YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GENETİK ALGORİTMA TABANLI OPTİMAL ADAPTİF FUZZY PID KONTROLCÜ TASARIMI

Bilgisayar Müh. Mustafa ÇELEBİ

FBE Elektrik Mühendisliği Anabilim Dalı Kontrol ve Otomasyon Programında Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı : Doç. Dr. İbrahim Beklan KÜÇÜKDEMİRAL (YTÜ)

İSTANBUL, 2011

İÇİNDEKİLER

Sayfa

SİMGE LİSTESİ

a_x	Üçgen üyelik fonksiyonu sol sınır değeri
b_x	Üçgen üyelik fonksiyonu merkez sınır değeri
c_x	Üçgen üyelik fonksiyonu sağ sınır değeri
ce	Hatanın değişimi
e	Hata
fit(x)	x niteliğine ait uygunluk fonksiyonu
K_d	PID türevsel kazanç katsayısı
K_i	PID entegral kazanç katsayısı
K_p	PID oransal kazanç katsayısı
N	Popülasyon büyüklüğü
Pc	Çaprazlama oranı
P_m	Mutasyon oranı
$\mu_A(x)$	A kümesine ait üyelik derecesi gösterimi
V	Bulanık küme maksimum operatörü
$\wedge$	Bulanık küme minimum operatörü

KISALTMA LİSTESİ

BMK	Bulanık Mantık Kontrolör
COG	Ağırlık Merkezi Yöntemi
FIS	Fuzzy Inference Systems – Bulanık Çıkarım Sistemleri
FLC	Bulanık Mantık Kontrol
FPID	Bulanık PID Kontrolör
GA	Genetik Algoritma
G/Ç	Giriş/Çıkış
IAE	Integral Absolute Error – Mutlak Hatanın Entegrali
IEEE	Institute of Electrical and Electronics Engineers – Elektrik ve Elektronik Mühendisleri Enstitüsü
MIMO	Multiple Input Multiple Output – Çok Giriş Çok Çıkış
MISO	Multiple Input Single Output – Çok Giriş Tek Çıkış
MOGA	Multi Objective Genetic Optimization – Çok Amaçlı Genetik Optimizasyon
PID	Oran – Entegral – Türev
PLC	Programmable Logic Controller – Programlanabilir Mantık Kontrolör

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖNSÖZ

Tez konusu seçiminden tez yazımına kadar geçen süreçte; desteğini ve yardımını hiçbir zaman esirgemeyen değerli hocam İbrahim Beklan Küçükdemiral'a çok teşekkür ederim. Kıymetli hocalarımız: Galip Cansever, Emin Olcay ve Murat Yeşiloğlu'na teşekkürlerimi sunarım. Başaracağıma her zaman inanan, değerli hocam Şeref Naci Engin'e teşekkürü bir borç bilirim. Hayatın her alanındaki maddi ve manevi destekleri için annem Kadriye Çelebi ve ablam Nida Çelebi'ye sonsuz teşekkür ederim.

ÖZET

Birbirinden farklı metodlar geliştirilmesine rağmen doğrusal olmayan kontrol sistemlerinde kullanılan klasik PID kontrolör performanslarının, doğrusal kontroldeki başarıları ile örtüşmediği gözlemlenmiştir. Doğrusal olmayan sistemler için doğrusal olmayan çıkışlar üretebilen ve uygulaması çok basit olan bulanık mantık kontrolörler (BMK) klasik PID performansına göre daha başarılı sonuç vermektedir.

Bulanık kontrolörün bu avantajına rağmen; kontrolör tasarım parametrelerinin (giriş/ çıkış fonksiyonları, fonksiyon sayısı, kural tabloları) sadece uzman bilgisi tarafından yada deneme yanılma yöntemi ile belirlenme gibi bir dezavantajı bulunmaktadır. Bu sebepten dolayı; çözümü istenen fakat matematiksel modeli hakkında hiçbir bilgiye sahip olunmadığı durumların optimizasyonunda güçlü olduğu bilinen Genetik Algoritma ve ilişkilendirmede ihtiyacımız olan için uygulama fonksiyonu Integral Absolute Error-Mutlak Hatanın Entegrali kullanılarak bulanık kontrolör parametreleri belirlenmektedir.

Klasik PID kontrolör tasarımındaki en önemli süreç olan kazanç katsayılarının (K_p, K_i, K_d) ayarlanması işlemi bulanık kontrolör ile online (eş zamanlı) olarak gerçekleşebilmektedir. Genetik Algoritma tarafından parametreleri belirlenen bulanık kontrolör ile PID kazanç katsayılarının ayarlanması birbirinden farklı uygulama koşullarında çok etkin ve verimli sonuçlar vermiştir.

Aynı amaç için kullanılan bu iki metodun ortak özelliği "doğrusal olmayan sistemlerde kullanılabilme" ve "sistemin matematiksel modeli hakkında bilgi sahibi olmadan etkin çalışabilme" çalışma kapsamında yapılan benzetimler ile disiplinler arası geniş bir alanda ve farklı süreçlerde uygulanabilirliğini göstermiştir.

Anahtar Kelimeler: Bulanık Mantık, PID, Genetik Algoritma, Optimizasyon

ABSTRACT

Although different methods are explored, success of conventional PID performance on non-linear control systems with linear control systems are not-matched. Fuzzy controllers which is very easy to apply and can generate non-linear outputs for non-linear systems; gives more successful results than conventional PID.

Despite of its advantage, fuzzy controller has a disadvantage; design parameters (input/ output functions, function number, rule tables) can only be determined by system experts (expert knowledge) or trial-error method. Due to this matter; problems that we have no prior knowledge about its mathematical model solved by powerful optimization tool Genetic Algorithm; for relationship requirements IAE (Integral Absolute Error) as a fitness function used and fuzzy controller parameters determined.

The most important process of designing conventional PID controller is determining PID gains (K_p, K_i, K_d) can be done online by fuzzy controller. Tuning PID gains by genetic algorithm based fuzzy controller gives an effective and powerful results on different operating conditions.

The same objective of these two methods: “useful on non-linear systems” and “work propoerly without non-requirement prior knowledge mathematical model of system” simulation results show that controller can be applicable on wide range and multidiciplinary proccesses.

Keywords: Fuzzy Logic, PID , Genetic Algorithm , Optimization

1. GİRİŞ

Matematiksel olarak modellenemeyen yada modellenmesi çok karmaşık olan; doğrusal olmayan sistemlerinin üzerinde kullanımda son derece başarılı sonuçlar veren bulanık mantık kontrolörler son yıllarda kontrol teorisinin en popüler araştırma konusu haline gelmiştir. Sistem hakkında yeterli bilginin olmadığı, sistem girişlerinin sensör değil; insan tarafından alındığında ve sistem çıkışları için net bir müdahale yerine daha hassas, dilsel değişken adı verilen komutlar ile işlem yapılması gerektiğinde "bulanık mantık" bizim için biçilmiş kaftan vazifesini görür. Bulanık mantık için keskin ve kesin sınırlar yoktur. Uygulaması esnasında olasılık ile değil oluşum derecesi ile ilgilenilir. Günlük hayatta kullandığımız "hızlı, çok hızlı, yavaş, çok yavaş" kelimelerinin dilsel terimlere dönüştürme işlemlerini gerçekleştirerek sistemi niteliksel olarak tanımlar ve yaşanan dünya ile ilişki kurmamızı sağlar.

Bununla birlikte BMK; doğrusal olmayan karmaşık sistemlerin bile kolayca tasarlanıp, düşük maliyette uygulanması gibi güçlü yanları olduğu gibi; uzman bilgisi ihtiyacı denilen bir dezavantajı bulunmaktadır. Sistemi iyi bilen, en iyi tanıyan kişinin sezileri; diğer deyişle "uzman bilgisi" ihtiyacı ve sistem parametrelerinin deneme yanılma yapılmadan öğrenilemeyecek olması problemimizin tanımıdır. Bizim amacımız bulanık mantık veri tabanı bileşenlerinden; giriş ve çıkış üyelik fonksiyonlarının sınır optimizasyonu ve kural tabanına ait kuralların optimizasyonunu gerçekleştirmektir. Bu optimizasyonu gerçekleştirmek için kullandığımız en sistematik global optimizasyon yöntemi olan genetik algoritma (GA)'dır.

En klasik tanımı ile Genetik algoritmalar, evrim sürecini örnek alan bir arama yöntemidir. Amacı, problemler için aranan çözümlerin; doğada geçerli olan en iyinin yaşaması kuralına dayanarak sürekli iyileşen çözümler bulmaktır. Bunu sağlamak için "iyi"nin ne olduğunu belirleyen bir uygunluk (fitness) fonksiyonu ve yeni çözümler üretmek için yeniden kopyalama, çaprazlama ve mutasyon gibi operatörleri kullanır. GA, ele alınan problemin belirlenmiş bir uygunluk koşuluna göre evrilerek mükemmelleşmesini amaçlayan bir evrim teorisine dayalı geliştirilmiş algoritma olarak tanımlanabilir. Genetik algoritmaların bir diğer önemli özelliği de bir grup çözümle uğraşmasıdır. Bu sayede çok sayıda çözümün içinden iyileri seçilip kötüleri elenebilmektedir.

Literatürde GA ve bulanık mantık ile ilgili birçok araştırma bulunmaktadır. Bununla birlikte BMK ait model parametrelerinin (üyelik fonksiyonu sınırları, kural tabanı) genetik algoritma ile optimizasyonu farklı bilim adamları tarafından farklı yöntemlerle geliştirilmiştir. Genel yöntem bulunamamasına rağmen geleneksel sonuçlara göre daha iyi sonuçlar veren çalışmalar

yapılmıştır. Aşağıda adı geçen çalışmalar bu optimizasyon yöntemlerinin kullanıldığı çalışmalardan bazılarıdır.

Ahmed Rubaai, Marcel J. Castro-Sitiriche ve Abdul R. Ofoli, endüstriyel motor sürücüleri için GA tabanlı melez bulanık PID kontrolörlerin gerçek zamanlı uygulaması üzerinde çalışma yapmıştır. Bulanık PID kontrolör (FPID) ve geleneksel PID kontrolör entegrasyonunda melez yapı oluşturulmuş; optimal giriş ölçeklendirme faktörü ve çıkış değerleri için GA kullanılmıştır. Asıl amaçları PID ve FPID kullanılarak; PID veya FPID ayrı ayrı kullanımından daha iyi sonuç verdiğini ispatlamaktır. Melez kontrolör kullanma amaçları olarak; halihazırda birçok durum için yeterli olan PID ile arkaplanda bekleyen ve gürültülerde devreye girecek olan FPID kullanımından bahsetmişlerdir. Yine bu melez kontrolör ile gerçek zamanlı fırçasız motor hız kontrolü uygulaması yapılmıştır (Rubaai vd., 2008).

Ding, Wang, Zhang ve Hu, “A fuzzy neutral network controller based on optimized genetic algorithm for UC rolling mill” adını verdikleri çalışmalarında; gövde mil bükme sistemi ile ilgili hem teorik hemde uygulamaya yönelik çalışma yapılmıştır. Kesin bir sonuca ulaşmak için nöral bulanık kontrolör tasarlanmış ve sisteme uygulanmıştır. Sinir ağlarının öğretme süreci optimizasyonu için basit GA kullanılmış ve daha yüksek hassasiyetli sonuç elde edilmiştir. Yapılan bu çalışma ile daha hızlı sistem cevabı ve model belirsizliğine karşı daha ileri seviyede gürbüzlük sağlanmıştır (Ding vd., 2009).

Chakraborty, Bandyopadhyay ve Patranabis, "Bulanık Genetik otomatik PID parametre optimizasyonu" adını verdikleri bir çalışma yapmışlardır. Çalışma bitkin (Dead-Beat) kontrol teorisi temelinde oluşturulmuştur. Bulanık kontrol tekniği kontrolör çıkışını tahmin etmesi için kullanılmış ve Takagi-Sugeno modeline ait kural optimizasyonu GA ile gerçekleştirilmiştir. Kuralların süreç uzmanları tarafından oluşturulduğu çalışmaların aksine kayde değer gelişme GA ile yapılan kural optimizasyonu olmuştur. Ziegler-Nichols formülü ile ayarlanan PID kontrolör performasının iyi olduğu gözlemlenmiştir (Chakraborty vd., 2001).

Lo ve Sadegh, enerji sistemlerinin kararlılık kontrolü için FPID kontrolör geliştirmişlerdir. PID kontrolör parametreleri Ziegler-Nichols ayar yöntemi ile ayar edilmiş ve GA ile edinilmiş; probleme endüstriyel yönden bir bakışaçısı ile yaklaşmışlardır. Bulanık mantık kural tabanı "kural oluşturucu" adını verdikleri fonksiyon tarafından otomatik olarak oluşturulmuştur. BMK vasıtası ile SVC (Static Var Compansator) sisteminin kompanzasyon kararlılığı sağlanmıştır. Geleneksel PID ile FPID kontrolör karşılaştırılmış ve benzetimler

yapılmıştır (Lo ve Sadegh, 2003).

Tang, Man, Chen ve Kwong; optimal FPID kontrolör mekanizmasını MOGA (Multi Objective Genetic Optimization Çok Amaçlı Genetik Optimizasyon) ile gerçekleştirmişlerdir. Geliştirilen bu kontrolör geleneksel PID ayrık versiyonu olarak oluşturulmuştur. FPID tasarım ve geliştirilme aşamasında; üyelik fonksiyonları üçgen fonksiyon olarak ve dört adet kural PI ve dört kural D olarak oluşturulmuştur. PID kazançları MOGA kullanılarak optimize edilmiş ve FPID kontrolör verimli hale getirilmiştir. GA tarafından kullanılan amaç fonksiyonu üç madde içermektedir: Maksimum aşımı minimize etmek, çıkışın yerleşme zamanını minimize etmek ve çıkışın yükselme zamanını minimize etmek. Bilgisayar benzetimleri ile karşılaştırmalar yapılmıştır ve geliştirilen kontrolörün verimli sonuç verdiği izlenmiştir (Tang vd., 2001).

Ying, yaptığı çalışmada Zadeh "ve operatörü" ve giriş bulanık setleri (yamuk, bazı özel durumlarda üçgen fonksiyonlar) kullanılarak giriş çıkış ilişkilerinin çıkarımı üzerine çalışmıştır. Giriş bulanık kümeleri ve giriş uzayının şekli arasında gerekli ilişkileri karakterize edecek koşulları incelemiştir. Bu analitik ilişkiye göre giriş alanları değişik bölgelerde farklılık göstermektedir. Her iki tipte (Mamdani ve Takagi Sugeno) kullanılabildiği fakat Mamdani tipi bulanık kontrolörler için giriş bulanık setlerinin seçiminin daha olumlu sonuçlar verdiğini belirtmişlerdir. Analizlerine göre ikizkenar yamuk şeklindeki bulanık setler diğer tüm setlerden daha anlaşılır ve esnektir. Ying'in yaptığı bu önemli tespite göre; bulanık set seçiminde ikizkenar yamuk ilk seçenek olmalıdır. Otomatik olarak bulanık girişleri oluşturmak için için yapay sinir ağları veya GA kullanılmasının faydalı olabileceğini önermiştir (Ying, 2006).

Baogang Hu, George K. I. Mann, ve Raymond G. Gosine; FPID ve GA kullanılarak sistem dizaynına yönelik teorik bir çalışma yapmışlardır. Kontrolörün yapısı bir giriş ve üç kuraldan toplamda altı ayar parametresinden oluşmaktadır. Tasarım stratejisi olarak garantili PID performanslı bulanık kontrolör düşünülmüştür. Çalışmada kullanılan GA, doğrusal kontrolörden kötü olmamak kaydı ile optimizasyon yapar. Doğrusal yaklaşım indisi ve doğrusal olmayan varyasyon indisi bulanık kontrolör tasarımında sonuç edinmede kullanılmştır (Hu vd., 1999).

Visioli; PID kontrolörlerin ayar edilmesi için temel olarak bulanık mantık kullanarak farklı metotları karşılaştırılmıştır. Her bir kontrolörün en iyi şekilde kullanılabilmesi ve parametre ayar işlemleri için (katsayıların ayarlanması, üyelik fonksiyonlarının belirlenmesi) GA

kullanılmıştır. Klasik PID kontrolör ile PID benzeri bulanık kontrolör karşılaştırılmalarını yapmış ve yüksek performanslı sonuçlar elde etmiştir (Visioli, 2001).

Seng, Marzuki ve Yusof, "Adaptif, kendinden ayarlamalı, bulanık kontrol sistemleri uygulaması" üzerine makale yayınlamıştır. Nöro-bulanık mantık kontrolörün tüm parametreleri GA tarafından eş zamanlı ayarlanmıştır. Yapay sinir ağı modeli ve gauss üyelik fonksiyonları kullanılarak yapılmıştır. GA çaprazlama ve mutasyon işlemleri ile daha hızlı bir yaklaşım metodu oluşturulmuştur. Bunun dışında esnek, farklı bir kodlama yapısı kullanılarak daha çabuk çözüme ulaşma ve optimizasyon yolu düşünülmüştür. Oluşturulan kontrolörün performansı geleneksel bulanık kontrolör ve klasik PID kontrolör ile karşılaştırılmıştır (Seng vd., 1999).

Visioli, bulanık çıkarım sistemine parametrelerinin GA ile ayarlandığı bir makale yayınlamıştır. Yapılan çalışmada oluşturulan bulanık kontrolörün hem üst aşım ve yükselme zamanını düşürdüğü gözlemlenmiştir. PID parametreleri Ziegler-nichols formülü ile tanımlanmıştır. Çok etkin bir kontrolördür ve endüstriye rahatça adapte edilebilir. Çünkü az bir programlama ile çeşitli parametre varyasyonlarında bile gürbüz bir kontrolör görünümündedir. Bulanık kontrolör parametreleri kolayca GA tarafından oluşturulmuş ve ayarlanmıştır (Visioli, 1999).

Joo Er ve Lei Sun, "Hybrid Fuzzy Proportional–Integral Plus Conventional Derivative Control of Linear and Nonlinear Systems" adını verdikleri çalışmalarında doğrusal olmayan sistemler için GA kullanılarak oluşturulan melez PID kontrolör geliştirmişlerdir. Melez PID kontrolör klasik PI kontrolör yerine kullanılmış; iki girişli normalleştirilmiş FLC ile D kontrolör artımsal formda işlenmiştir. Kontrolör kazanç katsayı optimizasyonu GA tarafından yapılmıştır. Doğrusal bir sistem olan "Sıcaklık kontrol sistemi" ve doğrusal olmayan sistem olan "füze sistemi modeli" üzerinde deneme yapılmıştır (Er ve Sun, 2001).

Hasan Ali, Murata ve Tamura, frenleme direnci sıcaklığının güç sistemlerinin geçici kararlılığı üzerindeki etkilerini bulanık mantık kontrollü uygulama üzerinde gözlemlemişlerdir. Direncin ortam sıcaklığından farklı şekilde ani artışları sistemi kötü yönde etkilemektedir. Farklı noktalarda yapılan kontrollerde bulanık mantık kontrollü sıcaklık artışının güç sistemi geçici kararlılığı üzerinde neredeyse etkisi olmamıştır. Bulanık kontrolör ile kontrol edilen frenleme direnci performansı geleneksel PID ile bulanık kontrolör ile karşılaştırılmış ve geliştirilen bulanık kontrolörün sade kullanımı kolay ve etkin olduğu gözlemlenmiştir (Ali vd., 2004).

Kobayashi, Hsu, Yamada ve Fujikawa, doğrusal sistemler için bulanık adaptif kontrolör adını verdikleri çalışmada zaman gecikmeli sistemler üzerinde çalışmışlardır. Bazı sistemlerde sinyal akışları arasında oluşan kaçınılmaz zaman gecikmesi bulanık kontrolör ile aşılmaya çalışılmıştır. Genelde bulanık kontrolör kural tablosu deneme yanılma yöntemi ile bulunurken; bu çalışmada bulanık kontrolör kural tablosu GA ile optimize edilmiş ve sisteme uygulaması yapılmıştır (Kobayashi vd., 1997).

Qin ve Du, "A fuzzy PID Controller Optimized By Genetic Algorithms Used for a Single Phase Power Factor Pre-Regulator" isimli çalışmalarında; tek faz güç faktörü için optimize edilmiş bulanık PID kontrolör kullanmışlardır. Giriş üyelik fonksiyonları, çıkış üyelik fonksiyonları ve bulanık kontrolör müdahale mekanizması kuralları seçim ve optimizasyonları GA ile yapılmıştır. Seçim ve optimizasyon kriteri zaman domeni üzerinde; cevap süresi, üst aşım oranına göre alınmıştır. Benzetim sonuçları bulanık kontrolör üyelik fonksiyonları optimizasyonunda ilgi çekici bir metod olduğunu göstermiştir (Qin ve Du, 1997).

Valarmathi, Devaraj ve Radhakrishnan, "Genetik Algoritma ve Sugeno Bulanık Mantık Temelli online ph ayarlama" adını verdikleri çalışmada; PID için optimal parametre değerlerinin ayarlanmasının PID tasarımında en önemli süreç olduğunu göstermiş, ph ayarlama sürecinde optimal PID kontrolörün GA ile parametreleri ayarlanmış Tekagi-Sugeno tipteki bulanık kontrolör ile tasarlandığı gösterilmiştir. Geliştirilen bulanık kontrolör PID parametrelerini birçok farklı ortam şartı için eş zamanlı olarak değiştirebilmektedir. Bilgisayar üzerinde benzetim sonuçları ile uygulamanın başarılı olduğu izlenmiştir (Valarmathi vd., 2008).

Literatür taramasında GA destekli bulanık mantık kontrolör (BMK) tasarımını ele alan makalelere yer verilmiştir. Yapılan araştırmalara rağmen bulanık parametreleri için (üyelik fonksiyonları tipleri ve sınırlarının belirlenmesi, kural tabanının belirlenmesi) için sistematik bir ayar yöntemi geliştirilmemiştir (Küçükdemiral, 2002).

Birçok makalede kullanılan sistemler ya doğrusal yada tamamen doğrusal olmayan kontrol sistemleri olmuştur. Yapılan tez çalışmasında öncelikli olarak bu eksiklik büyük ölçüde ortadan kaldırılmaya çalışılmıştır. Uygulama yapılan bütün sistemler doğrusal olmayan sistemlere ait transfer fonksiyonları seçilmiştir. FPID kontrolörlerin optimal tasarımı metodu üzerine yapılan bu tez çalışmasında; analitik olarak bağlantılar oluşturduktan sonra GA ile optimum çözüme ulaşmak amaçlandı. Sadece üyelik fonksiyonları sınırlarının optimizasyonu ve yeniden belirlenmesi ile yetinilmemiş; aynı zamanda kural tabanı optimizasyonu da

yapılan programa eklenmiştir. Hem GA üzerindeki uygunluk fonksiyonlarında hemde mevcut sistemleri birbiri ile karşılaştırmak için geleneksel PID kontrolörün IAE kriteri performans ölçütü olarak alınmıştır. Oluşturulan kontrolörün uygulaması doğrusal olmayan sistemlere yapılmıştır. Bu çalışma kapsamında ortaya çıkarılan, GA tabanlı optimal bulanık PID kontrolör tasarım stratejisi kontrol mühendisliğinde geniş kapsamlı kullanılabilecek bir algoritmadır.

Yüksek lisans tezi çalışması beş bölümden oluşmaktadır. Giriş bölümünde; çalışmanın konusu ve çalışmaya konu olan problemin çözümüne yönelik önceden yapılan çalışmalar hakkındaki araştırmalar ve bu çalışmaların içeriği belirtilmiştir. İkinci bölümde, bulanık mantık ve bulanık çıkarım sistemleri bileşenlerinin tanımları yapılmıştır. Üçüncü bölümde GA ve algoritmada kullanılan operatörlerin tanımları yapılmıştır. Son bölümde, yapılan uygulama farklı parametreleri için test edilerek performansları karşılaştırılmış; uygulama kaynak kodu ve sonuçları verilmiştir.

2. BULANIK MANTIK

2.1 Bulanık Mantık Tarihçesi

İnsan hayatı Aristo Mantığı yani keskin mantık ile ifade edilen 0 (yanlış) ve 1 (doğru)'den mi ibaret? Yoksa kalbin elektriksel aktivite grafiğinde olduğu gibi veya sinüzoidal bir işaret gibi mi? Yaşadığımız çevre ve hayat sadece "siyah" ve "beyaz" üzerine kurulmamıştır. Evrende gri renkler de vardır ve bulanık mantık çalışmalarının amacı doğadaki gri tonların bilim ve endüstride kullanılmasını sağlamaktır. Günlük hayatımızı kapsayan bir çok olay klasik mantığın belirttiği gibi net ve kesin çizgiler ile birbirinden ayrılmamaktadır. Olaylar ve durumlar bu kadar belirgin, doğrusal ve net değildir. Belirsizliklerin olduğu, karmaşık, doğrusal olmayan durumlar konusu olmaktadır. Bu belirsiz durumlar ve olaylar insanlar tarafından oluşturulan ve geliştirilen çeşitli kabuller ile giderilmekte yada tahmin edilmektedir. Azeri bilimadamı Lütfi Asker Zadeh'in araştırmaları ile derinleşen FCL günümüzde sıkça kullanılabilir duruma geldi.

M.Ö. IV. yüzyılda yaşamış olan Yunan filozofu "Aristoteles" mantık bilimine ve yorumuna verilen isimdir. Aristo mantığın kurucusu olarak kabul edilir. "Organon" isimli kitabı Aristo mantığı yani keskin mantık üzerinedir. Aristo mantığı iki değerli mantıktır. Aristo'nun büyük araştırmaları ve çabaları sonucu ortaya çıkan ve "Düşüncenin Kanunu" adı verilen teorisidir. Bu teoriye göre hayattaki herhangi bir öneri "doğru" yada "yanlış" tır. Örneğin; Bütün insanlar ölümlüdür. Sokrates bir insandır. Öyleyse Sokrates ölümlüdür.

Polonyalı bilim adamı Jan Lukasiewicz, çok değerli mantık adına öncü araştırmacı oldu. 1917 yılında ortaya attığı üç değerli önermeler hesabı ilk kez Aristoteles'in 0 ve 1'lerden oluşan ikili mantığına alternatif bir sistem oluşturdu. Jan Lukasiewicz bu araştırmasını 4, 5, 6 üzerinde değerleri de geliştirdi. Sonuç olarak buna Lukasiewicz mantığı adı verildi. Birçok matematikçi Jan Lukasiewicz değerlerinin nümerik olarak ifade etmeye çalışsalar da Lütfi A. Zadeh [0.0 1.0] aralığını sonsuz değerler ile gösteren "Bulanık Mantık" uygulaması kadar başarılı olmamıştır.

Bulanık mantık anlam olarak çok değerli mantıktır ve klasik mantıktaki 0, 1 aralığında sonsuz değer alır. Bulanık mantık, klasik mantıkdaki ve, veya, değil gibi tüm işlemleri içerir. Bu nedenle bulanık mantık klasik mantığa da uzanır "doğru" veya "yanlış" arasında "kısmen doğru" veya "kısmen yanlış" sonuçlarını içerir. Üyelik derecesi [0, 1] arasında herhangi bir gerçek sayı degerini alabilir. Bulanık mantık insan mantığı karakteristigine benzerdir ve

doğrulugun genel kavramı “bulanık” lık ile ifade eder (L. A. Zadeh, 1965).

İşte bu tanımdan ötürü bunu başarmak Lütfi A. Zadeh için kolay olmamıştır. Zadeh’in bu araştırmalarını radikal fikirlerinden dolayı hiçbir teknik dergi yayınlamayı kabul etmemiştir. Çünkü keskin mantığa belirgin bir alternatif oluşturmuştur. Zadeh daha sonraları bulanık mantık ve düşünüşün temelini oluşturan Bulanık Algoritması’nı ileri sürmüştür. Ardından “Bulanık Küme” ifadesi 1964 yılında California Berkeley Universitesinde sunulmuştu ve sonrasında makale ‘The Information and Control Jurnal’ dergisinde yayınlanmıstır. 1972 yılında Michio Sugeno bulanık ölçüm ve integral kavramlarıyla bulanık mantık konusuna yeni anlayış getirmiştir. 1974 yılında, Ebraham Mamdani bulanık mantığı ilk kez bir buhar makinesinin kontrol aşamasında kullanmıştır. Şu anda Mamdani en yaygın olarak kullanılan tiptir (Ross, 1995). Endüstrideki PID kontrolörlere ani pikleri önlemek için bulanık mantık fonksiyonu eklenmiştir. 2000 yılından sonra birçok endüstriyel kontrolör üreticileri bulanık kontrol yazılımlarını piyasaya sürmüşlerdir.

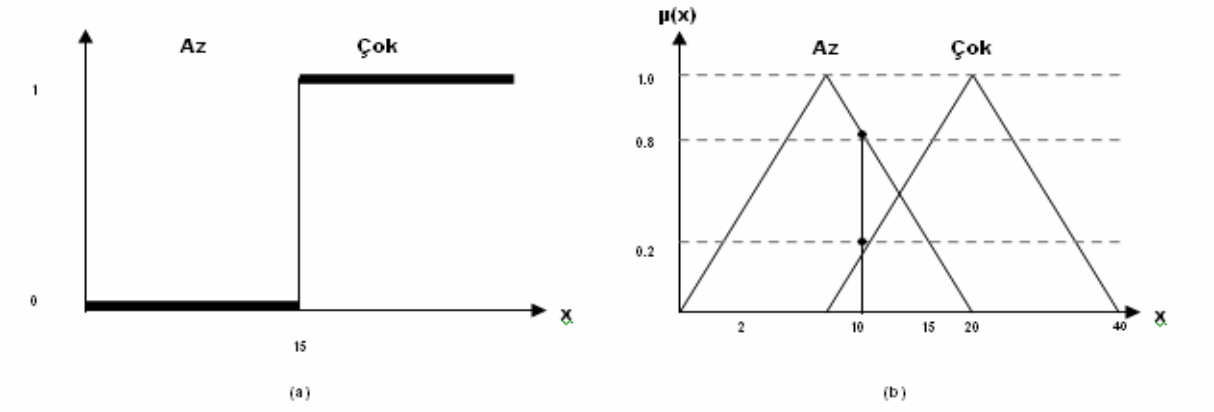
2.2 Bulanık Mantık Tanımı

Doğrusallık belirli sistemlerde sıkça karşımıza çıksa da belirsiz, karmaşık, doğrusal olmayan olaylar hayatın tamamına yakınını kapsar. Bilimsel araştırmalarda bunun için varsayımlar ve kabuller ile veya sistemler lineerleştirilerek bunun üstesinden gelmeye çalışırız. Sistem hakkında yeterli bilginin olmadığı, girişlerin sensör yerine insan tarafından alındığında ve çıkışlar için net bir müdahale yerine dilsel değişken adı verilen komutlar ile işlem yapılması gerektiğinde “bulanık mantık” bizim için biçilmiş kaftan vazifesini görür.

Bulanık mantıkta keskin sınırlar yoktur. Olasılık alanı değildir; olayların oluşum olasılığından çok oluşum derecesi ile ilgilenilir. Bulanık mantık günlük hayatta kullandığımız dilsel terimlere dönüştürme işlemlerini gerçekleştirerek sistemi niteliksel olarak tanımlayarak dünya ile ilişki kurar.

Oluşum derecesi olarak ifade ettiğimiz konu bulanık küme teorisinin temelini oluşturur. Bulanık küme teorisinde bulanık kümenin sınırları kesin ve keskin degildir. Bulunan elemanların bulanık kümelere ait olma miktarları üyelik dereceleri ile belirlenir. Dolayısıyla bulanık kümeler değişik üyelik derecesine sahip elemanları birarada içerebilmektedir. Bulanık küme teorisinde elemanlar birden fazla kümeye farklı derecelerde ait olabilirler. Klasik mantıkta “az” veya “çok” ile ayrılan sınırlar bulanık mantıkta üyelik fonksiyonları ve üyelik dereceleri ile belirlenir.

Örneğin; Bir kullanıcıya aylık enerji tüketimi hakkında yöneltilen soruya karşılık verilen cevap "az" olduğunda ortada bir belirsizlik olacaktır. Verilen cevabın 10 kw/saat olması herkes tarafından anlaşılacak sayısal bir değerdir. 10kw/saat değeri keskin mantıkta 15kw /saat altında olduğu için "az" kümesine ait olmaktadır. Bulanık mantıkta ise bu ifade iki küme içinde yer alabilir. "Az" kümesinde üyelik derecesi 0.8 iken "çok" kümesindeki üyelik derecesi 0.2 olarak gösterilir. Aşağıdaki şekilde örneğe ait klasik mantık ve bulanık mantık için belirtilen sınır ve üyelik dereceleri gösterilmiştir.



Şekil 2.1 (a) Keskin mantık (b) bulanık mantık

Bulanık küme ilişkilerindeki işlemlerin klasik kümelerdeki ilişkilerindeki işlemlerden farkı, işlem sırasında üyelik derecelerinin 0 ve 1 gibi iki değerli olmayıp, 0 ile 1 aralığında herhangi bir ondalık değeri de içermesidir. Küme teorisinde {0,1} şeklinde gösterilen yapı, bulanık küme teorisinde [0 1] olarak gösterilir.

Matematiksel olarak küme, kendisine ait olan ve olmayan elemanların kesin olarak bilindiği topluluktur. Küme setlerine ait elemanlar $\chi_A(x)$ şeklinde gösterilir.

Örnek: Üç elemanlı bir A kümesinin liste gösterimi

A = {3,4,5 }

Bulanık mantık küme teorisinde; uzayda x gibi herhangi bir eleman, A bulanık kümesine belli aitlikle üye ise, x elemanın bulanık kümeye aitlik miktarı üyelik derecesi olarak adlandırılır, $\mu_A(x)$ ile gösterilir. Eğer $\mu_A(x)$ =1 ise; "x, A kümesine tamamen aittir" anlamına gelir. (2.1) eşitliğinde üyelik fonksiyonu sürekli (2.2) eşitliğinde ayrık gösterim verilmiştir.

$$A = \sum \frac{\mu_A(x)}{x} \tag{2.1}$$

$$A = \int \frac{\mu_A(x)}{x} \tag{2.2}$$

Eğer A $R \in (-\infty,+\infty)$' da, söz konusu kümenin bir elemanı ise $\mu_A(x)$ üyelik fonksiyonu $R \to [0,1]$ aralığında oluşur. Diğer bir deyişle A kümesi $A = [a_1, a_2]$ aralığında ise genel olarak $\mu_A(x)$ üyelik fonksiyonu (2.3) kullanılarak gösterilir (Wang, 1997).

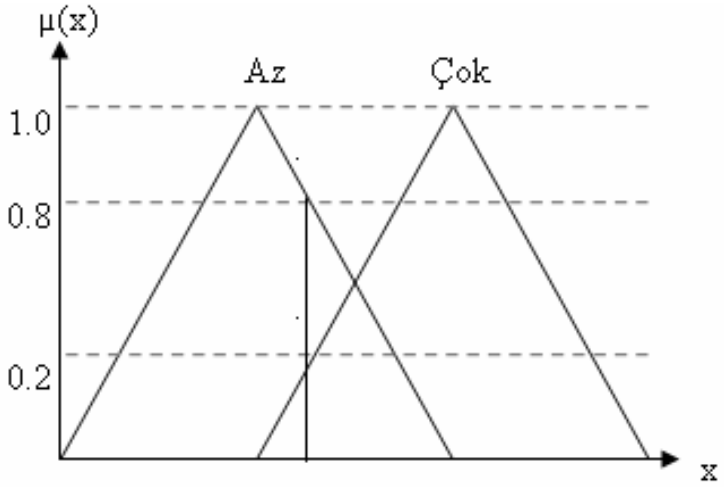
$$\mu_A(x) = \begin{cases} 0, & x < a_1 \\ 1, & a_1 \le x \le a_2 \\ 0 & x > a_2 \end{cases} \tag{2.3}$$

Birleşme $\mu_A U_B(x) = \mu_A(x) \text{ V } \mu_B(x)$

Kesişim $\mu_A \cap_B(x) = \mu_A(x) /\backslash \mu_B(x)$ (2.4)

Ters $\mu_A(x) = 1 - \mu_A(x)$

(2.4) eşitliğinde belirtilen özelliklerde, V ile gösterilen sembol maksimum alma operatörü ve /\ ile gösterilen sembol ise minimum alma operatördür. Bulanık mantık teorisinde kesişim minimum, birleşim maksimum işlemini ifade eder (Ross 1995).



Şekil 2.2 Bulanık mantıkta kesişim ve birleşim

2.3 Bulanık Mantık Avantaj ve Dezavantajları

Bulanık Mantık Avantajları:

- Matematiksel olarak modellemek zor veya imkansız olan sistemlerde, zamanla durumları değişen ve doğrusal olmayan sistemlerde başarılı sonuçlar verirler. Bulanık mantık denetimi; doğrusal olmayan sistemlerde, sistem analizinin klasik yöntemlerle yapılamadığı, sisteme ait bilgilerin niteliklerin belirsiz ve kesin olmadığı durumlarda uygulamak çok uygun olmaktadır.
- Sistem hakkındaki edinilen bilgiler matematiksel değilde dilsel değişkenler ise ve

klasik kontrolör tasarlamanın zor olduğu durumlarda uygulamak avantajlı olacaktır.

- Sistem çok seri şekilde modellenebilir çünkü kontrolör tasarımı yapılırken sistemin matematiksel modeline ihtiyaç yoktur; tamamen tasarımcının (operatörün) tecrübelerine dayalı bir kontrolör oluşturulur.
- Bulanık kontrol insa özgü olan kontrol stratejisini taklit ettiğinden dolayı (insan düşünme tarzına yakın olması) girişler ile çıkışlar arasında rahatça anlaşılabilen bağlantı kurar. Tüm değişkenler herkes tarafından algılanabilecek şekilde oluşturulabilir.
- Bulanık mantık uygulamaları klasik kontrol sistemleri ile uygulanabilir. Bu şekilde kullanılması en başarılı sonucu verecektir. Örneğin PID katsayı ayarlama işleminin bulanık kontrolör ile yapılması.
- Bulanık kontrolör üzerindeki parametrelere ekleme yapmak veya çıkarmak üyelik fonksiyonu toplam sayılarını değiştirmek, deneme/yanılma yöntemini kullanmak vasıtası ile sisteme özel çok başarılı doğrusal olmayan kontrolörler oluşturmak mümkündür.
- Bulanık mantık kullanılması sayesinde tam olarak bilinmeyen veya eksik girilen bilgiler ile sistem üzerinde çalışmak, uygulama yapmak ve sonuç almak son derece rahattır.
- Bilimadamları neden/sonuç ilişkisi ile ilgilendiklerinden FLC matematiksel temelinin zayıf olmasından ötürü kullanmaktan çekinmektedirler.
- Tasarım sürecinin kısa olması, yapılan işlemlerin basitliği, sisteme uygulamanın kolay olması ve yazılımının ucuz olması sebebiyle bulanık mantık ile kontrol düşük maliyetli bir kontrol yöntemidir.
- Sistem cevabındaki yüksek osilasyonlarını ve çıkıştaki aşırı sapmaları gidermek için bulanık kontrolöre daha fazla kural eklenmesi ve daha fazla üyelik fonksiyonu ile gidermek mümkündür.
- Bulanık mantık, tam olarak bilinmeyen veya eksik girilen bilgilere göre işlem yapma yeteneğine sahiptir. Deney ve deneyime dayanan metotdur. Kontrol edilecek sistemin matematiksel modeline ihtiyaç duymaz. Günümüzde doğrusal olmayan kontrol sistemleri ile sürekli uğraştığımız düşünülürse bulanık mantık verimli sonuç vermektedir.
- BMK tamamen tasarımcının ürünü olduğu için (kendi dilinde oluşturduğu değişken, kurallar ve fonksiyonlar); buna ait kontrol algoritması kolayca değiştirilebilir esnek bir yapıdadır (Sugeno, 1985).

Bulanık Mantık Dezavantajları:

- BMK genel olarak sisteme özeldir ve başka bir sisteme uygulanması neredeyse imkansızdır. Çünkü üyelik fonksiyonları sınır değerleri sisteme özel olması ve öğrenme yeteneğinin olmamasıdır.
- Sistem karalılık analizi, gözlemlenebilirlilik ve kontrol edilebilirlilik analizi yapılmasında ispat edilmiş matematiksel bir yöntem yoktur. Bulanık mantık denetim sisteminin kararlılığını sağlayabilecek model ancak geleneksel kontrol sistemleri ile birlikte kullanmak ile mümkün olabilir.
- Kolay tasarım yapılabilmesine karşın; sistem üzerinde uygulama ile deneme/yanılma yapılmadan hangi üyelik işlevinin ve hangi sayıda üyelik işlevlerinin kullanılması gerektiğini belirlemek çok zordur.
- Bulanık kontrolör tasarımında sistem matematiksel modeline ihtiyaç olmadığından; üyelik fonksiyonu ve sınırları belirleme, giriş/çıkış bilgileri, bulanık kurallar ile ilgili olarak uzman sezileri, bilgi ve deneyimlerine ihtiyaç bulunmaktadır. Örneğin giriş fonksiyonunun üyelik fonksiyonuna ait sınır değerinin yanlış belirtilmesi durumunda sistem kararsızlığa gidecektir.
- PID üzerinde kullanılan paramatre ayarlama işlemi bulanık kontrolörler için yeni araştırma konusu olmuştur ve online uygulaması çok zordur. Şu anki araştırmalarda sistem üzerinde offline çalıştırma ile sisteme online olarak uygulaması yapılabilir.
- Bulanık mantık kontrocüler genelde sisteme özeldir ve başka bir sisteme uygulanması neredeyse imkansızdır. Çünkü üyelik fonksiyonları sınır değerleri sisteme özel olması ve öğrenme yeteneğinin olmamasıdır.

2.4 Bulanık Mantık Uygulama Alanları

Bulanık mantığın kullanım alanlarının linear olmayan kontrol sistemleri için olduğunu ve dünyada birçok olayın doğrusal olmadığını düşünürsek bulanık mantık kullanım alanının ne kadar geniş olduğunu anlarız. Yazılım kolaylığı ve işlemci ile birlikte ek hafıza gerektirmeyen bulanık mantık popülaritesi günümüzde yükselmiştir.

LG (Kore) , Arçelik ve Vestel (Türkiye) Bulanık mantık Çamaşır makineleri üretmektedirler. Bulanık Mantık Kontrol Sistemli çamaşırları özelliklerine göre ideal yıkama programını kendisi belirliyor. Çamaşırların özelliklerine ve yıkanma ısısına uygun olmayan bir program seçilmesine izin vermiyor, ayrıca kullanıcı hatalarını belirten kod sistemi ile kişileri uyarıyor. Çamaşır miktarına göre su tüketimini ayarlayan sistemi ise ekonomik kullanım sunuyor.

Aalborg White (Danimarka) ve Condor Holding (Hollanda), bulanık mantık çimento döner fırını kullanmaktadır. Çimento üretim alanında değirmen içerisindeki sıcaklık ve oksijen oranı kaliteli bir ürün elde etmek için son derece önemlidir. Ayrıca ısı ve karbonmonoksit oranı gibi bilgilerin gerçekliği ve kontrolü iyi bir çalışma mantığı için gereklidir.

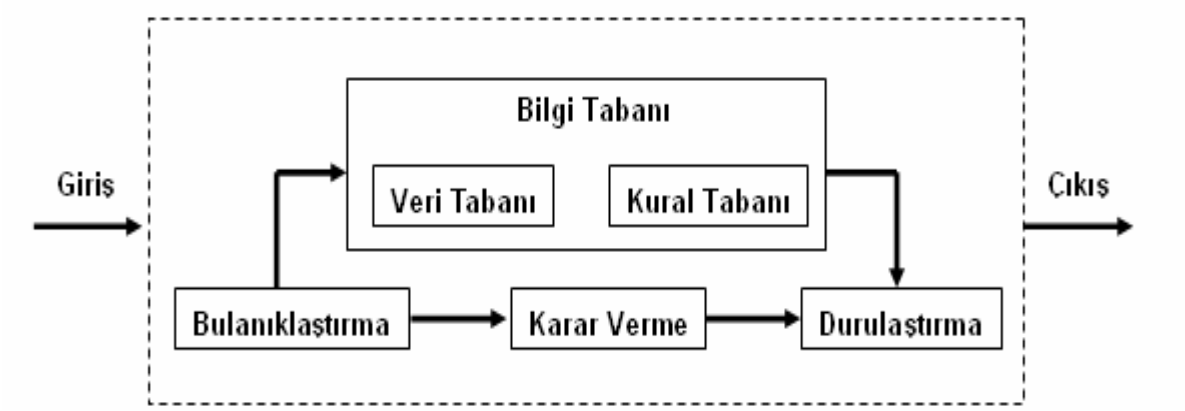
Hitachi (Japonya), Sendai Metrosu bulanık metro fren sistemi, Sendai' de kullanılan gerek insan sürücüler, gerekse geleneksel otomatik kontrolörden daha iyi çalısan, metro freni kontrolüdür. Sendai Metrosu kontrolünün böyle bir bulanık mantık sistemine terk edilmesinden sonra, bu alandaki çalısmalarda ve uygulama alanlarında son derece büyük bir artış olmuştur.

Günümüzde Japonya firması OMRON bu konuda öncü durumdadır. 1987 yılında en hızlı bulanık mantık kontrolörü üreten firmadır. İnsansı yada insan-gibi adını verilen bulanık mantık tekniğini OMRON firması iklimlendirme sistemlerinde bulunan sıcaklık kontrolünden, bilekten tansiyon ölçen cihazlara kadar birçok ürününde kullanmaktadır.

Omron firması aynı zamanda bulanık mantık kontrol yazılımı üretmektedir. “Fuzzy Support Software C500-SU981-E” isimli bulanık kontrol yazılımları ile oluşturulan bulanık mantık konrol blokları PLC (Programmable Logic Controller – Programlanabilir Mantık Kontrolör) üzerinde kullanılarak proseslere bulanık kontrol yapılması sağlanmaktadır. Aynı zamanda karma bloklar oluşturulup hem doğrusal hem doğrusal olmayan kontrol beraber sağlanabilir.

2.5 Bulanık Çıkarım Sistemleri

Fuzzy Inference Systems (FIS) - bulanık çıkarım sistemleri veya bulanık kontrolör modeli olarak literatürde anılmaktadır. FIS: bulanıklaştırma, karar verme, durulaştırma, bilgi olmak üzere dört ana bloktan oluşmaktadır.



Şekil 2.3 Bulanık çıkarım sistemi, mamdani tipi genel yapısı

Bulanıklaştırma : Ham girişleri dilsel değerler ile derecendirilmiş hallerine dönüştürür.

Bilgi Tabanı : İki bölümden oluşur; veri tabanı ve kural tabanı)

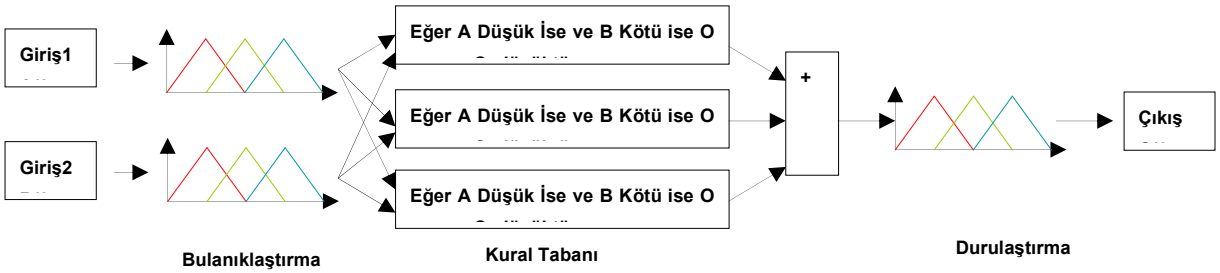
Veri tabanı : Bulanık kümelerin (g/ç) üyelik fonksiyonlarını ve sınırlarını içerir.

Kural tabanı : Kural tablolarını ve yapısını içerir

Karar verme : Kurallardaki çıkarım işlemlerinin gerçeklenmesidir.

Durulaştırma : Bulanık çıkarım ardından oluşan değerleri sayısal çıktılara dönüştürür.

Bulanık mantık ile kontrolör tasarlanırken ilk yapılması gereken giriş ve çıkış değişkenlerini tanımlamaktır. Sistemde kaç adet giriş ve kaç adet çıkış olacağı belirlenirken farklı kombinasyonlar seçilme esnekliği varır: MISO (Multiple Input Single Output, Çok giriş tek çıkış), MIMO (Multiple Input Multiple Output, Çok Giriş Çok Çıkış) Ardından bulanık alt kümelerin her bir değiskeni için belirli bir aralık (range-sınır) tanımlanır ve her birine dilsel etiket atanır. Daha sonra herbir bulanık alt küme için üyelik fonksiyonu belirlenir. Giriş ve durum değişkenlerine ait bulanık alt kümeler ile giriş deşişkenine ait alt kümeler arasında bulanık ilişkiler kurulur. Kontrolör tarafından girişler bulanıklaştırılır. Önceden belirlenmiş olan bulanık kurallar kullanılarak bulanık çıkarım yapılır. Bulanık kurallar tarafından belirlenen bulanık çıkışlardan tek bir bulanık değer elde edilir. Durulama yapılır ve keskin çıkış değeri elde edilir (Babaoğlan, 2009).



Şekil 2.4 Bulanık çıkarım sistemleri işlev şeması

2.6 Bulanıklaştırma

Sistem girişleri genellikle sayısal bir değer olur. Fakat bulanık mantık konusunda sözü geçen "bulanık" olma işleminin uygulanabilmesi için giriş değişkenlerinin bulanık olması gerekmektedir. Bu sebeple alınan girişler bulanıklaştırma işlemine tabi tutulurlar. Bulanıklaştırma sisteme sayısal olarak alınan giriş değerlerinin bilgilerine dilsel değişkenler oluşturulup üyelik derecelerinin atanması işlemleridir. Üyelik fonksiyonları vasıtası ile sistemde okunan bilgilerin ait olduğu fonksiyonlar/üyelik dereceleri tespit edilir ve sayısal olarak girilmiş değişkene çok yüksek , yüksek gibi değişkenler atanır. Sayısal giriş değişkeni

keskin değer olarakta adlandırılmaktadır. Bulanıklaştırma sonucunda elde edilen değişkenlere "dilsel değişkenler" adı verir. Dilsel değişken sayısal değerleri dilsel değerlere dönüştürür. Tüm bu bulanıklaştırma işleminin yapılabilmesi için her bir giriş için üyelik fonksiyonlarının oluşturulması gerekmektedir (Şen, 2000).

Bulanıklaştırma sırasında sayısal bilgiler belirli aralıklara bölünerek üyelik fonksiyonları ile ifade edilirler. Her üyelik fonksiyonu için ona ait geometrik bir şekli ve sözsel ifadesi vardır. Bu üyelik fonksiyonları belirli oranda iç-içe geçmiş şekilde de oluşturulmaktadırlar. Bazı durumlarda ki tezimizde bunu uyguladık üyelik fonksiyonları üzerinde normalleştirme şartı uygulanır.

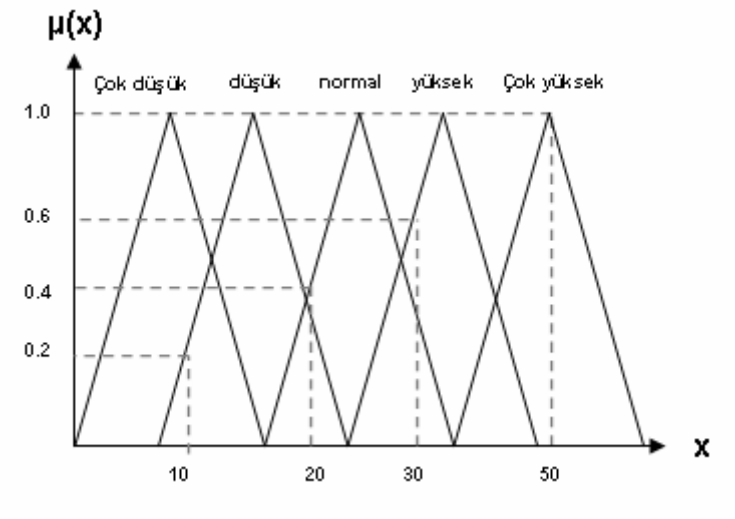
Bulanıklaştırmada şu adımlar izlenir:

- Giriş değişkenlerinin tartıp ölçülendirmek
- Belirtilen kümedeki değişkenlere karşılık giriş değişkenlerinin oranlama/skalalandırma işlemini yapmak
- Bulanıklaştırma birimine ait üyelik fonksiyonu etiketlerinin uygun dilsel değişkenlere dönüşüp atanmasını sağlamak.

Bulanıklaştırma işlemi bu açıklamalara göre kolay anlaşılır ve uygulanabilir gözükmektedir. Ama, önceden tekrar tekrar belirtilen uzman tectübeleri ve sezileri ihtiyacından dolayı bu işlemlerin yapılması kolay değildir. Sistemi işleten operatörlerin sistemde uyguladığı seçimler ve kontrol davranışı, sistemin matematiksel modelinden daha önemlidir. Bundan dolayıdır ki bulanıklaştırma aşamasına gelene kadar geçen süre normalden çok fazla olabilir.

Bulanıklaştırma işlemi sistemin tüm girişlerini ve çıkışlarını dilsel değişken adı verilen sözlerle ifade etmeyi ve çok karmaşık sistemlerin rahat şekilde herkes tarafından alaşılabilmesini sağlar. Bulanıklaştırma üzerinde yapılan girdi tekli şekilde olabilir. Önceki konuda üyelik fonksiyonlarının açıklaması yapılırken belirtildiği gibi seçilen fonksiyonlar sisteme özeldir, değişkenlik gösterebilir. Bu örnek için üçgen seçildiği gibi yamuk, çan eğrisi gibi çok değişik üyelik fonksiyonları seçilebilir. Giriş değerleri birden fazla üyelik fonksiyonlarında da değer alabilirler. Üyelik fonksiyonlarında kullanılacak etiket sayısı 3, 5, 7 olabilir. Yyazılım yapısı açısından tekil olmasının faydalı olduğu belirtilmiştir ve kullanıcı tarafından en uygun olanı seçilir. Dikkat edilmesi gereken nokta aynı değer birden farklı üyelik fonksiyonu ile kesişmekte fakat farklı üyelik dereceleri denk gelmektedir. Aşağıdaki şekilde beş etiketli (çok düşük, düşük, normal, yüksek, çok yüksek) BMK ait g/ç için üçgen

üyelik fonksiyonu ve sayısal değere karşılık gelen üyelik dereceleri gösterilmiştir (Karaoğlan, 2007).



Şekil 2.5 Üyelik fonksiyonu üzerinde derecelerin atanması

Şekil2.5'te gösterilen y-ekseni üzerinde gösterilen kısım üyelik derecesi {μ} olarak adlandırılmaktadır. Bu kısımnda sayısal giriş değerlerinin her üyelik fonksiyonu için etiketlere karşılık düşen değerleri temsil eder. Bu çalışma kapsamında üç adet etiket kullanılacaktır (pozitif, sıfır, negatif).

2.7 Bilgi Tabanı

Bulanık mantık sistemlerinin temellerinden olan bilgi tabanı; karar verme amaçlı oluşturulan birimi kural tabanının da kullandığı bilgileri aldığı veri tabanı(data base) ve kontrol amaçlarına uygun dilsel kontrol kurallarının bulunduğu kural tabanı(rule base) olmak üzere iki blok şeklindedir (Cordon ve diğerleri, 2001).

Edinilen bütün sonuçlar "çıkarım" ünitesi karar verme işlemlerinde bilgi tabanına getirilip; veri tabanı üzerinden üyelik fonksiyonlarıyla ilgili bilgileri, kural tabanından ise kombinasyonu oluşturulmuş değişik giriş değerleri için tespit edilmiş olan çıkış bilgilerini alır. Bu bakımdan kontrolör içerisinde bilgi tabanı ve çıkarım birimi sürekli haberleşme halindedir.

Bulanık kontrolör tasarımındaki önemli birimlerden olan bilgi tabanıdır. Bundan sonraki aşama ise bulanıklaştırma, karar verme ve durulaştırma birimlerine ait üyelik fonksiyonu için gerekli bilgileri sağlayan blok "veri tabanı"dır. Bulanık çıkarım yapmak için gerekli

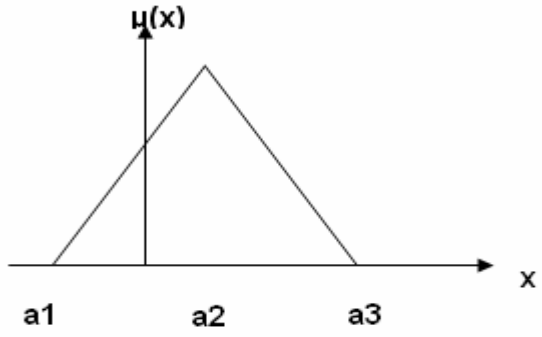
önermeler topluluğunun bulunduğu blok "kural tabanı"dır. Oluşturulmak istenen kurallar; sistemden elde edilen deneyimler, yapay sinir ağları veya genetik algoritmalar yoluyla elde edilebilir.

2.8 Veri Tabanı

BMK üzerinde yapılması gereken tüm işlemler için gerekli işlem üyelik fonksiyonlarını, dilsel niteleyicileri içeren blok veri tabanıdır. FLC uygulanacak sisteme ait bulanıklaştırma, bulanık çıkarım, durulama işlemleri sırasında ihtiyaç olan üyelik fonksiyonu ve kural tablosu bilgileri veri tabanı bloğundan sağlanır.

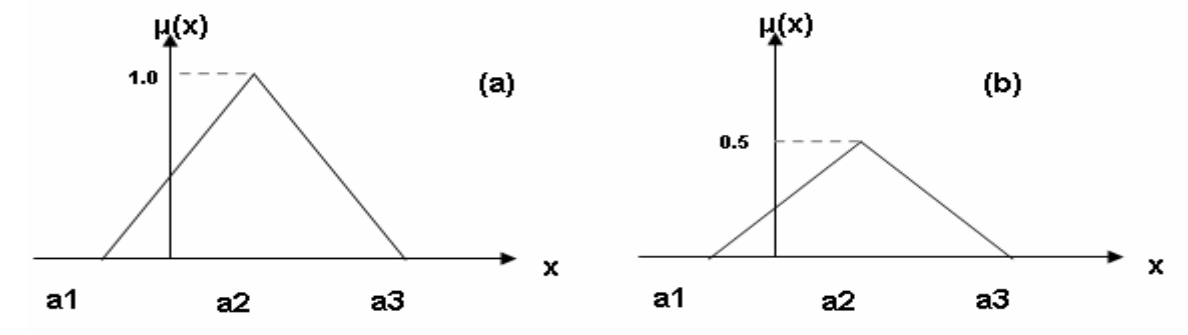
Kontrol edilen sistemdeki değişkenler oluşturulduktan sonra bir bulanık kelime veya ifadenin temsil ettiği sayısal aralık, ifade hakkında bilgi sahibi olan uzman kişi veya kişiler tarafından belirlenebilir. Örnek ile açıklamak gerekirse; PID kontrolü yapılacak bir sistem ile ilgili oransal kazanç değişkeni aralığı -50 ve +50 aralığında belirtilebilir. Bu aralık K_p değişkeni ile ilgili olarak üyelik fonksiyonları için alacağı minimum ve maksimum değeri belirtir (K_p uzayı). Ama bu aralığı konuşmada alt aralıklar ile ifade etmek gerekirse: çok pozitif, sıfır ve negatif şeklinde dilsel değişkenler oluşur. Bulanık kümelerde elemanlar birbiri ile kesin sınırlar ile ayrılmazlar ama aynı zamanda birbirleri ile örtüşmezler. Alt kümelere ait sınırlar söz konusu olursa negatif için -50 ve 0, sıfır için -25 ve +25 son olarak pozitif için 0 ve 50 aralığı belirlenebilir.

Bulanık mantık kümeler teorisinde elemanlar üyelik derecesi ile gösteriliyordu. Kümeye ait elemanların üyelik derecelerini ve buna karşılık gelen değerlerini gösteren şekle üyelik fonksiyonu (membership function) adı verilir. Doğrusal olmayan sistemlerdeki uygulamalarda yaygın olark kullanılan üyelik fonksiyonu çeşitleri, üçgen (triangle), çan (gauss) eğrisi ve yamuk (trapez) fonksiyon çeşitleridir. Her üyelik fonksiyonu kendisine ait matematiksel ifade ile elde edilir (Cordon ve diğerleri, 2001).



Şekil 2.6 Üçgen üyelik fonksiyonu

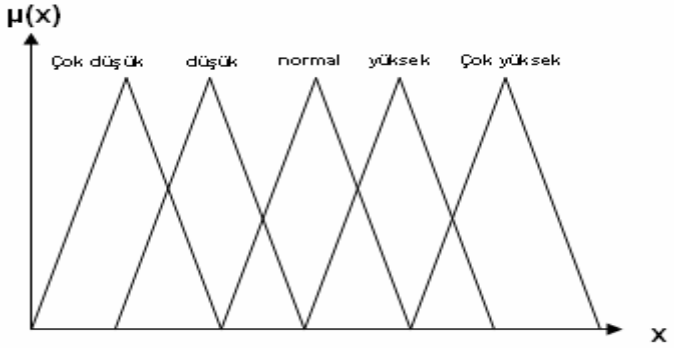
FIS üzerinde üçgen üyelik fonksiyonu a1, a2 ve a3 olmak üzere üç adet değişken ile tanımlanır. Matematiksel "dış bükey" olma özelliğine uyulması koşuluna göre öğelerin değişken değerleri a1 < a2 < a3 gibi bir sıralama bulunması gerekir. Üçgen üyelik fonksiyonu kullanılarak bir üyelik derecesinin hesaplanması, değişkenin değerine göre yapılır (Elmas, 2003). Aynı zamanda diğer bir bulanık mantık kuralı ise; kümenin normalleşmiş olmasıdır. Buda kümenin en az bir elemanının maksimum derecesinin "1" olması halidir.



Şekil 2.7 (a)Normalleşmiş üyelik fonksiyonu (b)normal olmayan üyelik fonksiyonu

$$\mu_A(x) = \begin{cases} 0, & x < a_1 \\ \dfrac{x - a_1}{a_2 - a_1}, & a_1 \le x \le a_2 \\ \dfrac{a_3 - x}{a_3 - a_2}, & a_2 \le x \le a_3 \\ 0, & x > a_3 \end{cases} \tag{2.5}$$

Bulanık mantık kullanırken her bir giriş ve çıkış için yaygın olarak üç adet (düşük, normal, yüksek) , beş adet (çok düşük, düşük, normal, yüksek, çok yüksek) veya yedi adet(çok fazla düşük, çok düşük, düşük, normal, yüksek, çok yüksek, çok fazla yüksek) olarak adlandırılan etiketler oluşturulur.



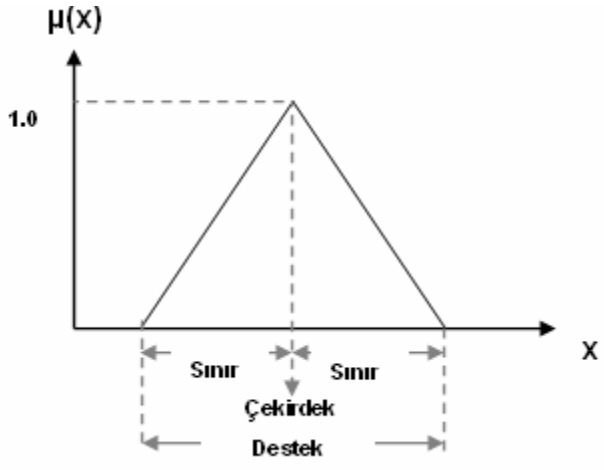
Şekil 2.8 Beş etiketli üçgen üyelik fonksiyonu

Bulanık mantık küme teorisine göre bir küme üç kısımdan oluşur: core (öz), support (dayanak) ve boundary (sınır).

Öz: Bir bulanık alt kümede, üyelik dereceleri "1" olan öğelerin toplandığı orta kısma öz denir. Burada yer alan tüm öğeler için $\mu_A(x) = 1$ olur.

Dayanak: Bir bulanık alt kümenin tüm öğelerini içeren aralığa dayanak adı verilir. Bulanık kümenin dayanak kısmında yer alan değişkenler, 0 ile 1 arasında üyelik derecesine sahiptir. Bu durum $\mu_A(x) > 0$ şeklinde ifade edilir.

Sınır: Bulanık alt kümenin üyelik derecesi 1'e veya 0'a eşit olmayan öğelerin oluşturduğu kısımlara üyelik fonksiyonuna ait sınırları veya üyelik fonksiyonun geçiş bölgeleri adı verilir. Bu kısımda yer alan elemanların tanımı $0 < \mu_A(x) < 1$ şeklindedir (Aras, 2010).



Şekil 2.9 Bulanık küme parçaları

2.9 Kural Tabanı

Bulanık mantık tanımında sürekli geçen uzman sezileri, bilgi ve deneyimleri bulanık mantık kontrol kurallarında ihtiyaç duyulan en önemli etkendir. Bulanık kuralların toplam sayısı ve kuralların doğruluğu sistem performansına doğrudan etkiyen faktörlerdir. Dilsel değişkenler kural tabanı kısmında belirli ifadeler halinde birleştirilerek bulanık kurallar meydana getirilir.

Uzman sezileri dışında kullanılan kural oluşturma şekilleri; sistem deneme yanılma ile kontrol edilerek yapılan işlemlerin daha sonrasında edinilen tecrübelerin kontrolöre aktarılması, yapay zeka teknikleri ve GA olarak sıralanabilir. Bulanık kural tablosu oluşturulurken üç kalıptan yararlanılır. Bunlar atama cümlesi, şart cümlesi ve şartsız cümleler.

Atama cümlesi bir durum değişkenine ait genel bir yargı yapar. Örneğin K_p oranı düşük. Şart cümlesi koşula bağlı olarak yargı yapılır. Örneğin eğer a küçük ise o halde b küçüktür. Şartsız cümle koşula bağlı olmadan yargı yapılır. Örneğin K_p değerini azalt. Bulanık kural tablolarında bulunan kurallar "şart cümlesi"dir. Bulanık kontrol tablosunda bulunan bulanık kurallar, "IF" ve "THEN " kuralları topluluğu ile temsil edilirler.

Örnek: Eğer Koşul1 ….Koşul n O Halde Çıkarım1 .... Çıkarım n

Önerme: Hata değeri yüksek iken sistemi osilasyona götürüyor, küçük hata değeri sistemi osilasyona götürmüyor.

Çıkarım: Eğer hata büyük ise küçük K_p değeri kullan (IF error BIG THEN K_p SMALL).

Bulanık mantık küme teorisinde bulanık ikişki matrisleri kontrolöre ait karar verme safhasında kullanılır. İlişki matrisi üyelik fonksiyonları matrisleri ve kendi matrislerinin kartezyen çarpımı şeklindedir. Bulanık ilişki matrislerinde iki uzay arasında olan ilişkiler karakteristik fonksiyonlar aracılığı ile ölçülmez, [0,1] değerleri arasında belirtilen üyelik fonksiyonları vasıtası ile belirlenir.

Dilsel değişkenlerin/kelimelerin sayısal değerlerini kapsayan üyelik fonksiyonları, dilsel değişkenlere ait yorumlardır. Bu kelime parçacıklarından manalı cümleler elde edebilmek için ve, veya, değil gibi klasik mantıkta da var olan kelimeler kullanılarak küme işlemleri yapılır. Kesişim alınması "minimum alma, /\" ile sağlanır (Babuška, 2001). (2.6) veya (2.7) gösterimlerinde olduğu gibi kesişim işlemi yapılır.

$$\mu_{\mathbf{A}} \cap_{\mathbf{B}} (\mathrm{x}) = \mu_{\mathbf{A}} (\mathrm{x}) / \backslash \mu_{\mathbf{B}} (\mathrm{y}) \tag{2.6}$$

$$\min[\mu_{\mathbf{A}} (\mathrm{x}), \mu_{\mathbf{B}} (\mathrm{y})] = \mu_{\mathbf{C}} (\mathrm{z}) \tag{2.7}$$

Çizelge 2.1 Örnek bulanık kural tablosu

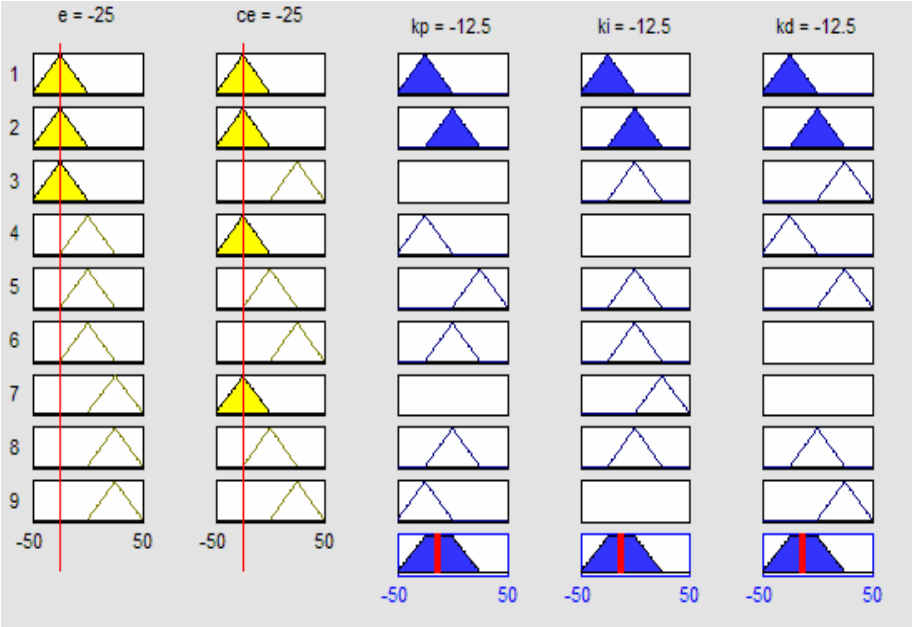
Koşul	Sonuç
Eğer (hata) Eksi ve (hatanın değişimi) Eksi O halde	(K_p) Eksi (K_i) Sıfır (K_d) Artı
Eğer (hata) Sıfır ve (hatanın değişimi) Eksi O halde	(K_p) Sıfır (K_i) Artı (K_d) Eksi
Eğer (hata) Sıfır ve (hatanın değişimi) Sıfır O halde	(K_p) Sıfır (K_i) Sıfır (K_d) Sıfır
Eğer (hata) Sıfır ve (hatanın değişimi) Artı O halde	(K_p) Sıfır (K_i) Eksi (K_d) Artı
Eğer (hata) Artı ve (hatanın değişimi) Eksi O halde	(K_p) Artı (K_i) Sıfır (K_d) Eksi
Eğer (hata) Artı ve (hatanın değişimi) Sıfır O halde	(K_p) Sıfır (K_i) Sıfır (K_d) Eksi
Eğer (hata) Artı ve (hatanın değişimi) Artı O halde	(K_p) Artı (K_i) Eksi (K_d) Sıfır

2.10 Bulanık Çıkarım

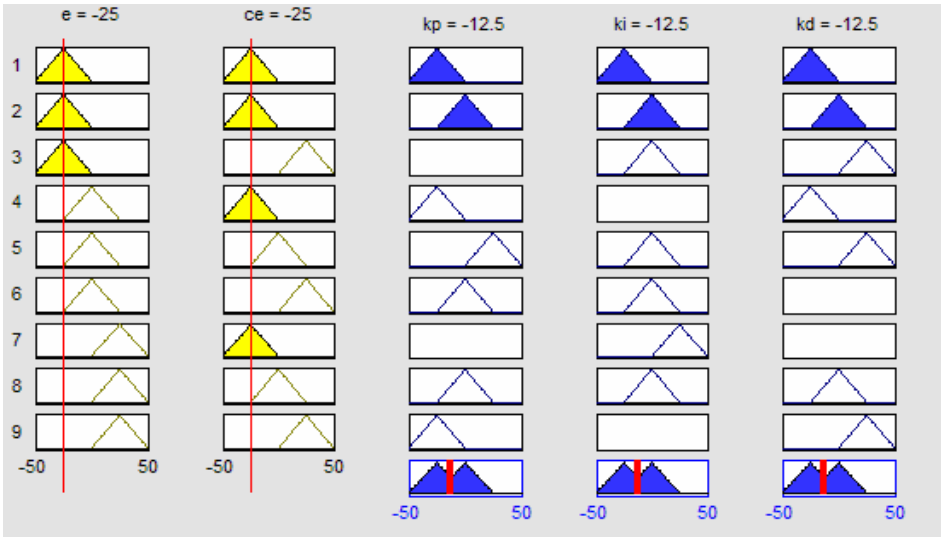
Tüm işlemlerin karara bağlanacağı bulanık mantığın can alıcı noktalarından biri de karar verme birimidir. Karar verme biriminin diğer bir adı da "çıkarım"dır. Sürekli bahsedilen insan gibi davranma insanı veya insan beynini taklit etme yeteneğinin en yoğun kullanıldığı çıkış değeri yada kümesinin oluşturulduğu bölümdür. Çıkarım bloğu çıkarım ve toplama adlı iki birimden oluşur. Her iki işlem için de kural tablosunu kullanır. Bulanık çıkarım sisteminde karar verme aşaması kuralların sınanması aşamasıdır, ve kuralların belirli bir yapıda birleştirilmesi gerekir (Mamdani ve Assilian, 1975).

Toplama işleminde; her bir kural için temsil edilen bulanık kümeler tek bir bulanık küme haline getirilir. Toplama işlemi, durulaştırma işleminden bir önceki işlemdir ve her bir giriş için bir kere yapılır. Toplama işleminin girişleri, çıkarım işleminden geçirilmiş olan kesikli çıkış fonksiyonlarıdır. Toplama işleminin çıkışı ise her bir çıkış için bir bulanık kümedir. Toplama işleminde değişme özelliği vardır; dolayısı ile hangi kuralın önce tetiklendiği/uygulandığı önemli değildir.

Aşağıda verilen iki örnekte Mamdani tipi kullanılmıştır. Bulanık kural olarak dokuz adet kural oluşturulmuş fakat tetiklenen kural sayısı iki olarak ayarlanmıştır. Sistemde varsayılan olarak "VE" operatörü kullanılmıştır. "Ve" operatörü kullanıldığı için minimum işlemi uygulanır (Mathworks, 2010).



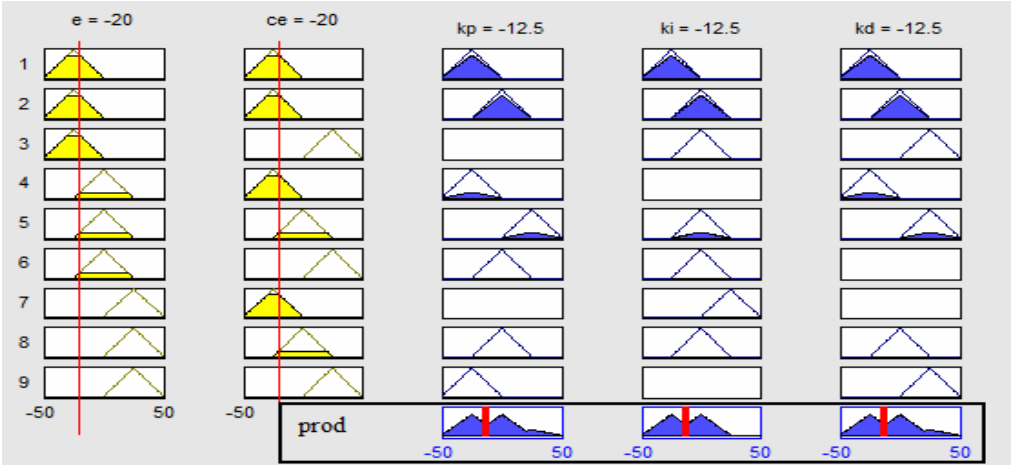
Şekil 2.10 Toplam (sum) bulanık çıkarım işlemi



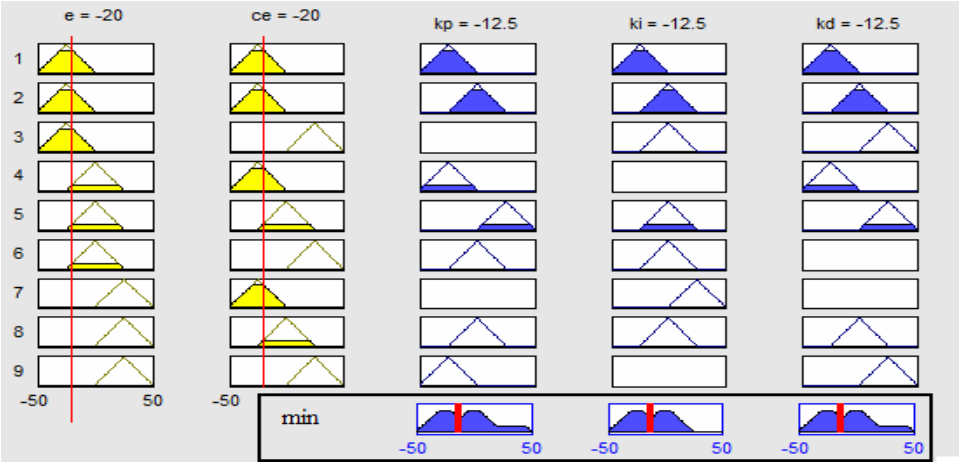
Şekil 2.11 Maksimum bulanık çıkarım işlemi

Çıkarım işlemi uygulanmadan önce her bir kural için "kural ağırlığı"nın belirlenmesi gerekmektedir. Birden fazla kural ile ilgili çıkarım yapılırken "ve" ve "veya" kullanılması işlem esnasında minimum veya maksimum olan girişi kullanma seçiminin yapılmasını sağlar. Benzer şekilde her bir kuralda 0-1 aralığında kural ağırlığı değere sahiptir. Genel olarak bir alınır fakat hassas kontrol yapılmak istendiğinde farklı kurallar için farklı ağırlık verilebilir. Her bir kural için ağırlık ataması yapıldıktan sonra çıkarım işlemi uygulanır. Bu işlemin sonucu üzerine dilsel etiket işlenmiş üyelik fonksiyonu kümesidir.

Tıpkı toplama işlemi gibi çıkarım işlemi de her bir kural için tek tek uygulanır. Min ve prod isimli yöntem sıkça kullanılmaktadır. "Min" yöntemi bulanık çıkış kümesini doğruluk derecesine göre hesaplama yapılan kural ile kırpar; prod ise bulanık çıkış kümesi üzerinde ölçeklendirme yapar.



Şekil 2.12 Prod bulanık çıkarım işlemi



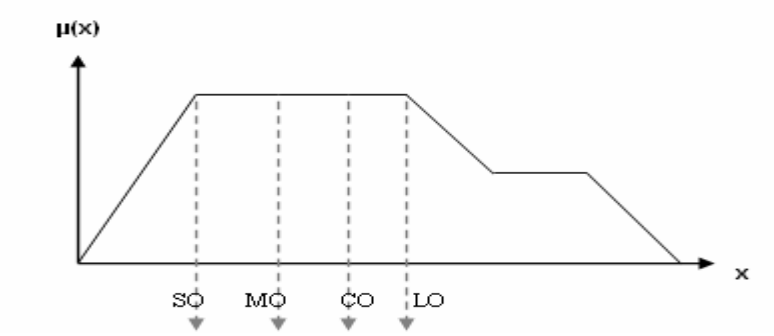
Şekil 2.13 Min bulanık çıkarım işlemi

2.11 Durulaştırma

Girişler alındı, bulanıklaştırma işlemine tabi tutuldu ve üyelik fonksiyonlarında yerlerine yerleştirildi, girişlere karşılık gelen üyelik dereceleri oluşturuldu, peki çıkışlar yine bulanık bir üyelik fonksiyonundan ibaret mi olacak? Gerçek hayattaki sistemlerin neredeyse hiçbiri bulanık değerler ile işlem yapmazlar. Kontrol edilecek sistemlerin büyük bir kısmı gerçek değerler ile çalışırlar. Bu sebeple bulanık kontrolörümüzün en son bloğu ve bulanık modelin son aşaması olan "durulaştırma" işlemine ihtiyaç duyarız. Durulaştırma işlemi; elimizde var olan bulanık değerlerin gerçek değerlere, kesin sonuçlara dönüştürülmesi işlemidir. İlk işlem olan bulanıklaştırma işleminini birebir zıttıdır da denilebilir (Mogaddas, 2001).

Çıkarım tipi olarak "Mamdani" veya "Larsen" kullanılıyorsa durulaştırma bloğu kullanılır. Eğer "TSK" veya "Tsukamoto" kullanılıyorsa durulaştırma bloğuna ihtiyaç duyulmaz. Çünkü ilk TSK ve Tsukamoto çıkarım tipinde gerçek (kesin) değerler ile işlem yapılır. Önce her kural için üyelik derecelerinden oluşan değer ve sonuç kural tespit edilir. Sonrasında en uygun yöntem seçilerek durulama yapılır. Burada bulanık kontrolör tasarımında yaygın olarak kullanılan birkaç durulaştırma tipi belirtilmiştir. Centroid yani ağırlık merkezi yöntemi en çok kullanılan durulaştırma yöntemidir (Yüzgeç, 1999).

- Üyelik fonksiyonu maksimumların ilk noktası, FOM
- Üyelik fonksiyonu maksimumların en küçük noktası, SOM
- Üyelik fonksiyonu maksimumların en büyüğü, LOM
- Üyelik fonksiyonu en yüksek noktaların ortalamaları, MOM
- Ağırlık Merkezi, COG

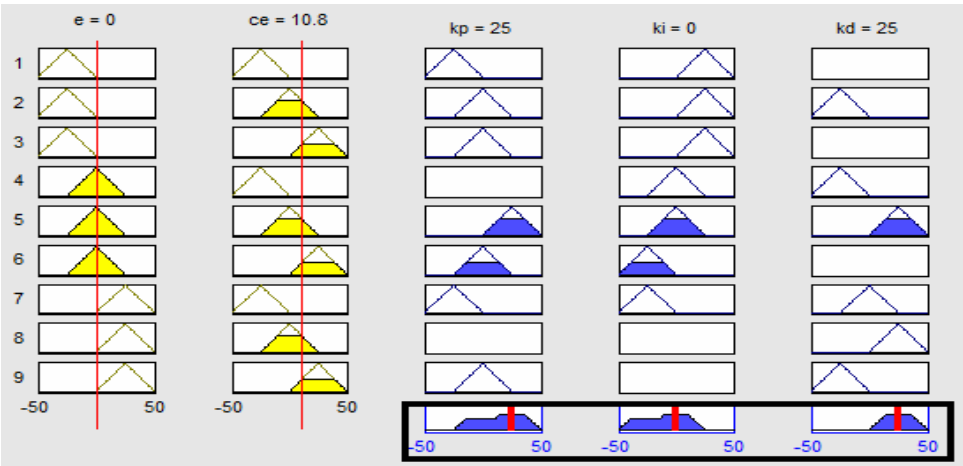


Şekil 2.14 Durulaştırma yöntemleri sonuç gösterimi.

Durulama işleminde kural veya kurallar için sonuçlar tayin edilir; sonrasında oluşan şekle seçilmiş olan yöntemle durulama işlemi gerçekleştirilir. Şöyle ki önce "ölçekleme" yapılarak üyelik derecelerinin karşılıkları bulunur, daha sonra "durulama" ile bulanık kontrolör ile bulunan bulanık değer, bulanık olmayan ve keskin olarak adlandırılan sayısal değer üretilir.

Aşağıdaki örnekte sistemimizde kullanılan üç üyelik fonksiyonu için yapılan durulaştırma işlemi gösterilmiştir. Sistemimizde iki giriş (e-hata, ce-hata türevi) üç adet çıkış (K_p, K_i, K_d) kullanılmıştır. Her giriş ve çıkış için üçer adet üyelik fonksiyonu tanımlanmıştır (N, Z, P). Alınan girişler seçilip üyelik fonksiyonlarındaki yerleri belirlendikten sonra kural tablosuna göre çıkışları belirlenir ve son olarak bulanık küme işlemlerinin ardından seçilmiş olan "durulaştırma" tipine göre sonuç belirlenir. Örnekte durulaştırma tipi olarak "centroid ağırlık merkezi" yöntemi seçilmiştir. (2.8) genel gösterimi verilmiştir.

$$A = \frac{\sum_{i=1}^{n} z_i w_i}{\sum_{i=1}^{n} w_i} \tag{2.8}$$



Şekil 2.15 Mom tipi durulaştırma sonuç gösterimi

Hangi durulaştırma yöntemi genel kullanıma hitap etmektedir? Bunun için kesin bir cevap veya ispat bulunmamaktadır. Eğer hızlı bir modelleme yapılmak isteniyorsa “centroid” yani “ağırlık merkezi” yöntemi uygun bir başlangıç olabilir. Tıpkı üyelik fonksiyonu ve kural tablosu oluşturma işlemlerinde olduğu gibi en uygun durulaştırma yöntemi seçimi deneme yanılma yöntemi ile bulunmaktadır. Fakat günümüzde geliştirilmiş olan öğrenme algoritmaları ve GA ile yapılan arama yöntemleri ile hangi durulaştırma yönteminin uygun olduğu uzman kişiler olmadan da belirlenebilmektedir.

3. GENETİK ALGORİTMA

3.1 Genetik Algoritma Tarihçesi

Charles Darwinian'ın 1859 yılında ortaya attığı evrim teorisi "Survival of the fittest – Uygun Olanın Hayatta Kalması" arama ve optimizasyon algoritmalarının temelini oluşturmuştur. Bu teorinin temeli doğal seleksiyondur. Doğal evrimin çalışmasının temelinde doğadaki canlıların sürekli olarak değişen çevreye uyum sağlamasının olduğu görülmüştür. Doğada varolmak için gerekli temel kaynakların kullanımı hedefi ile bir rekabet sürmekte ve bu rekabette diğer canlılardan daha uygun özelliklere sahip olan bireyler hayatta kalabilmektedir, güçlü olmayan bireyler ise gelecekte temsil edilmemektedir (Darwin, 1859).

Evrim; genetik bilgi taşıyan kromozomlar üzerinde genetik işlemlerin uygulanması sürecine verilen addır. Yani oluşturulan bir popülasyonun kopyalama, çaprazlama ve mutasyon gibi işlemlere tabi tutularak yeni topluluğun oluşumuna kadar geçen dönemdir. Bu dönem popülasyon üzerinde bütün genetik işlemlerin sırasıyla uygulandığı bir süreçtir. Evrim sürecine, başlangıçta belirlenen bir durdurma kriteri gerçekleşinceye veya probleme uygun bir çözüm bulununcaya kadar sürmektedir. GA oluşturulurken evrim sürecinden faydalanma fikri; bir bireyin hem annesinin hem de babasının özelliklerini taşıyabildiği gibi öncekilerden bile üstün olabilmesi; eldeki problemin bilinen anne ve babasından daha farklı ve üstün niteliklerini de taşıyabileceği bazı çözümlerinin melezlenmesiyle çok daha iyi çözümler üretilebileceği fikrini doğurmuştur.

GA ismini biyoloji ve bilgisayar biliminden almaktadır. GA üzerindeki her bir çözüm, birey veya kromozom olarak isimlendirilen dizinlerle gösterilir. Biyolojiden esinlenerek genellikle "0" ve "1" lerden oluşan dizilerle veya gerçek değerlerle ifade edilirler. Biyolojik kromozom üzerinde belirli genlerin belirli karakteristik özellikleri taşıması gibi genetik algoritmaların belirli kısımlarının da belirli özellikleriyle problemin çözümünü içerdiği kabul edilmektedir.

GA tanımı ilk defa J.D Bagley ait olan doktora tezinde konu edilmiştir. Bagley adaptif (uyarlamalı) sistemler üzerine çalışmış ve GA üzerine yoğunlaşmıştır (Bagley, 1967). Evrim kuramını esas alan Genetik Algoritmaların temel prensipleri ise ilk defa makine öğrenmesi (machine learning) konusunda çalışmalar yapan J. Holland tarafından ortaya konulmuştur. Michigan Üniversitesi'nde psikoloji ve bilgisayar bilimi uzmanı olan John Holland; yapmış olduğu çalışmalarda; canlıların evriminden, değişiminden etkilenerek, bu genetik evrim sürecini bilgisayar ortamına aktarmak sureti ile mekanik yapının öğrenme yeteneğini

geliştirmektense bir çok yapının bütününden çiftleşme, çoğalma, değişim olarak adlandırılan genetik süreçler sonunda üstün yeni bireylerin elde edilebileceğini ispatlamıştır. John Holland'ın asıl hedefi özel problemlerin çözümü için algoritma üretmek değildi. Doğal adaptasyon sürecini bilgisayar sistemlerine uyarlamaya çalışıyordu. Doğal ve Yapay Sistemlerde Adaptasyon "Adaptation in Natural and Artifical Systems" isimli kitabında genetik algoritmayı biyolojik evrimin girişi olarak tanımlamıştır (Holland, 1975).

John Holland'ın öğrencisi olan David Goldberg'in "Gaz Borularının GA İle Optimizasyonu" konulu doktora tezi, kendisine "National Science Foundation" Genç Araştırmacı ödülünü kazandırmış olmasının yanında genetik algoritmaların yalnızca teorik olmadığını ayrıca farklı alanlardaki çeşitli problemlerin çözümünde yaygın bir şekilde uygulanabileceğini gösterdi (Goldberg, 1989).

GA, farklı problem çeşitlerine uyarlanabilmesi ve etkin sonuçlar üretebilmesi sebebi ile araştırmacılar arasında oldukça geniş bir kullanım alanı bulmuştur. GA reel hayata ilişkin hemen hemen problemlerin (ayrık, stokastik veya yüksek dereceli doğrusal olmayan) çözümünde geleneksel optimizasyon problemlerine oranla daha başarılı sonuçlar üretirler (Gen ve Cheng, 1997).

Araştırmacıların daha iyi sonuçlar elde etme isteği ise GA üzerindeki çalışmaların artmasını sağlamıştır. John Koza (1992), yaptığı araştırmalar sonunda GA kullanarak çeşitli görevleri yerine getiren programlar geliştirmiş; geliştirdiği bu metoda "Genetik Programlama" adını vermiştir. Günümüzde değişik problemler için birçok arama ve optimizasyon yöntemi bulunmaktadır.

3.2 Genetik Algoritma ve Diğer Yöntemler Arasındaki Farklar

Literatürde, problemleri çözme hedefi ile uygulanan arama teknikleri; hesap-temelli ve direkt-arama teknikleri olarak sınıflandırılabilir. Çözülmek istenen problemler sayısal olarak iyi tanımlanabiliyorsa veya çözüm uzayı küçük ve tek ise, hesap temelli arama tekniği daha iyi sonuç verir. Buna rağmen hesap-temelli teknik, mühendislikte gittikçe artan en iyiyi bulma işlevlerinde oldukça zayıf kalmaktadır. Bu algoritmalar, kompleks bir sistemin sadece bir bölümünü optimize etmek için uygundurlar. Bu sebeple bu tür metodlar geniş araştırma uzayında kullanım için uygun değildirler. Çünkü tüm çözümlerin değerlendirilmesi oldukça zaman alması sebebi ile optimum çözümün bulunması için oldukça uzun bir süre gerektirecektir. Genetik algoritmalar ise;

- Arama işlemine tek bir noktadan değil, noktalar kümesinden (nüfus) başlar. Bu nüfus, problemin bütün olası çözümlerini temsil eden uzayı oluşturur ve "yerel en iyi" çözümde sıkışıp kalmazlar. GA tek yerden değil, bir grup çözüm içinden arama yapar. Birçok en iyileme probleminde; örneğin tepe tırmanma algoritmasında, tanım aralığı içindeki tek noktadan hareketle bazı geçiş kaidelerine göre inceleme noktaları bulunur (bir önceki durumdan daha iyi duruma geçmek). Bu, noktadan noktaya yönelme metodu, çok sayıda tepe noktası olan araştırma uzayı için risklidir, çünkü yerel en iyide takılır. GA, çok sayıda noktalardan oluşan dizi nüfusu ile çalıştığından aynı anda birçok tepe noktasına geçilebilir. Böylece noktadan noktaya geçme yöntemindeki yanlış tepe noktası bulunma ihtimali azalır.
- Problemlerin çözümünü parametrelerin değerleriyle değil, kodlarıyla arar. Parametreler kodlanabildiği sürece çözüm üretilebilir. Bu sebeple GA ne yaptığı konusunda bilgi içermez, fakat nasıl yaptığını bilir (Kuruca, 2009).
- Türev yerine uygunluk fonksiyonunun değerini kullanır. Bu değerin kullanılması ayrıca yardımcı bir bilginin kullanılmasını gerektirmez. Diğer yardımcı bilgileri değil sadece amaç işlev bilgisini kullanır. Kompleks matematik hesaplamaları yerine sadece giriş-çıkış bilgilerine ihtiyaç duyar. Çoğu metodun, doğru bir biçimde çalışması için yardımcı bir takım bilgilere ihtiyaçları vardır. GA hiçbir yardımcı bilgiye ihtiyaç duymaz. Verimli bir araştırma yapmak için lazım olan, her bir kromozomun değerlendirileceği bir uygunluk fonksiyonudur. Optimizasyon yapılması sırasında problemle ilgili özel bilgilerin kullanılmaması, sadece uygunluk fonksiyonu bilgilerinin kullanılması GA performansını yükseltmede son derece etkilidir (Mitchell, 1996).
- Diğer bir çok en iyileme metodlarınından farklı olarak, olasılığa dayalı geçiş kuralları kullanır. Sonuçta araştırma uzayının hangi bölgesine doğru yöneleceğine karar vermek için rastgele seçim kullanılır. Bu avantajından dolayı kompleks problemlerin optimizasyonu için tercih edilebilir bir yöntemdir. Özellikle doğrusal olmayan sistemlerde ve matematiksel olarak modellenmesi karmaşık olan problemlerde etkindirler. Oluşturulan başlangıç nüfusu girdiği GA döngüsü sonucunda en uygun çözümü rahatça bulabilir.

Sonuç olarak GA geleneksel arama yöntemlerinde olduğu gibi iyi bir sonucu ele alıp onu geliştirmek için çok çabalamaz. Tam tersi çok fazla sonucu aynı anda ele alır ve her bir sonuçta çok daha az çalışır. GA klasik yöntemlerin uzun zamanda bulabileceği çözümü yeterli doğrulukla çok kısa zaman diliminde bulabilir. Optimal değeri bulmayı kesin olarak güvence etmez, fakat uzun nesiller sonucunda en uygun değere yakın çözümler bulunmasını sağlar.

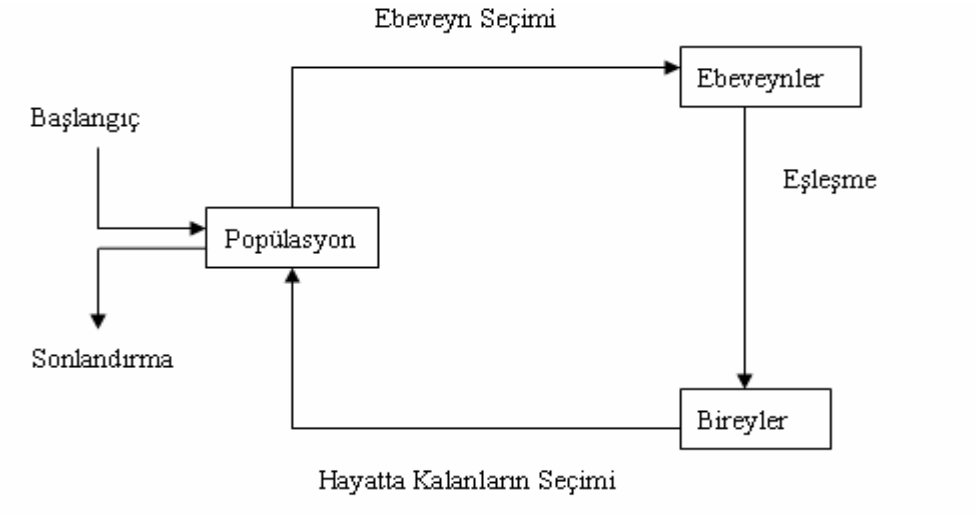
3.3 Kontrol Teorisinde Genetik Algoritma Uygulama Alanları

GA uygulama alanları olarak: ekonomi, iktisat, tüm mühendislik ve tıp alanları örnek verilebilir. Bunun yanında disiplinler arası kullanımına da sıkça rastlamak mümkündür. Konumuz kontrol teorisi olduğu için son yıllarda kontrol teorisinde yapılan uygulamalar ile ilgili IEEE (International Electrical & Electronics Engineers) topluluğundan örnekler verilmiştir [1].

- GA temelli hibrid kontrolör ile enerji sistemleri kararlılık kontrolü .
- Doğrusal olmayan kontrol sistemleri için GA ile çok amaçlı optimizasyon.
- Nükleer buhar jeneratörü için genel kullanım amaçlı ileribesleme ve geribeslemeli kontrolör tasarımı.
- PID kazanç katsayılarının GA ile eş zamanlı ayarlanması.
- Takagi Sugeno model ikinci dereceden kararlı ve zaman gecikmeli kontrol sistemleri için GA ile optimal kontrol.
- CPU sıcaklık düşürme prosesi için optimal tasarım ve kontrol.
- GA ile buhar türbini kontrol sistemi tasarımı.
- MIMO sistemler için önceden parametre ayarlanmış PID kontrolü.
- Uyarlanabilir hidrojen jerneratörünün GA ile ayarlanması.
- Makine öğrenmesi, yapay sinir ağları ve genetik algoritmalar.
- Elektrolitik kapasitörlerde hata yakalama ve güç kaybı yaklaşımı.
- GA ile paralel veri madenciliği.
- GA ile voltaj ve reaktif güç düzeltici kontrolör tasarımı.
- Doğrusal olmayan kontrol sistemleri için hibrid GA bulanık PID tasarımı.
- Çözünmüş oksijen seviyesi takibi ve kontrolü için genetik tabanlı hibrid öngörülü kontrolör tasarımı.
- Endüktif servo motor sürücünün GA tabanlı gelişrilmiş, kayan mod kontrolör ile hareket kontrolü.
- Bulanık mantık kontrolör veri tabanı ve kural tabanı parametrelerinin GA ile optimizayonu.
- Doğrusal olmayan ayrık zamanlı kontrol sistemleri için GA ile gürbüz bulanık gözlemleyici temelli bulanık kontrolör tasarımı.
- GA tabanlı model öngörülü kontrolör ile nükleer reaktör kontrolü.
- GA tabanlı çoklu etmen destekli öğrenme ile yük-frekans kontrolü.
- MIMO Sistemler için PID katsayılarının GA ile hesaplanması.

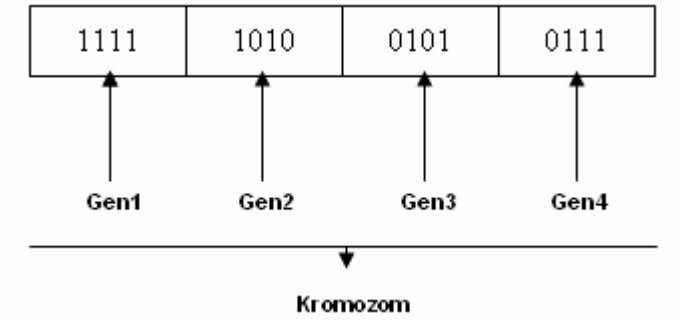
3.4 Genetik Algoritma Tanımı

GA, evrim sürecini örnek alan bir arama yöntemidir. Doğal evrim sürecinin, canlıların genetik ve kalıtım özelliklerinin GA bulunuşundaki katkısı oldukça fazladır. Uygun özelliklere sahip olmanın belirleyici faktörlerinden en mühimi ise değişen ortamlara uyum sağlama kabiliyetidir. GA amacı, problemler için aranan çözümlerin; doğada geçerli olan en iyinin yaşaması kuralına dayanarak sürekli iyileşen çözümler bulmaktır. GA tüm bunları yaparken tek çözüm ile değil gruplarla Bunu sağlamak için "iyi"nin ne olduğunu belirleyen bir uygunluk (fitness) fonksiyonu ve yeni çözümler üretmek için kopyalama, çaprazlama, mutasyon operatörleri ve sonlandırma koşullarını kullanır.



Şekil 3.1 Genetik algoritma genel yapısı

Çözümlerin Kodlanması: GA ilk Kendi başına manası olan ve genetik bilgi taşıyan en küçük genetik birim gendir. Kısmi bilgi taşıyan bu minik yapıların bir araya gelmesiyle bütün bilgileri kapsayan kromozomlar oluşur. Bir gen A, B gibi bir karakter olabileceği gibi 0 veya 1 ile adlandırılan herhangi bit dizisi olabilir.



Şekil 3.2 Gen ve kromozom gösterimi

Bir veya birden fazla genin bir araya gelmesiyle meydana gelen ve probleme ait tüm bilgileri içeren yapılara kromozom denir. Kromozomlar geçerli alternatif aday çözümler manasına gelmektedir.

İlk Popülasyonunun Oluşturulması: Popülasyon kromozomlar veya bireyler topluluğu olarak isimlendirilebilir. GA çalışmaya başlayabilmesi için olası çözümlerin kodlandığı bir çözüm grubu oluşturulmalıdır. GA çalışma esnasında popülasyonun bir kısım bireyleri yok olmakta ve yenileri gelmektedir. İkili kodlamanın kullanıldığı kromozomların gösteriminde, ilk popülasyon rastgele oluşturulabilir (Yeo ve Agyel, 1996).

Uygunluk Değerinin Hesaplanması: Bir kuşak oluşturulduktan sonraki ilk adım, popülasyondaki her üyenin uygunluk değerini hesaplama adımıdır. Bir çözümün uygunluk değeri ne kadar yüksekse, yaşama ve çoğalma şansı o kadar fazladır ve bir sonraki kuşakta temsil edilme oranı da o kadar yüksektir (Yeniay, 2001).

Ebeveyn Seçimi ve Üreme: Çoğalma işleminde diziler, amaç fonksiyonuna göre kopyalanır ve iyi kalıtsal özellikleri gelecek kuşağa daha iyi aktaracak bireyler seçilir. Üreme operatörü yapay bir seçimdir. Dizileri uygunluk değerlerine göre kopyalama, daha yüksek uygunluk değerine sahip dizilerin, bir sonraki kuşaktaki bir veya daha fazla yavruya daha yüksek bir olasılıkla katkıda bulunması anlamına gelmektedir. Seçim işlemi, bir sonraki kuşak için yeni nesil üretmek amacı ile hangi ebeveynlerin yer alması gerektiğine karar vermektedir.

Çaprazlama İşleminin Uygulanması: Farklı çözümler arasında bilgi değişimini sağlayarak arama uzayının araştırılmamış bölgelerine yakınsamasını sağlayan bir arama operatörüne çaprazlama denir. Çoğalma işlemi sonucunda elde edilen yeni populasyondan rastsal olarak iki kromozom seçilmekte ve karşılıklı çaprazlama işlemine tabi tutulmaktadır (Engin, 2001).

Mutasyon İşleminin Uygulanması: Çaprazlama, mevcut gen potansiyellerini araştırmak üzere kullanılır. Fakat popülasyon gerekli tüm kodlanmış bilgiyi içermez ise çaprazlama tatmin edici bir çözüm üretemez. Bundan dolayı, mevcut kromozomlardan yeni kromozomlar üretme yeteneğine sahip bir operatör gerekmektedir. Bu görevi mutasyon gerçekleştirir.

Yeni Popülasyonun Oluşması ve Algoritmanın Sonlandırılması: Yeni kuşak çoğalma, çaprazlama ve mutasyon işlemlerinden sonra tanımlanmakta ve bir sonraki kuşağın ebeveynleri olmaktadırlar. Süreç yeni kuşakla çoğalma için belirlenen uygunluk ile devam eder. Bu süreç önceden belirlenen kuşak sayısı kadar veya bir hedefe ulaşılıncaya kadar veya durdurma kriteri sağlanana kadar devam eder.

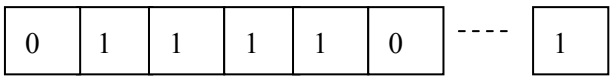
3.5 Kromozom Kodlama

GA diğer tüm arama metodlarından ayıran en önemli özellik, değişkenin kendisi yerine değişkenleri simgeleyen kromozomları kullanmasıdır. GA değişimli olarak kodlama uzayı ve çözüm uzayı üzerinde çalışır. Yani genetik operatörler kodlama uzayında gerçekleşirken, değerlendirme ve seçim işlemi ise çözüm uzayında gerçekleşmektedir. Bir problemin çözümü esnasında, çözümler arasından ötekilerine kıyasla en iyi olan araştırılır. Her olası çözüm uygunluk değeriyle belirtilir. Bu sebeple GA'nın uygulanmasında ilk adım, sorun için arama uzayını en iyi temsil eden uygun bir kodlama yapısının tercih edilmesidir. Bu problemin çözümü ile uğraşan uzmanlar için önemli vakit ayrılması gereken bölümdür. Kromozomlar ve değer dizileri, GA kuramının üzerinde uygulandığı en temel birimler olduğundan, bu önemli bilgisayar ortamında çok iyi ifade edilmesi gereklidir.

Kromozom kodu çözümünde, genotipin fenetipe dönüşüm prosesi çalışmaktadır. Genotip, biyolojideki tarifi gibi GA kullanımında bireyin genetik bünyesini temsil eden bit dizinini yani kromozomu ifade eder. Fenotip ise, GA çözümü ifade eden erişkin bireylerdir. Kromozomlar yaygın olarak GA içerisinde şu şekilde kodlanmaktadır:

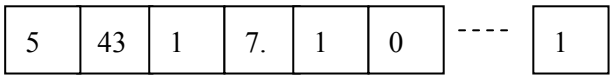
- İkili Diziler (1011, 0100)
- Gerçek sayılar (5.4, 6.5)
- Karakterler (A, B, X)
- Kural listeleri

İkili değer kodlaması: GA üzerinde yoğunlukla tatbik edilen metod ikilik tabanda kodlamadır. Goldberg, 1989 temelini attığı Basit GA kullanımında kodlama, ikili düzende kodlamadır. Bu kodlamada, her dizi "0" ve "1" değerlerinden oluşmaktadır. Bu yöntemde kromozomlar "0" ve "1"lerin meydana getirdiği bit dizilerinden oluşur. Kodlanmış tüm değişkenlerin ard arda dizilenmesiyle kromozomlar elde edilir. Kromozom kodlarının çözülmesiyle değişkenlerin özgün değerleri elde edilebilir. Bu sayı her değişken için ayrımlı seçilebilir. Genel olarak ikili kodlama ayrık sistemlerde kullanılmaktadır. Yapılan uygulamaya göre değişkenlik göstermekle birlikte ikili kullanımın avantajı; GA uygulamasında mutasyon işlemi esnasında, temsil edilen değer mutasyondan en az şekilde etkilenmesidir.



Şekil 3.3 İkili değerde kodlama

Gerçek değer kodlaması: Bünyesinde reel sayısal ölçüler kapsayan sorunlarda tatbik edilen kodlama şeklidir. Bu tür sorunları ikili kodlama ile çözmek amacı ile uğraşmak zorlayıcı bir seçenek olabilir. Böyle durumlarda, diziler gerçek değerlerle doğrudan kodlanır. GA kullanımında değişkenleri gerçek değerleriyle uygulamanın avantajı, dizinin her uygunluk değerinin hesaplanmasından önce gerçek değerine çevrilmesine ihtiyaç duyulmayacağı için işlem süresini kısaltması ve değişkenlerin kodlanması esnasında değişkenlerin oluşabilecek duyarlılık kaybı sıkıntılarını alt etmesidir.



Şekil 3.4 Gerçek değerde kodlama

3.6 İlk Popülasyonunu Oluşturulması

GA tanımına göre popülasyon; belli operatörler aracılığıyla dizilerle temsil edilen çözümlerden, soruna uygun bir sayıda bir araya gelmiş olanlarının meydana getirdiği çözüm setidir. GA ait her bir evrede, operatörlerin tatbik ettiği işlemler sonucu bu popülasyon yenilenmektedir. Her yeni popülasyonun bir öncekinden daha iyi sonuçlar ortaya çıkarması beklenmektedir. Sebebi ise üretim operatörü tatbik edilirken yani seçim yapılırken uygunluğu maksimum kromozomların tercih edilmesi daha yüksektir.

Tüm bu işlemlerin yapılabilmesi veya başlayabilmesi için ortada bir kromozom setinin bulunması gerekir. Başlangıçta elimizde var olan çözümler başlangıç neslini ifade eder. Şayet elimizde var olan bir başlangıç nesli yoksa başlangıç kromozomları rasgele üretilebilir. Daha sonra, doğal seleksiyon sürecini başlatabilmek için çaprazlama ve mutasyon operatörleri kullanılarak yeni nesiller türetilir. Bu çalışma kapsamında başlangıç nesli rastgele olarak üretilmektedir.

GA tatbikinde popülasyon genişliğinin değer olarak belirlenmesi gerekir. Eğer kromozom adeti düşük olursa optimizasyon sırasında çözüm aranan uzayın ancak bir bölümü gezilebilir ve çaprazlama için alternatif olmaz. Büyük popülasyonlarda çözüm uzayı iyi örneklendiği için aramanın gücü artar lakin uygun bir sürede yüksek kalitede bir çözüme ulaşılamayabilir. Genel uygulamada kullanılmak için belirlenmiş standart başlangıç popülasyonu sayısı bulunmamaktadır (Reeves, 1993).

Başlangıç popülasyonu genel olarak rastgele atama olarak oluşturulur. Örnek olarak ikili olarak kodlanmış her bir kromozomun 9 bitten oluştuğunu ve 3 adet gerçek değer ifade

ettiğini varsayalım. İlk nüfus değerinin "5" olduğu düşünülürse; rastgele olarak oluşturulmuş ilk popülasyonda bulunan kromozomların oluşturduğu topluluk şu şekilde olur:

Çizelge 3.1 Başlangıç popülasyonu örneği

Indis	Kromozom	Gerçek Değer
1	100 110 001	4, 6, 1
2	101 100 111	5, 4, 7
3	001 001 101	1, 1, 5
4	011 101 100	3, 5, 4
5	111 011 001	7, 3, 1

3.7 Uygunluk Fonksiyonu

Problem çözümünün GA kullanılarak sonuca ulaşılmak istendiğinde probleme uygun çözüm yığınının elde edilebilmesi için uygunluk fonksiyonunun belirlenmesi gerekmektedir. Uygunluk fonksiyonu optimizasyon yapmak istediğimiz fonksiyona verilen addır. Literatürde standart optimizasyonda amaç fonksiyonu-objective function olarak bilinir. GA uygulamasında efektif sonuç için uygunluk fonksiyonunun tüm işlemlerin başında, dikkatli şekilde, amaca uygun tanımlanması gerekmektedir. Bizim problemimiz bulanık kontrolöre ait üyelik fonksiyonları ve kural tablosunu optimize etmek olacaktır. Program her döngüde kontrolörü sisteme uygulayacak ve sistemdeki hatayı minimize etmeye çalışacaktır. Amacımıza uygun olan "uygunluk fonksiyonu" hatanın mutlak değerinin entegralini minimize etmek olacaktır.

$$Fit(IAE) = \int_0^\infty | y_{sp}(t) - y(t) | dt \tag{3.1}$$

GA bireylerin uygunluk ve iyiliklerine göre ayrılıp fark edilmesine ihtiyaç duyar. Uygunluk fonksiyonunun GA içerisinde kullanılması ile problemin çözümünde yarar sağlayacak olan kromozomların varlıklarını sürdürme şanslarının artırılması ve problemin çözümünde fayda kazandırmayan kromozomların nüfustan ayıklanmasına imkan sağlanır. Uygunluk, popülasyondaki bir kısım bireyin sorunu nasıl çözeceği için iyi bir kriterdir. Her kuşakta, mekanizma iyi bireyleri seçerek eşleşme havuzuna atar. Eşleşme havuzu gelecek nesili üretmek için kullanılır. Bir sonraki işlem olan yeniden üretim seçim operatörleri yardımıyla yapılabilir. Kurama göre iyi olan bireyler yaşamını sürdürmeli ve bu bireylerden yeni ve daha nitelikli bireyler oluşmalıdır. Kromozoma ait çözümün uygunluk değerinin azami yüksek

oluşu o bireyin hayatta kalma ve üreme şansının yüksek oluşu anlamına gelmektedir ve o bireyin bir sonraki nesilde temsil edilme oranı aynı oranda yüksektir (Dalci, Uzunoglu ve Küçükdemiral, 2004).

3.8 Çaprazlama

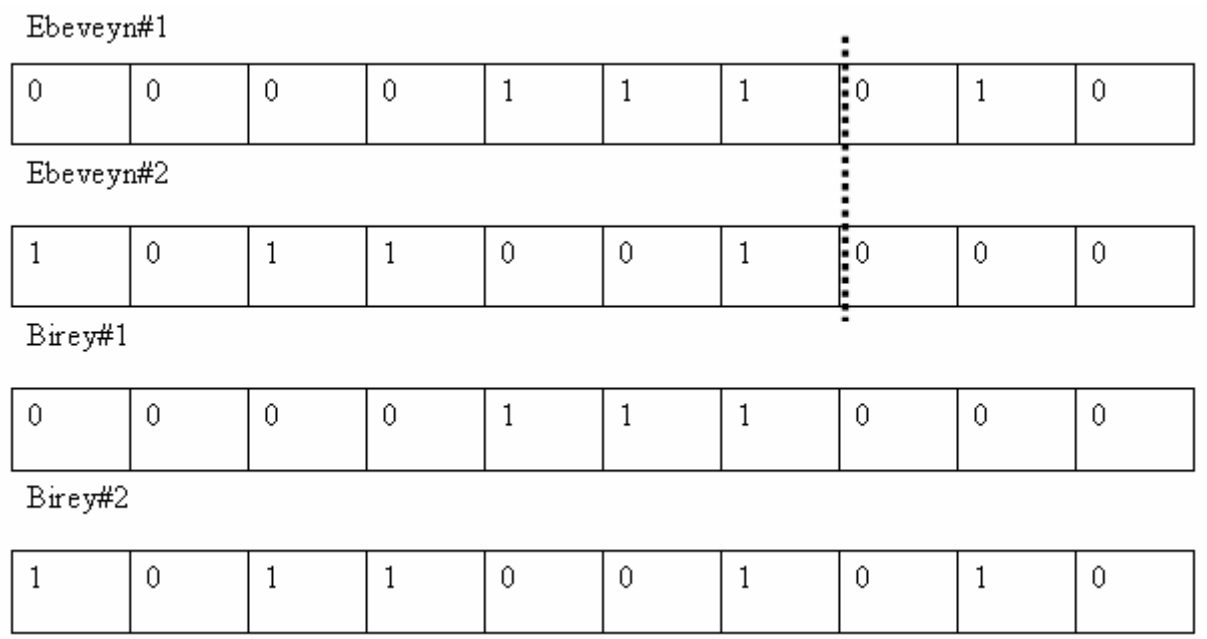
Çaprazlama, farklı çözümler arasında bilgi değişimini sağlayarak arama uzayının benzer ancak araştırılmamış bölgelerine varımını sağlayan bir arama operatörüdür. Bu işlem yapılırken her zaman sonuçlar önceden tahmin edilemez. Bu yüzden gelişigüzel yapılan değişikliklerde sonucun mükemmelliğe doğru gitmesi için belirli kıstaslar bulmak için çalışılır. Dizilerdeki genlerin yapısı ve etkileri araştırılarak, bu genlere yapılan müdahalelerle bireye bazı iyi özellikler kazandırılabilir. GA, çaprazlama işlemini uygunluk ölçütlerine göre seçilmiş iki ebeveyn bireyden, iyi özellikte yeni bireyler elde etmek için kullanır. Çaprazlama ile birlikte en iyi değere doğru nüfus yavaş yavaş yaklaşmaya başlayacaktır. Problemimizin çözümünde yani hem bulanık kontrolör üyelik fonksiyonları hem de kural optimizasonunda iki noktalı çaprazlama kullanıldı. Çünkü tıpkı diğer parametrelerin seçiminde olduğu gibi çaprazlama seçiminde de deneme yanılma yöntemi ile iki noktalı çaprazlamanın tek noktalı çaprazlamaya göre daha iyi sonuç verdiği görüldü.

Çözüm araması sırasında çözüme aday olan iki farklı karar uzayı noktası arasında yapılan bir çaprazlama ile öncekilerden çözüme daha yakın olabilecek iki tane yeni çözüm noktası ortaya çıkarılır. Bu iki yeni çözüm noktası önceki noktalardan kalıtımlı olarak doğar. Böylece çözüm popülasyonundaki karar noktaları daha iyiye doğru evrimleşecek biçimde yeni bir nüfus oluşturur. Burada iki çözüm noktasının sayısal değerlerinin bazı haneleri aralarında çaprazlama yaparak yeni çözüm noktalarının var olması sağlanır.

Çaprazlama ile işlem uygulanacak eşler gelişigüzel veya çeşitli yöntemlerle seçildikten sonra dizi eşlerinin hangi genlerden itibaren kesileceği gelişigüzel veya belirli oranlarla (P_c) belirlenir. Kesilen genler, dizi eşleri arasında karşılıklı değiştirilir. Buradaki hedef eldeki kromozomlardan yeni diziler oluşturmaktır. Bu uygulamada çaprazlanacak dizi eşleri ve bunların hangi genlerden itibaren çaprazlanacağı rastgele olarak belirlenir. GA tasarım aşamasında ve GA uygulaması sırasında performansı için etken iki çeşit çaprazlama bulunmaktadır.

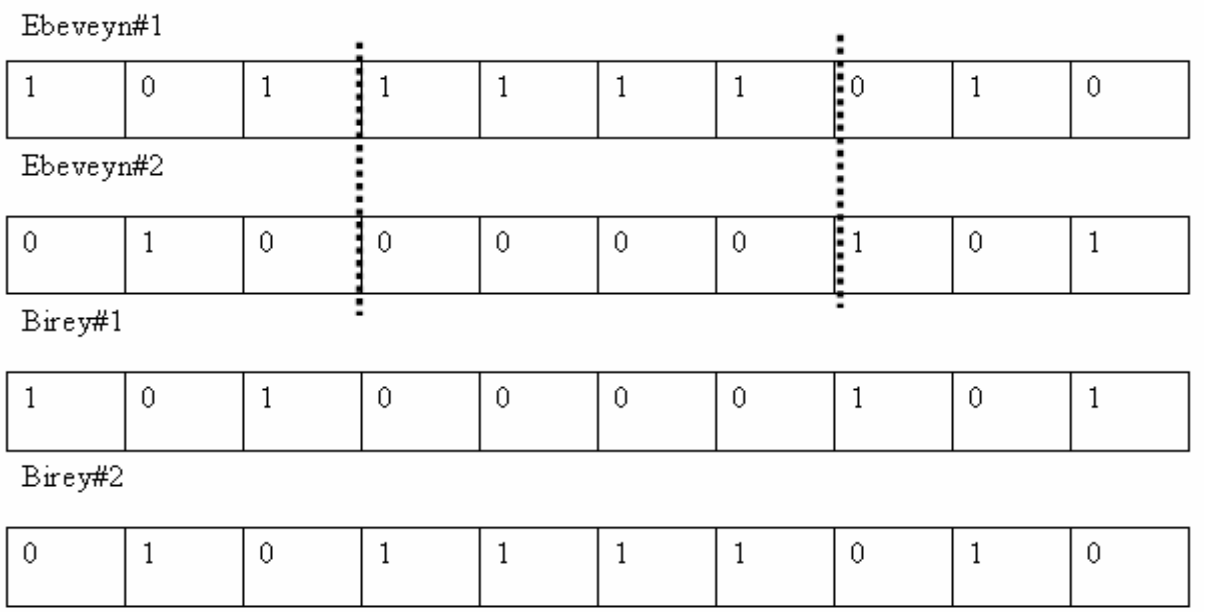
Tek noktalı çaprazlama operatörü: Tek çaprazlama işleminde, x uzunluğundaki iki zincirin birer noktalarından ikiye bölünmesiyle birincil ve ikincil parçalar yer değiştirmektedir.

Çaprazlama noktası 1 ile x-l arasından rassal olarak seçilir. Eşleşen iki dizide, bu çaprazlama noktasından sonraki bölümler yer değiştirerek yeni iki dizi elde edilir (Goldberg, 1989).



Şekil 3.5 İki ebeveynin (10 bit), rastgele seçilen (7. bit) tek noktalı çaprazlama

İki noktalı çaprazlama operatörü: İki noktalı çaprazlama operatörünü tek noktalı çaprazlama operatöründen ayıran nokta "1" ile "x-1" (kromozom uzunluğu) arasında çaprazlama için iki farklı nokta seçilmesidir. Yeni komozomlar çaprazlama sonrasında eşleşen dizilerin bu noktalar arasındaki bölgelerinin yer değiştirmesi ile elde edilir (Davis, 1991).



Şekil 3.6 İki ebeveynin (10 bit), rastgele seçilen (4. ve 8. bit) iki noktalı çaprazlama

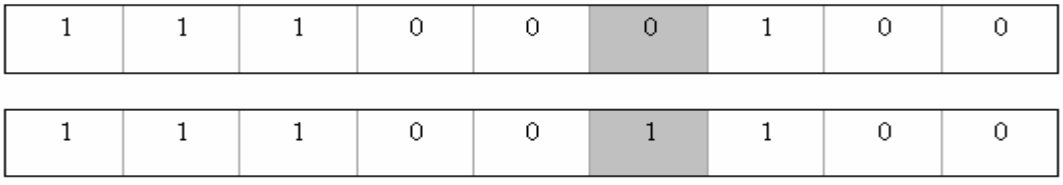
Literatürde tek ve iki noktalı çaprazlama dışında birçok çaprazlama çeşidi bulunmaktadır. Örneğin sıkça kullanılan maskeleme yöntemi ile rastgele ikili dizi oluşturulur. Oluşturulan bu ikili dizi ile ebeveynlere ait noktalar arasında rastgele ve çok noktalı çaprazlama gerçekleştirilir.

3.9 Mutasyon

Mutasyon ile popülasyonda içerisinde bulunan kromozom içerisindeki birkaç değeri çok küçük oranlar ile değiştirerek yeni kromozomların elde edilmesini sağlar (Goldberg, 1989). Mutasyon, kaybolan genetik malzemenin yeniden elde edilmesini sağlayan bir operatördür. Örneğin, nüfustaki her çözümün bit değeri 0 ise ne kadar çaprazlama yapılırsa yapılsın aynı bit için değeri "1" olan çözüm meydana gelmeyecektir. Nüfusta çeşitlilik yaratabilmek, çaprazlama sonucunda kaybolabilen iyi nitelikleri geri kazanabilmek ve en iyiye ulaşabilmek için bireylerdeki kodlar belli bir ihtimal ile P_m mutasyona uğratılmaktadır. Mutasyonun bazı durumlarda yararsız, hatta bozucu değişimler yaratabilme olasılığı bulunmaktadır. Yüksek mutasyon oranı GA döngüsünde rasgele aramaya sebebiyet verdiğinden popülasyonda yapılan aramalarda mutasyon olasılığının düşük olması tercih edilir.

Çaprazlama ile nispeten iyi olduğu gösterilen dizilerin birleşmeleri sağlanır. Seçme ve çaprazlama beraber yeni çözümler arasa bile hızlı yakınsamaya neden olurlar. Faydalı genetik gerecin yokolma riski nedeni ile mutasyon gerekli bir operatördür. Farklı mutasyon fonksiyonları bulunmaktadır. Bunlardan; gauss, düzgün (uniform) ve uyarlanabilir en sık kullanılan mutasyon fonksiyonlarıdır.

Yapılan çalışmalar mutasyonun oranının, kromozom uzunluğu ve probleme ait çözüm uzayı ile birebir orantılı olduğunu ispatlamıştır (Topuz, 2002). Birçok durumda düşük nüfus sayılarında sistemin performansını çaprazlama olasılığından daha çok mutasyon olasılığı belirlemektedir. Örneğin; ikili bir kodlamanın kullanıldığı bir dizide mutasyon operatörü ile rasgele seçilen eleman değeri "1" ise "0" ; veya "0" ise "1" olarak değiştirilerek yeni bir dizi elde edilir.



Şekil 3.7 Mutasyona uğratılmış kromozom (6.bit) öncesi ve sonrası

3.10 Seçim Metodları

Kromozom seçme operasyonu üremeye aday bireylerin seçilmesi olarak belirtilebilir. GA kullanımında seçme operatörünün fonksiyonu, yüksek uygunluk değerine sahip bireylerin sonraki nesillerde varlık gösterme ihtimalini fazlalaştırmaktır. Böylelikle seçim, arama uzayının başarılı olacağı beklenilen bölgelerde yoğunlaştırılır ve başarı şansı arttırılır (Kuvat, 2009). Bunun için üremeye bırakılacak bireyler, üreme havuzu olarak adlandırılan bir nüfustan gelişigüzel seçilerek eşleştirilir. Nüfusun çeşitliliğindeki en önemli etki seçim için oluşturulan dominanslıktır. Eğer azami derecede seçim baskısı uygulanırsa, GA hızla nüfus türlülüğünü kaybederek yakınsama prosesine girer. Zayıf seçicilik uygulandığında ise, aramanın aktifliği olumsuz etkilenir. Bunun için iyi bir GA başarımı elde etmek amacıyla, uygun göreli uygunluk mizanı ile yeterli seçim baskınlığını belirlemek gerekmektedir (Michalewicz, 1992).

Rulet Çemberi: Genetik algoritmada kullanılan en basit seçim mekanizması rulet çemberi yöntemidir. Rulet çemberi yönteminde kromozomlar uyumluluk işlevine göre bir rulet etrafına gruplanır (Goldberg 1989). Uygunluk fonksiyonu herhangi bir kıstasa uyan bireylerin seçilmesi için kullanılır. Bu rulet üzerinden gelişigüzel bir birey seçilir. Daha geniş bir alana sahip bireyin seçilme şansı daha fazla olacaktır (Çivril, 2009). Bu metod yardımıyla kromozomlar istatistiksel yöntemler kullanılarak uygunluk fonksiyonu değerlerinin toplam uygunluk fonksiyonuna oranları ölçüsünde seçilirler. Rulet çemberi metodu, basit olmasına karşın bir stokastik hataya sahiptir. Bu hata, yeni kümede her kromozomun beklenen kopya sayısı ile gerçekleşen kopya sayısı arasında büyük bir farkın olmasıdır. Tekrarlanan ardıştırmalarında bu örnekleme yanlış artmakta ve kuramsal olarak tahmin edilenden çok daha farklı yönlerde aramaya devam edilmesine neden olmaktadır (Booker ve Holland, 1989).

$$olasılık(birey) = \frac{uygunluk(i)}{\sum_{i} uygunluk(i)} \tag{3.2}$$

Rank Seçimi: Rank seçimi popülasyonunun uygunluk değerlerine göre sıralanmasıdır. Popülasyon içinde bulunan N tane kromozom arasından eşit olasılıkla seçim yapılır ve yüksek rank değerindeki kromozomlar yeniden üreme için seçilir. Uygunluk değerleri birbirinden çok farklı olan durumlar rulet çarkı seçimi için sorun teşkil yaratır. Rank seçiminde tüm bireylerin seçilme ihtimali bulunmaktadır. Rank seçimi yönteminde en iyi birey diğer bireylerden çok fazla değişiklik göstermediğinden bu yöntem yavaş bir yaklaşımdır.

$$uygunluk(birey_1) \geq uygunluk(birey_2) \tag{3.3}$$

Turnuva Seçimi: Kural dahilinde iki kromozom toplum içerisinden seçilerek uygunluk işlevi büyük olan kromozom eşleşme havuzuna gönderilir, diğeri ise havuzun içine tekrar bırakılır. Bu işleme yeni populasyon büyüklüğüne ulaşılıncaya kadar devam edilir. Bu yöntemin kazanımı herhangi bir kromozomun proses esnasında kaybedilme olasılığı rulet çemberi seçim yöntemine göre daha azdır. Turnuva seçimi en optimal çözümü bulmak için en etkin yöntemdir. Bu seçim yöntemini kodlamak ve uygulamak çok basittir (Goldberg ve Miller 1995).

Elitistik Seçim: Elitist seçimin anlamı; uygunluk derecesi en yüksek olan bireyin değişikliğe uğramadan, yani çaprazlama ve mutasyona uğramadan, hiçbir fonksiyon ile seçim uygulamadan yeni nüfusa direkt olarak kopyalanmasıdır.

3.11 Sonlandırma Koşulları

GA parametre seçimi yapıldıktan sonra kod çalıştırılır. Peki algoritma hangi koşulda sona ermelidir? İşte algoritmanın sonsuza dek çalışmaması, en uygun sonuçtan uzaklaşmaması, en uygun sonuca en yakın zamanda ve en uygun sürede varması veya daha uygun bir sonuç bulunması için oluşturulan koşullara sonlandırma koşulları adı verilir. GA ait olan parametreler kullanılarak sonlandırma koşulları uygulanır. Yaygın kullanılan sonlandırma koşulları:

- Jenerasyon sayısı: Önceden belirlenmiş adım sayısı sonunda GA durdurulur. Yeni nesil oluşturma işlemi bir adım anlamına gelmektedir. Standart bir optimizasyon için 100 adım uygun bir değerdir.
- Süre limiti: Optimizasyon için süre belirlemek tabiki mümkün değildir. Ama birçok parametrenin optimizasyonu yapıldığında yada sürenin genel kodlamada önem arzettiği durumlarda süre limiti koşulu konulabilir. Algoritma belirlenen süre aşıldığında durdurulur.
- Uygunluk değeri limiti: Algoritma çalışma esnasında önceden belirlenmiş uygunluk değeri limitini, yine önceden belirlenmiş olan koşula göre aşarsa veya altına düşerse arama sonlandırılır.
- Uygunluk süre limiti: Algoritma çalışma esnasında önceden belirlenmiş uygunluk değeri limitini, yine önceden belirlenmiş olan koşula göre aşarsa veya altına düşerse arama sonlandırılır.
- Teknik programlama yazılım firmalarından Mathworks, Matlab yazılımı GA aracına tolerans fonksiyonu adlı ek sonlandırma koşulu eklemiştir (Mathworks, 2010).

3.12 Algoritma Performansını Etkileyen Faktörler

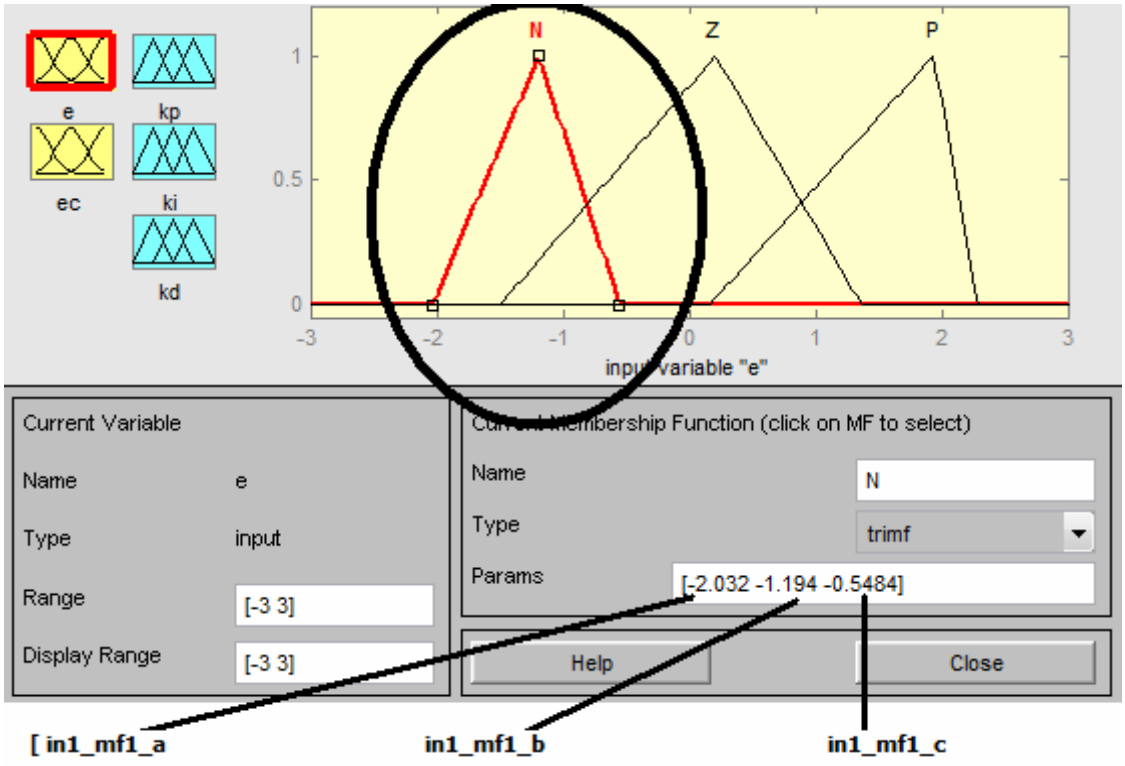
GA performansını etkileyen faktörler yani etkin çalışmasını sağlayan değerler bu algoritmaya ait parametrelerdir. GA için gerekli en uygun parametre seçimi için yüzlerce çalışma yapılmıştır fakat genel kullanım için parametre dizisi bulunamamıştır (Smith, 2004). Günümüzde parametre seçimi için yapay zeka teknikleri sıkça kullanılmaktadır. GA parametreleri çaprazlama ve mutasyon olasılığı, kromozom göserimi populasyon büyüklüğü, jenerasyon sayısı, seçim stratejisi ve en önemlisi uygunluk fonksiyonu tanımlanması işlemleridir. Bu değerlerin tümü arama yapılan problemden probleme farklılık göstermektedir.

- Uygunluk fonksiyonu tanımlaması ve başarı değerlendirmesinin nasıl yapıldığı: Akıllıca düşünülmemiş ve kodlanmamış bir değerlendirme fonksiyonu, algoritmanın çalışma süresini arttıracağı gibi sonuca hiçbir zaman ulaşmamasına sebebiyet verebilir. Bir sonuca varılsa bile kullanılan sonuç sistemin kararsız olmasına sebep olur.
- Jenerasyon ve popülasyon sayısının seçimi: Yanlış seçilmiş bir jenerasyon sayısı işlem süresini artırır veya yakınsama asla gerçekleşmez. Popülasyon sayısının seçimi de sistem hızına doğrudan etkiyen faktördür.
- Kromozom adeti: Dizi (kromozom) adetlerini çoğaltmak sürenin arttırılması açısından avantajdır; lakin azaltmak kromozom türlülüğünü ortadan kaldırmaktadır. Kromozom sayısı problemin en başında net bir şekilde belirlenmelidir, sonradan yapılacak değişiklik problemde en başa dönme ile eşdeğerdir.
- Kodlama gösteriminin nasıl yapıldığı: Ayrık sistemlerde ikili kodlama yapmanın avantaj olduğu belirtilmiştir. Gerçek gösterimde mutasyon işleminde değer ikili sisteme göre daha çok etkilenir. Bununla beraber ikili kodlamada işlemin hem başında hem de sonunda değer çevrimi yapılması işlem süresini arttıracaktır.
- Kaç noktalı çaprazlama yapılacağı: Normalinde çaprazlamanın tek noktada oluşmasına rağmen yapılan incelemelere göre bazı problemlerde çift noktalı çaprazlamanın daha faydalı olduğu belirlenmiştir.
- Mutasyon oranı: Kromozomlar birbirine benzerken, çözüm noktalarının uzağında bulunuyorsa, mutasyon yöntemini kullanmak GA için en elverişli çözümdür. Yüksek bir değer vermek sonucun kararlı bir noktaya varmasını engelleyecektir.

4. TASARIM ve BENZETİM SONUÇLARI

4.1 Genetik Algoritma ile Bulanık Kontrolör Parametre Optimizasyonu

Bulanık PID ve klasik PID karşılaştırmanın yeterli koşullar sağlanmadıkça mümkün olmadığını belirtilmiştik. Bulanık sistemler; kolay anlaşılabilirliği olmasına rağmen parametreler ve üyelik fonksiyonlarını oluşturmada sistem hakkında çok ileri seviyede bilgi gerektirmektedir. Bir bulanık kontrolör tasarımında, kontrolörün sağlıklı çalışabilmesi için, kontrol edilecek sistemin iyi incelenmesi ve tanınması gerekir. Bu sebeple bulanık PID kontrolörler genelde deneme yanılma yöntemi ile geliştirilmektedir. İşte bu noktada genetik algoritmanın "adaptif" üstünlüğü devreye girmektedir. Sistemin çıkışı ile istenen değer farkına (hata parametresi) göre kendi parametrelerini yeniler ve kontrolöre uygular.



Şekil 4.1 GA ile optimizasyon sonucu oluşan üyelik fonksiyonu

GA tabanlı bulanık PID kontrolörler için birçok metod oluşturulmuş ve devam eden birçok çalışma bulunmaktadır. Bu çalışmada amaç; GA ile arama yaparak bulanık kontrolöre ait üyelik fonksiyonlarını ve kural tablolarını optimize etmek ve sonrasında sisteme bu kontrolörü uygulamaktır. Oluşturacağımız bulanık kontrolör iki giriş, üç çıkış, dokuz kural ve üç üyelik fonksiyonuna sahiptir. Yapılması amaçlanan kontrolörün en önemli özelliği hiç şüphesiz sade yapısıdır. Dokuz kural ve iki giriş üç çıkışlı ayarlanabilir parametreli bulanık

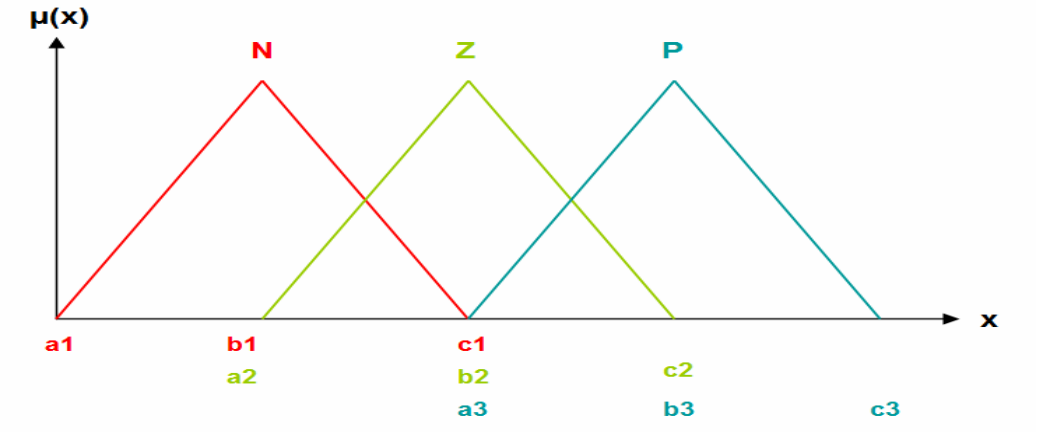
kontrolör kullanılıyor. Kontrol işareti için kapalı döngü çevrim lineer olmayan parametreler şeklinde tanımlanmıştır. Kesin sonuç verecek bir bulanık PID kontrolör oluşturulması amaçlanmıştır. Bulanık PID kontrolör kendisine ait doğrusal olmayan fonksiyonlarından doğrusal bir sinyal oluşturacak şekilde düşünülmüştür. Sistem üzerinde GA ile doğrusal olmayan bir arama yapılmaktadır. Problemimizdeki amaç sistemdeki hatayı minimize etmek olduğu için "uygunluk fonksiyonumuz" IAE kullanıldı. Kullanılan transfer fonksiyonlarımızda set değerinin altındaki ve üstündeki tüm hataları istenildiğinden, çok ani pikler olmadığından bu seçim yapıldı.

Bulanık kontrolörümüzün iki adet giriş ve üç adet çıkışa sahip olduğunu belirtmiştik.

Giriş değişkenleri: *e* (hata) , *ce* (hata değişimi)

Çıkış değişkenleri: K_p (oransal kazanç) , K_i (entegral kazanç) , K_D (türevsel kazanç)

Bulanık kontrolördeki iki adet giriş *e* ve *ce* ; üç adet çıkış K_p, K_i, K_d üçer adet "üçgen" üyelik fonksiyonlarına sahiptir. Bunlar N (negative/ eksi), Z (zero/ sıfır) , P (positive/ artı) şeklinde dilsel olarak ifade edilmiştir. GA üyelik fonksiyonları için sınır değerleri araması için elimizde bir bulanık kontrolör blok bulunması gerekmektedir. Bu blok hem optimize edilecek hemde sonuçlarımızda karşılaştırmada kullanılacaktır. Tüm işlemler öncesinde blok otomatik olarak oluşturulmalıdır. Optimize edilmiş bulanık kontrolör bloğu ile eşit şartlarda karşılaştırma yapılabilmesi için uygun üyelik fonksiyonları ile birlikte değerlendirilmelidir. Program içerisinde girilen sınır değerleri için otomatik bulanık kontrolör bloğu oluşturulur. Çünkü elle girilmiş değerler yada sistemle ilgisiz olan değerler sistemin oturma süresini uzatabilir ya da tamamen kararsız hale getirebilir. Aşağıdaki şekilde kontrolörde kullanılan üç üyelik fonksiyonu ve üyelik fonksiyonlarına ait değer ekseni üzerindeki sınırlar gösterilmiştir.



Şekil 4.2 Bulanık üyelik fonksiyonu genetik kodlama için hazırlanması

Problemimiz ayrık sistem olduğu için ve denemeler sonucunda daha iyi sonuç verdiği için ikili gösterim seçilmiştir. Gerçek gösterim daha doğal olduğu belirtilse de gerçek gösterimin birçok dezavantajı bulunmaktadır. Örneğin mutasyon işlemi sırasında birebir rakama müdahale edildiği için sonuçtan çok uzak bir rakam oluşabilir. Ama ikili gösterimde bu söz konusu değildir. Mutasyon işlemi sırasında sadece belirlenen bit işlemden etkilenir. İkili gösterimin tek dezavantajı olarak onluk-ikilik / ikilik-onluk çevirme süreleri belirtilebilir.

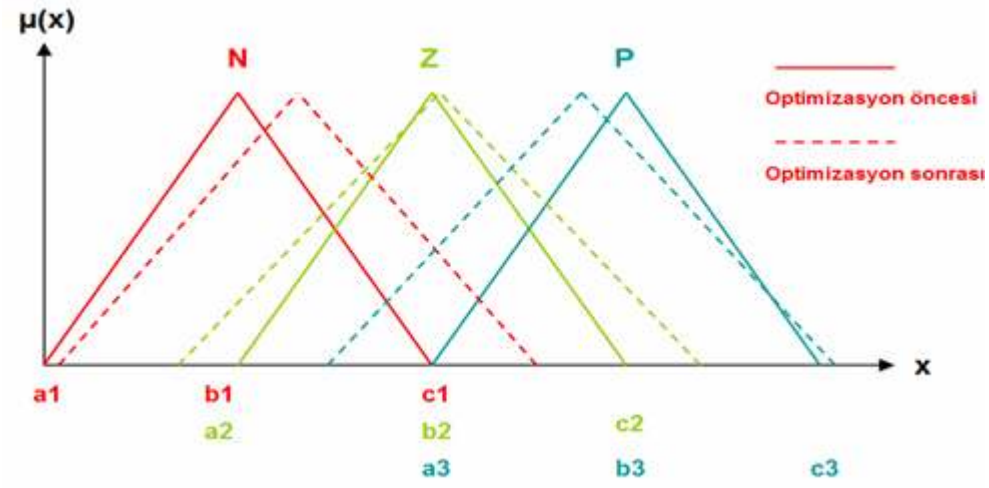
Üçgen üyelik fonksiyonuna ait sınır değerleri “x” eksenindeki sol sınır "a" orta noktası "b" ve x eksenindeki sağ sınırı "c" olarak ifade edilmiştir. Geliştirilen kod üzerinde değer hassasiyeti açısından her bir bölüm için standart olarak 5 bit (değiştirilebilir) yer ayrılıyor. Bir adet dilsel değişken için üç sınır değeri toplamı (a + b + c) 15 bit kullanılmaktadır.

Bu işlem tüm üyelik fonksiyonları için geçerlidir. Her bir giriş için üç adet üyelik fonksiyonu bulunduğundan toplam (15x3) 45 bit kullanılacaktır. Bulanık kontrolörümüzde iki giriş ve üç adet çıkış bulunduğu için (girişler *e* ve *ce*; çıkışlar K_p, K_i, K_d), toplam dizi uzunluğu (45x5) 225 bit olacaktır. GA üzerinde kullanılacak dizi uzunluğumuz, yani problemimizde bulmak istediğimiz optimum sonucun genetik gösterimi 225 bit olarak tanımlanır.

00001	01000	10101	...	11100	10101	00011
a1	b1	c1		a3	b3	c3
Giriş1	Giriş1	Giriş1		Çıkış3	Çıkış3	Çıkış3

Şekil 4.3 Üyelik fonksiyon sınırları için kromozom gösterimi

a1, b1, c1 = N (Giriş1) , a2, b2, c2 = Z (Giriş1) , a3, b3, c3 = P (Giriş1)

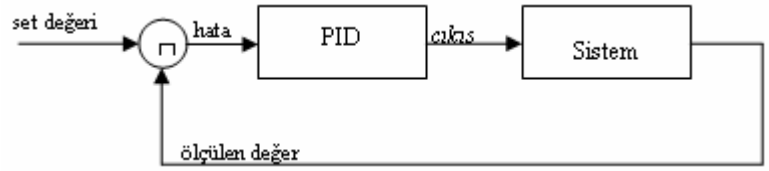


Şekil 4.4 Bulanık üyelik fonksiyonu sınırları, optimizasyon öncesi ve sonrası

Oluşturmuş kod varolan bir bulanık kontrolörün parametreleri üzerinden aramaya başlar ve belirtilen oluşturma sayısı süresince devam ederek bulanık kontrolör üyelik fonksiyonlarını ve kurallarını optimize eder. Bulanık kontrolörlerde kullanılacak g/ç fonksiyonlarının sayısının ve kural sayısının arttırılmasıyla yoluyla daha iyi değerler elde edilebilir. Bulanık kontrolör ile hatta GA tabanlı bulanık kontrolör ile klasik PID'den çok daha iyi performansa sahip kontrolör geliştirilebilir.

4.2 Bulanık Mantık Kontrolör ile PID Kazanç Katsayılarının Çıkarımı

PID, kontrol teorisinin ortaya çıkışından bu yana en çok kullanılan kontrol algoritmalarının başında gelmiştir. PID adını, proportional-integral-derivative kelimelerinin baş harflerinden almıştır. Türkçesi oransal-entegral-türev, kimi yerlerde oransal-tümlevsel-türevsel olarak bilinmektedir. PID kontrol; bu üç temel kontrol etkisinin üstünlüklerini tek bir birim içinde birleştiren etkidir. Entegral kontrolü, ortaya çıkabilecek kalıcı hal hatasını sıfırlarken türevsel kontrol, sadece PI kontrol kullanılması haline göre sistemin aynı bağıl kararlılığı için cevap hızını artırır. PID kontrol sistemde sıfır kalıcı durum hatası olan hızlı sistem cevabı sağlar. Katsayılarının ayarı iyi yapılmış PID kontrolör herhangi bir doğrusal sistemin hassas kontrolünde oldukça başarılıdır. Zaten kazanç katsayılarının belirlenmesi işlemi PID tasarımındaki en önemli süreçtir.



Şekil 4.5 Pid genel gösterim

Günümüzde geliştirilmiş kontrolörler PI, öz örgütlemeli PI, iki aşamalı PI, model tabanlı, adaptif network, kazanç-programlamalı PID, kendi ayarlamalı PID bunlardan birkaçıdır.

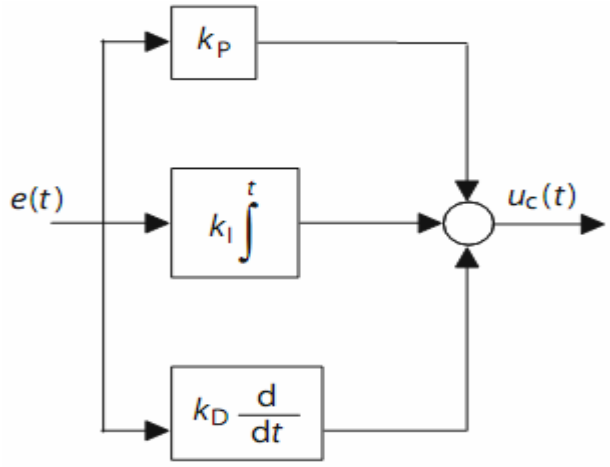
$$u(t) = K_P e(t) + K_I \int e(t)d(t) + K_D \frac{de(t)}{dt} \tag{4.1}$$

u(t) kontrol sinyalidir, hata sinyalinin belli bir katına getirilir. K_p oransal kontrolör kazancı olarak adlandırılabilir. *i(t)* kontrol sinyali, o andaki zamana bağlı olarak hata sinyalinin toplamları şeklinde, K_i sabitiyle ifade edilen bir katıdır. Burada K_i entegral kontrolör kazancını göstermektedir. Türev alıcı *d(t)* kontrol sinyali, hata sinyalinin aynı andaki değişim oranıdır ve K_d sabitiyle ifade edilir (Kıyak, 2008).

PID kontrolör tasarımında istenilen tepkiyi elde etmek için aşağıdaki adımlar izlenir:

- Açık döngü tepkisi bulunur ve ihtiyaçlar belirlenir.
- Yükselme zamanını düzeltmek için oransal kontrolör eklenir.
- Aşmayı düzeltmek için türevsel kontrolör eklenir.
- Kararlı hal hatasını yok etmek için entegral kontrolör eklenir.
- İstenilen tepki elde edilene kadar K_p, K_i ve K_d ayarlanır (Aydoğdu, 2006).

Hata büyük olduğu anlarda kontrolörün oransal kazancı odukça büyük olmalı fakat aşırı yükselmeleri önlemek için integral kazancı oldukça küçük olmalıdır. Buna göre kural olarak; Sistem osilasyonunu azaltmak için küçük oransal kazanç ve sürekli hal hatasını ortadan kaldırmak için ise büyük integral kazanç kullanmalıdır. Bununla birlikte PID algoritması, endüstriyel uygulamalarda çok sık karşılaşılan, ayar noktasının değişimi, çalışma şartlarının değişimi, sistemin durdurulup tekrar çalıştırılması ve dış etkilerin olması gibi durumlarda, prosesin değişik kriterlere göre optimumda kontrol edilmesine engel oluşturabilmektedir. Yani doğrusal olmayan sistemlerde PID kontrolör yetersiz kalmakta ve sistemi kararsızlığa götürmektedir (Küçükdemiral ve Cansever 2006).



Şekil 4.6 PID kontrolör ait genel blok şeması

Öncelikle çözülecek problem için bulanık mantık yaklaşımının doğru bir seçenek olup olmadığına karar vermeliyiz. Eğer uygulanacak sistemin davranışı kurallarla ifade edilebiliyorsa veya karmaşık bir matematiksel işlem gerektiriyorsa, bulanık mantık yaklaşımı uygulanabilir. Bulanık PID benzeri kontrolörler ile ilgili araştırmalara bakıldığında bu çalışmada tasarım parametrelerini yapısal parametreler ve ayar parametreleri olarak iki grupta sadeleştirildi. Yapısal parametreler ve ayar parametreleri GA vasıtası ile çevrim dışı (eş zamanlı olmayan) olarak ayarlanmakta ve kontrolör parametreleri işlenmektedir. Sabit olacak

bulanık kontrolör parametreleri ve bizim GA ile arama yapacağımız "ayarlanabilir parametreler" tabloda belirtilmiştir.

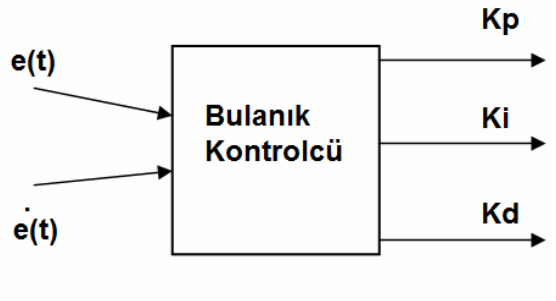
Çizelge 4.1 Tasarım parametreleri

Yapısal Parametreler	**Ayarlanabilir parametreler**
Giriş Değişkenleri	Hata (e), hata değişimi (ce)
Çıkış Değişkenleri	K_p, K_i, K_d
Bulanık Kural Tablosu	Öncül, artçıl
Üyelik Fonksiyonları	Üyelik fonksiyonu sınırları (a_x, b_x, c_x)
Dil kümeleri	Kural tablosu parametreleri

Kontrolör giriş olarak hata ve hata değişimini kullanır.

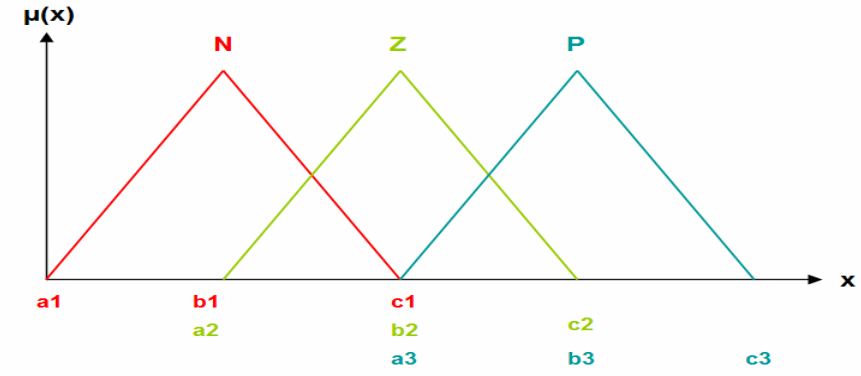
e (hata) = (istenen konum – o andaki konum)

de (hata değişimi) = (şimdiki hata – bir önceki hata)



Şekil 4.7 Bulanık kontrolör g/ç (Li-Xin Wang, 1997)

Hesaplamalarda kolaylık sağlaması amacıyla üçgen üyelik fonksiyonu kullanılmıştır. Tüm giriş ve çıkışlar için kullanılan üyelik fonksiyonları ve dilsel değişkenler N (Negative/ Negatif), Z (Zero/ Sıfır), P (Positive/ Pozitif)



Şekil 4.8 Kullanılan üyelik fonksiyonları gösterimi

Her sistem için kural tablosu farklı olacaktır. Bizim bunu bilmediğimizi ya da tek bir sisteme göre yapmadığımızı düşünürsek buna sistem kendisi karar verecektir. Eğriyi kendisi görecek ve ona göre ayarlayacaktır. Bizim kontrolörümüz ise yerine göre K_p ayarı verip yerine göre K_p+K_i yerine göre K_p+K_i+K_d ayarlaması yapmaktadır. Geliştirilen algoritma üzerinde bu şekilde esneklik sağlanmıştır. Bu bulanık kontrolördeki kural tabanı sayesinde gerçekleştirilmektedir.

Sistemde üç dilsel değişken ve iki giriş bulunduğundan sistemin ilk halinde oluşturulacak örnek kural listesi aşağıdaki şekilde olacaktır:

Eğer hata "N" ve hata_türevi "N" ise Kp "N" Ki "N" Kd "N"
Eğer hata "N" ve hata_türevi "Z" ise Kp "N" Ki "Z" Kd "Z"
Eğer hata "N" ve hata_türevi "Z" ise Kp "N" Ki "Z" Kd "Z"

Bulanık kontrolör üyelik fonksiyonlarına ait sınır değerleri sistemden sisteme değişiklik göstermekle beraber tecrübe ve deneme yanılma yöntemleri ile belirlenmiştir. Buda bulanık mantığın felsefesinden ileri gelir. Bu çalışmanın amacı GA ile optimum sınırların belirlenmesi ve uzman ihtiyacının devre dışı bırakılmasıdır.

4.3 Benzetim Sonuçları

Örnek Sistem1 $sys = \dfrac{400}{s^2 + 50s + 5}$

Oluşturmak istediğimiz kontrolör sistem davranışını kontrol etmek için optimum kontrol işaretini üretir. Kontrolör bulanık PID yapısı olmakla birlikte belirtilen her bir giriş ve çıkış sınır değerleri için optimizasyon yapılmaktadır. Yapılan benzetimler sırasında, Pentium Core2Due 2.2 GHz işlemci ve 2 GB RAM bilgisayar ve MATLAB programının 2009b sürümü kullanılmıştır.

Çizelge 4.2 Sistem1 - kontrolöre ait ga parametreleri

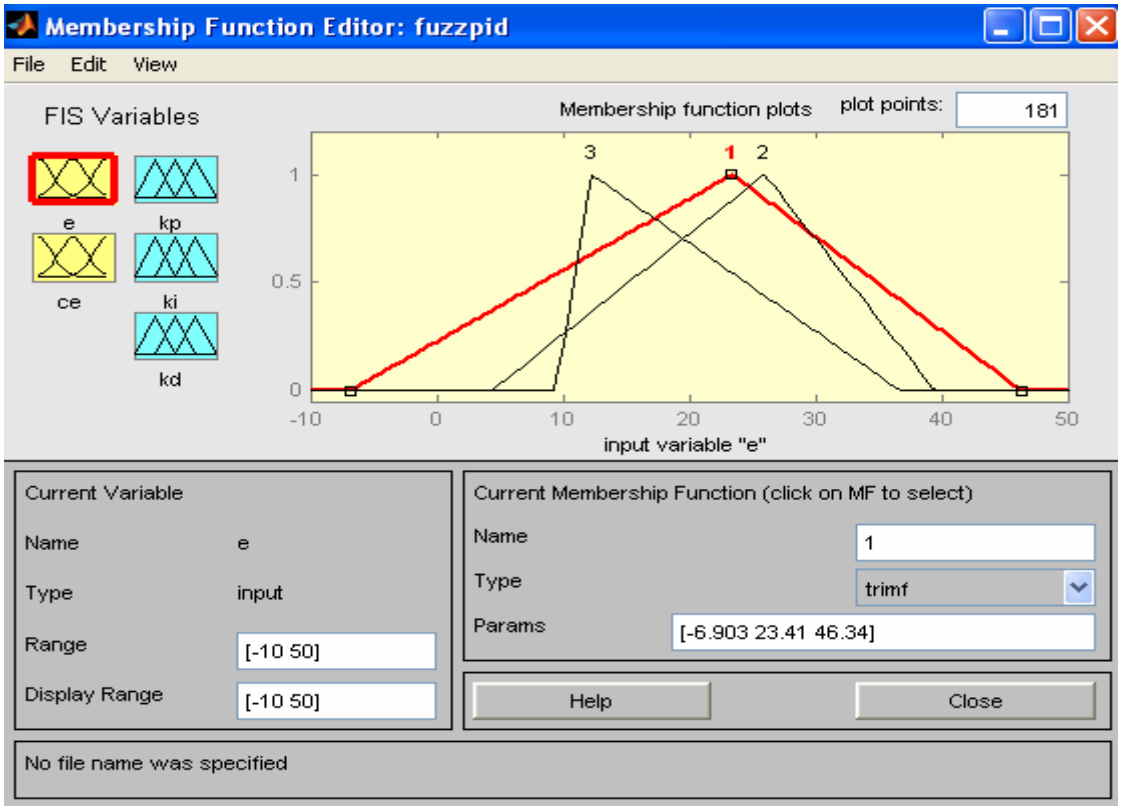
Parametreler	**Değer**
Popülasyon Büyüklüğü	15
Maksimum Jenerasyon Sayısı	15
Çaprazlama Yöntemi ve Oranı	İki Noktalı , Uniform
Mutasyon Oranı	0.001
Uygunluk Fonksiyonu	IAE

Sonlandırma Koşulu	Maksimum Jenerasyon Sayısı

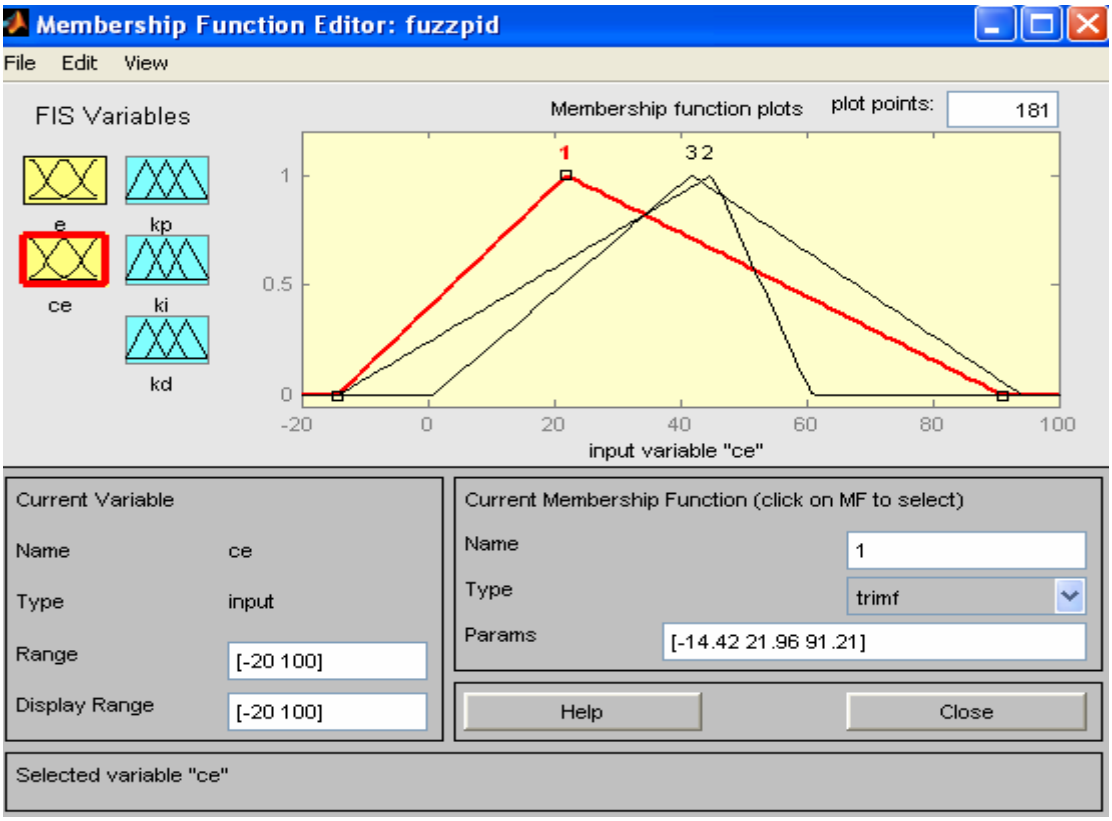
Çizelge 4.3 Sistem1 - bmk g/ç fonksiyonları için verilen sınır değerleri

Parametreler	**Sınırlar**
Giriş1 (*e*)	[-10,+50]
Giriş2 (*ce*)	[-20,+100]
Çıkış1 (K_p)	[2,+10]
Çıkış2 (K_i)	[0,+0.1]
Çıkış3 (K_d)	[0,+0.001]

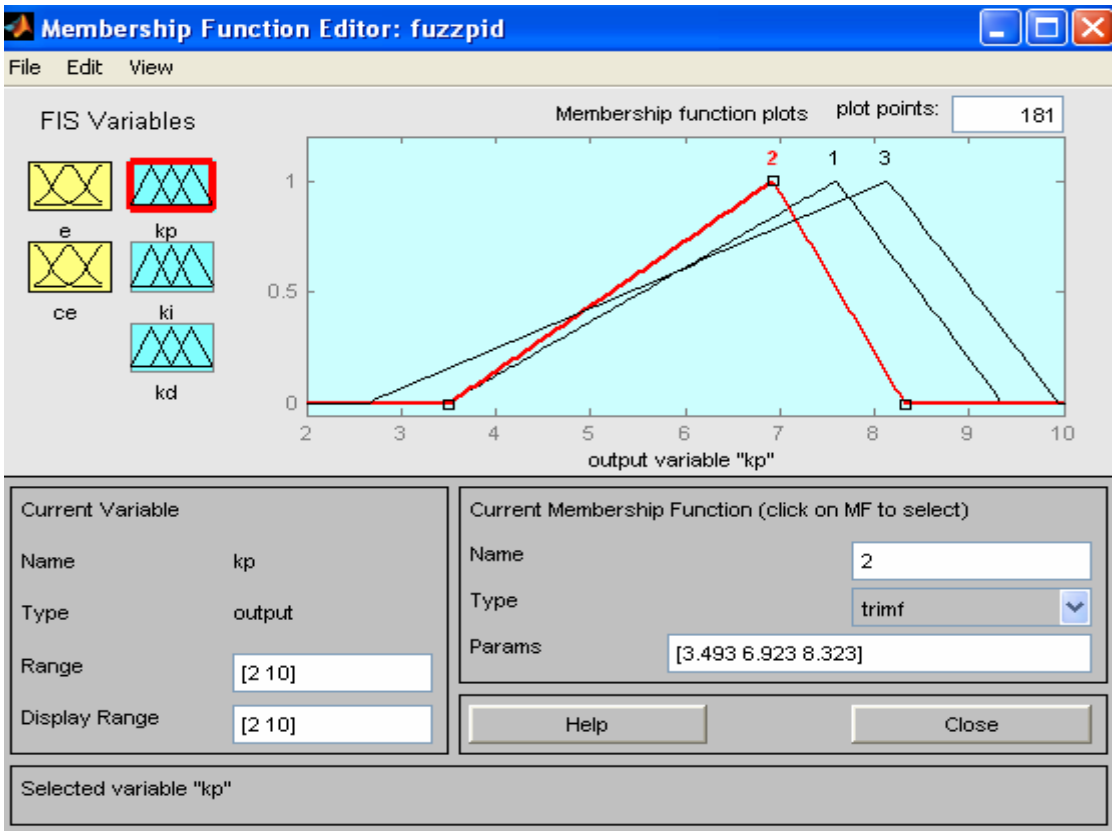
Klasik PID için manual deneme yapılıp uygun katsayılar bulunmuş; en iyi performan gösteren kontrolör sisteme uygulama yapılmıştır (K_p = 7.5 K_i =0.01 , K_d = 0.001). Genetik optimizasyon işlemi sırasında kullanılan parametreler Çizelge 4.2 ve Çizelge 4.3'de gösterilmiştir. Bulunan optimal kontrolör parametreleri ile bulunan bulanık mantık giriş ve çıkışlarına GA sonucu oluşan en iyi bireye ait üyelik fonksiyonları Şekil 4.9, 4.10, 4.11 , 4.12 ve 4.13'de gösterilmiştir.



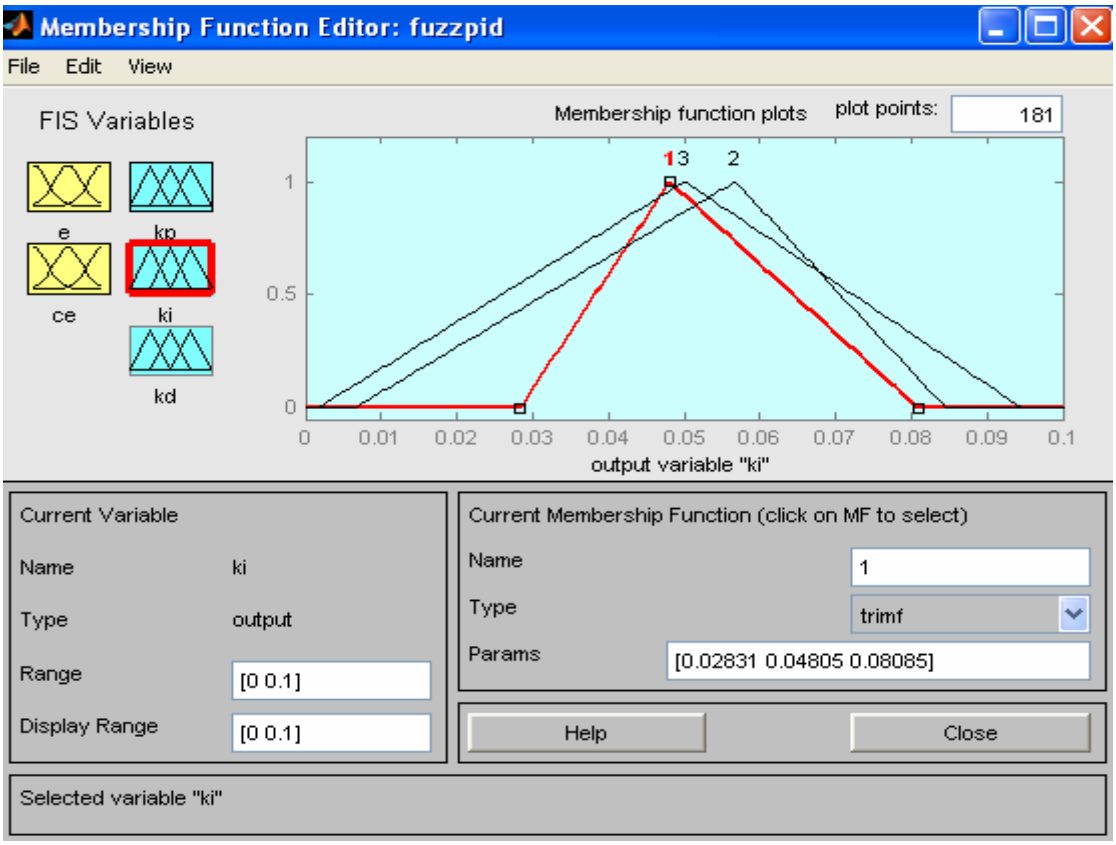
Şekil 4.9 Sistem1 – optimizasyon sonrası elde edilen giriş fonksiyonu (*e*)



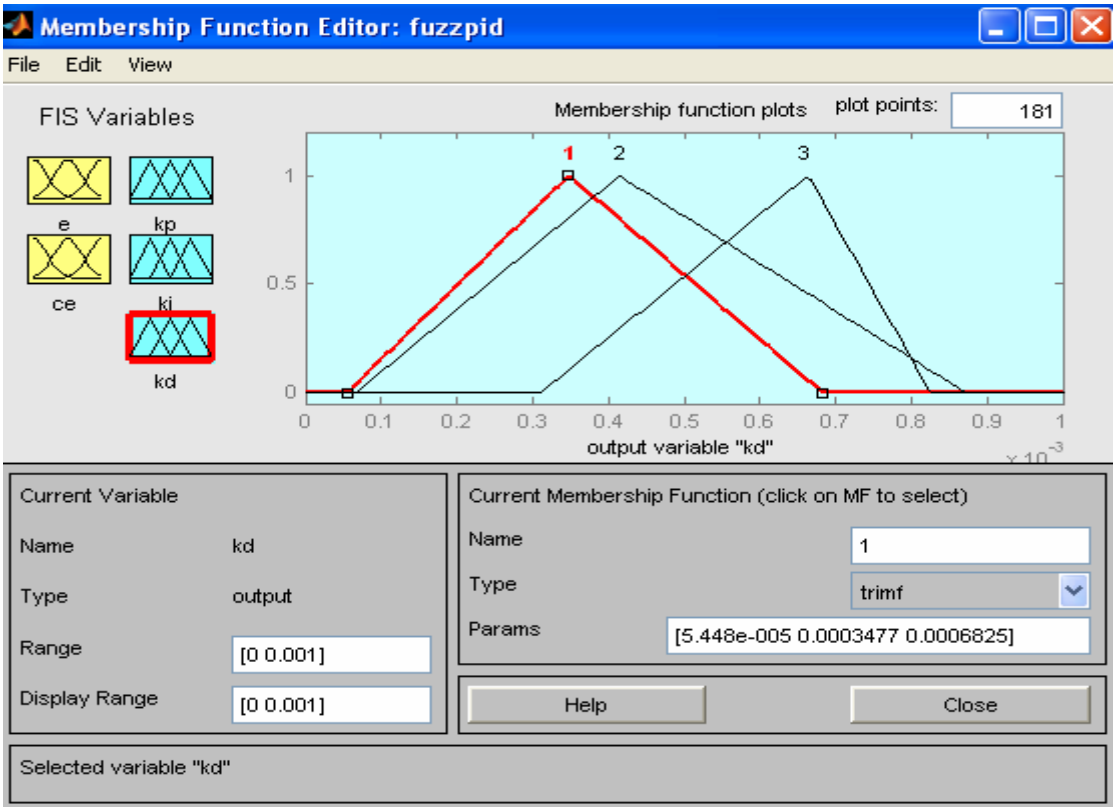
Şekil 4.10 Sistem1 – optimizasyon sonrası elde edilen giriş fonksiyonu (*ce*)



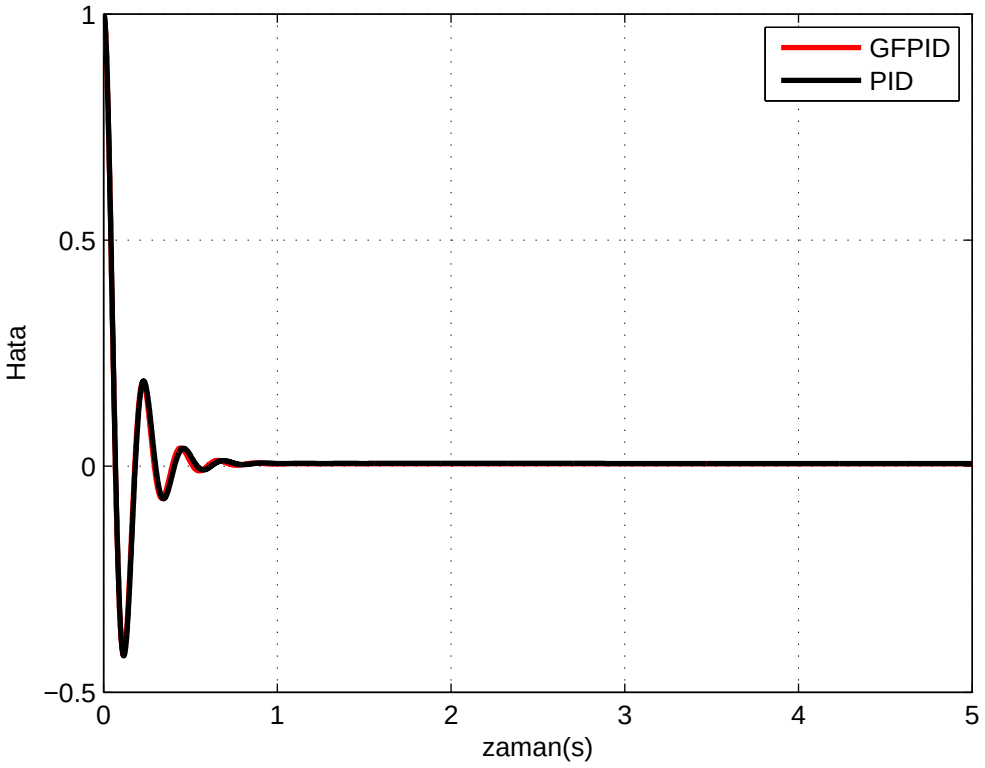
Şekil 4.11 Sistem1 – optimizasyon sonrası elde edilen çıkış fonksiyonu (*Kp*)



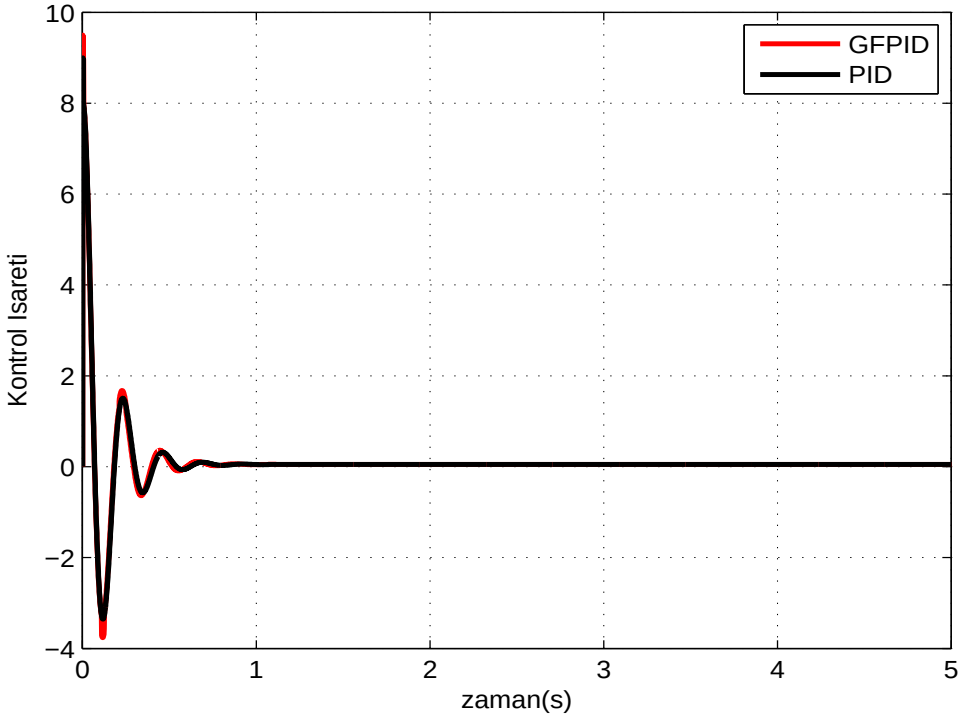
Şekil 4.12 Sistem1 – optimizasyon sonrası elde edilen çıkış fonksiyonu (*Ki*)



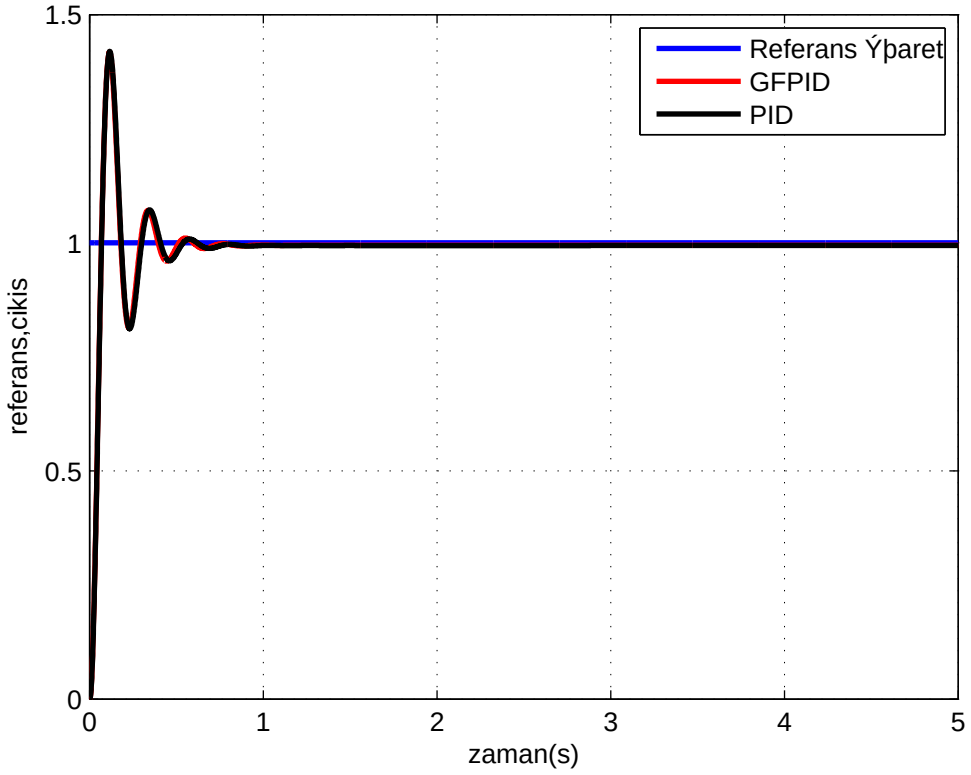
Şekil 4.13 Sistem1 – optimizasyon sonrası elde edilen çıkış fonksiyonu (*Kd*)



Şekil 4.14 Sistem1 – hatanın değişimi



Şekil 4.15 Sistem1 – kontrol işaretinin değişimi



Şekil 4.16 Sistem1 – sistem cevabı karşılaştırma (PID, FPID)

Oluşturulan Genetik Bulanık Kontrolörü klasik PID ile eşit şartlarda karşılaştırmak adına klasik PID için birkaç deneme yapılıp uygun katsayılar bulunarak sisteme uygulama yapılmıştır. Aynı işlem bulanık kontrolör için de geçerlidir. Uygulamada GA parametreleri kadar uygunluk fonksiyonuna ait döngü sayısı da önem arzetmektedir. Çünkü GA sırasında performans ölçütü olan uygunluk fonksiyonu IAE için kontrolör parametreleri sisteme belirli bir döngü sayısınca uygulanmaktadır. Bu döngü sayısı jenerasyon sayısından bağımsızdır. GA ile bulanık kontrolör ait üyelik fonksiyonlarının sınır değerleri ile birlikte kural tabloları da optimize edilmiştir. Tek başına üyelik fonksiyonlarının optimize edilmesi doğrusal olmayan sistemler için başarılı sonuç vermemektedir. Yapılan genetik optimizasyon sonucunda; klasik PID ve GA ile optimize edilmiş bulanık PID kontrolör (GFPID) sisteme uygulanmıştır. Benzetim sırasında sistemin basamak fonksiyonu şeklindeki referansa verdiği cevap şekil4.16'da gösterilmiştir. GFPID ve klasik PID eşit şartlarda aynı yükselme zamanına sahip PID üzerinde olması gerektiği gibi kalıcı hal hatasının sıfırladığı, üst aşımın olduğu görülmüştür. Aynı zamanda genetik optimizasyon yapılmış olan bulanık kontrolör ile klasik PID yerleşme süresinin aynı olduğu gözlemlenmiştir. Şekil 4.14'de FPID uygulamasındaki hata işaretinin değişimi ve Şekil 4.15 üzerinde FPID ait kontrolör işaretinin değişimi gösterilmiştir.

Örnek Sistem2 $sys = \dfrac{8152}{s^2 + 461.2s + 6}$

İlk oluşturduğumuz kontrolör ile aynı şekilde amaç sistem davranışını kontrol etmek için optimum kontrol işaretini üretmektir. GA parametreler aynı olmakla birlikte değişik parametreler ve sınır değerleri için optimizasyon yapılmış ve başarılı sonuçlar elde edilmiştir. GA parametrelerinden popülasyon büyüklüğü, maksimum jenerasyon sayısı ve uygunluk fonksiyonuna ait döngü sayısı değiştirilmiştir.

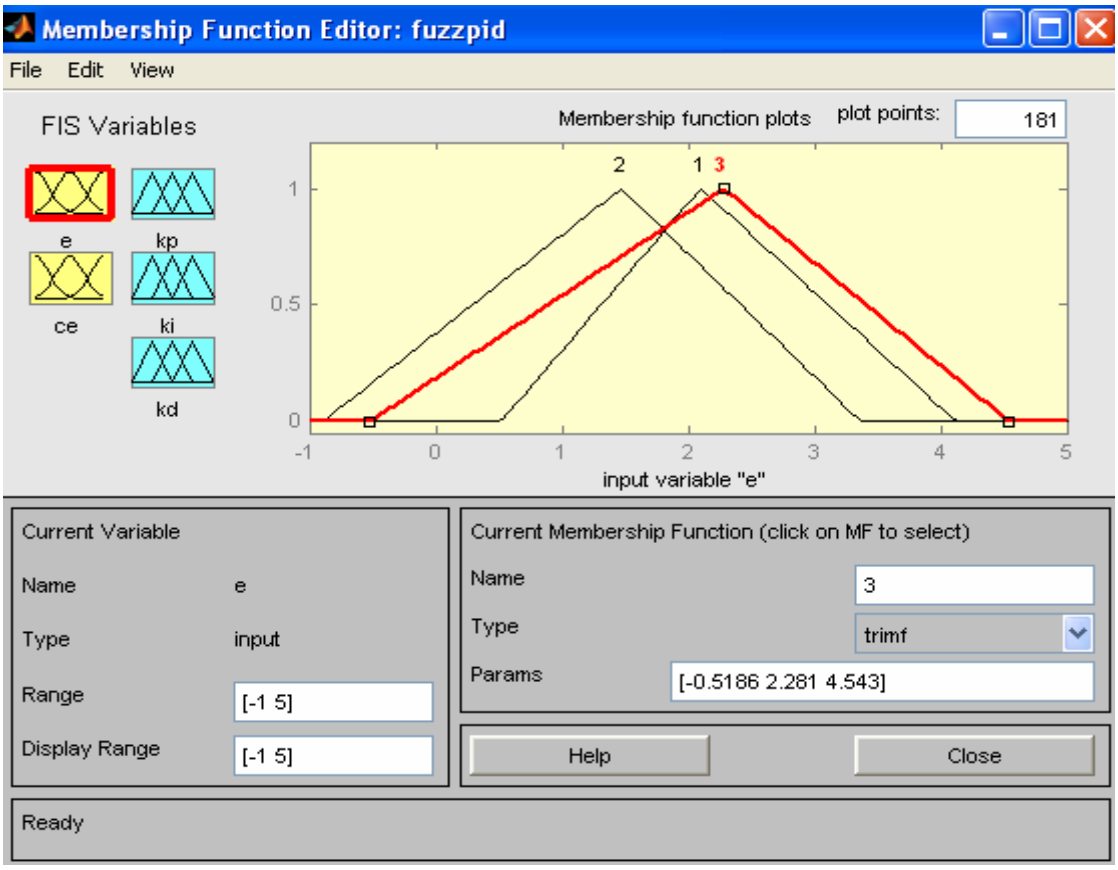
Çizelge 4.4 Sistem2 - kontrolöre ait ga parametreleri

Parametreler	**Değer**
Popülasyon Büyüklüğü	20
Maksimum Jenerasyon Sayısı	20
Çaprazlama Yöntemi ve Oranı	İki Noktalı , Uniform
Mutasyon Oranı	0.001
Uygunluk Fonksiyonu	IAE
Sonlandırma Koşulu	Maksimum Jenerasyon Sayısı

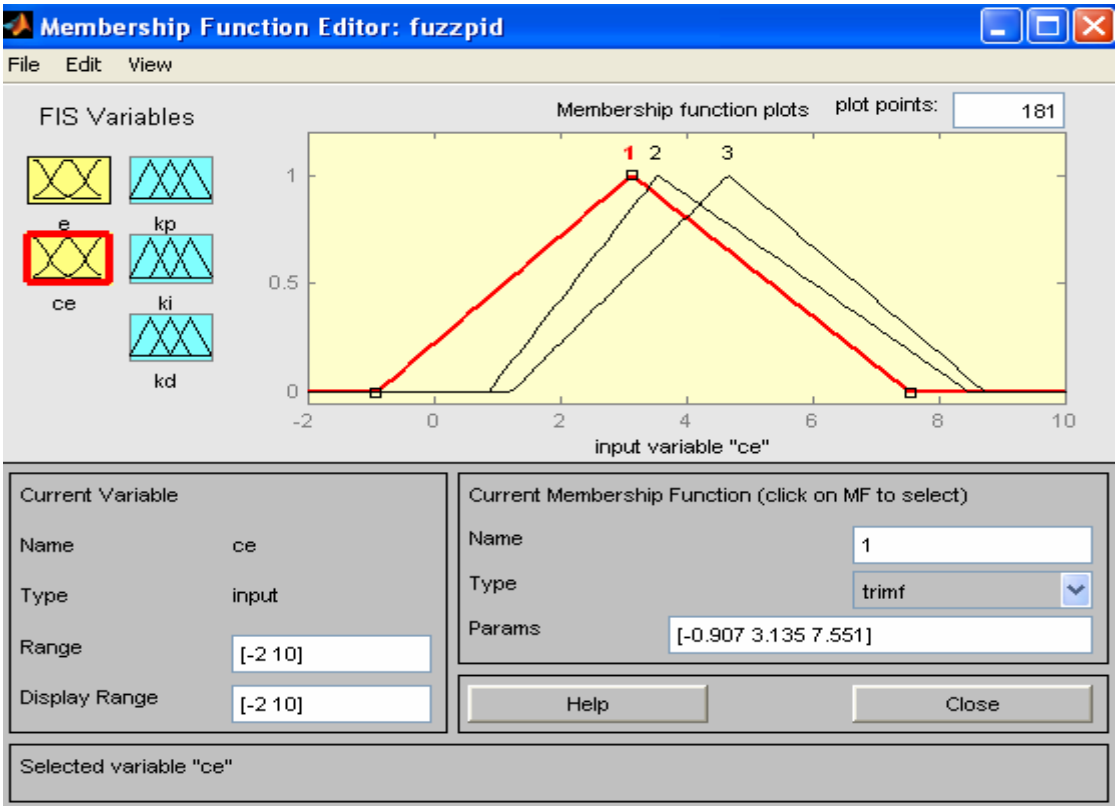
Çizelge 4.5 Sistem2 - bmk g/ç fonksiyonları için verilen sınır değerleri

Parametreler	**Sınırlar**
Giriş1 (*e*)	[-1,+5]
Giriş2 (*ce*)	[-2,+10]
Çıkış1 (K_p)	[0,+10]
Çıkış2 (K_i)	[0,+0.1]
Çıkış3 (K_d)	[0,+0.001]

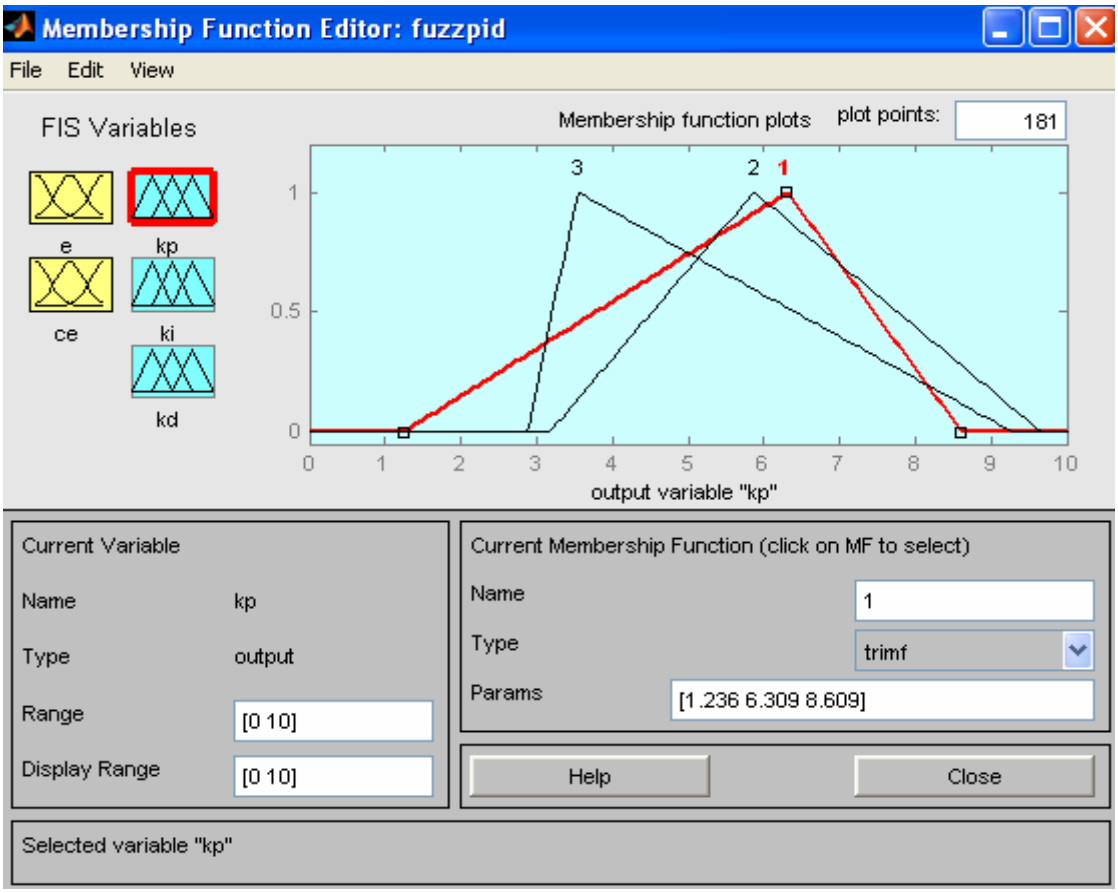
Önceki örnekte yapıldığı gibi; oluşturulacak GFPID ve klasik PID ile eşit şartlarda karşılaştırmak adına klasik PID için birkaç deneme yapılıp uygun katsayılar bulunmuş ardından sisteme uygulama yapılmıştır (K_p = 8 K_i =0.01 , K_d = 0.001). GA optimizasyon işlemi sırasında kullanılan parametreler Çizelge 4.4 ve Çizelge 4.5'de gösterilmiştir. BMK g/ç ait optimal kontrolör parametreleri ile GA sonucu oluşan en iyi bireye ait üyelik fonksiyonları Şekil 4.17, 4.18, 4.19 , 4.20 ve 4.21'de gösterilmiştir.



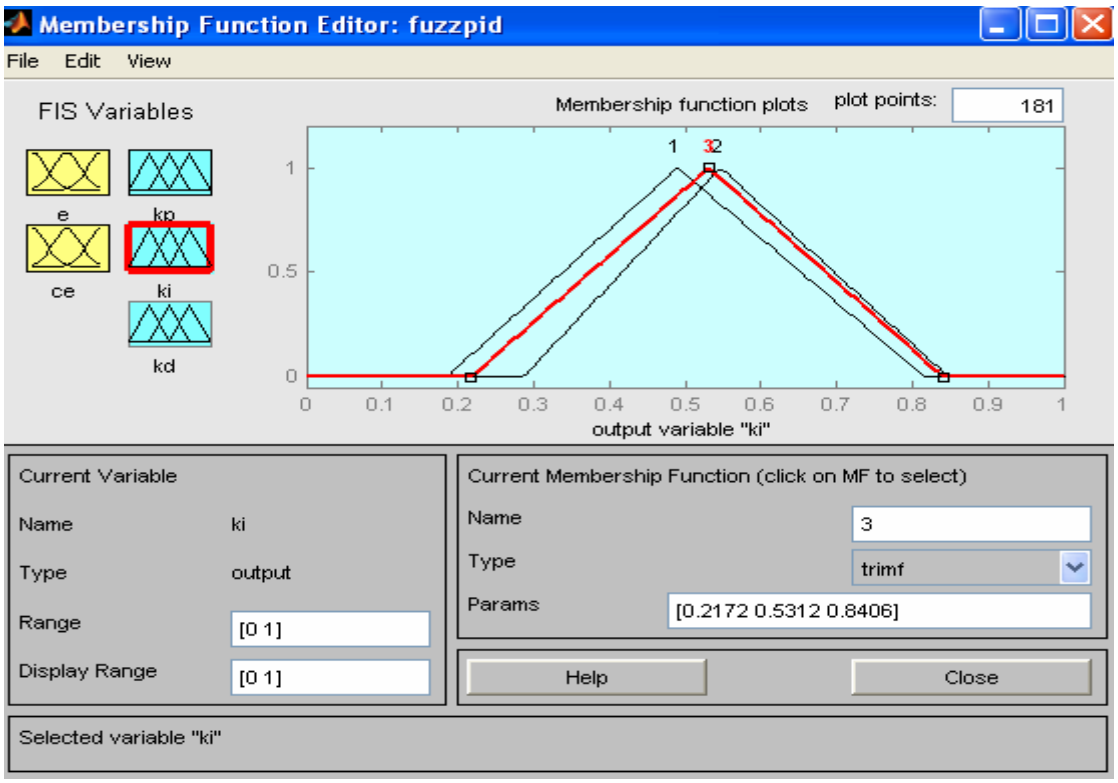
Şekil 4.17 Sistem2 – optimizasyon sonrası elde edilen giriş fonksiyonu (*e*)



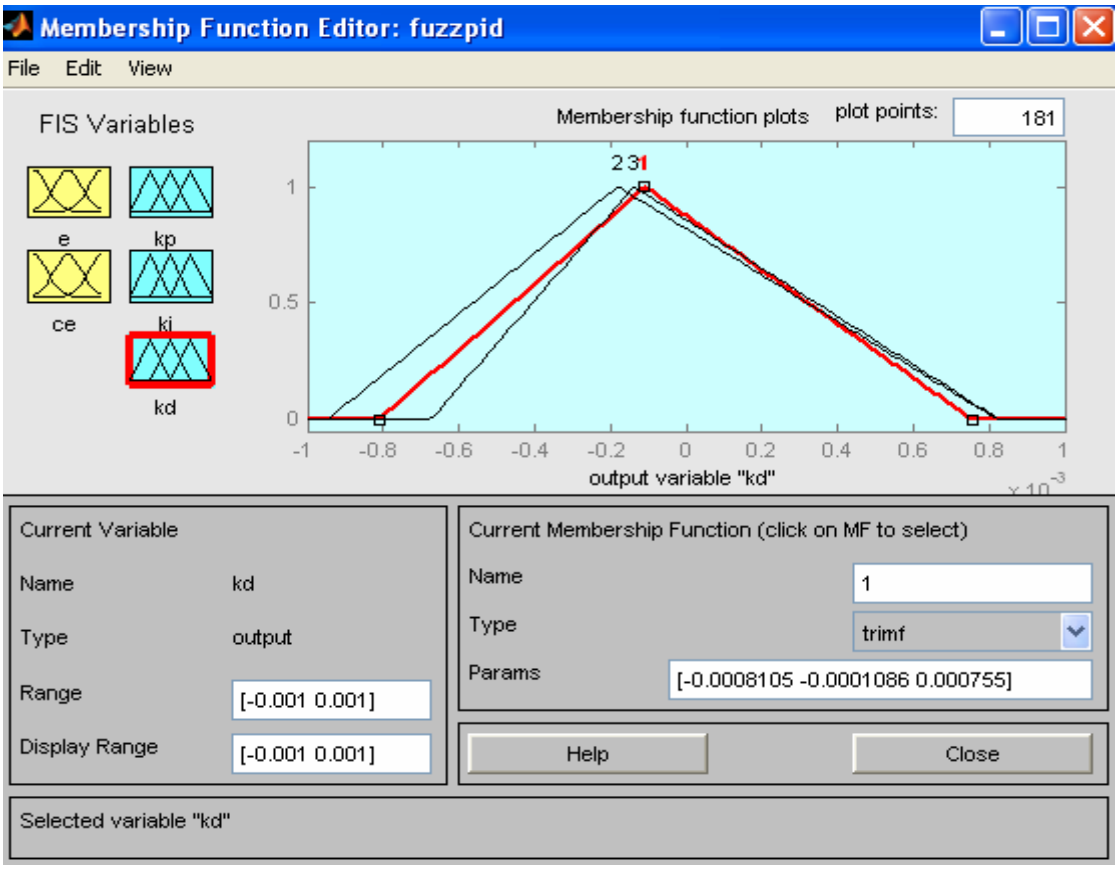
Şekil 4.18 Sistem2 – optimizasyon sonrası elde edilen giriş fonksiyonu (*ce*)



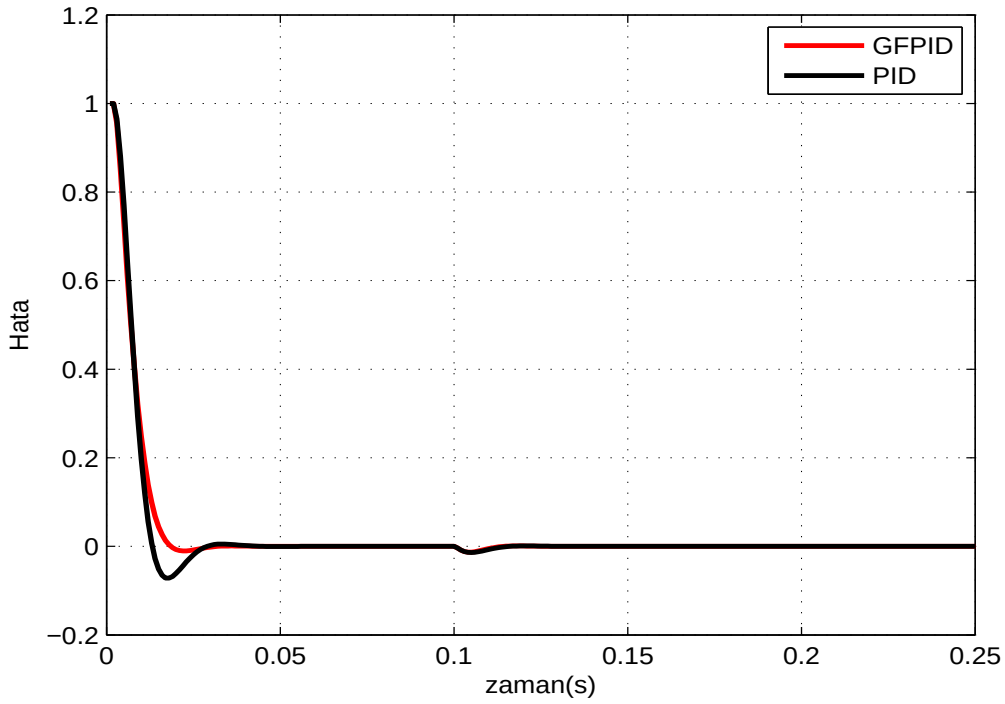
Şekil 4.19 Sistem2 – optimizasyon sonrası elde edilen çıkış fonksiyonu (*Kp*)



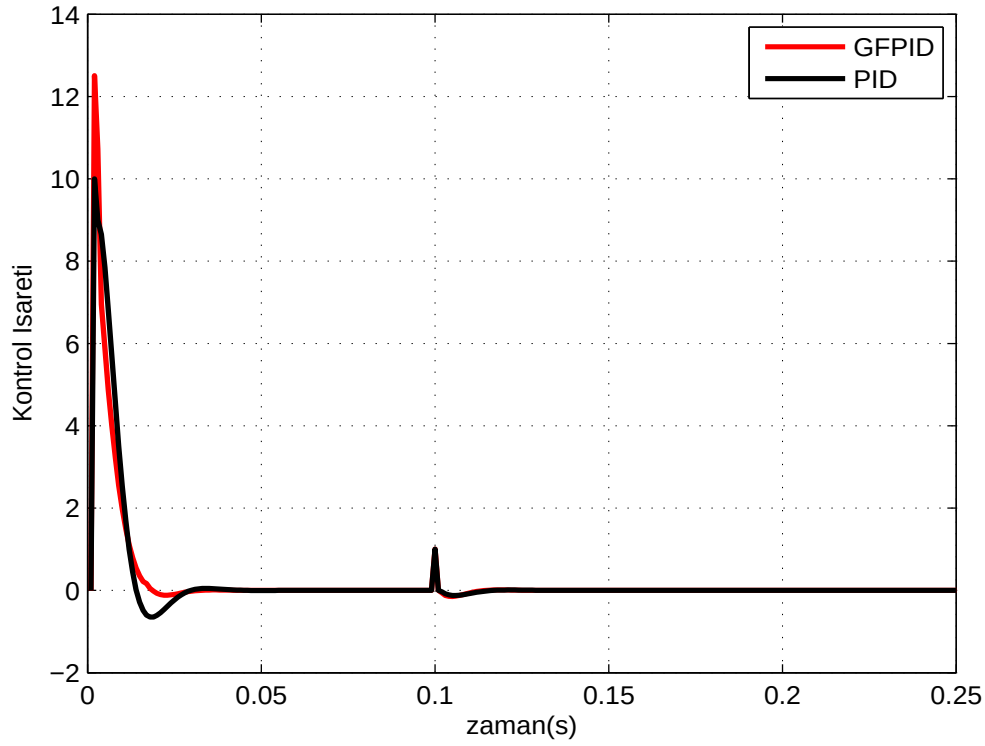
Şekil 4.20 Sistem2 – optimizasyon sonrası elde edilen çıkış fonksiyonu (*Ki*)



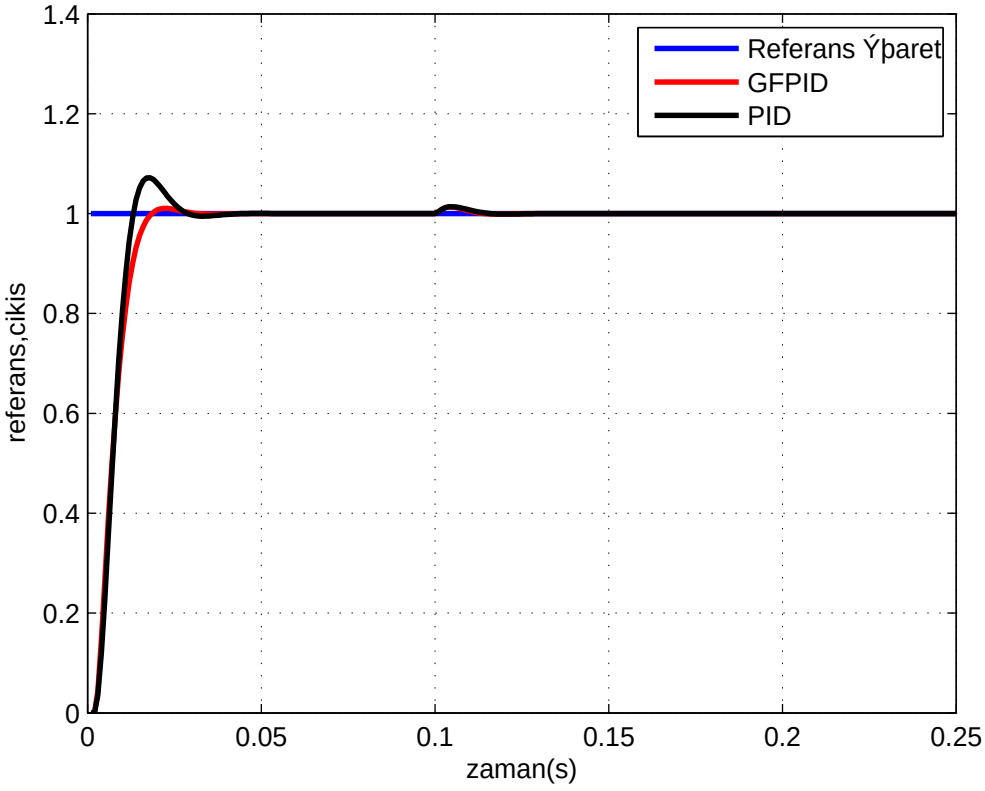
Şekil 4.21 Sistem2 – optimizasyon sonrası elde edilen çıkış fonksiyonu (*Kd*)



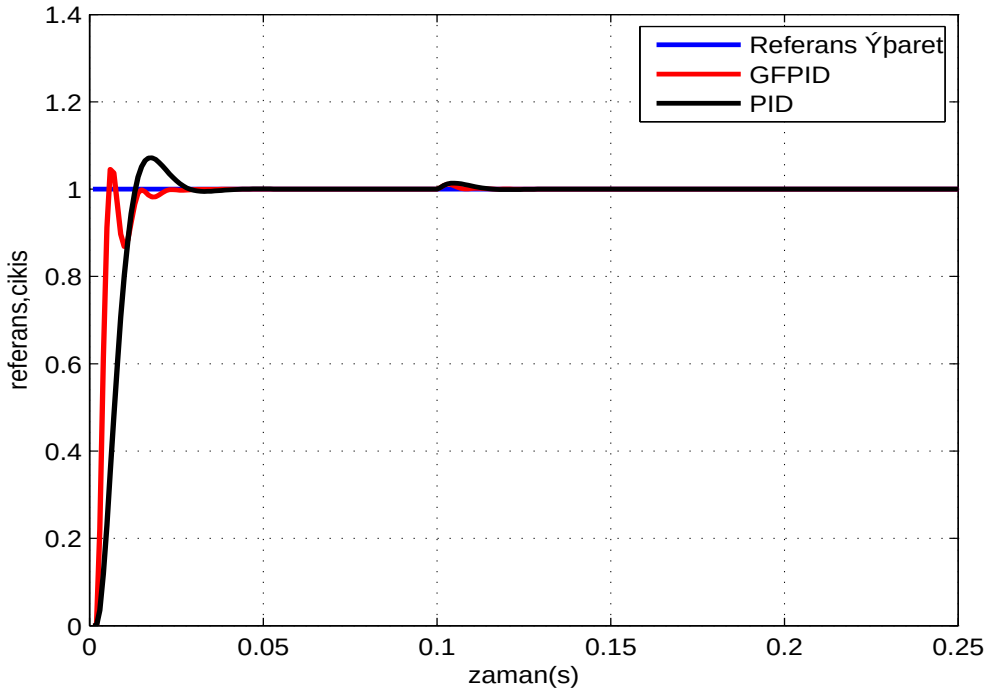
Şekil 4.22 Sistem2 – hatanın değişimi



Şekil 4.23 Sistem2 – kontrol işaretinin değişimi



Şekil 4.24 Sistem2 – sistem cevabı karşılaştırma (PID, FPID)



Şekil 4.25 Sistem2 – farklı parametreler için sistem cevabı karşılaştırma (PID, FPID)

Çizelge 4.6 Sistem2 - bmk g/ç fonksiyonları için verilen farklı sınır değerleri

Parametreler	**Sınırlar**
Giriş1 (*e*)	[-2,+4]
Giriş2 (*ce*)	[-3,+9]
Çıkış1 (K_p)	[3,+8]
Çıkış2 (K_i)	[0,+0.3]
Çıkış3 (K_d)	[0,+0.005]

Çalışmamızda mutasyon kullanılmaktadır ama oranı çok küçüktür. Mutasyon çeşitlilik için yapılmakta ve fayda sağlamaktadır fakat şuna dikkat edilmelidir ki; mutasyon oranının büyük seçilmesi bulanık mantığa ait üçgen üyelik fonksiyonu için değişmez kural olan “a>b” ifadesini “b>a” şeklinde bozabilir. Bunun için mutasyon oranının büyük seçilmesi durumunda kod içerisinde bulunan bit hassasiyeti yüksek seçilmeli ve ek kontrol koyulmalıdır. Referans ile cevabın farkındaki hata ve hatanın değişimine bulanık kontrolör uygulanırken limit konması gerekmektedir. Çünkü bulanık kontrolör hata ve hata değişimini giriş olarak almaktadır ve FPID sınırları içerisinde olmayan bir giriş değeri için çıkış değeri üretemez. Benzetimde kontrol işaretine 0.1s referans işareti kadar gürültü eklenmiştir. Şekil 4.24 üzerindeki sistem çıkışları gözlemlendiğinde GFPID üst aşımın klasik PID oranla yok

denecek kadar az olduğu izlenmiştir. Hatanın değişimi de klasik PID oranla daha başarılıdır. BMK için sınır değerlerinin değiştirilmesi sonrasında oluşturulan kontrolöre ait sistem cevabı karşılaştırması Şekil 4.25'te gösterilmiştir. GFPID ile daha hızlı sistem cevabı, sıfır üst aşım ve çok hızlı yerleşme süresi elde edilmiştir.

Örnek Sistem3 $sys = \dfrac{523500}{s^3 + 87.35s^2 + 10470s + 2}$

Üçüncü mertebeden örnek bir sistem üzerinde klasik PID ve GFPID uygulaması yapılmıştır.

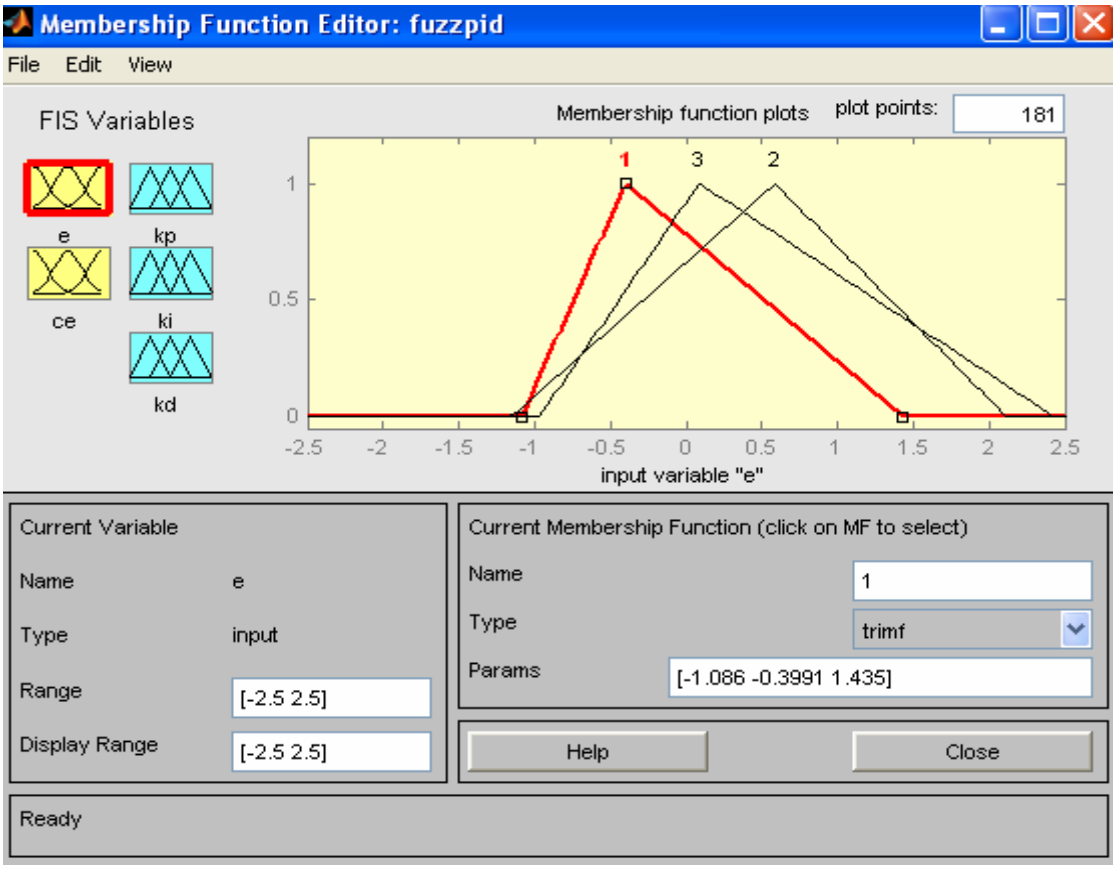
Çizelge 4.7 Sistem3 - kontrolöre ait ga parametreleri

Parametreler	**Değer**
Popülasyon Büyüklüğü	30
Maksimum Jenerasyon Sayısı	30
Çaprazlama Yöntemi ve Oranı	İki Noktalı , Uniform
Mutasyon Oranı	0.001
Uygunluk Fonksiyonu	IAE
Sonlandırma Koşulu	Maksimum Jenerasyon Sayısı

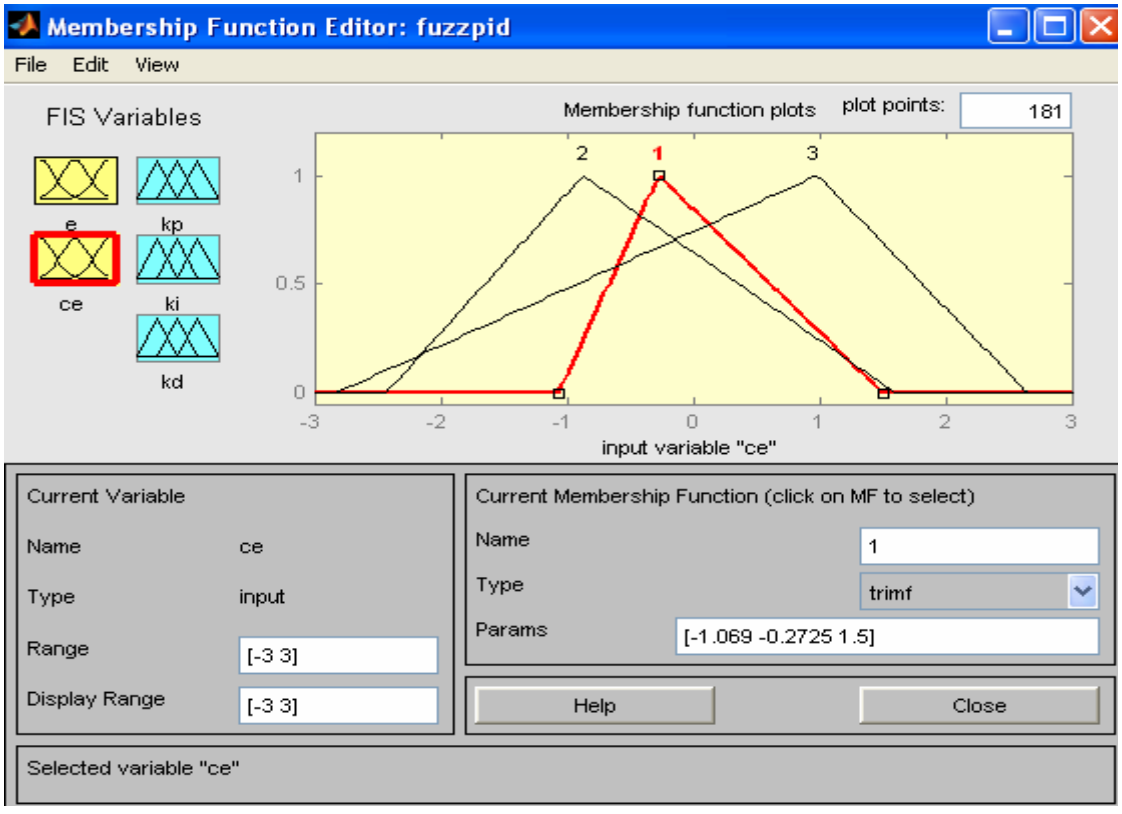
Çizelge 4.8 Sistem3 - bmk g/ç fonksiyonları için verilen sınır değerleri

Parametreler	**Sınırlar**
Giriş1 (*e*)	[-2.5,+2.5]
Giriş2 (*ce*)	[-3,+3]
Çıkış1 (K_p)	[+0.5,+2.8]
Çıkış2 (K_i)	[0,+0.3]
Çıkış3 (K_d)	[0,+0.003]

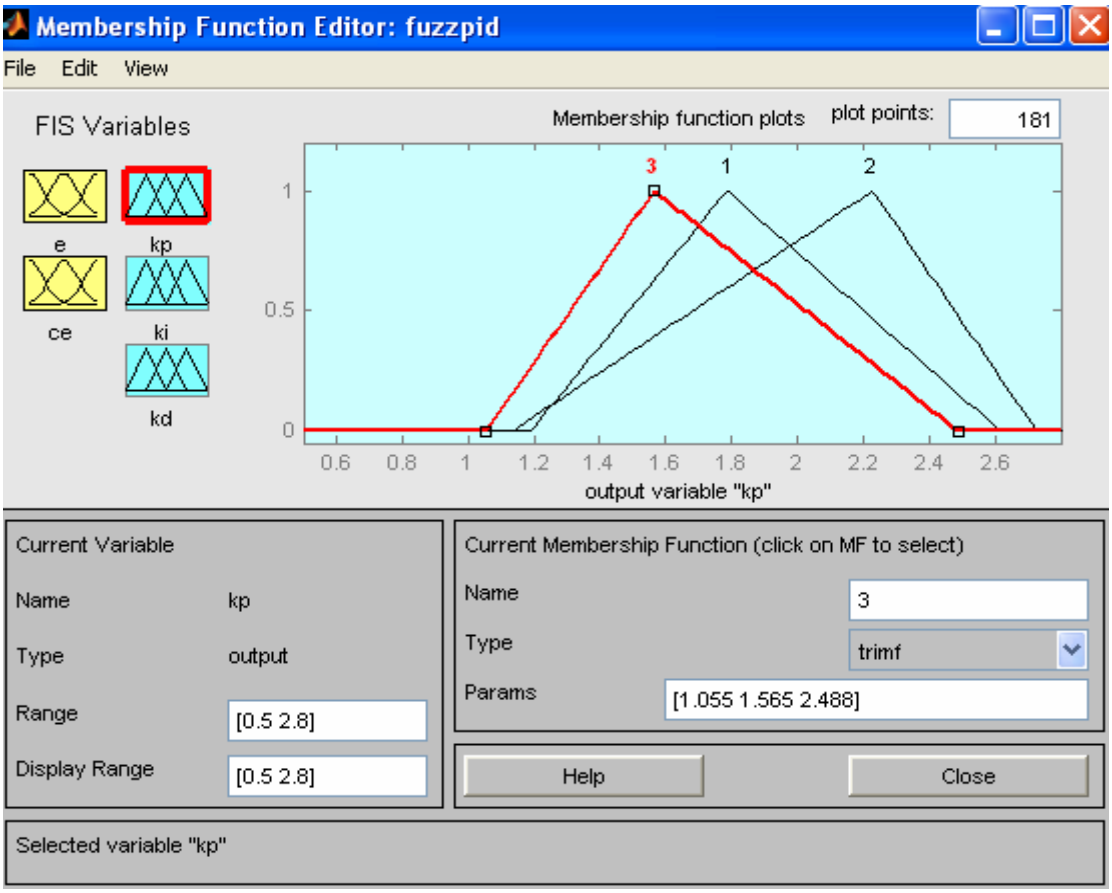
Diğer örneklerdeki gibi; oluşturulacak GFPID ve klasik PID ile eşit şartlarda karşılaştırmak adına klasik PID için birkaç deneme yapılıp uygun katsayılar bulunmuş ardından sisteme uygulama yapılmıştır (K_p = 2 K_i =0.015 , K_d = 0.002). Optimizasyon öncesinde ve sonrasında; okuma ve yazma işlemlerinde Matlab Fuzzy Toolbox kullanılmıştır. Genetik optimizasyon işlemi sırasında kullanılan parametreler Çizelge 4.7 ve Çizelge 4.8'de gösterilmiştir. GA Optimizasyon sonrasında oluşan BMK g/ç en iyi bireye ait üyelik fonksiyonları Giriş1 (*e*) Şekil 4.26, Giriş2 (*ce*) Şekil 4.27, Çıkış1 (K_p) Şekil 4.28, Çıkış2 (K_i) Şekil 4.29 ve Çıkış3 (K_d) Şekil 4.30'da gösterilmiştir.



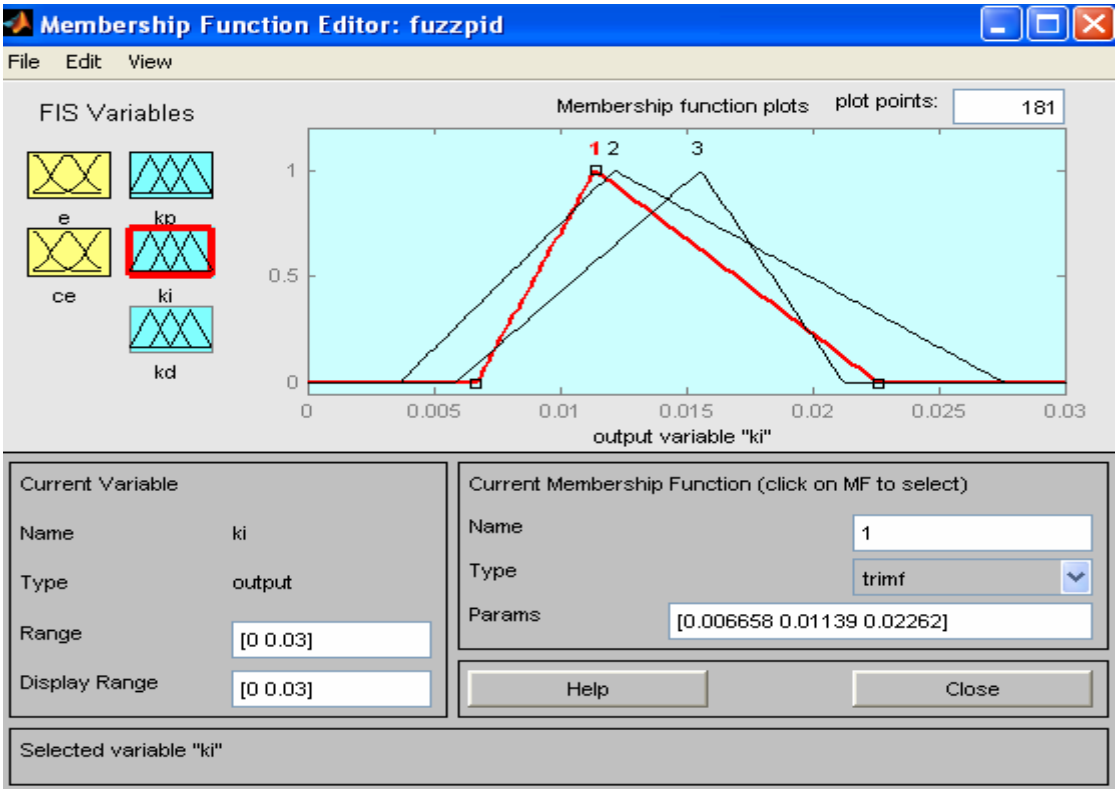
Şekil 4.26 Sistem3 – optimizasyon sonrası elde edilen giriş fonksiyonu (*e*)



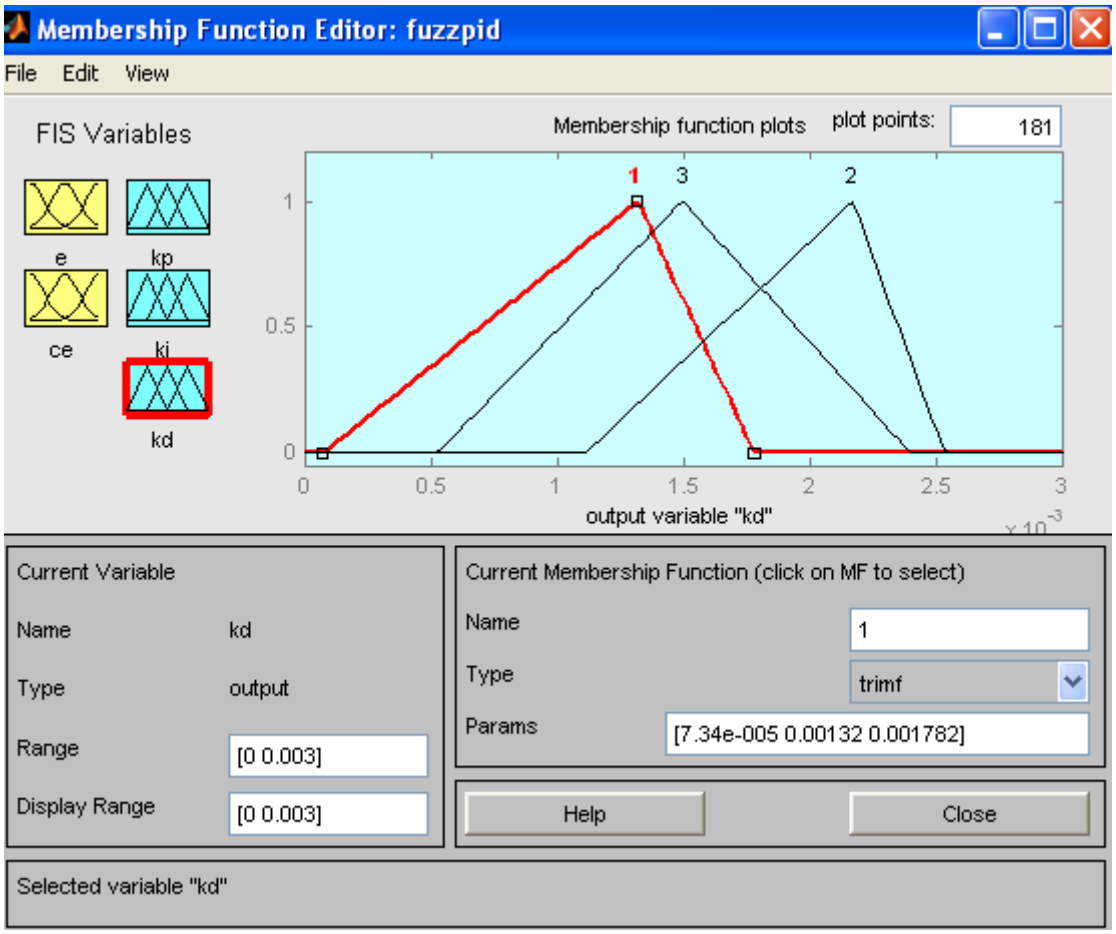
Şekil 4.27 Sistem3 – optimizasyon sonrası elde edilen giriş fonksiyonu (*ce*)



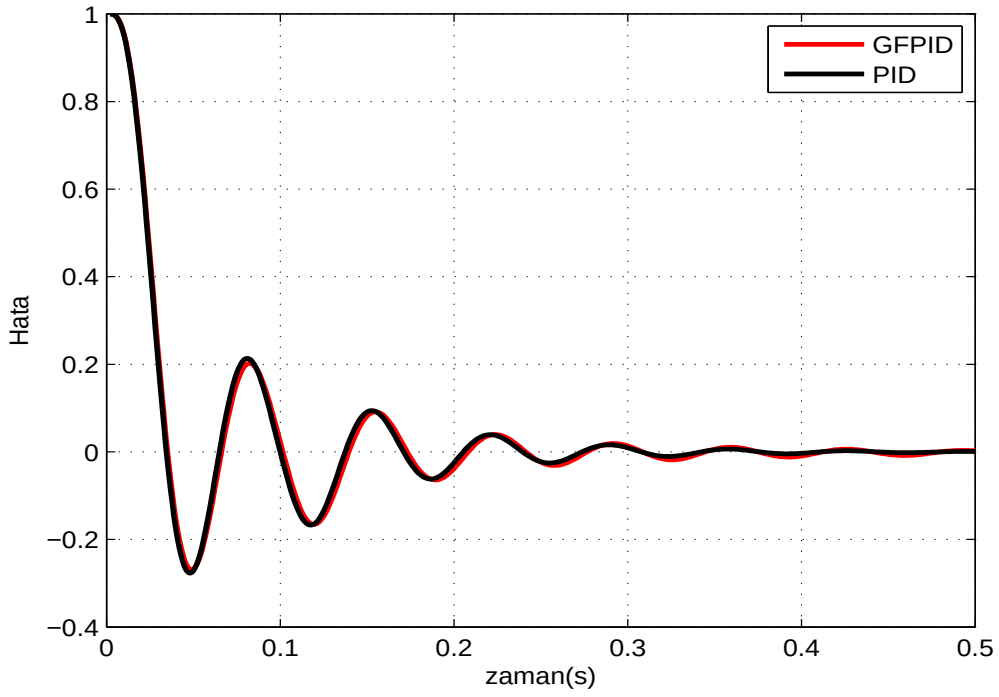
Şekil 4.28 Sistem3 – optimizasyon sonrası elde edilen çıkış fonksiyonu (*Kp*)



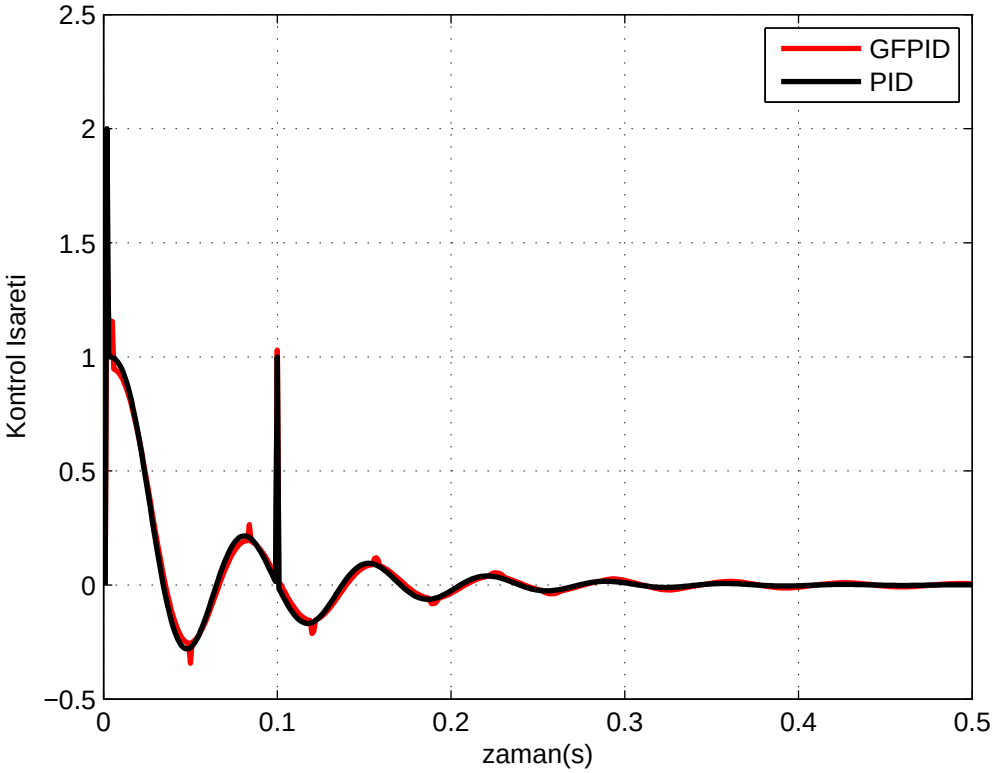
Şekil 4.29 Sistem3 – optimizasyon sonrası elde edilen çıkış fonksiyonu (*Ki*)



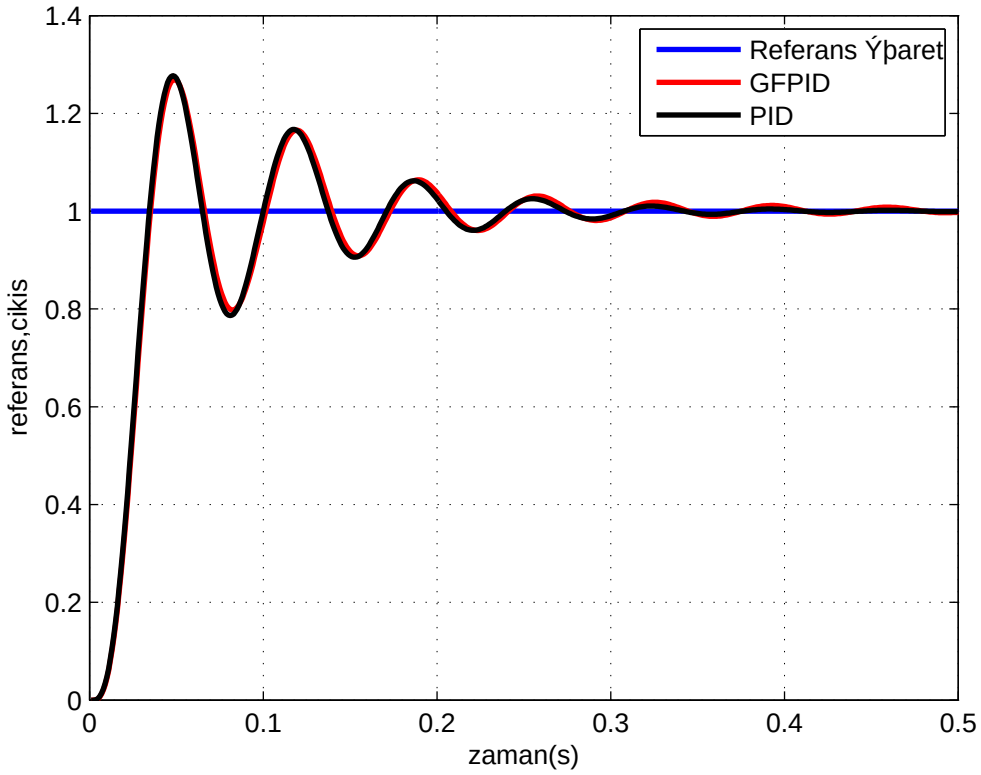
Şekil 4.30 Sistem3 – optimizasyon sonrası elde edilen çıkış fonksiyonu (*Kd*)



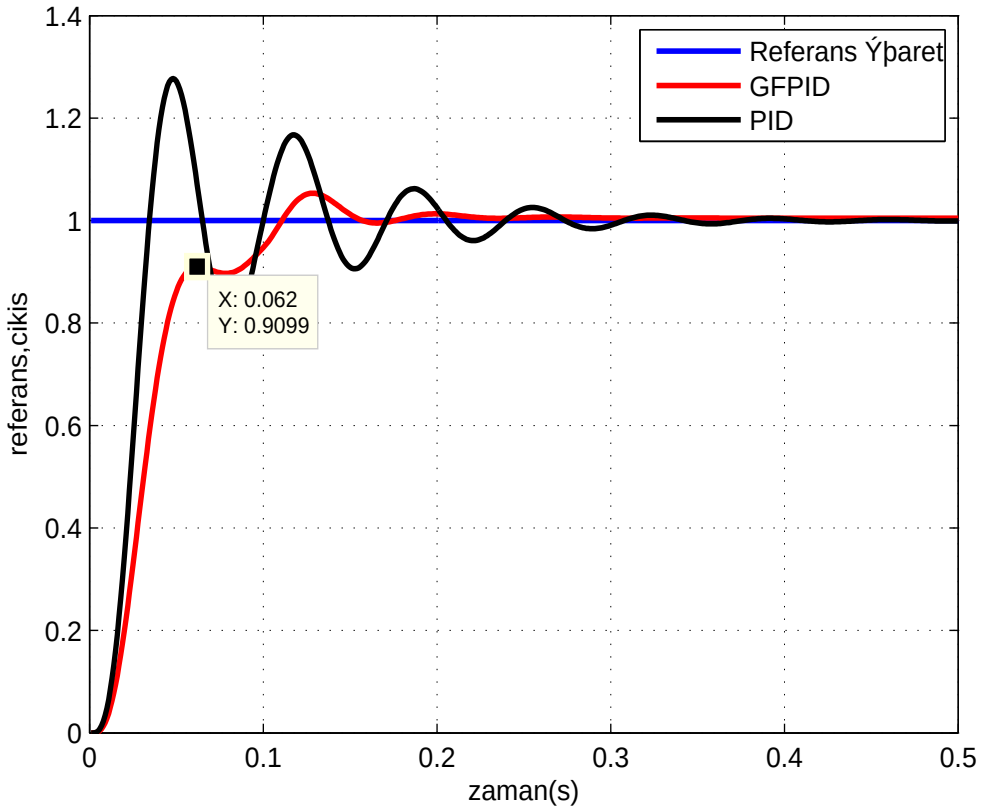
Şekil 4.31 Sistem3 – hatanın değişimi



Şekil 4.32 Sistem3 – kontrol işaretinin değişimi



Şekil 4.33 Sistem3 – sistem cevabı karşılaştırma (PID, FPID)



Şekil 4.34 Sistem3 – farklı parametreler için sistem cevabı karşılaştırma (PID, FPID)

Çizelge 4.9 Sistem3 - bmk g/ç fonksiyonları için verilen farklı sınır değerleri

Parametreler	**Sınırlar**
Giriş1 (e)	[-2,+2]
Giriş2 (ce)	[-4,+4]
Çıkış1 (K_p)	[+1.2,+3.2]
Çıkış2 (K_i)	[+0.1,+0.5]
Çıkış3 (K_d)	[+0.001,+0.005]

Tüm benzetimlerde olduğu gibi bu benzetimde kullanılan GA parametreleri(popülasyon büyüklüğü, maksimum jenerasyon sayısı, çaprazlama yöntemi ve oranı, mutasyon oranı, uygunluk fonksiyonu, sonlandırma koşulu) benzerlik göstermektedir. Farklı sınır değerleri için optimizasyon yapılmış ve başarılı sonuçlar elde edilmiştir. Sistem3 üzerinde diğer örnekte olduğu gibi iki farklı benzetim sonucunda karşılaştırma yapılmıştır. Benzetimde kontrol işaretine 0.1s referans işareti kadar gürültü eklenmiştir. İlk benzetim sonucunda basamak referansa sistemin cevabı Şekil 4.33'te gösterilmiştir. Bu benzetimde PID ve GFPID

sistem cevabı örtüşmektedir ve 0.3s kararlılığa kadar osilasyondadır. Kalıcı hal hataları PID olması gerektiği gibi “0” ve yerleşme zamanları eşittir.

İkinci benzetimde; Çizelge 4.9 üzerinde seçilen BMK ait sınır değerleri ile GA optimizasyonu yapılarak oluşturulan kontrolörün sisteme uygulandığında basamak referansa sistemin cevabı Şekil 4.34’te gösterilmiştir. GFPID kullanımında üst aşım neredeyse sıfırlanmıştır fakat sistem cevabı süresi azalmıştır. GFPID Yerleşme süresi klasik PID kontrolöre oranla daha başarılıdır.

SONUÇ VE ÖNERİLER

Tezdeki çalışmada kontrol teorisinin önemli konusu olan ve endüstride genel olarak kullanılan klasik PID Kontrolörler ve yeni yeni kullanılmaya başlayan bulanık kontrolörler ele alınmıştır. İkinci mertebeden transfer fonksiyonlarına uygulama yapılarak karşılaştırma yapılmış ve sistemlerin verdiği tepkiler incelenmiştir. Klasik PID tasarımında sıkıntılı süreç olarak bilinen kazanç katsayılarının ayarlanması BMK ile gerçeklenmiştir. Bulanık kontrolörleri tasarlamak için en önemli proses olan sistem uzmanı bilgisi ve işlem yapılacak sistem hakkındaki yüksek uzmanlık bilgisi GA kullanılarak en aza indirilmiştir. Bulanık kontrolör tasarım parametrelerinin optimizasyonu GA ile yapılmıştır. GA ile elde edilen kontrolör parametreleri FIS bloğuna işlenmiş ve sonrasında sisteme uygulaması yapılmıştır Yapılan çalışmada geliştirilmiş olan MATLAB kodu farklı sistemler üzerinde klasik PID ile karşılaştırması incelenmiştir.

Genel olarak FPID kontrolörlerin, klasik PID kontrolörlerden daha iyi çalıştığı birçok araştırmacı tarafından tecrübe edilmiştir. Fakat şu bir gerçektir ki; bu iki tip kontrolör (FPID ve Klasik PID) gerçekte birebir karşılaştırılamaz: çünkü her iki kontrolörün adilce karşılaştırılabileceği koşulları oluşturmak neredeyse mümkün değildir. Yapılan sadece PID kontrolörü sisteme uygulamak; iyi çalışıp çalışmadığını gözlemlemek ve daha iyi bir sonuç almak istendiğinde FPID geliştirmektir. Hatta bu tezde yapıldığı gibi bunu daha ileri götürüp BMK üyelik fonksiyonlarını GA ile ayarlamak ve bulanık kontrolör kurallarını optimize etmektir. Tez kapsamında dikkat edilen koşullar : her iki kontrolörün yükselme zamanlarının eşit olması, klasik PID için uygun K_p, K_i, K_d değerlerinin ayarlanması ve ayarlanan optimal değerlerin bulanık kontrolör için sınır olarak verilmesidir. Çünkü GA ile optimizasyon sırasında BMK için verilecek geniş sınırlar GFPID'nin klasik PID göre çok iyi sistem cevabına sahip olması anlamına gelecektir.

Benzetimler sonucunda görülmüştür ki GA bulanık kontrolörlerin veri tabanına ait parametreleri bulmak için güçlü bir optimizasyon aracıdır. GA kullanımının tek ve en büyük dezavantajı eş-zamanlı (online) çalışabilmesi için bulunması istenen parametre oranında yavaş olmasıdır. Jenerasyon sayısı, popülasyon büyüklüğü parametrelerin bulunmasını çok yoğun şekilde etkilemekle birlikte; mutasyon oranı, çaprazlama oranı ve string uzunluğu tasarımı GA için önemli seçim parametreleri olmuştur. Çünkü hem algoritmanın çalışma süresini hemde sistemin kararlılığı GA parametrelerine bağlıdır.

Oluşturulan GFPID çeşitli transfer fonksiyonlarına uygulanarak benzetimler yapılmıştır. Birebir şartlarda klasik PID ile aynı sonucu vermektedir. Yani literatürde anlatıldığı şekli ile

“PID like - PID gibi” kontrolör olarak çalışmaktadır. Ayrıca klasik PID göre daha az kontrol işareti üretilmektedir.

GA tabanlı melez bulanık PID kontrolörler kullanılarak klasik PID ve FPID birlikte kullanılabilir. FPID ve klasik PID kontrolör entegrasyonu yapılarak melez yapı oluşturulabilir. PID veya FPID beraber kullanılarak ayrı ayrı kullanımından daha iyi sonuçlar elde edilebilir. Klasik PID belirlenemeyen ve önceden kestirilmemiş gürültüler karşısında kısmen başarılı sonuç vermektedir. Melez kontrolör kullanarak; halihazırda birçok durum için yeterli olan klasik PID ile arkaplanda bekleyecek ve belirlenmiş veya belirlenmemiş gürültülerde devreye girecek olan FPID kullanımı başarılı sonuçlar verecektir. Yapılan simülasyonlarda çıkışa gürültü eklendiğinde FPID kural optimizasyonu sonrasında “D” kontrolcünün kısmen devre dışı bırakıldığı görülmüştür. Gürültü olan ortamlarda bu işlem klasik PID kullanımı sırasında manuel olarak gerçekleştirilmektedir.

GA vasıtası ile bulanık kontrolörlerin üyelik fonksiyonları ve kural tabloları offline (çevrim dışı) olarak hazırlanmış ve sisteme online olarak uygulaması yapılmıştır. Bulanık mantık tipi olarak Mamdani kullanılmıştır; bulanık kontrolör ağırlıkları ve bağlantılar (ve/ veya) elde edilmemiştir.

Çalışmanın ilerleyen aşamalarında BMK ait parametrelerin optimizasyonunun eş zamanlı yapılması sağlanabilir; üyelik fonksiyonu sınırları ve kural optimizasyonu online olarak gerçekleştirilebilir. İstenen tüm parametreler için en iyi sonuçlar bulunabilir; BMK üyelik fonksiyonu seçimi, bulanıklaştırma ve durulaştırma yöntemi seçimi optimizasyona katılırsa çok üst seviyede bir kontrolör tasarlanabilir.

Bu tez kapsamında kullanılan GA algoritması tek amaç fonksiyonuna sahiptir. Bu fonksiyon; referans işaret ile sistem cevabındaki farkı minimize etmektedir. MOGA yani çok amaç fonksiyonuna sahip GA kullanılarak; sistem cevabına ait maksimum üst aşımı minimize etmek, sistem cevabının yerleşme zamanını minimize etmek ve sistem cevabının yükselme zamanını minimize etmek mümkün olabilir.

Geliştirilmiş olan “Genetik Algoritma Tabanlı Optimal Adaptif Fuzzy PID Kontrolcü” her şeyden önce başarılı bir alternatif kontrolördür. Algoritma, sistem üzerinde kısa bir süre çevrimdışı (offline) çalışma yaptırılarak, optimum BMK parametrelerini bulmaktadır. Yapılan benzetimler sonucunda sistem cevabının örnek bir sistem için değil farklı sistemler için başarılı olduğu görülmüştür. Uygun sınırlar seçildiğinde; online PID katsayılarını ayarlayan bulanık kontrolör klasik PID göre gözle görülür şekilde iyi sonuçlar vermektedir.

KAYNAKLAR

Ali, H.M., Murata, T., Tamura, J., (2004), "The Effect of Temperature Rise of the Fuzzy Logic-Controlled Braking Resistors on Transient Stability", IEEE Transactions On Power Systems, 19(2).

Aras, Ö. (2009), "Bulanık Mantık Teknikleri ile Nötralizasyon Prosesinde On-Line Ph Kontrolü", Gebze İleri Teknoloji Enstitüsü.

Babaoğlan, İ., (2009), "MIMO Ofdm Sistemlerde Tepe Gücü/Ortalama Güç Oranını Düşürme Teknikleri Ve Bu Oranı Düşürmek İçin Bulanık Mantık Kullanımı", Erciyes Üniversitesi.

Babuška, R., (2001), "Bulanık and Neural Control Disc Course", Delft University of Technology, Delft, Netherlands.

Bagley, J.D., (1967), "The Behavior of Adaptive Systems Which Employ Genetic and Correlational Algorithms", Universtiy of Michigan.

Bakanay, D., (2009), "Mikro Öğretimde Performansın Bulanık Mantık Yöntemiyle Değerlendirilmesi", Marmara Üniversitesi.

Booker, L.B., Goldberg, D.E., Holland, J.H., (1989), "Classifier Systems and Genetic Algorithms", Artificial Intelligence, 40(1-3):235-282.

Chakraborty, U.K.R. Bandyopadhyay, D. Patranabis, (2001) , "A fuzzy-genetic approach for automatic tuning of a PID controller" , "23'3 Int. Conf. information Technology lnferfaces /TI 2007, Pula, Croatia".

Cordon O., Herrera F., Hoffman F. ve Magdalena L., (2001), Genetic Fuzzy Systems, World Scientific Pub Co Inc. 9810240163.

Çivril, H., (2009), "Hemşire Çizelgeleme Problemlerinin Genetik Algoritma İle Çözümü", Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü.

Dalcı, B., Uzunoğlu, M. ve Küçükdemiral, İ., (2004), "Genetic Algorithm Based Optimal Self-Tuning Fuzzy Logic Power System VAR Stabilizer", International Journal of Electrical Engineering Education, 41:71-89.

Darwin, C., (1859), On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life, John Murray, London, Modern reprint Charles Darwin, Julian Huxley, On The Origin of Species. Signet Classics. ISBN 0-451-52906-5.

Davis, L., (1991), Handbook of Genetic Algorithms, Van Nostrand Reinhold, NewYork.

Ding, X., Wang, Z., Zhang, C. ve Hu, Q., (2009), "A Fuzzy Neutral Network Controller Based on Optimized Genetic Algorithm for UC Rolling Mill", 2009 Second International Symposium on Knowledge Acquisition and Modeling.

Elmas, Ç., (2003), Bulanık Mantık Denetleyiciler, Seçkin Yayıncılık, Ankara, Türkiye.

Engin, O., (2001), "Akış Tipi Çizelgeleme Problemlerinin Genetik Algoritma ile Çözüm Performansının Artırılmasında Parametre Optimizasyonu", İTÜ Endüstri Mühendisliği.

Er, M. J., ve Sun, Y. L., (2001), "Hybrid Fuzzy Proportional–Integral Plus Conventional Derivative Control of Linear and Nonlinear Systems", IEEE Transactions On Industrial Electronics, 48(6).

Esen, M., (2006), "Bulanık Mantık Destekli Güç Akış Analizi", Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü.

Gen, M. ve Chen, R., (1997), Genetic Algorithms and Engineering Design, NewYork, Wiley.

Goldberg D.E., (1989), Genetic Algorithms in Search, Optimisation and Machine Learning, Addision-Wesley.

Goldberg ve Miller, (1995), "Genetic Algorithms, Tournament Selection, and the Effects of Noise, Complex Systems", University of Illinois, 9(3):193-212.

Holland, J.H. (1975), Adaptation in Natural and Artificial Systems, The University of Michigan Press, USA.

Hu, B., Mann, G. K. I., Gosine, R. G. (1999), "New Methodology for Analytical and Optimal Design of Fuzzy PID Controllers" , IEEE Transactions On Fuzzy Systems, 7(5).

Karaoğlan, Ö. G., (2007), "Kontrol Sistemleri İçin Bulanık PID Kontrolörlerin Genetik Algoritmalar Yardımıyla Ayarlanması", İstanbul Teknik Üniversitesi.

Kıyak, E. (2008), "Bir Uçağın Yatıs Kontrol Sistem Tasarımında Klasik ve Bulanık Denetleyici Etkileri" , Anadolu Üniversitesi, e-ISSN:1304-414.

Kobayashi, H., Hsu, C.C., Yamada, S.I., Fujikawa, H. (1997), "A Design of a Model Reference Fuzzy Adaptive Controller for Linear Systems with Time Delay Using MSGA", Industrial Electronics, Control and Ins., IECON97. 23rd International Conference.

Koza, J. R., (1992), Genetic Programming , Bradford Book, The MIT Press, Massachusetts.

Kuruca, E. (2009), "Gezgin Satıcı Problemi Tabanlı Bir Sistemin Dinamik Bulanık Genetik Algoritmalar İle Optimizasyonu", Yıldız Teknik Üniversitesi, İstanbul.

Kuvat, G. (2009), "Paralel Genetik Algoritmalarda Göç Yöntemleri ve Göç Parametrelerinin Dinamik Olarak Belirlenmesi " , Eskişehir Osmangazi Üniversitesi.

Küçükdemiral, İ. B., (2002), "Nöral-Genetik Tabanlı Optimal Bulanık Kontrolörün Gerçeklenmesi ve DC servomotora uygulanması", Yıldız Teknik Üniversitesi.

Küçükdemiral, İ. B., Cansever, G., (2006), "Sugeno Based Robust Adaptive Fuzzy Sliding-Mode Controller for SISO Nonlinear Systems", Journal of Intelligent and Fuzzy Systems, 17:113–124.

Lo, K. L. ve Sadegh, M.O., (2003), "Systematic method for the design of a full-scale fuzzy PID stability controller for svc to control power system", IEE Proc.-Gmm Tranrm. Dimib., 150(3).

Mamdani, E. H. , Assilian, S., (1975), "An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller", International Journal of Man-Machine Studies.

Mathworks, (2010), Matlab R2009b, Genetic Algorithm Toolbox; Matlab User's Guide.

Michalewicz, Z., (1992), Genetic Algorithms+Data Structures=Evolution P., Springer ,USA.

Mitchell, M., (1996), "Introduction to Genetic Algorithms" , "MIT press, Ann A., MI, USA".

Moghaddas, J., (2001), "A Fuzzy Neural Network Approach For Power System Evaluations" "New Mexico State University, New Mexico"

Qin, W., Du, S. (1997), "A Fuzzy PID Controller Optımized By Genetic Algorithms Used For A Single Phase Power Factor Pre-Regulator", Industrial Electronics, Control and Instrumentation, 1997. IECON 97. 23rd International Conference.

Reeves, C.R., (1993), Modern Heuristic Techniques for Combinatorial Problems, John Wiley & Sons, Inc., New York, NY.

Ross, T.J., (1995), "Fuzzy Logic with Engineering Applications", "Mc.Graw-Hill Publishing Co. New York".

Rubaai, A., Marcel J. Castro-Sitiriche, Abdul R. Ofoli, (2008) , "DSP-Based Laboratory Implementation of Hybrid Fuzzy-PID Controller Using Genetic Optimization for High-Performance Motor Drives" , IEEE Transactions On Industry Applications, Vol. 44, No. 6.

Seng, T.L., Bin Khalid, M., Rubiyah, Y., (1999), "Tuning of a Neuro-Fuzzy Controller by Genetic Algorithm", IEEE Transactions On Systems, Man, And Cybernetics PartB: Cybernetics, 29(2).

Smith, D. S., (2004), Digital Transmission Systems, Springer.

Sugeno, M., (1985), Industrial Applications of Fuzzy Control, Elsevier Science Pub. Co.

Şen, Z., (2000), Mühendislikte Bulanık Mantık ile Modelleme Prensipleri, Su Vakfı.

Tang, K.S., Man, K. F., (2001), "An Optimal Fuzzy PID Controller", IEEE Transactions On Industrial Electronics, 48(4).

Topuz, V., (2002), "Bulanık genetik proses kontrolü", Marmara Üniversitesi, İstanbul.

Valarmathi, K., Devaraj, D., Radhakrishnan, T.K., (2008), "A Combined Genetic Algorithm and Sugeno Fuzzy Logic based approach for on-line Tuning in pH process", Intelligent Systems, IS '08. 4th International IEEE Conference.

Visioli, A. (1999), "Fuzzy Logic Based Set-Point Weight Tuning of PID Controllers", IEEE Transactions On Systems, Man And Cyber. Part A: Systems And Humans, 29(6).

Visioli, A. (2001), "Tuning of PID controllers with fuzzy logic" , IEE Proc.-Control Theory Appl., 148(1).

Wang, L. X. (1997), A Course In Fuzzy Systems and Control, Prentice Hall PTR.

Yeniay, Ö. , (2001), "An Overview of Genetic Algorithms", Anadolu Üniversitesi BT Dergisi, 2(1).

Yeo, M.F., Agyel, (1996), "Optimizing Engineering Problems Using Genetic Algorithms",

Engineering Computations, 15(2):267-281.

Ying, H. (2006), “Deriving Analytical Input–Output Relationship for Fuzzy Controllers Using Arbitrary Input Fuzzy Sets and Zadeh Fuzzy AND Operator”, IEEE Transactions On Fuzzy Systems, 14(5).

Zadeh, L., (1965) , “Fuzzy Sets”, Information and Control, 8:338–3

İnternet Kaynakları

[1] www.ieee.org

EKLER

Ek 1 Matlab Kaynak Kodları

```
%Ana program.m

% Genetik Algoritma Tabanlı Optimal Bulanık PID Kontrolcü Tasarımı

% Genel program , alt programlar buradan çağrılacak

clc; close all; clear all;

global dongu_sayisi;

dongu_sayisi=2000;

global genetik_dongu_sayisi;

genetik_dongu_sayisi=400;

global populasyon_sayisi ;

populasyon_sayisi = 10;

global jenerasyon_sayisi ;

jenerasyon_sayisi = 10;

global sys ; %transfer fonksiyonu

global N nbit limit; %değişken sayısı ve hassasiyet

N=45; %Değişken sayısı

sys=tf(400,[1,50,0]); %Denklem1

%sys=tf(8152,[1,461.2,0]); %Denklem2

%ucgen fonksiyon icin range degerleri

global e_min  e_max ce_min ce_max kp_min kp_max ki_min ki_max kd_min kd_max  ;

e_min = -30 ; e_max = 30; ce_min = -30; ce_max = 30 ; kp_min = -5; kp_max = 5 ; ki_min =
-0.1 ; ki_max = 0.1; kd_min = -0.05 ; kd_max = 0.05 ;

%limit = [-0.5 -0.25; -0.25 0.25; 0.25 0.5;-0.5 -0.25; -0.25 0.25; 0.25 0.5;-0.5 -0.25; -0.25
0.25; 0.25 0.5;-0.5 -0.25; -0.25 0.25; 0.25 0.5;-0.5 -0.25; -0.25 0.25; 0.25 0.5;-0.5 -0.25; -0.25
```

```
0.25; 0.25 0.5; -0.5 -0.25; -0.25 0.25; 0.25 0.5;-0.5 -0.25; -0.25 0.25; 0.25 0.5;-0.5 -0.25; -0.25
0.25; 0.25 0.5;-0.5 -0.25; -0.25 0.25; 0.25 0.5;-0.5 -0.25; -0.25 0.25; 0.25 0.5;-0.5 -0.25; -0.25
0.25; 0.25 0.5;-0.5 -0.25; -0.25 0.25; 0.25 0.5;-0.5 -0.25; -0.25 0.25; 0.25 0.5;-0.5 -0.25; -0.25
0.25; 0.25 0.5];

e_limit = membership(e_min,e_max) ;

ce_limit = membership(ce_min,ce_max) ;

kp_limit = membership(kp_min,kp_max) ;

ki_limit = membership(ki_min,ki_max) ;

kd_limit = membership(kd_min,kd_max) ;

limit=[e_limit ; ce_limit ;kp_limit ;ki_limit ; kd_limit] ;

fuzzy_initialize(); %Fuzzy blogu initial degerler icin olusturuluyor

[ex_time,ex_referans,ex_yout,ex_hata] = calistir(); %eski Fuzzy sistem üzerinde calistiriliyor

bit_hassasiyeti=5; % değişkenler için kullanılacak ikili hassasiyet

nbit = bit_hassasiyeti * ones(1,N) ; % her değişken için ayrı bit kullanılacak

%nbit=[5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5];

%genetik algoritma toolbox gönderme kısmı - fuzzy membership için

FitnessFunction = @fitness_function;

string_uzunlugu = N * nbit(1); % dizi = degisken sayısı * tek değişken için bit sayisi

options = gaoptimset('populationsize',populasyon_sayisi,'Populationtype','bitstring',
'CrossoverFcn',{@crossoversinglepoint},'MutationFcn', {@mutationuniform, 0.012},...

'Generations',jenerasyon_sayisi,'StallGenLimit',inf,'StallTimeLimit',inf,'PlotFcns',{@gaplotbe
stf,@gaplotstopping});

%'CrossoverFcn',{@crossoversinglepoint},'MutationFcn', {@mutationuniform,0.012},...

[x,FVAL,REASON,OUTPUT,POPULATION,SCORES] =
ga(FitnessFunction,string_uzunlugu,options);

sonuc = string_parcala(N,x,nbit,limit); %String parcalama islemi degiskenler
```

```
fuzzy_yazma(sonuc); %Olusan parametre degerleri fuzzy membershiplere yaziliyor
kural();
pid_cikis = pid_calistirma() ;
[time,referans,yout,hata] = calistir(); %yeni Fuzzy sistem üzerinde calistiriliyor
close;
figure(1);plot(time,referans,'b',time,yout,'r' , time,pid_cikis,'k','LineWidth',2);
hleg1 = legend('Referans İşaret','Genetik Bulanık Kontrolcü','PID Çıkış');
%figure(1);plot(time,referans,'b',time,yout,'LineWidth',2);
xlabel('time(s)');ylabel('referans,cikis');
grid on
% Program sonu anaprogram.m
%baslangic calistir.m
function [time,referans,cikis,hata]=calistir()
global a;
global dongu_sayisi;
global sys;
global e_min  e_max ce_min ce_max  ;
%Transfer fonksiyonu
ts=0.001;
dsys=c2d(sys,ts,'z');
[num,den]=tfdata(dsys,'v');
%Referans sinyal
yref=ones(1,dongu_sayisi)*1;
```

```
referans=yref;
%İlk andaki değerler kodun index hatası vermemesi için
u_1=0.0;u_2=0.0;y_1=0;y_2=0;
x_=[0;0;0];
error_1=0; hata=0; hata_turev=0;
kp0=0.0; kd0=0.0; ki0=0.0;
for k=1:dongu_sayisi
time(k)= ts* k;
    fuzzypid=evalfis([hata hata_turev],a); %Fuzzy evaluate işlemi
       kp(k)=(kp0+fuzzypid(1)) ;
       ki(k)=(ki0+fuzzypid(2)) ;
       kd(k)=(kd0+fuzzypid(3)) ;
       u(k)=kp(k)*x_(1)+kd(k)*x_(2)+ki(k)*x_(3);  % Kontrol sinyali
yout(k)=-den(2)*y_1-den(3)*y_2+num(1)*u(k)+num(2)*u_1+num(3)*u_2;
%sys=tf(400,[1,50,0]);
% Transfer function:
%   400
% ----------
% s^2 + 50 s
error(k)=yref(k)-yout(k);
%Sonraki adım için atamalar
  u_2=u_1;   u_1=u(k);
  y_2=y_1;   y_1=yout(k);
  x_(1)=error(k);              %  P
```

```
  x_(2)=error(k)-error_1;        %  D
  x_(3)=x_(3)+error(k);          %  I
  error_1=error(k);
  hata=x_(1); hata_turev=x_(2)
%Fuzzy PID sonu
end
cikis = yout ;
hata = u;
%Programın sonu
%Program sonu calistir.m
%fitness_function.m
%uygunluk fonksiyonu
function [J]=fitness_function(x)
global genetik_dongu_sayisi;
global N nbit limit
sonuc = string_parcala(N,x,nbit,limit); %String parcalama islemi degiskenler
fuzzy_yazma(sonuc); %Olusan parametre degerleri fuzzy membershiplere yaziliyor
[~,~,~,error] = sistem;
J=0;
for i=1:genetik_dongu_sayisi
   J=J+abs(error(i));
end
% program sonu fitness_function.m
%fitness_function_rule.m
```

```
%fitness function - kural optimizasyonu için

function [J]=fitness_function_rule(x)

global genetik_dongu_sayisi;

global e_min  e_max ce_min ce_max  ;

global a ;

global sys;

a=readfis('fuzzpid');

N=27; %Değişken sayısı

nbit=[2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]; %2 2 2 2 2 2

% Genetik algoritma sonucu stringi parçalama işlemi Binary->Decimal

b1=0; bn=0;

for j=1:N

   sum(j)=0;

   n=nbit(j);

   if (j==1)

      b1=1;

   else

      b1=b1+nbit(j-1);

   end

   bn=bn+nbit(j);

   for k=b1:bn

      n=n-1;

      sum(j)=sum(j)+x(k)*(2^n);

   end
```

```
end
for z=1:9
   a.rule(z).consequent(1) =  sum(z*3-2);
   a.rule(z).consequent(2) =  sum(z*3-1);
   a.rule(z).consequent(3) =  sum(z*3);
end
%Transfer fonksiyonu
ts=0.001;
dsys=c2d(sys,ts,'z');
[num,den]=tfdata(dsys,'v');
%Referans sinyal
yref=ones(1,genetik_dongu_sayisi)*1;
u_1=0.0;u_2=0.0;
y_1=0;y_2=0;
x_=[0;0;0];  %İlk matris üzerinden gitmemek için dummy matris
error_1=0; hata=0; hata_turev=0;
kp0=0.5; kd0=1.0; ki0=0.0; %PID initial değerlerin verilmesi
for k=1:genetik_dongu_sayisi
      fuzzypid=evalfis([hata hata_turev],a);
      kp(k)=(kp0+fuzzypid(1)) ;
      ki(k)=(ki0+fuzzypid(2)) ;
      kd(k)=(kd0+fuzzypid(3)) ;
      u(k)=kp(k)*x_(1)+kd(k)*x_(2)+ki(k)*x_(3);
yout(k)=-den(2)*y_1-den(3)*y_2+num(1)*u(k)+num(2)*u_1+num(3)*u_2;
```

```
error(k)=yref(k)-yout(k);
%Sonraki adım için atamalar
  u_2=u_1;  u_1=u(k);
  y_2=y_1;  y_1=yout(k);
  x_(1)=error(k);              %  P - Oransal
  x_(2)=error(k)-error_1;      %  D - Türev
  x_(3)=x_(3)+error(k);        %  I - Integral
  error_1=error(k);
  hata=x_(1); hata_turev=x_(2);
end
J=0;
for i=1:genetik_dongu_sayisi
   J=J+abs(error(i));
end
%program sonu fitness_function_rule.m
%fuzzy_initialize.m
%optimize edilecek fuzzy bloğu oluşturmak
function fuzzy_initialize()
global a ; %global degisken
global e_min  e_max ce_min ce_max kp_min kp_max ki_min ki_max kd_min kd_max  ;
a=newfis('fuzzpid','mamdani','min','max','min','max','centroid');
a=addvar(a,'input','e',[e_min,e_max]);                    %hata e
a=addmf(a,'input',1,'1','trimf',[e_min,e_min/2,0]);
a=addmf(a,'input',1,'2','trimf',[e_min/2,0,e_max/2]);
```

```
a=addmf(a,'input',1,'3','trimf',[0,e_max/2,e_max]);
a=addvar(a,'input','ce',[ce_min,ce_max]);                    %hata e
a=addmf(a,'input',2,'1','trimf',[ce_min,ce_min/2,0]);
a=addmf(a,'input',2,'2','trimf',[ce_min/2,0,ce_max/2]);
a=addmf(a,'input',2,'3','trimf',[0,ce_max/2,ce_max]);
a=addvar(a,'output','kp',[kp_min,kp_max]);                    %hata e
a=addmf(a,'output',1,'1','trimf',[kp_min,kp_min/2,0]);
a=addmf(a,'output',1,'2','trimf',[kp_min/2,0,kp_max/2]);
a=addmf(a,'output',1,'3','trimf',[0,kp_max/2,kp_max]);
a=addvar(a,'output','ki',[ki_min ,ki_max]);                %ki output2
a=addmf(a,'output',2,'1','trimf',[ki_min , ki_min/2, 0]);
a=addmf(a,'output',2,'2','trimf',[ki_min/2, 0 ,ki_max/2]);
a=addmf(a,'output',2,'3','trimf',[0, ki_max/2 , ki_max]);
a=addvar(a,'output','kd',[kd_min, kd_max]);                %ki output2
a=addmf(a,'output',3,'1','trimf',[kd_min, kd_min/2, 0]);
a=addmf(a,'output',3,'2','trimf',[kd_min/2 ,0 ,kd_max/2]);
a=addmf(a,'output',3,'3','trimf',[0, kd_max/2,kd_max]);
kurallar= [1 1 1 1 1 1 1 ;
1 2 1 2 2 1 1 ;
1 3 1 3 3 1 1 ;
2 1 2 1 1 1 1 ;
2 2 2 2 2 1 1 ;
2 3 2 3 3 1 1 ;
3 1 3 1 1 1 1 ;
```

```
3 2 3 2 2 1 1 ;

3 3 3 3 3 1 1];

a=addrule(a,kurallar);

a=setfis(a,'DefuzzMethod','mom');

writefis(a,'fuzzpid');

%program sonu fuzzy_initialize.m

%fuzzy_yazma.m

%Edinilen genetik algoritma sonuclarının fuzzy yazılması

function fuzzy_yazma(sonuc)

global a ; %global degisken

global e_min  e_max ce_min ce_max kp_min kp_max ki_min ki_max kd_min kd_max ;

% Scaling functions tanımlama kısmı - e input1

in1_mf1_a=sonuc(1) ; % giris1 uyelik fonksiyonu1 a

in1_mf1_b=sonuc(2) ; % giris1 uyelik fonksiyonu1 b

in1_mf1_c=sonuc(3) ; % giris1 uyelik fonksiyonu1 c

in1_mf2_a=sonuc(4) ; % giris1 uyelik fonksiyonu2 a

in1_mf2_b=sonuc(5) ; % giris1 uyelik fonksiyonu2 b

in1_mf2_c=sonuc(6) ; % giris1 uyelik fonksiyonu2 c

in1_mf3_a=sonuc(7) ; % giris1 uyelik fonksiyonu3 a

in1_mf3_b=sonuc(8) ; % giris1 uyelik fonksiyonu3 b

in1_mf3_c=sonuc(9) ; % giris1 uyelik fonksiyonu3 c

a.input(1,1).range = [e_min e_max];

a.input(1,1).mf(1,1).params = [ in1_mf1_a in1_mf1_b in1_mf1_c] ;

a.input(1,1).mf(1,2).params = [ in1_mf2_a in1_mf2_b in1_mf2_c] ;
```

```
a.input(1,1).mf(1,3).params = [ in1_mf3_a in1_mf3_b in1_mf3_c] ;
% Scaling functions tanımlama kısmı - ce giris2
in2_mf1_a=sonuc(10) ; % giris2 uyelik fonksiyonu1 a1
in2_mf1_b=sonuc(11) ; % giris2 uyelik fonksiyonu1 b1
in2_mf1_c=sonuc(12) ; % giris2 uyelik fonksiyonu1 c1
in2_mf2_a=sonuc(13) ; % giris2 uyelik fonksiyonu2 a2
in2_mf2_b=sonuc(14) ; % giris2 uyelik fonksiyonu2 b2
in2_mf2_c=sonuc(15) ; % giris2 uyelik fonksiyonu2 c2
in2_mf3_a=sonuc(16) ; % giris2 uyelik fonksiyonu3 a3
in2_mf3_b=sonuc(17) ; % giris2 uyelik fonksiyonu3 b3
in2_mf3_c=sonuc(18) ; % giris2 uyelik fonksiyonu3 c3
a.input(1,2).range = [ ce_min ce_max];
a.input(1,2).mf(1,1).params = [ in2_mf1_a in2_mf1_b in2_mf1_c] ;
a.input(1,2).mf(1,2).params = [ in2_mf2_a in2_mf2_b in2_mf2_c] ;
a.input(1,2).mf(1,3).params = [ in2_mf3_a in2_mf3_b in2_mf3_c] ;
% Scaling functions tanımlama kısmı - kp cikis1
out1_mf1_a=sonuc(19) ; % cikis1 uyelik fonksiyonu1 a1
out1_mf1_b=sonuc(20) ; % cikis1 uyelik fonksiyonu1 b1
out1_mf1_c=sonuc(21) ; % cikis1 uyelik fonksiyonu1 c1
out1_mf2_a=sonuc(22) ; % cikis1 uyelik fonksiyonu2 a2
out1_mf2_b=sonuc(23) ; % cikis1 uyelik fonksiyonu2 b2
out1_mf2_c=sonuc(24) ; % cikis1 uyelik fonksiyonu2 c2
out1_mf3_a=sonuc(25) ; % cikis1 uyelik fonksiyonu3 a3
out1_mf3_b=sonuc(26) ; % cikis1 uyelik fonksiyonu3 b3
```

```
out1_mf3_c=sonuc(27) ; % cikis1 uyelik fonksiyonu3 c3
a.output(1,1).range = [ kp_min kp_max];
a.output(1,1).mf(1,1).params = [ out1_mf1_a out1_mf1_b out1_mf1_c] ;
a.output(1,1).mf(1,2).params = [ out1_mf2_a out1_mf2_b out1_mf2_c] ;
a.output(1,1).mf(1,3).params = [ out1_mf3_a out1_mf3_b out1_mf3_c] ;
% Scaling functions tanımlama kısmı - ki cikis2
out2_mf1_a=sonuc(28) ; % cikis2 uyelik fonksiyonu1 a1
out2_mf1_b=sonuc(29) ; % cikis2 uyelik fonksiyonu1 b1
out2_mf1_c=sonuc(30) ; % cikis2 uyelik fonksiyonu1 c1
out2_mf2_a=sonuc(31) ; % cikis2 uyelik fonksiyonu2 a2
out2_mf2_b=sonuc(32) ; % cikis2 uyelik fonksiyonu2 b2
out2_mf2_c=sonuc(33) ; % cikis2 uyelik fonksiyonu2 c2
out2_mf3_a=sonuc(34) ; % cikis2 uyelik fonksiyonu3 a3
out2_mf3_b=sonuc(35) ; % cikis2 uyelik fonksiyonu3 b3
out2_mf3_c=sonuc(36) ; % cikis2 uyelik fonksiyonu3 c3
a.output(1,2).range = [ ki_min ki_max ];
a.output(1,2).mf(1,1).params = [ out2_mf1_a out2_mf1_b out2_mf1_c] ;
a.output(1,2).mf(1,2).params = [ out2_mf2_a out2_mf2_b out2_mf2_c] ;
a.output(1,2).mf(1,3).params = [ out2_mf3_a out2_mf3_b out2_mf3_c] ;
% Scaling functions tanımlama kısmı - kd cikis3
out3_mf1_a=sonuc(37) ; % cikis3 uyelik fonksiyonu1 a1
out3_mf1_b=sonuc(38) ; % cikis3 uyelik fonksiyonu1 b1
out3_mf1_c=sonuc(39) ; % cikis3 uyelik fonksiyonu1 c1
out3_mf2_a=sonuc(40) ; % cikis3 uyelik fonksiyonu2 a2
```

```
out3_mf2_b=sonuc(41) ; % cikis3 uyelik fonksiyonu2 b2

out3_mf2_c=sonuc(42) ; % cikis3 uyelik fonksiyonu2 c2

out3_mf3_a=sonuc(43) ; % cikis3 uyelik fonksiyonu3 a3

out3_mf3_b=sonuc(44) ; % cikis3 uyelik fonksiyonu3 b3

out3_mf3_c=sonuc(45) ; % cikis3 uyelik fonksiyonu3 c3

a.output(1,3).range = [ kd_min kd_max];

a.output(1,3).mf(1,1).params = [ out3_mf1_a out3_mf1_b out3_mf1_c] ;

a.output(1,3).mf(1,2).params = [ out3_mf2_a out3_mf2_b out3_mf2_c] ;

a.output(1,3).mf(1,3).params = [ out3_mf3_a out3_mf3_b out3_mf3_c] ;

writefis(a,'fuzzpid');

%program sonu fuzzy_yazma.m

%kural.m    kural optimizasyonu

function kural()

%genetik algoritma toolbox gönderme kısmı - fuzzy rulebase için

global a ;

%a=readfis('fuzzpid');

global y;

global populasyon_sayisi ;

global jenerasyon_sayisi ;

y= zeros(27,9);

N=27; %Değişken sayısı

nbit=[2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]; %2 2 2 2 2 2

FitnessFunction = @fitness_function_rule;

string_uzunlugu = 54; % bit sayisi
```

```
options = gaoptimset('populationsize',populasyon_sayisi,'Populationtype','bitstring',
'CrossoverFcn',{@crossoversinglepoint},...
'Generations',jenerasyon_sayisi,'StallGenLimit',inf,'StallTimeLimit',inf,'PlotFcns',{@gaplotbe
stf,@gaplotstopping});
[x,~,~,~,~,~] = ga(FitnessFunction,string_uzunlugu,options);
% Genetik algoritma sonucu stringi parçalama işlemi Binary->Decimal
b1=0; bn=0;
for j=1:N
   sum(j)=0;
   n=nbit(j);
   if (j==1)
      b1=1;
   else
      b1=b1+nbit(j-1);
   end
   bn=bn+nbit(j);
   for k=b1:bn
      n=n-1;
      sum(j)=sum(j)+x(k)*(2^n);
   end
end
for z=1:9
   a.rule(z).consequent(1) =  sum(z*3-2);
   a.rule(z).consequent(2) =  sum(z*3-1);
   a.rule(z).consequent(3) =  sum(z*3);
```

```
end
%program sonu kural.m
%membership.m Membershipleri ayirmak
function [deger]=membership(minimum,maximum)
%Tüm değerler sıfırlanıyor.
for i=1:9
   for j=1:2
      deger(i,j)=0;
   end
end
%baslangic kosulu
min = minimum ;
max = maximum ;
deger(1,1)= min;
deger(1,2)= min + (abs(min) + abs(max)) /3 ;
%limitler belirleniyor
for i=2:9
   deger(i,1) = deger(i-1,2) ;
   if(i==4 || i==7)
      deger(i,1) = min;
   end
   deger(i,2) = deger(i,1) + abs(min)/3+abs(max)/3 ;
end
%program sonu membership.m
```

```
%pid_calistirma.m
%Artımsal pid ile tek katsayı değeri ile sisteme pid uygulanması
function [pid_cikis]=pid_calistirma()
global sys;
global dongu_sayisi;
ts=0.001;
dsys=c2d(sys,ts,'z');
[num,den]=tfdata(dsys,'v');
u_1=0.0;u_2=0.0;
y_1=0;y_2=0;
x=[0,0,0]';
kp=5.0;
ki=0.5;
kd=0.5;
error_1=0;
for k=1:1:dongu_sayisi
ref(k)=1.0;
u(k)=kp*x(1)+kd*x(2)+ki*x(3);
yout(k)=-den(2)*y_1-den(3)*y_2+num(1)*u(k)+num(2)*u_1+num(3)*u_2; %den3 yok :D
error(k)=ref(k)-yout(k);
u_2=u_1;u_1=u(k);
y_2=y_1;y_1=yout(k);
x(1)=error(k);
x(2)=(error(k)-error_1)/ts;
```

```
x(3)=x(3)+error(k)*ts;

error_1=error(k);

end

pid_cikis = yout;

%program sonu pid_calistirma.m

%sistem.m Sonuçların sisteme uygulanması

function [time,referans,cikis,hata]=sistem()

global a;

global genetik_dongu_sayisi;

global sys;

global e_min  e_max ce_min ce_max  ;

%Transfer fonksiyonu

ts=0.001;

dsys=c2d(sys,ts,'z');

[num,den]=tfdata(dsys,'v');

yref=ones(1,genetik_dongu_sayisi)*1;

referans=yref;

%İlk andaki değerler kodun index hatası vermemesi için

u_1=0.0;u_2=0.0;y_1=0;y_2=0;

x_=[0;0;0];

error_1=0; hata=0; hata_turev=0;

kp0=0.0; kd0=0.0; ki0=0.0;

for k=1:genetik_dongu_sayisi

time(k)= ts* k;
```

```
    fuzzypid=evalfis([hata hata_turev],a); %Fuzzy evaluate işlemi
        kp(k)=(kp0+fuzzypid(1)) ;
        ki(k)=(ki0+fuzzypid(2)) ;
        kd(k)=(kd0+fuzzypid(3)) ;
        u(k)=kp(k)*x_(1)+kd(k)*x_(2)+ki(k)*x_(3);  % Kontrol sinyali
 yout(k)=-den(2)*y_1-den(3)*y_2+num(1)*u(k)+num(2)*u_1+num(3)*u_2;
%sys=tf(400,[1,50,0]);
error(k)=yref(k)-yout(k);
   u_2=u_1;   u_1=u(k);
   y_2=y_1;   y_1=yout(k);
   x_(1)=error(k);              %  P
   x_(2)=error(k)-error_1;      %  D
   x_(3)=x_(3)+error(k);        %  I
   error_1=error(k);
   hata=x_(1); hata_turev=x_(2);
end
cikis = yout ;
hata = u;
function [geri_don]=string_parcala(N,x,nbit,limit)
b1=0; bn=0;
for j=1:N
    sum(j)=0;
    n=nbit(j);
    if (j==1)
```

```
        b1=1;
    else
        b1=b1+nbit(j-1);
    end
    bn=bn+nbit(j);
    for k=b1:bn
        n=n-1;
        sum(j)=sum(j)+x(k)*(2^n);
    end
    var(j)=sum(j);
end
for j=1:N
    sonuc(j)=limit(j,1)+(limit(j,2)-limit(j,1))/(2^nbit(j)-1)*var(j);
end
for i=1:3:45
if sonuc(i)==sonuc(i+1) || sonuc(i)>sonuc(i+1)
sonuc(i)=sonuc(i+1)-0.1;
end
if sonuc(i+1)==sonuc(i+2) || sonuc(i+1)>sonuc(i+2)
sonuc(i+1)=sonuc(i+2)-0.1;
end
end
geri_don = sonuc;
%program sonu string_parcala.m
```

ÖZGEÇMİŞ

Doğum tarihi	14.06.1981	
Doğum yeri	İstanbul	
Lise	1996-1999	Haydarpaşa Teknik Lisesi Hidrolik Pnömatik Bölümü
Lisans	2000-2005	Girne Amerikan Üniversitesi Bilgisayar Mühendisliği Bölümü
Yüksek Lisans	2008-2011	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektrik Müh. ABD, Kontrol ve Otomasyon Programı

Çalıştığı Kurumlar

2006-2009	AKBİL A.Ş. – Proje Mühendisi
2009-2010	ARTI A.Ş. – Proje Mühendisi
2010-2011	ARTEL Ltd. Şti. – Proje Mühendisi