İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ANTROPOMORFİK ROBOTLARIN DİNAMİĞİ VE ADAPTİF KONTROL UYGULAMALARI: MATLAB/SİMULİNK MODELLEME

YÜKSEK LİSANS TEZİ

Muhammet ÖZTÜRK

Uçak ve Uzay Mühendisliği Anabilim Dalı

Uçak ve Uzay Mühendisliği Programı

MAYIS 2014

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ANTROPOMORFİK ROBOTLARIN DİNAMİĞİ VE ADAPTİF KONTROL UYGULAMALARI: MATLAB/SİMULİNK MODELLEME

YÜKSEK LİSANS TEZİ

Muhammet ÖZTÜRK
511111126

Uçak ve Uzay Mühendisliği Anabilim Dalı

Uçak ve Uzay Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Elbrus CAFEROV

MAYIS 2014

İTÜ, Fen Bilimleri Enstitüsü'nün 511111126 numaralı Yüksek Lisans Öğrencisi **Muhammet ÖZTÜRK**, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı **"ANTROPOMORFİK ROBOTLARIN DİNAMİĞİ VE ADAPTİF KONTROL UYGULAMALARI: MATLAB/SİMULİNK MODELLEME"** başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmustur.

Tez Danışmanı :	**Prof. Dr. Elbrus CAFEROV** İstanbul Teknik Üniversitesi	.....
Jüri Üyeleri :	**Yrd. Doç. Dr. Turgut Berat Karyot** İstanbul Teknik Üniversitesi	.....
	Yrd. Doç. Dr. S. Murat Yeşiloğlu İstanbul Teknik Üniversitesi	.....

Teslim Tarihi : **2 Mayıs 2014**
Savunma Tarihi : **28 Mayıs 2014**

Aileme,

ÖNSÖZ

Değerli tez danışman hocam Prof. Dr. Elbrus CAFEROV'a bilgece yönlendirmeleri ve şevk veren önerileri için teşekkürü borç bilirim.
Tez çalışmalarında katkılarından ve desteklerinden ötürü Prof. Dr. Hakan TEMELTAŞ, Yar. Doç. Dr. Turgut Berat KARYOT ve Yar. Doç. Dr. S. Murat YEŞİLOĞLU'na yardımlarından ötürü arkadaşım Günay YILDIZ'a, öğrenim hayatım boyunca maddi ve manevi desteklerini hiçbir zaman esirgemeyen değerli aileme sonsuz teşekkürlerimi sunarım.

Mayıs 2014

Muhammet ÖZTÜRK
Uçak ve Uzay Mühendisi

İÇİNDEKİLER

Sayfa

KISALTMALAR

DH **:** Denavit Hartenberg
SOA **:** Uzaysal Operator Cebri
DQ **:** İkili Quaternion
PD **:** Oransal Türevsel Kontrol
CTC **:** Hesaplanmış Tork Kontrolü

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

ANTROPOMORFİK ROBOTLARIN DİNAMİĞİ VE ADAPTİF KONTROL UYGULAMALARI: MATLAB/SİMULİNK MODELLEME

ÖZET

Robotlar günümüzde bir çok alanda insanların yaşayışlarını ve işlerini kolaylaştırmak için kullanılan mekanizmalardır. İlk zamanlarda basit makineler şeklinde dizayn edilmiş olan robotlar zamanla ilerlemiş ve insanımsı yapılara dönüşmüşlerdir.

Robotların tarihteki gelişimi incelendiğinde karşımıza robot alanında ilk çalışmaları yapan ve ilk sibernetikçi kabul edilen Ebul-iz İsmail bin ar-Razzaz el-Cezeri çıkmaktadır. Ebul-iz İsmail bin ar-Razzaz el-Cezeri 1205-1206 yıllarında yazdığı "Kitab-ül'-Camü Beyne'l-İlmi-i ve'l-amelen-Nafi' Fi Sınaati'l-Hiyel" adlı kitabın içinde, 300'e yakın otomatik makine ve sistemleri ile ilgili bilgi verdikten sonra çalışma özelliklerini şemalarla göstermiştir. Sadece suyun kaldırma ve basınç gücünü kullanarak tamamen yeni bir teknik ve sistem kurmuş, çok yönlü otomatik hareketler elde edebilmiştir. Kurmuş olduğu otomatik sistemlerde ses (kuş, davul, zurna, ıslık vs) ya da çığlık çıkması gerektiği anda bu sesleri de sağlayabilmiştir.

17. ve 18. yüzyıllarda Avrupa'da ilkel otomatlar bulunmuştur. 1940'lı yıllardan sonra gelişen teknolojiye bağlı olarak robot teknolojisinde büyük gelişmeler görülür ve endüstriyel robotların üretimi başlar.

Günümüzde robotların sanayi robotu, insansı robotlar gibi değişik çeşitleri bulunmaktadır. Bizim incelediğimiz kısım endüstriyel robotlardır. Endüstriyel robot; genel amaçlı, insana benzer özelliklere sahip programlanabilir bir makinedir. Bu robotun insana benzeyen en önemli özelliği onun koludur. Tutma ve yerleştirme işlemlerinde robot kolu kullanılır. Robot kolu, başka bir makineyle birleştirilerek, malzemenin yüklenmesi ve takım değiştirme işlemini yapmaktadır. Kesme, şekil verme, yüzey kaplama, silindirik ve düzlem yüzey taşlama gibi imalat işlemlerini gerçekleştirir. Montaj ve kontrol uygulamalarında kullanılmaktadır. Genellikle hantal ve sabit konumludurlar.

Bu çalşımada kullanılan sanayi robotu ABB firmasına ait IRB 140 robotudur. Bu robot üzerinde kullanılan gerçek veriler üzerinden kinematik ve dinamik kontrolleri yapılmıştır. Kontrol sistemleri genel olarak lineer ve lineer olmayan kontrol sistemleri olarak ikiye ayrılmaktadır. 20. yuzyılın ikinci yarısından itibaren lineer kontrol sistemlerinin yetersiz kaldığı bir çok durum meydana çıkmaya başlamıştır.

İlk olarak uçaklarda karşılaşılan lineer sistemlerin yetersizliği problemi lineer olmayan yöntemlerin kullanılmaya başlanılmasıyla çözüme kavuşmaya başlamıştır. Ancak şu da belirtilmelidir ki lineer olmayan kontrol sistem tasarımları lineer sistemlere göre daha zordur. Lineer olmayan kontrol sistemleri ile alakalı bir çok yönteme ulaşmak mümkündür. Ancak hem uçaklarda hem de robotlarda kullanılan genel yöntemler olarak gürbüz kontrol, uyarlamalı kontrol gibi sistemler

kullanılmaktadır. Bunların yanında robotlarda kullanılan kayma kipli kontrol yöntemi de bulunmaktadır.

Bütün eklemlerin bütün verilerinin çok iyi bir şekilde bilindiği durumlarda hesaplanmış tork yöntemi ile sistemi çözmek ve iyi sonuçlara ulaşmak mümkündür ancak sistemin bütün parametreleri hatasız olarak bilebilmek mümkün olmayabilmektedir. Bunun için kullanılan yöntemlerden uyarlamalı kontrol sistemi üzerinde durulmuştur. Uyarlamalı kontrol yönteminde sistemin parametrik hataları giderilebilmiştir.

Bu yöntemlerde ve çalışmalarda Denavit-Hartenberg yöntemi ve Matlab kullanılmıştır. Robot hakkındaki bütün bilgiler ABB firmasından alınmıştır.

ANTHROPOMORPHIC ROBOT'S DYNAMICS AND ADAPTIVE CONTROL APPLICATIONS: MATLAB/SIMULINK MODELING

SUMMARY

Nowadays, robots are used to simplify the works and human lives in many areas. In the beginning, simple machines had been designed but with the time, complicated robots as humanoid robots have been structured.

When we examine the improvement of robotics in history, we meet with Ebul-iz İsmail bin ar-Razzaz el-Cezeri that made the first robotic labors. He is accepted as the first cybernetic. He has written a book named "Kitab-ül'-Camü Beyne'l-İlmi-i ve'l-amelen-Nafi' Fi Sınaati'l-Hiyel" that contains nearby 300 automatic machines and systems. In the book, all systems have been viewed clearly with diagrams.

By the water's lift force and pressure, he has used a new technic and performed very effective systems. Besides, in the automatic systems he could provide some voices like birds, timpani, clarions, whistles and screams.

In the seventeenth and eighteenth centuries, primary automatic systems have been found in Europe. Barely with the developing technologies by 1940, there have been big developments in robot technology and industrial robots started to be produced.

Now there are types of robots like industrial, humanoid etc. We are interested in industrial robotics. Industrial robots are used for general purposes and can behave like human.

Industrial robots generally designed as an arm. All the works have been done with that arm. Those arms can work with each other and can some works like loading material, changing tools, cutting, forming, coating and grinding cylindrical and plane surface. Those arms are generally cumbersome and fixed.

Robotic systems that being used in the industry have a rising line that any machine in industry can only be used for a purpose. A robot can be used for several purposes like human. The robots can be serial or parallel.

The serial robots like human and can do many things like human and can have some solutions for barriers. However, the parallel robots cannot behave like the serials. They cannot move like human but if you need very big resolution and powers in a restricted area, you must use parallel. Generally, in industry, serial robots are being used and our work is about serial robots.

When you start to work in the field of robotics, you must coordinate the robotic systems. There are simple diversity between plane and robots that was shown in thesis. There are mainly two methods to find equations as spatial operator algebra (SOA) and Denavit Hartenberg (DH). In this thesis, Denavit Hartenberg has been

used and examined in detail. DH is a simple method to learn and to use so DH is being used in many robotic systems continuously.

In the main idea in DH robot needs to a main coordinate axis and depending on the main axis other axes being determined. In DH, you need coordinate axes for every link and all of the axes must be related to previous one. How to determine all the axes have shown in thesis.

In thesis, there are mainly two problems that kinematic and dynamic. Kinematic system has two titles that forward and inverse kinematics. Forward kinematic works from links to tip point namely you know the link's angle and location values and so you can find the tip point. Inverse kinematic determine the link's values from tip points. Inverse kinematic must be calculated to work in robotic area. Kinematic systems have been examined in second and third chapters.

Dynamic systems have the same situations like kinematic systems that are forward dynamic and inverse dynamics. In forward dynamics, we have the torque values for every links, we find acceleration values for each links, and so we can find acceleration and so velocity, location at any moment. In the inverse dynamics, we have the link's location, velocity and acceleration values at any moment so we get the torque values for every links. Each of these systems are nonlinear and so control system must be nonlinear but when we put the inverse and forward dynamics one after another we have inputs as location, velocity and acceleration and have outputs as acceleration, velocity and location.

There are some types of controllers for systems like robots. Generally, there are two ways to solve the control problems as linear and nonlinear systems. Linear systems are simple and easily applicable and so in first times generally used but linear control systems was not enough for some systems. Because some systems are so complicated and have nonlinear differential equations.

Firstly, airplane systems needed to nonlinear control systems. The nonlinear systems were successful so they were used in the other systems like robots. The dynamics systems are nonlinear systems and control systems must be nonlinear. In this thesis some control mechanism has been dealt like computed torque, robust and adaptive.

Computed torque gives fast and good results. It works both forward and inverse dynamic systems. If there is not any parametric fault, the system works very good and systems become like linear. However, if there are some parametric errors the computed torque cannot tolerate the error. Robust control systems tolerate the error that comes from outside but dynamic model must be very good. In adaptive control systems dynamic model does not have to be perfect that adaptive control system tolerates the errors that come from model as link mass, length and any others, besides tolerate the faults from outer region like robust do.

In mathematical equations adaptive are similar to computed torque but we have some addition on it. Namely, the dynamic equation is accepted as combination of two equations as regression matrix and parameter faults. When you study on that terminology, you reach to a control equation that related to parametric faults. This mathematical structure has been shown in thesis.

In this thesis, adaptive control systems have been used. Firstly, some parameters accepted with errors and afterwards the system was run. Adaptive control system has tolerated the error but computed torque was not any effect on the error.

All results have been showed and explained in detail. Besides simulation has been done with virtual reality toolbox in Matlab.

The robot that used in the thesis was IRB 140 that belongs to ABB firm. All the knowledge and drawing folders has been taken from them like measures of link, some other specific values. That studied robot is a currently used robot that is controlled again with the real data.

All the results have been given and compared each other. Besides, some addition to system has been done and the results were compared.

1. GİRİŞ

Robot kelime olarak kökeni köle olan bir anlam ifade etmektedir ve insanların ihtiyaçlarına uygun olarak her türlü görevde kullanılabilecek sistemlerdir. Robot sistemleri disiplinlerarası bir çalışma olup mekanik, elektronik-elektronik, kontrol gibi birbirleriyle uyumlu çalışması gereken disiplinlerden oluşmaktadır.

Günümüzde bir robottan aşağıda verilen 3 özelliği yerine getirmesi beklenmektedir.

1. İşlem yapma yetisi: Bir işlemi fiziksel veya farazi olarak yerine getirebilmelidir yoksa robot olmaz sadece bir madde olur.

2. İşlemin sonucunu belirleme yetisi: İşlemi yaptıktan sonra mutlak olarak işlemin sonucunu belirlemelidir ki işlem tam olarak yapılmış olsun.

3. Karar verme yetisi : İşlemin sonucuna göre ya da dış etmenlere göre mutlaka bir yargı kurabilmelidir.

Robotların tarihteki gelişimi incelendiğinde karşımıza robot alanında ilk çalışmaları yapan ve ilk sibernetikçi kabul edilen Ebul-iz İsmail bin ar-Razzaz el-Cezeri çıkmaktadır. Ebul-iz İsmail bin ar-Razzaz el-Cezeri 1205-1206 yıllarında yazdığı "Kitab-ül'-Camü Beyne'l-İlmi-i ve'l-amelen-Nafi' Fi Sınaati'l-Hiyel" adlı kitabın içinde, 300'e yakın otomatik makine ve sistemleri ile ilgili bilgi verdikten sonra çalışma özelliklerini şemalarla göstermiştir. Sadece suyun kaldırma ve basınç gücünü kullanarak tamamen yeni bir teknik ve sistem kurmuş, çok yönlü otomatik hareketler elde edebilmiştir. Kurmuş olduğu otomatik sistemlerde ses (kuş, davul, zurna, ıslık vs) ya da çığlık çıkması gerektiği anda bu sesleri de sağlayabilmiştir.

17. ve 18. yüzyıllarda Avrupa'da ilkel otomatlar bulunmuştur. 1940'lı yıllardan sonra gelişen teknolojiye bağlı olarak robot teknolojisinde büyük gelişmeler görülür ve endüstriyel robotların üretimi başlar.

Günümüzde robotlar yapısal durumlarına göre endüstriyel ve otomasyonsal robot olarak 2 kısma ayrılabilmektedir. Bizim incelediğimiz kısım endüstriyel robotlardır. Endüstriyel robot; genel amaçlı, insana benzer özelliklere sahip programlanabilir bir

makinedir. Bu robotun insana benzeyen en önemli özelliği onun koludur. Tutma ve yerleştirme işlemlerinde robot kolu kullanılır. Robot kolu, başka bir makineyle birleştirilerek, malzemenin yüklenmesi ve takım değiştirme işlemini yapmaktadır. Kesme, şekil verme, yüzey kaplama, silindirik ve düzlem yüzey taşlama gibi imalat işlemlerini gerçekleştirir. Montaj ve kontrol uygulamalarında kullanılmaktadır. Genellikle hantal ve sabit konumludurlar.

Endüstriyel robotlar sanayide kullanılan robot tipleri olup genellikle 6 eksenli yapılardan oluşurlar. Bu tür robotlarda ilk 3 eksen robotun taşıyıcı kısmı olup gövde olarak adlandırılır. Son 3 eksen ise robotun yönelmesini sağlayan kısımdır ve bilek olarak adlandırılır. Bileklerin her yere girebilmesi için eksenleri tamamı dönel eksenlerden seçilmektedir. Bizim çalışmalarımızda incelediğimiz robot 6 eksenli endüstriyel bir robottur.

1.1 Tezin Amacı

Robot modelleme yapılırken de ilk olarak bütün verilerin kusursuz olarak bilindiği varsayılır ve bazı kabuller yapılır. Ancak gerçek bir sistem üzerinde çalışırken sistemin kütlelerinden, atalet momentlerine kadar bir çok konuda kesin bilgiye sahip olmamız neredeyse imkânsızdır. Bu tezde kullnaılan robot üzerinde kinematik ve dinamik olmak üzere matematiksel tasarımlar ve kontrol sistemleri uygulanmıştır. Robotumuzun dinamik sistemleri lineer olmayan bir sistemdir bu sebeple lineer olmayan kontrol yöntemleri uygulanmıştır. Denavit Hartenberg yöntemi ile çalışılan bu sistemde dinamik modelleme için temelde hesaplanmış tork yöntemi kullanılmıştır. Hesaplanmış tork yöntemi sistemi lineer hale getiren bir yöntemdir. Girişler eklemlerin açısal ve açıya bağlı değerleri iken çıkışlarda yanı birimde olmaktadır.

Tezimizin asıl amacı bu sistemlerin düzgün çalışmasının ardından uçaklarda olduğu gibi değişik etkilere maruz kalırsa ne gibi cevaplar verebileceğini görmek ve çözümler üretebilmektir. Parametrik hata diye belirtilen eklemlerin kütleleri üzerinde bazı değişikliklere gidilmiş ve hesaplanmış tork yönteminin bu değişimlere verdiği cevaplar incelenmiştir. Bu cevapların yeterli düzeyde olmadığı görülmüş ve uyarlamalı kontrol yöntemleri uygulanmıştır. Bu kontrol sistemi ile parametrik hatalar ortadan kaldırılarak sistemin düzenli bir şekilde çalışması garanti altına alınmıştır.

Tezin amacı belirli hatalar için belirtilen sistemlerin etkilerini inceleyip bu yöntemlerin sistemlere etkilerini belirlemek ve karşılaştırmalarını yapmaktır. Sonuçlar kısmında bu sistemlerin grafiklerle desteklenmek suretiyle karşılaştırılması yapılmıştır.

1.2 Literatür Araştırması

Robot modelleme yapılırken genel olarak kullanılan Uzaysal Operatör Cebri, İkili Quaternion ve Denavit-Hartenberg yöntemi gibi değişik yöntemler mevcuttur. Her yöntemin avantajları ve dezavantajları mevcuttur.

İkili quaternion yönteminde eklemlerin birbirleriyle alakalı bağlantıları modellenirken homojen matrisler yerine Clifford cebri diye bilinen bir cebir kullanılır. Bu cebirsel yöntem kinematik modelleme de oldukça esnek bir yapı sunar ancak ters kinematikte tekillikler karşısında ciddi sıkıntılar yaşabilmektedir [1].

Modelleme yapılırken işlem yükü önemlidir. Örnek olarak Denavit Hartenberg yönteminim işlem yükü serbestlik derecesinin karesiyle orantılı olarak artmaktadir. Bu sebeple bu üstel artışın yerine kullanılabilecek ve yüksek serbestlik derecelerinde daha az işlem yükü gerektirecek değişik yöntemler üzerinde çalışıldı ki bunlardan biri Jain ve Rodriguez'in üzerinde çalışmış olduğu kütle matrisinin duyarlılığının hesaplanması işleminde kullandığı uzaysal operatör cebridir [2]. Bu cebir Rodriguez tarafından da 1987 yılında üzerinde çalışılmış bir yöntemdir ve yüksek dereceli sistemleri daha etkili şekilde modelleme imkanı sunmaktadır [3, 4, 5]. Uzaysal Operatör Cebri serbestlik derecesi ile orantılı olarak artmaktadır. Bu durumda DH yönteminde olduğu gibi çok büyük işlem yükleri getirmemektedir [6].

Denavit Hartenberg yöntemi (Denavit ve Hartenberg) 1955 yılında iki koordinat ekseni arasındaki hareket bağıntısını ifade etmek için oluşturulmuş olan basit matematiksel ifadelerden oluşmaktadır. Bu yöntemde robotun veya makinenin eklem bağıntıları için belli bir kural çerçevesinde koordinat eksenleri atanır ve dönüşüm matrisleri ile bu koordinat eksenleri arasında ilişkiler ifade edilir. Yapılan çalışmalar göstermiştir ki Denavit-Hartenberg katı modeller için oldukça uygun ve uygulanabilirliği kolay bir yöntemdir [7]. DH yönteminin kolay ve uygulanabilirliğinin yüksek olmasının yanında bazı dezavantajlarıda bulunmaktadır. Örnek olarak 5 serbestlik derecesine kadar güzel bir çalışma sunarken 5. serbestlik

derecesinden sonrasında işlem yükü çok ağırlaşmaktadır. Bu sebeple 4 ayaklı robotlar gibi belli yüksek serbestlik dereceli sistemlerde başka yöntemler daha etkili olabilmektedir [6].

Bu tezde yukarıda bahsedilen yöntemler arasında anlaşılması ve basit bir şekilde uygulanabilirliği açısından Denavit Hartenberg yöntemi kullanılmıştır. Her ne kadar Uzaysal Operatör Cebri yüksek serbestlik derecelerinde çok etkili olsa da Denavit Hartenberg sanayide ve insansı robot, ayaklı robot, robot el vb. bir çok araştırma konularında genel olarak kullanılan bir yöntemdir [8, 9, 10]. DH en geniş olarak kullanılan bir yöntemdir genel olarak robot üreticileri tarafından kullanılmaktadır [11,12]. Denavit Hartenberg minimum parametre ile kinematik problemleri çözüme kavuşturmaktadır ki bu diğer yöntemlere nazaran avantaj oluşturmaktadır [12].

Bu tezde dinamik kontrol çalışmasında ilk olarak Hesaplanmış Tork yöntemi kullanılmıştır. Eğer sistemin parametrik değerleri doğru bir şekilde biliniyorsa Hesaplanmış Tork kontrolü çok iyi performans verir [13, 14]. Bu modelin seri robotlarda yüksek dinamik işlemlerini desteklediği literatürde genişçe işlenmiştir [15]. Özellikle paralel robotlarda kinematik problemler karmaşık olmasına karşın dinamik model yüksek dereceden lineer olmayan terimler içermektedir ve bu sebeple Oransal Türevsel kontrolcü kullanmak işe yaramamaktadır. Bu durumda uygun kontrolcü tasarlamak zordur ve bunun için Hesaplanmış Tork yöntemi geliştirilmiştir. Hesaplanmış Tork yöntemi PD kontrol terimleri ile geri beslemeli dinamik model terimlerini kapsamaktadır ve lineer olmayan terimler barındırmaktadır [16, 17, 14]. Hesaplanmış Tork yönteminde sisteme konum hız ve ivme değerleri girilerek aynı birimlerde ve aynı sayıda çıkış alınmaktadır [18]. Hesaplanmış Tork Kontrolcüleri belirsizlik içeren lineer olmayan sistemlerde kullanılan en iyi doğrusal olmayan sağlam kontrolcülerdir [13, 17].

İleri dinamik ve geri dinamik denklemlerinin birleşimini içeren hesaplanmış tork yöntemi bu şekilde kapalı bir sistem oluşturmakta ve tamamıyla veya dinamik modelin kesinliğine bağlı olarak kısmi lineer yapı oluşturmaktadır ve bu şekilde robot manipulatorlere PD ve PID kontrolcüleri uygulanabilmektedir [17]. Bu durumda bizim için asıl problem dinamik modelin tam olarak çıkarılıp çıkarılamadığıdır bu problem ile alakalı bir çok çalışma yapılmıştır [17].

Dinamik modellemenin kusursuz bir şekilde yapıldığı öngörülürdüğü halde mekanik ve biyolojik sistemlerde kesin bilgiye sahip olmak mümkün olmamaktadır ve parametrik hatalar gibi bazı hatalar sebebiyle de kontrol sistemleri üzerinde bozuntular meydana gelebilmektedir. Eklem kütleleri, eklemlerin eylemsizlik momentleri, sürtünme katsayıları gibi fiziksel parametreleri kesin olarak bilmek mümkün olamamaktadır [17, 19]. Ayrıca engebeli yerlerde, değişken hava koşullarında bilinmeyen ve değişen sürtünme katsayıları sistemleri belirsizliğe götürmektedir. Bütün bu belirsizlikler bizleri uyarlamalı kontrol kullanımına yönlendirmektedir [19]. Uyarlamalı kontrol yöntemi gürbüz kontrol uylamasını da içeren güçlü özelliklere sahip bir kontrol yöntemidir. Uyarlama kontrol sistemini tasarlarken öncelikle uyarlama kontrol kuralını bulmak gerekir [20].

Uyarlamalı Kontrol yönteminde kontrol kuralını bulmak için yapılan bir çok yöntem olmakla birlikte bu çalışmada Lyapunov testi yapılmıştır [6, 21]. Bu test Uyarlamalı kontrol kısmında incelenmiştir. Bütün çalışmalar gerçek bir robot üzerinden denenmiş ve ABB şirketi ile iletişime geçilerek gerçek veriler kullanılmıştır [22, 23].

1.3 Sorunlar ve Çözümler

Gerek robotlarda gerekse uçaklarda dinamik sistemi kurmak bir şekilde mümkün olmaktadır ancak dış bozuntulara karşı farklı yöntemler geliştirmek gerekmektedir. Örneğin uçaklarda hava koşulları gibi dış etmenlere karşı gürbüz kontrol uygulaması, robotlarda iç dinamik parametrik hatalara karşı adaptif (uyarlamalı) kontrol uygulaması oldukça etkili olmaktadır.

Peki bu bozuntular sistemin içerisinden geliyorsa ne gibi çözümler oluşturulabilir? Bu oluşturulan çözümlerin uygulanışı ve karşılaştırılmaları nasıl olabilir? Bu sorulara cevap bulabilmek amacıyla bir sanayi robotuna adaptif (uyarlamalı) kontrol sistemi tasarımı yapılabilir ancak sonuçları hangi derecede etkili olacaktır ve farklı uygulamaları mevcut mudur? Adaptif (uyarlamalı) kontrol gibi bir kontrol metodu uygulamadan da sistemi belli bir hata sınırında tutmak mümkün olabilmekte midir?

Bu tezde bu sorulara cevaplar bulunmaya çalışılmış, sonuçlar kısmında bu sorular incelenmiş ve en uygun kontrol çalışması yapılmaya çalışılmıştır.

Bütün bu çalışmalar sonucunda adaptif (uyarlamalı) kontrolün sistemde oluşan parametrik hataları ortadan kaldırması ve bunun simulasyon şeklinde gösterilebilmesi beklenmektedir.

2. KİNEMATİK MODELLEME

2.1 Amaç

Bir robotun üzerinde kontrol sistemlerini uygulamadan önce matematik modellemesini yapmak ve bu model çerçevesinde kontrol algoritamlarına girmek gerekecektir. Bu çalışmada öncelikle kinematik modelleme ardından da Dinamik modelleme hakkında bilgi verilmiştir.

Eğer bir robot üzerinde modelleme yapılacaksa bunun için öncelikle kinematik hesabın düzgün bir şekilde yapılması gerekmektedir. Bu işlemlerle alakalı olarak birçok değişik modelleme yöntemi olmakla beraber genel olarak kullanılan bir sistemolan Denavit-Hartenberg yöntemi işlenecektir.

2.2 Denavit Hartenberg Yöntemi

Denavit ve Hartenberg'in 1955 yılında geliştirdiği Denavit-Hartenberg yöntemi robot kinematik hesaplarındaki karmaşıklığı büyük oranda çözmek ve daha basit modellenebilir hale getirmek için önerilmiştir [24].

Denavit Hartenberg yönteminde belirli kurallara göre önce koordinat çerçeveleri atanır, daha sonra dönüşüm için gerekli uzuv ve eklem parametreleri bulunur. Bu yöntemde her bir eksen bir önceki eksene bağlı olduğundan herhangi bir eklemin uç noktasının, referans noktasına oranla koordinatlarını bulmak mümkündür [24].

Eksen takımlarının atamaları 3 temel koşula bağlı olarak yapılır;

1- $z_i$ekseni, i+1 ekleminin hareket ekseni boyuncadır.
2- $x_i$ekseni, z_{i-1} eksenine diktir ve z_i eksenine doğru yönelmiştir.
3- $y_i$ekseni, sağ el yasasına uyacak şekilde saptanır [24].

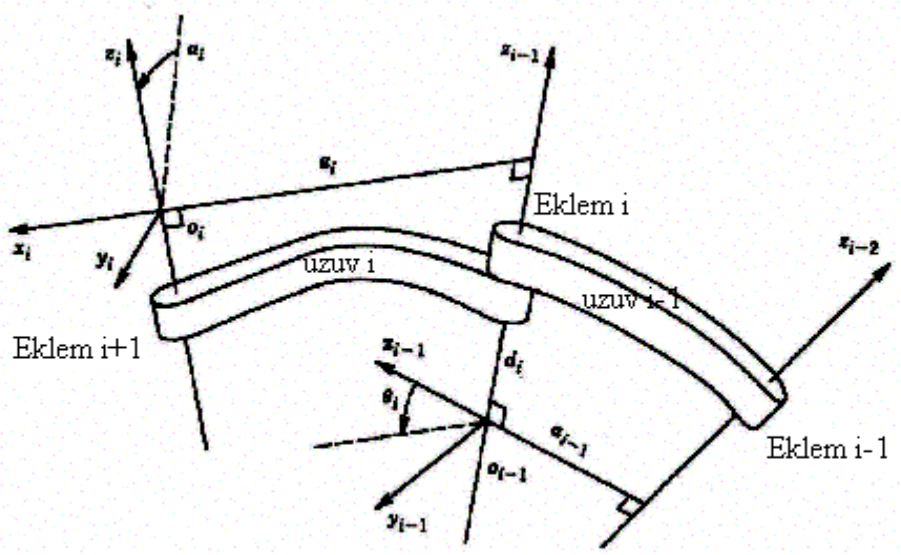
Daha sonra dönüşüm matrislerinin hesabında kullanılmak üzere gerekli parametrelerin ve değişkenlerin belirlenmesi gerekir.

a_{i-1}, Z_{i-1} ile Z_i arasında X_i boyunca belirlenen uzunluktur.

α_{i-1}, Z_{i-1} ile Z_i arasında X_i boyunca ölçülen açıdır.

d_i, X_{i-1} ile X_i arasında Z_{i-1} boyunca belirlenen uzunluktur.

θ_i, X_{i-1} ile X_i arasında Z_{i-1} boyunca ölçülen açıdır [25].



Şekil 2.1 : Eklem değişkenlerinin belirlenmesi [Url-1].

Herhangi bir robotun modellemesini yaparken sırasıyla önce x,y,z koordinatlarını belirleyip ardından a, α, d, θ değerlerini bulup bir tablo halinde yazmamız gerekir. Bu işlemlerin ardından ileri kinematik ve ters kinematik hesaplanabilir. Bir robotun kinematik modellemesi için bunlar zaruri bilgilerdir.

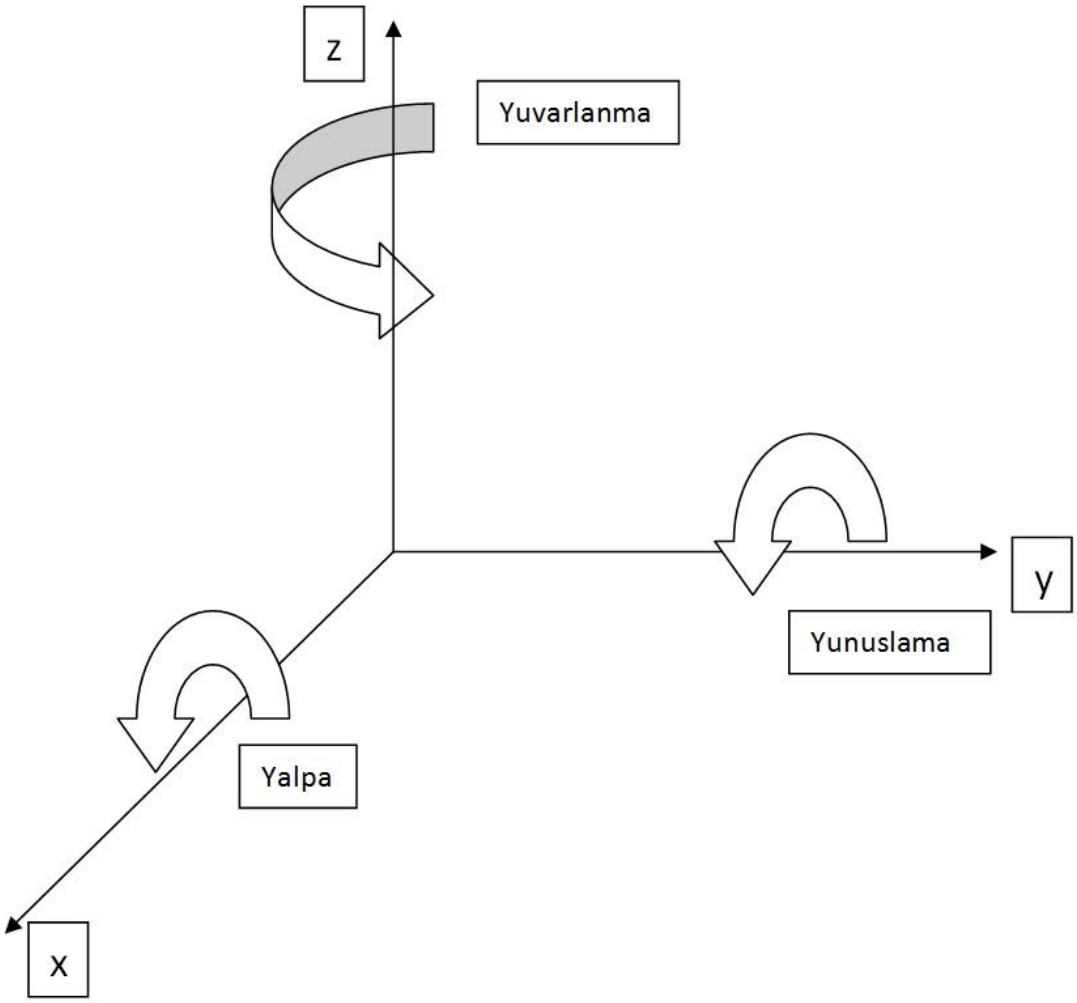
2.3 İleri Kinematik

Rigid bir eklemin dönme matrisi 3x3 olarak ifade etmek mümkündür zira sadece 3 eksende dönmek imkanı vardır. Dönme matrisi litaratürde R harfi ile gösterilir ve x ekseni, y ekseni z ekseni etrafında dönmesine göre farklılık gösterir.

Robotikte x ekseni etrafında dönme yalpa (yaw), y ekseni etrafında dönme yunuslama (pitch) ve z ekseni etrafında dönme yuvarlanma (roll) olarak ifade edilmektedir.

Uçaklarda bu dönme eksenleri farklılık göstermektedir ve aşağıdaki gibidir. x ekseni etrafında dönme uvarlanma (roll), y ekseni etrafında dönme yunuslama (pitch) ve z ekseni etrafında dönme yalpa (yaw) olarak ifade edilmektedir.

Uçak ve robot sistemleri temelde aynı mantık üzerine kuruludur. Sadece sisteme etkiyen kuvvetler farklılık göstermektedir.



Şekil 2.2 : Yalpa, yunuslama, yuvarlanma açıları.

Bütün eksenleri kendi içinde ayrı ayrı incelendiğinde aşağıdaki dönme formülleri ortaya çıkmaktadır.

$$R_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{bmatrix} R_{y,\theta} = \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix} R_{z,\theta} = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Bu matrisler dönme matrisleri olup x, y ve z eksenlerinde beraber bir dönüş olduğunda bu 3 matrisin çarpılması gerekecektir.

$$R = R_{z,\phi} R_{y,\theta} R_{x,\psi} = \begin{bmatrix} cos\phi & -sin\phi & 0 \\ sin\phi & cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\psi & -sin\psi \\ 0 & sin\psi & cos\psi \end{bmatrix}$$

$$= \begin{bmatrix} cos\phi cos\theta & -sin\phi cos\psi + cos\phi sin\theta sin\psi & sin\phi sin\psi + cos\phi sin\theta cos\psi \\ sin\phi cos\theta & cos\phi cos\psi + sin\phi sin\theta sin\psi & -cos\phi sin\psi + sin\phi sin\theta cos\psi \\ -sin\theta & cos\theta sin\psi & cos\theta cos\psi \end{bmatrix}$$

Ancak bu dönme (rotasyon) matrisleri sadece sabit bir noktada dönmeyi ifade etmek için kullanılır. Biz ayni zamanda ilerlemeyi de inceleyeceğimiz için transformation matrisleri ismiyle ifade edilen dönüşüm matrislerini kullanacağız. Bu matrislerin itibariyle dönme matrisleri ve yer değiştirme matrislerinden oluşmaktadır.

$$T = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} \quad \textbf{(2.1)}$$

Olayın daha iyi anlaşılması için her 3 eksende dönme ve ilerleme matrislerini verelim.

$$R_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\theta & -sin\theta & 0 \\ 0 & sin\theta & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_{x,a} = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_{y,\theta} = \begin{bmatrix} cos\theta & 0 & sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -sin\theta & 0 & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_{y,b} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_{z,\theta} = \begin{bmatrix} cos\theta & -sin\theta & 0 & 0 \\ sin\theta & cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_{z,c} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Dördüncü sütun x,y ve z eksenlerinde yer değiştirmeleri göstermektedir. Hem 3 eksendeki yer değiştirmeyi hemde 3 eksendeki dönmeyi ifade edebilmek için 2.2 nolu denklem kullanılmaktadır.

$$T = R_{z,\theta} T_{z,d} T_{x,a} R_{x,\alpha} \quad \textbf{(2.2)}$$

$$T = R_{z,\theta} T_{z,d} T_{x,a} R_{x,\alpha}$$

$$= \begin{bmatrix} cos\theta & -sin\theta & 0 & 0 \\ sin\theta & cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\alpha & -sin\alpha & 0 \\ 0 & sin\alpha & cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) * \cos(\alpha_i) & \sin(\theta_i) * sin(\alpha_i) & \mathrm{a_i} * \cos(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) * \cos(\alpha_i) & -\cos(\theta_i) * sin(\alpha_i) & \mathrm{a_i} * \sin(\theta_i) \\ 0 & \sin(\alpha_i) & \cos(\alpha_i) & b_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(2.3)}$$

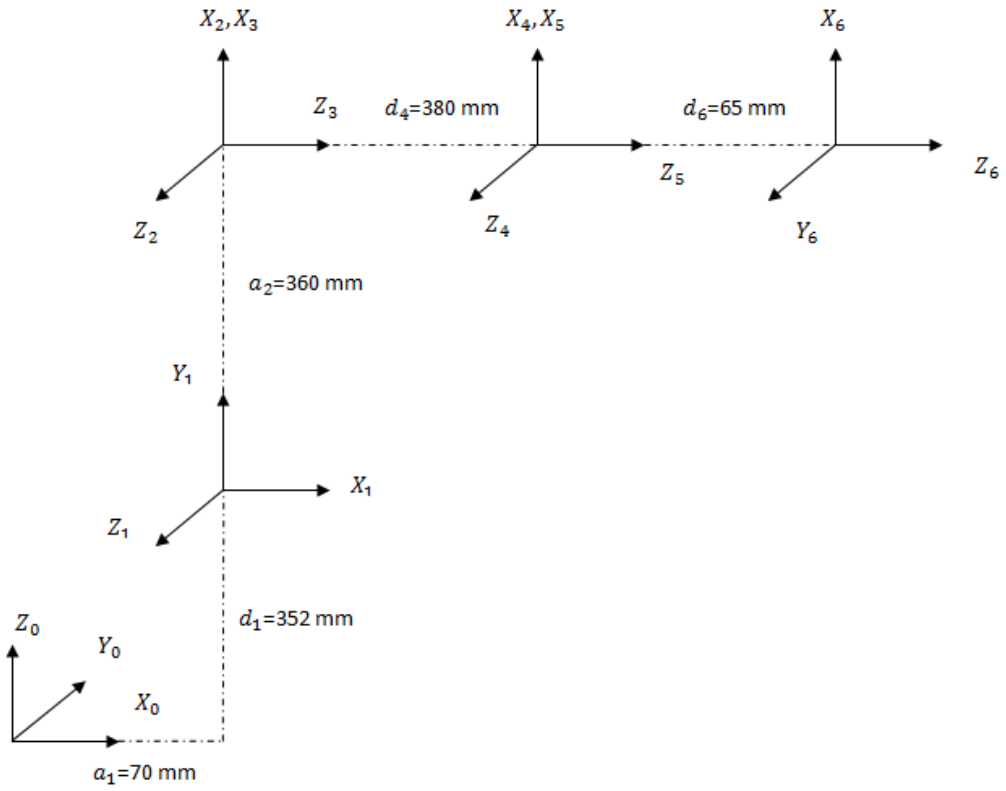
2.3 nolu denklemle herhangi bir robotun ileri kinematik denklemleri rahatça oluşturulabilir. Bu çalışmada örnek olarak ABB firmasına ait 6 eksenli IRB 140 robotu seçilmiştir. 810 mm lik mesafede 6 kg'lık yük kapasitesine sahip bir robottur.

Şekil 2.3 : ABB firmasının IRB 140 kodlu sanayi robotu [Url-2].

İleri kinematik denklemlerini oluşturmak için öncelikle Denavit Hartenberg yöntemine göre eksen atamalarının yapılması gerekmektedir. Bu işlem için öncelikle sıfır noktası ve belirlenir ve diğer eksenler daha önce belirtildiği gibi eklemlere bağlı olarak atanır.

Eksen atamaları kurallara uygun yapıldıktan sonra parametleri belirleyip tabloda yerine koyulur.

Bu değerlerin tabloda yerine konmasının ardından dönüşüm matrisleri hesaplanır. Bu hesap yapılırken 2.3 nolu denklem kullanılır.



Şekil 2.4 : 6 eklemli robotumuzun eksen atamaları.

Çizelge 2.1 : 6 eklem için D-H tablosu.

	a (xi boyunca)	α (xi boyunca)	d (zi-1 boyunca)	θ (zi-1 boyunca)	Hazırol durumu
1	70	π/2	352	θ_1	0
2	360	0	0	θ_2	π/2
3	0	π/2	0	θ_3	0
4	0	- π/2	380	θ_4	0
5	0	π/2	0	θ_5	0
6	0	0	65	θ_6	0

$$T_{0,1} = \begin{bmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & 70 * \cos(\theta_1) \\ \sin(\theta_1) & 0 & -\cos(\theta_1) & 70 * \sin(\theta_1) \\ 0 & 1 & 0 & 352 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{1,2} = \begin{bmatrix} \cos(\theta_2 + pi/2) & -\sin(\theta_2 + pi/2) & 0 & 360 * \cos(\theta_2 + pi/2) \\ \sin(\theta_2 + pi/2) & \cos(\theta_2 + pi/2) & 0 & 360 * \sin(\theta_2 + pi/2) \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{2,3} = \begin{bmatrix} \cos(\theta_3) & 0 & \sin(\theta_3) & 0 \\ \sin(\theta_3) & 0 & -\cos(\theta_3) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} T_{3,4} = \begin{bmatrix} \cos(\theta_4) & 0 & -\sin(\theta_4) & 0 \\ \sin(\theta_4) & 0 & \cos(\theta_4) & 0 \\ 0 & -1 & 0 & 380 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{4,5} = \begin{bmatrix} \cos(\theta_5) & 0 & \sin(\theta_5) & 0 \\ \sin(\theta_5) & 0 & -\cos(\theta_5) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} T_{5,6} = \begin{bmatrix} \cos(\theta_6) & -\sin(\theta_6) & 0 & 0 \\ \sin(\theta_6) & \cos(\theta_6) & 0 & 0 \\ 0 & 0 & 1 & 65 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Bu dönüşüm matrisleri sırasıyla eklemler arasındaki dönüşümleri vermektedir. Uç noktasının merkez eksene göre konumunu bulmak için bu 6 matrisi birbiriyle çarpmamız gerekecektir. Bu işlemi elle yapmak zor olacağından Ek A.1'de gösterilen Matlab programı kullanılmıştır:

$$T_{0,6} = T_{0,1}T_{1,2}T_{2,3}T_{3,4}T_{4,5}T_{5,6} \quad \textbf{(2.4)}$$

Elde edilen sonuçlar Ek A.2'de gösterildiği gibi 4x4'lük ve uzun denklemler içeren bir matris olarak çıkmıştır.

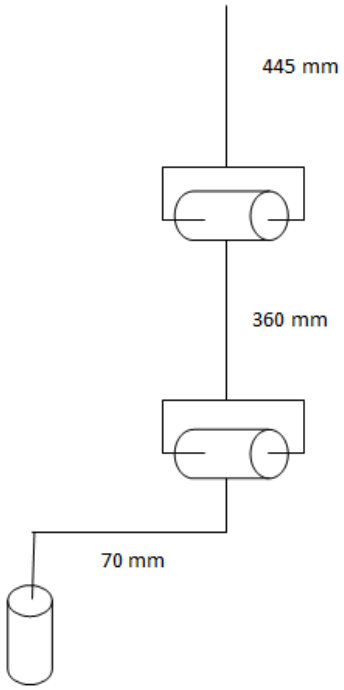
Bu sonuç bize herhangi bir eklemi ne kadar döndürdüğümüzde uç organın nerede ve hangi konumda olduğunu verir. Ancak sanayi de ve robot kullanımlarında önemli olan uç organın istenilen konuma gitmesini sağlamak ve bunun için de ters kinematik hesap yaparak her bir eklemin ne konumda ve hızda olduğunun bilgisine ulaşılabilmesini sağlamaktır. Bu sebeple bir sonraki konumuz ters kinematik olacaktır.

2.4 Ters Kinematik

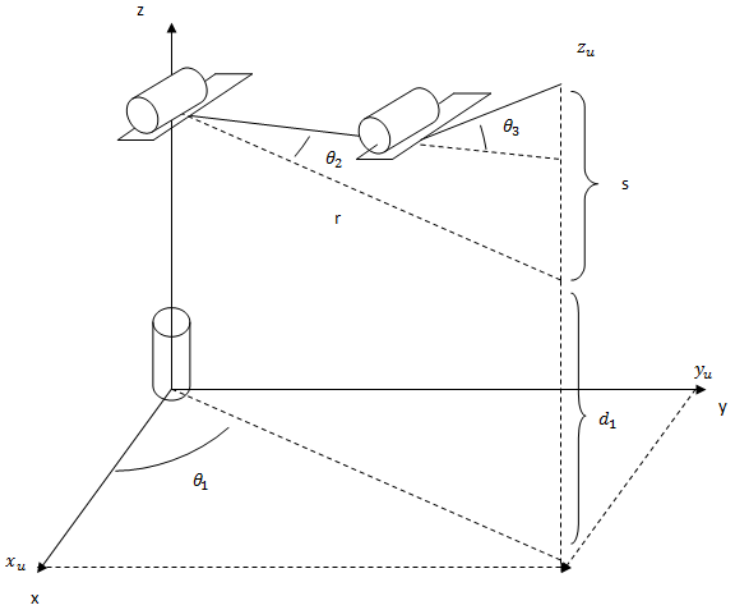
Ters kinematik problemi için farklı yaklaşımlar bulunmaktadır. Eklem sayısı arttıkça ters kinematik modellemenin zorluğu arttığından birçok teknik kullanılmakta ve sıkıntılar aşılmaya çalışılmaktadır. Bu bölümde geometrik ve analitik yaklaşımlardan kısaca bahsedilecek ve genel bir bilgi verildikten sonra örnek olarak robotumuzla alakalı bilgi verilecektir.

2.4.1 Geometrik yaklaşım

Bu yaklaşım türünde sistemin geometrik yapısı incelenir ve ve bir formül üretilir. İşlemin basit olması adına ilk 3 eksen için bu işlemi yapalım.



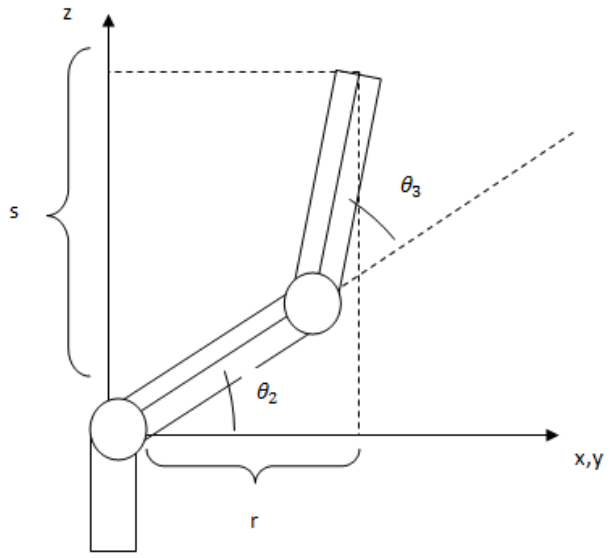
Şekil 2.5 : İlk üç eksenin izometrik görünümü



Şekil 2.6 : 3 eksenli yapının incelenmesi.

Bizim robotumuzun bütün eksenleri dönel eksendir ve ilk 3 eksen şekil 2.5'teki gibi bir yapıya sahiptir. Birinci eksen robotun hangi tarafa ve kaç derece yöneleceğini

belirtmektedir ve bu sebeple sadece x ve y eksenlerine bakarak derecesini öğrenmek mümkün olabilecektir.



Şekil 2.7 : 2. ve 3. eklem açılarının bulunması.

$$\theta_1 = atan2(x_u; y_u) \quad \textbf{(2.5)}$$

Şekil 2.6'da ilk 3 eksen için genel bir görüntü verilmiş ve ilk eksen için 2.5 nolu denklem verilmiştir. 2 ve 3. eksenin değişken değerleri birbirleriyle bağlantılı olduğu için işlemin daha kolay olması adına öncelikle 3. Eksen için bir denklem oluşturulur. Şekil 2.7'de olduğu gibi robota yandan baktığımızda 2. ve 3. Eklemin uzunluklarını bildiğimizden 3. Eksenin değeri için cosinus teoreminden faydalanabiliriz.

$$r^2 + s^2 = a_2^2 + a_3^2 + 2a_2a_3\cos\theta_3 \;\Rightarrow$$

$$cos\theta_3 = \frac{r^2+s^2-a_2^2-a_3^2}{2a_2a_3} = \frac{x_u^2+y_u^2-d^2+(z_u-d_1)^2-a_2^2-a_3^2}{2a_2a_3} = D$$

Bu eklemin hem yukarı hemde aşağı yönde hareketi olabilir. Bu durumda sinüs ve cosinüsün oranlarından meydana gelen tanjant ile gerekli denklemi aşağıdaki gibi üretilebilir.

$$\theta_3 = Atan2(D, \sqrt{1-D^2}) \quad \textbf{(2.6)}$$

2. eklem için de aşağıdaki formül çıkartılabilir.

$$\theta_2 = Atan2\left(\sqrt{x_u^2 + y_u^2 - d^2}, z_u - d_1\right) - Atan2(a_2 + a_3c_3, a_3s_3) \quad \textbf{(2.7)}$$

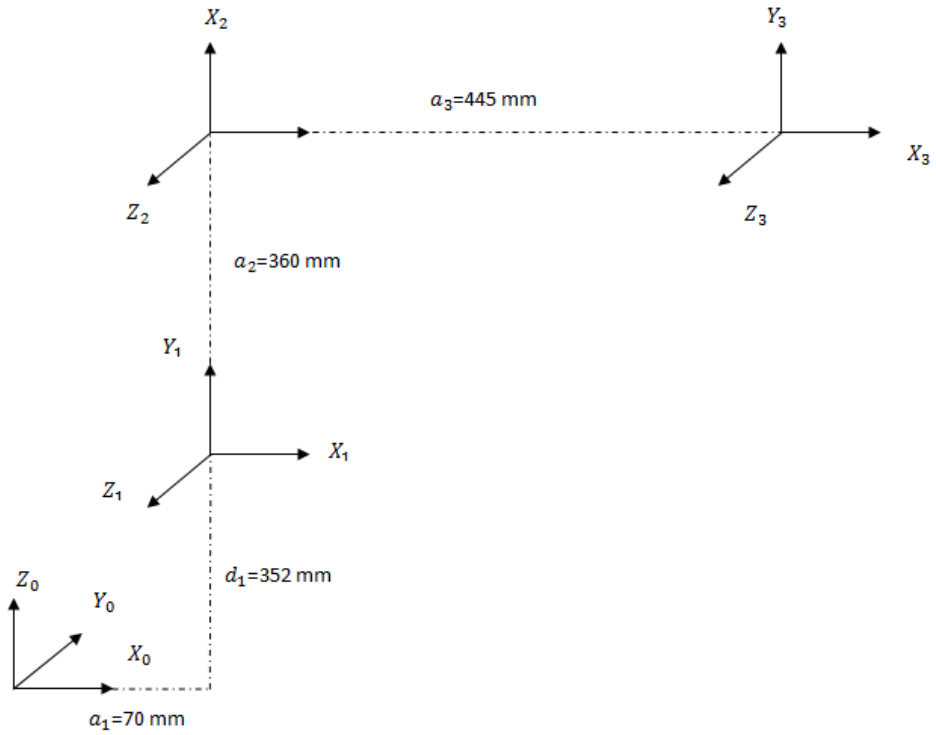
Bu konuda detaylı bilgi için Spong'un Robot Modelling and Control kitabına bakılabilir [26].

2.4.2 Analitik yaklaşım

Bu yaklaşımda öncekinin aynı sisteme sahiptir ancak bu sefer işlemleri dönüşüm matrisleri üzerinde yapacağız. Öncelikle bizim ileri kinematik denklemlerimiz 9 nolu denklemde belirtildiği şekilde bulunmaktadır. Ayrıca 13 nolu denklemde belirtildiği gibi bir matris tanımlarız.

$$T_{0,6} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(2.8)}$$

Bu matrisi dönüşüm matrisi ile eşleştirip ters kinematik formülleri çıkartabiliriz. Basit olması açısından bunu yine ilk 3 eklem ile yapalım. Bunun için öncelikle robotumuzun D-H tablosunu hazırlamak gerekir. Daha sonra hesaplanacak olan ileri kinematik matrislerini 14 ve 15 nolu denklemlerde olduğu gibi birbirlerine eşitlemek suretiyle $\theta_1, \theta_2\ ve\ \theta_3$ değerlerine ulaşmak mümkün olacaktır.



Şekil 2.8 : İlk 3 eklem için DH tablosu.

Çizelge 2.2 : 3 eklem için DH tablosu.

	a (xi boyunca)	α (xi boyunca)	d (zi-1 boyunca)	θ (zi-1 boyunca)	Hazırol durumu
1	70	π/2	352	θ_1	0
2	360	0	0	θ_2	π/2
3	445	0	0	θ_3	-π/2

$T_{0,3} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$ olsun aynı zamanda $T_{0,3} = T_{0,1} * T_{1,2} * T_{2,3}$

olduğundan aşağıdaki işlemler yapılabilir:

$$T_{0,3}=\begin{bmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & 70 * \cos(\theta_1) \\ \sin(\theta_1) & 0 & -\cos(\theta_1) & 70 * \sin(\theta_1) \\ 0 & 1 & 0 & 352 \\ 0 & 0 & 0 & 1 \end{bmatrix} *$$

$$\begin{bmatrix} \cos(\theta_2 + pi/2) & -\sin(\theta_2 + pi/2) & 0 & 360 * \cos(\theta_2 + pi/2) \\ \sin(\theta_2 + pi/2) & \cos(\theta_2 + pi/2) & 0 & 360 * \sin(\theta_2 + pi/2) \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} *$$

$$\begin{bmatrix} \cos(\theta_3 - pi/2) & 0 & \sin(\theta_3 - pi/2) & 445 * \cos(\theta_3 - pi/2) \\ \sin(\theta_3 - pi/2) & 0 & -\cos(\theta_3 - pi/2) & 445 * \sin(\theta_3 - pi/2) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

=

$$\begin{bmatrix} \cos(\theta_2 + \theta_3)\cos(\theta_1) & \sin(\theta_1) & \sin(\theta_2 + \theta_3)\cos(\theta_1) & \cos(\theta_1)\,[445\cos(\theta_2 + \theta_3) - 360\sin(\theta_2) + 70] \\ \cos(\theta_2 + \theta_3)\sin(\theta_1) & -\cos(\theta_1) & \sin(\theta_2 + \theta_3)\sin(\theta_1) & \sin(\theta_1)\,[445\cos(\theta_2 + \theta_3) - 360\sin(\theta_2) + 70] \\ \sin(\theta_2 + \theta_3) & 1 & -\cos(\theta_2 + \theta_3) & 445\sin(\theta_2 + \theta_3) + 360\cos(\theta_2) + 352 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Bu işlemden sonra aşağıda görüldüğü gibi eşitliklerden yola çıkılarak 2.9 nolu denklemler türetilebilir.

$$p_x = \cos(\theta_1)\,[445\cos(\theta_2 + \theta_3) - 360\sin(\theta_2) + 70]$$

$$p_y = \sin(\theta_1)\,[445\cos(\theta_2 + \theta_3) - 360\sin(\theta_2) + 70]$$

$$p_z = 445\sin(\theta_2 + \theta_3) + 360\cos(\theta_2) + 352$$

$$\left[T_{0,1}\right]^{-1} * T_{0,3} = T_{1,2} * T_{2,3} \quad \textbf{(2.9)}$$

$$\left[T_{0,1} * T_{1,2}\right]^{-1} * T_{0,3} = T_{2,3}$$

Bu denklemlerin karşılıklı olarak sadeleştirilmesi sonucu sistemin ters kinematik denklemlerine ulaşmak mümkündür.

2.5 Jakobiyen Matrisi

Daha önce göstermiş olduğumuz ileri ve geri kinematik denklemleri eklemlerin birbirlerine göre konumlarını ifade etmektedir, jacobean matrisi ise eklemler arasındaki hız ilişkilerini ifade eder. Jakobiyen matrisi bu yönüyle robotik işlemlerde çok kullanılan bir araçtır.

Jakobiyen matrisine girdi olarak robotumuzun uç organının anlık hız değerlerini verir çıktı olarak ise her bir eklemin hız değerlerini alırız. Klasik olarak ters kinematik işlemlerde konumlar üzerinden hareket edilirken jakobiyen matrisi bu konum ilişkilerini hız olarak tanımlamamızı sağlar.

Jacobiyen matrisi doğrusal ve açısal hız olmak üzere iki hızı tanımlar. Linear hız için 3 ve açısal hız için 3 olmak üzere her bir eklem için toplam 6 değişkeni hesaplar.

$$J = \begin{bmatrix} J_v \\ J_w \end{bmatrix} = \begin{bmatrix} v_{3x1} \\ w_{3x1} \end{bmatrix} \quad \textbf{(2.10)}$$

Jakobiyen matrisi hesabında her bir eklemin açısal ve doğrusal hızı bir önceki ekleme göre hesaplanır. Bu sebeple eklemin prizmatik veya dönel olması açısal ve doğrusal hızları değiştirecektir. Lineer ve açısal hızlar için 3'er değişkenin olması x, y ve z yönündeki lineer ve açısal hızlardan kaynaklanmaktadır. Jacobean matrisi bulunurken 2.10 nolu denklem kullanılır.

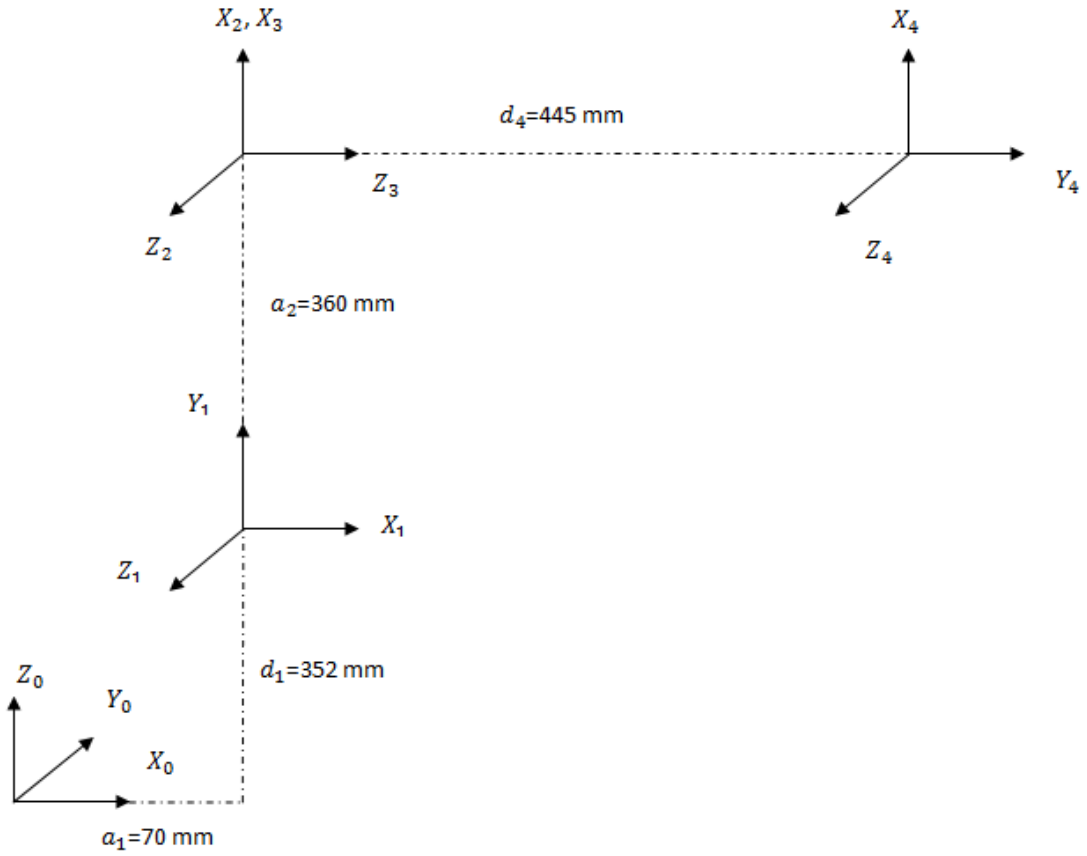
$$J_{v_i} = \begin{Bmatrix} z_{i-1} x (O_n - O_{i-1}) & dönel\ eklem\ ise \\ z_{i-1} & prizmatik\ eklem\ ise \end{Bmatrix}$$

$$J_{w_i} = \begin{Bmatrix} z_{i-1} & dönel\ eklem\ ise \\ 0 & prizmatik\ eklem\ ise \end{Bmatrix} \quad \textbf{(2.11)}$$

Bu denklemlerde z_i eklemin z eksenindeki dönmeyi gösterirken O_n ise yer değiştirmeyi göstermektedir. Z eksenindeki dönmeyi dönüşüm matrisinin 3. sütunu

alınarak bulunabilir. Aynı şekilde dönüşüm matrisinin 4. sütunu kullanılarak yer değiştirme de rahat bir şekilde bulunabilir.

IRB 140 isimli robotumuzun Jacobean matrisini bulalım: Bunun için işlemlerin daha kolay yapılabilmesi amacıyla ilk 3 eksen üzerinde çalışılmıştır. Bu sebeple eksen atamaları aşağıdaki tabloda görüldüğü gibi değiştirilmiş ve 4 eksen ile ifade edilmiştir ancak işlemler 3 değişken üzerinden yapılmıştır.



Şekil 2.9 : 4'lü eksen ataması.

Çizelge 2.3 : 4 eklem için D-H tablosu.

	a (xi boyunca)	α (xi boyunca)	d (zi-1 boyunca)	θ (zi-1 boyunca)	Hazırol durumu
1	70	π/2	352	θ_1	0
2	360	0	0	θ_2	π/2
3	0	π/2	0	θ_3	0
4	0	-π/2	445	θ_4	0

Jacobean matrisi hesabı yapılırken öncelikle ileri kinematik matrislerini hesaplarız.

$$T_{0,1} = \begin{bmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & 70 * \cos(\theta_1) \\ \sin(\theta_1) & 0 & -\cos(\theta_1) & 70 * \sin(\theta_1) \\ 0 & 1 & 0 & 352 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{1,2} = \begin{bmatrix} \cos(\theta_2 + pi/2) & -\sin(\theta_2 + pi/2) & 0 & 360 * \cos(\theta_2 + pi/2) \\ \sin(\theta_2 + pi/2) & \cos(\theta_2 + pi/2) & 0 & 360 * \sin(\theta_2 + pi/2) \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{2,3} = \begin{bmatrix} \cos(\theta_3) & 0 & \sin(\theta_3) & 0 \\ \sin(\theta_3) & 0 & -\cos(\theta_3) & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} T_{3,4} = \begin{bmatrix} \cos(\theta_4) & 0 & -\sin(\theta_4) & 0 \\ \sin(\theta_4) & 0 & \cos(\theta_4) & 0 \\ 0 & -1 & 0 & 445 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{0,4} = \begin{bmatrix} s_1 * s_4 - c_1 * c_2 * c_4 * s_3 - c_1 * c_3 * c_4 * s_2 & -c_{(2+3)} * c_1 & c_4 * s_1 + c_1 * c_2 * s_3 * s_4 + c_1 * c_3 * s_2 * s_4 & 5 * c_1 * (89 * c_{(2+3)} - 72 * s_2 + 14) \\ -c_1 * s_4 - c_2 * c_4 * s_1 * s_3 - c_3 * c_4 * s_1 * s_2 & -c_{(2+3)} * s_1 & c_2 * s_1 * s_3 * s_4 - c_1 * c_4 + c_3 * s_1 * s_2 * s_4 & 5 * s_1 * (89 * c_{(2+3)} - 72 * s_2 + 14) \\ c_{(2+3)} * c_4 & -s_{(2+3)} & -c_{(2+3)} * s_4 & 445 * s_{(2+3)} + 360 * c_2 + 352 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$T_{0,4}$matrisinde kısaltma amacıyla sinüs yerine s cosinüs yerine c harfi kullanılmış ve θ_3 ifadeleri rakamlarla ifade edilmiştir. Daha okunaklı olması amacıyla sonuç tekrardan verilmiştir.

Birinci satır : $[\, s_1 * s_4 - c_1 * c_2 * c_4 * s_3 - c_1 * c_3 * c_4 * s_2 \,,\, -c_{(2+3)} * c_1,\, c_4 * s_1 + c_1 * c_2 * s_3 * s_4 + c_1 * c_3 * s_2 * s_4,\, 5 * c_1 * (89 * c_{(2+3)} - 72 * s_2 + 14)]$

İkinci satır :$[-c_1 * s_4 - c_2 * c_4 * s_1 * s_3 - c_3 * c_4 * s_1 * s_2, -c_{(2+3)} * s_1,\, c_2 * s_1 * s_3 * s_4 - c_1 * c_4 + c_3 * s_1 * s_2 * s_4,\, 5 * s_1 * (89 * c_{(2+3)} - 72 * s_2 + 14)]$

Üçüncü satır : $[c_{(2+3)} * c_4,\, -s_{(2+3)},\, -c_{(2+3)} * s_4,\, 445 * s_{(2+3)} + 360 * c_2 + 352]$

Dördüncü satır : [0, 0, 0, 1]

IRB 140 robotunun eklemlerinin hepsi dönel eklem olduğundan 2.11 nolu denklemin dönel eklemler için olan kısımlarını kullanacağız.

$$z_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} z_1 = \begin{bmatrix} \sin(\theta_1) \\ -\cos(\theta_1) \\ 0 \end{bmatrix} z_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} z_3 = \begin{bmatrix} \sin(\theta_3) \\ -\cos(\theta_3) \\ 0 \end{bmatrix} z_4 = \begin{bmatrix} -\sin(\theta_4) \\ \cos(\theta_4) \\ 0 \end{bmatrix}$$

$$Q_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} Q_1 = \begin{bmatrix} 70 * \cos(\theta_1) \\ 70 * \sin(\theta_1) \\ 0 \end{bmatrix} Q_2 = \begin{bmatrix} 360 * \cos(\theta_2 + pi/2) \\ 360 * \sin(\theta_2 + pi/2) \\ 0 \end{bmatrix} Q_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} Q_4 = \begin{bmatrix} 0 \\ 0 \\ 445 \end{bmatrix}$$

IRB 140 robotu için uyguladığımız genel formül aşağıda verilmiştir.

$$J = \begin{bmatrix} cross(z_0, (Q_4 - Q_0)) & cross(z_1, (Q_4 - Q_1)) & cross(z_2, (Q_4 - Q_2)) \\ z_0 & z_1 & z_2 \end{bmatrix}$$

$$J = \begin{bmatrix} 0 & -93\cos(\theta_1) & 360\cos(\theta_2) \\ 0 & -93\sin(\theta_1) & 360\sin(\theta_2) \\ 0 & -70 & 0 \\ 0 & \sin(\theta_1) & 0 \\ 0 & -\cos(\theta_1) & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

Jakobiyen eklemlerin hızlarını ifade etmek için kullanılan bir matris olup bu matrisi uygun şekilde kullanabilmek zorundayız aksi halde tek başına bir şey ifade etmiyecektir. Bu sebeple robotumuz için uygun bir kontrol sistemi tasarlayıp Jakobiyen matrisini de içine koymak durumundayız.

2.6 Matlab'ta Yörünge Planlaması

Matlab üzerinde modelleme yaparken eklem uzayında ve kartezyen uzayında olmak üzere genel olarak 2 yöntem kullanılmaktadır. Bu yazıda daha çok kartezyen yörünge planlaması üzerinde durulacak ve hız kontrolü kullanılarak incelenecektir.

2.6.1 Kartezyen-eklem uzayında yörünge planlaması

Yörünge planlaması yapılırken önemli olan nokta şudur ki biz bu robotun her anda bir nokta verip orada olmasını mı istiyoruz yoksa başlangıç ve bitiş noktaları verip aradaki yolu nasıl gittiğini önemsemiyor muyuz?

Bu durumu daha iyi ifade edebilmek için yörünge planlamasında kullanılan eklem uzayı ve kartezyen uzayı hareketlerini inceleyelim.

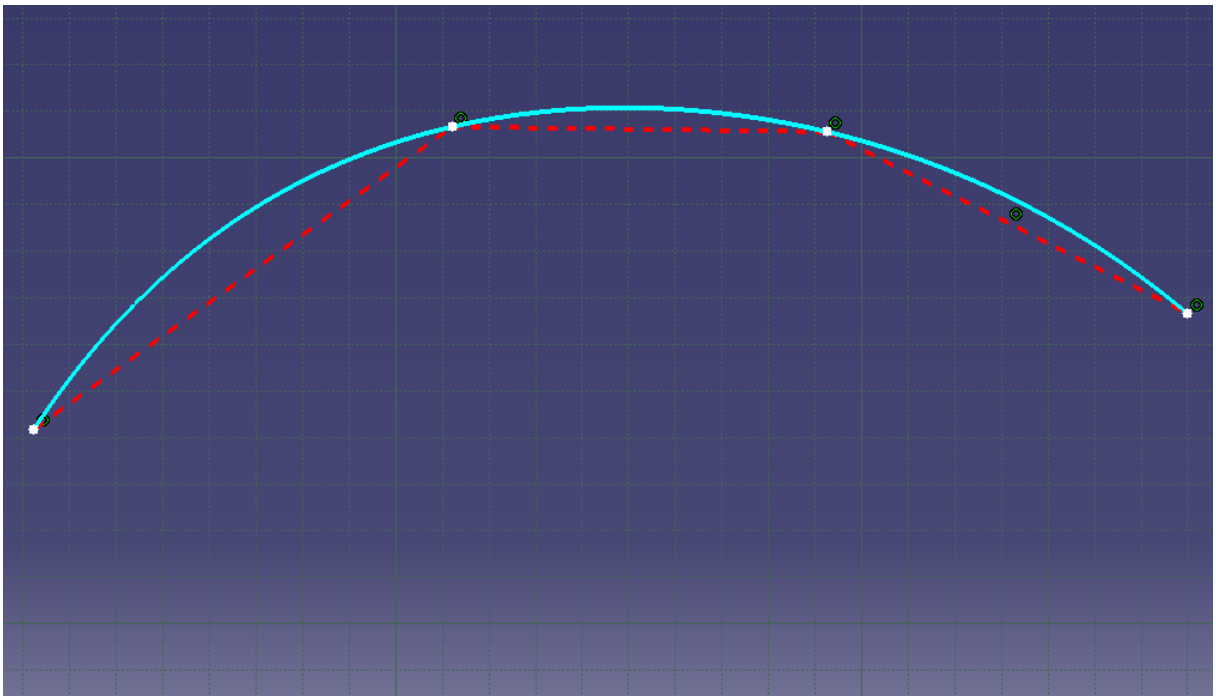
Eklem uzayında; sistemin bizden istediği ilk ve son konum, ayrıca bu 2 konum arasındaki mesafeyi robotun ne kadar sürede gideceğinin bilinmesidir. Bu planlamanın yapılmasının ardından robot uç organı bu 2 nokta arasında kendine uygun bir yörünge çizip ilerleyecek ve bize anlık olarak konum, hız ve ivme değerlerini verecektir. Kullanıcının yörüngeye müdahale şansı olmadığı gibi robotun uç organının hızına da bir etkisi bulunmamaktadır [27].

Kartezyen uzayında; durum biraz daha farklıdır zira 3 aşamalı bir yapı mevcuttur. Bu modelde robot için maksimum ivme değerleri göz önünde bulundurulur ve robot bu ivmesel değerleri aşamaz. Eğer tasarımda ufak bir değişiklik yapılırsa robota sabit bir hız değeri girilip robotun bu değerde sürekli olarak hareket etmesi sağlanabilir. Bu

sayede robot kamsimum ivme ile bu hıza gelecek ve bu sabit hızda sürekli olarak devam edecektir.

Kartezyen uzayı eklem uzayına göre bir çok yönden avantaj sağlamasına karşın yine de bazı sıkıntıları mevcuttur. Örnek olarak verilen 2 nokta arasında düz bir çizgi halinde gitmekte ve sizin isteyeceğiniz karmaşık bir yörüngeyi takip etmekten uzak olacaktır. Ancak sanayide kullanımı oldukça uygundur.

Kartezyen ve eklem uzayında yörünge planlamanın farkının daha iyi anlaşılabilmesi için Şekil 2.10'a bakılabilir. Şekilden görüldüğü gibi kartezyen uzayında robot direkt hedefe yönelmeyi seçerken eklem uzayında uygun bir rota çizmektedir [27].



Şekil 2.10 : Kartezyen (kırmızı kesikli çizgi) ve eklem (mavi düz çizgi) uzayı yörüngeleri.

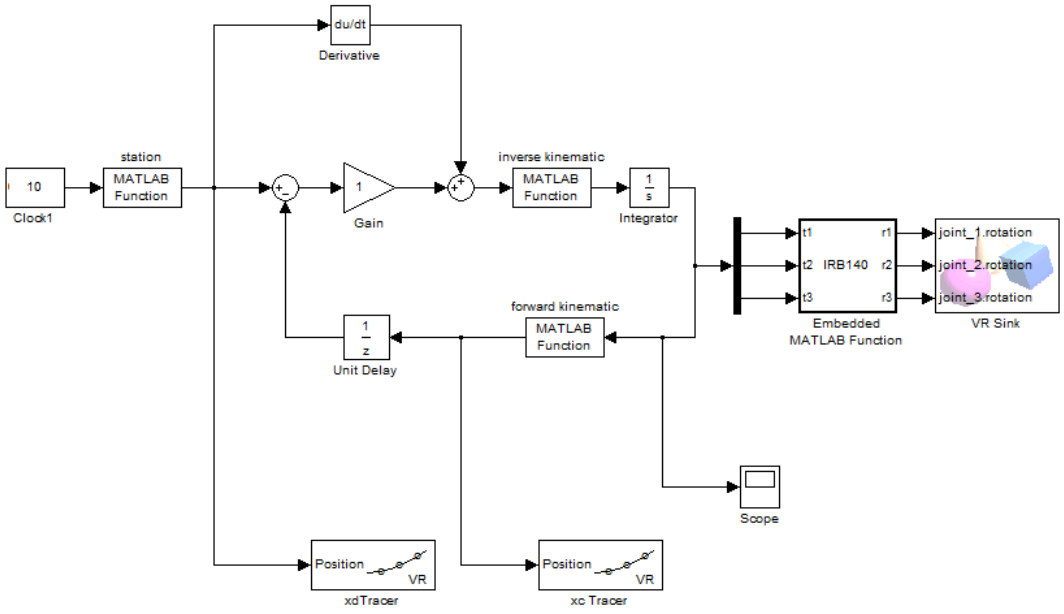
2.6.2 Hız kontrollü yörünge planlaması

Jakobiyen kullanılarak yapılan yörünge planlamasında ters kinematik denklemlerinden gelen bazı tekillik problemleri ortadan kalkmakta ve sisteme istenilen karmaşıklıkta yörünge verilebilmektedir. Bu sistem modelleme aşağıdaki ifade edildiği gibi yapılmaktadır.

Matematiksel olarak ifade edilen sistemin modellemesini yapmak için simulink diyagramlarını kullanamak uygun görüldü ve Şekil 2.11'de ki gibi bir model

hazırlandı. Bu model önceki bölümlerde bahsedildiği gibi yalnızca kinematik modellemeyi içermektedir.

Bu modelde en solda görülen clock bloğu zamanı göstermektedir ve sistem ilk olarak o bloğun çalışmasıyla başlar. Bunun ardından station diye ifade edilen bloğa gelir. Bu bloğun için robotumuzun uç noktasının izleyeceği rota bulunmaktadır. Clock bloğundan gelen zamana göre uç noktasının hangi zamanda nerede olması gerektiği çok açık olarak ifade edilmiştir.



Şekil 2.11 : IRB 140 yörünge modeli.

Station bloğu kodları:

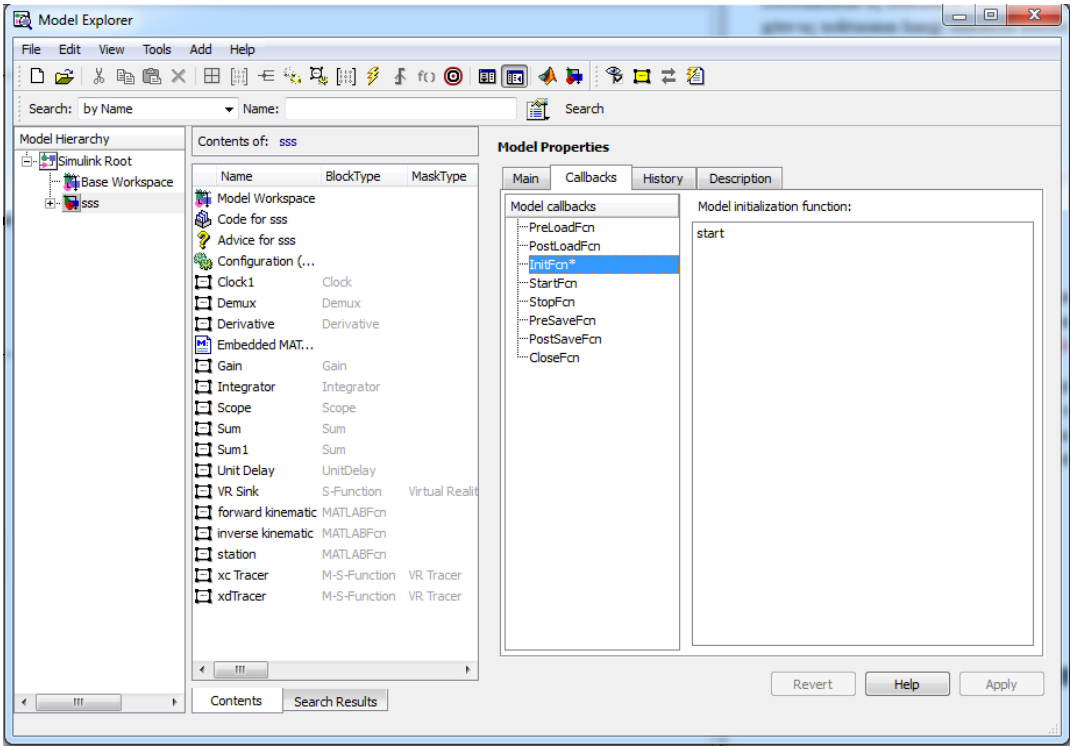
```
function xd = station(t)
global xd;

w=0.6;
xd=[0.512+0.2*cos(2*w*t)*cos(w*t) 0.2*cos(2*w*t)*sin(w*t) 0.712]';
%kelebek

end
```

Bu kodta belirtilen yörünge aynı zamanda 3 boyutlu olarak gösterilebilmesi amacıyla xd tracer diye adlandırılan bir bloğa bağlanarak 3 boyutlu olarakta gösterilmiştir.

Bu sistemde aldığımız yörüngeyi gain değeri ile çarpıp hız bilgisi şeklinde inverse kinematic diye belirtilen bloğa bağlayıp ters kinematic yardımıyla her bir eklemin ne

kadar hareket etmesi gerektiği hesabında kullanmak istedik. Bunun için Jakobiyen matrisini kullandık ve hız tanımları üzerinden hareket edildi.

İlk başlangıç olmak üzere bir Jakobiyen matrisi tanımlanıp sistemin içine yerleştirildi. Bu matris her döngüde değişecek olmasına karşın sistemin çalışmaya başlaması için gereklidir. Bu sebeple start isminde bir function dosyası yazıp bunu simulink dosyamızın içindeki view ikonundan model explorer pencerisini açarak şekil 2.12'da ekranda görülen İnitialFunction (İlk fonksiyon) kısmına ekleyip kaydediyoruz. Bu şekilde ilk önce start dosyasını okuyup çalıştıracaktır. Start dosyası Ek A.3'te gösterilmiştir.



Şekil 2.12 : IRB 140 yörünge modeli başlama dosyasını sisteme ekleme.

Start dosyasının en son kısmında t_1, t_2, t_3 şeklinde belirtilen eklem değişkenlerinin 0 değerleri ile değiştirildiği görülmektedir. Buradaki amaç sistemin ilk başlama anında bu değişkenlerin 0 olarak kabul etmektir. Bu sayede bu sistem ne zaman çalışmaya başlarsa başlasın her zaman için robot sıfır durumundan harekete başlayacaktır.

İnverse kinematic bloğuna giren x, y, z hız değerleri inverse kinematic bloğuna girdikten sonra her bir eklemin hız değeri olarak çıkacaktır.

İnverse Kinematic Bloğu kodları:

```
function[thetadot]=inverse(Vt)

global Jvi;
```

```
thetadot=inv(Jvi)*Vt;

end
```

Yukarıda verilen kodlarda da görüldüğü gibi Jakobiyen fonksiyonunun tersi ile girdi değerleri çarpılarak her bir eksenin hızları hesap edilmiştir. Bu hesap konusunda biraz açıklama yapmakta fayda vardır:

Jakobiyen matrisi ile robotun hızı arasında aşağıda belirtildiği gibi bir ilişki mevcuttur.

$$V=J*\dot{\theta} \tag{2.12}$$

$\dot{\theta}$ her bir eklemin hızını gösterirken V ise uç noktanın hızlarını göstermektedir ve beklenenin aksine ters hız kinematiği ters konum kinematiğinden daha kolaydır. Eğer hesaplamış olduğumuz matris kare matris ise ve tekillik özelliği gösteriyorsa bu matrisi 2.13'te ki denkleme çevirmek mümkündür [26]. Ancak Jakobiyen matrisinin tersini almak kare matrisi olmadığı için mümkün olmamaktadır. Bu sebeple Jakobiyen matrisinin sözde tersini almak gerekir. Bunun için matlabta "pinv" komutunu kullanmak yeterli olacaktır.

$$\dot{\theta} = J^{-1} * V \tag{2.13}$$

İnverse kinematic bloğundan çıkan hız verileri integrator yardımıyla konum verilerine dönüşür ve hedefe gönderilir. Bu veriler aynı zamanda döngü olarak şekilde görülen forward kinematic bloğuna gönderilerek tekrardan ileri kinematik ile uç noktanın koordinatlarının belirlenmesi sağlanır.

Forward Kinematic Bloğu kodları:

```
function[xc]=forward(th)

global Jv Jvi H4 t1 t2 t3 t4 xc;

t11=th(1);t22=th(2);t33=th(3);

xc1=subs(H4,[t1 t2 t3 t4],[t11 t22 t33 0]);
xc=xc1(1:3,4);
Jvi=subs(Jv,[t1 t2 t3 t4 ],[t11 t22 t33 0]);

end
```

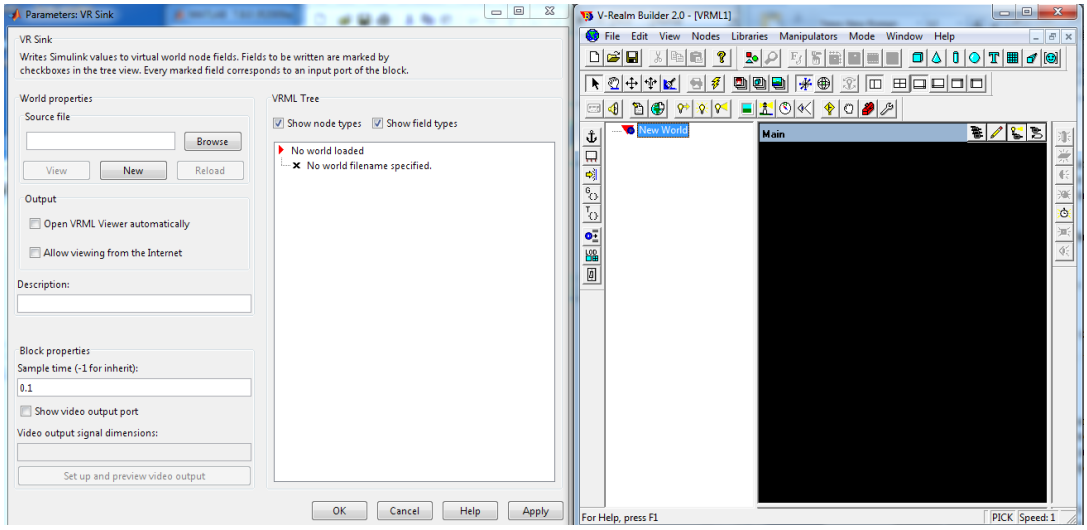
Sistemin geri beslemeli yapılmasının sebebi sistemin zamanla daha yüksek hatalara gitmesidir. Geri besleme ile bunun önüne geçilirken Unit Delay bloğu ile sistem ilk

çalışmaya başladığı anda geri besleme bloğunun çalışmasını anlık olarak engellemektir. Bu sayede sistemin ilk çalışmasında Jakobiyen değeri start dosyasından alınmış ve sonrasında geri beslemeden alınmaya başlanmış olur.

Bu şekilde sistem güzel çalışıyor gibi görülse de bazı hatalara sebebiyet verdiği sonradan görülmüştür. Şöyle ki; eğer hata değerlerini geri dönüşüm ile station bloğu çakıştıktan sonra alırsanız hatanızın her zaman belli bir seviyede kaldığını ve sıfıra inemediğini görürsünüz. Bunun sebebi station bloğundan gelen geri dönüşümden gelen uç organ noktası her seferinde station durağından gelen uç organ noktasının zman olarak bir gerisinde olacaktır. Bu sebeple hatayı minimize etmek çok zor olacaktır. Bu sebeple delay sonraki çalışmalarda delay zamanının kaldırıldığını göreceksiniz.

2.7 Virtual Reality Toolbox

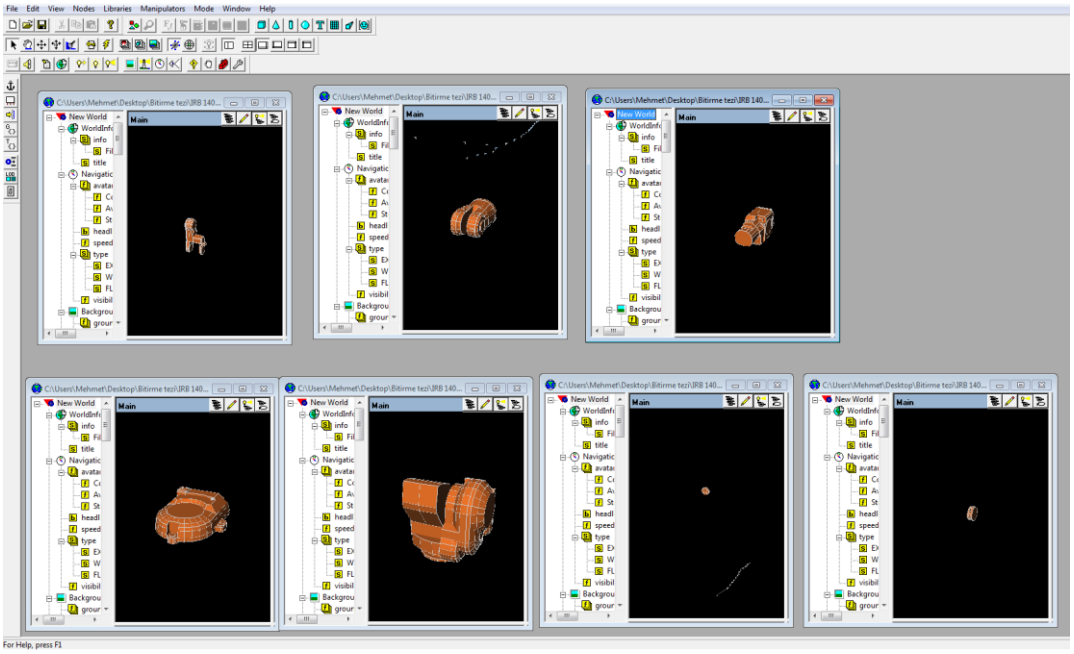
VR Sink bloğu sistemin 3 boyutlu görselini sağlamak için kullanılmıştır. ABB firmasının sitesinden alınan CAD çizimleri .wrml dosyası haline getirilip Virtual Toolbox içine yerleştirilmiştir.



Şekil 2.13 : Virtual reality toolbox arayüzü

Çizimlerin virtual içine yerleştirilmesi işlemi daha sonra oluşabilecek hataların önüne geçilebilmesi bakımından çok önemlidir. Öncelikle her bir parça ayrı ayrı .wrml dosyası haline getirilip sırasıyla virtual reality toolbox içinde açılmalıdır. Bunun için bir simulink dosyası açılıp "Simulink 3d Animation" ismi altında bulunan "VR Sink" bloğu dosyanın içine atılır ve bloğa çift tıklamak suretiyle karşımıza çıkacak ekrandan new seçeneğinin seçilmesiyle 3 boyutlu ekran kaşımıza çıkacaktır.

Açılan ekranda .wrml formatında kaydettiğimiz dosyaların hepsini açıp hepsini şekil 2.13'te olduğu gibi gösterebiliriz.



Şekil 2.14 : Virtual reality toolbox dosyasının hazırlanması.

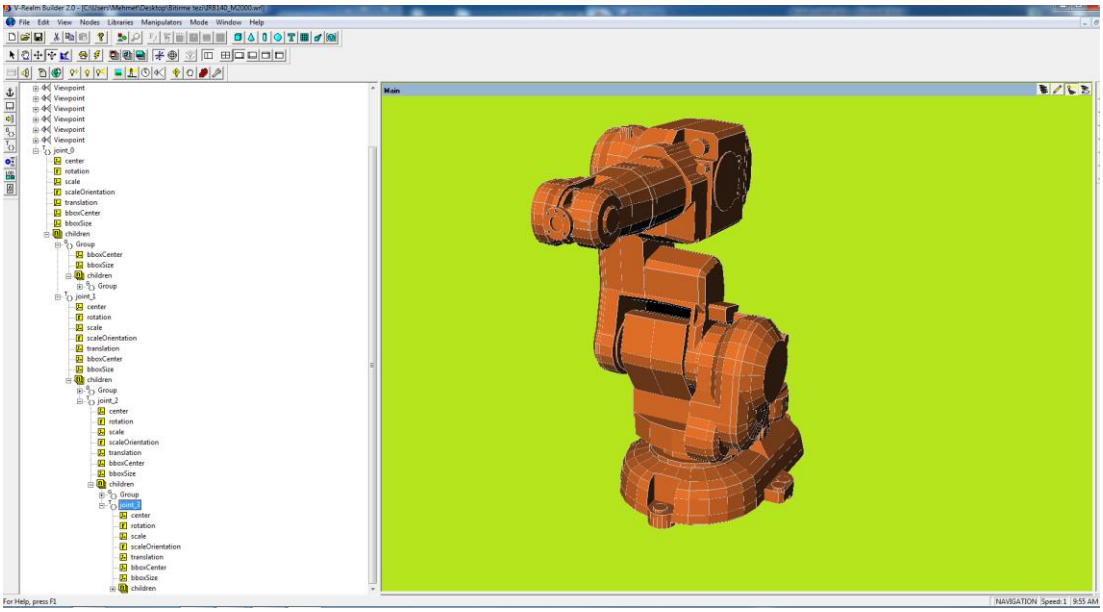
Bu parçaları sırasıyla virtual reality toolboxta birleştirmek için öncelikle en alttan en üste işlemleri yapmak gerekir. Eklenecek her bir parçanın transform dosyası kopyalanır ve bir önceki parçanın children bölümüne yapıştırılır. Bu şekilde bütün parçalar arka arkaya eklenmiş olur. Bu işlemde karşılaşılabilcek bazı sorunlar aşağıda ifade edilmiştir:

1- Kopyalanan parçanın boyutları belli bir oranda düşük olarak kopyalanmış olabilir. Bunun için Scale seçeneğini işaretleyip boyutlandırmayı tekrardan yapmak gerekir.

2- Kopyalanan her bir parça kopyalandığı parçanın koordinat merkezine göre yerleşir. Bu sebeple parçaların boyutlarını bilip buna göre parçaların konumlarını değiştirmek gerekir. Bunu translation komutuyla rahatça yapabilirsiniz.

3- Dönme ekseninde kaymalar oluşabileceğini her zaman göz önünde bulundurmak zorundayız. Bu sebeple center komutuyla parçanın merkezinin değiştirilebilecek olması bilinmek durumundadır.

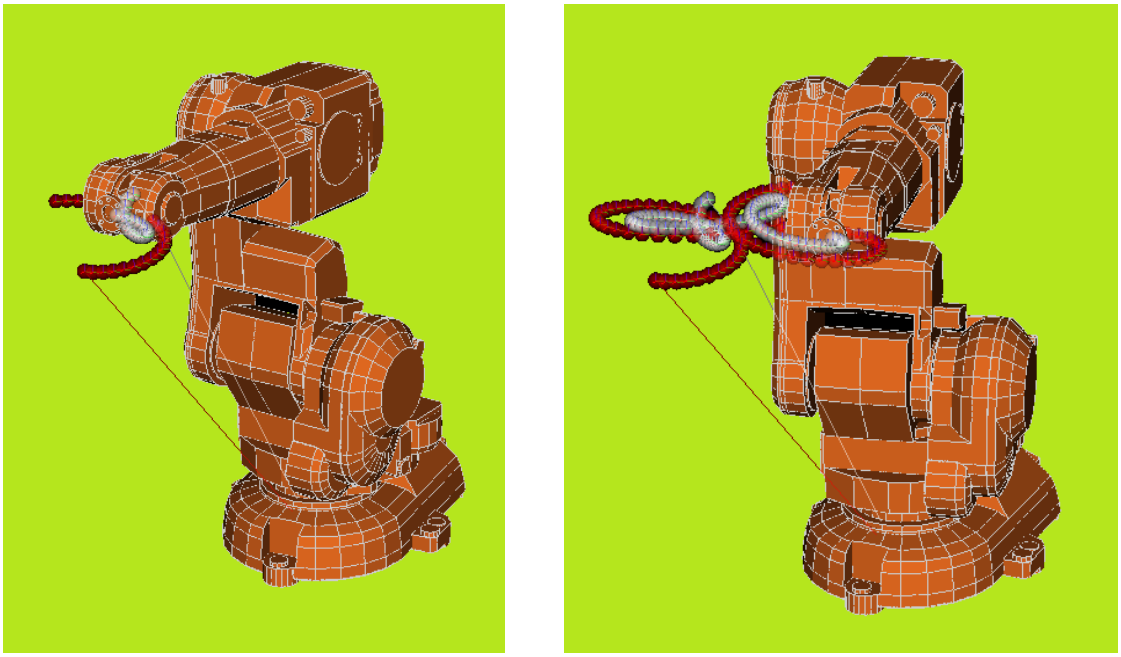
Şu belirtilmelidir ki .wrml çizim konusunda yeterli desteği sağlayan bir program değildir. Bu araç yapılan işlemleri görselleştirmek amacıyla kullanılmaktadır.



Şekil 2.15 : Virtual reality toolbox dosyalarının birleştirilmiş durumu.

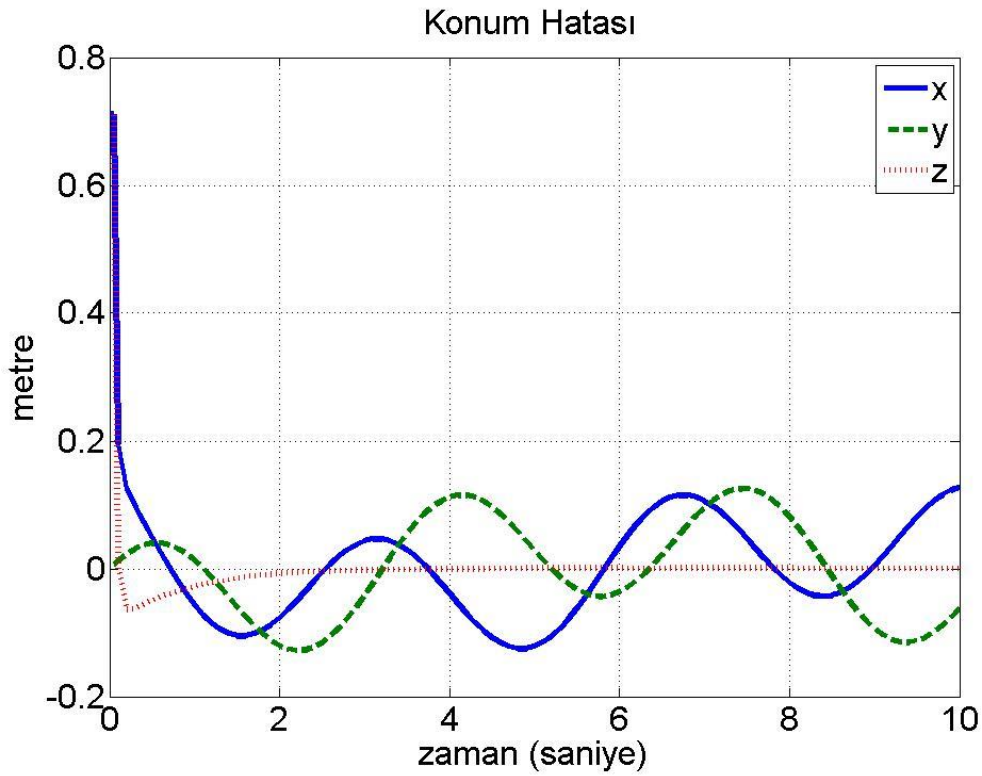
2.8 Modelin Sonuçları

Bizim kurmuş olduğumuz model kinematik hesaplar işin içine katılarak yapılan geri beslemeli bir sistemdir. Geri beslemeli sistem tek başına bir sistemin çalışmasında yeterli iyileştirmeyi gösteremediğinden ciddi düzeyde hataların meydana geldiği görülmektedir. ABB firmasının yaptığı IRB 140 adlı robotumuzun çalışma esnasında vermiş olduğumuz kelebek şeklindeki rotayı izleme durumu Şekil 2.16'da verilmiştir.



Şekil 2.16 : Robotun çalışma anı.

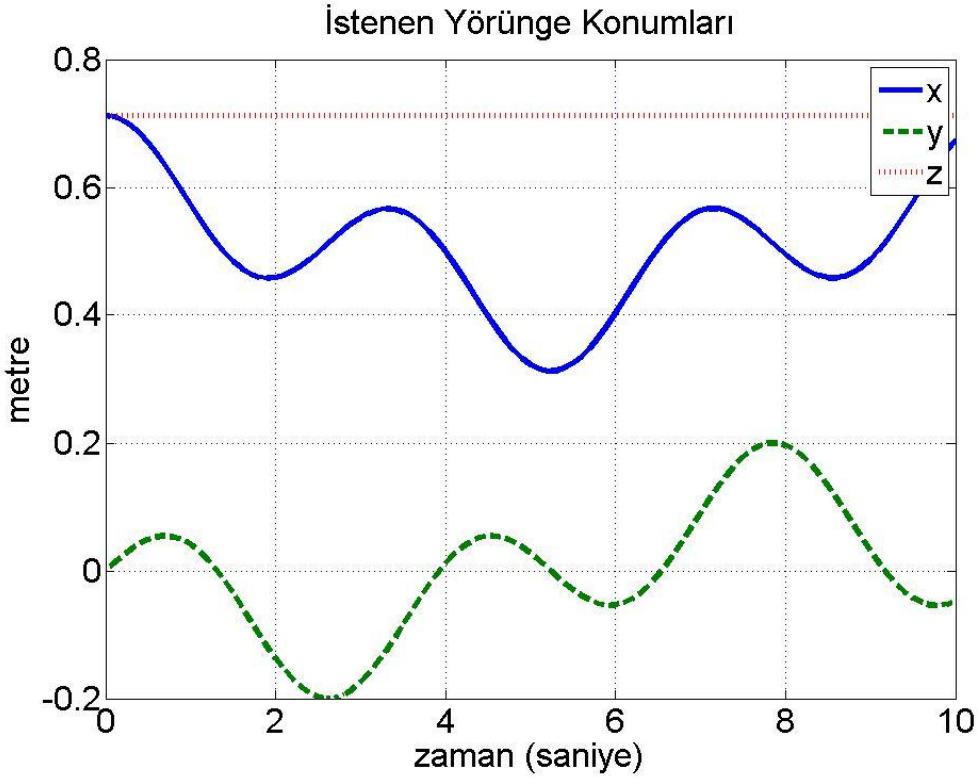
3 boyutlu sistemde belirgin olarak görülen hata, scope bloğu ile detaylı olarak incelenebilmektedir.



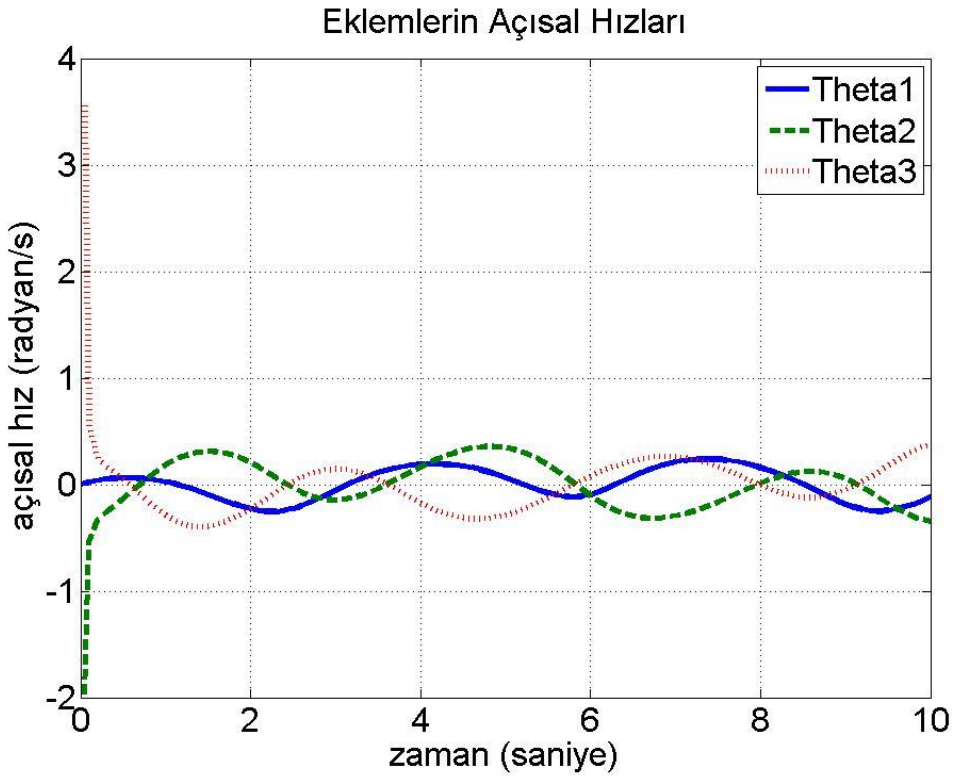
Şekil 2.17 : Uç noktasının yörüngeden sapma hatası.

İzlenecek rotanın t=0 anında başlangıç noktası [0.712 0 0.712] olarak kabul edilmiş ve robotun yükseklik değişmeyecek şekilde bir şekil çizmesi istenmiştir. IRB 140 robotu 1 m civarında bir boya sahip olup işlemin [0 0 0] noktasından başladığını kabul edersek bu durumda ilk hataların 0.7 metre olması kaçınılmazdır. Sistemin çalışmaya başlaması ile yükseklik değeri 3 sn. içinde tamamıyla oturmakta ancak x ve y eksenleri belirli hatalarla çalışmaya devam etmektedir. Bu çalışmada şekilde de görüldüğü üzere hiç bir kontrol uygulanmamış ve hatanın ±11 cm civarında olduğu görülmüştür. Bu hata oranı sanayide veya değişik çalışma ortamlarında kabul edilemez bir hata oranı olarak görülmesi muhtemel bir orandır. Bu sebeple bu hata oranın azaltacak uygulamaların yapılması kaçınılmazdır.

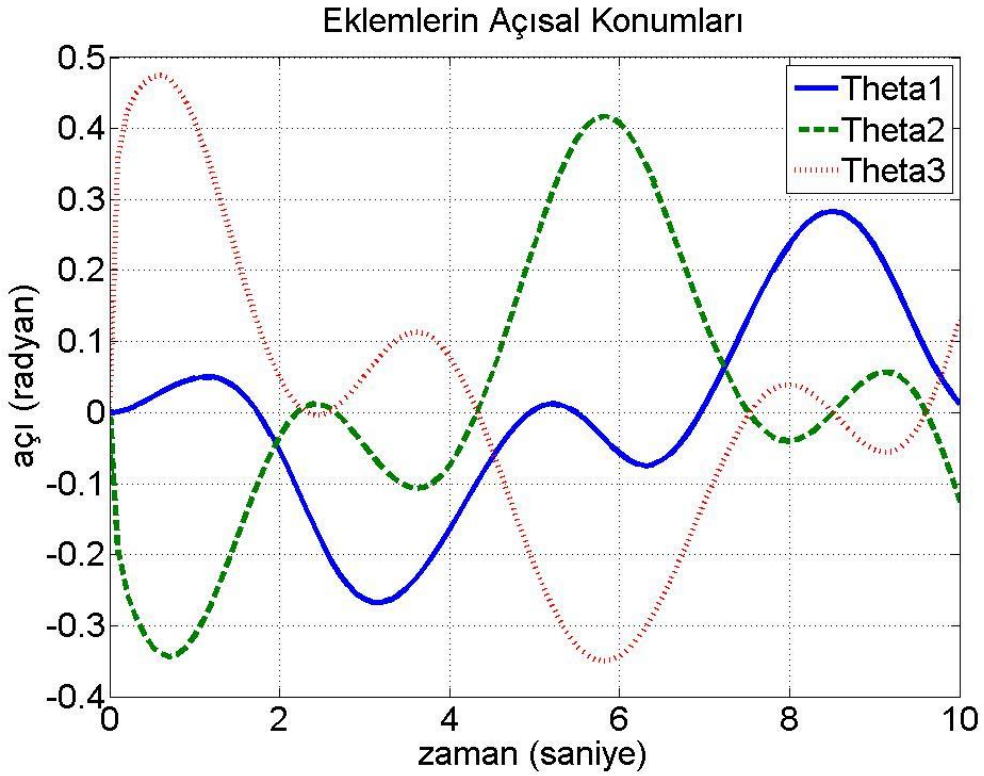
Simulink bloklarında robotumuzdan talep ettiğimiz rota ve eklemlerin konum ve hızları şekil 2. 18,2.19 ve 2.20'de gösterilmiştir.



Şekil 2.18 : Robotun izlemesi istenen rota.



Şekil 2.19 : Eklemlerin hızları.



Şekil 2.20 : Eklemlerin konumları.

Sistemimizin nonlinear bir sistem olmasından dolayı hata kaçınılmazdır ve kontrol sistemi kullanmaksızın gözardı edilemeyecek bir seviyede ilerlemektedir. Bu hatanın minimum düzeye indirilmesi için PID, Fuzzy, Uyarlamalı gibi değişik yöntemler uygulanabilir. Bu sistemin nonlinear olması nedeniyle PID uyguladığımızda hata istediğimiz seviyeye ulaşamayabilir ancak tolere edilebilir bir durumda ise bizim için uygun olacaktır.

3. KİNEMATİK KONTROL

3.1 Amaç

2. bölümde modellemesi yapılan sistemin ciddi manada hatalar içerdiği görülmüştür. Bu hataları en aza indirmek için sistem üzerinde bazı kontrol yöntemleri uygulanabilir. Bu yöntemler içinde kulllanım yaygınlığı da göz önüne alındığında ilk olarak PID kullanılması uygun görülmüştür.

3.2 PID

Dünya üzerinde günümüzde kullanılan kontrol sistemlerinin yarıdan fazlası PID kontrolörlerdir [28]. Bu derece yaygın olması kolay kullanımı ve iyi sonuç verebilmesinden kaynaklıdır.

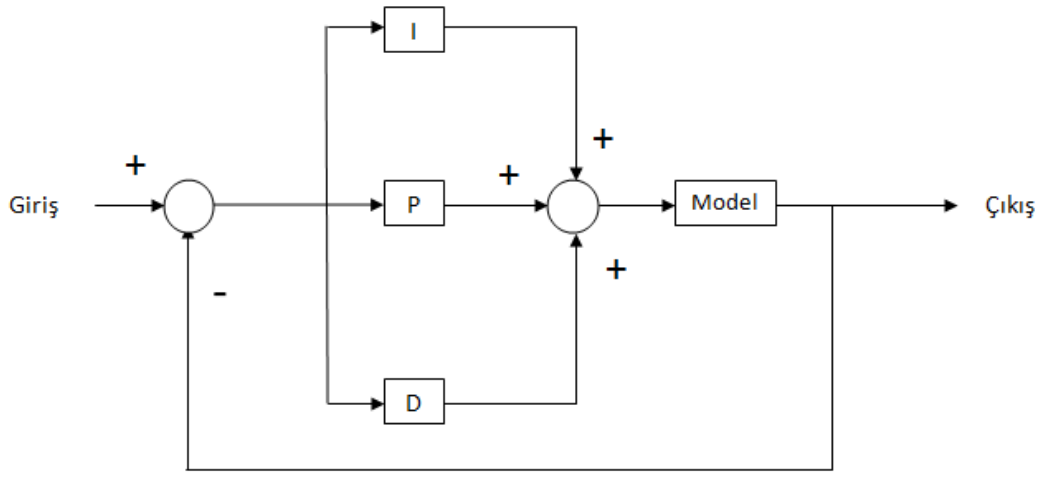
PID kontrol sistemi üç terimli bir yapıdır. Oransal (proportional), integrator (integrator), türevsel (derivative) kontrollerin birleşimi şeklindedir ve genel olarak geri beslemeli bir sistem şekildeki gibi verilebilir. Bu sistemde PID bir nevi denetleyici görevi görmekte ve hataları minimize etmektedir. Bu şekilde process adıyla belirtilen modelimizin içine girip çıkan sonuçların geri besleme yoluyla tekrardan incelenmesi sonucu sistem sürekli olarak devam edecektir. Kp hata değerinin oransal kazancının katını, Kd hatanın türevinin katını Ki ise hatanın integralinin katını ifade etmektedir [29].

Bir PID sistemi incelenirken Kp, Kd ve Ki değerlerinin sisteme ne şekilde etkiyeceğini bilmek durumundayız.

Kp; oransal kontrol olup yükselme zamanını azaltacak fakat kalıcı hal hatasını hiç bir zaman ortadan kaldırmayacaktır.

Ki; integral kontrol olup kalıcı hal hatasını azaltmakta etkili fakat geçici hata üzerinde olumsuz tesir bırakabilir.

Kd; türevsel kontrol olup sistemin kararlılığını artırmada, aşımı azaltmada ve geçici azalmanın gelişmesinde etkilidir. Genel olarak sistemi şekilde verildiği gibi ifade edebiliriz.



Şekil 3.1 : PID geri beslemeli sistem.

Çizelge 3.1 : PID kontrol parametrelerinin sistem üzerindeki etkileri.

Kontrol	Yükselme zamanı	Aşım	Yerleşme zamanı	Sürekli hal hatası
K_p	Azalır	Artar	Az Değişir	Azalır
K_i	Azalır	Artar	Artar	Sıfırlar
K_d	Az Değişir	Azalır	Azalır	Değişmez

Burada bilinmesi gerekn nokta şudur ki bu Kp, Ki, Kd değerlerinin hepsi birbirleriyle ilişkilidir ve birindeki değişim diğer ikisini etkileyebilecektir. Bu durum sebebiyle modelleme yapılırken hepsi göz önüne alınmalıdır [29].

PID katsayılarının hesabında katsayıların daha iyi ve tutarlı hesaplanabilmesi için John G. Ziegler ve Nathaniel B. Nichols bir yöntem ileri sürülmüş ve genel olarak ta kabul almıştır. Ziegler-Nichols yöntemi diye tabir edilen bu yöntemde bizim inceleyeceğimiz kısım kapalı çevrim olacaktır.

Kapalı çevrim ile uğraşırken alacağımız hata sinyali genel olarak bir salınım şeklinde sürekli hataya giden bir salınım olacaktır. Bu salınımın periyodu ve aşımı dikkate alınarak Ziegler-Nichols yöntemi ile sonuca ulaşmak mümkün olabilecektir.

Salınımın her bir periyodunu Pcr ile ve salınım kazancı Kcr ile ifade edilirse Kp, Ki, Kd değerleri için Çizelge 3.2'teki formüller kullanılabilir [30].

Çizelge 3.2 : Ziegler-Nichols kontrol katsayılarının belirlenmesi.

Kontrol	K_p	K_i	K_d
P	$0.5 * K_{cr}$	∞	0
PI	$\frac{K_{cr}}{2.2}$	$\frac{P_{cr}}{1.2}$	0
PID	$0.6 * K_{cr}$	$P_{cr}/2$	$P_{cr}/8$

Bu formülleri sisteme uygulayabilmek için öncelikle bizim sistemimizin salınımının belli bir seviyede ilerlemesi lazım. Ancak o şekilde belli bir periyod değeri ve aşım hesaplama imkânımız olabilecektir. Bizim sistemimiz nonlinear bir sistem olduğundan bu sisteme uygulayacağımız bir Kp katsayılı oransal kontrol ile sistemi lineer hale yaklaştırabiliriz.

Öncelikle hatamızı en aza indirebilecek şekilde bir Kp değeri seçmeye çalıştık. Seçilen Kp değerlerine göre hata durumları incelendiğinde Ek B.1'deki sonuçlar meydana gelecektir. Grafikleri yorumladığımız zaman gördüğümüz sonuç şu olacaktır ki: Grafiklerde de görüldüğü gibi hata oranları Kp değeri büyüdükçe küçülmekte ancak özellikle z ekseninde olmak üzere bazı bozuntular meydana gelmeye başlamaktadır. Çizelge 3.3'den Kp değerlerine bağlı olarak hata değerleri incelenmiştir.

Çizelge 3.3 : Kp değerlerine bağlı hata durumu.

Kp değerleri	**X ekseni**	**Y ekseni**	**Z ekseni ilk aşım**
0.5	160 mm	160 mm	0
1	90 mm	90 mm	4 mm
1.5	90 mm	90 mm	4 mm
2	75 mm	75 mm	5 mm
2.5	70 mm	70 mm	6 mm
3	55 mm	55 mm	8 mm
4	40 mm	40 mm	10 mm
5	30 mm	30 mm	15 mm
7	20 mm	20 mm	25 mm
10	15 mm	15 mm	55 mm
15	10 mm	10 mm	110 mm
20	1500 mm	750 mm	1100 mm

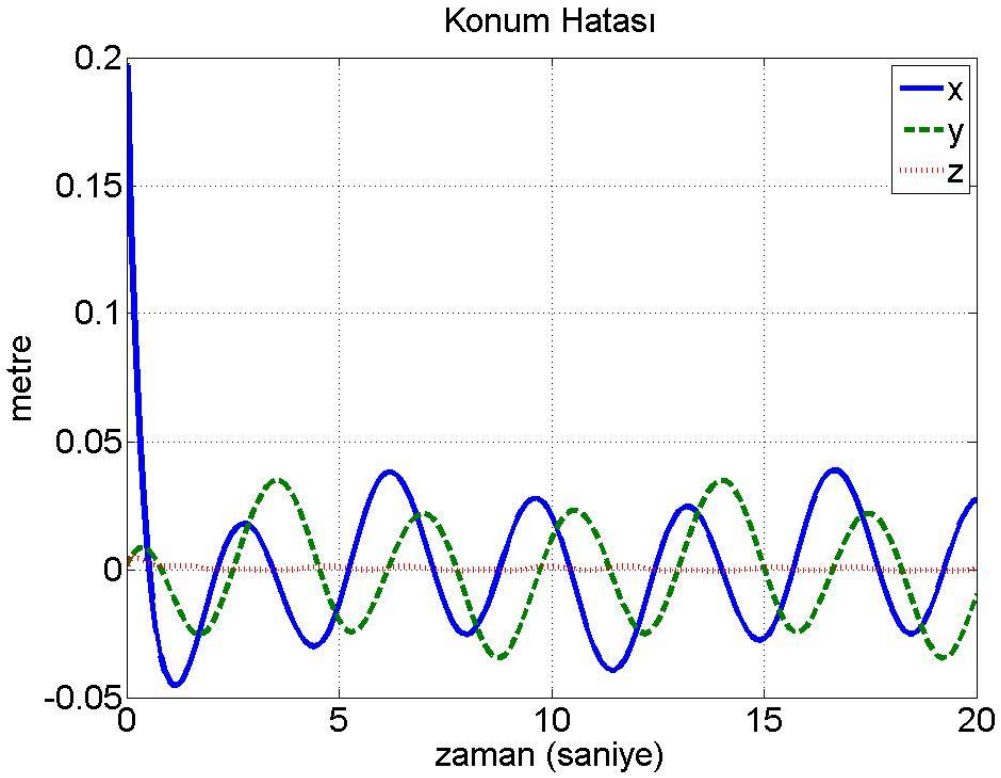
Çizelge3.3'te de görüldüğü gibi Kp değeri arttıkça x ve y eksenlerinde ki hata oranı düşmekte ancak z eksenindeki ilk aşım artmaktadır. Bu sebeple seçeceğimiz nokta her 3 eksen içinde en uygun değer olmalıdır. En son olarak görülmüştür ki belli bir noktadan sonra Kp değeri sistemi olumsuz noktada etkilemektedir.

3.2.1 Ziegler-Nichols ile katsayı belirleme

Bu yöntemi kullanımında ilk yapılacak iş Kp değerinin belirlenmesi olacaktır. Biz Kp değerini 3 olarak kabul ettik zira 2 ile yaptığımı denemlerde de gördük ki en az hatayı 3 ile sağlama imkânımızolmaktadır. Her bir eksenin periyodu farklı olduğundan her eksen ayrı bir PID kontrol sistemi oluşturduk. Sonuçta oluşturduğumuz sistem Çizelge 3.4'teki gibi olmuştur.

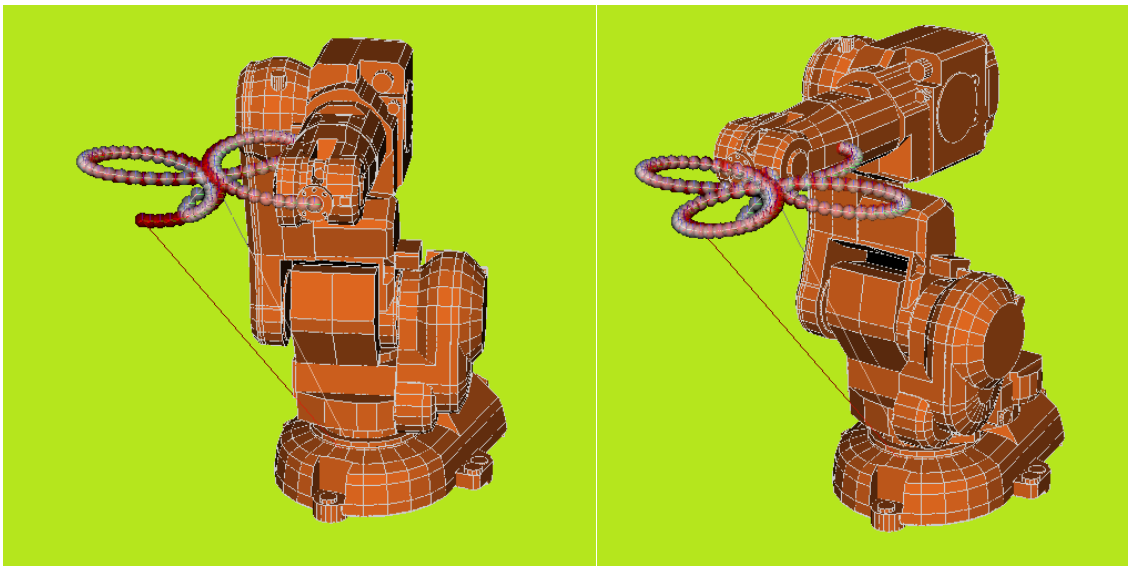
Çizelge 3.4 : Uygun görülen PID kontrol katsayıları.

	Kp	Ki	Kd
X ekseni	3	1.5	0.375
Y ekseni	3	1.8	0.45
Z ekseni	3	0.375	1.5



Şekil 3.2 : Ziegler-Nichols yöntemi ile hata analizi.

Bu şekilde sistemi çalıştırdığımızda aldığımız hatalar Şekil 3.2'de verilmiştir. Şekilde de görüldüğü üzere 55 mm boyutlarında olan hata 40 mm boyutlarına inmiştir. 40 mm sanayi robotlarında kabul edilebilir bir hata değildir ancak Kp değerini daha fazla artırmamız z ekseni üzerinde negatif etki yapacağından uygun değildir. Bu sebeple Kp değerini 3 tutmak suretiyle hatayı daha da azaltmak mecburiyetindeyiz. Daha öncede ifade ettiğimizi gibi integraör kalıcı hal atasını uzun sürede ortadan kaldırmayı hedefleyen bir etkendir oysaki türevsel kontrol kısa zamanda işlev gören bir etkendir. Bu sebeple türevsel kontrolün daha etkin olduğu yeni bir model tasarladık. Bir sonraki bölümde görülecek olan bu tasarım ile hatayı minimize etmiyi ve kabul edilebilir bir düzeye getirmeyi hedefledik.



Şekil 3.3 : Robotun çalışma anı.

3.3 PD Tasarımı

Ziegler-Nichols kuralı ile PID uyguladığımızda aldığımız 40 mm cevabı bizim için pek uygun bir cevap olmamıştır ve bunda integral kontrolünün büyük etkisi olmuştur. Bu sebeple Kp=3 olmak suretiyle sadece PD kontrolünü kabul ettiğimiz bir kontrol uygulamaya karar verdik. Bu uygulama sonunda Çizelge 3.5'te verilen sonuçlara ulaşılmıştır. Çalışmada z ekseni sürekli makul bir değerde tutulduğundan tabloda eklenmemiştir.

Çizelge 3.5 : Kp ve Kd değerlerine bağlı hata durumları.

Kp değerleri	Kd değerleri	X ekseni	Y ekseni
1	0.8	29 mm	25.5 mm
1	0.9	15 mm	14.5 mm
1	1	9 mm	6.5 mm
1	1.1	9.7 mm	8 mm
1	1.5	46 mm	47 mm
1	2	101 mm	98 mm
1.5	0.8	19 mm	18 mm
1.5	0.9	10.5 mm	9.3 mm
1.5	1	4.4 mm	3.6 mm
1.5	1.1	10 mm	9.5 mm
1.5	1.5	42.5 mm	42 mm
1.5	2	82.6 mm	83.6 mm
2	0.8	14.5 mm	13.5 mm
2	0.9	8.5 mm	8 mm
2	1	4.15 mm	3.8 mm
2	1.1	10.2 mm	10.4 mm
2	1.5	38 mm	39 mm
2	2	74 mm	74 mm
3	0.8	7.65 mm	6.8 mm
3	0.9	4 mm	3 mm
3	0.91	3.72 mm	2.5 mm
3	0.92	3.51 mm	2.05 mm
3	0.93	3.34 mm	2.5 mm
3	0.94	3.21 mm	2.89 mm
3	1	5.4 mm	4.7 mm
3	1.1	10.5 mm	9.8 mm
3	1.5	32.5 mm	32 mm
3	2	60 mm	59 mm

Çizelge 3.5'teki veriler ışığında şu ifade edilebilir ki Kp oransal kontrol olup yükseltilmesi hatayı azaltıcı bir rol oynamaktadır. Ancak Ek B.2'deki grafiklere bakıldığında Kp değerini yükseltmenin aynı zamanda bir bozucu etkiye de sebep

olduğu görülecektir. Bu durum grafiklere bakıldığında net olarak görülmektedir. Özellikle Kp=3 ve Kd=2 ve 3 alındığı grafiklerde barizdir. Biz işlemlerimizde bu bozucu etkinin ve minimum hatanın en uygun noktasını bulmak durumundayız.

Kp'yi yükseltmek Kalıcı durum hatası diye ifade edilen hatayı kabul edilemeyecek şekilde yükseltmektedir. Bu sebeple kalıcı durum hatasını sıfırlaması için bir çok çalışmada Ki kullanılmakktadır. İntegral kontrolcü her devirde hataların toplamı üzerinden hareket eder ve hedefe ulaşmayı daha uzun vadeye bırakır ancak kalıcı durum hatasını ortadan kaldırarak sonuca minimum hatayla ulaşmayı sağlar [31].

Biz bu durumda Ek B.2'de görüldüğü gibi Kp değerini 3 kabul ettiğimizde ilk 2 saniye boyunca ciddi derecede bozuntu ile karşılaştık ve hata değerlerimiz yaklaşık 5mm civarında seyretti ancak Kp=2 değerinde ise bozuntu ilk 1 saniye içinde kendini telafi etmiş ve hata durumu 4 mm civarında seyretmiştir. Bu sebeple Kp=3 kazanç değerini seçip Kd değerini 0.9 yaptık ve hem bozuntunun yarım saniyede bittiğini hem de sürekli hatanın 3 mm civarında seyrettiğini gördük. Bu sebeple bir kaç deneme daha yapıp en son Kp=3 ve Kd=0.92 olarak kabul ettik. Bu durumda toplamda maximum 3.5 mm civarında hatayla yörüngeyi izliyor durumda olacağız. Bu çoğunlukla daha az olurken hatanın maximum ulaşacağı seviye olarak görülebilecektir.

4. DİNAMİK MODELLEME

Bir robotun üzerinde kontrol sistemlerini uygulamadan önce matematik modellemesini yapmak ve bu model çerçevesinde kontrol algoritamlarına girmek gerekecektir. Bu çalışmada öncelikle kinematik modelleme ardından da Dinamik modelleme hakkında bilgi verilmiştir.

Eğer bir robot üzerinde modelleme yapılacaksa bunun için öncelikle kinematik hesabın düzgün bir şekilde yapılması gerekmektedir. Bu işlemlerle alakalı olarak birçok değişik modelleme yöntemi olmakla beraber genel olarak kullanılan bir sistemolan Denavit-Hartenberg yöntemi işlenecektir.

4.1 Amaç

2. bölümde modellemesi yapılan sistemin ciddi manada hatalar içerdiği görülmüştür. Bu hataları en aza indirmek için sistem üzerinde bazı kontrol yöntemleri uygulanabilir. Bu yöntemler içinde kulllanım yaygınlığı da göz önüne alındığında ilk olarak PID kullanılması uygun görülmüştür.

Önceki bölümde robot kolunun kinematik denklemlerini sunmuş ve bu denklemlerle basit bir modelin nasıl yapılabileceğini ifade etmeye çalışmıştık. Bu bölümde inceleyeceğimiz robot dinamiği, robot kollarının kütle, yerçekimi gibi kuvvetlerin de içinde olduğu matematik ifadeleriyle oluşturacağımız denklemlerin çözüme kavuşturulmasını ve bu denklemler yardımıyla bu kuvvetlerin modellenmesini içerecektir.

Dinamik modelde dikkat edilmesi gereken husus her bir eklemin dönme ekseninde motor olduğu öngörülerek işlem yapılacak ve gerek yerçekimi, gerekse hareketten doğan kuvvetlerin bu motorlara bindirdiği anlık yükler hesaplanarak hem robotun dur konumundaki hem de hareket anındaki anlık etkileşimler incelenecektir. Bu sebeple dinamik denklemlerin daha karmaşık ve işlem yükü getireceği aşikardır.

Biz bu bölümde robot dinamiğinde genel olarak kullanılan 2 yöntemden bahsedeceğiz. Bu yöntemler Newton-Euler ve Newton-Lagrange olarak ifade edilen

yöntemlerdir. Bu iki yöntem arasında bir karşılaştırma ile işe başlayacak olursak her ikisinin de ne amaçla kulanıldığına bağlı olarak diğerine üstünlük sağlayabildiği görülmektedir.

Newton-Euler metodunda temel olarak kuvvetler dikkate alınarak oluşturulan denklemler aracılığı ile dinamik modelleme yapılır. Dinamik eşitlikler her bir link için ayrı ayrı ve birbirinin peşi sıra yazılır ve işlemler sayısal değerler üzerinden yapılır ve dinamik özelliklerin çıkarılması için uygun bir yapıya sahiptir [32].

Lagrange-Euler yönteminde bütün yapı bir bütün olarak ele alınır ve potansiyel enerji ile kinetik enerji farkının hesabıyla dinamik model çıkartılır. Lagrange-Euler metodunda semboller üzerinden hareket edilir ve kontrol sistemleri uygulanması için oldukça uygundur.

Newton-Euler yöntemi basit işlemler gerektiren bir yapıya sahip olmasına, Euler-Lagrange metodunun dinamik modellemede çok hesap yüküne sahip olmasına ve Newton-Euler yöntemine göre daha etkisiz olmasına rağmen genel olarak Lagrange-Euler yöntemi tercih edilmektedir. Bunu sebebi günümüzdeki teknolojik gelişmeye bağlı olarak yapılacak hesapların artık hızlı olması ve Lagrange-Euler metodunun kontrolör tasarımındaki olumlu etkisidir [33].

Robotumuzda da hesap yükü fazla olmasına karşın en çok kullanılan yöntem olması sebebiyle Lagrange-Euler metodunu kullanmayı uygun gördük.

4.2 Lagrange-Euler Metodu

Lagrange-Euler yöntemi potansiyel enerji ve kinetik enerjinin farkı üzerinden sonuca ulaşılan bir metottur.

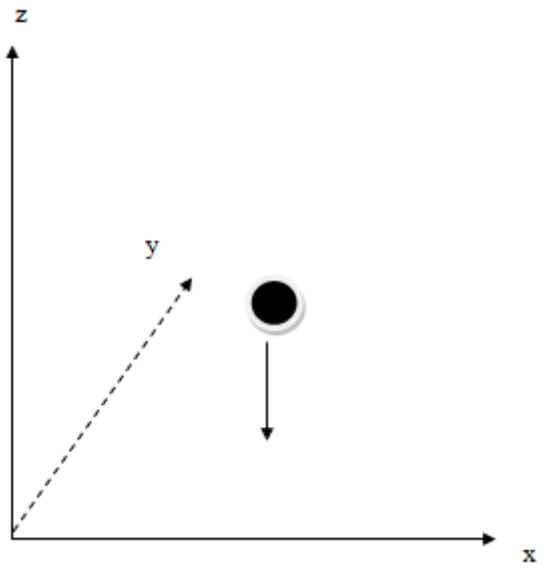
$$L = K - P \tag{4.1}$$

4.1 nolu denklemde K kinetik enerjiyi P ise potansiyel enerjiyi temsil etmektedir. Kinetik enerji formülü 4.2‘de verildiği gibidir ve hız ifadesinden gelmektedir. Potansiyel enerji ise tamamıyla konuma ve yerçekimine bağlıdır. Bu sebeple formülü 4.3’teki gibi genelleyebiliriz.

$$K = \frac{1}{2} m v^2 \tag{4.2}$$

$$L = \frac{1}{2}mv^2 - mgh \tag{4.3}$$

Şekil 4.1'de olduğu gibi yüksekten düşen bir topun kinetik ve potansiyel enerjisini hesaplamak istediğimizde 4.4 nolu formüldeki gibi denklemleri rahatlıkla kullanabiliriz.



Şekil 4.1 : Z ekseni boyunca düşen top.

$$K = \frac{1}{2}m\dot{z}^2 P = mgzL = \frac{1}{2}m\dot{z}^2 - mgz \tag{4.4}$$

Serbest düşüşteki top için dinamik bir modelleme yapmaya kalkarsak topu yeryüzüne doğru çeken yerçekiminden kaynaklanan bir kuvvet (F=m*a) vardır. Aynı şekilde topun yüzeye düşme hızını yavaşlatan "f" ile ifade edebileceğimiz sürtünme kuvveti de mevcuttur. Bu durumda topun dinamik modelini 4.5 nolu denklemde ki gibi ifade edebiliriz.

$$m\ddot{z} = mg - f \tag{4.5}$$

z yönündeki ivmelenme sürtünme kuvvetinin büyüklüğüne bağlı olarak değişecektir. Bu durum robot mekaizmalarında çok önem ifade etmese de uçak gibi hava araçlarında ciddi manalar ifade edecektir.

4.4 numaralı formülde ifade edildiği gibi bir L fonksiyonuna sahip olduğumuzu düşündüğümüzde L'nin hıza göre türevi kinetik enerjinin hıza göre türevine, L'nin konuma göre türevi potansiyel enerjinin konuma göre türevine eşit olacaktır. Bu durumu 4.5 ve 4.6 nolu formüllerde daha açık olarak görebiliriz.

$$\frac{\partial L}{\partial \dot{z}} = \frac{1}{2} m * 2 * \dot{z} = m\dot{z} \frac{\partial K}{\partial \dot{z}} = \frac{1}{2} m * 2 * \dot{z} = m\dot{z} \quad \textbf{(4.5)}$$

$$\frac{\partial K}{\partial z} = mg \frac{\partial L}{\partial z} = mg \frac{d}{dt} \frac{\partial L}{\partial \dot{z}} = m\ddot{z} \quad \textbf{(4.6)}$$

4.5 nolu formülümüzü hatırlayacak olursak o ifadenin yerine aşağıdaki ifade rahatça yazılabilir.

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{z}} = \frac{\partial L}{\partial z} - f \Longrightarrow f = \frac{\partial L}{\partial z} - \frac{d}{dt} \frac{\partial L}{\partial \dot{z}} \quad \textbf{(4.7)}$$

f bizim örneğimizde sürtünme kuvvetini temsil etmektedir. Kullanıldığı modele göre gerekli kuvveti ifade etmek gibi değişik manalara da gelebilir.

4.3 Teorik Model

IRB 140 sanayi robotunun dinamik denklemlerini çıkarırken Lagrange-Euler metodunu kullandık ve bu sebeple işe öncelikle kinetik enerji ve potansiyel enerji denklemlerinden başlamak zorundayız.

Bizim robotumuz 6 eklemli bir yapıya sahip olup her bir eklem bir öncekine bağlı şekilde hareket etmektedir. Uç eklemin hareketleri ilk eklemi yalnızca x,y,z doğrultusunda doğrusal bir kuvvet ile etkilememekte ayrıca açısal kuvvetlere de maruz bırakmaktadır. Bu sebeple kinetik enerji yalnızca doğrusal hızların değil ayrıca açısal hızların da bir bileşimi şeklinde olacaktır. Bu sebeple kinetik enerji denklemini 2.11 nolu formüldeki gibi ifade edebiliriz [34].

$$K = \frac{1}{2} * m * \dot{z}^2 + \frac{1}{2} * I * \dot{w}^2 \quad \textbf{(4.8)}$$

Burada z ve w ifadeleri hız ifadesinin yerine kullanılmıştır. Dolayısıyla bu ifadelerin yerine hız için genel ifade olan "v" ifadesini koyarsak daha anlaşılır olacaktır.

$$K = \frac{1}{2} * m * v^T * v + \frac{1}{2} * w^T * I * w \quad \textbf{(4.9)}$$

4.9 nolu denklemde m kütleyi v ise doğrusal hızı ifade etmektedir. Bu ifadelerde "v" ve "w" değerlerini kinematik bölümünde ifade ettiğimiz Jakobiyen matrislerinden çekeceğiz. I ise atalet tensörü diye isimlendirilen bir terimdir ve katı bir cismin kütlesel özelliklerinin temsil etmektedir. Her bir cismin atalet tensörü o cismin ağırlık merkezi temel alınmak suretiyle hesaplanır. Robot sisteminde bütün işlemler ana koordinat sistemine göre hesaplanması gerekmektedir. Bu nedenle 4.10 'da ki gibi hesaplanacak olan atalet tensörlerinin dönme matrisleri ile çarpılarak ana koordinat sistemine göre yeniden düznlenmesi gerekmektedir.

$$\mathrm{I} = \begin{bmatrix} \mathrm{I_{xx}} & \mathrm{I_{xy}} & \mathrm{I_{xz}} \\ \mathrm{I_{yx}} & \mathrm{I_{yy}} & \mathrm{I_{yz}} \\ \mathrm{I_{zx}} & \mathrm{I_{zy}} & \mathrm{I_{zz}} \end{bmatrix} \quad \textbf{(4.10)}$$

Gerekli düzenlemeler yapıldıktan sonra 4.9 nolu denklem aşağıdaki denkleme dönüşecektir.

$$K = \frac{1}{2} * m * (J_v * \dot{q})^T * (J_v * \dot{q}) + \frac{1}{2} * (J_w * \dot{q})^T * (R * I * R^T) * (J_w * \dot{q}) \quad \textbf{(4.11)}$$

Denklemde görüldüğü gibi $\dot{q}^T$ ve $\dot{q}$ terimleri ortak olarak bulunmaktadır. Bu durumda denklemi yeniden düzenleyip 2.16'da ki gibi yazabiliriz.

$$K = \frac{1}{2} * \dot{q}^T * \left[\mathrm{m} * {J_v}^T * J_v + {J_w}^T * (R * I * R^T) * J_w\right] * \dot{q} \quad \textbf{(4.12)}$$

Denklemlerin daha sade yazılabilmesi amacıyla köşeli parantez içinde gösterilen kısım "D" ile gösterilip kütle matrisi olarak ifade edilmektedir.

$$D = m * {J_v}^T * J_v + {J_w}^T * (R * I * R^T) * J_w \quad \textbf{(4.13)}$$

Potansiyel enerji için klasik ifade olan P=mgh eşitliğini kullanırasak çözüm ifademiz 4.13'de gösterildiği gibi olacaktır.

$$L = K - P = \frac{1}{2} * \dot{q}^T * \mathrm{D} * \dot{q} - m * g * h \quad \textbf{(4.14)}$$

Genel fikir vermesi açısından çıkarılan 4.13 nolu denkleme ek olarak robot kinematiğini hesaplamak için bazı bozuntuları da hesaba katmak zorundayız. Örnek olarak çalışan 2 eksenle bir robot mekanizmasını kafamızda canlandıralım. Dönel eklemlere sahip bu robotumuzda 2. eklem dönmektedir. Bu dönen eklem 1. ekleme bağlıdır ve dönmeden kaynaklanan hareket kuvvet sebebiyle 1. eklem üzerinde bir etki oluşturmaktadır. Eğer aradaki bağlantı bir an için gevşeyecek olsa 2. eklemin hareketinden oluşan kuvvet sebebiyle aradaki bağlantı kopacak ve robot dağılacaktır. **Corialis** ismi verilen bu hareketin kuvvet hesabı nasıl yapılmalıdır?

Bu kuvvetin hesabı yapılırken **Christoffel Symbolleri** diye adlandırılan denklemler kullanılacaktır. Denklem 4.15'de görülen bu denklemde k ifadesi link sayısını ifade etmektedir.

$$c_{ijk} = \frac{1}{2} * \left[\frac{\partial d_{kj}}{\partial q_i} + \frac{\partial d_{ki}}{\partial q_j} - \frac{\partial d_{ij}}{\partial q_k} \right] \quad \textbf{(4.15)}$$

Bizim genel formülümüz 4.16'da görüldüğü gibi kinetik enerji,potansiyel enerji ve sürtünme etkilerinden meydana gelmektedir.

$$D(q) * \ddot{q} + C(q, \dot{q}) * \dot{q} + g(q) + f(\dot{q}) = \tau \quad \textbf{(4.16)}$$

"g" terimi yerçekimi etkisini "f" ise sürtünmeyi ifade etmektedir.

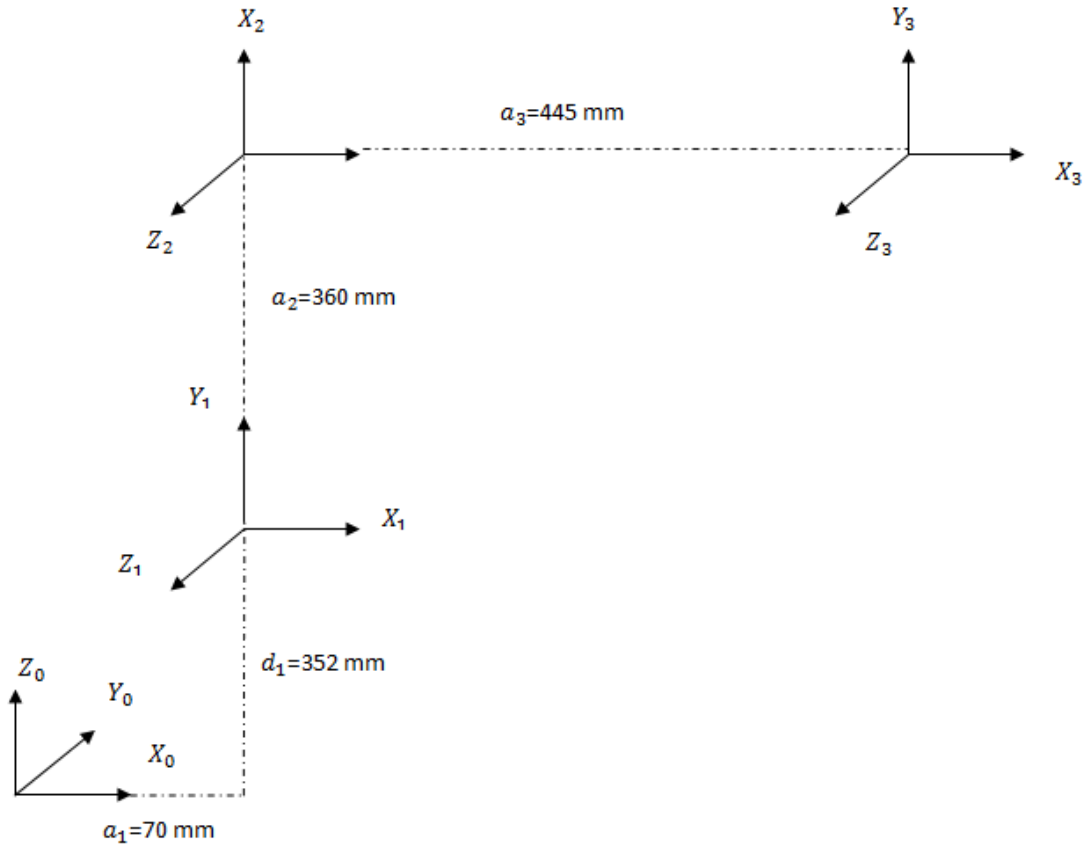
4.4 Matematiksel Modelleme

Matematiksel modelleme kısmında işin teorisi üzerine konuşuldu ancak bu teoriyi gerçek sisteme uyarladığımızda takip edeceğimiz problemler ve model şemaları farklılıklar gösterebilir. Dinamik model çıkarılırken dikkat edilecek ilk nokta kinematik denklemlerin düzgün şekilde çıkarılması olacaktır.

Üzerinde çalıştığımız robotda Denavit-Hartenberg yöntemini kullandık. Bu yöntemde yeni geliştirilen dinamik denklemler ciddi düzeyde hesap yüküne sahip olduğundan ancak ilk 3 eksen için işlem yapabilme durumundayız.

Robotik uygulamalarda kullanılan ikinci bir yöntem olan skrew teorisinde ise bu işlemler çok daha basit bir şekilde ve bilgisayara çok yük bindirmeden yapılabilmektedir.

Robotik uygulamalarında genel olarak kullanılması sebebiyle bir dinamik modellemede Denavit-Hartenberg yöntemini kullandık ancak bilgisayar yükünün aşırı olması sebebiyle ancak ilk 3 eksen için işlem yapabilme şansına sahip olduk. Bu sebeple dinamik model için uygun olabilmek koşuluyla dönüşüm matrisleri tekrar çıkarılmıştır.



Şekil 4.2 : IRB 140 robotunun 3 eklemli eksen ataması.

Bu durumda oluşacak olan matrisler aşağıda verilmiştir.

$$T_{0,1} = \begin{bmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & 70 * \cos(\theta_1) \\ \sin(\theta_1) & 0 & -\cos(\theta_1) & 70 * \sin(\theta_1) \\ 0 & 1 & 0 & 352 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{1,2} = \begin{bmatrix} \cos(\theta_2 + pi/2) & -\sin(\theta_2 + pi/2) & 0 & 360 * \cos(\theta_2 + pi/2) \\ \sin(\theta_2 + pi/2) & \cos(\theta_2 + pi/2) & 0 & 360 * \sin(\theta_2 + pi/2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{2,3} = \begin{bmatrix} \cos(\theta_3 - pi/2) & -\sin(\theta_3 - pi/2) & 0 & 445 * \cos(\theta_3 - pi/2) \\ \sin(\theta_3 - pi/2) & \cos(\theta_3 - pi/2) & 0 & 445 * \sin(\theta_3 - pi/2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Robot sistemimizin dinamik modellemesinde ilk iş olarak kütle matrislerini hesaplamak durumundayız. Kütle matrislerinin hesaplanması işlemi için 4.13 nolu denklemi kullacağız. Bu denklemi 3 eklemli bir yapı için düzenlersek aşağıdaki gibi bir denklem ortaya çıkacaktır.

$$D = m_1\left[{J_{v_1}}^T J_{v_1}\right] + m_2\left[{J_{v_2}}^T J_{v_2}\right] + m_3\left[{J_{v_3}}^T J_{v_3}\right] + {J_{v_1}}^T\left[R_{01} I_1 {R_{01}}^T\right] J_{w_1} + {J_{v_2}}^T\left[R_{02} I_2 {R_{02}}^T\right] J_{w_2} + {J_{v_3}}^T\left[R_{03} I_3 {R_{03}}^T\right] J_{w_3} \quad \textbf{(4.17)}$$

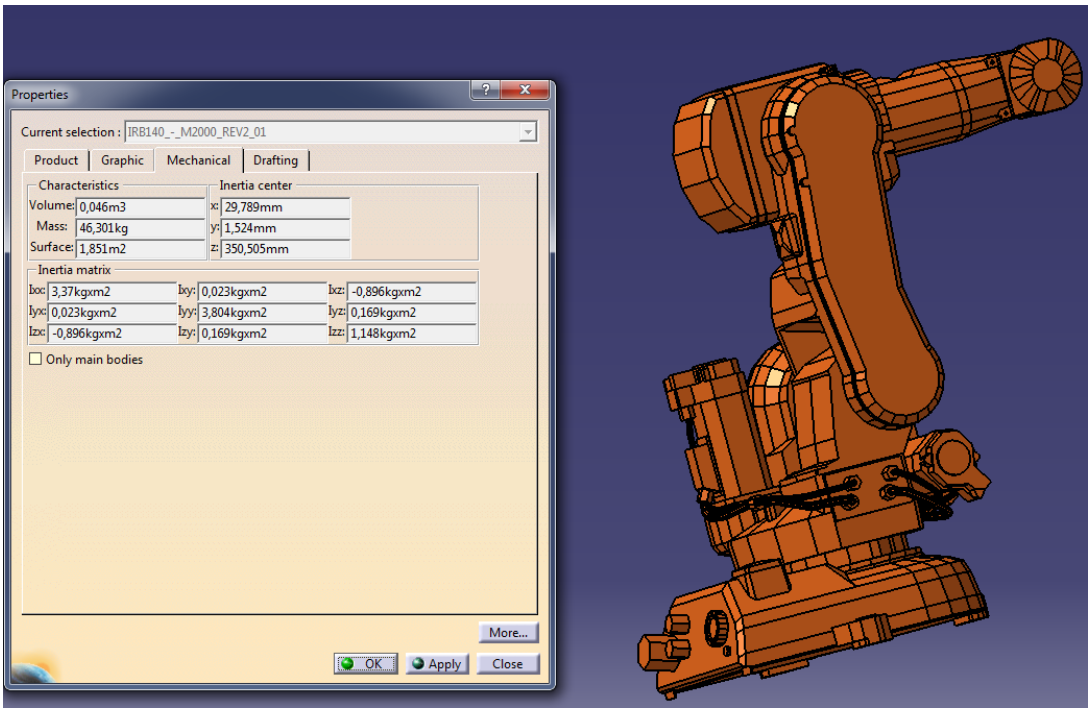
Bu denklemi çözmeye başladığımızda öncelikle jakobiyen matrislerini hesaplamak durumundayız. Bu işlemleri sırasıyla nasıl yaptığımızı gösterelim.

4.4.1 Ağırlık merkezi

Eklem 1

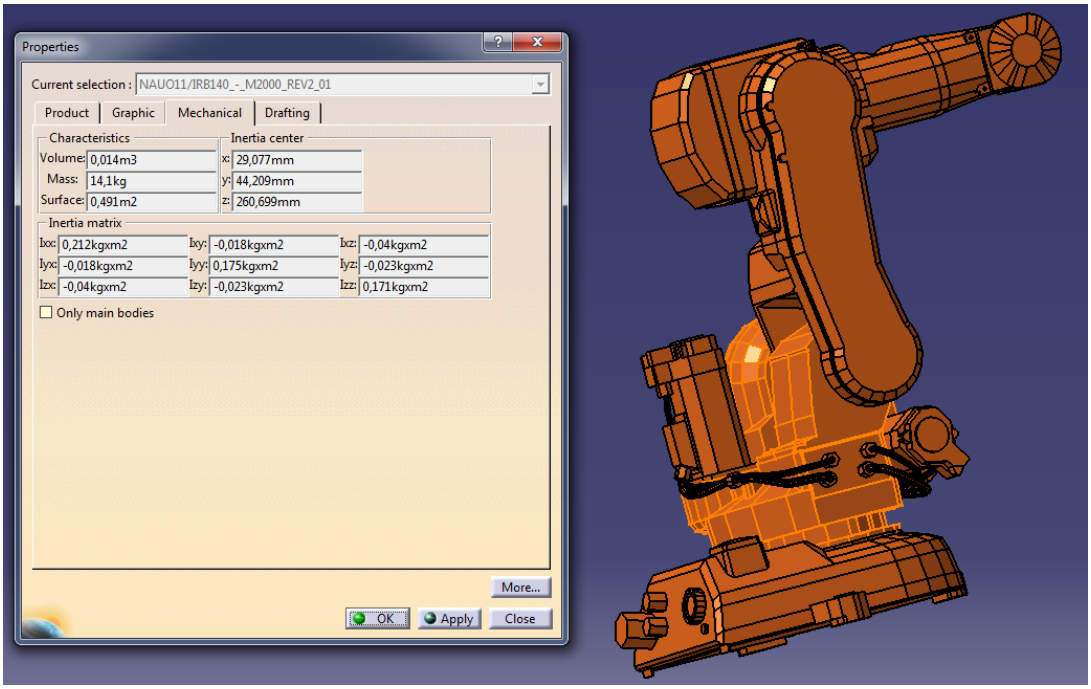
```
%% A1 and A1c
A1=[cos(t1),  0 ,   sin(t1),  0.07*cos(t1);
sin(t1),   0 ,  -cos(t1),  0.07*sin(t1);
0  ,       1 ,         0,  0.352;
0  ,       0 ,         0,  1];

r0c1=[0.0298  0.2607   -0.04421]';

R01=A1(1:3,1:3);
A1_r0c1=R01*r0c1;

A1c=[cos(t1),   0 ,   sin(t1),  A1_r0c1(1,1);
sin(t1),   0 ,  -cos(t1),  A1_r0c1(2,1);
0  ,       1 ,         0,  A1_r0c1(3,1);
0  ,       0 ,         0,  1]
```

Öncelikle 1. eklem için kullandığımız dönüşüm matrisini A1 olarak sisteme ilave ettik. Bunun ardından 1. eklemin ağırlık merkezini r0c1 matrisine yazdık. Bizim eklemlerimiz düzgün geometrik şekillere sahip eklemler olmadığından bu eklemlerin merkezlerinin bulunması bizim için oldukça zahmetli bir problemdir. Bu sebeple bu problemi ABB firmasından almış olduğumuz cad dosyasını Catia programında açmak suretiyle çözdük.



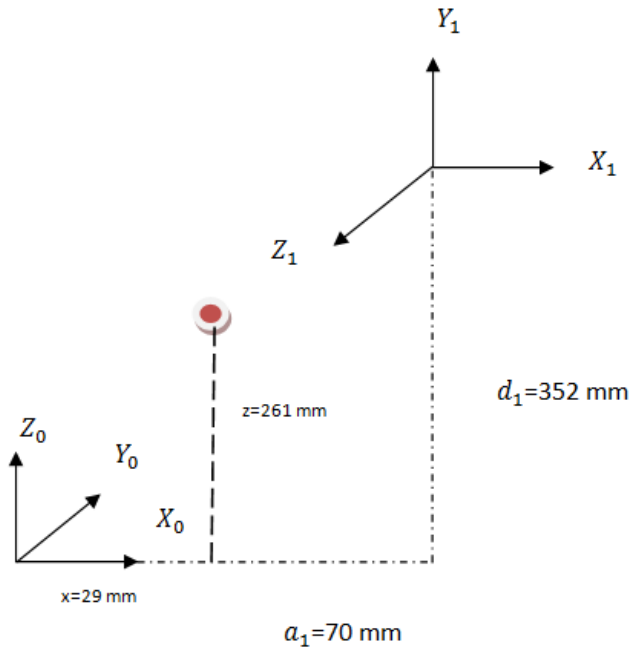
Şekil 4.3 : IRB 140 robotunun hacim, yüzey alanı, ağırlık merkezi ve atalet moment değerleri.

Şekil 4.3'te görüldüğü gibi robotumuzun bir bütün olarak değerlerine ulaşabilmemiz mümkündür. Birinci eklem için bu işlemi yapmaya kalktığımızda şekil 4.4'te ki değerleri kullanmamız gerekecektir.



Şekil 4.4 : IRB 140 robotunun ilk ekleminin hacim, yüzey alanı, ağırlık merkezi ve atalet moment değerleri.

Şekil 4.4'ten görüldüğü gibi ağırlık merkezlerimiz [29,077mm 44,209 mm 260,699mm] olarak görülmektedir. Ancak bu değerler ana koordinat sistemine göre düzenlenmiş koordinatlardır. Oysaki bize kullandığımız eklemin koordinat eksenine göre ağırlık merkezi gerekir. Yani 1. Eklem için 1 nolu ekseni kullanmak durumundayız. Bu işlemi 1. eklem için Şekil 4.5'te ele aldık.



Şekil 4.5 : IRB 140 robotunun ilk ekleminin ağırlık merkezi hesabı.

Bu işlemi adım adım yapalım;

X ekseni için verilen 29,077 mm değeri X_1 aynı yönde olduğu için değişmeyecektir.

Y ekseni Z_1 ile aynı doğrultuda ve farklı yönlerde olduğundan Y ekseni için verilen 44,209 değeri Z_1'de -44,209 mm olacaktır.

Z ekseni için verilen 260,699 mm değeri Y_1 ile aynı doğrultuda ve yönde olduğundan değişim yaşamayacaktır.

Burada görüldüğü gibi 1. Eksene göre değişiklik yapıldığında [29,077mm 260,699 mm -44,209mm] şeklinde bir durum ortaya çıkacaktır. Bu şekilde birinci ekleme yerleştirilen koordinat sistemine göre birinci eklemin kütle merkezinin konumunu bulmuş olduk ancak ana koordinat sistemine göre de birince eksenin kütle

merkezinin koordinatlarını bulmak durumundayız ve bu işlem için elimizdeki kütle merkezi matrisini dönme matrisi ile çarparız [34].

Sağlama: Bulmuş olduğumuz yeni dönüşüm matrisinin doğru olduğunu anlamak isteyebiliriz. Zira bu aşamada yapılacak bir hata bütün sistemimizi tehlikeye sokacaktır. Bunun en son bulduğumuz A1c matrisinde t1 zamanını sıfıra eşitleyerek sonucu inceleyelim.

$$A_{1c} = \begin{bmatrix} \cos(\theta_1) & 0 & \sin(\theta_1) & \frac{149 * \cos(\theta_1)}{5000} - \frac{4421 * \sin(\theta_1)}{100000} \\ \sin(\theta_1) & 0 & -\cos(\theta_1) & \frac{4421 * \cos(\theta_1)}{100000} + \frac{149 * \sin(\theta_1)}{5000} \\ 0 & 1 & 0 & 2607/10000 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

t1=0 için

$$A_{1c} = \begin{bmatrix} 1 & 0 & 0 & 0.0298 \\ 0 & 0 & -1 & 0.04421 \\ 0 & 1 & 0 & 0.2607 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Görüldüğü gibi t1=0 anında metre cinsinden olmak koşuluyla ana koordinat sistemine göre vermiş olduğumuz değerlere birebir ulaşabilmekteyiz. Bu durumda bu sistemin çözümnün doğru olduğunu kabul etmek durumundayız.

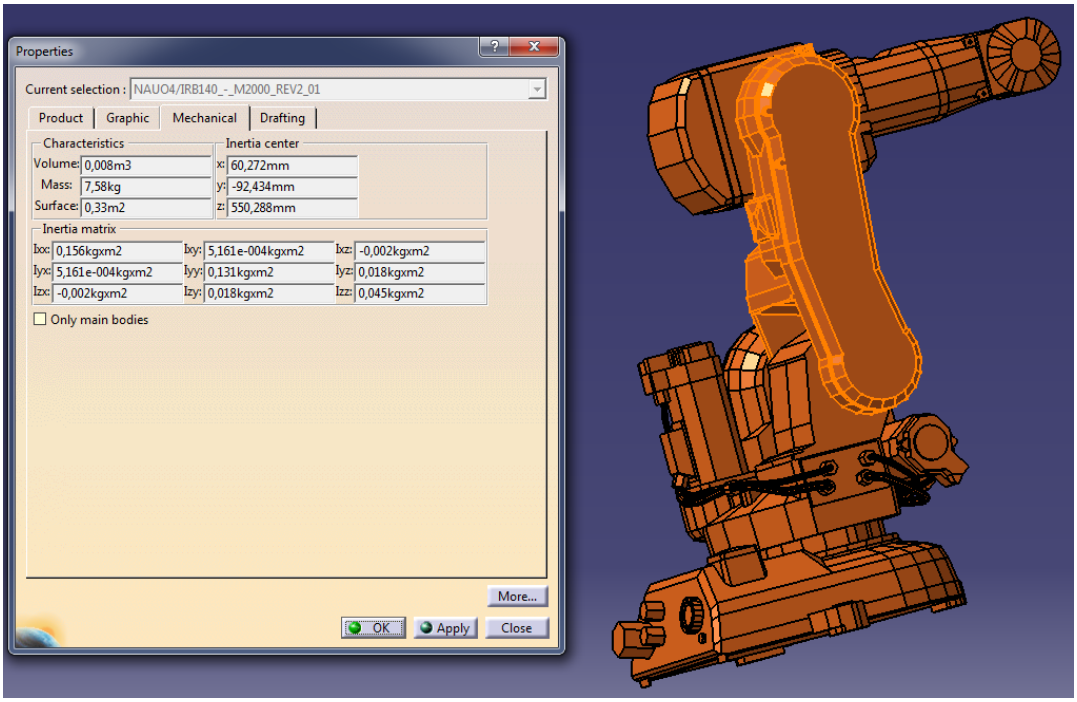
Eklem 2

```
%% A2 and A2c
A2=[cos(t2+pi/2),  -sin(t2+pi/2),  0,   0.36*cos(t2+pi/2);
sin(t2+pi/2),   cos(t2+pi/2),  0,   0.36*sin(t2+pi/2);
    0,         0,                  1,   0;
    0,         0,                  0,   1];

r1c2=[0.19829  0.0093   0.09243]'

R12=A2(1:3,1:3);
A2_r1c2=R12*r1c2;

A2c=[cos(t2+pi/2),  -sin(t2+pi/2),  0,   A2_r1c2(1,1);
sin(t2+pi/2),   cos(t2+pi/2),  0,   A2_r1c2(2,1);
     0,         0,                  1,   A2_r1c2(3,1);
     0,         0,                  0,   1]
```

Şekil 4.6'dan görüldüğü gibi ağırlık merkezlerimiz [60,272mm -92,434 mm 550,288mm] olarak görülmektedir. Ancak bu değerler ana koordinat sistemine göre düzenlenmiş koordinatlardır. Oysaki bize kullandığımız eklemin koordinat eksenine göre ağırlık merkezi gerektir. Yani 2. eklem için 2 nolu ekseni kullanmak durumundayız. Bu işlemi 2. eklem için Şekil 4.7'de ele aldık.

Şekil 4.7'de ele alınan işlemde ana koordinat merkez sistemine göre alınmış olan ağırlık merkezi 2. eklem koorinat sistemine göre tekrardan düzenlenmiş ve aşağıda ifade edilen denklemlerdeki gibi bulunmuştur.

X= 550,288-352=198,288 mm Y= 70-60,272=9,728 mm Z= 92,434 mm



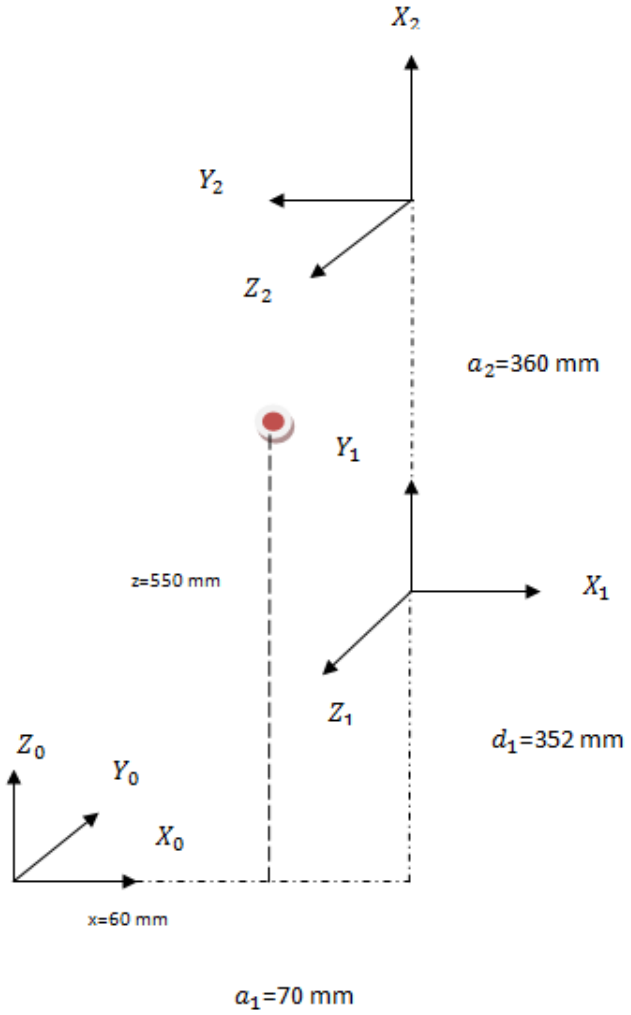
Şekil 4.6 : IRB 140 robotunun ikinci ekleminin hacim, yüzey alanı, ağırlık merkezi ve atalet moment değerleri.

X ekseni için verilen 60,272 mm değeri Y_2 aynı doğrultuda farklı yönde olduğu için eksi olacak ancak 70 mmlik eksen kaymasını hesaba kattığımızda 70-60,272=9,728 mm olacaktır.

Y ekseni Z_2 ile aynı doğrultuda ve farklı yönlerde olduğundan Y ekseni için verilen -92,434 değeri Z_2'de 92,434 mm olacaktır.

Z ekseni için verilen 550,288 mm değeri X_2 ile aynı doğrultuda ve yönde olduğundan işaretinde bir değişim yaşanmayacak ama ilk eklemden doğan 352 mm lik fark sebebiyle 550,288-352=198,288 mm olacaktır.

Burada görüldüğü gibi 2. Eksene göre değişiklik yapıldığında [198,288 mm 9,728 mm 92,434 mm] şeklinde bir durum ortaya çıkacaktır.



Şekil 4.7 : IRB 140 robotunun ikinci ekleminin ağırlık merkezi hesabı.

Sağlama: Bulmuş olduğumuz bu matrisin doğru olduğunu anlamak istediğimizde;

$$A_{2c} = \begin{bmatrix} \cos(\theta_2 + pi/2) & -\sin(\theta_2 + pi/2) & 0 & -\frac{93 * \cos(\theta_2)}{10000} - \frac{19829 * \sin(\theta_2)}{100000} \\ \sin(\theta_2 + pi/2) & \cos(\theta_2 + pi/2) & 0 & \frac{19829 * \cos(\theta_2)}{100000} - \frac{93 * \sin(\theta_2)}{10000} \\ 0 & 0 & 1 & 9243/100000 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

t1=0 için

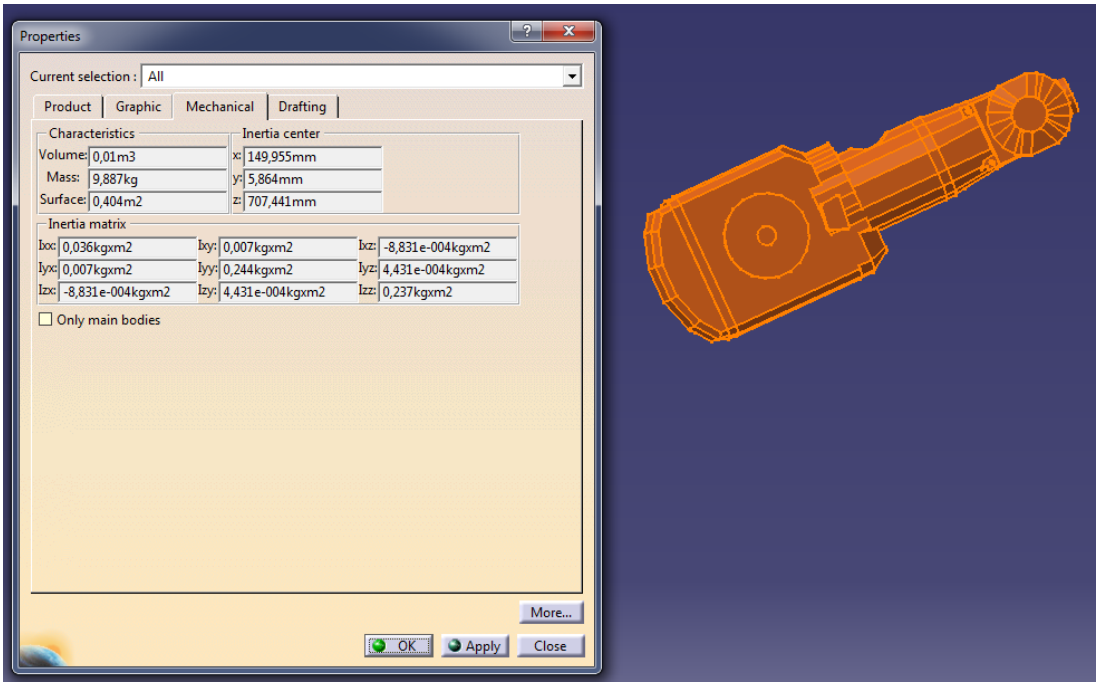
$$A_{2c} = \begin{bmatrix} 0 & 1 & 0 & 0.0093 \\ 1 & 0 & 0 & 0.19829 \\ 0 & 0 & 1 & 0.09243 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Görüldüğü gibi t1=0 anında metre cinsinden olmak koşuluyla ana koordinat sistemine göre vermiş olduğumuz x ve y'nin yer değiştirmesi koşuluyla ulaşmak mümkündür. Bu yer değiştirmenin sebebi de sistemin 90 derecelik bir kayma ile çalışmaya başlamasıdır.

3. eklemin hesabı yapılırken son 3 eklem birleştirilerek tek bir eklem gibi işlem yapılmıştır. Bu sayede matematikyükünün daha az olması amaçlanmıştır. 6 eksenli sistem tasarımı yapılmak istendiğinde 3 eklemli yapının aynısı 6 eksen için uygulanabilmektedir.

Eklem 3

```
%% A3 and A3c
A3=[cos(t3-pi/2),  -sin(t3-pi/2),  0,  0.445*cos(t3-pi/2);
sin(t3-pi/2),    cos(t3-pi/2),  0,  0.445*sin(t3-pi/2);
0 ,                 0 ,         1,        0;
0 ,                 0 ,         0,        1];

r2c3=[0.07996  -0.00456   -0.00586]';
R23=A3(1:3,1:3);
A3_r2c3=R23*r2c3;

A3c=[cos(t3-pi/2),  -sin(t3-pi/2),  0,  0.07996*sin(t3)-
0.00456*cos(t3);
sin(t3-pi/2),    cos(t3-pi/2),  0,  -0.07996*cos(t3)-
0.00456*sin(t3);
0 ,                 0 ,         1 , -0.00586;
0 ,                 0 ,         0 ,        1];
```



Şekil 4.8 : IRB 140 robotunun üçüncü eklemi hacim, kütle, yüzey alanı, ağırlık merkezi ve atalet momenti değerleri.

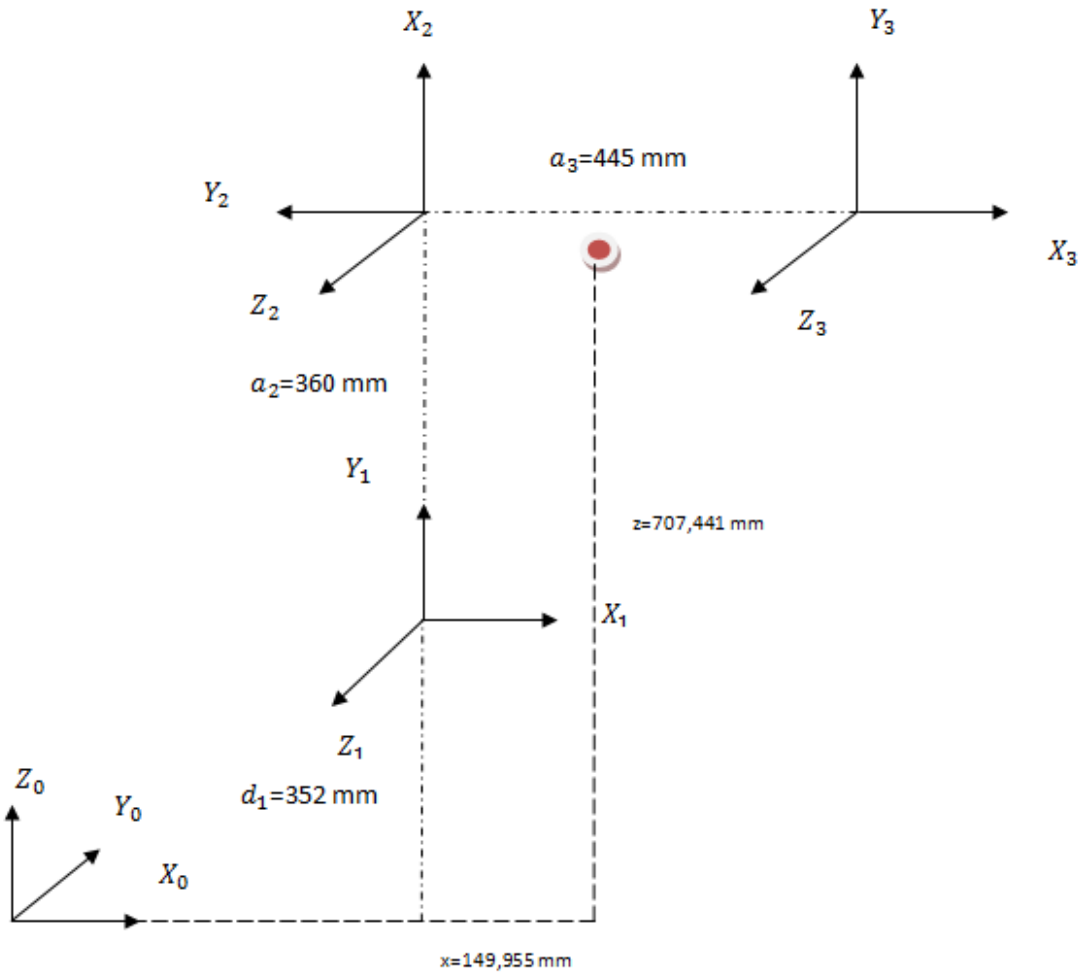
Şekil 4.8'de görüldüğü gibi ağırlık merkezlerimiz [149,955 mm 5,864 mm 707,441 mm] olarak görülmektedir. Bu işlemi 3. eklem için Şekil 4.9'da ele aldık.

X ekseni için verilen 149,955 mm değeri X_3 aynı doğrultuda ve yönde olduğu için işaret değişimi yaşanmayacak 70 mmlik eksen kaymasını hesaba kattığımızda 149,955-70=79,955 mm olacaktır.

Y ekseni Z_3 ile aynı doğrultuda ve farklı yönlerde olduğundan Y ekseni için verilen 5,864 değeri Z_3'te -5,864 mm olacaktır.

Z ekseni için verilen 707,441 mm değeri Y_3 ile aynı doğrultuda ve yönde olduğundan işaretinde bir değişim yaşanmayacak ama ilk iki eklemden doğan 712 mm'lik fark sebebiyle 707,441-712=-4,559 mm olacaktır.

Burada görüldüğü gibi 3. eksene göre değişiklik yapıldığında [79,955 mm -4,559 mm -5,864 mm] şeklinde bir durum ortaya çıkacaktır.



Şekil 4.9 : IRB 140 robotunun üçüncü eklemi ağırlık merkezi.

Sağlama:

A_{3c}

$$= \begin{bmatrix} \cos(\theta_3 - pi/2) & -\sin(\theta_3 - pi/2) & 0 & \frac{1999 * \cos\left(\theta_3 - \frac{pi}{2}\right)}{25000} + \frac{57 * \sin\left(\theta_3 - \frac{pi}{2}\right)}{12500} \\ \sin(\theta_3 - pi/2) & \cos(\theta_3 - pi/2) & 0 & \frac{1999 * \sin\left(\theta_3 - \frac{pi}{2}\right)}{25000} - \frac{57 * \cos\left(\theta_3 - \frac{pi}{2}\right)}{12500} \\ 0 & 0 & 1 & -293/50000 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

t3=0 için

$$A_{3c} = \begin{bmatrix} 0 & 1 & 0 & -0.00456 \\ -1 & 0 & 0 & -0.07996 \\ 0 & 0 & 1 & -0.00586 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Görüldüğü gibi t3=0 anında metre cinsinden olmak koşuluyla ana koordinat sistemine göre vermiş olduğumuz x ve y'nin yer değiştirmesi koşuluyla ulaşmak mümkündür. Bu yer değiştirmenin sebebi de sistemin 90 derecelik bir kayma ile çalışmaya başlamasıdır.

4.4.2 Jakobiyen matrisi

Jakobiyen matrisi hesabının nasıl yapılduığını kinematik bölümünde incelemiştik. Şimdi aynı işlemin bir benzerini dinamik modelleme için yapacağız ancak dikkat edilmesi gerekn bir kaç nokta bulunmaktadır. Şöyle ki; bizim yaptığımız hesaplamalarda bir eklemin ağırlık merkezi hesaplamaları hep bir önceki eklemin eksen atamalarına göre yapılmış ve sonuç bulunmuştu. Kütle jakobiyenlarını bulurken de aynı işlem tekrarlanacaktır.

```
%% center of mass jacobian

T01=A01
T02=A01*A12
T03=A01*A12*A23

T01c=A01c
T02c=A01*A12c
T03c=A01*A12*A23c

R01=T01(1:3,1:3);
R02=T02(1:3,1:3);
R03=T03(1:3,1:3);

z0=[0 0 1]';  Q0=[0 0 0]';
```

```
z1=T01(1:3,3); Q1=T01(1:3,4); Q1c=T01c(1:3,4);
z2=T02(1:3,3); Q2=T02(1:3,4); Q2c=T02c(1:3,4);
z3=T03(1:3,3); Q3=T03(1:3,4); Q3c=T03c(1:3,4);

Jvc1=[cross(z0,(Q1c-Q0)) zeros(3,1) zeros(3,1)]
Jwc1=[      z0           zeros(3,1) zeros(3,1)]

Jvc2=[cross(z0,(Q2c-Q0)) cross(z1,(Q2c-Q1)) zeros(3,1)]
Jwc2=[      z0                z1           zeros(3,1)]

Jvc3=[cross(z0,(Q3c-Q0)) cross(z1,(Q3c-Q1)) cross(z2,(Q3c-Q2))]
Jwc3=[      z0                z1                z2]
```

4.4.3 Kütle matrisi hesabı

Kütle matrisinin bulunması ile ilgili denklemi daha önce vermiştik. Bu denklemin kullanımında bize her bir eklemin kütlesi ve atalet momentleri gerekecektir. Bu değerleri Şekil 4.4, 4.6, 4.8'de ki değerlerden okumak mümkündür.

$$m = \begin{bmatrix} 14{,}1 \\ 7{,}58 \\ 9{,}887 \end{bmatrix} kg I_1 = \begin{bmatrix} 0{,}212 & -0{,}018 & -0{,}04 \\ -0{,}018 & 0{,}175 & -0{,}023 \\ -0{,}04 & -0{,}023 & 0{,}171 \end{bmatrix} kgm^2$$

$$I_2 = \begin{bmatrix} 0{,}156 & -0{,}0005161 & -0{,}002 \\ -0{,}0005161 & 0{,}131 & 0{,}018 \\ -0{,}002 & 0{,}018 & 0{,}045 \end{bmatrix} kgm^2$$

$$I_3 = \begin{bmatrix} 0{,}036 & 0{,}007 & -0{,}0008831 \\ 0{,}007 & 0{,}244 & 0{,}0004431 \\ -0{,}0008831 & 0{,}0004431 & 0{,}237 \end{bmatrix} kgm^2$$

Bu verileri kullanarak işleme tabi tuttuğumuzda kütle matrisine Ek B.1'deki gibi ulaşılmış olacaktır. Ancak ulaşılmış olan bu değer anlık bir ifadeyi ele almaktadır ve zamana bağlı olarak değişen açı değerleriyle değişim göstermektedir.

Bu kütle matrisi 4.19 nolu denklemde görüldüğü üzere her bir link değişkeninin 2. dereceden türeviyle çarpılmak suretiyle kinetik enerjiyi ifade edecektir yani şu an kütle matrisi 2.dereceden türev modundadır. Simulink model oluştururkn 2 kez integral çarpanı ile çarparak kütle matrisini konum seviyesine indirgedik ve işlemleri sonuca götürdük.

4.4.4 Christofel sembolleri

Christofel sembolleri daha önce de bahsettiğimiz gibi sistemin dönme hareketinden kaynaklanan bir kuvvettir ve 4.19 nolu denklemde ifade ediildiği gibi 1. dereceden türev mertebesindedir.

Bu kuvvetlerin hesabında C ile göstereceğimiz Coriolis matrislerini kullandık [35].

$$C_{ij} = \sum_{k=1}^{n} c_{ijk}(q)\dot{q}_k = \sum_{k=1}^{n} \frac{1}{2}\left[\frac{\partial D_{kj}}{\partial q_k} + \frac{\partial D_{ki}}{\partial q_j} - \frac{\partial D_{ij}}{\partial q_i}\right]\dot{q}_k \quad \textbf{(4.18)}$$

Bu işlemlerde i,j,k değerlerini kullanış durumunuz sisteminizin teorisine göre değişiklikler gösterebilir yani aynı sistemi k yerine i'yi 1'den n'e kadar değişken kabul ederekte yapabilirsiniz zira sistemin sağlıklı çalışması adına hepsini 1'den n'e kadar değişen değişkenler olarak kabul etmek durumundayız. Bu sistemi Ek1'de göreceğiniz gibi bir while döngüsü içinde çalıştırdığımızda Coriolis matrisleri Ek B.2'deki gibi çıkmaktadır.

4.4.5 Potansiyel enerji

Robotun işlemleri içinde potansiyel enerji genel olarak kullanılan mgh formülü ile ifade edilecektir. Her bir eklemin potansiyel enerjisi ayrı ayrı hesaplanacağından denklem 4.19'daki gibi bir durum meydana gelecektir. Dönüşüm matrislerinin 3. Satır ve 4. Sütunları yüksekliği vermektedir. Bizim yüksekliğimiz her an değişeceği için böyle pratik bir çözüme gidilmiştir.

$$P_1 = m_1 * g * T01c(3,4);\ \ P_2 = m_2 * g * T02c(3,4);\ P_3 = m_3 * g * T03c(3,4) \quad \textbf{(4.19)}$$

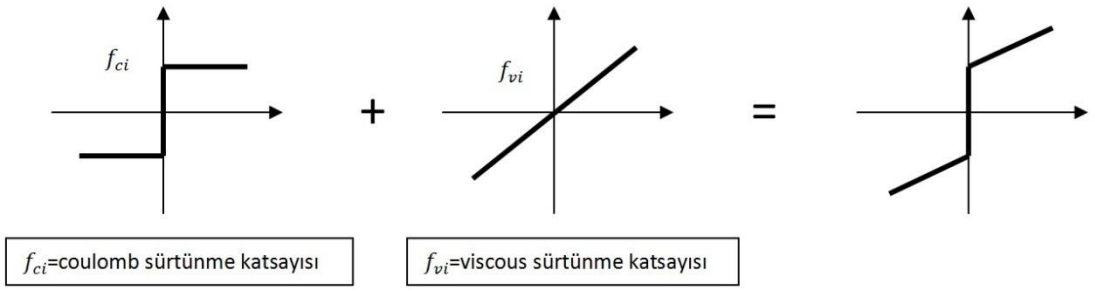
4.4.6 Sürtünme kuvveti

Sürtünme kuvveti robot sistemine iki başlık altında eklenmiştir. Bunlar viscous sürtünme ve coulomb sürtünmedir. Viscous sürtünme hıza bağlı olarak değişen bir kuvvettir. Coulomb sürtünme ise hızın artış veya eksilişine bağlı olarak anlık değişen bir yapıya sahiptir.

$$F(\dot{q}) = F_v\dot{q} + F_c * sgn(\dot{q}) \quad \textbf{(4.20)}$$

$$sgn(x) = \begin{cases} +1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases} \quad \textbf{(4.21)}$$

Viscous ve coulomb sürtünmelerinin birbirlerine etkilerini daha iyi anlayabilmek için aşağıdaki şekile bakılabilir.



Şekil 4.10 : Sürtünme katsayıları.

Şekilde de görüldüğü üzere coulomb friction ivmenin negatif veya pozitif olmasına bağlı olarak -1 veya +1 değerlerini almaktadır.

4.5 Matlab Model

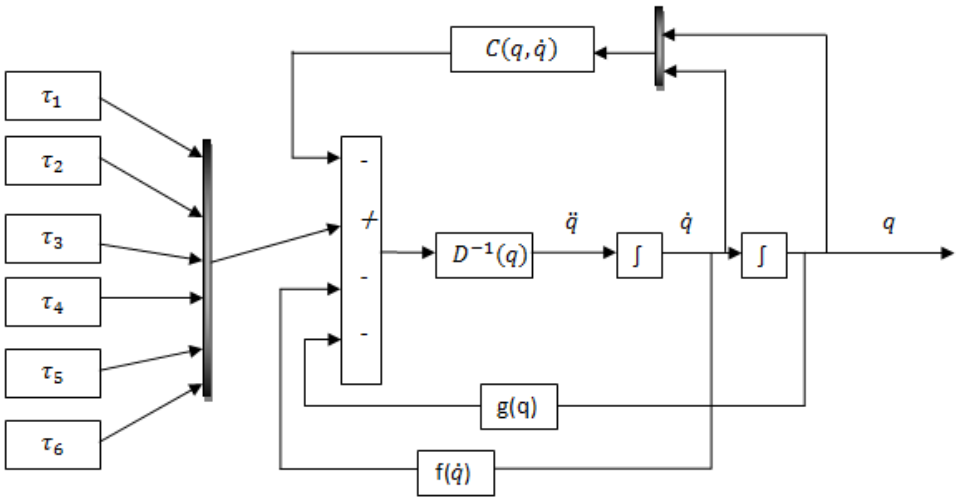
4.19 nolu denklemde ifade edildiği gibi kütle matrisinin 2. dereceden türevi christofel sembollerinin 1. dereceden türevi, yerçekimi kuvveti ve eğer eklenmek isterse sürtünme kuvvetlerinin eklenmesi ile motorların üzerinde ihtiyaç olunan torq miktarları bulunabilir. Ancak bizim robotun hareket mekanızmasını kurarken dikkat edeceğimiz nokta bu torkların sebep olacağı hareketlerdir. Bu sebeple 4.19 nolu denklemden hareketle 4.22'deki gibi bir denklem yazılabilir.

$$D(q) * \ddot{q} = \tau - C(q, \dot{q}) * \dot{q} - g(q) - f(\dot{q}) \qquad \textbf{(4.22)}$$

Matrislerin temel özelliklerini dikkate alarak bu denklemi düzenlersek 4.23'teki denkleme ulaşırız.

$$\ddot{q} = D^{-1}(q) * [\tau - C(q, \dot{q}) * \dot{q} - g(q) - f(\dot{q})] \qquad \textbf{(4.23)}$$

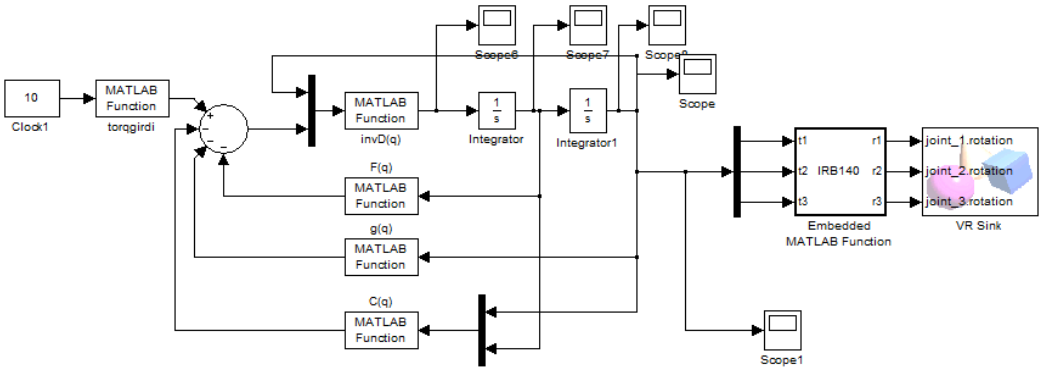
Bu denklemde $\ddot{q}$ eklemlerin 2. dereceden türevlerini ifade etmekte ve integral yardımıyla bu ivme ifadesi konum ifadelerine çevrilebilmektedir. Bu sistemi çalıştırabileceğimiz bir model kurmak istersek şekil 4.11'deki gibi bir model kurmamız mümkün olabilir.



Şekil 4.11 : Dinamik model şeması.

Bu modelde $\ddot{q}$ teriminin görüldüğü noktaya kadar olan kısımda 4.23 nolu denklem halledilmiş demektir. Bundan sonra integral yardımıyla ivme ifadeleri konum ifadeleri haline getirilmiş ve simulasyona gönderilmiştir.

Burada dikkat edilmesi gerekn nokta sürtünme kuvveti için hız hareketten kaynaklanan corialis kuvvetleri için ise konum ve hızın birlikte kullanıldığı potansiyel enerjide ise sadece konum bilgilerinin kulanılmış olmasıdır. Bu şemaya uygun olarak simulink yardımıyla hazırlanan robotumuzun dinamik modeli aşağıdaki şekilde gösterilmiştir.



Şekil 4.12 : IRB 140 robotunun dinamik modeli.

Bu modelde görüldüğü gibi 10sn lik bir zaman diliminde "torkgirdi" bloğundan istenilen torkların girmesi ile başlayan sistemde öncelikle kütle matrisi hesabı

yapılmaktadır. Kütle matrisi bloğunda zamana bağlı olarak 6 giriş bulunmaktadır. Bunlar $\theta_1, \theta_2, \theta_3, \tau_1, \tau_2, \tau_3$ girişleridir. Şekil 4.11'de gösterilenin aksine Tork girişlerinin 3 tane olması bizim 3 eksen için işlem yapıyor olmamızdır. Dolayısıyla bize 3 eklem noktası için tork hesabı gerekmektedir.

Ek B.3'te görüldüğü gibi ilk 3'ü konum ve son 3'ü tork hesapları olmak üzere 6 girdili sistemde kütle matrisinin tersi alınmış ve ve tork matrisi ile çarpılarak ivme matrisi hesaplanmıştır.

İvme matrisi integrali alınarak hız ve konum değerleri haline getirilmiştir. Coriolis kuvvetleri hesabı için konum ve hız değerlirinden alınan 3'er girdi değeri C fonksiyonuna girdi olarak verilmiştir.

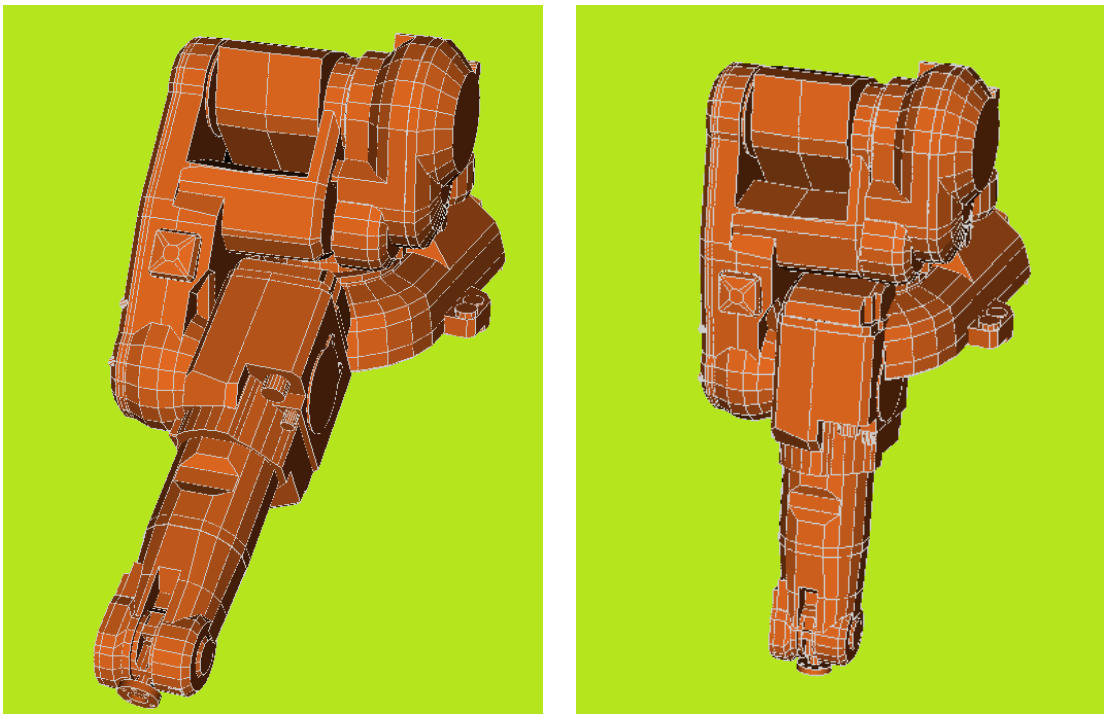
Ek B.4'te görüleceği üzere ilk 3 girdi ile Christofel sembolleri hesap edilmiş ve bu semboller hızlarla çarpılarak yeni Corialis kuvvetleri hesaplanmıştır.

Yer çekimi kuvvetlerinin hesabında da modelde görüleceği üzere sadece 3 eksenin anlık değerleri alınmış ve daha önce bulunan yerçekimi matrisinin içine eklenerek anlık olarak potansiyel enerji hesabına gidilmiştir. Bu işlemin aynısı hız değerleri kullanılmak suretiyle sürtünme içinde kullanılmıştır.

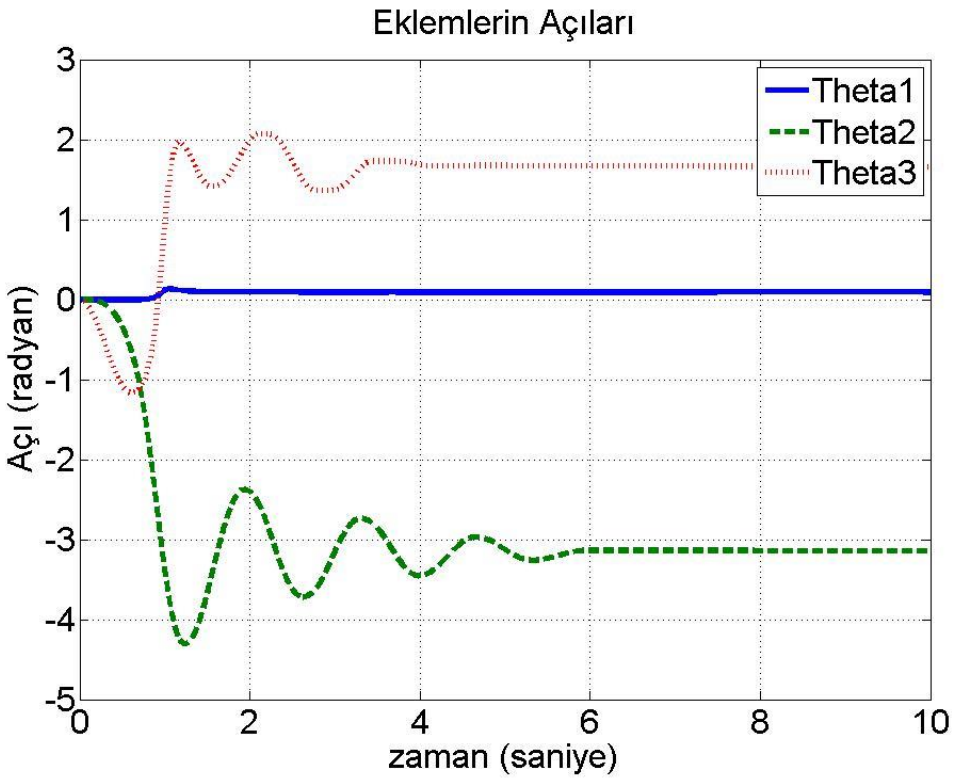
4.6 Modelin Sonuçları

Bizim modelimizde tork girdisini sıfır olarak kabul ettiğimizde yerçekimine maruz kalan bir cismin hareketi gibi robotumuzun alt tabanı sabit olmak üzere üst eklemleri yere düşmek zorundadır. Simulasyonumuzu çalıştırdığımızda bu etkiyi görmekteyiz. Aşağıdaki şekilde 2.sn ve 5.sn'deki durumlarını göstermektedir. Ayrica Şekil 4.14'ten 10 saniyelik zaman diliminde θ değerlerini görmek mümkündür. Bu şekil üzerinden robotun eklemlerinin hareketleri anlaşılabilmektedir.

Şekil 4.14'ten görülebildiği üzere robotumuzun eksenleri ilk 4 sn içinde yerçekiminin etkisi ile kararlı hale gelmiştir. Hiç bir kuvvet uygulamadığımızda yerçekiminin etkisiyle düşmeye başlayan sistem belli bir salınımdan sonra durmakta ve şekil 4.13'ta görüldüğü gibi durmaktadır. Şekil 4.14'te mavi çizgi ile gösterilen kısım 1. Eksen olup yatay eksende etkili olduğundan düşey eksenlerdeki bu düşme hareketinden minimum derecede etkilenmiştir. Birinci eklemin aksine 2. ve 3. eklemler ciddi bir salınım içine girmişlerdir.



Şekil 4.13 : Soldan sağa 2. saniye ve 5. saniyede robotun durumları.

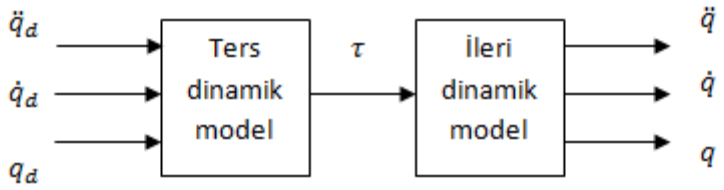


Şekil 4.14 : Dinamik model sonuçları.

5. DİNAMİK KONTROL

Bir dinamik modelde bizden beklenen giriş tork değerlerine göre çıkış konumlarının ne olacağıdır, yani motorların uyguladıkları kuvvetlere göre ne gibi sonuçlarla karşılaşçağımızı bilmek isteriz. Ancak güncel yaşamda anlık motor kuvvetini hesap ederek iş yapılmaz. Bizden beklenen robota belli bir yörünge takibi yaptırılması, bu yörünge takibinde gerekli tork hesabının robot sistemi tarafından hesaplanıp motorlar aracılığı ile sağlanmasıdır. Bizim motorlardan aldığımız bu torkların robotumuzun uç organ konumunu nerelere götürdüğünü hesap etmek ve ne gibi bozuntular içerdiğini belirlemek zorundayız.

İlk ifadelerimde de kullandığım gibi tork girdilerine karşın robot uc organının uzaydaki konumunu hesap ettiğimiz sistem ileri dinamik modeli içermektedir. Biz ilk olarak hedef yörünge verip bunun için gerekli tork kuvvetlerini istediğimizden bunu ters dinamik model olarak ifade edebiliriz. Bu durumu şematik olarak göstermek istersek Şekil 5.1'deki gibi bir şemayı kullanabiliriz.



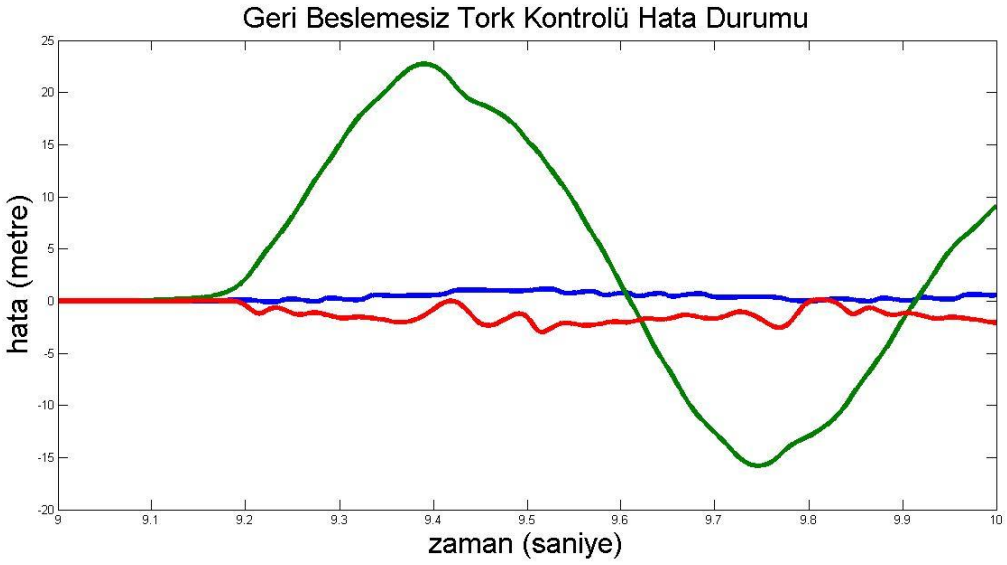
Şekil 5.1 : Hesaplanmış tork modeli.

Bu şemada q_d ile gösterilen kısımlar bizim istediğimiz görüngeyi ifade etmektedir. İstediğimiz görüngeyi robotun takip edebilmesi için gerekli olan tork kuvvetleri ters kinematik model ile hesap edilmekte ve motorlara iletilmektedir. Ardından bu kuvvetlerin çıkarqacağı sonuçta hesaplanıp tekrar geri besleme ile sisteme gönderilerek sürekli kontrol altında olan bir sistem geliştirilebilir. Bunun için öncelikle ters dinamik modelin matematiksel altyapısını anlamaya çalışalım.

Ters dinamik model 4.23 nolu denklemde de gürüldüğü gibi bir yapıya sahiptir. Bu denklemin bloklar halinde modele eklenmesi işlemi için Şekil 5.2'deki gibi bir model oluşturulabilir.

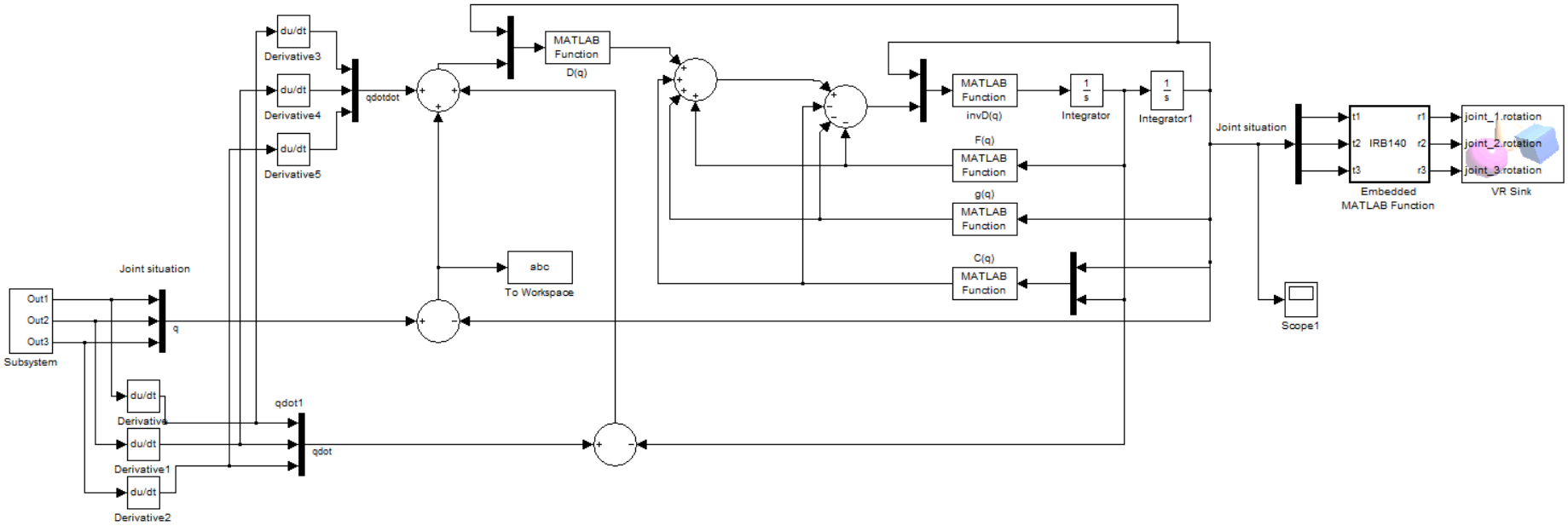
Bu modelde Şekil 2.11'de ki sisteme benzer şekilde bir yörünge modeli oluşturulmuş ve sistemin bu yörüngeyi takip etmesi istenmiştir. Ancak bu sefer dinamik model için uğraştığımızdan yörünge üzerinden alınan verilerin 1. dereceden ve 2. dereceden türevleri alınarak ileri tork hesabı yapılmış ve bu şekilde her bir eklem için gerekli olacak olan tork değerleri hesaplanmıştır. Bu tork hesabının bulunmasının ardından Şekil 4.12'de ki sistem devreye girmekte ve uç organın hareketleri hesap edilebilmektedir.

Bu sistemi ilk olarak geri beslemesiz bir şekilde ele aldığımızda Şekil 3.6'daki gibi bir hata durumuyla karşılaşırız.



Şekil 5.2 : Geri beslemesiz tork kontrolü hata durumu.

Hataların metre cinsinden olduğunu göz önünde bulundurduğumuzda hatanın ne derece absürd olduğunu anlamak mümkün olacaktır. Bu nedenle aşağıdaki gibi bir geribeslemeli sistem inşa etmek durumundayız.

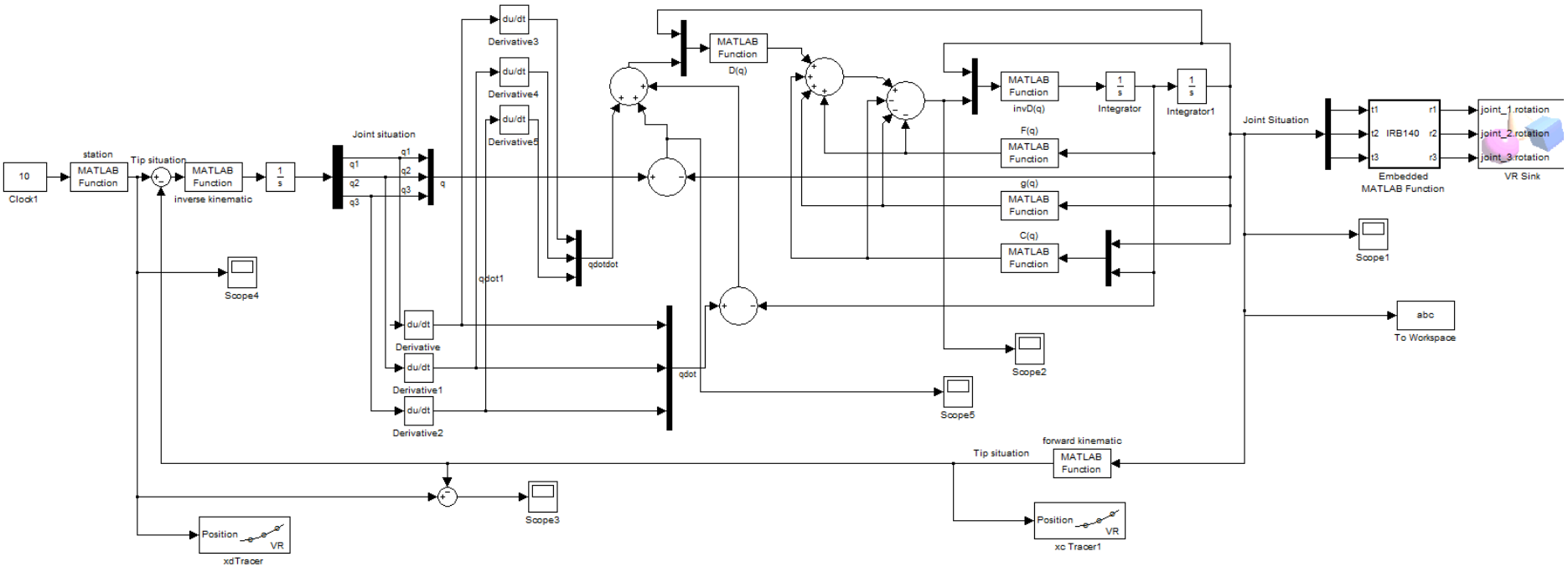


Şekil 5.3 : IRB 140 robotunun geri beslemeli dinamik modeli.

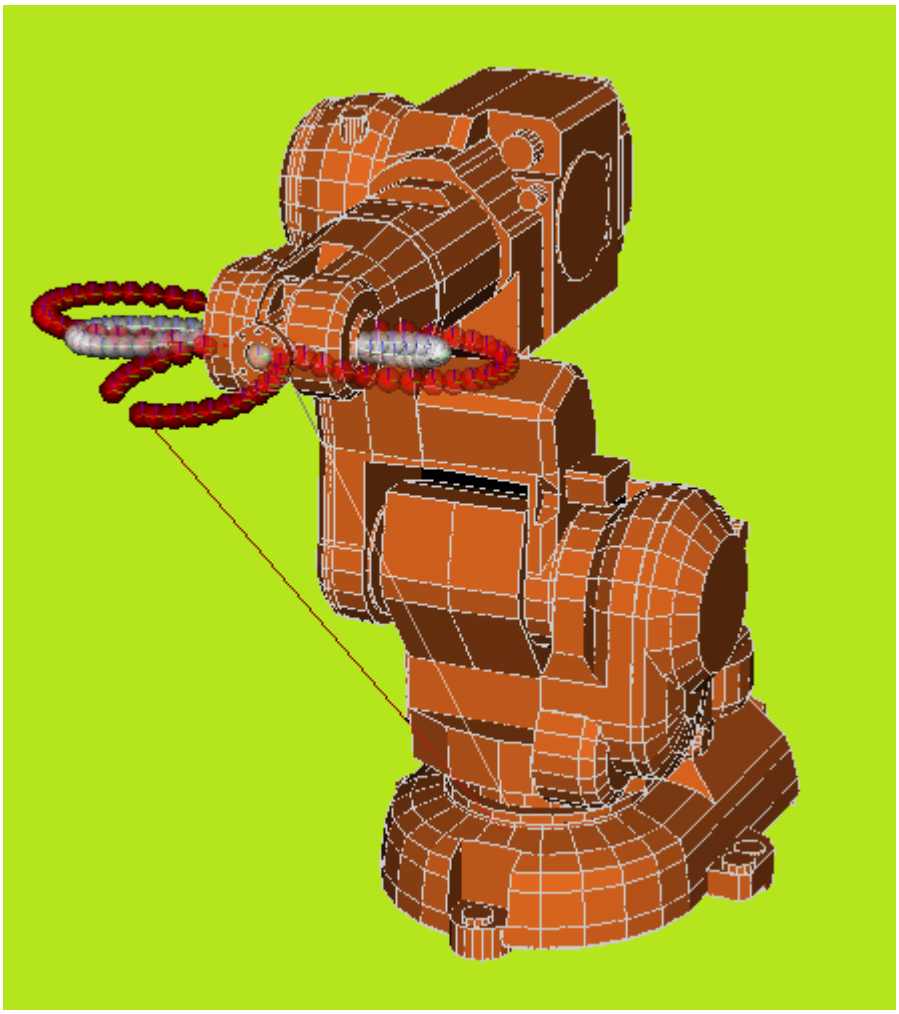
Bu sistemde eklem konumları olarak anlık sinüs dalgaları girilmiş, bu veriler ileri dinamik model ile tork hesabına dönüştürülmüş ve ardından da ters dinamik model ile yeni konum verileri elde edilmiştir. Burada robotun eklemlerinde istediğimiz durumdan robotun eklemlerinin gerçek durumlarını çıkararak hata oranını bulmuş ve bunu sisteme ekleyerek hata oranını düşürmüş durumdayız. Ancak şunu dikkate almalıyız ki robot eklemlerine verilen sinüs dalgalarına bağlı olarak belli ve tahmin edilebilir bir yörüngede ilerlemekte ve istediğimiz yörüngeyi verememekteyiz. Bu sistemi tasarlamak adına aşağıdaki gibi bir tasarım gerçekleştirdik.

Bu sistemde Şekil 4.12'de ki görülen sistem dinamik modelin girişine eklenerek zamana bağlı olarak bir yörünge oluşumu sağlanmıştır. Ardından bu yörünge ters kinematik bloğu sayesinde eklemlerin hızları, integratör sayesinde eklemlerin konumları şeklini almış ve dinamik modele gönderilmiştir.

Bu modelde dikkat edilmesi gereken husus şudur ki animasyon bloğuna gönderilen değerler her bir eklemin konum bilgilerini içermektedir. Karışıklık olmaması için de "joint_situation" şeklinde belirtilen bu değerleri geri besleme yapabilmek için kinematik modelleme kısmından hatırladığımız formüller yardımıyla oluşturduğumuz "forward_kinematic" bloğuna girdi olarak veririz. Bloktan uç organın anlık konumları çıkmakta ve istenilen uç organ konumu ile arasındaki fark hesap edilerek tekrardan ters kinematik denklem bloğuna sokularak her bir eklemin konumları (açısal değerleri) hesap edilmektedir. Bu döngü sürekli olarak devam etmektedir. Bu sayede sisteme istediğimiz şekilde bir yörünge verebilmekte ve hareketlerini takip edebilmekteyiz.



Şekil 5.4 : IRB 140 robotunun yörünge fonksiyonlu dinamik modeli.

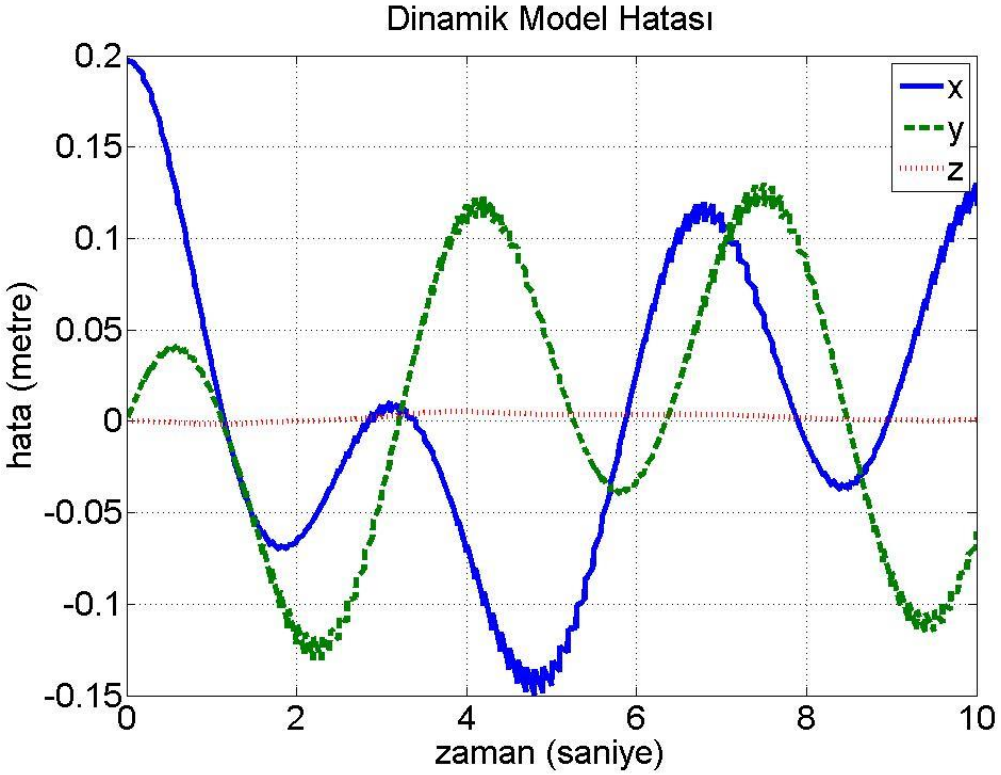


Şekil 5.5 : Geri beslemeli dinamik modeli.

Yukarıdaki şekilde görüldüğü gibi geri beslemeli dinamik modelimiz yörüngeyi belli bir oranda takip edebilmekte ve geri beslemesiz sistemin yaptığı uçuk hareketlerde bulunmamaktadır. Bu sistemde ki hata değerlerini aşağıdaki şekilde incelediğimizde bazı eklemlerde 15 cm'e varan değerler okumaktayız ki bu bir sanayi robotu için yüksek hata oranını ifade etmektedir. Bu sebeple sisteme uygun bir kontrolcü eklemek durumundayız. Ekleyeceğimiz bu kontrolcü için PD-PID, Robust, Uyarlamalı, Bulanık gibi kontrol yöntemleri kullanabiliriz. Bu yöntemlerin herbirinin diğer yöntemlere göre avantajları ve dezavantajları bulunmaktadır. Ancak genel olarak 2 yöntemden bahsedilebilir.

Robust kontrol, sistemin üzerinde oluşabilecek olan bozuntuların göz önünde bulundurularak yapıldığı bir tasarımdır ve bu bozuntuların en kısa sürede sönümlenerek sistemin düzgün bir şekilde çalışmasını hedefler.

Uyarlamalı kontrol, sistemi modellerken kullandığımız temel bilgilerin bile hatalı olduğu durumlarda sistemin kendi kendini kararlı duruma getirerek çalışabildiği bir kontrol modelidir.



Şekil 5.6 : Geri beslemeli dinamik model hatası.

5.1 Lineerleştirme

Robot sisteminin dinamik eşitlikleri aşağıdaki gibidir. Bu denklem yerçekimi ve sürtünme kuvvetleri de baz alınarak oluşturulmuştur.

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) = \tau \qquad \textbf{(5.1)}$$

Ters dinamik sistemde Dinamik Modelleme bölümünde de bahsettiğimiz gibi tork girişi yapılmakta ve konum bilgileri alınmaktadır. Bizim hesaplanmış tork sistemi tasarlamaktaki amacımız istediğimiz konum bilgisini girmek ve sistemden de konum bilgisine ulaşmaktır. Eğer mükemmel ve ideal ortamda çalışan bir robot sistemimiz olsaydı istenilen konum ile üretilen konum eşit olurdu. Bu durumda $\theta(t) = \theta_d(t)$ ve $\dot{\theta}(t) = \dot{\theta}_d(t)$ olurdu ki bunu aşağıdaki denklemle ifade edebiliriz. Biz denklemlerimizde gerek uzunluk ve gerek açısal değişkenler için q ifadesini kullanıyoruz bu yüzden q=θ olarak alınmıştır [36].

$$D(q_d)\ddot{q}_d + C(q_d, \dot{q}_d)\dot{q}_d + G(q_d) + F(\dot{q}_d) = \tau \quad \textbf{(5.2)}$$

Yukarıdaki denklemlerde belirttiğimiz bu durumlar sadece istenilen hedef ile ulaşılan hedefin aynı olması durumu için geçerlidir ancak bu hiçbir zaman garanti edilemez ve hem dış faktörler hem de mekaniksel sebeplerden dolayı neredeyse imkansızdır. Bu sebeple biz sistemimizin içine geri besleme koymak durumundayız ki bizim bu kapalı devremiz istikrarlı olmalıdır. Bu sebeple öncelikle sisteme konum girilerek tork hesabı ve bu tork hesabı girişi ile tekrardan robotun uç noktasının konum bilgilerine ulaşılmıştır. $\ddot{\theta}(t) = \ddot{\theta}_d(t)$ şeklinde yapılan kabul gerçekliği karşılayamayacağından dolayı $\ddot{\theta}(t) = \ddot{\theta}_d(t) - K_v\dot{e} - K_p e$ şeklinde yazılır. Bu şekilde belli bir hata faktörü oluşturulur ve sürekli değişen bu hata değeri ile sistem hem lineerleştirilmiş hem de sıfır hataya indirgenme özelliği kazanmış olur. Bu durum bir sonraki bölümde ayrıntılarıyla açıklanmıştır [36].

5.2 Hesaplanmış Tork

Hesaplamalı Tork kontrolünü incelemeden sistemin dinamik modellemesini irdelemek durumundayız. 4.22 nolu denklemde görüldüğü gibi sistemimizin ideal durumunda denklemin sol yanında bulunan değerlerin istediğimiz tork kuvvetini vermesi gerekmektedir ancak yukarıdaki şekilde görüldüğü gibi bu mümkün olmamaktadır. Bu sebeple işlem kolaylığı için yerçekimi ve sürtünme kuvvetlerini ihmal ederek denklemi tekrardan yazalım.

$$D(q)\ddot{q} + C(q, \dot{q}) + \tau_d = \tau \quad \textbf{(5.3)}$$

Bu denklemde τ_d sürtünme, yerçekimi, motor, çevre koşulları gibi sebeplerle oluşabilecek olan bozuntuları ifade etmektedir. Bu sistemi çalıştırdığımızda meydana gelen hatayı aşağıdaki denklemde görüldüğü ifade edebiliriz.

$$e(t) = q_d(t) - q(t) \quad \textbf{(5.4)}$$

Bu denklemde q_d istediğimiz eklem hareketlerini ifade etmekte q ise gerçekleşen durumu belirtmektedir. Biz yukarıdaki denklemi 5.3 nolu denklemde yerine koymak istersek 2. dereceden türevlerini almak zorundayız.

$$\dot{e}(t) = \dot{q}_d(t) - \dot{q}(t) \qquad \ddot{e}(t) = \ddot{q}_d(t) - \ddot{q}(t) \qquad \textbf{(5.5)}$$

5.5 nolu denklemleri 5.3 nolu denklemde yerine koyarak hata için genel bir formül üretebiliriz.

$$\ddot{q} = \frac{\tau - C(q, \dot{q}) - \tau_d}{D(q)} \qquad \textbf{(5.6)}$$

$$\ddot{e} = \ddot{q}_d + \frac{C(q, \dot{q}) + \tau_d - \tau}{D(q)} \qquad \textbf{(5.7)}$$

Bu denklemin içinde normal tork denklemimizin yanında bozuntu içeren ve τ_d işaretiyle gösterdiğimiz bir eklenti de mevcuttur. Bu denklemin içinden bu eklentiyi çıkarmak için aşağıdaki denklemde olduğu gibi bir çıkarımda bulunabiliriz.

$$\ddot{e}(t) = u(t) + w(t) \qquad \textbf{(5.8)}$$

Bu durumda u (kontrol fonksiyonu) ve w (bozuntu fonksiyonu) aşağıdaki denklemde görüldüğü gibi olacaktır.

$$u(t) = \ddot{q}_d(t) + \frac{C(q, \dot{q}) - \tau}{D(q)} \qquad w(t) = \frac{\tau_d}{D(q)} \qquad \textbf{(5.9)}$$

Bizim sistemimizdeki irdeleyeceğimiz kontrol fonksiyonu (u) olacaktır. Bu sebeple kontrol fonksiyonuna bağlı bir tork hesabı oluşturmalıyız. Yukarıdaki denklemden aşağıdaki fonksiyona ulaşılabilir.

$$\tau = C(q, \dot{q}) - D(q)(u - \ddot{q}_d) \qquad \textbf{(5.10)}$$

Bu denklemde u yerine konulup işlem yapıldığında ideal durumda tam bir eşitlik sağladığı görülür. Daha önce de bahsettiğimiz gibi u fonksiyonu ne kadar iyi tasarlanırsa sistemimiz o derece iyi işleyecek demektir ki bunun için PID, robust, uyarlamalı, bulanık mantık gibi çeşitli metodlar mevcuttur.

5.2.1 Hesaplamalı tork kontrol oransal, türevsel katsayı hesabı

PD kontrol uygulamasında bize gerekli olan eklemlerin konumları ve türevleri olan hız verileri olacaktır. PD kontrol fonksiyonunu aşağıda denklemdeki gibi tanımlayabiliriz.

$$u = -K_p e - K_d \dot{e} \tag{5.11}$$

Bu durumda 5.10 nolu denklemde göstermiş olduğumuz tork ve hata denklemleri aşağıdaki denklemlerde ki gibi bir hal alacaktır.

$$\begin{aligned} \tau &= C(q,\dot{q}) - D(q)(-K_p e - K_d \dot{e} - \ddot{q}_d) \\ \tau &= C(q,\dot{q}) + D(q)(K_p e + K_d \dot{e} + \ddot{q}_d) \\ \ddot{e}(t) &= -K_p e - K_d \dot{e} + w(t) \end{aligned} \tag{5.12}$$

Bu denklemden yapılacak bir çıkarımla aşağıdaki denkleme ulaşılabilecektir.

$$w(t) = \ddot{e}(t) + K_d \dot{e} + K_p e \tag{5.13}$$

Bu durumda denklemin karakteristik polinomu aşağıdaki gibi olacaktır.

$$\Delta(s) = s^2 I + K_d s^1 + K_p s^0 \tag{5.14}$$

2. dereceden bir polinomun genel olarak formu aşağıdaki gibi ifade edilmektedir.

$$p(s) = s^2 + 2\xi w_n s + {w_n}^2 \tag{5.15}$$

Bu durumda $K_d = 2\xi w_n$ ve $K_p = {w_n}^2$ olmaktadır. Bu durumda sönümleme oranını ve doğal frekansı bilmek durumundayız.

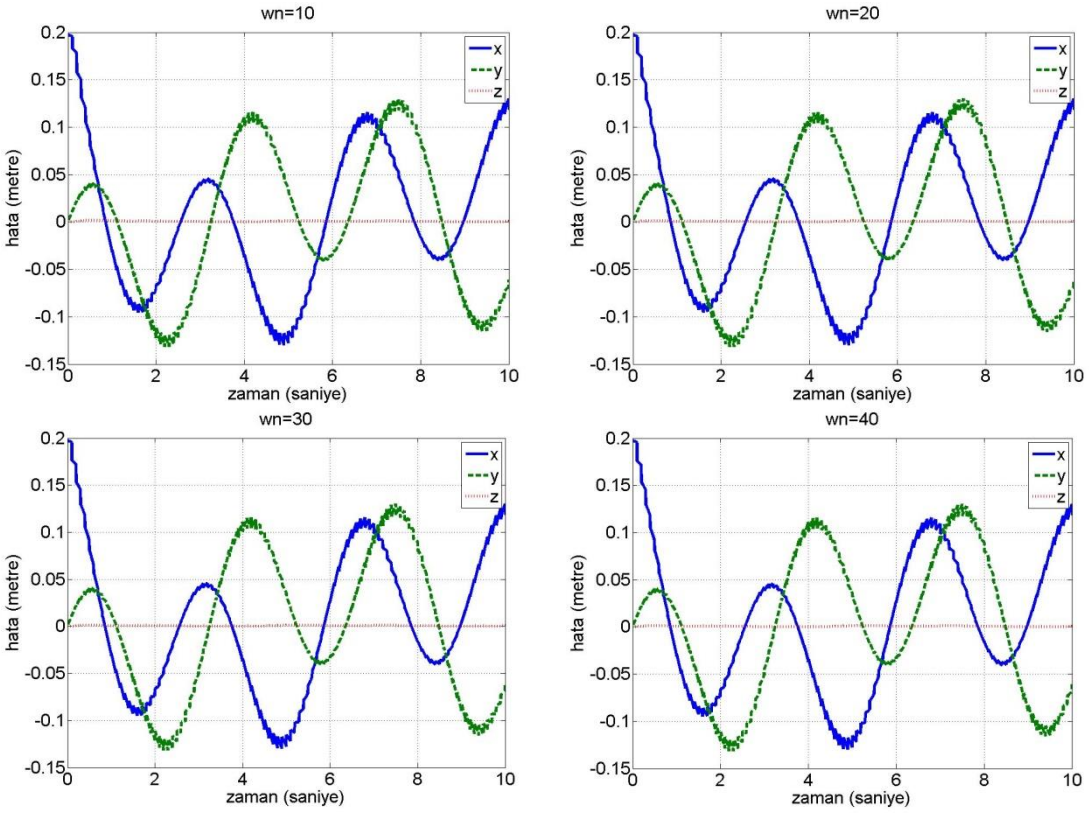
Genel olarak sönümleme oranı PD kontrolcülerde 1 alınmaktadır. Doğal frekans ise yüksek olmalıdır zira hatalara verilen cevap süresi ile ilişkilidir ve yüksek olması demek hataların daha hızlı şekilde minimumu indirilebilmesi demektir.

Doğal frekansın üst limitinin hesaplanması ile alakalı birçok yöntem bulunmakla beraber en genel olarak kullanılan denklem aşağıda belirtilmiştir.

$$w_n = \frac{w_r}{2} \qquad w_r = \sqrt{\frac{k_r}{J}} \tag{5.16}$$

Bu denklemde J ifade ettiği eklemin atalet momentini k_r stifness diye tabir edilen katılık değerini belirtir. J değişken bir yapıya sahip olup en yüksek olduğu durumu ele almak daha mantıklıdır.

Biz doğal frekans değeri için sırasıyla 10,20,30 ve 40 değerlerini almak suretiyle hata oranını denedik ve aşağıdaki şekilde belirtilen sonuçlara ulaştık.

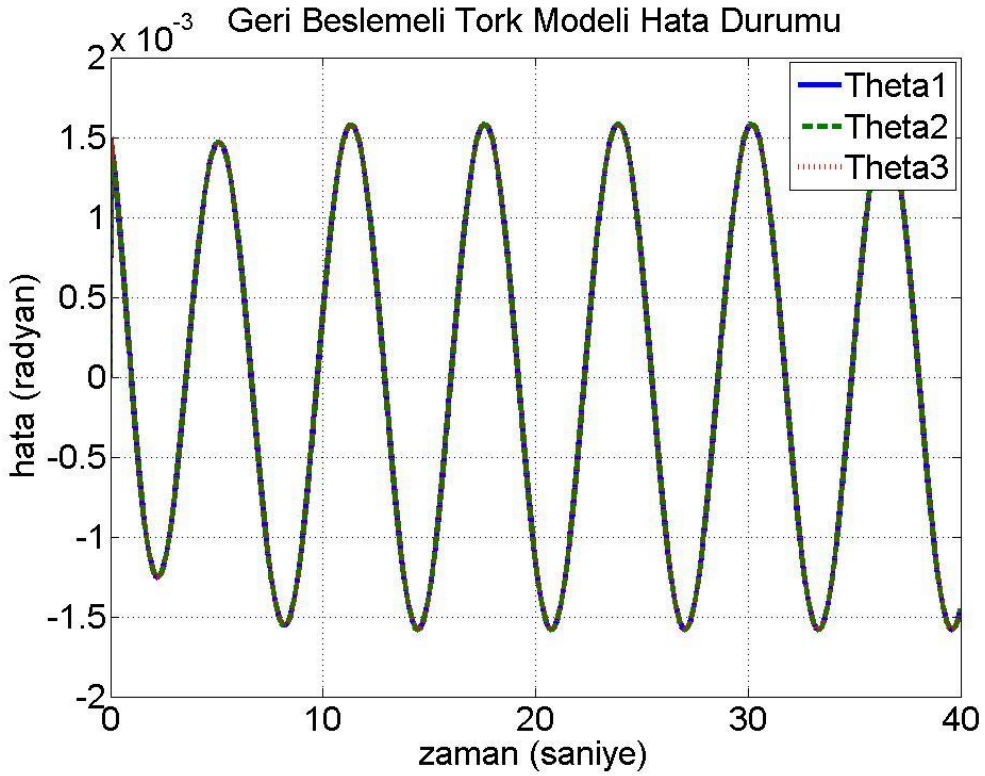


Şekil 5.7 : Doğal frekanslara bağlı dinamik model hataları.

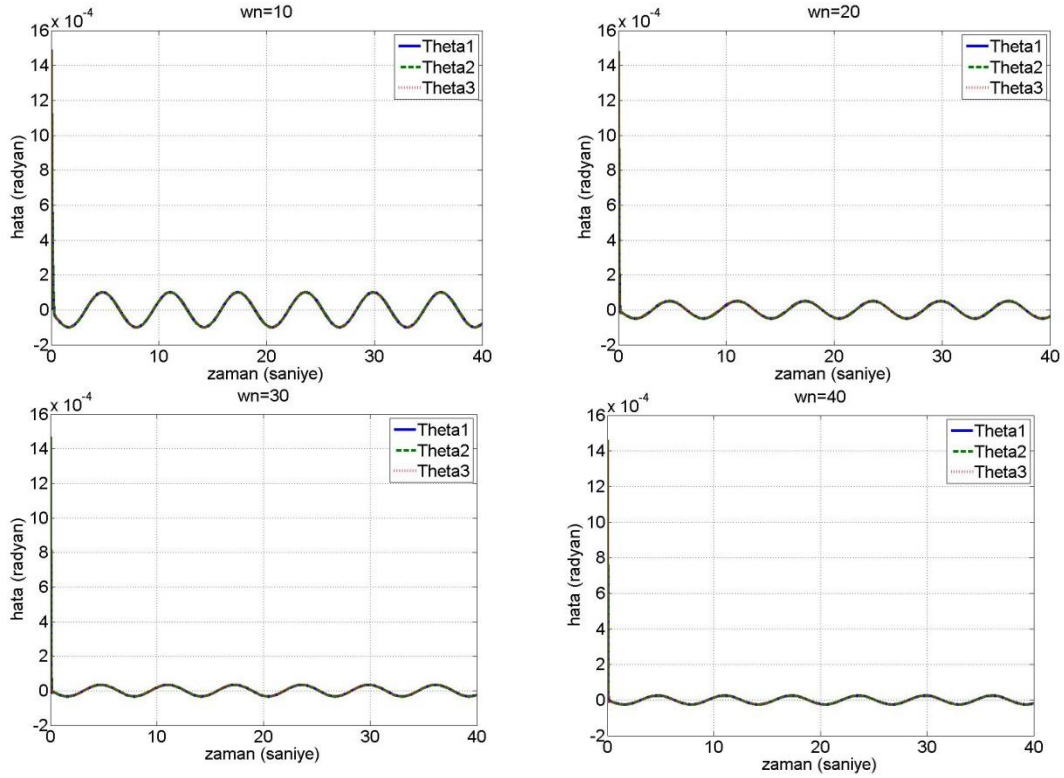
Bizim PD uyguladığımız model sadece tork modeli olmasına karşın sistemimiz yörünge ve tork modelinin birleşimiydi. Biz bu sebeple tork kontrol sistemlerimizi sadece tork kontrol sistemine uygulamaya ve yörünge sistemine ayrı bir bölümde incelemeye karar verdik. Bunun için öncelikle şekil 5.3'te ki sistemi kullandık ve aşağıdaki sonuçlara ulaştık.

Aşağıdaki şekilde görüldüğü üzere hata durumu 0.0015 radyan civarında olup oldukça düşük düzeyde seyretmektedir. Ancak sürekli olmakta ve bir iyileşme

berlitisi göstermemektedir. Bu sebeple bu sisteme PD tork uygulayarak değişik doğal frekans değerlerinde görülen hata durumlarını görmek istedik.



Şekil 5.8 : Geri beslemeli tork modeli hata durumu.

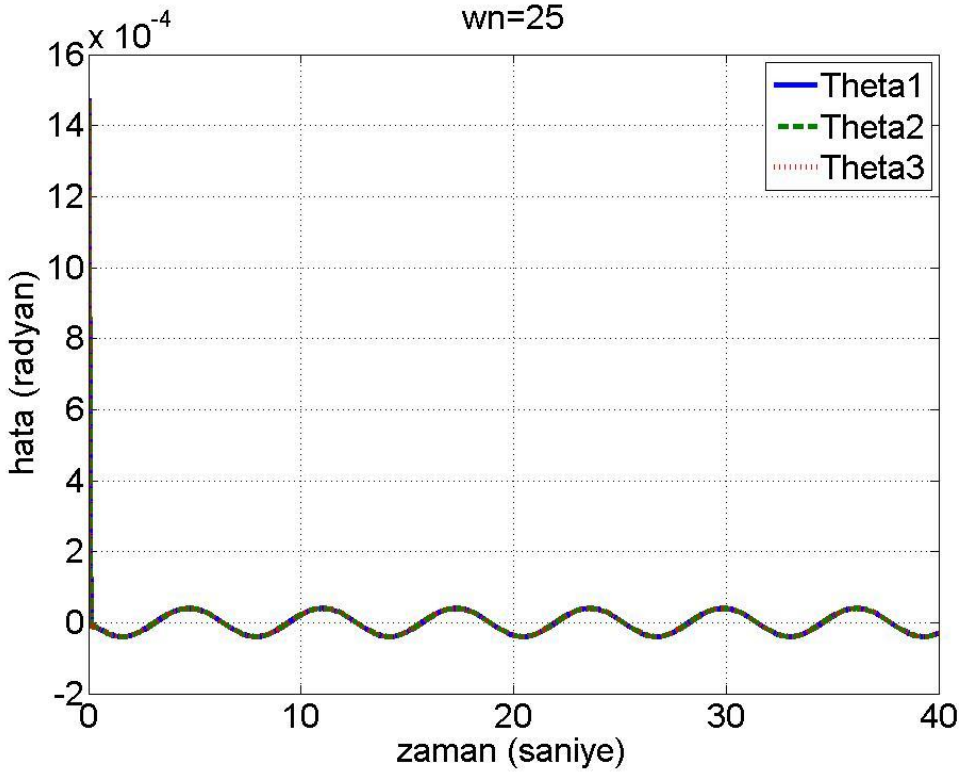


Şekil 5.9 : Doğal frekanslara bağlı PD dinamik model hataları.

Yukarıdaki şekilden çıkartacağımız sonuç PD kontrolcü sisteminde sürekli olarak bir hatanın varlığının olacağıdır. Bu sürekli hal hatasını sıfırlamak PD kontrolcüde mümkün değildir ancak minimum seviyeye indirmek mümkündür. Şekilden de okunabileceği gibi bu sürekli hal hatasını en aza indirebilmek için doğal frekansı yüksek tutmak gerekmektedir. Doğal frekansın olabilecek en yüksek düzeye ulaştırılması gerekmektedir ki bunun için 5.16 nolu denklemi kullanabiliriz.

Bu denklemi kullanabilmek için öncelikle atalet tensörü ve sertlik katsayılarının bulunması gerekmektedir. J değerini 0.012 kgm^2 'dir [22]. Katılık değerini 20 $\frac{m}{N}$ olarak aldık. Bu durumda doğal frekans 20 Hz olarak çıkacaktır.

Genel olarak sanayi robotlarında kullanılan rezonans frekansı genelde 50 Hz olarak görülür [37]. 5.15 nolu denklemden doğal frekans rezonansın yarısı olarak ele alındığına göre bizim doğal frekansımız 25 Hz olacaktır. Bu frekans değerini sisteme eklersek $K_p = 625$ ve $K_d = 50$ olarak ele alınacaktır ve sonuçları şekil 5.10'da görülmektedir.



Şekil 5.10 : PD dinamik model hatası.

PD Dinamik modelinde 0.001 radyan hata ile sistem başlamaktadır. Hata 0.0015 radyana hızlı bir şekilde çıkmakta ve 0.115 sn içinde sıfıra inmekte, ardından da maksimum 4.734'üncü saniyede 0.00004 radyana kadar hata oluşmaktadır. Bu hata

sürekli olarak ±0.00004 radyan olarak devam etmektedir. Bu hata durumu bizim için kabul edilebilir bir hata durumudur. Eğer bu hatayı tamamıyla ortadan kaldırmak istersek PD uygulamasında bu hatayı sıfırlama imkânı söz konusu değildir ancak bu hatayı sıfıra indirmek için PID uygulayabiliriz.

5.2.2 Hesaplanmış tork oransal, integral, türevsel katsayı hesabı

PID kontrolcü tasarımında sisteme hatanın integralini eklemek durumundayız ve bunun için aşağıda denklemdeki gibi bir kabul yaparak ifade edebiliriz.

$$\dot{\varepsilon} = e \tag{5.17}$$

Bu ifadeyi kontrol fonksiyonuna eklediğimizde aşağıdaki denklemi elde ederiz.

$$u = -K_i\varepsilon - K_p e - K_d\dot{e} \tag{5.18}$$

Bu durumda 3.10 denklemde ifade edilne tork denklemleri aşağıdaki gibi bir ifadeye bürünecektir.

$$\tau = C(q,\dot{q}) + D(q)(K_i\varepsilon + K_p e + K_d\dot{e} + \ddot{q}_d) \tag{5.19}$$

Hatanın 2. dereceden türevi daha önceden ifade ettiğimiz gibi kontrol fonksiyonu ile bozuntu fonksiyonunun toplamı olarak ifade edilmektedir. Kontrol fonksiyonumuz artık 5.18 nolu denklemde olduğu gibidir. Bu durumda zamana bağlı olarak hatayı aşağıdaki denklemde olduğu gibi ifade edebiliriz.

$$\ddot{e}(t) = -K_i\varepsilon - K_p e - K_d\dot{e} + w(t) \tag{5.20}$$

Bu denklemi düzenlersek;

$$w(t) = \ddot{e}(t) + K_d\dot{e} + K_p e + K_i\varepsilon \tag{5.21}$$

Bu durumda denklemin karakteristik polinomu aşağıdaki gibi olacaktır.

$$\Delta(s) = s^3 I + K_d s^2 + K_p s^1 + K_i s^0 \tag{5.22}$$

Bu denklemden bir çıkarım yapmak için Routh-Hurwitz stabilite kriterini kullanabiliriz. Routh-Hurwitz kriteri lineer sistemlerin stabilitesinin devamı ve testi için kullanılan bir kriterdir ve biz bu sistemi integral çarpanını seçmek için

kullanacağız. Bu sayede sistemimizin stabilitesini bozmayacak bir integrator çarpanının sınırlarını belirlemiş olacağız.

Routh-Hurwitz kriteri; $\Delta(s) = a_n s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0 = 0$ şeklindeki karakteristik polinomlar üzerinden hesaplanmaktadır. Bir karakteristik polinomun testini yapmak için kullanılacak Routh-Hurwitz kriteri tablosu aşağıdaki gibidir [38].

$$\begin{Bmatrix} s^n \\ s^{n-1} \\ s^{n-2} \\ s^{n-3} \end{Bmatrix} \begin{vmatrix} a_n & a_{n-2} & a_{n-4} \\ a_{n-1} & a_{n-3} & a_{n-5} \\ b_{n-1} & b_{n-3} & b_{n-5} \\ c_{n-1} & c_{n-3} & c_{n-5} \end{vmatrix} \quad \textbf{(5.23)}$$

Bu tablo n sayısı sonsuza gidecek şekilde devam etmektedir. Bu denklemde gösterilen b ve c değerleri aşağıdaki denklemlerle hesaplanabilir.

$$b_{n-1} = \frac{a_{n-1}a_{n-2} - a_n a_{n-3}}{a_{n-1}} = \frac{-1}{a_{n-1}} \begin{bmatrix} a_n & a_{n-2} \\ a_{n-1} & a_{n-3} \end{bmatrix} \quad \textbf{(5.24)}$$

$$b_{n-3} = \frac{-1}{a_{n-1}} \begin{bmatrix} a_n & a_{n-4} \\ a_{n-1} & a_{n-5} \end{bmatrix} \quad \textbf{(5.25)}$$

$$c_{n-1} = \frac{-1}{b_{n-1}} \begin{bmatrix} a_{n-1} & a_{n-3} \\ b_{n-1} & b_{n-3} \end{bmatrix} \quad \textbf{(5.26)}$$

Yukarıdaki denklem gözönüne alındığında bizim 3.20 nolu denklemimiz $q(s) = a_3 s^3 + a_2 s^2 + a_1 s + a_0$ ve $a_3 = 1,\ a_2 = K_d,\ a_1 = K_p,\ a_0 = K_i$ şeklindedir. Buradan çözüme devam edersek;

$$\begin{Bmatrix} s^3 \\ s^2 \\ s^1 \\ s^0 \end{Bmatrix} \begin{vmatrix} a_3 & a_1 \\ a_2 & a_0 \\ b_1 & 0 \\ c_1 & 0 \end{vmatrix} = \begin{Bmatrix} s^3 \\ s^2 \\ s^1 \\ s^0 \end{Bmatrix} \begin{vmatrix} 1 & K_p \\ K_d & K_i \\ b_1 & 0 \\ c_1 & 0 \end{vmatrix}$$

$$b_1 = \frac{a_2 a_1 - a_0 a_3}{a_2} \text{ ve } c_1 = \frac{b_1 a_0}{b_1} = a_0$$

Routh-Hurwitz kriterinde sistemin kararlı olması için gerekli koşul ilk sütunun pozitif ve sıfırdan farklı olmasıdır. Bu durumda bizim b1 ve c1 değerlerimizin pozitif olması gerekir. Bu durumda a_0 değeri yani integral çarpanı 0 olamaz ayrıca bu

şartlardan $a_2a_1 - a_0a_3 > 0$ eşitliği çıkar. Bu eşitlikteki terimler yerine bizim kontrol çarpanlarını koyarsak aşağıdaki sonuç çıkar.

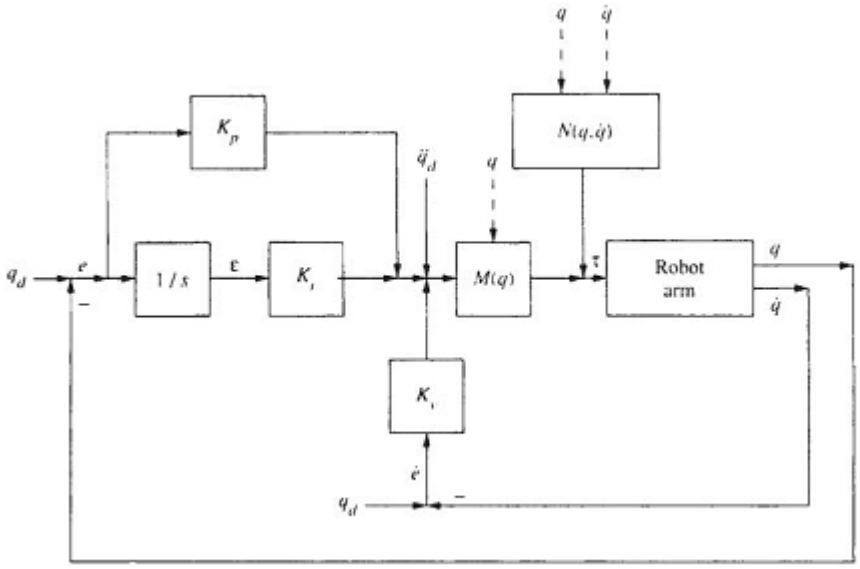
$$K_dK_p > K_i \quad \textbf{(5.27)}$$

Yukarıdaki denklemde görüldüğü gibi integratör çarpanı sıfırdan büyük ve türev ile oransal katsayıların çarpımından küçük olmak zorundadır ki aşağıdaki gibi ifade edebiliriz.

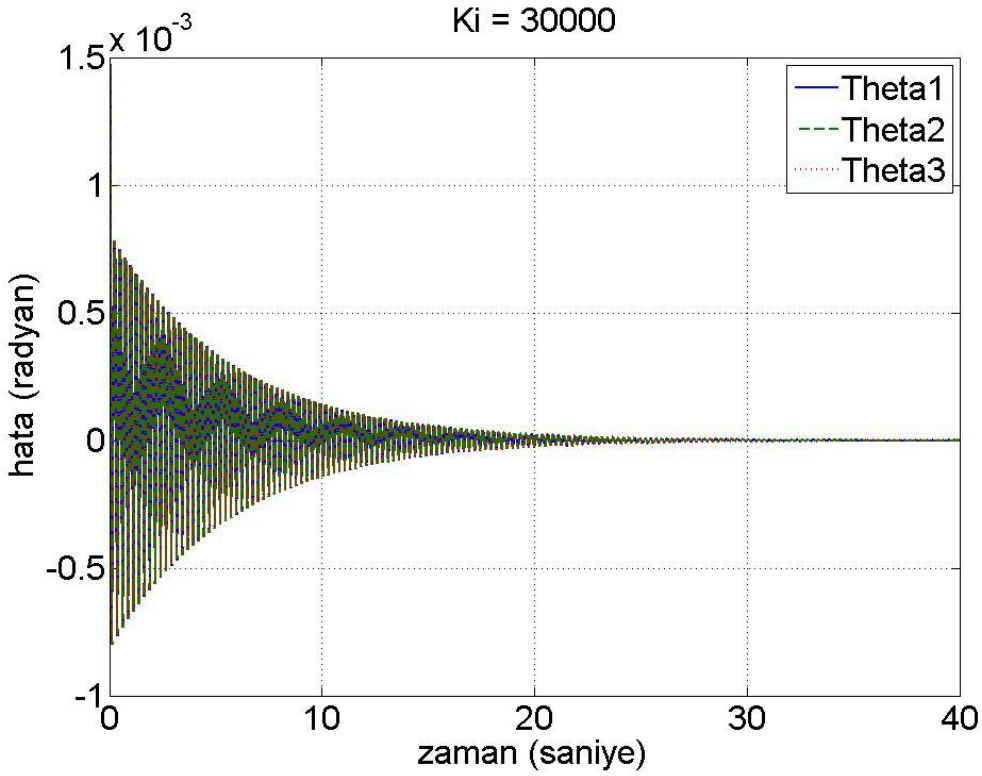
$$50 * 625 > K_i > 0 \quad \textbf{(5.28)}$$

Bu durumda integrator çarpanı 31250'den küçük olmak durumundadır ve ilk olarak ki=30000 olarak kabul edip inceledik.

PID kontrol sistemi; için öncelikle şekil 5.11'deki gibi bir sistem tasarladık. Bu işlemde dikkat edilmesi gereken nokta integrator çarpanı girişinin konum hatasından alınmasıdır zira bizim bütün kontrol sistemi hatalar üzerinden işlem yapmaktadır.

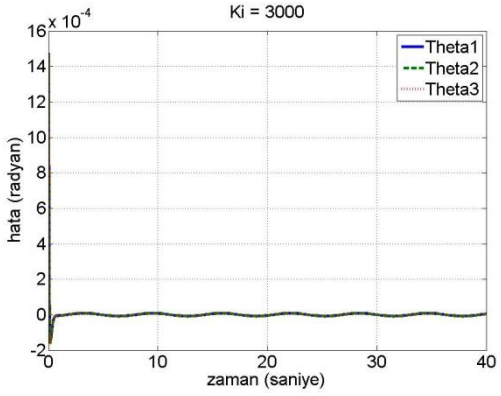


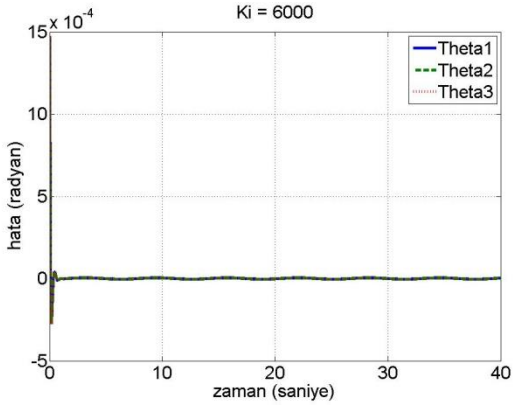
Şekil 5.11 : PID dinamik model [34].

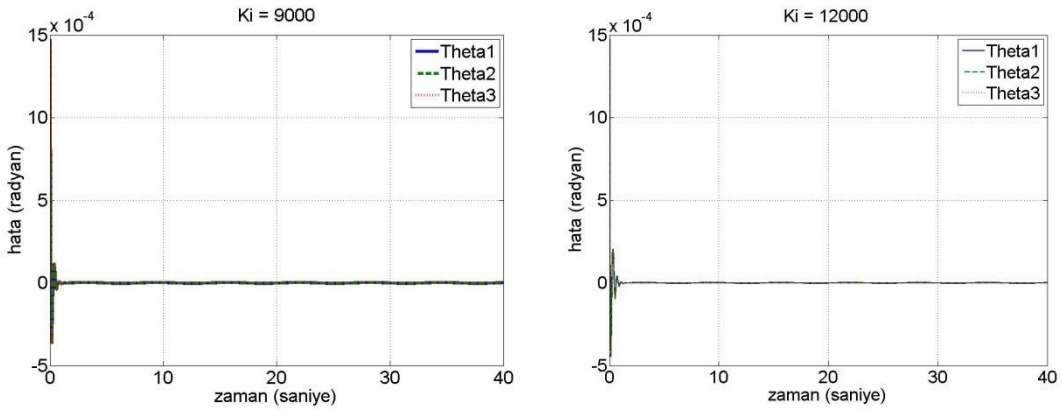


Şekil 5.12 : PID tork kontrolü model hatası.

Yukarıdaki şekilde görüldüğü gibi şekil 5.8'de görülen 0.0015 radyanlık hata yine aynı düzeyde başlamakta ama PID kontrolcüde zamanla azalan bir grafik çizmektedir. Halbuki Şekil 5.10'u incelediğimizde 0.00004 radyan mertebesinde neredeyse yok denebilecek düzeyde hata oluşmaktadır. Bizim PID sistemimizinde kısa sürede bu hata oranına ulaşıp tamamıyla sönümlemesini beklemekteyiz. Bu sebeple integratör çarpanlarını 3000,6000,9000 ve 12000 olarak ele alıp teker teker inceledik ve Şekil 5.13'te ki gibi sonuçlar çıktı.







Şekil 5.13 : İntegrator çarpanına bağlı tork kontrolü model hatası.

Bu grafikleri genel olarak incelersek aşağıdaki sonuçlara ulaşırız.

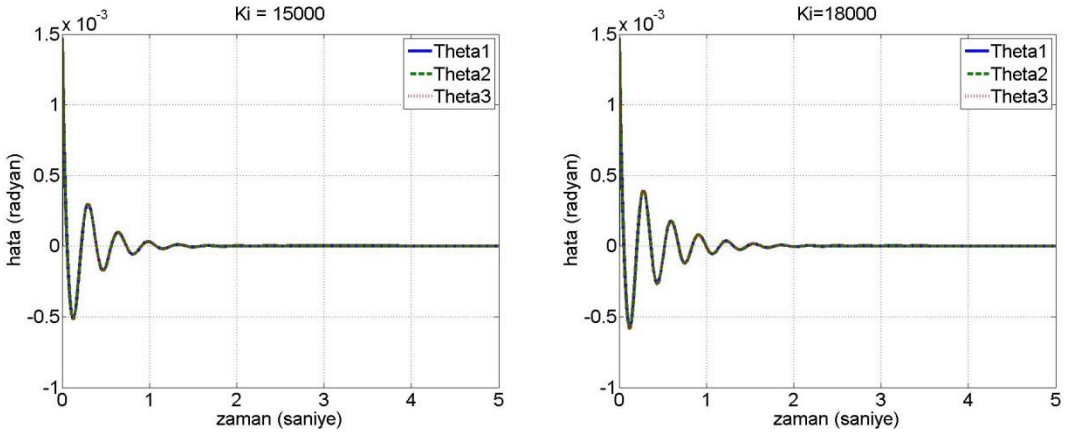
Ki=3000 için sistem 1.72 saniyede sıfırdan başlayacak şekilde bir sürekli hal hatası sirkülasyonuna girmiştir. Sürekli hal hatası 3.2'inci saniyede en yüksek noktaya ulaşmaktadır ve $\pm 8.3059 * 10^{-6}$ radyandır. Bu değer şekil 5.10'da gösterilen $\pm 4 * 10^{-5}$ radyanlık sürekli hal hatasının yanında oldukça iyi bir performanstır.

Ki=6000 için sistem 1.609 saniyede sıfırdan başlayacak şekilde bir sürekli hal hatası sirkülasyonuna girmiştir. Bu durum Ki=3000'den daha iyidir. Sürekli hal hatası 3.183'üncü saniyede en yüksek noktaya ulaşmaktadır ve $\pm 4.186 * 10^{-6}$ radyandır. Bu değer integrator çarpanının 3000 olduğu durumdan daha iyi sonuç vermiştir.

Ki=9000 için sistem 1.205 saniyede sıfırdan başlayacak şekilde bir sürekli hal hatası sirkülasyonuna girmiştir. Ayrıca bu saniyeye gelmeden önce de hafif bir sirkülasyona da girildiği görülmektedir yani keskin şekilde sıfıra yaklaşmamış hafif bir sirkülasyonla sıfıra yaklaşmıştır. Bu durum Ki=6000'den daha iyidir. Sürekli hal hatası 3.152'inci saniyede en yüksek noktaya ulaşmaktadır ve $\pm 2.7915 * 10^{-6}$ radyandır. Bu değer integratör çarpanını 6000 olduğu durumdan 2 kat daha az hatayı ifade etmektedir.

Ki=12000 için 1.189 saniyede sıfırdan başlayacak şekilde sürkli bir hal hatası sirkülasyonuna girmiştir. Bu sürekli hal hatasına girmeden önceki durumda da belli bir sirkülasyona sahiptir ve çok düşük hatalar içermektedir. Sürekli hal hatası 1.472'inci saniyede en üst noktaya ulaşmaktadır ve $\pm 1.7831 * 10^{-6}$ radyandır. Bu değerler oldukça iyi düzeydedir.

Bu durumda integral çarpanı yükseldikçe sonuç daha iyi bir duruma gitmektedir. Ancak Şekil 5.14'te görüleceği üzere oturma zamanı artmaktadır. Bu nedenle Ki=12000 değerini PID için en uygun değer olarak kabul edebiliriz.

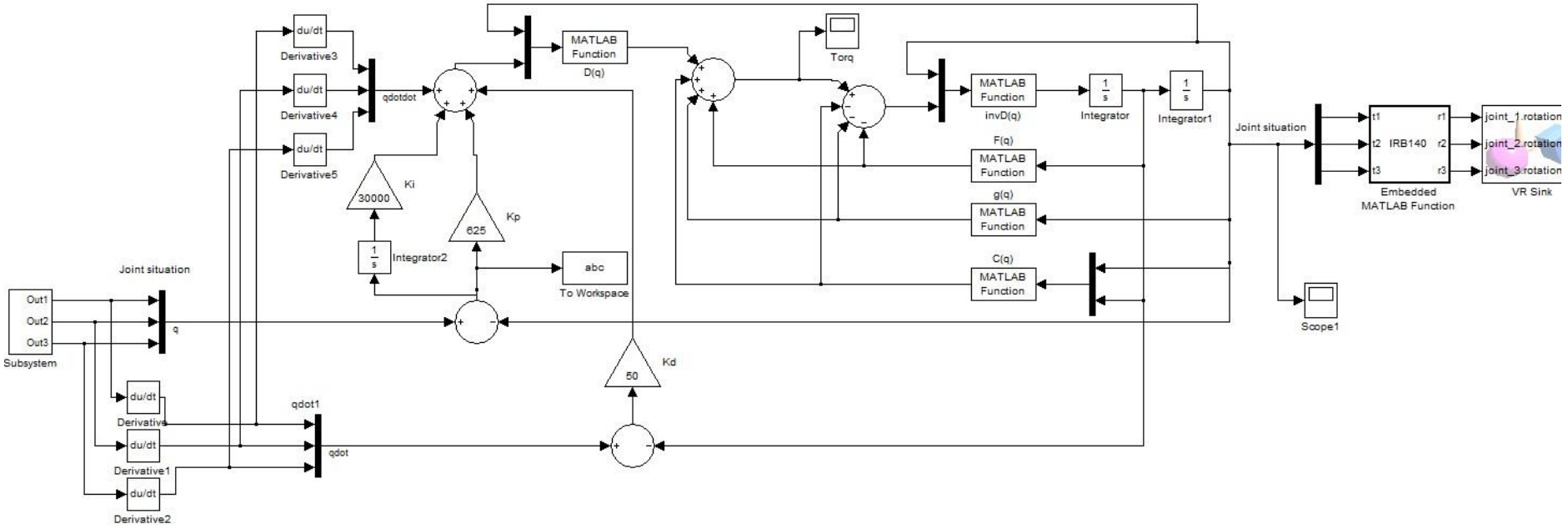


Şekil 5.14 : Ki = 15000, 18000 için tork kontrollü model hatası.

PID kullanımında hataların sıfıra yaklaştığı görülmekle beraber PID kullanımı fiziksel sistemlerde birçok probelmleri de beraberinde getirmektedir. Örnek olarak motorların üretebileceği tork miktarını verebiliriz. Bu problem PID sistemlerde PD sistemlere nazaran daha yüksek oranlarda görülmektedir ve ilerleyen bölümlerde bu durumlardan bahsedilecektir [34].

Tork kontrolü hesabında doğal frekansın yüksek olması sebebiyle kazanç katsayıları da yüksek olmuştu. Bu katsayıların yüksek olması sebebiyle bu sistemi ayrıca ziegler-Nichols yöntemi ile de incelemeye karar verdik.

Ziegler-Nichols Yöntemi; PID kontrolör tasarımlarında kullanılan genel bir yöntemdir ve bu yöntemi kullanırken Şekil 5.8'de gösterilen geri beslemeli tork modeli üzerinde çalışılacaktır. Bunun için öncelikle kazancın ve periyodun bilinmesi icap eder. Şekil 5.8'de kazancımız $1.5 * 10^{-3}$ tür ve periyodumuzda 7 sn'dir. Bu durumda Çizelge5.2'de verilen formüllerin içinde bu değerler yazarsak Çizelge 5.1'deki sonuçlara ulaşırız.

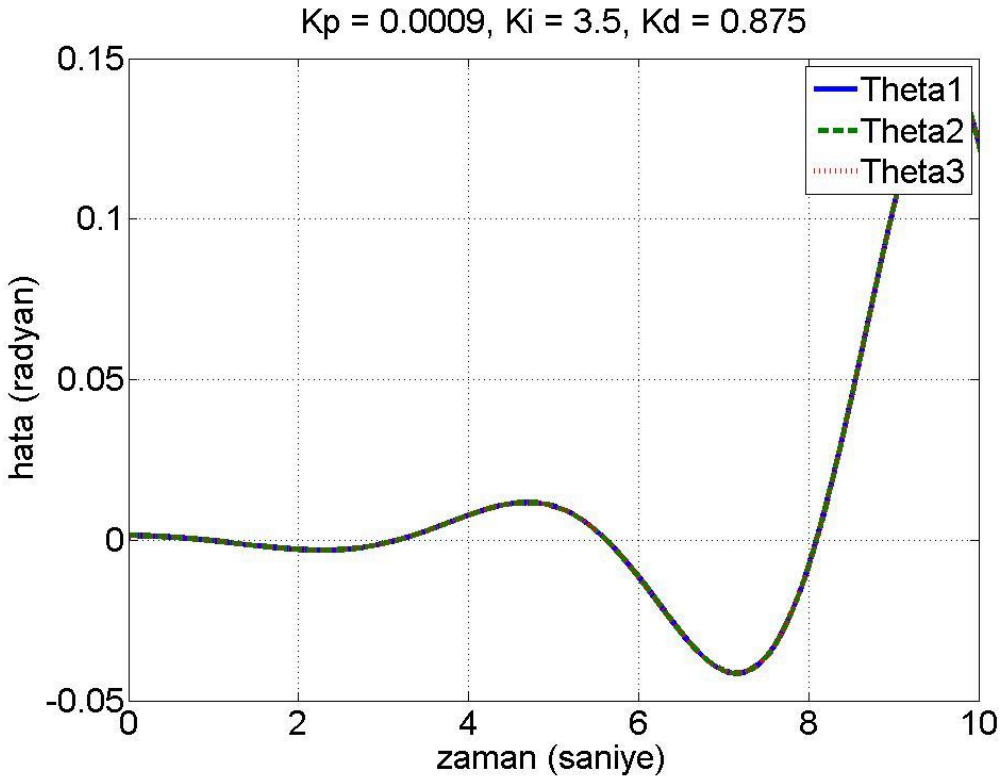


Şekil 5.15 : PID tork kontrol modeli.

Çizelge 5.1 : Ziegler-Nichols yöntemi ile çarpanların tespiti.

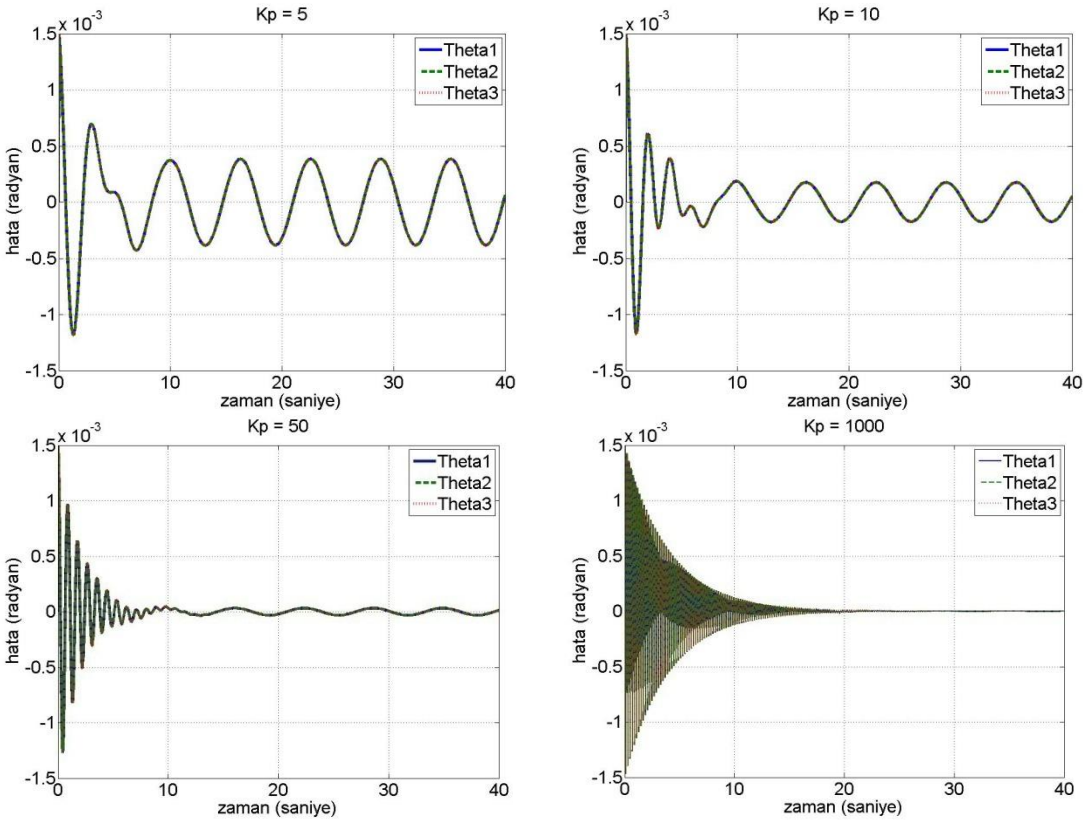
Kontrol	K_p	K_i	K_d
P	$7.5 * 10^{-4}$	∞	0
PI	$6.818 * 10^{-4}$	5.833	0
PID	$9 * 10^{-4}$	3.5	0.875

Bu durumda oluşacak sonuçlar aşağıdaki şekilde verilmiştir ve devamlı kötüye giden sonuçlardır. Bunun sebebi ise Kp tercihini en uygun koşul için yapmamış olmamızdır.

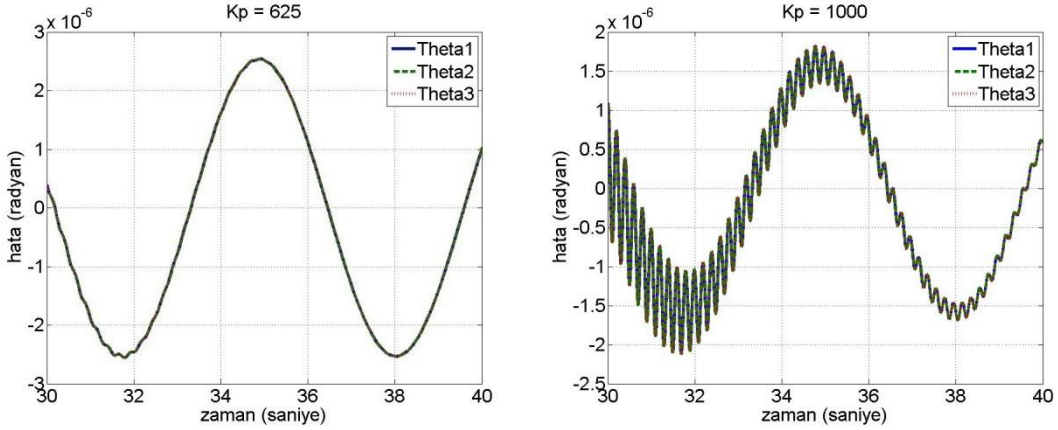


Şekil 5.16 : Kp = 0.0009, Ki = 3.5, Kd = 0.875 için dinamik model hatası.

Oransal çarpanın düşük ve integral çarpanının yüksek olması sistem üzerinde bozucu etki yapmıştır. Oransal kontrol çarpanının nihai değerini bulmak üzere bir kaç deneme yapmayı uygun gördük. Bu amaçla yapılan bazı denemeler aşağıdaki şekilde görülmektedir ki oransal kazanç katsayısı belli bir değerden sonra aşırı derecede sirkülasyona neden olmaktadır. Ancak bu sirkülasyona rağmen sürekli hal hatasının da sürekli olarak düştüğü gözlemlenmiştir. Bu durumda oransal kontrol değeri tercihi yapılırken daimi hal hatasının minimum olduğu ve düzenli olduğu bir değeri almak zorundayız. Şekil 5.18'de oransal çarpanların 625 ve 1000 olduğu değerlerde 30 ve 40. Saniyeler arası incelenmiştir.



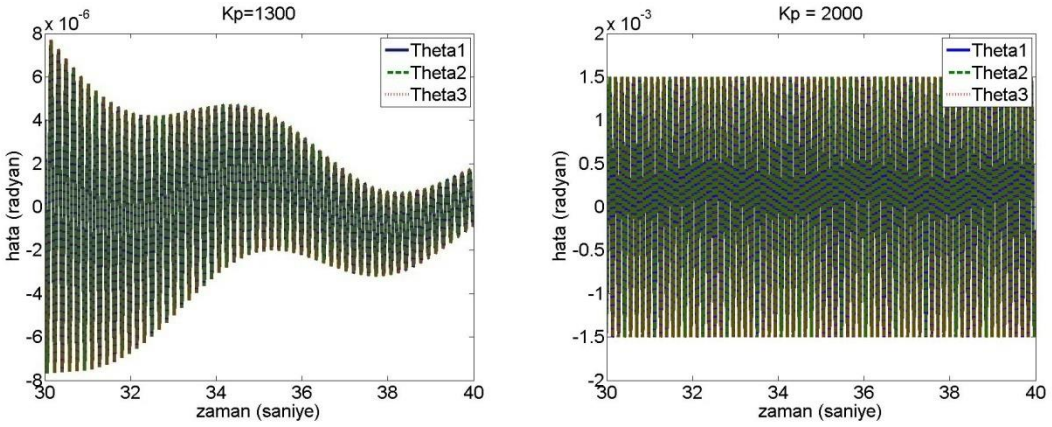
Şekil 5.17 : Kp değerlerine bağlı tork kontrol hataları.



Şekil 5.18 : Kp = 625 ve 1000 için tork kontrol hataları.

Yukarıdaki şekilde görüldüğü gibi oransal kontrol kazancının 625 ve 1000 olduğu değerlerde sürekli hal hataları 10^{-6} mertebelerinde olmaktadır ve eşit düzeydedir.Ancak oransal kazancın 1000 olduğu durumda sirkülasyon hala devam etmektedir. Bu durumda oransal kazancı 625 almak ve Ziegler-Nichol yöntemini bu değere göre kullanmak en uygun seçim olacaktır. Zira bu hata oransal kazancın 100 olduğu durum için 10^{-5} mertebelerindedir.

Daha yüksek oransal kazanç değerlerinde sürekli hal hatası değişime uğramazken sirkülasyon ciddi derecelerde artmaktadır. Bu durumu incelemek için oransal kontrolün 1300 ve 2000 olduğu değerleri aşağıdaki şekilde sunduk.



Şekil 5.19 : Kp = 1300 ve 2000 için tork kontrol hataları.

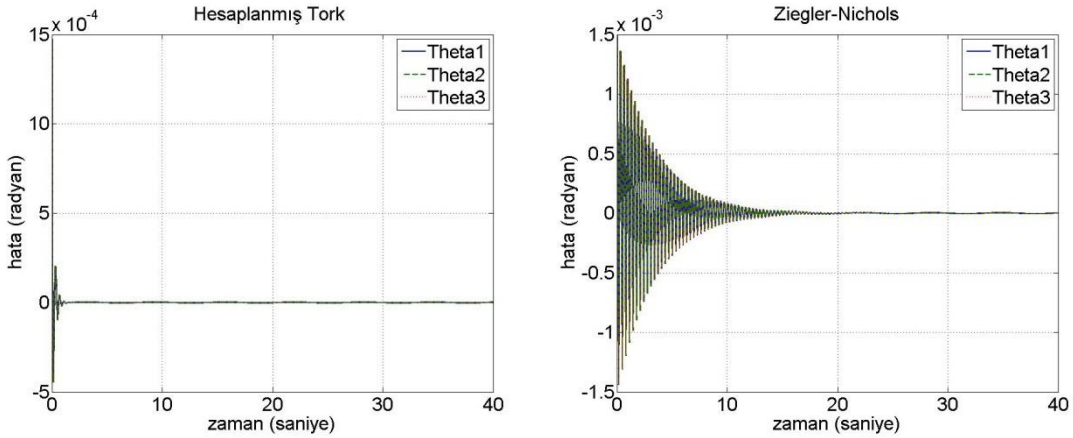
Oransal çarpanın 2000 değerini geçmesinden sonra artık oransal kontrol sisteme fayda yerine zarar verecek ve örnek olarak oransal kazancın 10000 olduğu durumda hatalar 10^{27} radyan mertebesine ulaşacaktır.

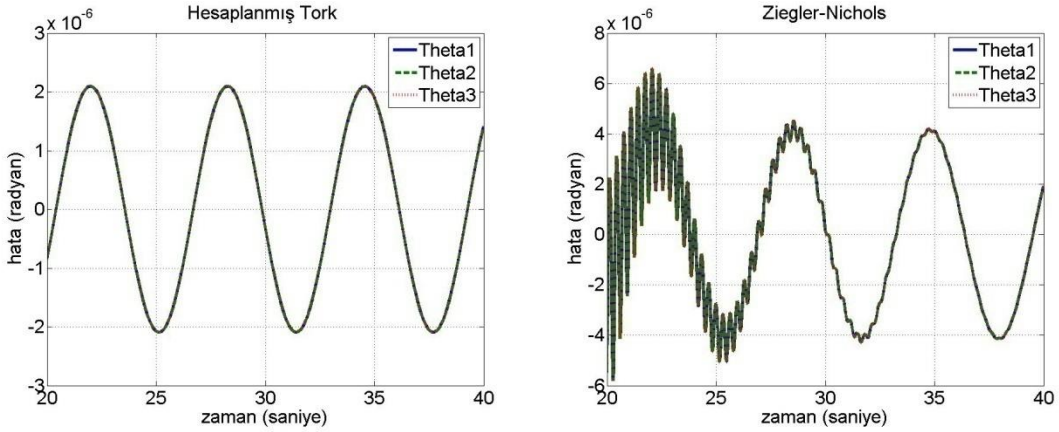
Oransal kazacı 625 alarak Ziegler-Nichols yöntemini uyguladığımızda periyodun ne olduğunu bilmemiz gerekir. Şekil 5.18'den incelediğimizde periyodun yaklaşık olarak 6.2 saniye olduğu görülecektir.

Çizelge 5.2 : Kcr=625 için Ziegler-Nichols çizelgesi.

Kontrol	K_p	K_i	K_d
P	312.5	∞	0
PI	284.1	5.167	0
PID	375	3.1	0.775

Yukarıda belirtilen değerleri kullandığımızda aşağıdaki sonuçlara ulaşıldı.



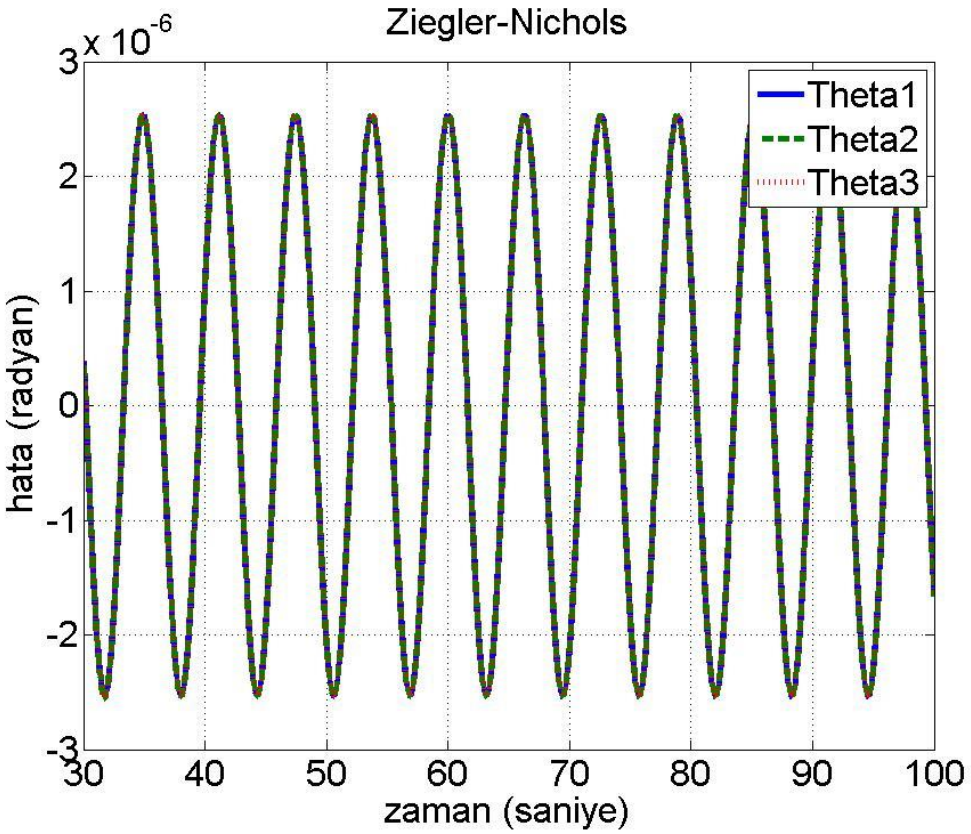


Şekil 5.20 : Hesaplanmış tork yöntemi ile Ziegler-Nichols uygulamasının karşılaştırması.

Yukarıdaki şekilde görüldüğü gibi Ziegler yöntemi yerine "Hesaplanmış Tork" adıyla ifade edilen yöntemin karşılaştırılması yapılmıştır. 0-40 saniye aralıklarında çalışıldığında Ziegler-Nichols yönteminin ancak 20 saniye gibi bir sürede kalıcı hal hatası durumuna girebildiği görülürken Hesaplanmış Tork yönteminde bu süre 1.189 saniye ile sınırlı kalmıştır. Bu süre bizim için oldukça uygun bir süre olup kalıcı durum hataları karşılaştırıldığında Hesaplanmış Tork yönteminin $2 * 10^{-6}$ radyan mertebelerindeki hatasına karşılık Ziegler-Nichols yönteminde bu hataların $4 * 10^{-6}$ değerlerinde olduğu ve ancak 35. saniyeden sonra düzgün bir çizgi takip edebildiği görülmüştür. Ziegler-Nichols yönteminde kalıcı durum hatasının daha iyi anlaşılabilmesi için aşağıdaki şekilde 30. Saniyeden sonrası gösterilmiştir.

Sonuç olarak; robot dinamik sistemi tasarımında hesplanmış tork yöntemi genel olarak kullanılan bir yöntemdir. Ziegler-Nichos yöntemi gibi kazanç katsayılarının belirlenmesinde kullanılan ek yöntemlere ihtiyaç duyulmadan kendi içinde gerekli değerler hesaplanabilmektedir. Şekillerden de görüleceği gibi kalıcı durum hatasa olarak uzun vadede çok fazla farklılık göze çarpmamasına karşın kısa vadede hesaplanmış tork yöntemi çok güzel sonuçlar vermekte ve salınımı minimumda tutmaktadır.

Önemli olan bir diğer konuda hesaplanmış tork yönteminin integratör çarpanına ihtiyaç duymadan bu işlemleri yapması ve busayede sistem üzerinde ki matematiksel yükünün de kıyas yapıldığında az olmasıdır.



Şekil 5.21 : Ziegler-Nichols uygulaması kalıcı durum hatası

6. UYARLAMALI (ADAPTİF) KONTROL

6.1 Uyarlamalı Kontrol Hakkında Genel Bilgi

Bu bölüme kadar yapılan işlemler mükemmel ortam olarak tarif edebileceğimiz bir ortamda olduğumuz varsayılarak yapılmıştır. Yani robotumuza dış hiç bir etkenin etkimediği ve iç dinamikleri tam manasıyla bildiğimiz öngörülmüştür. Ancak laboratuar şartlarında bile yapılması mümkün olmayan bu durumları gerçek çalışma ortamlarında yapmak mümkün olamamaktadır. Bütün bu bilinmezliklerle beraber kontrol sistemini de geliştirmek ve gerek dış gerekse iç bozuntuları hesaba katmak zorundayız. Bu bölümde en yaygın olarak kullanılan dayanıklı (gürbüz) ve uyarlamalı kontrol sistemlerinden bahsedilecek ve uyarlamalı sistem uygulanıp hesaplanmış tork yöntemi üzerine ne gibi artıları olduğu incelenecektir.

Dayanıklı kontrol; genel olarak sistemin iç dinamiğinden belirsizlikler yerine dış bozucuların sistem üzerinde oluşturacağı bozuntulara karşı dayanıklı bir yapı içerir. Dayanıklı kontrol teorisinde örnek olarak bir uçağın kontrol sistemini verebiliriz. Uçağın bütün parametrelerini doğru bir şekilde hesaplayıp dinamik sistemini oluşturmuş olsak dahi hava koşullarını her an için doğru hesaplamak mümkün olamayabilir. Bu durumda sisteme sürekli olarak dışarıdan bir bozuntu müdahelesi bulunmaktadır. Bunu engellemek için sisteme tutarsızlık etmeni diye de ifade edilen bir değer ataması yapılır. Bu sistem hazırlamasında H-İnfinity teorisi oldukça etkili bir yöntemdir.

Uyarlamalı kontrol; sistemin iç dinamiğinde hata olduğu varsayımı üzerine yapılan bir çalışmadır. Örnek olarak uçakların kanatlarında yakıt vs. sebeplerle ağırlık veya ölçü farkı olduğu varsayılabilir. Sistemimizde varsayacağımız gibi herhangi bir eklemin üzerindeki ağırlık değerinin yanlış olduğu veya üzerine fazladan yük bindiği varsayılabilir. Uyarlamalı kontrol sistemi bu hatayı farkedip çözmeli ve sistemi istenen duruma getirmelidir.

Genel olarak robotik üzerine çalışan araştırmacılar robotik sistemlerinde parametrelerin belirsizliği sorununu yenmek amacıyla uyarlamalı kontrol kullanılmaktadır zira bütün dinamiği mükemmel biliyor olsanız da robotun taşıyacağı

yükte belirsizlikler olması muhtemeldir ve uyarlamalı kontrol sistemi buna çok iyi cevap vermektedir.

Robot üzerine yapılan uyarlamalı kontrol çalışmalarının genellikle bozuntu etkilerinden kaynaklanan yörünge hataları üzerine yapılan yaklaşımlar içerisinde oldukça iyi olduğu görülmüştür. Ancak şu belirtilmelidir ki uyarlamalı kontrol sisteminin uygulamasında yerel lineerleştirme teknikleri, zamanla değişen öngörüler gibi değişik yöntemleri kullanmak zordur [34].

6.2 Hesaplamalı Tork Yaklaşımı ile Uyarlamalı Kontrol Yaklaşımı

Hazırlanan sistem modelleri ne kadar mükemmel olursa olsun gerçek hayatta iki belirsizlik durumunu tam manasıyla modellemek mümkün olamamaktadır. Bunlar eklemlerin ağırlıkları ve eklemler arasındaki sürtünme kuvvetleridir. Bu belirsizlikleri önlemek veya gözardı edilebilecek seviyeye çekebilmek için iyi uyarlamalı kontrol sistemine sahip olunmalı ve bu belirsizlikler için sağlam bir güncelleme kuralı oluşturulmalıdır.

Güncelleme kuralı; anlık olarak takip etmek durumunda bulunduğumuz hatalara bağlı olarak nasıl bir tepki verileceğini belirten bir formül yapısıdır. Uyarlamalı hesaplamalı tork kontrolcüsü üzerine çalışmak için ilk olarak hata sistemi için bir formül üretilmelidir ki aşağıda verilmiştir [34].

$$\tau = W(q, \dot{q}, \ddot{q})\varphi \qquad \textbf{(6.1)}$$

Bu hata formülünde $W(q, \dot{q}, \ddot{q})$ zamana bağlı bir fonksiyondur ve φ bilinmeyen parametreleri ifade etmektedir ki eklem ağırlıkları ve sürtünme kuvvetleri bunlardan ikisidir. Yukarıda belirtilen formül önemlidir zira bilinen ve bilinmeyen parametreleri ayırma imkânı sağlamıştır. Yukarıdaki formülü 3. bölümde ele alınan tork formülü ile eşleştirirsek ortaya aşağıdaki gibi bir formül çıkacaktır.

$$W(q, \dot{q}, \ddot{q})\varphi = D(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + F(\dot{q}_d) \qquad \textbf{(6.2)}$$

Yukarıda belirtilen formül sadece tork denklemlerini içeren bir formüldür ve hesaplanmış tork yöntemine ait değerleri içermemektedir. Bu sebeple sistemi düzenlersek aşağıdaki formüle ulaşırız.

$$W(q,\dot{q},\ddot{q})\varphi = D(q)(\ddot{q}_d + K_d\dot{e} + K_p e) + C(q,\dot{q})\dot{q} + G(q) + F(\dot{q}_d) \quad \textbf{(6.3)}$$

Yukarıda gösterilen formüle nasıl ulaşıldığı Dinamik Kontrol bölümünde incelenmiştir. Sistem üzerinde oluşan her türlü bozuntu direkt olarak kütle matrisine, christofel sembollerine, yerçekimine ve sürtünmeye etkiyecektir. Bu sebeple biz bu değerlerin gerçek değerini bilemiyor olsak bile olduklarını varsayacağımız tahminsel ifade eklemek durumundayız. Bu sebeple formülü aşağıdaki gibi tekrardan yazabiliriz.

$$W(q,\dot{q},\ddot{q})\varphi = \widehat{D}(q)(\ddot{q}_d + K_d\dot{e} + K_p e) + \hat{C}(q,\dot{q})\dot{q} + \hat{G}(q) + \hat{F}(\dot{q}_d) \quad \textbf{(6.4)}$$

Yukarıdaki formülde " ^ " işareti tahmin edilen dinamik değerleri ifade etmektedir. Dinamik kontrol bölümün 5.4 ve 5.5 nolu denklemlerle ifade edildiği gibi hata durumu istediğimiz konum ile meydana gelen konum durumu arasındaki farktan kaynaklanmaktadır ve bu denklemler üzerinden aşağıdaki gibi bir çıkarım yapılabilir.

$$\ddot{q}_d = \ddot{e} + \ddot{q} \quad \textbf{(6.5)}$$

Bu denklemi 6.4 nolu denklemin içine eklersekaşağıdaki gibi sonuç çıkacaktır.

$$\tau = \widehat{D}(q)\left(\ddot{e} + K_d\dot{e} + K_p e\right) + \widehat{D}(q)\ddot{q} + \hat{C}(q,\dot{q})\dot{q} + \hat{G}(q) + \hat{F}(\dot{q}_d) \quad \textbf{(6.6)}$$

Denklem 6.2'den hatırlanacağı üzere klasik tork denklemini yukarıdaki formüle eklersek aşağıdaki gibi bir denkleme ulaşılır.

$$\tau = \widehat{D}(q)\left(\ddot{e} + K_d\dot{e} + K_p e\right) + W(q,\dot{q},\ddot{q})\hat{\varphi} \quad \textbf{(6.7)}$$

Yukarıdaki denklemde görüldüğü gibi bilinmeyen parametre hataları tahmin olarak işin içine sokulmuştur. Parametre hatası aşağıdaki gibi ifade edilmektedir.

$$\tilde{\varphi} = \varphi - \hat{\varphi} \quad \textbf{(6.8)}$$

φ gerçek parametre değerlerini $\hat{\varphi}$ tahmin edilen parametre değerlerini ifade etmektedir. Bu ikisi birbirinden çıkarıldığında parametre hatalarına " $\tilde{\varphi}$ " ulaşılır. Bu durumda 6.1 ve 6.7 nolu denklemleri toplarsak aşağıdaki denklemlere ulaşırız.

$$\widehat{D}(q)\left(\ddot{e} + K_d\dot{e} + K_p e\right) + W(q,\dot{q},\ddot{q})\hat{\varphi} = W(q,\dot{q},\ddot{q})\varphi \quad \textbf{(6.9)}$$

$$\ddot{e} + K_d\dot{e} + K_p e = \widehat{D}^{-1}(q)W(q,\dot{q},\ddot{q})(\varphi - \hat{\varphi}) \quad \textbf{(6.10)}$$

$$\ddot{e} + K_d\dot{e} + K_p e = \widehat{D}^{-1}(q)W(q,\dot{q},\ddot{q})\tilde{\varphi} \quad \textbf{(6.11)}$$

Bu denklem durum uzayı formunda yazılırsa oluşacak denklem, hata vektörü ve A, B katsayıları aşağıdaki gösterildiği gibi olacaktır. I_n birim matris ve O_n sıfır matrisi olarak alınmıştır [34].

$$\dot{e} = Ae + BD^{-1}(q)W(q,\dot{q},\ddot{q})\tilde{\varphi} \quad \textbf{(6.12)}$$

$$e = \begin{bmatrix} e \\ \dot{e} \end{bmatrix} \qquad B = \begin{bmatrix} O_n \\ I_n \end{bmatrix} \quad A = \begin{bmatrix} O_n & I_n \\ -K_p & -K_v \end{bmatrix} \quad \textbf{(6.13)}$$

Bu işlemlerden sonra hata vektörünün kararlı olup olmadığı bilinmelidir. Bunun için Lyapunov kararlılık analizi yapılabilir. Lyapunov kararlılık analizi yapılırken öncelikle V(x) şeklinde bir Lyapunov fonksiyonu belirlenmelidir. Bu denklemler x=0 için aşağıdaki eşitliklere uyuyorsa sistemin kararlı olduğu anlaşılmış olur.

$$\begin{gathered} x \neq 0 \; için \; V(x) > 0 \\ x \neq 0 \; için \; V(x) < 0 \\ \|x\| \to \infty \; için \; V(x) \to \infty \\ V(0) = 0 \end{gathered} \quad \textbf{(6.14)}$$

Sistemin kararlı olabilmesi için gerekli durumlar Lyapunov metoduyla uyarlamalı kontrol sistemi içine yerleştirilebilir ki bu sistem 4 adımda aşağıda anlatılmıştır;

- Öncelikle hata eşiliklerinin türevi alınmalıdır.
- İkinci olarak, parametre hataları (sistemin iç dinamiğinden gelen hatalar) ve sinyal hatalarını (iç dinamiklerin hatasız kabul edildiği kontrol sisteminden kaynaklanan hatalar) ifade eden bir denklem oluşturulmalıdır. Sinyal hatasını " e " ve parametre hatalarını " $\tilde{\varphi}$ " ile gösterdiğimiz hatırlanırsa aşağıdaki formül seçilebilir.

$$V = e^T Pe + \tilde{\varphi}^T \Gamma^{-1}\tilde{\varphi} \quad \textbf{(6.15)}$$

Bu denklemde V'nin Lyapunov analizine uygun bir yapıda olabilmesi için P pozitif, sabit ve simetrik bir matris ve Γ köşegen ve pozitif bir matris olmalıdır. " γ " pozitif ve sabit olmak koşuluyla " Γ " matrisini aşağıdaki gibi yazmak mümkündür.

$$\Gamma = diag(\gamma_1, \gamma_2, \dots, \gamma_r) \tag{6.16}$$

- Üçüncü olarak, yukarıda seçilmiş olan Lyapunov fonksiyonunun zamana göre türevi alınır. Eğer ki türev " – " ise kararlılık belirlenmiş olur.

$$\dot{V} = -e^T Qe + (parametre\ hatasına\ bağlı\ bazı\ terimler) \tag{6.17}$$

P ve Q matrisleri Lyapunov analizi ile belirlenebilir. Q matrisi pozitif tanımlı, simetrik bir matristir. Pozitif bir Q matrisi Lyapunov eşitliklerinden çıkarımla beraberinde pozitif simetrik bir P matrisi getirecektir.

$$A^T P + PA = -Q \tag{6.18}$$

- Dördüncü olarak, bazı ekstra eklentiler yardımıyla da ifade edilebilmektedir ki bu eklentiler sıfıra eşit olmalıdır. Kararlı uyarlamalı kontrol modeli elde etmek için bazı değişikliklere ihtiyaç duyulur. ε durum hatasının elementlerinin lineer bir kombinasyonunu ifade etmektedir. ξ ise w vektörüne bağlıdır [39].

$$\dot{\theta} = -\Gamma \varepsilon \xi \tag{6.19}$$

Robotik sistemimizi Lyapunov teorsi kullanarak kararlılık analizi yapmak istediğimizde öncelikle zamana bağlı türevlerinin alınması gerekmektedir. Bu sebeple 6.15 nolu denklemin zamana bağlı türevleri alınırsa aşağıdaki gibi olacaktır.

$$\dot{V} = e^T P\dot{e} + \dot{e}^T Pe + 2\tilde{\varphi}^T \Gamma^{-1} \dot{\tilde{\varphi}} \tag{6.20}$$

Yukarıdaki denklem yazılırken aşağıdaki eşitlikten faydalanılmıştır.

$$[\tilde{\varphi}^T \Gamma^{-1} \tilde{\varphi}]^T = \tilde{\varphi}^T \Gamma^{-1} \dot{\tilde{\varphi}} \tag{6.21}$$

Daha önce sistemin durum uzayı modeli 6.12 nolu denklemde verilmiştir. Bu denklem 6.20 nolu denklemin içine yerleştirdiğinde aşağıdaki sonuçlar meydana çıkar [6].

$$\dot{V} = e^T P(Ae + BD^{-1}(q)W(.)\tilde{\varphi}) + (Ae + BD^{-1}(q)W(.)\tilde{\varphi})^T Pe + 2\tilde{\varphi}^T \Gamma^{-1}\dot{\tilde{\varphi}} \quad \textbf{(6.22)}$$

$$\dot{V} = -e^T Qe + 2\tilde{\varphi}^T(\Gamma^{-1}\dot{\tilde{\varphi}} + W^T(.)\widehat{D}^{-1}(q)B^T Pe) \quad \textbf{(6.23)}$$

6.22 nolu denklemdeki terimler birleştirilir ve denklem sadeleştirilirse 6.23 nolu denklem oluşur. Bu denklem görüleceği üzere 6.17 nolu denkleme benzemektedir. 6.23 nolu denklemde "$\dot{\tilde{\varphi}}$ " yerine aşağıdaki eşitliği yazar ve 6.17 nolu denklemde belirtildiği gibi eklenti terimini çıkartırsak 6.25 nolu denkleme ulaşırız [6].

$$\dot{\tilde{\varphi}} = -\Gamma W^T(.)\widehat{D}^{-1}(q)B^T Pe \quad \textbf{(6.24)}$$

$$\dot{V} = -e^T Qe \quad \textbf{(6.25)}$$

Bu noktadan sonra uyarlamalı kontrol güncelleme kuralı oluşturulabilir. Bunun için 6.8 nolu denklemi 6.24 nolu denklemin içerisine sokarak işlem yapılır. Bu işlemde dikkat edilmesi gereken durum gerçek parametrik değerlerin değişmediğidir. Bu durumda gerçek parametre değerlerinin türevi sıfıra eşit olacaktır. Bu durum 6.26 nolu denklemde ifade edilmiştir [6].

$$\dot{\tilde{\varphi}} = \dot{\varphi} - \dot{\hat{\varphi}} \Longrightarrow \dot{\tilde{\varphi}} = 0 - \dot{\hat{\varphi}} \quad \textbf{(6.26)}$$

6.26 nolu eşitlik 6.24 nolu eşitliğe ilave edildiğinde aşağıdaki sonuç ortaya çıkacaktır.

$$\dot{\hat{\varphi}} = \Gamma W^T(.)\widehat{D}^{-1}(q)B^T Pe \quad \textbf{(6.27)}$$

Bu denklem bizim uyarlamalı güncelleme kuralımızdır ve sistemin içine eklenecektir. Sistem kararlılık analizine V fonksiyonu üzerinden devam edildiğinde sınır koşulları sebebiyle aşağıdaki denklemlere ulaşılacaktır ve sistemimizin kararlı olduğu görülmüş olacaktır [6].

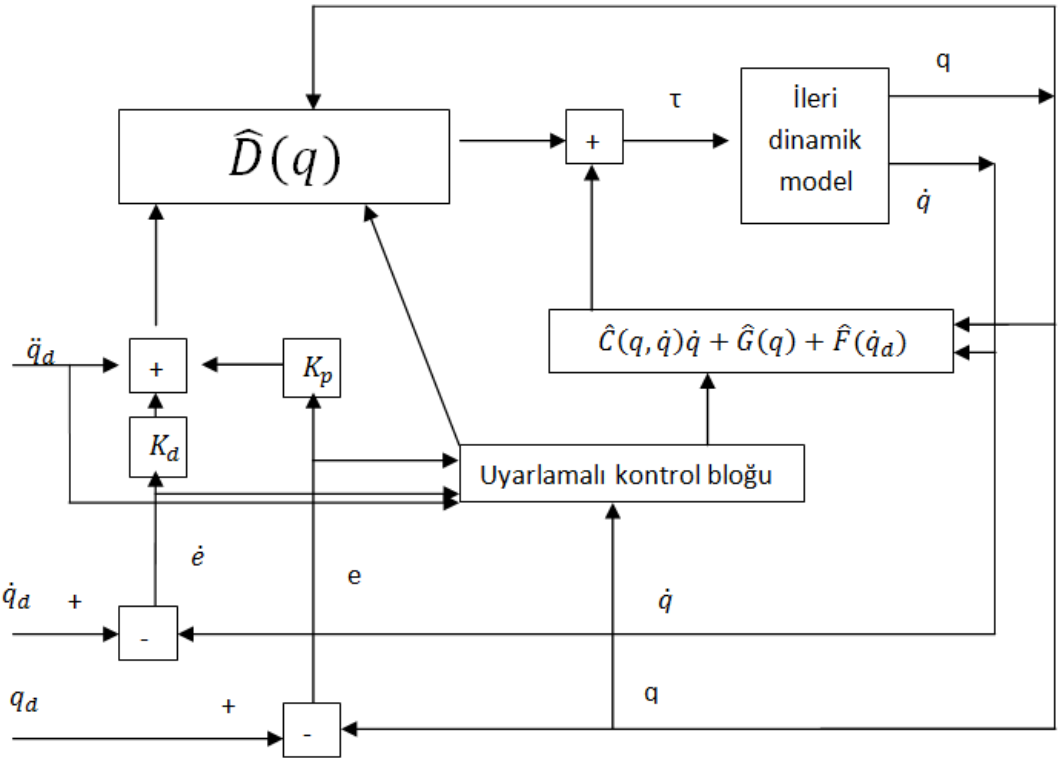
$$\lim_{t\to\infty} V = V_\infty$$

$$\lim_{t\to\infty} \dot{V} = 0 \quad \textbf{(6.28)}$$

Sonuç olarak tork kontrolcümüz olarak 4.6 nolu denklem kullanılacak, güncelleme kuralımız 6.27 nolu denklemde verilen formüller olacaktır ki diğer eşitliklerin sağlanması zaruridir.

6.3 Uyarlamalı Kontrol Sistem Tasarımı

Uyarlamalı kontrol sistem tasarımında asıl hedef güncelleme kuralını oluşturabilmektir. Güncelleme kuralı sürekli olarak bir önceki işlemin sonucuna göre bize yeni parametre değerleri verecek ve bu şekilde sürekli olarak bir güncelleme yapacaktır. Güncelleme kuralı oluşturulduktan sonra sıra hesaplamalı tork sisteminin üzerinde yapılacak değişiklere gelecektir. Bu değişiklikler aşağıdaki şekilde görüldüğü gibi ters dinamik denklem üzerinde yapılmaktadır. Uyarlamalı kontrol bloğundan alınan yeni parametre değerleri ters dinamik denklem birimlerinin içerisine dahil edilerek sistemin sürekli olarak kendini düzeltmesi amaçlanmaktadır.



Şekil 6.1 : Uyarlamalı tork kontrol şeması.

Bizim sistemimizde bu değişken parametreler eklemlerin kuvvetleridir. Bu sebeple ters tork denklemlerinin içinde bulunan kütle değerleri değişken olarak tanımlanmıştır. Uyarlamalı kontrol bloğunun içinde 6.27 nolu denklemde ifade edilen güncelleme kuralı kullanılmıştır.

"Γ" işareti ile gösterilen (r x r) boyutlu pozitif kazanç matrisi olup yapılan denemeler sonucunda sistemimizde 400 olarak kabul edilmiştir. Oransal kontrol katsayısında olduğu gibi belli bir değerden sonra bozuntuya sebebiyet vermeye başlamakta ve kalıcı durum hatasını olumsuz etkilemeye başlamaktadır. Sistemimiz 3 eklemli olduğundan aşağıdaki gibi yazılabilir.

$$\Gamma = 400 * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \textbf{(6.29)}$$

"W" işareti ile gösterilen regresyon matrisi 6.1 nolu denklemde belirtildiği gibi bilinmeyen parametrelerle çarpıldığı zaman tork kuvvetine eşit olmaktadır. İncelediğimiz sistemde bilinmeyen parametreleri m1, m2, m3 olarak düşünürsek regresyon matrisi bu parametrelerin tork denkleminden çekilmesiyle sağlanabilir. Ancak tork denklemimiz 3 sayfaya yakın çıkmaktadır. Bu sebeple her iki tarafı bilinmeyen parametrelerin tersi ile çarpmak suretiyle regresyon matrisini bulmak mümkün olabilmiştir.

$$\tau\varphi^{-1} = W(q, \dot{q}, \ddot{q})\varphi\varphi^{-1} = W(q, \dot{q}, \ddot{q}) \quad \textbf{(6.30)}$$

"$\widehat{D}$" kütle matrisinin içinde bulunan bilinmeyen parametrelerin yerine sürekli olarak değişken olarak parametre değerlerini ifade etmektedir. Tahmin edilen parametre değişkenlerine bağlı oalrak bu kütle matrisi sürekli olarak değişmektedir.

Sistem B ve kontrol amaçlı olarak kullanılan P matrisi ile çarpılır. Bu değer hata değeri ile anlık olarak çarpılarak güncelleme kuralı oluşturulmuş olur. Bu hata değeri 6x1 lik matris olup hataların türevlerini de içermektedir. Bu değerler üzerinden güncelleme kuralı oluşturulmuş ve sisteme dâhil edilmiştir.

Bu sistemin analizi denklem 6.18'de verildiği gibi yapılmıştır. A matrisi 6.13 nolu denklemdeki gibi alınmıştır. P ve Q matrislerinin formülleri aşağıda verilmiştir. Analiz sonucunda eşitliğe doğru bir şekilde ulaşılmış ve sistemin düzgün olduğuna karar kılınmıştır.

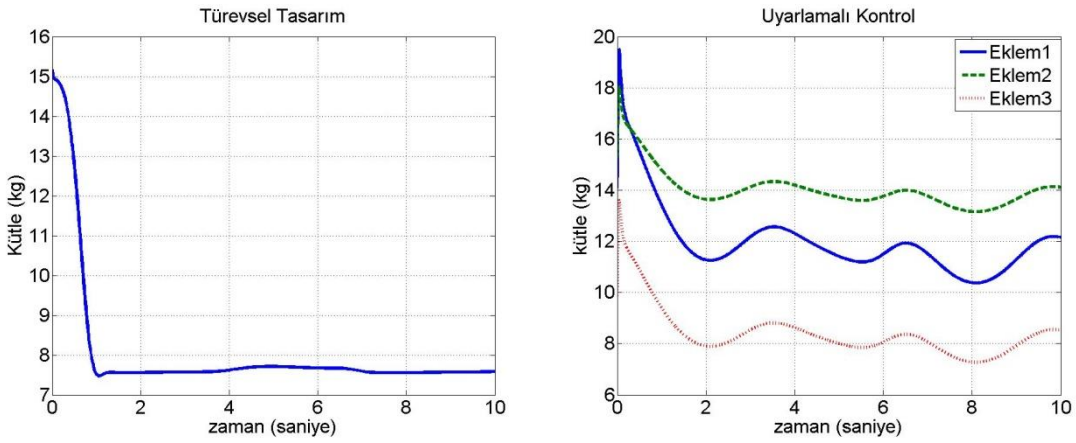
$$P = \begin{bmatrix} P_1 I_n & P_2 I_n \\ P_2 I_n & P_3 I_n \end{bmatrix} = \frac{1}{4}\begin{bmatrix} (2k_p + k_d) I_n & I_n \\ I_n & 2I_n \end{bmatrix} \quad \textbf{(6.31)}$$

$$Q = \begin{bmatrix} 0.5k_p I_n & 0_n \\ 0_n & 2I_n \end{bmatrix} \quad \textbf{(6.32)}$$

6.3.1 Uyarlamalı kontrol sistemi türevsel tasarım

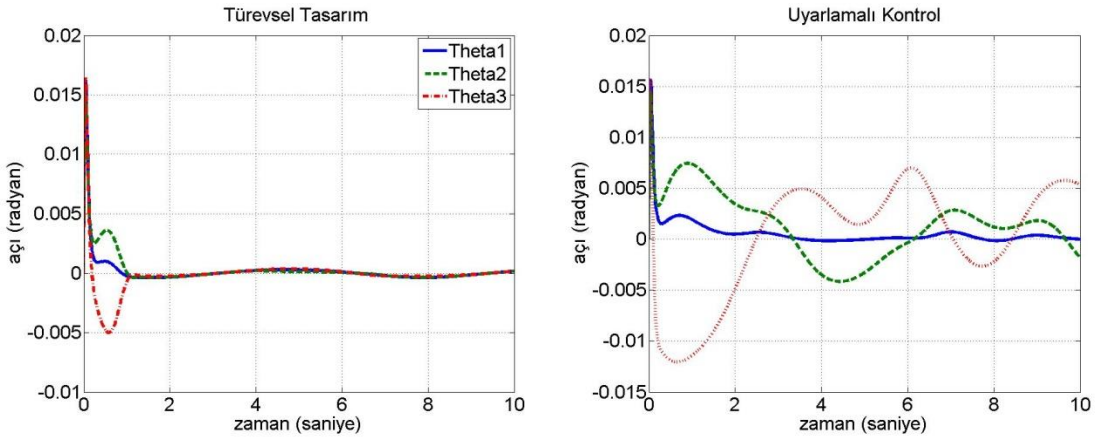
Uyarlamalı konrtol sistemi oluşturulurken temelde regresyon matrisinin doğru bir şekilde oluşturulması yatmaktadır. Bu işlem denklem 6.30'da verildiği şekilde yapılabildiği gibi tork denkleminin değişmesi öngörülen parametreye göre türevi alınarak yapılabilir. Bu durumda P matrisi gibi iç kontrol matrislerine gerek kalmamakta ve o eklemin çok yüksek hatalarına rağmen güzel cevap verdiği görülebilmektedir.

Örnek olarak eklemlerin hatalarını 3 katına aldığımızda uyarlamalı kontrol blokları çalışmıyorken türevsel olarak yaptığımız uyarlamalı kontrol 10 katı hatada bile düzgün sonuçlar vermektedir. Ayrıca tek bir bilinmeyen parametrede normal tasarım cevabının hatası türevsel tasarımdan çok daha yüksek olmaktadır.



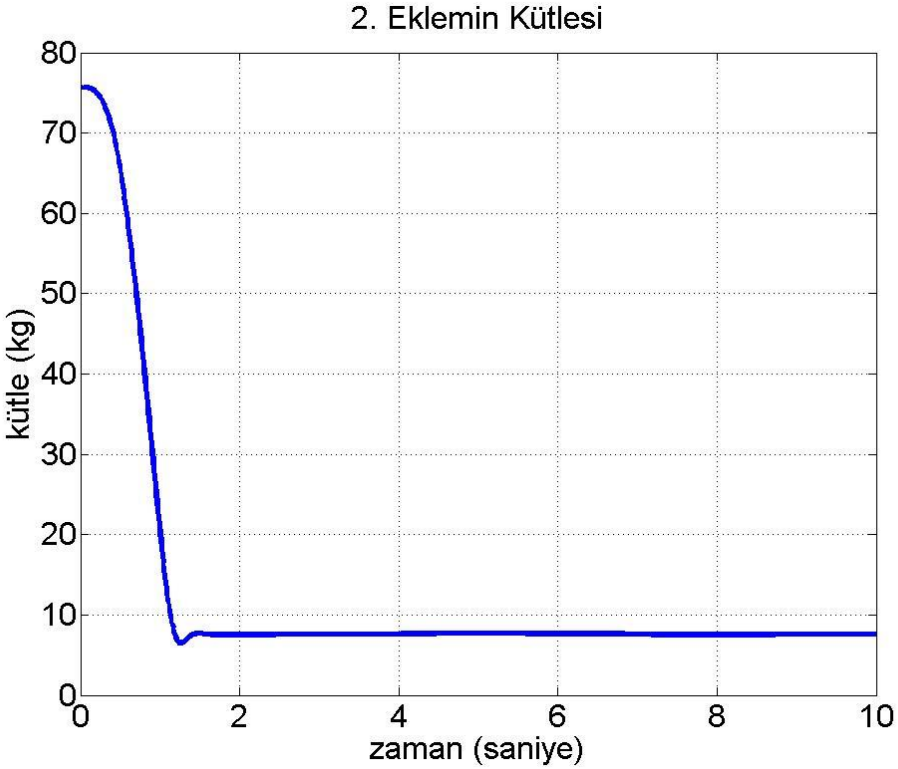
Şekil 6.2 : 2. Eklemin kütlesi 2 katına çıkarıldığında sistem cevapları.

Yukarıdaki şekilde dikkat edilecek bir noktada şudur ki biz sadece bir eklem için kütlesel değişim yapmış olmamıza rağmen sistem 3 eklem için de benzer hata varmış gibi işlem yapmaktadır. Sonuç olarak ciddi oranda hata oluşturmakta ve bu hatayı çözmek için ağır matematiksel işlemlere girmektedir. Ayrıca tam manasıyla çözüme de ulaşamamaktadır.



Şekil 6.3 : 2. Eklemin kütlesi 2 katına çıkarıldığında açısal hatalar.

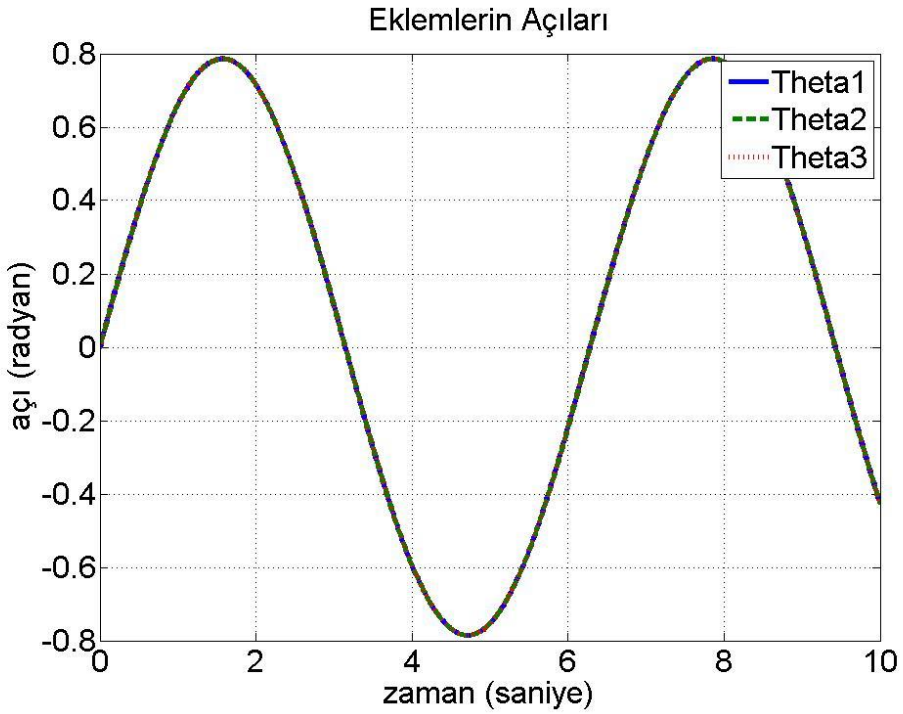
Şekil 6.2 ve 6.3'te görüldüğü üzere sadece bir eklemin değişmesi durumunda türev alma yöntemi kullanılarak regresyon matrisinin bulunduğu yöntem çok daha fazla etkili olmaktadır. Ancak bu yöntem sadece bir değişken için kullanılabilmektedir. Genel kullanım olarak sunulan uyarlamalı kontrol sistemi ise her bir eklem için kullanılabilmekte ve oldukça iyi sonuçlar verebilmektedir.



Şekil 6.4 : 2. Eklemin kütlesi 10 katına çıkarıldığında sistemin etkisi.

6.4 Uyarlamalı Kontrol Sistem Tasarımı

Uyarlamalı kontrol sistemini incelenirken öncelikle girişler sinus dalgası şeklinde verilmiştir. Bu şekilde bozuntuyu anlamak daha rahat olacaktır.

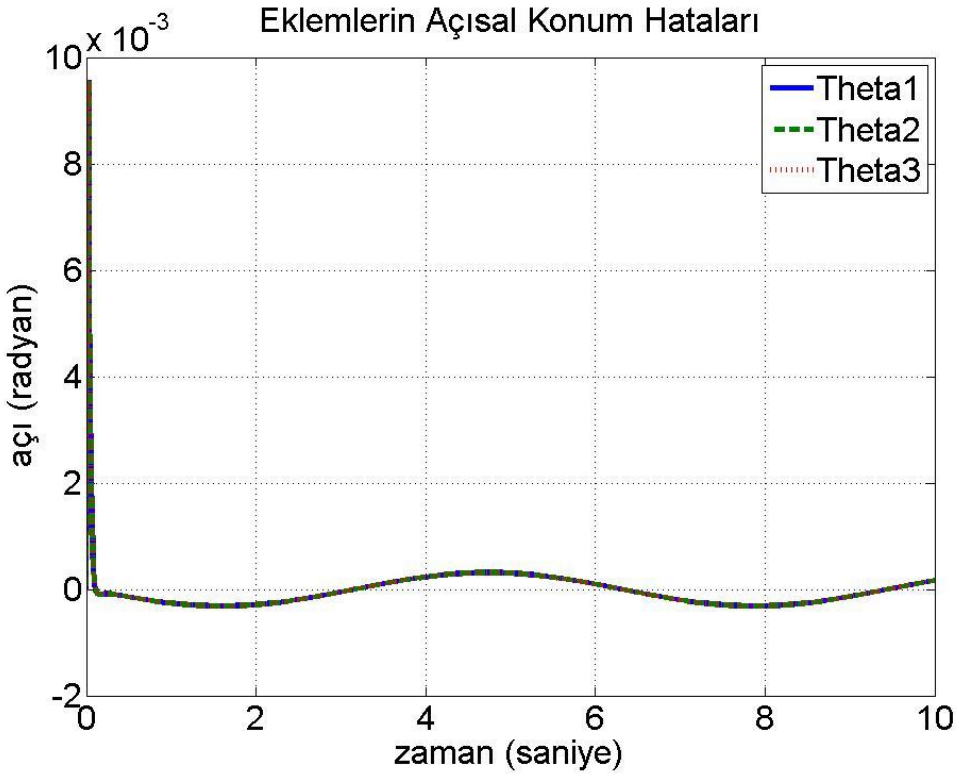


Şekil 6.5 : İzlenmesi istenen yol için eklemlerin zamana bağlı açısal değerleri.

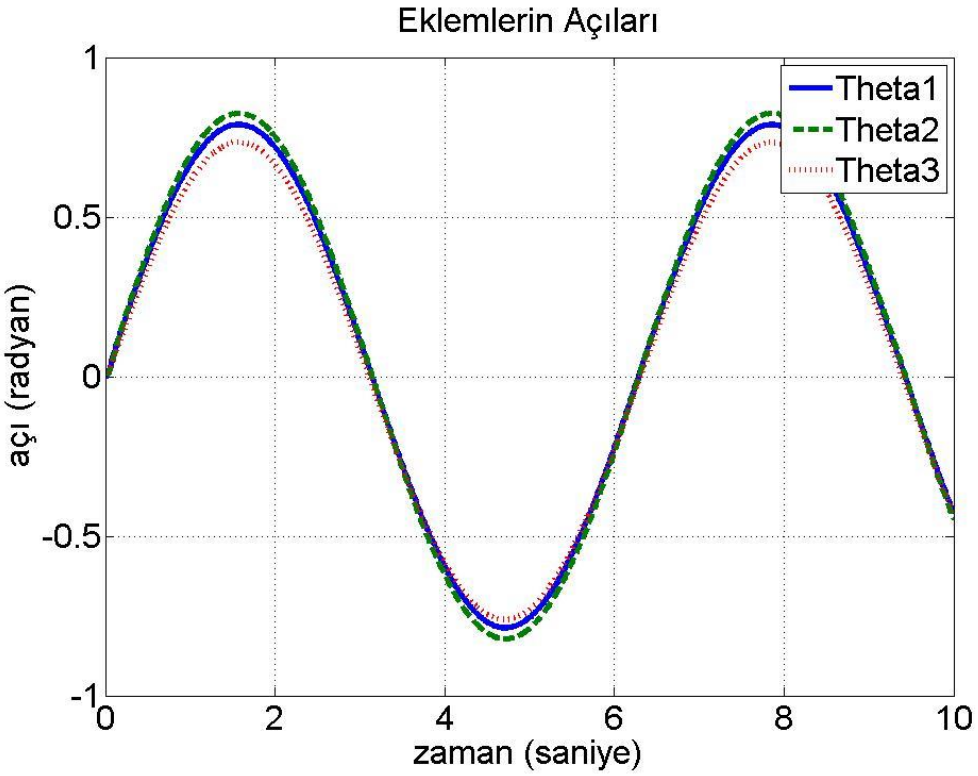
6.4.1 Hesaplamalı tork yöntemi simulasyon sonuçları

Hesaplamalı tork yönteminde tahmin edilen eklemlerin ağırlıkları üzerinde hiç bir değişim olmamaktadır ve olduğu gibi sisteme dâhil olmaya devam etmektedir. Eklemlerimizin hataları sırasıyla 1,0.5,1,5 ve 2 ile çarpılarak sonuçlar incelenirse aşağıdaki gibi olacaktır.

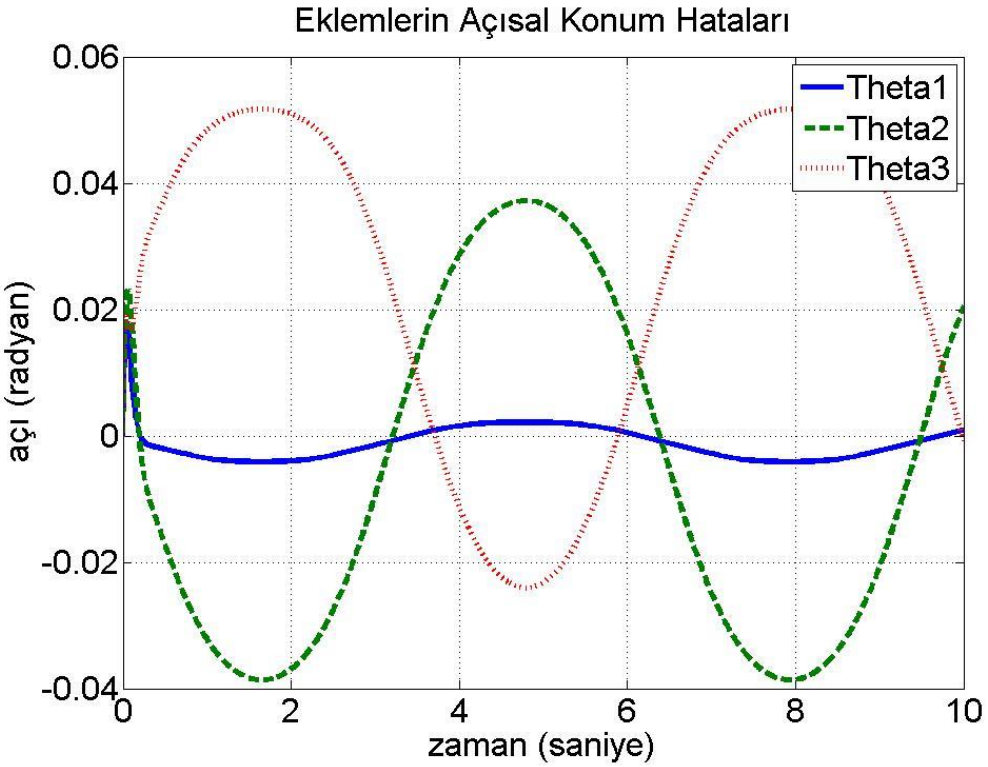
Aşağıda şekilde görüldüğü gibi bütün eklemlerin kütlelerinin bilinmesi durumunda eklermelerin açısal hataları 10^-5 seviyelerindedir. Ancak hata faktörleri işin içine girdiğinde hatalar artmaya başlamaktadır. Örnek olarak sistemi 0.5 hata ile çarptığımızda Şekil 6.8'de görüldüğü gibi 0.05 radyan seviyelerinde hataya sebebiyet vermektedir ki bu da yaklaşık 3 dereceye tekabül etmektedir. Bu hata sürekli olarak devam etmektedir.



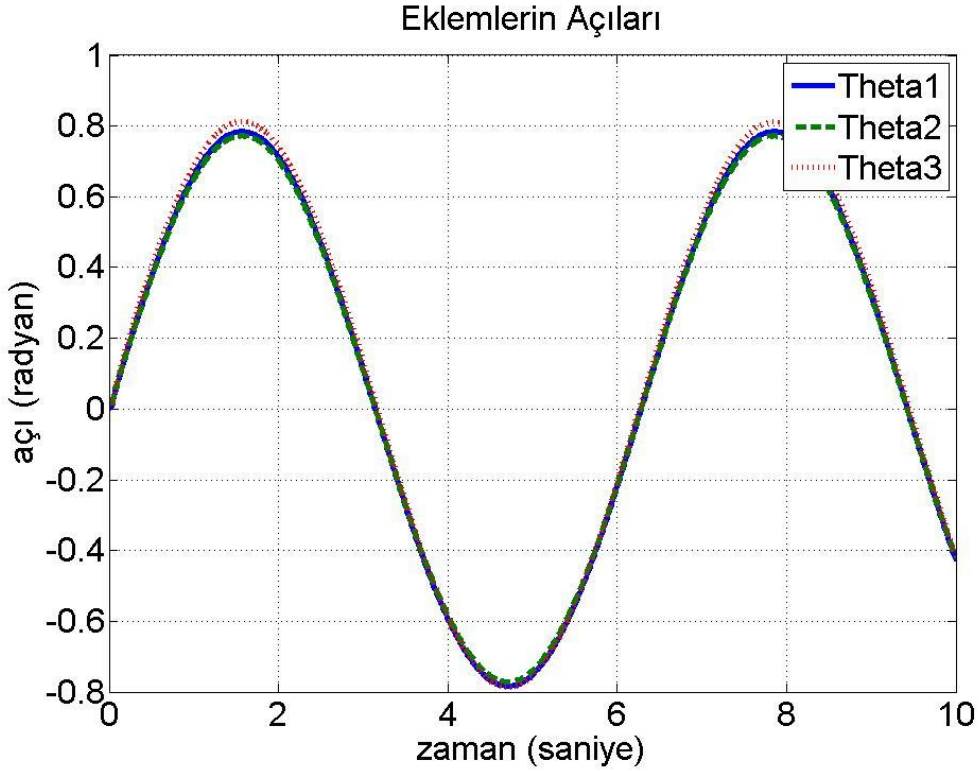
Şekil 6.6 : Bütün kütleler bilindiğinde eklemlerin açısal konum hataları.



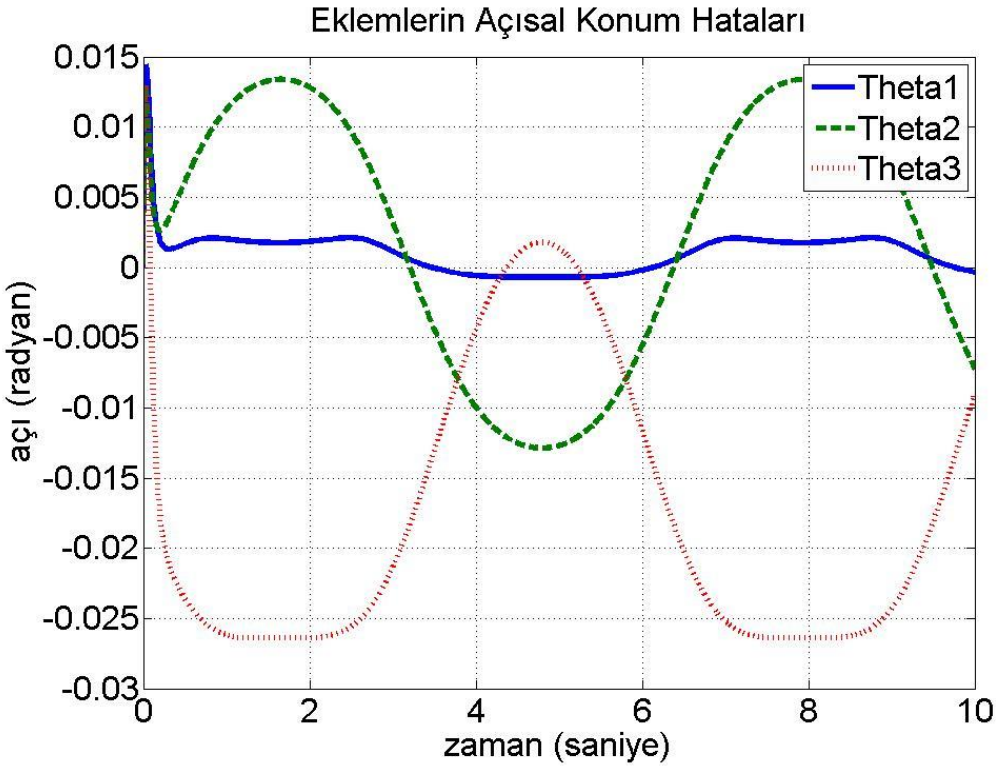
Şekil 6.7 : Eklem kütleleri 0.5 ile çarpıldığında eklemlerin açıları.



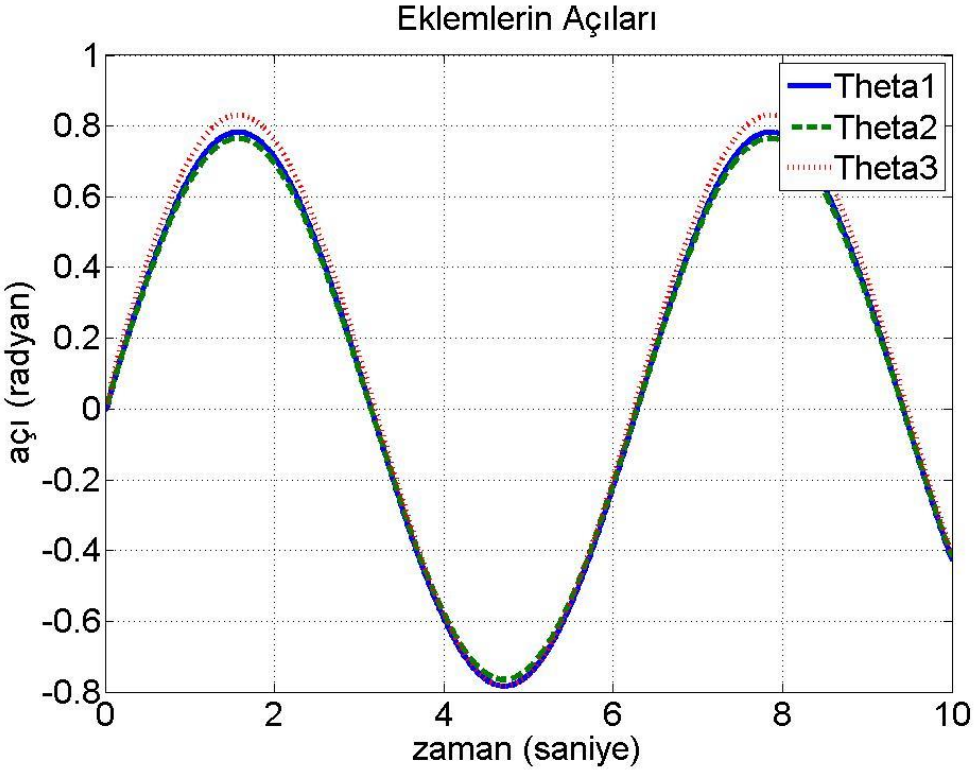
Şekil 6.8 : Eklem kütleleri 0.5 ile çarpıldığında eklemlerin açısal konum hataları.



Şekil 6.9 : Eklem kütleleri 1.5 ile çarpıldığında eklemlerin açıları.



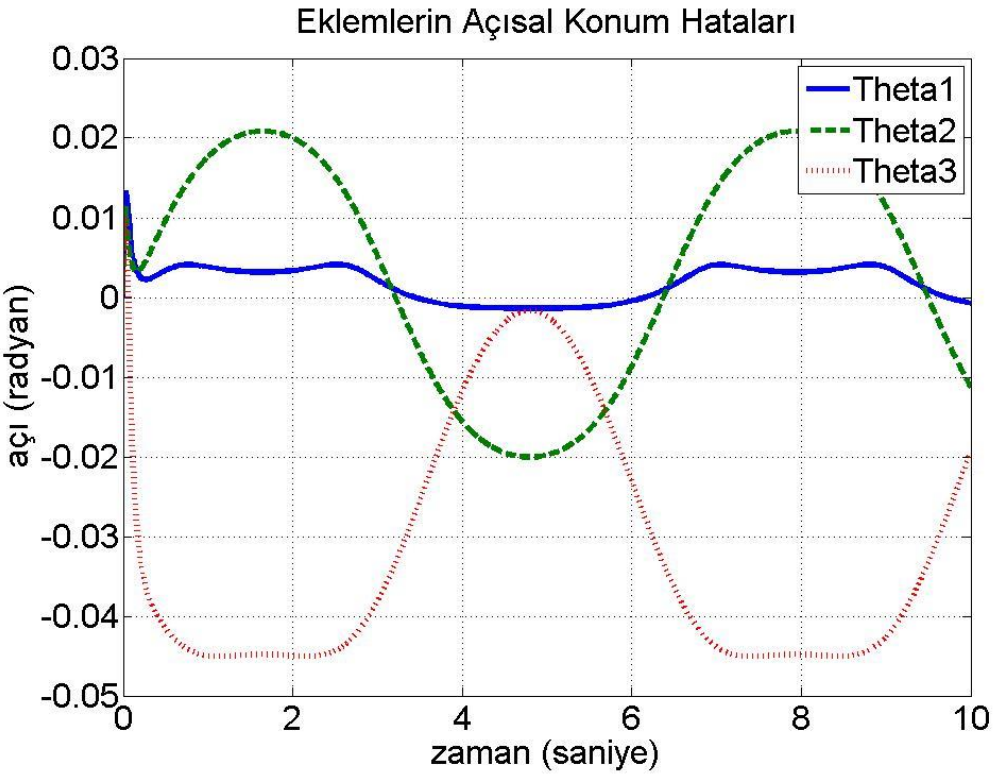
Şekil 6.10 : Eklem kütleleri 1.5 ile çarpıldığında eklemlerin açısal konum hataları.



Şekil 6.11 : Eklem kütleleri 2 ile çarpıldığında eklemlerin açıları.

Yukarıdaki şekillerde görüleceği gibi eklem kütleleri 1.5 katlık bir çarpım hatasıyla çalıştırıldığında 0.5 çarpımına göre oransal olarak eklem üzerinde daha az hata

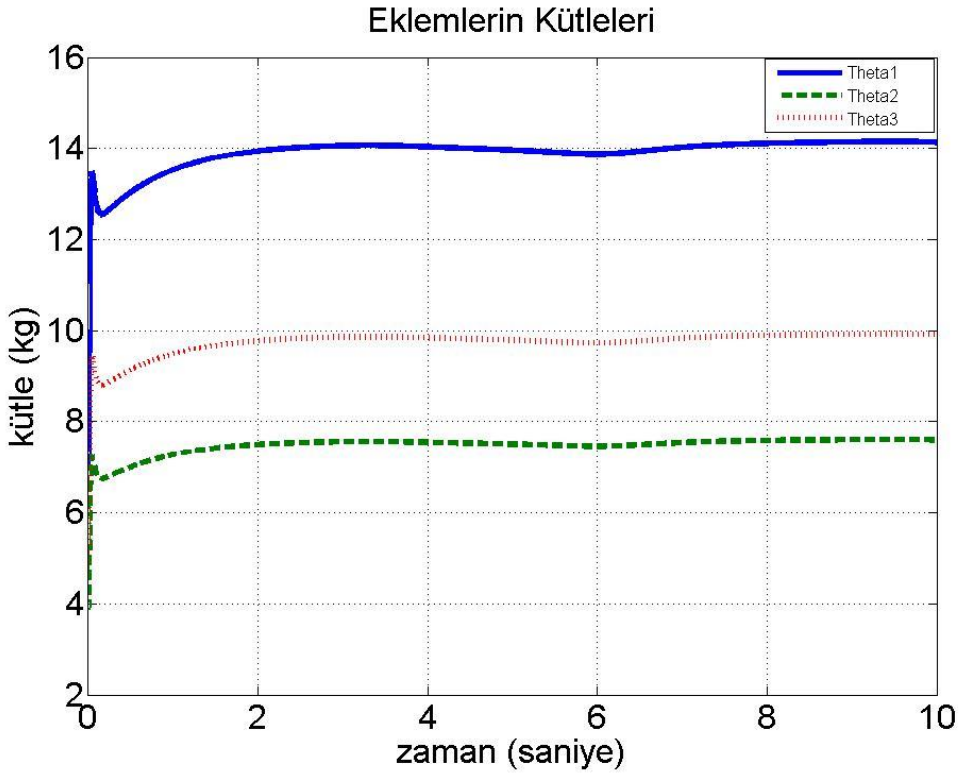
oluşturulduğundan daha az hataya sebebiyet vermektedir. Ancak eklem hatası 2'ye artırıldığında hata durumu da artmaktadır. Bu hatalar her bir eklemde 3-4 dereceye kadar varabilmektedir. Bu durumda bu hataların toplamı robotun uç konumunda ciddi hatalara sebebiyet verebilecek bir duruma gelebilmektedir ve Uyarlamalı kontrol sisteminde olduğu gibi güncelleyici bir sistem olmadığında hata sonsuza kadar devam edecektir.



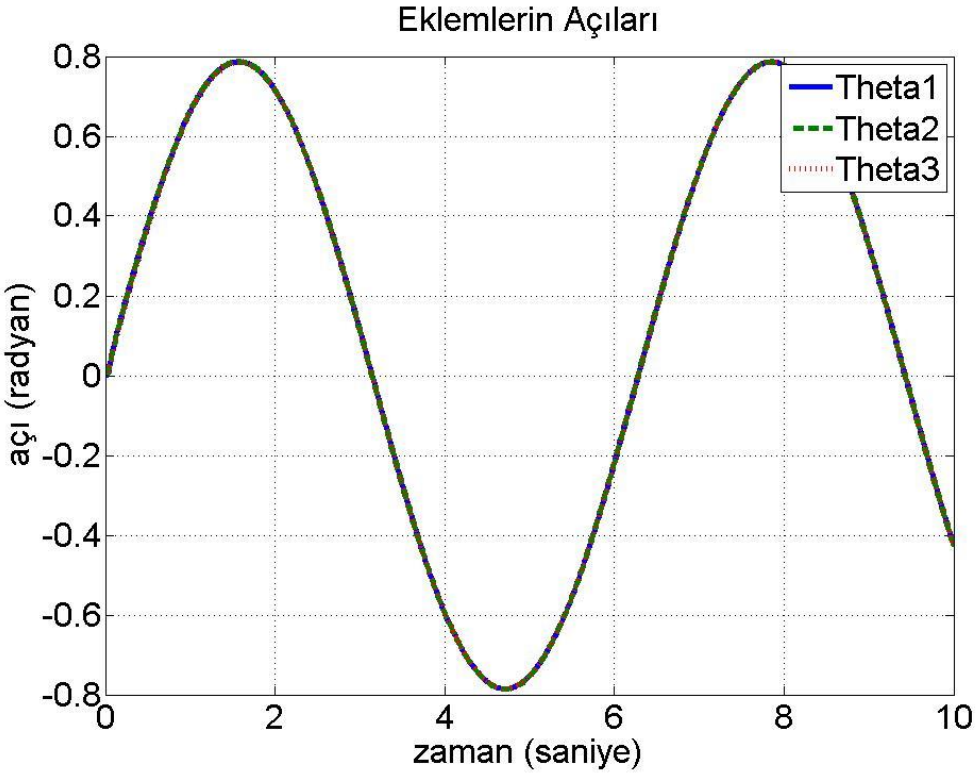
Şekil 6.12 : Eklem kütleleri 2 ile çarpıldığında eklemlerin açısal konum hataları.

6.4.2 Uyarlamalı kontrol sistemi simulasyon sonuçları

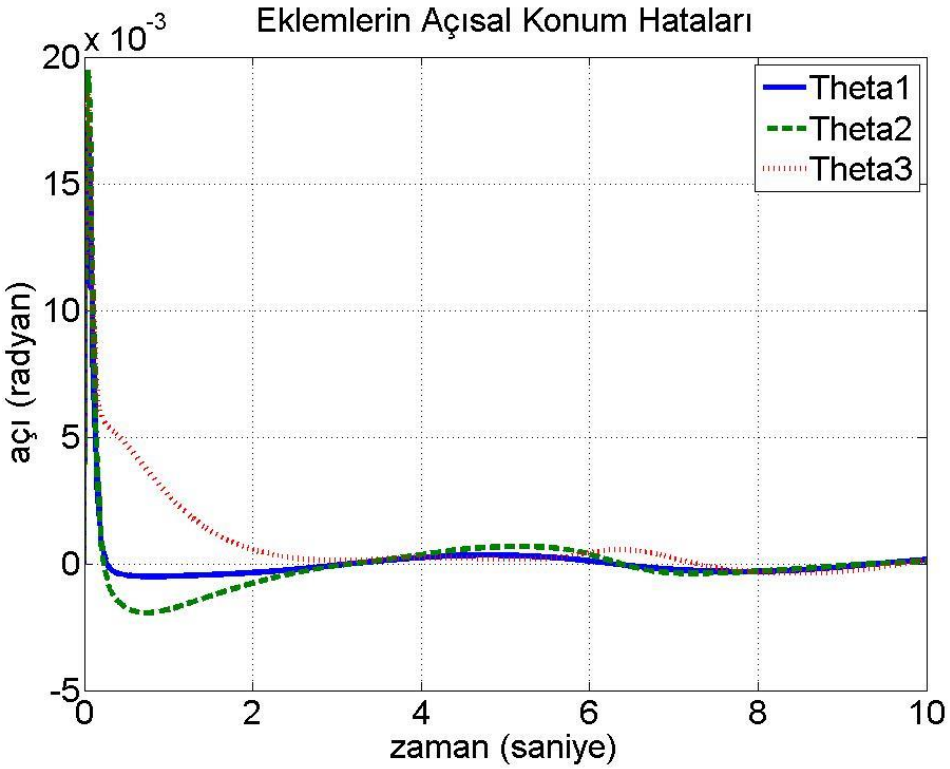
Hesaplanmış tork yöntemine uyarlamalı kontrol eklediğimizde dışarıdan verdiğimiz parametrik hatalar güncelleme kuralının sisteme dâhil olmasıyla beraber hızlı bir şekilde ortadan kalkmaktadır. Bunun sonuçlarını aşağıdaki şekillerde görmek mümkündür. Aşağıdaki şekillerde parametrik değerlerin 0.5,1.5 ve 2 ile çarpımı sonucu oluşan hatalar incelenmiştir.



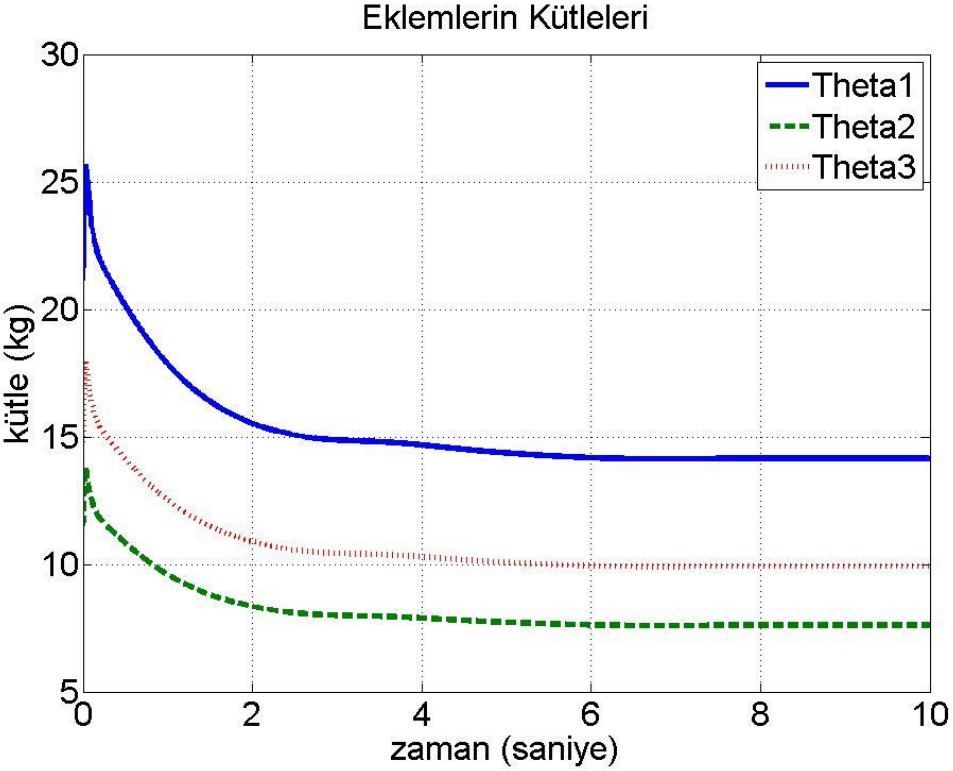
Şekil 6.13 : Eklem kütleleri 0.5 ile çarpıldığında eklemlerin kütleleri.



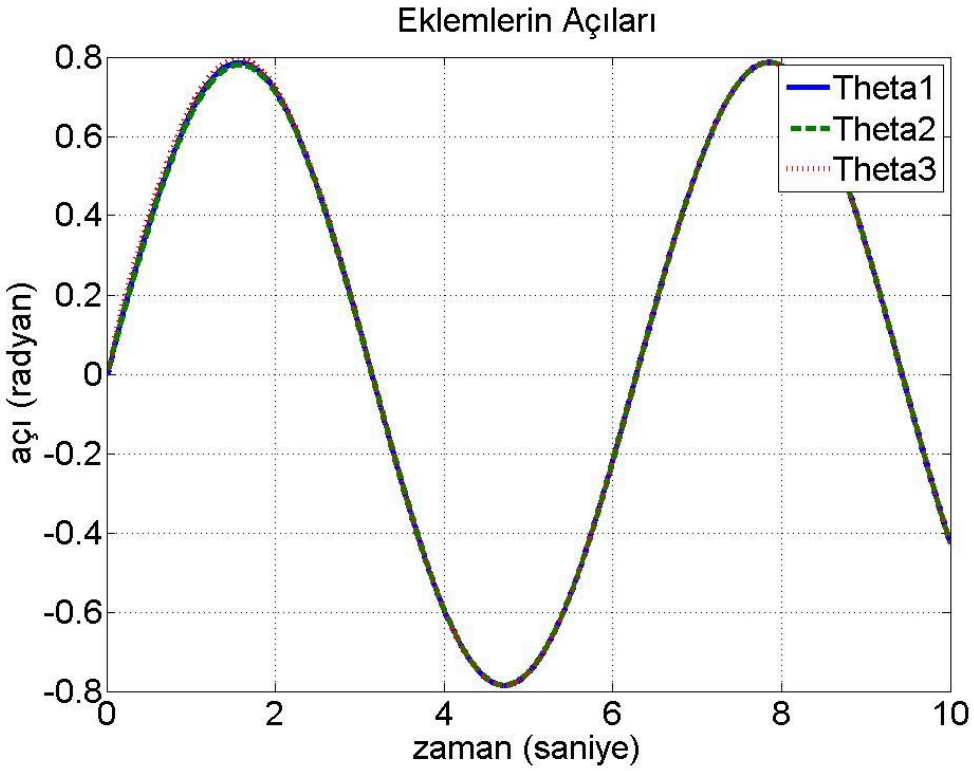
Şekil 6.14 : Eklem kütleleri 0.5 ile çarpıldığında eklemlerin açıları.



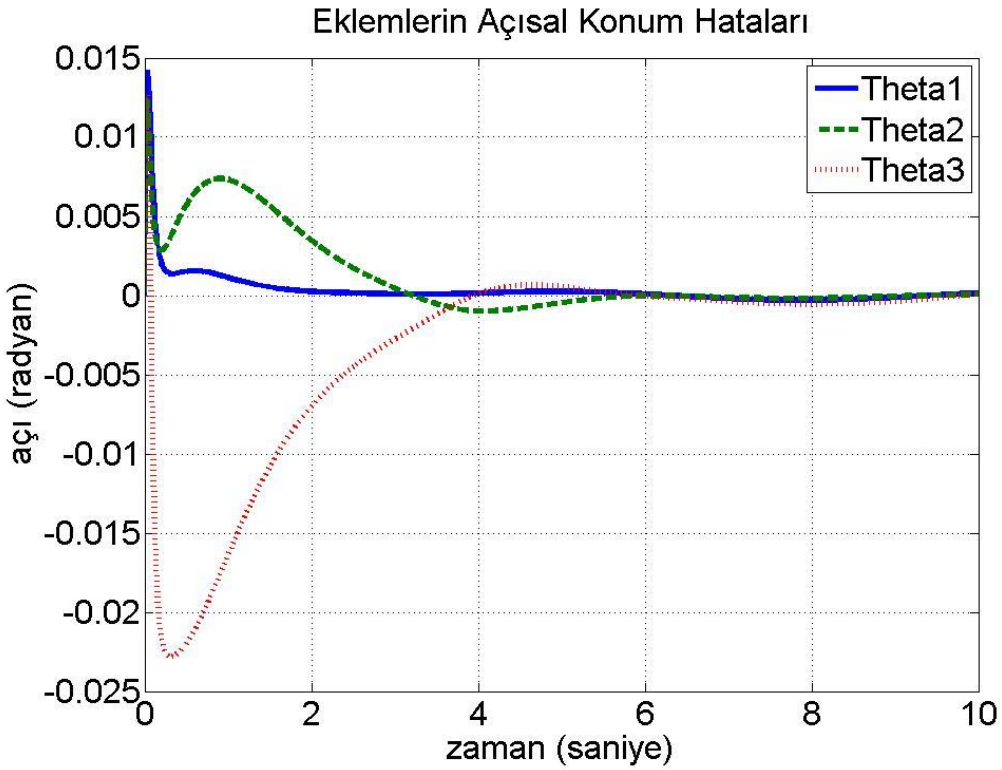
Şekil 6.15 : Eklem kütleleri 0.5 ile çarpıldığında eklemlerin açısal konum hataları.



Şekil 6.16 : Eklem kütleleri 1.5 ile çarpıldığında eklemlerin kütleleri.



Şekil 6.17 : Eklem kütleleri 1.5 ile çarpıldığında eklemlerin açıları.



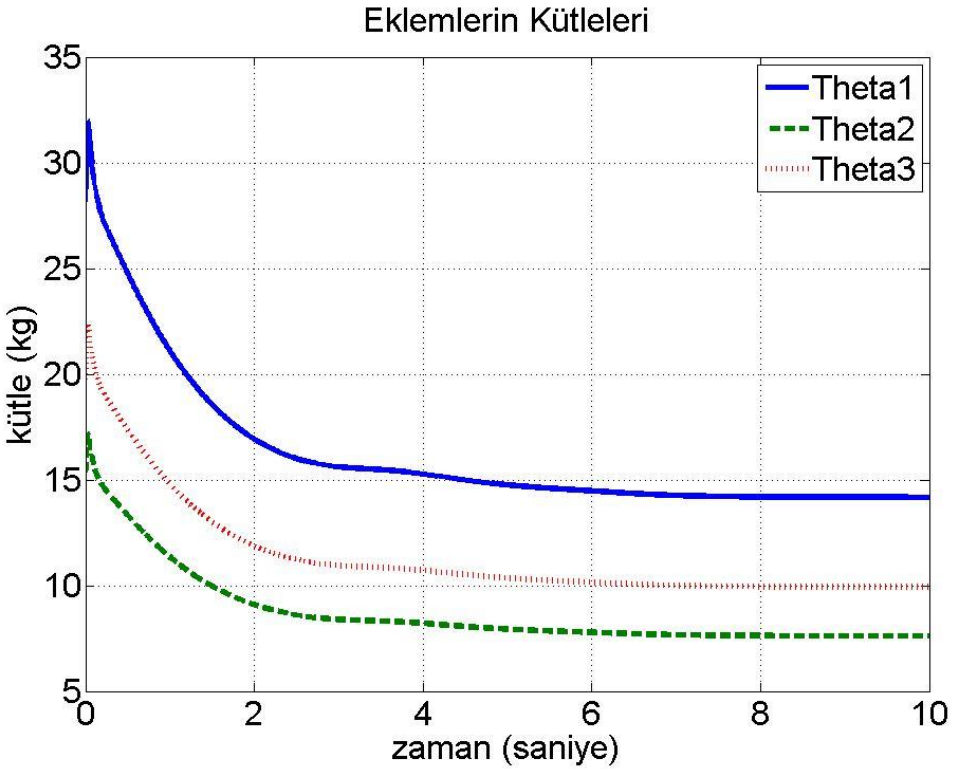
Şekil 6.18 : Eklem kütleleri 1.5 ile çarpıldığında eklemlerin açısal konum hataları.

Şekillerde görüldüğü gibi uyarlamalı kontrol sisteminde parametrik hatalar öncelikle sistem üzerinde ciddi etkiler bırakmakla beraber zaman içinde oturmakta ve neredeyse sıfır durumuna gelmektedir.

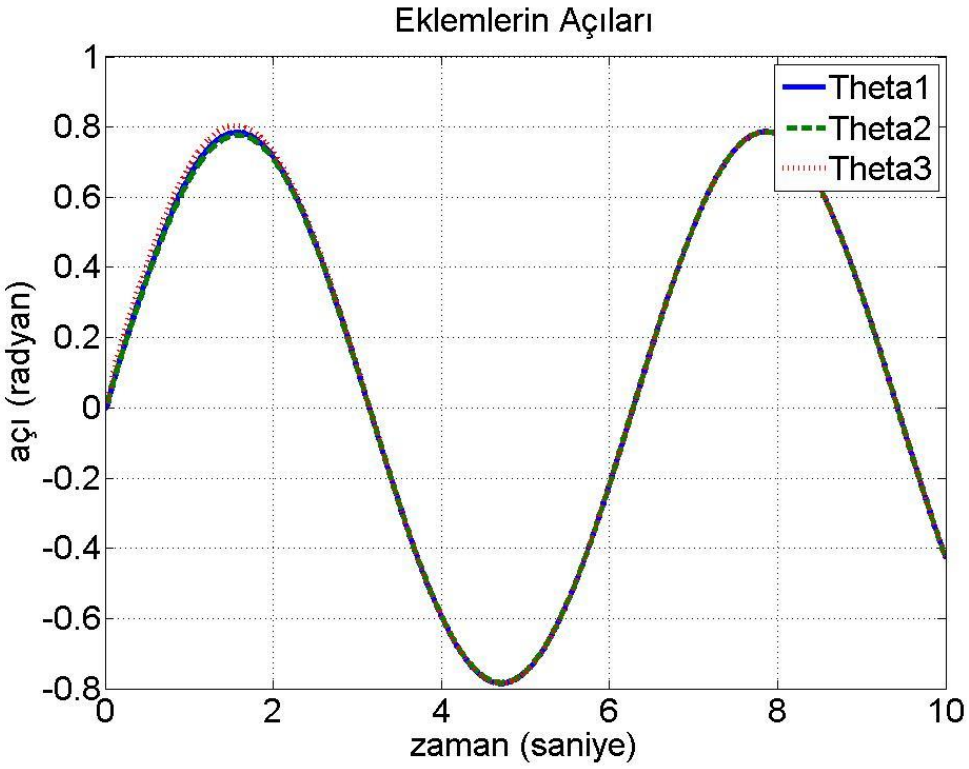
Öncelikle belirtmek gerekirki eklem açıları incelendiğinde ilk saniyelerde ufakta olsa oluşan hatalardan dolayı eklemlerin açıları birbirinden farkedilebilir durumda iken zamanla bu farkındalık ortadan kalkmakta ve sanki hiç hata yokmuş gibi görülmektedir. Oysaki aynı hatalar hesaplanmış tork yöntemine eklendiğinde bu farklılık sürekli olarak göze çarpmaktaydı zira hatalar süreklilik arz etmekteydi.

Sistemlerin açısal konum hatalarına baktığımızda göreceğimiz sonuç aşağıdaki çizelge 6.1'deverilmiştir.

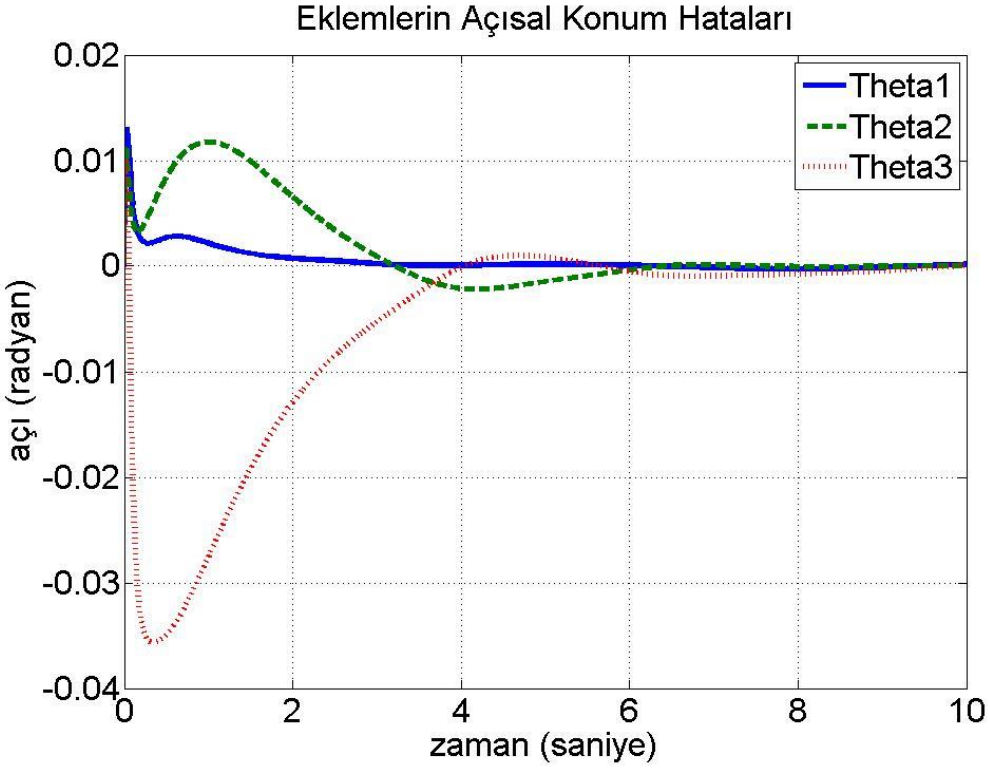
Çizelgeyi incelediğimizde uyarlamalı kontrol sisteminde hata ne olursa olsun sistemin belli bir sürede belli bir hata oranına yaklaştığı görülmektedir. Bu bizim için mükemmel denenebilecek bir sonuçtur. Zira sistem 3 içinde düzgün bir döngüye girmekte ve hatayı derece cinsine çevirdiğimizde maksimum 0.032 derecelik bir hatayı görmektedir.



Şekil 6.19 : Eklem kütleleri 2 ile çarpıldığında eklemlerin kütleleri.



Şekil 6.20 : Eklem kütleleri 2 ile çarpıldığında eklemlerin açıları.



Şekil 6.21 : Eklem kütleleri 2 ile çarpıldığında eklemlerin açısal konum hataları.

Çizelge 6.1 : Hesaplanmış tork ve uyarlamalı tork kontrolün karşılaştırması.

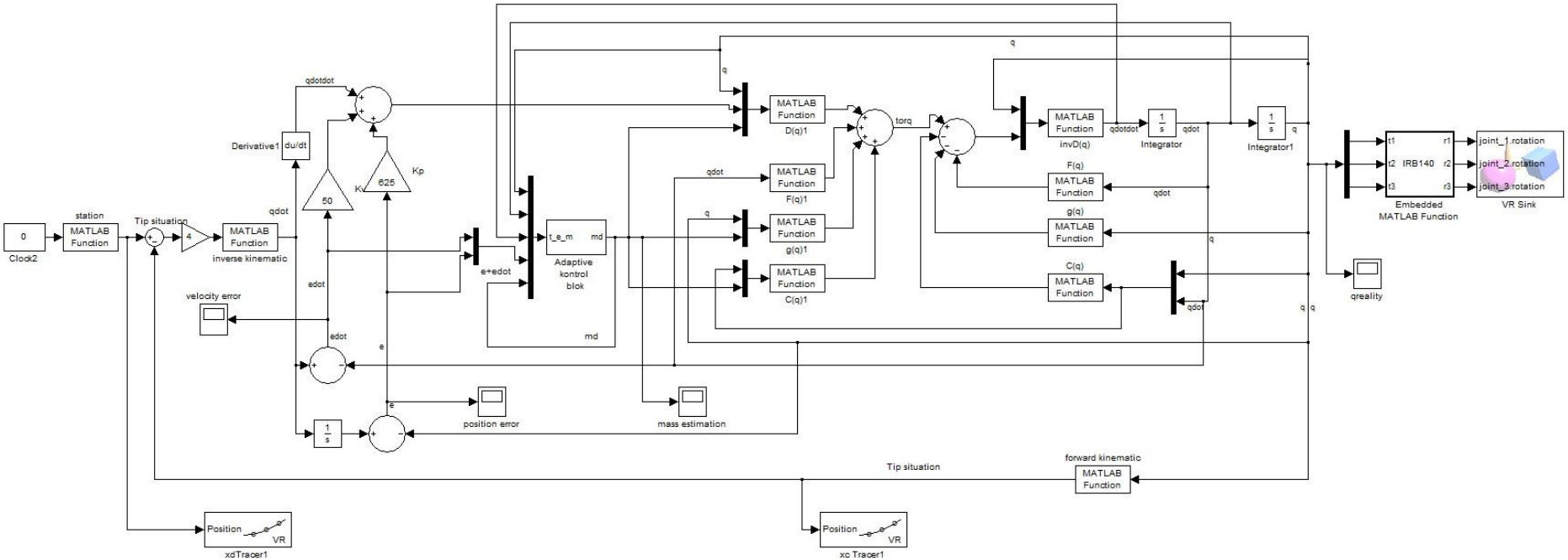
	Hesaplanmış Tork			**Uyarlamalı Kontrol**		
Parametrik Hata durumu	Aşım (radyan)	Yerleşme Zamanı (sn)	Sürekli Hal Hatası (radyan)	Aşım (radyan)	Yerleşme Zamanı (sn)	Sürekli Hal Hatası (radyan)
0.5m	0.05	Yok	0.05	$1.942 * 10^{-3}$	3	$5.65 * 10^{-4}$
1.5m	0.027	Yok	0.027	0.02	3	$5.65 * 10^{-4}$
2m	0.045	Yok	0.045	0.035	3	$5.65 * 10^{-4}$

Sonuç olarak, uyarlamalı kontrol parametrik hataları düzenlemek için oldukça uygun bir seçenek olarak karşımıza çıkmaktadır.

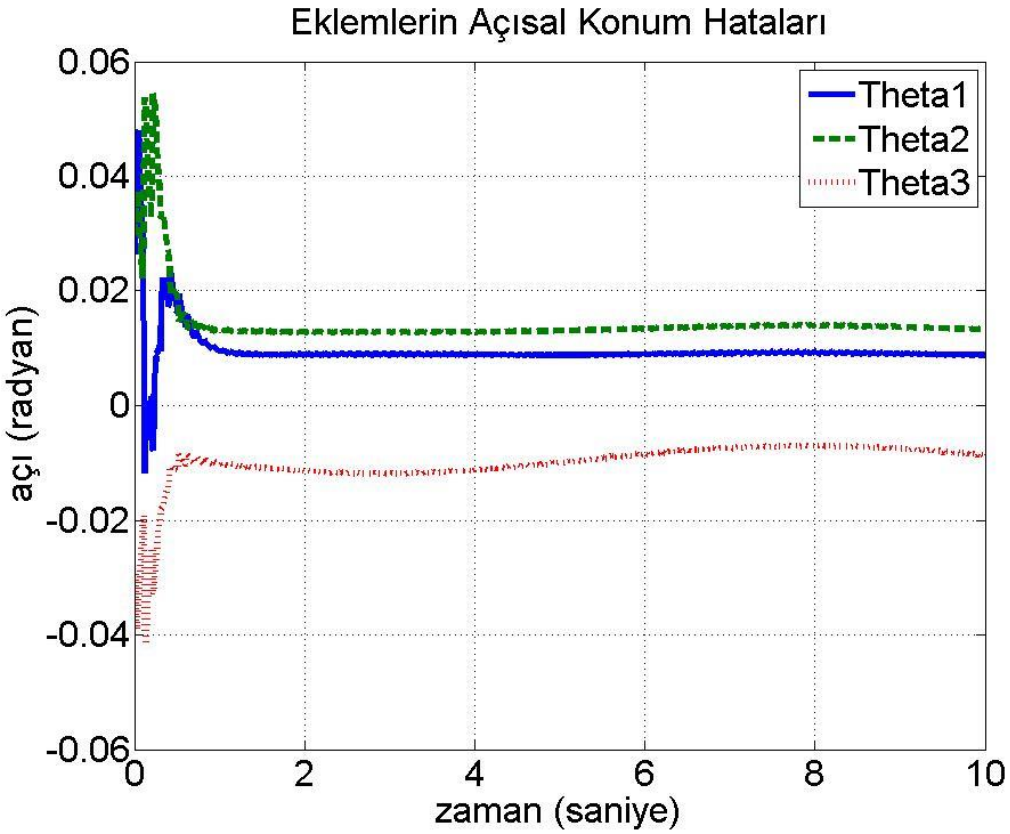
6.5 Uyarlamalı Kontrol Sistemi Yörünge Planlaması

Hesaplamalı tork yöntemine uygulayacağımız uyarlamalı kontrolün ardından belli bir yörüngeyi takip etmek istediğimizde sistemin vereceği cevabı daha net görmek için matlabda animasyon çalışması yapıldı. Sistem Şekil 6.22'de görüldüğü gibi hazırlanmıştır.

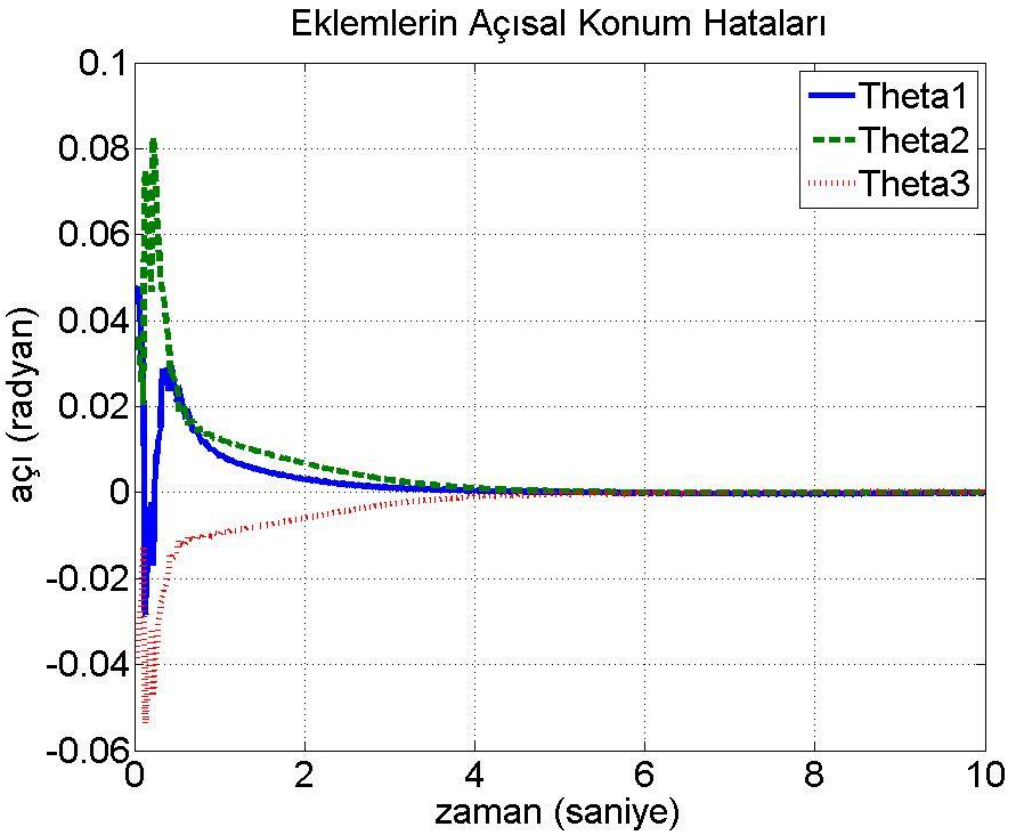
Sistemin çalışma mantığı incelenirse tork denkleminden çıkan eklem açılarından ileri kinematik yardımıyla uç organ konum bilgileri alınmakta ve hedef olarak belirlenen yeni konum ile arasındaki mesafe ölçülmektedir. Bu mesafe oransal kontrolcüye hata olarak girmekte ve belli bir büyültme oranından sonra ters kinematik yardımıyla eklemlerin açısal hızları belirlenip tork sistemine iletilmektedir. Bu döngü sürekli olarak devam etmektedir. Burada dikkat edilmesi gereken konu her iki sistemin diğer sistem üzerinde pozitif etkisi olmasıdır. Birbirlerinin hatalarını tolere etmeye eğilimli iki yapı şeklindedirler. Bu düzeltme eğilimi üzerinde hatayı ortadan kaldıracak kontrol uygulanmamış parametrik hatalara sahip hesaplanmış tork sisteminde daha iyi görmek mümkündür. Bunun karşılaştırması aşağıdaki şekillerde yapılmıştır.



Şekil 6.22 : Hesaplanmış tork üzerine uygulanan uyarlamalı kontrol sisteminde yörünge planlaması.



Şekil 6.23 : Hesaplanmış tork geri beslemeli yörünge sistemi.



Şekil 6.24 : Uyarlamalı kontrol geri beslemeli yörünge sistemi.

2. eklem kütlesinin 5 kat artılarak yapıldığı bu sistemlerde yörünge sisteminin geri beslemeli ve kontrollü olmasından dolayı her ne kadar dinamik model içinde hata olsa da sistem sürekli olarak düzelmeye gitmektedir. Şekil 6.23 ve 6.24'te gösterilen açısal hatalarda uyarlamalı kontrol ileri seviyede düzelme sağlarken hesaplanmış torkun ancak belli bir seviyeye kadar düzelme sağlayabildiği görülmüştür. Esasında bu işlemler eklemler üzerinde bulunan motorlar için sorun teşkil edebilmektedir zira dinamik modelde ki hata yüzünden sürekli bir hata durumu oluşmakta ve eklemler üzerinde daha fazla kuvvetsel değişim meydana gelmektedir.

Sonuç olarak; eklemler üzerinde belli derecelerde hataların mevcudiyeti göz önünde bulundurulduğunda uyarlamalı kontrol oldukça elverişli bir kontrol sistemidir.

7. SONUÇ VE ÖNERİLER

Bu çalışmada seri bir sanayi robotunun dinamik analizi yapılmış ve sonuçları dinamik kontrol bölümünde tartışılmıştır. Dinamik modelleme ve kontrol bölümlerinde de incelendiği gibi robotik sistemler oldukça karmaşık lineer olmayan denklemler içermektedir. Matlab programı kullanılarak yapılan bu işlemlerde hipotezde bulunan 3 soruya cevap bulunmaya çalışılmış ve aşağıdaki sonuçlara ulaşılmıştır.

Robot sistemimizin kendi iç dinamiğinden gelen bozuntular için literatürde bulunan ve üzerinde çalışılmış olan bir çok yöntem mevcuttur. Gürbüz kontrol, uyarlamalı kontrol, bulanık kontrol bunlardan bir kaçıdır. Literatürde bir çok çalışmada özellikle fuzzy ve uyarlamalı kontrolün bir arada yapıldığı bir çok çalışma mevcuttur. Bu tezde bu kontrol yöntemlerinden klasik olarak kullanılan uyarlamalı kontrol yöntemi uygulanmış ve parametrik hataların ortadan kalktığı gözlemlenmiştir.

Uyarlamalı kontrol sistemi kendi içinde atalet momenti tabanlı uyarlamalı kontrol gibi değişik metodlara sahiptir. Bu tezde de farklı bir şekilde yapılmış bir sisteminde incelemesi ve karşılaştırması yapılmış eksileri ve artıları incelenmiş, tek bir parametre için çalıştığında mükemmel sonuç verdiği gözlemlenmiştir. Ancak bizim uyguladığımız çalışmada türevsel yöntem olarak ifade ettiğimiz yöntem tek bir parametre için çalıştığından pek etkin olarak kullanımı mümkün görülmemektedir.

Uyarlamalı kontrol iç parametrik hataları incelemek amacıyla yapılan bir kontrol uygulaması olduğundan hesaplanmış tork yöntemi gibi bir yöntemin onun yerine geçmesi mümkün olamamaktadır. Ancak en son bölümde incelendiği üzere kinematik sistemle bütünleşik heraket ettirildiği taktirde sistem sürekli alarak hataya gidememekte ve kararlılığa eğilim göstermektedir. Bu şekilde sürekli olarak dinamik sistem hata verirken kinematik geri besleme sebebiyle sürekli olarak kendini düzeltmeye çalışmaktadır. Ancak bu durumun bize hem güç olarak hem de hassasiyet olarak ciddi derecede külfetleri olmaktadır. Bu sebeple bu şekilde bir kullanım uygun değildir.

Sonuç olarak bu çalışmada gerçek bir sanayi robotunun uyarlamalı kontrolü ve simulasyonu yapılmıştır ve bu şekilde bir robot üretimi yapılacak olsa önceden bilgisayardan test etme imkanı elde edilmiştir.

Aynı şekilde bu kontrol metodlarını diğer tip robotlara ve makinelere de uygulamak mümkündür.

KAYNAKLAR

[1] **Radavelli L., Simoni R., De Pieri E.R., Martin D.** (2012, November). *A comparative study of the Kinematics of Robots Manipulators by Denavit-Hartenberg and dual Quaternion,* Asociación Argentina de Mecánica Computacional, 31, 2833-2848.

[2] **Mohan A., Saha S.K.** (2008, September). *A Recursive, Numerically Stable, and Efficient Simulation Algorithm for Serial Robots with Flexible Links,* Multibody System Dynamic, 21, 1-35.

[3] **Rodriguez G., Kreutz-Delgado K., Jain A.** (1991, August). *A Spatial Operator for Manipulator Modeling and Control,* International Journal of Robotics Research, 10, 371-381.

[4] **Rodriguez G., Kreutz-Delgado K., Jain A.** (1992, March). *Spatial Operator Algebra for Multibody System Dynamics,* The Journal of the Astronautical Sciences, 40, 27-50.

[5] **Zhixiang T., Hongtao W.** (2010, July). *Spatial Operator Algebra for Free-Floating Space Robot Modeling ans Simulation,* Chinese Journal of Mechanical Engineering, 23.

[6] **Yıldız G.** (2013). *Nonlınear Control Methods Of Industrıal Serıal Robots.* Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi/Kontrol ve Otomasyon Mühendisliği Anabilim Dalı Kontrol ve Otomasyon Mühendisliği Programı, İstanbul.

[7] **Abdel-Malek K., Othman S.** (1999, June). *Multiple Sweeping Using the Denavit-Hartenberg Representation Method,* Computer-Aided Design, 31, 567-583.

[8] **Shah S.V., Saha S.K., Dutt J.K.** (2013). *Dynamics of Tree-Type Robotic Systems,* Springer, New York.

[9] **Murray R.M., Li Z., Sastry S.S.** (1994). *DA Mathematical Introcudtion to Robotic Manipulation,* CRC Press.

[10] **Shah S.V., Saha S.K., Dutt J.K.** (2012, April). *Denavit-Hartenberg Parameterization of Euler Angles,* Journal of Computational and Nonlinear Dynamics, vol. 7, 021006 1-10.

[11] **Hayat A.A., Chittawadigi R.G., Udai A.D., Saha S.K.** (2013, July). *Identification of Denavit-Hartenberg Parameters of an Industrial Robot,* Proceedings of Conference on Advances in Robotics, New York,USA.

[12] **Rocha C.R., Tonetto C.P., Dias A.** (2011, August). *A Comparison between the Denavit-Hartenberg and the Screw-Based Methods Used in Kinematic Modeling of Robot Manipulators,* Robotics and Computer-Integrated Manufacturing, 27, 723-728.

[13] **Piltan F., Soltani S.** (2011, September). *Design Artificial Nonlinear Controller Based on Computed Torque like Controller with Tunable Gain,* World Applied Sciences Journal, 14, 1306-1312.

[14] **Piltan F., Mirzaei M., Shahriari F., Nazari I., Emamzadeh S.** (2012). *Design Baseline Computed Torque Controller,* International Journal of Engineering, 6, 129-141.

[15] **Zubizarreta A., Marcos M., Cabanes I., Pinto C.** (2011, May). *A procedure to evaluate Extended Computed Torque Control configurations in the Stewart-Gough platform,* Robotics and Autonomous Systems, 59, 770-781.

[16] **Le T.D., Kang H.J., Suh Y.S., Ro Y.S.** (2013, September). *A online self-gain tuning method using neural networks for nonlinear PD computed torque controller of a 2-dof parallel manipulator,* Neurocomputing, 116, 53-61.

[17] **Piltan F., Keshavarz M., Badri A., Zargari A.** (2012). *Design novel Nonlinear Controller Applied to Robot Manipulator: Design New Feedback Linarization Fuzzy Controller With Minimum Rule Base Tuning Method,* International Journal of Robotics and Automation, vol.3, issue. 1.

[18] **Zelei A., Kovacs L.L., Stepan G.** (2011, May). *Computed torque of an under-actuated service robot platform modeled by natural coordinates,* Commun Nonlinear Sci Numer Simulat, 16, 2205-2217.

[19] **Behn C., Zimmermann K.** (2010, June). *Straight Worms under Adaptive Control and Friction: Adaptive Control,* IUTAM Symposium on Dynamics Modeling and Interaction Control in Virtual and Real Environments, 30, 65-72.

[20] **Özçelik S., Yeşiloğlu S.M., Temeltaş H.** (2006, June). *Direct Adaptive Control of Flexible Manipulators,* American Control Conference, Mineapolis, MN, USA.

[21] **Parlakçı M. N. A.** (2003). *Robust Variable Structure Controllers Design for Robot Manipulators with Parameter Perturbations.* Doktora Tezi, Boğaziçi Üniversitesi/Fen bilimleri Fakültesi, İstanbul.

[22] **ABB Robotics** (2010). *IRB 140.* ABB Flexible Automation

[23] **ABB Robotics** (2012). *Product Manual IRB 140.* ABB Flexible Automation

[24] **Karyot T. B.** (2010). Robotiğe Giriş Ders Notları.

[25] **Bingül Z. Ve Küçük S.** (2005). *Robot Tekniği.* Birsen Yayınevi.

[26] **Spong M. W., Hutchinson S., Vidyasagar M.** (2006). *Robot Modeling and Control.* John Wiley & Sons,Inc.

[27] **Öztürk M.** (2010). *Robotik Yapının Kinematik Modellenmesi.* Lisans Tezi, İTÜ, Uçak ve Uzay Bilimleri Fakültesi, İstanbul.

[28] **Ogata K.** (2002). *Modern Control Engineering*. Prentice Hall.

[29] **Yılmaz S.** (2013). PID Denetleyici Tasarımı Ders Notları.

[30] **DiCesare A.,Gustafson K.,Lindenfelzer P.** (2012). *Design Optimization of a Quad-rotor Capable of Autonomous Fligh.* Worcester Polytechnic Institute.

[31] **Krass M.** (2006). *PID Control Theory.*

[32] **Hoifodt H.** (2011). *Dynamic Modeling and Simulation of Robot Manipulators*. Norwegian University of Science and Technology.

[33] **Bingül Z., Küçük S.,** (2008). *Robot Dinamiği ve Kontrolü.* Birsen Yayınevi.

[34] **L.Lewis F., M.Dawson D., T. Abdallah C** (2004). *Robot Manipulator Control Theory and Practice.* Marcel Dekker Inc.

[35] **M. Muray R., Li Z., Shankar Sastry S.** (1994). *A Mathematical Introduction to Robotic Manipulation.* CRC Press.

[36] **Muray R. M., Li Z., Sastry S. S**. (1994). *A Mathematical Introduction to Robotic Manipulation.* CRC Press.

[37] **Hearne J.** (2009). *Posture Dependent Vibration Resistance of Serial Robot Manipulators to Applied Oscillating Loads.* Yüksek Lisans Tezi, University of Waterloo, Makine mühendisliği, Ontario, Canada.

[38] **Dorf R. C., Bishop R. H.** (2008). *Modern Control Systems.* (11th ed.).Pearson Prentice Hall.

[39] **Yüksek B.** (2013). *Modelling and Control of a Fixed Wing Unmanned Aerial Vehicle.* Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi/Mekatronik Mühendisliği Anabilim Dalı Mekatronik Mühendisliği Programı, İstanbul.

[40] **Saba A. B. M. A.** (1993). *Dynamic Analyzes And Synthesis of Multi Variable Control System for Robot with Cylindrical Coordinate*, Doktora Tezi, San Peterburg State Technical Universitesi, San Peterburg

Url-1 <*http://www2.omu.edu.tr/akademik_birimler/muhendislik/elektr*>, alındığı tarih: 18.03.2014.

Url-2 <*http://search.abb.com/library/Download.aspx?DocumentID*>, alındığı tarih: 18.03.2014.

EKLER

EK A.1

```
clc;
formatshort
syms pi11l2teta1teta2teta3teta4teta5teta6

a=    [70      360          0       0       0       0  ];
alfa=[pi/2     0            pi/2    -pi/2   pi/2    0  ];
d=    [352     0            0       380     0       65 ];
teta=[teta1    teta2+pi/2   teta3    teta4   teta5   teta6];

%1. koordinat sisteminden ana koordinat sistemine transformation
matrisi
H01=[cos(teta(1))  -sin(teta(1))*cos(alfa(1))
sin(teta(1))*sin(alfa(1))   a(1)*cos(teta(1));
sin(teta(1))  cos(teta(1))*cos(alfa(1))    -
cos(teta(1))*sin(alfa(1))  a(1)*sin(teta(1));
     0             sin(alfa(1))                cos(alfa(1))
d(1);
     0             0                           0
1 ]
%2. koordinat sisteminden ana koordinat sistemine transformation
matrisi
H12=[cos(teta(2))  -sin(teta(2))*cos(alfa(2))
sin(teta(2))*sin(alfa(2))   a(2)*cos(teta(2));
sin(teta(2))  cos(teta(2))*cos(alfa(2))    -
cos(teta(2))*sin(alfa(2))  a(2)*sin(teta(2));
     0             sin(alfa(2))                cos(alfa(2))
d(2);
     0             0                           0
1 ]
%3. koordinat sisteminden ana koordinat sistemine transformation
matrisi
H23=[cos(teta(3))  -sin(teta(3))*cos(alfa(3))
sin(teta(3))*sin(alfa(3))  a(3)*cos(teta(3));
sin(teta(3))  cos(teta(3))*cos(alfa(3))    -
cos(teta(3))*sin(alfa(3))  a(3)*sin(teta(3));
     0             sin(alfa(3))                cos(alfa(3))
d(3);
     0             0                           0
1 ]
%4. koordinat sisteminden ana koordinat sistemine transformation
matrisi
H34=[cos(teta(4))  -sin(teta(4))*cos(alfa(4))
sin(teta(4))*sin(alfa(4))   a(4)*cos(teta(4));
sin(teta(4))  cos(teta(4))*cos(alfa(4))    -
cos(teta(4))*sin(alfa(4))  a(4)*sin(teta(4));
     0             sin(alfa(4))                cos(alfa(4))
d(4);
     0             0                           0
1 ]
%5. koordinat sisteminden ana koordinat sistemine transformation
matrisi
H45=[cos(teta(5))  -sin(teta(5))*cos(alfa(5))
sin(teta(5))*sin(alfa(5))   a(5)*cos(teta(5));
sin(teta(5))  cos(teta(5))*cos(alfa(5))    -
cos(teta(5))*sin(alfa(5))  a(5)*sin(teta(5));
     0             sin(alfa(5))                cos(alfa(5))
d(5);
```

```
     0             0                              0
1 ]
%6. koordinat sisteminden ana koordinat sistemine transformation
matrisi
H56=[cos(teta(6))  -sin(teta(6))*cos(alfa(6))
sin(teta(6))*sin(alfa(6))   a(6)*cos(teta(6));
sin(teta(6))  cos(teta(6))*cos(alfa(6))    -
cos(teta(6))*sin(alfa(6))  a(6)*sin(teta(6));
     0             sin(alfa(6))                  cos(alfa(6))
d(6);
     0             0                              0
1 ]

H36=H34*H45*H56
H06=H01*H12*H23*H34*H45*H56
simplify(H06)
```

EK A.2

$T_{0,6} =$

Birinci satır: [(cos(t1)*sin(t3)*sin(pi/2 + t2) - cos(t1)*cos(t3)*cos(pi/2 + t2))*(sin(t4)*sin(t6) - cos(t4)*cos(t5)*cos(t6)) + sin(t1)*(cos(t4)*sin(t6) + cos(t5)*cos(t6)*sin(t4)) - cos(t6)*sin(t5)*(cos(t1)*cos(t3)*sin(pi/2 + t2) + cos(t1)*cos(pi/2 + t2)*sin(t3)), (cos(t1)*sin(t3)*sin(pi/2 + t2) - cos(t1)*cos(t3)*cos(pi/2 + t2))*(cos(t6)*sin(t4) + cos(t4)*cos(t5)*sin(t6)) + sin(t1)*(cos(t4)*cos(t6) - cos(t5)*sin(t4)*sin(t6)) + sin(t5)*sin(t6)*(cos(t1)*cos(t3)*sin(pi/2 + t2) + cos(t1)*cos(pi/2 + t2)*sin(t3)), cos(t5)*(cos(t1)*cos(t3)*sin(pi/2 + t2) + cos(t1)*cos(pi/2 + t2)*sin(t3)) + sin(t1)*sin(t4)*sin(t5) - cos(t4)*sin(t5)*(cos(t1)*sin(t3)*sin(pi/2 + t2) - cos(t1)*cos(t3)*cos(pi/2 + t2)), 70*cos(t1) + 360*cos(t1)*cos(pi/2 + t2) + (cos(t1)*cos(t3)*sin(pi/2 + t2) + cos(t1)*cos(pi/2 + t2)*sin(t3))*(65*cos(t5) + 380) + 65*sin(t1)*sin(t4)*sin(t5) - 65*cos(t4)*sin(t5)*(cos(t1)*sin(t3)*sin(pi/2 + t2) - cos(t1)*cos(t3)*cos(pi/2 + t2))]

İkinci satır: [- (sin(t4)*sin(t6) - cos(t4)*cos(t5)*cos(t6))*(cos(t3)*cos(pi/2 + t2)*sin(t1) - sin(t1)*sin(t3)*sin(pi/2 + t2)) - cos(t1)*(cos(t4)*sin(t6) + cos(t5)*cos(t6)*sin(t4)) - cos(t6)*sin(t5)*(cos(t3)*sin(t1)*sin(pi/2 + t2) + cos(pi/2 + t2)*sin(t1)*sin(t3)), sin(t5)*sin(t6)*(cos(t3)*sin(t1)*sin(pi/2 + t2) + cos(pi/2 + t2)*sin(t1)*sin(t3)) - cos(t1)*(cos(t4)*cos(t6) - cos(t5)*sin(t4)*sin(t6)) - (cos(t6)*sin(t4) + cos(t4)*cos(t5)*sin(t6))*(cos(t3)*cos(pi/2 + t2)*sin(t1) - sin(t1)*sin(t3)*sin(pi/2 + t2)), cos(t5)*(cos(t3)*sin(t1)*sin(pi/2 + t2) + cos(pi/2 + t2)*sin(t1)*sin(t3)) - cos(t1)*sin(t4)*sin(t5) + cos(t4)*sin(t5)*(cos(t3)*cos(pi/2 + t2)*sin(t1) - sin(t1)*sin(t3)*sin(pi/2 + t2)), 70*sin(t1) + 360*cos(pi/2 + t2)*sin(t1) + (cos(t3)*sin(t1)*sin(pi/2 + t2) + cos(pi/2 + t2)*sin(t1)*sin(t3))*(65*cos(t5) + 380) - 65*cos(t1)*sin(t4)*sin(t5) + 65*cos(t4)*sin(t5)*(cos(t3)*cos(pi/2 + t2)*sin(t1) - sin(t1)*sin(t3)*sin(pi/2 + t2))]

Üçüncü satır: [cos(t6)*sin(t5)*(cos(t3)* cos(pi/2 + t2) - sin(t3)*sin(pi/2 + t2)) - (sin(t4)*sin(t6) - cos(t4)*cos(t5)*cos(t6))*(cos(t3)*sin(pi/2 + t2) + cos(pi/2 + t2)*sin(t3)), - (cos(t6)*sin(t4) + cos(t4)*cos(t5)*sin(t6))*(cos(t3)*sin(pi/2 + t2) + cos(pi/2 + t2)*sin(t3)) - sin(t5)*sin(t6)*(cos(t3)*cos(pi/2 + t2) - sin(t3)*sin(pi/2 + t2)),

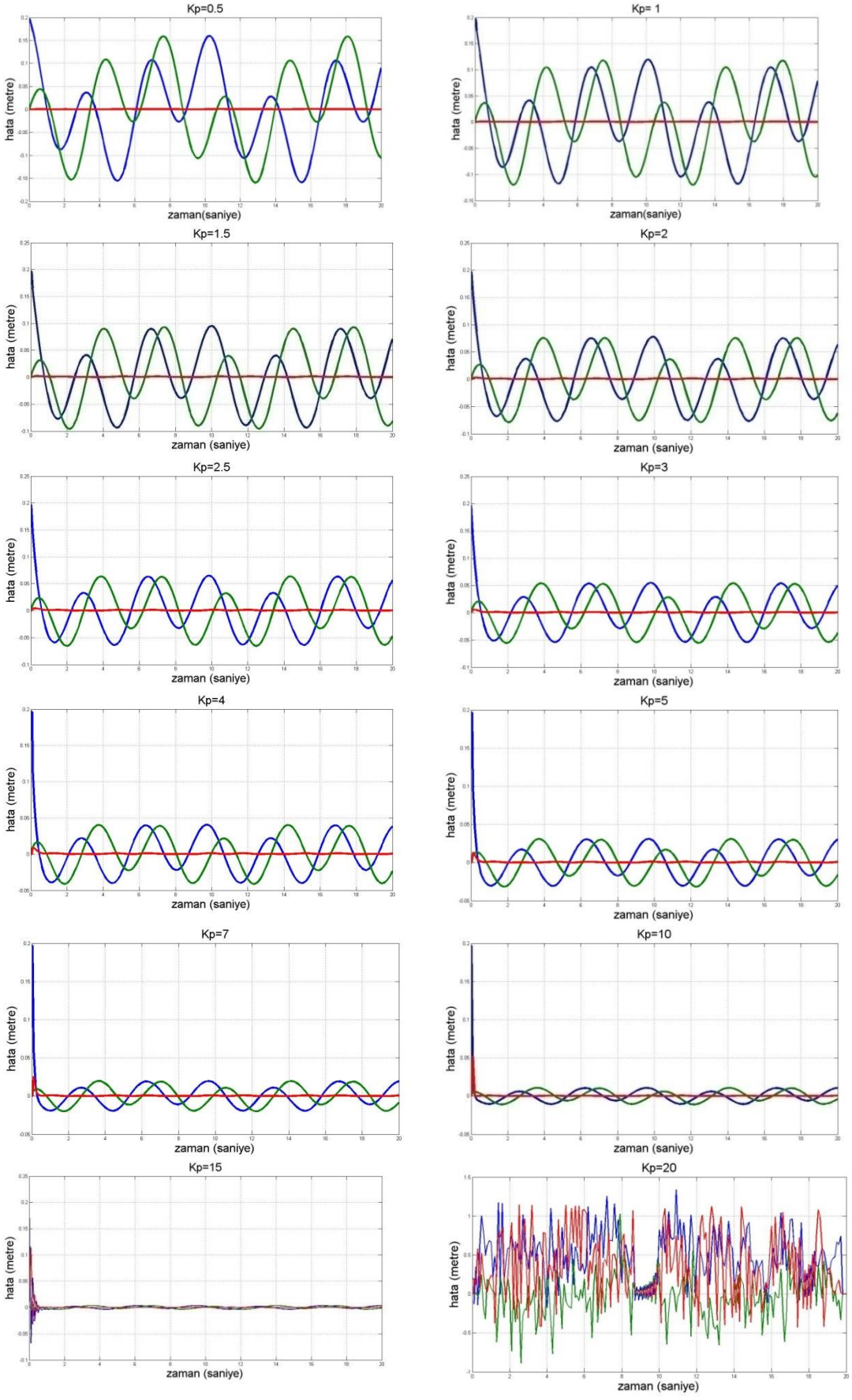
cos(t4)*sin(t5)*(cos(t3)*sin(pi/2 + t2) + cos(pi/2 + t2)*sin(t3)) - cos(t5)*(cos(t3)*cos(pi/2 + t2) - sin(t3)*sin(pi/2 + t2)), 360*sin(pi/2 + t2) - (65*cos(t5) + 380)*(cos(t3)*cos(pi/2 + t2) - sin(t3)*sin(pi/2 + t2)) + 65*cos(t4)*sin(t5)*(cos(t3)*sin(pi/2 + t2) + cos(pi/2 + t2)*sin(t3)) + 352]

Dördüncü satır: [0, 0, 0, 1]

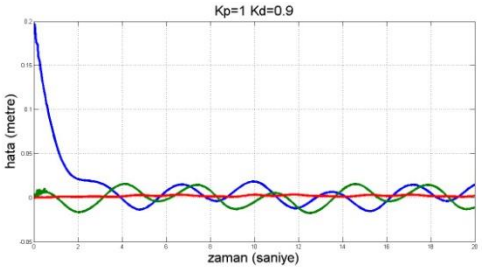
EK A.3

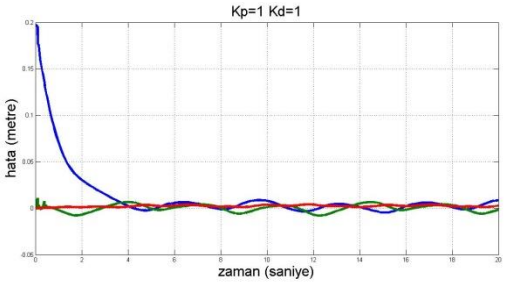
```
clear all;
clc;
global Jv Jw J Jvi H4 t1 t2 t3 t4 xc xd;

syms aa1a2a3a4a5a6dd1d2d3d4d5d6tetat1t2t3t4t5t6alfa;

A1=[cos(t1),  0 ,   sin(t1),  0.07*cos(t1);
sin(t1),   0 ,  -cos(t1),  0.07*sin(t1);
0  ,       1 ,         0,  0.352;
0  ,       0 ,         0,  1];

A2=[cos(t2+pi/2),  -sin(t2+pi/2),  0 ,   0.36*cos(t2+pi/2);
    sin(t2+pi/2),   cos(t2+pi/2),  0 ,   0.36*sin(t2+pi/2);
    0,          0,         1 ,   0;
    0,          0,         0 ,   1];

A3=[cos(t3),  0 ,  sin(t3),    0;
sin(t3),   0 ,  -cos(t3),   0;
0 ,        1 ,   0 ,        0;
0 ,        0 ,   0 ,        1];

A4=[cos(t4),  0 ,  -sin(t4),   0;
sin(t4),   0,   cos(t4),    0;
   0 ,       -1 ,  0  ,       0.445;
0 ,        0 ,  0 ,         1];

H4=A1*A2*A3*A4;

T1=A1;
T2=A1*A2;
T3=A1*A2*A3;
T4=A1*A2*A3*A4;

z0=[0 0 1]';  Q0=[0 0 0]';
z1=T1(1:3,3); Q1=T1(1:3,4);
z2=T2(1:3,3); Q2=T2(1:3,4);
z3=T3(1:3,3); Q3=T3(1:3,4);
z4=T4(1:3,3); Q4=T4(1:3,4);

J=[cross(z0,(Q4-Q0))  cross(z1,(Q4-Q1))  cross(z2,(Q4-Q2))
          z0                 z1                 z2       ]

  Jv=J(1:3,1:3);
  Jvi=subs(Jv,[t1 t2 t3],[0 0 0]);
  Jw=J(4:6,1:3);
```

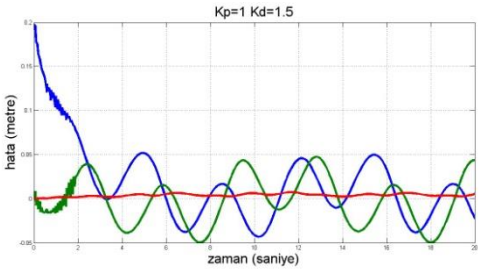
EK B.1

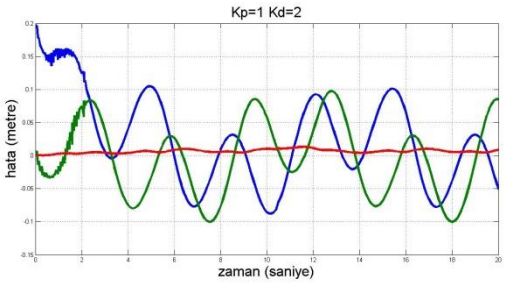

Kp=0.5
Kp= 1
Kp=1.5
Kp=2
Kp=2.5
Kp=3
Kp=4
Kp=5
Kp=7
Kp=10
Kp=15
Kp=20
hata (metre)
zaman(saniye)
zaman (saniye)

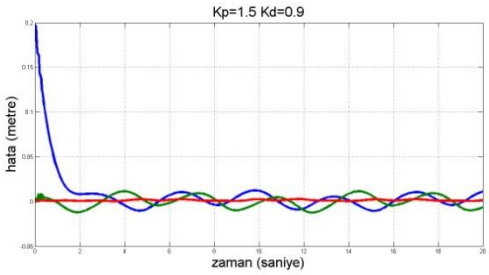
EK B.2

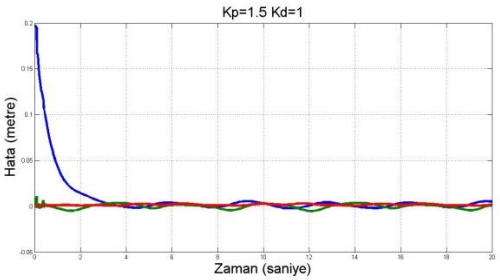

Kp=1 Kd=0.9
hata (metre)
zaman (saniye)

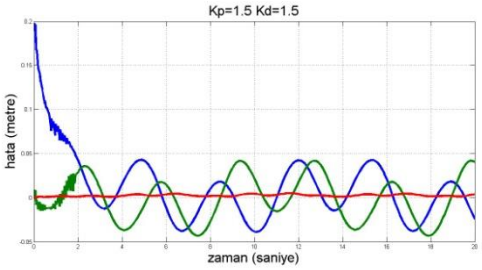

Kp=1 Kd=1
hata (metre)
zaman (saniye)

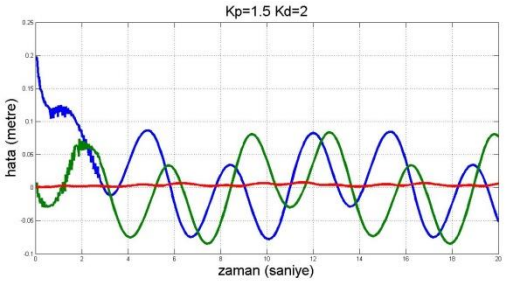

Kp=1 Kd=1.5
hata (metre)
zaman (saniye)

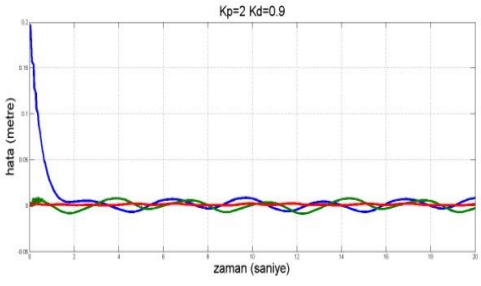

Kp=1 Kd=2
hata (metre)
zaman (saniye)

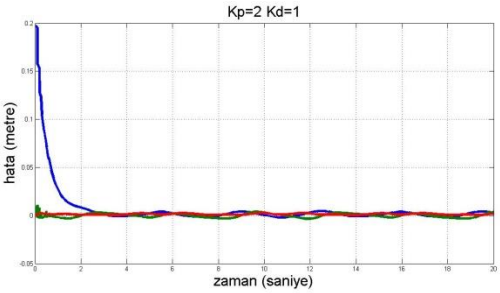

Kp=1.5 Kd=0.9
hata (metre)
zaman (saniye)

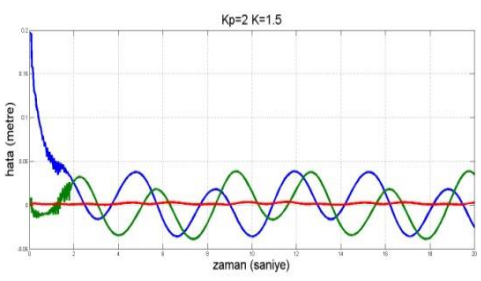

Kp=1.5 Kd=1
Hata (metre)
Zaman (saniye)

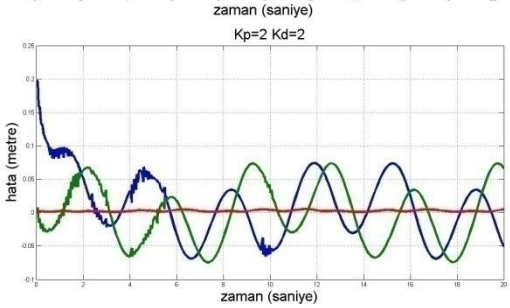

Kp=1.5 Kd=1.5
hata (metre)
zaman (saniye)

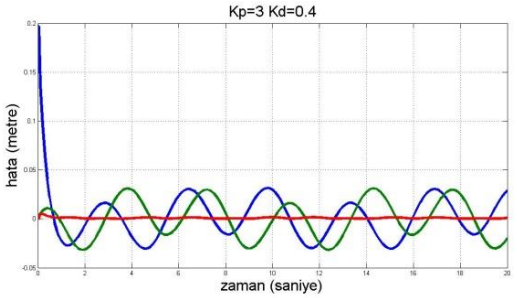

Kp=1.5 Kd=2
hata (metre)
zaman (saniye)

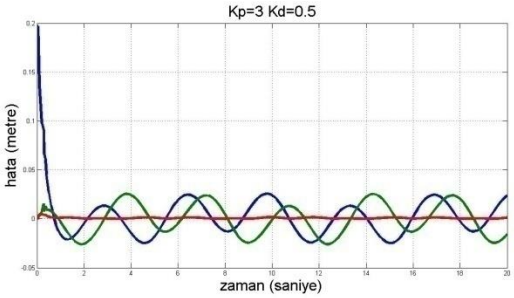

Kp=2 Kd=0.9
hata (metre)
zaman (saniye)

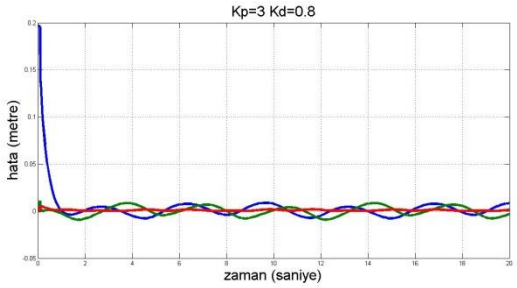

Kp=2 Kd=1
hata (metre)
zaman (saniye)

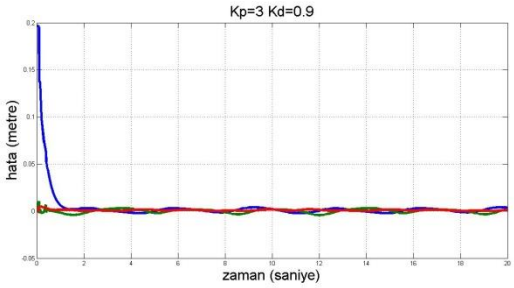

Kp=2 K=1.5
hata (metre)
zaman (saniye)

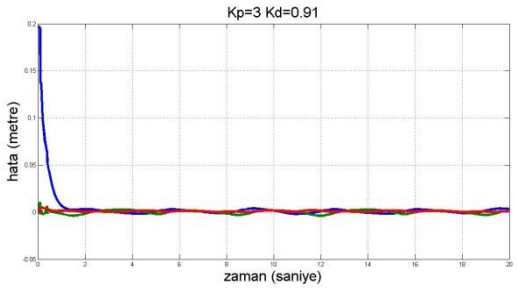

Kp=2 Kd=2
hata (metre)
zaman (saniye)

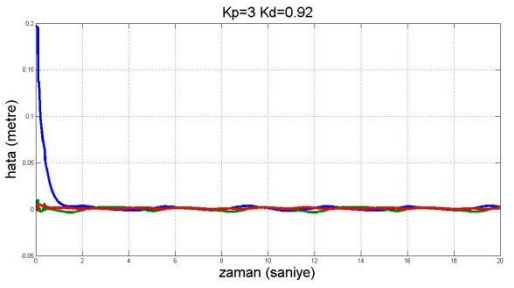

Kp=3 Kd=0.4
hata (metre)
zaman (saniye)

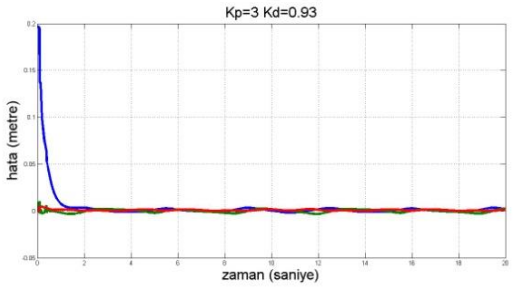

Kp=3 Kd=0.5
hata (metre)
zaman (saniye)

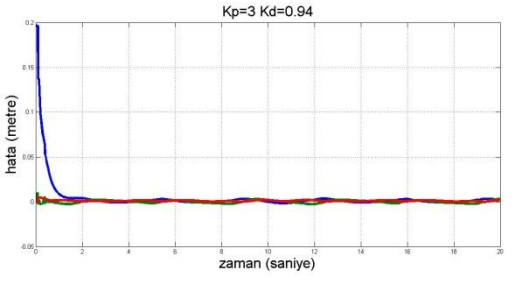

Kp=3 Kd=0.8
hata (metre)
zaman (saniye)

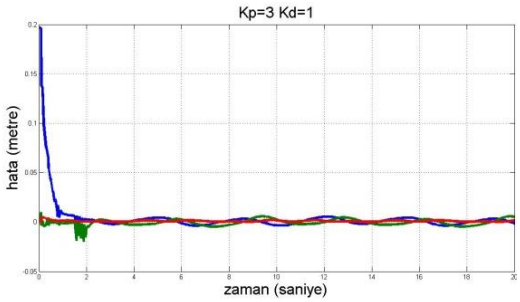

Kp=3 Kd=0.9
hata (metre)
zaman (saniye)

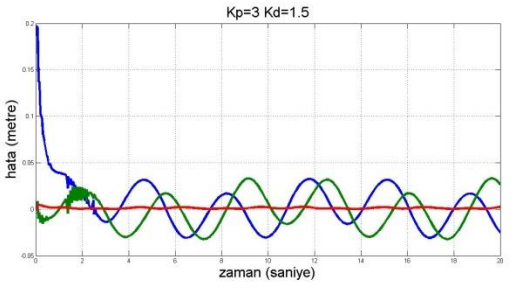

Kp=3 Kd=0.91
hata (metre)
zaman (saniye)

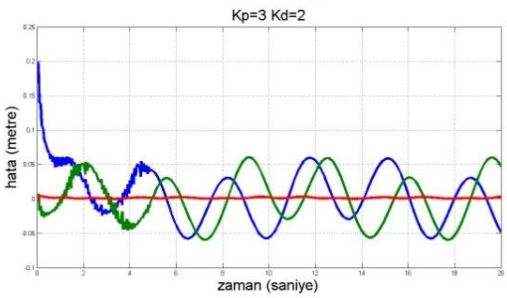

Kp=3 Kd=0.92
hata (metre)
zaman (saniye)

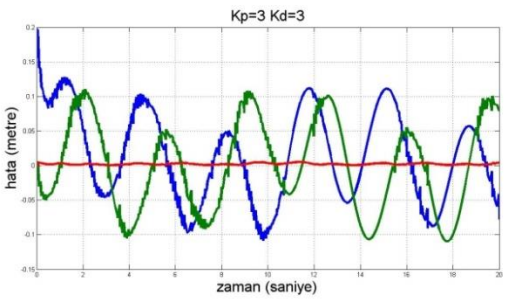

Kp=3 Kd=0.93
hata (metre)
zaman (saniye)


Kp=3 Kd=0.94
hata (metre)
zaman (saniye)


Kp=3 Kd=1
hata (metre)
zaman (saniye)


Kp=3 Kd=1.5
hata (metre)
zaman (saniye)


Kp=3 Kd=2
hata (metre)
zaman (saniye)


Kp=3 Kd=3
hata (metre)
zaman (saniye)

EK B.3

$$D(1,1) = \Big[0{,}0039508461887 * \sin(\theta_2 - \theta_3)^2 - 0{,}0039508461887 * \sin(\theta_3 - \theta_2)^2 - 0{,}032460998 * \sin\left(\theta_2 - \frac{\theta_3}{2}\right)^2 + 0{,}01973832 * \sin\left(\frac{\theta_2}{2}\right)^2 + 0{,}032460998 * \sin\left(\frac{\theta_3}{2}\right)^2 - 0{,}0039508461887 * \sin(-\theta_2 - \theta_3)^2 - 0{,}027669758 * \sin\left(-\theta_2 - \frac{\theta_3}{2}\right)^2 - 0{,}193688307 * \sin\left(\frac{\theta_2}{2} + \frac{\theta_3}{2}\right)^2 + 0{,}267057106 * \sin(\theta_2 + \theta_3)^2 + 0{,}0063118608 * sin(\theta_2 + \theta_3) + 1{,}55373705 * sin(\theta_2)^2 - 0{,}708730148 * \sin(\theta_2) - 0{,}284603227 * \sin(2 * \theta_2 + \theta_3) + 0.01449435526 * \sin(2 * \theta_2) + 0.0106049742112 \sin(2\theta_2 + 2\theta_3) + 0.2846032272 * \sin(\theta_3) + 0.9304452265964\Big]$$

$$D(1,2) = [0{,}0001789035408 * \cos(\theta_2 + \theta_3) + 0.0037496080872 * \sin(\theta_2 + \theta_3) - 0.120068205626 * \cos(\theta_2) - 0.01148423958 * \sin(\theta_2)]$$

$$D(1,3) = [0{,}0001789035408 * \cos(\theta_2 + \theta_3) + 0.0037496080872 * \sin(\theta_2 + \theta_3)]$$

$$D(2,1) = [0{,}0001789035408 * \cos(\theta_2 + \theta_3) + 0.0037496080872 * \sin(\theta_2 + \theta_3) - 0.120068205626 * \cos(\theta_2) - 0.02451576042 * \sin(\theta_2)]$$

$$D(2,2) = [0.5692064544 * \sin(\theta_3) - 0.0324609984 * \cos(\theta_3) + 1.9254673642204]$$

$$D(2,3) = [0.2846032272 * \sin(\theta_3) - 0.0162304992 * \cos(\theta_3) + 0.3004191253424]$$

$$D(3,1) = [0{,}00017890354 * \cos(\theta_2 + \theta_3) + 0.0037496080872 * \sin(\theta_2 + \theta_3)]$$

$$D(3,2) = [0.2846032272 * \sin(\theta_3) - 0.0162304992 * \cos(\theta_3) + 0.3004191253424]$$

$$D(3,3) = [0.3004191253424]$$

EK B.4

$$C(1,1) = [0.01449435526 * \dot{q}_2 * \cos(2\theta_2) + 0.776868525239 * \dot{q}_2 * \sin(2\theta_2) + 0.0003944913 * \dot{q}_2 * \cos(-\theta_2 - \theta_3) - 0.0177877017 * \dot{q}_2 * \cos(-2\theta_2 - \theta_3) + 0.0177877017 * \dot{q}_2 * \cos(2\theta_2 - \theta_3) + 0.0001126554441 * \dot{q}_2 * \cos(-2\theta_2 - 2\theta_3) + 0.0001126554441 * \dot{q}_2 * \cos(2\theta_3 - 2\theta_2) - 0.0001126554441 * \dot{q}_2 * \cos(2\theta_2 - 2\theta_3) + 0.0104923187671 * \dot{q}_2 * \cos(2\theta_2 + 2\theta_3) + 0.0003944913 * \dot{q}_3 * \cos(-\theta_2 - \theta_3) + 0.0001126554441 * \dot{q}_3 * \cos(-2\theta_2 - 2\theta_3) + 0.0001126554441 * \dot{q}_3 * \cos(2\theta_3 - 2\theta_2) - 0.0001126554441 * \dot{q}_3 * \cos(2\theta_2 - 2\theta_3) + 0.0104923187671 * \dot{q}_3 * \cos(2\theta_2 + 2\theta_3) - 0.00196899852175 * \dot{q}_2 * \sin(2\theta_3 - 2\theta_2) - 0.00196899852175 * \dot{q}_2 * \sin(2\theta_2 - 2\theta_3) - 0.135503976348 * \dot{q}_2 * \sin(2\theta_2 + 2\theta_3) - 0.00196899852175 * \dot{q}_3 * \sin(2\theta_3 - 2\theta_2) - 0.00196899852175 * \dot{q}_3 * \sin(2\theta_2 - 2\theta_3) - 0.135503976348 * \dot{q}_3 * \sin(2\theta_2 + 2\theta_3) + 0.0027614391 * \dot{q}_2 * \cos(\theta_2 + \theta_3) + 0.0027614391 * \dot{q}_3 * \cos(\theta_2 + \theta_3) - 0.0553395164 * \dot{q}_2 * \sin(\theta_2 + \theta_3) - 0.0553395164 * \dot{q}_3 * \sin(\theta_2 + \theta_3) - 0.354365074 * \dot{q}_2 * \cos(\theta_2) + 0.1423016136 * \dot{q}_3 * \cos(\theta_3) + 0.00493458 * \dot{q}_2 * \sin(\theta_2) + 0.0081152496 * \dot{q}_3 * \sin(\theta_3) - 0.0003944913 * \dot{q}_2 * \cos(\theta_2 - \theta_3) + 0.0003944913 * \dot{q}_2 * \cos(\theta_3 - \theta_2) - 0.0177877017 * \dot{q}_2 * \cos(\theta_3 - 2\theta_2) - 0.2668155255 * \dot{q}_2 * \cos(2\theta_2 + \theta_3) + 0.0003944913 * \dot{q}_3 * \cos(\theta_2 - \theta_3) + 0.0003944913 * \dot{q}_3 * \cos(\theta_3 - \theta_2) - 0.1423016136 * \dot{q}_3 * \cos(2\theta_2 + \theta_3) - 0.0162304992 * \dot{q}_2 * \sin(2\theta_2 + \theta_3) - 0.0081152496 * \dot{q}_3 * \sin(2\theta_2 + \theta_3)]$$

$$C(1,2) = [0.01449435526 * \dot{q}_1 * \cos(2\theta_2) + 0.776868525239 * \dot{q}_1 * \sin(2\theta_2) + 0.0106049742112 * \dot{q}_1 * \cos(2\theta_2 + 2\theta_3) - 0.135503976348 * \dot{q}_1 * \sin(2\theta_2 + 2\theta_3) + 0.0031559304 * \dot{q}_1 * \cos(\theta_2 + \theta_3) + 0.0037496080872 * \dot{q}_2 * \cos(\theta_2 + \theta_3) + 0.0037496080872 * \dot{q}_3 * \cos(\theta_2 + \theta_3) - 0.0553395164 * \dot{q}_1 * \sin(\theta_2 + \theta_3) - 0.0001789035408 * \dot{q}_2 * \sin(\theta_2 + \theta_3) - 0.0001789035408 * \dot{q}_3 * \sin(\theta_2 + \theta_3) - 0.354365074 * \dot{q}_1 * \cos(\theta_2) - 0.01148423958 * \dot{q}_2 * \cos(\theta_2) + 0.00493458 * \dot{q}_1 * \sin(\theta_2) + 0.120068205626 * \dot{q}_2 * \sin(\theta_2) - 0.2846032272 * \dot{q}_1 * \cos(2\theta_2 + \theta_3) - 0.0162304992 * \dot{q}_1 * \sin(2\theta_2 + \theta_3)]$$

$$C(1,3) = [0.0106049742112 * \dot{q}_1 * \cos(2\theta_2 + 2\theta_3) - 0.135503976348 * \dot{q}_1 * \sin(2\theta_2 + 2\theta_3) + 0.0031559304 * \dot{q}_1 * \sin(2\theta_2 + \theta_3) + 0.0037496080872 * \dot{q}_2 * \cos(\theta_2 + \theta_3) + 0.0037496080872 * \dot{q}_3 * \cos(\theta_2 + \theta_3) - 0.0553395164 * \dot{q}_1 * \sin(\theta_2 + \theta_3) - 0.0001789035408 * \dot{q}_2 * \sin(\theta_2 + \theta_3) - 0.0001789035408 * \dot{q}_3 * \sin(\theta_2 + \theta_3) + 0.1423016136 * \dot{q}_1 * \cos(\theta_3) + 0.0081152496 * \dot{q}_1 * \sin(\theta_3) - 0.1423016136 * \dot{q}_1 * \cos(2\theta_2 + \theta_3) - 0.0081152496 * \dot{q}_1 * \sin(2\theta_2 + \theta_3)]$$

$$C(2,1) = [0.2846032272 * \dot{q}_1 * \cos(2\theta_2 + \theta_3) + 0.0162304992 * \dot{q}_1 * \sin(2\theta_2 + \theta_3) - 0.01449435526 * \dot{q}_1 * \cos(2\theta_2) - 0.776868525239 * \dot{q}_1 * \sin(2\theta_2) - 0.0106049742112 * \dot{q}_1 * \cos(2\theta_2 + 2\theta_3) + 0.135503976348 * \dot{q}_1 * \sin(2\theta_2 + 2\theta_3) - 0.0031559304 * \dot{q}_1 * \cos(\theta_2 + \theta_3) + 0.0553395164 * \dot{q}_1 * \sin(\theta_2 + \theta_3) + 0.354365074 * \dot{q}_1 * \cos(\theta_2) - 0.00493458 * \dot{q}_1 * \sin(\theta_2)]$$

$$C(2,2) = [0.018 * \dot{q}_1 * \cos(\theta_2) + 0.2846032272 * \dot{q}_3 * \cos(\theta_3) + 0.0162304992 * \dot{q}_3 * \sin(\theta_3)]$$

$$C(2,3) = \left[\frac{88983 * (\dot{q}_1 + \dot{q}_2) * (1999 * \cos(\theta_3) + 114 * \sin(\theta_3))}{625000000}\right]$$

$$C(3,1) = [0.0003944913 * \dot{q}_1 * \cos(\theta_2 - \theta_3) - 0.0003944913 * \dot{q}_1 * \cos(\theta_3 - \theta_2) + 0.1423016136 * \dot{q}_1 * \cos(2\theta_2 + \theta_3) + 0.0081152496 * \dot{q}_1 * \sin(2\theta_2 + \theta_3) - 0.0003944913 * \dot{q}_1 * \cos(-\theta_2 - \theta_3) - 0.0001126554441 * \dot{q}_1 * \cos(-2\theta_2 - 2\theta_3) - 0.0001126554441 * \dot{q}_1 * \cos(2\theta_3 - 2\theta_2) - 0.0001126554441 * \dot{q}_1 * \cos(2\theta_2 - 2\theta_3) - 0.0104923187671 * \dot{q}_1 * \cos(2\theta_2 + 2\theta_3) + 0.00196899852175 * \dot{q}_1 * \sin(2\theta_3 - 2\theta_2) + 0.00196899852175 * \dot{q}_1 * \sin(2\theta_2 - 2\theta_3) - 0.0027614391 * \dot{q}_1 * \cos(\theta_2 + \theta_3) + 0.055339516400000 * \dot{q}_1 * \sin(\theta_2 + \theta_3) - 0.1423016136 * \dot{q}_1 * \cos(\theta_3) - 0.0081152496 * \dot{q}_1 * \sin(\theta_3)]$$

$$C(3,2) = \left[\frac{-88983 * \dot{q}_2 * (1999 * \cos(\theta_3) + 114 * \sin(\theta_3))}{625000000}\right]$$

$$C(3,3) = [0]$$

ÖZGEÇMİŞ

Ad Soyad: Muhammet ÖZTÜRK

Doğum Yeri ve Tarihi: Eyüp, İstanbul, 1988

Adres: İTÜ Uçak ve Uzay Bilimleri Fakültesi

E-Posta: ozturkmuh@itu.edu.tr

Lisans: Uzay Mühendisliği

Yayın ve Patent Listesi:

1. Muhammet Öztürk, Bahadır Koçer, Recep Uygun, **Design of SMA Based Actuators Used in Robotics**, İnternational Clawar Conference, University of Technology Sydney, Sydney, 14-17 Temmuz 2013
2. Muhammet Öztürk, Muhsin Hançer, M. Sefa Ulutaş, Murat Can, Musa Tartik, Adem Günel, Uğur Şahin, A. Duran Şahin, D. Eren Akyüz, **Bulanık Mantık Hesaplamalarına Dayalı Binalarda Isı Kayıp-Kazanç Yaklaşımı**, Bilimde Modern Yöntemler Sempozyumu, Dicle Üniversitesi Kongre Merkezi, İstanbul, 14-16 Ekim 2010
3. Nihat İlgen, Deniz Ballı, Muhammet Öztürk, Ekin Ceylan, M. Yasin Yalçın, Muharrem Ulu, Esra Çelik, S. Murat Yeşiloğlu, **Kinematics of Quadruped Walking on Discontinuous Terrain without a Priori Knowledge of the Terrain**, Uluslararası Clawar Konferansı, Boğaziçi Üniversitesi, İstanbul, 09-11 Ekim 2009