T.C.
MARMARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MODEL ÖNGÖRÜLÜ KONTROL BENZETİMİ İÇİN EĞİTİM VE UYGULAMA ARAYÜZ TASARIMI

ERAY YILMAZLAR

YÜKSEK LİSANS TEZİ
MEKATRONİK ANABİLİM DALI

DANIŞMAN
Yrd. Doç. Dr. Erkan KAPLANOĞLU

İSTANBUL (2012)

T.C.
MARMARA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MODEL ÖNGÖRÜLÜ KONTROL BENZETİMİ İÇİN EĞİTİM VE UYGULAMA ARAYÜZ TASARIMI

ERAY YILMAZLAR
(526209004)

YÜKSEK LİSANS TEZİ
MEKATRONİK ANABİLİM DALI

DANIŞMAN
Yrd. Doç. Dr. Erkan KAPLANOĞLU

İSTANBUL (2012)

TEŞEKKÜR

Yüksek lisans eğimimde ve tez çalışmamda desteğini, rehberliğini ve değerli zamanını esirgemeden bana yol gösteren her zaman yardımcı olan tez danışmanım Sayın Yrd. Doç.Dr. Erkan KAPLANOĞLU'na ve çalışmalarım sırasında bana destek olan aileme sonsuz teşekkürlerimi sunarım.

İÇİNDEKİLER

ÖZET

MODEL ÖNGÖRÜLÜ KONTROL BENZETİMİ İÇİN EĞİTİM VE UYGULAMA ARAYÜZ TASARIMI

Son senelerde özellikle süreç kontrol sistemlerinde, ileri kontrol tekniklerinin önemi endüstride giderek artmaktadır. Günümüzdeki süreç kontrol yapılarına bakıldığında ise kontrol sürecine etki eden birim sayısı her geçen gün artmaktadır. Gelenekselde ise bu durum, basınç, sıcaklık, sıvı seviyesi ve akış gibi değişkenlerin her birini ayrı ayrı kontrol etmektedir. Bu durum, süreç otomasyonu için ideal bir çözüm değildir. Bütün değişkenleri aynı anda kontrol edebilen tek bir kontrolör tasarlanması arzu edilir. Model öngörülü kontrol yöntemi ise gelişen bu kontrol süreçlerinde hem öngörü işlemini yapabilmesi hemde birden fazla etkeni kontrol edebilmesi bakımından önemli bir araştırma alanı haline getirmiştir.

Bu çalışmada öncelikle model öngörülü kontrolün yapısı, tarihi, çalışma şekli, çeşitleri, sistemlere uygulama şekilleri hakkında bilgiler verilmiştir. MATLAB MPC Toolbox kullanılarak model öngörülü kontrol sürecinle ilişkili algoritmaların, komutların uygulanışı ve kullanılma yöntemleri hakkında bilgiler verilmiş, hesaplamaları yapılmıştır. MATLAB GUI arayüz programı kullanılarak model öngörülü kontrol yönteminin eğitiminin gerçekleşmesi için uygulama arayüzleri tasarlanmıştır.Bu tasarımlara ait kodlar MATLAB m file üzerinde yazılarak tasarım çalıştırılmıştır.

Sonuç olarak tek girişli tek çıkışlı ya da çok girişli çok çıkışlı sistemlerin kısıtlamalı, kısıtlamasız model öngörülü kontrol işlemlerinin gerçekleşebildiği ve gözlemlenebildiği eğitim ve uygulama arayüzleri gerçekleştirilmiştir.

02.2012 **Eray YILMAZLAR**

ABSTRACT

DESIGN OF EDUCATION AND APPLICATION INTERFACE FOR SIMULATION OF MODEL PREDICTIVE CONTROL

Recently, the importance of the advanced control techniques are increasing especially in the process control systems. In today's looking at the structures of the process control, the number of the units is increasing every day affect the control process. In the conventional controllers each variable is controlled separately such as pressure, temperature, liquid level, flow etc. This is not ideal solution for the process automation. Able to control all the variables at the same time it is desirable to design a single controller. In model predictive control method forecasting operation and make the process of developing this control both the to control in terms of more then one factor has become an important research area.

In this study, model predictive control structure, history, mode of operation, types and systems application typess information about the was given. The model predictive control algorithms, which is prepared using MATLAB MPC Toolbox, information about command procedure and user manual was given. System's calculations were performed. MATLAB GUI interface program will take place using a model predictive control process control and system information input is designed interfaces. Main interface for calculating and tranining of MPC has been designed with MATLAB GUI. All codes of this design written with MATLAB M. files.

As a result, single-input single-output and multi-input multi-output systems constrained or unconstrained model predictive control carried out and observable in the training and application interfaces.

02.2012 **Eray YILMAZLAR**

SEMBOLLER

A,B,C,D **:** Durum uzayı Matrisleri

d : Bozucu sinyali

f **:** Serbest cevap vektörü.

G **:** Transfer fonksiyon matrisi

$\mathbf{G_{N123}}$ **:** GPC de ufuk değerleri ile öngörü matrisi

$\mathbf{I_{nxn}}$ **:** (nxn) Birim matris

J **:** Öngörü maliyet (ölçüt)fonksiyonu

K **:** Kontrolör kazanç matrisi

M : Kontrol ufku

N **:** Ufuk sayısı

$\mathbf{N_1}$ **:** Ufuk endüşük değeri

$\mathbf{N_2}$ **:** Ufuk enbüyük değeri

Nu **:** Kontrol ufuk değeri

r **:** Referans girişi

P **:** Öngörü ufku

u **:** Kontrol sinyali.

uwt : Giriş ağırlık matrisi

ulim : Giriş kısıtlama matrisi

v : Ölçülemeyen bozucu

Y **:** Sistem çıkış transfer fonk.matrisi

$\hat{\mathbf{Y}}$ **:** Öngörü çıkışı vektörü

ylim : Çıkış kısıtlama matrisi

ywt : Çıkış ağırlık matrisi

ω_n **:** Doğal freakans

w **:** Referans vektörü

z : Ölçülemeyen gürültü

τ **:** Zaman sabiti

Λ **:** Bağıl kazanç matrisi

λ **:** Kontrol sinyali artışı için ağırlık faktörü

KISALTMALAR

AMPC **:** Adaptive Model Predictive Control

DMC **:** Dynamic control Matrix

GMPC **:**Generalized Model Predictive Control (Genelleştirilmiş Model Öngörülü Kontrol)

GPC :Generalised Predictive Control (Genelleştirilmiş Öngörülü Kontrol)

GUI : Grafik Kullanıcı Arayüzü

QDMC : Quadratic Dynamic Matrix Control

LMI : Lineer Matrix Inequalities

MIMO **:** Multi Input Multi Output (Çok Giriş Çok Çıkış Sistemleri)

MPC **:** Model Predictive Control (Model Öngörülü Kontrol)

MVC **:** Multivariable Control(Çokdeğişkenli kontrol)

PID :Proportional Integral Derivative(Oransal İntegral Türev Kontrol)

PFC : Predictive Functional Control

ŞEKİLLER

SAYFA NO

TABLOLAR

SAYFA NO

BÖLÜM I

GİRİŞ VE AMAÇ

Model öngörülü kontrol yönteminin kullanımı günümüzde hızla artmaktadır. Model öngörülü kontrolün yapısı incelendiğinde kontrol edilecek sistemin matematiksel modelleri ile sistemin gelecekteki davranışını tahmin ve eniyileme prensibine göre sistemden çıkış alan kontrolörlere genel olarak model öngörülü kontrol adı verilmektedir. Model öngörülü kontrol yönteminin rahatlıkla uygulanabileceği ve gözlemlenebileceği bir arayüz tasarımı ile de sistemi oluşturan modellerin arayüz üzerine girilmesi ve bu model bilgileri üzerinden kontrol işlemlerinin kolaylıkla yapılması hedeflenmektedir.

Model öngörülü kontrol yönteminin uygulanışı ve çalışma şekli hakkında günümüzde uluslararası alanda pek çok çalışmalar ve yenilikler yapılmaktadır. Ülkemizde ise bu durumum, 2010 yılında yapılan TÜBİTAK'ın "Bir Kısıtlamalı Öngörülü Kontrol Algoritmasının Endüstriyel Uyarlaması ve Endüstri Beklentileri Açısından Türkiye'deki Mevcut Durum" projesinden elde edilen bilgiler eşliğinde öngörülü kontrol algoritmaları hakkında bu konuyla ilgili çalışanlarda yeterli bilgiye sahip olmadıkları tespit edilmiştir. Üniversitelerin ilgili bölümlerinde verilen dersler ve içerikleri incelendiğinde ise öngörülü kontrolün bazı yüksek lisans derslerinde bir bölüm başlığı olarak okutulduğu gözlemlenmiştir [1].

Bu çalışmalardan yola çıkılarak süreç kontrol algoritmaları içerisinde önemli bir yere sahip olan model öngörülü kontrol algoritmasının öğretilmesini yaygınlaştırmak için eğitim ve uygulama arayüzü tasarlanmıştır. Bu arayüzle kullanıcılar model öngörülü kontrolün diğer kontrol sistemlerinden farkını, avantajlarını, dezavantajlarını görebilecek, hangi sistemlere nasıl ve ne şekilde uygulanabileceği ile ilgili bilgi veren eğitim materyallerine ulaşabilecek ve çeşitli sistemler için uygulamalar gerçekleştirebileceklerdir.

BÖLÜM II

GENEL BİLGİLİER

II.1 ÖNGÖRÜLÜ KONTROL

Öngörülü kontrol, endüstriyel kontrolde önemli bir yeri olan gelişmiş bir kontrol metodudur. Önceleri sadece Petro-kimya endüstrisinde uygulanırken günümüzde diğer sistem kontrol sektörlerinde kullanımı artmaktadır. Elbette ki öngörülerin kullanılması kontrolde çok eski ve alışılmış bir fikirdir. Örneğin iyi bilinen Smith öngörücüsü geniş zaman gecikmeleri olan sistemler için kullanılan bir kontrolördür. Bu kontrolörün temelinde zaman gecikmesi olmayan dolayısıyla sistem çıkışının öngörüsünü sağlayan bir sistem modeli vardır. Öngörünün elde edilmesi sistemde gecikme ile ortaya çıkan geniş faz aralıklarına sebep olan faz yolunun elde edilmesi için kullanılan bir araçtır. Daha temel olarak türev davranışını kullanan her kontrolör veya faz yolu düzenleyicisi bazı sinyallerin öngörüsünün sağlayıcısı olarak görülebilir.

Öngörülü kontrolde kavramlar sezgisel ve parametre ayarları diğer yöntemlere göre kolaydır. Çok basit dinamiğe sahip sistemlerden, kararsız, minimum fazlı olmayan ya da çok uzun ölü zamanı bulunan sistemler gibi daha karmaşık dinamiklere sahip çok çeşitli süreçlerin kontrolünde kullanılabilir. Önemli yönlerinden biri çok değişkenli sistemlerin kontrolünde rahatlıkla kullanılabilir olmasıdır. Yapısı gereği ölü zaman etkisini giderme özelliğine sahiptir. Kontrol sonucunda uygulaması kolay lineer bir kontrol kuralı elde edilir. Geleceğe ilişkin giriş işareti bilindiğinde oldukça yararlı ve kullanışlıdır. Belirli temel ilkeler üzerine kurulmuş geliştirmeye tamamen açık bir metoda sahiptir. En önemli avantajlarından biri de kısıtlamalarla başa çıkabilecek, kısıtlamaları tasarım sürecine sistematik olarak katabilen geliştirilmiş biçimleri olmasıdır.

Sonuçta uygulaması kolay, işlem karmaşıklığı düşük olan bir kontrol kuralı elde edilse de, bu kontrol kuralının elde edilmesi PID kontrolörlerinden daha

karmaşıktır. Sistem dinamiğinin değişmediği durumlarda kontrolörün elde edilme işlemi önceden yapılabilir, ancak uyarlamalı kontrol yapılıyorsa tüm hesap işlemleri her örnekleme zamanında tekrarlanır. Sınırlandırmalar göz önüne alındığında işlem karmaşıklığı daha da artacaktır. En önemli sorun ise sisteme ilişkin uygun modelin elde edilmesidir. Algoritma sistem modelinden yararlanarak geleceğe ilişkin sistem yanıtının elde edilmesine dayandığından yapılacak olan bir hatada gerçek sistemle yanıt arasındaki farklılık artıkça istenilen sistem yanıtını elde etmekte sorunlar yaşanır [2,3,4].

Öngörülü kontrolün bu sistemlerde başarıya ulaşmasındaki ana sebepler;

1. Çok değişkenli kontrol problemlerin ele alınması
2. Eyleyici sınırlamalarının dikkate alınabilirliği
3. Kontrol güncellenme oranı diğer sistemlerle karşılaştırıldığında oldukça düşüktür. Bu sebeple çevrim içi hesaplama için oldukça fazla zamana gereklidir.

II.1.1 Öngörülü Kontrol Literatür Taraması

Öngörülü kontrol ile ilgili yapılan çalışmalar incelendiğinde ilk olarak Kalman'nın 1960 yılında ikinci dereceden lineer regülatör (LQR) dizaynı ile başlamıştır. 1963 yılında da Propoi tarafından önerilen ve model öngörülü kontrolün en önemli kısımlarından olan kayan ufuk prensibi (receding horizon princible) üzerinde çalışmalar yapılmıştır.

1970'lerin sonuna doğru Model Öngörü Kontrol yönteminin endüstride uygulanması ile ilgili pek çok makale çıkmaya başlamıştır. 1978 yılında Richalet, model öngörülü kontrolü Model öngörülü sezgisel control adı altında sunmuş ve PID kontrol sistemleri ile gerçekleşen problemleri sezgisel kavramlarla birleştirerek ilgili parametreleri daha kolay kontrol etmiştir [5].

1980'li yıllara gelindiğinde Cuter ve Ramerker öngörülü kontrol üzerinde yapmış oldukları çalışmalar sonucunda dinamik matris kontrol(DMC) yöntemini geliştirmişlerdir. Bu kontrol yöntemi ile kısıtlamalı sistemlerin işletiminde optimal çözümler getirilmiş ve kontrol sinyali, doğrusal programlama probleminin ard arda çözülmesi ile hesaplanmıştır. DMC en çok bilinen ticari öngörülü kontrol yöntemi olarak günümüzde de kullanılmaktadır. Clarke ve diğerleri. 1987 yılında Genelleştirilmiş Öngörülü Kontrol(GPC) yöntemini önermişlerdir. Bu yöntem

Genelleştirlmiş Minimum Varyans (GMV) fikrine dayanmaktadır. Akademik ve endüstri çalışmalarında en yaygınlaşan yöntemlerden biridir [6,7].

1990'lı yıllar incelendiğinde, Rawling ve Muske 1993'de sonsuz öngörü ufku üzerinde çalışmalar gerçekleştirmişlerdir. Morari'nin 1994'de durum uzay matrislerinin model öngörülü kontrol algoritmalarına uyarlanabilmesini sağlamıştır. Bu çalışma sayesinde liner olamayan ve karmaşık sistemlerin kontrolünde büyük fayda sağlamıştır [8].

2000'li yıllara geldiğimizde Maciejowski'nin öngörü kontrol ve kısıtlama adlı kitabı model öngörülü kontrol açıklayan en iyi kitap seçilmiştir ayrıca çağımızda yazılımcılığın gelişmesiyle de birçok firma model öngörülü kontrol ile ilgili çalışmalar yapmış ve model öngörülü kontrolü bilgisayar tabanlı bir kontrol sistemi haline dönüştürmüşlerdir.

II.1.2 Kayan Ufuk (Horizon) Kavramı

Öngörülü kontrol algoritmasının en temel kavramı olan ufuk kavramı ve temel stratejisi şekil II.1'de görülmektedir. Örnek olarak ayrık zamanda SISO sistem alınmıştır. Sistemin her hangi bir andaki zamanı k ile tanımlanmıştır. k anındaki sistemin çıkışı y(k) ve önceki sistem çıkışı da şekilde görülmektedir. İstenen değer yörüngesi her bir t anı için s(t) olarak gösterilmiştir. Sistem modelinden yararlanarak, öngörü ufku denilen, belirlenmiş bir ufuk boyunca gelecekteki sistem yanıtı hesaplanır. Sistemin zaman sabiti T_{ref} sistemin cevabından hesaplanır. Eğer sistemde o andaki hata;

$$\varepsilon\ k\ = s\ k\ - y(k) \tag{II.1}$$

ise referans yörüngesi, hata gibi i adım sonrası için seçilebilir. Eğer çıkış tam olarak takip ettiği düşünülürse;

$$\varepsilon\ k+i\ = e^{-iT_s/T_{ref}}\varepsilon(k) \tag{II.2}$$

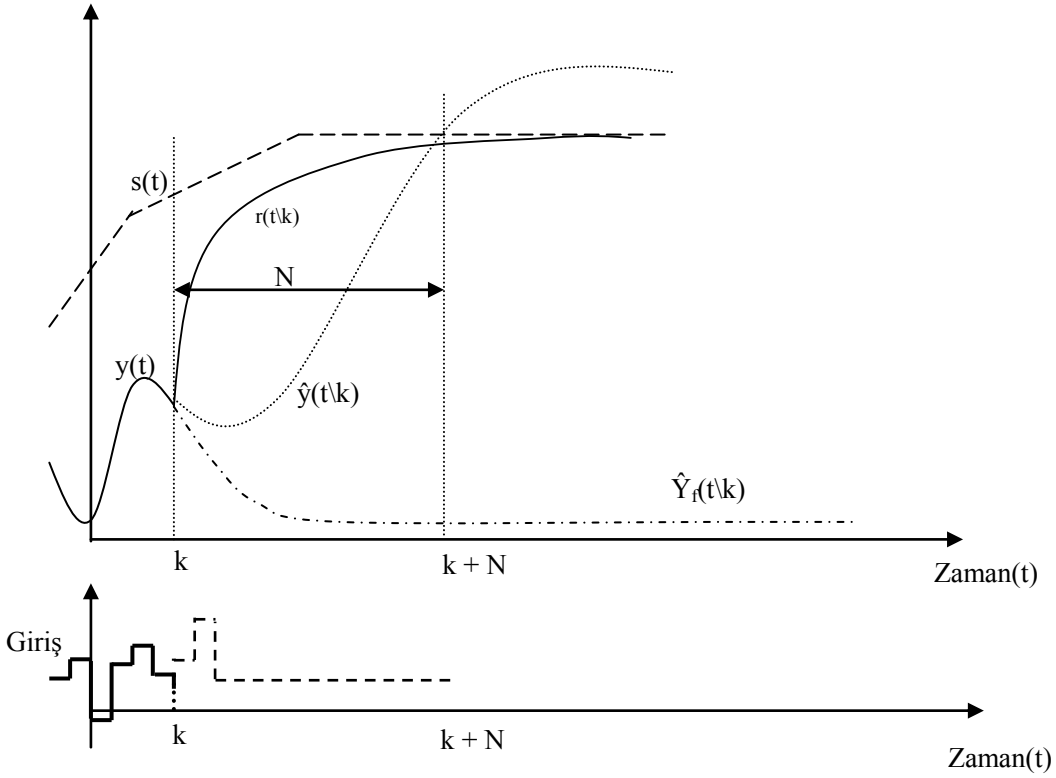
olabilir. Burada T_s örnekleme zamanı ve $\lambda = e^{iT/_s T_{ref}}$ $(0<\lambda<1)$ Yani, Referans yörünge aşağıdaki gibi tanımlanabilir.

$$r(k+i/k) = s(k+i) - \varepsilon(k+i) \tag{II.3}$$

$$\varepsilon\ k+i\ = e^{-iT_s/T_{ref}}\varepsilon(k) \tag{II.4}$$

r(k+i/k),k anındaki koşullara bağlı olarak referans yörüngeyi gösterir. Referans yörüngesi başka bir değişle, o andaki çıkış değerden başlayarak ilerideki bir zamanda istenen değere s(t) ulaşan yörünge olarak da tanımlanabilir. Öngörülü kontrolör,

geçerli bir zamandan başlayarak öngörü ufku boyunca sistemin davranışını öngören "iç model"e sahiptir. Bu öngörü davranışı varsayılan giriş yörüngesine $u(k+i/k)\ i = 0,1,\ldots,N-1)$ bağlıdır. İç model doğrusal olarak kabul edilebilir. Bu şekilde en iyi giriş hesaplanması diğerlerine nazaran daha kolay şekilde yapılabilir.



Şekil II.1 Öngörülü Kontrolün Temel Kavramı

Ölçülen çıkış $y(k)$, $u(k)$ giriş değerine göre elde edilebilir. İç modele göre $y(k)$ sadece $u(k)$ değerine değil geçmiş giriş değerlerine $(u(k-1),\ u(k-2),\ldots.)$ de bağlıdır. En basit durumda giriş yörüngesini $k+N$ zamanında öngörü ufkunun sonundaki sistem çıkışını gerekli $r(k+N)$ değerine getirerek seçilmesi denenebilir. Bu durumda Richalet [9] terminolojisini kullanarak $k+Hp$ zamanında tek bir tesadüfî nokta olacağı söylenebilir. Bunu yapabilen pek çok giriş yörüngesi $\{\hat{u}(k|k), \hat{u}(k+1|k), \ldots, \hat{u}(k+N-1|k)\}$ ve bunların içerisinden örneğin, en küçük giriş enerjisi gerektiren herhangi birisi seçilebilir. Fakat giriş yörüngesine daha az sayıdaki değişken tarafından parametrize edilmiş bazı temel yapıları uygulamak daha tercih edilebilir ve uygundur. Şekil II.1 girişin öngörü ufkunun ilk 3 basamağı üzerindeki değişimini ve daha sonra da sabit kalmasını göstermektedir: $\hat{u}(k+2|k) = \hat{u}(k+3|k) = \ldots \hat{u}(k+Hp-1|k)$, böylece burada seçilebilecek 3 parametre

$\hat{u}(k|k), \hat{u}(k+1|k), \hat{u}(k+2|k)$ vardır. Olası en basit yapı girişi öngörü ufkunda sabit tutmaktır.

$\hat{u}(k|k), = \hat{u}(k+1|k) = \ldots = \hat{u}(k|k)$, bir giriş yörüngesi seçildiğinde yörüngenin sadece ilk elemanı sisteme giriş olarak uygulanır. Yani $u(k), \hat{u}(k|k)$ olarak ayarlanır. Burada $u(k)$ uygulanan asıl sinyali göstermektedir. Daha sonra çıkış ölçümünün tüm çevrimi, öngörü, ve giriş yörünge tespiti tekrarlanır, bir örnekleme aralığı sonrasında yeni çıkış ölçümü $y(k+1)$ elde edilir; yeni bir referans yörüngesi $r(k+i|k+1)$ $i(=2,3,\ldots)$ tespit edilir: öngörüler $i = 1,2,\ldots,Hp$ ile $k+1+i$ ufuğu üzerinden yapılır. Yeni bir giriş yörüngesi $i = 0,1,\ldots,Hp-1$ ile $\hat{u}(k+1+i|k+1)$ seçilir ve son olarak sonraki giriş sisteme uygulanır $u(k+1) = \hat{u}(k+1|k+1)$. Öngörü ufku önceki ile aynı uzunlukta kaldığı ancak yol boyunca her bir adımda tek bir örnekleme aralığı boyunca kaydığı için bu tip bir sistem kontrolü geri çekilen ufuk stratejisi olarak adlandırılır [2].

II.2 MODEL ÖNGÖRÜLÜ KONTROL

Model Öngörülü Kontrol (MPC) yapısı geniş bir tarihi vardır. Endüstriyel uygulamalarla ilgili ilk çalışmalardan biri Richatel'e aittir [6]. İlk yöntemler darbe ya da basamak tepkisi modellerine dayanırken sonraki yöntemlerde daha etkili modeller esas alınmıştır.

MPC özel bir kontrol şekli değildir; fakat kontrol metotlarının birçok alanından daha fazla alanda belirli bir genel fikir etrafında geliştirilmiştir. Bu tasarım yöntemleri uygulamalı olarak aynı yapıya sahip doğrusal denetleyicilere yol gösterir ve yeterli derecede serbestlik sunar. Çeşitli MPC algoritmaları, maliyet fonksiyonu, gürültü ve sürecin gösterilmesinde kullanılan modele göre aralarında küçük farklılıklar gösterirler. Model Öngörülü Kontrol, sistemin gelecekteki davranışını optimize etmek için uygulanması gereken kontrol dizisini hesaplayan bir kontrolör sınıfını nitelemektedir. Modele Dayalı Öngörülü Kontrol yöntemleri sadece, kullanılan sistem modeli, gürültü modelleri ve minimize edilecek olan ölçütler bakımından farklılık gösterirler. İlk olarak güç reaktörlerindeki ve petrol rafinerilerindeki kontrol gereksinimlerine bağlı olarak özel geliştirilmiş olan bu yöntemler, günümüzde kimya, gıda, otomotiv, havacılık, metalürji ve kağıt endüstrilerini kapsayan geniş bir alana da uygulanmaktadır. Şu anda başarıyla

kullanılan birçok MPC denetimi vardır. Yalnızca süreç endüstrisinde değil, aynı zamanda robot manipülatörlerinden, klinik anesteziye kadar birçok farklı alanlarda uygulamalarına rastlanmaktadır. Bu yöntemlerin ortak özelliği sistem modelinin doğrudan kullanılması ve kontrol işaretinin belirli bir ölçüte göre minimumlaştırarak elde edilmesidir. Bu tasarım yöntemleri ile elde edilen kontrolörler lineer yapıdadır [11,12].

II.2.1 MPC Algoritmasının Temel Adımları

N ufku ile tanımlı gelecek çıkışlar (öngörülü ufuk diye adlandırılır) süreç modeli kullanılarak her t anında önceden bulunur. Bu öngörülü çıkışlar $k = 1 \ldots\ldots\ldots N$ için $y(t + t / k)$, t'den hemen önceki bilinen değerlere, hesaplanmış ve sisteme gönderilmiş olan $u(t + t / k)$, $k = 0 \ldots\ldots N - 1$ gelecek kontrol sinyallerine bağlıdır.

Geleceğe ilişkin kontrol işareti dizisi bir başarını ölçütünü minimumlaştırarak, genellikle öngörülen sistem çıkışı ile referans yörüngesi $r(t + k)$ arasındaki hatayı minimumlaştırarak hesaplanır. Kontrolün amacı pek çok durumda nesnel fonksiyonda yer alır. Model doğrusal ise ve sınırlama yoksa ve ölçüt karesel hatadan oluşursa, kesin bir çözüm elde edilir. Bu durumun aksinde bir tekrarlanmış fonksiyon yöntemi kullanılmalıdır.

Kontrol sinyali u(t/t), hesaplanan gelecek kontrol sinyalleri alınmadığı halde, sürece gönderilir; çünkü gelecek örnekleme anında $y(t + 1)$ zaten bilinmektedir ve 1.adım bu yeni değerle tekrarlanır ve tüm diziler yenilenerek alınır. Böylece $u(t + 1/t + 1)$ kayan ufuk stratejisi kullanılarak hesaplanabilir (prensipte mevcut yeni bilgiden dolayı $u(t + 1/t)$'den farklı olacaktır). Eğer model lineer, ölçüt karesel ise ve sınırlandırmalar yoksa analitik bir çözüm bulunabilir, aksi durumlarda diğer optimizasyon yöntemlerini kullanmak gerekir [2,4,12].

Genel MPC yapısı şekil III.2 de verilmiştir. Burada yr referans yörüngeyi, ym model çıkışını, u kontrol işaretini, y süreç çıkışını, h model çıkışı ile süreç çıkışı arasındaki hatayı ve d ise süreçteki gürültüyü temsil etmektedir. Tüm MPC algoritmalarının sahip olduğu genel kavramlar vardır. Bunlar, öngörü modeli, maliyet fonksiyonu ve kontrol kuralını elde edilmesidir.

Kullanılan model ve yöntem açısından değişik Model Öngörülü Kontrol teknikleri vardır. Bunlara örnek olarak

• Dinamik Matris Kontrol (Dynamic Matrix Control: DMC)

• Genişletilmiş Öngörülü Özuyarlamalı Kontrol (Extended Prediction Self Adaptive Control EPSAC)

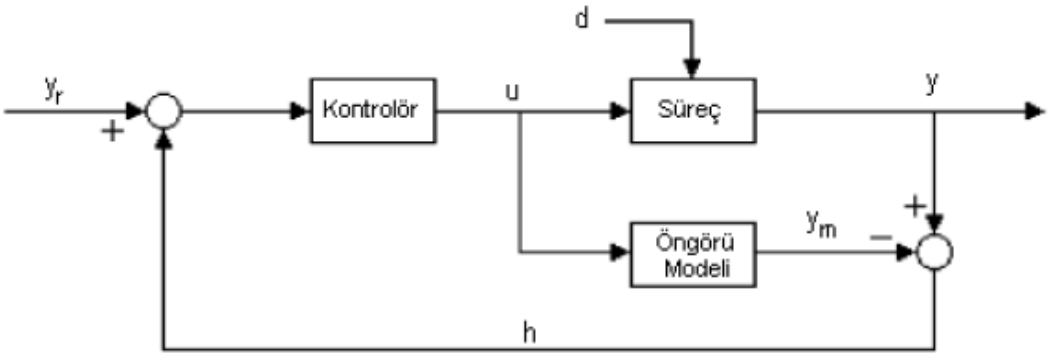
• Genelleştirilmiş Öngörülü Kontrol (Generalised Predictive Control:GPC)

• Model Algoritmik Kontrol (Model Algorithmic Control: MAC)

• Öngörülü Fonksiyon Kontrol (Predictive Functional Control:PFC)

• Ardışık Açık Çevrim Optimizasyon Kontrol (Sequebtial Open Loop Optimization:SOLO)

verilebilir. Öngörülü kontrol algoritmaları literatürde genel olarak, Model Öngörülü Kontrol MPC veya Modele Dayalı Öngörülü Kontrol MBPC isimleri kullanılır.



Şekil II.2 Model Öngörülü Kontrol Genel Yapısı

Literatürde birçok Model Öngörülü Kontrol algoritması bulunur. Farklı algoritmalarda kullanılan çok sayıda faklı modeller vardır. Bu modellerden en çok kullanılanı basamak yanıtı modelidir. Bu modelin uygulanması kolaydır, çünkü sisteme basamak girişi uygulandığında sistem çıkışını ölçmek model parametrelerini bulmak için yeterlidir. Bu model çok değişkenli sistemler için de kullanılabilir olduğundan endüstride yaygın olarak kullanılmaktadır. MPC yöntemlerinin geliştirilmesinde optimizasyon işlemi, önemli bir konudur. Eğer ölçüt karesel ise, tam çözümü geçmiş giriş ve çıkışlara ve gelecekteki girişlere bağlı olarak lineer bir fonksiyonla ifade edilebilir. Kısıtlamaların olması durumunda çözüm ancak daha çok işlem gerektiren nümerik bir algoritma kullanılarak elde edilebilir. Optimizasyon problemi değişken sayısına ve ufuk uzunluğuna bağlıdır [12]. MPC'nin diğer kontrolörlere göre bazı üstünlükleri vardır.

• Kavramlar sezgisel olduğu ve aynı zamanda parametre ayarları diğer yöntemlere göre kolay olduğu için kontrol bilgisi sınırlı olan çalışanlara çekici gelmektedir.

• Çok basit dinamiğe sahip süreçlerden, kararsız, minimum fazlı olmayan ya da çok uzun ölü zamanı bulunan süreçler gibi daha karmaşık dinamiklere sahip çok çeşitli süreçlerin kontrollünde kullanılabilir.

• Çok değişkenli sistemlerin kontrolünde rahatlıkla kullanılabilir.

• Yapısı gereği ölü zaman etkisini giderme özelliğine sahiptir.

• Sonuçta uygulaması kolay lineer bir kontrol kuralı elde edilir.

• Sınırlandırmalarla başa çıkabilecek, sınırlandırmaları tasarım sürecine sistematik olarak katabilen geliştirilmiş biçimleri vardır.

• Geleceğe ilişkin çıkış işareti hesaplanabilindiğinde oldukça yararlı ve kullanışlıdır.

• Belirli temel ilkeler üzerine kurulmuş geliştirilmeye tamamen açık bir metodolojidir.

Bu üstünlüklerine karşın Modele Dayalı Öngörülü Kontrolün eksik yönleri de vardır.

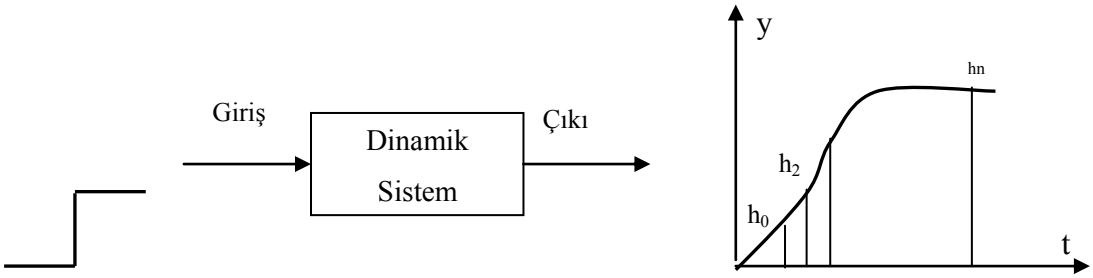
• Sonuçta uygulaması kolay, işlem karmaşıklığı düşük olan bir kontrol kuralı elde edilse de, bu kontrol kuralının elde edilmesi PID kontrolörlerinden daha karmaşıktır. Sistem dinamiğinin değişmediği durumlarda kontrolörün elde edilme işlemi önceden yapılabilir, ancak uyarlamak kontrol yapılıyorsa tüm hesap işlemleri her örnekleme zamanında tekrarlanır. Sınırlandırmalar göz önüne alındığında işlem karmaşıklığı daha da artacaktır.

• En önemli sorun ise sisteme ilişkin uygun modelin elde edilmesidir. Algoritma sistem modelinden yararlanılarak geleceğe ilişkin sistem yanıtının elde edilmesine dayanmaktadır, ancak gerçek sistemle yanıt arasındaki farklılık artıkça istenilen sistem yanıtını elde etmekte sorunlar yaşanır. Model öngörülü kontrol süreç modelleri aşağıdaki gibidir [13].

II.2.1.1 Basamak Cevabı Modeli

Yaygın olarak kullanılan modeldir. Açık çevrim basamak cevabı şekil II.3'te verilmiştir. Bu modelin uygulanması kolaydır, çünkü sisteme basamak girişi uygulandığında sistem çıkışını ölçmek model parametrelerini bulmak için yeterlidir.

Bu model çok değişkenli sistemler için de kullanılabilir olduğundan endüstride yaygın olarak kullanılmaktadır.



Şekil II.3 Birim Basamak Cevabı

Birim basamak cevabının Δ_t süresince değerleri h_1, h_2.....h_T ile tanımlanır. Δ_t sistemin oturma zamanından alınabilir. N model ufkudur. Basamak cevabı girişteki değişim Δ_u h'dan alınır. Çıkışın öngörü değeri ŷ ve n. örnekleme zamanındaki u_n giriş değişkeni olarak belirlenir. Ayrıca y_m gerçek çıkış değeri olarak tanımlanır. Sistemde hata ve bozucu olmadığı durumda $\hat{y}= y_m$ olacaktır. $\Delta_u=u_i-u_{i-1}$ ve y_0 çıkış değerinin başlangıç değeri olacak şekilde öngörü modeli;

$$y_{h+1} = y_0 + \sum_{i=1}^{T} h_i \Delta u_{n+1-i} \qquad \text{(II.5)}$$

şeklinde hesaplanır. Sistem çıkışına ilişkin ilk N değer göz önüne alınmış, sonsuz toplam yapılmamıştır. Bu nedenle bu model integratör içermeyen ve kararlı lineer sistemler için uygundur.

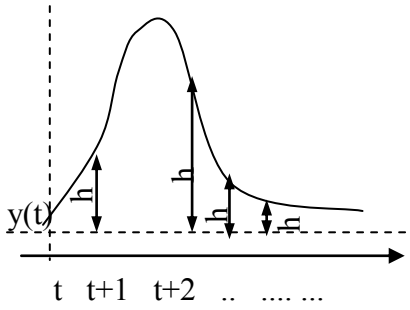
II.2.1.2 Darbe Cevabı Modeli

Bu model basamak modeline benzemektedir. Modelde giriş çıkış ilişkisi;

$$y(t) = \sum_{i}^{N} h_i\, u\ t-i\ = H\ z^{-1}\ u(t) \qquad \text{(II.6)}$$

şeklinde ifade edilir. h_i'ler sisteme darbe girişi uygulandığında elde edilen çıkışın örneklenmiş değerleridir(Şekil II.4). Bu toplam, sadece N değerine kadar yapılmıştır. Burada $H(z-1) = h_1 z-1 + h_2 z-2+\ldots\ldots+h_N z-N$ ile tanımlanır. Bu model kullanılarak elde edilen öngörü ifadesi denklem II.7 de verilmiştir.

$$y(t+k/t) = \sum_{i=1}^{T} h_i\, u\ t+k-i\ = H\ z^{-1}\ u(t+k/t) \qquad \text{(II.7)}$$



Şekil II.4 Darbe Cevabı Modeli

II.2.1.3 Transfer Fonksiyonu Modeli

Bu model, parametre sayısı az olduğu için ve her türlü lineer sistem için uygundur. Bu nedenle birçok uygulamada kullanılmaktadır. Giriş u(t) ve çıkış y(t) olmak üzere;

$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} + \ldots\ldots\ldots + a_{na} z^{-na} \quad \text{(II.8a)}$$

$$B(z^{-1}) = b_0 + b_1 z^{-1} + b_2 z^{-2} + \ldots\ldots\ldots + b_{nb} z^{-nb} \quad \text{(II.8b)}$$

$$A(z^{-1}) y(t) = B(z^{-1}) u(t-1) \quad \text{(II.8c)}$$

model eşitlikleri verilebilir. Bu model kullanılarak elde edilen öngörü ifadesi;

$$y\ \ t + \frac{k}{t}\ \ = \frac{B\ z^{-1}}{A\ z^{-1}} u\ \ t + \frac{k}{t} \quad \text{(II.9)}$$

şeklindedir.

II.2.1.4 Durum Uzayı Modeli

Çok değişkenli sistemlerin kolaylıkla tanımlanabildiği bir modeldir. Bu model en çok Öngörülü Fonksiyonel Kontrol (PFC) algoritmasında kullanılır. Durum uzay modelini formu aşağıda verilmiştir.

$$x(k+1) = Ax(k) + Bu(k) \quad \text{(II.10)}$$

$$y\ k\ \ = C_y x(k) \quad \text{(II.11)}$$

$$z\ k\ \ = C_z x(k) \quad \text{(II.12)}$$

Burada x, n boyutlu durum vektörü, u, l boyutlu giriş vektörü, y, ölçülen çıkışların my boyutlu vektörü ve z kontrol edilecek olan çıkışların mz boyutlu vektörüdür. y ve z'deki değişkenler genellikle geniş bir kapsam ile örtüşmektedir ve bunlar sıklıkla aynı olur. Bu da tüm kontrollü çıkışlar sıklıkla ölçülmesi demektir.

Çoğunlukla y, z olarak kabul edilir ve C'yi hem Cy hem de Cz, m'yi de hem my hem de mz olarak kullanılır. k indeksi ise zaman adımlarını göstermektedir.

Bu formatta bir standart kullanılmasının sebebi lineer sistemlerin ve kontrolün standart teorisi ile paralel olmasını sağlamaktır. Bu modeli ölçülen veya ölçülmeyen hataların ve ölçme gürültüsünün etkilerini dahil etmek için genelleştirilebilir.

Davranışların k zaman adımlarındaki sırası şu şekilde kabul edilir.

1.Ölçme değerleri alınır y(k)

2.Gerekli sistem girişleri hesaplanır u(k)

3.u(k) sisteme uygulanır

Ölçülen y(k) ve sisteme uygulanan u(k) arasında daima bir gecikme söz konusudur. Dolayısıyla ölçülen çıkış eşitliğinde, u(k)'dan y(k)'ya asla bir "doğrudan besleme" olmayacaktır. Bu sebeple model "tam anlamıyla uygun" denilebilir.

Kontrol edilen çıkışlar z(k) prensip olarak u(k)'ya bağlı olabilir.

$$z\ k\ = C_z x\ k\ + D_z u(k) \quad \text{(II.13)}$$

Burada D_z=0 almak bazı durumlarda uygun olabilir. Diğer taraftan optimal u(k)'nın hesaplanmasını az da olsa karmaşıklaştırabilir. Bu karışıklıktan hiçbir şeyi kaybetmeden kontrol çıkışlarına yeni bir vektör tanımlayarak kurtulabilinir.

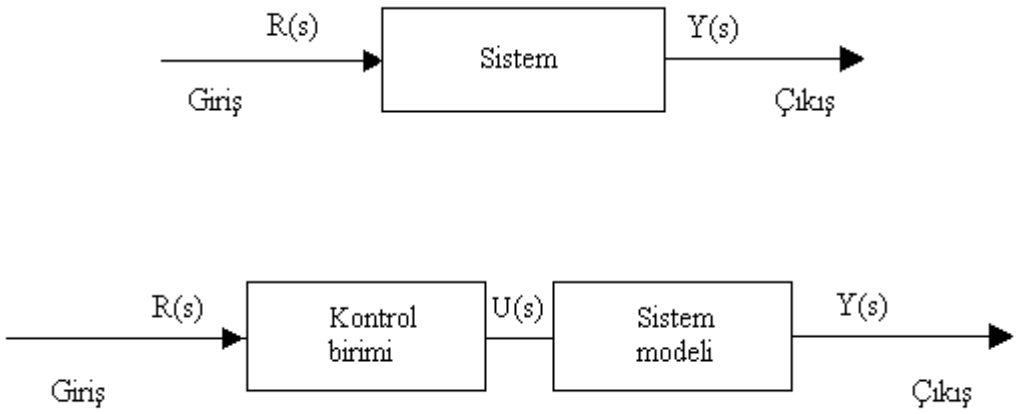
$$z\ k\ = z\ k\ + D_z u(k) \quad \text{(II.14)}$$

Bu açık olarak sadece x(k)'ya u(k):ž(k)=Czx(k)'dan direk besleme olmadan bağlıdır. Neticesinde maliyet fonksiyonunda ve kısıtlardaki değişiklikler kolayca yapılmaktadır [4].

Kontrol kuralı, sistem durumlarının bazen fiziksel anlamları taşımamasına rağmen durum vektörlerinin doğrusal kombinasyonun geri beslemesidir. Bu modellerin yanı sıra son yıllarda yapay sinir ağları(Neural Net) ve bulanık mantık (Fuzzy Logic) gibi modellerde kullanılmaktadır [12].

II.3 TEK GİRİŞLİ TEK ÇIKIŞLI SİSTEM TANIMI

Tek bir çıkışın tek bir değişken tarafından kontrol edildiği sistemler tek giriş tek çıkış sistemler (Single Input Single Output-SISO) olarak sınıflandırılırlar. Bu sistemler basit kontrol sistemlerinde kullanılırlar. Örneğin ısıtma, havalandırma sistemlerinin kontrolü, hareket kontrolleri gibi sitemlerin kontrolleri genellikle tek giriş tek çıkış kontrol sistemleriyle gerçekleşir. Kontrol edilen sistemden kasıt ise bütünü oluşturan, birbirleri ile bağlı olan ya da belli bir işlev için bir araya getirilmiş olan elemanlar düzenine ya da kümesine denir. Sistemi oluşturan bu elemanları



Şekil II.5 Tek Girişli Tek Çıkışlı Bir Sistemin Genel Yapısı

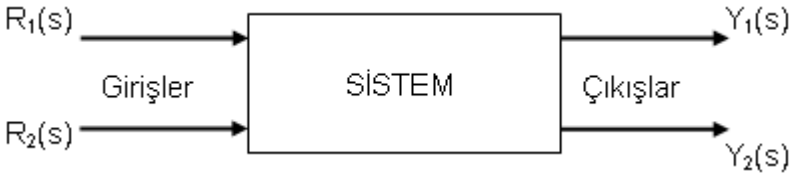
kontrol etmek için ise elemanlarının birbiri ile bağlantılarını bir bütün şeklinde değerlendirilir, sistemin kontrolü için gerekli olan model hesaplanır ve böylece tek bir girişin kontrol edebileceği tek bir model ve bu modelin kontrol birimi ile etkileşimi sonucunda istenilen yönde tek bir çıkış sağlanır.

II.4 ÇOK GİRİŞLİ ÇOK ÇIKIŞLI SİSTEM TANIMI

Birden fazla çevrime sahip sistemlere çok girişli çok çıkışlı sistemler(Multi Input Multi Output-MIMO) denir [14]. Çok girişli çok çıkışlı sistemler tek giriş tek çıkışlı sistemlerden farklıdır. Örneğin herhangi bir ürünün imalat veya rafine işlemlerini yapan bir birimin kontrolü tek kontrollü bir çevrim ile yapılamaz. Hatta her bir birim işlemi en az iki değişken üzerinde kontrol gerektirir. Bu yüzden genellikle en az 2 tane çevrimle baş etmek gerekir.

Çok girişli çok çıkışlı sistemlerde herhangi bir çıkış, diğer girişten ya da herhangi bir giriş, diğer bir çıkıştan etkilenmektedir. Bu etkileşimin hangi düzeyde olduğunu bilinmesi gerekir ancak bu şekilde kontrol tasarımın başarıya ulaşması sağlanır.

Çok değişkenli sistem analizi için örnek olarak 2 Giriş,2 Çıkışlı (2X2 Two Input-Two Output -TITO) süreç örnek olarak verilmiştir (Şekil II.6). Çok değişkenli sistemlerin ve sinyallerin normları ile kazancın, band genişliğinin ve diğer sistem özelliklerinin elde edilmesi mümkündür.



Şekil II.6 MIMO Sistem Genel Yapısı (2X2)

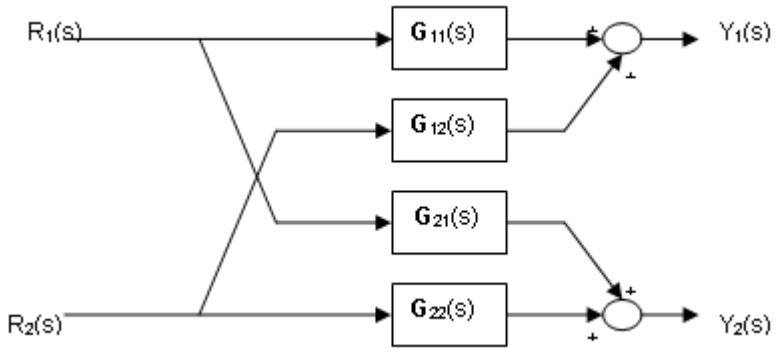
Transfer fonksiyon ilişkilerini kullanarak bu sistem değişkenlerini,

$$Yi(s) = Gii(s)Ri(s) + Gij(s)Rj(s) \quad \text{(II.15)}$$

$$Y1(s) = G11(s)R1(s) + G12(s)R2(s) \quad \text{(II.15a)}$$

$$Y2(s) = G21(s)R1(s) + G22(s)R2(s) \quad \text{(II.15b)}$$

denklemleri ile tanımlayabiliriz. Burada $G_{ij}(s)$ fonksiyonu i. çıkış değişkeni ile j. giriş değişkeni arasındaki ilişkiyi içeren transfer fonksiyonudur. Bu denklem setinin blok diyagram ile gösterimi de şekil II. 7'de görülmektedir.



Şekil II.7 MIMO Sistem Genel Yapısı

Bu sistemi tamamlayan transfer fonksiyonunu kısaca; Y=GR şeklinde de ifade edebiliriz ki burada; Y ve R matrisleri I çıkış ve J giriş değişkenlerini ifade eden sütun matrislerdir, G ise I den J ye transfer fonksiyonunu ifade eder. Matris gösterimi birçok değişken içeren karmaşık çok değişkenli sistemlerin iç ilişkilerin gösterilmesinde kullanılır.

II.4.1 Çok Girişli Çok Çıkışlı Sistemlerin Durum Uzayı Modeli

Çok değişkenli sistemlerin dinamik davranışları sunulmasında iki form vardır. Bunlar, Durum Uzayı ve Transfer Fonksiyonu formlarıdır. Her iki formda da modeller çok değişkenli doğası nedeniyle vektör matrisi şeklinde yazılır [15].

II.4.1 Durum Uzayı Modeli

Denklem II.16 ve II.17 da durum uzayı modelini genel gösterimi bulunmaktadır.

$$x = A\ x\ + B\ u\ + \Gamma d(t) \quad \text{(II.16)}$$

$$y\ t\ = Cx(t) \quad \text{(II.17)}$$

burada;

x= Durum değişkenlerinin ℓ-boyutlu vektörü

u= Giriş(kontrol) değişkenlerinin m-boyutlu vektörü

y=Çıkış değişkenlerinin n-boyutlu vektörü

d= Bozucu değişkenlerinin k-boyutlu vektörü

A,B,C ve Γ ise çarpım işlemi gerçekleştirilen sistem matrislerinin vektörleridir.

Doğrusal olmayan çok değişkenli sistemler genelde denklem II.18 de verilen diferansiyel denklemlerle ifade edilir.

$$\frac{dx(t)}{dt} = f(x,u,d),\ y(t) = h(x(t)) \quad \text{(II.18)}$$

burada f(x,u,d) ve h(x(t)) genelde doğrusal olmayan fonksiyonlarının vektörleridir.

II.4.2 Transfer Fonksiyonu Modeli

SISO sistemlere benzer olarak çok değişkenli sistemlerde de transfer fonksiyonu, giriş ve çıkış değişkenleri arasındaki ilişki veren Laplace transformasyon ile tanımlanır.

$$Y(s) = G(s)U(s) + G_d(s)D(s) \quad \text{(II.19)}$$

burada m-giriş ve n-çıkışlı bir sistem için m x n boyutundaki G(s) transfer fonksiyon matrisi;

$$G(s) = \begin{matrix} g_{11}(s) & g_{12}(s) & \cdots & g_{1m}(s) \\ g_{21}(s) & \cdots & \cdots & g_{2m}(s) \\ \cdots & \cdots & \cdots & \cdots \\ g_{n1}(s) & g_{n2}(s) & \cdots & g_{mm}(s) \end{matrix} \quad \text{(II.20)}$$

Her bir g_{ij} indisi SISO transfer fonksiyonu özelliği taşır. G(d) n x k boyutunda benzer transfer fonksiyonlarından oluşur.

SISO sistemlerdeki gibi durum uzayı modelinden transfer fonksiyona ulaşılması mümkündür. Denklem II.15a ve II.15b'deki eşitliklerin Laplace dönüşümleri alınır, X(s) durum değişkenleri vektörünün Laplace dönüşümleri elde edilir.

$$Sx(s) = Ax(s) + Bu(s) + \Gamma d(s) \quad \text{(II.21)}$$

$$y(s) = Cx(s) \quad \text{(II.22)}$$

gerekli düzenlemeler matris matematiğine göre yapılırsa,

$$Y\ s\ =\ C\ sI - A\ ^{-1}b\ u\ s\ +\ C\ sl - A\ ^{-1}\Gamma\ \mathrm{d}\ \mathrm{s} \quad \text{(II.23)}$$

elde edilir. Bundan sonra Denklem II.19'la karşılaştırılarak durum uzayı denklemi ile transfer fonksiyonu arasındaki ilişki tanımlayan

$$G(s) = [C\ sI - A\ ^{-1}B] \quad \text{(II.24)}$$

$$G_d(s) = [C\ sI - A\ ^{-1}\Gamma] \quad \text{(II.25)}$$

eşitlikleri elde edilir [13].

II.5 KISITLAMALI ÖNGÖRÜLÜ KONTROL

Çok değişkenli sistemlerin kontrolünde çeşitli kısıtlamaların olduğu bir gerçektir. Bu kısıtlamalar hem sistem girişlerinde hem de sistem çıkışlarında olabilir. Sistem girişlerindeki kısıtlamaların sebebi eyleyiciler, çıkışlardakilerin sebebi ise genelde kalite ve güvenliktir. Diğer kontrol yöntemlerinde tasarım parametrelerinin ayarlanması ile bu kısıtlamalar tanımlanabilir. Ancak bu durumda sistemin kontrolünde sorunlar yaşanabilir. Kontrolün başarıma ulaşması için problemin kısıtlamaları içerecek şekilde formüle edilerek çözülmesi gerekir. Öngörülü kontrol yöntemindeki formülasyon yapısı kısıtlamaların kolaylıkla eklenmesine olanak vermektedir. Son yıllarda yapılan çalışmalarda kısıtlamalarında dikkate alınarak ve tam olarak sağlandığı görülmektedir [16].

Öngörülü kontrolünün problemleri çoğunlukla kısıtlamaları olduğu için elde edilen kontrol kuralı da genellikle doğrusal değildir. Standart bir öngörülü kontrolörün aslında sistem kısıtlamalardan uzak oldukça nasıl güvenli çalışıyorsa benzer şekilde de uzun süre doğrusal, kısıtlamalara yaklaşıldıkça da doğrusal değildir [2].

Giriş üzerinden yapılan kısıtlamalarda denklem II.26'daki, çıkış üzerinde yapılan kısıtlamalarda ise denklem II.27'deki şekilde gerçekleşir.

$$u_{min}(l) \leq u(k+l) \leq u_{max}(l) \quad \text{(II.26)}$$

$$y_{min}(l) \leq y(k+l/k) \leq y_{max}(l) \quad \text{(II.27)}$$

II.6 SİSTEMLERİN MODELLENMESİ VE SİSTEM TİPLERİ

Mekaniksel, elektriksel, ısıl, akışkan, ekonomik, biyolojik vb. pek çok dinamik sistem matematik modeli diferansiyel denklemler yoluyla karakterize edilebilir. Bir dinamik sistemin bir giriş fonksiyonuna karşı göstereceği tepki veya cevap ise bu diferansiyel denklemin çözümünden elde edilir.

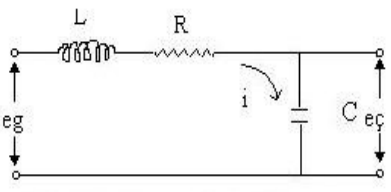
Modeller gerçeğin sadeleştirilmiş durumlarıdır. Bir model, uçak biçim belki de renk olarak gerçek uçağa benzemeyebilir, fakat boyu veya yapısal karmaşıklık açısından farklıdır. Mühendislik alanında gerçek sistemlerle deney yapmanın pratik olmadığı durumlarda fiziksel modeller ya da fiziksel modeli soyut bir şekilde temsile edebilecek matematiksel modeller mevcuttur. Bir sistemin dinamik karakteristiğinin matematiksel tanımına matematiksel model denir. Bir sistemin çözümlenmesinde ilk adım onun modelini çıkartmaktır. Bu çözümleme işlemi sürecin en önemli kısmıdır. Sisteme uygun matematiksel model elde edildikten sonra analitik veya sayısal çözüm teknikeri ile aranan çözümü bulmak mümkündür.

Sistemi oluşturan bileşenler incelendiğinde ve bunların her birinin sisteme etkisi, bir birleri ile enerji dönüşümleri gibi işlemleri hesaplanması karmaşık ve zor bir süreçtir. Sistemin modelini hesaplamada sistemi oluşturan elemanları bölüm bölüm ayırmak ve bunlar arasındaki ilişki incelemek bu süreci kolaylaştıracaktır.

Hesaplanan sistem denklemleri bize o sisteme uygulanabilecek olan ani darbe giriş fonksiyonu cevabının lablace dönüşümüdür. Bu dönüşümde bize transfer fonksiyonun yani sistemin modelinin matematiksel bir şekilde eşdeğerini verir. Bu eş değerde doğrusal şekilde sistemin çıkışının girişine oranıyla ifade edilir.

II.6.1 Elektriksel Sistemler

Basit bir elektriksel sistem direnç, kapasite ve indüktans elemanlarından oluşur. Bu elemanlar değişik şekillerde birbirine bağlanması ile farklı elektrik devreleri oluşturulur.



Şekil II.8 Elektrik Devresi

Ardışık bağlı L-R-C devresi şekilde gözüktüğü gibidir. Uygulanan eg giriş gerilimine karşı i akımı ve eç çıkış gerilimi oluşmaktadır. Böyle bir sistemin matematiksel modelini çıkarmak için bazı hesaplamalar yapılır. Bu hesaplamalar:

$e_g = e_l + e_r + e_c$ veya $e_g = L\frac{di}{dt} + Ri + \frac{1}{C}\int_0^{\infty} idt$ (II.28)

$e_{ç} = \frac{1}{C}\int idt$ şeklinde ifade edilir. (II.29)

Bu denklemler sistemin dinamik davranışlarını tanımlayan diferansiyel denklem takımıdır. Bu denklemlerinde laplace dönüşümü alındıktan sonra çıkışı girişe oranlayarak sistemin transfer fonksiyonu elde edilir.

$e_g = LC\frac{d^2 e_{ç}}{dt^2} + RC\frac{de_{ç}}{dt} + e_{ç}$ (II.30)

$\frac{E_g(s)}{E_{ç}(s)} = \frac{1}{LCs^2+RCs+1}$ (II.31)

II.6.2 Mekanik Sitemler

Öteleme hareketi yapan kütle, yay, sönümleme elemanları yada dönme hareketi yapan eylemsizlik moment, burulma yayı ve dönel sönümleme elemanlarından meydana gelir. Ötelemede elemanların çeşitli kuvvetlerin etkisi sonucunda titreşim hareketi, dönmede ise dönme hareketinin çeşitli servo mekanizmalarının giriş, açısal konumu hızı gibi değişimlerin inceleme ve hesaplamaları yapılır.

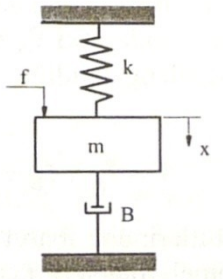
II.6.2.1 Öteleme Yapan Sistemler

$f_{net} = ma$ (II.32)

$F\ t\ = m\frac{d^2X}{d^2} + B\frac{dX}{dt} + kX - mg$ (II.33)

Kütle m, ivme a ile kütlenin yer değiştirme mesafesi x şeklinde, yayın katsayısı k ile gösterilmiştir. B ise sürtünme etkisinin oluşturduğu zorluk. Sistemin transfer fonksiyonun çıkarmak için gerekli düzenlemeler ve laplace dönüşümü alındığında sistemin kuvvet karşısında titreşimini gösteren transfer fonksiyonu elde edilir.

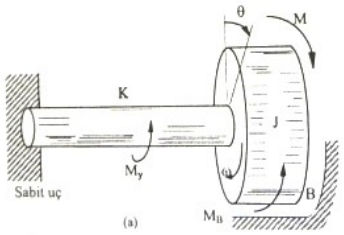
$\frac{X(s)}{F(s)} = \frac{1}{ms^2+Bs+k}$ (II.34)



Şekil II.9 Kütle-Yay-Sönümleyici Sistem

II.6.2.2 Dönme Yapan Sistemler

j eylemsizlik kütlesi, k burulma yayı, B burulma sönümleme elemanını θ açısal yer değiştirme, M'de dönme momentini ifade etmektedir. Sistemin transfer fonksiyonu için yine sistemin laplace dönüşümü alındığında transfer fonksiyonu elde edilir.



Şekil II.10 Dönel Kütle-Yay- Sönümleme Sistemi

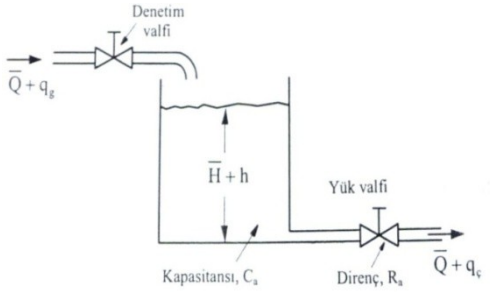
$$M\ t\ = j\frac{d^2\theta}{d^2} + B\frac{d\theta}{dt} + k\theta \tag{II.35}$$

$$\frac{X(s)}{F(s)} = \frac{1}{js^2+Bs+k} \tag{II.36}$$

II.6.3 Sıvı Seviye Sistemleri

Seviye kontrol sistemleri genelde sıvı seviyesini kontrol etmek amacıyla kullanılmaktadır. Sıvı seviye sistemleri de üstü açık bir tank ve bu tanka giren bir akışkan debisi ile tankı terk eden akışkan debisi farkına bağlı olarak sıvı seviyesine bağlı değişim ile modellenir.

$$q_g - q_1 = C_1\frac{dh_1}{dt} \tag{II.37}$$



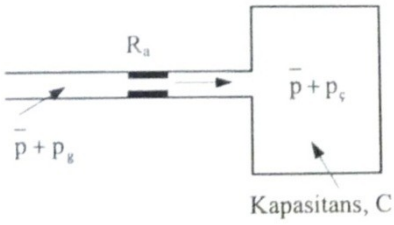
Şekil II.11 Sıvı Seviye Sistemi

Q sabit akış debisi, H sıvı seviyesi, q debide meydana gelen sapma, h sıvı seviyesinde meydana gelen sapma. Sistemin transfer fonksiyonu için yine sistemin laplace dönüşümü alındığında transfer fonksiyonu elde edilir.

$$\frac{H(s)}{Q_g(s)} = \frac{R_a}{R_a C_a s+1} \tag{II.38}$$

II.6.4 Basınç Sistemleri

Gaz akışlarının yer aldığı kapalı tanklarda basınç değişimi ve basınç denetimi esastır. Gaz akışkan sistemlerinde gazın sıkıştırılabildiğinin hesaba katılması gerekir. Bu durumda sistemin akışkan kapasitansı gaz kanunlarına göre elde edilir.



Şekil II.12 Basınç Tank Sistemi

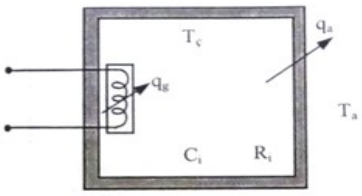
$$q = \frac{p_g - p_ç}{R_a} \text{ ve } q = C_a \frac{dp_ç}{dt} \tag{II.39}$$

Sistemin diferansiyel denkleminin laplace dönüşümü alındığında transfer fonksiyonu aşağıdaki gibi olur.

$$\frac{P_ç(s)}{P_g(s)} = \frac{1}{R_a C_a s+1} \tag{II.40}$$

II.6.5 Isıl Sistemler

Basit bir ısıl sistem bir direnç ve bir kapasitans elemanlarından ibaret modellenebilir. Isıl sistemlerde enerjinin indüktif etki ile depolanması yoktur ve bu nedenle ısıl indüktans elemanından söz edilemez. Sistem incelendiğinde q_g giren ısıl güç, qa çıkan ısıl güç, R_i ortamın ısıl direnci, Ci ortamın ısıl kapasitesi, $T_ç$ iç ortamda meydana gelen sıcaklık değişimi, T_a dış ortamda meydana gelen sıcaklık değişimi. Sistemi oluşturan her elemanın diferansiyel denklemleri çıkarıldığında



Şekil II.13 Isıl Sistem

$$R_i q_g = -\ T_ç - T_a\ = R_i C_i \frac{dT_ç}{dt} \tag{II.41}$$

denkleminin laplace dönüşümü alındığında

$$T_ç\ s\ = \frac{R_i}{T_i s+1} Q_g\ s\ + \frac{1}{T_i s+1} T_a(s) \tag{II.42}$$

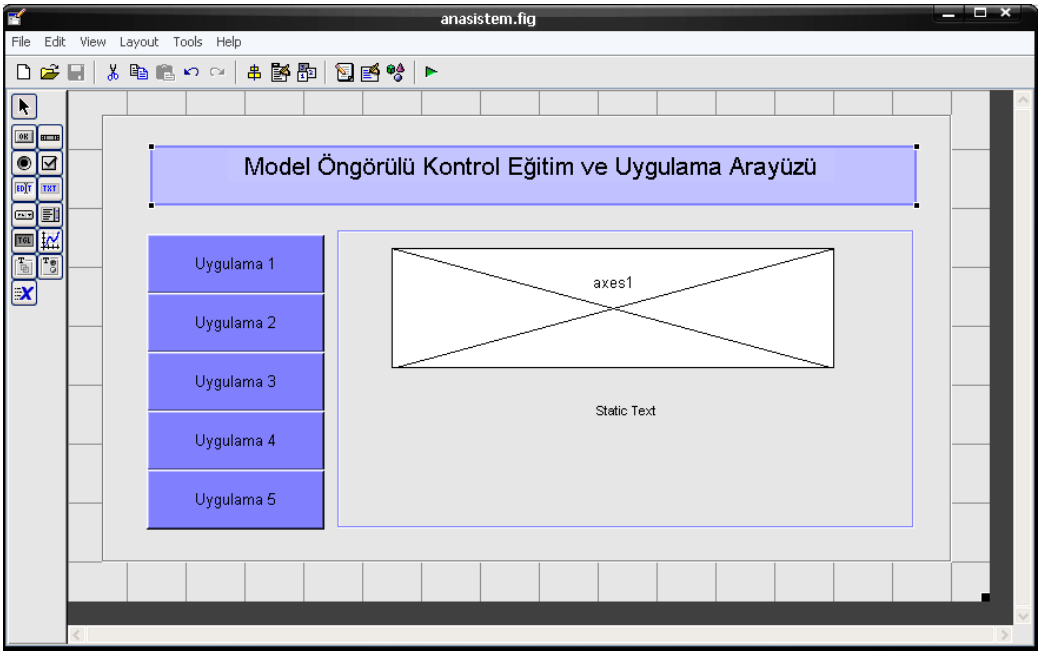
sistemin transfer fonksiyonu elde edilir.

Yukarıda açıklanan bu sistemler en yaygın olarak kullanılan sistemlerdir. Bunların dışında elektromekanik sistemler, akışkan güç iletim sistemleri, elektrohidrolik, elektropnömatik sistemler gibi sistemlerde bulunmaktadır [17].

II.7 MATLAB GRAFİKSEL KULLANICI ARA YÜZÜ (GUI)

İçeriğinde yer alan nesnelerin kullanılması ile kullanıcıya etkileşim sağlayan ve bir işin veya bir programın koşturulmasını sağlayan grafiksel bir program arayüzüdür. Açılımı Graphical User Interface (GUI) dir.

GUI nesneleri menüler, araç çubukları, radio butonlar, liste kutuları veya kaydırıcılar olabilir. Bunların yanında MATLAB GUI ile MATLAB'in sunduğu hesaplama imkânları kullanılarak da veri alımı ve grafik çizimi gibi pek çok işlem gerçekleştirilebilir.



Şekil II.14 Örnek Bir GUI Arayüzü

II.7.1 Grafiksel Kullanıcı Arabiriminin(GUI) Çalışıma Yapısı

Her bir nesne GUI için tanımlanan programlama dosyasında callback diye adlandırılan ayrı alt rutin programlama parçalarına sahiptir. Bu şekilde her bir nesnede oluşan olaylara (örnek olarak bir buton nesnesinin tıklanması ile click event oluşması gibi) GUI o olaya ait callback rutinlerini icra ettirir. Yani, GUI hem bir arayüz hem de bir program çağrılarını icra ettirme mekanizması olarak çalışır.

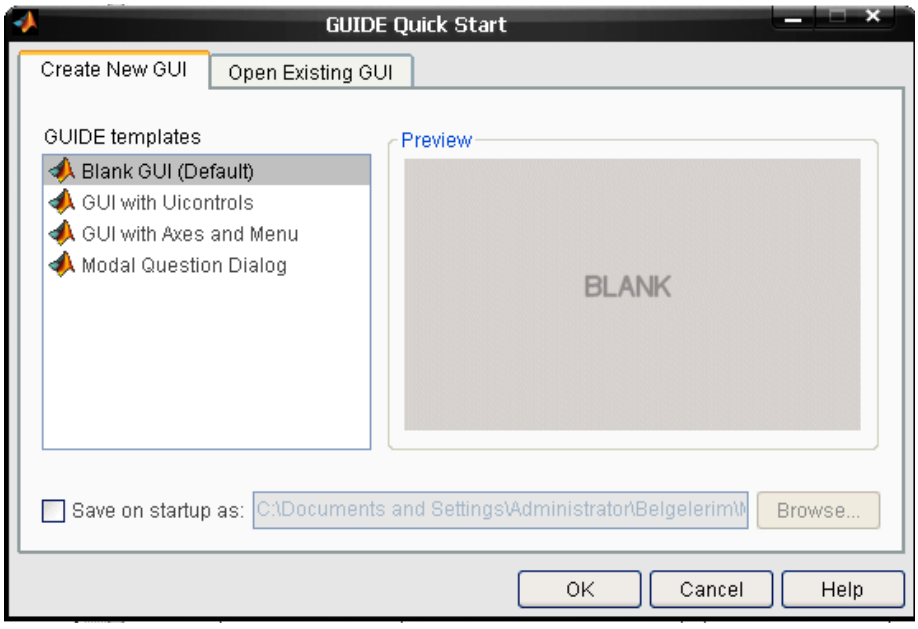
II.7.2 MATLAB'te GUI Oluşturma Yöntemleri

Özellikle GUI tasarımında hızlı arayüzler dizayn etmek ve bu işe ilk başlayan programcılar için MATLAB GUIDE aracının kullanılması büyük bir kolaylık sağlar. Bu aracın kullanılması ile GUI arabirimi kolaylıkla ve yorulmadan sürükle bırak ve açılan pencerelerde özelliklerin değiştirilmesine dayanan bir yöntem kullanılır. GUI arabirimleri tasarlandıktan sonra GUI'nin arka planında yani MATLAB m file

içersinde, oluşturduğumuz ara bilimlere ilişkin fonksiyonlar ve bu fonksiyonların yapacakları görevler belirlenir.

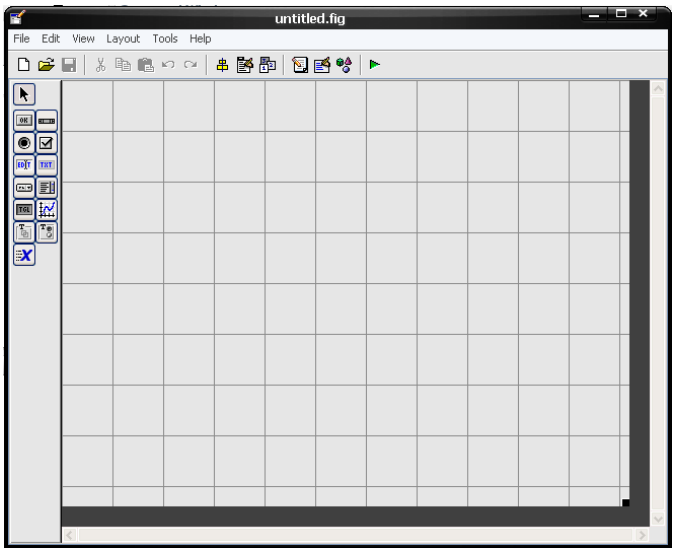
GUIDE MATLAB'ın GUI tasarımcılarına sunduğu içerisinde çeşitli araçlar içeren ve kolaylık sağlayan bir grafiksel GUI geliştirme ortamıdır. GUIDE kullanılarak tıkla ve sürükle-bırak tekniği ile GUI ara üne nesneler (örneğin butonlar, text kutuları, liste kutuları, grafikler v.s.) kolaylıkla eklenebilir. Ayrıca, eklenen nesnelerin hizalanması, tab sırasının değiştirilmesi, görsel ayarlar üzerinde manipülasyonlar yapılması da bu ortamın tasarımcılara sunduğu imkânlardan bazılarıdır.

MATLAB programı üzerinde file menüsü new GUI seçeneği tıklanıldığında şekil II.15'deki gibi bir pencere gelir. Bu pencereden eğer yeni bir GUI tasarımı yapacak isek Blank GUI seçeneğini seçilir. Şayet önceden yapılmış bir tasarımı açılmak isteniyor ise Open Existing GUI sekmesinden sonra istenilen dosyayı seçilir.



Şekil II.15 Yeni Bir GUI Oluşturulması

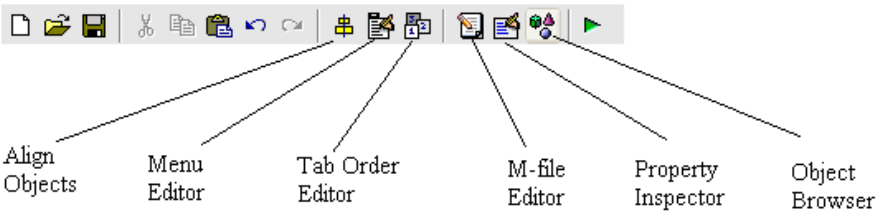
Burada yeni bir tasarım oluşturulacağını kabul edilir ve blank GUI sekmesi üzerinden ok butonu tıklanılır. Şekil II.16'daki GUIDE LAYOUT Editor (GUIDE Çalışma Alanı) penceresine ulaşılır.



Şekil II.16 GUI Çalışma Alanı

Çalışma alanına arayüzü oluşturacak birimlerin eklenmesi için sol tarafta bulunan nesne butonlarından istenilen nesneye ait buton tıklanır ve daha sonra çalışma alanında uygun görülen bir noktaya tıklandığında o noktaya ilgili nesne eklenmiş olacaktır. İstenirse çalışma alanındaki bir nesne farenin sol tuşu ile tıklanıp bırakılmadan çalışma alanının herhangi bir yerine sürüklenebilir.

II.7.3 GUIDE Araç Çubuğunun İncelenmesi



Şekil II.17 GUIDE Araç Çubuğunun İncelenmesi

II.7.3.1 Menu Editor

GUI uygulamasına istenilirse File, Edit… v.b. gibi menü içeren programlarda olduğu gibi arayüz üzerine bir menü eklenmesi ve eklenen menü ile ilgili işlemelerin yapılması bu araç vasıtasıyla sağlanır.

II.7.3.2 Align Objects

Bu araç sayesinde GUI çalışma alanına eklenilen nesnelerin yatay ya da dikey olarak hizalanması işlemleri gerçekleştirilir.

II.7.3.3 Tab Order Editor

Tab Order Editor kullanılarak GUI yüzeyindeki nesnelerin birinden diğerine tab tuşu ile geçiş sırası (örneğin bir buton seçili ve aktif iken bir başka butona ya da bir liste kutusuna tab tuşu kullanılarak geçilmesi gibi.) değiştirilebilir.

II.7.3.4 Property İnspector

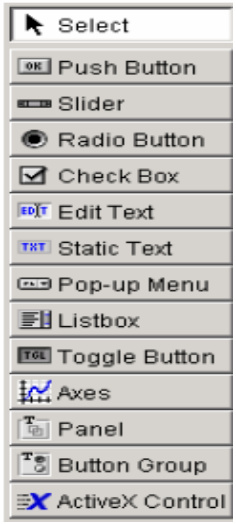
Bu pencere sayesinde de GUI uygulamasına eklenen nesnelerin özellikleri değiştirilebilir ya da var olan özelliklerinin ve değerlerinin neler olduğu gözlenilebilir.

II.7.3.5 Object Browser

Bu araç ile tasarımcı GUI uygulamasına eklemiş olduğu nesnelerin ve isimlerinin neler olduğu genel hali ile bakabilir.

II.7.3.6 M-File editor

Hazırlanmış olan GUI uygulaması ile ilgili komutları görebilmek ve üzerinde değişiklik yapabilmek için bu araç kullanılır.



Şekil II.18 Component Panel

II.7.4 GUI Nesnelerinin Açıklanması

MATLAB GUIDE aracı kullanarak boş (blank) bir GUI çalışma ekranını açtığımızda aşağıda görülen component panel pek çok nesnenin kullanılabileceği görülmektedir. Bu nesnelerin sırasıyla özelliklerini aşağıda belirtilmiştir.

II.7.4.1 Push Button

Normal bir buton özelliği taşımaktadır. Bir buton üzerine tıklanması ile yapılacak işlemlerin komutları m file üzerinde ilgili callback'lerin altına yazılır.

II.7.4.2 Toggle Buton

Çift durumlu bir buton özelliği taşıyan bu nesne ile iki farklı seçenek içeren durumlarda örneğin bu buton basılı ise bir işlemin, bu buton basılmamış ise başka işlemlerin yapılması gerektiği yerlerde tercih edilen bir nesnedir. Buton grubu nesnesi ile beraber kullanımı tavsiye edilir.

II.7.4.3 Radio Buton

Birden fazla seçeneğin olduğu, ancak seçeneklerden sadece herhangi birinin seçilebileceği hallerde bu nesne kullanılır. Buton grupları ile kullanılması genellikle tercih edilir.

II.7.4.4 Check Box

Kullanıcıya seçim yapabileceği ve birden fazla şıkkı işaretleyebileceği durumlarda bu nesne kullanılır.

II.7.4.5 Edit Text

Bir kullanıcıdan bilgi girişi ya da bir değerin alınması söz konusu olduğunda giriş elemanı olarak sıklıkla kullanılan bir nesnedir.

II.7.4.6 Static Text

Kullanıcıya herhangi bir bilgi verme ya da bulunan bir sonuç veya değeri gösterme amacıyla sıklıkla kullanılan bir nesnedir.

II.7.4.7 Slider

Kullanıcıdan bir giriş değerini kaydırılmak suretiyle kolaylıkla alınmasına imkân veren bir nesnedir.

II.7.4.8 List Box

Kullanıcıya bilgi verme amacıyla kullanılabileceği gibi bir değeri listeden seçmek amacıyla da kullanılan sabit bir liste kutusu niteliğinde kullanılan bir nesnedir.

II.7.4.9 Pop-Up Menu

Kullanıcıdan alınmak istenilen bilgileri açılan bir listeden seçme özelliği taşıyan bir nesnedir.

II.7.4.10 Axes

Yapılan iş ile ilgili grafik çizimlerinin kullanıcıya gösterilmesini sağlayan bir nesnedir.

II.7.4.11 Panel

GUI yüzeyi nesnelerinin kullanıcıya daha anlamlı ve güzel gözükmesini sağlayan, ayrıca tasarımcıya GUI dizaynında kolaylık sunan bir nense olup, GUI yüzeyi nesnelerinin gruplanması ve bir arada gösterilmesi amacıyla kullanılır.

II.7.4.12 Button Group

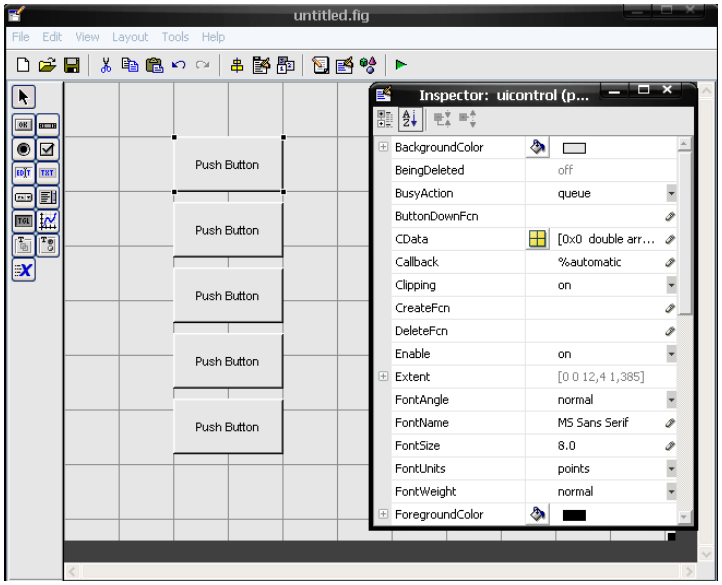
Radio veya toggle tipteki buton nesnelerinin bir arada kullanılarak kullanıcının birden fazla seçenekten sadece bir tanesini seçmesini sağlamak amacıyla kullanılan bir nesnedir

II.7.4.13 ActiveX Component

MATLAB GUI tasarımları sadece yukarıda belirtilen nesneler ile sınırlı değildir. Tasarıcı ve programcı ayrıca, ActiveX adı verilen ve değişik alternatifi olan nesnelerin kullanılmasına da imkân verir. Böylece hem tasarımcı hem tasarlanılacak GUI arayüzünün kullanımı bakımından kullanıcıya esneklik sağlanmış olunur.

II.7.4.14 Nesneler Üzerinde Değişiklik Yapmak

Çalışma ortamına getirilen nesneler üzerinde isim, yazı karakteri boyutu, renk, koordinat, arka plan rengi, hedefleme(tag) tanımlanması gibi işlemlerin yapılması için nesnenin üzerine iki kere tıklanarak açılan inspector penceresiyle bu işlemler gerçekleştirilebilir.

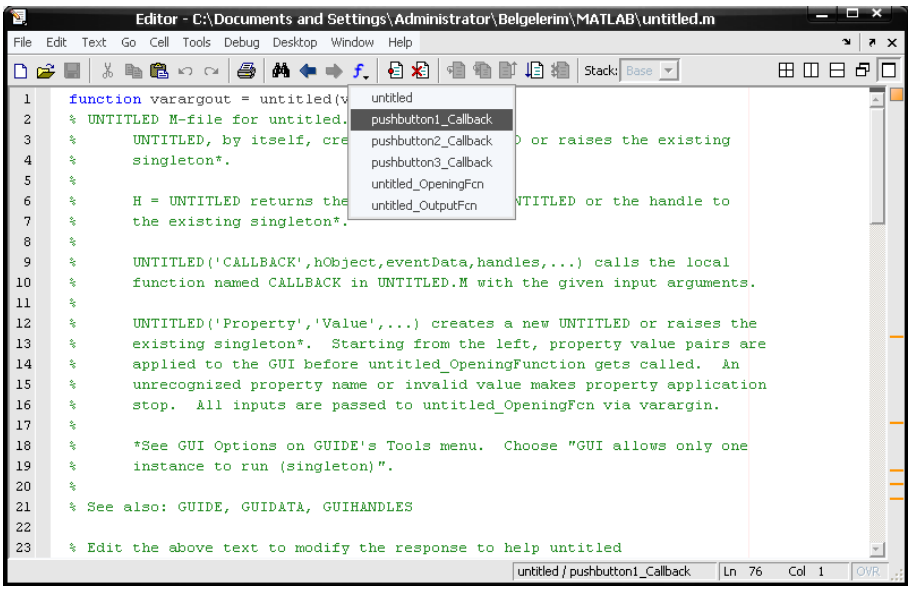


Şekil II.19 Inspector Düzenleme Penceresi

II.7.5 GUI Arayüzünün Programlanması

Bir GUI arayüzünün programlanması demek o çalışmanın kaydedildiği isimle aynı zamanla oluşturulan .m uzantılı dosya içerisine kodlama satırlarının eklenmesi demektir. Bu dosyanın içine görebilmek, değişiklik yapabilmek için GUIDE çalışma ekranı penceresinden View/M-File Editor komutu kullanılabilinir. Ardından karşımıza şekil II.20'deki gibi bir pencere gelecektir.

Şekil II.20'deki pencerede hazırlamış olduğumuz GUI tasarımına ait kodlar gözükmektedir. Burada pek çok kodun hazır eklenmiş olduğu görülecektir. Bu kodlar otomatik olarak MATLAB GUIDE tarafından eklenmiştir. Biz burada ilgili butonlara ve liste kutularına ya da istenilen bir nesneye ait callback isimli alt program parçalarına ilgili kodları yazacağız. Bir nesneye ait callback in bulunduğu satıra gitmek için araç çubuğunda yer alan f simgeli butona tıklanır ve açılan listeden ilgili nesneye ait callback in ismi seçilir.



Şekil II.20 GUI Kodlamalarının Yazıldığı M File Penceresi

II.7.5.1 Push Button

Arayüz içersinde oluşturulan bir butona tıklandığında bu butonun yapılması isten görevi, m file üzerindeki fonksiyonun altındaki komutlara göre yapar örneğin

function pushbutton1_Callback(hObject, eventdata, handles)

A=5; B=10; C=A+B;

butonun yapmasını istediğimiz bu basit toplama işlemini, arayüz üzerinde butona tıkladığımız an gerçekleştirir.

II.7.5.2 Toggle Buton

Bir toggle buton çift durumlu çalışır. Bir kere tıklandığında basılı kalır. Bir daha tıklanırsa basılı kalmayıp eski konumuna geri döner. Böyle bir nesnede geçerli buton konumunu öğrenmek ve kullanabilmek için aşağıdaki komut satırları kullanılmalıdır.

function togglebutton1_Callback(hObject, eventdata, handles)

button_state = get(hObject,'Value');

if button_state == get(hObject,'Max')

% Toggle buton basıldığında yapılacak işlemler

elseif button_state == get(hObject,'Min')

% Toggle buton basılmadığı durumda yapılacak işlemler

End

II.7.5.3 Radio Buton

Radio buton buton group nesnesi ile birlikte kullanıldığında daha etkili sonuçlar alınır. Ancak, kodlama yolu ile de radio butonların konumu kontrol edilebilir. Bir radio butonun basılıp basılmadığının kontrolü için şu kodlar kullanılabilir:

```
if (get(hObject,'Value') == get(hObject,'Max'))
% Radio buton basıldığında yapılacak işlemler
else
% Radio buton basılmadığı durumda yapılacak işlemler
End
```

II.7.5.4 Check Box

Check Box nesnesinin konum kontrolü de radio butonlarınkine benzer şekildedir.

```
function checkbox1_Callback(hObject, eventdata, handles)
if (get(hObject,'Value') == get(hObject,'Max'))
% Checkbox nesnesi işaretlendiğinde yapılacak işlemler
else
% Checkbox nesnesi işaretlenmediği durumda yapılacak işlemler
end
```

II.7.5.5 Edit Text

Edit text'te yazı yada rakam yazdırtabilir bu arayüz üzerinden bilgiler alınabilir. Alınan bilgiler string tiptedir ve sayısal olarak kullanılamazlar. Sayısal olarak kullanabilmek için öncelikle edit box içerikleri sayısala dönüştürülmelidir. Örneğin

```
a=get(handles. editText,'String');
```

Arayüzde tanımlanan edit text üzerine yazılan bilgileri string formatında a ya tanımlar ve a üzerinde yapılacak işlemler fonksiyon içerisine yazılır.

II.7.5.6 Static Text

Edit Text nesnesi ile benzer özelliklere sahiptir. Bu nesnenin edit text ten tek farkı kullanıcıdan bilgi girişi alınamamasıdır, sadece bilgi göstermek amaçlı kullanılır. Kodlama mantığı edit text nesnesi ile aynıdır.

```
set(handles.answer_staticText,'String',c);
```

c içersinde tanımlan string formatındaki bilgiyi arayüz üzerindeki ilgili static text içersine yazdırır.

II.7.5.7 Slider

Bir kaydırıcı (slider) nesnesinin geçerli değerini program yoluyla okumak için gerekli komut satırları şöyle olmalıdır.

```
function slider1_Callback(hObject, eventdata, handles)
slider_value = get(hObject,'Value');
% Callback bloğunun devamı ve diğer komutlar
```

Bir kaydırıcı nesnesinin en küçük ve en büyük değerlerinin de ayarlanması bir programcı için gereklidir. Bunun için bu nesnenin max ve min özellikleri kullanılmalıdır.

II.7.5.8 List Box

List box nesnelerinin liste tipindeki string içeriğinin kullanılabilmesi için bu nesnelerin value ve string özellikleri birlikte kullanılır. Kodlama mantığı şu şekilde olacaktır:

```
function listbox1_Callback(hObject, eventdata, handles)
index_selected = get(hObject,'Value');
list = get(hObject,'String');
item_selected = list{index_selected}; % Hücre dizisinden çevirme işlemi
```

II.7.5.9 Pop-Up Menu

Popup menü nesnelerinde seçilen bir öğenin hangisi olduğu anlamak için bu nesnelerin Value özelliğinden yararlanılır. Kodlama örneği aşağıda gösterilmiştir.

```
function popupmenu1_Callback(hObject, eventdata, handles)
val = get(hObject,'Value');
switch val
case 1
% Birinci öğe seçili iken yapılacak işlemler
case 2
% İkinci öğe seçili iken yapılacak işlemler
% Callback bloğunun devamı ve diğer komutlar
```

II.7.5.10 Axes

Grafik çizimlerinin ve resimlerin kullanıcıya sunulmasında sıklıkla kullanılan bir nesne olan axes için bir GUI arayüzüne içersinde oluşturulan axes alanı ile çizimlerin ve resimlerin gösterimi yapılabilir.

axes(handles.axes1)

plot(A)

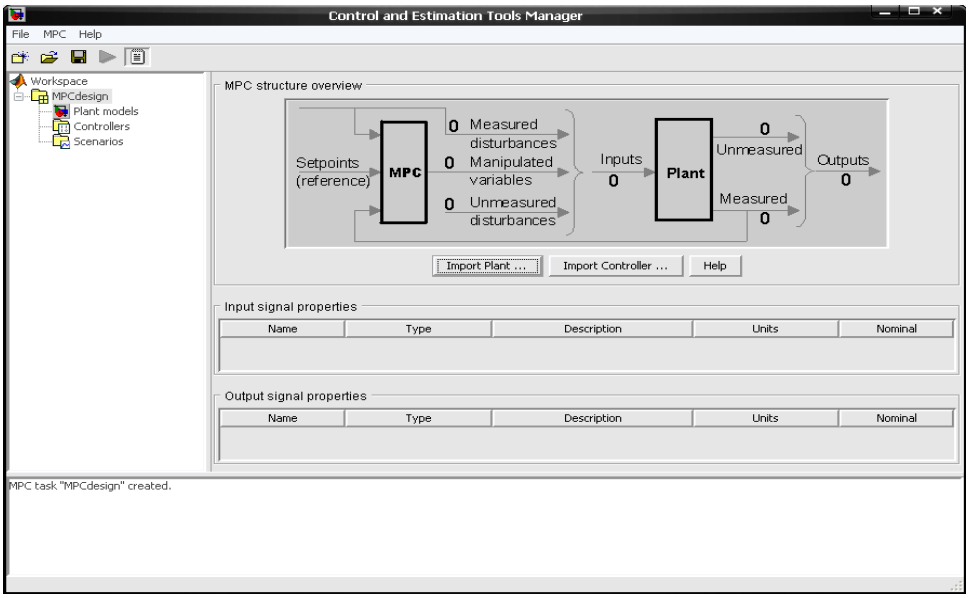
Bu komutlar ile A içersinde kayıtlı bulunan grafiğin ya da reisimin axes1 içersinde gösterilmesidir.

II.7.5.11 Panel

Bu nesneler programlama amacı taşımayan yapıdadırlar. Panellerin kullanım amacı görsel olarak GUI uygulamasını zengin kılmak ve kullanıcıya yapacağı işlemler ile ilgili kullanımını kolaylaştırmaktır [18].

II.8 MATLAB MPC TOOLBOX

MATLAB ortamında model öngörülü kontrol işlemlerinin yapılığı kısımlar MPC Toolbox altında çalışmaktadır. MPC Toolbox MATLAB'in ana penceresi üzerinde bulunan start sekmesinin altında bulunur. MPC Toolbox çalıştırıldığında şekil II.21'deki gibi çalışma penceresi açılır.

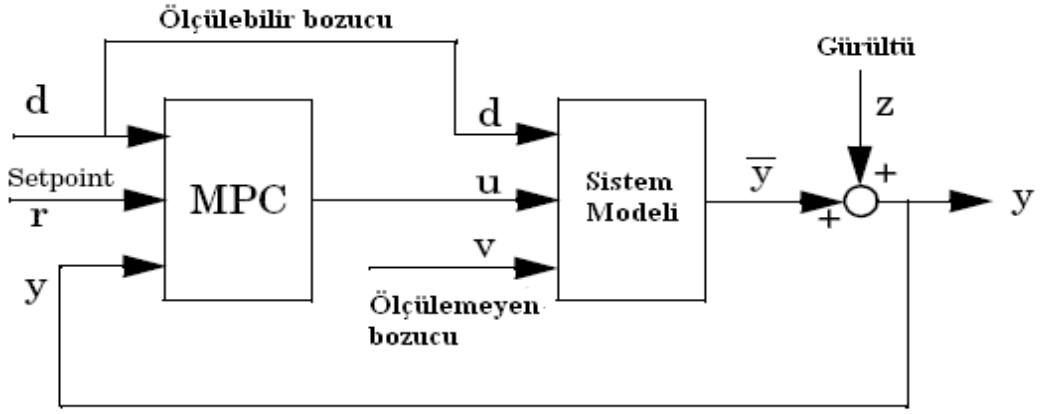


Şekil II.21 MATLAB MPC Toolbox'ın Görünüşü

Bu toolbox çalışmasında MATLAB üzerindeki bilgilerin workspace tanımlanmasıyla gerçekleşir. Bu işlem toolbox üzerindeki import plant, import controller butonları ile gerçekleşir. MATLAB'in sunduğu imkânlardan biride import işlemini gerçekleştirmeden MATLAB'in alt yapısında tasarlanan MPC Toolbox'a ait olan komutlar ile de model öngörülü kontrol işlemi gerçekleşebilir. BU komutların yapısını ve MATLAB üzerinde MPC işleminin yapılması aşağıdaki kısımlarda anlatılmıştır.

II.8.1 MATLAB Programında Sistemin MPC Sürecine Hazırlanışı

MATLAB m file üzerinde yapılan MPC ile ilgili çalışmalar MPC Toolbox tarafından referans alınarak gerçekleşmiştir. Bu sürecin blok diyagramı şekil II.22'teki gibidir.



Şekil II.22 MPC Toolbox'a Göre Model Öngörülü Kontrol Bloğu

Şekil II.22 üzerinde gösterilen simgelerin anlamları tablo II.1 üzerinde açıklanmıştır. Model öngörülü kontrol işlemine başlamadan önce sistemi oluşturan modellerin MATLAB ve MPC Toolbox'ta kullanılan formata çevrilmesi gerekmektedir. Bu işlemde kullanılan MATLAB komutları ise aşağıdaki alt başlıklarda açıklanmıştır.

v	Ölçülemeyen bozucu sinyaldir. Saha üzerindeki etkisi belli olmayan bozuculardır. MPC bu durumu geri besleme üzerinden düzeltmektedir.
r	Referans girişi sinyalidir.
u	Ayarlanabilir değişken sinyaldir. MPC kontrol bloğunun çıkışından alınan sinyaldir.
d	Ölçülebilir bozucu sinyaldir. Çevreden gelen ölçülebilir bozucuların etkisini öngörülü kontrol ile minimize edebilmek için kontrolöre ve sahaya ileri besleme ile dahil edilen sinyaldir.
$\boldsymbol{y}$	Çıkış sinyalidir.
y	Ayarlanabilir çıkış sinyalidir.
z	Ölçülemeyen gürültü. Elektriksel gürültüler, örnekleme hataları, kalibrasyon farkı gibi istenmeyen etkenler.

Tablo II.1 Model Öngörülü Kontrol Sürecinde Kullanılan Semboller

II.8.1.1 Poly2tfd

g=poly2tfd(num, den)

g=poly2tfd(num, den, delt, delay)

z yada laplace domaininde tanımlanmış transfer fonksiyonunu MATLAB standartlarına (polynominal) uygun hale getirir.

$G\ s\ = \frac{b_0 s^n + b_1 s^{n-1} + \cdots + b_n}{a_0 s^n + a_1 s^{n-1} + \cdots + a_n}$ $G\ z\ = \frac{b_0 + b_1 z^{-1} + \cdots + b_n z^{-n}}{a_0 + a_1 z^{-1} + \cdots + a_n z^{-n}}$

$num = [b_0 + b_1 + \cdots + b_n]$ delt= Örnekleme zamanı

$den = [a_0 + a_1 + \cdots + a_n]$ delay= Bekleme süresi

Örneğin

$G\ s\ = \frac{3s-1}{5s^2+2s+1} e^{-2.5s}$ Şeklinde tanımlanan transfer fonksiyonun polynominal formata çevrilmesi aşağıdaki gibi gerçekleşir.

$num =\ 3\ -1\ ;\ den =\ 5\ 2\ 1\ ;\ delt = 0;\ delay = 2.5;$

$g = poly2tfd([3\ -1], [5\ 2\ 1], 0, 2.5)$

$$g = \begin{matrix} 0 & 0 & 2 \\ 5 & 2 & 1 \\ 0 & 2.5 & 0 \end{matrix}$$

II.8.1.2 Tfd2step

Adım cevabı yöntemi kullanılan sisteler için modelin MPC adım formatına dönüşümünü ve sistemi oluşturan modellerin bir bütün olarak değerlendirilmesini sağlar.

$plant = tfs2step(tfinal, delt, nout, g1, \dots, gn)$

tfinal=Adım cevabının bitiş süresi.

delt=Adım cevabı için istenilen örnekleme zamanı.

nout=Çıkış kararlılığı göstergesi.

$g1, g2, \dots$ gn= N sayıda tf formatında transfer fonksiyonu dizisi. N birden büyük olduğunda aşağıdaki gibi matris oluşturur. N maksimum 25 olabilir.

$$\begin{matrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n_u} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,n_u} \\ \vdots & \vdots & \ddots & \vdots \\ g_{n_y,1} & g_{n_y,2} & \cdots & g_{n_y,n_u} \end{matrix}$$

Örneğin

$g11 = poly2tfd(12.8, [16.7\ 1], 0, 1);$

$g21 = poly2tfd(6.6, [10.9\ 1], 0, 7);$

$g12 = poly2tfd(-18.9, [21.0\ 1], 0, 3);$

$g22 = poly2tfd(-19.4, [14.4\ 1], 0, 3);$

$delt = 3;\ ny = 2;\ tfinal = 90;$

$plant = tfd2step(tfinal, delt, ny, g11, g21, g12, g22);$

Verilen transfer fonksiyonları poly formatına çevrilmiş ve tfd2step komutuyla plant adı altında bir bütün olarak birleştirilmiştir.

II.8.1.3 Tfd2mod

Çok girişli çok çıkışlı sistemlerin modellerinden elde edilen transfer fonksiyonlarının poly formatına çevriminden sonra MPC kontrol biriminin içerisinde hesaplanabilmesi için sistemin modellerinin tek bir model dizisi haline dönüştürülmesidir. Bu komut genellikle durum-uzay sistemli işlemlerde kullanılmaktadır.

$model = tfd2mod(delt, ny, g1, g2, g3, \dots, g25)$

ny=Ayarlanabilir değişkenlerin çıkış sayısı.

delt=Adım cevabı için istenilen örnekleme zamanı.

$g1, g2, \dots$ gn= N sayıda tf formatında transfer fonksiyonu dizisi.

Örneğin

$g1 = poly2tfd(12.8, [16.7\ 1],0,1);$

$g2 = poly2tfd(6.6, [10.9\ 1],0,7);$

$delt = 3$

$ny = 2$

$model\ =\ tfd2mod(delt, ny, g1, g2)$

Verilen transfer fonksiyonları tf formatından poly formatına çevrilmiş ve tfd2mod komutu ile sistem bütünleşik olarak mod formatına dönüşmüştür.

II.8.1.4 C2dmp

Sürekli zamandaki durum-uzay modelini mod formatına çevirir.

$[phi, gam] = c2dmp(a, b, T);$

a ve b durum-uzay matrisleri, T örnekleme zamanıdır.

II.8.1.5 Ss2mod

Ayrık zamandaki durum-uzay modellerini mod formatına dönüştürür.

$mod = ss2mod(phi, gam, c, d, T)$

phi ve gam mod formatına çevrilmiş durum-uzay matrisleri, c ve d ayrık zamandaki durum uzay matrisleri.

II.8.2 MPC Kazanç Katsayısının Hesaplanması

Bir önceki konularda sistemin model öngörülü kontrol işlemi yapılmadan önceki durumunu MATLAB programında kullanılan formata dönüştürülmesi işlemlerinin hangi komutlar ile nasıl yapıldığından bahsedilmiştir. Bu bölümde uygun formata dönüştürülen sistemler üzerinden model öngörülü kontrol kazanç katsayısı hesaplanacaktır. Model öngörülü kontrol kazanç katsayısı sadece kısıtlamasız sistemlerde hesaplanır kısıtlama olan sistemlerde ise kazanç katsayısı tüm kontrol sisteminin hesaplandığı süreç içerisinde hesaplanır. Kazanç katsayısını hesaplanmasıyla ilgili MPC Toolbox içerisindeki komutlar aşağıdaki başlıklar altında açıklanmıştır.

II.8.2.1 Mpccon

Adım formatındaki MPC hesaplamalarında, MPC kontrol sinyalinin kazancını hesaplar.

$Kmpc = mpccon(model, ywt, uwt, M, P)$

model=Kontrol edilecek sistemin adım formatına uygun modeli.

ywt=Referans girişindeki sinyalleri izlemede oluşabilecek hatalara karşı uygulanacak ağırlık matrisi.

uwt=Ölçülebilir değişkenleri yani MPC kontrol sinyalini izlemede oluşabilecek hatalara karşı uygulanacak ağırlık matrisi.

M=Kontrol ufku.

P=Öngörü ufku.

Komutun kullanımıyla ilgili örnekler bölüm III'te anlatılacaktır.

II.8.2.2 Smpccon

Durum uzay tabanlı kontrol sistemlerinde MPC kontrol kazancının hesaplandığı komuttur.

$Ks = smpccon(imod, ywt, uwt, M, P)$

ywt =Referans girişindeki sinyalleri izlemede oluşabilecek hatalara karşı uygulanacak ağırlık matrisi.

uwt=Ölçülebilir değişkenleri yani MPC kontrol sinyalini izlemede oluşabilecek hatalara karşı uygulanacak ağırlık matrisi.

M=Kontrol ufku.

P=Öngörü ufku.

imod= Modellerin transfer fonksiyonlarını, MPC durum uzay formatına uygun şekle dönüşülmüş hali.

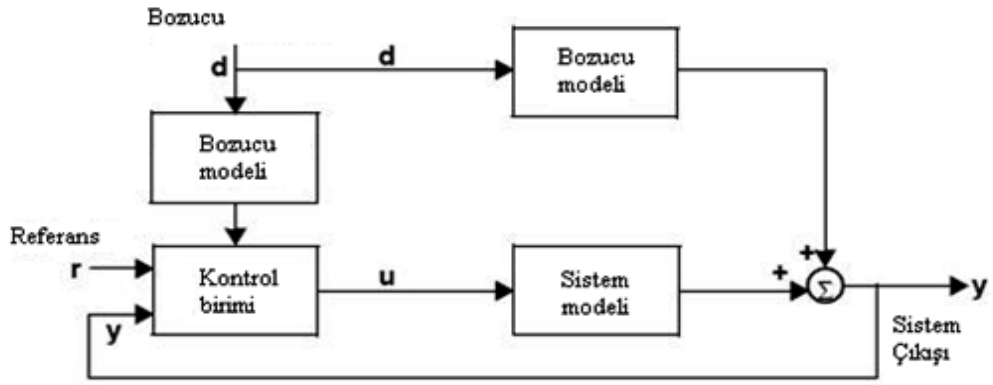
Komutun kullanımıyla ilgili örnekler bölüm III'te anlatılacaktır.

II.8.3 Model Öngörülü Kontrol İşleminin Gerçekleştirilmesi

Bu bölümde anlatılacak olan komutlar MPC Toolbox ile doğrudan ilişkilendirilerek model öngörülü kontrol sürecini gerçekleştireceklerdir. Bu komutların uygulandığı model tipine göre işlevleri aşağıdaki alt başlıklarda açıklanacaktır ve çalışma şekilleri bölüm III'te uygulanacaktır.

II.8.3.1 Mpcsim

Adım cevabı yöntemiyle kontrol edilecek MPC sistemlerini formata uygun hale getirilen sistem modelleri üzerinden ayarlanabilir değişkenleri kontrol ederek model öngörülü kontrol işlemini gerçekleştirir.



Şekil II.23 Tek Girişli Tek Çıkışlı Sistemin MPC Blok Diyagramı

Tek girişli tek çıkışlı bir sistemin adım cevabı yöntemi ile model öngörülü kontrol işlemi aşağıdaki komut ile gerçekleşir.

$y = mpcsim(plant, model, Kmpc, tend, r)$

model=Kontrol edilecek sistemin adım formatına uygun şekilde hesaplanan, kontrol için durum tahminine imkan sağlayacak model.

plant =Sistemin MPC adım cevabı yöntemine uygun formatı.

Kmpc= mpccon komutuyla elde edilen MPC kazanç matrisi.

tfinal=Adım cevabının bitiş süresi.

r=Referans girişi.

II.8.3.2 Cmpc

Kapalı cevrim sistemlerindeki kontrolü, ayarlanabilir değişkenler ya da sistem çıkışı üzerinden sabit bir sınır getirerek yani kısıtlama yaparak kontrolü gerçekleştir. Sistemin blok diyagramı yukarıdaki şekil II.24 ile aynıdır. Burada kontrol birimine ek olarak kısıtlamalar konulmuştur. Kısıtlama ile birlikte model öngörülü kontrol işlemini gerçekleştiren komut aşağıdaki gibidir.

$[y\ u] =$

$cmpc(plant, model, ywt, uwt, M, P, tend, r, ulim, ylim, tfilter, dplant, dmodel)$

model=Kontrol edilecek sistemin adım formatına uygun şekilde hesaplanan, kontrol için durum tahminine imkan sağlayacak model.

plant =Sistemin MPC adım cevabı yöntemine uygun formatı.

ywt=Referans girişindeki sinyalleri izlemede oluşabilecek hatalara karşı uygulanacak ağırlık matrisi.

uwt=Ölçülebilir değişkenleri yani MPC kontrol sinyalini izlemede oluşabilecek hatalara karşı uygulanacak ağırlık matrisi.

M=Kontrol ufku.

P=Öngörü ufku.

tend=Öngörülü kontrolün bitişi süresi

r=Sistem referans girişi

ulim=Ayarlanabilir değişkenlerin kısıtlandırılacağı limit. Limitlerin sayısı ve değerleri ise kontrol edilen giriş çıkış miktarına göre, sistemin negatif ya da pozitif değerlerde kontrol yaptığı durumlarda göre değişmektedir.

Örneğin $u_{min} = -10$ $u_{max} = 10$ ayarlanabilir değişkenlerin -10 ve 10 aralığı arasında kısıtlanır. yada ulim=[inf inf 5 25] şeklinde yapılan kısıtlamalarda her dizideki sıraya göre birinci saniye inf yani kısıtlama yok, ikinci saniye kısıtlama yok, üçüncü saniye artı sonsuzdan 5'e kadar, dördüncü saniyede 5 ile 25 değerleri arasında kısıtlama yapılır.

ylim=Sistem çıkışındaki sinyalleri kısıtlamak amacıyla kullanılır. Kısıtlamanın yapılma şekli ayarlanabilir değişkenlerin kısıtlamasıyla aynıdır.

tfilter= Zaman içersindeki ölçülemeyen gürültülerin etkisini azaltmak için kullanılan sabit matrislerdir.

dmodel=Kontrol edilecek sisteme etki eden ölçülebilir bozucuların adım formatına uygun şekilde hesaplanan, kontrol için durum tahminine imkan sağlayacak modeli.

dplant =Ölçülebilir bozucuların modelli.

II.8.3.3 Smpcsim

Kapalı çevrim sistemlerindeki durum-uzay sistemlerinin uygun formatlara dönüştürülmesiyle elde edilen ayarlanabilir değişkenlerin kontrolü üzerinden yapılan çok girişli çok çıkışlı sistemlerin model öngörülü kontrolünü hesaplar.

$[y, u] = smpcsim(pmod, imod, Ks, tend, r, usat, z, d, w)$

pmod=Sistem modelini durum uzay formatından MPC formatına göre temsil edildiği sistemin model bilgisi.

$imod$=Kontrol edilecek sistemin durum uzay formatından MPC formatına göre temsil edildiği model bilgisi.

Ks= smpccon fonksiyonuyla hesaplanan MPC kazanç matrisi.

tend=Öngörülü kontrolün bitişi süresi.

r= Sistem referans girişi.

usat=Ayarlanabilir değişkenler kısıtlama matrisi.

z=Ölçülebilir gürültü.

d=Ölçülebilir bozucu.

w=Ölçülemeyen bozucu.

II.8.3.4 Scmpc

Kapalı cevrim sistemlerindeki kontrolü, ayarlanabilir değişkenler yada sistem çıkışı üzerinden sabit bir sınır getirerek yani kısıtlama yaparak kontrolü gerçekleştir.

$[y\ u] = cmpc(pmod, imod, ywt, uwt, M, P, tend, r, ulim, ylim, z, d, w)$

pmod=Sistem modelini durum uzay formatından MPC formatına göre temsil edildiği sistemin model bilgisi.

$imod$=Kontrol edilecek sistemin durum uzay formatından MPC formatına göre temsil edildiği model bilgisi.

uwt=Ölçülebilir değişkenleri yani MPC kontrol sinyalini izlemede oluşabilecek hatalara karşı uygulanacak ağırlık matrisi.

M=Kontrol ufku.

P=Öngörü ufku.

tend=Öngörülü kontrolün bitişi süresi

r=Sistem referans girişi

ulim=Ayarlanabilir değişkenlerin kısıtlandırılacağı limit. Limitlerin sayısı ve değerleri ise kontrol edilen giriş çıkış miktarına göre, sistemin negatif yada pozitif değerlerde kontrol yaptığı durumlarda göre değişmektedir.

ylim=Sistem çıkışındaki sinyalleri kısıtlamak amacıyla kullanılır. Kısıtlamanın yapılma şekli ayarlanabilir değişkenlerin kısıtlamasıyla aynıdır [19].

z=Ölçülebilir gürültü.

d=Ölçülebilir bozucu.

w=Ölçülemeyen bozucu.

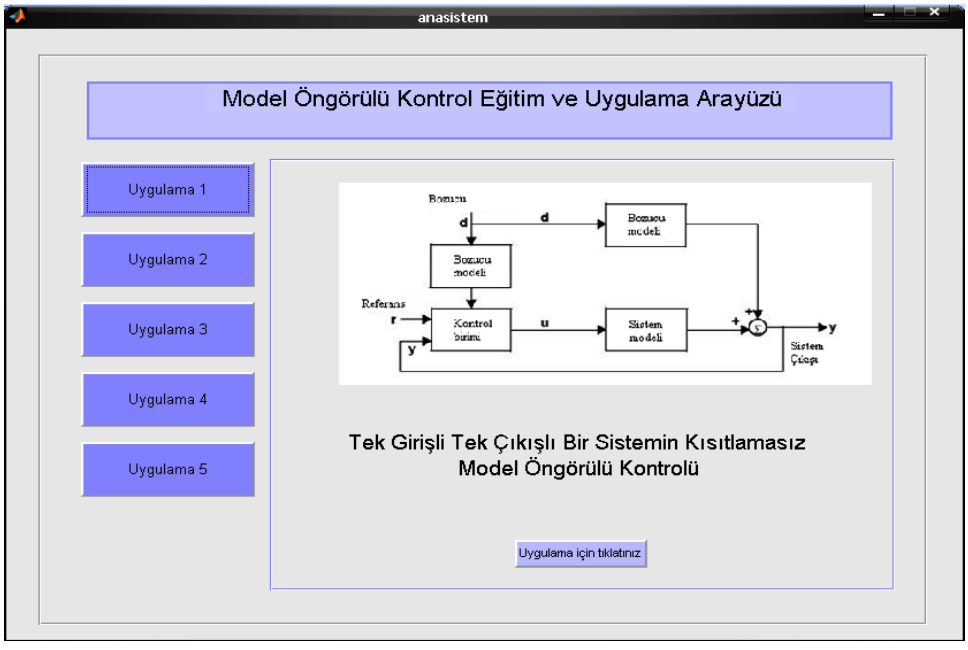
BÖLÜM III

MODEL ÖNGÖRÜLÜ KONTROL EĞİTİM VE UYGULAMA ARAYÜZÜ TASARIMI UYGULANMASI

Yapılan çalışmada ilk olarak hazırlanan eğitim ve uygulama arayüzlerinin ana bir pencerede sunulması ve uygulayıcının istenilen çalışmaya rahatlıkla ulaşabilmesi hedeflenmiştir. Yapılan ana arayüz tasarımı şekil III.1'de gözükmektedir.

Bu arayüz tasarımı kendi içerisinde 5 adet uygulama içermektedir. Bu uygulamalar aşağıdaki gibidir.

1. Tek girişli tek çıkışlı bir sistemin kısıtlamasız model öngörü kontrolü
2. Tek girişli tek çıkışlı bir sistemin kısıtlamalı model öngörü kontrolü
3. Çok girişli çok çıkışlı bir sistemin kısıtlamasız model öngörü kontrolü
4. Çok girişli çok çıkışlı bir sistemin kısıtlamalı model öngörü kontrolü
5. Çok girişli çok çıkışlı bir sistemin durum uzay matrisleri ile kısıtlamalı model öngörü kontrolü



Şekil III.1 Model Öngörülü Kontrol ve Uygulama Arayüzünün Ana Penceresi

Yukarıda ifade edilen uygulamalar için yapılacak olan çalışmalar ilk olarak model öngörülü kontrol işleminin MATLAB MPC Toolbox'a göre gerçekleşmesini daha sonra model ve kontrol sistemlerine ait bilgileri GUI arayüzü üzerinde tasarlanılarak model öngörülü kontrol işleminin bu arayüz üzerinde gerçekleşmesini, gözlemlenebilmesini ve değiştirilebilmesini sağlayacaktır. Bu uygulamalar için yapılacak çalışmalar sırasıyla alt başlıklarda aşağıda açıklanmıştır.

III.1 TEK GİRİŞLİ TEK ÇIKIŞLI BİR SİSTEMİN KISITLAMASIZ MODEL ÖNGÖRÜLÜ KONTROLÜ

Bir model öngörülü kontrolün en basit şekilde uygulandığı tek girişi olan ve tek bir çıkışı olan sistemdir. Şekil II.23 verilen tek girişli tek çıkışı bir sistemin blok diyagramı incelendiğinde sisteme giriş yapan referans ve birde bozucu bulunmaktadır. Bu girişler istenilen sonuç için kontrol biriminde yani MPC toolbox içersinde hesaplatılarak sistemin modeline etki etmektedir. Model ile etkileşimi sonucunda da girişin istediği şekilde çıkış vermeye dönük olarak hazırlanmıştır. Sistemi en basit şekilde formülle gösterecek olursak aşağıdaki gibi gösterilir.

$$y = G_p\,[u] + G_d\,[d] \tag{III.1}$$

III.1.1 Kontrol Edilecek Sistemin Yapısı

Sistemin yapısı denklem III.1'e göre incelendiğinde kontrol edilecek olan model G_p ve sisteme etki eden ölçülebilir bozucunun modeli G_d ile tanımlanmıştır. u kontrol biriminden çıkan MPC toolbox tarafından kontrol edilen ayarlanabilir değişkenler şeklinde adlandırılan kontrol sinyalidir. Bu sinyallin içeriğinde giriş ve geri bildirim değerleri ile ölçülebilir bozucu sinyallerinin etkileşimi ve öngörü, kontrol ufuklarının bu sürece etkileri üzerine hesaplanmış kontrol sinyalleri yer almaktadır.

$$G_p = \frac{5.72}{60s+1} e^{-14s} \tag{III.2a}$$

$$G_d = \frac{1.52}{25s+1} e^{-15s} \tag{III.2b}$$

denklemlerinde ise üzerinde çalışma yapılan tek giriş tek çıkışlı kısıtlamasız model öngörülü kontrol sisteminin sistem modelini denklem III.2a ve bozucu modelinin

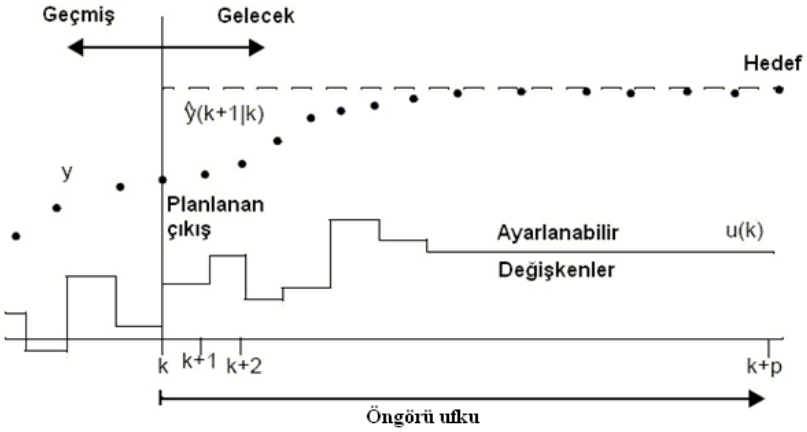
denklem III.2b 'de ifade edilen transfer fonksiyonları gözükmektedir. Bu transfer fonksiyonları sistem modelini oluşturur.

III.1.2 Sistemin Model Öngörülü Kontrolü

Sistemin MPC ile kontrolünü ve temel çalışma mantığını şekil III.2'de gözükmektedir. Hedef olarak tanımlanan y çıkışı sistemin referans girişine göre en istenen şekilde çalışmasını amaçlamaktadır. Bu çalışma sürecine etki eden değişkenler yani ayarlanabilir değişkenler çıkışın hedefe ulaşması üzerinde etkisi büyüktür. Bu etkinin kontrolünde kullanılan MPC yöntemi ile ayarlanabilir değişkenlerdeki değişmeleri öngörü ufku boyunca tahmin etmekte ve bu değişimlerin istenilen hedefe ulaşmada oluşturabileceği sorunları mümkün olduğunca en kısa zamanda ortadan kaldırmaktadır.

Model öngörülü kontrolde kullanılan MATLAB programının MPC Toolbox'ı model öngörülü kontrol işleminin gerçekleşmesine büyük kolaylık sağlamaktadır. MPC Toolboxında MPC ile ilgili kontrol algoritmaları bulunmaktadır. Yapılan çalışmada da bu algoritmalar kullanılmış, algoritma içersindeki formüller üzerinde tanımlamalar ve değişiklikler yapılmıştır.

MPC toolboxını kullanmadan önce bazı verilerin MATLAB'in anlayabileceği formata dönüştürülmesi gerekmektedir. Yapılan bu çalışmada ilk olarak MPC Toolbox'ın kullanımına hazırlık daha sonra kontrol biriminin oluşturulmasından bahsedilecektir.



Şekil III.2 Öngörülü Kontrolü Çalışma Süreci

III.1.2.1 Sistem Transfer Fonksiyonlarının Uygun Standartlara Getirilmesi

Sistemin yapısında verilen kontrol edilecek modelin ve bozucu modelin transfer fonksiyonlarını MATLAB MPC Toolbox tarfından anlaşılabilicek ve adım cevabı yöntemiyle kontrolü yapılacak formata dönüştürülmesi gerekmektedir. Bu işlemler için denklem III.2a'da verilen G_p transfer fonksiyonu MATLAB m file üzerinde $num = 5.72$ $den = 60\ 1$ $delt = 0$ $delay = 14$ şeklinde tanımladığında ve bölüm II'de çalışması anlatılan poly2tfd komutu kullanıldığında aşağıdaki sonuç elde edilir.

$$\mathrm{G_p} = poly2tfd([5.72],[60\ \ 1],0,14) \quad \text{(III.3)}$$

$$\mathrm{G_p} = \begin{matrix} 0 & 5.72 \\ 60 & 1 \\ 0 & 14 \end{matrix} \quad \text{(III.3a)}$$

Aynı işlem $\mathrm{G_d}$ transfer fonksiyonuna da uygulandığında

$$\mathrm{G_d} = poly2tfd([1.52],[25\ \ 1],0,15) \quad \text{(III.4)}$$

$$\mathrm{G_d} = \begin{matrix} 0 & 1.52 \\ 25 & 1 \\ 0 & 15 \end{matrix} \quad \text{(III.4a)}$$

Sonuçları elde edilir. MATLAB standartlarına uygun hale getirilen model ve bozucu transfer fonksiyonlarını adım cevabı ile kontrolü için adım formatına dönüşmesi gerekmektedir. Bunun için yine bölüm II'de anlatılan tfd2step komutu kullanılır.

$tfinal$=250; % Sistemin çalışma süresi

$delt$=5; % Sistemin örnekleme zamanı

$nout$=1; % Sistemin çıkış kararlılığı

$$\mathrm{G_p}plant = tfd2step(tfinal, delt, nout, \mathrm{G_p}) \quad \text{(III.5)}$$

İşlemi yapıldığında sistemin transfer fonksiyonu adım formatına dönüşür. Aynı işlem $\mathrm{G_d}$ transfer fonksiyonuna da uygulandığında

$$\mathrm{G_d}plant = tfd2step(tfinal, delt, nout, \mathrm{G_d}) \quad \text{(III.6)}$$

adım cevabına uygun formatta bozucu modeli elde edilir. Bu kısma kadar yapılan işlemlerde sistem bilgilerinin MPC Toolbox formatında kullanıma hazır hale getirilmiştir. Kontrol sürecinde yapılacak işlemlerde ise ilk olarak sistem modeline uygulanacak olan kazancı öngörü ve kontrol ufukları içersinde hesaplanmasıdır.

III.1.2.2 Model Öngörülü Kontrol Kazanç Katsayısının Hesaplanması

Tek girişli tek çıkışlı sistemlerin model öngörülü kontrol kazanç katsayısı mpccon komutu ile hesaplanır. Bu komutun çalışma şekli bölüm II'de açıklanmıştır. mpccon komutunun MPC Toolbox içerisindeki algoritmasının önemli bir kısmı aşağıdaki gibidir.

```
for i=1:p
  if i <= nywt
        ywt=[ywt;yweight(i,:)'];
  elseif i==p & horizon == Inf,
    ywt=[ywt;100*max([yweight;zeros(1,ny)])'];
  elseif i==p-1 & horizon == Inf,
    ywt=[ywt;max([yweight;zeros(1,ny)])'.*(1+intm*100)];
  else
    ywt=[ywt;yweight(nywt,:)'];
  end
end

for i=1:m
  if i <= nuwt
    uwt=[uwt;uweight(i,:)'];
  else
    uwt=[uwt;uweight(nuwt,:)'];
  end
end
X = [diag(ywt)*Su;diag(uwt)]\[diag(ywt);zeros(mnu,pny)]
```

Buradaki algoritmada sistem modelinin girişteki ve çıkıştaki ağırlık matrislerini yani giriş ve çıkıştaki sinyalleri izlemede oluşabilecek izleme hatalarına karşı tanımlar yapılmış bu tanımlara bağlı olarak öngörü ve kontrol ufkunun süreçteki etkileri hesaba katılarak kazanç katsayısı elde edilmiştir.

Bu işlemin yapılması için kullanılan mpccon komutu ile sisteme uygulanacak kazanç katsayısının bulunması aşağıdaki şekilde gerçekleşir.

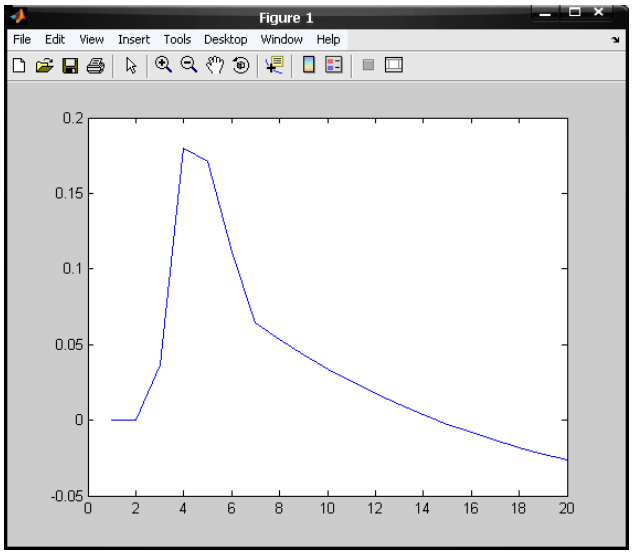
$$Kmpc = mpccon(\mathrm{G_p}plant, ywt, uwt, M, P) \qquad \text{(III.7)}$$

$\mathrm{G_p}plant$ değeri denklem III.6 ile hesaplamıştır burada sisteme etki eden P öngörü ufkunu 20s, M kontrol ufkunu ise 5s alınarak ve ywt, uwt ağırlık matrislerini sırasıyla 1,1 alındığında kazanç katsayısı aşağıdaki gibi bulunur.

```
ywt=1;          % Çıkış ağırlık matrisi
uwt=1;          % Giriş ağırlık matrisi
P=20;           % Öngörü ufku
M=5;            % Kontrol ufku
```

$Kmpc = mpccon(\mathrm{G_p}plant, 1,1,5,20)$

Bu işlemin sonunda 20 saniyelik öngörü ufku boyunca ayarlanabilir değişkenlerin durumuna bağlı şekilde değişerek aşağıdaki şekil III.3'teki gibi bir kazanç katsayısı elde edilir.



Şekil III.3 Model Öngörülü Kontrol Kazanç Katsayısı

Hesaplan kazanç katsayısı sistem modeline uygulanıldığında sistemden alınan çıkışla birlikte ölçülebilir bozucularda sisteme etki etmektedir. Bu etkileri ortadan kaldırmada ise MPC'nin tüm süreci değerlendirerek kontrol etmesi gerekmektedir. Bu işlemi de MPC Toolbox içersinde tanımlanan mpcsim komutu ile gerçekleştirmektedir.

III.1.2.3 Model Öngörülü Kontrolün Gerçekleşmesi

Mpcsim komutunun içeriği incelendiğinde MPC Toolbox içerisinde çalışan ve yaklaşık bin satırlık kod listesi ve bu kodların model öngörülü kontrol sürecinde sisteme etki eden her türlü bilgilerin ayrıştırılıp hesaplandığı bir algoritmadır. Komutun çalışması ile ilgili bilgiler bölüm II'de açıklanmıştır. Komutun uygulanması ve çalışması aşağıdaki gibi gerçekleşir.

$[y\ u] = mpcsim(\mathrm{G_p}plant, \mathrm{G_p}model, Kmpc, tend, r, \mathrm{G_d}plant, \mathrm{G_d}dmodel)$ (III.8)

tend=250; % Kontrolün bitiş süresi

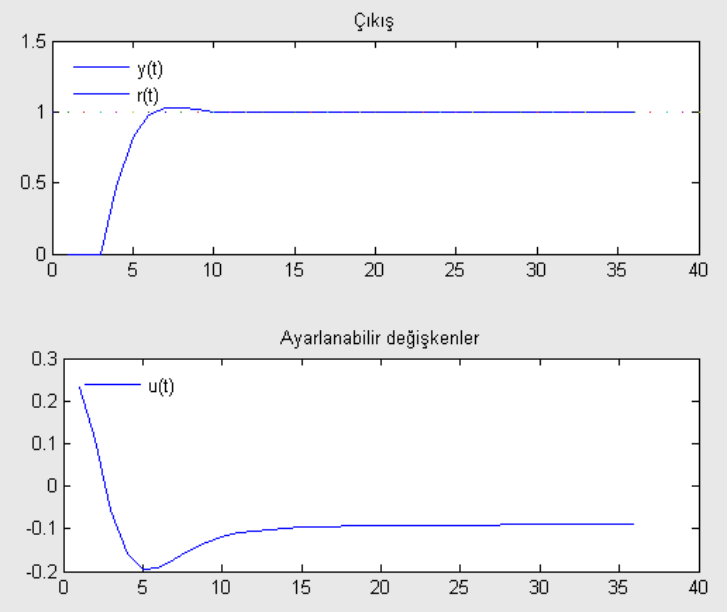
r=1; % Sisteme etki eden referans girişi

$\mathrm{G_p}plant = \mathrm{G_p}model$ %Hesaplamaların daha verimli olması içi eşit alınmıştır

$\mathrm{G_d}plant = \mathrm{G_d}dmodel$ %Hesaplamaların daha verimli olması içi eşit alınmıştır

Yukarıdaki komutlara uygun olarak daha önceden hesaplan değerleri de mpcsim komutu içersinde yerine konulduğunda; mpcsim komutu hesaplamaları kendi içerisindeki algoritmalara gömerek, y sistem çıkışını ve bu süreçteki ayarlanabilir değişkenlerde meydana gelen değişimleri hesaplamaktadır. Bu hesaplar aşağıdaki grafikteki gibi gözükmektedir.

Şekil III.4 üzerindeki sinyallerin süreç içerisindeki değişimi incelendiğinde ilk olarak ayarlanabilir değişkenler üzerinde 10. saniyeye kadarki süreçte sisteme etki eden ayarlanabilir değişken sinyali üzerinde değişimler gözükmektedir. Bu değişimler hem öngörü ve kontrol ufkunun etkisiyle elde edilmiş MPC kazanç katsayısı hem de sistemdeki bozucular ve geri bildirim sinyallerinin süreç içersindeki değimlerine paralel olarak gerçekleşmektedir. Ayarlanabilir değişkenlerin değişimi incelendiğinde 5. saniyeye kadar azalmakta ve -0.2 değerine kadar düşmektedir. Çıkış sinyali incelediğimizde 5. saniyede referans değerine yani istenilen çıkış seviyesine ulaşmaktadır fakat ayarlanabilir değişken 6. saniyede de değerini -0.2 seviyesinde tuttuğunda çıkış sinyali üzerinde aşmalar meydana gelmektedir geri bildirim ile bu işlem 13. saniye kadar rayına oturmakta ve sistemin kontrolü kararlı bir şekilde gerçekleşmektedir.



Şekil III.4 Tek Girişli Tek Çıkışlı Olarak Verilen Sistemin Model Öngörülü Kontrolü

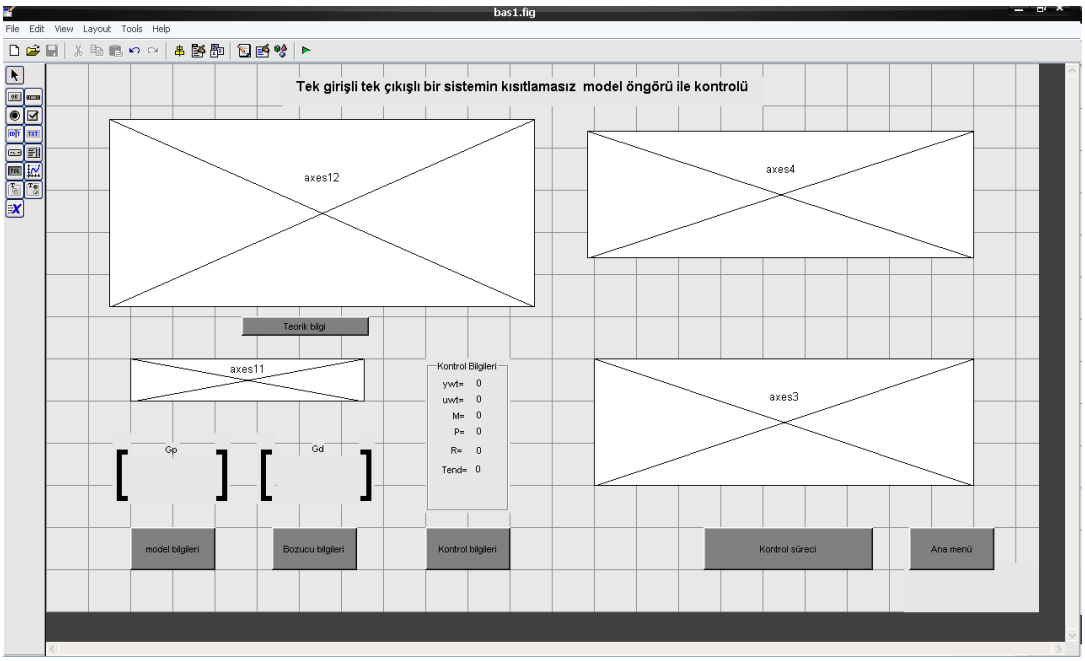
Öngörü ve kontrol ufuklarının etkileri incelendiğinde ufuklarının değerleri düşürülseydi sistemin çalışmasının kararlı bir hale gelmesi daha uzun zaman alacaktı bu da sistemin verimli çalışmasına, zaman ve maliyet gibi etkenlere olumsuz bir şekilde yansıyacaktır.

III.1.3 Tek Girişli Tek Çıkışlı Bir Sistemin Kısıtlamasız Model Öngörülü Kontrolünün Arayüz Tasarımın MATLAB GUI Üzerinde Gerçekleşmesi

Kontrol işlemlerini gerçekleştirildiğimiz tek giriş tek çıkışlı bir sistemin uygulamalarda ve eğitimde kolaylık sağlaması, model öngörülü kontrol işleminin rahatlıkla gözlenmesi açısından arayüz tasarımı büyük bir fayda sağlamaktadır. Arayüz tasarımı MPC'ninde rahatlıkla çalışabileceği MATLAB programının GUI tabanıyla tasarlanmıştır. GUI ile ilgili temel bilgiler bölüm II'de ayrıntılı bir şekilde açıklanmıştır. Bu kısımda model öngörülü kontrolün bir GUI sisteminde nasıl oluşturulduğu ve planlandığından bahsedilecektir.

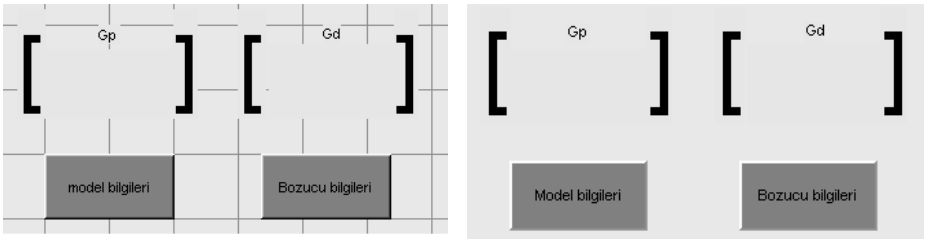
III.1.3.1 Sistem Bilgilerinin GUI Üzerinde Tasarlanması

Şekil III.5'te tek girişli tek çıkışlı bir sistemin kısıtlamasız model öngörülü kontrolü arayüzünün GUI de tasarımının tamamlanmış hali gözükmektedir.



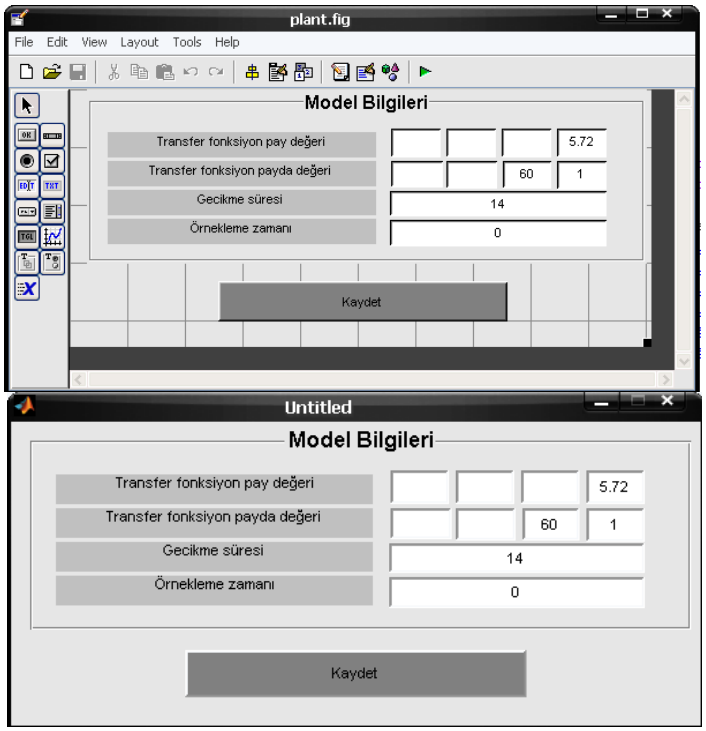
Şekil III.5 Tek Girişli Tek Çıkışlı Bir Sistemin Kısıtlamasız Model Öngörülü Kontrolü Arayüzü Taslağı

Arayüzü oluşturtan birimler ayrı ayrı incelendiğinde ilk olarak sistemin matematiksel modelini yansıtan transfer fonksiyonlarını ve ölçülebilir bozucunun transfer fonksiyonlarının arayüze giriş bölümleri oluşturulmuştur. Bu bölümde sistemin model bilgilerinin giriş yapılacağı bir buton yine bozucu sistemin bilgilerinin giriş yapılacağı buton konulmuştur.



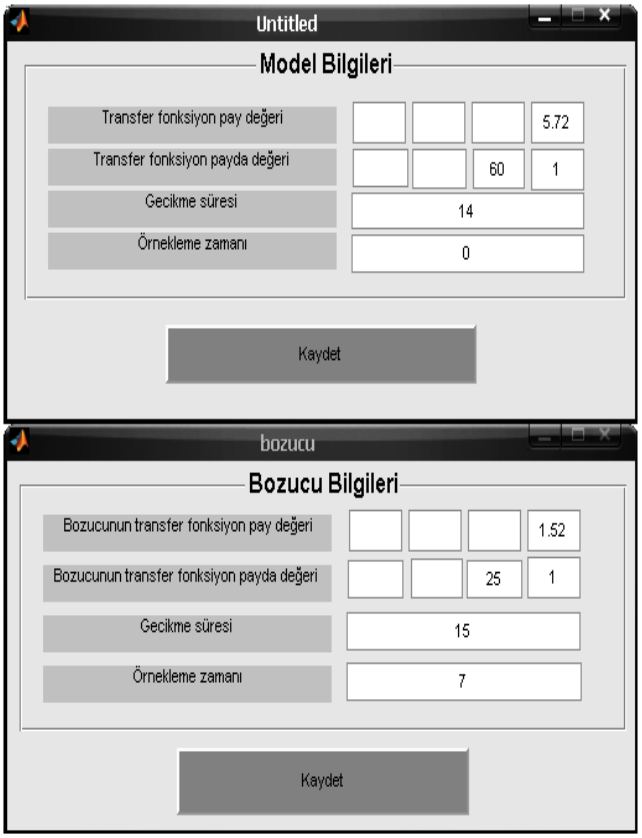
Şekil III.6 Sistemin ve Bozucuların Modellerinin Bilgileri

Şekil III.6 gözüken GUI'nin pasif ve aktif durumundaki sisteme ve bozuculara ait bilgilerin girilebileceği ve poly formatında gözlemlenebileceği kısımlar gözükmektedir.



Şekil III.7 Sistem Modellerinin Girileceği GUI Penceresi

Bu pencerelerde sisteme ait bilgilerin üçüncü dereceden bir transfer fonksiyonun yazımına uygun olarak tasarlanmıştır. Tasarlan bu sistemde transfer



Şekil III.8 Bozucu Modellerinin Girileceği GUI Penceresi

fonksiyon pay değeri için bırakılan boşluklar a ile ifade edildiğinde $a_4s^3 + a_3s^2 + a_2s + a_1$ şeklinde payda için ise boşluklar b ile ifade edildiğinde $b_4s^3 + b_3s^2 + b_2s + b_1$ şeklinde gösterilmektedir. Gecikme süresi olarak ifade edilen kısım ise transfer fonksiyonun $g = \frac{a_4s^3+a_3s^2+a_2s+a_1}{b_4s^3+b_3s^2+b_2s+b_1} e^{-ts}$ t ile tanımlanmış kısımdır. Her iki bilgi girişinde de sistem yapıları bu şeklide tanımlanmaktadır. Tanımlanan bu bilgi girişleri ise GUI'nin arka planında aşağıdaki kodlarla gerçekleşir.

```
%Düzenleme kutuları içerisindeki bilgilerin m file üzerinde tanımlanması
pay1=get(handles.pay1, 'String');          payda1 = get(handles.payda1,'String');
pay2 = get(handles.pay2,'String');         payda2 = get(handles.payda2,'String');
pay3 = get(handles.pay3,'String');         payda3 = get(handles.payda3,'String');
pay4 = get(handles.pay4,'String');         payda4 = get(handles.payda4,'String');
delay1 = get(handles.delay1,'String');     delt1 = get(handles.delt1,'String');
%Transfer fonksiyonunun oluşturulması
NUM1=[str2num(pay4) str2num(pay3) str2num(pay2) str2num(pay1)];
```

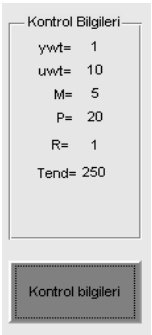
```
DEN1=[str2num(payda4) str2num(payda3) str2num(payda2) str2num(payda1)];
delt1=str2num(delt1);
delay1=str2num(delay1);
% Poly formatına dönüştürülüp ana sayfaya aktarılması
gp=poly2tfd(NUM1,DEN1,delt1,delay1);
gp=num2str(gp)
set(bas1GUIdata.iki,'string',gp)
```

Şekil III.9 Sistem Bilgilerinin Uygulama Penceresinde Görünümü

Bu tanımlamalar hem model hemde bozucular için gerçekleştiğinde sistem bilgileri yukardaki şekildeki gibi gözükmektedir.

III.1.3.2 Kontrol bilgilerinin GUI üzerinde tasarlanması

Sistemin kontrol bilgilerinin giriş yapıldığı kısım ise ana sayfa üzerinde şekil III.10'daki gibi tasarlanmıştır.



Şekil III.10 Kontrol Bilgilerinin Uygulama Penceresi Üzerindeki Görünümü

Kontrol bilgileri butonuna basıldığında kontrol değişkenleri adlı sayfa açılır bu sayfa üzerinde tanımlanan MPC kontrol sisteminde yer alan bilgilerin girişi yapılır. Bu bilgi girişlerinin tanımı yine düzenleme kutuları üzerinde tanımlanan ve

GUI'nin arka planında bulunan m file üzerinde kayıt edilir. Kayıt edilen bu bilgiler ana penceredeki Kontrol bilgileri kısmında da gözükmektedir.

Şekil III.11 Kontrol Bilgilerinin Girildiği GUI Pencereleri

Kontrol değişkenleri penceresindeki düzenleme kutuları üzerinde tanımlanan kontrol birimleri ve bunların ana pencerede gösterilmesini sağlayan kodlar aşağıdaki gibi m file üzerine yazılmıştır.

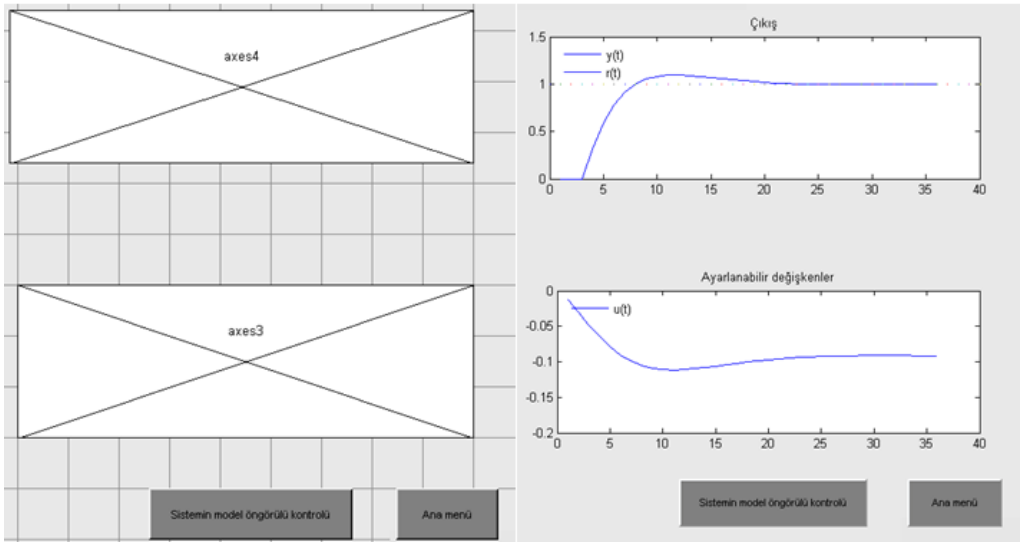
```
ywt = get(handles.ywt,'String');        set(bas1GUIdata.ywt,'string',ywt)
P = get(handles.P,'String');            set(bas1GUIdata.P,'string',P)
uwt = get(handles.uwt,'String');        set(bas1GUIdata.uwt,'string',uwt)
M = get(handles.M,'String');            set(bas1GUIdata.M,'string',M)
P = get(handles.P,'String');            set(bas1GUIdata.P,'string',P)
```

tend= get(handles.tend,'String'); set(bas1GUIdata.tend,'string',tend)

r = get(handles.r,'String'); set(bas1GUIdata.r,'string',r);

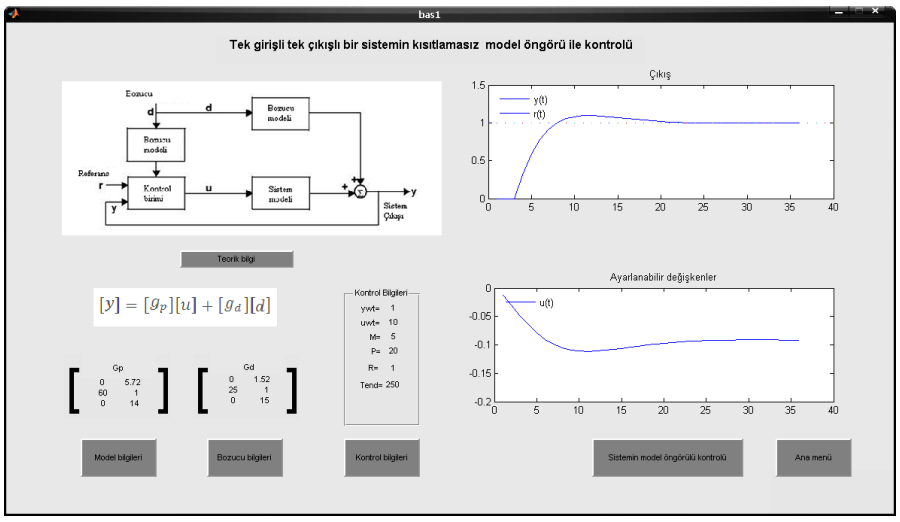
III.1.3.3 Model Öngörülü Kontrol İşleminin Gerçekleşeceği Ve Gözlemleneceği GUI Arayüzü Tasarımı

Tek giriş tek çıkışlı bir sistemin tüm bilgileri ana pencere üzerinde oluşturulduktan sonra model öngörülü kontrolün yapılması ve bu kontrol sürecinin grafiklerinin gösterimi ise GUI de axes olarak tanımlanan grafik ve şekil göstericileri tarafından gerçekleşmektedir. Aşağıdaki şekil III.12'de 2 adet axes grafiği ve 2 adet buton bulunmaktadır. Sistemin model öngörülü kontrolü adıyla tanımlanan butonda tüm sistemin bilgilerine göre gerçekleşen model öngörülü kontrol işlemi yapılmaktadır. Bu işlem GUI'nin arka planında çalışan m file üzerinde gerçekleşmektedir. M file üzerindeki bu kodlamalar ek 1, 1-1, 1-2'de açıklanmıştır. Ana menü butonu ise bu çalışmayı kapatır ve model öngörülü kontrol uygulamaların yer aldığı açılış sayfasına geri döner.



Şekil III.12 GUI Uygulama Arayüzdendeki MPC İşleminin Gerçekleştirilmesi

Yapılan arayüz çalışması çalıştırıldığında ise aşağıdaki gibi gözükmektedir. Bu uygulamada tek girişli tek çıkışlı sistemlerin kısıtlamasız model öngörülü kontrol yöntemiyle kontrol işlemi gerçekleşmiştir.



Şekil III.13 Tek Girişli Tek Çıkışlı Bir Sistemin Kısıtlamasız Model Öngörülü Kontrol Eğitim ve Uygulama Arayüzü

III.2 TEK GİRİŞLİ TEK ÇIKIŞLI BİR SİSTEMİN KISITLAMALI MODEL ÖNGÖRÜLÜ KONTROLÜ

Bir önceki uygulamada yapılan tek girişli tek çıkışlı bir sistemin kısıtlamasız model öngörülü kontrolü uygulamasında kontrol sürecini biraz daha istenilen yönde müdahale edebilen yani ayarlanabilir değişkenler ve sistem çıkışı üzerinde belirli sınırlamalar, kısıtlamalar getirilerek kontrol sürecini daha verimli bir şekilde gerçekleştirilebilir. Bu sistemi uygulamada tek giriş tek çıkışlı bir sistemin kısıtlamalı model öngörülü kontrolü uygulamasında şeklinde gerçekleşecektir.

Yapılan bu çalışmada tek girişli tek çıkışlı bir sistemin kontrolüne ilişkin bir önceki uygulamaya ek olarak kontrol sistemine kısıtlamalar eklenmiştir. Bu kısıtlamalar ayarlanabilir değişken kısıtlamaları $u_{min} \leq u\ k+l\ \leq u_{max}$ şeklinde çıkış kısıtlamaları ise $y_{min} \leq y\ k+1\ \leq y_{max}$ şeklinde ifade edilebilir. Kısıtlamalı kontrolde MPC kazanç katsayısı, öngörü ufku, kontrol ufku, sistem modelleri, kısıtlamalar ve giriş çıkış ağırlık matrisleri bir bütün olarak kontrol edilir. Kısıtlamasız kontrol sistemlerinde ise ilk olarak kazanç elde edilir daha sonra kontrol sistemine eklenirdi. Kısıtlamalı kontrolde ise ayarlanabilir değişkenlerinde kısıtlanması ve bunun kazanç katsayısını etkilemesinden dolayı tüm kontrol işlemi bir bütün olarak yapılır.

Kısıtlamalı model öngörülü kontrol işlemin yapıldığı MPC Toolbox'ı bu işlemleri cmpc komutun içerisindeki algoritmalara sistem bilgilerini gömerek gerçekleştirmektedir. Cmpc komutunun çalışması ve kullanılmasıyla ilgili bilgi bölüm II'de anlatılmıştır. Cmpc komutunu kontrol sistemi için uygulanması aşağıdaki gibi gerçekleşir.

$[y\ u] = cmpc(\mathrm{G_p}plant, \mathrm{G_p}model, ywt, uwt, M, P, tend, r, ulim, ylim, \ldots$

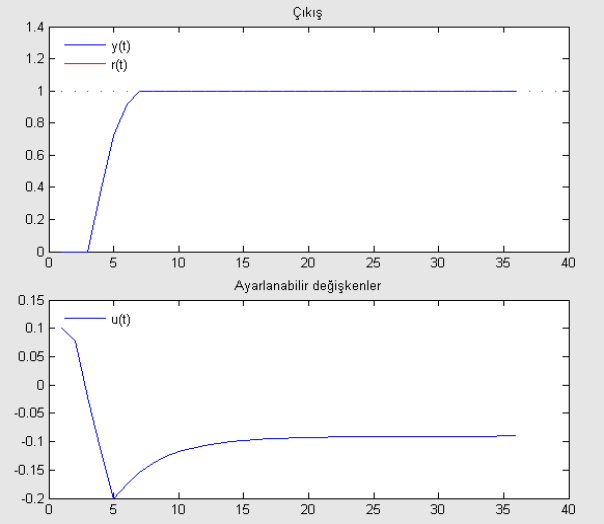
$\mathrm{G_d}plant, \mathrm{G_d}model)$

Komut içerisindeki bilgiler birinci uygulama ile aynıdır. Burada ek olarak kontrol sistemine eklenen ulim, ylim ifadeleri kısıtlamaları göstermektedir. Bu kısıtlamalar aşağıdaki gibi gösterilir.

$ulim = [-0.4 \ \ -inf \ \ 0.1];$

$ylim = [\,]\,;$

Yapılan bu kısıtlama ayarlanabilir değişkenler üzerine yapılmış çıkış üzerinde bir kısıtlama yapılmamıştır. Bir önceki sistemdeki veriler de komut içerisinde kullanıldığında sistem çıkışı ve ayarlanabilir değişkenler üzerindeki süreç aşağıdaki grafikteki gibi gerçekleşir. Buradaki kısıtlama sürecinde birinci saniyede ayarlanabilir değişkene 0.1'lik, ikinci saniyede negatif yönde kısıtlama yapılamamış fakat birinci saniye içerisindeki kısıtlama süreç içersinde kendisinden daha büyük bir değer girilmedikçe devam etmektedir. Üçüncü saniyede ise negatif yönde yapılmayan kısıtlamaya karşı ayarlanabilir değişkenlerin -0.4'lük değerde sınırlandırılmıştır.



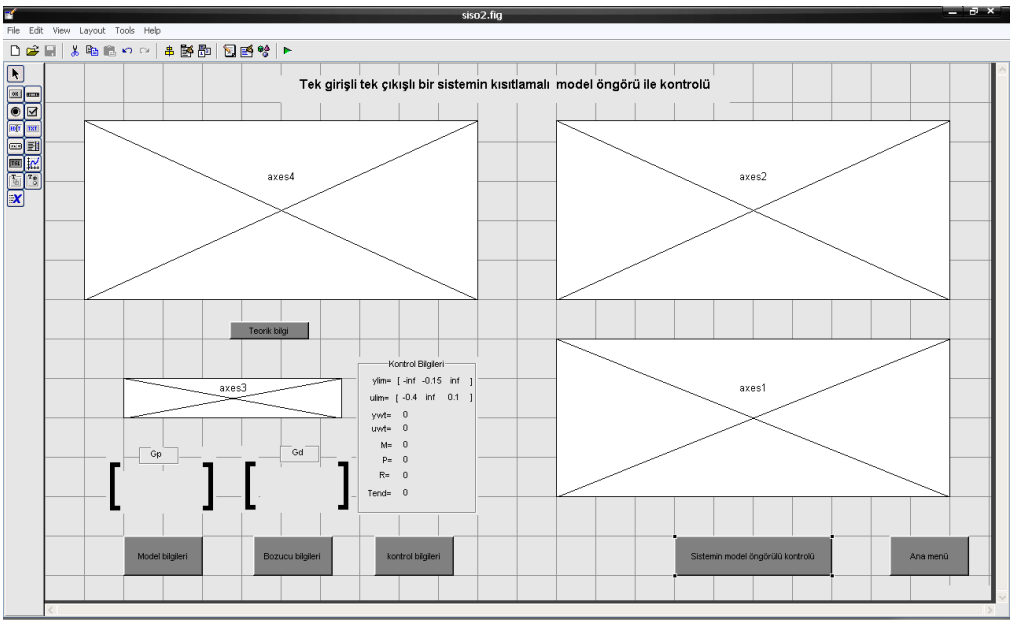
Şekil III.14 Tek Girişli Tek Çıkışlı Sistemin Kısıtlamalı Model Öngörülü Kontrolü

Genel olarak kısıtlama sürecine bakıldığında 0.1 ile -0.4 değerleri arasında kalmaktadır. Bu kısıtlamalara bağlı olarak da kısıtlamalı model öngörülü kontrol işlemi gerçekleşmiştir.

Bu kontrol değişkenleri sisteme uygulandığında model öngörülü kontrol ayarlanabilir değişkenler ve çıkış üzerindeki etkisi yukarıdaki şekilde gözüktüğü gibi gerçekleşir.

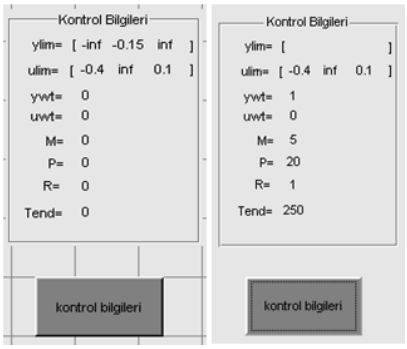
II.2.1 Tek Giriş Tek Çıkışlı Bir Sistemin Kısıtlamalı Model Öngörülü Kontrolü Arayüzü Tasarımı

Bir önceki arayüz çalışmasında yapılan tek girişli tek çıkışlı bir sistemin model öngörülü kontrolü uygulamasından yola çıkılarak kontrol sisteminin daha da gelişmesine katkıda bulunan kısıtlama sürecinin uygulanması ayrı bir arayüz tasarımı gerektirmektedir. Yapılan bu arayüz tasarımı aşağıdaki şekil III.15 üzerinde görülmektedir.

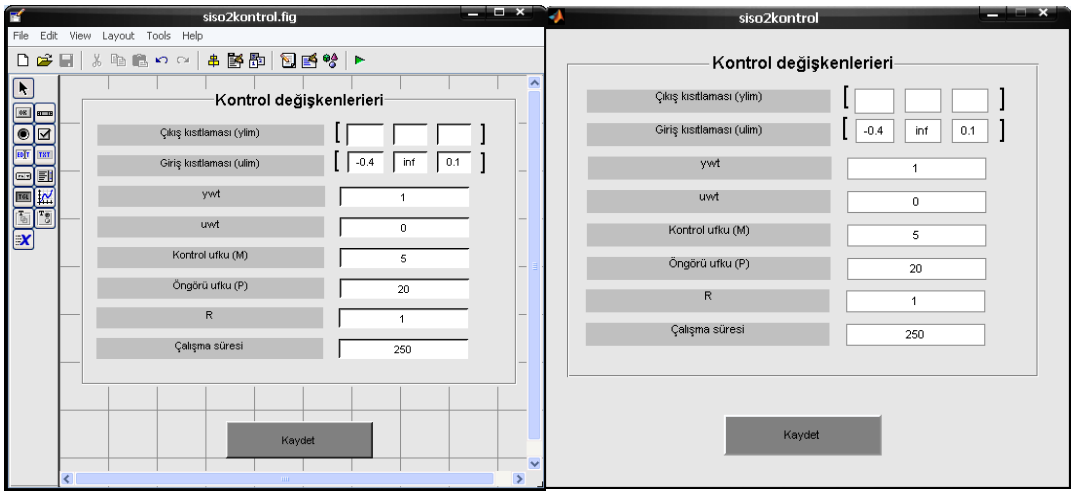


Şekil III.15 Tek Girişli Tek Çıkışlı Bir Sistemin Kısıtlamalı Arayüzü Tasarımı

Yapılan bu arayüz tasarımında bir önceki arayüz tasarımına ek olarak kontrol bilgileri kısmına, ayarlanabilir değişkenler ve çıkış üzerinde etkili olabilecek kısıtlama bilgileri eklenmiştir. Eklenen bu kısıtlamalar aşağıdaki şekil III.16 üzerinde gözükmektedir. Sistemin model ve bozucu bilgileri ise bir önceki yapılan tasarımla aynıdır.



Şekil III.16 Kontrol Bilgilerine Eklenen Kısıtlama Bilgileri



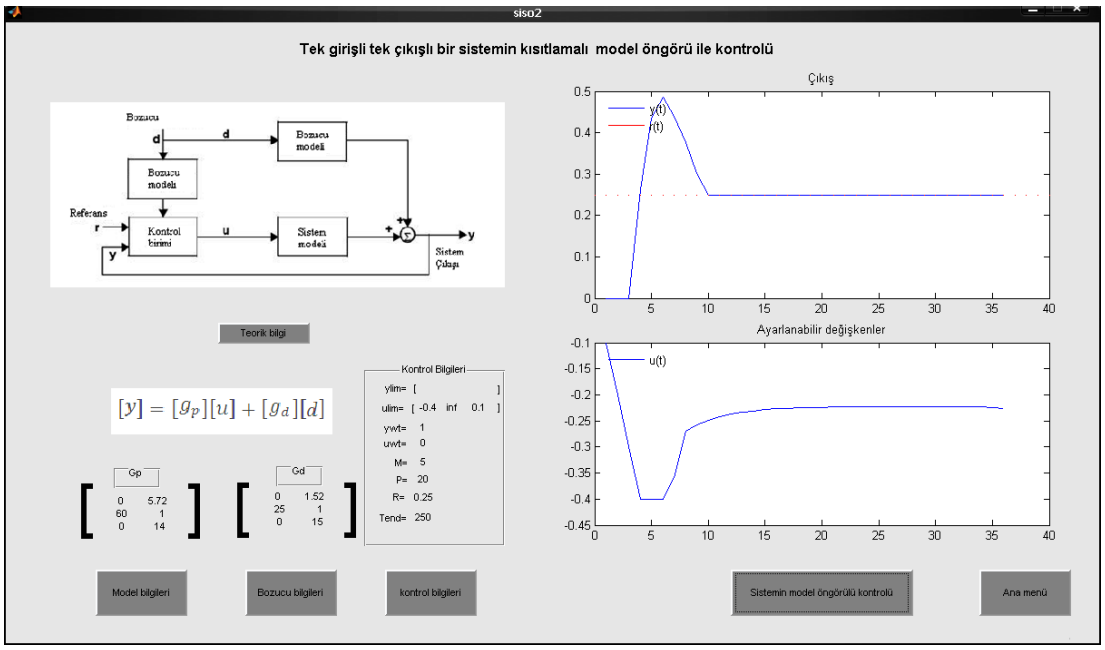
Şekil III.17 Kontrol ve Kısıtlama Bilgilerinin Girileceği GUI Penceresi

Tasarımın GUI arka planındaki programlanması birinci uygulamadaki gibidir. Kontrol sürecine eklenen kısıtlamaların kodları ise aşağıdaki gibidir.

```
yl1 = get(handles.yl1,'String');        yl1=str2num(yl1);
yl2 = get(handles.yl2,'String');        yl2=str2num(yl2);
yl3 = get(handles.yl3,'String');        yl3=str2num(yl3);
ul1 = get(handles.ul1,'String');        ul1=str2num(ul1);
ul2 = get(handles.ul2,'String');        ul2=str2num(ul2);

ul3 = get(handles.ul3,'String');        ul3=str2num(ul3);
ulim=[ul3 ul2 ul1];
ylim=[yl3 yl2 yl1];
```

Burada açılan kontrol değişkenleri penceresinde kısıtlamalarla ilgili veriler ilgili düzenleme kutularında yazılıp kaydedildiğinde, bu veriler ana pencere üzerindeki kontrol bilgileri kısmına yazılmaktadır. MPC kontrol sürecindeki bu veriler de ana pencere üzerindeki bu kısımlar üzerinden alınmaktadır. Bu veriler GUI'nin arka planında çalışan m file üzerinde MATLAB programı tarafından MPC sürecinde kullanılmaktadır. Tasarlanan bu arayüzün çalışmasına ilişkin kodlar ek 2, 2, 2-1 ve 2-2 üzerinde açıklanmışıdır.



Şekil III.18 Tek girişli Tek Çıkışlı Bir Sistemin Kısıtlamalı Model Öngörülü Kontrol Eğitim ve Uygulama Arayüzü

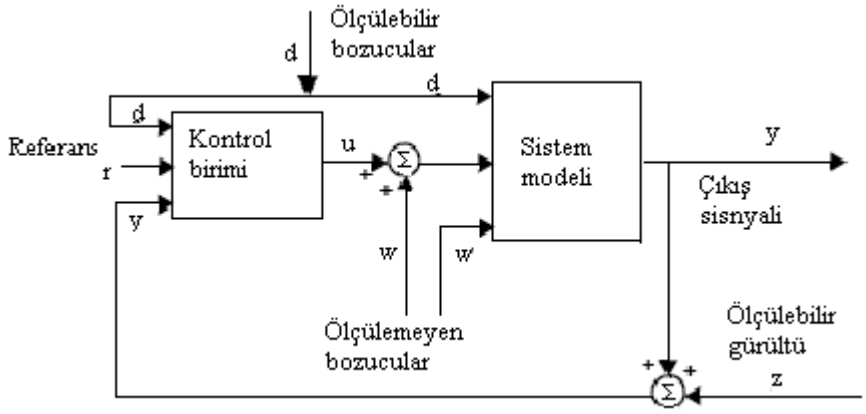
Arayüzün çalışması incelendiğinde ilk olarak kontrol edilecek modele ait bilgilerin girilmesi gerekmektedir. Bu süreçte tasarım üzerinde bulunan model bilgileri adlı butona tıklanıldığında modele ait transfer fonksiyonlarının değerlerini girebileceğimiz bir pencere açılmaktadır. Bu pencereye ilgili veriler yazılıp kayıt edildiğinde MATLAB m file üzerinde hesaplanarak ana pencere üzerinde MPC formatına uygun bir şekilde yazılır. Bozucu modelinin girişinde aynı süreç izlenir. Kontrol bilgilerinin girişinde ise kontrol bilgileri adlı butona tıklanıldığında açılan pencere üzerinde sistemin referans girişleri, ufuklar, kısıtlamalar yer almaktadır. Buradaki verilerde girildiğinde hesaplanmak üzere ana pencere üzerinde yer alır ve

sistemin model öngörülü kontrolü adlı butonu ile de MPC Toolbox içerisinde hesaplamalar yapılır sonuçlar grafiklere yansıtılır.

III.3 ÇOK GİRİŞLİ ÇOK ÇIKIŞLI BİR SİSTEMİN KISITLAMASIZ MODEL ÖNGÖRÜLÜ KONTROLÜ

III.3.1 Kontrol Edilecek Sistemin Yapısı

Bundan önceki uygulamalarda yapılan kontrol işlemi tek giriş tek çıkışlı basit sistemler üzerinde uygulanmıştır. Bu çalışmada ise model öngörülü kontrol işleminin daha kapsamlı ve karışık sistemlere yani sisteme birden fazla girdinin ve çıktının olduğu sistemlere nasıl uygulandığını ve süreç içersindeki kontrolün nasıl gerçekleştiği üzerinde durulacaktır. İlk olarak sistemin blok diyagramını şekil III.19 üzerinde incelediğimizde sisteme etki edenden referans girişleri, ölçülebilir bozucular, ölçülemeyen bozucular, ölçülebilir gürültüler yer almaktadır.



Şekil III.19 Çok Girişli Çok Çıkışlı Bir Sistemin Yapısı

Kontrol birimine yani ayarlanabilir değişkenler olarak tanımlanan kontrol oluşturulduğu birime etki eden sinyaller incelendiğinde referans girişleri, ölçülebilir bozucular ve geri bildirim sinyallerinden oluşmaktadır. Kontrol birimi bu sinyalleri kendi içerisinde değerlendirerek ayarlanabilir değişkenler sinyalini oluşturmaktadır. Bu sinyal sistem modeline etki ederek istenilen sonuçta çıkış vermeyi hedeflemektedir. Bu sürece bazen istenmeyen durumlar etki etmektedir. Bunlar ölçülemeyen bozucu ve gürültü sinyalleridir. Bu sinyallerin oluşturduğu etki ancak

geri bildirilim ile düzeltilebilmektedir. Şekil III.19'a göre 2 girişli 2 çıkışlı bir sistemin denklemini çıkardığımızda aşağıdaki sonuç elde edilir.

$$\begin{matrix} Y_1 \\ Y_2 \end{matrix} = \begin{matrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{matrix} u + \begin{matrix} G_{13} \\ G_{23} \end{matrix} d \qquad \text{(III.8)}$$

III.3.1.1 Sistemin Transfer Fonksiyonları

Sistemin blok diyagramından çıkardığımız denklemi incelediğimizde iki girişin uygulanacağı iki sistem modeli vardır. Her girişin kendi sistem modelinin yanında diğer girişe ait sistem modeline de etki etmektedir. Bundan dolayı oluşan sistem modellerinin sayısı 4 adettir. Bu modeller denklem üzerinde G_{11}, G_{12}, G_{21}, G_{22} şeklinde ifade edilmiştir. Sistem modeline etkide eden diğer bir etkende ölçülebilir bozucu sinyalleridir. Bu bozcu sinyaller her girişe ayrı ayrı etki etmektedir. Sistemde iki giriş olduğundan dolayı etki eden bozucu modeli de iki adet olmaktadır. Bu modeller G_{13} ve G_{23} şeklinde ifade edilmiştir. Bu modellerin transfer fonksiyonları aşağıdaki gibidir.

$$G_{11} = \frac{12.8}{16.7s+1}e^{-1s} \quad G_{12} = \frac{-18.9}{21s+1}e^{-3s} \quad G_{13} = \frac{3.8}{14.9s+1}e^{-8s} \qquad \text{(III9. a,b,c)}$$

$$G_{21} = \frac{6.6}{10.9s+1}e^{-7s} \quad G_{22} = \frac{-19.4}{14.4s+1}e^{-3s} \quad G_{23} = \frac{4.9}{13.2s+1}e^{-3s} \qquad \text{(III.9 d,e,f)}$$

Transfer fonksiyonları yerine yerleştirdiğimizde

$$\begin{matrix} Y_1 \\ Y_2 \end{matrix} = \begin{matrix} \frac{12.8}{16.7s+1}e^{-1s} & \frac{-18.9}{21s+1}e^{-3s} \\ \frac{6.6}{10.9s+1}e^{-7s} & \frac{-19.4}{14.4s+1}e^{-3s} \end{matrix} u + \begin{matrix} \frac{3.8}{14.9s+1}e^{-8s} \\ \frac{4.9}{13.2s+1}e^{-3s} \end{matrix} d \qquad \text{(III.10)}$$

denklemini elde edilir.

III.3.1.2 Sistem Transfer Fonksiyonlarının Uygun Standartlara Getirilmesi

Model öngörülü kontrolün temel yapısı hakkında önceki bölümlerde bahsedilmiştir. Çok girişli çok çıkışlı sistemlerde ise model öngörülü kontrol işlemi her giriş için ayrı ayrı yapılmakta fakat tüm sistem bir bütün olarak değerlendirilmektedir. Dikkat edilmesi gereken bir nokta ise tek girişli tek çıkışlı sistemlerde kontrolü adım cevabı model üzerinden gidilmiştir. Çok girişli çok çıkışlı sistemlerde ise girişler birbirlerinin modellerini etkilediklerinden dolayı sistem modellerinde durum uzayı modeli üzerinden gidilecektir. Model öngörülü kontrol işlemini yukarıda özellikleri verilen sistem üzerinden MATLAB MPC Toolbox içerisinde uygulanması ve kontrolü aşağıda açıklanmıştır. İlk olarak sisteme ait

modellerin transfer fonksiyonun MATLAB programının kullanacağı formata dönüştürülmesi gerekmektedir. Bu işlemler aşağıdaki gibi gerçekleşmektedir.

G_{11} için verilen transfer fonksiyonu MATLAB üzerinde $num = 12.8$ $den = 16.7\ 1$ $delt = 0$ $delay = 1$ şeklinde tanımladığında ve bölüm 3'te çalışması anlatılan poly2tfd komutu kullanıldığında aşağıdaki gibi sonuç elde edilir.

$$\mathrm{G}_{11} = poly2tfd([12.8],[16.7\ \ 1],0,1) \quad \text{(III.11)}$$

$$\mathrm{G}_{11} = \begin{matrix} 0 & 12.8 \\ 16.7 & 1 \\ 0 & 1 \end{matrix}$$

Bu işlem diğer transfer fonksiyonlarına da uygulandığında aşağıdaki sonuçlar elde edilir.

$$\mathrm{G}_{11} = \begin{matrix} 0 & 12.8 \\ 16.7 & 1 \\ 0 & 1 \end{matrix} \qquad \mathrm{G}_{12} = \begin{matrix} 0 & -18.9 \\ 21 & 1 \\ 0 & 3 \end{matrix} \qquad \mathrm{G}_{13} = \begin{matrix} 0 & 3.8 \\ 14.9 & 1 \\ 0 & 8 \end{matrix}$$

$$\mathrm{G}_{21} = \begin{matrix} 0 & 6.6 \\ 10.9 & 1 \\ 0 & 7 \end{matrix} \qquad \mathrm{G}_{22} = \begin{matrix} 0 & -19.4 \\ 14.4 & 1 \\ 0 & 3 \end{matrix} \qquad \mathrm{G}_{23} = \begin{matrix} 0 & 4.9 \\ 13.2 & 1 \\ 0 & 3 \end{matrix}$$

Bu bilgileri model öngörülü kontrol işleminin gerçekleşmesi için durum uzay ve mod formatına dönüştürme işlemi yapılmalıdır bu işlem tfd2mod komutu gerçekleşir. Sistem modeli için oluşturulan kısmın hesaplamalarına umod, ölçülebilir bozucular için kısım ise dmod olarak adlandırılmıştır.

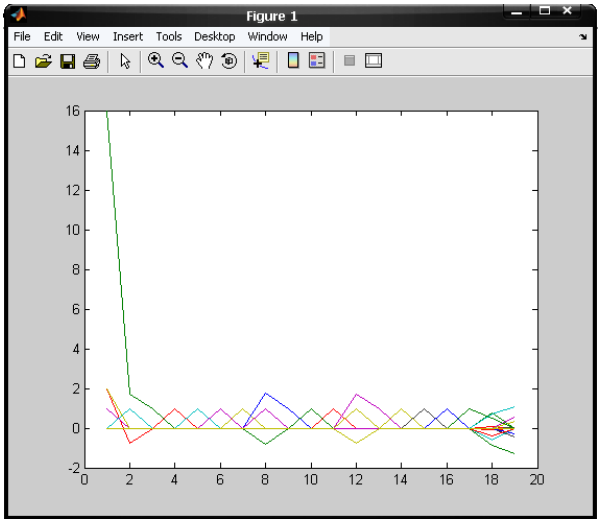
$$umod = tfdmod\ \ delt, ny, \mathrm{G}_{11}, \mathrm{G}_{12}, \mathrm{G}_{21}, \mathrm{G}_{22} \quad \text{(III.12)}$$

$$dmod = tfdmod\ \ delt, ny, \mathrm{G}_{13}, \mathrm{G}_{23} \quad \text{(III.13)}$$

delt sistemin örnekleme zamanı 2 olarak ve ny ayarlanabilir değişken çıkışı iki girişli bir sistem olduğundan dolayı 2 olarak alınmıştır. Bu komut ile sistem mod formatına dönüştürüldüğünde 13x13'lük bir sistem matrisi, ölçülebilir bozucular için ise 9x8'lik sistem matrisi elde edilir. Bu iki matris gurubu MPC Toolbox içerisinde hesaplanırken birleşik olarak hesaplanır. Bu birleştirme işlemini addumd adlı komut gerçekleştirir. Bu komutun uygulanması aşağıdaki gibi gerçekleşir.

$$pmod = addumd(umod, dmod) \quad \text{(III.14)}$$

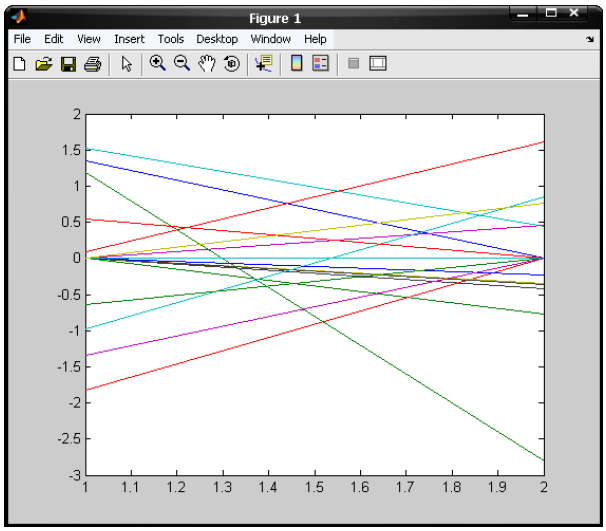
addumd komutunun uygulanmasından sonra sistem 19x20 matrislik MPC kontrol sistemine iletilecek sistem modeli bütünü haline gelir.



Şekil III.20 Sistem Modellerinin Mod Formatında Bütünleştirilmiş Matrisi (pmod)

III.3.1.3 Model Öngörülü Kontrol Kazanç Katsayısının Hesaplanması

MPC Toolbox'da kontrol süreci incelendiğinde sistem kısıtlamasız olduğundan dolayı sistem modeline uygulanacak kazanç katsayısının bulunması gerekmektedir. Kazanç katsayını bulmak için kullanılan komut ise tek giriş tek çıkış sistemlerin kullanılandan farklıdır. Smpccon adıyla tanımlan bu kod içerisinde öngörü ve kontrol ufukları, giriş ve çıkış ağırlık matrisleri ve pmod ismiyle tanımlan sistemin modelinden oluşur. Bu komutun uygulanması aşağıdaki gibi gerçekleşir.



Şekil III.21 MPC Kazanç Katsayısı (Ks)

Ağırlık matrislerini 0 ve öngörü ve kontrol ufuklarını 5 değerlerinde aldığımızda komut içerisindeki işlemler MPC Tollbox içerisindeki algoritmalar tarafından hesaplanır. Hesaplanan kazanç katsayısının grafiği aşağıdaki şekil III.21 üzerinde gözükmektedir.

$$Ks = smpccon(pmod, ywt, uwt, M, P) \quad \text{(III.15)}$$

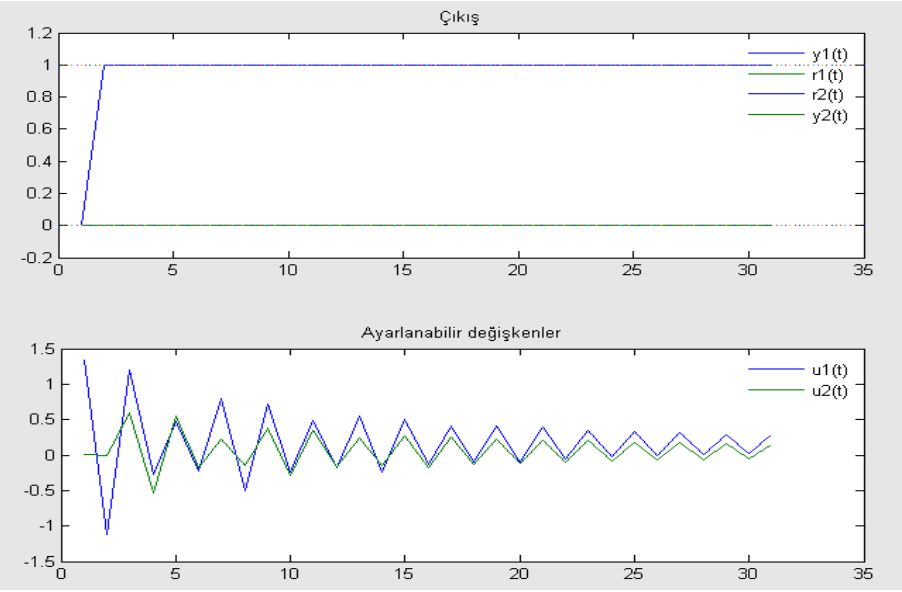
Elde edilen kazanç katsayı matrisi, sayısı 6 adet olan sistem modellerine uygulanacak olan katsayıyı göstermektedir.

III.3.1.4 Model Öngörülü Kontrolün Gerçekleşmesi

Kazanç katsayısı hesaplandıktan sonra model öngörülü kontrol işleminin gerçekleşmesi kontrol sinyali yani ayarlanabilir değişken sinyallerinin sistem modeline uygulanması gerekmektedir. Bu işlemi MATLAB Toolbox içerisinde smpcsim adlı komut tarafından yapılamaktadır. Bu komut sistem ve süreçle ilgili verileri komut içerisindeki algoritmaya gömerek hesaplamaları yapmaktadır. Bu komutun kullanımı aşağıdaki gibidir. Komutun çalışmasıyla ilgili detaylı bilgiler bölüm II'de açıklanmıştır.

```
pmod=imod;
tend=60;
r=[1 0];
```

$$y\,u = smpcsim(pmod, imod, Ks, tend, r) \quad \text{(III.16)}$$

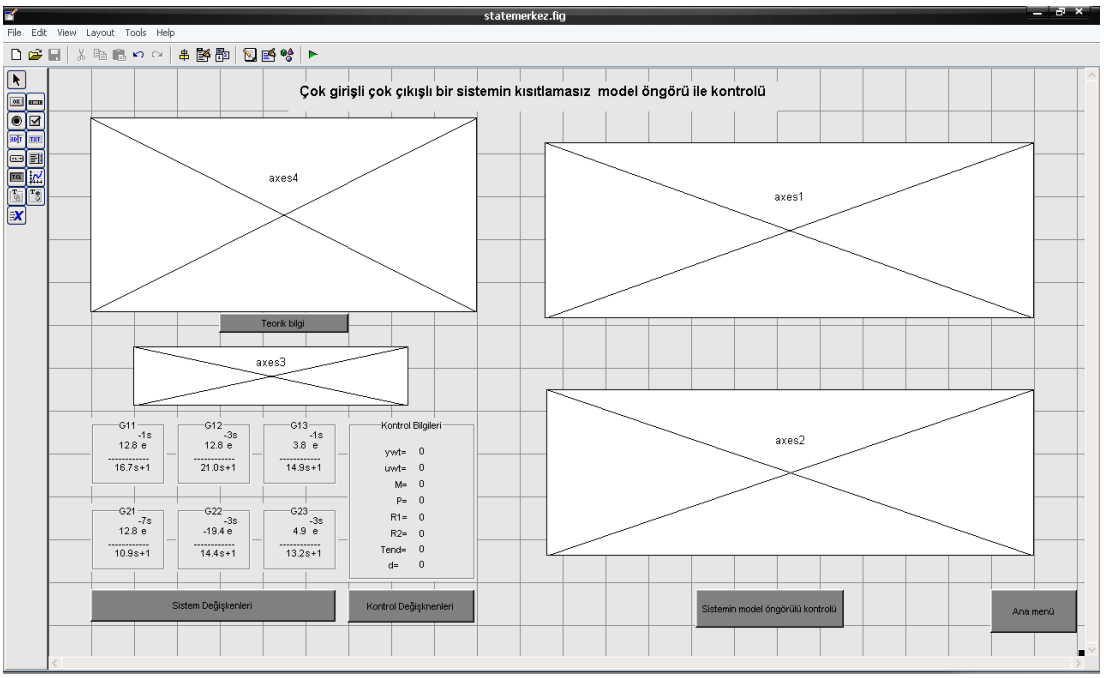


Şekil III.22 Sistemin Model Öngörülü Kontrolü

Sistem modelinin daha verimli çalışabilmesi için MPC Toolbox tarafından pmod, imod ile eşitlenmiştir. Kontrol süreci olarak belirtilen tend 60 saniye, referans girişleri ise [1 0] birinci girişin değeri 1 ikinci girişinki ise 0 olarak değer verilmiştir. İşlemi uyguladığımızda şekil III.22'deki gibi sonuç alırız.

III.3.2 Çok Girişli Çok Çıkışlı Bir Sistemin Kısıtlamasız Model Öngörülü Kontrolü Arayüzü Tasarımı

Çok giriş çok çıkışlı bir sistemin kısıtlamasız model öngörülü kontrolü için hazırlanan arayüzün MATLAB GUI üzerindeki tasarımı aşağıdaki gibidir.



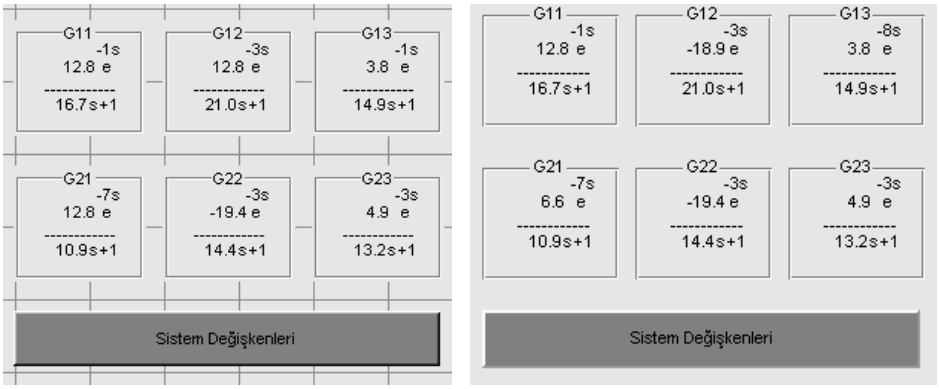
Şekil III.23 Çok Girişli Çok Çıkışlı Bir Sistemin Kısıtlamasız Model Öngörülü Kontrolü Arayüzü Tasarımı

Tasarımın incelendiğinde sistemi oluşturan transfer fonksiyonları, ölçülebilir bozucuların transfer fonksiyonun bulunduğu sistem bilgileri, model öngörülü kontrol bilgilerinin yer aldığı kontrol bilgileri kısmı, model öngörülü kontrolle ilgili teorik bilgi ve blok diyagramının yer aldığı kısım ve model öngörülü kontrol işleminin gözlendiği grafik pencereleri yer almaktadır. Bu bölümler ve tasarımın yapılışı hakkında bilgiler aşağıdaki kısımlarda açıklanmaktadır.

III.3.2.1 Sistem Değişkenleri Bölümü

Arayüz üzerinde tasarlanan, sistemi ve bozucu modellerin transfer fonksiyonun yer alığı kısımlar aşağıdaki şekil üzerinde gözükmektedir. Şeklin sol tarafında bulunan ksım GUI'nin pasif halinde tasarlanan birimleri göstermektedir.Bu kısımda transfer fonksiyonları ve sistem değişkenleri adlı butondan oluşmaktadır. Sağ taraftaki kısım ise GUI'nin aktif durumda iken sistem değişkenleri adlı butona tıklanmasıyla açılan yardımcı pencere üzerinde transfer fonksiyonlarına ait verilerin girilip kayıt edilemesiyle oluşan bilgiler görülmektedir.

Sistem değişkenleri adlı butona tıklandığında açılacak yardımcı pencerenin tasarımı ise şekil III.24'te gözükmektedir. Yardımcı pencere açma ve aralarında verileri paylaşmayla ilgili kodlamalar şu şekildedir.



Şekil III.24 Sistem Değişkenleri Tasarımı

function pushbutton1_Callback(hObject, eventdata, handles) statesistem()

Bu fonksiyonda butona tıklandığında statesistem adlı GUI penceresinin açılması gerçekleşecektir. Veri alış verişinde ise şu kodlamalar yapılır.

statemerkezGUIhandle= statemerkez

statemerkezGUIdata= guidata(statemerkezGUIhandle);

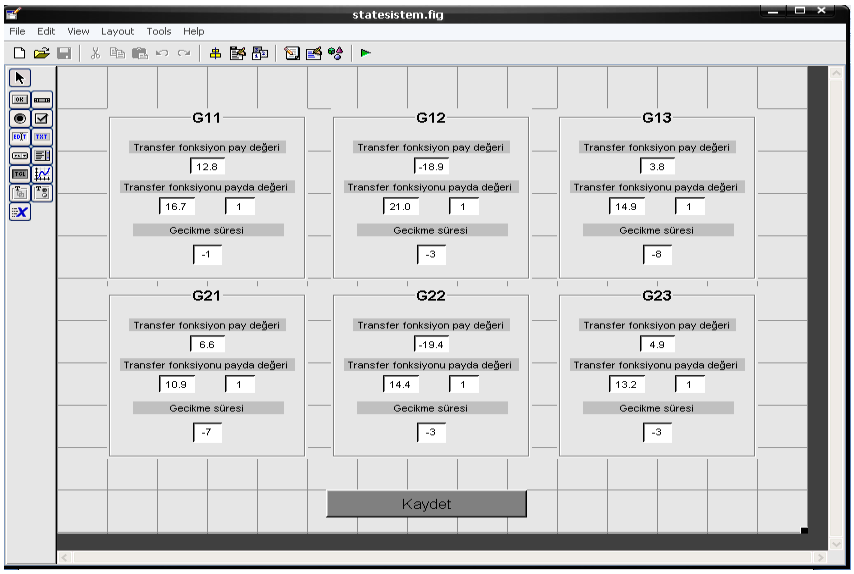
Bu kodlamada ana pencerenin ismi olan statemerkez GUI tasarımı üzerinde veri ilişkilendirilmesi yapılmaktadır. Bu kodlamalar yardımcı pencerenin arka planında çalışan m file üzerinde haberleşmesi yaptırılacak verilerin bulunduğu fonksiyonun başlangıçına yazılırlar. Ana pencereye veri gönderim işlemi ise aşağıdaki şekilde gerçekleşir.

pay1=get(handles.pay1, 'String');

set(statemerkezGUIdata.pay1,'string',pay1)

Burada yardımcı pencere üzerinde pay1 isemiyle hedeflenmiş düzenleme kutularının içeriğindeki bilgiyi ana penceredeki yani statemerkez adlı pencere üzerinde pay1 isimli sabat kutular üzerine gönderilir. Yardımcı pencereden ana pencereye veri transferi kodlama sistemindeki değişiklerle sağlanmış olur.

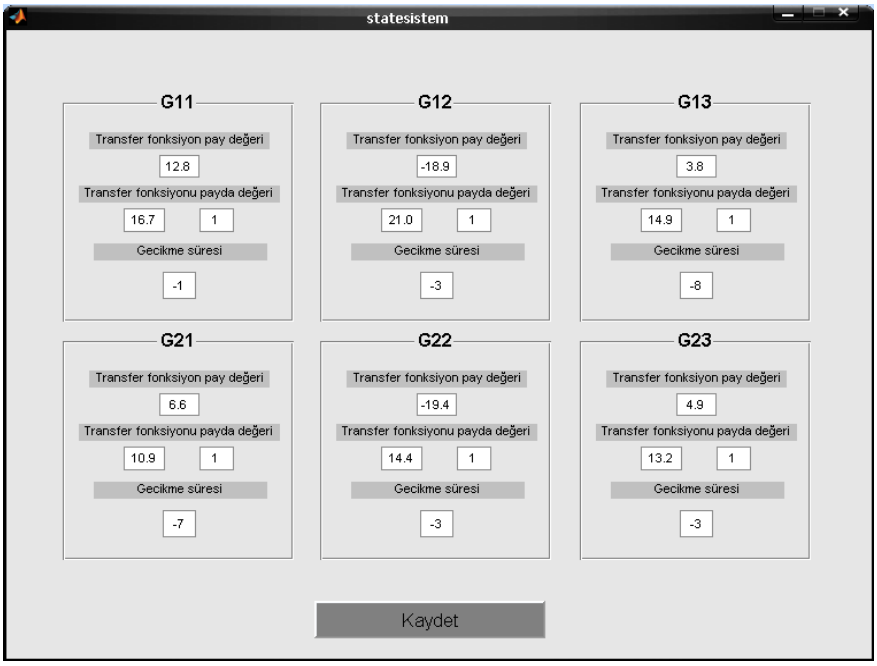
Sistem butonuna tıklandığında açılacak olan yardımcı pencerenin tasarımı ise aşağıdaki şekil III.25 üzerinde gösterilmektedir. Bu pencerede her transfer fonksiyonun pay payda değerleri geçikme süreleri için düzenleme kutuları üzerinden girişler tasarlanmıştır.



Şekil III.25 Sistem Değişkenleri ile İlgili Verililerin Girildiği GUI Penceresi Tasarımı

Sistem değişkenleri penceresinin GUI'nin aktif haldeki görüntüsü ise aşağıdaki şekil III.26'daki gibidir. Bu pencerede sistemin ve bozucunun transfer fonksiyonlarına ait verilerin girişleri yapılıp kayıt ediğinde pencere otomatik olarak kapanacaktır ve girilen verilerin arka planda m file üzerinden ana pencereye gönderilmesi gerçekleşecektir. Bu sürece ilişkin m file üzerindeki kodlamaların bir kısmı aşağıdaki gibidir. Kodlamaların tamamı ek 3 üzerinde verilmiştir.

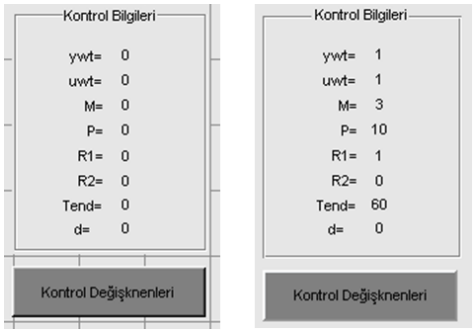
```
statemerkezGUIhandle= statemerkez;
statemerkezGUIdata= guidata(statemerkezGUIhandle);
pay1=get(handles.pay1, 'String');      set(statemerkezGUIdata.pay1,'string',pay1)
pay2=get(handles.pay2, 'String');      set(statemerkezGUIdata.pay2,'string',pay2)
pay3=get(handles.pay3, 'String');      set(statemerkezGUIdata.pay3,'string',pay3)
```



Şekil III.26 Sistem Değişkenlerine Ait Verilerinin Girildiği GUI Penceresinin Aktif Haldeki Görünümü

III.3.2.2 Kontrol Bilgileri Bölümü

Ana pencere üzerinde tasarlanan kontrol bilgileri bölümünde kontrol sürecine ilişkin referans bilgileri, ufuklar, ağırlık matrisleri, çalışma süresi gibi bilgiler yer almaktadır. Ana pencere üzerinde gözlemlenmesi için yer alan bu bilgilerin girişi kontrol değişkenleri adlı butona tıklanarak açılan yardımcı pencere üzerinden gerçekleşmektedir.



Şekil III.27 Kontrol Değişkenlerinin Uygulama Penceresi Üzerindeki Görünümü

Şekil III.28 Kontrol Değişkenlerine Ait Verilerin Girildiği GUI Penceresi

Kontrol değişkenleri adlı butona tıklanıldığında açılacak olan pencerenin tasarımı ve aktif haldeki durumu şekil III.28 üzerinde gösterilmiştir. Ana pencere üzerinde bulunan bilgilerin gözleneceği kısım ve butona tıklanıldığında açılacak olan pencerenin görünüşü verilmiştir. M file üzerindeki kodlamalar ise aşağıdaki gibidir.

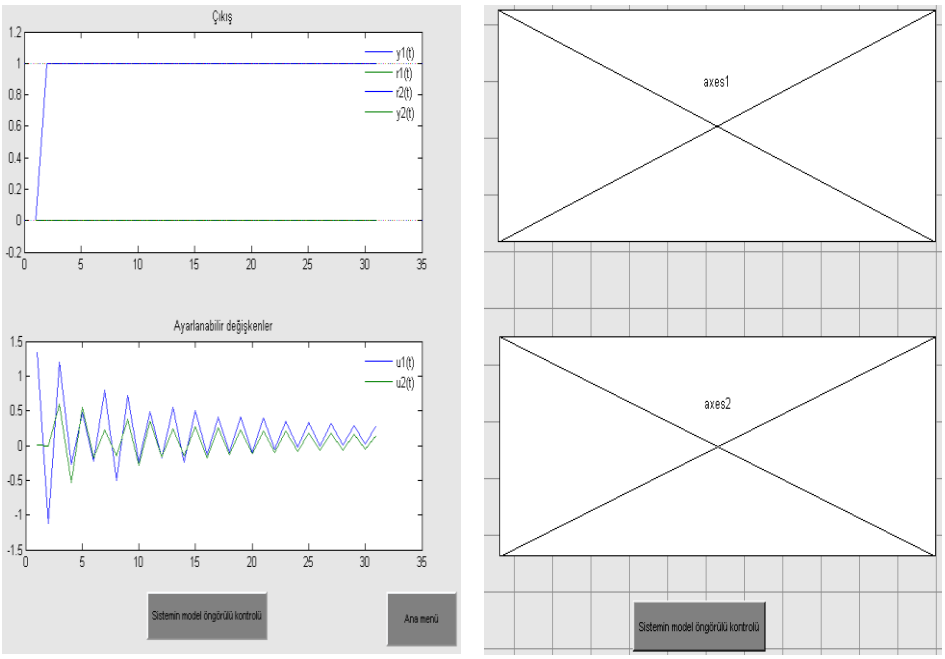
```
statemerkezGUIhandle= statemerkez;
statemerkezGUIdata= guidata(statemerkezGUIhandle);
ywt = get(handles.ywt,'String');        set(statemerkezGUIdata.ywt,'string',ywt)
uwt = get(handles.uwt,'String');        set(statemerkezGUIdata.uwt,'string',uwt)
M = get(handles.M,'String');            set(statemerkezGUIdata.M,'string',M)
P = get(handles.P,'String');            set(statemerkezGUIdata.P,'string',P)
R1 = get(handles.R1,'String');          set(statemerkezGUIdata.R1,'string',R1)
```

```
R2 = get(handles.R2,'String');          set(statemerkezGUIdata.R2,'string',R2)
tend = get(handles.tend,'String');      set(statemerkezGUIdata.tend,'string',tend)
w= get(handles.w,'String');             set(statemerkezGUIdata.w,'string',w)
guidata(statemerkez, statemerkezGUIdata);
close(statekontrol);
```

Bu kodlamalarda yardımcı pencere üzerindeki düzenleme kutuları üzerine girilen verilerin ana penceredeki ilgili kısımlara gönderilmesi gerçekleşmiştir.

III.3.2.3 Model Öngörülü Kontrol İşleminin Gerçekleşmesi

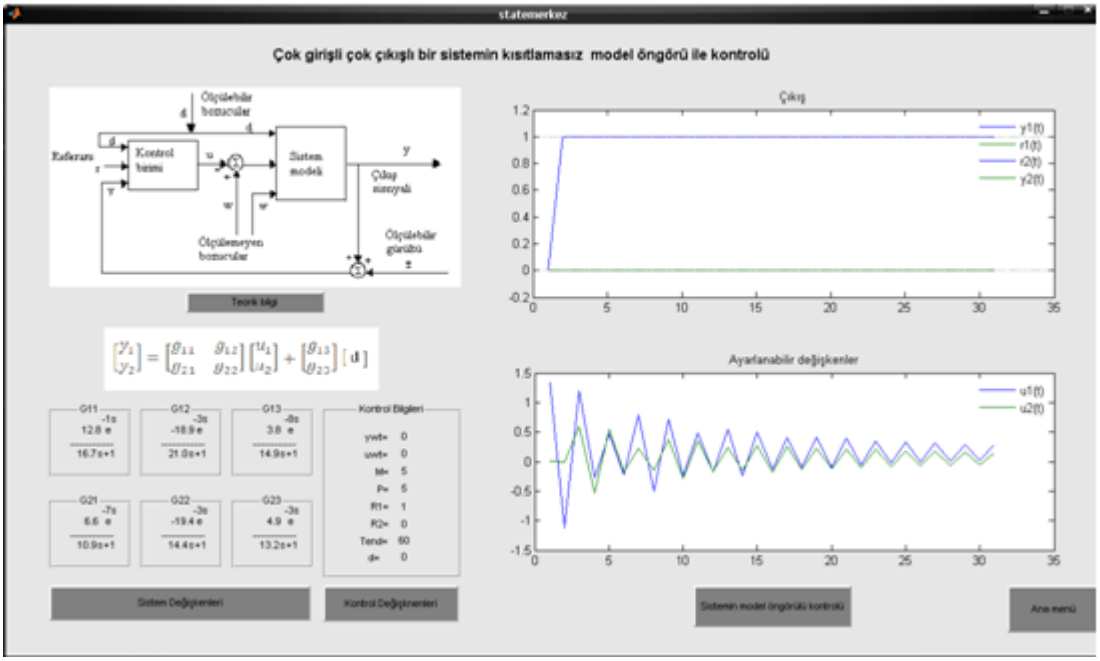
Arayüz tasarımında buraya kadarki yapılan işlemlerde model öngörülü kontrol sürecine ilişkin yer alan model bilgileri ve kontrol sistemine ait bilgileri yardımcı pencereler ile ana pencere üzerinde toplanmıştır. Model öngörülü kontrol işleminin gerçekleşmesi ve bu kontrol sürecinin gözlemlenmesi için ana pencere üzerinde aşağıdaki şekil III.29'daki gibi tasarım yapılmıştır.



Şekil III.29 Model Öngörülü Kontrolün Hesaplanması ve Gösterimi

Ana pencere üzerine yapılan bu tasarım ile GUI'nin aktif haldeki durumu yukarıdaki şekil üzerinde gösterilmiştir. Bu tasarımda sistemin model öngörülü kontrol adlı butona tıklanıldığında arka planda çalışan m file üzerinde sistemin model öngörülü kontrolü hesaplanır ve Axes'lerde gösterilir. M file üzerinde ki kodlar ek 3, 3-1 ve 3-2 üzerinde verilmiştir.

Tasarlanan arayüzün tamamlanmış ve aktif haldeki görünüşü aşağıdaki şekil üzerinde gözükmektedir.



Şekil III.30 Çok Girişli Çok Çıkışlı Bir Sistemin Kısıtlamasız Model Öngörülü Kontrol Eğitim ve Uygulama Arayüzü

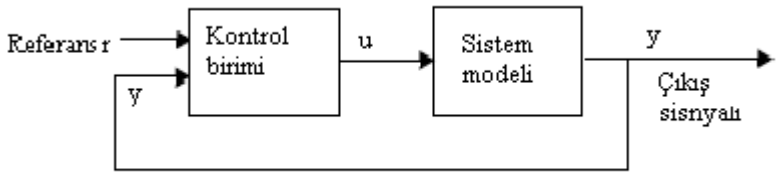
Yapılan bu arayüz tasarımı ile çok girişli çok çıkışlı sistemlerin model öngörülü kontrolü işlemi gerçekleşmiştir. Bu arayüz tasarımı ile farklı sistemlere ait modellerin bilgileri arayüze girilerek ve kontrol seçenekleri seçilerek sistemin model öngörülü kontrolü gerçekleşir.

III.4 ÇOK GİRİŞLİ ÇOK ÇIKIŞLI BİR SİSTEMİN KISITLAMALI MODEL ÖNGÖRÜLÜ KONTROLÜ

III.4.1 Kontrol Edilecek Sistem

Bir önceki uygulamada yapılan çok girişli çok çıkışlı bir sistemin kısıtlamasız model öngörülü kontrolü uygulamasında kontrol sürecini biraz daha istenilen yönde müdahale edebilen yani ayarlanabilir değişkenler ve sistem çıkışı üzerinde belirli sınırlamalar, kısıtlamalar getirilerek kontrol sürecini daha verimli bir şekilde gerçekleştirilebilen sistem tasarlandı. Bu uygulama çok girişli çok çıkışlı bir sistemin kısıtlamalı model öngörülü kontrolü uygulaması ile gerçekleşecektir. Uygulama işleminde kısıtlama işlemi tek girişli tek çıkışlı bir sistemin kısıtlamalı model öngörülü kontrolü uygulamasındaki gibidir. Kısıtlama işlemi ayarlanabilir

değişkenler ve sistem çıkışı üzerinde gerçekleşir. MPC kazanç katsayısı ayrı bir şekilde hesaplamaz tüm sistem bilgileriyle beraber hesaplanır.



Şekil III.31 Kontrol Edilecek Sistemin Blok Diyagramı

Yapılan bu arayüz tasarımında sisteme etki eden ölçülebilir bozucular uygulamaya katılmamıştır. Dolayısıyla sisteme etki eden sadece sistem modelleri ve kısıtlamalardır. Yapılan bu uygulamada iki adet giriş ve iki adet çıkış sistemine göre tasarlanmıştır. Sistemi şekil III.31'deki blok diyagramından yola çıkılarak basit bir şekilde aşağıdaki denklemdeki gibi ifade edilebilir.

$$\begin{matrix} Y_1 \\ Y_2 \end{matrix} = \begin{matrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{matrix} u \qquad \text{(III.17)}$$

III.4.1.1 Sistemin Yapısı

Sisteme ait model bilgileri bir önceki uygulamadaki bilgiler ile aynıdır fakat sisteme etkide eden ölçülebilir bozucular yapılan bu uygulamada sistem dışında bırakılmıştır. Sistemin blok diyagramından çıkardığımız denklemi incelediğimizde iki girişin uygulanacağı iki sistem modeli vardır. Her girişin kendi sistem modelinin yanında diğer girişe ait sistem modeline de etki etmektedir. Bundan dolayı oluşan sistem modellerinin sayısı 4 adettir. Bu modeller denklem üzerinde G_{11}, G_{12}, G_{21}, G_{22} şeklinde ifade edilmiştir. Bu modellerin transfer fonksiyonları aşağıdaki gibidir.

$$G_{11} = \frac{12.8}{16.7s+1} e^{-1s} \qquad G_{12} = \frac{-18.9}{21s+1} e^{-3s} \qquad \text{(III.17 a,b)}$$

$$G_{21} = \frac{6.6}{10.9s+1} e^{-7s} \qquad G_{22} = \frac{-19.4}{14.4s+1} e^{-3s} \qquad \text{(III.17 c,d)}$$

$$\begin{matrix} Y_1 \\ Y_2 \end{matrix} = \begin{matrix} \frac{12.8}{16.7s+1} e^{-1s} & \frac{-18.9}{21s+1} e^{-3s} \\ \frac{6.6}{10.9s+1} e^{-7s} & \frac{-19.4}{14.4s+1} e^{-3s} \end{matrix} u \qquad \text{(III.18)}$$

Modellere ait verilen bu transfer fonksiyonları bir önceki uygulamadaki veriler ile aynıdır. Bu verilerin MATLAB formatına uygun hale getirilme şekilleri de aynı işlemlerle gerçekleşir. Bu işlemler bir önceki uygulamayla aynı olduğundan dolayı anlatılmamıştır.

III.4.1.2 Sistemin Model Öngörülü Kontrolü

Kısıtlamalı kontrolde kısıtlamasıza göre kontrol sistemine eklenen kısıtlama bilgileri yer almaktadır. Bu kısıtlama bilgilerinin çalışma şekli tek girişli tek çıkışlı bir sistemin kısıtlamalı model öngörülü kontrolündeki gibidir. Verilen her kısıtlama bilgileri ayarlanabilir değişkenler ya da çıkış üzerinde zaman sıralamasına göre sınır getirerek gerçekleşir. Sisteme uygulanacak olan kısıtlama ise aşağıdaki gibidir.

$ulim = [-inf \;\; -0.5 \;\; inf \;\; inf \;\; 0.1 \; 100];$ *(III.19)*

$ylim = [\,]\,;$ (III.20)

Ayarlanabilir değişkenler üzerine yapılan bu kısıtlama denklem III.19 üzerinde gözükmektedir. Çıkış üzerinde ise her hangi bir kısıtlama yapılmamıştır.

Sistemin kontrolü MPC Toolbox içerisindeki şu komutlar ile gerçekleşir.

$imod = tfd2mod(delt, ny, g11, g21, g12, g22);$

$pmod = imod;$

$ywt = \;\; 0\;;$

$uwt = \;\; 0\;;$

$M = 2;$

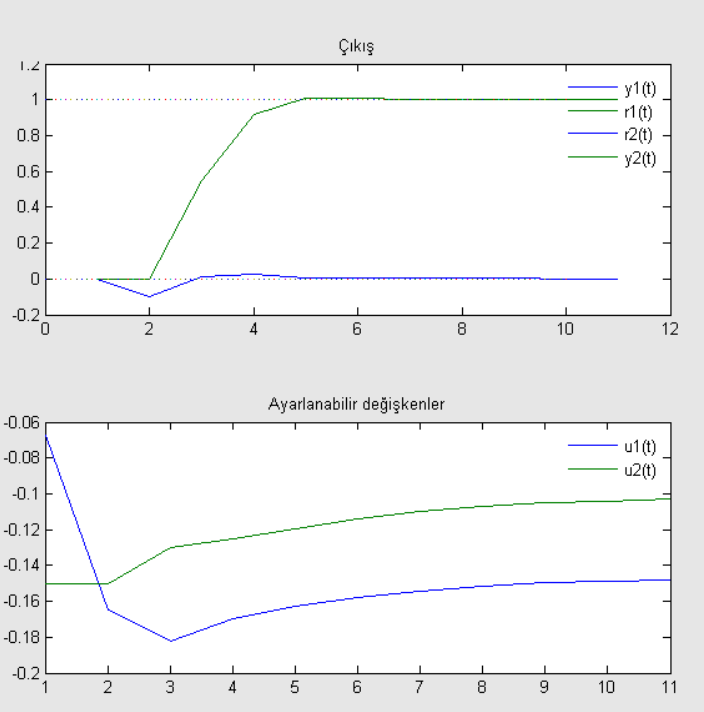
$P = 6;$

$tend = 30;$

$r = [0\;1];$

$[y, u] = scmpc(pmod, imod, ywt, uwt, M, P, tend, r, ulim, ylim);$

İlk olarak sistemi oluşturan transfer fonksiyonlarının MATLAB formatına dönüştürülmesi ve bütünleşik hale getirilmesiyle gerçekleşir. Bütünleşik hale getirilen model bilgileri imod şeklinde tanımlanmıştır. Toolbox içerisinde pmod olarak adlandırılan kısım ise imod ile aynıdır. Bu eşitlik kontrolün daha verimli gerçekleşmesini sağlar. ywt ve uwt değerleri ise bu kontrol sisteminde 0 olarak tanımlanmıştır. Kontrol ufku 2, öngörü ufku ise 6, çalışma süresi 30s ve giriş değerleri birinci giriş 0, ikinci giriş değeri ise 1 olarak tanımlanmıştır. Kontrol işlemi gerçekleştirildiğinde aşağıdaki grafikteki sonuç elde edilir.

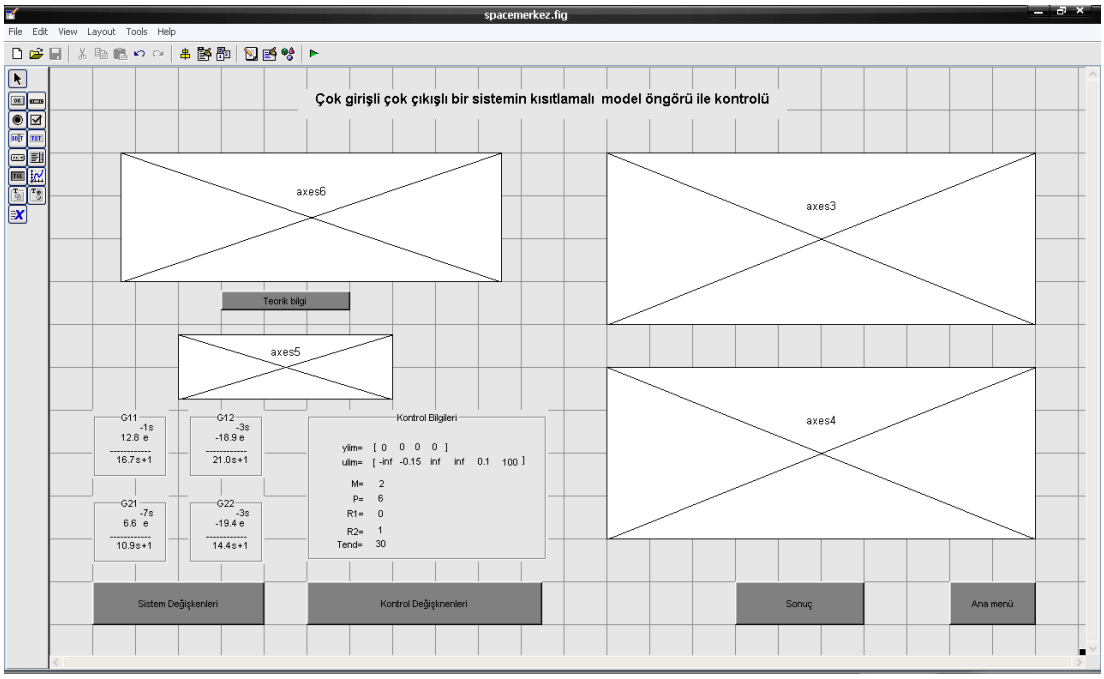


Şekil III.32 Çok Girişli Çok Çıkışlı Bir Sistemin Kısıtlamalı Model Öngörülü Kontrolü

III.4.2 Çok Girişli Çok Çıkışlı Bir Sistemin Kısıtlamalı Model Öngörülü Kontrolü Arayüzü Tasarımı

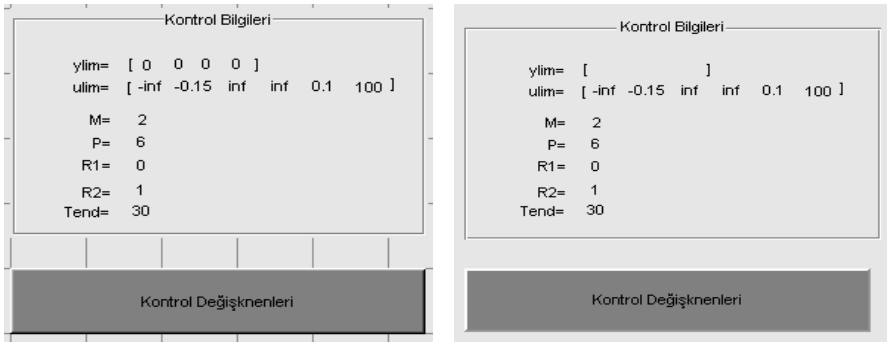
Bir önceki arayüz çalışmasında yapılan çok girişli çok çıkışlı bir sistemin model öngörülü kontrolü uygulamasından yola çıkılarak kontrol sisteminin daha da gelişmesine katkıda bulunan kısıtlama sürecinin uygulanması ayrı bir arayüz tasarımı gerektirmektedir. Yapılan bu arayüz tasarımı aşağıdaki şekil üzerinde gözükmektedir.

Yapılan bu arayüz tasarımında bir önceki arayüz tasarımına ek olarak kontrol bilgileri kısmına, ayarlanabilir değişkenler ve çıkış üzerinde etkili olabilecek kısıtlama bilgileri eklenmiştir. Eklenen bu kısıtlamalar aşağıdaki şekil III.34 üzerinde gözükmektedir. Sistemin model bilgileri ise bir önceki yapılan tasarımla aynıdır.



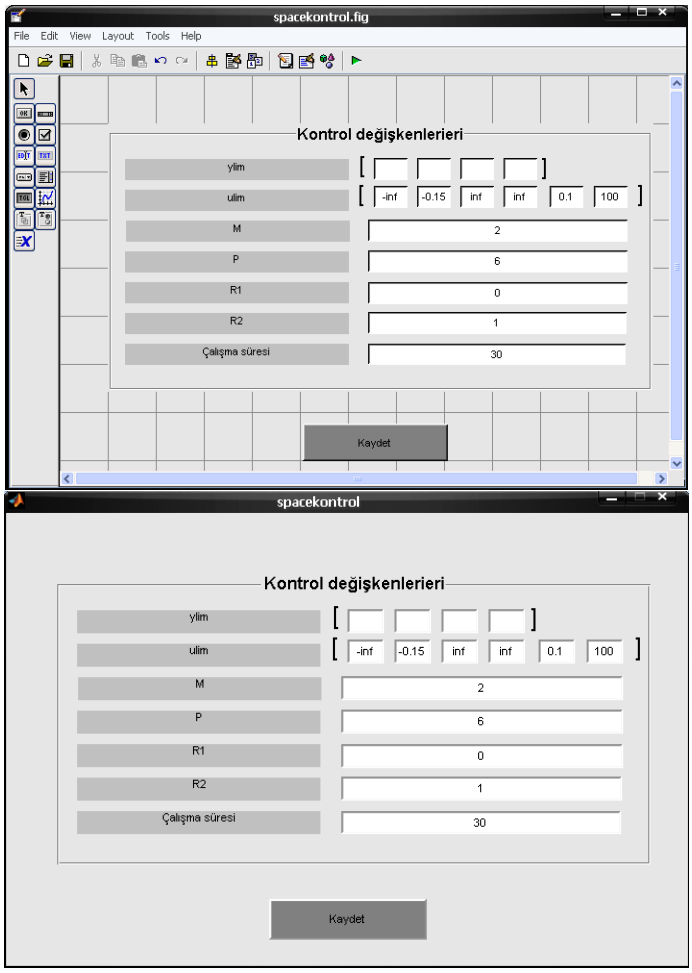
Şekil III.33 Çok Girişli Çok Çıkışlı Bir Sistemin Kısıtlamalı Model Öngörülü Kontrolü Arayüzü Tasarımı

Kontrol sistemiyle ilgili verilerin düzenlenmesinde ise aşağıdaki gibi tasarım yapılmıştır. Şekil III.34 üzerinde gözüken GUI tasarımında ana pencere üzerinde kontrol bilgilerini gösteren veri gösterim bölümü ve bu verilerin girilebildiği yardımcı pencerenin açılmasını sağlayacak bir buton tasarlanmıştır.



Şekil III.34 Uygulama Penceresindeki Kontrol Değişkenleri Görünümü

Kontrol değişkenleri adlı butona tıklanıldığında açılacak olan yardımcı pencerenin tasarımı ve GUI'nin aktif durumundaki görünüşü ise aşağıdaki şekil III.35 üzerinde gösterilmiştir.



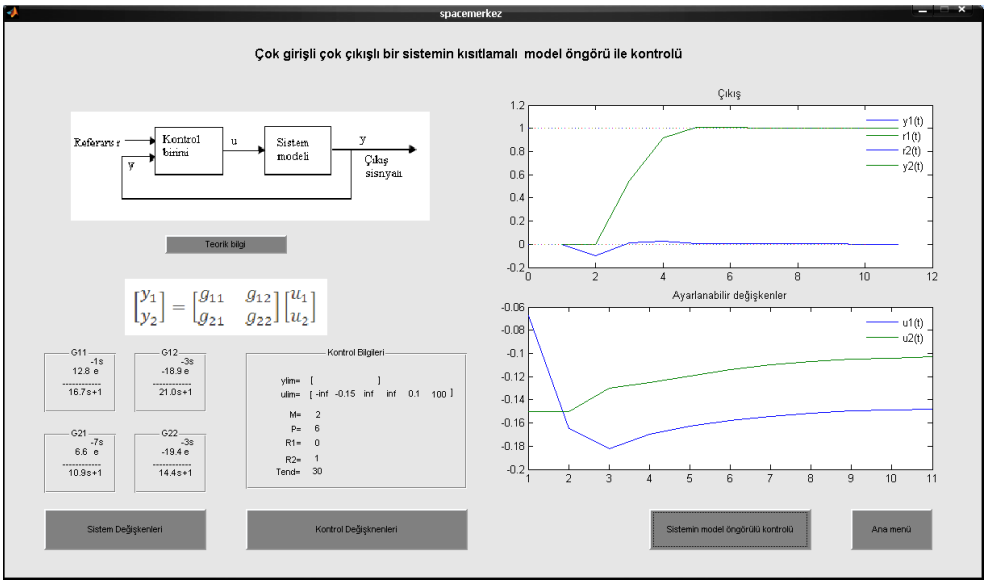
Şekil III.35 Kontrol Değişkenlerine ve Kısıtlamalara Ait Bilgilerin Girildiği GUI Penceresi

Tasarım penceresinde model öngörülü kontrolü etkileyen öngörü ve kontrol ufukları, ayarlanabilir değişken ve çıkış kısıtlamaları, referans girişleri ve çalışma süresi gibi bilgiler yer almaktadır. Bu bilgilerin girişinde GUI programının düzenleme kutuları ara yolundan faydalanılmıştır. Düzenleme kutuları üzerine girilen bu bilgiler arka planda çalışan m file üzerinden işlenmektedir. Bu tasarımın amacı kontrol bilgilerinin rahatlıkla girilebileceği ayrı bir kısım ve bu bilgilerin ana pencere üzerinde gözlenebileceği bir bölüme aktarmasıdır.

Bu tasarıma ait m file üzerine yazılan kodların bir kısmı aşağıdaki gibidir. Kodların tamamı ek 4, 4-1 ve 4-2 üzerinde verilmiştir.

```
spacemerkezGUIhandle= spacemerkez;
spacemerkezGUIdata= guidata(spacemerkezGUIhandle);
 y1 = get(handles.y1,'String');          set(spacemerkezGUIdata.y1,'string',y1)
u5 = get(handles.u5,'String');           set(spacemerkezGUIdata.u5,'string',u5)
M = get(handles.M,'String');             set(spacemerkezGUIdata.M,'string',M)
 P = get(handles.P,'String');            set(spacemerkezGUIdata.P,'string',P)
R1 = get(handles.R1,'String');           set(spacemerkezGUIdata.R1,'string',R1)
R2 = get(handles.R2,'String');           set(spacemerkezGUIdata.R2,'string',R2)
tend = get(handles.tend,'String');       set(spacemerkezGUIdata.tend,'string',tend)
guidata(spacemerkez, spacemerkezGUIdata);
close(spacekontrol);
```

Yardımcı pencerenin veriler girilip kaydedildikten sonra pencere kendiliğinden kapanacaktır ve ana pencereye dönecektir. Ana pencere üzerinde sistem modeline ait bilgiler de girildiğinde sistem model öngörülü kontrole hazır hale gelir. Sistemin çalışmasını gözlemlemek için tasarım üzerinde buluna sistemin model öngörülü kontrolü adlı butona tıklanıldığında ayarlanabilir değişkenlere ve çıkışa ait grafiklerde pencere üzerinde gözükecektir. Tasarım uygulanmış son durumu aşağıdaki şekil III.36 üzerinde gözükmektedir. Yapılan bu arayüz sayesinde çeşitli sistemlere ait transfer fonksiyonları arayüze girilerek ve kontrol birimleri durumlara göre değiştirilerek sistemin model öngörülü kontrolü gerçekleştirilir.



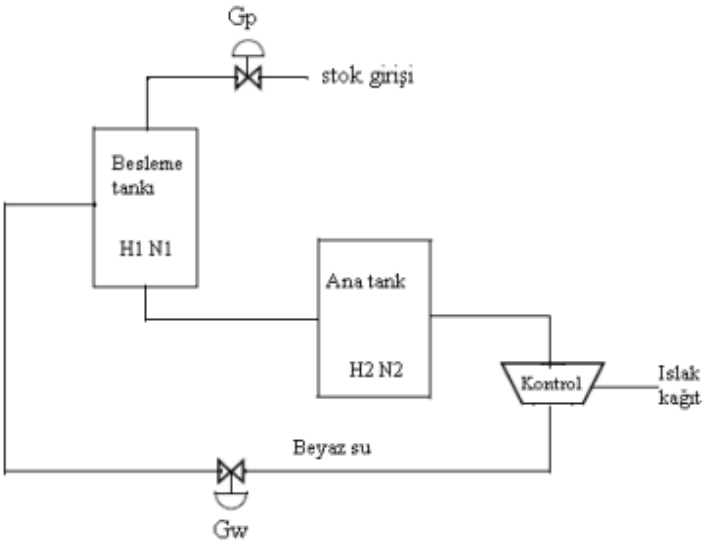
Şekil III.36 Çok Girişli Çok Çıkışlı Bir Sistemin Kısıtlamalı Model Öngörülü Kontrolü Eğitim ve Uygulama Arayüzü

III.5 ÇOK GİRİŞLİ ÇOK ÇIKIŞLI BİR SİSTEMİN DURUM UZAY MATRİSLERİ İLE KISITLAMALI MODEL ÖNGÖRÜLÜ KONTROLÜ

III.5.1 Kontrol Edilecek Sistemin Yapısı

Bir önceki yapılan çok girişli çok çıkışlı sistemlerin model öngörülü kontrolü uygulamalarından farklı olarak bu uygulamada gerçek bir sistem üzerinden verilen durum uzay matrisleri kullanılarak sistemin model öngörülü kontrolü amaçlanmaktadır.

Durum uzay matrisleri verilen sistemin görünüşü şekil III.37 üzerinde gösterilmiştir. Aşağıdaki sistem bir kağıt makinesinin üretilmesiyle ilgilidir. Sistemin yapısı incelendiğinde H1 ve H2 sıvı seviyesini göstermektedir. N1 ve N2 ise akış oranlarındaki tutarlılığı göstermektedir. Sistemin çıkışı ise y=[H2 N2] den oluşmaktadır. Sistem içerisindeki ayarlanabilir değişkenler ise u=[Gp Gw] şeklinde tanımlanmıştır. Gp stok girişinin debisi Gw beyaz su girişinin debisini oluşturmaktadır.



Şekil III.37 Durum Uzay Matrisleri Verilen Sistemin Görünüşü

Sistemin denklemlerini yazdığımızda durum matrislerini elde etmek amacıyla (y=0, u=0, v=0, d=0) sistem bilgilerini sıfır olarak kabul edilir ve yapılan hesaplamalar sonucunda durum uzay matrisleri elde edilir. Sisteme ait durum uzay matrisleri ise aşağıdaki gibidir.

$$y = Ay + B_0 u + B_v v + B_d d \qquad \text{(III.21)}$$

$$A = \begin{matrix} -1.93 & 0 & 0 & 0 \\ 0.394 & -0.426 & 0 & 0 \\ 0 & 0 & 0.63 & 0 \\ 0.82 & -0.784 & 0.413 & -0.426 \end{matrix} \qquad C = \begin{matrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix} \qquad \text{(III.22 a,c)}$$

$$B = \begin{matrix} 1.274 & 1.274 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1.314 & -0.65 & 0.203 & 0.406 \\ 0 & 0 & 0 & 0 \end{matrix} \qquad D = \begin{matrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{matrix} \qquad \text{(III.23 b,d)}$$

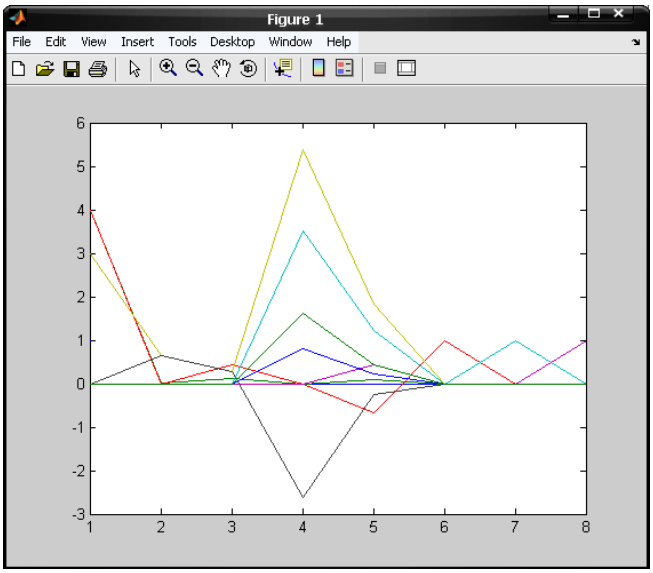
Sisteme ait bu durum uzay matrislerini MATLAB formatına dönüşmesi gerekmektedir. Bu dönüşüm işlemlerinden ilk olarak c2dmp komutu kullanılarak sürekli zamandan ayrık zaman formatına dönüşüm yapılır bu işlem aşağıdaki gibidir.

[PHI, GAM]=c2dmp(A, B, dt); (III.24)

Matrisler ve dt örnekleme zamanı 1 olarak komut içerisine girildiğinde MATLAB tarafından hesaplanır ve PHI, GAM isimlerinde tanımlandırılır.

$imod = ss2mod(PHI, GAM, C, D);$ (III.25)

ss2mod komutu ile de MATLAB'ın MPC Toolbox içerisinde anlayabileceği mod formatına dönüştürülmüştür.



Şekil III.38 Durum Uzay Matrislerinin Mod Formatına Dönüştürülmesi (imod)

Sistemin tüm model bilgilerinin bütünleştirilmiş durumu şekil III.38 üzerinde gözükmektedir.

III.5.1.1 Sistemin Kontrol Edilmesi

Sistem kısıtlamalı olduğundan dolayı MPC kazanç katsayısı bulunmamaktadır ve kontrol işlemi bir bütün olarak yapılmaktadır. Kontrol sistemine ait bilgiler aşağıdaki gibidir.

$P = 10;$

$M = 3;$

$ywt = [1,0,1];$

$uwt = [1\ 1];$

$ulim = [-10 * [1\ 1]\ 10 * [1\ 1]\ 2 * [1\ 1]];$

$ylim = [\,];$

$pmod = imod;$

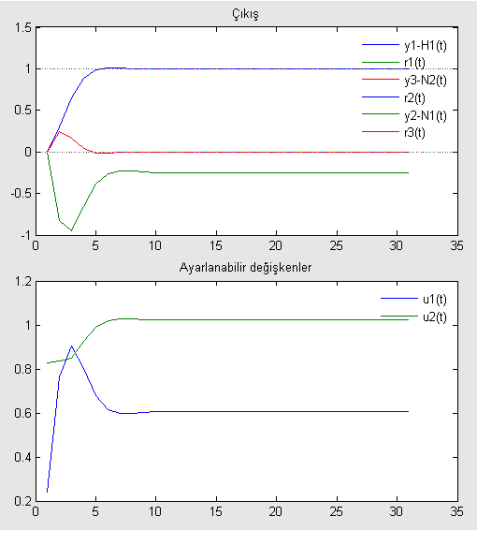
$tend = 30;$

$setpts = [1,0,0];$

Verilen kontrol bilgilerinde ayarlanabilir değişkenler üzerinde kısıtlama yapılmıştır. Öngörü ufku 10, kontrol ufku 3 olarak alınmıştır. Giriş ve çıkış ağırlık matrisleri ise 1 katsayısı ile çarpılmaktadır. Sistemin model öngörülü kontrolü ise MPC Toolbox içerisinde scmpc komutu içerisindeki algoritmalara gömülerek gerçekleşir. Bu komutun uygulanışı aşağıdaki gibidir.

$$[y,u] = scmpc(pmod, imod, ywt, uwt, M, P, tend, setpts, ulim, ylim); \quad \text{(III.26)}$$

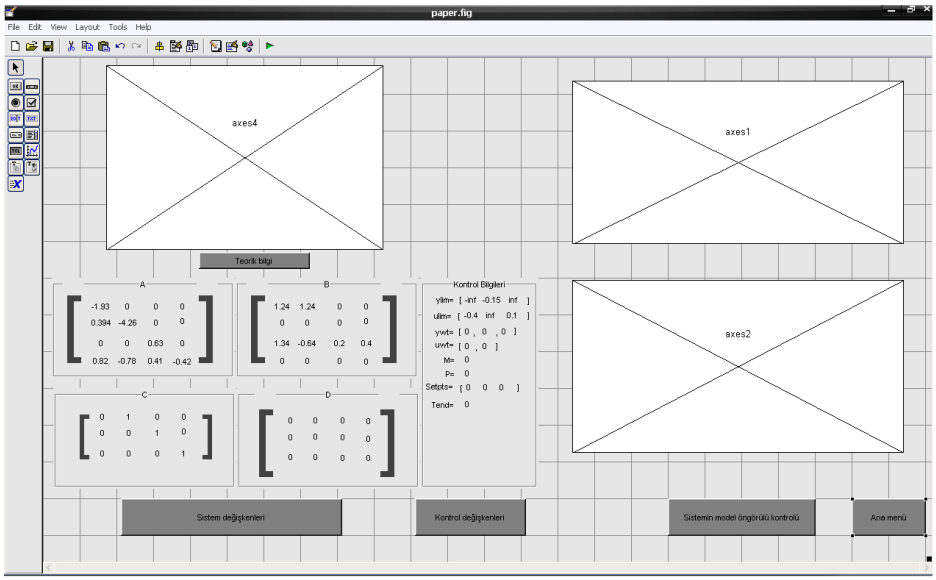
Model öngörülü kontrol işleminin sonucunda elde edilen ayarlanabilir değişken çıkılar üzerindeki değişimler aşağıdaki şekil III.39 üzerindeki gösterilmektedir.



Şekil III.39 Sistemin Model Öngörülü Kontrolü

III.5.2 Çok Girişli Çok Çıkışlı Bir Sistemin Durum Uzay Matrisleri ile Kısıtlamalı Model Öngörülü Kontrolü Arayüzü Tasarımı

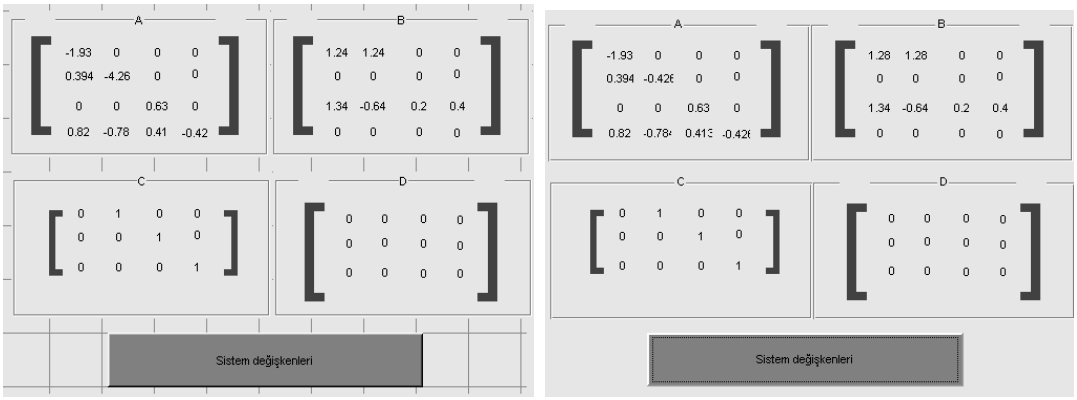
Sistemin arayüz tasarımı diğer uygulamalardaki gibidir. Sistemi oluşturan modellerin yer aldığı sistem bilgileri, kontrol bilgilerinin yer aldığı kısım ve model öngörülü kontrolünü gözlemlenebildiği kısımdan oluşmaktadır. Tasarımın tamamlanan hali aşağıdaki gibidir.



Şekil III.40 Durum Uzay Matrisleri Verilen Sistemin Model Öngörülü Kontrol Arayüzü Tasarımı

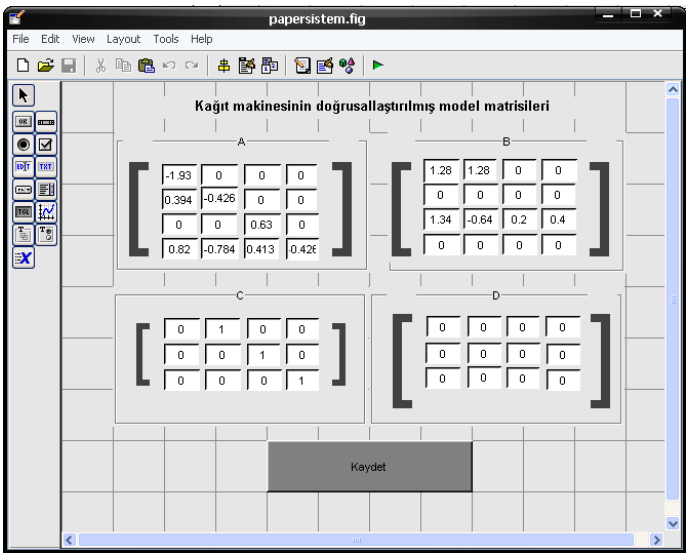
III.5.2.1 Sistem Değişkenlerinin Tasarlanması

Sistemin bilgilerini oluşturan durum uzay matrislerinin tasarıma uygulanması ve matris haline dönüştürülmesi aşağıdaki gibi gerçekleşmiştir.



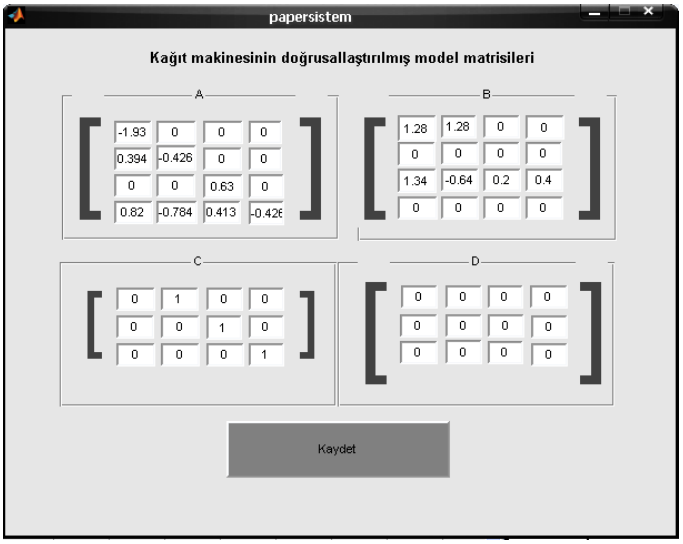
Şekil III.41 Durum Uzay Matrislerinin Uygulama Penceresindeki Görünümü

GUI tasarımı aktif hale getirildiğinde sistem bilgilerinin görünüşü şekil III.41 sağdaki gibidir. Buradaki ya da farklı verileri sisteme yüklemek için sistem değişkenleri adlı butona tıklanıldığında açılan yardımcı pencere üzerinden bu bilgiler sisteme girilir. Yardımcı pencerenin tasarımı ise şekil III.42 üzerinde gösterilmiştir.



Şekil III.42 Kontrol Değişkenlerinin Girildiği Pencerenin Tasarımı

Bu arayüz tasarımında oluşturulan ve sistem bilgilerinin girilmesi amaçlanan yardımcı pencere durum uzay matrislerinin girilmesi yönelik hazırlanmıştır. Matrislerin yazımı gurup gurup yapılmıştır ve her bir satırda ve sütundaki bilgilerin girilmesi için düzenleme kutularından yararlanılmıştır. Yardımcı arayüz tasarımının aktif haldeki görüşü şekil III.43 üzerinde gözükmektedir.



Şekil III.43 Kontrol Değişkenlerinin Girildiği Pencerenin Aktif Durumu

Arayüzün arkasında m file üzerinde çalışan kodların yazımı ise aşağıdaki gibidir.

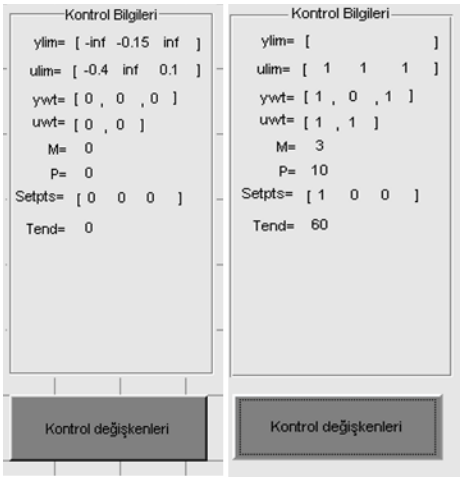
```
c11 = get(handles.c11,'String');        set(paperGUIdata.c11,'string',c11)
c12 = get(handles.c12,'String');        set(paperGUIdata.c12,'string',c12)
c13 = get(handles.c13,'String');        set(paperGUIdata.c13,'string',c13)
c14 = get(handles.c14,'String');        set(paperGUIdata.c14,'string',c14)
```

Matrisin her birimi için bu kodlar sırasıyla yazılamıştır. Kaydet butonuna tıklanıldığında be veriler ana pencere üzerindeki adreslerine kayıt edilecektir. Bu verileri matris şeklinde ifade edilebilesi ise aşağıdaki gibi gerçekleşir.

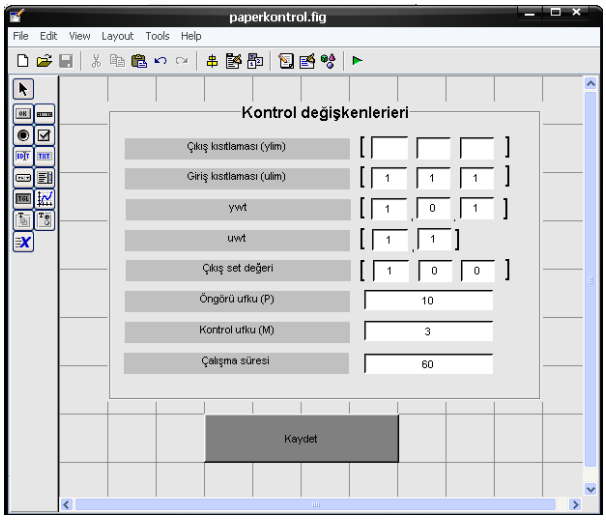
```
C=[c11 c12 c13 c14; c21 c22 c23 c24; c31 c32 c33 c34];
```

III.5.2.2 Kontrol Değişkenlerinin Tasarlanması

Kontrol değişkenlerine ait bilgilerin GUI'nin pasif ve aktif oldukları duruma göre gösterim şekil III.44 üzerinde gösterilmiştir. Bu bilgileri girilmesi için ise kontrol değişkenleri adlı butona tıklanıldığında açılan yardımcı pencere ile gerçekleşmektedir. Yardımcı pencerenin tasarımı ise şekil III.45 üzerinde gösterilmiştir.

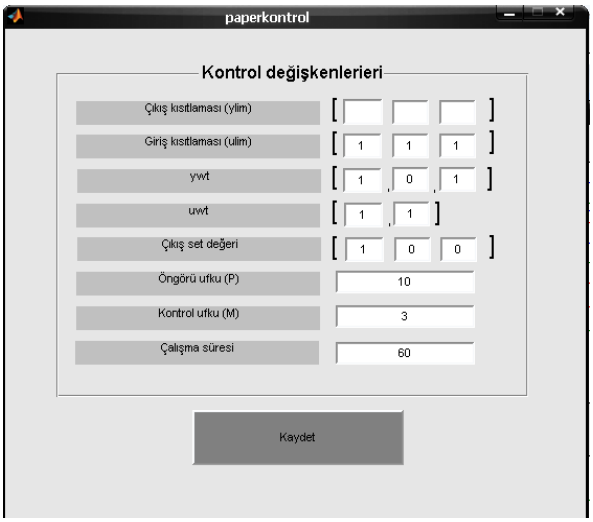


Şekil III.44 Kontrol Değişkenlerinin Uygulama Penceresi Üzerindeki Görünümü



Şekil III.45 Kontrol Değişkenlerine Ait Verilerin Girildiği GUI Penceresinin Tasarımı

Tasarımda yer alan bilgilerin girilmesi için düzenleme kutuları hedeflenmiştir. GUI aktif hale getirildiğinde ise aşağıdaki şekil III.46'daki gibi görünür.



Şekil III.46 Kontrol Değişkenlerine Ait Verilerin Girildiği GUI Penceresinin Aktif Görünümü

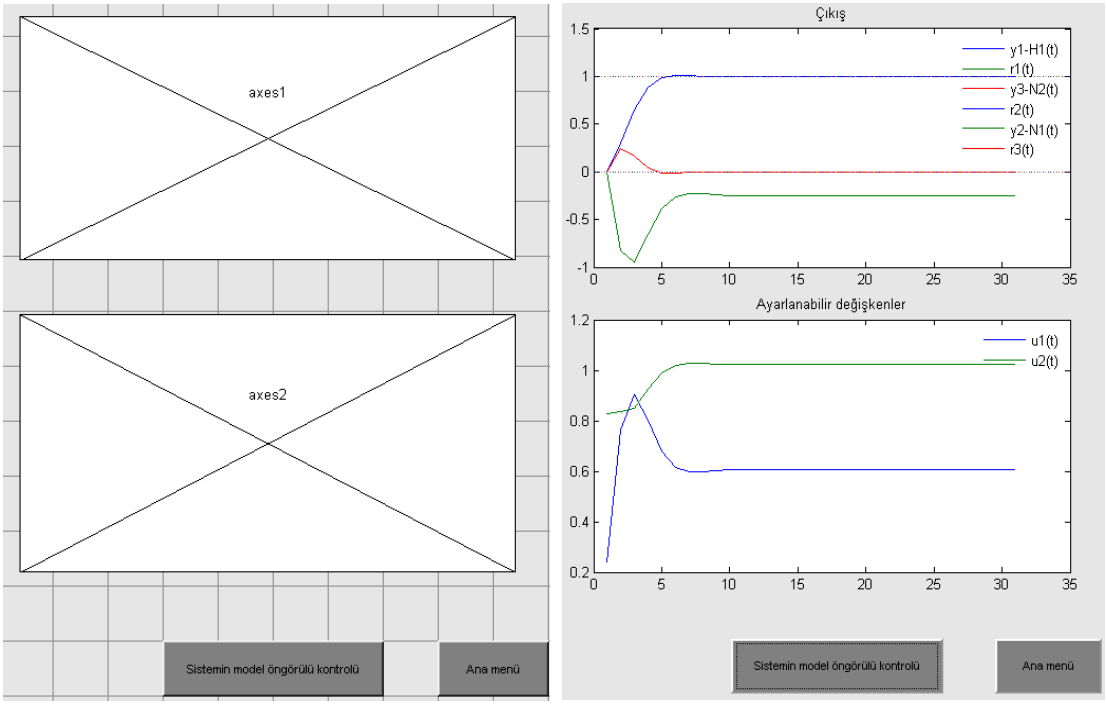
Kontrol sistemini oluşturan bilgiler düzenleme kutuları üzerinden girilir ve kaydedilir. Kayıt işlemi yapıldığında açılan bu yardımcı pencere kapanır ve girdiğimiz veriler ana pencere üzerinde gözükür. Bu işlemi arka planda çalışan m

file üzerine yazılmış kodlar tarafından gerçekleşir. Bu kodların bir kısmı aşağıdaki gibidir.

```
yw1 = get(handles.yw1,'String');      set(paperGUIdata.yw1,'string',yw1)
yw2 = get(handles.yw2,'String');      set(paperGUIdata.yw2,'string',yw2)
yw3 = get(handles.yw3,'String');      set(paperGUIdata.yw3,'string',yw3)
uw1 = get(handles.uw1,'String');      set(paperGUIdata.uw1,'string',uw1)
uw2 = get(handles.uw2,'String');      set(paperGUIdata.uw2,'string',uw2)
 M = get(handles.M,'String');         set(paperGUIdata.M,'string',M)
 P = get(handles.P,'String');         set(paperGUIdata.P,'string',P)
 r1 = get(handles.r1,'String');       set(paperGUIdata.r1,'string',r1)
r2 = get(handles.r2,'String');        set(paperGUIdata.r2,'string',r2)
r3 = get(handles.r3,'String');        set(paperGUIdata.r3,'string',r3)
```

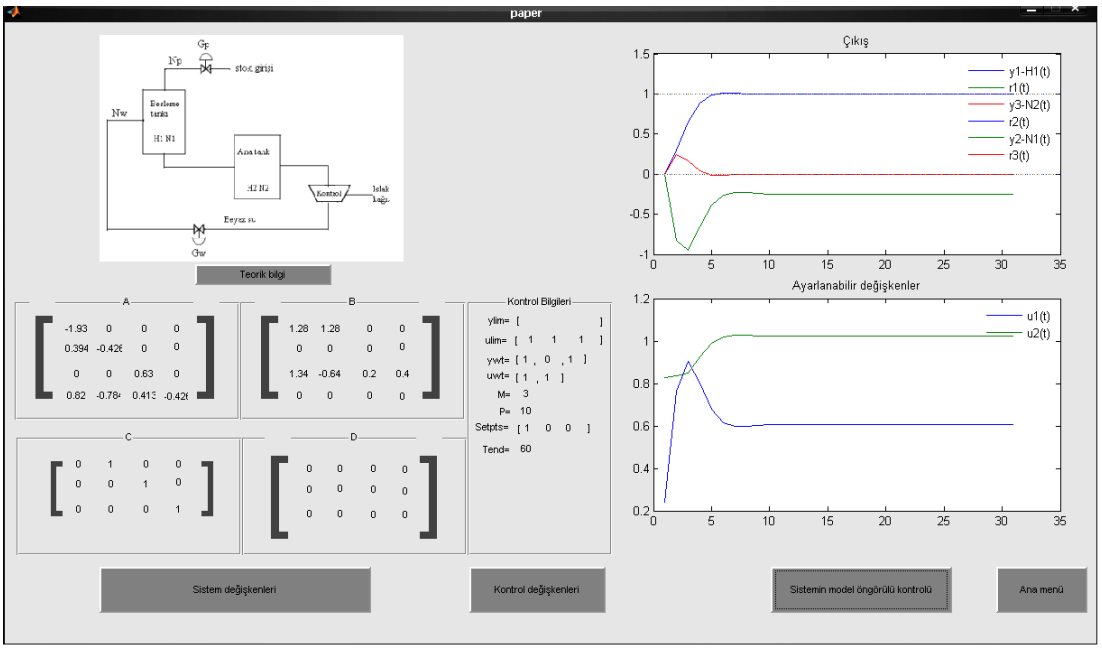
III.5.2.3 Model Öngörülü Kontrolün Gerçekleşmesi

Sistem ve kontrole ait bütün bilgiler ana pencere üzerinde girildikten sonra sistemin model öngörülü kontrolünün gerçekleşmesi ve gözlemlenmesi için aşağıdaki tasarım bölümü ana pencere üzerinde yapılmıştır.



Şekil III.47 Model Öngörülü Kontrol İşleminin Yaptırılacağı ve Gözlemleneceği Arayüzün Tasarımı

Bu tasarımda çıkış ve ayarlanabilir değişkenlerinin gözlenebilineceği iki adet axes ve sistemin model öngörülü kontrolünü hesaplama işleminin gerçekleşmesi içinde bir buton konulmuştur. Butona tıklanıldığında arka planda çalışan m file üzerinde çalışan kodlar ile model öngörülü kontrol işlemi gerçekleşir. M file üzerine yazılan bu kodlar ek 5, 5-1 ve 5-2'de verilmiştir.



Şekil III.48 Çok Girişli Çok Çıkışlı Bir Sistemin Durum Uzay Matrisleri ile Kısıtlamalı Model Öngörülü Kontrolü Eğitim ve Uygulama Arayüzü

BÖLÜM IV.

SONUÇLAR VE TARTIŞMA

IV.I MODEL ÖNGÖRÜLÜ KONTROL İŞLEMİNİN GERÇEKLEŞTİRİLMESİ

Bölüm III'te ele alınan ve beş alt başlık içerisinde anlatılan çeşitli durumlar ve şartlara göre model öngörülü kontrol işlemi gerçekleşmiştir. Bu durumlar tek girişli tek çıkışlı sistemler, çok girişli çok çıkışlı sistemlerden oluşmaktadır. Bu durumlara göre MATLAB MPC Toolbox içerisinde uygulanan algoritmaların türü de farklılaşmıştır. Verilen çeşitli durumların genel olarak en yaygın kullanılan sistem tiplerinin seçimine dikkat edilmiştir. Her farklı durumu yansıtan süreçte model öngörülü kontrol işlemi yapılmış ve her duruma göre kullanılan komut ve algoritmalar açıklanmıştır.

Yapılan bu çalışmaların sonucunda MATLAB MPC Toolbox'la tasarlanan sistemlerin model öngörülü kontrol ile nasıl kontrol edileceği, kullanılan terim ve kalıpların nasıl anlamlandırıldığı, model öngörülü kontrolün çalışma şeklinin öğrenilmesi ve kavranması gerçekleşmiştir.

IV.II. MATLAB GUI'DE MODEL ÖNGÖRÜLÜ KONTROL ARAYÜZÜ TASARIMI

Model öngörülü kontrolü bölüm III'te gerçekleştirilen beş farklı duruma ait beş farklı arayüz tasarlanmıştır. Arayüz tasarımları MATLAB GUI üzerinde gerçekleştirilmiştir. Arayüzler üzerinde genel olarak sistem modeli, kontrol verileri ve model öngörülü kontrolün gözlemlendiği kısımlar yer almaktadır.

Yapılan arayüz tasarımları sayesinde uygulayıcılar kontrol edecekleri sistem yapılarına ait verileri model öngörülü kontrol sürecine rahatlıkla dahil edebilmektedirler. Model öngörülü kontrol sürecinde kontrol bilgileri, ufuk bilgileri, referans bilgileri de kolaylıkla arayüz üzerine girilerek model öngörülü kontrol

işlemi kolaylıkla yapılabilmektedir. Kullanıcıların bu şekilde model öngörülü kontrol sürecine ilişkin gözlemlerini kontrol sürecine ilişkin değişimlerini ve bu değişimlerinin kontrol üzerindeki etkilerini rahatlıkla gözlemleyebilmektedir. Ayrıca arayüzler üzerinde verilen teorik bilgiler adlı buton ile açılan sayfada model öngörülü kontrolünün aşamalarını gerçekleşme şekli hakkında da bilgi vermektedir.

BÖLÜM V.

SON DEĞERLENDİRMELER VE ÖNERİLER

Bu çalışmada model öngörülü kontrol çeşitleri, çalışmaları, kullanımları hakkında bilgiler verilmiştir. Bu bilgilerin uygulanabileceği bir arayüz tasarımı gerçekleştirilmiştir. Bu tasarımda bulunan model öngörülü kontrol işleminin uygulanabileceği ve gözlemlenebileceği metaryeller ve uygulamalar, özellikle proses kontrol ile ilgili derslerde ve eğitimlerde kullanabilir durumdadır. Yapılan çalışma, Marmara Üniversitesi Fen Bilimleri Enstitüsü Mekatronik bölümünde okutulan Proses Dinamiği ve Kontrol dersini alan öğrenciler ile dönem içinde test edilmiş ve tasarımın eksiklikleri giderildikten sonra konunun anlaşılmasında büyük ölçüde fayda sağladığı görülmüştür. Arayüzün eğitim öğretimde başarı değerlendirilmesi amacıyla tasarım, eğitim ölçme değerlendirme kriterleri ile 2011-2012 bahar döneminde Marmara Üniversitesi Teknik Eğitim Fakültesi Mekatronik Bölümü 4.sınıf dersi olan Proses Kontrol dersinde kullandırılarak sistemin başarısı test edilecektir.

Yapılan çalışma MATLAB programına bağlı olarak çalışmaktadır. Bu da arayüz tasarımının ve MPC hesaplamalarının MATLAB programı dışında çalışmasını engel teşkil etmektedir. Bu arayüz, MATLAB programından bağımsız çalışabilecek bir yazılım haline (exe formatına) dönüştürülebilir.

Çalışmalar tek girişli tek çıkışlı ve iki girişli iki çıkışlı sistemlere göre gerçekleşmiştir. Bu giriş ve çıkış sayıları artırılarak model öngörülü kontrol işlemi daha da ön plana çıkarılarak kapsamlı hale getirilebilir ve kullanıcılar süreç içerisindeki etkenlerin bir birleri ile ilişkisini daha rahat gözlemleyebilirler.

Hazırlanan bu arayüzün web üzerinden erişim ve gerçek zamanlı kontrol modülü eklenerek uzaktan eğitimde kullanılabilir hale getirilebilir.

Bu değerlendirme ve öneriler sonucunda model öngörülü kontrol eğitim ve uygulama arayüzü tasarımı daha kullanışlı ve kapsamlı hale gelecektir.

KAYNAKLAR

[1] Kaplanoğlu,E.; Yılmazlar,E.; "Bir kısıtlamalı öngörülü kontrol algoritmasının endüstriyel uyarlaması" *Otomasyon Dergisi Mart 92-100,* **(2011)**

[2] Kaplanoğlu,E. "Bir kısıtlamalı öngörülü kontrol algoritmasının endüstriyel uyarlaması" *Doktora Tezi,* Marmara Üniversitesi Fen Bilimleri Enstitüsü, **(2006)**

[3] Smith., O.,J.,M.; "Close control of Loops With Deadtime"; *Chemical Engineering Progress;53:217,* **(1975)**

[4] Maciejowiski,J.M.,:"Predictive Control With Constraints", Prentice Hall **(2002)**

[5] Richalet,J.;Rault,A.,;Testud,L.,;Papon,J.,:"Model Heuristic Control: Application To Industrial Processes"; *Automatica 14:413-428,* **(1978)**

[6] Cutler ,C.R.;Ramaker, B.,L.:"Dynamic Matrix Control-a Computer Control Algorithm"; *Proceeding American Control Conference SanFrancisco,* **(1980)**

[7] Clarke, D.W.; Scattolini, R.: "Constrained Receding-Horizon Predictive Control." *IEE Proc. D 138 (4), 347-354*, **(1991)**

[8] Ricker, N. L., Subramanian, T.,; Sim T.,: "Case Studies of Model Predictive Control in Pulp and Paper Production, " : *Model Based Process Control, (ed. T. J. McAvoy, Y. Arkun, and E. Zafiriou), Pergamon Pres,* **(1989)**

[9] Richalet,J.,:"Industrial Applications of Model Based Predictive Control"; *Automatica 29,1,* **(1993)**

[10] Camacho, F. E.; Bordons. C.: "Model Predictive Control", *Springer Press*, (**2000)**

[11] Temurtas, F.; Temurtas, H.; Yumusak, N.; Oz, C.:"Effects of Trajectory Planning on the Model Based Predictive Robotic Manipulator" *Control. ISCIS 2003, LNCS 2869, 545-552,* **(2003)**

[12] Camacho, F. E.; Bordons. C.: "Model Predictive Control", *Springer Press*, (**2000)**

[13] Yalçın,Y.,;"Modele Dayalı Öngörülü Endüstriyel Kontrolörlerin Gerçeklenmesi", *YSL Tezi İTÜ Fen Bilimleri Ens,* **(2004)**

[14] Tham, M.T.: ”Multivariable Control (An Introduction To Decoupling Control), *Department of Chemical Process Engineering University of Newcastle*, **(1999)**

[15] Ogunnaike,A.B.;Ray W.H.:”Process Dynamics,Modelling and Control” Oxford University Press, **(1994)**

[16] Yavuzyılmaz,Ç.;Demircioğlu, H,”Kısıtlamalı Çokdeğişkenli SÜGÖNDE”, *TOK2005 Otomatik Kontrol Ulusal Toplantısı bildirler kitabı,* **(2005)**

[17] Yüksel,İ, "Otomatik kontrol sistem dinamiği ve denetim sistemleri" *Nobel Yayınları, 5. Basım*, **(2006)**

[18] Savaş, K, "Kontrol sistemleri için MATLAB GUI tasarıları" *Lisans Tezi,* Marmara Üniversitesi Teknik Eğitim Fak, **(2007)**

[19] Ricker, N; Morari, M, "Model predictive control toolbox" *User Gui, MathWorks,* **(1998)**

EKLER

EK 0-1 ANA GUI TASARIMININ KODLARI

```
%---Uygulama 1 için-----------

function sis1_Callback(hObject, eventdata, handles)
panels=[handles.uipanel1 handles.uipanel2 handles.uipanel3
handles.uipanel4 handles.uipanel5 handles.uipanel6];
   handles.currPanel=1;
[resim1]=imread('siso.bmp');
axes(handles.axes1);
imshow(resim1);
set(panels(2),'Visible','on','Position',[75 4 50 4]);
set(panels(3),'Visible','off','Position',[75 4 50 4]);
set(panels(4),'Visible','off','Position',[75 4 50 4]);
set(panels(5),'Visible','off','Position',[75 4 50 4]);
set(panels(6),'Visible','off','Position',[75 4 50 4]);
A='Tek Girişli Tek Çıkışlı Bir Sistemin Kısıtlamasız Model Öngörülü
Kontrolü','FontSize',10
set(handles.text1,'String',A);

%---Uygulama 2 için-----------
function sis2_Callback(hObject, eventdata, handles)
panels=[handles.uipanel1 handles.uipanel2 handles.uipanel3
handles.uipanel4 handles.uipanel5 handles.uipanel6];
   handles.currPanel=1;
   [resim1]=imread('siso.bmp');
axes(handles.axes1);
imshow(resim1);
set(panels(2),'Visible','off','Position',[75 4 50 4]);
set(panels(3),'Visible','on','Position',[75 4 50 4]);
set(panels(4),'Visible','off','Position',[75 4 50 4]);
set(panels(5),'Visible','off','Position',[75 4 50 4]);
set(panels(6),'Visible','off','Position',[75 4 50 4]);
A='Tek Girişli Tek Çıkışlı Bir Sistemin Kısıtlamalı Model Öngörülü
Kontrolü','FontSize',10
set(handles.text1,'String',A);

%---Uygulama 3 için-----------
function sis3_Callback(hObject, eventdata, handles)
panels=[handles.uipanel1 handles.uipanel2 handles.uipanel3
handles.uipanel4 handles.uipanel5 handles.uipanel6];
   handles.currPanel=1;
   [resim1]=imread('durum.bmp');
axes(handles.axes1);
imshow(resim1);
set(panels(2),'Visible','off','Position',[75 4 50 4]);
set(panels(3),'Visible','off','Position',[75 4 50 4]);
set(panels(4),'Visible','on','Position',[75 4 50 4]);
set(panels(5),'Visible','off','Position',[75 4 50 4]);
set(panels(6),'Visible','off','Position',[75 4 50 4]);
A='Çok Girişli Çok Çıkışlı Bir Sistemin Kısıtlamasız Model Öngörülü
Kontrolü','FontSize',10
set(handles.text1,'String',A);
```

```
%---Uygulama 4 için-----------
function sis4_Callback(hObject, eventdata, handles)
panels=[handles.uipanel1 handles.uipanel2 handles.uipanel3
handles.uipanel4 handles.uipanel5 handles.uipanel6];
   handles.currPanel=1;
   [resim1]=imread('durum1.bmp');
axes(handles.axes1);
imshow(resim1);
set(panels(2),'Visible','off','Position',[75 4 50 4]);
set(panels(3),'Visible','off','Position',[75 4 50 4]);
set(panels(4),'Visible','off','Position',[75 4 50 4]);
set(panels(5),'Visible','on','Position',[75 4 50 4]);
set(panels(6),'Visible','off','Position',[75 4 50 4]);
A='Çok Girişli Çok Çıkışlı Bir Sistemin Kısıtlamalı Model Öngörülü
Kontrolü','FontSize',10
set(handles.text1,'String',A);

%---Uygulama 5 için-----------
function sis5_Callback(hObject, eventdata, handles)
panels=[handles.uipanel1 handles.uipanel2 handles.uipanel3
handles.uipanel4 handles.uipanel5 handles.uipanel6];
   handles.currPanel=1;
   [resim1]=imread('paper.bmp');
axes(handles.axes1);
imshow(resim1);
set(panels(2),'Visible','off','Position',[75 4 50 4]);
set(panels(3),'Visible','off','Position',[75 4 50 4]);
set(panels(4),'Visible','off','Position',[75 4 50 4]);
set(panels(5),'Visible','off','Position',[75 4 50 4]);
set(panels(6),'Visible','on','Position',[75 4 50 4]);
A='Çok Girişli Çok Çıkışlı Bir Sistemin Durum Uzay Matrsileri ile
Kısıtlamalı Model Öngörülü Kontrolü','FontSize',10
set(handles.text1,'String',A);

%---Uygulama pencerelerinin açılmı----------

function uy1_Callback(hObject, eventdata, handles)
bas1()
function uy2_Callback(hObject, eventdata, handles)
siso2()
function uy3_Callback(hObject, eventdata, handles)
statemerkez()
function uy4_Callback(hObject, eventdata, handles)
spacemerkez()
function uy5_Callback(hObject, eventdata, handles)
paper()
```

EK I TEK GİRİŞLİ TEK ÇIKIŞLI BİR SİSTEMİN KISITLAMASIZ MODEL ÖNGÖRÜLÜ KONTROLÜ TASARIMI KODLARI

```
varargout{1} = handles.output;
[resim1]=imread('denklem1.bmp');
axes(handles.axes11);
imshow(resim1)
[resim2]=imread('siso.bmp');
axes(handles.axes12);
imshow(resim2)
```

```
function plantt_Callback(hObject, eventdata, handles)
plant();
function bozucuu_Callback(hObject, eventdata, handles)
bozucu();
function kontroll_Callback(hObject, eventdata, handles)
kontrol();

function deger_Callback(hObject, eventdata, handles)
pay1=get(handles.onbir, 'String');
pay1=str2num(pay1);
pay2=get(handles.oniki, 'String');
pay2=str2num(pay2);
pay3=get(handles.onuc, 'String');
pay3=str2num(pay3);
pay4=get(handles.ondort, 'String');
pay4=str2num(pay4);
payda1=get(handles.onbes, 'String');
payda1=str2num(payda1);
payda2=get(handles.onalti, 'String');
payda2=str2num(payda2);
payda3=get(handles.onyedi, 'String');
payda3=str2num(payda3);
payda4=get(handles.onsekiz, 'String');
payda4=str2num(payda4);
%
pay21=get(handles.yirmibir, 'String');
pay21=str2num(pay21);
pay22=get(handles.yirmiki, 'String');
pay22=str2num(pay22);
pay23=get(handles.yirmiuc, 'String');
pay23=str2num(pay23);
pay24=get(handles.yirmidort, 'String');
pay24=str2num(pay24);
payda21=get(handles.yirmibes, 'String');
payda21=str2num(payda21);
payda22=get(handles.yirmialti, 'String');
payda22=str2num(payda22);
payda23=get(handles.yirmiyedi, 'String');
payda23=str2num(payda23);
payda24=get(handles.yirmisekiz, 'String');
payda24=str2num(payda24);
%
delay1 = get(handles.delay1,'String');
delay1=str2num(delay1)
nout1 =1;
delt1 = get(handles.delt1,'String');
delt1=str2num(delt1)
delt2 = get(handles.delt2,'String');
delt2=str2num(delt2)
delay2 = get(handles.delay2,'String');
delay2=str2num(delay2)
nout2 =1;
%
ywt = get(handles.ywt,'String');
ywt=str2num(ywt);
uwt = get(handles.uwt,'String');
uwt=str2num(uwt);
M = get(handles.M,'String');
M=str2num(M);
P = get(handles.P,'String');
```

```
P=str2num(P);
tend= get(handles.tend,'String');
tend=str2num(tend);
tfinal=tend
r= get(handles.r,'String');
r=str2num(r);
%
NUM1=[pay4 pay3 pay2 pay1];
DEN1=[payda4 payda3 payda2 payda1];
NUM2=[pay24 pay23 pay22 pay21];
DEN2=[payda24 payda23 payda22 payda21];
%
gp=poly2tfd(NUM1,DEN1,delt1,delay1);
plant=tfd2step(tfinal,delt2,nout1,gp);
model=plant;
gd=poly2tfd(NUM2,DEN2,delt1,delay2);
dplant=tfd2step(tfinal,delt2,nout2,gd);
usat=[]
tfilter=[]
dstep=1
gp=num2str(gp);
gd=num2str(gd);
set(handles.iki,'string',gp);
set(handles.uc,'string',gd);
%
Kmpc1=mpccon(model,ywt,uwt,M,P);
x=0:1:40
modeld=dplant
[y1,u1]=mpcsim(plant,model,Kmpc1,tend,r,usat,tfilter,dplant,modeld,d
step);
axes(handles.axes4);
plot(y1); hold; plot(x,r,'-');
legend('y(t)', 'r(t)', 'Location','NorthWest'); legend boxoff;
title('Çıkış');
axes(handles.axes3);
plot(u1);
legend('u(t)','Location','NorthWest'); legend boxoff;
title('Ayarlanabilir değişkenler');
function pushbutton14_Callback(hObject, eventdata, handles)
close(bas1)
```

EK I-1 Sistem değişkenleri için tasarlanan yardımcı GUI'nin m file üzerine girilen kodlar

```
bas1GUIhandle= bas1;
bas1GUIdata= guidata(bas1GUIhandle);
pay1=get(handles.pay1, 'String');
set(bas1GUIdata.onbir,'string',pay1)
pay2 = get(handles.pay2,'String');
set(bas1GUIdata.oniki,'string',pay2)
pay3 = get(handles.pay3,'String');
set(bas1GUIdata.onuc,'string',pay3)
pay4 = get(handles.pay4,'String');
set(bas1GUIdata.ondort,'string',pay4)
payda1 = get(handles.payda1,'String');
set(bas1GUIdata.onbes,'string',payda1)
payda2 = get(handles.payda2,'String');
set(bas1GUIdata.onalti,'string',payda2)
```

```
payda3 = get(handles.payda3,'String');
set(bas1GUIdata.onyedi,'string',payda3)
payda4 = get(handles.payda4,'String');
set(bas1GUIdata.onsekiz,'string',payda4)
delay1 = get(handles.delay1,'String');
set(bas1GUIdata.delay1,'string',delay1)
delt1 = get(handles.delt1,'String');
set(bas1GUIdata.delt1,'string',delt1)
%
NUM1=[str2num(pay4) str2num(pay3) str2num(pay2) str2num(pay1)];
DEN1=[str2num(payda4) str2num(payda3) str2num(payda2)
str2num(payda1)];
delt1=str2num(delt1);
delay1=str2num(delay1);
gp=poly2tfd(NUM1,DEN1,delt1,delay1);
gp=num2str(gp)
set(bas1GUIdata.iki,'string',gp)
guidata(bas1, bas1GUIdata);
close(plant);
```

EK I-2 Bozucu değişkenler için tasarlanan yardımcı GUI'nin m file üzerine girilen kodlar

```
function save2_Callback(hObject, eventdata, handles)
bas1GUIhandle= bas1;
bas1GUIdata= guidata(bas1GUIhandle);
pay21=get(handles.pay21, 'String');
set(bas1GUIdata.yirmibir,'string',pay21)
pay22 = get(handles.pay22,'String');
set(bas1GUIdata.yirmiki,'string',pay22)
pay23 = get(handles.pay23,'String');
set(bas1GUIdata.yirmiuc,'string',pay23)
pay24 = get(handles.pay24,'String');
set(bas1GUIdata.yirmidort,'string',pay24)
payda21 = get(handles.payda21,'String');
set(bas1GUIdata.yirmibes,'string',payda21)
payda22 = get(handles.payda22,'String');
set(bas1GUIdata.yirmialti,'string',payda22)
payda23 = get(handles.payda23,'String');
set(bas1GUIdata.yirmiyedi,'string',payda23)
payda24 = get(handles.payda24,'String');
set(bas1GUIdata.yirmisekiz,'string',payda24)
delay2 = get(handles.delay2,'String');
set(bas1GUIdata.delay2,'string',delay2)
delt2 = get(handles.delt2,'String');
set(bas1GUIdata.delt2,'string',delt2)
NUM1=[str2num(pay24) str2num(pay23) str2num(pay22) str2num(pay21)];
DEN1=[str2num(payda24) str2num(payda23) str2num(payda22)
str2num(payda21)];
delt2=str2num(delt2);
delay2=str2num(delay2);
gd=poly2tfd(NUM1,DEN1,delt2,delay2);
gd=num2str(gd)
set(bas1GUIdata.uc,'string',gd)
guidata(bas1, bas1GUIdata);
close(bozucu);
```

EK I-3 Kontrol değişkenleri için tasarlanan yardımcı GUI'nin m file üzerine girilen kodlar

```
bas1GUIhandle= bas1;
bas1GUIdata= guidata(bas1GUIhandle);
ywt = get(handles.ywt,'String');
set(bas1GUIdata.ywt,'string',ywt)
uwt = get(handles.uwt,'String');
set(bas1GUIdata.uwt,'string',uwt)
M = get(handles.M,'String');
set(bas1GUIdata.M,'string',M)
P = get(handles.P,'String');
set(bas1GUIdata.P,'string',P)
tend= get(handles.tend,'String');
set(bas1GUIdata.tend,'string',tend)
tfinal = get(handles.tend,'String');
set(bas1GUIdata.tend,'string',tfinal);
r = get(handles.r,'String');
set(bas1GUIdata.r,'string',r);
guidata(bas1, bas1GUIdata);
close(kontrol);
```

EK II TEK GİRİŞLİ TEK ÇIKIŞLI BİR SİSTEMİN KISITLAMALI MODEL ÖNGÖRÜLÜ KONTROLÜ TASARIMI KODLARI

```
function pushbutton2_Callback(hObject, eventdata, handles)
siso2plant();
function pushbutton3_Callback(hObject, eventdata, handles)
siso2kontrol();
function pushbutton4_Callback(hObject, eventdata, handles)
siso2bozucu();

function pushbutton1_Callback(hObject, eventdata, handles)

pay1=get(handles.pay1, 'String');
pay1=str2num(pay1);
pay2=get(handles.pay2, 'String');
pay2=str2num(pay2);
pay3=get(handles.pay3, 'String');
pay3=str2num(pay3);
pay4=get(handles.pay4, 'String');
pay4=str2num(pay4);
payda1=get(handles.payda1, 'String');
payda1=str2num(payda1);
payda2=get(handles.payda2, 'String');
payda2=str2num(payda2);
payda3=get(handles.payda3, 'String');
payda3=str2num(payda3);
payda4=get(handles.payda4, 'String');
payda4=str2num(payda4);
pay21=get(handles.pay11, 'String');
pay21=str2num(pay21);
pay22=get(handles.pay22, 'String');
pay22=str2num(pay22);
pay23=get(handles.pay33, 'String');
pay23=str2num(pay23);
pay24=get(handles.pay44, 'String');
```

```
pay24=str2num(pay24);
payda21=get(handles.payda11, 'String');
payda21=str2num(payda21);
payda22=get(handles.payda22, 'String');
payda22=str2num(payda22);
payda23=get(handles.payda33, 'String');
payda23=str2num(payda23);
payda24=get(handles.payda44, 'String');
payda24=str2num(payda24);
%
delay1 = get(handles.delay1,'String');
delay1=str2num(delay1)
nout1 =1;
delt1 = get(handles.delt1,'String');
delt1=str2num(delt1)
delt2 = get(handles.delt2,'String');
delt2=str2num(delt2)
delay2 = get(handles.delay2,'String');
delay2=str2num(delay2)
nout2 =1;
%
ywt = get(handles.ywt,'String');
ywt=str2num(ywt);
uwt = get(handles.uwt,'String');
uwt=str2num(uwt);
M = get(handles.M,'String');
M=str2num(M);
P = get(handles.P,'String');
P=str2num(P);
tend= get(handles.tend,'String');
tend=str2num(tend);
tfinal=tend
r= get(handles.r,'String');
r=str2num(r);
yl1 = get(handles.yl1,'String');
yl1=str2num(yl1);
yl2 = get(handles.yl2,'String');
yl2=str2num(yl2);
yl3 = get(handles.yl3,'String');
yl3=str2num(yl3);
ul1 = get(handles.ul1,'String');
ul1=str2num(ul1);
ul2 = get(handles.ul2,'String');
ul2=str2num(ul2);
ul3 = get(handles.ul3,'String');
ul3=str2num(ul3);
%
NUM1=[pay4 pay3 pay2 pay1];
DEN1=[payda4 payda3 payda2 payda1];
NUM2=[pay24 pay23 pay22 pay21];
DEN2=[payda24 payda23 payda22 payda21];
%
gp=poly2tfd(NUM1,DEN1,delt1,delay1);
plant=tfd2step(tfinal,delt2,nout1,gp);
model=plant;
gd=poly2tfd(NUM2,DEN2,delt1,delay2);
dplant=tfd2step(tfinal,delt2,nout2,gd);
modeld=[]
usat=[]
tfilter=[]
dstep=1
```

```
gp=num2str(gp);
gd=num2str(gd);
set(handles.gp,'string',gp);
set(handles.gd,'string',gd);
%
ulim=[ul3 ul2 ul1];
ylim=[yl3 yl2 yl1];
 x=0:1:40
modeld=dplant
[y1,u1]=cmpc(plant,model,ywt,uwt,M,P,tend,r,ulim,ylim,tfilter,dplant
,modeld,dstep);
axes(handles.axes2);
plot(y1);hold; plot(x,r,'r-');
legend('y(t)','r(t)','Location','NorthWest'); legend boxoff;
title('Çıkış');
 axes(handles.axes1);
plot(u1);
legend('u(t)','Location','NorthWest'); legend boxoff;
title('Ayarlanabilir değişkenler');

function ana_Callback(hObject, eventdata, handles)
close(siso2)
```

EK II-1 Sistem değişkenleri için tasarlanan yardımcı GUI'nin m file üzerine girilen kodlar

```
siso2GUIhandle= siso2;
siso2GUIdata= guidata(siso2GUIhandle);
pay1=get(handles.pay1, 'String');
set(siso2GUIdata.pay1,'string',pay1)
pay2 = get(handles.pay2,'String');
set(siso2GUIdata.pay2,'string',pay2)
pay3 = get(handles.pay3,'String');
set(siso2GUIdata.pay3,'string',pay3)
pay4 = get(handles.pay4,'String');
set(siso2GUIdata.pay4,'string',pay4)
payda1 = get(handles.payda1,'String');
set(siso2GUIdata.payda1,'string',payda1)
payda2 = get(handles.payda2,'String');
set(siso2GUIdata.payda2,'string',payda2)
payda3 = get(handles.payda3,'String');
set(siso2GUIdata.payda3,'string',payda3)
payda4 = get(handles.payda4,'String');
set(siso2GUIdata.payda4,'string',payda4)
delay1 = get(handles.delay1,'String');
set(siso2GUIdata.delay1,'string',delay1)
delt1 = get(handles.delt1,'String');
set(siso2GUIdata.delt1,'string',delt1)
NUM1=[str2num(pay4) str2num(pay3) str2num(pay2) str2num(pay1)];
DEN1=[str2num(payda4) str2num(payda3) str2num(payda2)
str2num(payda1)];
delt1=str2num(delt1);
delay1=str2num(delay1);
gp=poly2tfd(NUM1,DEN1,delt1,delay1);
gp=num2str(gp)
set(siso2GUIdata.gp,'string',gp)
guidata(siso2, siso2GUIdata);
close(siso2plant);
```

EK II-2 Bozucu değişkenler için tasarlanan yardımcı GUI'nin m file üzerine girilen kodlar

```
siso2GUIhandle= siso2;
siso2GUIdata= guidata(siso2GUIhandle);
pay21=get(handles.pay1, 'String');
set(siso2GUIdata.pay11,'string',pay21)
pay22 = get(handles.pay2,'String');
set(siso2GUIdata.pay22,'string',pay22)
pay23 = get(handles.pay3,'String');
set(siso2GUIdata.pay33,'string',pay23)
pay24 = get(handles.pay4,'String');
set(siso2GUIdata.pay44,'string',pay24)
payda21 = get(handles.payda1,'String');
set(siso2GUIdata.payda11,'string',payda21)
payda22 = get(handles.payda2,'String');
set(siso2GUIdata.payda22,'string',payda22)
payda23 = get(handles.payda3,'String');
set(siso2GUIdata.payda33,'string',payda23)
payda24 = get(handles.payda4,'String');
set(siso2GUIdata.payda44,'string',payda24)
delay2 = get(handles.delay2,'String');
set(siso2GUIdata.delay2,'string',delay2)
delt2 = get(handles.delt2,'String');
set(siso2GUIdata.delt2,'string',delt2)
NUM1=[str2num(pay24) str2num(pay23) str2num(pay22) str2num(pay21)];
DEN1=[str2num(payda24) str2num(payda23) str2num(payda22)
str2num(payda21)];
delt2=str2num(delt2);
delay2=str2num(delay2);
gd=poly2tfd(NUM1,DEN1,delt2,delay2);
gd=num2str(gd)
set(siso2GUIdata.gd,'string',gd)
guidata(siso2, siso2GUIdata);
close(siso2bozucu);
```

EK II-3 Kontrol değişkenleri için tasarlanan yardımcı GUI'nin m file üzerine girilen kodlar

```
siso2GUIhandle= siso2;
siso2GUIdata= guidata(siso2GUIhandle);
yl1 = get(handles.yl1,'String');
set(siso2GUIdata.yl1,'string',yl1)
yl2 = get(handles.yl2,'String');
set(siso2GUIdata.yl2,'string',yl2)
yl3 = get(handles.yl3,'String');
set(siso2GUIdata.yl3,'string',yl3)
ul1 = get(handles.ul1,'String');
set(siso2GUIdata.ul1,'string',ul1)
ul2 = get(handles.ul2,'String');
set(siso2GUIdata.ul2,'string',ul2)
ul3 = get(handles.ul3,'String');
set(siso2GUIdata.ul3,'string',ul3)
ywt = get(handles.ywt,'String');
set(siso2GUIdata.ywt,'string',ywt)
```

```
uwt = get(handles.uwt,'String');
set(siso2GUIdata.uwt,'string',uwt)
M = get(handles.M,'String');
set(siso2GUIdata.M,'string',M)
P = get(handles.P,'String');
set(siso2GUIdata.P,'string',P)
tend= get(handles.tend,'String');
set(siso2GUIdata.tend,'string',tend)
r = get(handles.r,'String');
set(siso2GUIdata.r,'string',r);
guidata(siso2, siso2GUIdata);
close(siso2kontrol);
```

EK III ÇOK GİRİŞLİ ÇOK ÇIKIŞLI BİR SİSTEMİN KISITLAMASIZ MODEL ÖNGÖRÜLÜ KONTROLÜ TASARIMI KODLARI

```
varargout{1} = handles.output;
 [resim1]=imread('denklem3.bmp');
axes(handles.axes3);
imshow(resim1)
[resim2]=imread('durum.bmp');
axes(handles.axes4);
imshow(resim2)
function pushbutton1_Callback(hObject, eventdata, handles)
statesistem()
function kontrol_Callback(hObject, eventdata, handles)
statekontrol()

function pushbutton2_Callback(hObject, eventdata, handles)

pay1=get(handles.pay1, 'String');
pay1=str2num(pay1);
pay2=get(handles.pay2, 'String');
pay2=str2num(pay2);
pay3=get(handles.pay3, 'String');
pay3=str2num(pay3);
pay4=get(handles.pay4, 'String');
pay4=str2num(pay4);
pay5=get(handles.pay5, 'String');
pay5=str2num(pay5);
pay6=get(handles.pay6, 'String');
pay6=str2num(pay6);
payda1=get(handles.payda1, 'String');
payda1=str2num(payda1);
payda2=get(handles.payda2, 'String');
payda2=str2num(payda2);
payda3=get(handles.payda3, 'String');
payda3=str2num(payda3);
payda4=get(handles.payda4, 'String');
payda4=str2num(payda4);
payda5=get(handles.payda5, 'String');
payda5=str2num(payda5);
payda6=get(handles.payda6, 'String');
payda6=str2num(payda6);
payda11=get(handles.payda11, 'String');
payda11=str2num(payda11);
payda22=get(handles.payda22, 'String');
payda22=str2num(payda22);
```

```
payda33=get(handles.payda33, 'String');
payda33=str2num(payda33);
payda44=get(handles.payda44, 'String');
payda44=str2num(payda44);
payda55=get(handles.payda55, 'String');
payda55=str2num(payda55);
payda66=get(handles.payda66, 'String');
payda66=str2num(payda66);
us1=get(handles.us1, 'String');
us1=str2num(us1);us1=-1*us1;
us2=get(handles.us2, 'String');
us2=str2num(us2);us2=-1*us2;
us3=get(handles.us3, 'String');
us3=str2num(us3);us3=-1*us3;
us4=get(handles.us4, 'String');
us4=str2num(us4);us4=-1*us4;
us5=get(handles.us5, 'String');
us5=str2num(us5);us5=-1*us5;
us6=get(handles.us6, 'String');
us6=str2num(us6);us6=-1*us6;
ywt=get(handles.ywt, 'String');
ywt=str2num(ywt);
uwt=get(handles.uwt, 'String');
uwt=str2num(uwt);
M=get(handles.M, 'String');
M=str2num(M);
R1=get(handles.R1, 'String');
R1=str2num(R1);
R2=get(handles.R2, 'String');
R2=str2num(R2);
r=[R1 R2];
P=get(handles.P, 'String');
P=str2num(P);
tend=get(handles.tend, 'String');
tend=str2num(tend);
w=get(handles.w, 'String');
w=str2num(w);
%
delt = 2;
ny = 2;
g11 = poly2tfd(pay1,[payda1 payda11],0,us1);
g21 = poly2tfd(pay3,[payda3 payda33],0,us3);
g12 = poly2tfd(pay2,[payda2 payda22],0,us2);
g22 = poly2tfd(pay4,[payda4 payda44],0,us4);
umod = tfd2mod(delt,ny,g11,g21,g12,g22)
% Defines the effect of u inputs
g13 = poly2tfd(pay5,[payda55 payda55],0,us5);
g23 = poly2tfd(pay6,[payda6 payda66],0,us6);
dmod = tfd2mod(delt,ny,g13,g23)
pmod = addumd(umod,dmod);
 imod = pmod;
ywt = [ ];
uwt = [ ];
v=[];
z=[];
ulim=[];
Kest=[];
w=[w];
x=0:0.2:35;
Ks = smpccon(imod,ywt,uwt,M,P);
[clmod,cmod] = smpccl(pmod,imod,Ks);
```

```
smpcpole(cmod)
Ks = smpccon(imod,ywt,uwt,M,P);
[y,u] = smpcsim(pmod,imod,Ks,tend,r,ulim,Kest,z,v,w);
assignin('base', 'y',y);
assignin('base', 'delt',delt);
assignin('base', 'u',u);
axes(handles.axes1);
plot(y); hold; plot(x,R1); plot(x,R2);
title('Çıkış');
legend(
'y1(t)','r1(t)','r2(t)','y2(t)','Location','NorthEast');legend
boxoff;
axes(handles.axes2);
plot(u); title('Ayarlanabilir değişkenler');
legend('u1(t)','u2(t)','Location','NorthEast'); legend boxoff;

function pushbutton6_Callback(hObject, eventdata, handles)
close
```

EK III-1 Sistem değişkenleri için tasarlanan yardımcı GUI'nin m file üzerine girilen kodlar

```
function save_Callback(hObject, eventdata, handles)
statemerkezGUIhandle= statemerkez;
statemerkezGUIdata= guidata(statemerkezGUIhandle);
pay1=get(handles.pay1, 'String');
set(statemerkezGUIdata.pay1,'string',pay1)
pay2=get(handles.pay2, 'String');
set(statemerkezGUIdata.pay2,'string',pay2)
pay3=get(handles.pay3, 'String');
set(statemerkezGUIdata.pay3,'string',pay3)
pay4=get(handles.pay4, 'String');
set(statemerkezGUIdata.pay4,'string',pay4)
pay5=get(handles.pay5, 'String');
set(statemerkezGUIdata.pay5,'string',pay5)
pay6=get(handles.pay6, 'String');
set(statemerkezGUIdata.pay6,'string',pay6)
us1=get(handles.us1, 'String');
set(statemerkezGUIdata.us1,'string',us1)
us2=get(handles.us2, 'String');
set(statemerkezGUIdata.us2,'string',us2)
us3=get(handles.us3, 'String');
set(statemerkezGUIdata.us3,'string',us3)
us4=get(handles.us4, 'String');
set(statemerkezGUIdata.us4,'string',us4)
us5=get(handles.us5, 'String');
set(statemerkezGUIdata.us5,'string',us5)
us6=get(handles.us6, 'String');
set(statemerkezGUIdata.us6,'string',us6)
payda11=get(handles.payda11, 'String');
set(statemerkezGUIdata.payda11,'string',payda11)
payda22=get(handles.payda22, 'String');
set(statemerkezGUIdata.payda22,'string',payda22)
payda33=get(handles.payda33, 'String');
set(statemerkezGUIdata.payda33,'string',payda33)
payda44=get(handles.payda44, 'String');
set(statemerkezGUIdata.payda44,'string',payda44)
payda55=get(handles.payda55, 'String');
```

```
set(statemerkezGUIdata.payda55,'string',payda55)
payda66=get(handles.payda66, 'String');
set(statemerkezGUIdata.payda66,'string',payda66)
payda1=get(handles.payda1, 'String');
set(statemerkezGUIdata.payda1,'string',payda1)
payda2=get(handles.payda2, 'String');
set(statemerkezGUIdata.payda2,'string',payda2)
payda3=get(handles.payda3, 'String');
set(statemerkezGUIdata.payda3,'string',payda3)
payda4=get(handles.payda4, 'String');
set(statemerkezGUIdata.payda4,'string',payda4)
payda5=get(handles.payda5, 'String');
set(statemerkezGUIdata.payda5,'string',payda5)
payda6=get(handles.payda6, 'String');
set(statemerkezGUIdata.payda6,'string',payda6)
guidata(statemerkez, statemerkezGUIdata);
close(statesistem);
```

EK III-2 Kontrol değişkenleri için tasarlanan yardımcı GUI'nin m file üzerine girilen kodlar

```
function save_Callback(hObject, eventdata, handles)
statemerkezGUIhandle= statemerkez;
statemerkezGUIdata= guidata(statemerkezGUIhandle);
ywt = get(handles.ywt,'String');
set(statemerkezGUIdata.ywt,'string',ywt)
uwt = get(handles.uwt,'String');
set(statemerkezGUIdata.uwt,'string',uwt)
M = get(handles.M,'String');
set(statemerkezGUIdata.M,'string',M)
P = get(handles.P,'String');
set(statemerkezGUIdata.P,'string',P)
R1 = get(handles.R1,'String');
set(statemerkezGUIdata.R1,'string',R1)
R2 = get(handles.R2,'String');
set(statemerkezGUIdata.R2,'string',R2)
tend = get(handles.tend,'String');
set(statemerkezGUIdata.tend,'string',tend)
w= get(handles.w,'String');
set(statemerkezGUIdata.w,'string',w)
guidata(statemerkez, statemerkezGUIdata);
close(statekontrol);
```

EK IV ÇOK GİRİŞLİ ÇOK ÇIKIŞLI BİR SİSTEMİN KISITLAMALI MODEL ÖNGÖRÜLÜ KONTROLÜ TASARIMI KODLARI

```
varargout{1} = handles.output;
[resim1]=imread('denklem4.bmp');
axes(handles.axes5);
imshow(resim1)
[resim2]=imread('durum1.bmp');
axes(handles.axes6);
imshow(resim2)

function sistem_Callback(hObject, eventdata, handles)
spacesistem()
```

```
function kontrol_Callback(hObject, eventdata, handles)
spacekontrol()
function ana_Callback(hObject, eventdata, handles)
close

function sonuc_Callback(hObject, eventdata, handles)
pay1=get(handles.pay1, 'String');
pay1=str2num(pay1);
pay2=get(handles.pay2, 'String');
pay2=str2num(pay2);
pay3=get(handles.pay3, 'String');
pay3=str2num(pay3);
pay4=get(handles.pay4, 'String');
pay4=str2num(pay4);
%
payda1=get(handles.payda1, 'String');
payda1=str2num(payda1);
payda2=get(handles.payda2, 'String');
payda2=str2num(payda2);
payda3=get(handles.payda3, 'String');
payda3=str2num(payda3);
payda4=get(handles.payda4, 'String');
payda4=str2num(payda4);
%
payda11=get(handles.payda11, 'String');
payda11=str2num(payda11);
payda22=get(handles.payda22, 'String');
payda22=str2num(payda22);
payda33=get(handles.payda33, 'String');
payda33=str2num(payda33);
payda44=get(handles.payda44, 'String');
payda44=str2num(payda44);
%
us1=get(handles.us1, 'String');
us1=str2num(us1);us1=-1*us1;
us2=get(handles.us2, 'String');
us2=str2num(us2);us2=-1*us2;
us3=get(handles.us3, 'String');
us3=str2num(us3);us3=-1*us3;
us4=get(handles.us4, 'String');
us4=str2num(us4);us4=-1*us4;
%
y3=get(handles.y3, 'String');
y3=str2num(y3);
y4=get(handles.y4, 'String');
y4=str2num(y4);
y5=get(handles.y5, 'String');
y5=str2num(y5);
y6=get(handles.y6, 'String');
y6=str2num(y6);
%
u1=get(handles.u1, 'String');
u1=str2num(u1);
u2=get(handles.u2, 'String');
u2=str2num(u2);
u3=get(handles.u3, 'String');
u3=str2num(u3);
u4=get(handles.u4, 'String');
u4=str2num(u4);
u5=get(handles.u5, 'String');
u5=str2num(u5);
```

```
u6=get(handles.u6, 'String');
u6=str2num(u6);
M=get(handles.M, 'String');
M=str2num(M);
R1=get(handles.R1, 'String');
R1=str2num(R1);
R2=get(handles.R2, 'String');
R2=str2num(R2);
r=[R1 R2];
%
ylim=[y3 y4 y5 y6];
ulim=[u1 u2 u3 u4 u5 u6];
P=get(handles.P, 'String');
P=str2num(P);
tend=get(handles.tend, 'String');
tend=str2num(tend);
delt=3;
ny=2;
ywt=[];
uwt=[1 1];
g11 = poly2tfd(pay1,[payda1 payda11],0,us1);
g21 = poly2tfd(pay3,[payda3 payda33],0,us3);
g12 = poly2tfd(pay2,[payda2 payda22],0,us2);
g22 = poly2tfd(pay4,[payda4 payda44],0,us4);
imod=tfd2mod(delt,ny,g11,g21,g12,g22);
pmod=imod
[y,u]=scmpc(pmod,imod,ywt,uwt,M,P,tend,r,ulim,ylim);
x=0:0.1:10;
axes(handles.axes3);
plot(y); hold; plot(x,R1); plot(x,R2);
title('Çıkış');
legend(
'y1(t)','r1(t)','r2(t)','y2(t)','Location','NorthEast');legend
boxoff;
axes(handles.axes4);
plot(u);title('Ayarlanabilir değişkenler');
legend('u1(t)','u2(t)','Location','NorthEast'); legend boxoff;
```

EK IV-1 Sistem değişkenleri için tasarlanan yardımcı GUI'nin m file üzerine girilen kodlar

```
function pushbutton1_Callback(hObject, eventdata, handles)
spacemerkezGUIhandle= spacemerkez;
spacemerkezGUIdata= guidata(spacemerkezGUIhandle);
pay1=get(handles.pay1, 'String');
set(spacemerkezGUIdata.pay1,'string',pay1)
pay2=get(handles.pay2, 'String');
set(spacemerkezGUIdata.pay2,'string',pay2)
pay3=get(handles.pay3, 'String');
set(spacemerkezGUIdata.pay3,'string',pay3)
pay4=get(handles.pay4, 'String');
set(spacemerkezGUIdata.pay4,'string',pay4)
us1=get(handles.us1, 'String');
set(spacemerkezGUIdata.us1,'string',us1)
us2=get(handles.us2, 'String');
set(spacemerkezGUIdata.us2,'string',us2)
us3=get(handles.us3, 'String');
set(spacemerkezGUIdata.us3,'string',us3)
us4=get(handles.us4, 'String');
```

```
set(spacemerkezGUIdata.us4,'string',us4)
payda11=get(handles.payda11, 'String');
set(spacemerkezGUIdata.payda11,'string',payda11)
payda22=get(handles.payda22, 'String');
set(spacemerkezGUIdata.payda22,'string',payda22)
payda33=get(handles.payda33, 'String');
set(spacemerkezGUIdata.payda33,'string',payda33)
payda44=get(handles.payda44, 'String');
set(spacemerkezGUIdata.payda44,'string',payda44)
payda1=get(handles.payda1, 'String');
set(spacemerkezGUIdata.payda1,'string',payda1)
payda2=get(handles.payda2, 'String');
set(spacemerkezGUIdata.payda2,'string',payda2)
payda3=get(handles.payda3a, 'String');
set(spacemerkezGUIdata.payda3,'string',payda3)
payda4=get(handles.payda4, 'String');
set(spacemerkezGUIdata.payda4,'string',payda4)
guidata(spacemerkez, spacemerkezGUIdata);
close(spacesistem);
```

EK IV-2 Kontrol değişkenleri için tasarlanan yardımcı GUI'nin m file üzerine girilen kodlar

```
function pushbutton1_Callback(hObject, eventdata, handles)
spacemerkezGUIhandle= spacemerkez;
spacemerkezGUIdata= guidata(spacemerkezGUIhandle);
y3 = get(handles.y3,'String');
set(spacemerkezGUIdata.y3,'string',y3)
y4 = get(handles.y4,'String');
set(spacemerkezGUIdata.y4,'string',y4)
y5 = get(handles.y5,'String');
set(spacemerkezGUIdata.y5,'string',y5)
y6 = get(handles.y6,'String');
set(spacemerkezGUIdata.y6,'string',y6)
u1 = get(handles.u1,'String');
set(spacemerkezGUIdata.u1,'string',u1)
u2 = get(handles.u2,'String');
set(spacemerkezGUIdata.u2,'string',u2)
u3 = get(handles.u3,'String');
set(spacemerkezGUIdata.u3,'string',u3)
u4 = get(handles.u4,'String');
set(spacemerkezGUIdata.u4,'string',u4)
u5 = get(handles.u5,'String');
set(spacemerkezGUIdata.u5,'string',u5)
u6 = get(handles.u6,'String');
set(spacemerkezGUIdata.u6,'string',u6)
M = get(handles.M,'String');
set(spacemerkezGUIdata.M,'string',M)
P = get(handles.P,'String');
set(spacemerkezGUIdata.P,'string',P)
R1 = get(handles.R1,'String');
set(spacemerkezGUIdata.R1,'string',R1)
R2 = get(handles.R2,'String');
set(spacemerkezGUIdata.R2,'string',R2)
tend = get(handles.tend,'String');
set(spacemerkezGUIdata.tend,'string',tend)
guidata(spacemerkez, spacemerkezGUIdata);
close(spacekontrol);
```

EK V ÇOK GİRİŞLİ ÇOK ÇIKIŞLI BİR SİSTEMİN DURUM UZAY MATRİSLERİ İLE KISITLAMALI MODEL ÖNGÖRÜLÜ KONTROLÜ TASARIMI KODLARI

```
varargout{1} = handles.output;
[resim2]=imread('paperr.bmp');
axes(handles.axes4);
imshow(resim2)

function pushbutton1_Callback(hObject, eventdata, handles)
papersistem();

function pushbutton2_Callback(hObject, eventdata, handles)
paperkontrol();

function pushbutton3_Callback(hObject, eventdata, handles)
%-------------A--------------%
a11=get(handles.a11, 'String');
a11=str2num(a11);
a12=get(handles.a12, 'String');
a12=str2num(a12);
a13=get(handles.a13, 'String');
a13=str2num(a13);
a14=get(handles.a14, 'String');
a14=str2num(a14);
a21=get(handles.a21, 'String');
a21=str2num(a21);
a22=get(handles.a22, 'String');
a22=str2num(a22);
a23=get(handles.a23, 'String');
a23=str2num(a23);
a24=get(handles.a24, 'String');
a24=str2num(a24);
a31=get(handles.a31, 'String');
a31=str2num(a31);
a32=get(handles.a32, 'String');
a32=str2num(a32);
a33=get(handles.a33, 'String');
a33=str2num(a33);
a34=get(handles.a34, 'String');
a34=str2num(a34);
a41=get(handles.a41, 'String');
a41=str2num(a41);
a42=get(handles.a42, 'String');
a42=str2num(a42);
a43=get(handles.a43, 'String');
a43=str2num(a43);
a44=get(handles.a44, 'String');
a44=str2num(a44);
%-------------B----------------%
b11=get(handles.b11, 'String');
b11=str2num(b11);
b12=get(handles.b12, 'String');
b12=str2num(b12);
b13=get(handles.b13, 'String');
b13=str2num(b13);
b14=get(handles.b14, 'String');
```

```
b14=str2num(b14);
b21=get(handles.b21, 'String');
b21=str2num(b21);
b22=get(handles.b22, 'String');
b22=str2num(b22);
b23=get(handles.b23, 'String');
b23=str2num(b23);
b24=get(handles.b24, 'String');
b24=str2num(b24);
b31=get(handles.b31, 'String');
b31=str2num(b31);
b32=get(handles.b32, 'String');
b32=str2num(b32);
b33=get(handles.b33, 'String');
b33=str2num(b33);
b34=get(handles.b34, 'String');
b34=str2num(b34);
b41=get(handles.b41, 'String');
b41=str2num(b41);
b42=get(handles.b42, 'String');
b42=str2num(b42);
b43=get(handles.b43, 'String');
b43=str2num(b43);
b44=get(handles.b44, 'String');
b44=str2num(b44);
%----------------C--------------%
c11=get(handles.c11, 'String');
c11=str2num(c11);
c12=get(handles.c12, 'String');
c12=str2num(c12);
c13=get(handles.c13, 'String');
c13=str2num(c13);
c14=get(handles.c14, 'String');
c14=str2num(c14);
c21=get(handles.c21, 'String');
c21=str2num(c21);
c22=get(handles.c22, 'String');
c22=str2num(c22);
c23=get(handles.c23, 'String');
c23=str2num(c23);
c24=get(handles.c24, 'String');
c24=str2num(c24);
c31=get(handles.c31, 'String');
c31=str2num(c31);
c32=get(handles.c32, 'String');
c32=str2num(c32);
c33=get(handles.c33, 'String');
c33=str2num(c33);
c34=get(handles.c34, 'String');
c34=str2num(c34);
%---------------D---------------%
d11=get(handles.d11, 'String');
d11=str2num(d11);
d12=get(handles.d12, 'String');
d12=str2num(d12);
d13=get(handles.d13, 'String');
d13=str2num(d13);
d14=get(handles.d14, 'String');
d14=str2num(d14);
d21=get(handles.d21, 'String');
d21=str2num(d21);
```

```
d22=get(handles.d22, 'String');
d22=str2num(d22);
d23=get(handles.d23, 'String');
d23=str2num(d23);
d24=get(handles.d24, 'String');
d24=str2num(d24);
d31=get(handles.d31, 'String');
d31=str2num(d31);
d32=get(handles.d32, 'String');
d32=str2num(d32);
d33=get(handles.d33, 'String');
d33=str2num(d33);
d34=get(handles.d34, 'String');
d34=str2num(d34);
%--------------------------------%
A=[a11 a12 a13 a14; a21 a22 a23 a24; a31 a32 a33 a34; a41 a42 a43
a44];
B=[b11 b12 b13 b14; b21 b22 b23 b24; b31 b32 b33 b34; b41 b42 b43
b44];
C=[c11 c12 c13 c14; c21 c22 c23 c24; c31 c32 c33 c34];
D=[d11 d12 d13 d14 ; d21 d22 d23 d24 ; d31 d32 d33 d34];
%-----------------------------------------------------------------
dt=2;
[PHI,GAM]=c2dmp(A,B,dt);
minfo=[dt,4,2,1,1,3,0];
imod=ss2mod(PHI,GAM,C,D,minfo);
%-----------------------------------------------------------
yl3=get(handles.yl3, 'String');
yl3=str2num(yl3);
yl2=get(handles.yl2, 'String');
yl2=str2num(yl2);
yl1=get(handles.yl1, 'String');
yl1=str2num(yl1);
%
ul3=get(handles.ul3, 'String');
ul3=str2num(ul3);
ul2=get(handles.ul2, 'String');
ul2=str2num(ul2);
ul1=get(handles.ul1, 'String');
ul1=str2num(ul1);
%
yw1=get(handles.yw1, 'String');
yw1=str2num(yw1);
yw2=get(handles.yw2, 'String');
yw2=str2num(yw2);
yw3=get(handles.yw3, 'String');
yw3=str2num(yw3);
%
uw1=get(handles.uw1, 'String');
uw1=str2num(uw1);
uw2=get(handles.uw2, 'String');
uw2=str2num(uw2);
%
tend=get(handles.tend, 'String');
tend=str2num(tend);
M=get(handles.M, 'String');
M=str2num(M);
P=get(handles.P, 'String');
P=str2num(P);
R1=get(handles.r1, 'String');
R1=str2num(R1);
```

```
R2=get(handles.r2, 'String');
R2=str2num(R2);
R3=get(handles.r3, 'String');
R3=str2num(R3);
r=[R1 R2 R3];
ylim=[yl3 yl3  yl2 yl2  yl1 yl1];
ulim=[-10*[ul3 ul3] 10*[ul2 ul2] 2*[ul1 ul1]];
ywt=[yw3,yw2,yw1];
uwt=0.6*[uw2 uw1];
Kest=[];
pmod=imod;
setpts=[R1 R2 R3];
[y,u,ym] = scmpc(pmod,imod,ywt,uwt,M,P,tend,setpts,ulim,ylim,Kest);
axes(handles.axes1);
x=0:0.2:35;
plot(y);hold; plot(ym); plot(x,R1); plot(x,R2); plot(x,R3);
title('Çıkış');
legend( 'y1-H1(t)','r1(t)','y3-N2(t)','r2(t)','y2-
N1(t)','r3(t)','Location','NorthEast');legend boxoff;
axes(handles.axes2);
plot(u); title('Ayarlanabilir değişkenler');
title('Ayarlanabilir değişkenler');
legend('u1(t)','u2(t)','Location','NorthEast'); legend boxoff;
function ana_Callback(hObject, eventdata, handles)
close(paper)
```

EK V-1 Durum uzay matrisleri için tasarlanan yardımcı GUI'nin m file üzerine girilen kodlar

```
paperGUIhandle= paper;
paperGUIdata= guidata(paperGUIhandle);
a11 = get(handles.a11,'String');
set(paperGUIdata.a11,'string',a11)
a12 = get(handles.a12,'String');
set(paperGUIdata.a12,'string',a12)
a13 = get(handles.a13,'String');
set(paperGUIdata.a13,'string',a13)
a14 = get(handles.a14,'String');
set(paperGUIdata.a14,'string',a14)
a21 = get(handles.a21,'String');
set(paperGUIdata.a21,'string',a21)
a22 = get(handles.a22,'String');
set(paperGUIdata.a22,'string',a22)
a23 = get(handles.a23,'String');
set(paperGUIdata.a23,'string',a23)
a24 = get(handles.a24,'String');
set(paperGUIdata.a24,'string',a24)
a31 = get(handles.a31,'String');
set(paperGUIdata.a31,'string',a31)
a32 = get(handles.a32,'String');
set(paperGUIdata.a32,'string',a32)
a33 = get(handles.a33,'String');
set(paperGUIdata.a33,'string',a33)
a34 = get(handles.a34,'String');
set(paperGUIdata.a34,'string',a34)
a41 = get(handles.a41,'String');
set(paperGUIdata.a41,'string',a41)
a42 = get(handles.a42,'String');
```

```
set(paperGUIdata.a42,'string',a42)
a43 = get(handles.a43,'String');
set(paperGUIdata.a43,'string',a43)
a44 = get(handles.a44,'String');
set(paperGUIdata.a44,'string',a44)
b11 = get(handles.b11,'String');
set(paperGUIdata.b11,'string',b11)
b12 = get(handles.b12,'String');
set(paperGUIdata.b12,'string',b12)
b13 = get(handles.b13,'String');
set(paperGUIdata.b13,'string',b13)
b14 = get(handles.b14,'String');
set(paperGUIdata.b14,'string',b14)
b21 = get(handles.b21,'String');
set(paperGUIdata.b21,'string',b21)
b22 = get(handles.b21,'String');
set(paperGUIdata.b22,'string',b22)
b23 = get(handles.b23,'String');
set(paperGUIdata.b23,'string',b23)
b24 = get(handles.b24,'String');
set(paperGUIdata.b24,'string',b24)
b31 = get(handles.b31,'String');
set(paperGUIdata.b31,'string',b31)
b32 = get(handles.b32,'String');
set(paperGUIdata.b32,'string',b32)
b33 = get(handles.b33,'String');
set(paperGUIdata.b33,'string',b33)
b34 = get(handles.b34,'String');
set(paperGUIdata.b34,'string',b34)
b41 = get(handles.b41,'String');
set(paperGUIdata.b41,'string',b41)
b42 = get(handles.b42,'String');
set(paperGUIdata.b42,'string',b42)
b43 = get(handles.b43,'String');
set(paperGUIdata.b43,'string',b43)
b44 = get(handles.b44,'String');
set(paperGUIdata.b44,'string',b44)
c11 = get(handles.c11,'String');
set(paperGUIdata.c11,'string',c11)
c12 = get(handles.c12,'String');
set(paperGUIdata.c12,'string',c12)
c13 = get(handles.c13,'String');
set(paperGUIdata.c13,'string',c13)
c14 = get(handles.c14,'String');
set(paperGUIdata.c14,'string',c14)
c21 = get(handles.c21,'String');
set(paperGUIdata.c21,'string',c21)
c22 = get(handles.c21,'String');
set(paperGUIdata.c22,'string',c22)
c23 = get(handles.c23,'String');
set(paperGUIdata.c23,'string',c23)
c24 = get(handles.c24,'String');
set(paperGUIdata.c24,'string',c24)
c31 = get(handles.c31,'String');
set(paperGUIdata.c31,'string',c31)
c32 = get(handles.c32,'String');
set(paperGUIdata.c32,'string',c32)
c33 = get(handles.c33,'String');
set(paperGUIdata.c33,'string',c33)
c34 = get(handles.c34,'String');
set(paperGUIdata.c34,'string',c34)
```

```
d11 = get(handles.d11,'String');
set(paperGUIdata.d11,'string',d11)
d12 = get(handles.d12,'String');
set(paperGUIdata.d12,'string',d12)
d13 = get(handles.d13,'String');
set(paperGUIdata.d13,'string',d13)
d14 = get(handles.d14,'String');
set(paperGUIdata.d14,'string',d14)
d21 = get(handles.d21,'String');
set(paperGUIdata.d21,'string',d21)
d22 = get(handles.d21,'String');
set(paperGUIdata.d22,'string',d22)
d23 = get(handles.d23,'String');
set(paperGUIdata.d23,'string',d23)
d24 = get(handles.d24,'String');
set(paperGUIdata.d24,'string',d24)
d31 = get(handles.d31,'String');
set(paperGUIdata.d31,'string',d31)
d32 = get(handles.d32,'String');
set(paperGUIdata.d32,'string',d32)
d33 = get(handles.d33,'String');
set(paperGUIdata.d33,'string',d33)
d34 = get(handles.d34,'String');
set(paperGUIdata.d34,'string',d34)
guidata(paper, paperGUIdata);
close(papersistem);
```

EK V-2 Kontrol değişkenleri için tasarlanan yardımcı GUI'nin m file üzerine girilen kodlar

```
function pushbutton1_Callback(hObject, eventdata, handles)
paperGUIhandle= paper;
paperGUIdata= guidata(paperGUIhandle);
yl3 = get(handles.yl3,'String');
set(paperGUIdata.yl3,'string',yl3)
yl2 = get(handles.yl2,'String');
set(paperGUIdata.yl2,'string',yl2)
yl1 = get(handles.yl1,'String');
set(paperGUIdata.yl1,'string',yl1)
ul1 = get(handles.ul1,'String');
set(paperGUIdata.ul1,'string',ul1)
ul2 = get(handles.ul2,'String');
set(paperGUIdata.ul2,'string',ul2)
ul3 = get(handles.ul3,'String');
set(paperGUIdata.ul3,'string',ul3)
yw1 = get(handles.yw1,'String');
set(paperGUIdata.yw1,'string',yw1)
yw2 = get(handles.yw2,'String');
set(paperGUIdata.yw2,'string',yw2)
yw3 = get(handles.yw3,'String');
set(paperGUIdata.yw3,'string',yw3)
uw1 = get(handles.uw1,'String');
set(paperGUIdata.uw1,'string',uw1)
uw2 = get(handles.uw2,'String');
set(paperGUIdata.uw2,'string',uw2)
M = get(handles.M,'String');
set(paperGUIdata.M,'string',M)
P = get(handles.P,'String');
set(paperGUIdata.P,'string',P)
```

```
r1 = get(handles.r1,'String');
set(paperGUIdata.r1,'string',r1)
r2 = get(handles.r2,'String');
set(paperGUIdata.r2,'string',r2)
r3 = get(handles.r3,'String');
set(paperGUIdata.r3,'string',r3)
tend = get(handles.tend,'String');
set(paperGUIdata.tend,'string',tend)
guidata(paper, paperGUIdata);
close(paperkontrol)
```

ÖZGEÇMİŞ

Eray YILMAZLAR, 17 Kasım 1988 tarihinde İstanbulda doğdum. Orta öğretimimi 2005 yılında Küçükköy Endüstri Meslek Lisesinde başarıyla tamamladım. Yükseköğretimime 2005 yılında Marmara Üniversitesi Teknik Eğitim Fakultesi Mekatronik Öğretmenliği bölümünde başladım. 2009 yılında teknik eğitim fakultesinden mevzun olduktan sonra Maramara Üniversitesi Fen bilimleri Enstitüsü Mekatronik Anabilim dalında öğrenime halan devam etmekteyim.