabandaki karşılıklarını FieldD alanındaki her değişkenin kapladığı alana, alt ve üst sınırlarına ve kodlama türüne göre hesaplar. Bu fonksiyon

altında 'CRTBASE' adlı istenen sayı tabanında satır vektörü oluşturmayı sağlayan bir fonksiyon çalışır.

run_fuzzysimmodel: Girdi olarak gerçek değerli popülasyonu alarak, her birey için Simulink modelini çalıştırır ve belirlenen performans kriterine göre bireylerin amaç fonksiyonu değerlerini hesaplayarak çıkışında [N_{ind}x1] boyutunda bir matrisle verir.

FIS yapısının oluşturulabilmesi için bu fonksiyonun altında 'CREATE_FIS' isimli bir fonksiyon çalışmaktadır. Bunun da altında çalışan 'CREATE_MFS' ve 'CREATE_RULES' isimli iki fonksiyon da üyelik fonksiyonlarının ve kural tabanının oluşturulmasını sağlar.

Bulanık Kontrolör Yapısı: Bulanık kontrolörde yapılan kabuller şunlardır:

- FIS (Fuzzy Interference System) yapısı iki giriş ve tek çıkıştan oluşmaktadır ve Mamdani tipi bulanık kurallar kullanılmaktadır.

- Modelde üçgen tipi üyelik fonksiyonları kullanılmaktadır.

- Her giriş/çıkış değişkeni [-1, +1] aralığında tanımlanır.

- Üyelik fonksiyonlarının sayısı [3, 9] arasında herhangi bir tek sayıdır.

- Üyelik fonksiyonlarının oluşturduğu şekil y-eksenine göre simetriktir.

- İlk üyelik fonksiyonunun tepe noktası -1, son üyelik fonksiyonunun ise +1 dir. İlk üyelik fonksiyonu hariç bir üyelik fonksiyonu kendisinden önceki üyelik fonksiyonun tepe noktasından başlar ve son üyelik fonksiyonu hariç kendisinden sonra gelen üyelik fonksiyonun tepe noktasında biter (Bir giriş, en fazla iki üyelik fonksiyonunu ateşleyebilir).

Kullanılan bu yöntemde, giriş/çıkış değişkenleri için üyelik fonksiyonlarının sayısı ve üyelik fonksiyonlarının sınır noktaları n ve c_m değişkenleri vasıtasıyla genetik algoritmalar tarafından ayarlanmaktadır.

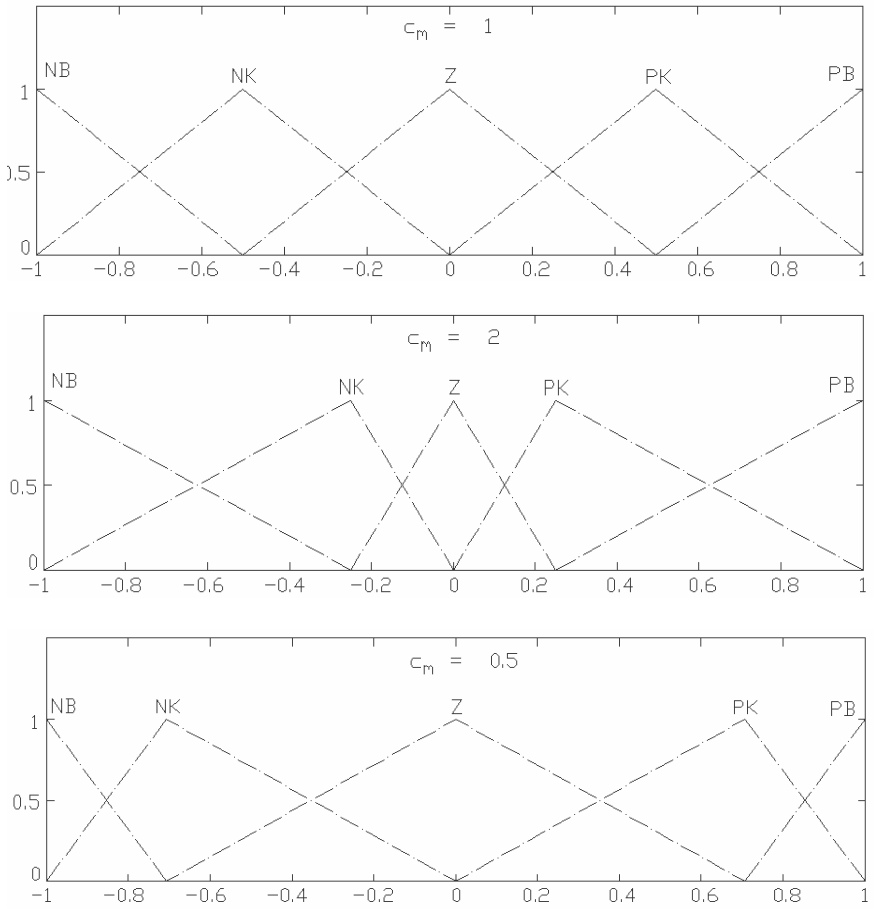
n: Bulanık kontrolördeki her giriş/çıkışın kaçar üyelik fonksiyonuyla tanımlandığını belirtir. Üyelik fonksiyonlarının sayısı en az üç en fazla dokuz ve tek sayı olduğundan bu değer için 2 bitlik bir alan yeterlidir. Kullanılan kontrolör iki girişli, tek çıkışlı olduğundan her bireyde altı bitlik alan üyelik fonksiyonlarının sayısının belirlenmesi için kullanılır.

c_m= Üçgen üyelik fonksiyonlarının tepe noktalarını belirlemekte kullanılır. Yukarıda anlatılan kabuller altında üyelik fonksiyonlarının tepe noktaları belirlenirse üyelik fonksiyonunun sınırları belirlenmiş olur. Girilen c_m değerine göre merkez noktalar $\left(\frac{i}{n}\right)^{\wedge} c_m$ formülüyle hesaplanır.

$$n = \left(\frac{\text{üyelik_fonk_sayisi} - 1}{2}\right)$$

i=1,2,…..n

Örnek olarak beş üyelik fonksiyonu olduğunu düşünürsek

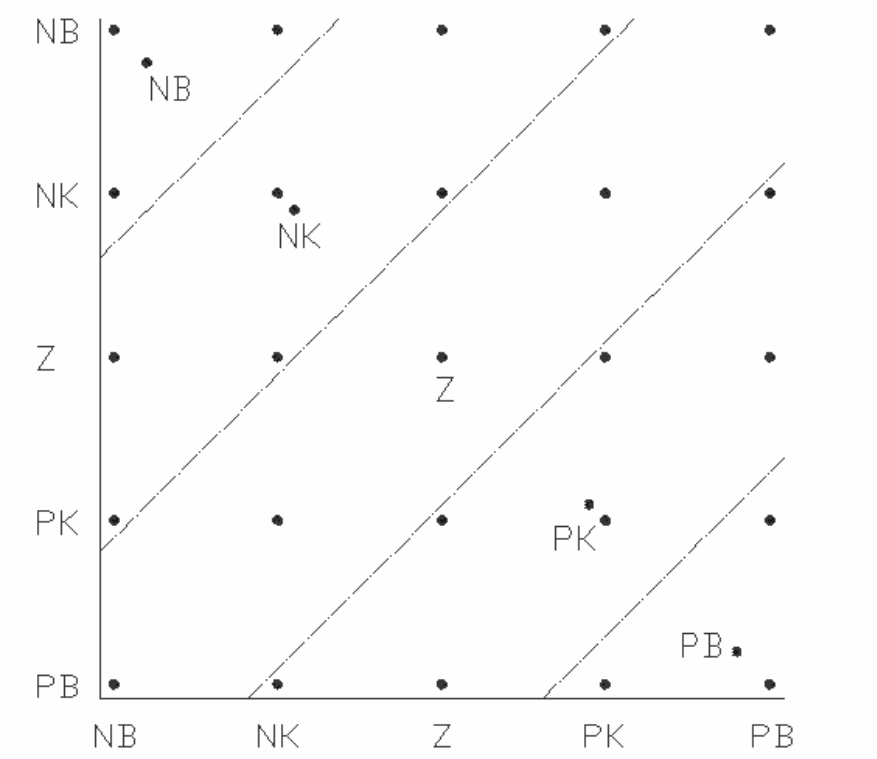


Şekil 4.1: c_m değerinin üyelik fonksiyonlarına etkisi

Görüldüğü gibi c_m>1 için tepe noktalar orijine yaklaşırken c_m<1 için sınırlara doğru yaklaşmaktadır. c_m değeri FieldD alanında iki kısımdan oluşur. c_{m1} her giriş/çıkış için büyüklüğü [0.1, 1] aralığında 7 bitlik bir diziyle ifade edilirken c_{m2} değeri -1 yada +1 olan bir bitlik yer kaplar. (değer 0 ise -1, 1 ise +1).

$c_m=c_{m1}\hat{}c_{m2}$ şeklinde hesaplanır. c_{m1} değeri c_m'in büyüklüğünü c_{m2} ise üyelik fonksiyonunun tepe noktasının merkeze mi yoksa sınıra doğru mu yaklaşacağını belirlemiş olur.

p_m= Bu değer kural tabanının oluşturulmasında c_m'e benzer bir görevde kullanılır. Bilindiği üzere kurallar oluşturulurken girişlerin tüm birleşimleri için bir çıkış belirlenir. Bunu bir örnekle açıklamak gerekirse beşer dilsel terimle tanımlı iki giriş için 25 kural vardır. Bu birleşim düzlemde 25 noktadan oluşan bir ızgara yapısı şeklinde düşünülebilir. p_m alanının ilk iki parçası kullanılan iki giriş için bu ızgara noktalarının x ve y koordinatlarını (x koordinatları birinci giriş y koordinatları ise ikinci giriş için) üyelik fonksiyonlarının orta noktalarını belirlemekte kullanılan yöntemdeki gibi hesaplar. p_m'in üçüncü parçası ise aynı yöntemle çıkış için x ekseni üzerinde 5 nokta belirler. 'tan' değişkeni bu beş noktanın hangi açıda bir doğrunun üzerinde yer alacağını belirler, yani bu x koordinatları 'tan' değeriyle çarpılarak y koordinatları hesaplanır. Her bir kurala karşılık düşen çıkış üyelik fonksiyonunu belirlemek için, kurallara karşılık gelen 25 ızgara noktasının (grid point) her birisiyle çıkışlara karşılık gelen beş noktanın arasındaki mesafe hesaplanır. Her bir kural için bu beş mesafe değerinden en küçüğü çıkış üyelik fonksiyonunu belirler.



Şekil 4.2: Bulanık kuralların oluşturulması

Scale: Bu kısım kontrolörde kullanılan ölçekleme katsayılarının kaçar bitlik yer kapladığını gösterir. Bu katsayıların alabileceği alt ve üst sınırlar kullanıcı tarafından girilmelidir.

FitnV=Obj_fun1(ObjV): Bu fonksiyon, bireylerin amaç fonksiyonu değerlerini uygunluk değerlerine çevirir. Bu örnek için Obj_fun1 isminde bir fonksiyon kullanılmıştır. ObjV ve FitnV [N_{ind}x1] boyutunda sütun vektörleridir. Tezle birlikte verilen diskette sıralı (Ranking) uygunluk değeri atanmasını sağlayan 'RANKING' isimli bir fonksiyon da mevcuttur.

SelCh=SELECT('SUS', Chrom, FitnV, GGAP): Bu fonksiyon populasyon içinden bireylerin seçimini sağlar. Bunu da fonksiyonun girdi parametrelerinden biri olan 'FitnV' uygunluk değerleri vektörüyle yapar. Elitist model kullanıldığından her populasyondan N_{ind}x(1-GGAP) kadar birey doğrudan yeni nesle taşınırken, kalan kısım için bireyler, kimin hangi miktarda üreyeceğinin belirlenmesi için birbirleriyle uygunluk değerleri baz alınarak yarışır. SELECT fonksiyonunun altında çalışan, seçim işleminin uygulanmasını sağlayan SRS(Stochastic Remainder Selection), RWS(Roulette Wheel Selection), SUS(Stochastic Universal Sampling) fonksiyonları disketin içinde mevcuttur. SelCh, seçilen bireyleri saklayan [(N_{ind}xGGAP), L_{ind}] boyutunda bir matristir.

SelCh=RECOMBIN('XOVSP', SelCh, 0.7): Diğer nesle doğrudan aktarılacak bireylerin dışındaki seçilmiş bireyler işlem havuzuna alınırlar. Burada RECOMBIN ana fonksiyonunun altında çalışan, çaprazlama metodunu belirleyen fonksiyona göre, değeri kullanıcı tarafından belirlenen çaprazlama olasılığı(P_c) oranında çaprazlamaya tabi tutulurlar. Bu işlemi sağlayan 'XOVSP'(Crossover Single Point) ve 'XOVMP'(Crossover Multi Point) fonksiyonları diskette bulunmaktadır.

SelCh=MUT(SelCh, 0.01): Bu fonksiyon çaprazlamaya göre çok daha küçük olasılıklarla bireylerin mutasyon işlemine tabi tutulmasını sağlar.

[Chrom,ObjV]=REINS(Chrom, SelCh, 1, 1, ObjV, ObjVSel): REINS fonksiyonu mutasyon işleminden çıkan bireylerin eski bireylerle yer değiştirmesini sağlar. SelCh populasyonundaki bireyler Chrom populasyonundaki N_{ind}xGGAP bireyin yerini alır. Chrom içindeki N_{ind}x(1-GGAP) birey değişmeden kalır. Fonksiyona girdi olan 3. parametredeki 1, Chrom içindeki uygunluk değeri en düşük elemalardan başlanılarak yer değiştirmenin yapılacağını belirtir. Bu değer 0 seçilirse yer değiştirme rastgele

yapılır. 4. parametre ise SelCh'taki bütün bireylerin yeni populasyona katılacağını belirtir. Bu oran 1'den küçük seçilerek de yer değiştirme yapılabilir. Bu fonksiyonun çıkışında yeni popülasyondaki bireylerin kodlu hali ve bireylerin amaç fonksiyonu değerleri elde edilir.

Bu fonksiyondan sonra kullanılan 'deger' fonksiyonu ve diğer değişkenler popülasyonda kullanılan her bireyin bilgilerini depolamak için kullanılır.

N_{ind} bireyden oluşan X nesil boyunca çalışan ve bulanık kontrolörde Y ölçekleme katsayısının olduğu bir programda

kk: N_{ind}xY

ll: (N_{ind} x 1) (1x1) struct

katsayilar: N_{ind}xYxX

bulanik_sistem: (N_{ind} x X) (1x1) struct

marek: X x N_{ind} x N_{var}

karek: N_{ind} x N_{var} x X

ObjaV: N_{ind}xX

FitnaV: N_{ind}xX

boyutundadır.

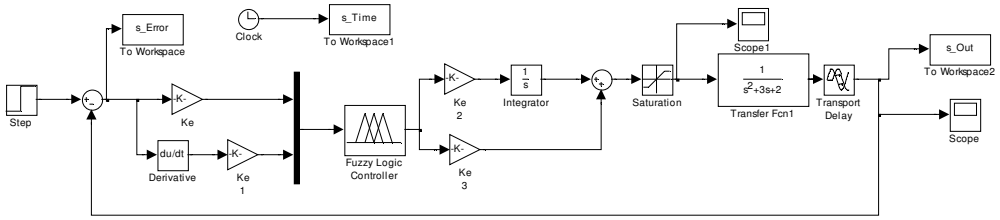
4.2 Simülasyonu Yapılan Sistemler

Yapılan simülasyonlarda popülasyonlar 50 bireyden oluşmakta ve bireyler ikilik tabanda kodlanmaktadır. Birinci sistem için yapılan simülasyonlar on saniye, ikinci sistem için yapılan simülasyonlar ise altı saniye boyunca çalıştırılmış, tüm simülasyonlarda örnekleme periyodu 0.01 sn. alınmış ve çözüm metodu olarak Ode4 (Runge-Kutta) çözücüsü seçilmiştir. Simülasyon çalışmasını bitirdiğinde genetik algoritma tarafından bulunan en başarılı bireyler bulanık kontrolöre uygulanmaktadır.

Yöntemin etkinliğini tespit etmek amacıyla yapılan simülasyonlar iki farklı yöntemle karşılaştırılmıştır. Birinci yöntemde bulanık kontrolör için kuralları önceden belirlenmiş köşegen kural tablosu [35] kullanılmış, bulanık kontrolörün seçilen bazı ölçekleme çarpanları genetik algoritmalar tarafından ayarlanmıştır. Kullanılan diğer

yöntemde ise katsayıları genetik algoritmalar tarafından ayarlanan klasik-PID kontrolör kullanılmıştır.

Sistem 1: $T(s)=\frac{e^{-0.5s}}{s^2+3s+2}$ (İkinci dereceden lineer, zaman gecikmeli sistem)



Şekil 4.3: Birinci Sistem için Simulink modeli

Kullanılan kontrolör bulanık PID yapısındadır. Çıkışta kullanılan 'saturation' bloğu kontrol işaretinin [-10, +10] aralığında kalmasını sağlar.

Durma koşulu: programın 100 nesil boyunca çalışması

$$\text{FieldD} = \begin{bmatrix} 2 & 2 & 2 & 6 & 6 & 6 & 6 & 6 & 6 & 1 & 1 & 1 & 1 & 1 & 1 & 10 & 10 & 13 & 13 & 8; \\ 3 & 3 & 3 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0; \\ 9 & 9 & 9 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 20 & 10 & 500 & 500 & 2*pi; \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0; \end{bmatrix}$$

GGAP= 0.9

Amaç fonksiyonu:

0.03*((abs(s_Error))'*s_Time) + abs(max(s_Out)-1) +0.1*settlingT;

Amaç fonksiyonunda ITAE (Integral time absolute error- Hatanın mutlak değerinin zaman değeriyle çarpımının integrali), maksimum aşım ve yerleşme zamanı değerleri belirli oranlarda kullanılmıştır.

Uygunluk ataması: RANKING

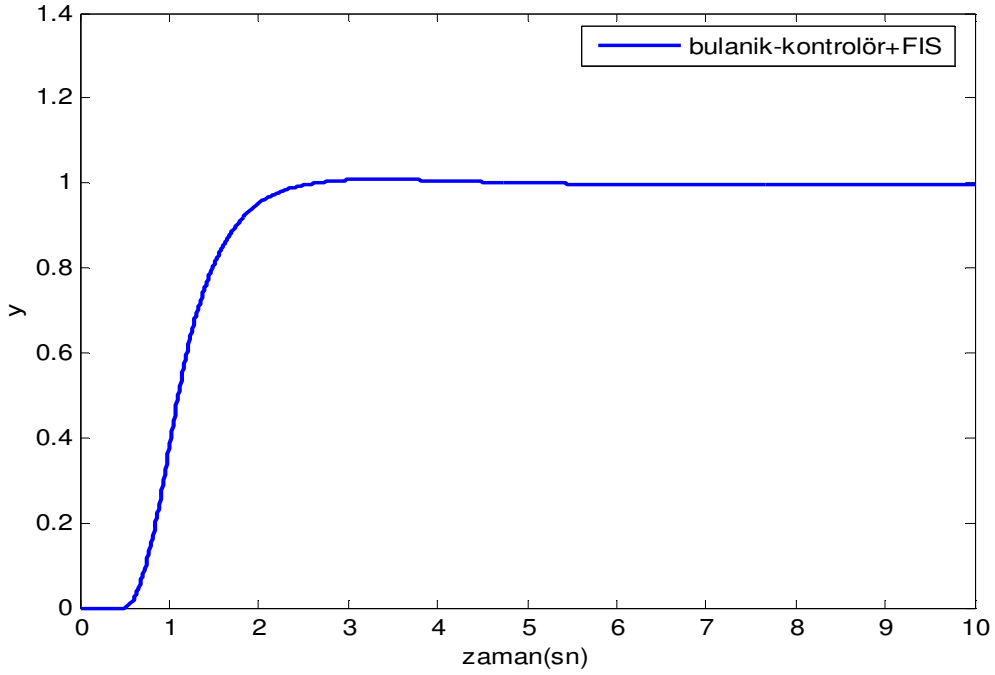
Seçim metodu: SUS (Stochastic Universal Sampling)

Çaprazlama metodu: XOVSP (Tek kesim noktalı çaprazlama P_c=0.7)

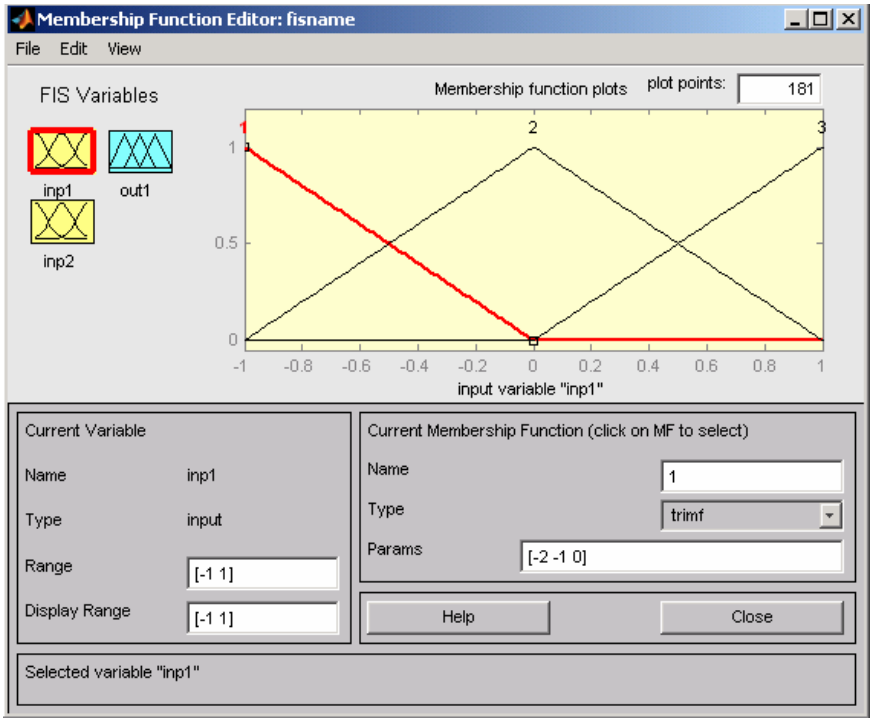
Mutasyon metodu: MUT(İkilik tabanda mutasyon P_m= 0.7/ L_{ind})

Yer değiştirme metodu: Üretilen bütün bireyler yeni popülasyona dahil edilmekte ve yer değiştirme, uygunluk değeri düşük olan bireylerden başlanılarak yapılmaktadır.

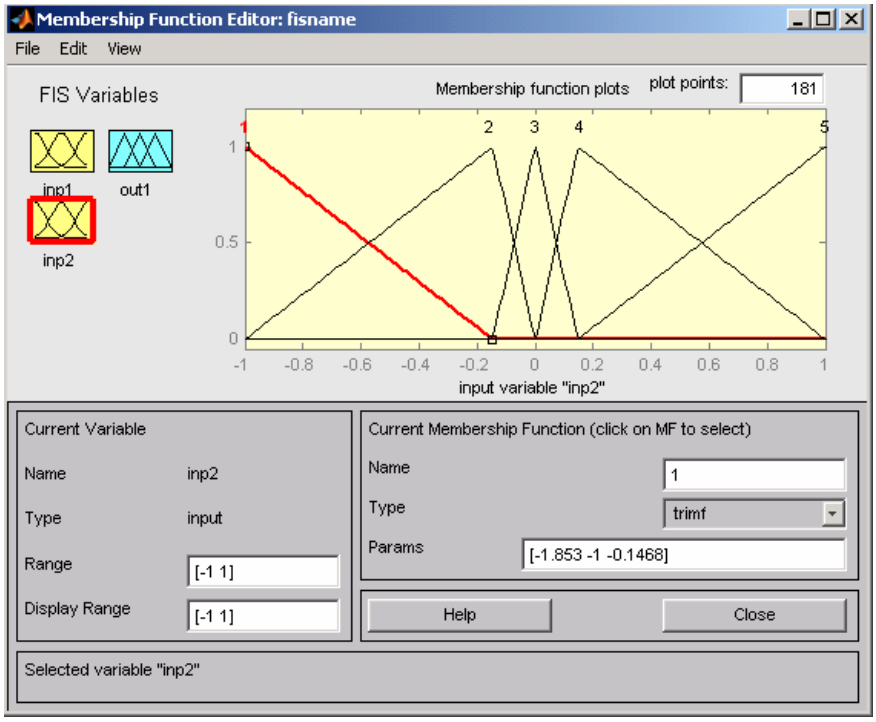
Bu yöntem kullanılarak yapılan simülasyon sonucunda elde edilen çıkış Şekil 4.4'teki gibi olmaktadır. Bulunan en iyi bireydeki üyelik fonksiyonları ve kuralların oluşturduğu yüzey görünümü sırasıyla Şekil 4.5, 4.6, 4.7 ve 4.8'de gösterilmiştir.



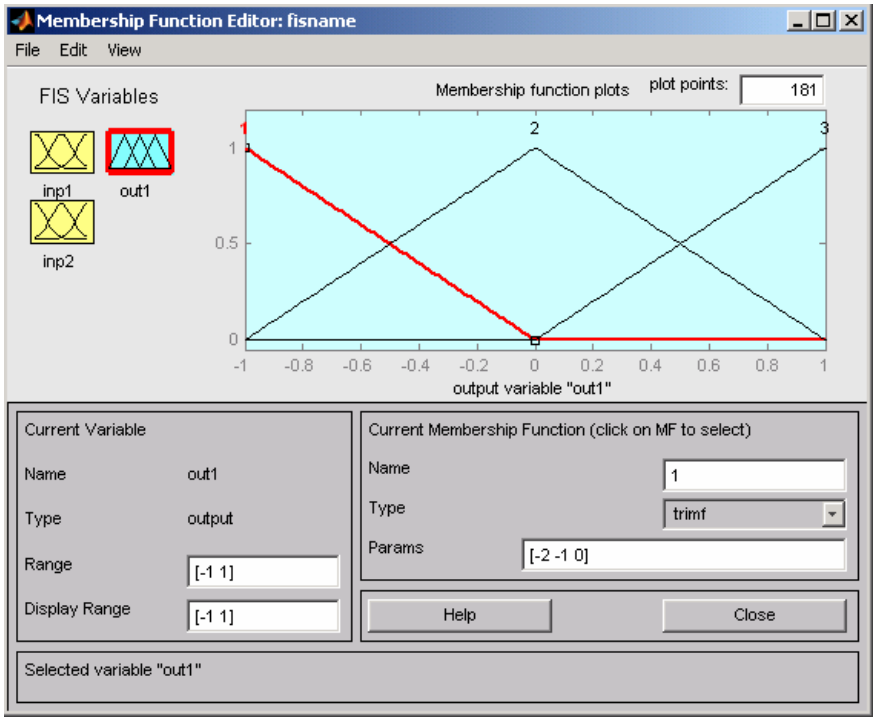
Şekil 4.4: Sistem çıkışı



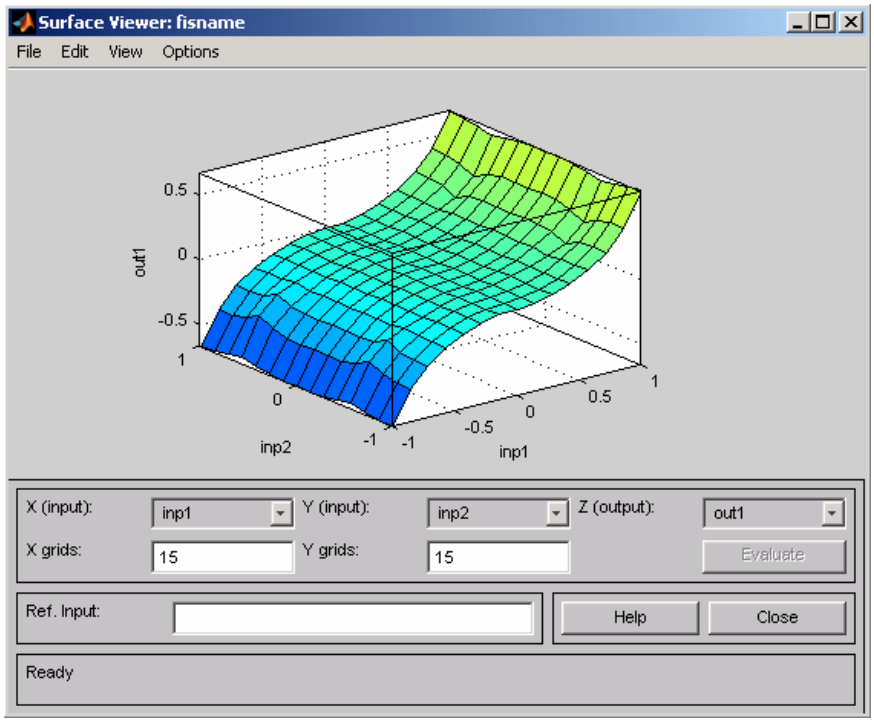
Şekil 4.5: Birinci giriş için üyelik fonksiyonları



Şekil 4.6: İkinci giriş için üyelik fonksiyonları



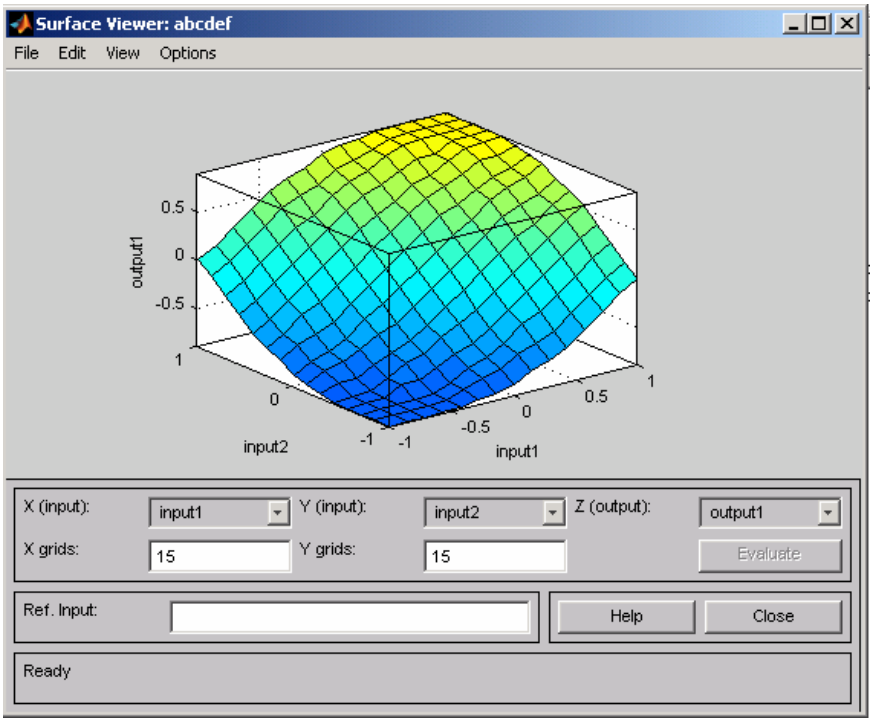
Şekil 4.7:Çıkış üyelik fonksiyonları



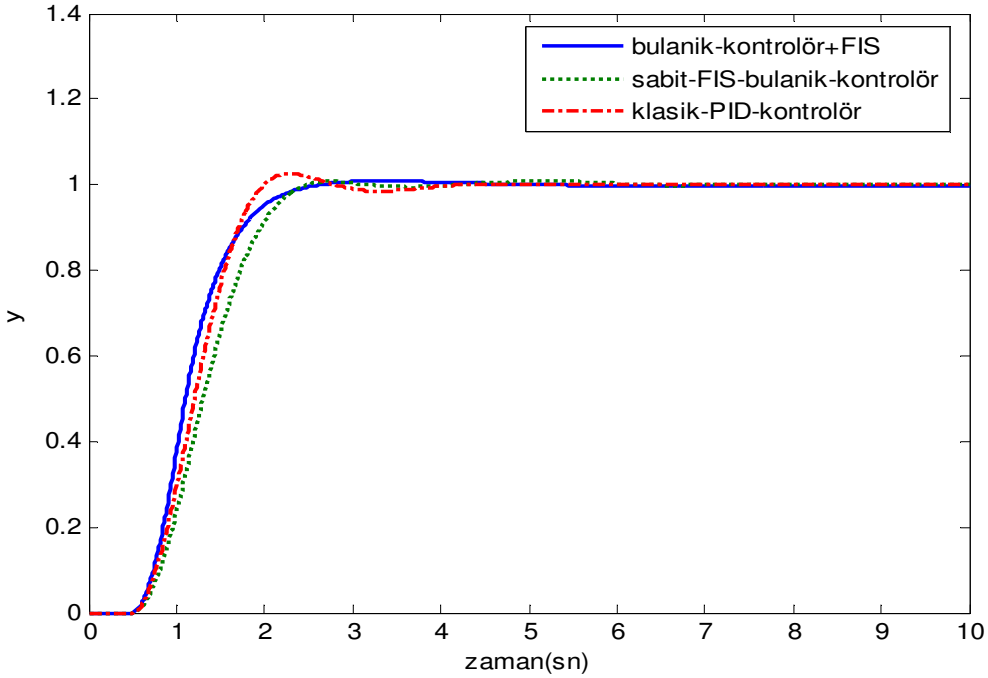
Şekil 4.8: Kural tabanının oluşturduğu yüzey

Kullanılan ikinci yöntemde ise yine bulanık kontrolör kullanılmış fakat sadece ölçekleme çarpanlarından üçü genetik algoritmalar tarafından ayarlanmıştır. Seçilen üyelik fonksiyonları üçgen tipinde olup girişler ve çıkış için yedişer tane üyelik fonksiyonu [-1, 1] aralığında tanımlıdır. Kullanılan köşegen kural tabanının yüzey görünümü ise Şekil 4.9'daki gibidir. Zaman gecikmeli sistem için yapılan simülasyonda, hata girişine bağlı giriş ölçekleme çarpanının değeri sabit kabul edilmiş, diğer katsayılar genetik algoritmalar tarafından ayarlanmıştır.

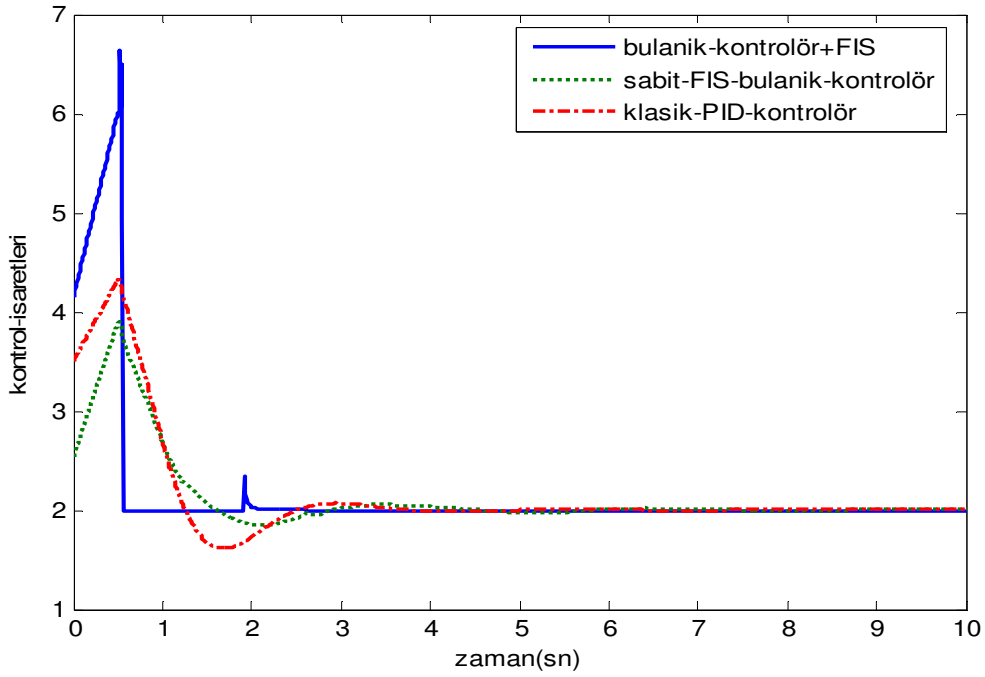
Kullanılan üçüncü yöntemde ise klasik PID kontrolörü kullanılmıştır. Genetik algoritmalar kontrolörün oransal, integral ve türev katsayılarını ayarlamaktadır. Kullanılan üç yöntem sonucunda elde edilen sistem çıkışları Şekil 4.10'da, kontrol işaretleri ise Şekil 4.11'de gösterilmiştir.



Şekil 4.9: Köşegen kural tablosu



Şekil 4.10: Zaman gecikmeli sistem için çıkışlar



Şekil 4.11: Zaman gecikmeli sistem için kontrol işaretleri

Birinci yöntem: En iyi bireyin sıra numarası: 4608 (93.nesil 8. birey)

Bulunan kontrolör katsayıları: K_1: 0.317 K_2: 9.805 K_3: 73.462 K_4: 81.852

İkinci yöntem: Birinci simülasyonda bulunan K_1=0.317 değeri sabit değer olarak girilmiş ve K_2, K_3, K_4 ölçekleme çarpanlarının en uygun değerleri genetik algoritmalar tarafından aranmıştır.

En iyi bireyin sıra numarası: 4870 (98.nesil 20. birey)

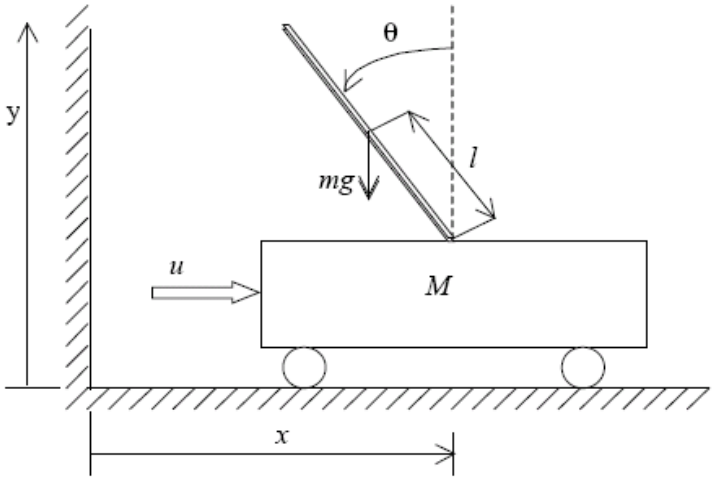
Bulunan katsayılar: K_1: 0.317 K_2: 0.179 K_3: 8.636 K_4: 8.175

Üçüncü yöntem:En iyi bireyin sıra numarası: 4062 (82. nesil 12. birey)

Bulunan katsayılar: K_1=3.506 K_2=1.388 K_3=1.642

Simülasyon sonuçları incelendiğinde, kullanılan ilk yöntemin daha az aşımlı bir sistem cevabı verdiği ve yerleşme zamanının diğerlerine göre daha kısa olduğu görülmektedir. Performans ölçütü olarak ITSE (Integral Time Square Error) alınırsa sırasıyla ITSE1: 49.74, ITSE2: 56.08, ITSE3: 65.83 değerleri elde edilmektedir. Bu veriden de anlaşıldığı gibi ilk yöntem diğerlerine göre daha başarılı bir kontrol sağlamaktadır. Fakat ilk yöntemde diğerlerine göre aratılacak daha fazla parametre olduğundan işlem süresi diğerlerine göre daha fazla olmaktadır.

Sistem 2: Ters Sarkaç Sistemi

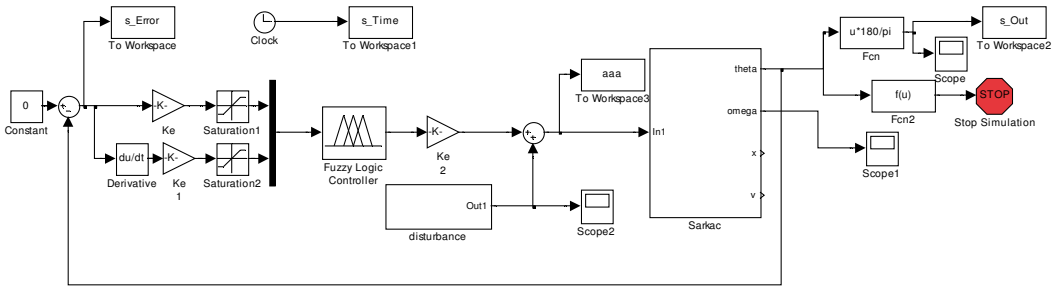


Şekil 4.12: Ters sarkaç sistemi

$$\ddot{\theta} = \frac{(M+m)g\sin\theta + \cos\theta\, u - ml\dot{\theta}^2 \cos\theta \sin\theta}{\frac{4}{3}(M+m)l - ml\cos^2\theta} \tag{4.1}$$

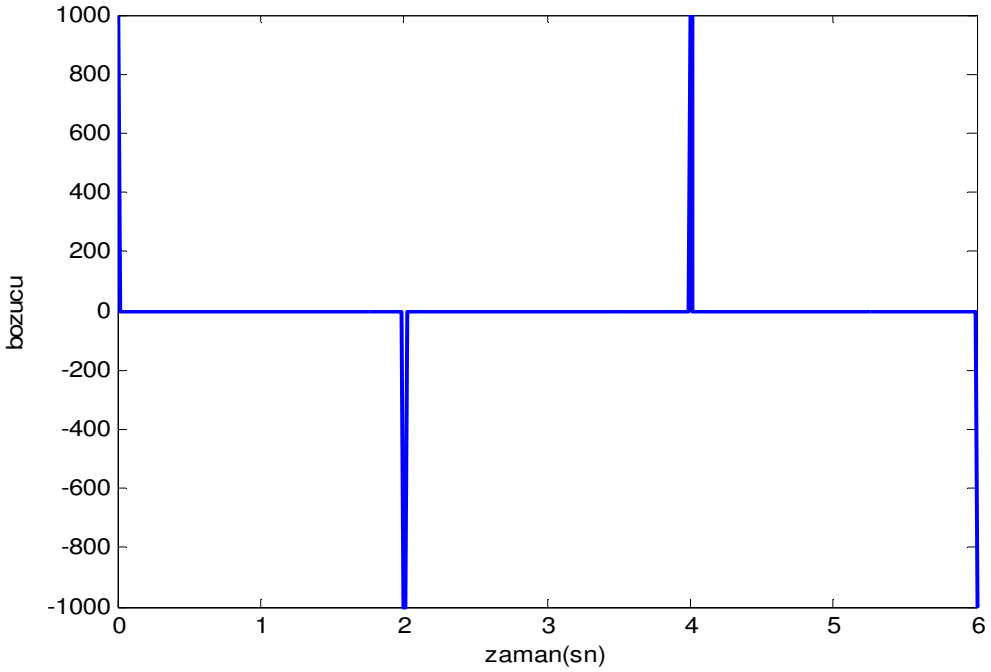
M=Arabanın ağırlığı: Simulasyonda 1 kg alınmıştır
m= Sarkacın ağırlığı: Simulasyonda 0.1 kg alınmıştır.
l=eksenle sarkacın ağırlık merkezi arasındaki mesafe: Simulasyonda 0.5 mt alınmıştır.

$$\ddot{\theta} = \frac{(11\sin\theta + \cos\theta\, u - 0.05\,\dot{\theta}^2 \cos\theta \sin\theta)}{0.733 - 0.05\cos^2\theta} \tag{4.2}$$



Şekil 4.13: İkinci Sistem için Simulink modeli

Bu modelde sarkaç açısı -90^0 veya $+90^0$ olduğunda simülasyon süresinin dolması beklenmeden simülasyon durdurulmaktadır. Girişte kullanılan 'saturation' blokları girişlerin [-1, +1] aralığında olmasını sağlamaktadır. +1000 büyüklüğünde ve her seferde yönü değişen bir bozucu(disturbance) 2 sn aralıkla ve 2 örnekleme periyodu süresince sisteme Şekil 4.14'teki gibi uygulanmaktadır.



Şekil 4.14: Bozucu etki (disturbance)

Kullanılan kontrol bulanık PD yapısındadır.

Durma koşulu: programın 100 nesil boyunca çalışması

$$\text{FieldD} = \begin{bmatrix} 2 & 2 & 2 & 7 & 7 & 7 & 7 & 7 & 7 & 1 & 1 & 1 & 1 & 1 & 1 & 10 & 10 & 17 & 11; \\ 3 & 3 & 3 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & -1 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0; \\ 9 & 9 & 9 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 20 & 10 & 10000 & 2*pi; \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0; \end{bmatrix}$$

GGAP=0.95

Amaç fonksiyonu: ((s_Error.^2)'*s_Time

Amaç fonksiyonunda ITSE (Integral Time Square Error) hatanın karesinin zaman değeriyle çarpımının integrali performans ölçütü olarak seçilmiştir.

Uygunluk değeri ataması: Obj_fun1

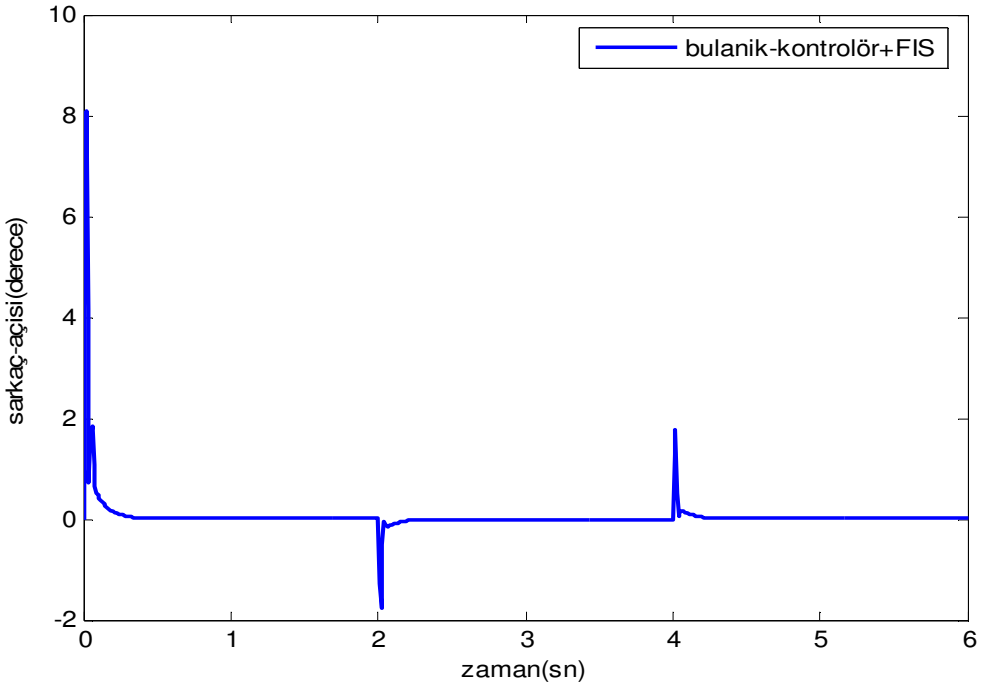
Seçim metodu: SRS (Stochastic Remainder Selection)

Çaprazlama metodu: XOVSP (Tek kesim noktalı çaprazlama P_c=0.5)

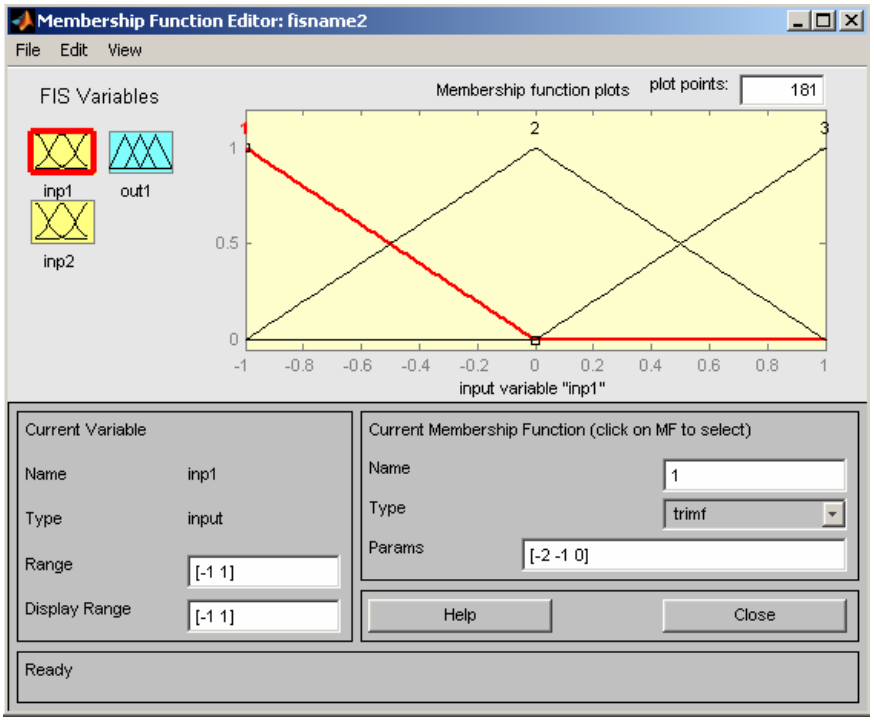
Mutasyon metodu: MUT (İkilik tabanda mutasyon P_m= 0.05)

Yer değiştirme metodu: Üretilen bütün bireyler yeni popülasyona dahil edilmekte ve yer değiştirme, uygunluk değeri düşük olan bireylerden başlanılarak yapılmaktadır.

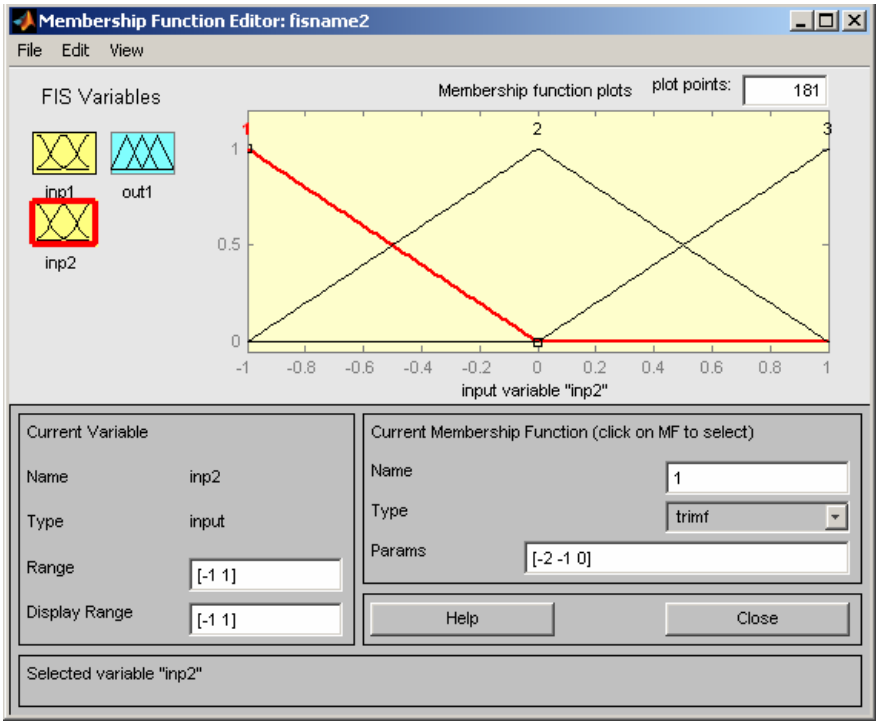
Bu yöntem kullanıldığında elde edilen sistem çıkışı Şekil 4.15'te, giriş ve çıkış üyelik fonksiyonları Şekil 4.16, 4.17 ve 4.18'de, kural tabanının oluşturduğu yüzey görünümü ise Şekil 4.19'da görülmektedir.



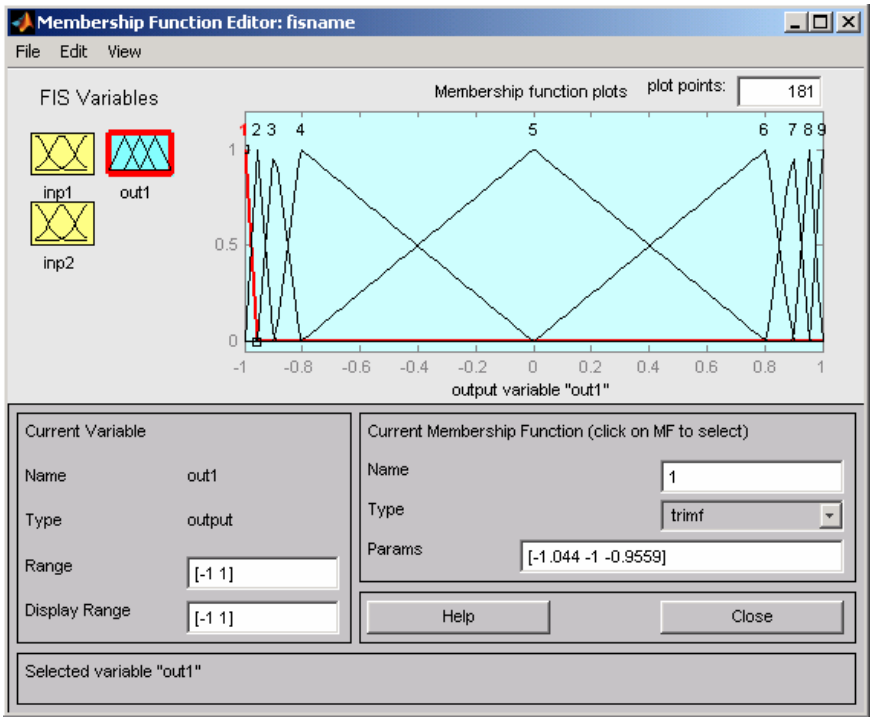
Şekil 4.15: Sistem çıkışı (Sarkaç açısı (derece))



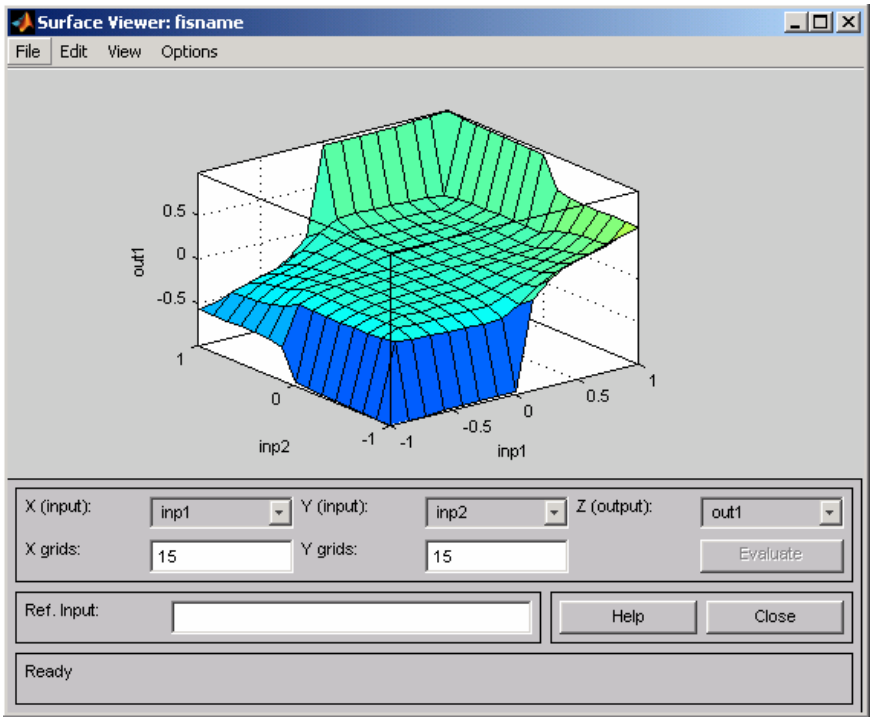
Şekil 4.16: Birinci giriş için üyelik fonksiyonları



Şekil 4.17: İkinci giriş için üyelik fonksiyonları



Şekil 4.18: Çıkış üyelik fonksiyonları



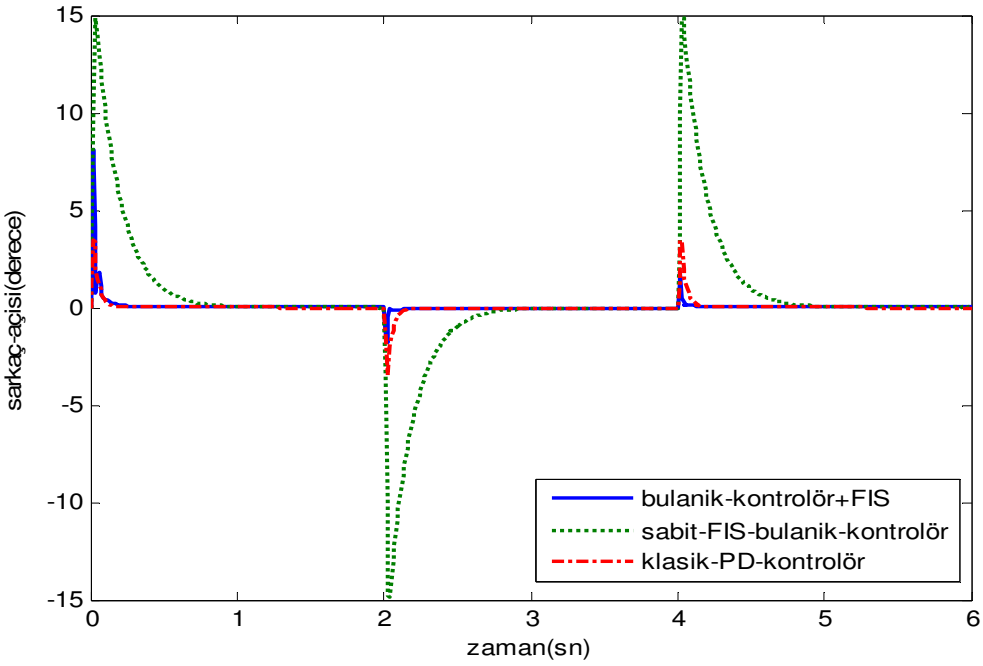
Şekil 4.19: Kural tabanının oluşturduğu yüzey

Sabit kural tablolu bulanık kontrolör yapısında ise girişler ve çıkış için yedişer tane gauss tipi üyelik fonksiyonları kullanılmış, giriş ölçekleme çarpanları sabit kabul edilerek sadece çıkış ölçekleme çarpanının en uygun değeri aratılmıştır. Kullanılan diğer kontrolör ise klasik-PD kontrolörüdür. Genetik algoritmalar, kontrolörün oransal ve türev katsayıları ayarlamakta kullanılmaktadır. Kullanılan üç yöntem sonucunda elde edilen sistem çıkışları Şekil 4.20'da, kontrol işaretleri ise Şekil 4.21, 4.22 ve 4.23'te gösterilmiştir.

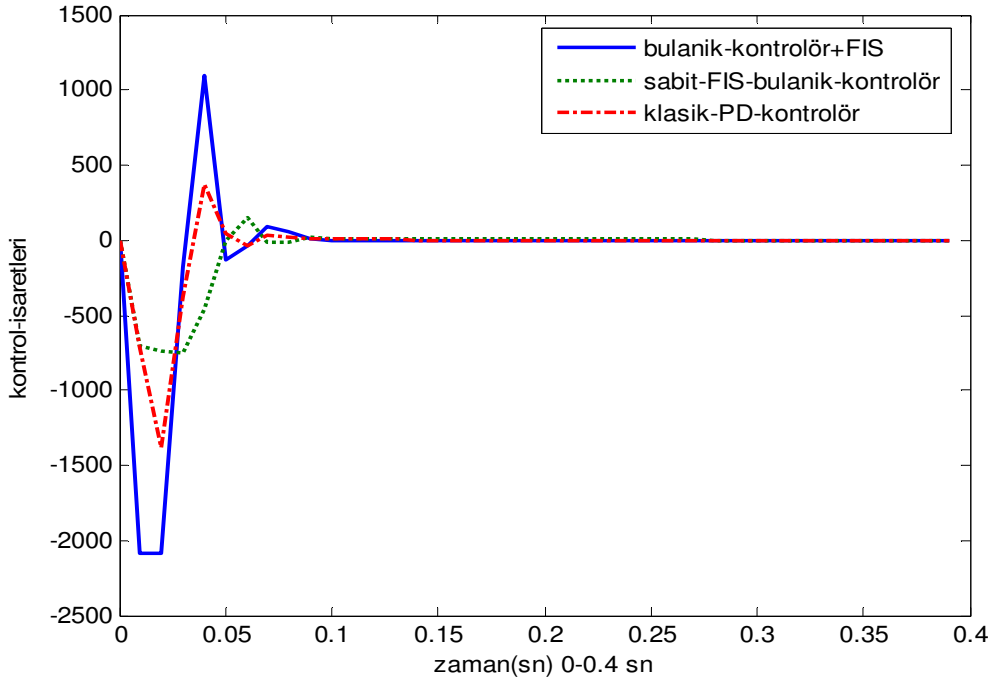
Birinci yöntem: En iyi bireyin sıra numarası: 4718 (95.nesil 18. birey)
Bulunan ölçekleme çarpanları: K_1: 2.151 K_2: 0.362 K_3: 2187.3

İkinci yöntem: Birinci simülasyonda bulunan K_1 ve K_2 değerleri sabit değer olarak girilmiş ve K_3 çıkış ölçekleme çarpanının en uygun değeri genetik algoritmalar tarafından aranmıştır.
En iyi bireyin sıra numarası: 4114 (83.nesil 14. birey)
Bulunan katsayılar: K_1: 2.151 K_2: 0.362 K_3: 850.27

Üçüncü yöntem: En iyi bireyin sıra numarası: 4582 (92. nesil 32. birey)
Bulunan katsayılar: K_1=6458 K_2=248.44

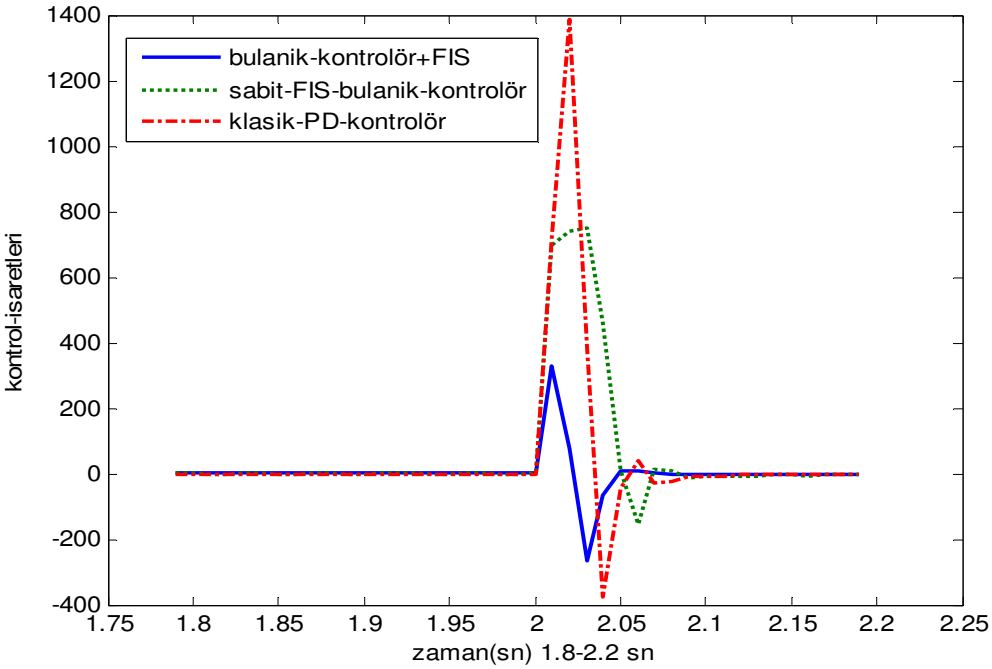


Şekil 4.20: Ters sarkaç sistemi için çıkışlar

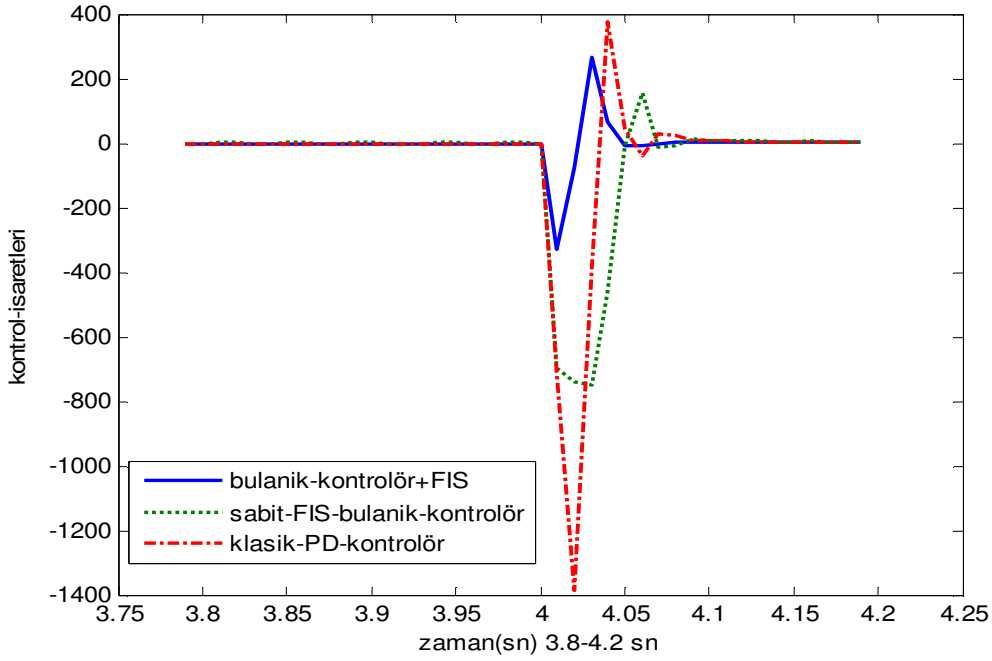


Şekil 4.21: Kontrol işaretleri (0 – 0.4 saniye arası)

Kontrol işaretlerinin değişimini daha net görebilmek için çizimler, sadece bozucunun uygulandığı anlarda 0.4 sn süresince çizdirilmiştir.



Şekil 4.22: Kontrol işaretleri (1.8 – 2.2 saniye arası)



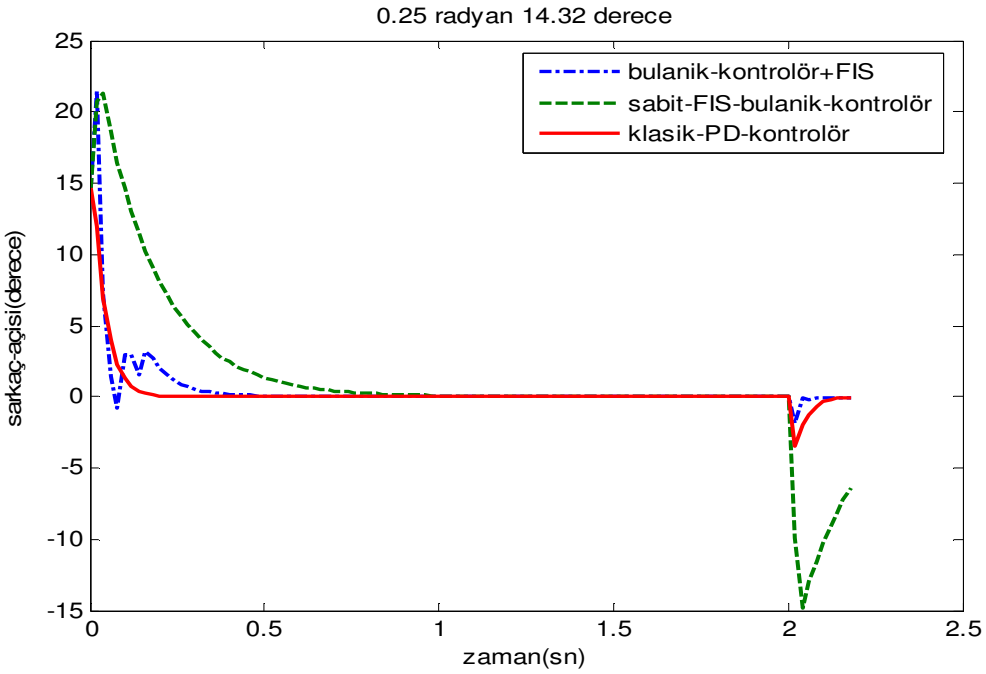
Şekil 4.23: Kontrol işaretleri (3.8 – 4.2 saniye arası)

Simülasyon sonuçları incelendiğinde, kullanılan ilk yöntemin diğer iki yönteme göre daha başarılı olduğu görülmektedir. ITSE değeri performans ölçütü olarak alınırsa ITSE1: 0.0106, ITSE2: 4.3548, ITSE3: 0.0649 verileri alınmıştır. Bu değer genetik

algoritmanın amaç fonksiyonu değeridir ve minimizasyon problemlerinde amaç fonksiyon değeri en düşük olan bireyler en başarılı olan bireylerdir. Fakat, bu yöntemin dezavantajı olarak aratılacak parametre sayısı diğerlerine göre daha fazla olduğundan programın sonuç üretmesi gereken süre diğerlerine göre daha fazladır.

Başlangıçta denge durumunda olan sarkaç için tasarlanan kontrolörlerin sarkacın bir başlangıç açısına sahip olması durumunda başarımlarını görmek için sisteme uygulanan bozucu ve denge durumundaki sarkaç için elde edilen kontrolör katsayıları değiştirilmeden, sarkacın farklı başlangıç açıları için simülasyonlar yapılmıştır. Bu bölümde 0.25, -0.25, 1.5 ve -1.5 radyan için yapılan simülasyonların sonuçları verilmiştir. Yapılan simülasyonların tamamı EK1'de verilmiştir.

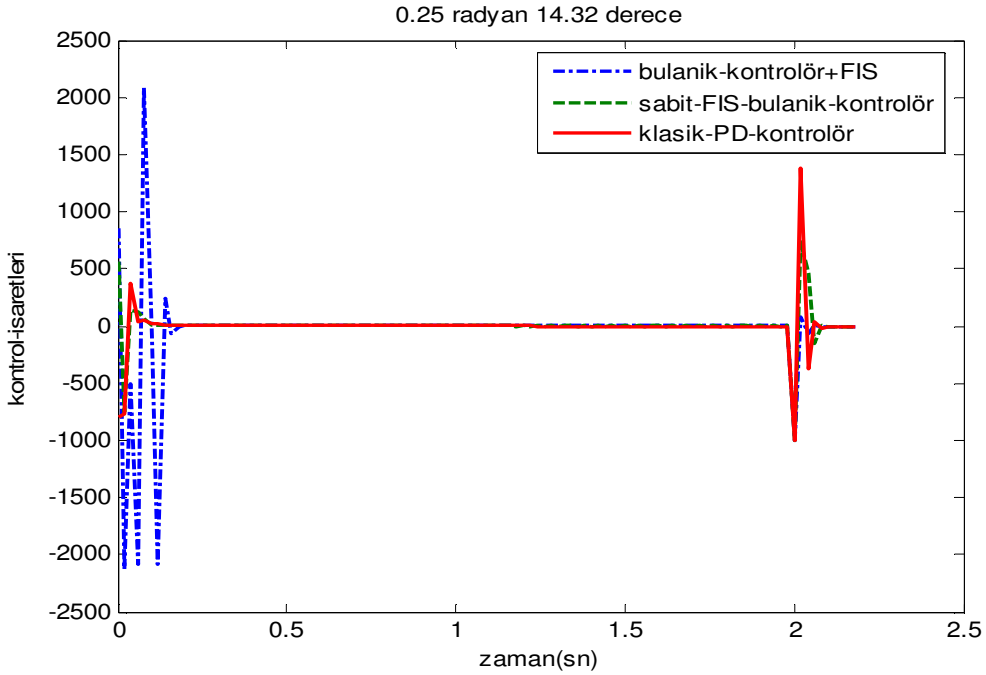
0.25 radyan ≈ 14.32 derece



Şekil 4.24: 0.25 radyan başlangıç açısı için sistem çıkışları

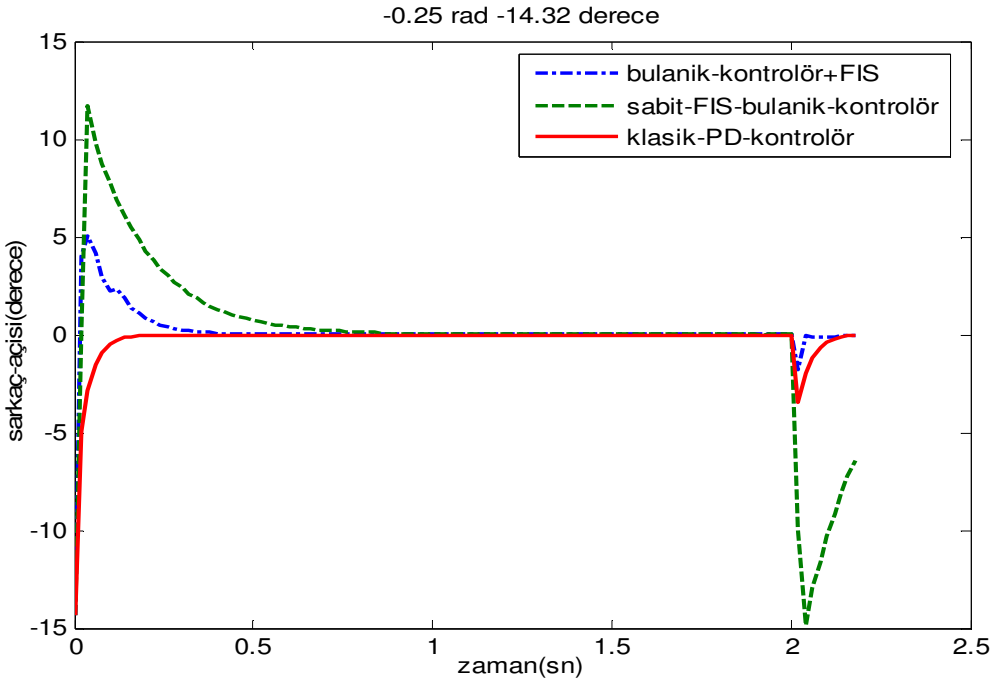
Simülasyon sonucundan da görüldüğü gibi, önerilen ilk yöntem olan hem bulanık kontrolör kural tablosunun hem de ölçekleme faktörlerinin genetik algoritmalar tarafından ayarlanması esasına dayanan yöntem bu başlangıç açısı için en salınımlı cevabı vermektedir. En başarılı kontrolör ise 0.2 sn yerleşme zamanına sahip olan klasik PD kontrolörüdür. Sabit kural tablolu bulanık kontrolör ise klasik PD kontrolöre göre daha yavaş bir cevap vermekte ve bu kontrolörün kullanıldığı

durumda, sonraki bozucu darbesinde sarkaç denge noktasından 15^{o} civarında saparak diğer kontrolörlere göre daha kötü bir performans göstermektedir. -0.25 radyan başlangıç açısı için ise yine klasik PD kontrolörü en başarılı kontrolü sağlamaktadır.

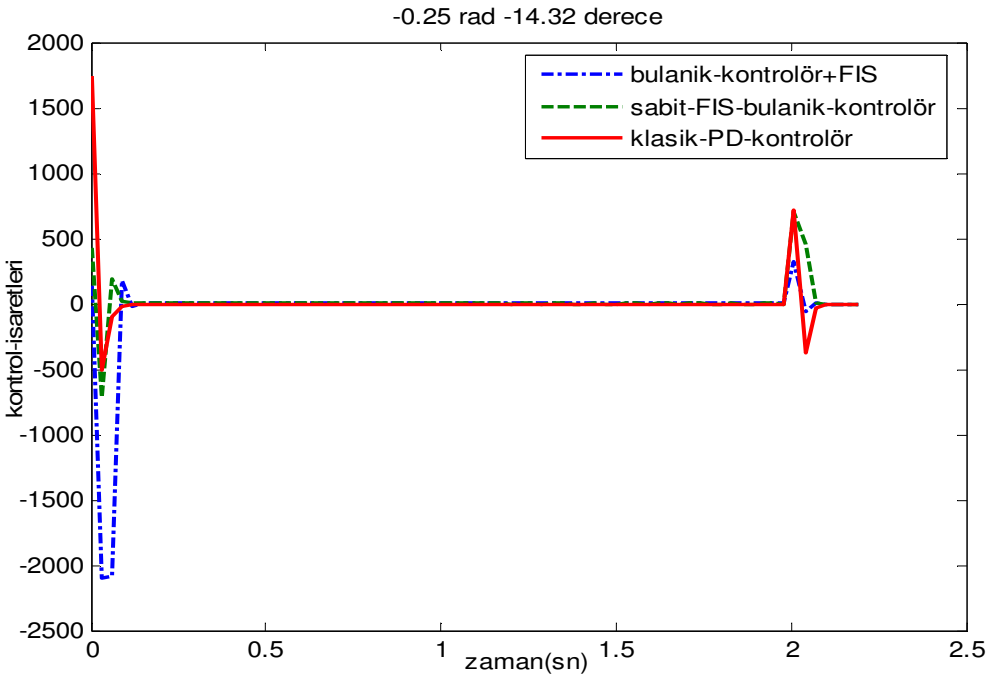


Şekil 4.25: 0.25 radyan başlangıç açısı için kontrol işaretleri

-0.25 radyan ≈ -14.32 derece

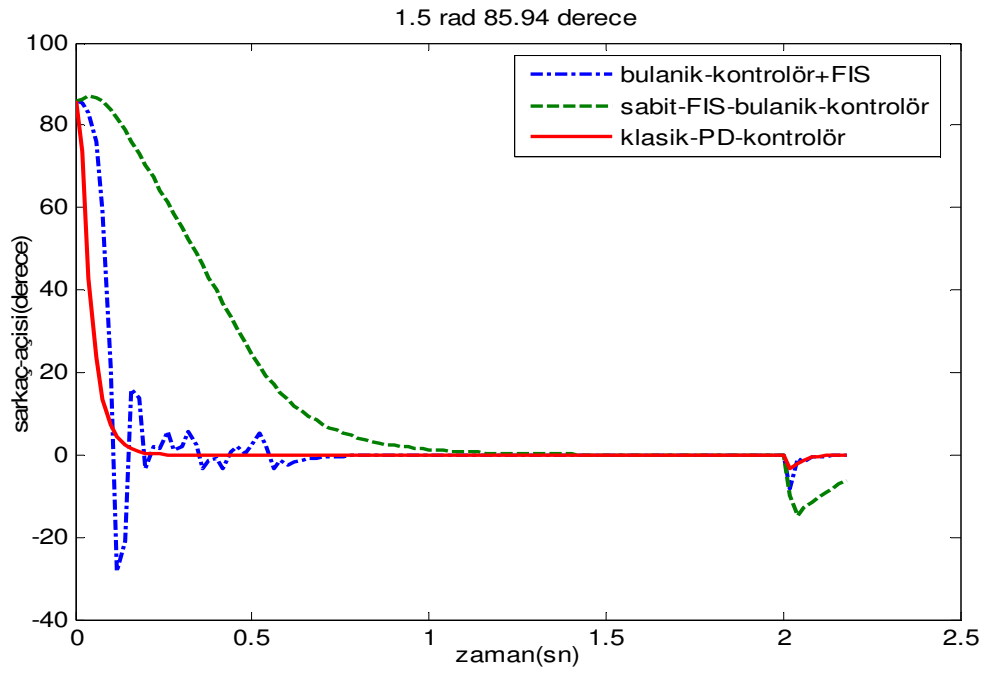


Şekil 4.26: -0.25 radyan başlangıç açısı için sistem çıkışları



Şekil 4.27: -0.25 radyan başlangıç açısı için kontrol işaretleri

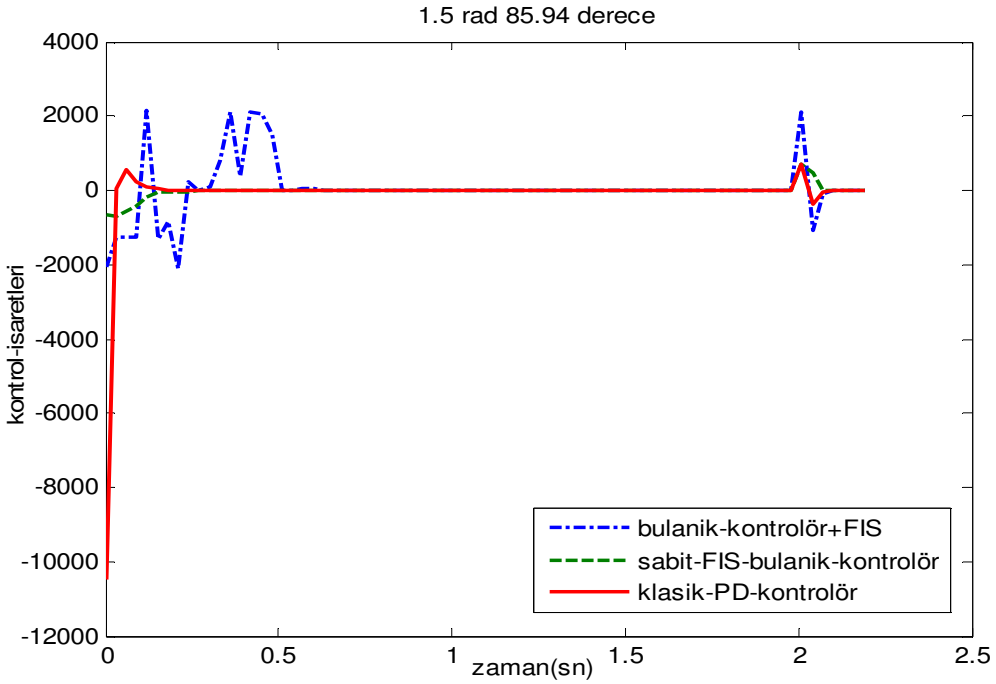
1.5 radyan ≈ 85.94 derece



Şekil 4.28: 1.5 radyan başlangıç açısı için sistem çıkışları

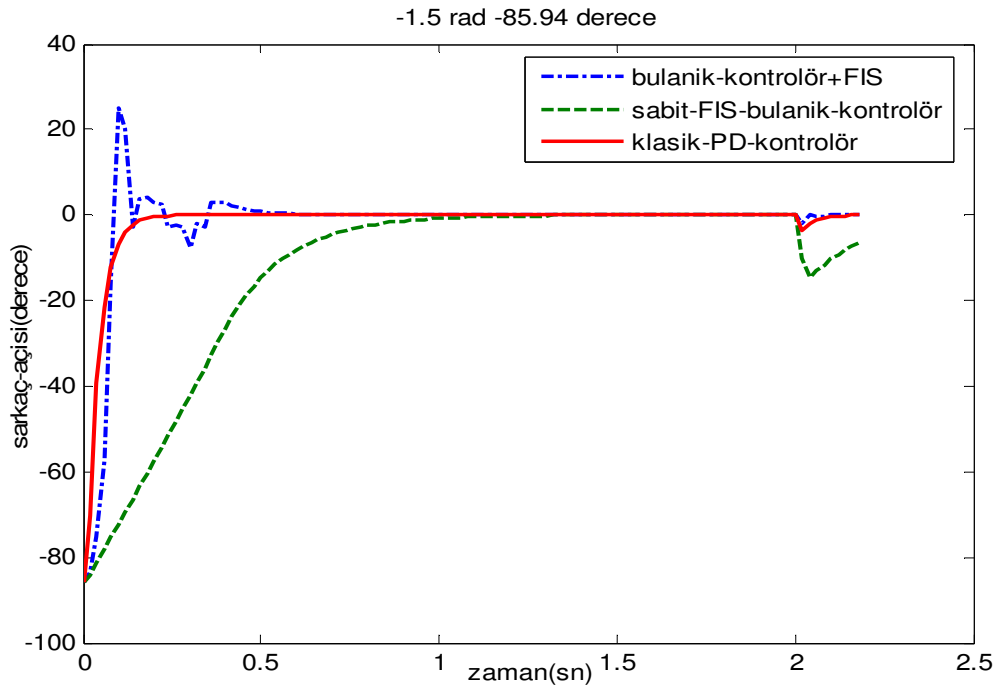
1.5 radyan başlangıç açısı için yine klasik kontrolörün en düzgün ve yerleşme zamanı en düşük olan kontrolü sağladığı görülmektedir. Fakat hem 1.5 radyan hem de -1.5

radyan başlangıç açısının olduğu durumlardaki kontrol işaretlerinin ± 10000 seviyelerine ulaştığı görülmektedir.

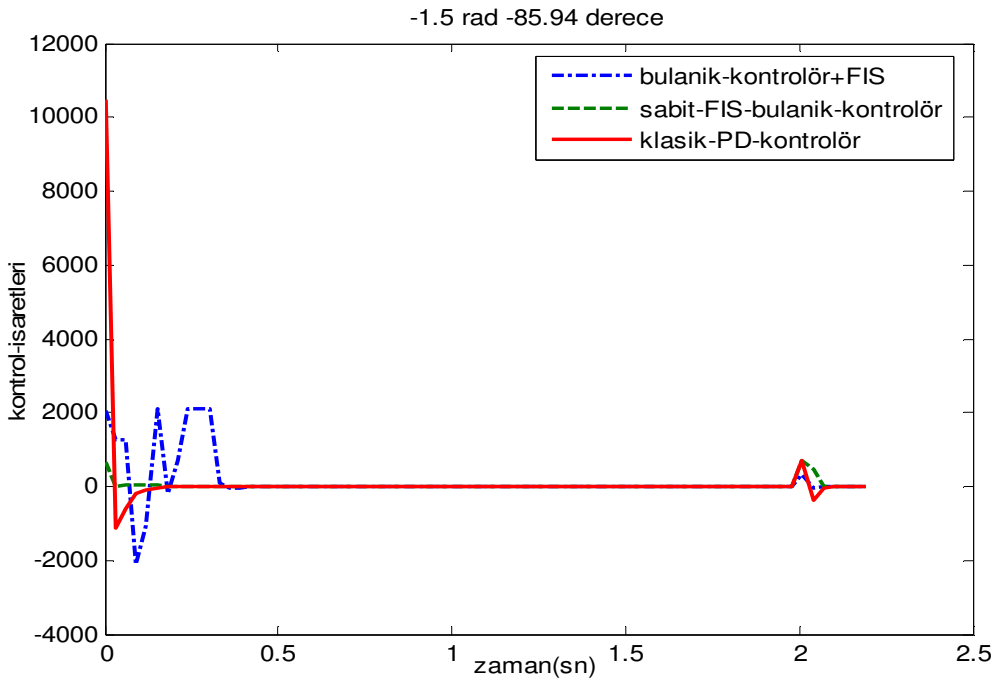


Şekil 4.29: 1.5 radyan başlangıç açısı için kontrol işaretleri

-1.5 radyan ≈ -85.94 derece



Şekil 4.30: -1.5 radyan başlangıç açısı için sistem çıkışları



Şekil 4.31: -1.5 radyan başlangıç açısı için kontrol işaretleri

Simülasyonlar, kontrolörlerin başlangıç açısına verdiği cevabı incelemek için 1.9 sn ve ikinci saniyede uygulanan bozucunun da etkisini hesaba katmak için 2.2 sn çalıştırılarak ITSE değerleri Tablo 4.1'deki gibi elde edilmiştir. Yapılan simülasyon sonuçları karşılaştırılacak olursa, klasik PD kontrolörün diğerlerine göre daha başarılı olduğu gözükmektedir. Fakat bu kontrolör kullanılacak olursa özellikle yüksek başlangıç açısının olduğu durumlarda çok büyük kontrol işaretlerine ihtiyaç duyulur. Örneğin -1.5 radyan ≈ -85.94 derece ilk açının olduğu durumda +10000 büyüklüğünü geçen bir kontrol işaretine ihtiyaç duyulur ki böyle bir büyüklükte bir işareti elde etmek her zaman mümkün olmayabilir ve de ayrıca tercih edilmez. Kural tabanı ve üyelik fonksiyonlarının sabit olduğu, sadece ölçekleme çarpanlarının genetik algoritmalar tarafından ayarlandığı bulanık kontrolör ise diğerlerine göre daha yavaş olmasına rağmen bütün ilk açı değerleri için salınım yapmadan sarkacı kontrol etmektedir fakat bu kontrolörde de bozucunun uygulandığı diğer anlarda sarkaç denge noktasından diğer kontrolörlerdeki durumlara göre daha fazla sapmaktadır.

Çalışmada önerilen hem ölçekleme çarpanlarının hem de kural tabanının genetik algoritmalar tarafından ayarlandığı kontrolörün ise bazı ilk açılarda ideal kontrolör performansından uzaklaştığı, sarkacın beklenenden fazla salınım yapmasından

anlaşılmaktadır. Fakat genel olarak sistem cevabı hızlıdır ve de kontrol işaretleri çok büyük değerlere çıkmamaktadır.

Tablo 4.1: Başlangıç açısına sahip sarkaç için kontrolörlerin ITSE değerleri

		0 - 1.9 sn arası	0 - 2.2 sn arası
0.14 radyan	bulanık-kontrolör+FIS	0,0046	0,0078
	sabit-FIS-bulanık-kontrolör	0,1139	1,3644
	klasik-PD-kontrolör	0,0018	0,0235
-0.14 radyan	bulanık-kontrolör+FIS	0,0063	0,0738
	sabit-FIS-bulanık-kontrolör	0,0449	1,2958
	klasik-PD-kontrolör	0,0009	0,0218
0.25 radyan	bulanık-kontrolör+FIS	0,0116	0,0149
	sabit-FIS-bulanık-kontrolör	0,1501	1,4010
	klasik-PD-kontrolör	0,0041	0,0258
-0.25 radyan	bulanık-kontrolör+FIS	0,0045	0,0078
	sabit-FIS-bulanık-kontrolör	0,0412	1,2918
	klasik-PD-kontrolör	0,0008	0,0225
0.55 radyan	bulanık-kontrolör+FIS	0,0163	0,0837
	sabit-FIS-bulanık-kontrolör	0,3519	1,6025
	klasik-PD-kontrolör	0,0144	0,0361
-0.55 radyan	bulanık-kontrolör+FIS	0,0192	0,0224
	sabit-FIS-bulanık-kontrolör	0,0468	1,2976
	klasik-PD-kontrolör	0,0073	0,0290
0.9 radyan	bulanık-kontrolör+FIS	0,0744	0,0776
	sabit-FIS-bulanık-kontrolör	1,5485	2,7992
	klasik-PD-kontrolör	0,0355	0,0572
-0.9 radyan	bulanık-kontrolör+FIS	0,1260	0,1934
	sabit-FIS-bulanık-kontrolör	0,0246	1,2769
	klasik-PD-kontrolör	0,0247	0,0463
1.5 radyan	bulanık-kontrolör+FIS	0,8523	0,9198
	sabit-FIS-bulanık-kontrolör	11,4926	12,7428
	klasik-PD-kontrolör	0,1502	0,1718
-1.5 radyan	bulanık-kontrolör+FIS	0,4619	0,4651
	sabit-FIS-bulanık-kontrolör	6,7992	8,0503
	klasik-PD-kontrolör	0,1320	0,1536

Bu tasarıma yapılacak eklentiler ile bu kontrolörün değişen ilk açılara karşı daha başarılı sonuçlar üretmesini sağlayacak uyarlamalı bir yapı sağlanabilir. Bu çalışmanın bir ileriki aşaması hem farklı ilk açılara hem de değişik tipteki ve büyüklükteki bozuculara karşı ideal kontrolör performansından sapmayacak bir kontrolör tasarlamak olmalıdır.

5. SONUÇLAR VE TARTIŞMA

Bu tez çalışmasında genetik algoritmaların bulanık mantık kontrolörlerinde kullanımı incelenmiş ve iki sistem için simülasyonlar yapılmıştır. Kullanılan yöntemde, bulanık kontrolörün ölçekleme katsayıları, üyelik fonksiyonlarının en uygun değerleri ve bulanık kurallar genetik algoritma tarafından bulunmakta ve daha sonra sisteme uygulanmaktadır. Bu yöntemin başarısı klasik-PID kontrolör ve köşegen kural tablolu bulanık kontrolörle karşılaştırılmıştır.

Simülasyonların sonuçları incelendiğinde, bulanık kontrolörün ölçekleme çarpanlarının ve FIS yapısının genetik algoritmalar tarafından ayarlandığı kontrolörün diğer kontrolörlere göre daha başarılı olduğu gözlenmiştir. Fakat, bu kontrolörde aratılan parametre sayısı fazla olduğu için işlem süresi diğerlerine göre uzun olmaktadır.

Kullanılan kontrolörler off-line (çevrimdışı) olarak çalışmaktadır. Bu tasarımlara yapılacak eklentilerle on-line (çevrimiçi) çalışma sağlanabilir, kullanılan kontrolörler uyarlamalı hale getirilebilirse, gerçek zamanlı çalışma esnasında değişen sistem parametrelerine karşın kontrolörler düzgün çalışmaya devam eder.

Genetik algoritmaların işleyişi konusunda belirlenmesi gereken en önemli fonksiyonlar amaç ve uygunluk fonksiyonlarıdır. Bu fonksiyonların iyi belirlenmemesi, genetik algoritmaların istenilenden çok farklı sonuçlara ulaşılmasına neden olur. Literatürde, genetik algoritmalar için hangi durumlarda hangi seçim, çaprazlama, mutasyon çeşidinin daha başarılı sonuçlar vereceği ve çaprazlama ve mutasyon oluşma olasılığının değerinin seçimi konusunda kesin kurallar olmadığından, en uygun değerlerin tespiti için bunların farklı kombinasyonları kullanılarak simülasyonlar yapılarak en uygun seçimler yapılabilir.

Çalışmanın son kısmında sarkacın bir başlangıç açısına sahip olması durumu göz önüne alınmış ve değişik ilk açılar için simülasyonlar yapılmıştır. Çalışmada önerilen hem ölçekleme çarpanlarının hem de kural tabanının genetik algoritmalar tarafından

ayarlandığı kontrolörün bazı ilk açılarda ideal kontrolör performansından uzaklaştığı, sarkacın beklenenden fazla salınım yapmasından anlaşılmaktadır. Fakat genel olarak sistem cevabı hızlıdır ve de kontrol işaretleri çok büyük değerlere çıkmamaktadır. Bu çalışmanın bir ileriki aşaması hem farklı ilk açılara hem de değişik tipteki ve büyüklükteki bozuculara karşı ideal kontrolör performansından sapmayacak bir kontrolör tasarlamak olmalıdır.

KAYNAKLAR

[1] **Rechenberg, I.,** 1973. Evolutionsstrategie: Optimierung tecnisher Systeme nach Prinzipien der biologischen Evolution, Fromman-Holzboog Verlag, Sttutgart

[2] **Goldberg, D.E.,** 1989. Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley Publishing Company, USA

[3] **Holland, J.H.,** 1975. Adaptation in Natural and Artificial Systems, Ann Arbor, The University of Michigan Press, USA

[4] **Bäck, T.,** 1996. Evolutionary Algorithms in Theory and Practice, Oxford University Press, NJ, USA

[5] **Michalewicz, Z.,** 1992. Genetic Algorithms+Data Structures=Evolution Programs, Springer Verlag, USA

[6] **Davis, L.,** 1988. Genetic Algoritms and Simulated Annealing, Morgan Kaufmann Publishers, CL, USA

[7] **Mitchell, M.,** 1996. An Introduction to Genetic Algorithms, MIT Press, England

[8] **Banzhalf, W., Nordin, P., Robert, E.K. and Frank D.F.,** 1998. Genetic Programming, Morgan Kaufman Publishing, SF, USA

[9] **Koza, J.R.,** 1992. Genetic Programming, Bradford Book, The MIT Press, Massachusetts

[10] **Wang, Li-Xin,** 1997. A Course in Fuzzy Systems and Control, Prentice Hall

[11] **Üstoğlu, İ.,** 1999. Bulanık Kontrol Sistemlerinde Kararlılık, Yüksek Lisans Tezi, İ.T.Ü. Fen Bilimleri Enstitüsü, İstanbul

[12] **Linkens, D.A. and Nyongesa, M.,** 1995. Genetic Algorithms for Fuzzy Control Part 1: Offline systems development and applications, IEE Proc.- Control Theory Applications, 142

[13] **Linkens, D.A. and Nyongesa, M.,** 1995. Genetic Algorithms for Fuzzy Control Part 2: Online systems development and applications, IEE Proc.-Control Theory Applications, 142

[14] **Herrera, F., Lozano, M., Verdegay, J.L.,** 1995. Tuning Fuzzy Logic Controllers by Genetic Algorithms, International Journal of Approximate Reasoning,12: 299-315

[15] **Lee, M.A., Takagi,** 1993. Integrating Design Stages of Fuzzy Systems Using Genetic Algorithms, Proc. 2nd IEEE Int. Conf.- Fuzzy Systems, San Francisco, 612-617

[16] **Belarbi, K. and Titel, F.,** 2000. Genetic algorithm for the design of a class of fuzzy controllers: An alternative approach, IEEE transactions on fuzzy systems, Vol 8, Iss4, 398-405

[17] **Park, Y.J., Cho, H.S. and Cha, D.H.,** 1995. Genetic Algorithm-Based Optimization of Fuzzy Logic Controller Using Characteristic Parameters, Proceedings of the 1995 International Conference on Evolutionary Computation, 831-836

[18] **Cheong, F. and Lai, R.,** 2000. Constraining the Optimization of a Fuzzy Logic Controller Using an Enhanced Genetic Algorithm, IEEE Transactions on Systems, Man and Cybernetics- Part B: Cybernetics, Vol 30, No.1

[19] **Homaifar, A. and Mc Cormic, E.,** 1995. Simultaneous Deisgn of Membership Functions and Rule Sets for Fuzzy Controllers Using Genetic Algorithms, IEEE Transactions on Fuzzy Systems, 3, May

[20] **Shi, Y., Eberhert, R. and Chen, Y.,** 1999. Implementation of Evolutionary Fuzzy Systems, IEEE Transactions on Fuzzy Systems, 7, April

[21] **Ng, C.K. and LI. Y.,** 1994. Design of Sophisticated Fuzzy Logic Controllers Using Genetic Algorithms, Techical ReportTR-Control 931228 University Of Glasgow

[22] **Karr, C.L., and Gentry, J.E.,** 1993. Fuzzy Control of pH Using Genetic Algorithms, IEEE Transactions of Fuzzy Systems, 1, February

[23] **Mann, K.F., Tang, K.S., Kwong, S. and LiuI, Z.F.,** 1998. Minimal Fuzzy Memebership and Rules Using Hierarchical Genetic Algorithms, IEEE Transaction on Industrial Electronics, 45, February

[24**] Chipperfield, A., Fleming, P., Pohlheim, H. and Fonseca, C.,** Genetic Algorithm Toolbox Version 1.2 User's Guide, University of Sheffield

[25] **Erol, O.K.,** 1995. Bulanık kontrolör karar tablosunun genetik algoritmalarla oluşturulması, Yüksek Lisans Tezi, İ.T.Ü. Fen Bilimleri Enstitüsü, İstanbul

[26] **Eksin, İ.,** 2004. Bulanık Modelleme ve Kontrol Yüksek Lisans Ders Notları, İ.T.Ü. Kontrol ve Otomasyon mühendisliği

[27] **Valenzuela, C.L.,** 1995. Evolutionary Divide and Conquer: a novel genetic approach to the TSP, Imperial College

[28] **Holstien, R.B.,** 1971. Artificial Genetic Adaptation in Computer Control Systems, PhD Thesis, Departmant of Computer and Communication Sciences, University of Michigan, Ann Arbor

[29] **Coruano, R.A. and Schaffer, J.D.,** 1988. Represantation and Hidden Bias: Gray vs Binary Coding, Proc. Of 6th Int. Conf. Machine Learning, 153-161

[30] **Çetin, N.,** 2002. Genetik Algoritma, Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul

[31] **Topuz, V.,** 2002. Bulanık Genetik Proses Kontrolü, Doktora Tezi, Marmara Üniveristesi Fen Bilimleri Enstitüsü, İstanbul

[32] **Fogarty, T.C.,** 1989. Varying the probability of mutation in the genetic algorithm, Proceeding of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, NJ, USA

[33] **Chambers, L.,** 1995. Practical Handbook of Genetic Algorithms Volume 1, CRC Press, FL, USA

[34] **Klir, G.J., and Juan, B.,** 1993. Fuzzy Sets and Fuzzy Logic Theory and Applications, Prentice Hall

[35**] Mizumoto, M.,** 1992. Realization of PID controllers by fuzzy control methods, Fuzzy Sets and Systems,70,171-182

[36] **Zadeh, L.A.,** 1965. Fuzzy Sets, Information and Control Vol.8, No.3

[37] **Güzelkaya, M., Eksin, I. and Gürleyen, F.,** 2001. A new methodology for designing a fuzzy logic controller and PI, PD belending mechanism. Journal of Intelligent and Fuzzy Systems, 11, 85-98

[38] **Li, H.X. and Gatland, H.B.,** 1996. Conventional fuzzy control and its enhancement, IEEE Trans. On Systems, Man, Cybernetics-Part(B) 26(5) 791-797

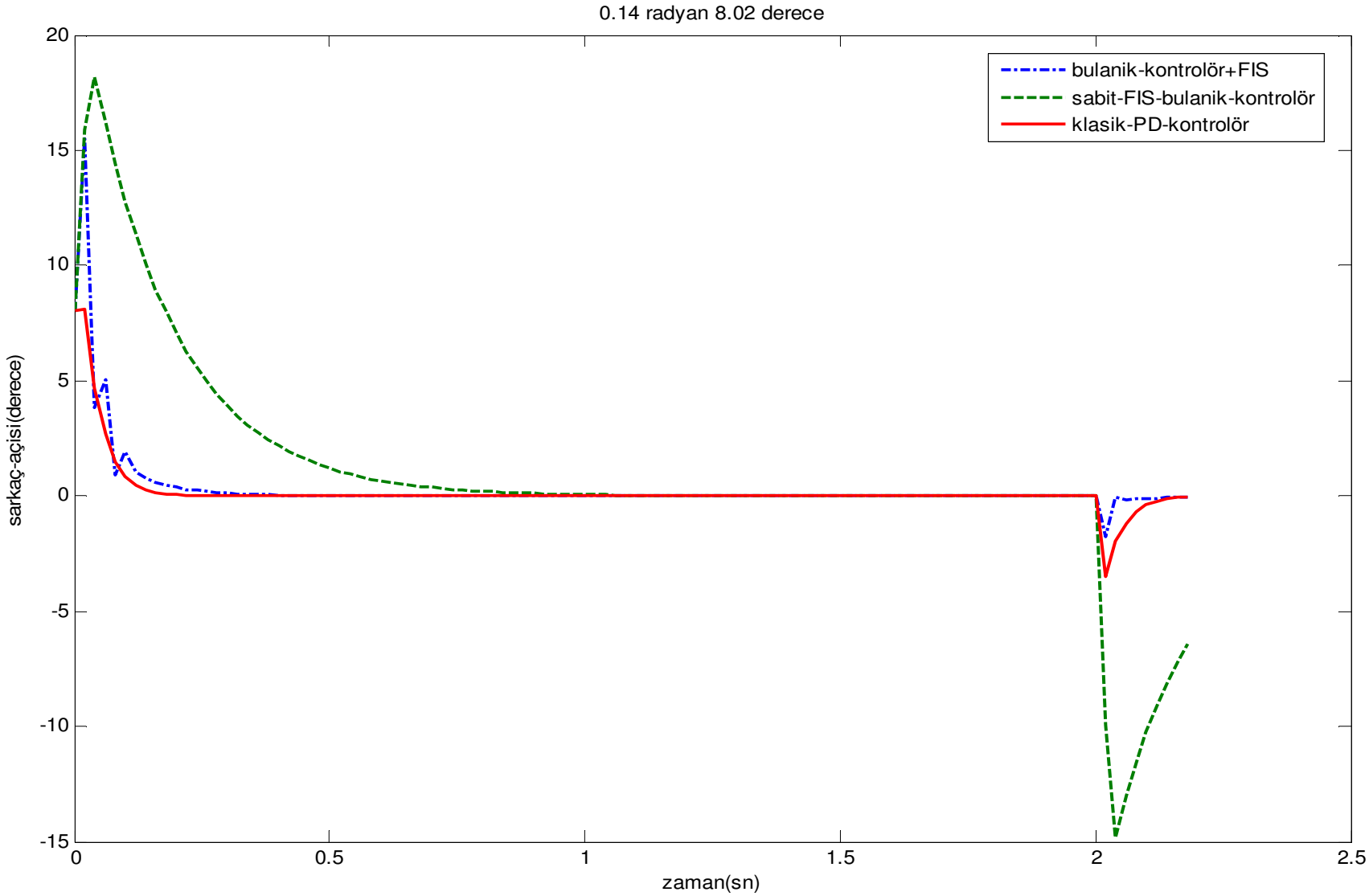
[39] **Qiao, W.Z. and Mizumoto, M.,** 1996. PID type fuzzy controller and parameters adaptive method, Fuzzy Sets and Systems,78, 23-35

[40] **Hu, B., Mann, G.K.I and Gosine, R.G.,** 1999. A new methodology for an analytical and optimal design of fuzzy PID controller, IEEE Trans. On Fuzzy Systems, 7(5), 521-539

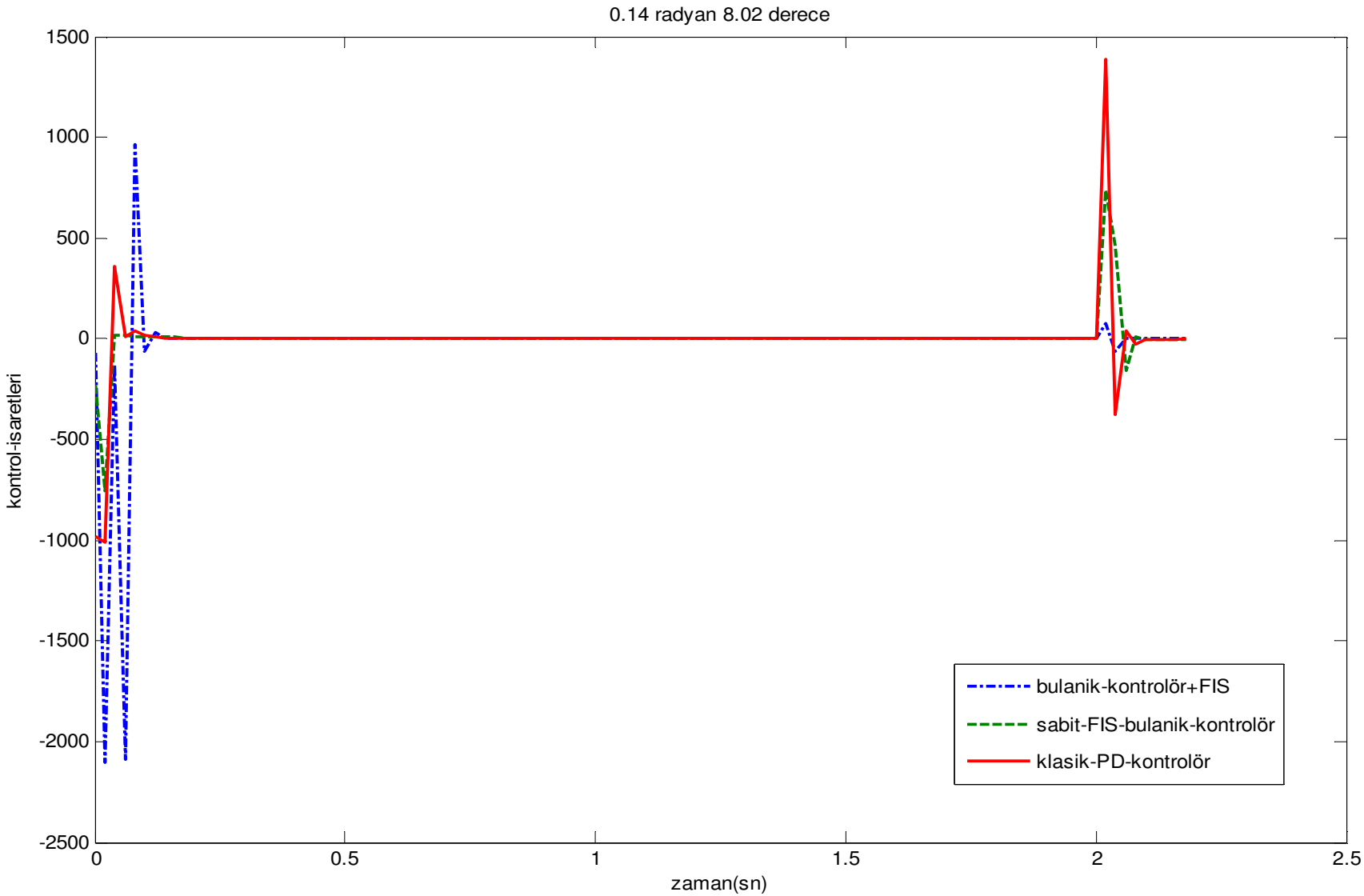
EKLER

EK 1: Başlangıçta denge durumunda olmayan sarkaç için yapılan simülasyonlar

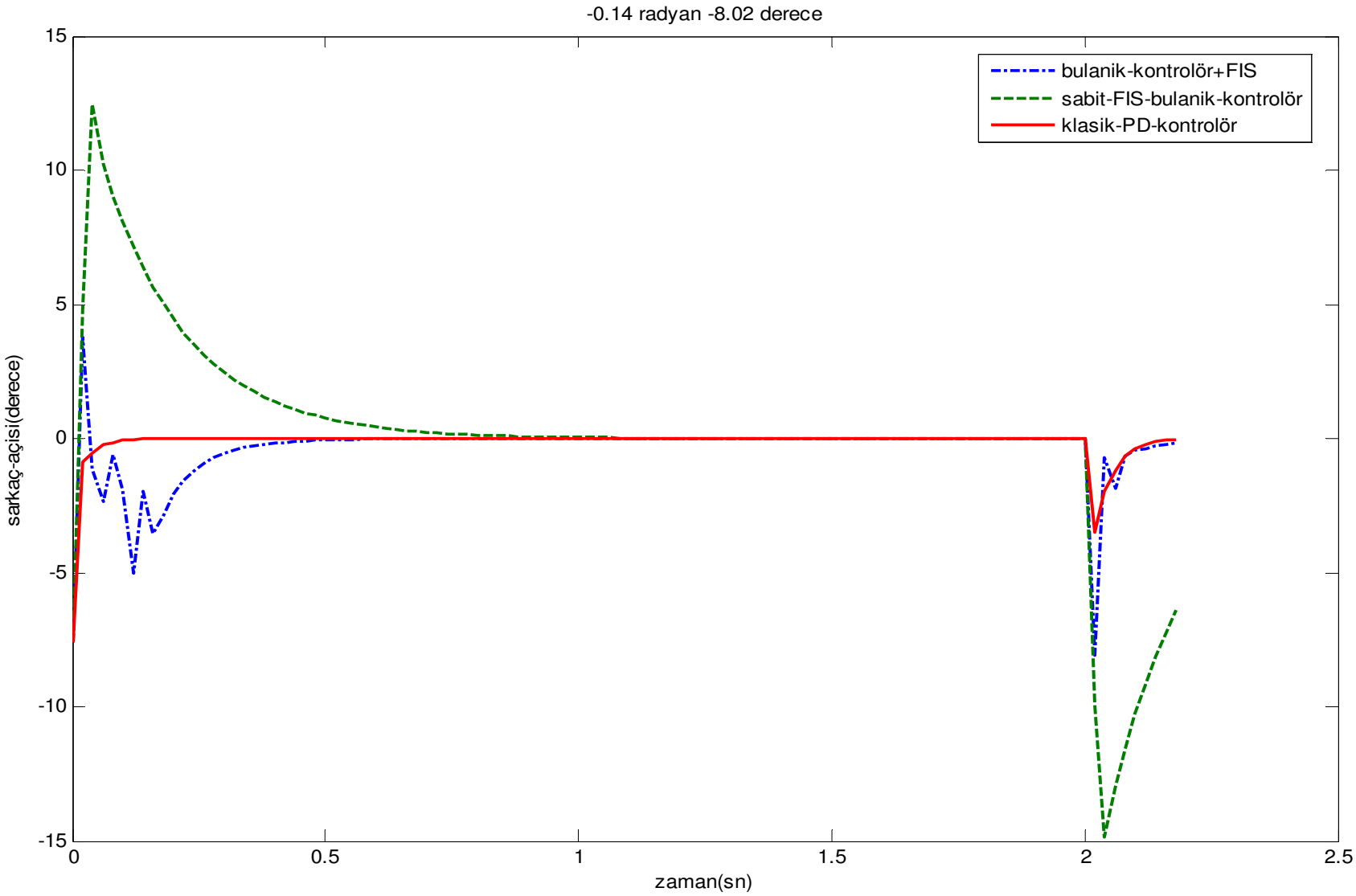
EK 2: Bilgisayar programı disketle beraber verilmiştir.



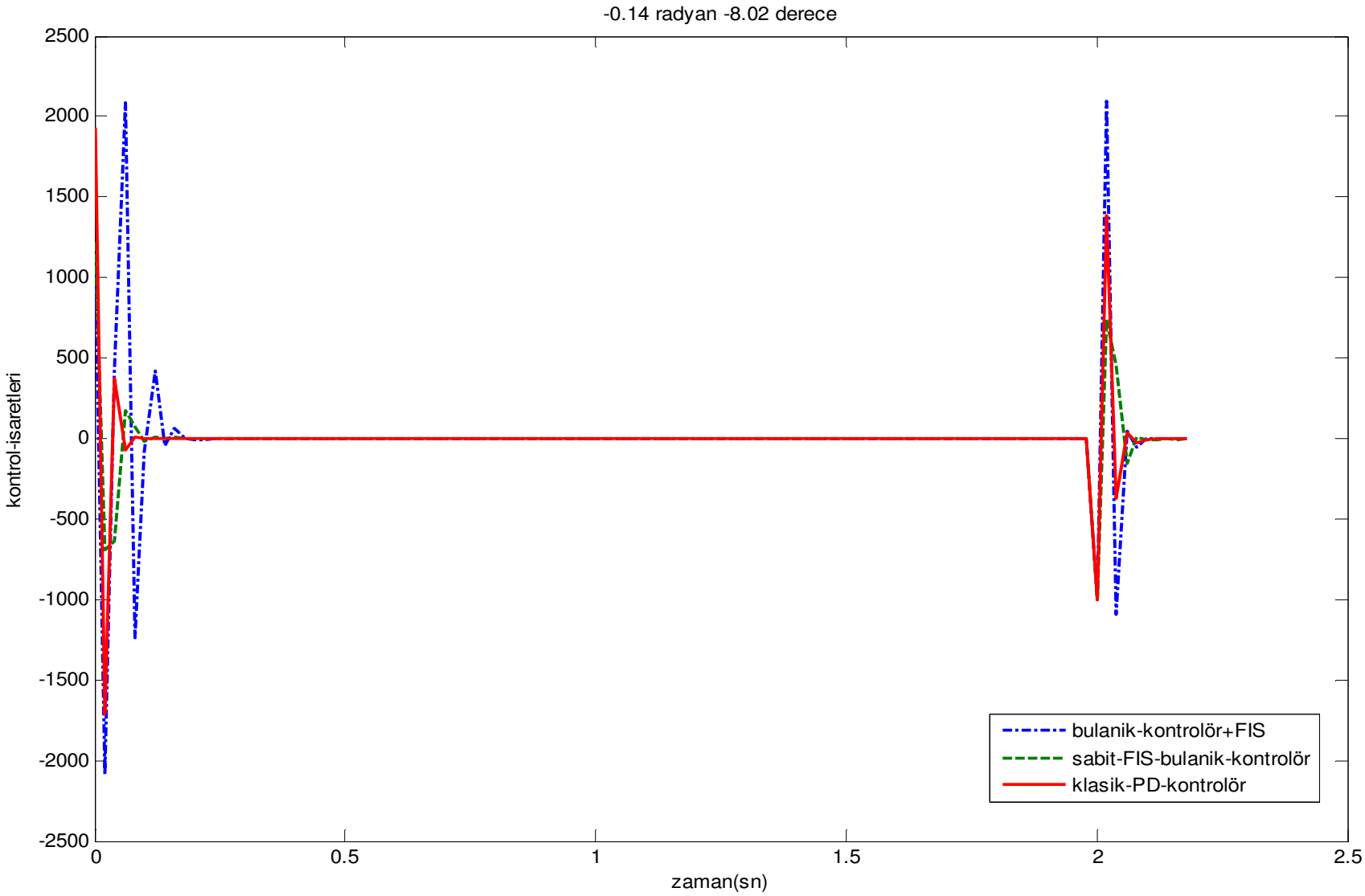
Şekil E.1: 0.14 radyan başlangıç açısı için çıkışlar



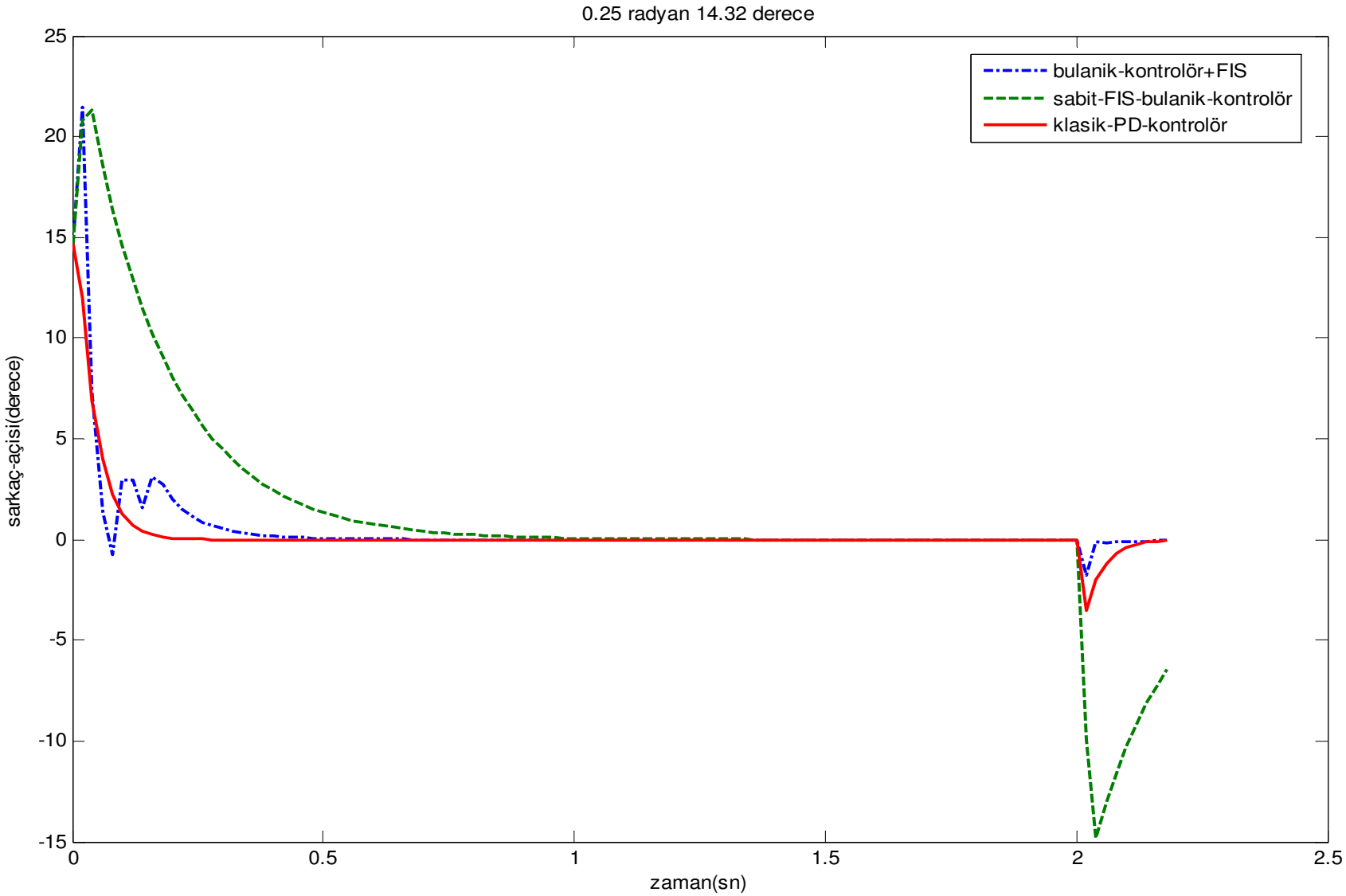
Şekil E.2: 0.14 radyan başlangıç açısı için kontrol işaretleri



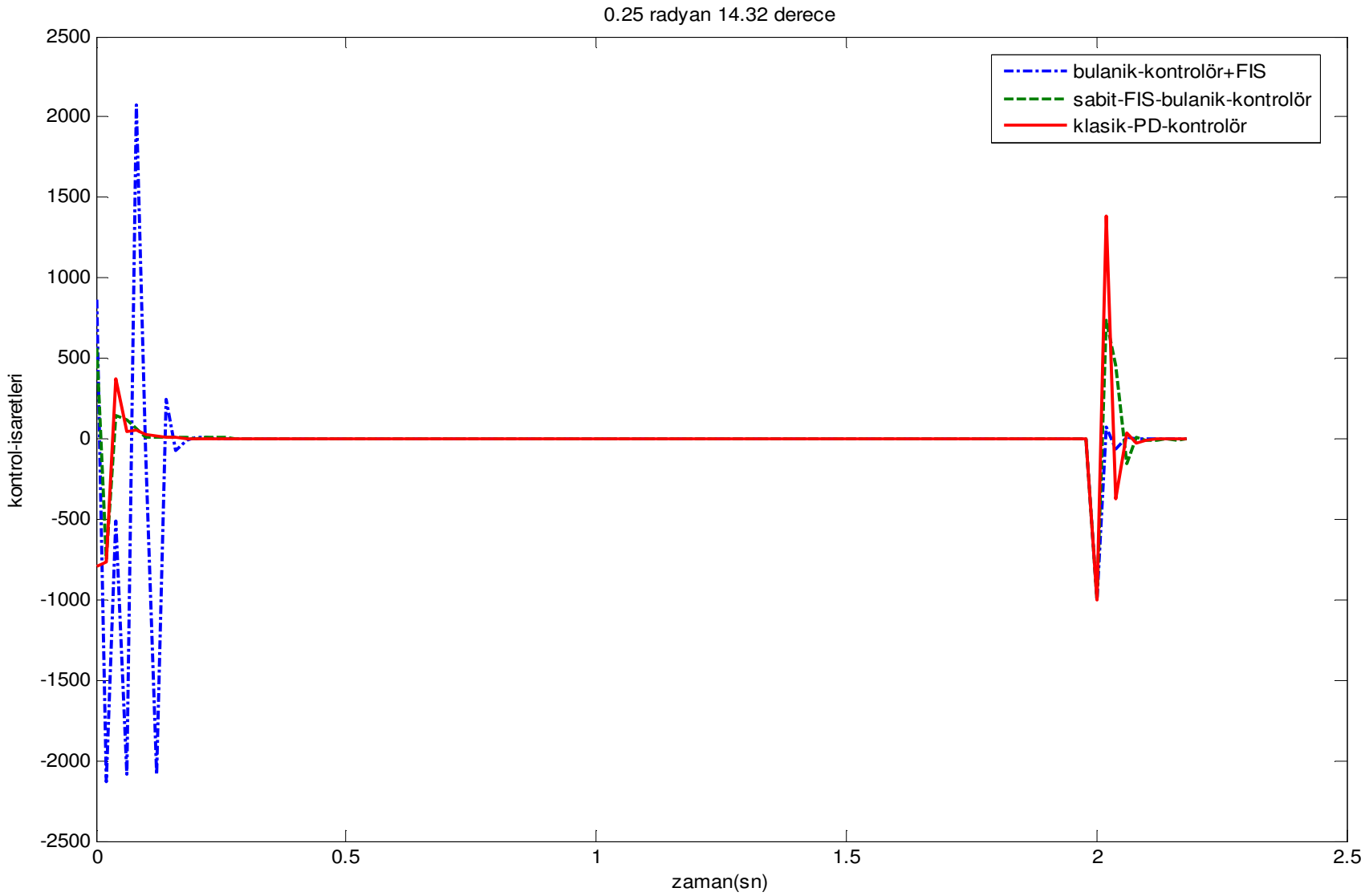
Şekil E.3: -0.14 radyan başlangıç açısı için çıkışlar



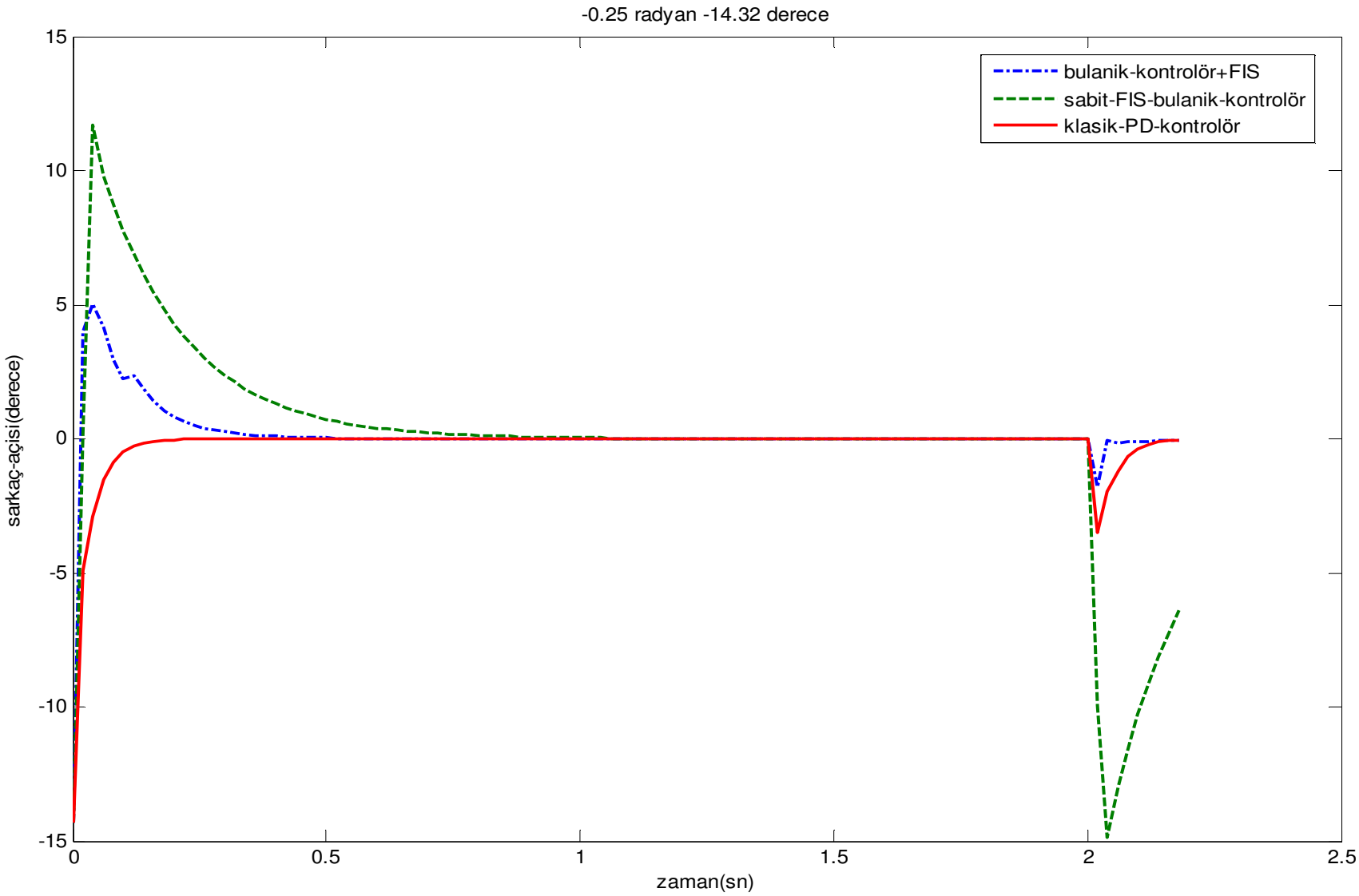
Şekil E.4: -0.14 radyan başlangıç açısı için kontrol işaretleri



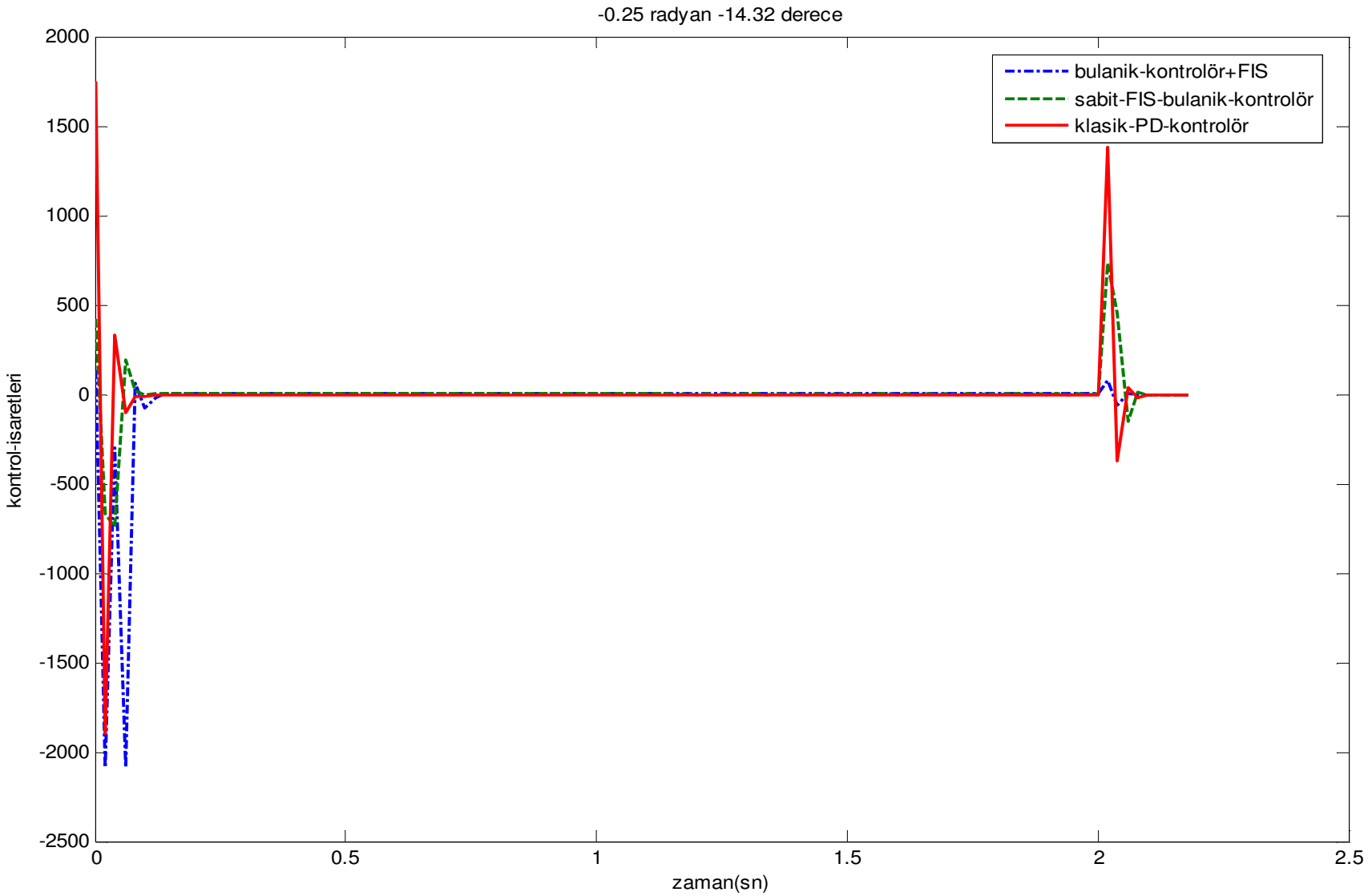
Şekil E.5: 0.25 radyan başlangıç açısı için çıkışlar



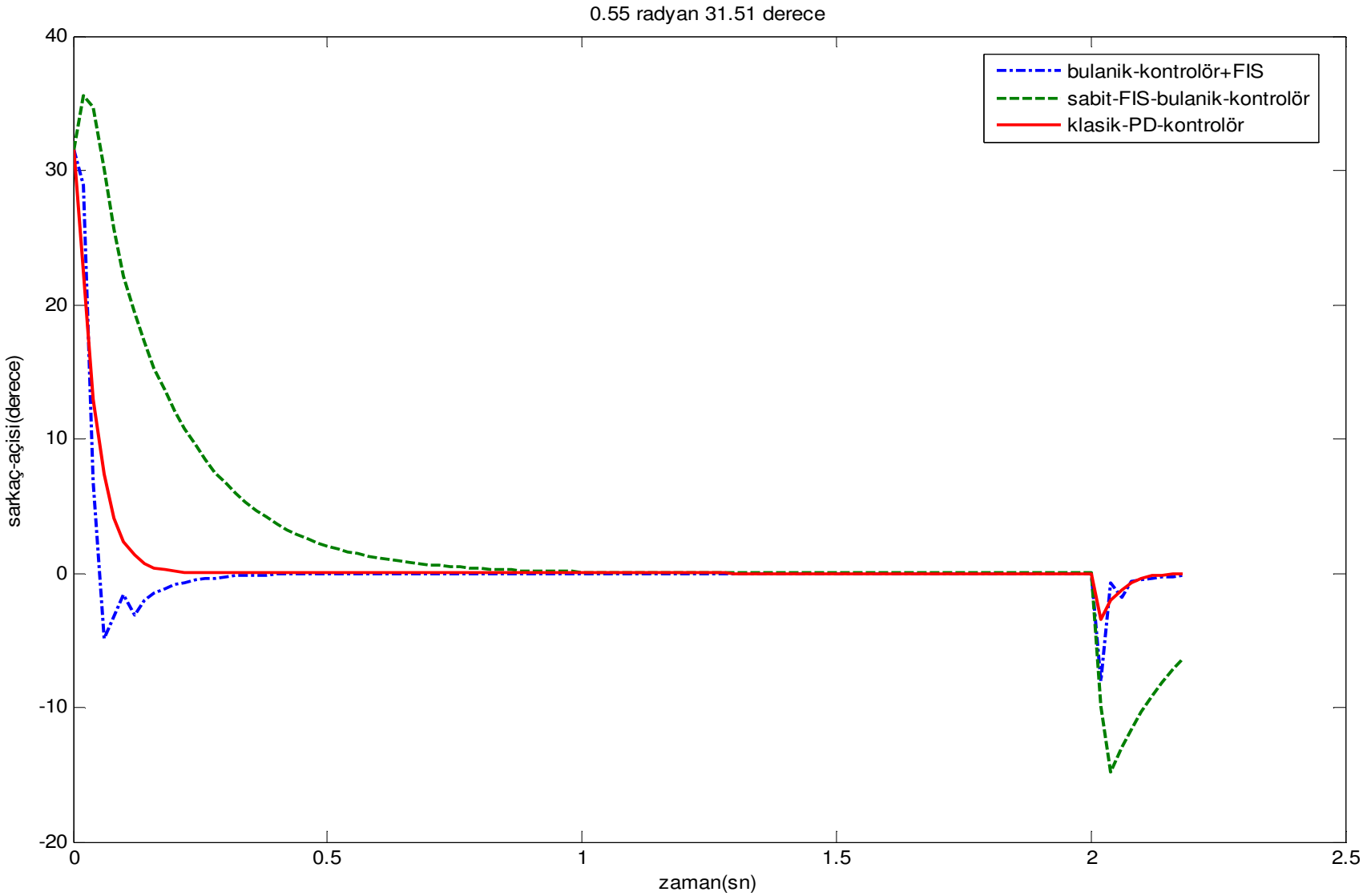
Şekil E.6: 0.25 radyan başlangıç açısı için kontrol işaretleri



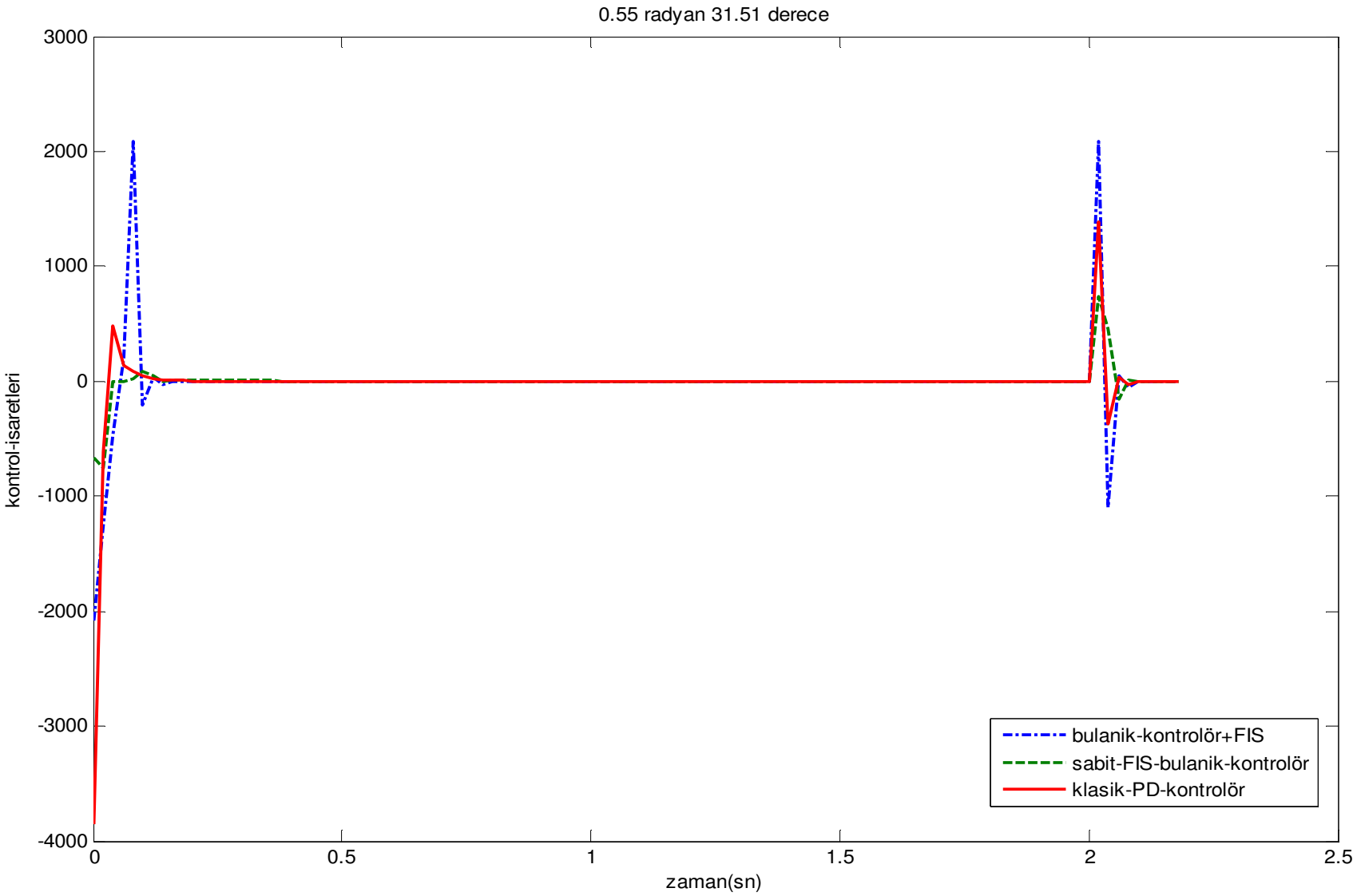
Şekil E.7: -0.25 radyan başlangıç açısı için çıkışlar



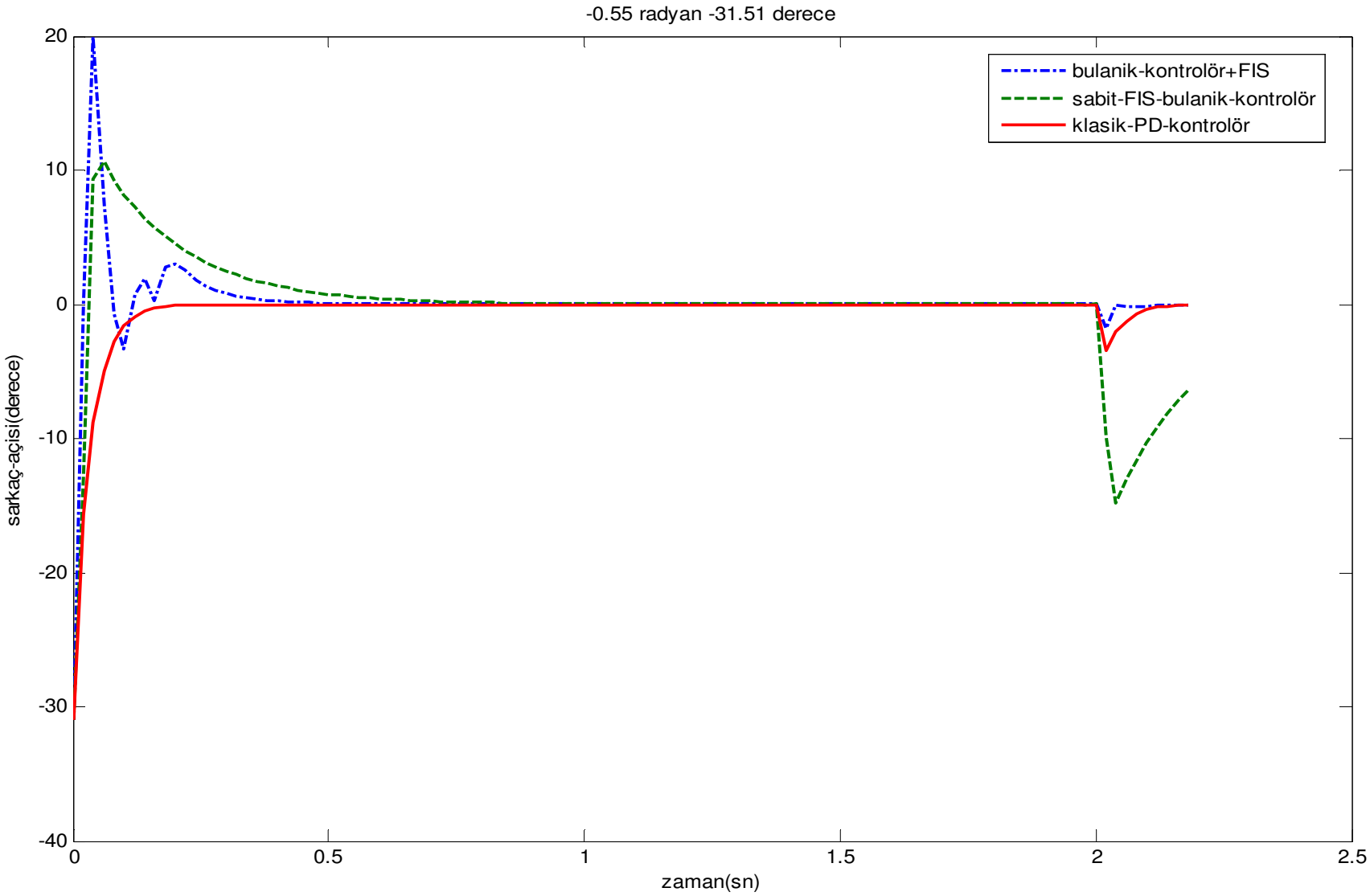
Şekil E.8: -0.25 radyan başlangıç açısı için kontrol işaretleri



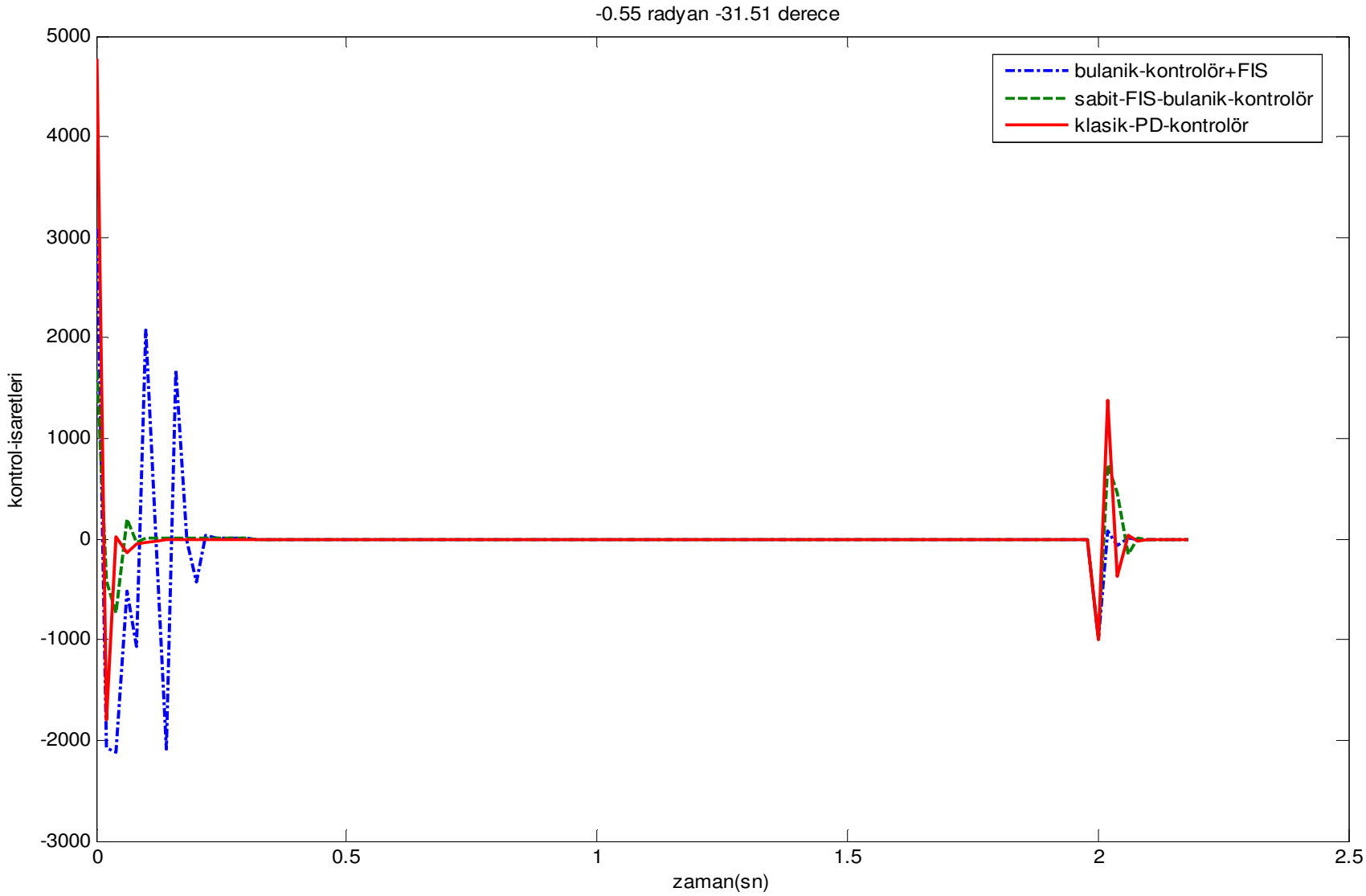
Şekil E.9: 0.55 radyan başlangıç açısı için çıkışlar



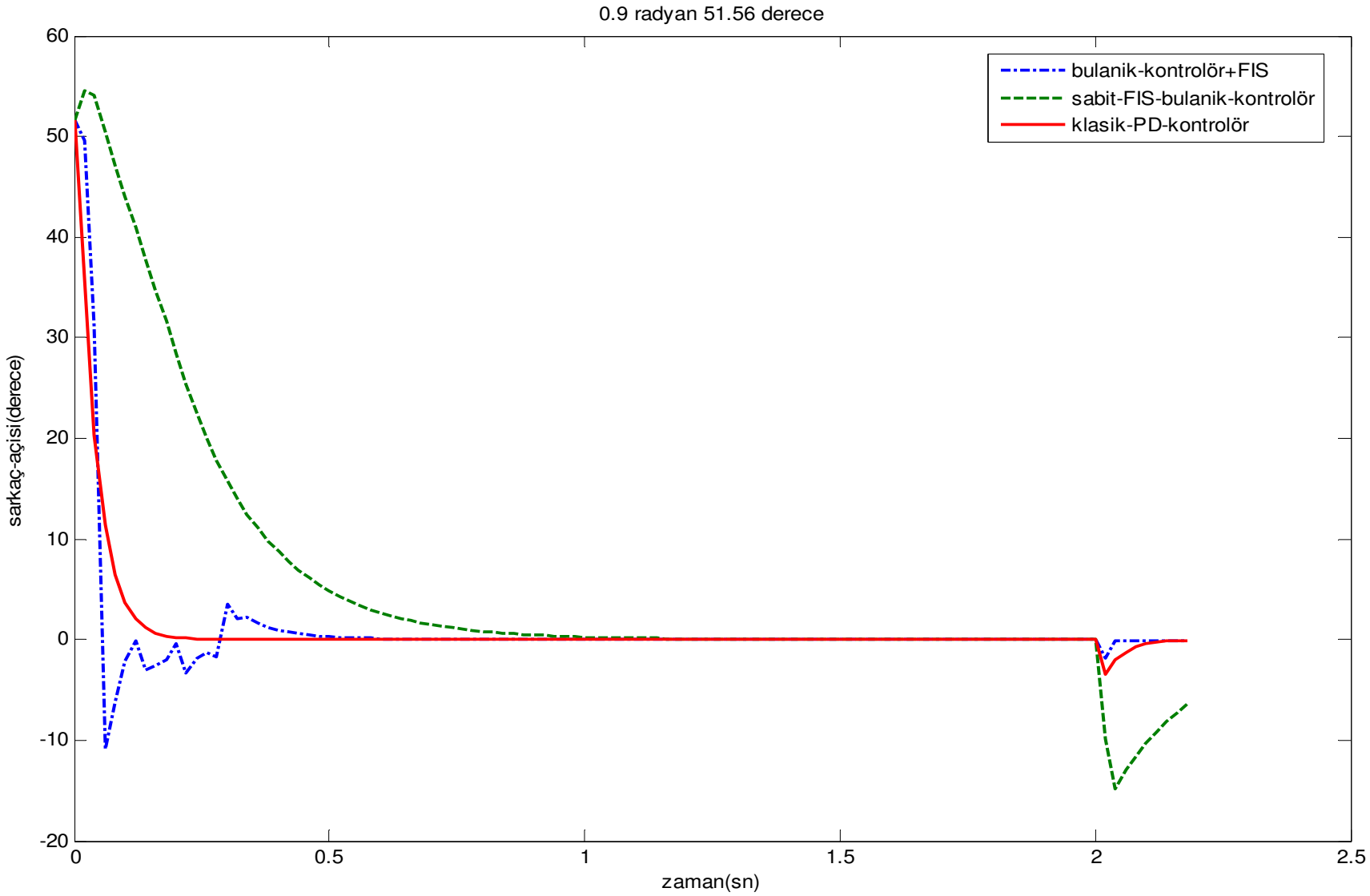
Şekil E.10: 0.55 radyan başlangıç açısı için kontrol işaretleri



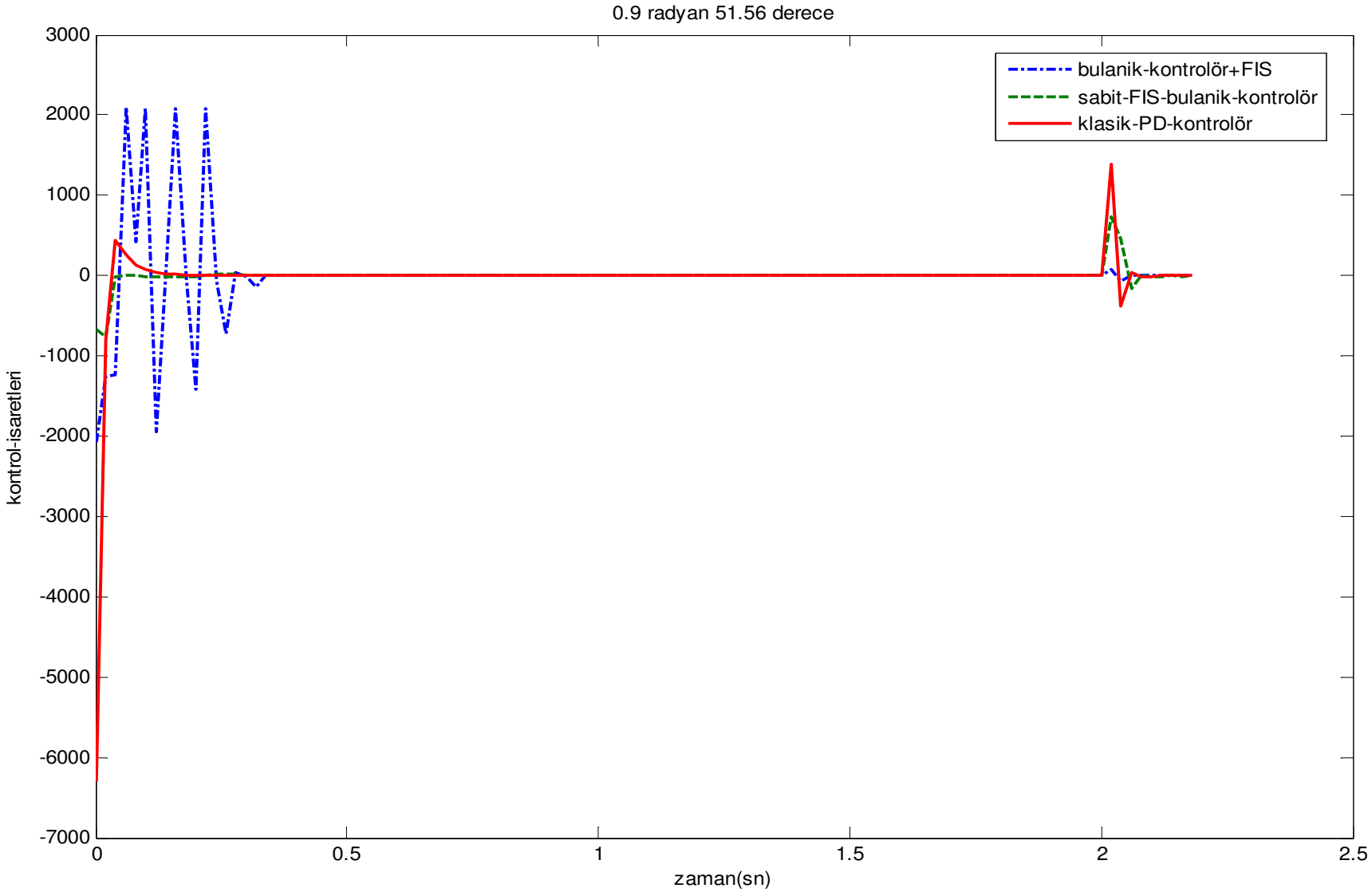
Şekil E.11: -0.55 radyan başlangıç açısı için çıkışlar



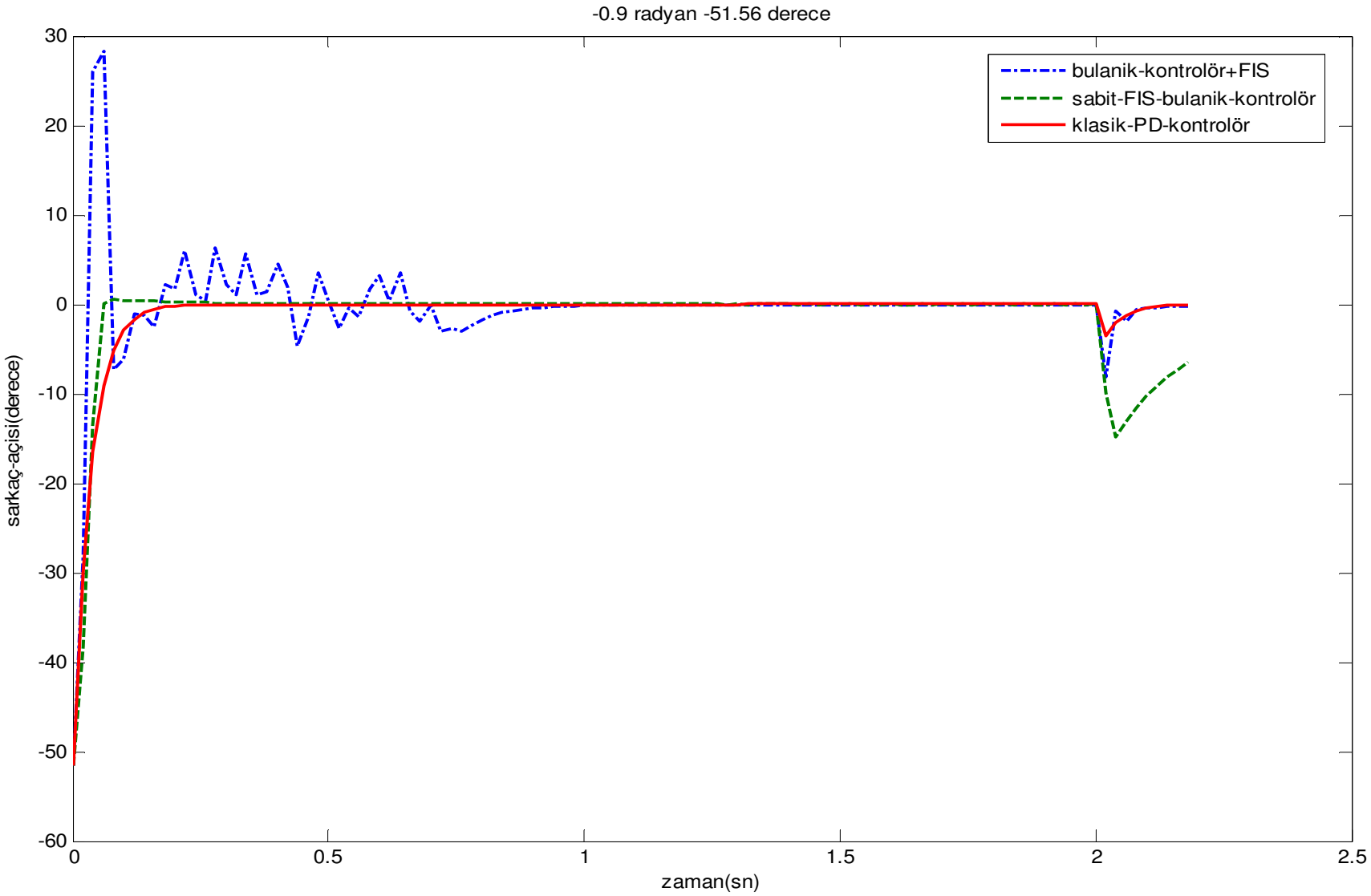
Şekil E.12: -0.55 radyan başlangıç açısı için kontrol işaretleri



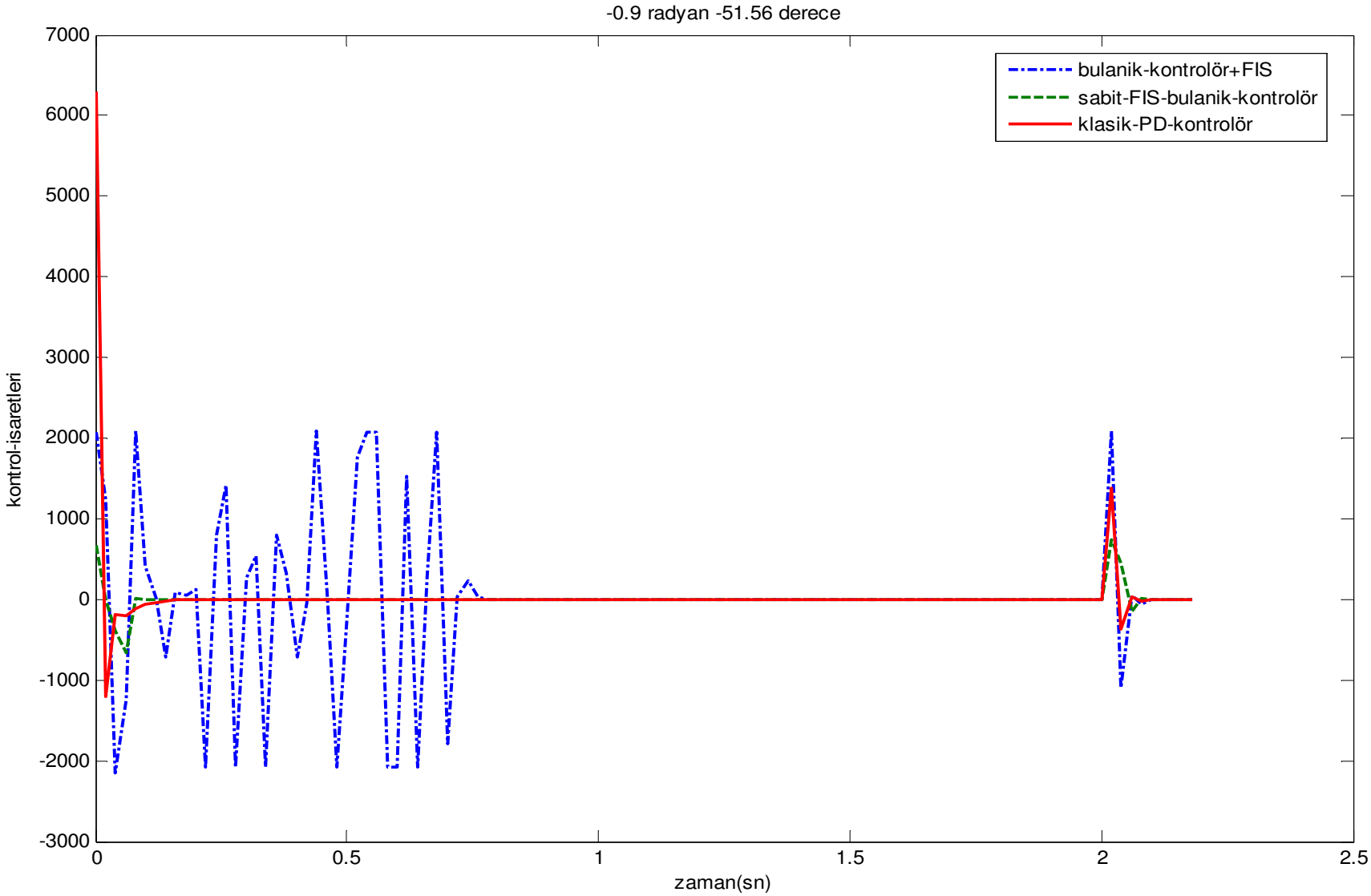
Şekil E.13: 0.9 radyan başlangıç açısı için çıkışlar



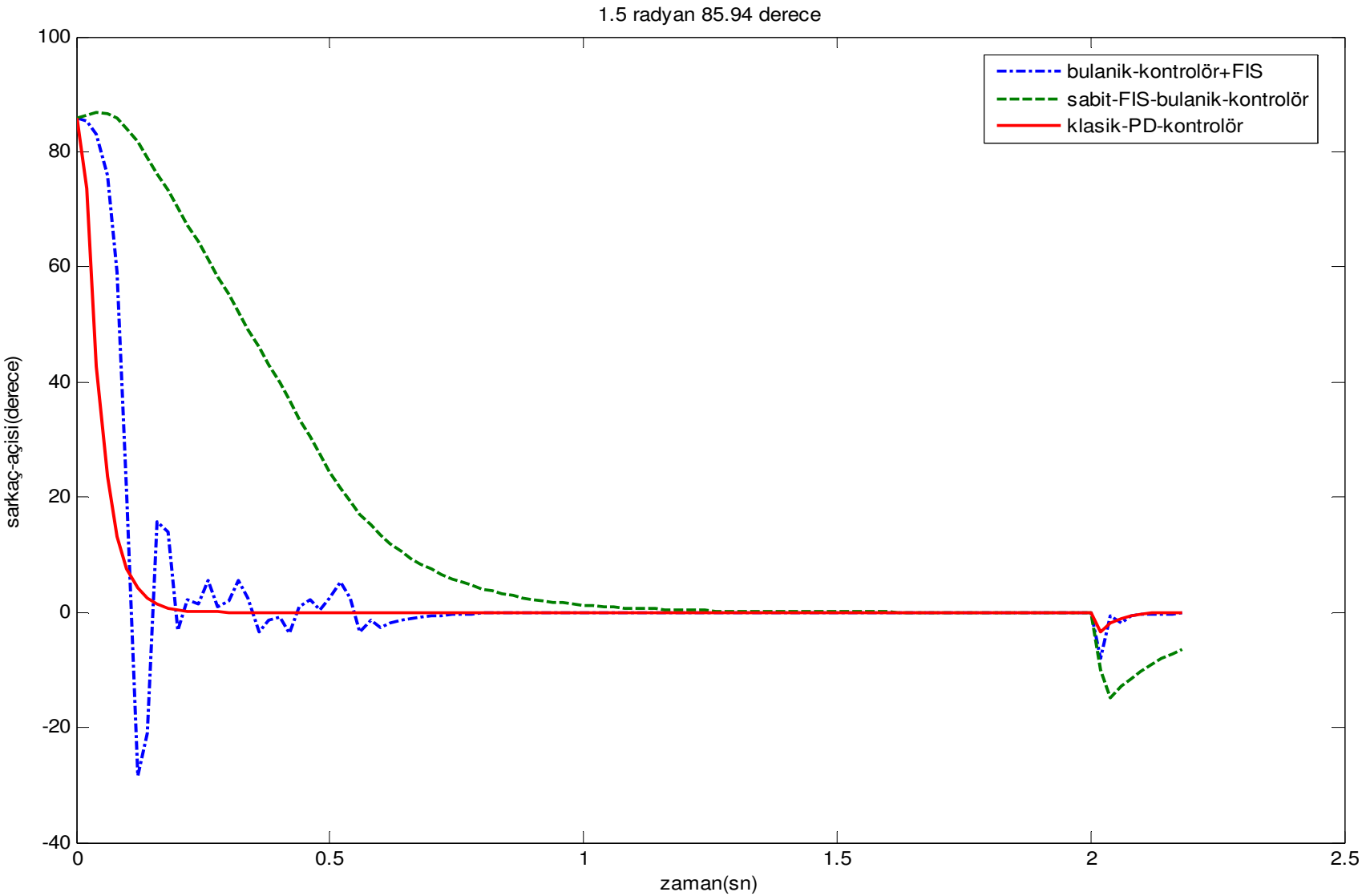
Şekil E.14: 0.9 radyan başlangıç açısı için kontrol işaretleri



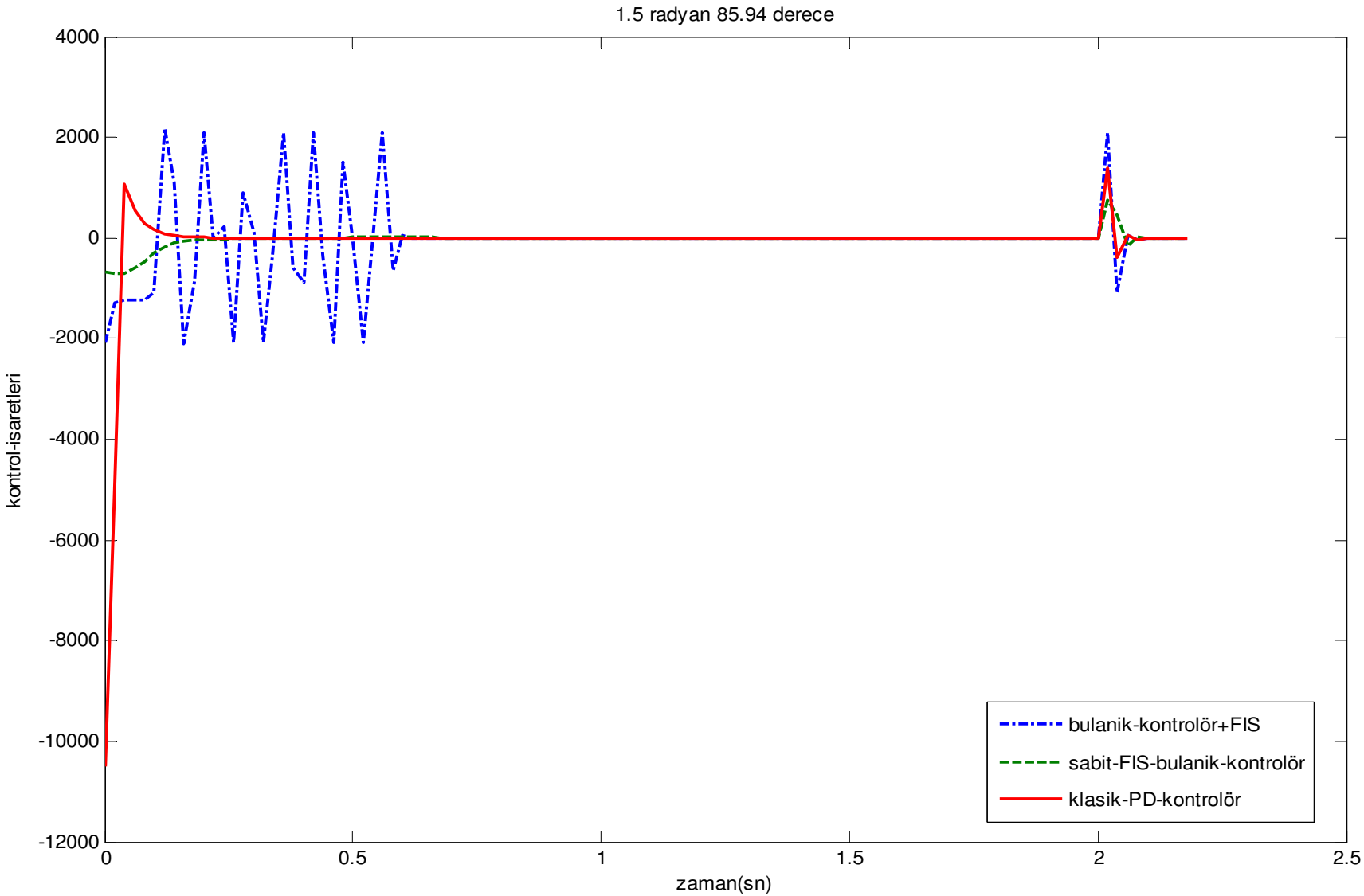
Şekil E.15: -0.9 radyan başlangıç açısı için çıkışlar



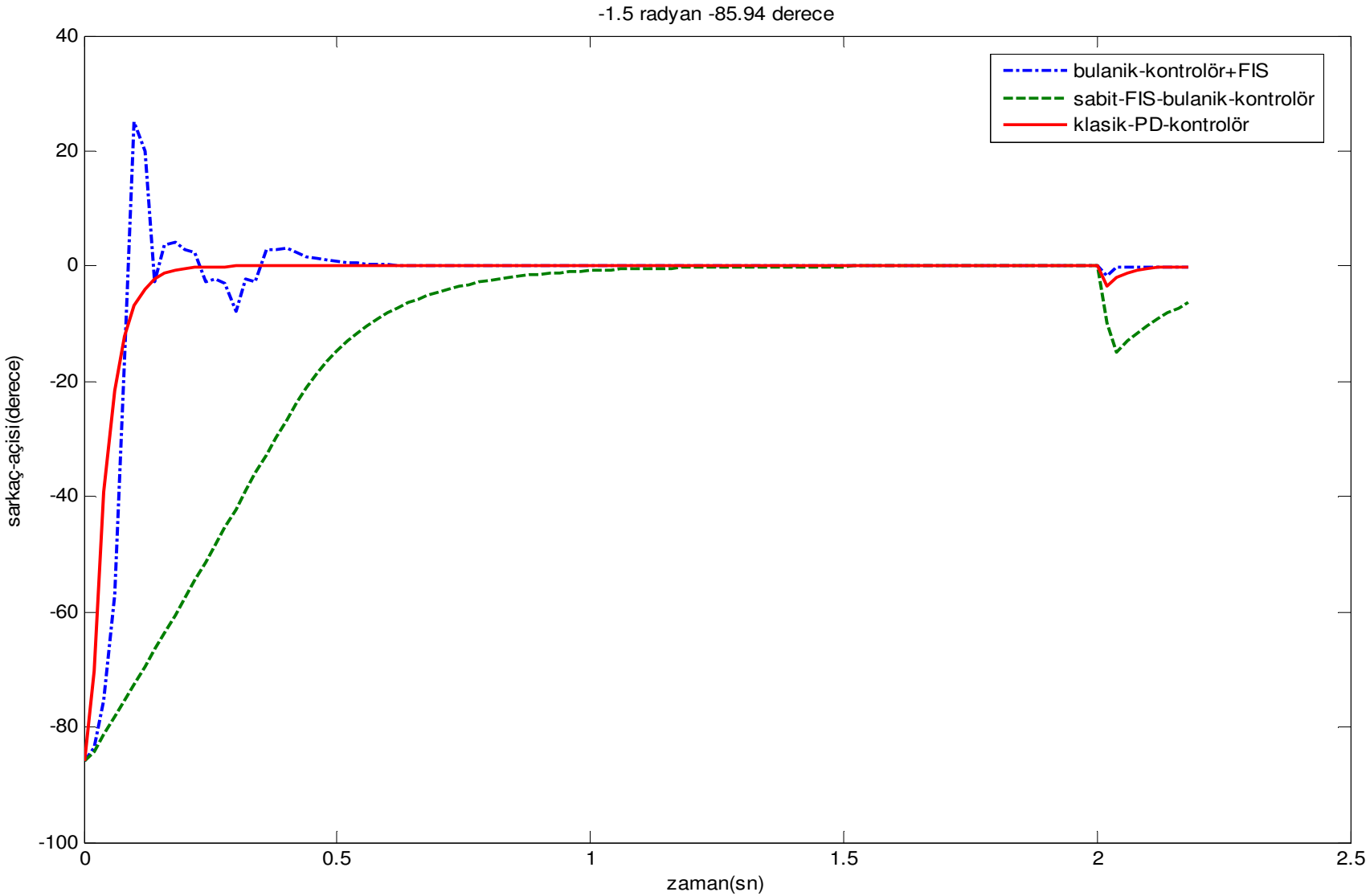
Şekil E.16: -0.9 radyan başlangıç açısı için kontrol işaretleri



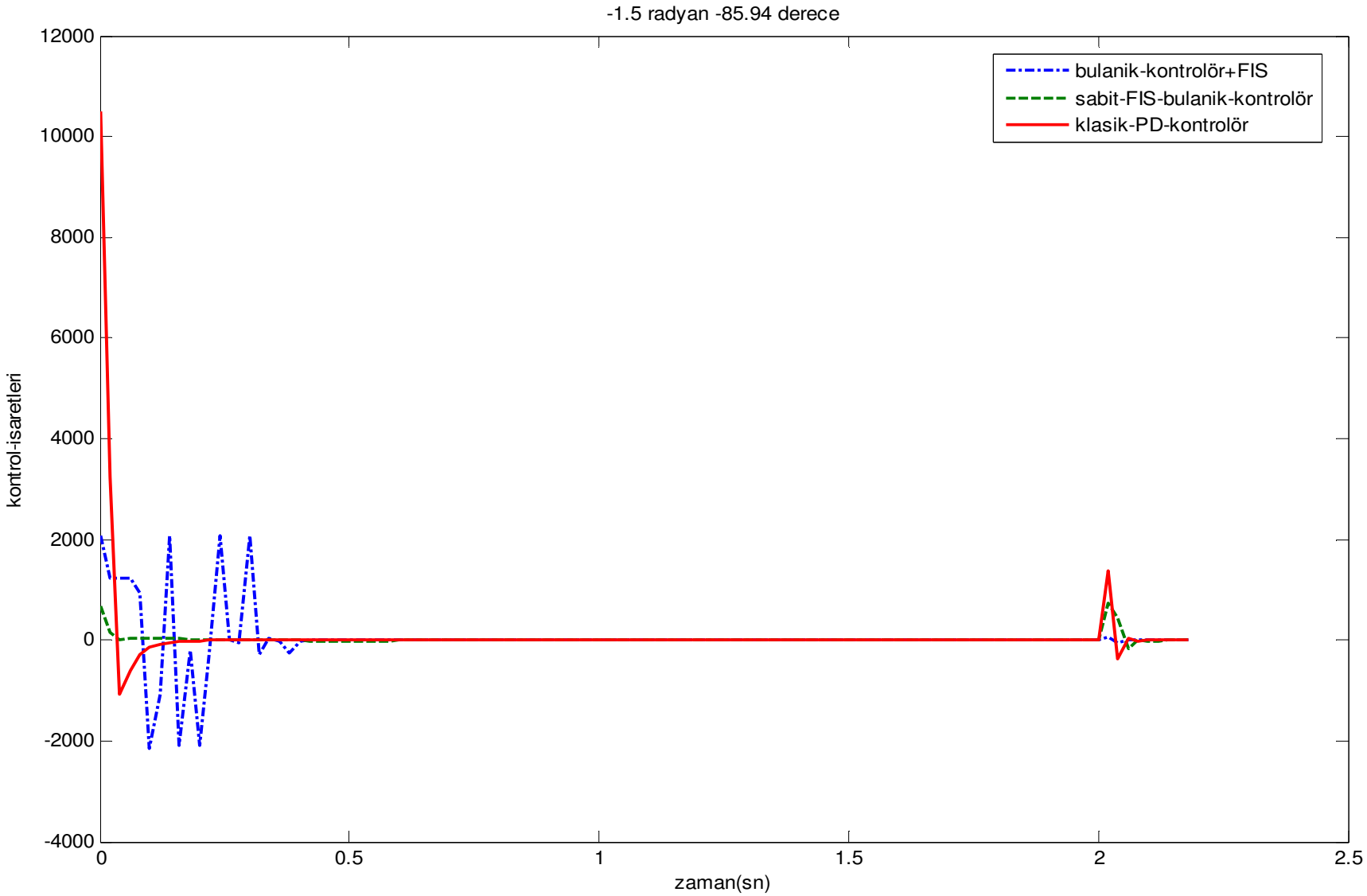
Şekil E.17: 1.5 radyan başlangıç açısı için çıkışlar



Şekil E.18: 1.5 radyan başlangıç açısı için kontrol işaretleri



Şekil E.19: -1.5 radyan başlangıç açısı için çıkışlar



Şekil E.20: -1.5 radyan başlangıç açısı için kontrol işaretleri

ÖZGEÇMİŞ

Ömer Güvenç Karaoğlan, 1980 yılında Samsun'da doğdu. İlköğrenimini Samsun'un Havza ilçesinde, orta ve lise öğrenimini ise Samsun Anadolu Lisesinde tamamladı. Haziran 2003 tarihinde Gazi Üniversitesi Elektrik-Elektronik mühendisliği bölümünden mezun oldu. 2003-2004 akademik yılı bahar döneminde İ.T.Ü Fen Bilimleri Enstitüsü'ne bağlı Kontrol ve Otomasyon mühendisliği bölümünde yüksek lisans öğrenimine başladı.