Karadeniz Teknik Üniversitesi
Mühendislik Fakültesi

Bilgisayar Mühendisliği Bölümü

Bulanık Mantık

DÖNEM PROJESİ

Feyzullah YILDIZ

15 Mayıs 2015

ÖNSÖZ

Bulanık mantık dersini alan bir öğrenci olarak çok faydalandığımı söyleyebilirim. Bir olayın bulanık olarak ifade edilmesi hakkında bilgi sahibi olmak ve verilen ödev ile bunu uygulamanın faydalı olduğunu düşünüyorum. Bu derste öğrendiğimiz bilgi ve becerileri akıllı sistemler, robotlar gibi alanlarda kullanabiliriz. Bu derste öğrendiklerimiz hem yazılım hem de elektronik devre olarak tasarlanabilmektedir. Biz bilgisayar mühendisi olarak yazılım kısmına ağırlık verdik. Bu dersi bilgisayar mühendisliği öğrencilerine anlatan ve öğreten Prof. Dr. İsmail H. ALTAŞ hocama teşekkür ediyorum.

Feyzullah YILDIZ
2015 - Trabzon

ÖZET

Bulanık mantık basit tanımı ile bir sistemin sonuç kümesinin "çok az, az, fazla, çok fazla" gibi ifadelerle bulanıklaştırmaktır. Mesela bir klima sıcak bir odayı soğutmak için odanın sıcaklığını ölçer ve buna göre estirilen sıcaklığın değerini az veya çok olarak ayarlar. Bu şekilde odayı sürekli aynı sıcaklıkta tutmaya çalışır. Bu örnek bulanık mantığın uygulama alanlarından bir tanesi olarak gösterilebilir.

Bu projede çeşitli üyelik fonksiyonları kullanarak matlab ortamında grafiksel sonuçlar elde ettik. Ayrıca derste gördüğümüz birçok konuyu bu projede uygulayarak öğrendiklerimizi pekiştirdiğimize inanıyorum. Son olarak bize verilen sistemlerden bana düşen 138. sistemi gerçekleştirdim.

Aşağıdaki olayların MATLAB fonksiyonlarını derste öğrendiklerimle ve çeşitli kaynaklardan araştırarak geliştirdim.

- Üçgen, yamuk, gaussian, çan, cauchy, sinüsoid, ve sigmoid üyelik fonksiyonları
- Kesişim, birleşim ve değilleme işlemleri
- Bulanık kümelerin tüm özelliklerini veren MATLAB fonksiyonu
- Açınım ilkesi
- Bulanık sonuçlandırma işlemleri
- Mamdani, sugeno, tsukamoto bulanık karar verme işlemleri
- Runge-Kutta Algoritması
- Oransal integral denetimi (PID)
- Bulanık denetim

Bu projede MATLAB R2014b sürümünü kullandım. MATLAB yeni sürümlerindeki bazı yenilikleri eski sürümler desteklemediğinden dolayı projedeki bazı fonksiyonlar eski sürümlerde çalışmayabilir.

Rapor içerisinde kodlara yer verilmemektedir. Kodlar ekteki dosyalarda mevcuttur. Kodların bazı kısımları raporda açıklanmıştır.

İÇİNDEKİLER

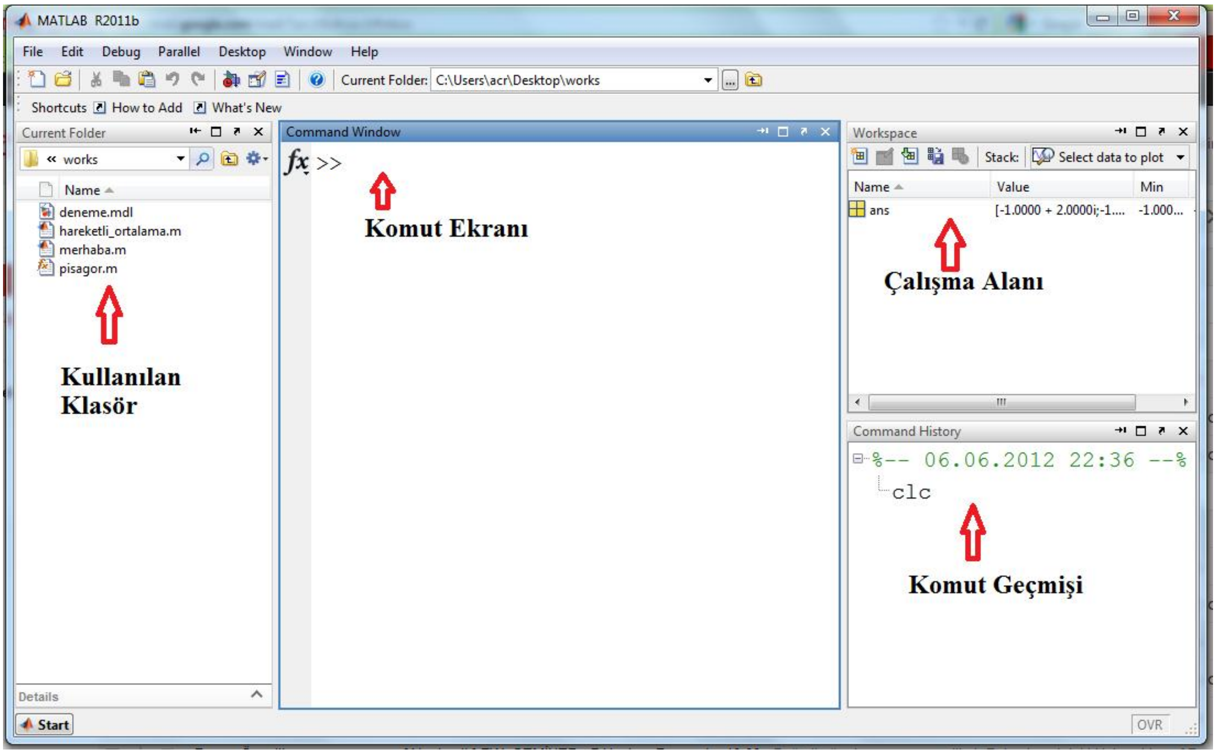
Önsöz i
Özet ii
İçindekiler iii
1. Bulanık Mantık ve Oluşturulan Kütüphane 4
1.1. Giriş 5
1.2. Fonksiyon Parametreleri 6
1.3. Üyelik Fonksiyonları 6
1.4. Bulanık Küme İşlemleri 6
1.5. Bulanık Küme Özellikleri 7
1.6. Açınım İlkesi 8
1.7. Bulanık Sonuçlandırma İşlemleri 9
1.8. Mamdani, Sugeno, Tsukamoto Bulanık Modelleme 10
1.9. Sonuçlar 11
1.10. Değerlendirmeler 12
2. Sistem Kontrolü: PID ve Bulanık Mantık Denetleyici Tasarımı 13
2.1. Giriş 14
2.2. Tasarım 15
2.3. Simülasyon Yazılımını Gerçekleme 16
2.4. Sonuçlar 17
2.5. Değerlendirmeler 18
3. Kaynaklar 19

1. BULANIK MANTIK VE OLUŞTURULAN KÜTÜPHANE

1.1. Giriş

Bulanık mantık, olayları üyelik dereceleri ile ifade etmektir. Mesela oda sıcak ise üyelik derecesi 1, sıcak değil ise 0'dır. Bulanık mantıkta önemli olan bu iki değer arasını ifade edebilmektir. Eğer oda biraz soğuk ise üyelik derecesi 0.25 olur. Aynı şekilde oda ılık ise soğuk kümesinin üyelik deresi 0.5 olur.

Bu proje için MATLAB kullanacağımdan öncelikle *mathworks* firmasına ait olan bu yazılımı elde ettim. Bilgisayar mühendisliği öğrencisi olarak daha önce matlab yazılımını kullanmadığım için biraz araştırma yapmam gerekti. Öğrenmek kolay oldu çünkü diğer programlama dillerine göre epey kolay bir yapısı vardı. Fakat yine de simulink kısmını pek anlayamadım.



Şekil 1 Klasik MATLAB görünümü

1.2. Fonksiyon Parametreleri

Fonksiyonları yazarken kullandığım parametrelerin anlamlarını aşağıda özetledim.

- `function [ x, mu ] = ucgen( x1, xT, x2, deger, geridonus )`

  ```
  x1: Üçgenin sol ayağı
  xT: Üçgenin tepe noktası
  x2: Üçgenin sağ ayağı
  deger: Üyelik derecesi bulunacak nokta
  geridonus: Geriye üçgen döndürmesini istiyorsam 1, "deger" noktasının üyelik
  derecesini istiyorsam 0 veriyorum.
  ```

- `function [ x, mu ] = yamuk(x1, xT1, xT2, x2 , kesin, geridonus )`

```
x1: Yamuğun sol ayağı
xT1: Yamuğun sol tepe noktası
xT2: Yamuğun sağ tepe noktası
x2: Yamuğun sağ ayağı
kesin: Üyelik derecesi bulunacak nokta
geridonus: Geriye üçgen döndürmesini istiyorsam 1, "deger" noktasının üyelik
derecesini istiyorsam 0 veriyorum.
```

- `function [ x, mu ] = sinusoid(xT, kesin, geridonus)`

```
xT: Fonksiyonun tepe noktası
kesin: Üyelik derecesi bulunacak nokta
geridonus: Geriye üçgen döndürmesini istiyorsam 1, "deger" noktasının üyelik
derecesini istiyorsam 0 veriyorum.
```

- `function [ x, mu ] = sigmoid( aL, cL,   cR,aR, kesin, geridonus )`

```
aL:  sol sigmoid tepe noktası
cL: sol sigmoid ayağı
cR: sağ sigmoid tepe noktası
aR:  sağ sigmoid ayağı
kesin: Üyelik derecesi bulunacak nokta
geridonus: Geriye üçgen döndürmesini istiyorsam 1, "deger" noktasının üyelik
derecesini istiyorsam 0 veriyorum.
```

- `function [ x, mu ] = gaussian( xT, w, kesin, geridonus )`

```
xT: Tepe noktası
w: Genişlik
kesin: Üyelik derecesi bulunacak nokta
geridonus: Geriye üçgen döndürmesini istiyorsam 1, "deger" noktasının üyelik
derecesini istiyorsam 0 veriyorum.
```

- `function [ x, mu ] = cauchy( xT, d, m, kesin, geridonus )`

```
xT: Yamuğun sol ayağı
d: Genişlik
m: Eğim
kesin: Üyelik derecesi bulunacak nokta
geridonus: Geriye üçgen döndürmesini istiyorsam 1, "deger" noktasının üyelik
derecesini istiyorsam 0 veriyorum.
```

- `function [ x, mu ] = canuyelik( xT, d, m, kesin, geridonus )`

```
xT: Tepe noktası
d: Genişlik
m: Eğim
kesin: Üyelik derecesi bulunacak nokta
geridonus: Geriye üçgen döndürmesini istiyorsam 1, "deger" noktasının üyelik
derecesini istiyorsam 0 veriyorum.
```

- `function [x1, y  ] = birlesim(  x1, y1, x2, y2 )`

```
x1: Birinci fonksiyonun x uzayı
y1: Birinci fonksiyonun üyelik dereceleri
x2: İkinci fonksiyonun x uzayı
y2: İkinci fonksiyonun üyelik dereceleri
```

- `function [ x1, y ] = kesisim( x1, y1, x2, y2 )`

```
x1: Birinci fonksiyonun x uzayı
y1: Birinci fonksiyonun üyelik dereceleri
x2: İkinci fonksiyonun x uzayı
y2: İkinci fonksiyonun üyelik dereceleri
```

- `function [ x1, y ] = degilleme( x1, y1)`

```
x1: Fonksiyonun x uzayı
y1: Fonksiyonun üyelik dereceleri
```

- `function [  ] = ozellikler( x1, y1, alfa )`

```
x1: Fonksiyonun x uzayı
y1: Fonksiyonun üyelik dereceleri
```

```
alfa: alfa kesmesi için kullanılan alfa değeri
```

- ```
  function [ yenix, yeniy  ] = acinim( x1, y1, a,b,c )
  ```

  ```
  x1: Fonksiyonun x uzayı
  y1: Fonksiyonun üyelik dereceleri
  a, b, c: Açınım yapılacak fonksiyonun katsayıları (ax^2 + bx + c)
  ```

- ```
  function[w, z]= bulanik_modelleme(Ax, Ay, Bx, By, CX, CY , k, xgiris, ygiris,
  sugeno_p, sugeno_q, sugeno_r)
  ```

  ```
  Ax, Ay: 1. kural fonksiyonları
  Bx, By: 2. kural fonksiyonları
  CX, CY: Çıkış fonksiyonları
  k: Kural Sayısı
  xgiris: 1. Kural için verilen giriş
  ygiris: 2. Kural için verilen giriş
  sugeno_p, sugeno_q, sugeno_r: Sugeno durulaştırma yapmak için gerekli katsayılar
  ```

- ```
  function [ x3,y3 ] = TGTK( x1, y1, x2, y2, x3, y3 )
  ```

  ```
  x1, y1: Giris fonksiyonu
  x2, y2: Kural fonksiyonu
  x3, y3: Çıkış fonksiyonu
  ```

- ```
  function[w, z]= CGTK(Ax, Ay, Bx, By, Cx, Cy, k)
  ```

  ```
  Ax, Ay: Giriş fonksiyonları
  Bx, By: Kural fonksiyonları
  Cx, Cy: Çıkış fonksiyonu
  k: Kural sayısı
  ```

- ```
  function[w, z]= CGCK(Ax, Ay, Bx, By, CX, CY , k)
  ```

  ```
  Ax, Ay: Giriş fonksiyonları
  Bx, By: Kural fonksiyonları
  Cx, Cy: Çıkış fonksiyonları
  k: Kural sayısı
  ```

1.3. Üyelik Fonksiyonları (uygulama_uyelikfonksiyonlari.m)

Derste takip ettiğimiz sunumlardan faydalanarak aşağıdaki üyelik fonksiyonlarını çizdirdim.

- Üçgen Üyelik Fonksiyonu (ucgen.m)
- Yamuk Üyelik Fonksiyonu (yamuk.m)
- Gaussian Üyelik Fonksiyonu (gaussian.m)
- Çan Üyelik Fonksiyonu (canuyelik.m)
- Sinüsoid Üyelik Fonksiyonu (sinusoid.m)
- Sigmoid Üyelik Fonksiyonu (sigmoid.m)

Sunumlardaki üyelik dereceleri fonksiyonlarına göre belli aralıktaki x değerleri için üyelik dereceleri bulunup fonksiyonlar elde edilmektedir. Bütün üyelik fonksiyonlarının geri dönüş tipleri [x mu] şeklindedir. Çünkü bu geriye dönen x ve üyelik dereceleri ileride kullanılacaktır. Sistemler hesaplanırken oransal integral ve bulanık denetim fonksiyonları kullanılacak ve bu fonksiyonlar üyelik fonksiyonlarındaki belli bir noktanın üyelik derecesini istediğinden dolayı deger ve geridonus parametresi alınıyor. Geridonus parametresi 0 olduğunda girilen değerin üyelik derecesi hesaplanıp hem x hem mu ile geri döndürülmektedir. Bu şekilde üyelik
  ```

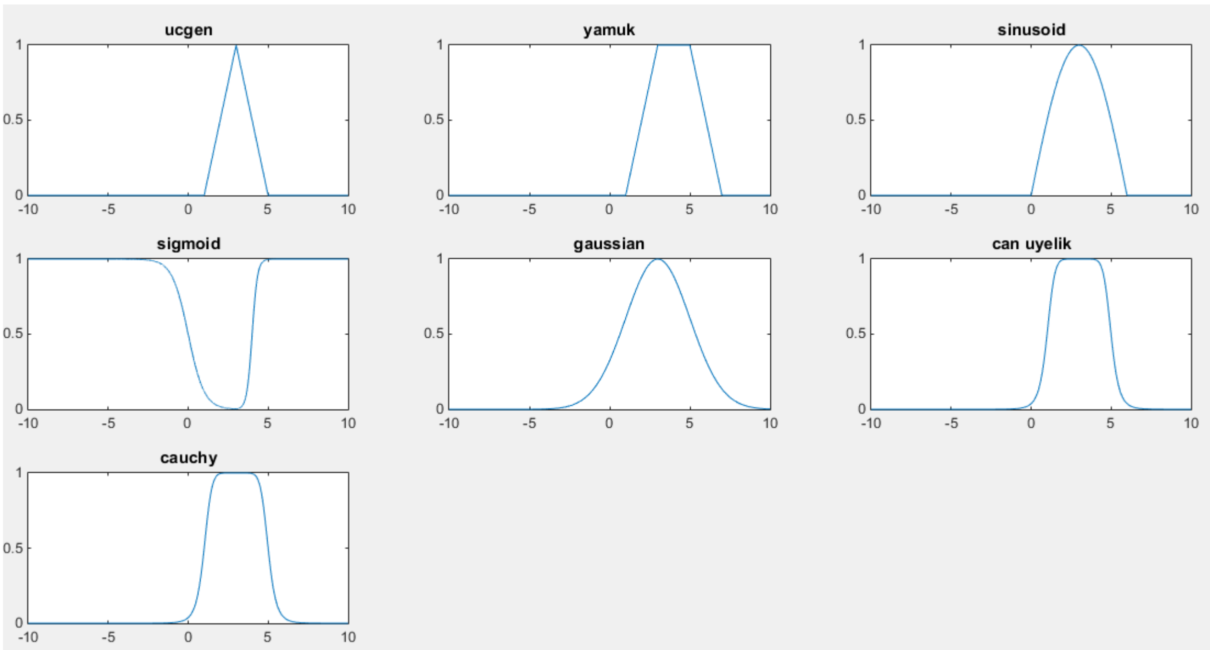
fonksiyonları istenildiğinde bir noktanın üyelik derecesini döndürebilmekte istenildiğinde de fonksiyonun bütün üyelik derecelerini döndürebilmektedir.

Aşağıda *ucgen.m* fonksiyonu yazılmıştır.

```
function [ x, mu ] = ucgen( x1, xT, x2, deger, geridonus )

for t=1:1
    mu1=(deger-x1)/(xT-x1);
    mu2=(x2-deger)/(x2-xT);
    x=max(min(mu1,mu2),0);
    mu = x;
    if(geridonus == 0) break; end
    x = -10:0.1:10;
    a = (x-x1)/(xT-x1);
    b = (x2 - x) / (x2 - xT);
    mu = max(min(a,b), 0.0);
    % plot(x, mu);
    % xlabel('x kesin sayisi'); ylabel('y kesin sayisi');
end
end
```

uygulama_uyelikfonksiyonlari.m çalıştırıldığında bütün üyelik fonksiyonlarının örnek bir çıktısının elde edilmesi sağlanmıştır.



1.4. Bulanık Küme İşlemleri (uygulama_bulanikislemler.m)

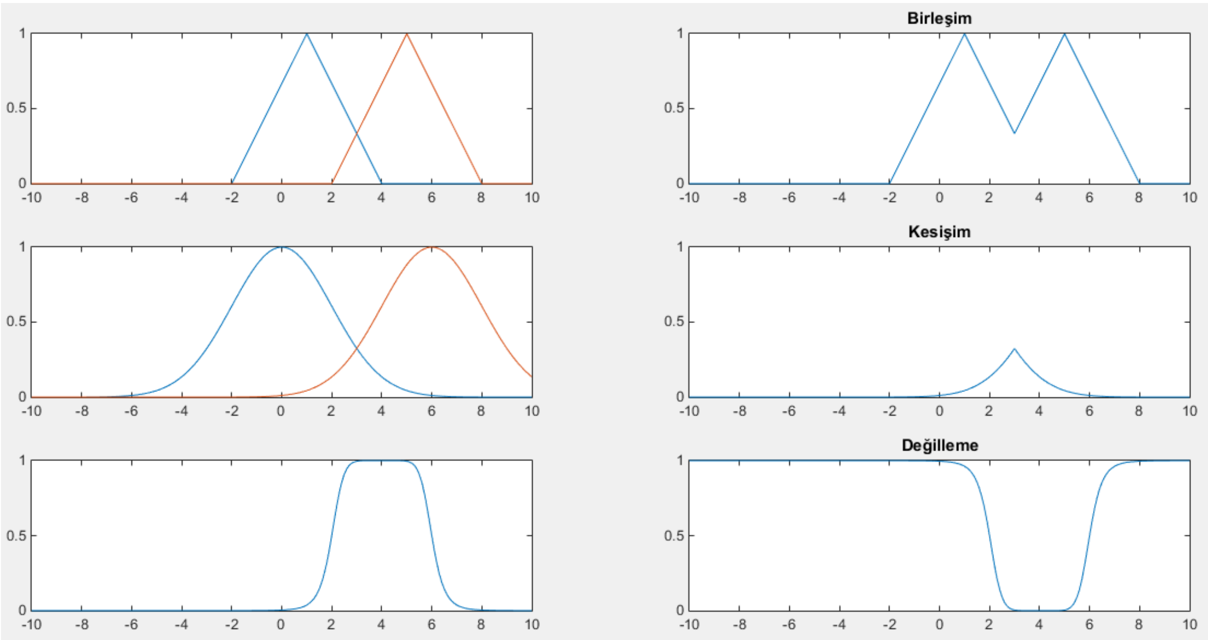
Bulanık kümelerde kesişim, birleşim ve değilleme işlemleri yapılabilmektedir. Yapılan işlem kesin kümelerdeki işlemlere benzemektedir. Birleşim işleminde aynı x değerindeki maksimum üyelik derecesi seçilir. Kesişim işleminde minimum üyelik derecesi seçilir. Böylelikle birleşim ve kesişim işlemi tanımlanabilmektedir. Yazdığım fonksiyonlar farklı üyelik fonksiyonları üzerinde işlem yapabilmektedir. Bir de değilleme işlemi vardır. Burada verilen üyelik derecelerini 1 den çıkartarak tersini buluyoruz.

Fonksiyon aşağıdaki gibidir.

Birleşim	Kesişim	Değilleme

function [x1,y]=
birlesim(x1,y1,x2,y2)
y=[];
for k=1:length(x1)
  y(k)=max(y1(k),y2(k));
end
end
	function [x1,y] =
	kesisim(x1,y1,x2,y2)
	y=[];
	for k=1:length(x1)
	   y(k)=min(y1(k),y2(k));
	end
	end
		function [x1,y] =
		degilleme(x1,y1)
		y = [];
		for k=1:length(x1)
		   y(k)= 1 - y1(k);
		end
		end

Uygulama_bulanikislemler.m fonksiyonu çalıştırıldığında bunların görselleştirilmiş uygulaması görülecektir. Çıktısı aşağıdaki gibidir.



1.5. Bulanık Küme Özellikleri (uygulama_ozellikler.m)

Bulanık kümelerin bazı özellikleri bulunmaktadır. Bunlar aşağıda açıklanmıştır.

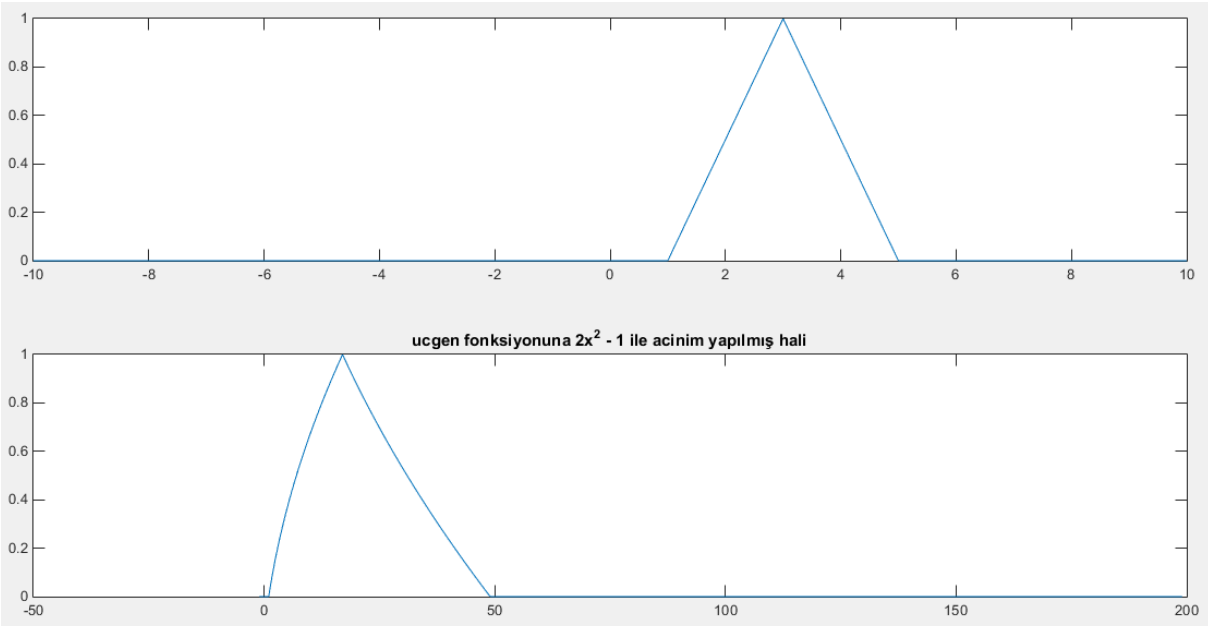
ÖZELLİK	AÇIKLAMA
Göbek	Üyelik derecesi 1 olan kısımlar
Geçiş Noktaları	Üyelik derecesi 0 olan noktalar
Sınır	Üyelik derecesi 0 ve 1 olmayan kısımlar
Destek	Üyelik derecesi 0 dan büyük olan kısımlar
Alfa – Kümesi	Üyelik derecesi ‘alfa’ değerine eşit veya büyük olan kısımlar
Bant Genişliği	Üyelik derecesi 0.5 ve daha büyük olan kısımlar

Bu özelliklerin matlab ortamında bulunmasını sağlayan fonksiyon *ozellikler.m* dosyasındadır. Uygulaması ise *uygulama_ozellikler.m* dosyasındadır.

1.6. Açınım İlkesi (uygulama_acinim.m)

Açınım ilkesi bir f(x) fonksiyonu ile bulanık kümenin farklı bir uzaya taşınması işlemidir. Bu ilke 1978 yılında Zadeh tarafından tanıtılmıştır.

Bu fonksiyon *acinim.m* dosyasında tanımlanmıştır. Uygulaması ise *uygulama_acinim.m* dosyasındadır. Bu uygulamanın çıktısı aşağıdaki gibidir. Bu uygulamada (1, 3, 5) üçgenine $2x^2 - 1$ fonksiyonu ile açınım ilkesi uygulanmıştır.



1.7. Bulanık Sonuçlandırma İşlemleri

Bulanık sonuçlandırmada belirlenen kurallar ile girişler karşılaştırılır. Bu karşılaştırmaya göre çıkış kümesinin maksimum üyelik derecesi belirlenir. Burada yapılan işlem giriş kümesi ile kural kümesinin kesişiminin alınması ve bu kesişimin tepe noktasının çıkışa aktarılarak çıkış kümesinin elde edilmesidir.

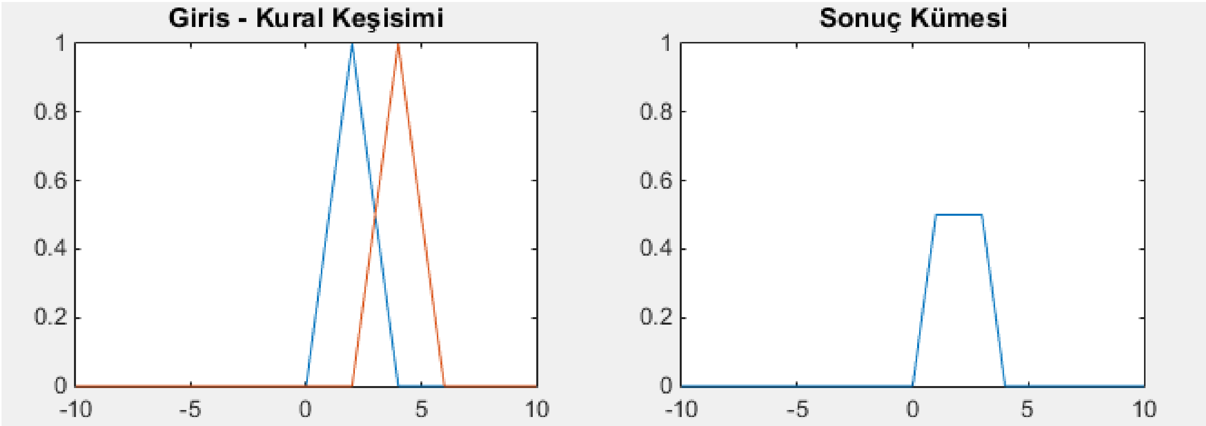
Bulanık sonuçlandırmada üç farklı yöntem bulunmaktadır. Bunlar;

- Tek Girişli Tek Kurallı Sonuçlandırma (TGTK.m)
- Çok Girişli Tek Kurallı Sonuçlandırma (CGTK.m)
- Çok Girişli Çok Kurallı Sonuçlandırma (CGCK.m)

Matlab ortamında üç yöntem ayrı ayrı oluşturulmuştur. Yukarıda belirtilen isimlerdeki dosyalarda mevcuttur. Bunların uygulaması *uygulama_TGTK*, *uygulama_CGTK*, *uygulama_CGCK* dosyalarında mevcuttur. Şimdi bunların sonuçlarını inceleyelim.

- **Tek Girişli Tek Kurallı Sonuçlandırma (uygulama_TGTK.m)**

Burada tek giriş kümesi ve tek kural kümesi olabilir. Bunlar üyelik fonksiyonları ile belirlenmektedir. Örnekte giriş, kural ve çıkış üçgen fonksiyonu ile tanımlanmıştır.

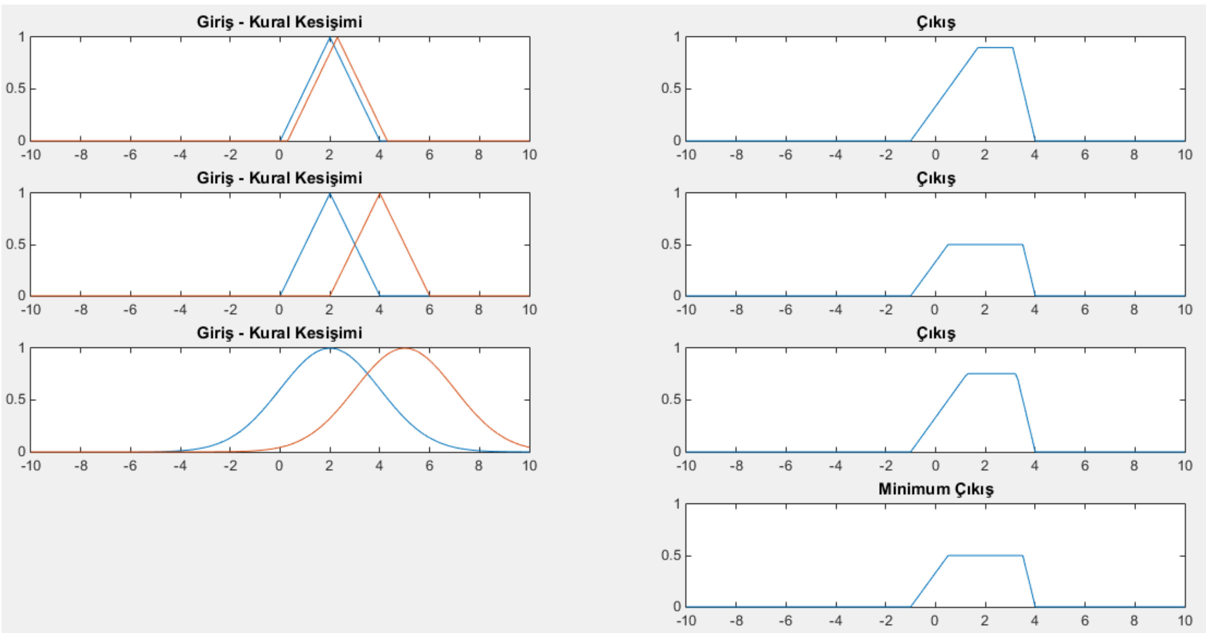


Bu örnekte giriş ve kural 0.5 noktasında kesişmiş ve bu çıkışa aktarılıp çıkış 0.5 noktasında kesilmiştir.

- **Çok Girişli Tek Kurallı Sonuçlandırma (uygulama_CGTK.m)**

Burada tek kural üzerine birden fazla giriş verilmektedir. Bütün girişlerin aynı kural ile kesişimin tepe noktası alınıp bunların minimumu çıkış kümesinin tepe noktasını belirlemektedir.

Örnekte kural kümemiz yamuktur. Üç farklı girişimiz var. İki tanesi üçgen bir tanesi ise gaussian fonksiyonudur.

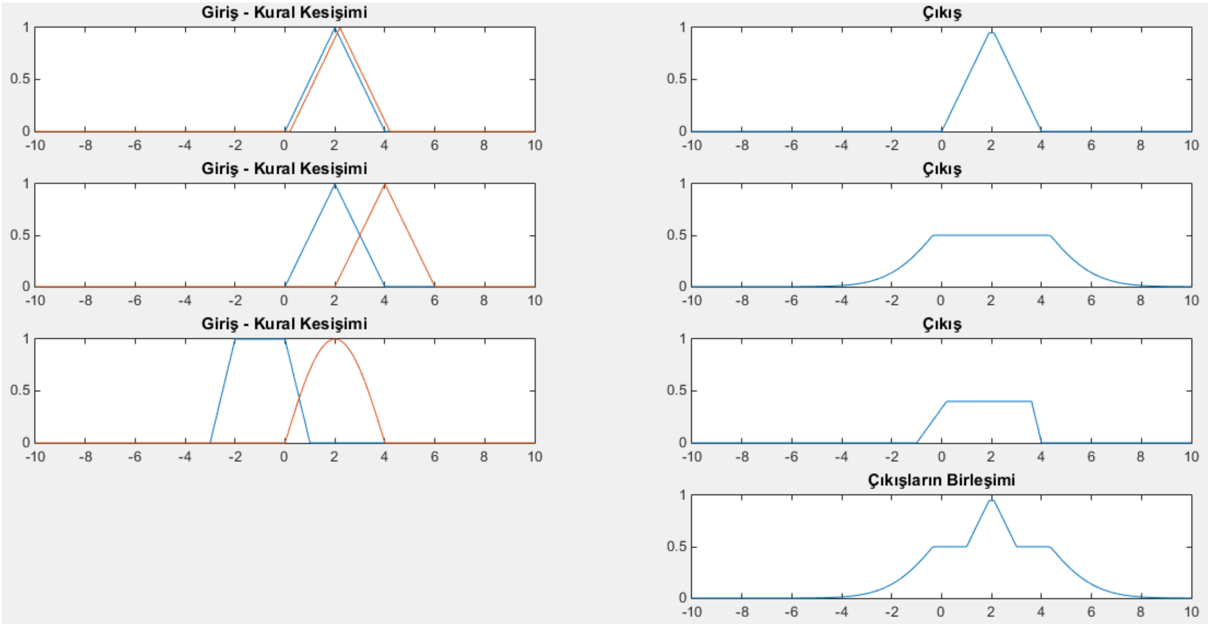


Burada ikinci giriş kural kesişim çıkışa aktarılmıştır. Çünkü minimum çıkış orada elde edilmiştir.

- Çok Girişli Çok Kurallı Sonuçlandırma (uygulama_CGCK.m)

Burada diğerlerinden farklı olarak çok kural kullanabiliyoruz. Aynı şekilde çok fazla kural da kullanıyoruz. Diğerleri gibi kesişim noktalarının maksimumunu alıyoruz. Ardında bütün çıkışları birleştiriyoruz ve çıkışımızı elde ediyoruz.

Örnekte iki kural üçgen bir kural sinusoid fonksiyondur. Girişlerin ikisi üçgen birisi yamuktur. Çıkışlar sırayla üçgen, gaussian ve yamuk fonksiyonlarından oluşmaktadır.



1.8. Mamdani, Sugeno, Tsukamoto Bulanık Modelleme (uygulama_bulanikModelleme.m)

Bulanık modellemede kesin girişler alınır ve bu girişlerin kuralları kestiği noktanın minimumu alınarak çıkışa aktarılır. Mamdani yönteminde çıkışların tepe noktalarının sonuç kümesine düştüğü nokta baz alınarak durulaştırma yapılır.

z = (u(z1)*z1 + u(z2)*z2) / (u(z1) + u(z2));

Tsukamoto durulaştırma yönteminde ise kesişim noktasının çıkış kümesine çarptığı nokta baz alınarak işlem yapılır.

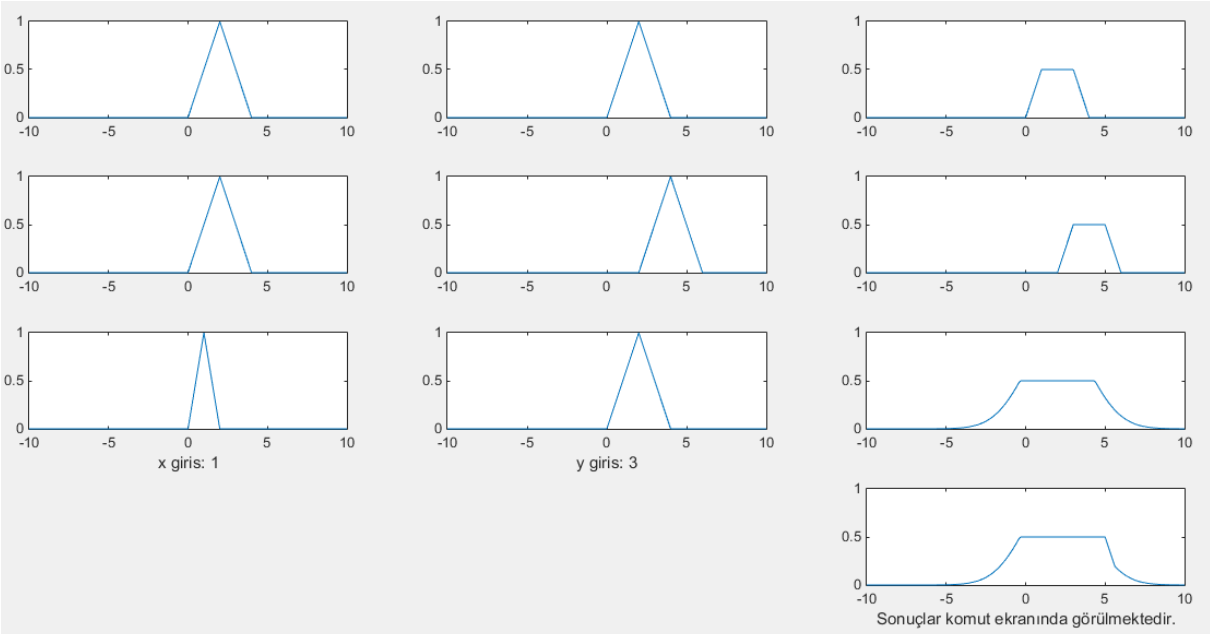
z = (u(z1) * z1 + u(z2)*z2) / (u(z1) + u(z2));

Sugeno durulaştırma yönteminde ise belirlenen fonksiyon ile z değerleri bulunur ve bunun sonucunda durulaştırma yapılır.

$z_i = p_i*x + q_i*y + r_i$;
z = (w1*z1 + w2*z2) / (w1 +w2);

Aşağıdaki uygulamanın durulaştırma sonuçları şu şekildedir.

Mamdani Sonucu: 2.716667
Tsukamoto Sonucu: 4.102138
Sugeno Sonucu: 4.500000



1.9. Sonuçlar

Sonuç olarak Bulanık Mantık dersinde gördüğümüz konuları MATLAB ortamında uygulama şansımız oldu. Bu şekilde dersteki kavramlar daha iyi anlaşıldı.

1.10. Değerlendirmeler

Bu ödevin birçok yönden olumlu etkileri oldu. Daha önce MATLAB yazılımını kullanma fırsatım olmamıştı. Bu ödev ile MATLAB hakkında gerekli önbilgiye sahip oldum.

İkinci olarak ileride çok işime yarayacağımı düşündüğüm karar verme işlemlerini hem anlayıp hem uyguladım. Bundan dolayı değerli hocama teşekkür ediyorum.

2. SİSTEM KONTROLÜ: PID VE BULANIK MANTIK DENETLEYİCİ TASARIMI

2.1. Giriş

Burada verilen sistemin girilen referans değerine ulaşıp kararlı duruma gelip gelmediğine bakılacaktır. Amacımız sistemde belli bir süre sonunda hatanın sıfıra inmesi, sürekli(kararlı) değere ulaşması ve aşma durumunun olmamasıdır. Bu sistem simülasyonunun yapılabilmesi için runge-kutta algoritmasına gerek duyulmaktadır. Sunumlardaki kodlardan faydalanarak runge-kutta fonksiyonunu yazdım.

Runge- kutta, sayısal çözümlemede diferensiyel denklemlerin çözümü için kullanılmaktadır. 2, 4, 5 vs. adımlı Runge-Kutta algoritmaları mevcuttur. Sunumda 4 adımlı kullanıldığından bende 4 adımlıyı tercih ettim. Buna göre yazdığım runge-kutta fonksiyonu *rungekutta.m* dosyasındadır.

Projede hem PI denetleyici hem de bulanık denetleyici kullanılarak sistem çözümü gerçeklenmiştir.

Ben 138. sistemi gerçekleştireceğim. Bu sistem 3 boyutlu doğrusal bir sistemdir. Bu sistemin nasıl bir sistem olduğu sonuçlar kısmında incelenmiştir.

2.2. Tasarım

Sistem parametreleri aşağıdaki gibidir.

```
A = [-4 -8 -4
      1  0  0
      0  1  0];
B = [1  0   0
     0  0   0
     0  0   0];
C = [4  1   6];
x =  [0  0   0];
U = [10 10 10];
D =  0;
dt = 0.01;
tend = 5;
t0 = 0;
k =1;
```

Burada değişkenlerin anlamı şu şekildedir.
A: Sistem matrisi
B: Giriş Matrisi
C: Çıkış Matrisi
X: Durum vektörüdür. Başlangıç değeri verilmiştir
U: Denetim çıkışı(Amaç otomatik ayarlama yapmaktır.)
tend: Bitiş süresi

Bu sistemin simülasyonunu yaparken dersteki örnek sistemlerden faydalanacağım. Sistem çıkışlarını gözlemleyip sistem hakkında yorum yapacağım.

2.3. Simülasyon Yazılımı Gerçekleme

Sistem hem PI denetleyici hem de Bulanık denetim ile simülasyon edilmiştir.

- **PI Denetleyici (oransal_integral_denetim.m)**

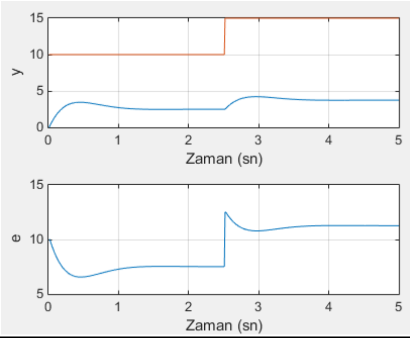
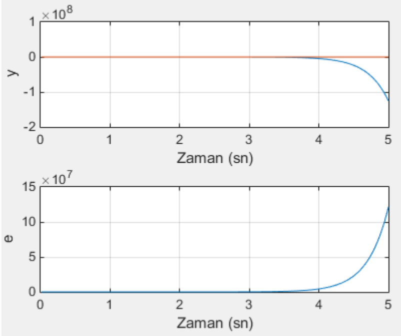
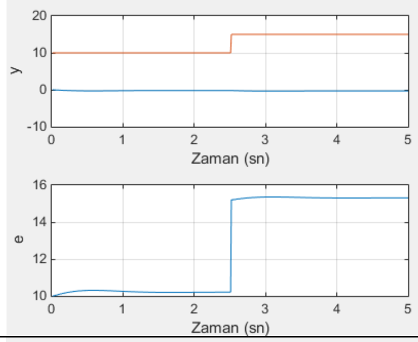
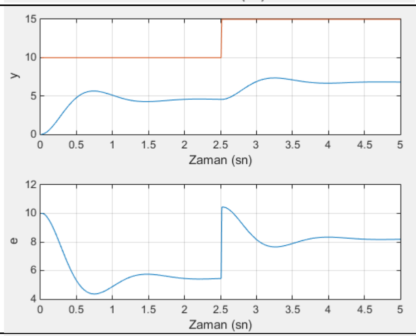
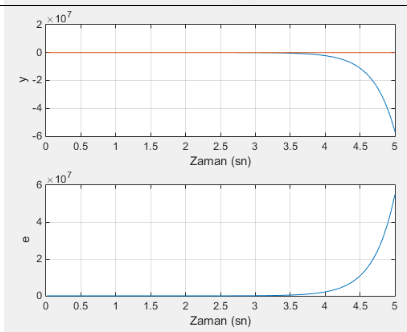
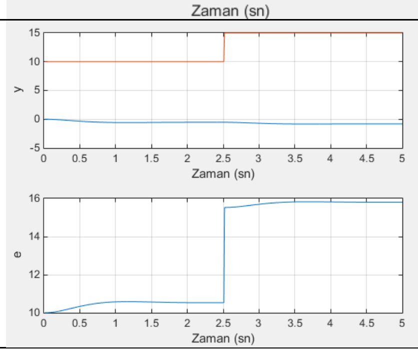
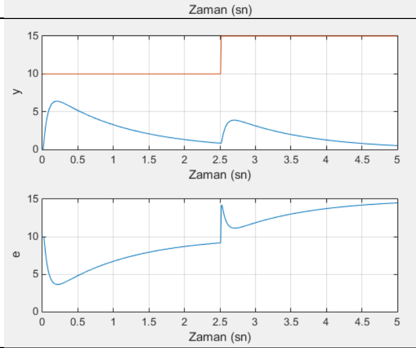
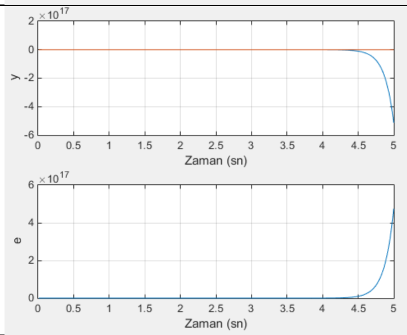
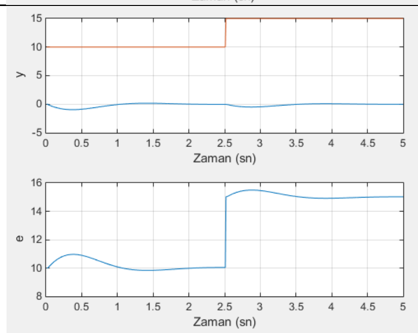
Bu sistemin kodu şu şekilde düzenlenmiştir.

```
function [  ] = oransal_integral_denetim(  )
clear;
A=[-4 -8 -4; 1 0 0;0 1 0];
B=[1 0 0 ; 0 0 0; 0 0 0];    C=[ 4 1 6];     D=0;
u=[10,10,10];
dt=0.01;    tend=5; t0=0;   k=1;
y0=0;  e10=0;

%   Baslangiç degerleri
U0=[10,10,10];    BOY=size(A);  LS=BOY(1); LK=BOY(2);
for n=1:LS
    x0(n)=0;
end
% -----------------Denetim için gerekli veriler
R0=input('Referans girisi degeri ==> ');
KP=input('KP girisi degeri ==> ');
KI=input('KI denetleyici kazancını giriniz ==> ');

   while t0<tend-dt
        if t0>(tend/2)  r0=R0+0.5*R0;
        else  r0=R0;
        end
        e0=r0-y0;
        ekp=KP*e0;
        e1=e10+dt*KI*e0;
        e10=e1;   u=(e1+ekp)*U0;
        [x1,x2,x3]=rungekutta(A,B,u,x0,dt);  %Runge ile denklem çöz.
        UU(k)=u(1);
        e(k)=e0;    r(k)=r0;   y(k)=x0(1); y0=y(k);
        t(k)=t0+dt;    t0=t(k);
        x0(1)=x1(1); x0(2)=x2(1);  x0(3)=x3(1);
        XX(k,1)=x1(1); XX(k,2)=x2(1); XX(k,3)=x3(1);
        k=k+1;
end;
% -------------------------- Grafikler
subplot(211)
plot(t,y,t,r); xlabel('Zaman (sn)'); ylabel('y');grid
subplot(212)
plot(t,e);  xlabel('Zaman (sn)');  ylabel('e'); grid
end
```

Sisteme verilen farklı girişlerde elde edilen sonuçlar aşağıda gösterilmiştir.

Simulator	B = [1; 0; 0]	B = [0; 1; 0]	B = [0; 0; 1]
R0 = 10 KP = 0.2 KI = 0.4			
R0 = 10 KP = 0 KI = 1			
R0 = 10 KP = 1 KI = 0			

Burada elde edilen grafikler sonuçlar başlığı altında değerlendirilecektir.

- **Bulanık Denetim**

Bulanık denetim kodu şu şekildedir.

```
function [  ] = bulanik_denetim(  )
clear;
 A=[-4 -8 -4; 1 0 0;0 1 0];
 B=[1 0 0 ; 0 0 0; 0 0 0];   C=[ 4 1 6];     D=0;
u1=10;   u2=10; u3 = 10;
dt=0.01;    tend=5; t0=0;   k=1;
%   Baslangiç degerleri
U0=[u1;u2;u3];    BOY=size(A);  LS=BOY(1); LK=BOY(2);
for n=1:LS
    x0(n)=0;
end
% -----------------Denetim için gerekli veriler
r0=input('Referans girisi degeri ==> ');
tend=input('Simülasyon bitis zamani ==> ');
    EMAX  = r0;        EMIN  = -EMAX;
    DEMAX = EMAX/10;   DEMIN = -DEMAX;
    DUMAX = 1;         DUMIN = -1;
    NLe=EMIN;   NTe=NLe;    NRe=0;
    SLe=NTe;    STe=0;      SRe=EMAX;
    PLe=STe;    PTe=EMAX;   PRe=PTe;
    NLde=DEMIN;     NTde=NLde;     NRde=0;
    SLde=NTde;      STde=0;        SRde=DEMAX;
    PLde=STde;      PTde=DEMAX;    PRde=PTde;
    NLdu=DUMIN;       NTdu=DUMIN;     NRdu=0;
    SLdu=NTdu;        STdu=0;         SRdu=DUMAX;
    PLdu=STdu;        PTdu=DUMAX;     PRdu=PTdu;
%  Hata ve denetleyici için baslangiç degerleri
```

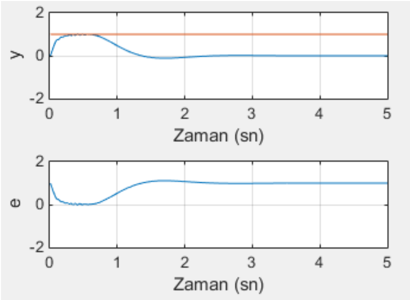
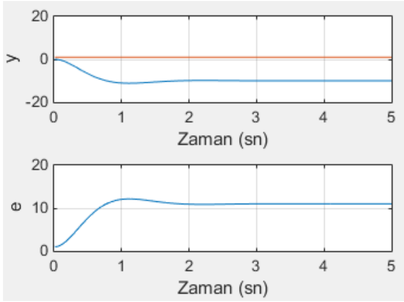
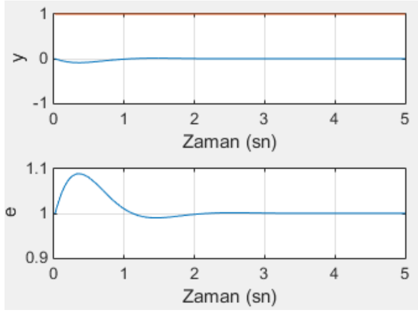
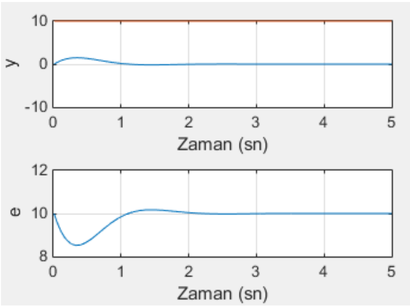
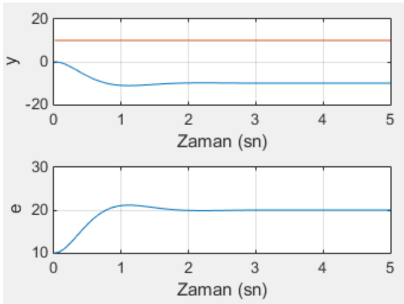
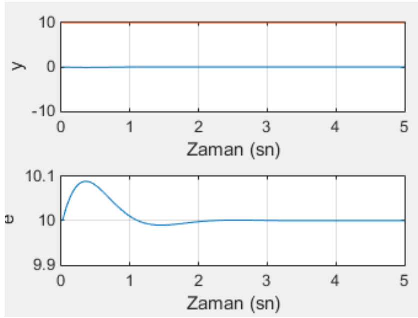
```
   ee=EMAX;   dee=0;  E=EMAX;  dE=0;
   e0=EMAX;   e(1)=0; e(2)= 0;   de=e(2)-e(1);
   C(1)=0;
% membership matrix
     DU=[ NTdu NTdu STdu
          NTdu STdu PTdu
          STdu PTdu PTdu ];
% -------------- iteratif çözüm
while t0<tend-dt
      % Bulaniklastirma
      E=limiter(EMIN,EMAX,ee);%---------- limit E
      [x, mu] = ucgen(NLe,NTe,NRe,E,0);
      FSE(1) = x;
      %FSE(1)=ucgen(NLe,NTe,NRe,E,0);
      [x, mu]=ucgen(SLe,STe,SRe,E,0);
      FSE(2) = x;
      [x, mu]=ucgen(PLe,PTe,PRe,E,0);
      FSE(3) = x;
      DE=limiter(DEMIN,DEMAX,dee); %----- limit DE
      [x, mu]=ucgen(NLde,NTde,NRde,DE,0);
      FSDE(1) = x;
      [x, mu]=ucgen(SLde,STde,SRde,DE,0);
      FSDE(2) = x;
      [x, mu]=ucgen(PLde,PTde,PRde,DE,0);
      FSDE(3) = x;
      N1=length(FSE);
      N2=length(FSDE);
      NM=N1*N2;
      nn=1;
      for mm=1:N1
        for qq=1:N2
            FSDU(nn)=min( [FSE(mm) FSDE(qq)] );
            nn=nn+1;
        end
      end
      nn=1;
      for mm=1:N1
         for qq=1:N2
             DDU(nn)=FSDU(nn)*DU(mm,qq);
             nn=nn+1;
         end
      end
      DUTOP1 = sum(DDU) ;
      DUTOP2 = sum(FSDU);
      DV = (DUTOP1/DUTOP2);

     % ------------------------------- PI etki
     C(k+1) = C(k) + DV;
     CC=limiter(0,DUMAX,C(k+1));
     UU0 = CC*U0;
     [x1,x2,x3]=rungekutta(A,B,UU0,x0,dt);%Runge ile denklem çöz.
     UU(k)=UU0(1);
     t(k)=t0+dt;    t0=t(k);
     r(k)=r0;       y(k)=x0(1);
     e(k)=r(k)-y(k);    de(k)=e(k)-e0;
     ee=e(k);  dee=de(k);   e0=e(k);   duty(k)=CC;
       x0(1)=x1(1);  XX(k,1)=x1(1);
       x0(2)=x2(1);  XX(k,2)=x2(1);
       x0(3)=x3(1);  XX(k,3)=x3(1);
      k=k+1;
end;
% ------------------------- Grafikler
subplot(211)
plot(t,y,t,r); xlabel('Zaman (sn)'); ylabel('y');grid

subplot(212)
plot(t,e);  xlabel('Zaman (sn)');  ylabel('e'); grid

end
```

Bulanık denetim ile elde edilen sonuçlar şu şekildedir.

	B = [1; 0; 0]	B = [0; 1; 0]	B = [0; 0; 1]
r0:1 tend:5			
r0:10 tend:5			

2.4. Sonuçlar

PI denetleyici sonuçlarına göre girişimiz ne olursa olsun sistemimiz hiçbir zaman referans değerine ulaşamamaktadır. Fakat giriş [1;0;0] olduğunda sistem belli bir süre sonra kararlı duruma geçiyor fakat referans değerine ulaşamıyor. Mesela bir fanlı soğutma sistemine ortamı 10 derece soğutması için referans değeri giriyoruz. Fakat bu soğutma sistemi ortamı belli bir derece soğutabilir. Çünkü klima gibi herhangi bir soğuk hava estirmiyor ortamda dönerek hava esintisi oluşturmaktadır.

Bulanık denetim incelendiğinde ise referans değerine ulaşmak için bir salınım yapmakta fakat yine başlangıç değerine dönmektedir. Referans değeri küçük olduğunda kısa bir süre referans değeri etrafında salınım yapmakta ve yine başlangıç değerine düşmektedir. Fakat referans değerimiz büyük olduğunda referans değerine ulaşmadan giriş değeri etrafında salınım yapmakta ve giriş değerine düşmektedir.

2.5. Değerlendirmeler

İkinci kısımda herhangi bir sistemin tasarlanmasını gördük. Robot yapımında kullanılabilecek sistemlere benzemektedir. Bu derste kararlı sistemlerin nasıl oluşturulabileceğini ve tasarlanabileceğini gördük. Yapılan uygulamanın mesleki gelişime çok katkısı olduğunu düşünmekteyim.

3. KAYNAKLAR

ALTAŞ, İsmail H., Bulanık Mantık Ders Sunumları, (Trabzon, 2008)
https://tr.wikipedia.org/ (Özgür Ansiklopedi)
http://www.ee.hacettepe.edu.tr/~solen/Matlab/Coskun%20Tasdemir%27den/Matlab%20Turkce%20kullanma%20kilavuzu.pdf