İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

YÜRÜYEN ROBOTLARIN KİNEMATİK VE DİNAMİK MODELLERİNİN MODÜLER YAKLAŞIM İLE ELDE EDİLMESİ

YÜKSEK LİSANS TEZİ

Süleyman Baran HEPGÜVEN

Kontrol ve Otomasyon Mühendisliği Anabilim Dalı

Kontrol ve Otomasyon Mühendisliği Programı

Eylül 2016

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

YÜRÜYEN ROBOTLARIN KİNEMATİK VE DİNAMİK MODELLERİNİN MODÜLER YAKLAŞIM İLE ELDE EDİLMESİ

YÜKSEK LİSANS TEZİ

Süleyman Baran HEPGÜVEN

(504131127)

Kontrol ve Otomasyon Mühendisliği Anabilim Dalı

Kontrol ve Otomasyon Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Hakan TEMELTAŞ

Eylül 2016

İTÜ, Fen Bilimleri Enstitüsü'nün 504131127 numaralı Yüksek Lisans Öğrencisi Süleyman Baran HEPGÜVEN, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı "YÜRÜYEN ROBOTLARIN KİNEMATİK VE DİNAMİK MODELLERİNİN MODÜLER YAKLAŞIM İLE ELDE EDİLMESİ" başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

Tez Danışmanı : **Prof. Dr. Hakan TEMELTAŞ**

İstanbul Teknik Üniversitesi

Jüri Üyeleri : **Yrd. Doç. Dr. Sıddık Murat YEŞİLOĞLU**

İstanbul Teknik Üniversitesi

Yrd. Doç. Dr. Ersan OĞUZ

Hava Harp Okulu

Teslim Tarihi : 5 Eylül 2016

Savunma Tarihi : 28 Eylül 2016

Aileme,

ÖNSÖZ

Tez teori ve simülasyon çalışmaları sırasında sürekli olarak desteğini esirgemeyen danışmanım sayın Prof. Dr. Hakan TEMELTAŞ'a teşekkürlerimi sunarım. Ayrıca moral ve motivasyonumu yüksek tutmama yardımcı olan aileme de teşekkür ederim.

Eylül 2016

Süleyman Baran Hepgüven
Kontrol Mühendisi

İÇİNDEKİLER

KISALTMALAR

ZMP	**:** Zero Moment Point
IPM	**:** Inverted Pendulum Model
DeNOC	**:** Decoupled Natural Orthogonal Complement
GIM	**:** Generalized Inertia Matrix
DOF	**:** Degree of Freedom
DH	**:** Denavith Hartenberg
NE	**:** Newton Euler

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

YÜRÜYEN ROBOTLARIN KİNEMATİK VE DİNAMİK MODELLERİNİN MODÜLER YAKLAŞIM İLE ELDE EDİLMESİ

ÖZET

Yürüyen robotlar uzun yıllardan beri araştırma konusu olmuştur. Mobil robotlardan farklı olarak, yürüyen robotlar canlı uzuvlarına benzer yapılar içeren robot ailesini temsil etmektedir. Örnek olarak; İki ayaklı yürüyen robot, dört ayaklı yürüyen robot, sekiz ayaklı yürüyen robot gibi yapılar söylenebilir. Bu tip robotlar zorlu ve engebeli arazi koşullarında hareket edebilme kabiliyetine sahip olduğu için uygulama alanları da çok çeşitli olabilmektedir. Mesela askeri amaçla kullanımda ağır askeri mühimmatı engebeli arazi de taşıma işlemi, günlük yaşamda mobil robotların hareket etmesinin kısıtlı kaldığı merdiven, basamak, tümsek gibi yapılarda ilerleyebilme kabiliyeti gibi özellikler yürüyen robotların kullanım alanı olarak verilebilecek en basit ve ilk akla gelen örneklerdir.

Yürüyen robotlar, endüstriyel tek zincirli robotlara ve mobil robotlara göre oldukça karmaşık kinematik yapılara sahiptir. Ayrıca bu tip robotlar çok fazla serbestlik derecesine sahip olmaktadır. Örneğin İki ayaklı yürüyen robot 12 adet serbestlik derecesine sahiptir. Bacak sayısı arttıkça serbestlik derecesi de artmaktadır. Bu yapılar göz önünde bulundurulduğunda klasik yöntemler hem yetersiz hem de işlem gücü açısından maliyetli olmaktadır. Bu problemlerin önüne geçmek için özyineleme dinamik modelleme yöntemlerine başvurulmaktadır. "Spatial Operator Algebra" olarak Featherstone tarafından ileri sürülen modelleme yöntemi temelinde Newton-Euler dinamiğine dayanmaktadır. Bu modelleme tekniği daha çok tek zincirli yapılarda kullanılmıştır. Ağaç yapılı karmaşık yapılara da uygulaması bulunmaktadır. Ancak paralel kollara sahip robotlar için sistematik bir yaklaşım sunmamaktadır.

Öne sürülen en son yöntemde Saha, robotik yapıları modüller ve modüllerin içinde bulunan linkler olarak modellemekte ve böylece sistematik bir biçimde karmaşık yapılı kinematik yapıların modellenmesini göstermektedir.

Robot kinematik modellemesi bu yöntemlerde hız domeninde gerçekleştirilmekte ve ileri yani tabandan uca doğru bir özyineleme ile çözülmektedir. İleri kinematik çıktısı olarak "Twist" ismi verilen eklemlere ait 6-boyutlu vektörler hesaplanmaktadır. Bu vektörler hem lineer hem de açısal hız vektörlerini içermekte ve eklemin sabit olarak belirlenmiş eksen takımına göre veya link koordinatlarına göre tanımlanabilmektedirler.

Dinamik modelleme iki aşamadan meydana gelmektedir: İleri ve ters dinamik. Ters dinamik kontrol amacıyla kullanılırken, ileri dinamik robot simülasyonu için kullanılmaktadır. Ters dinamik, doğrudan Newton-Euler dinamiği ile elde edilirken,

ileri dinamik biraz daha işlem gerektirmektedir. Bunun nedeni ileri dinamik hesaplamasında robot eylemsizlik matrisinin tersinin gerekmesidir. Matris tersi alma işlemi nümerik ya da analitik olarak hesaplanabilmektedir. Nümerik ters alma daha fazla işleme neden olurken, analitik ters alma işlemi daha hızlı bir şekilde sonuçları vermektedir.

Dinamik modelleme, modül seviyesinde kalınarak çözülebileceği gibi, link seviyesinde işlemler gerçekleştirilerek de hesaplanabilir. Hem ters hem de ileri dinamik için modül seviyesinde tüm matrisleri oluşturup işleme sokmak, link seviyesindeki özyineleme şemasını ortadan kaldırarak hesaplama maliyetini oldukça fazla arttırmaktadır. Öte yandan aynı sonucu elde eden yapıyı link seviyesine kurmak, hesaplamada ortaya çıkan matris işlemlerini, link seviyesinde özyinelemeyi beraberinde getirdiğinden önemli ölçüde azaltmaktadır.

Elde edilen ileri ve ters dinamik modeller, örneklerle uygulamaya dönüştürülmüştür. Bu uygulamalar için geliştirilen dinamik denklemler MATLAB ortamında oluşturulmuştur. Oluşturulan algoritmaların testi ise güvenilirliği endüstride uzun yıllar kullanılarak tecrübe edilmiş dinamik çözüm yapabilen yazılımlarla sağlanmıştır. Bu çalışmada özellikle çok gövdeli dinamik konusuna ağırlık veren ADAMS programı karşılaştırma için seçilmiştir. Tezde bulunan tüm örneklerin ileri ve ters dinamik simülasyonlarının sağlaması ADAMS ile yapılmıştır.

Yürüyen robotların yörünge planlaması düzgün ve doğal bir hareket elde etmek için gereklidir. Robotların hareketinde hedeflenen, gövde olarak belirlenen modülün belirli bir yüksekte istenen bir hızda istenen koordinata tekrarlı adımlarla gitmesidir. Gövde için belirlenen yörüngeye ek olarak ayakların nasıl hareket etmesi gerektiği de ayak yörüngeleri ile belirlenmelidir. Bu yörünge, ayağın adım atma esnasında ne kadar yükseleceği, ne kadar uzunlukta adım atılacağı parametrelerini içermektedir. Gövde yörüngesi “ZPM” prensibi ile hesaplanabilmektedir. Daha basitleştirilmiş hali olan “IPM” yöntemi ile yörünge hesaplanması daha hızlı bir biçimde yapılabilmektedir. Ayak yörüngeleri ise trigonometrik fonksiyonlarla hesaplanmaktadır.

KINEMATICAL AND DYNAMICAL MODELING OF WALKING ROBOTS USING MODULAR APPROACH

SUMMARY

Walking robots are interested in researchers for many years. They are different from mobile robots which have wheels to move around, and represent a robotic family that have structures like human or animal limbs. For instance; Biped, Quadruped and Hexapod. This type of robots are practicable to many areas since they are capable of moving at very tough and rugged lands. The process of carrying heavy military ammos at tough lands, the capabilty of moving at stairs, basements, bumps like structures where mobile robots have restricted movement capabilities are two simple examples for walking robots.

Unlike the industrial single chain low degree of freedom robots and mobile robots, walking robots have very complex kinematical chains and number of degree of freedom is very high. A two legged spatial biped has 12 degree of freedom and when the number of leg is increased, degree of freedom reaches very high numbers. As a result, classical approaches are not suitable and have high cost to solve dynamical equations for this type of structures. To overcome these problems, recursive dynamical modelling algorithms come forward. A systematic approach called "Spatial Operator Algebra" which is proposed by Featherstone is built based on Newton-Euler dynamics. This modeling approach is applied single chain robots mostly. "Spatial Operator Algebra" is applied for tree-structure complex robots however it does not contain systematic approach to complex robots which have parallel branches.

As most recent approach which is proposed by Saha, robots are seperated to modules and each module have links that form a single chain. By using Saha's modelling technic, complex robot kinematical and dynamical equation are calculated in a systematic way and in a recursive manner.

Robot kinematical model is calculated in velocity domain in explained methods, and solved from base to tip recursively. 6-dimension "Twist" vectors which belongs to joints, are output of the forward kinematics. "Twist" vectors include both linear and angular velocity vectors and can be defined at fixed inertial coordinate frame or at link coordinate frames.

Dynamical modeling have two branches: Forward Dynamics and Inverse Dynamics. Inverse dynamics is usefull as control purposes, on the other hand, forward dynamics is applied for robot simulation. Inverse dynamics is directly calculated from Newton-Euler Equations, however forward dynamics is much more complicated since it requires the inversion of the robot inertia matrix. The inversion of the inertia matrix can be calculated in a numerical or analytical way. Numerical inversion is not a cost

effective way since robot inertia matrix is a sparce matrix. However analytical approach is more cost effective method.

Dynamical models can be solved in two ways: at module level and link level. If all huge-sized matrices at module level is built seperately and is processed them at module level dynamical equations, computational effort increases while vanishin the link level recursive scheme. On the other hand, if the structure that reaches same results is built at link level, matrix operations which appears at calculation decreases since it brings link level recursion scheme as well.

Observed dynamical equations are simulated by applying them to examples. MATLAB environment is used to simulate and animate to examples. The developed dynamical models for examples can be tested with softwares which are capable of solving the rigid body dynamics and are examined for long terms in industry. At this thesis, ADAMS software is selected to prove the accuracy of the developed dynamical equations since ADAMS is based on especially for "Multibody Dynamics" subject.

Walking robots are desired to move in a natural and straight trajectory. It is necessary that the body of the robot keeps in constant height form ground and moves forward or backward to target coordiantes at a desired speed with repetitive steps. In addition to body trajectory, foot trajectory also required to determine how feet behaves. Foot trajectory defines maximum height of the foot during step, longiness of the step size. The body trajectory is can be calculated by using "ZPM" principle. The simplest form of "ZPM" is called "IPM" method and it requires less computational time than "ZPM". On the other hand, foot trajectories are in trigonometric form.

1. GİRİŞ

Günümüzde robotlar endüstride ağır, rutin ve insan hayatını riske atabilecek işlerde yoğun bir şekilde kullanılmaktadır. Bu robotların neredeyse tamamı tek bir zincir halinde bulunmakta ve klasik yöntemlerle kinematik ve dinamik denklemleri elde edilebilmektedir. Robotları günlük hayata uyarlama çabası olarak isimlendirilebilecek yürüyen robotlar, endüstride kullanılan robotlara göre yapısal olarak daha karmaşık, çoğunlukla tek zincirli robotlara göre daha fazla serbestlik derecesine, birden fazla kinematik zincire, birden fazla serbestlik derecesi içeren eklemlere ve sabit olmayan bir tabana sahiptirler. Klasik yöntemlerden ziyade özyinelemeli yöntemle bu tip karmaşık ve çok serbestlik derecesine sahip robotların kinematik ve dinamik modellerini elde etmek, hem modellemeyi basitleştirmekte hem de gereksiz hesaplamaları ortadan kaldırarak hesaplama zamanını düşürmektedir[2,9].

Özyinelemeli yöntem, literatürde robot dinamiğinde çokça kullanılan ve kapalı formda olmayan yapıya sahiptir. Bu yapı temelde Newton-Euler dinamiğini içermektedir. Newton-Euler dinamiği, tüm sisteme ilişkin kuvvet ve moment vektörlerinin tamamını hesaplamaktadır. Bu sayede, bir sisteme hareket anında etkiyen kuvvetleri hesaplamada bu yapı doğrudan kullanılabilmektedir. Tüm kuvvetlerin hesaplanması sınırlandırılmış, yani harekete neden olmayan kuvvetlerin hesaplanmasını da beraberinde getirmektedir. Sınırlandırılmış kuvvetlerin eliminasyonunu hesaplamak sisteme ilişkin DeNOC matrisleri ile mümkün olmaktadır. Sonuç olarak robot dinamiği iki ana bölümü içermektedir: Newton-Euler dinamiği ve DeNOC konsepti. DeNOC matrisleri robotik sistemin kinematik analizi yapılarak elde edilmektedir[9].

Bu tezde ilgilenilen alan, robotların bir alt kümesi olan yürüyen robotlar ve modüler tabanlı kinematik ve dinamik modellemenin bu tip robotlar üzerinde uygulanmasıdır. Modellemeye ilişkin teori verildikten sonra bu teorinin simülasyonları basit robotik sistemler üzerinde simüle edilmiştir. Ardından daha kapsamlı bir robotik sistem üzerinde kinematik ve dinamik modellerin simülasyonları gerçekleştirilmiştir.

1.1. Literatür Araştırması

Özyinelemeli robot dinamiğine ilişkin olarak Featherstone [2], yayınlanan kitabında detaylı bir biçimde bu çalışmasını aktarmıştır. "Spatial" vektörleri kullanan Featherstone robotik sistemlere ilişkin hem ters hem ileri dinamik denklemleri, özyinelemeli yöntemi kullanarak vermiştir. Ayrıca kullandığı yöntem açık zincirlere ek olarak hem kapalı zincirlere hem de ağaç yapılı sistemlere uygulanabilmektedir. Tüm işlemleri link seviyesinde gerçekleştirmektedir.

Featherstone'un çalışmalarını eksik tahrikli sistemlere uygulayan Jain ve Rodriguez [4], çalışmalarında "Spatial Operator" algoritmalarını O(n) mertebesinde tutarak etkili dinamik model algoritmaları oluşturmuşlardır. Buna benzer bir çalışma Yeşiloğlu ve Temeltaş[12] tarafından uzay manipülatörleri üzerinde gösterilmiştir. Bu çalışmada iki uzay manipülatörünün ortak bir yükü taşıması üzerine yoğunlaşılmış ve "Spatial Operator" yöntemiyle O(n) mertebesinde etkili algoritmalar ile simülasyonlar gerçekleştirilmiştir.

Saha, Shah ve Dutt [9], Featherstone'un link seviyesinde açıklamış olduğu robot dinamiğini modüler hale getirerek, özellikle birden fazla kinematik zincire sahip olan robotik sistemlere sistematik bir yaklaşım getirmişlerdir. Ayrıca bu yapı yürüyen robotlara ve kapalı zincirli sistemlere de aynı kaynakta uygulanmış ve simülasyon sonuçları verilmiştir.

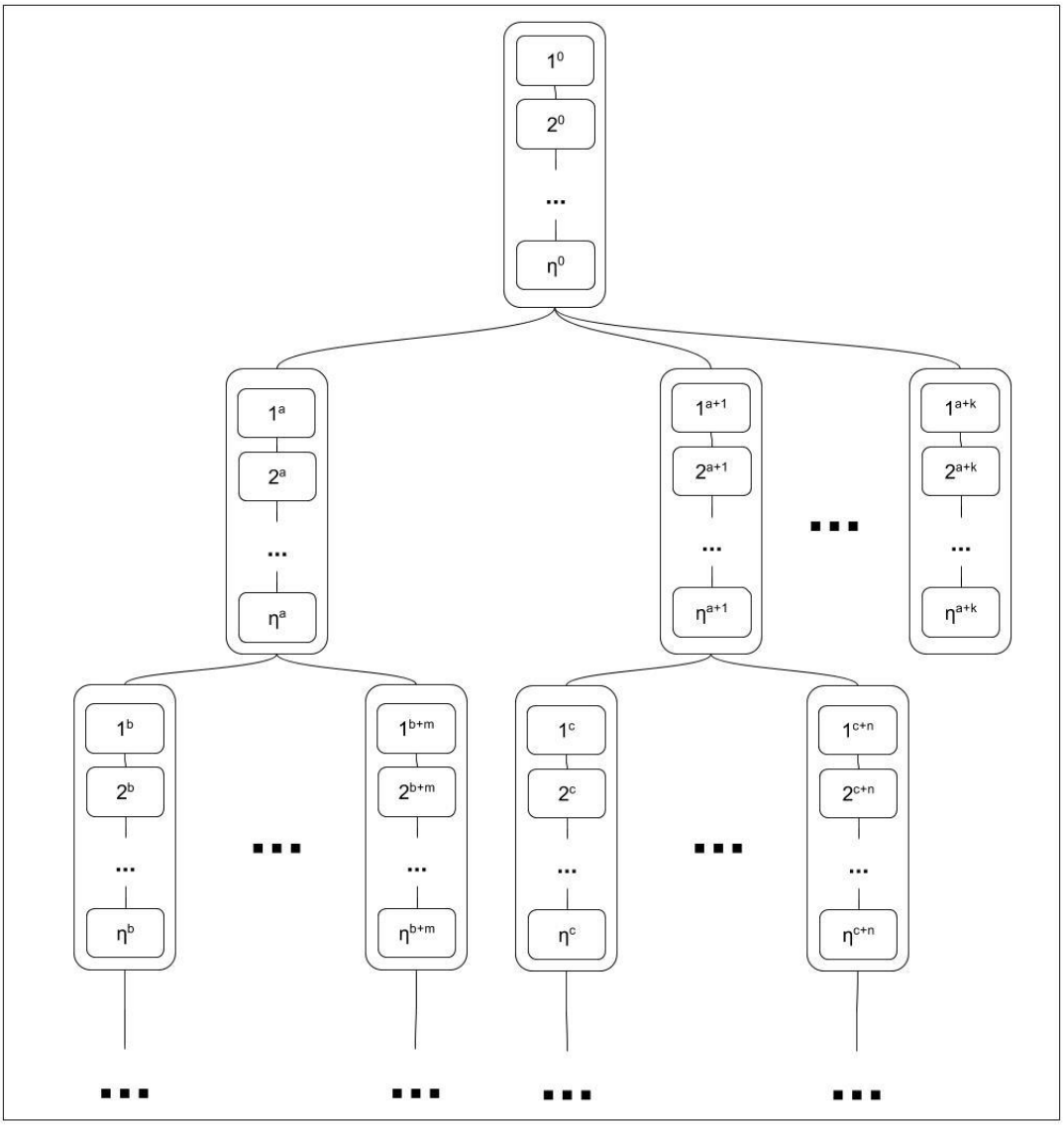
Yürüyen robotların dinamiğine ilişkin olarak literatürde çeşitli yöntemler bulunmaktadır. Bunlar genel olarak ikiye ayrılabilir: robotun konfigürasyonuna bağımlı ve robotun konfigürasyonundan bağımsız. Robotun anlık konfigürasyonuna bağlı robot dinamiğinde, taban serbest olarak alınmamakta bu nedenle robot hareket ettikçe oluşan kapalı ve açık zincir konfigürasyonları için ayrı ayrı modelleme yapılmasını zorunlu kılmaktadır[9]. Shih [8], iki aşamalı olarak oluşturduğu robot dinamik modelinde, taban olarak yürüyen robotun gövdesini değil, ayaklardan bir tanesini seçmiştir, bu seçim doğal olarak sistemi sabit tabanlı olarak modellemesine yol açmış ve sonuç olarak hareketten kaynaklı açık ve kapalı zincir yapıları göz önünde bulundurmak durumunda kalmıştır. Robotun konfigürasyonundan bağımsız bir şekilde dinamik modelin elde edilmesi, tabanın hareketli gövde olarak seçilmesiyle ve ayak-yer ilişkisinin tanımlanmasıyla mümkün olmaktadır[9,2].

Vukobratovic[10], “insansı” olarak isimlendirdiği yürüyen robotu “uçan” olarak tanımlamış ve bu kabul ile özyinelemeli dinamik modeli geliştirmiştir.

Yürüyen robotların insansı hareket etmesini sağlamak amacıyla ters kinematiğine yönelik yapılan çalışmalar bulunmaktadır. Vukabratovic[11] öne sürdüğü “Sıfır Moment Noktası” yöntemi ile özellikle iki ayaklı robotların hareket esnasındaki dengelenme problemine çözüm bulmuştur. Öne sürülen yönteme göre robotun ağırlığı ve hareketinden kaynaklı olarak oluşan kuvvet ve momentler bir noktada sıfırlanmalıdır. Bu nokta eğer robotun temas eden ayağının sınırları içerisinde kalıyorsa, robot dengede kalacaktır. Ancak bu nokta dışarda kalıyorsa, robotun dengesi de bozulacaktır. Vukobratovic’in öne sürdüğü yöntemin basitleştirilmiş hali “Ters Sarkaç Model” olarak literatüre geçmiştir[5]. Kajita ve Toni[5], “Ters Sarkaç Model” ile yürüyen robotların tüm kütlesini gövdede toplanmış olarak varsaymış, geriye kalan uzuvların kütlelerini ihmal etmiştir. Bu durum hesaplama açısından Vukobratovic’in yöntemine göre daha hızlıdır[9]. Çünkü sıfır moment noktasının hesaplanması diferansiyel denklemlerin her tekrarlamada çözülmesi ile gerçekleşmekte fakat “Ters Sarkaç Model” yaklaşımında bu denklemler bir sefer çözülerek yürüyen robotu dengede tutacak gövde yörüngesi hesaplanabilmektedir[9]. Gövde yörüngesinin yanında ayak yörüngesi de yürüyen robotun düzgün bir şekilde hareket edebilmesi için gereklidir. Saha, Shah ve Dutt [9], bu yörüngeyi trigonometrik fonksiyonlarla ifade etmişlerdir. Gövde yörüngesi ve ayak yörüngeleri zamana bağlı fonksiyonlar olarak elde edildiğinden, bu yörüngeler senkron bir biçimde hareket edecektir.

1.2. Takip Edilen Yöntemler

Yürüyen robotlar, ağaç yapılı robotların bir alt konusu olarak görüldüğü takdirde, modül tabanlı modelleme teknikleri için uygun bir forma sahip olmaktadır. Uygun form ile söylenmek istenen, yürüyen robotun serbest bir gövdeye sahip olmasıdır. Gövdenin serbest olarak belirlenmesi, Saha, Shah ve Dutt’un kullandığı özyinelemeli modüler tekniğin uygulanmasını kolaylaştırmaktadır. Bu yöntemde robotik sistem modüllere ayrıştırılmakta ve her modül içerisinde tek bir zincir barındırmaktadır. Şekil 1.1 modül yapısını göstermektedir.



Şekil 1.1 : Ağaç yapılı robotik sistemin modüler gösterimi

Kinematik modelleme "Twist" vektörlerinin hesaplanmasıyla gerçekleştirilmektedir. "Twist" vektörleri Saha, Shah ve Dutt tarafından öne sürülmüş ve bir rijit cismin uzaydaki lineer ve dönel hızlarını içeren vektörlerdir. Her ekleme ait "Twist" vektörü tabandan uç noktaya doğru hesaplanmaktadır. Bu yöntemin kötü yönlerinden bir tanesi, klasik yöntemde de karşılaşılan, ters kinematik hesaplanmasını standart bir formatta verememesidir. Bunun için ileri de yürüyen robot için ters kinematik bölümünde de bahsedilecek olan geometrik yaklaşımlar kullanılmaktadır.

Dinamik modelleme robotikte ikiye ayrılmaktadır. Ters dinamik modelleme olarak isimlendirilen ve eklem hareketine karşılık eklem torklarının hesaplanmasını sağlayan modelleme daha çok kontrol amaçlı kullanılırken, ileri dinamik modelleme simülatör amacıyla kullanılmaktadır. Ters dinamik modelleme Newton-Euler ve

DeNOC matrisleri ile doğrudan hesaplanabilirken, ileri dinamik modelleme robot eylemsizlik matrisinin tersinin alınmasını gerektirdiği için doğrudan hesaplanamaz[9]. Bu nedenle analitik veya nümerik olarak robot eylemsizlik matrisinin tersinin alınması için bir yöntem kullanılmalıdır. Saha, Shah ve Dutt bu konuyla ilgili olarak [9] nolu kaynakta bu yöntemi "GIM Ayrıştırması" olarak tanımlamışlardır. "GIM" robot eylemsizlik matrisini ifade etmekte ve bu yöntem sayesinde "GIM" üç matrisin çarpımı ile ifade edilmektedir. Bu matrisler üst üçgen ve köşegen yapıda olduğu için ters alma işlemi analitik yöntemle gerçekleştirilebilmektedir.

Yürüyen robotların yörünge planlaması gövde ve ayak yörüngelerinin planlanmasına dayanmaktadır. Gövde yörüngesinin planlanmasında "Ters Sarkaç Model" yöntemi bu çalışmada kullanılmaktadır. Bu yöntemde hedeflenen nokta gövdenin yörüngesini yürüyen robotun ağırlığından kaynaklı devrilmeyi engelleyecek şekilde belirlemektir. Ayak yörüngesi ise trigonometrik bir fonksiyon ile oluşturulmuştur[9].

Bölüm 2'de karmaşık robotik yapıların kinematik modellemesine ilişkin bilgiler verilmiştir. Robotik yapı burada modüller arası ve modül içi olarak iki aşamada incelenmiş ve kinematik modele ilişkin eşitlikler verilmiştir. Bölüm 3, dinamik modellemeyi konu almaktadır. Temeli Newton-Euler dinamiğine dayanan ve yine robotik yapıyı modüllere ayrıştırarak inceleyen bir dinamik modelleme yöntemini içerir. Ayrıca robotlarda ihtiyaç duyulan ters ve ileri dinamik modellerin elde edilmesi de yine bölüm 3'te gösterilmiştir. Biri sabit tabanlı diğeri sabit olmayan tabanlı iki örnek ile ileri ve ters dinamik modellemesinin basit formdaki simülasyonları gerçekleştirilmiştir. Bu örnekler düzlemsel yapıda olup sadece tek serbestlik dereceli eklemlerden oluşmaktadır. Bölüm 4, daha kapsamlı bir örneğin kinematik, ileri ve ters dinamik modellerinin simülasyonlarını ve doğruluklarını içermektedir. Bu örnek 3 boyutlu yapıda bulunan ve bir, iki ve üç serbestlik dereceli eklemler içeren yürüyen bir robottur. Son bölüm olan bölüm 5'te ise sonuçlar ve tezde işlenen konuyla ilgili ilerde gerçekleştirilmesi olası olan konular verilmiştir.

2. KİNEMATİK MODELLEME

Robotların yapısı karmaşıklaştıkça klasik yöntemler ya yetersiz kalmakta ya da modellemeyi karmaşık bir şekilde çözmektedir. Yürüyen robotlarda da bu durum söz konusudur. Karmaşık yapılı robotların kinematik ve dinamik modellemesini yapabilmek için robotu, modüller ve modüller içinde bulunan eklemler olarak tanımlamak, ortaya çıkan karmaşıklığı azaltmakta ve problemin çözümüne sistematik bir şekilde yaklaşabilmeyi sağlamaktadır. Buna yönelik olarak [9]'de bulunan ağaç yapılı modelleme yöntemi bu çalışmada kullanılmıştır.

Ağaç yapılı mimaride robotik sistem, modüllerin birleşiminden meydana gelmektedir[9]. Her modül seri bağlı linkleri içinde barındırmaktadır. Taban modülü sabit veya hareketli olarak belirlenir ve "0" nolu modül olarak isimlendirilir. Diğer modüller bu modüle direk ya da dolaylı olarak bağlıdır. Modül tanımı için aşağıdaki özellikler sağlanmalıdır:

- Her modül seri bağlı linkleri içermelidir.
- Her modül bir önceki modülün son linkinden başlamalıdır.

Modüller arasındaki ilişki, ebeveyn-çocuk ilişkisi olarak tanımlanmaktadır. Buna göre seri bağlı iki modülden taban modülüne daha yakın olan ebeveyn, diğeri ise çocuk modülü olarak isimlendirilir. Notasyon karmaşasını engellemek için bu çalışmada kullanılan notasyon, [9] ile aynı tutulmuştur. Buna göre, modüller "M_i" ile belirtilmekte ve alt indis modül numarasını göstermektedir. Modül içinde bulunan eklemler şöyle numaralandırılır:

$$1^i, 2^i, \dots, k^i, \dots, \eta^i$$

Önceden de belirtildiği üzere eklemler birden fazla serbestlik derecesine sahip olabilir. Serbestlik derecesi sayısı "ε_k" sembolü ile belirtilir.

2.1. Modül İçi Hız Dağılımları

Kullanılan kinematik yaklaşım modeli, klasik yöntemde olduğu gibi eklemler arası pozisyon ilişkilerini değil, hız ilişkilerini vermektedir. Bu modelin anlaşılabilmesi için ilk olarak "Twist" vektörlerinin açıklanması gereklidir. 6 boyuta sahip bu vektör uzayda herhangi bir noktanın tanımlı olduğu koordinat eksenine göre açısal ve lineer hızlarını içerir. İlk üç elemanı açısal hız bileşenini, son üç elemanı ise lineer hızları içerir.

$$t_k = [\omega_k^T \quad \dot{o}_k^T]^T \tag{2.1}$$

2Açısal hız vektörünün 2-norm'u açısal hızı verirken, bu vektör yönündeki birim vektör ise dönme eksenini verir.

Birbirine bağlı eklem yapısında bulunan robotik sistemlerde, bu yönteme göre her eksene bir twist vektörü tanımlanır. Hız domeninde sisteme yaklaşıldığında, alt yani tabana daha yakın olan eklemlerin hızları, kendilerine bağlı üst eklemleri etkilemektedir. Yani üst eklemlerin hızları, alt eklemlerin hızlarından etkilenmektedir. Bu ilişki, "Twist Yayılım Matrisi" konseptiyle tanımlanmaktadır[2,9].

Bu ilişkiyi açıklayan denklemleri verebilmek için şu tanımlama yapılabilir: #(k-1) nolu link ile #(k) nolu link, (k) nolu 1-DOF rotasyonel veya lineer hareket eden eklem ile birbirine bağlansın.

Bu durumda #(k) nolu linkin "Twist" vektörü bir önceki linke göre hesaplanması denklem 2.2'de belirtilmiştir.

$$t_k = A_{k,k-1}\, t_{k-1} + p_k \dot{\theta}_k \tag{2.2}$$

Bu denkleme ait terimleri açıklarsak, "t" ile belirtilen vektörler indislerinin belirlediği Twist vektörleridir. A terimi 6x6 boyutunda Twist yayılım matrisi olarak isimlendirilen matristir. "p" vektörü eklem hareket eksenini veren hareket yayılım vektörü olarak isimlendirilen 6x1 boyutunda bir vektördür. Bu vektörün ilk üç terimi rotasyonel, son üç terimi lineer hareket eksenini tanımlamaktadır. Son olarak "$\dot{\theta}$" eklem açısal ya da lineer pozisyon değişimini temsil eder.

Twist yayılım matrisi, bir vektöre bağlı olarak oluşturulur. Bu vektör iki ardışıl link arasında tanımlanmıştır. Linklere ait koordinat eksenlerinin merkezleri arasında uzanan bu vektörün yönü sonraki linkten önceki linke doğrudur. Bu vektör DH parametreleri ile ifade edilmektedir:

$$\boldsymbol{a}_{k,k-1} = -\boldsymbol{a}_{k-1,k} = [-a_k \quad b_k \sin(\alpha_k) \quad -b_k \cos(\alpha_k)]^T \tag{2.3}$$

Twist yayılım matrisi ise bu vektöre göre denklem 2.4'deki gibi oluşturulur.

$$A_{k,k-1} = \begin{bmatrix} 1_{3x3} & 0_{3x3} \\ \boldsymbol{a}_{k,k-1} \times & 1_{3x3} \end{bmatrix} \tag{2.4}$$

İki link arasında tanımlanan bu ifadeleri genelleştirerek seri bağlı linkler içeren bir modüle ait genelleştirilmiş "Twist" vektörü elde edilebilir. "i" nolu modüle ilişkin "Twist" vektörleri aşağıdaki gibi elde edilir.

Modülün ilk eklemi için "Twist" eşitliği yazıldığı takdirde bir önceki modülün son linki ile bağlantılı olduğu için, bu link ile önceki modülün son linki arasında bulunan Twist yayılım matrisi kullanılır.

$$t_{1^i} = A_{1^i,l}\, t_l + p_{1^i}\, \dot{\theta}_{1^i} \tag{2.5}$$

Geri kalan eklemler modülün içinde birbirleriyle seri olarak bağlı olduğundan Twist yayılım matrisleri modül içindeki vektörlerle tanımlanmaktadır. "2", "k" ve "η" nolu eklemlere ilişkin "Twist" vektörleri şu şekilde elde edilir.

$$t_{2^i} = A_{2^i,1^i}\, t_{1^i} + p_{2^i}\, \dot{\theta}_{2^i} \tag{2.6a}$$

$$t_{k^i} = A_{k^i,k-1^i}\, t_{k-1^i} + p_{k^i}\, \dot{\theta}_{k^i} \tag{2.6b}$$

$$t_{\eta^i} = A_{\eta^i,\eta-1^i}\, t_{\eta-1^i} + p_{\eta^i}\, \dot{\theta}_{\eta^i} \tag{2.6c}$$

Anlaşılması açısından tek serbestlik dereceli eklemlerle oluşturulan bu ifadeler kompakt forma sokularak, tüm modüle ait "Twist" ifadesi oluşturulabilir.

$$\bar{t}_i = \bar{N}_i\, \dot{\bar{q}}_i \tag{2.7}$$

"$\bar{t}_i$" vektörü modülde bulunan tüm linklerin "Twist" vektörlerini içerirken, "$\bar{N}_i$" matrisi ise Twist yayılım matrisleri ve hareket yayılım vektörlerinden oluşur. "$\dot{\bar{q}}_i$ " ise eklem hızlarını barındırır.

$$\bar{t}_i = \begin{bmatrix} t_{1^i} \\ \vdots \\ t_{k^i} \\ \vdots \\ t_{\eta^i} \end{bmatrix}, \quad \bar{N}_i = \begin{bmatrix} p_{1^i} & 0 & & \cdots & & 0 \\ A_{2^i,1^i}\, p_{2^i} & p_{2^i} & 0 & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & & \vdots \\ A_{k^i,1^i}\, p_{1^i} & \cdots & A_{k^i,k-1^i}\, p_{k-1^i} & p_{k^i} & & \\ \vdots & \ddots & & \ddots & \ddots & 0 \\ A_{\eta^i,1^i}\, p_{1^i} & \cdots & & A_{\eta^i,k^i}\, p_{k^i} & & p_{\eta^i} \end{bmatrix} \tag{2.8}$$

$$\dot{\bar{q}}_i = \begin{bmatrix} \dot{\theta}_{1^i} \\ \vdots \\ \dot{\theta}_{k^i} \\ \vdots \\ \dot{\theta}_{\eta^i} \end{bmatrix}$$

Kompakt formda dikkat edilirse, modülün ilk linkine ilişkin açık form denklemindeki "$A_{1^i,l}\, t_l$" ifadesi kaybolmaktadır. Bu ifade bir sonraki bölümde anlatılacağı üzere modüller arası ilişkiyi tanımladığından, sonra eklenecektir.

Çok serbestlik dereceli eklemlerde iç hız dağılımı yine 1-DOF eklemli sistemlerdeki gibi yapılmaktadır. Fakat burada eklem içinde tüm eksenler tek bir noktada kesiştiği için "a" vektörleri "0" vektörlerine eşittir. Bu nedenle iç hız dağılımı yapılırken "a" vektörlerinden elde edilen Twist yayılım matrisleri birim matrise dönüşmektedir. Bu özellik göz önünde bulundurulduğunda denklem 2.9'daki gibi bir genel yaklaşım yapılabilir.

$$\begin{aligned} t_{k_j} &= A_{k,k-1}\, t_{k-1_{\varepsilon(k-1)}} + p_{k_j}\dot{\theta}_{k_j}\ e\breve{g}er\ j = 1 \\ t_{k_j} &= t_{k_{(j-1)}} + p_{k_j}\dot{\theta}_{k_j}\ e\breve{g}er\ j > 1 \end{aligned} \tag{2.9}$$

2.2. Modüller Arası Hız Dağılımları

Eklemler arası ilişkiler elde edildikten sonra, seri bağlı eklemlerden meydana gelen modüllerin arasındaki hız dağılımları elde edilmiştir. Burada modüller tek bir link gibi düşünülmekte ve modül içi Twist yayılım denklemlerine benzer denklemlerle

hız dağılımları gösterilebilmektedir. Notasyona bağlı kalınarak, bir terimin modül olduğunu belirtmek için üzerinde " ¯ " ifadesi bulunmaktadır.

İki adet ardışıl modülün numaraları, "β" ve "i" olsun. Ayrıca "β", "i"'nin ebeveyn modülü olsun. Bu durumda denklem 2.10'de belirtilen hız dağılım eşitliği geçerlidir.

$$\bar{t}_i = \bar{A}_{i,\beta}\bar{t_\beta} + \bar{N}_i\dot{\bar{q}}_i \tag{2.10}$$

"$\bar{A}_{i,\beta}$" matrisi modül-Twist yayılım matrisi ve "$\bar{N}_i$" matrisi ise modül-hareket yayılım matrisi olarak isimlendirilmektedir. "$\bar{A}_{i,\beta}$" matrisi, "β" nolu ebeveyn modülün son ekleminden "i" nolu çocuk modülün tüm eklemlerine olan Twist yayılım matrislerini içermektedir. "$\bar{N}_i$" matrisi ise, "β" nolu modülün modül içi hareket yayılım vektörlerini ve Twist yayılım matrislerini barındırır.

Modüller birbirine seri olmayabilir. Hatta hiçbir bağlantıları bile olmayabilir. Bu durum robotun en genel kinematik eşitliğini elde etmek için göz önünde bulundurulmalıdır. Sonuç olarak aşağıdaki eşitlik sadece ardışıl olarak birbirine bağlı bulunan modüller için geçerlidir.

$$\bar{A}_{j,h} * \bar{A}_{h,i} = \bar{A}_{j,i} \tag{2.11}$$

Modül hız dağılımı daha genelleştirilerek tüm sisteme ait twist vektörü elde edilebilir.

$$\boldsymbol{t} = \boldsymbol{A}\,\boldsymbol{t} + \boldsymbol{N_d}\,\dot{\boldsymbol{\theta}} \tag{2.12}$$

Bu ifadeler daha açıklayıcı olması açısından detaylı bir biçimde açıklanırsa denklem 2.13'deki eşitlikler elde edilir.

$$\boldsymbol{t} = \begin{bmatrix} \bar{t}_0 \\ \bar{t}_1 \\ \vdots \\ \bar{t}_k \\ \vdots \\ \bar{t}_n \end{bmatrix}, \quad \boldsymbol{A} = \begin{bmatrix} 0 & & & \cdots & & & 0 \\ \bar{A}_{1,\beta} & 0 & & & & & \\ 0 & \bar{A}_{2,\beta} & \ddots & & & & \\ \vdots & & \ddots & 0 & & & \vdots \\ & & & \bar{A}_{k,\beta} & \ddots & & \\ & & & \vdots & \ddots & 0 & \\ 0 & & & \cdots & & \bar{A}_{n,\beta} & 0 \end{bmatrix} \tag{2.13}$$

$$\boldsymbol{N_d} = \begin{bmatrix} \bar{N}_0 & & & \cdots & & & 0 \\ 0 & \bar{N}_1 & & & & & \\ & 0 & \bar{N}_2 & & & & \\ & & \ddots & \ddots & & & \vdots \\ \vdots & & & 0 & \bar{N}_k & & \\ & & & & \ddots & \ddots & \\ 0 & & & \cdots & & 0 & \bar{N}_n \end{bmatrix}, \quad \boldsymbol{\dot{\theta}} = \begin{bmatrix} \bar{\theta}_0 \\ \bar{\theta}_1 \\ \vdots \\ \bar{\theta}_k \\ \vdots \\ \bar{\theta}_n \end{bmatrix}$$

Bu denklem daha kompakt hale getirilerek robota ilişkin DeNOC matrisleri elde edilebilir.

$$\boldsymbol{t = N_l\, N_d\, \dot{q}\ ve\ N_l = (1 - A)^{-1}} \tag{2.14}$$

Kompakt formda bulunan "$\boldsymbol{N_l}$" ifadesi denklem 2.15'deki gibi oluşturulur.

$$\boldsymbol{N_l} = \begin{bmatrix} 1 & 0 & & \cdots & & & 0 \\ \bar{A}_{1,0} & 1 & & & & & \\ \bar{A}_{2,0} & \bar{A}_{2,1} & \ddots & & & & \\ \vdots & \vdots & \ddots & 1 & & & \vdots \\ \bar{A}_{k,0} & \bar{A}_{k,1} & & \bar{A}_{k,\mathrm{k-1}} & \ddots & & \\ \vdots & \vdots & & \vdots & \ddots & 1 & 0 \\ \bar{A}_{n,0} & \bar{A}_{n,0} & & \cdots & & \bar{A}_{n,\mathrm{n-1}} & 1 \end{bmatrix} \tag{2.15}$$

"$\boldsymbol{N_l}$" terimi görüldüğü üzere tüm modüller arasındaki modül Twist yayılım matrislerini içermektedir. Fakat sistemde yukarıda da bahsedildiği üzere bazı modüller birden fazla modüle bağlıyken, bazı modüllerin hiçbir bağlantısı bulunmayabilir. Bu durumda bağlantısı bulunmayan iki modüle ilişkin modül Twist yayılım matrisi "sıfır" matrisine eşit olacaktır.

Modül içi ve modüller arası olarak incelenen robot kinematiğinde en sonda elde edilen denklem kinematik modeli kompakt bir biçimde vermektedir. Kompakt form, kinematik modeli basit ve tek bir denklem olarak vermesi açısından teorik olarak

pozitif bir etki yaratırken, ilerde dinamik modelleme bölümünde detaylı bir biçimde anlatılacağı üzere simülasyona uygulanması esnasında getirdiği fazladan işlem yükü nedeniyle negatif bir etki oluşturmaktadır.

2.3. Simülasyon Çalışmaları

Kinematik modelleme, sadece ileri kinematik ile ilgilendiği için, iki farklı örnek üzerinde ileri kinematik modelleme simülasyonları gerçekleştirilmiştir. Simülasyonlardan ilki sabit tabanlı bir robotik sistem olan düzlemsel robotik tutucu, diğeri ise sabit olmayan tabanlı düzlemsel yürüyen robotu içermektedir. Simülasyonlarda basitlik açısından örnekler düzlemsel tutulmuş ve birden fazla serbestlik derecesine sahip eklemler kullanılmamıştır.

Simülasyonların doğruluğunu sağlamak amacıyla VRML editör ile robotik sistemler görsel olarak oluşturulmuştur. İleri kinematik girdisi eklem hızları ve çıktısı eklemlerin uzaydaki lineer ve açısal hızlarıdır. Sağlama işlemi, kinematik çıktı olarak elde edilen eklem lineer ve açısal hızlarının görsel arayüzde kullanılması yöntemi üzerinden gerçekleştirilmiştir.

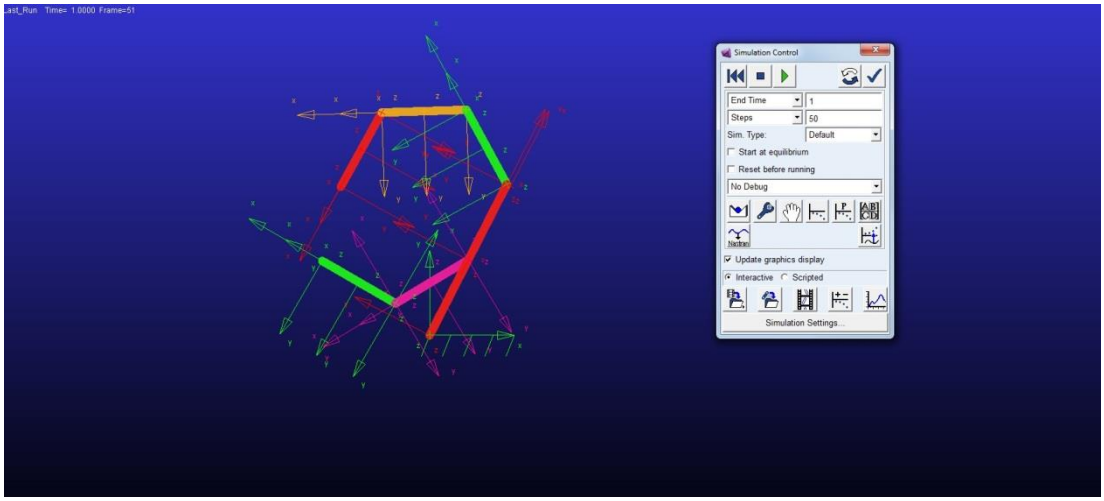
VRML editör ile robotik sistem üzerinde eklemler arasında bağlantılar gerçekleştirilebilmektedir. Buna göre eklemlere hız uygulandığı takdirde, bu bağlantılar sayesinde otomatik olarak sistem hareket etmektedir. Yani VRML editör doğrudan kinematik çıktıyı vermektedir. Diğer yandan bu özellik kinematik sağlama aracı olarak kullanılmaya olanak sağlamaktadır.

İleri kinematik çıktısı olarak elde edilen eklem lineer ve açısal hızları integre edilerek lineer konum ve oryantasyon hesaplanmaktadır. Bu sonuçları VRML editör ile örtüştürmek adına, görsel arayüzde robotik sistemden bağımsız parçalar oluşturulmuştur. Bu parçaların başlangıç konumları, robotun başlangıç konfigürasyonundaki eklem konum ve oryantasyonları ile örtüşecek şekilde belirlenmiştir. Simülasyon esnasında bu parçalara ilgili kinematik çıktıları beslenerek, algoritmanın hatalı çalışıp çalışmadığı görülmektedir.

2.3.1. Robotik tutucu problemi

Üzerinde çalışılan robotik tutucu, 3 modülden oluşmaktadır “1” nolu modül bir link, “2” nolu modül üç link ve “3” nolu modül ise iki linkten oluşmaktadır. 2 ve 3 nolu

modül 1 nolu modüle direk bağlıdır. 2 ve 3 nolu modül arasında herhangi bir bağlantı yoktur. 1 nolu modül sabit bir noktaya bağlıdır. Ayrıca tüm eklemler rotasyonel hareket etmekte ve bir serbestlik derecesine sahiptir.



Şekil 2.1 : İki boyutlu robotik tutucu

Modül seviyesinde "Twist" vektörleri denklem 2.16'daki gibi elde edilebilir:

$$\begin{aligned} \bar{t_1} &= \overline{N_1}\,\dot{\overline{q_1}} \\ \bar{t_2} &= \overline{A_{2,1}}\,\bar{t_1} + \overline{N_2}\,\dot{\overline{q_2}} \\ \bar{t_3} &= \overline{A_{3,1}}\,\bar{t_1} + \overline{N_3}\,\dot{\overline{q_3}} \end{aligned} \tag{2.16}$$

2 ve 3 nolu modül birbirine bağlı olmadığından ve her ikisi de 1 nolu modüle bağlı olduğundan, 2 ve 3 nolu "Twist" vektörünün hesaplanmasında 1 nolu modülün "Twist" vektörü kullanılmaktadır. Modül mertebesindeki matrislerin ve vektörlerin detaylı olarak açılmış hali denklem takımı 2.17'de verilmiştir.

$$\overline{t_1} = [t_{1^1}], \qquad \overline{N_1} = [p_{1^1}], \qquad \overline{\dot{q}_1} = [\dot{\theta}_{1^1}]$$

$$\overline{t_2} = \begin{bmatrix} t_{1^2} \\ t_{2^2} \\ t_{3^2} \end{bmatrix}, \qquad \overline{N_2} = \begin{bmatrix} p_{1^2} & 0 & 0 \\ A_{2^2 1^2}\, p_{1^2} & p_{2^2} & 0 \\ A_{3^2 1^2}\, p_{1^2} & A_{3^2 2^2} p_{2^2} & p_{3^2} \end{bmatrix}^{2}$$

$$\overline{\dot{q}_2} = \begin{bmatrix} \dot{\theta}_{1^2} \\ \dot{\theta}_{2^2} \\ \dot{\theta}_{3^2} \end{bmatrix}, \qquad \overline{A_{2,1}} = \begin{bmatrix} A_{1^2 1^1} \\ A_{2^2 1^1} \\ A_{3^2 1^1} \end{bmatrix} \tag{2.17}$$

$$\overline{t_3} = \begin{bmatrix} t_{1^3} \\ t_{2^3} \end{bmatrix}, \qquad \overline{N_3} = \begin{bmatrix} p_{1^3} & 0 \\ A_{2^3 1^3}\, p_{1^3} & p_{2^3} \end{bmatrix}$$

$$\overline{\dot{q}_3} = \begin{bmatrix} \dot{\theta}_{1^3} \\ \dot{\theta}_{2^3} \end{bmatrix}, \qquad \overline{A_{3,1}} = \begin{bmatrix} A_{1^3 1^1} \\ A_{2^3 1^1} \end{bmatrix}$$

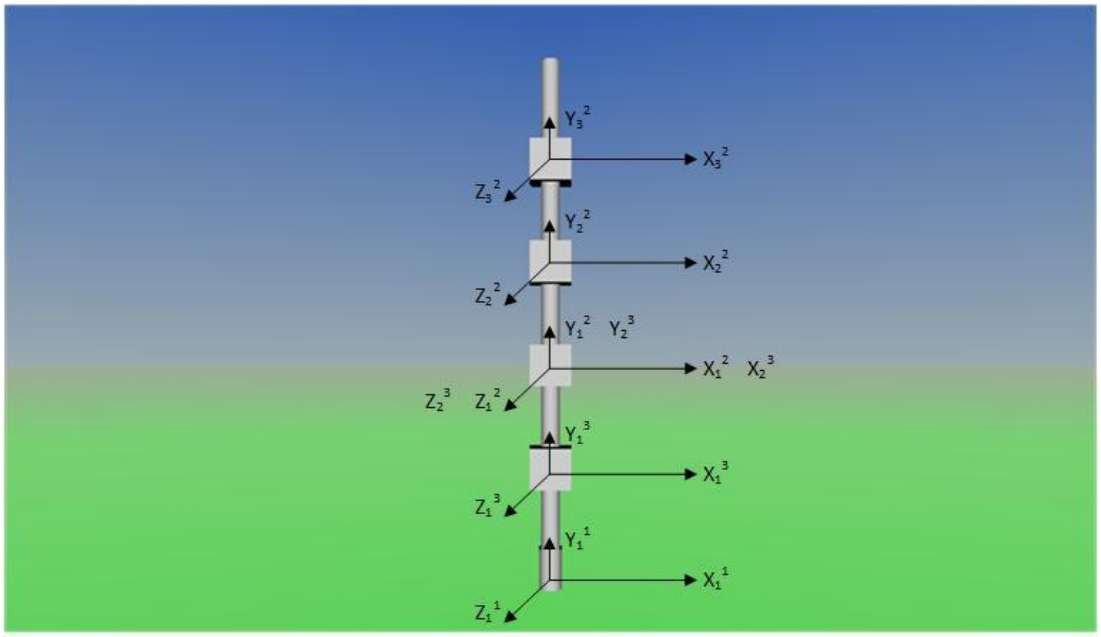
Bu ifadeler doğrudan oluşturularak modül mertebesinde kalınarak kinematik çözüm elde edilebilir. İleride de açıklanacağı üzere modül mertebesi, link seviyesindeki özyineleme şemasını ortadan kaldırarak hesaplama zamanını arttırmaktadır. Modül seviyesinde bulunan ifadelere ek olarak link seviyesinde çalışıldığında denklem takımı 2.18'deki sonuçlar elde edilir:

$$\begin{aligned} t_{1^1} &= p_{1^1}\, \dot{\theta}_{1^1} \\ t_{1^2} &= A_{1^2 1^1}\, t_{1^1} + p_{1^2}\, \dot{\theta}_{1^2} \\ t_{2^2} &= A_{2^2 1^2}\, t_{1^2} + p_{2^2}\, \dot{\theta}_{2^2} \\ t_{3^2} &= A_{3^2 2^2}\, t_{2^2} + p_{3^2}\, \dot{\theta}_{3^2} \\ t_{1^3} &= A_{1^3 1^1}\, t_{1^1} + p_{1^3}\, \dot{\theta}_{1^3} \\ t_{2^3} &= A_{2^3 1^3}\, t_{1^3} + p_{2^3}\, \dot{\theta}_{2^3} \end{aligned} \tag{2.18}$$

İlk modül ve ilk ekleme ilişkin denklemin diğerlerinden farklı olarak Twist yayılım matrisi içermemesi sabit bir noktaya bağlanmasından kaynaklanmaktadır.

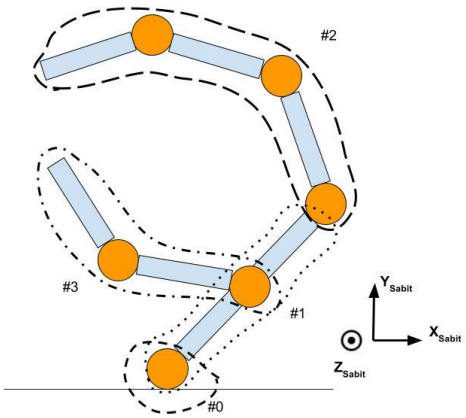
Bu ifadelerde bulunan Twist yayılım matrisi ve hareket yayılım matrisi ifadeleri, önceden de bahsedildiği üzere vektörlere bağlıdır. Bu vektörler, robotun başlangıç kofigürasyonuna bağlı olarak oluşturulmakta (Modifiye DH yöntemi) ve simülasyon sırasında güncellenmektedir. Bu işlemi kolaylaştırmak için, robot üzerinde tanımlı tüm vektörler kendi koordinat eksenlerine bağlı olarak ifade edilebilir. Eklemlere

atanan ve aynı numaralara sahip bu koordinat eksenleri simülasyon boyunca eklem açısal hızlarına bağlı olarak güncellendiği takdirde tüm vektörler otomatik olarak güncellenmiş olur. Kullanılan yöntem gereği, eklemlerin hareket eksen vektörleri ve hareket hızları var olduğundan, bu değerler kullanılarak rotasyon matrisleri hesaplanmaktadır. Daha sonrasında ise bu rotasyon matrisleri kullanılarak tüm eksenler güncellenmektedir.



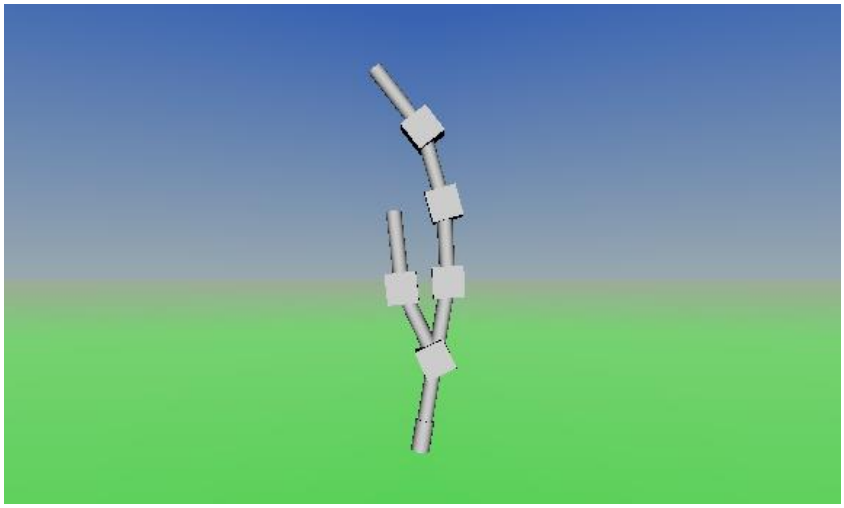
Şekil 2.2 : Robotik tutucu başlangıç konumu ve eksen atamaları

Robotik tutucu başlangıç pozisyonunda tüm modüller Y ekseni yönünde olacak şekilde konumlandırıldı. Şekil 2.2, robotik tutucunun başlangıç konfigürasyonunu ve eklem eksen atamalarını içermektedir.

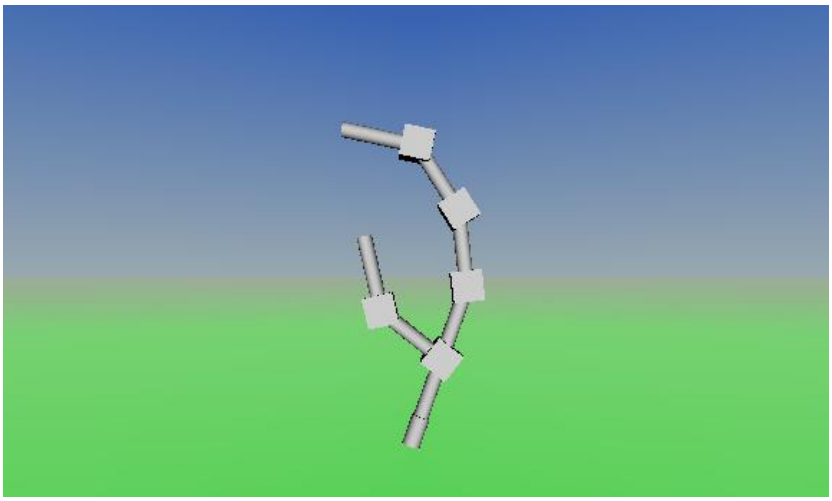


Şekil 2.3 : Robotik tutucu modül gösterimi

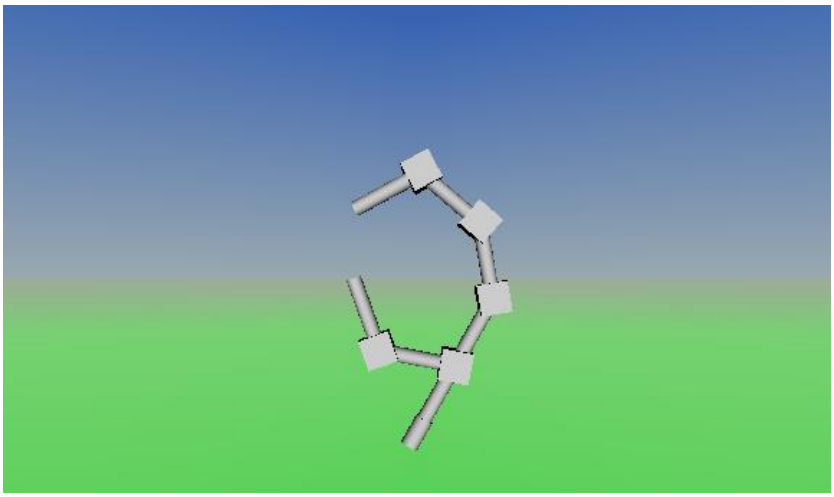
Oluşturulan kinematik modelin doğruluğunu test etmek için denklem çıktısı olarak karşımıza çıkan “Twist” vektörleri kullanılmıştır. Simülasyonun görsel arayüzüne robotik tutucuya ek olarak, tüm eklemleri takip etmek üzere beş adet küp eklenmiştir. Bu küpler herhangi bir cisimle bağlantılı olmayacak şekilde tanımlanmış ve simülasyon boyunca üç eksen doğrusal pozisyonları ve oryantasyonları beslenmiştir. Bunun için her simülasyon “step”inde hesaplanan “Twist” vektörlerinin lineer hızları içeren bölümü doğrusal pozisyonlar için, açısal hızları içeren bölümü ise oryantasyon için kullanılmıştır. Bu hızlar nümerik integrasyon ile ilgili olduğu küplere aktarılmış ve robotun eklemlerini simülasyon boyunca takip ettiği gözlenmiştir.



Şekil 2.4 : Robotik tutucu kinematik simülasyon görüntü 1



Şekil 2.5 : Robotik tutucu kinematik simülasyon görüntü 2

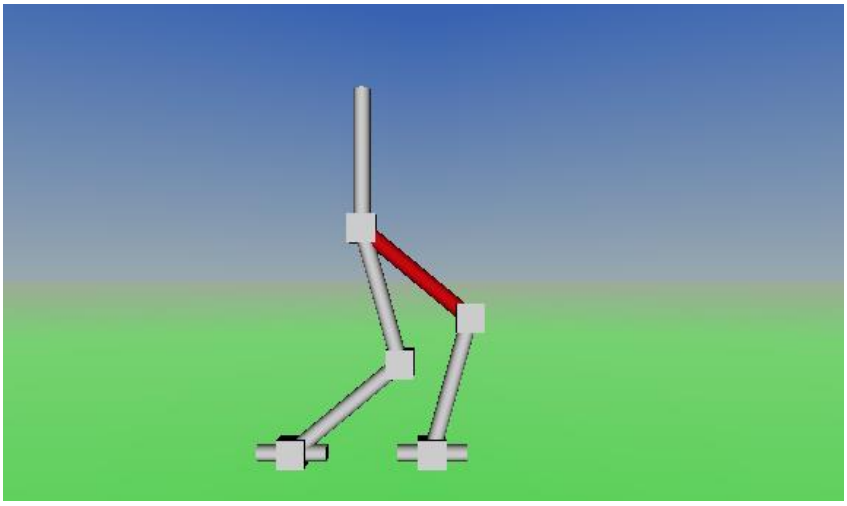


Şekil 2.6 : Robotik tutucu kinematik simülasyon görüntü 3

Sonuç olarak düzlemsel robotik tutucu probleminde ileri kinematik çözüm için MATLAB ortamında bir simülasyon oluşturulmuş ve bu simülasyon görsel arayüz olanağı sunan VRML editör ile doğrulanmıştır. Şekil 2.4, Şekil 2.5 ve Şekil 2.6 incelendiğinde eklem pozisyon ve oryantasyonlarının simülasyon boyunca iyi bir şekilde takip edildiği görülmektedir.

2.3.2. Düzlemsel yürüyen robot problemi

Simülasyon örneği olarak belirlenen düzlemsel iki ayaklı yürüyen robot, sabit olmayan tabanlı bir yapıya sahiptir. Burada taban sabit olmadığı için "0" nolu modül olarak belirlenmiştir. Robotun bacakları üç adet link ve üç adet eklem içermektedir. Ayrıca tüm eklemler rotasyonel harekete sahip ve bir serbestlik derecelidir.



Şekil 2.7 : Düzlemsel iki ayaklı yürüyen robot yapısı

Modelleme için öncelikle robota ait modüller belirlenmiştir. Burada taban sabit olarak alınmamış, bunun sonucu olarak da bir modül olarak tanımlanma ihtiyacı ortaya çıkmıştır. Bacaklarda kendi içlerinde tek zincir yapısına sahip olduğundan bunlara da birer modül tanımlanmıştır. Bu durumda kinematik modelleme modül seviyesinde denklem takımı 2.19'da oluşturulmuştur.

$$
\begin{aligned}
t_0 &= P_0\,\dot{q_0} \\
\overline{t_1} &= \overline{A_{1,0}}\,t_0 + \overline{N_1}\,\overline{\dot{q_1}} \\
\overline{t_2} &= \overline{A_{2,0}}\,t_0 + \overline{N_2}\,\overline{\dot{q_2}}
\end{aligned}
\tag{2.19}
$$

Görüldüğü üzere tabana ilişkin "Twist" vektörü diğer modüllere göre farklıdır. İlk göze çarpan kısım üzerinde herhangi bir işaret olmamasıdır. Ayrıca taban hareketi 3-boyutlu uzayda 6 eksende gerçekleşebileceği için "$\dot{q_0}$" terimi skalar değil, altı bileşenlidir. "N" matrisi yerini alan "P_0" matrisi ise denklem 2.20'de bulunan formda bulunmaktadır.

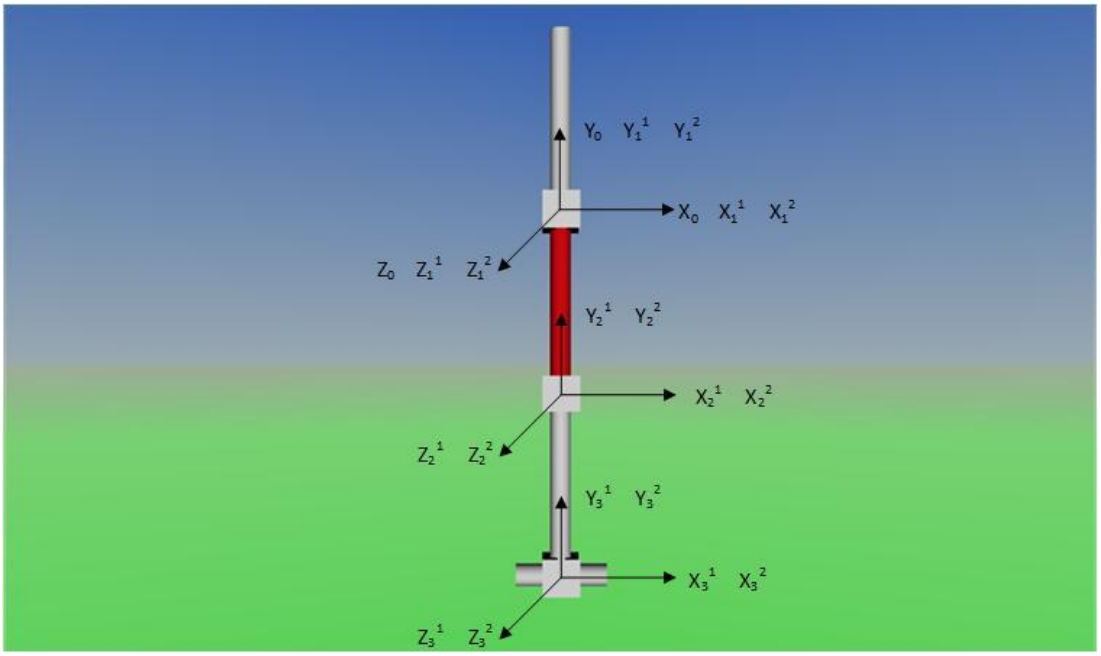
$$
P_0 = \begin{bmatrix} L_0 & 0 \\ 0 & 1 \end{bmatrix}
\tag{2.20}
$$

Bu ifadede bulunan "L_0" terimi taban oryantasyonunu belirleyen üç açının nasıl tanımlandığına göre değişkenlik göstermektedir. Bu çalışmada "L_0" terimi Euler XYZ açılarını temsil etmektedir.

Modül seviyesinde elde edilen denklemlerin açılmış hali olan link seviyesindeki karşılıkları ise denklem takımı 2.21'de belirtilmiştir.

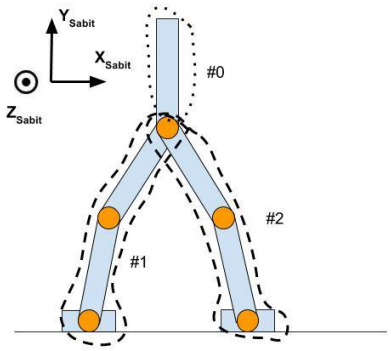
$$
\begin{aligned}
t_0 &= P_0\,\dot{q_0} \\
t_{1^1} &= A_{1^1\,0}\,t_0 + p_{1^1}\,\dot{\theta}_{1^1} \\
t_{2^1} &= A_{2^1 1^1}\,t_{1^1} + p_{2^1}\,\dot{\theta}_{2^1} \\
t_{3^1} &= A_{3^1 2^1}\,t_{2^1} + p_{3^1}\,\dot{\theta}_{3^1} \\
t_{1^2} &= A_{1^2\,0}\,t_0 + p_{1^2}\,\dot{\theta}_{1^2} \\
t_{2^2} &= A_{2^2 1^2}\,t_{1^2} + p_{2^2}\,\dot{\theta}_{2^2} \\
t_{3^2} &= A_{3^2 2^2}\,t_{2^2} + p_{3^2}\,\dot{\theta}_{3^2}
\end{aligned}
\tag{2.21}
$$

Twist yayılım matrisi ve hareket yayılım vektörü terimleri, eklemler üzerinde tanımlı vektörlere bağlı olarak oluşturulmaktadır. Bu vektörler simülasyon boyunca robotun hareketine göre güncellenerek doğru kinematik sonuçlar elde edilmiştir. Eksen atamaları Şekil 2.8'de görüldüğü gibi yapılmıştır.



Şekil 2.8 : Düzlemsel iki ayaklı yürüyen robot başlangıç konumu ve eksen atamaları

Düzlemsel yürüyen robotun kinematik çözümünün daha anlamlı olması bakımından yürüme hareketine yönelik çalışmalar da bu kapsamda araştırılmıştır. Yürüme hareketi, gövdenin ve ayakların periyodik yörüngelerinin oluşturulması ve bu yörüngelere göre ters geometrik yaklaşım ile eklem hareketlerinin belirlenmesinden ibarettir. Bu konu ileriki başlıklarda daha detaylı incelenecektir.

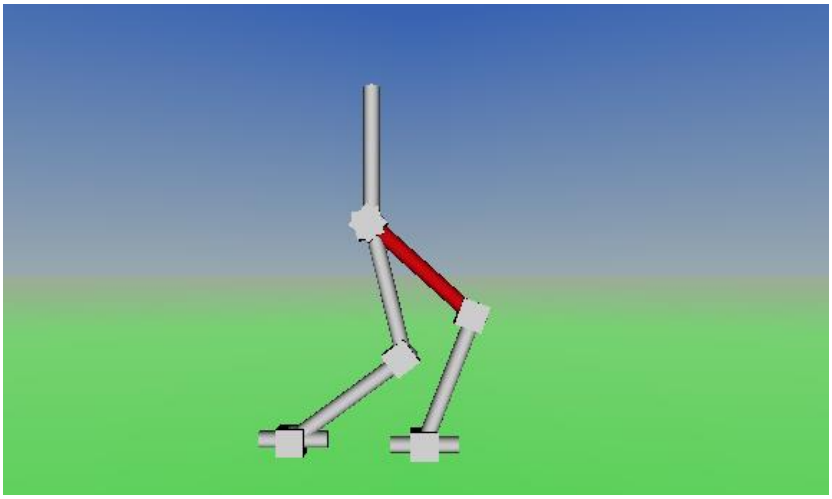


Şekil 2.9 : Düzlemsel yürüyen robot modül gösterimi

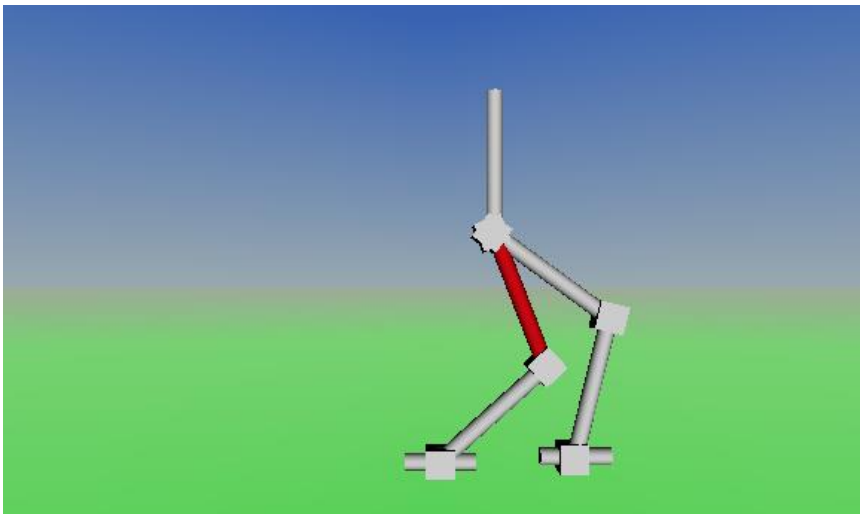
Kinematik simülasyonda, yürüyen robota verilen adım özellikleri şu şekildedir:

- Maksimum Ayak Yüksekliği: 0.2 metre
- Adım Uzunluğu: 0.5 metre

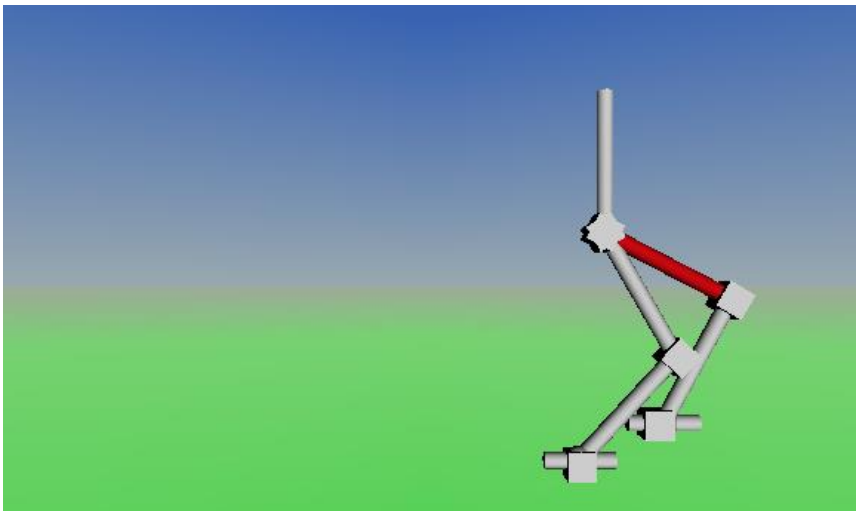
Bu özellikler göz önünde bulundurularak gövde ve ayak yörüngeleri oluşturulmuş ve animasyon ekranında serbest halde bulunan küp şeklindeki cisimlerin simülasyon boyunca eklemlerle çakışık kalıp kalmadığı gözlenmiştir. Bununla ilgili olarak Şekil 2.10, Şekil 2.11 ve Şekil 2.12'de bulunan simülasyonun farklı zamanlarına ait kareler eklenmiştir.



Şekil 2.10 : Düzlemsel iki ayaklı yürüyen robot kinematik simülasyon görüntü 1



Şekil 2.11 : Düzlemsel iki ayaklı yürüyen robot kinematik simülasyon görüntü 2



Şekil 2.12 : Düzlemsel iki ayaklı yürüyen robot kinematik simülasyon görüntü 3

Düzlemsel yürüyen robotun ileri kinematik çözümüne yönelik, yürüme hareketini de kapsayan simülasyon başarılı bir biçimde gerçekleştirilmiştir. Şekil 2.10, Şekil 2.11 ve Şekil 2.12'de simülasyondan alınan farklı zamanlardaki ekran alıntıları incelendiğinde eklemlere ilişkin olan bağımsız küplerin ileri kinematik algoritması sayesinde görsel olarak eklemlerle çakışık olduğu görülmektedir. Hem lineer konumların hem de oryantasyonların simülasyon boyunca eklemleri takip etmesi kinematik algoritmasının doğruluğunu göstermiştir.

3. DİNAMİK MODELLEME

Robotların dinamik modellemesinde iki temel yöntem bulunmaktadır. Bunlar Euler-Lagrange ve Newton-Euler yaklaşımlarıdır. Euler-Lagrange yaklaşımı robot dinamiğini kapalı bir formda verirken, Newton-Euler ise özyinelemeli dayalı bir yapı oluşturmaktadır. Literatür incelendiğinde [2,9], iterasyona dayalı açık formdaki Newton-Euler dinamik formülasyonunun, kapalı formlara göre işlem gücü açısından daha üstün olduğu belirtilmektedir. Özellikle bu fark çok serbestlik derecesi içeren robotik yapılarda daha belirgindir. Ayrıca enerji tabanlı olan Lagrange-Euler dinamiği sistem üzerinde bulunan tüm kuvvetleri vermemekte ancak vektörel tabanlı Newton-Euler dinamiği tüm kuvvetleri hesaplamaktadır.

Tüm kuvvetlerin elde edildiği bu hareket denklemleri "ayrık" hareket denklemleridir ve sistem üzerinde iş yapmayan kuvvetleri de içermektedir. Robot dinamiği ile elde edilmek istenen eklem torkları ile eklem ivmeleri arasındaki ilişkidir. Bu ilişkinin elde edilebilmesi için öncelikle "ayrık" olarak oluşturulan Newton-Euler denklemlerinin "birleştirilmiş" hale dönüştürülmesi gerekmektedir. Kinematik modelleme başlığı altında belirtilen ve Twist yayılım matrislerinden meydana gelen "DeNOC" matrisi bu aşamada kullanılmaktadır[9]. "DeNOC" matrisi sayesinde sisteme ait Newton-Euler dinamik denklemlerinden, sistem üzerinde iş yapan kuvvetler hesaplanabilmektedir. Buna "en düşük dereceli dinamik denklem" ismi verilmektedir.

Ayrıca bu işlem ile Newton-Euler dinamiğinin Lagrange-Euler formu elde edilmiş ve Lagrange-Euler dinamiği temel alınarak oluşturulan tüm kontrol yöntemlerinin rekürsif Newton-Euler dinamiğinde kullanılmasının yolu açılmış olmaktadır.[3]

3.1. Newton-Euler Dinamiği

Bir eklem ve ona bağlı linke ilişkin açısal ve lineer momentum ifadeleri denklem takımı 3.1'de verilmektedir[1].

$$M_k = m_k \dot{c}_k$$
$$L_k = I_k^c * \omega_k \tag{3.1}$$

M_k lineer momentumu, L_k açısal momentumu ifade etmektedir. Lineer momentum cismin lineer hızı ve kütlesi ile ilişkili iken, açısal momentum cismin eylemsizlik matrisi ve açısal hızı ile ilişkilidir. Kuvvet ve tork ifadeleri bu momentumların türevleri ile elde edilebilir. Böylece "ayrık" Newton-Euler denklemleri ortaya çıkmış olur.

$$I_k^c \dot{\omega}_k + \omega_k \times I_k^c \omega_k = n_k^c$$
$$m_k \ddot{c}_k = f_k^c \tag{3.2}$$

Denklem takımı 3.2'deki tüm ifadeler cismin kütle merkezine göre ifade edilmektedir. Bu ifadelerin cismin hareket ekseni olan eklem hareket eksenine indirgenmesi gerekmektedir.

Hareket ekseninde tanımlanması gereken değişkenler eylemsizlik matrisi, moment ve lineer ivme ifadeleridir.

$$\ddot{c}_k = \ddot{o}_k - d_k \times \dot{\omega}_k - \omega_k \times (d_k \times \omega_k)$$
$$I_k^c = I_k + m_k (d_k \times 1)(d_k \times 1)$$
$$f_k^c = f_k \tag{3.3}$$
$$n_k^c = n_k - d_k \times f_k$$

Denklem 3.3'deki ifadelerde $\ddot{c}_k$ kütle merkezinin lineer ivmesini, $\ddot{o}_k$ hareket ekseninin lineer ivmesini, d_k hareket ekseninden kütle merkezine olan vektörü, $\dot{\omega}_k$ açısal ivmeyi, ω_k açısal hızı, I_k^c kütle merkezine göre eylemsizlik matrisini, I_k hareket eksenine göre eylemsizlik matrisini, m_k cismin kütlesini, f_k^c kütle merkezi üzerindeki kuvvet vektörünü, f_k hareket ekseni üzerindeki kuvvet vektörünü, n_k^c kütle merkezine göre moment vektörünü ve n_k hareket eksenine göre moment vektörünü ifade etmektedir. Artık hareket eksenine göre indirgenen tüm ifadeler kullanılarak "ayrık" Newton-Euler denklemleri elde edilebilir.

İki denklem halinde ifade edilen denklemler daha kompakt hale getirildiğinde denklem 3.4 ile karşılaşılır[9].

$$M_k \dot{t}_k + \Omega_k M_k E_k t_k = w_k \tag{3.4}$$

Terimleri açıklamak gerekirse; M_k matrisi kütle, Ω_k matrisi hız ve E_k coupling matrisidir. w_k vektörü, 6 boyutlu "wrench" isimli k nolu linke ilişkin moment ve kuvvet vektörlerini içeren vektördür. Wrench vektörü üç bileşenin toplamını barındırır. Bunlardan ilki linki süren moment ve kuvvetlerdir, ikincisi sınırlandırılmış moment ve kuvvetler, üçüncü bileşen ise dış etmenlerden gelen moment ve kuvvetlerdir. Sınırlandırılmış moment ve kuvvetler, sistem üzerinde herhangi bir harekete neden olmayan bileşendir. "Wrench" vektörünün ayrıştırılması denklem 3.5'te verilmiştir.

$$w = w^D + w^C + w^F \tag{3.5}$$

Elde edilen denklem 3.4, çok kolay bir şekilde modül ve robot düzeyinde ifade edilebilir.

3.2. En Düşük Dereceli Hareket Denklemi

"Ayrık" yani tüm kuvvet ve moment bileşenlerini içeren Newton-Euler dinamik denklemlerinin, hareket analizi ve kontrolör geliştirme sürecinde kullanılabilmesi için "Birleştirilmiş" yani sadece harekete neden olan bileşenlere indirgenmesi gerekmektedir. Sınırlandırılmış kuvvetlerin çıkarılmış halini ifade eden "en düşük dereceli" dinamik denklemlerine, kinematik modellemede elde edilen DeNOC matrisleri sayesinde ulaşılır. Şöyle ki, DeNOC matrisleri denklem 3.6'da görülen ilişkiyi sağlamaktadır.

$$t = N\,\dot{q} \tag{3.6}$$

Denklem 3.6'dan da anlaşılacağı üzere "twist" vektörü ile eklem hızları arasındaki ilişki "N" matrisi ile verilmektedir. Bu matris, eklemlerin hareket eksenlerini içerdiğinden sınırlandırılmış kuvvet ve moment vektörleri bu matrisin sütunlarına diktir [9]. İki dik vektörün "skaler çarpım" işlem sonucu "0" vektörünü üreteceği için, "ayrık" olarak bir önceki bölümde verilen Newton-Euler denklemlerinden "en düşük dereceli" dinamik denklemlerine geçişte bu matris kullanılmaktadır.

$$N_d^T N_l^T \, w^C = 0$$
$$N_d^T N_l^T (M\dot{t} + \Omega M E t) = N_d^T N_l^T (w^D + w^F) \tag{3.7}$$

Bu ifadede "twist" vektörlerinin yerlerine de eklem hız eşitlikleri yerleştirilirse, Newton-Euler dinamik denklemlerinin Lagrange-Euler formuna ulaşılmış olur. Bu denklik özellikle özyinelemeli dinamik denklemlerde uygulanması zor olan kontrol yöntemlerinin kullanılabilmesinin yolunu açmaktadır[3].

$$I\,\ddot{q} + C\,\dot{q} = \tau + \tau^F \tag{3.8}$$

Eğer denklem 3.7 ile 3.8 arasında karşılıklı ifadeler eşleştirilirse denklem 3.9'daki eşitlikler elde edilir.

$$I = N^T M N, \qquad C = N^T \left(M \dot{N} + \Omega \, \mathrm{M} \, \mathrm{E} \, \mathrm{N}\right)$$
$$\tau = N^T \, w^D, \qquad \tau^F = N^T w^F \tag{3.9}$$

Son elde edilen denklem 3.9'daki ifadeler robotik sistemi bir bütün halinde ifade eden kompakt yapıyı belirtmektedir. Ayrıca bu eşitlik robot dinamiğinin tersi olarak literatürde yer almaktadır. Ters dinamik, kontrol aşamasında çokça başvurulan bir denklik olup "geri besleme lineerleştirme" tekniğini içeren kontrol yöntemlerinde kullanılmaktadır. Ters dinamiğin Newton-Euler denklemleriyle doğrudan elde edilebilmesine karşın ileri dinamik, bu denklemlerden türetilmekte ve zorlayıcı aşamalar içermektedir. İleri dinamiği bu ifadeden elde edebilmek için robot eylemsizlik matrisinin tersinin alınması gerekmektedir. Eylemsizlik matrisinin "seyrek" yapısı ve bu nedenle nümerik ters alma da ortaya çıkabilecek işlem yükü nedeniyle, bu matrisin tersinin alınması işlemi literatürde daha çok analitik yöntemlerle sağlanmaktadır [9].

3.3. Robot Eylemsizlik Matrisi

Robot eylemsizlik matrisi, robot kütle matrisi ve DeNOC matrislerinden meydana gelmektedir.

$$I = N^T M N \tag{3.10}$$

Denklem 3.10'da verilen eşitlik, NE denklemlerinin Lagrange-Euler formundan elde edilmişti. Kütle matrisleri (M) bir link veya modülün serbest halde bulunduğu durumdaki kütle ve eylemsizlik bilgilerini içerir. Fakat robotik sistemlerde linkler serbest bir biçimde değil, birbirlerine rijit bir biçimde bağlı halde bulunmaktadır. Bu durum, eklemdeki tork değerinin doğru bir biçimde hesaplanmasında etkilidir. Robot eylemsizlik matrisinin, serbest halde tanımlı kütle matrislerinden DeNOC matrisleri ile çarpılarak elde edilmesi, eylemsizlik matrisine seri bağlı olma özelliğinden kaynaklı etkileri eklemektedir.

Yani DeNOC matrisleri, genel anlamda serbest cisimler için tanımlanmış "ayrık" Newton-Euler denklemlerini alıp, birbirine eklemlerle bağlı linklerden meydana gelen robotik sistemlere bu dinamik denklemleri indirgeyerek, bağlantılardan kaynaklı etkilerin dinamik denklemlere eklenmesini sağlar. Robot eylemsizlik matrisi denklem 3.11'de olduğu gibi ifade edilebilir[9]:

$$I = N_d^T \, \widetilde{M} \, N_d \qquad \widetilde{M} = N_l^T \, M \, N_l \tag{3.11}$$

Denklem 3.11'de bulunan "$\widetilde{M}$" terimi, serbest kütle matrisinden elde edilen ve linklerin birbirine bağlı olmasından kaynaklı etkileri de içeren matristir. Genelleştirilmiş "kompozit modül" matrisi olarak ifade edilen bu matrisin yapısı denklem 3.12'de görülmektedir.

$$\widetilde{M} = \begin{bmatrix} \widetilde{\widetilde{M}}_1 & \dots & \dots & sym \\ \widetilde{\widetilde{M}}_2 \, \bar{A}_{2,1} & \widetilde{\widetilde{M}}_2 & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \widetilde{\widetilde{M}}_n \, \bar{A}_{n,1} & \widetilde{\widetilde{M}}_n \, \bar{A}_{n,2} & \dots & \widetilde{\widetilde{M}}_n \end{bmatrix} \tag{3.12}$$

Denklem 3.12'deki matris, robotun tamamına ait olduğu için matris elemanları modülleri işaret etmektedir. Modüllere ilişkin elemanlardan herhangi bir tanesinin yapısı denklem 3.13'de verilmiştir.

$$\widetilde{\widetilde{M}}_1 = \begin{bmatrix} \widetilde{M}_1 & \dots & \dots & sym \\ \widetilde{M}_2 \, A_{2,1} & \widetilde{M}_2 & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \widetilde{M}_n \, A_{n,1} & \widetilde{M}_n \, A_{n,2} & \dots & \widetilde{M}_n \end{bmatrix} \tag{3.13}$$

Artık denklem 3.13'de elemanlar linklere ilişkindir. Robotik sistemlerde altta bulunan eklemler, hem kendilerine rijit bir şekilde bağlı linki hemde bu linke üstten

bağlanan linkleri taşımak durumundadır. Bu durumda bu ekleme ilişkin tork ifadesi, Newton-Euler dinamik denklemleri ile hesaplanmak istendiği takdirde, denklemlerde bulunan kütle matrisi şu değerleri içermelidir:

- Ekleme rijit bir biçimde bağlı bulunan linkin kütle matrisini
- Bu linke bağlı üst linklere ilişkin kütle matrislerini

Bu değerleri içinde barındıran kütle matrisine önceden de belirtildiği gibi "kompozit modül" kütle matrisi ismi verilmektedir[2,9]. Link seviyesindeki eşitliği denklem 3.14 ile aktarılmıştır.

$$\widetilde{M}_k = M_k + \sum_{j \in \gamma_i} A_{j,k}^T \, M_j \, A_{j,k} \tag{3.14}$$

Görüldüğü üzere, üst linklere ilişkin kütle matrisleri eklenerek, ilgili ekleme etkiyen toplam kütle matrisi hesaplanmaktadır. Tüm kütle matrisleri, ilişkili oldukları eklemlere referans alınarak oluşturulduğu için, k-nolu "kompozit modül" kütle matrisi hesaplanırken, üst linklerin kütle matrisleri k-nolu ekleme indirgenmelidir. Bu işlem kütle matrislerini Twist yayılım matrisleri ile çarparak gerçekleştirilir.

"Kompozit modül" kavramı sadece bir konsepti ifade etmektedir. Bu konsept açıklanarak robot kütle matrisinin daha iyi açıklanması amaçlanmıştır. Özyineleme şemasında bu konsept direk olarak kullanılmamaktadır.

3.4. Sabit Tabanlı Sistemler

3.4.1. Sabit tabanlı sistem için ters dinamik model

"En düşük dereceli" hareket denkleminin aslında robotun ters dinamik ifadesi olduğu belirtilmişti. Newton-Euler dinamik denklemlerinin Lagrange-Euler formu olarak ifade edilen bu hareket denklemi robotun dinamiğini kompakt yapıda vererek, anlaşılırlığı arttırmakta ve karmaşıklığı azaltmaktadır.

$$I \, \ddot{q} + C \, \dot{q} = \tau + \tau^F \tag{3.15}$$

Robot eylemsizlik matrisi ve Coriolis matrisi daha açık şekilde yazılırsa denklem 3.16 elde edilir.

$$N^T M N \ddot{q} + N^T \left(M \dot{N} + \Omega \, \mathrm{M} \, \mathrm{E} \, \mathrm{N}\right) \dot{q} = \tau + \tau^F \tag{3.16}$$

Elde edilen denklem 3.16 robotu kompakt bir şekilde sunmaktadır. Bu ifadede görülen tüm matrisler herhangi bir özyineleme şeması oluşturulmadan doğrudan hesaplanabilir. İterasyon şeması olmaması, linklerin birbirine olan etkilerinin olmadığı anlamına gelmemektedir. Burada örneğin, herhangi bir linkin "Twist" ifadesi, kompakt yapı açıldığında modül seviyesinde denklem 3.17'deki gibi elde edilir.

$$\bar{t}_i = \bar{A}_{i,1} \bar{N}_1 \dot{\bar{q}}_1 + \bar{A}_{i,2} \bar{N}_2 \dot{\bar{q}}_2 + \cdots + \bar{N}_i \dot{\bar{q}}_i \tag{3.17}$$

Görüldüğü üzere özyineleme şeması olmamasına karşın, "i" nolu "Twist" vektörü kendisi ve önceki eklemlerin hızlarına bağlıdır. Fakat bu ifade çok sayıda toplam ifadesi içermektedir. "i" nolu "Twist" için "n" adet matris toplama işlemi varsa, "i+1" nolu "Twist" için "n+1" adet matris toplama işlemi olacak ve bu böylece devam edecektir.

Toplam ifadelerinin fazlalığının yanı sıra modül Twist yayılım matrisleri ardarda matris çarpımları içermektedir. Görüldüğü üzere linklerin birbirine olan etkileri kompakt yapıda bulunmakta ancak hesaplama açısından maliyetli olmaktadır. Bu maliyet düşürülebilir çünkü bu ifade ile bir önceki modüle ilişkin "Twist" vektörleri ortak bileşenler içermektedir. Bu ortak bileşenler kullanılarak verilen yukarıdaki ifade sadece bir önceki modüle bağımlı olarak tekrar ifade edilebilir ve işlem sayısı azaltılabilir. Sonuç olarak sadece robot kompakt hareket denklemi kullanılarak doğru sonuçlar elde edilebilir ancak işlem sayısı bakımından son derece problemli bir yapı ortaya çıkmış olur.

Kompakt yapıda kalmadan alt seviyelerde işlemlerin çözülmesi için hareket denklemi iki aşamalı şekilde tekrar yazılırsa denklem takımı 3.18 elde edilir[9].

$$\begin{aligned} M \dot{t} + \Omega M E t &= w \\ N_d^T N_l^T w &= \tau - \tau^F \end{aligned} \tag{3.18}$$

Denklem 3.18'de görülen ilk denklem "Twist" vektörlerini içerdiğinden ileri özyinelemeye, ikinci denklem ise DeNOC matrislerinin transpozunu içerdiğinden geri özyinelemeye neden olmaktadır. Bu konseptler ilerki bölümlerde açıklanmıştır.

3.4.1.1. İleri özyineleme

Robota ilişkin kompakt "Wrench" denklemi "Twist" vektörlerini içermektedir. Robot kompakt "Twist" vektörü denklem 3.19'da hesaplanmaktadır.

$$t = N\,\dot{q} \tag{3.19}$$

Bu kompakt yapı n adet modüle sahip robot için modül seviyesinde ayrıştırılırsa ilk iki modül ve i nolu modül denklem 3.20'de görüldüğü şekilde elde edilir.

$$\begin{aligned} \bar{t}_1 &= \bar{N}_1\dot{\bar{q}}_1 \\ \bar{t}_2 &= \bar{A}_{2,1}\bar{N}_1\dot{\bar{q}}_1 + \bar{N}_2\dot{\bar{q}}_2 \\ \bar{t}_i &= \bar{A}_{i,1}\bar{N}_1\dot{\bar{q}}_1 + \bar{A}_{i,2}\bar{N}_2\dot{\bar{q}}_2 + \cdots + \bar{N}_i\dot{\bar{q}}_i \end{aligned} \tag{3.20}$$

Sadece kompakt yapı kullanıldığında hesaplanması zorunlu olan denklem 3.20'deki ifadeler, modül numarası arttıkça daha çok işlem kapasitesi gerektirmektedir. Denklemler incelendiğinde bu ifadelerin sayısı bir ileri özyineleme yapısı oluşturularak azaltılabilir. Saha denklem 3.21'deki yapıyı ileriye sürmüştür.

$$\bar{t}_i = \bar{A}_{i,i-1}\,\bar{t}_{i-1} + \bar{N}_i\dot{\bar{q}}_i \tag{3.21}$$

Denklem 3.21'deki yapıda hem toplam ifadeleri azaltılmış hemde modül Twist yayılım matrisleri sadece bir önceki modül ile ilişkili olduğundan çarpım ifade sayıları da otomatik olarak azalmıştır.

Modül seviyesinde verilen denklem 3.21'deki özyineleme şeması doğrudan kullanıldığında link seviyesinde fazladan ve gereksiz işlemlerin yapılmasına neden olmaktadır. Robot kompakt yapısından modül seviyesine geçişe benzer şekilde, modül seviyesinden link seviyesine geçiş yapılması daha etkili bir algoritma üretmeyi sağlamaktadır.

"i" nolu modüle ilişkin kompakt yapı link seviyesinde ayrıştırılırsa "1", "2" ve "k" nolu linklere ilişkin "Twist" vektörleri denklem 3.22'deki gibi elde edilir.

$$t_{1^i} = A_{1^i,\beta}\, t_\beta + p_{1^i}\, \dot{\theta}_{1^i}$$

$$t_{2^i} = A_{2^i,\beta}\, t_\beta + A_{2^i,1^i}\, p_{1^i}\, \dot{\theta}_{1^i} + p_{2^i}\, \dot{\theta}_{2^i} \tag{3.22}$$

$$t_{k^i} = A_{k^i,\beta}\, t_\beta + A_{k^i,1^i}\, p_{1^i}\, \dot{\theta}_{1^i} + A_{k^i,2^i}\, p_{2^i}\, \dot{\theta}_{2^i} + \cdots + p_{k^i}\, \dot{\theta}_{k^i}$$

"β" terimi "ebeveyn" modülün son linkini temsil etmektedir. "Modül içindeki link sayısı arttıkça, toplam ifadeleri ve çarpım ifadeleri, modül seviyesinde olduğu gibi artış göstermektedir. Ayrıca tüm modül içindeki "Twist" ifadelerinin hesaplanmasında, önceki modülün son linkine ilişkin "Twist" ifadesi de yer almaktadır. Tüm bu fazladan işlemleri ortadan kaldırmak için Saha denklem 3.23'de bulunan modül içi ileri özyineleme algoritmasını öne sürmüştür[9].

$$\begin{aligned} t_{1^i} &= A_{1^i,\beta}\, t_\beta + p_{1^i}\, \dot{\theta}_{1^i} \\ t_{k^i} &= A_{k^i,k-1^i}\, t_{k-1^i} + p_{k^i}\, \dot{\theta}_{k^i} \quad k = 2, \dots, \eta \end{aligned} \tag{3.23}$$

Görüldüğü üzere, i nolu modülün ilk linki, önceki modülün son linkine bağlı olduğundan (modül tanımına göre) "$A_{1^i,\beta}\, t_\beta$" ifadesini içermektedir. Geriye kalan i nolu modül içindeki linkler bir önceki link cinsinden yazıldığı için, işlem sayıları önemli ölçüde azalmıştır. Ayrıca bu yapı link seviyesinde bir ileri özyineleme şeması ortaya çıkarmıştır.

Twistlerin en az işlemle elde edilmesi gösterildikten sonra iki aşamadan oluşan hareket denkleminin ilk aşaması denklem 3.24, 3.25 ve 3.26 görüldüğü şekilde elde edilebilir.

$$\begin{aligned} t_{1^i} &= A_{1^i,\beta}\, t_\beta + p_{1^i}\, \dot{\theta}_{1^i} \\ t_{k^i} &= A_{k^i,k-1^i}\, t_{k-1^i} + p_{k^i}\, \dot{\theta}_{k^i} \quad k = 2, \dots, \eta \end{aligned} \tag{3.24}$$

$$\begin{aligned} \dot{t}_{1^i} &= \dot{A}_{1^i,\beta}\, t_\beta + A_{1^i,\beta}\, \dot{t}_\beta + \dot{p}_{1^i}\, \dot{\theta}_{1^i} + p_{1^i}\, \ddot{\theta}_{1^i} \\ \dot{t}_{k^i} &= \dot{A}_{k^i,k-1^i}\, t_{k-1^i} + A_{k^i,k-1^i}\, \dot{t}_{k-1^i} + \dot{p}_{k^i}\, \dot{\theta}_{k^i} + p_{k^i}\, \ddot{\theta}_{k^i} \quad k = 2, \dots, \eta \end{aligned} \tag{3.25}$$

$$w_{k^i} = M_{k^i} \dot{t}_{k^i} + \Omega_{k^i} M_{k^i} E_{k^i} t_{k^i} \tag{3.26}$$

İleri özyinelemeye ilişkin tüm şema "Wrench" ve "Wrench" hesaplanmasında gerekli olan "Twist" ve "Twist" türevinin hesaplanmasından ibarettir. "Twist" ve "Twist"

türevi modülün ilk linki olup olmamalarına göre farklı eşitlikler içermektedir. Tüm bölüm boyunca açıklanmaya çalışılan nokta robot kompakt yapısının maliyetli bir çözüm olduğu, öbür taraftan link seviyesinin ise kompakt yapıya göre daha az işlem gerektirdiğidir. Bu nedenle simülasyonlarda link seviyesinde işlemler gerçekleştirilmiştir.

3.4.1.2. Geri özyineleme

İki aşamalı hareket denkleminin ikinci aşamasına ilişkin, "Wrench", tork eşitliğini veren denklem, denklem 3.27'de olduğu şekildedir.

$$N_d^T N_l^T \, w = \tau - \, \tau^F \tag{3.27}$$

DeNOC matrislerinin etkisi önceki bölümlerde açıklanmıştı. Denklem 3.27'deki ifade robotun tamamına ilişkin olan kompakt yapıdır. Herhangi bir iterasyon şeması oluşturulmadan, DeNOC matrisleri ve ileri özyineleme sonucu elde edilen "Wrench" vektörleri kullanılarak eklem torkları elde edilebilir. Fakat bu kompakt yapı, "Wrench" eşitliğinde olduğu gibi gereksiz işlemlerin yapılmasına yol açmaktadır. DeNOC matrislerinin transpoz alınmış hali incelendiğinde denklem 3.28 ortaya çıkmaktadır.

$$N^T = \begin{bmatrix} \bar{N}_1{}^T & \bar{N}_1{}^T \bar{A}_{2,1}{}^T & \cdots & \bar{N}_1{}^T \bar{A}_{i,1}{}^T & \cdots & \bar{N}_1{}^T \bar{A}_{n,1}{}^T \\ 0 & \bar{N}_2{}^T & \cdots & \bar{N}_2{}^T \bar{A}_{i,2}{}^T & \cdots & \bar{N}_2{}^T \bar{A}_{n,2}{}^T \\ 0 & 0 & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \bar{N}_i{}^T & \ldots & \bar{N}_i{}^T \bar{A}_{n,i}{}^T \\ 0 & 0 & 0 & 0 & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \bar{N}_n{}^T \end{bmatrix} \tag{3.28}$$

Robotun tamamına ilişkin tork-"Wrench" eşitliği modül seviyesinde ayrıştırılırsa "1", "i" ve "n" nolu modül tork ifadeleri denklem takımı 3.29'daki şekilde elde edilmektedir.

$$\begin{aligned} \bar{\tau}_1 &= \bar{N}_1{}^T \bar{w}_1 + \bar{N}_1{}^T \bar{A}_{2,1}{}^T \bar{w}_2 + \cdots + \bar{N}_1{}^T \bar{A}_{i,1}{}^T \bar{w}_i + \cdots + \bar{N}_1{}^T \bar{A}_{n,1}{}^T \bar{w}_n \\ \bar{\tau}_i &= \bar{N}_i{}^T \bar{w}_i + \cdots + \bar{N}_i{}^T \bar{A}_{n,i}{}^T \bar{w}_n \\ \bar{\tau}_n &= \bar{N}_n{}^T \bar{w}_n \end{aligned} \tag{3.29}$$

Denklem 3.29'da görüldüğü üzere altta kalan eklem torkları üstte bulunan linkin "Wrench" ifadelerinden etkilenmektedir. Ancak bu denklemlerde herhangi bir iterasyon şeması görülmemektedir. Ayrıca burada kendini tekrar eden işlemler bulunmakta ve bu işlemler gereksiz bir biçimde işlem sayısını arttırmaktadır. Bunun önüne geçmek için Saha "$\widetilde{\overline{w}}$" ifadesini oluşturmuştur. Bu ifade ile hem işlem sayısı azaltılmış hem de tork hesaplama denklemleri sadece bir üst modüllere (birden fazla olabilir.) bağlı yazılarak geri özyineleme algoritması oluşturulmuştur. Modül seviyesindeki bu yapıya ait "n", "i" ve "1" nolu tork ifadeleri denklem 3.30'daki şekildedir.

$$\bar{\tau}_n = \bar{N}_n{}^T \bar{w}_n$$

$$\bar{\tau}_i = \bar{N}_i{}^T \widetilde{\overline{w}}_i \qquad \widetilde{\overline{w}}_i = \overline{w}_i + \sum_{\lambda \in \xi_i} \bar{A}_{\lambda,i}{}^T \widetilde{\overline{w}}_\lambda \tag{3.30}$$

$$\bar{\tau}_1 = \bar{N}_1{}^T \widetilde{\overline{w}}_1 \qquad \widetilde{\overline{w}}_1 = \overline{w}_1 + \sum_{\lambda \in \xi_1} \bar{A}_{\lambda,1}{}^T \widetilde{\overline{w}}_\lambda$$

Denklem 3.30'daki ifadelerde görülen "ξ_i", "i" nolu modüle üstten bağlı modüllerin kümesidir. Önemli ölçüde azaltılmış ve özyineleme şemasının oluşmasını sağlamış bu konsept ile modül seviyesi tamamlanmış olmaktadır. Ters dinamik ileri özyineleme bölümünde olduğu gibi modül seviyesinde kalındığı takdirde yine aynı şekilde fazladan işlemler ortaya çıkmaktadır. Bunun önüne geçmek açısından link seviyesinde "$\widetilde{w}$" konsepti Saha tarafından oluşturulmuştur.

"i" nolu modüle ilişkin tork-"Wrench" ifadesi "1", "k", "η-1" ve "η" nolu linkler için açılırsa denklem takımı 3.31 elde edilir.

$$\tau_{\eta^i} = p_{\eta^i}{}^T w_{\eta^i} + p_{\eta^i}{}^T \sum_{\beta \in \xi_i} A_{\beta,\eta^i}{}^T w_j$$

$$\tau_{\eta-1^i} = p_{\eta-1^i}{}^T w_{\eta-1^i} + p_{\eta-1^i}{}^T A_{\eta^i,\eta-1^i}{}^T w_{\eta^i} + p_{\eta-1^i}{}^T A_{\eta^i,\eta-1^i}{}^T \sum_{\beta \in \xi_i} A_{\beta,\eta^i}{}^T w_j$$

$$\tau_{k^i} = p_{k^i}{}^T w_{k^i} + p_{k^i}{}^T A_{k+1^i,k^i}{}^T w_{k+1^i} + \cdots + p_{k^i}{}^T A_{\eta^i,k^i}{}^T \sum_{\beta \in \xi_i} A_{\beta,\eta^i}{}^T w_j \tag{3.31}$$

$$\tau_{1^i} = p_{1^i}{}^T w_{1^i} + p_{1^i}{}^T A_{2^i,1^i}{}^T w_{2^i} + \cdots + p_{1^i}{}^T A_{\eta^i,1^i}{}^T \sum_{\beta \in \xi_i} A_{\beta,\eta^i}{}^T w_j$$

Modüllere ilişkin ifadenin detaylı hali olan link seviyesindeki karşılıkları denklem takımı 3.31'de görülmektedir. Alt linklere indikçe işlem sayısı önemli ölçüde artmakta ve ayrıca her link için, ait olduğu modüle üstten bağlı modülün linklerinden gelen etkilerde tekrar tekrar hesaplanmaktadır. Bu denkliklerin işlem sayısı Saha'nın öne sürdüğü link seviyesi "$\widetilde{w}$" konsepti ile önemli ölçüde azaltılmaktadır. Ayrıca modül seviyesinde olduğu gibi link seviyesinde de özyineleme şemasını ortaya çıkartmaktadır. "i" nolu modülün "1", "k", "η-1" ve "η" nolu modülleri üzerinden açıklanan geri özyineleme algoritması denklem 3.32'deki şekildedir.

$$\begin{aligned}
\tau_{\eta^i} &= {p_{\eta^i}}^T \widetilde{w}_{\eta^i} \qquad \widetilde{w}_{\eta^i} = w_{\eta^i} + \sum_{\beta \,\epsilon\, \xi_i} {A_{1^\beta,\eta^i}}^T \widetilde{w}_{1^\beta} \\
\tau_{\eta-1^i} &= {p_{\eta-1^i}}^T \widetilde{w}_{\eta-1^i} \qquad \widetilde{w}_{\eta-1^i} = w_{\eta-1^i} + {A_{\eta^i,\eta-1^i}}^T \widetilde{w}_{\eta^i} \\
\tau_{k^i} &= {p_{k^i}}^T \widetilde{w}_{k^i} \qquad \widetilde{w}_{k^i} = w_{k^i} + {A_{k+1^i,k^i}}^T \widetilde{w}_{k+1^i} \\
\tau_{1^i} &= {p_{1^i}}^T \widetilde{w}_{1^i} \qquad \widetilde{w}_{1^i} = w_{1^i} + {A_{2^i,1^i}}^T \widetilde{w}_{2^i}
\end{aligned} \tag{3.32}$$

Son aşamada elde edilen ters dinamik geri özyineleme denklemleri, robot kompakt denkleminin direk uygulanmasına göre önemli ölçüde az sayıda işlem içermektedir. Tekrarlı olarak aynı işlemin yapılmasını engelleyen "$\widetilde{w}$" hem işlem sayısının azaltılmasına hemde geri özyineleme algoritmasının link seviyesinde oluşturulmasını sağlamıştır.

Geri özyinelemenin tamamlanması ile robota ilişkin özyinelemeli ters dinamik algoritması açıklanmış olmaktadır. İlk olarak kompakt formda oluşturulan robot hareket denklemi, doğrudan bu yapının kullanılmasıyla ortaya çıkan hesaplama zamanı problemi ile karşı karşıya kalmıştır. Matris yapısındaki bu hareket denklemi öncelikle iki aşamalı olarak düzenlenmiştir. İlk aşama modül ve link seviyesinde incelenerek ileri özyinelemeyi ortaya çıkartmış, ikinci aşama da yine modül ve link seviyesinde incelenerek geri özyinelemenin oluşturulmasını sağlamıştır. Burada önemli nokta, eğer hesaplama zamanı çok önem arz ediyorsa, tüm işlemlerin link seviyesinde gerçekleştirilmesi gereğidir. Eğer önem arz etmiyorsa, kompakt yapıda kalınabilir.

3.4.2. Sabit tabanlı sistem için ileri dinamik model

İleri dinamik modelleme robot eklem torkları ile eklem ivmeleri arasındaki ilişkiyi vermektedir. Türetilmesi ters dinamik üzerinden gerçekleştirilmektedir. Robot simülasyonu için son derece önemli olan ileri dinamik model, ters dinamik modellemeye göre daha kompleks bir yapıda bulunmaktadır. Robot kompakt hareket denkleminden elde edilen robot ileri dinamik denklemi, denklem 3.33'de verilmiştir.

$$\ddot{q} = I^{-1}\,(\tau - h) \qquad h = C\,\dot{q} + \tau^{F} \tag{3.33}$$

Denklem 3.33'de bulunan en önemli ve zorlayıcı nokta robot eylemsizlik matrisinin tersinin alınmasıdır. İşlem maliyeti yüksek olan bu ters alma işlemi için literatürde çeşitli analitik yöntemler mevcuttur.

3.4.2.1. Robot eylemsizlik matrisinin tersi

Robot eylemsizlik matrisinin ters alma işlemi, seri robotik sistemler için Saha tarafından öne sürülen yöntem ile analitik olarak gösterilmiştir[7]. Bu yöntemde genelleştirilmiş eylemsizlik matrisi, üç matrisin çarpımı şeklinde ifade edilmiştir.

$$I = U\,D\,U^{T} \tag{3.34}$$

Denklem 3.34'de bulunan "U" matrisi üst üçgen matris ve "D" matrisi köşegen bir matristir. Seri sistemler için öne sürülen bu yöntem, modüler olarak ifade edilen karmaşık yapılı robotik sistemlere uygulanmıştır[9]. Ayrıştırma işlemi "ters Gaussian eliminasyon" metodu ile gerçekleştirilmektedir. Bu işlem ile ilk olarak robot eylemsizlik matrisi iki matrise ayrıştırılır. Bunlardan biri üst üçgen matris iken diğeri alt üçgen matristir. Sonrasında alt üçgen matris bir adet köşegen ve elde edilen üst üçgenin transpoze çarpımı şeklinde ifade edilerek ayrıştırma tamamlanmış olmaktadır.

"Ters Gaussian eliminasyon" metodu "Gaussian eliminasyon" üzerinden hareket edilerek oluşturulan bir metottur. Bu nedenle "Gaussian eliminasyon" metodu iyi bir şekilde anlaşılmalıdır.

Gaussian eliminasyon metodu

"Gaussian eliminasyon" metodu, bir lineer denklem takımını tek değişken kalacak şekilde basitleştirme işlemini satır işlemleri ile gerçekleştirir. Denklem 3.35'deki eşitlikle tanımlanmış lineer denklem takımları bu metot ile çözülebilmektedir.

$$A\,x = b \tag{3.35}$$

Genel anlamda "Gaussian eliminasyon" metodu Şekil 3.1'deki şemaya sahiptir.

Şekil 3.1 : Gaussian eliminasyon metodu şeması

Tanımlanan satır işlemleri belirli bir düzeni takip edecek şekilde düzenlenmektedir. Bu düzen sırasıyla ilk sütundan başlayarak "A" matrisinin üst üçgen matrise dönüştürülmesini sağlar. Bu düzen aslında denklem 3.36'daki işleme denk gelmektedir.

$$M^{n-1} \ldots M^{1}\, A = U \tag{3.36}$$

"M" matrislerinin sayısı, "A" matrisinin boyutuna göre şekillenmektedir. "A" matrisi ile aynı boyuta sahip "M" matrisleri denklem 3.37'de verilmiştir[6].

$$M^{(k)} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \ddots & 1 & 0 & \ddots & 0 \\ 0 & 0 & \ddots & -m_{k+1,k} & 1 & \ddots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & -m_{n,k} & 0 & 0 & 1 \end{bmatrix} \tag{3.37}$$

"M" matrisindeki katsayılar, "A" matrisinin elemanları belirli bir formda ise, bu formlardan yararlanılarak, bu katsayılar genel bir biçimde yazılabilir.

Örnek

Denklem 3.38'deki matrisin üst üçgene dönüşümü "M" matrisleri kullanılarak şu şekilde gerçekleştirilir.

$$A = \begin{bmatrix} 3 & 1 & 2 \\ 2 & 4 & 9 \\ 8 & 6 & 5 \end{bmatrix} \tag{3.38}$$

Bu matris "3x3" olduğu için 2 adet, uygun katsayılara sahip "M" matrisi ile çarpıldığı takdirde üst üçgen karşılığı elde edilmiş olur.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -m_{3,2} & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ -m_{2,1} & 1 & 0 \\ -m_{3,1} & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 & 1 & 2 \\ 2 & 4 & 9 \\ 8 & 6 & 5 \end{bmatrix} = \begin{bmatrix} u_{1,1} & u_{1,2} & u_{1,3} \\ 0 & u_{2,2} & u_{2,3} \\ 0 & 0 & u_{3,3} \end{bmatrix} \tag{3.39}$$

$$\begin{bmatrix} 3 & 1 & 2 \\ -3\,m_{2,1} + 2 & -m_{2,1} + 4 & -2\,m_{2,1} + 9 \\ 3\,m_{3,2}\,m_{2,1} - 3\,m_{3,1} - 2\,m_{3,2} + 8 & m_{3,2}\,m_{2,1} - m_{3,1} - 4\,m_{3,2} + 6 & 2\,m_{3,2}\,m_{2,1} - 2m_{3,1} - 9\,m_{3,2} + 5 \end{bmatrix} = \begin{bmatrix} u_{1,1} & u_{1,2} & u_{1,3} \\ 0 & u_{2,2} & u_{2,3} \\ 0 & 0 & u_{3,3} \end{bmatrix}$$

Son elde edilen matris eşitliğinde üst üçgenin "0" elemanlarını sağlaması gereken üç adet denklem ortaya çıkmaktadır.

$$\begin{aligned} 3\,m_{3,2}\,m_{2,1} - 3\,m_{3,1} - 2\,m_{3,2} + 8 &= 0 \\ m_{3,2}\,m_{2,1} - m_{3,1} - 4\,m_{3,2} + 6 &= 0 \\ 2\,m_{3,2}\,m_{2,1} - 2m_{3,1} - 9\,m_{3,2} + 5 &= 0 \end{aligned} \tag{3.40}$$

Bu üç eşitlikten "M" matrisi katsayıları hesaplanabilir ve sonuçta üst üçgen matris elde edilebilir.

$$m_{3,2} = \frac{3}{11}, \qquad m_{3,2}\,m_{2,1} - m_{3,1} = \frac{54}{11} \tag{3.41}$$

Bu sonuçlara göre "$m_{2,1}$" ve "$m_{3,1}$" eşitliği sağlayacak şekilde seçilir.

$$m_{2,1} = 3 \qquad m_{3,1} = \frac{45}{11} \tag{3.42}$$

3.4.2.2. Üst üçgen matrisin hesaplanması

Üst üçgen matris olan "U" matrisi "ters Gausian eliminasyon"(RGE) metodu ile oluşturulmaktadır. "Gaussian eliminasyon" metodunda matris ilk sütunundan başlanarak elimine edilmekte ve sonuçta üst üçgen bir matris elde edilmektedir. "RGE" metodunda ise, elimine edilmek istenen matris son sütundan başlanarak elimine edilir ve sonuçta alt üçgen bir matris elde edilir.

Robot eylemsizlik matrisinin elemanları modüler yapıdan dolayı skaler değil matris formundadır ve belirli bir eşitliğe sahiptir.

$$\bar{I}_{ij} = \bar{N}_i^{\,T}\, \tilde{\bar{M}}_i\, \bar{A}_{i,j}\, \bar{N}_j \tag{3.43}$$

Denklem 3.43'deki eşitlik sayesinde önceden de bahsedildiği üzere üst üçgen "U" matrisinin elemanlarına ilişkin ifadeler oluşturulabilmektedir.

"n" adet modül içeren robota ilişkin "I" robot eylemsizlik matrisi "n-1" adet üst üçgen ile çarpıldığında "RGE" metodunun yardımıyla alt üçgen matrise çevrilebilir.

$$B_1\, B_2 \ldots B_{n-1}\, I = L \tag{3.44}$$

"B" matrisleri "Gaussian eliminasyon" başlığı altında anlatılan "M" matrislerine benzer bir biçimde oluşturulmaktadır. "B" matrislerinin elemanları matris olan üst üçgen matris iken "M" matrisleri elemanları skaler olan alt üçgen matrislerdir.

$$B_i = \begin{bmatrix} 1 & 0 & \cdots & -\bar{U}_{0,i+1} & \cdots & 0 \\ 0 & 1 & 0 & \vdots & \ddots & \vdots \\ \vdots & \ddots & \ddots & -\bar{U}_{i,i+1} & \ddots & 0 \\ \vdots & \ddots & \ddots & 1 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \tag{3.45}$$

Denklem 3.44'de "I" matrisi yalnız bırakılırsa denklem 3.46 elde edilir.

$$B_1\, B_2 \ldots B_{n-1}\, I = L \tag{3.46}$$

"B" matrislerinin tersinin alınması işlemi sadece değişken terimlerinin işaretinin değiştirilmesi ile tamamlanmaktadır.

$$B_i^{-1} = \begin{bmatrix} 1 & 0 & \cdots & \bar{U}_{0,i+1} & \cdots & 0 \\ 0 & 1 & 0 & \vdots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \bar{U}_{i,i+1} & \ddots & 0 \\ \vdots & \ddots & \ddots & 1 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \tag{3.47}$$

Burada denklem 3.48'deki tanımlama yapılsın.

$$I = U\, L \tag{3.48}$$

$$U = \begin{bmatrix} 1 & \overline{U}_{0,1} & \overline{U}_{0,2} & \cdots & \overline{U}_{0,i+1} & \cdots & \overline{U}_{0,n} \\ 0 & 1 & \overline{U}_{1,2} & \cdots & \vdots & \ddots & \vdots \\ \vdots & \ddots & 1 & \ddots & \vdots & \ddots & \overline{U}_{i,n} \\ \vdots & \ddots & \ddots & \ddots & \overline{U}_{i,i+1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \overline{U}_{n-1,n} \\ 0 & 0 & \cdots & \cdots & 0 & \cdots & 1 \end{bmatrix}$$

Görüldüğü üzere "I" robot eylemsizlik matrisi, üst üçgen ve alt üçgen olarak iki matrisin çarpımı şeklinde ifade edilmiş oldu. Ayrıca "I" matrisinin elemanları belirli bir formda olduğu için "U" matrisinin elemanları da denklem 3.49'da gösterilen formda oluşturulabildi[9].

$$\overline{U}_{i,j} = \overline{N}_i^{\,T} \; \bar{A}_{j,i}^{\,T} \; \overline{N}_j^{\,-T} \tag{3.49}$$

"I" matrisi, bir üst üçgen ve bir alt üçgenin çarpımı şekline dönüştürülmesinin ardından alt üçgen matris yine iki matris şeklinde tekrar oluşturulabilmektedir[9].

$$L = D \; U^T \tag{3.50}$$

Burada "D" matrisi köşegen bir matristir ve denklem 3.51 formundadır[9].

$$D = \begin{bmatrix} \hat{\bar{I}}_1 & & & 0 \\ & \hat{\bar{I}}_2 & & \\ & & \ddots & \\ 0 & & & \hat{\bar{I}}_n \end{bmatrix} \tag{3.51}$$

"D" matrisinin elemanları ise denklem 3.52 ile aynı formda bulunmaktadır[9].

$$\hat{\bar{I}}_i = \overline{N}_i^{\,T} \; \hat{\overline{M}}_i \; \overline{N}_i \tag{3.52}$$

Alt üçgen matrisin eşitliği yerine koyulduğu takdirde denklem 3.53'deki eşitlik elde edilir;

$$I = U \; D \; U^T \tag{3.53}$$

"I" robot eylemsizlik matrisinin bu ayrıştırma işlemi ters alma işleminde temel oluşturmaktadır.

$$I^{-1} = (U\ D\ U^T)^{-1} = U^{-T}\ D^{-1}\ U^{-1} \tag{3.54}$$

Ayrıştırma işlemi sonucu elde edilen denklem 3.4'deki matris kombinasyonunun tersinin alınması, üç matrisin ayrı ayrı tersinin alınması ile gerçekleştirilmektedir. Üst üçgen matrisin tersinin alınması işlemi önceden açıklandığı gibi matris tersi alınmadan sağlanmaktadır. Köşegen matrisin tersinin alınması ise köşegen elemanların ayrı ayrı tersinin alınması ile gerçekleştirilmektedir. Bu köşegen matris, pozitif tanımlı modül kütle matrislerinden oluştuğu için herhangi bir singüler duruma neden olmamaktadır.

$$U^{-1} = \begin{bmatrix} 1 & -\overline{U}_{0,1} & -\overline{U}_{0,2} & \cdots & -\overline{U}_{0,i+1} & \cdots & -\overline{U}_{0,n} \\ 0 & 1 & -\overline{U}_{1,2} & \cdots & \vdots & \ddots & \vdots \\ \vdots & \ddots & 1 & \ddots & \vdots & \ddots & -\overline{U}_{i,n} \\ \vdots & \ddots & \ddots & \ddots & -\overline{U}_{i,i+1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & -\overline{U}_{n-1,n} \\ 0 & 0 & \cdots & \cdots & 0 & \cdots & 1 \end{bmatrix} \tag{3.55}$$

$$D^{-1} = \begin{bmatrix} \hat{\bar{I}}_1^{-1} & & & 0 \\ & \hat{\bar{I}}_2^{-1} & & \\ & & \ddots & \\ 0 & & & \hat{\bar{I}}_n^{-1} \end{bmatrix}$$

İleri dinamik modellemenin kompakt yapıdan özyinelemeli yapıya dönüştürülmesi işlemi ilerleyen başlıkların altında açıklanmıştır. Özyinelemeli yapıda ileri dinamik iki aşamadan meydana gelmektedir. Öncelikle tork yayılımının gerçekleştirildiği geri özyineleme bölümü yapılmakta ardından ivmelerin ve hızların yayılımının yapıldığı ileri özyineleme yapılmaktadır. İki adet aşamanın oluşması ileri dinamik denklem çözümünün denklem 3.56 şeklinde yazılması ile mümkün olmaktadır[9].

$$\begin{aligned} \hat{\varphi} &= U^{-1}\ (\tau - h) \\ \tilde{\varphi} &= D^{-1}\hat{\varphi} \\ \ddot{q} &= U^{-T}\tilde{\varphi} \end{aligned} \tag{3.56}$$

Üç denklem halinde ifade edilen ileri dinamik kompakt denkleminde ilk denklem geri özyinelemeye neden olmakta, ikinci denklem "D" matrisinin köşegen olmasından kaynaklı olarak herhangi bir özyineleme şeması oluşturmamakta, üçüncü

denklem ise ileri özyineleme şemasının ortaya çıkmasını sağlamaktadır. Ters dinamikte olduğu gibi kompakt yapıda kalınarak giriş torklarına karşılık, eklem ivmeleri hesaplanabilir. Ancak önceki duruma benzer bir şekilde bu yapıda da işlem yükü önemli ölçüde fazladır. Bu fazlalığın büyük bölümü kompakt yapıda aynı işlemlerin tekrar tekrar gerçekleştirilmesidir. Özyinelemeli yapıya geçildiği takdirde gereksiz bir biçimde aynı işlemlerin gerçekleştirilmesi engellenmektedir.

3.4.2.3. Geri özyineleme

Bu aşama, üç denklem ile ifade edilen robot kompakt ileri dinamik denkleminin, ilk iki denkleminin verilen giriş torklarına göre hesaplanmasını sağlamaktadır. Geri özyinelemeyi meydana getiren asıl denklem, ilk denklemdir. İkinci denklem doğrudan, özyinelemeye ihtiyaç duymadan hesaplanmaktadır.

$$\hat{\varphi} = U^{-1}\,\varphi$$
$$\tilde{\varphi} = D^{-1}\hat{\varphi} \tag{3.57}$$

"$\varphi = (\tau - h)$" eşitliği oluşturularak bu kompakt denklemlerin ayrıştırılması kolaylaştırılmıştır. "h" terimi "Coriolis", yerçekimi ve sürtünme matrislerini içermektedir. "h" teriminin hesaplanması, sabit taban durumunda ters dinamik uygulanarak bulunabilir[9]. Bahsedilen iki denklem kompakt olarak, özyineleme şeması oluşturulmadan hesaplanabilir. Ancak bu yapı aynı işlemlerin tekrar tekrar hesaplanmasını içerdiği için maliyetlidir. Bu kompakt yapıdan özyinelemeli yapıya geçiş için ilk denklemin modül düzeyinde açılmış hali incelenmelidir.

$$\begin{bmatrix} \hat{\bar{\varphi}}_1 \\ \vdots \\ \hat{\bar{\varphi}}_i \\ \vdots \\ \hat{\bar{\varphi}}_n \end{bmatrix} = \begin{bmatrix} 1 & -\bar{U}_{1,2} & -\bar{U}_{1,3} & \cdots & -\bar{U}_{1,i} & \cdots & -\bar{U}_{1,n} \\ 0 & 1 & -\bar{U}_{2,3} & \cdots & \vdots & \ddots & \vdots \\ \vdots & \ddots & 1 & \ddots & \vdots & \ddots & -\bar{U}_{i,n} \\ \vdots & \ddots & \ddots & \ddots & -\bar{U}_{i-1,i} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & -\bar{U}_{n-1,n} \\ 0 & 0 & \cdots & \cdots & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} \bar{\varphi}_1 \\ \vdots \\ \bar{\varphi}_i \\ \vdots \\ \bar{\varphi}_n \end{bmatrix} \tag{3.58}$$

Matris formu açılıdığı takdirde, "1", "i" ve "n" nolu terimler denklem takımı 3.59 şeklinde elde edilmektedir.

$$\hat{\bar{\varphi}}_1 = \bar{\varphi}_1 - \bar{U}_{1,2}\,\bar{\varphi}_2 - \cdots - \bar{U}_{1,i}\,\bar{\varphi}_i - \cdots - \bar{U}_{1,n}\bar{\varphi}_n$$

$$\hat{\bar{\varphi}}_i = \bar{\varphi}_i - \bar{U}_{i,i+1}\,\bar{\varphi}_{i+1} - \cdots - \bar{U}_{i,n}\bar{\varphi}_n \tag{3.59}$$

$$\hat{\bar{\varphi}}_n = \bar{\varphi}_n$$

Bir önceki bölümde, "I" robot eylemsizlik matrisi ayrıştırılması aşamasında "$\bar{U}_{i,j}$" terimleri açıklanmıştı. Bu denklikler, denklem 3.59 ile açıklanan denklemlere eklenirse denklem takımı 3.60'a dönüşür.

$$\begin{aligned}\hat{\bar{\varphi}}_1 &= \bar{\varphi}_1 - \bar{N}_1^{\,T}\,\bar{A}_{2,1}^{\,T}\,\bar{\psi}_2\,\bar{\varphi}_2 - \cdots - \bar{N}_1^{\,T}\,\bar{A}_{i,1}^{\,T}\,\bar{\psi}_i\,\bar{\varphi}_i - \cdots \\ &\quad - \bar{N}_1^{\,T}\,\bar{A}_{n,1}^{\,T}\,\bar{\psi}_n\,\bar{\varphi}_n \\ \hat{\bar{\varphi}}_i &= \bar{\varphi}_i - \bar{N}_i^{\,T}\,\bar{A}_{i+1,i}^{\,T}\,\bar{\psi}_{i+1}\,\bar{\varphi}_{i+1} - \cdots - \bar{N}_i^{\,T}\,\bar{A}_{n,i}^{\,T}\,\bar{\psi}_n\,\bar{\varphi}_n \\ \hat{\bar{\varphi}}_n &= \bar{\varphi}_n\end{aligned} \tag{3.60}$$

Burada görüldüğü üzere modül sayısı arttıkça toplam ifadeleri de artmaktadır. Ayrıca toplam ifadelerinde bulunan çarpanlar incelendiğinde, bu çarpanlar birçok ortak terimi barındırmaktadır. Bu da aynı işlemlerin tekrar tekrar yapıdığı anlamına gelmektedir. İşlemleri azaltmak adına "$\tilde{\bar{\eta}}_i$" konsepti kompakt yapıya Saha tarafından eklenmiştir[9].

$$\tilde{\bar{\eta}}_i = \sum_{\lambda\,\epsilon\,\xi_i} \bar{A}_{\lambda,i}^{\,T}\,\bar{\eta}_\lambda \qquad \bar{\eta}_\lambda = \bar{\psi}_\lambda\,\hat{\bar{\varphi}}_\lambda + \tilde{\bar{\eta}}_\lambda \tag{3.61}$$

"$\tilde{\bar{\eta}}_i$" konsepti eklendiğinde, ifadelerden de görüldüğü üzere ilgili modüle ilişkin "$\hat{\bar{\varphi}}_i$" değişkeni bir üst modül ifadelerine bağlandığı için bir geri özyineleme algoritması ortaya çıkmıştır.

Bu konseptin eklenmiş olduğu modül seviyesi denklemleri, "n" ve "i" nolu modüller için denklem takımı 3.62 ile aktarılmıştır.

$$\begin{aligned}\hat{\bar{\varphi}}_n &= \bar{\varphi}_n \\ \hat{\bar{\varphi}}_i &= \bar{\varphi}_i - \bar{N}_i^{\,T}\,\tilde{\bar{\eta}}_i \quad , \quad \tilde{\bar{\eta}}_i = \sum_{\lambda\,\epsilon\,\xi_i} \bar{A}_{\lambda,i}^{\,T}\,\bar{\eta}_\lambda \quad , \quad \bar{\eta}_\lambda = \bar{\psi}_\lambda\,\hat{\bar{\varphi}}_\lambda + \tilde{\bar{\eta}}_\lambda\end{aligned} \tag{3.62}$$

Görüldüğü üzere kompakt yapının açılmış halinde herhangi bir özyineleme bulunmazken, "$\tilde{\tilde{\eta}}_i$" konsepti ile hem işlem sayısı azaltılmış, hemde alt modüllerin hesaplanması, üst modüllere bağlanarak geri özyineleme şeması oluşturulmuştur.

Modül seviyesinde açılan kompakt yapı, aynı şekilde link seviyesine açılmalıdır ki daha etkili bir geri özyineleme algoritması elde edilsin. Bunun için i nolu modüle ilişkin "$\tilde{\tilde{\eta}}_i$" konsepti eklenmiş kompakt yapı link seviyesinde "η^i", "$\eta - 1^i$" ve "k" nolu linkler için açılırsa denklem 3.63, 3.64 ve 3.65 elde edilir.

$$\hat{\varphi}_{\eta^i} = \varphi_{\eta^i} + p_{\eta^i}{}^T \tilde{\eta}_{\eta^i}, \quad \tilde{\eta}_{\eta^i} = \sum_{\beta \in \xi_{\eta^i}} A_{\beta,\eta^i}{}^T \eta_\beta \quad , \quad \eta_\beta = \psi_\beta \hat{\varphi}_\beta + \tilde{\eta}_\beta \tag{3.63}$$

$$\hat{\varphi}_{\eta-1^i} = \varphi_{\eta-1^i} + p_{\eta-1^i}{}^T A_{\eta^i,\eta-1^i}{}^T \psi_{\eta^i} \varphi_{\eta^i} + p_{\eta-1^i}{}^T A_{\eta^i,\eta-1^i}{}^T \tilde{\eta}_{\eta^i}$$

$$\tilde{\eta}_{\eta^i} = \sum_{\beta \in \xi_{\eta^i}} A_{\beta,\eta^i}{}^T \eta_\beta \qquad \eta_\beta = \psi_\beta \hat{\varphi}_\beta + \tilde{\eta}_\beta \tag{3.64}$$

$$\hat{\varphi}_{k^i} = \varphi_{k^i} + p_{k^i}{}^T A_{k+1^i,k^i}{}^T \psi_{k+1^i} \varphi_{k+1^i} + \cdots + p_{k^i}{}^T A_{\eta^i,k^i}{}^T \tilde{\eta}_{\eta^i}$$

$$\tilde{\eta}_{\eta^i} = \sum_{\beta \in \xi_{\eta^i}} A_{\beta,\eta^i}{}^T \eta_\beta \qquad \eta_\beta = \psi_\beta \hat{\varphi}_\beta + \tilde{\eta}_\beta \tag{3.65}$$

Modül seviyesinde yapılan işlem sayısını azaltma yöntemi link seviyesine uygulanmadığı takdirde hesaplanması gerekli denklemler yukarda gösterilmiştir. Burada da aynı işlemlerin birden fazla yerde yapılması söz konusudur. Bunun önüne geçmek adına link seviyesinde "$\tilde{\eta}_{k^i}$" konsepti oluşturulmuştur[9].

$$\tilde{\eta}_{k^i} = \sum_{\beta \in \xi_{\eta^i}} A_{\beta,k^i}{}^T \eta_\beta \qquad \eta_\beta = \psi_\beta \hat{\varphi}_\beta + \tilde{\eta}_\beta \tag{3.66}$$

Bu konsept eklendiğinde elde edilen link seviyesi denklemleri, denklem 3.67, 3.68 ve 3.69 ile verilmiştir.

$$\hat{\varphi}_{\eta^i} = \varphi_{\eta^i} + {p_{\eta^i}}^T \tilde{\eta}_{\eta^i}, \quad \tilde{\eta}_{\eta^i} = \sum_{\beta \in \xi_{\eta^i}} {A_{\beta,\eta^i}}^T \eta_\beta \quad , \quad \eta_\beta = \psi_\beta \hat{\varphi}_\beta + \tilde{\eta}_\beta \tag{3.67}$$

$$\begin{aligned} \hat{\varphi}_{\eta-1^i} &= \varphi_{\eta-1^i} + {p_{\eta-1^i}}^T \tilde{\eta}_{\eta-1^i}, \\ \tilde{\eta}_{\eta-1^i} &= {A_{\eta^i,\eta-1^i}}^T \eta_{\eta^i} \qquad \eta_{\eta^i} = \psi_{\eta^i} \hat{\varphi}_{\eta^i} + \tilde{\eta}_{\eta^i} \end{aligned} \tag{3.68}$$

$$\begin{aligned} \hat{\varphi}_{k^i} &= \varphi_{k^i} + {p_{k^i}}^T \tilde{\eta}_{k^i}, \\ \tilde{\eta}_{k^i} &= {A_{k+1^i,k^i}}^T \eta_{k+1^i} \qquad \eta_{k+1^i} = \psi_{k+1^i} \hat{\varphi}_{k+1^i} + \tilde{\eta}_{k+1^i} \end{aligned} \tag{3.69}$$

Modül seviyesinin doğrudan kullanılması durumunda üst modülün linklerinin etkileri birden fazla kez hesaplanmak durumunda kalınırken, link seviyesinde oluşturulan geri özyineleme ile bu hesaplama bir kez modülün son linki için yapılmaktadır. Diğer linkler bir üstlerin bulunan linklere bağlı yazıldığı için bu etkiler yeniden hesaplanmaz. Ayrıca "A" Twist yayılım matrislerinin kullanımı da önemli ölçüde azaltılmaktadır. Sonuç olarak link seviyesinde minimum işlem sayısıyla geri özyineleme algoritması oluşturulmuştur.

İkinci denklem olan "$\tilde{\varphi} = D^{-1}\hat{\varphi}$" herhangi bir fazladan işlem içermemektedir. Bu ifade "1", "i" ve "n" nolu modüller için açıldığında bu durum görülmektedir.

$$\begin{aligned} \tilde{\hat{\varphi}}_1 &= {\hat{\hat{I}}_1}^{-1} \hat{\hat{\varphi}}_1 \\ \tilde{\hat{\varphi}}_i &= {\hat{\hat{I}}_i}^{-1} \hat{\hat{\varphi}}_i \\ \tilde{\hat{\varphi}}_n &= {\hat{\hat{I}}_n}^{-1} \hat{\hat{\varphi}}_n \end{aligned} \tag{3.70}$$

Denklem 3.70'deki ifadeler robot seviyesinde hesaplanabilir. Bu ifadelerin verilmesiyle geri özyineleme tamamlanmış olmaktadır. Özyineleme ile ilgili sürekli bahsedildiği üzere işlemlerin link seviyesinde yapılması etkili bir algoritma için gereklidir.

3.4.2.4. İleri özyineleme

Robot kompakt ileri dinamik denkleminin üç denklem halinde yazılmasının ardından ilk iki denklemin geri özyinelemeyi, son denklemin ise ileri özyinelemeyi oluşturduğu belirtilmişti. Üçüncü ve eklem ivmelerinin hesaplandığı denklem, denklem 3.71 ile verilmiştir.

$$\ddot{q} = U^{-T}\tilde{\varphi} \tag{3.71}$$

Bu kompakt yapı modül seviyesinde incelenirse denklem 3.72'deki eşitlik ile karşılaşılır.

$$\begin{bmatrix} \ddot{q}_1 \\ \vdots \\ \ddot{q}_i \\ \vdots \\ \ddot{q}_n \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & \cdots & 0 \\ \bar{U}_{1,2}{}^T & 1 & 0 & \cdots & \vdots & \ddots & \vdots \\ \bar{U}_{1,3}{}^T & \bar{U}_{2,3}{}^T & 1 & \ddots & \vdots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 & \ddots & \vdots \\ \bar{U}_{1,i}{}^T & \ddots & \ddots & \bar{U}_{i-1,i}{}^T & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \bar{U}_{1,n}{}^T & \cdots & \cdots & \bar{U}_{i-1,n}{}^T & \cdots & \bar{U}_{n-1,n}{}^T & 1 \end{bmatrix} \begin{bmatrix} \tilde{\bar{\varphi}}_1 \\ \vdots \\ \tilde{\bar{\varphi}}_i \\ \vdots \\ \tilde{\bar{\varphi}}_n \end{bmatrix} \tag{3.72}$$

Matris formundaki bu ifade daha detaylı incelendiğinde eklem torkları modül seviyesinde "1", "i" ve "n" nolu modüller için verilen denklem 3.73, 3.74 ve 3.75'deki işlemlerin gerçekleştirilmesi ile elde edilebilmektedir.

$$\ddot{\bar{q}}_1 = \tilde{\bar{\varphi}}_1 \tag{3.73}$$

$$\ddot{\bar{q}}_i = \bar{U}_{1,i}{}^T \tilde{\bar{\varphi}}_1 + \bar{U}_{2,i}{}^T \tilde{\bar{\varphi}}_2 + \cdots + \bar{U}_{i-1,i}{}^T \tilde{\bar{\varphi}}_{i-1} + \tilde{\bar{\varphi}}_i \tag{3.74}$$

$$\ddot{\bar{q}}_n = \bar{U}_{1,n}{}^T \tilde{\bar{\varphi}}_1 + \bar{U}_{2,n}{}^T \tilde{\bar{\varphi}}_2 + \cdots + \bar{U}_{n-1,n}{}^T \tilde{\bar{\varphi}}_{n-1} + \tilde{\bar{\varphi}}_n \tag{3.75}$$

Bu ifadede ortak terimler fazla belirgin olmadığı için "$\bar{U}_{i,j}$" ifadelerinin karşılıkları yazılmalıdır. Denklem 3.76, 3.77 ve 3.78'de, bu ifadenin eklenmiş hali bulunmaktadır.

$$\ddot{\bar{q}}_1 = \tilde{\bar{\varphi}}_1 \tag{3.76}$$

$$\ddot{\bar{q}}_i = \bar{\psi}_i{}^T \bar{A}_{i,1} \bar{N}_1 \tilde{\bar{\varphi}}_1 + \bar{\psi}_i{}^T \bar{A}_{i,2} \bar{N}_2 \tilde{\bar{\varphi}}_2 + \cdots + \bar{\psi}_i{}^T \bar{A}_{i,i-1} \bar{N}_{i-1} \tilde{\bar{\varphi}}_{i-1} + \tilde{\bar{\varphi}}_i \tag{3.77}$$

$$\ddot{\bar{q}}_n = \bar{\psi}_n{}^T \bar{A}_{n,1} \bar{N}_1 \tilde{\bar{\varphi}}_1 + \bar{\psi}_n{}^T \bar{A}_{n,2} \bar{N}_2 \tilde{\bar{\varphi}}_2 + \cdots + \bar{\psi}_n{}^T \bar{A}_{n,n-1} \bar{N}_{n-1} \tilde{\bar{\varphi}}_{n-1} + \tilde{\bar{\varphi}}_n \tag{3.78}$$

Denklem 3.76, 3.77 ve 3.78'de bulunan ortak terimler işlem sayısını arttırmaktadır. Saha'nın öne sürdüğü "$\tilde{\bar{\mu}}_i$" konsepti ile modül seviyesinde ileri özyineleme şeması oluşturularak işlem sayısı azaltılmaktadır.

$$\tilde{\bar{\mu}}_i = \bar{A}_{i,i_p}\,\bar{\mu}_{i_p}\,, \quad \bar{\mu}_{i_p} = \bar{N}_{i_p}\,\ddot{\bar{q}}_{i_p} + \tilde{\bar{\mu}}_{i_p} \tag{3.79}$$

"$\tilde{\bar{\mu}}_i$" konsepti kompakt yapının açılmış haline uygulanırsa, "1", "i" ve "n" nolu modüllere ilişkin ifadeler denklem 3.80, 3.81 ve 3.82'de görüldüğü şekildedir.

$$\ddot{\bar{q}}_1 = \tilde{\bar{\varphi}}_1 \tag{3.80}$$

$$\ddot{\bar{q}}_i = \tilde{\bar{\varphi}}_i - \bar{\psi}_i{}^T\,\tilde{\bar{\mu}}_i \quad , \quad \tilde{\bar{\mu}}_i = \bar{A}_{i,i_p}\,\bar{\mu}_{i_p}\,, \quad \bar{\mu}_{i_p} = \bar{N}_{i_p}\,\ddot{\bar{q}}_{i_p} + \tilde{\bar{\mu}}_{i_p} \tag{3.81}$$

$$\ddot{\bar{q}}_n = \tilde{\bar{\varphi}}_n - \bar{\psi}_n{}^T\,\tilde{\bar{\mu}}_n \quad , \quad \tilde{\bar{\mu}}_n = \bar{A}_{n,n_p}\,\bar{\mu}_{n_p}\,, \quad \bar{\mu}_{n_p} = \bar{N}_{n_p}\,\ddot{\bar{q}}_{n_p} + \tilde{\bar{\mu}}_{n_p} \tag{3.82}$$

Modül düzeyinde oluşturulan ileri özyineleme şeması diğer bölümlerde olduğu gibi link seviyesine göre fazladan işlemler içermektedir. Bunu göstermek için modül seviyesinde bulunan matrisler incelenmelidir.

Ana denkleme ilişkin matris ve vektörler denklem 3.83'deki şekildedir.

$$\tilde{\bar{\varphi}}_i = \begin{bmatrix} \tilde{\varphi}_{1^i} \\ \vdots \\ \tilde{\varphi}_{k^i} \\ \vdots \\ \tilde{\varphi}_{\eta^i} \end{bmatrix}, \qquad \bar{\psi}_i{}^T = \begin{bmatrix} \psi_{1^i} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \psi_{\eta^i} \end{bmatrix}, \qquad \tilde{\bar{\mu}}_i = \begin{bmatrix} \tilde{\mu}_{1^i} \\ \vdots \\ \tilde{\mu}_{k^i} \\ \vdots \\ \tilde{\mu}_{\eta^i} \end{bmatrix} \tag{3.83}$$

Geri kalan iki denkleme ilişkin matris ve vektörler ise denklem 3.84'de görüldüğü şekliyle açılabilir.

$$\bar{A}_{i,i_p} = \begin{bmatrix} 0 & \cdots & A_{1^i,\eta_{pre}{}^{i_p}} \\ 0 & \cdots & A_{2^i,\eta_{pre}{}^{i_p}} \\ \vdots & \cdots & \vdots \\ 0 & \cdots & A_{\eta^i,\eta_{pre}{}^{i_p}} \end{bmatrix}, \quad \bar{\mu}_{i_p} = \begin{bmatrix} \mu_{1^{i_p}} \\ \vdots \\ \vdots \\ \mu_{\eta^{i_p}} \end{bmatrix}$$

$$\bar{N}_{i_p} = \begin{bmatrix} p_{1^{i_p}} & 0 & \cdots & \cdots & 0 \\ A_{2^{i_p},1^{i_p}}\,p_{1^{i_p}} & p_{2^{i_p}} & 0 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ A_{\eta-1^{i_p},1^{i_p}}\,p_{1^{i_p}} & A_{\eta-1^{i_p},2^{i_p}}\,p_{2^{i_p}} & \ldots & p_{\eta-1^{i_p}} & 0 \\ A_{\eta^{i_p},1^{i_p}}\,p_{1^{i_p}} & A_{\eta^{i_p},2^{i_p}}\,p_{2^{i_p}} & \cdots & A_{\eta^{i_p},\eta-1^{i_p}}\,p_{\eta-1^{i_p}} & p_{\eta^{i_p}} \end{bmatrix} \tag{3.84}$$

$$\ddot{\bar{q}}_{i_p} = \begin{bmatrix} \ddot{q}_{1^{i_p}} \\ \vdots \\ \vdots \\ \ddot{q}_{\eta^{i_p}} \end{bmatrix}, \quad \tilde{\bar{\mu}}_{i_p} = \begin{bmatrix} \tilde{\mu}_{1^{i_p}} \\ \vdots \\ \vdots \\ \tilde{\mu}_{\eta^{i_p}} \end{bmatrix}$$

İleri dinamik ileri özyineleme bölümü modül seviyesinde bulunan tüm denklemler ve denklemlerin içinde bulunan matris ve vektörler açıklandıktan sonra modül seviyesi ifadeleri daha ayrıntılı incelenebilir. "1", "k" ve "η" nolu linkler için elde edilen denklemler, denklem 3.85, 3.86 ve 3.87'de görülmektedir.

$$\ddot{q}_{1^i} = \tilde{\varphi}_{1^i} - \psi_{1^i} A_{1^i,\eta_{pre}{}^{i_p}} \left[A_{\eta^{i_p},1^{i_p}} p_{1^{i_p}} \ddot{q}_{1^{i_p}} + \cdots + A_{\eta^{i_p},\eta-1^{i_p}} p_{\eta-1^{i_p}} \ddot{q}_{\eta-1^{i_p}} + \ddot{q}_{\eta^{i_p}} + \tilde{\mu}_{\eta^{i_p}}\right] \tag{3.85}$$

$$\ddot{q}_{k^i} = \tilde{\varphi}_{k^i} - \psi_{k^i} A_{k^i,\eta_{pre}{}^{i_p}} \left[A_{\eta^{i_p},1^{i_p}} p_{1^{i_p}} \ddot{q}_{1^{i_p}} + \cdots + A_{\eta^{i_p},\eta-1^{i_p}} p_{\eta-1^{i_p}} \ddot{q}_{\eta-1^{i_p}} + \ddot{q}_{\eta^{i_p}} + \tilde{\mu}_{\eta^{i_p}}\right] \tag{3.86}$$

$$\ddot{q}_{\eta^i} = \tilde{\varphi}_{\eta^i} - \psi_{\eta^i} A_{\eta^i,\eta_{pre}{}^{i_p}} \left[A_{\eta^{i_p},1^{i_p}} p_{1^{i_p}} \ddot{q}_{1^{i_p}} + \cdots + A_{\eta^{i_p},\eta-1^{i_p}} p_{\eta-1^{i_p}} \ddot{q}_{\eta-1^{i_p}} + \ddot{q}_{\eta^{i_p}} + \tilde{\mu}_{\eta^{i_p}}\right] \tag{3.87}$$

Denklem 3.85, 3.86 ve 3.87'deki denklemlerin hesaplanması eğer modül seviyesinde kalınırsa zorunludur. Görüldüğü üzere link ivmeleri hesaplanırken, her link için aynı işlemler tekrar edilmektedir. Bunu engellemek için Saha, "$\tilde{\mu}_i$" konseptini, ileri dinamik üçüncü denklem çözümü link seviyesine uygulamıştır[9]. Bu konsept denklem 3.88 ile gösterilebilir.

$$\ddot{q}_{k^i} = \tilde{\varphi}_{k^i} - \psi_{k^i} \tilde{\mu}_{k^i}, \quad \tilde{\mu}_{k^i} = A_{k^i,k-1^i} \mu_{k-1^i}$$
$$\mu_{k-1^i} = p_{k-1^i} \ddot{\theta}_{k-1^i} + \tilde{\mu}_{k-1^i} \tag{3.88}$$

Oluşturulan bu ifadelerle, robotun bir modülüne ilişkin eklem ivmelerinin hesaplanmasında, aynı işlemlerin tekrar tekrar yapılması engellenmektedir. Ayrıca önceki modülün son linkinden şu anki modülün linklerine olan Twist yayılım matrisleri de önemli ölçüde yer kaplamaktadır. Bu konsept ile algoritmada kullanılan Twist yayılım matrislerinin sayıları da azaltılmaktadır. Öne sürülen "$\tilde{\mu}_i$" konsepti algoritmaya aktarılırsa "i" nolu modül, "1", "k" ve "η" nolu linklerin ivmeleri denklem 3.89, 3.90 ve 3.91 ile elde edilebilir.

$$\ddot{q}_{1^i} = \tilde{\varphi}_{1^i} - \psi_{1^i} \tilde{\mu}_{1^i}, \quad \tilde{\mu}_{1^i} = A_{1^i,\eta^{i_p}} \mu_{\eta^{i_p}}, \quad \mu_{\eta^{i_p}} = p_{\eta^{i_p}} \ddot{\theta}_{\eta^{i_p}} + \tilde{\mu}_{\eta^{i_p}} \tag{3.89}$$

$$\ddot{q}_{k^i} = \tilde{\varphi}_{k^i} - \psi_{k^i}\,\tilde{\mu}_{k^i}, \quad \tilde{\mu}_{k^i} = A_{k^i,k-1^i}\,\mu_{k-1^i}, \quad \mu_{k-1^i} = p_{k-1^i}\,\ddot{\theta}_{k-1^i} + \tilde{\mu}_{k-1^i} \tag{3.90}$$

$$\ddot{q}_{\eta^i} = \tilde{\varphi}_{\eta^i} - \psi_{\eta^i}\,\tilde{\mu}_{\eta^i}, \quad \tilde{\mu}_{\eta^i} = A_{\eta^i,\eta-1^i}\,\mu_{\eta-1^i}, \quad \mu_{\eta-1^i} = p_{\eta-1^i}\,\ddot{\theta}_{\eta-1^i} + \tilde{\mu}_{\eta-1^i} \tag{3.91}$$

Etkili bir hale getirilen algoritmada görüldüğü üzere aynı işlemlerden kaçınılmıştır. Ayrıca, bir çok 6x6 matris çarpımına neden olan Twist yayılım matrislerinin uzak linkler için tanımlanması durumu ortadan kaldırılmıştır. Sadece bir önceki linke bağlı olma özelliğini getiren "$\tilde{\mu}_i$" konseptinin oluşturulması ile sabit taban ileri dinamik algoritması tamamlanmış olmaktadır.

3.5. Serbest Tabanlı Sistemler

Serbest tabanlı robotik sistemler için elde edilen dinamik modeller kompakt formda, sabit tabanlı robotik sistemlerle aynıdır.

$$I\,\ddot{q} + h = \tau\,, \quad h = C\,\dot{q} + \tau^F \tag{3.92}$$

Bu ifade sabit tabanda özyinelemeli algoritmaların nasıl oluşturulduğu sorusuna cevap aranırken ayrıntılı bir biçimde matris elemanlarına ayrıştırılarak incelenmişti. Sabit tabanda, taban hareket etmediği için, eklem ivmelerinden etkilenmemekte ve sonuç olarak robot hareket denkleminde "0" indisli ifadeler görülmemektedir. Serbest taban durumu için robot hareket denklemi incelendiğinde denklem 3.93'deki eşitlikle karşılaşılmaktadır[9].

$$\begin{bmatrix} I_0 & I_{\theta 0}^T \\ I_{\theta 0} & I_\theta \end{bmatrix} \begin{bmatrix} \ddot{q}_0 \\ \ddot{q}_\theta \end{bmatrix} = \begin{bmatrix} 0 \\ \tau_\theta \end{bmatrix} - \begin{bmatrix} h_0 \\ h_\theta \end{bmatrix} \tag{3.93}$$

Bu ifade iki eşitlik halinde ayrıştırılırsa denklem 3.94 ve 3.95 elde edilir.

$$I_0\,\ddot{q}_0 + I_{\theta 0}^T\,\ddot{q}_\theta = 0 - h_0 \tag{3.94}$$

$$I_{\theta 0}\,\ddot{q}_0 + I_\theta\,\ddot{q}_\theta = \tau_\theta - h_\theta \tag{3.95}$$

Hareket denklemi eklem torkları yalnız bırakılacak şekilde düzenlenirse denklem 3.96'ya ulaşılır.

$$\tau_\theta = I_{\theta 0}\,\ddot{q}_0 + I_\theta\,\ddot{q}_\theta + h_\theta \tag{3.96}$$

Görüldüğü üzere eklem torkları, denkleme girdi olarak verilen eklem ivmeleri ve serbest halde bulunan taban ivmelerine bağlıdır.

$$\ddot{q}_0 = {I_0}^{-1}\,(-I_{\theta 0}^{T}\;\ddot{q}_\theta\; - h_0) \tag{3.97}$$

Denklem 3.96 ve 3.97'de bulunan "$I_{\theta 0}$", taban ivmelerinin eklem torklarına olan etkisini aktarır. "I_θ" ise robotun sabit haldeki eylemsizlik matrisini içerir. "h" ifadeleri "Coriolis", yerçekimi ve sürtünme matrislerini içerir.

Eklem torklarının, taban ivmesinden etkilenmesi, sabit tabanda uygulanan özyinelemeli yapılarda değişiklik yapılmasına neden olmaktadır. Bu, hem ileri dinamik hem de ters dinamik için geçerlidir.

3.5.1. Serbest tabanlı sistem için ters dinamik modeli

Serbest tabana yönelik elde edilen hareket denkleminde, yeni bir tanımlama yapılması gerekmektedir. Eklem torkları denklem 3.98'de bulunan eşitlikle hesaplanmaktadır.

$$\tau_\theta = I_{\theta 0}\,\ddot{q}_0 + I_\theta\,\ddot{q}_\theta + h_\theta \tag{3.98}$$

Burada denklem 3.99'daki tanımlama yapılırsa, sabit taban algoritması üzerinden bir algoritma oluşturulabilir.

$$\tilde{h}_\theta = I_\theta\,\ddot{q}_\theta + h_\theta \tag{3.99}$$

Görüldüğü üzere, serbest taban hareket denklemi içinde, sabit taban hareket denklemi de bulunmaktadır. "$\tilde{h}_\theta$" ifadesi, serbest taban ivme etkisini içermeyen eklem torklarını vermektedir. Bu nedenle bu ifade sabit taban ters dinamik algoritması ile hesaplanabilir. Serbest taban ivme etkilerini içeren eklem torklarının hesaplanması için gerekli olan "$I_{\theta 0}$" ve "$\ddot{q}_0$" ifadeleri ise geri özyineleme bölümünde ve sabit taban ters dinamik algoritmasına ek olarak gelen ileri özyineleme ile elde edilir.

3.5.1.1. İleri özyineleme

İleri özyineleme bölümü sabit taban ters dinamik ileri özyineleme bölümünde olduğu gibi hesaplanır. Serbest taban durumundan kaynaklı olarak bu bölüme "0" indisli "Twist", "Twist" türevi ve "Wrench" ifadeleri eklenmektedir. Sabit tabandan farklı olarak, "Twist" ifadesi özel bir durum barındırmaktadır.

$$t_0 = P_0 \, \dot{q}_0 \tag{3.100}$$

Denklem 3.100'deki ifade de dikkat edilmesi gereken nokta, "$\dot{q}_0$" vektörünün "6x1" boyutunda olmasıdır. Taban hareketinin serbestlik derecesinin "6" olmasından kaynaklı olan "$\dot{q}_0$" vektörünün "6x1" boyutunda olması durumu, taban "Twist" ifadesini robot eklem "Twist" ifadelerinden ayıran ilk etmendir. İkinci etmen ise "P_0" matrisidir. "P_0" matrisi taban ivmelerini, taban "Twist" vektörüne dönüştüren Twist yayılım matrisidir.

$$P_0 = \begin{bmatrix} L_0 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.101}$$

Twist yayılım matrisinde bulunan "L_0" elemanı, taban açısal hızlarının tanımlandığı Euler açıları cinsinden oluşturulan rotasyon matrisidir. Bu matris sabit koordinat eksenine bağlı olarak tabanın oryantasyonunun hesaplanmasında kullanılır.

3.5.1.2. Geri özyineleme

Serbest taban durumu ters dinamik modeli geri özyineleme bölümü, sabit taban ters dinamik geri özyineleme bölümünü içermekte, ve buna ek olarak "$I_{\theta 0}$"nun hesaplanmasıyla ilgili işlemlerde bulunmaktadır[9].

$$\tilde{\bar{h}}_i = \bar{N}_i^T \, \tilde{\bar{w}}_i \, , \quad \tilde{\bar{w}}_i = \bar{w}_i + \bar{A}_{i+1,i}^T \, \tilde{\bar{w}}_{i+1} \tag{3.102}$$

Önceden de söylendiği gibi "$\tilde{\bar{h}}_i$", serbest taban ivmelerinin etkisini içermeyen eklem torklarını verir. "$I_{\theta 0}$" taban ivmelerinin etkisini eklemlere aktaran matristir. Bu matris tabandan tüm eklemlere tanımlıdır. Yapısı denklem 3.103 şeklindedir.

$$I_{\theta 0} = [\bar{I}_{1,0} \quad \cdots \quad \bar{I}_{i,0} \quad \cdots \quad \bar{I}_{n,0}] \tag{3.103}$$

"$\bar{I}_{i,0}$"nun hesaplanması ise önceden tanımlandığı şekilde yapılmaktadır[9].

$$\bar{I}_{i,0} = \bar{N}_0{}^T \; \bar{A}_{i,0}^T \, \bar{\tilde{M}}_i \, \bar{N}_i \tag{3.104}$$

"$\bar{I}_{i,0}$" teriminin bu aşamada hesaplanmasının nedeni, "$\bar{\tilde{M}}_i$" matrisini barındırıyor olmasıdır.

$$\bar{\tilde{M}}_i = \bar{M}_i + \sum_{j \in \xi_i} \bar{A}_{j,i}^T \, \bar{M}_j \, \bar{A}_{j,i} \tag{3.105}$$

"$\bar{\tilde{M}}_i$" matrisi, üst modüllere bağlı yapısından dolayı geri özyineleme bölümünde hesaplanmaktadır. Tüm bölümlerde olduğu gibi modül mertebesinde kalınması fazladan işlemlerin yapılmasına neden olmaktadır. Bu nedenle "$\bar{\tilde{M}}_i$" ifadesinin link seviyesindeki karşılığının kullanılması daha etkili bir algoritma için şarttır.

Modül seviyesindeki "$\bar{\tilde{M}}_i$" ifadesinin link seviyesinde açılmış hali denklem 3.106 ile verilmiştir.

$$\tilde{M}_{k^i} = M_{k^i} + A_{k+1^i,k^i}^T \, M_{k+1^i} A_{k+1^i,k^i} + \cdots + A_{\eta^n,\eta-1^n}^T \, M_{\eta^n} \, A_{\eta^n,\eta-1^n} \tag{3.106}$$

Saha bu ifadedeki ortak noktaları göz önüne alarak link seviyesinde denklem 3.107'deki ifadeyi oluşturmuştur.

$$\tilde{M}_i = M_i + \sum_{j \in \xi_i} A_{j,i}^T \, M_j \, A_{j,i} \tag{3.107}$$

Bu tanımlama ile geri özyineleme bölümü tamamlanmış olmaktadır.

3.5.1.3. İleri özyineleme

Geri özyineleme bölümünde hesaplanan "$\bar{I}_{i,0}$" terimleri taban ivme etkisini eklem torklarına aktarmak için bu aşamada kullanılır. Bunun için öncelikle taban ivmelerinin hesaplanması gereklidir.

$$\ddot{\bar{q}}_0 = -\bar{I}_{0,0}^{-1} \, \bar{\tilde{h}}_0 \tag{3.108}$$

Serbest taban ters dinamik başlığında ilk olarak hareket denklemi verilmişti. Bu hareket denklemi incelendiğinde eklem torklarının, taban ivmelerine bağlı olduğu görülmekteydi. İlk ileri özyineleme ve geri özyineleme sonrası, serbest taban etkisini içermeyen ancak robot eklemlerinin birbirine olan etkilerini içeren eklem torkları

hesaplandı. İkinci ileri özyinelemenin ilk aşamasında ise eklem torklarının tabana yansıyan ivmesi hesaplandı. Böylece tüm veriler elde edildikten sonra serbest taban etkisini de içeren eklem torklarının elde edilme aşamasına gelindi.

$$\bar{\tau}_i = -\bar{I}_{i,0}^T \, \ddot{\bar{q}}_0 + \tilde{\bar{h}}_i \tag{3.109}$$

Son aşamadaki işlemler esasında herhangi bir özyinelemeye ihtiyaç duymamaktadır. Çünkü denklemler incelendiğinde herhangi bir alt modüllere veya üst modüllere bağımlılık bulunmamaktadır. Bu nedenle işlemlerin tamamı modül mertebesinde yapılabilir.

3.5.2. Serbest tabanlı sistem için ileri dinamik modeli

Ters dinamik modelinde olduğu gibi, ileri dinamik modellemede de kompakt form ileri dinamik hareket denklemi serbest taban ve sabit tabanda aynıdır.

$$\ddot{q} = I^{-1}(\tau - h) \tag{3.110}$$

İleri dinamik hareket denklemi, ters dinamik üzerinden gidilerek hesaplanmakta ve robot eylemsizlik matrisinin tersini içerdiği için işlem yükü durumuyla karşı karşıya kalmaktadır. Sabit taban robotik sistemlerde bahsedildiği üzere serbest tabanlı sistemlerde de "UDU^T" ayrıştırması ile eylemsizlik matrisinin tersi alınmakta ve matris tersi alınmasının getirdiği işlem yükünden böylece kaçınılmaktadır.

Serbest tabanın bulunması durumunda ileri dinamik sabit taban algoritmasına ek olarak bir ileri özyineleme eklenmektedir. Bunun nedeni olarak Saha, Coriolis, yerçekimi ve sürtünme kuvveti etkilerini içeren "h" vektörünün hesaplanma ihtiyacını öne sürmektedir.

İlk olarak ileri özyineleme ile hesaplanan "h" vektörü sonrasında, sabit tabanda olduğu gibi aşağıdaki üç denklemin çözümünü içeren sırasıyla geri ve ileri özyineleme aşamaları yapılmaktadır.

$$\begin{aligned} \hat{\varphi} &= U^{-1}\,(\tau - h) \\ \tilde{\varphi} &= D^{-1}\hat{\varphi} \\ \ddot{q} &= U^{-T}\tilde{\varphi} \end{aligned} \tag{3.111}$$

Denklem 3.111'deki matrisler sabit tabanda "0" indisli elemanları içermezken serbest tabanda "0" indisli elemanlar içermektedir.

3.5.2.1. İleri özyineleme

İleri özyinelemede "h" vektörünün hesaplanması için gerekli olan "$\overline{w}_i$" hesaplanır. Burada "$\overline{w}_i$" "Wrench" vektörü, "Twist" ve "Twist" türevi vektörlerini içerdiği için önceki bölümlerde anlatıldığı üzere ileri özyineleme ile hesaplanmaktadır. Yine önceden bahsedildiği üzere, "$\overline{w}_i$" vektörlerinin modül seviyesinde kalınmayıp, link seviyesinde hesaplanması en etkili yöntemdir.

Saha "h" vektörlerinin hesaplanma problemini ileri özyineleme bölümü ekleyerek çözmesinin nedenini, hesaplama zamanındaki iyileştirme olarak belirtmektedir[9]. İleri özyinelemede bulunan önemli nokta, ileri dinamik hesaplaması yapıldığı ve sonuçta eklem ivmeleri hesaplanacağı için özyinelemeli yapının ilk adımı olan ileri özyineleme bölümünde eklem ivmeleri "0" olarak alınır[9].

Sonuç olarak buradaki algoritma, eklem ivmelerinin "0" alınması haricinde, serbest taban ters dinamik ilk ileri özyineleme bölümüyle aynıdır.

3.5.2.2. Geri özyineleme

Geri özyineleme, sabit tabanla aynı şekilde ilk ve ikinci denklemin çözümünü herhangi bir singüler duruma yol açacak matris tersi almadan vermektedir. İlk iki denklem ile "$\hat{\varphi}$" ve "$\tilde{\varphi}$" hesaplanır. Sabit tabandan farklı olarak, indisler en üst modüllerden, tabanda dahil olacak şekilde (0) oluşturulur. Aynı denklemler hem modül seviyesinde hemde link seviyesinde geçerlidir. En az işlem sayısına sahip olan link seviyesi denklemi, denklem 3.112, 3.113 ve 3.114 ile tekrar verilsin.

$$\hat{\varphi}_{\eta^i} = \varphi_{\eta^i} + p_{\eta^i}{}^T \left(\tilde{\eta}_{\eta^i} + w_{\eta^i} \right), \quad \tilde{\eta}_{\eta^i} = \sum_{\beta \,\epsilon\, \xi_{\eta^i}} A_{\beta,\eta^i}{}^T \, \eta_\beta$$
$$\eta_\beta = \psi_\beta \, \hat{\varphi}_\beta + \tilde{\eta}_\beta \tag{3.112}$$

$$\hat{\varphi}_{\eta-1^i} = \varphi_{\eta-1^i} + p_{\eta-1^i}{}^T \left(\tilde{\eta}_{\eta-1^i} + w_{\eta-1^i} \right), \; \tilde{\eta}_{\eta-1^i} = A_{\eta^i,\eta-1^i}{}^T \, \eta_{\eta^i}$$
$$\eta_{\eta^i} = \psi_{\eta^i} \, \hat{\varphi}_{\eta^i} + \tilde{\eta}_{\eta^i} \tag{3.113}$$

$$\hat{\varphi}_{k^i} = \varphi_{k^i} + {p_{k^i}}^T \left(\tilde{\eta}_{k^i} + w_{k^i} \right), \;\; \tilde{\eta}_{k^i} = {A_{k+1^i,k^i}}^T \eta_{k+1^i} \qquad (3.114)$$

$$\eta_{k+1^i} = \psi_{k+1^i} \hat{\varphi}_{k+1^i} + \tilde{\eta}_{k+1^i}$$

İlk denklemin çözümünü gösteren denklem 3.112, 3.113 ve 3.114'deki link seviyesindeki ifadelerle, ikinci denklem çözülebilir. İkinci denklemde sabit tabanda olduğu gibi bir matris tersi alma işlemi vardır fakat bu matris her zaman pozitif tanımlı olduğundan herhangi bir singüler duruma yol açmamaktadır. Özyineleme şeması içermeyen ikinci denklem modül seviyesinde çözülebilir.

$$\begin{aligned} \tilde{\ddot{\varphi}}_1 &= {\hat{\hat{I}}_1}^{-1} \; \hat{\ddot{\varphi}}_1 \\ \tilde{\ddot{\varphi}}_i &= {\hat{\hat{I}}_i}^{-1} \; \hat{\ddot{\varphi}}_i \\ \tilde{\ddot{\varphi}}_n &= {\hat{\hat{I}}_n}^{-1} \; \hat{\ddot{\varphi}}_n \end{aligned} \qquad (3.115)$$

Böylece eklem torklarının birbirine olan etkileri ve tüm eklem torklarının tabanda ne ölçüde bir torka neden olduğu hesaplanmış oldu. Sonraki bölümde de taban da dahil olmak üzere tüm ivmeler hesaplanmaktadır.

3.5.2.3. İleri özyineleme

Üçüncü denklem ile ilişkili olan bu özyineleme bölümü, sabit tabanla benzer şekilde çözülmektedir. İlk olarak, sabit tabandan farklı olarak, serbest taban ivmeleri hesaplanmaktadır.

$$\ddot{\tilde{q}}_0 = \ddot{\tilde{\varphi}}_0 \qquad (3.116)$$

İleri özyineleme kısmı kompakt formda kalınarak çözülebilir ancak önceki aşamalarda olduğu gibi hesaplama zamanı ve işlem sayısı bakımından olumsuz bir durum oluşturur. Bu işlemlerin "$\tilde{\mu}_i$" konsepti ile, sabit taban ileri dinamik ileri özyineleme aşamasında olduğu gibi link seviyesinde yapılması en hızlı sonucu vermektedir. Algoritma sabit taban ileri dinamik ileri özyineleme ile aynıdır. Tek fark indislerin "0" dan başlamasıdır.

$$\begin{aligned} \ddot{q}_{1^i} &= \tilde{\varphi}_{1^i} - \psi_{1^i} \tilde{\mu}_{1^i}, \quad \tilde{\mu}_{1^i} = A_{1^i,\eta^{i_p}} \mu_{\eta^{i_p}} \\ \mu_{\eta^{i_p}} &= p_{\eta^{i_p}} \ddot{\theta}_{\eta^{i_p}} + \tilde{\mu}_{\eta^{i_p}} \end{aligned} \qquad (3.117)$$

$$\ddot{q}_{k^i} = \tilde{\varphi}_{k^i} - \psi_{k^i}\,\tilde{\mu}_{k^i}, \quad \tilde{\mu}_{k^i} = A_{k^i,k-1^i}\,\mu_{k-1^i}$$
$$\mu_{k-1^i} = p_{k-1^i}\,\ddot{\theta}_{k-1^i} + \tilde{\mu}_{k-1^i} \tag{3.118}$$

$$\ddot{q}_{\eta^i} = \tilde{\varphi}_{\eta^i} - \psi_{\eta^i}\,\tilde{\mu}_{\eta^i}, \quad \tilde{\mu}_{\eta^i} = A_{\eta^i,\eta-1^i}\,\mu_{\eta-1^i}$$
$$\mu_{\eta-1^i} = p_{\eta-1^i}\,\ddot{\theta}_{\eta-1^i} + \tilde{\mu}_{\eta-1^i} \tag{3.119}$$

Tüm ifadelerin tamamlanmasının ardından, sabit taban ve serbest tabanlı karmaşık yapılı robotik sistemlerin ileri ve ters dinamik modelleri, hesaplama zamanı da göz önüne alınarak elde edilmiştir.

3.6. Simülasyon Çalışmaları

Kinematik analiz bölümünde, kinematik simülasyonları gerçekleştirilen sabit ve sabit olmayan tabanlı örnekler için, bu aşamada ileri ve ters dinamik analizleri gerçekleştirilmiştir. Simülasyonlarda basitlik açısından yine, örnekler düzlemsel tutulmuş ve birden fazla serbestlik derecesine sahip eklemler kullanılmamıştır.

Dinamik simülasyonların doğruluğunu göstermek amacıyla, MATLAB harici bir dinamik simülatör programına ihtiyaç duyulmuştur. Bu program ADAMS olarak belirlenmiş ve örnekler ADAMS ortamında oluşturulmuştur. İleri dinamik için hem MATLAB algoritmasında hem de ADAMS ortamında aynı torklar eklemlere uygulanmış ve eklem ivmeleri karşılaştırılmıştır. Ters dinamikte ise yine aynı şekilde hem MATLAB algoritmasında hem de ADAMS ortamında aynı ivmeler eklemlere uygulanmış ve hesaplanan tork değerleri kıyaslanmıştır.

3.6.1. Robotik tutucu problemi

Kinematik analizi yapılan robotik tutucunun yapısı hatırlatılırsa; 3 modülden meydana gelen bu robotik sistemde, “1” nolu modül bir link, “2” nolu modül üç link ve “3” nolu modül ise iki linkten oluşmaktadır. İlk olarak ters dinamik analizi incelenmiş hemen ardından ise ileri dinamik çözümü aktarılmıştır. Her iki analizde de ayrıntılı bir biçimde özyineleme şemasına ait eşitlikler ve simülasyon çıktıları verilmiştir.

Simülasyonların doğruluğunu göstermek için ADAMS çıktıları da grafik olarak verilmiş ve algoritma ile ADAMS çıktıları arasındaki farkları içeren tüm eklemlere ilişkin grafiklerde gösterilmiştir.

3.6.1.1. Ters dinamik

Robotik tutucu ters dinamik algoritması, MATLAB ortamında m-file sentaksı ile oluşturulmuştur. Bu algoritmanın oluşturulması için dinamik modelleme bölümünde anlatılan özyinelemeli yapıya ait eşitlikler, robotik tutucu yapısı için sentezlenmiştir. İlk olarak elde edilen bu eşitlikler ayrıntılı bir biçimde aktarılmış, ardından da simülasyon şeması ve doğruluk testleri verilmiştir.

İleri Özyineleme

İleri özyineleme, teoriye bağlı kalınarak ilk olarak modül seviyesinde, sonrasında ise link seviyesinde açıklanmıştır. 3 adet modülden meydana gelen robotik tutucu sistemi için modüller arası ileri özyineleme ilişkisi aşağıdaki şekilde verilebilir.

- Modül 1 için modüller arası ilişki:
  - $\overline{t_1} = \overline{N_1}\,\overline{\dot{q}_1}$
  - $\overline{\dot{t}_1} = \overline{\dot{N}_1}\,\overline{\dot{q}_1} + \overline{N_1}\,\overline{\ddot{q}_1}$
  - $\overline{w_1}^* = \overline{M_1}\,\overline{\dot{t}_1} + \overline{\Omega_1}\,\overline{M_1}\,\overline{E_1}\,\overline{t_1}$
- Modül 2 için modüller arası ilişki:
  - $\overline{t_2} = \overline{A_{2,1}}\,\overline{t_1} + \overline{N_2}\,\overline{\dot{q}_2}$
  - $\overline{\dot{t}_2} = \overline{\dot{A}_{2,1}}\,\overline{t_1} + \overline{A_{2,1}}\,\overline{\dot{t}_1} + \overline{\dot{N}_2}\,\overline{\dot{q}_2} + \overline{N_2}\,\overline{\ddot{q}_2}$
  - $\overline{w_2}^* = \overline{M_2}\,\overline{\dot{t}_2} + \overline{\Omega_2}\,\overline{M_2}\,\overline{E_2}\,\overline{t_2}$
- Modül 3 için modüller arası ilişki:
  - $\overline{t_3} = \overline{A_{3,1}}\,\overline{t_1} + \overline{N_3}\,\overline{\dot{q}_3}$
  - $\overline{\dot{t}_3} = \overline{\dot{A}_{3,1}}\,\overline{t_1} + \overline{A_{3,1}}\,\overline{\dot{t}_1} + \overline{\dot{N}_3}\,\overline{\dot{q}_3} + \overline{N_3}\,\overline{\ddot{q}_3}$
  - $\overline{w_3}^* = \overline{M_3}\,\overline{\dot{t}_3} + \overline{\Omega_3}\,\overline{M_3}\,\overline{E_3}\,\overline{t_3}$

Modül seviyesinde verilen matris ve vektörlerin içeriğine bakılırsa aşağıda verilen eşitliklere ulaşılmaktadır. Ve bu eşitlikler incelendiğinde link seviyesinde bir

özyinelemeye ihtiyaç duymadan modül seviyesinde (fazladan işlem yaparak) ters dinamik ileri özyinelemeyi çözmek mümkündür.

- Modül 1 için:

$$\overline{w_1}^* = [{w_1}^*], \quad \overline{M_1} = [M_1], \qquad \dot{\overline{t_1}} = [\dot{t_1}], \qquad \overline{\Omega_1} = [\Omega_1], \qquad \overline{E_1} = [E_1],$$

- Modül 2 için:

$$\overline{w_2}^* = \begin{bmatrix} {w_{1^2}}^* \\ {w_{2^2}}^* \\ {w_{3^2}}^* \end{bmatrix}, \qquad \overline{M_2} = \begin{bmatrix} M_{1^2} & 0 & 0 \\ 0 & M_{2^2} & 0 \\ 0 & 0 & M_{3^2} \end{bmatrix},$$

$$\dot{\overline{t_2}} = \begin{bmatrix} \dot{t_{1^2}} \\ \dot{t_{2^2}} \\ \dot{t_{3^2}} \end{bmatrix}, \quad \overline{\Omega_2} = \begin{bmatrix} \Omega_{1^2} & 0 & 0 \\ 0 & \Omega_{2^2} & 0 \\ 0 & 0 & \Omega_{3^2} \end{bmatrix},$$

$$\overline{E_2} = \begin{bmatrix} E_{1^2} & 0 & 0 \\ 0 & E_{2^2} & 0 \\ 0 & 0 & E_{3^2} \end{bmatrix}$$

- Modül 3 için:

$$\overline{w_3}^* = \begin{bmatrix} {w_{1^3}}^* \\ {w_{2^3}}^* \end{bmatrix}, \qquad \overline{M_3} = \begin{bmatrix} M_{1^3} & 0 \\ 0 & M_{2^3} \end{bmatrix}, \qquad \dot{\overline{t_3}} = \begin{bmatrix} \dot{t_{1^3}} \\ \dot{t_{2^3}} \end{bmatrix}, \quad \overline{\Omega_3} = \begin{bmatrix} \Omega_{1^3} & 0 \\ 0 & \Omega_{2^3} \end{bmatrix},$$

$$\overline{E_3} = \begin{bmatrix} E_{1^3} & 0 \\ 0 & E_{2^3} \end{bmatrix}$$

Dinamik modelleme bölümünde bahsedildiği üzere yukarıda açıklanan modül seviyesindeki denklemler yerine link seviyesindeki denklemler daha az işlem içerdiğinden simülasyonda kullanılmıştır. Bu durumda link seviyesindeki eşitlikler verilmelidir.

- Modül 1 için linkler arası ilişki:
  - $t_{1^1} = p_{1^1}\, q_{1^1}^{\cdot}$
  - $\dot{t_{1^1}} = \dot{p_{1^1}}\, q_{1^1}^{\cdot} + p_{1^1}\, \ddot{q_{1^1}}$
  - ${w_{1^1}}^* = M_{1^1}\, \dot{t_{1^1}} + \Omega_{1^1}\, M_{1^1}\, E_{1^1}\, t_{1^1}$
- Modül 2 için linkler arası ilişki:
  - Link 1 için:
    - $t_{1^2} = A_{1^2,1^1}\, t_{1^1} + p_{1^2}\, q_{1^2}^{\cdot}$
    - $\dot{t_{1^2}} = \dot{A}_{1^2,1^1}\, t_{1^1} + A_{1^2,1^1}\, \dot{t}_{1^1} + \dot{p_{1^2}}\, q_{1^2}^{\cdot} + p_{1^2}\, \ddot{q_{1^2}}$

- $w_{1^2}{}^* = M_{1^2}\, \dot{t}_{1^2} + \Omega_{1^2}\, M_{1^2}\, E_{1^2}\, t_{1^2}$

- Link 2 için:
  - $t_{2^2} = A_{2^2,1^2}\, t_{1^2} + p_{2^2}\, \dot{q}_{2^2}$
  - $\dot{t}_{2^2} = \dot{A}_{2^2,1^2}\, t_{1^2} + A_{2^2,1^2}\, \dot{t}_{1^2} + \dot{p}_{2^2}\, \dot{q}_{2^2} + p_{2^2}\, \ddot{q}_{2^2}$
  - $w_{2^2}{}^* = M_{2^2}\, \dot{t}_{2^2} + \Omega_{2^2}\, M_{2^2}\, E_{2^2}\, t_{2^2}$
- Link 3 için:
  - $t_{3^2} = A_{3^2,2^2}\, t_{2^2} + p_{3^2}\, \dot{q}_{3^2}$
  - $\dot{t}_{3^2} = \dot{A}_{3^2,2^2}\, t_{2^2} + A_{3^2,2^2}\, \dot{t}_{2^2} + \dot{p}_{3^2}\, \dot{q}_{3^2} + p_{3^2}\, \ddot{q}_{3^2}$
  - $w_{3^2}{}^* = M_{3^2}\, \dot{t}_{3^2} + \Omega_{3^2}\, M_{3^2}\, E_{3^2}\, t_{3^2}$

- Modül 3 için linkler arası ilişki:
  - Link 1 için:
    - $t_{1^3} = A_{1^3,1^1}\, t_{1^1} + p_{1^3}\, \dot{q}_{1^3}$
    - $\dot{t}_{1^3} = \dot{A}_{1^3,1^1}\, t_{1^1} + A_{1^3,1^1}\, \dot{t}_{1^1} + \dot{p}_{1^3}\, \dot{q}_{1^3} + p_{1^3}\, \ddot{q}_{1^3}$
    - $w_{1^3}{}^* = M_{1^3}\, \dot{t}_{1^3} + \Omega_{1^3}\, M_{1^3}\, E_{1^3}\, t_{1^3}$
  - Link 2 için:
    - $t_{2^3} = A_{2^3,1^3}\, t_{1^3} + p_{2^3}\, \dot{q}_{2^3}$
    - $\dot{t}_{2^3} = \dot{A}_{2^3,1^3}\, t_{1^3} + A_{2^3,1^3}\, \dot{t}_{1^3} + \dot{p}_{2^3}\, \dot{q}_{2^3} + p_{2^3}\, \ddot{q}_{2^3}$
    - $w_{2^3}{}^* = M_{2^3}\, \dot{t}_{2^3} + \Omega_{2^3}\, M_{2^3}\, E_{2^3}\, t_{2^3}$

Tek serbestlik dereceli eklemler içerdiğinden serbestlik derecesi seviyesine inmeye gerek kalmadan tüm eşitlikler elde edilmiştir. Ters dinamik ileri özyineleme kısmı tamamlandıktan sonra bir ileri aşamaya geçilmiştir.

Geri Özyineleme

İleri özyinelemede elde edilen ve robot linklerini birbirinden bağımsız varsayarak hesaplanan "$w_i{}^*$" kuvvet vektörü, DeNOC matrisleri ile, "$\widetilde{w}_i$" vektörüne dönüştürülür. "$\widetilde{w}_i$" kuvvet vektörü, linklerin birbirine olan etkilerini içermektedir. DeNOC matrisleri bu etkileri uçtan tabana doğru yaymayı sağladığı için geri özyineleme algoritması ortaya çıkmaktadır.

Geri özyineleme, teoriye bağlı kalınarak tüm aşamalarda olduğu gibi ilk olarak modül seviyesinde, sonrasında ise link seviyesinde açıklanmıştır. 3 adet modülden meydana gelen robotik tutucu sistemi için modüller arası geri özyineleme ilişkisi aşağıdaki şekilde verilebilir.

- Modül 3 için modüller arası ilişki:
  - $\widetilde{\overline{w}}_3 = \overline{w}_3{}^*$
  - $\bar{\tau}_3 = \bar{N}_3{}^T \widetilde{\overline{w}}_3$
- Modül 2 için modüller arası ilişki:
  - $\widetilde{\overline{w}}_2 = \overline{w}_2{}^*$
  - $\bar{\tau}_2 = \bar{N}_2{}^T \widetilde{\overline{w}}_2$
- Modül 1 için modüller arası ilişki:
  - $\widetilde{\overline{w}}_1 = \overline{w}_1{}^* + \bar{A}_{2,1}{}^T \widetilde{\overline{w}}_2 + \bar{A}_{3,1}{}^T \widetilde{\overline{w}}_3$
  - $\bar{\tau}_1 = \bar{N}_1{}^T \widetilde{\overline{w}}_1$

Modül seviyesindeki ilişkilerde bulunan matris ve vektörler incelendiğinde:

- Modül 3 için:

$$\widetilde{\overline{w_3}} = \begin{bmatrix} \widetilde{w_{1^3}} \\ \widetilde{w_{2^3}} \end{bmatrix}, \qquad \overline{N_3}^T = \begin{bmatrix} {p_{1^3}}^T & (A_{2^3 1^3}\, p_{1^3})^T \\ 0 & {p_{2^3}}^T \end{bmatrix}$$

- Modül 2 için:

$$\widetilde{\overline{w_2}} = \begin{bmatrix} \widetilde{w_{1^2}} \\ \widetilde{w_{2^2}} \\ \widetilde{w_{3^2}} \end{bmatrix}, \qquad \overline{N_2}^T = \begin{bmatrix} {p_{1^2}}^T & (A_{2^2 1^2}\, p_{1^2})^T & (A_{3^2 1^2}\, p_{1^2})^T \\ 0 & {p_{2^2}}^T & (A_{3^2 2^2} p_{2^2})^T \\ 0 & 0 & {p_{3^2}}^T \end{bmatrix}^2$$

- Modül 1 için:

$$\widetilde{\overline{w_1}} = [\widetilde{w_{1^1}}], \qquad \overline{N_1}^T = [{p_{1^1}}^T]$$

Görüldüğü üzere, modül seviyesindeki denklemlerin içeriğinde bulunan matris ve vektörler doğrudan oluşturulup, tork değerleri hesaplanabilir. Ancak bu durum, dinamik modelleme başlığında anlatıldığı üzere gereksiz işlemlerin yapılmasını gerektirmektedir. Bu nedenle link seviyesindeki özyinelemeli yapı verilmelidir.

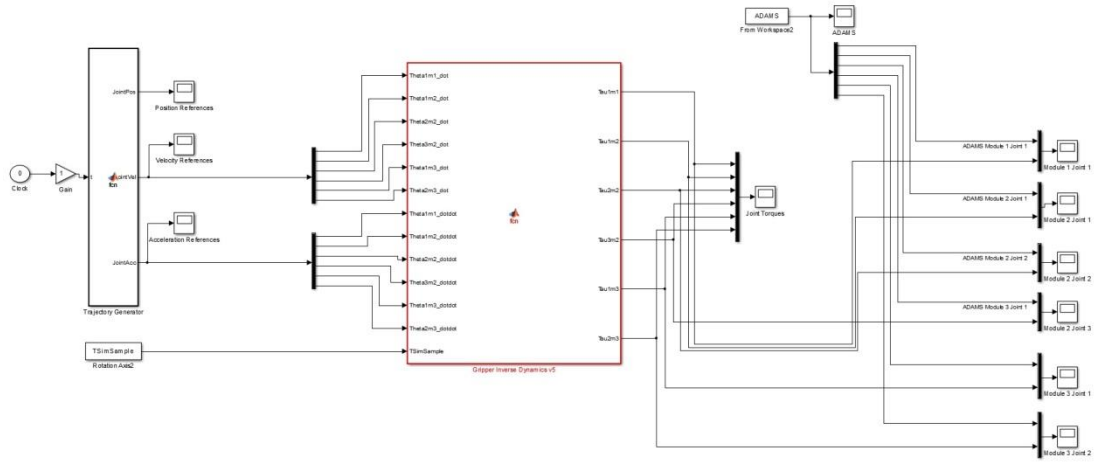
- Modül 3 için:
  - Link 2 için:

        - $\widetilde{w}_{2^3} = {w_{2^3}}^*$
        - ${\tau}_{2^3} = {p_{2^3}}^T \widetilde{w}_{2^3}$
    - Link 1 için:
        - $\widetilde{w}_{1^3} = {w_{1^3}}^* + (A_{2^3 1^3})^T \widetilde{w}_{2^3}$
        - ${\tau}_{1^3} = {p_{1^3}}^T \widetilde{w}_{1^3}$
- Modül 2 için:
    - Link 3 için:
        - $\widetilde{w}_{3^2} = {w_{3^2}}^*$
        - ${\tau}_{3^2} = {p_{3^2}}^T \widetilde{w}_{3^2}$
    - Link 2 için:
        - $\widetilde{w}_{2^2} = {w_{2^2}}^* + (A_{3^2 2^2})^T \widetilde{w}_{3^2}$
        - ${\tau}_{2^2} = {p_{2^2}}^T \widetilde{w}_{2^2}$
    - Link 1 için:
        - $\widetilde{w}_{1^2} = {w_{1^2}}^* + (A_{2^2 1^2})^T \widetilde{w}_{2^2}$
        - ${\tau}_{1^2} = {p_{1^2}}^T \widetilde{w}_{1^2}$
- Modül 1 için:
    - Link 1 için:
        - $\widetilde{w}_{1^1} = {w_{1^1}}^* + (A_{1^2 1^1})^T \widetilde{w}_{1^2} + +(A_{1^3 1^1})^T \widetilde{w}_{1^3}$
        - ${\tau}_{1^1} = {p_{1^1}}^T \widetilde{w}_{1^1}$

Link mertebesinde elde edilen geri özyineleme eşitlikleri ile robotik tutucu probleminin ters dinamik aşaması tamamlanmış ve algoritma yapısı genel anlamda ortaya çıkmıştır.

Simülasyon

Robotik tutucu ters dinamik simülasyonu MATLAB Simulink ortamında yapılmıştır. Doğruluğunu göstermek adına ADAMS ortamında da aynı link uzunlukları ve kütlelerine sahip bir robotik tutucu yapısı oluşturulmuştur. MATLAB Simulink algoritmasının genel görünümü Şekil 3.2'de verilmiştir.



Şekil 3.2 : Robotik tutucu ters dinamik MATLAB Simulink diyagramı

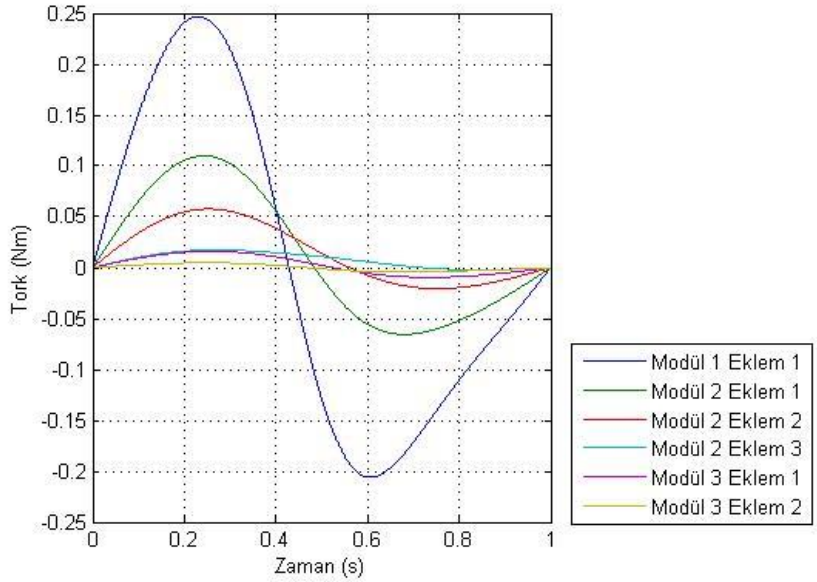
Diyagramda görülen gömülü fonksiyon bloğu tüm ters dinamik algoritmasını içermektedir. "m-file" sentaksında yazılan bu bloğa giriş olarak verilen eklem ivme yörüngeleri denklem 3.120 ile oluşturulmuştur.

$$\ddot{\theta}_i = \frac{2\,\pi}{T_{Periyot}^2}\left(\theta_{final_i} - \theta_{başlangıç_i}\right)\sin\left(\frac{2\,\pi\,t}{T_{Periyot}}\right) \tag{3.120}$$

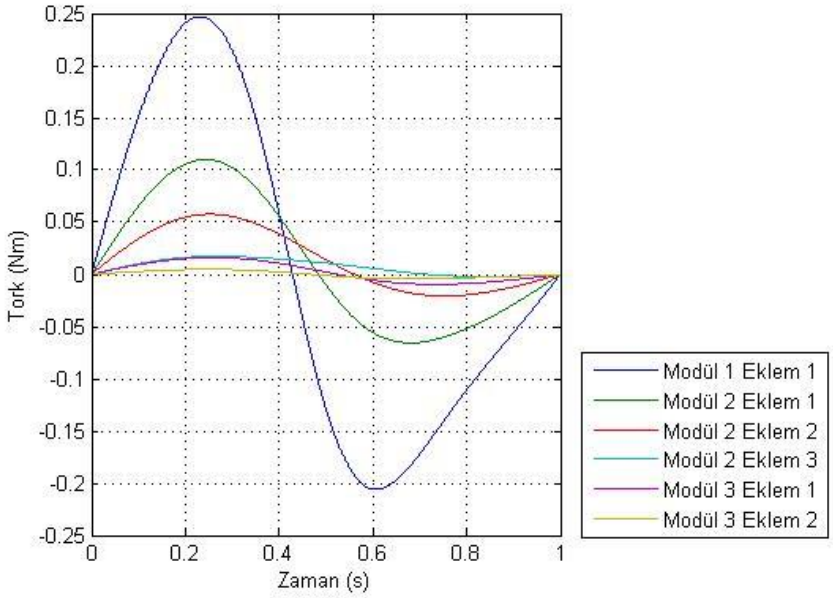
Tablo 3.1 : Eklem başlangıç ve final pozisyonları

	1^1	1^2	2^2	3^2	1^3	2^3
$\theta_{başlangıç_i}$	0	0	0	0	0	0
θ_{final_i}	60	60	60	60	60	-60

Bu yörüngeler hem MATLAB ortamında hem de ADAMS ortamında uygulanmış ve elde edilen tork değerleri çizdirilmiştir. Ayrıca grafikleri üst üste gösterebilmek için MATLAB ortamına, ADAMS dataları alınmıştır.



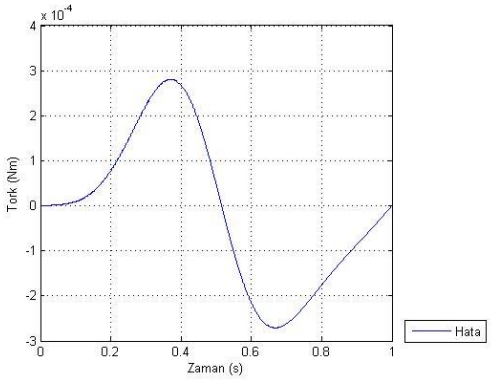
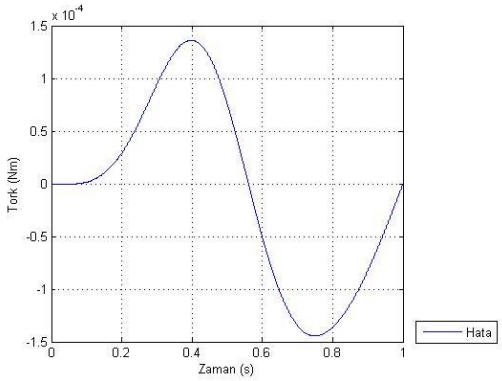
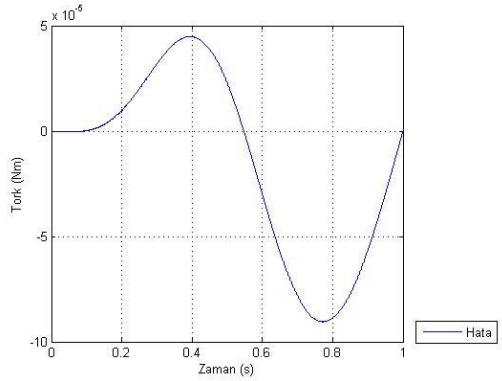
Şekil 3.3 : Robotik tutucu ters dinamik MATLAB algoritması tork çıktıları

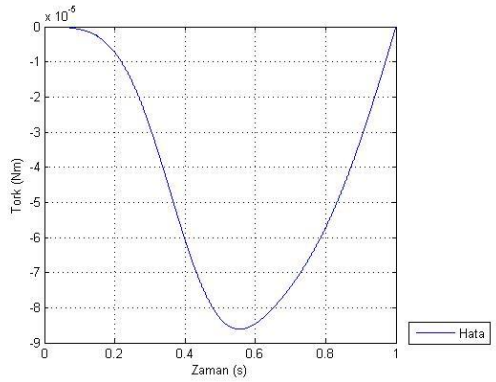
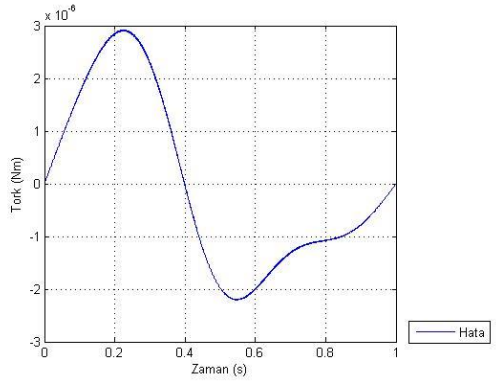
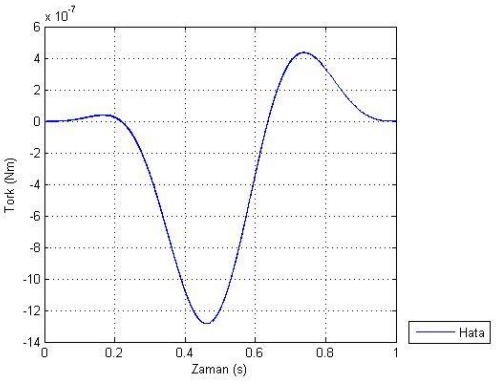


Şekil 3.4 : Robotik tutucu ters dinamik ADAMS tork çıktıları

Şekil 3.3 ve Şekil 3.4'te görüldüğü üzere ADAMS ve MATLAB algoritması tork çıktıları form olarak aynıdır. Bunu daha detaylı görebilmek için her ekleme ait ayrı grafiklerde ADAMS ile MATLAB algoritma sonuçları arasındaki farklar çizdirilmiştir.

Tablo 3.2 : Robotik tutucu ters dinamik eklem tork hesaplama hatası tablosu

Eklem No	Eklem Tork Hesaplama Hatası Grafiği
1^1	
1^2	
2^2	

3^2	
1^3	
2^3	

Tablo 3.2'den de görüldüğü üzere ADAMS ile MATLAB algoritması arasındaki hatalar oldukça düşük mertebelerde kalmaktadır. "$3x10^{-4}$" ile en yüksek hata ilk modülün ilk eklemine aittir. Bu hata da kabul edilebilir mertebede kalmaktadır. Bu sonuçlar ışığında, algoritmanın doğru sonuç verdiği söylenebilir.

3.6.1.2. İleri dinamik

İleri dinamik, robot eklemlerine tork uygulanması halinde nasıl bir yörünge izleyeceğini belirlemektedir. Geri ve ileri özyineleme ile hesaplanan ileri dinamik,

yine teorik olarak modüller arası ve modül içi denklemlerden oluşmaktadır. İleri dinamiğe giriş olarak verilen eklem torkları denklem 3.121 ile belirlenmektedir.

$$\tau = \tau_{input} - (C\,\dot{q} - \tau^F) \tag{3.121}$$

Aslında burada öne sürülen ileri dinamik çözüm sadece analitik olarak robot eylemsizlik matrisinin tersinin alınmasından ibarettir.

$$\tau = I^{-1}\ddot{q}, \qquad I = U\,D\,U^T \tag{3.122}$$

İlk olarak modül seviyesindeki denklemlerin verilmesinden sonra, dinamik modelleme bölümünde anlatıldığı üzere simülasyonlarda kullanılan link seviyesindeki denklemler verilmiştir. Denklemlerin ardından simülasyon şeması ve doğruluk testleri verilmiştir.

Geri Özyineleme

Geri özyineleme, üçe ayrıştırılan robot eylemsizlik matrisinin "U" teriminin tersinin girdi torklarıyla çarpımıyla oluşmaktadır.

$$\hat{\tau} = U^{-1}\,\tau \tag{3.123}$$

Denklemde görülen ters alma işlemi bölüm 3'te anlatıldığı üzere analitik olarak gerçekleştirilmiştir. Modül seviyesinde geri özyineleme şeması şu şekildedir:

- Modül 3 için:
  - $\hat{\bar{\tau}}_3 = \bar{\tau}_3$
  - $\tilde{\bar{\tau}}_3 = \hat{\bar{I}}_3^{\,-1}\,\hat{\bar{\tau}}_3$
  - $\bar{w}_3 = \bar{\Psi}_3\,\hat{\bar{\tau}}_3$
- Modül 2 için:
  - $\hat{\bar{\tau}}_2 = \bar{\tau}_2$
  - $\tilde{\bar{\tau}}_2 = \hat{\bar{I}}_2^{\,-1}\,\hat{\bar{\tau}}_2$
  - $\bar{w}_2 = \bar{\Psi}_2\,\hat{\bar{\tau}}_2$

- Modül 1 için:
  - $\bar{\bar{w}}_1 = \bar{A}_{2,1}{}^T \, \bar{w}_2 + \bar{A}_{2,1}{}^T \, \bar{w}_3$
  - $\hat{\bar{\tau}}_1 = \bar{\tau}_1 - \bar{N}_1{}^T \, \bar{\bar{w}}_1$
  - $\bar{\bar{\tau}}_1 = \hat{\bar{I}}_1^{-1} \, \hat{\bar{\tau}}_1$

Modüller arası denklemlerde üstteki modüllerin torkları ilk olarak "Wrench" kuvvet vektörlerine dönüştürülmekte ardından etki ettikleri modüle indirgenmektedirler. Sonrasında indirgendikleri modüllerde oluşturdukları torklar hesaplanarak, giriş torklarından çıkarılmaktadırlar.

Modül mertebesindeki eşitliklere ait matris ve vektörlerin içerikleri link seviyesinde oluşturulan değişkenlerden meydana gelmektedir. Eğer bu değişkenler ile doğrudan modül seviyesindeki matrisler oluşturulup, modül seviyesinde yukarıda verilen denklemler kullanılırsa, fazla işlem gücüne ihtiyaç duyulur. Bu nedenle link seviyesindeki denklemlerin kullanılması simülasyonda tercih edilmiştir.

- Modül 3 için:
  - Link 2 için:
    - $\hat{\tau}_{2^3} = \tau_{2^3}$
    - $\hat{m}_{2^3} = p_{2^3}^T \, \hat{M}_{2^3} \, p_{2^3}$
    - $\tilde{\tau}_{2^3} = {\hat{\tau}_{2^3}}/{\hat{m}_{2^3}}$
    - $w_{2^3} = \psi_{2^3} \, \hat{\tau}_{2^3}$
  - Link 1 için:
    - $\tilde{w}_{1^3} = A_{2^3 1^3}^T \, w_{2^3}$
    - $\hat{\tau}_{1^3} = \tau_{1^3} - p_{1^3}^T \, \tilde{w}_{1^3}$
    - $\hat{m}_{1^3} = p_{1^3}^T \, \hat{M}_{1^3} \, p_{1^3}$
    - $\tilde{\tau}_{1^3} = {\hat{\tau}_{1^3}}/{\hat{m}_{1^3}}$
    - $w_{1^3} = \psi_{1^3} \, \hat{\tau}_{1^3}$

- Modül 2 için:
  - Link 3 için:
    - $\hat{\tau}_{3^2} = \tau_{3^2}$
    - $\widehat{m}_{3^2} = p_{3^2}^T \widehat{M}_{3^2} p_{3^2}$
    - $\tilde{\tau}_{3^2} = {\hat{\tau}_{3^2}}/{\widehat{m}_{3^2}}$
    - $w_{3^2} = \psi_{3^2} \hat{\tau}_{3^2}$
  - Link 2 için:
    - $\widetilde{w}_{2^2} = A_{3^2 2^2}^T w_{3^2}$
    - $\hat{\tau}_{2^2} = \tau_{2^2} - p_{2^2}^T \widetilde{w}_{2^2}$
    - $\widehat{m}_{2^2} = p_{2^2}^T \widehat{M}_{2^2} p_{2^2}$
    - $\tilde{\tau}_{2^2} = {\hat{\tau}_{2^2}}/{\widehat{m}_{2^2}}$
    - $w_{2^2} = \psi_{2^2} \hat{\tau}_{2^2}$
  - Link 1 için:
    - $\widetilde{w}_{1^2} = A_{2^2 1^2}^T w_{2^2}$
    - $\hat{\tau}_{1^2} = \tau_{1^2} - p_{1^2}^T \widetilde{w}_{1^2}$
    - $\widehat{m}_{1^2} = p_{1^2}^T \widehat{M}_{1^2} p_{1^2}$
    - $\tilde{\tau}_{1^2} = {\hat{\tau}_{1^2}}/{\widehat{m}_{1^2}}$
    - $w_{1^2} = \psi_{1^2} \hat{\tau}_{1^2}$
- Modül 1 için:
  - Link 1 için:
    - $\hat{\tau}_{1^1} = \tau_{1^1} - p_{1^1}^T \widetilde{w}_{1^1},$
    - $\widetilde{w}_{1^1} = A_{1^2 1^1}^T w_{1^2} + A_{1^3 1^1}^T w_{1^3}$
    - $\widetilde{\tau_{1^1}} = {\widehat{\tau_{1^1}}}/{\widehat{m_{1^1}}}$
    - $\widehat{m_{1^1}} = p_{1^1}^T \widehat{M_{1^1}} p_{1^1}$
    - $w_{1^1} = \psi_{1^1} \hat{\tau}_{1^1}$

Tüm modüller için, link düzeyindeki eşitlikler elde edildikten sonra ileri özyinelemeye geçilebilir.

İleri Özyineleme

Bu aşama da, üçe ayrıştırılan robot eylemsizlik matrisinin "U^T" teriminin tersinin çarpımıyla oluşmaktadır.

$$\ddot{q} = U^{-T}\,\tilde{\tau} \tag{3.124}$$

İlk olarak modüller arası aktarılmış, ardından simülasyonda kullanılan link seviyesindeki eşitlikler verilmiştir.

- Modül 1 için:
  - $\ddot{\tilde{q}}_1 = \tilde{\tilde{\tau}}_1$
  - $\bar{\mu}_1 = \overline{N_1}\,\overline{\ddot{q}_1}$
- Modül 2 için:
  - $\ddot{\tilde{q}}_2 = \tilde{\tilde{\tau}}_2 - \overline{\Psi}_2^{\,T}\,\tilde{\tilde{\mu}}_2$
  - $\tilde{\tilde{\mu}}_2 = \bar{A}_{2,1}\,\bar{\mu}_1$
- Modül 3 için:
  - $\ddot{\tilde{q}}_3 = \tilde{\tilde{\tau}}_3 - \overline{\Psi}_3^{\,T}\,\tilde{\tilde{\mu}}_3$
  - $\tilde{\tilde{\mu}}_3 = \bar{A}_{3,1}\,\bar{\mu}_1$

İleri özyinelemede ivme yayılımı gerçekleştirilmektedir. İlk modül sabit bir noktaya bağlı olduğundan geri özyineleme sonucu hesaplanan "$\tilde{\tilde{\tau}}_1$" değeri doğrudan eklem ivmesine atanmıştır. Diğer modüller, 1 nolu modüle bağlı olduğu için, o modüllerin geri özyineleme sonucu elde edilen "$\tilde{\tilde{\tau}}_k$" değerleri, 1 nolu modülün ivmesi ile işleme sokularak (indirgenme) ivmeleri belirlenmiştir.

Simülasyonda kullanılan ve daha az işlem içeren link seviyesi eşitlikler şu şekilde verilebilir:

- Modül 1 için:
  - $\ddot{\theta}_{1^1} = \tilde{\tau}_{1^1}$
  - $\mu_{1^1} = p_{1^1}\,\ddot{\theta}_{1^1}$
- Modül 2 için:
  - Link 1 için:
    - $\ddot{\theta}_{1^2} = \tilde{\tau}_{1^2} - \psi_{1^2}^T\,\tilde{\mu}_{1^2}$
    - $\tilde{\mu}_{1^2} = A_{1^2 1^1}\,\mu_{1^1}$
    - $\mu_{1^2} = p_{1^2}\,\ddot{\theta}_{1^2}$

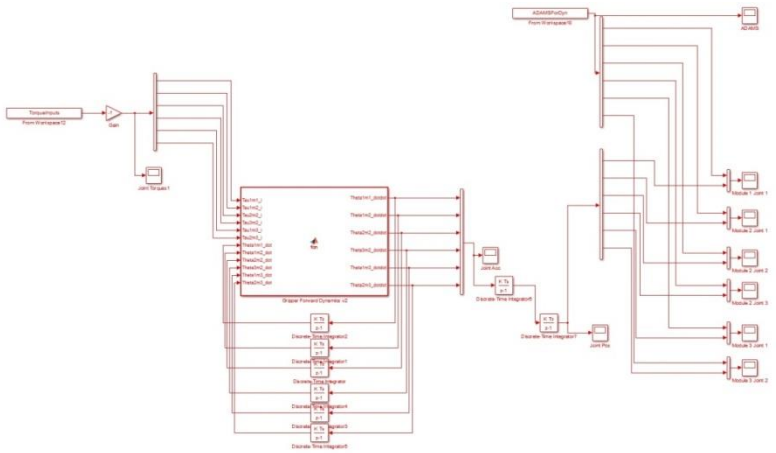
- Link 2 için:
  - $\ddot{\theta}_{2^2} = \tilde{\tau}_{2^2} - \psi_{2^2}^T \tilde{\mu}_{2^2}$,
  - $\tilde{\mu}_{2^2} = A_{2^2 1^2} \mu_{1^2}$
  - $\mu_{2^2} = p_{2^2} \ddot{\theta}_{2^2} + \tilde{\mu}_{2^2}$
- Link 3 için:
  - $\ddot{\theta}_{3^2} = \tilde{\tau}_{3^2} - \psi_{3^2}^T \tilde{\mu}_{3^2}$
  - $\tilde{\mu}_{3^2} = A_{3^2 2^2} \mu_{2^2}$,

- Modül 3 için:
  - Link 1 için:
    - $\ddot{\theta}_{1^3} = \tilde{\tau}_{1^3} - \psi_{1^3}^T \tilde{\mu}_{1^3}$
    - $\tilde{\mu}_{1^3} = A_{1^3 1^1} \mu_{1^1}$
    - $\mu_{1^3} = p_{1^3} \ddot{\theta}_{1^3}$
  - Link 2 için:
    - $\ddot{\theta}_{2^3} = \tilde{\tau}_{2^3} - \psi_{2^3}^T \tilde{\mu}_{2^3}$
    - $\tilde{\mu}_{2^3} = A_{2^3 1^3} \mu_{1^3}$

İleri dinamik özyineleme algoritması, robot eylemsizlik matrisinin ayrıştırılması yoluyla hem modül hem link seviyesinde elde edilmiştir. Simülasyonlarda link seviyesindeki eşitlikler kullanılmıştır. Tüm eşitlikler verildikten sonra simülasyon sonuçlarına geçilebilir.

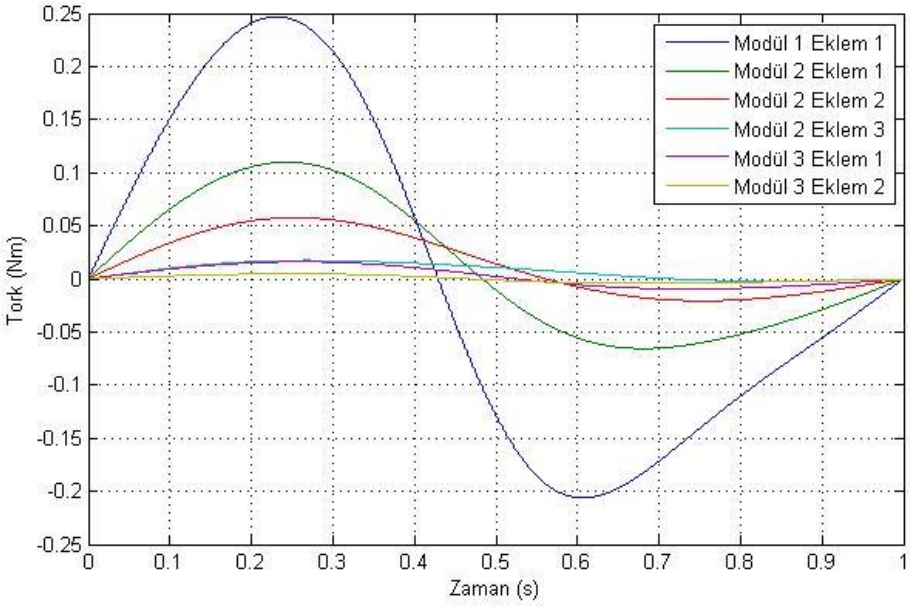
Simülasyon

Robotik tutucu ileri dinamik simülasyonuna ilişkin blok diyagram Şekil 3.5'de görülmektedir.



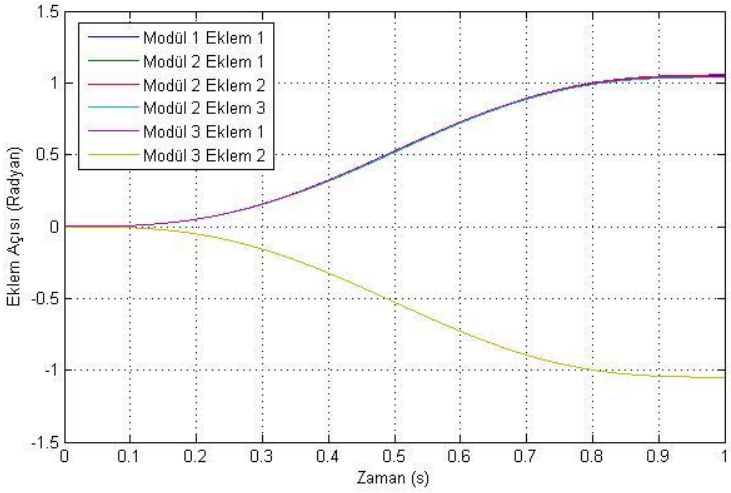
Şekil 3.5 : Robotik tutucu ileri dinamik MATLAB blok diyagram

Ters dinamikte olduğu gibi ileri dinamikte de tüm algoritma “m-file” sentaksında oluşturulmuştur. ADAMS dataları karşılaştırma için MATLAB ortamına alınmıştır. Hem ADAMS hem de MATLAB algoritmasında kullanılan eklem tork ifadeleri Şekil 3.6’da görülmektedir.

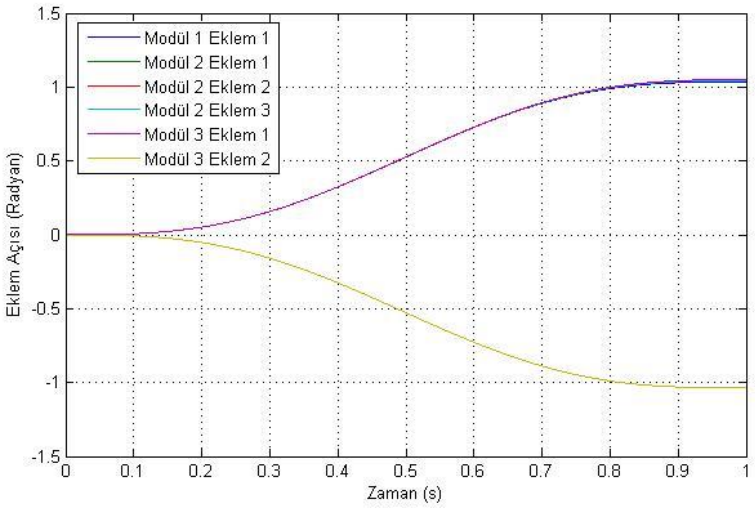


Şekil 3.6 : Robotik tutucu ileri dinamik eklem tork girişleri

Girdi olarak kullanılan bu tork girdileri, ters dinamik çıktısı olarak elde edilmiş nümerik datalarla oluşturulmuştur. O nedenle herhangi bir formüle sahip değildir. Bu girdiler sonucu elde edilen eklem pozisyonları ADAMS ve MATLAB algoritmasında Şekil 3.7 ve Şekil 3.8’de verilmiştir.



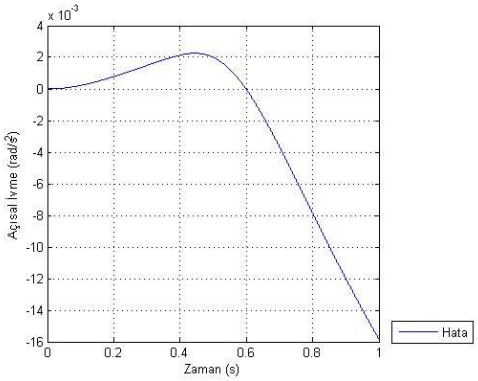
Şekil 3.7 : Robotik tutucu ileri dinamik MATLAB algoritması pozisyon çıktısı

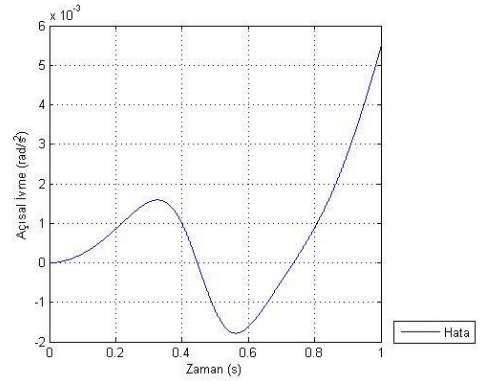
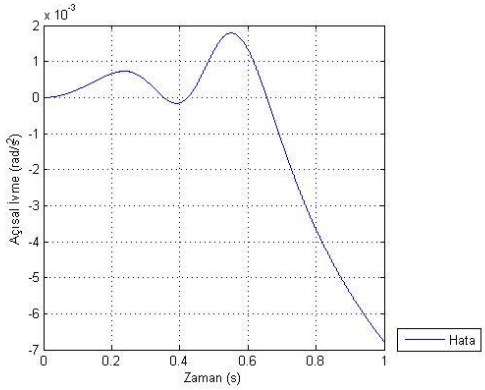
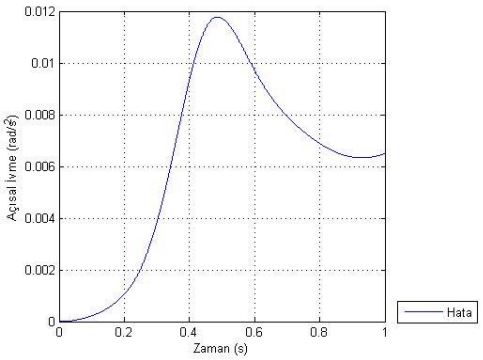


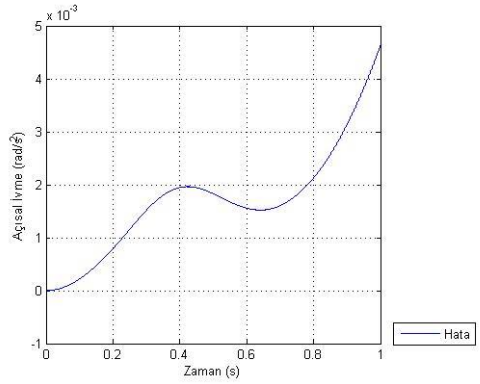
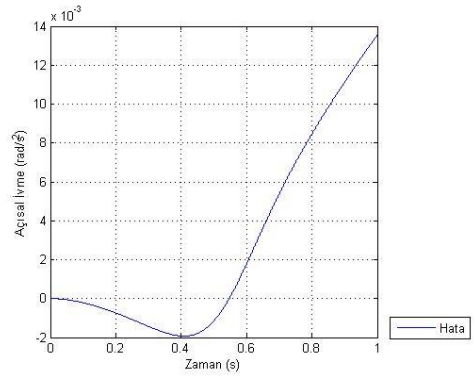
Şekil 3.8 : Robotik tutucu ileri dinamik ADAMS pozisyon çıktısı

Pozisyon dataları, MATLAB algoritmasında ivme çıktıları integre edilerek elde edilmiştir. Eklemlere uygulanan torklara göre hem ADAMS'ta hem de MATLAB algoritmasında "3" nolu modülün "2" nolu linki -60 dereceye giderken, diğer eklemler 60 dereceye doğru gitmektedir. Eklem ivme hatalarına ilişkin oluşturulan Tablo 3.3, eklem bazında grafiklerle oluşturularak, sonuçların yakınlığı daha net bir biçimde gösterilmiştir.

Tablo 3.3 : Robotik tutucu ileri dinamik çıktılarını karşılaştırma tablosu

Eklem No	Eklem İvme Hesaplama Hatası Grafiği
1^1	

1^2	 x 10^-3 Açısal İvme (rad/s²) Zaman (s) Hata
2^2	 x 10^-3 Açısal İvme (rad/s²) Zaman (s) Hata
3^2	 Açısal İvme (rad/s²) Zaman (s) Hata

1^3	
2^3	

Model hataları incelendiğinde en yüksek hatanın "$14x10^{-3}$" Nm olarak modül 3 eklem 2'ye ait olduğu görülmektedir. Bu hata kabul edilebilir mertebede kaldığı için sabit tabanlı ileri dinamik modele ilişkin oluşturulmuş olan robotik tutucu MATLAB algoritması doğrulanmıştır.

3.6.2. Düzlemsel yürüyen robot

Simülasyon örneği olarak belirlenen düzlemsel yürüyen robotun yapısı hatırlatılırsa; bu robot sabit olmayan tabanlı bir yapıya sahiptir. 3 adet modülden oluşmaktadır. Burada taban sabit olmadığı için "0" nolu modül olarak belirlenmiştir. Diğer iki modül robotun bacaklarını oluşturmakta ve her iki modülde üçer adet link içermektedir. Ayrıca tüm eklemler rotasyonel harekete sahip ve bir serbestlik derecelidir.

İlk olarak ters dinamik özyineleme yapısına ilişkin hem modül seviyesinde hem de link seviyesinde eşitlikler aktarılmıştır. Ardından algoritmanın doğruluğunun açıklandığı aşama gelmektedir. Aynı akış, ileri dinamik için de uygulanmıştır.

3.6.2.1. Ters dinamik

Sabit olmayan tabanlı sistemler için bölüm 3'te açıklandığı üzere, ters dinamik algoritması üç aşamadan meydana gelmektedir. İlk olarak bir ileri özyineleme ile "Twist" ve "Wrench" vektörleri hesaplanmakta, ardından gelen geri özyineleme ile taban etkisinin içerilmediği ancak eklem etkilerinin bulunduğu "Wrench" vektörü hesaplanmakta ve hemen ardından gelen bir ileri özyineleme ile taban ivmesi ve bu serbest taban etkisinin dahil edildiği eklem torkları hesaplanmaktadır.

Ters dinamik algoritması MATLAB ortamında m-file sentaksı ile oluşturulmuştur. İlerleyen bölümlerde ayrıntılı eşitlikler ve simülasyon sonuçları aktarılmaktadır.

İleri Özyineleme

İleri özyineleme bölümü ilk olarak "Twist", "Twist" türevi ve "Wrench" vektörleri hesaplanmaktadır. Teoriye bağlı kalınarak öncelikle modül mertebesinde eşitlikler verilmiş ardından ise simülasyonda kullanılan link seviyesindeki özyineleme yapısı verilmiştir.

- Modül 0 için modüller arası ilişki:
  - $t_0 = P_0 \dot{q}_0$
  - $\dot{t}_0 = \dot{P}_0 \dot{q}_0$
  - $w^* = M_0\, \dot{t}_0 + \Omega_0\, M_0\, E_0\, t_0$
- Modül 1 için modüller arası ilişki:
  - $\bar{t}_1 = \bar{A}_{1,0}\, t_0 + \bar{N}_1\, \dot{\bar{q}}_1$
  - $\dot{\bar{t}}_1 = \dot{\bar{A}}_{1,0}\, t_0 + \bar{A}_{1,0}\, \dot{t}_0 + \dot{\bar{N}}_1\, \dot{\bar{q}}_1 + \bar{N}_1\, \ddot{\bar{q}}_1$
  - $\bar{w}_1{}^* = \bar{M}_1\, \dot{\bar{t}}_1 + \bar{\Omega}_1\, \bar{M}_1\, \bar{E}_1\, \bar{t}_1$
- Modül 2 için modüller arası ilişki:
  - $\bar{t}_2 = \bar{A}_{2,0}\, t_0 + \bar{N}_2\, \dot{\bar{q}}_2$
  - $\dot{\bar{t}}_2 = \dot{\bar{A}}_{2,0}\, t_0 + \bar{A}_{2,0}\, \dot{t}_0 + \dot{\bar{N}}_2\, \dot{\bar{q}}_2 + \bar{N}_2\, \ddot{\bar{q}}_2$
  - $\bar{w}_2{}^* = \bar{M}_2\, \dot{\bar{t}}_2 + \bar{\Omega}_2\, \bar{M}_2\, \bar{E}_2\, \bar{t}_2$

Modül mertebesindeki denklemler robotun genel şemasını göstermekte ancak girişte bahsedildiği üzere simülasyonlarda aşağıda açıklanan link mertebesindeki denklemler kullanılmıştır.

- Modül 0 için linkler arası ilişki:
  - Link 1 için:
    - $t_0 = p_0\, \dot{q_0}$
    - $\dot{t_0} = \dot{p_0}\, \dot{q_0} + p_0\, \ddot{q_0}$
    - ${w_0}^* = M_0\, \dot{t_0} + \Omega_0\, M_0\, E_0\, t_0$
- Modül 1 için linkler arası ilişki:
  - Link 1 için:
    - $t_{1^1} = A_{1^1,0}\, t_0 + p_{1^1}\, \dot{q_{1^1}}$
    - $\dot{t_{1^1}} = \dot{A}_{1^1,0}\, t_0 + A_{1^1,0}\, \dot{t}_0 + \dot{p_{1^1}}\, \dot{q_{1^1}} + p_{1^1}\, \ddot{q_{1^1}}$
    - ${w_{1^1}}^* = M_{1^1}\, \dot{t_{1^1}} + \Omega_{1^1}\, M_{1^1}\, E_{1^1}\, t_{1^1}$
  - Link 2 için:
    - $t_{2^1} = A_{2^1,1^1}\, t_{1^1} + p_{2^1}\, \dot{q_{2^1}}$
    - $\dot{t_{2^1}} = \dot{A}_{2^1,1^1}\, t_{1^1} + A_{2^1,1^1}\, \dot{t}_{1^1} + \dot{p_{2^1}}\, \dot{q_{2^1}} + p_{2^1}\, \ddot{q_{2^1}}$
    - ${w_{2^1}}^* = M_{2^1}\, \dot{t_{2^1}} + \Omega_{2^1}\, M_{2^1}\, E_{2^1}\, t_{2^1}$
  - Link 3 için:
    - $t_{3^1} = A_{3^1,2^1}\, t_{2^1} + p_{3^1}\, \dot{q_{3^1}}$
    - $\dot{t_{3^1}} = \dot{A}_{3^1,2^1}\, t_{2^1} + A_{3^1,2^1}\, \dot{t}_{2^1} + \dot{p_{3^1}}\, \dot{q_{3^1}} + p_{3^1}\, \ddot{q_{3^1}}$
    - ${w_{3^1}}^* = M_{3^1}\, \dot{t_{3^1}} + \Omega_{3^1}\, M_{3^1}\, E_{3^1}\, t_{3^1}$
- Modül 2 için linkler arası ilişki:
  - Link 1 için:
    - $t_{1^2} = A_{1^2,0}\, t_0 + p_{1^2}\, \dot{q_{1^2}}$
    - $\dot{t_{1^2}} = \dot{A}_{1^2,0}\, t_0 + A_{1^2,0}\, \dot{t}_0 + \dot{p_{1^2}}\, \dot{q_{1^2}} + p_{1^2}\, \ddot{q_{1^2}}$
    - ${w_{1^2}}^* = M_{1^2}\, \dot{t_{1^2}} + \Omega_{1^2}\, M_{1^2}\, E_{1^2}\, t_{1^2}$

- Link 2 için:
  - $t_{2^2} = A_{2^2,1^2}\, t_{1^2} + p_{2^2}\, \dot{q_{2^2}}$
  - $\dot{t_{2^2}} = \dot{A}_{2^2,1^2}\, t_{1^2} + A_{2^2,1^2}\, \dot{t}_{1^2} + \dot{p_{2^2}}\, \dot{q_{2^2}} + p_{2^2}\, \ddot{q_{2^2}}$
  - $w_{2^2}{}^* = M_{2^2}\, \dot{t_{2^2}} + \Omega_{2^2}\, M_{2^2}\, E_{2^2}\, t_{2^2}$
- Link 3 için:
  - $t_{3^2} = A_{3^2,2^2}\, t_{2^2} + p_{3^2}\, \dot{q_{3^2}}$
  - $\dot{t_{3^2}} = \dot{A}_{3^2,2^2}\, t_{2^2} + A_{3^2,2^2}\, \dot{t}_{2^2} + \dot{p_{3^2}}\, \dot{q_{3^2}} + p_{3^2}\, \ddot{q_{3^2}}$
  - $w_{3^2}{}^* = M_{3^2}\, \dot{t_{3^2}} + \Omega_{3^2}\, M_{3^2}\, E_{3^2}\, t_{3^2}$

"Wrench" değerlerinin link seviyesinde elde edilmesi ile en az işlem sayısına sahip olacak bir biçimde algoritmanın ileri özyineleme bölümünün oluşturulması sağlanmıştır.

Geri Özyineleme

Üst eklemlerin etkilerinin robot uç noktasından tabana doğru yayıldığı bu bölümde yine aynı şekilde ilk olarak modül seviyesindeki eşitlikler, hemen ardından da simülasyonda kullanılan link seviyesi aktarılmıştır.

- Modül 2 için modüller arası ilişki:
  - $\tilde{\bar{w}}_2 = \bar{w}_2{}^*$
  - $\bar{h}_2 = \bar{N}_2{}^T\, \tilde{\bar{w}}_2$
  - $\tilde{\bar{M}}_2 = \bar{M}_2$
  - $\bar{K}_2 = \tilde{\bar{M}}_2\, \bar{N}_2$
  - $\tilde{\bar{K}}_2 = \bar{A}_{2,0}{}^T\, \bar{K}_2$
- Modül 1 için modüller arası ilişki:
  - $\tilde{\bar{w}}_1 = \bar{w}_1{}^*$
  - $\bar{h}_1 = \bar{N}_1{}^T\, \tilde{\bar{w}}_1$
  - $\tilde{\bar{M}}_1 = \bar{M}_1$
  - $\bar{K}_1 = \tilde{\bar{M}}_1\, \bar{N}_1$
  - $\tilde{\bar{K}}_1 = \bar{A}_{1,0}{}^T\, \bar{K}_1$

- Modül 0 için modüller arası ilişki:
  - $\bar{\tilde{w}}_0 = \bar{w}_0{}^* + \bar{A}_{2,0}{}^T \bar{\tilde{w}}_1 + \bar{A}_{1,0}{}^T \bar{\tilde{w}}_1$
  - $\bar{h}_0 = \bar{N}_0{}^T \bar{\tilde{w}}_0$
  - $\bar{\tilde{M}}_0 = \bar{M}_0 + \bar{A}_{1,0}{}^T \bar{\tilde{M}}_1 \bar{A}_{1,0} + \bar{A}_{2,0}{}^T \bar{\tilde{M}}_2 \bar{A}_{2,0}$
  - $\bar{K}_0 = \bar{\tilde{M}}_0 P_0$
  - $\bar{I}_{0,0} = \bar{K}_0{}^T P_0$

Geri özyineleme ile eklemler arasındaki tork etkileri aktarımı modül mertebesinde incelenmiştir. Ayrıca serbest tabana ilişkin, bir sonraki ileri özyinelemede taban ivmesini hesaplamak için kullanılan "taban eylemsizlik matrisi" hesaplanmıştır. Simülasyonda kullanılan link mertebesi eşitlikleri şu şekilde verilebilir:

- Modül 2 için:
  - Link 3 için:
    - $\tilde{w}_{3^2} = w_{3^2}{}^*$
    - $h_{3^2} = p_{3^2}{}^T \tilde{w}_{3^2}$
    - $\tilde{M}_{3^2} = M_{3^2}$
    - $K_{3^2} = \tilde{M}_{3^2} p_{3^2}$
    - $\tilde{K}_{3^2} = A_{3^2,0}{}^T K_{3^2}$
  - Link 2 için:
    - $\tilde{w}_{2^2} = w_{2^2}{}^* + A^T_{3^2 2^2} \tilde{w}_{3^2}$
    - $h_{2^2} = p_{2^2}{}^T \tilde{w}_{2^2}$
    - $\tilde{M}_{2^2} = M_{2^2} + A^T_{3^2 2^2} \tilde{M}_{3^2} A_{3^2 2^2}$
    - $K_{2^2} = \tilde{M}_{2^2} p_{2^2}$
    - $\tilde{K}_{2^2} = A_{2^2,0}{}^T K_{2^2}$
  - Link 1 için:
    - $\tilde{w}_{1^2} = w_{1^2}{}^* + A^T_{2^2 1^2} \tilde{w}_{2^2}$
    - $h_{1^2} = p_{1^2}{}^T \tilde{w}_{1^2}$
    - $\tilde{M}_{1^2} = M_{1^2} + A^T_{2^2 1^2} \tilde{M}_{2^2} A_{2^2 1^2}$
    - $K_{1^2} = \tilde{M}_{1^2} p_{1^2}$
    - $\tilde{K}_{1^2} = A_{1^2,0}{}^T K_{1^2}$

- Modül 1 için:
  - Link 3 için:
    - $\widetilde{w}_{3^1} = {w_{3^1}}^*$
    - ${h_{3^1}} = {p_{3^1}}^T \widetilde{w}_{3^1}$
    - $\widetilde{M}_{3^1} = M_{3^1}$
    - $K_{3^1} = \widetilde{M}_{3^1} p_{3^1}$
    - $\widetilde{K}_{3^1} = {A_{3^1,0}}^T K_{3^1}$
  - Link 2 için:
    - $\widetilde{w}_{2^1} = {w_{2^1}}^* + A^T_{3^1 2^1} \widetilde{w}_{3^1}$
    - ${h_{2^1}} = {p_{2^1}}^T \widetilde{w}_{2^1}$
    - $\widetilde{M}_{2^1} = M_{2^1} + A^T_{3^1 2^1} \widetilde{M}_{3^1} A_{3^1 2^1}$
    - $K_{2^1} = \widetilde{M}_{2^1} p_{2^1}$
    - $\widetilde{K}_{2^1} = {A_{2^1,0}}^T K_{2^1}$
  - Link 1 için:
    - $\widetilde{w}_{1^1} = {w_{1^1}}^* + A^T_{2^1 1^1} \widetilde{w}_{2^1}$
    - ${h_{1^1}} = {p_{1^1}}^T \widetilde{w}_{1^1}$
    - $\widetilde{M}_{1^1} = M_{1^1} + A^T_{2^1 1^1} \widetilde{M}_{2^1} A_{2^1 1^1}$
    - $K_{1^1} = \widetilde{M}_{1^1} p_{1^1}$
    - $\widetilde{K}_{1^1} = {A_{1^1,0}}^T K_{1^1}$
- Modül 0 için:
  - Link 1 için:
    - $\widetilde{w}_0 = {w_0}^* + A^T_{1^1 0} \widetilde{w}_{1^1} + A^T_{1^2 0} \widetilde{w}_{1^2}$
    - ${h_0} = {p_0}^T \widetilde{w}_0$
    - $\widetilde{M}_0 = M_0 + A^T_{1^1 0} \widetilde{M}_{1^1} A_{1^1 0} + A^T_{1^2 0} \widetilde{M}_{1^2} A_{1^2 0}$
    - $K_0 = \widetilde{M}_0 P_0$
    - $I_{0,0} = {K_0}^T P_0$

Ayrıntılı bir şekilde yürüyen robota ilişkin ters dinamik geri özyineleme bölümü oluşturulduktan sonra son ileri özyineleme bölümüne geçilmiştir.

İleri Özyineleme

Son ileri özyineleme ile taban ivmeleri hesaplanmakta ve bu taban ivmelerin etkileri eklemlere tekrar aktarılmaktadır. Modüller arası incelendikten sonra link seviyesi açıklanmıştır.

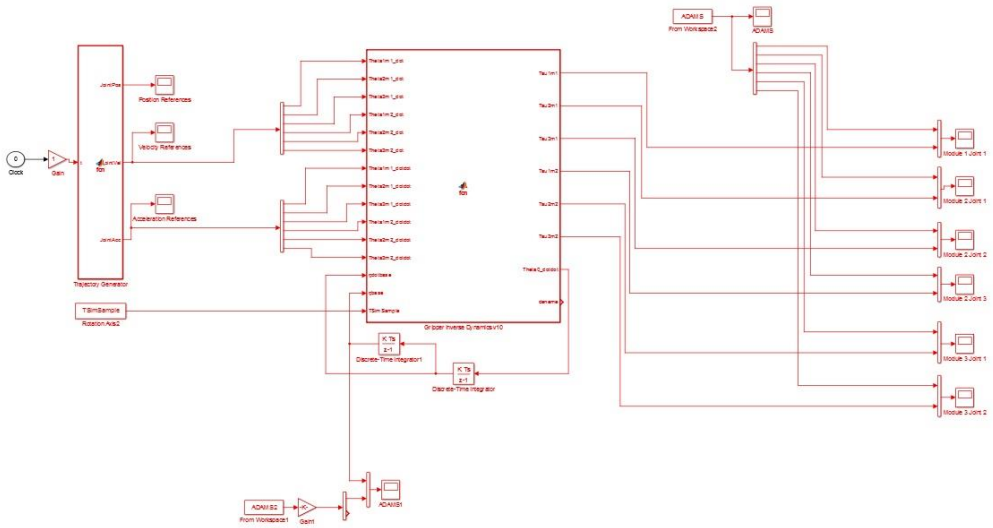
- Modül 0 için:
  - $\ddot{q}_0 = -I_{0,0}^{-1}\,\tau_0$
  - $\tilde{q}_0 = P_0\,\ddot{q}_0$
- Modül 1 için:
  - $\bar{\tau}_1 = \bar{\bar{K}}_1^{\,T}\,\tilde{q}_0 + \bar{h}_1$
- Modül 2 için:
  - $\bar{\tau}_2 = \bar{\bar{K}}_2^{\,T}\,\tilde{q}_0 + \bar{h}_2$

Serbest taban ters dinamik son ileri özyinelemesine ilişkin modül mertebesi eşitliklerinin ardından aşağıda link mertebesindeki taban ivme yayılımı aktarılmıştır.

- Modül 0 için:
  - Link 0 için:
    - $\ddot{q}_0 = -I_{0,0}^{-1}\,\tau_0$
    - $\tilde{q}_0 = P_0\,\ddot{q}_0$
- Modül 1 için:
  - Link 1 için:
    - $\tau_{1^1} = \widetilde{K}_{1^1}^{\,T}\,\tilde{q}_0 + h_{1^1}$
  - Link 2 için:
    - $\tau_{2^1} = \widetilde{K}_{2^1}^{\,T}\,\tilde{q}_0 + h_{2^1}$
  - Link 3 için:
    - $\tau_{3^1} = \widetilde{K}_{3^1}^{\,T}\,\tilde{q}_0 + h_{3^1}$
- Modül 2 için:
  - Link 1 için:
    - $\tau_{1^2} = \widetilde{K}_{1^2}^{\,T}\,\tilde{q}_0 + h_{1^2}$
  - Link 2 için:
    - $\tau_{2^2} = \widetilde{K}_{2^2}^{\,T}\,\tilde{q}_0 + h_{2^2}$
  - Link 3 için:
    - $\tau_{3^2} = \widetilde{K}_{3^2}^{\,T}\,\tilde{q}_0 + h_{3^2}$

Simülasyon

Yürüyen robota ilişkin elde edilen tüm link seviyesindeki eşitlikler MATLAB ortamında kullanılarak, ters dinamik algoritması oluşturulmuştur. Simulink diyagramının genel görünümü Şekil 3.9'da görülmektedir.



Şekil 3.9 : Düzlemsel yürüyen robot ters dinamik MATLAB algoritması

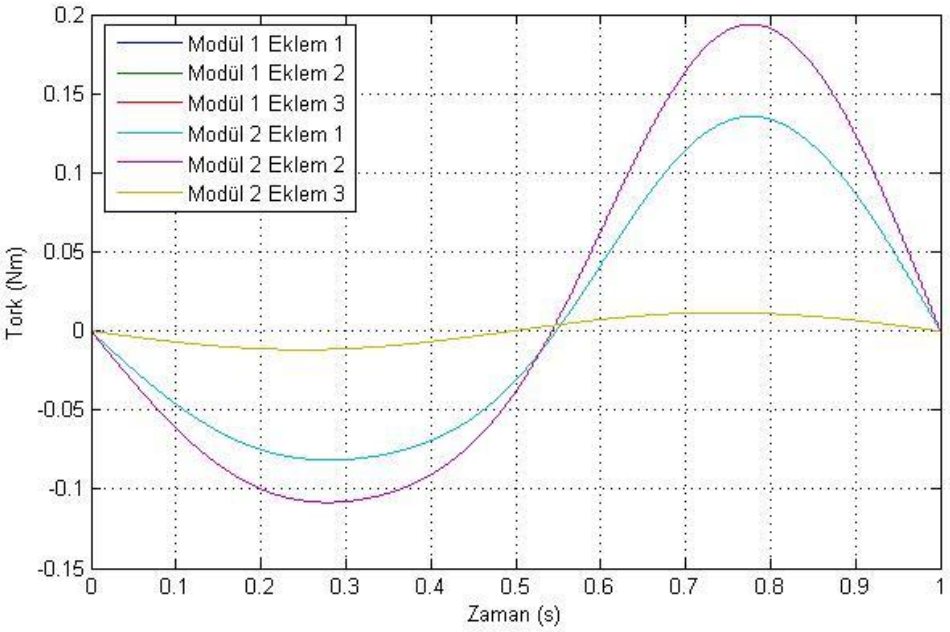
Diyagramda görülen gömülü fonksiyon bloğu tüm serbest tabanlı ters dinamik algoritmasını içermektedir. "m-file" sentaksında yazılan bu bloğa giriş olarak verilen eklem ivme yörüngeleri denklem 3.124'ün ikinci türevi ile oluşturulmuştur.

$$\theta_i = \frac{2\,\pi}{T_{Periyot}^2}\left(\theta_{final_i} - \theta_{başlangıç_i}\right)\sin\left(\frac{2\,\pi\,t}{T_{Periyot}}\right) \qquad (3.124)$$

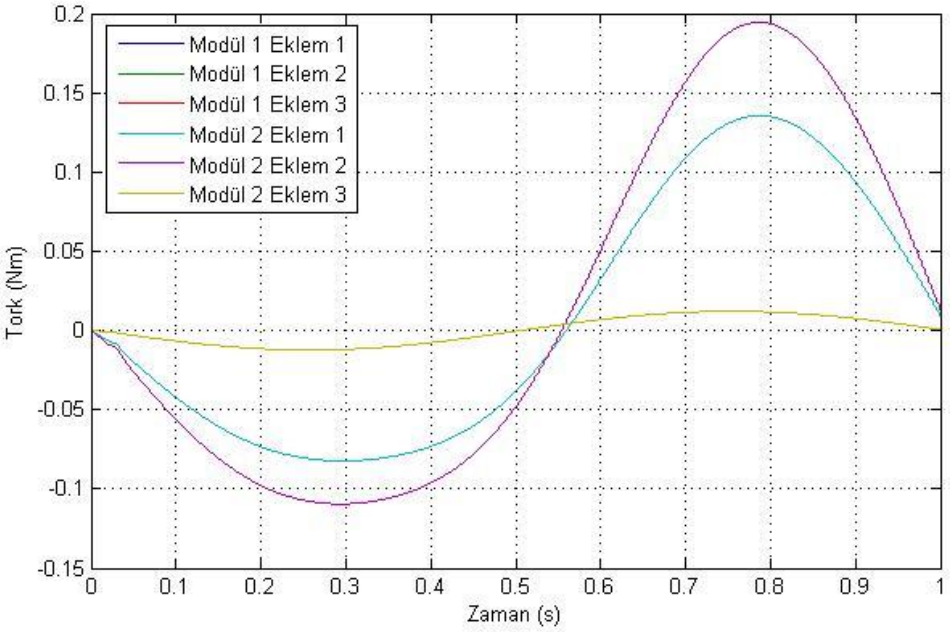
Tablo 3.4 : Eklem başlangıç ve final pozisyonları

	1^1	2^1	3^1	1^2	2^2	3^2
$\theta_{başlangıç_i}$	0	0	0	0	0	0
θ_{final_i}	-60	-60	-60	-60	-60	-60

Tüm eklemlere aynı açı farkları verilmiştir. Bu referanslar hem MATLAB algoritmasında hem de ADAMS ortamında uygulanarak, tork ölçümleri karşılaştırılmıştır.



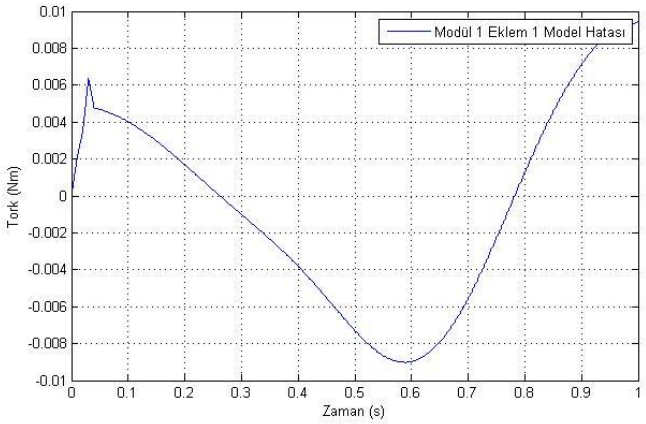
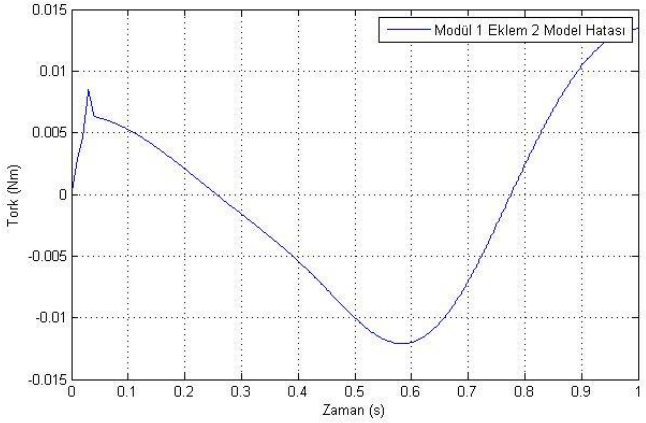
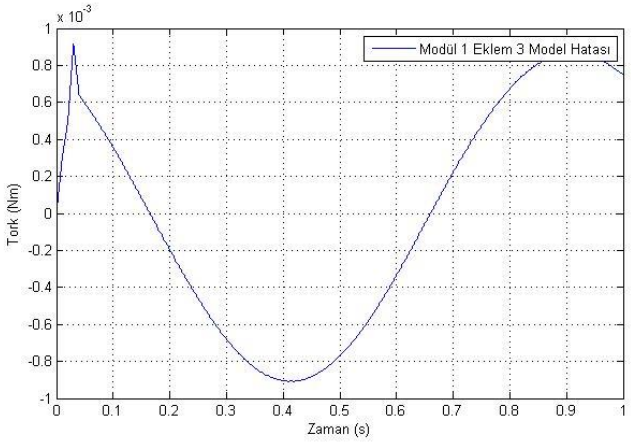
Şekil 3.10 : Yürüyen robot ters dinamik MATLAB algoritması tork çıktıları

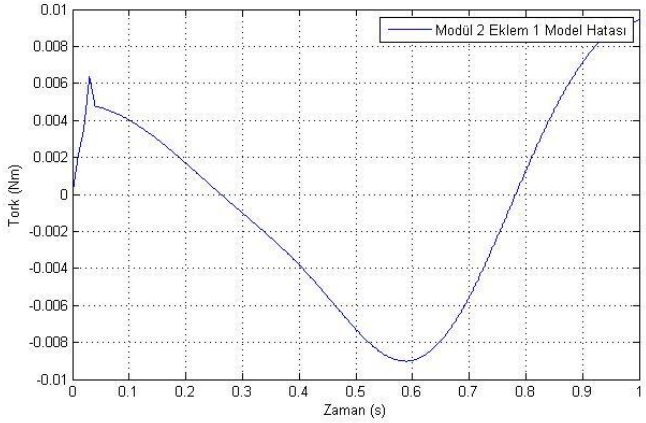
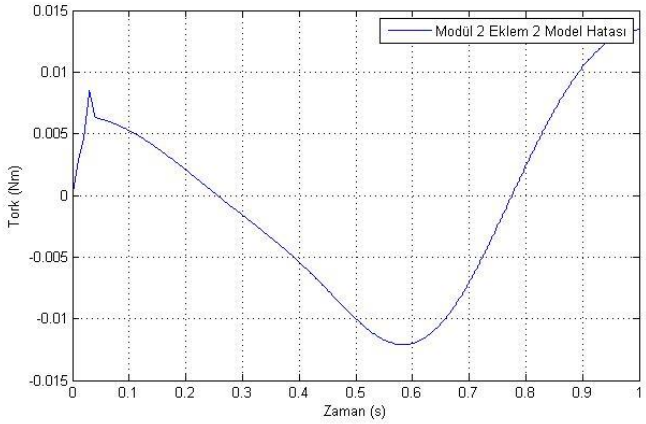
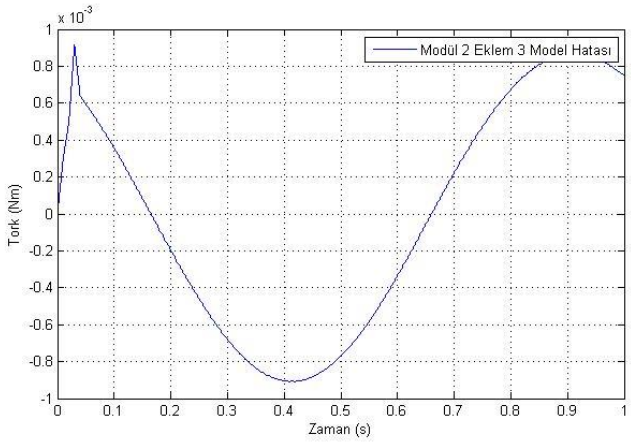


Şekil 3.11 : Yürüyen robot ters dinamik ADAMS tork çıktıları

Şekil 3.10 ve Şekil 3.11'de görüldüğü üzere ADAMS ve MATLAB algoritması tork çıktıları form olarak benzerdir. Bunu daha detaylı görebilmek için her ekleme ait ayrı grafiklerde ADAMS ile MATLAB algoritma sonuçları arasındaki farklar çizdirilmiştir.

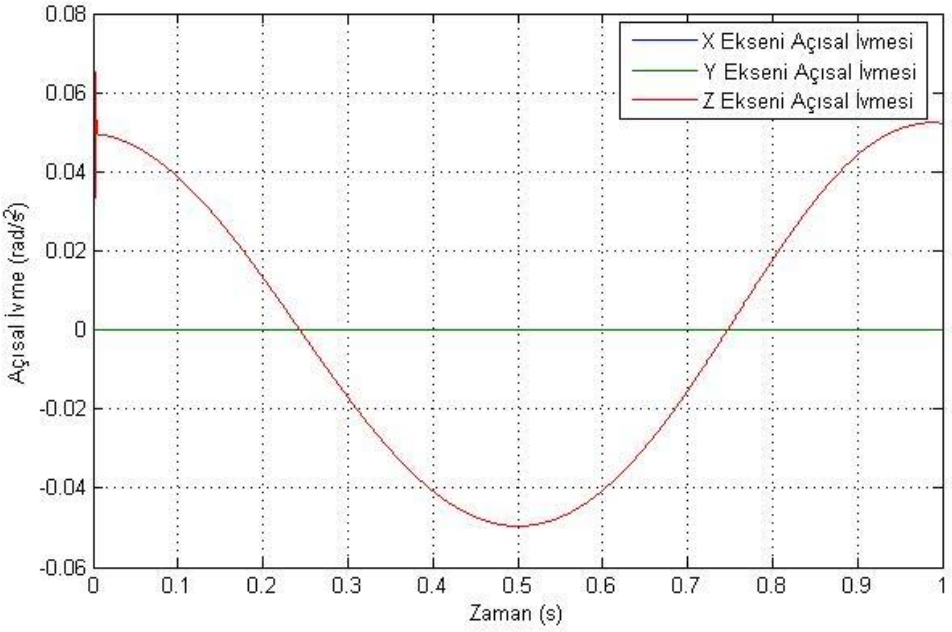
Tablo 3.5 : Yürüyen robot ters dinamik eklem tork karşılaştırma tablosu

Eklem No	Eklem Tork Hesaplama Hatası Grafiği
1^1	
2^1	
3^1	

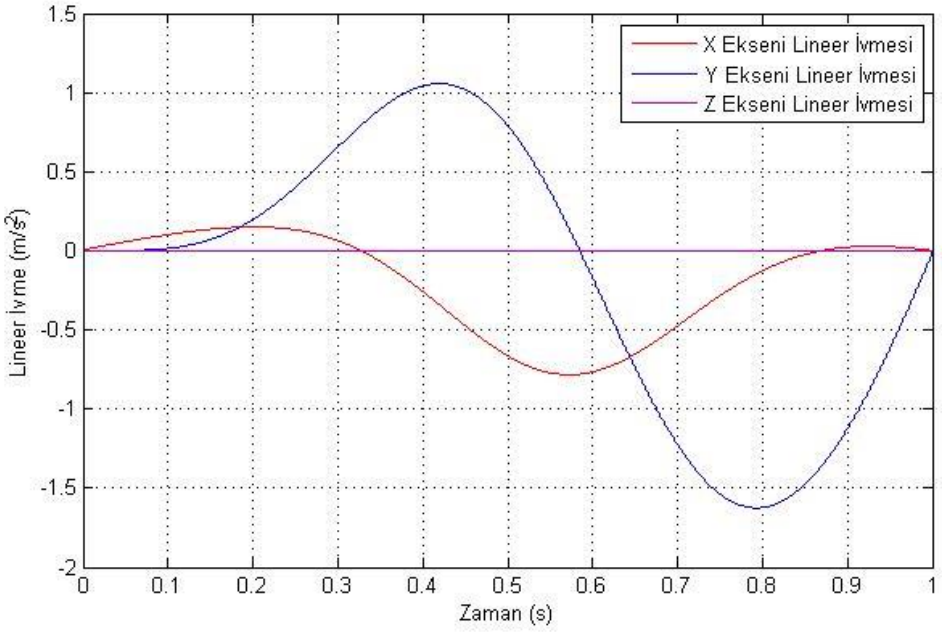
1^2	
2^2	
3^2	

Son tabloda aktarılan modelleme hatalarına ilişkin grafikler incelendiğinde ADAMS ile MATLAB algoritması arasındaki en yüksek farkın her iki modülün "2" nolu eklemlerinde meydana geldiği ve değerin 0.015 Nm civarında olduğu görülmektedir. Tüm eklemlere aynı anda hareket verilmesi durumu göz önüne alındığında bu fark

makul değerler aralığında kalmaktadır. Eklem tork karşılaştırmalarının ardından serbest taban ivmelerinin karşılaştırmalarına ilişkin grafikler Şekil 3.12 ve Şekil 3.13 ile verilmiştir.



Şekil 3.12 : Yürüyen robot serbest taban açısal ivme hesaplama hatası



Şekil 3.13 : Yürüyen robot serbest taban lineer ivme hesaplama hatası

Serbest taban ivmelerinin hatalarına bakıldığında açısal ivmelerdeki hataların düşük kaldığı ancak, lineer ivme hesaplama hatasının görece daha fazla olduğu

görülmektedir. Eklemlerde meydana gelen hatanın taban ivmelerindeki sapmalardan kaynaklı olduğu söylenebilir. Yine de eklem torklarındaki hataların makul çıkması algoritmanın doğru çalıştığını söylemek için yeterlidir denebilir.

3.6.2.2. İleri dinamik

Sabit olmayan tabanlı örnek olan yürüyen robot probleminin ileri dinamik çözümü, bölüm 3'te anlatıldığı biçimde özyinelemeli olarak çözülmüştür. Serbest taban ters dinamiğinde olduğu gibi burada da üç aşama bulunmaktadır. Sırasıyla ileri, geri ve son ileri özyineleme aşamalarını içeren ileri dinamik kısmı, teorik olarak modül ve link seviyesinde açıklanmıştır. Aynı şekilde burada da link seviyesindeki denklemler ile simülasyon şeması oluşturulmuştur.

Serbest taban bulunduğu için verilen eklem tork referanslarına göre ileri dinamik hem eklem ivmelerini hem de serbest taban ivmelerini hesaplamaktadır. İlk ileri özyineleme, ters dinamik ilk ileri özyineleme bölümü ile eklem ivmelerinin "0" alınması dışında aynıdır. Bu nedenle direk geri özyineleme bölümü denklemlerinin verilmesi ile ileri dinamik açıklanması başlatılmıştır.

İleri dinamik algoritması MATLAB ortamında m-file sentaksı ile oluşturulmuştur. Bu eşitlikler ilerleyen bölümlerde sırayla açıklanmıştır.

İleri Özyineleme

Giriş bölümünde kısaca bahsedildiği gibi, eklem ivmelerinin "0" alınması dışında bu bölüm serbest taban ters dinamik ilk ileri özyineleme bölümü eşitliklerini içerir.

Geri Özyineleme

Girdi torklarına göre eklemlerin birbirine olan etkilerini de dahil edecek biçimde, eklemlerde oluşan gerçek torkları hesaplar. Ancak serbest taban etkisini içermez. Modül mertebesindeki eşitlikler şu şekilde verilebilir:

- Modül 2 için:
  - $\hat{\bar{\tau}}_2 = \bar{\tau}_2$
  - $\tilde{\bar{\tau}}_2 = \hat{\bar{I}}_2^{-1}\, \hat{\bar{\tau}}_2$
  - $\bar{\eta}_2 = \bar{\Psi}_2\, \hat{\bar{\tau}}_2$

- Modül 1 için:
  - $\hat{\bar{\tau}}_1 = \bar{\tau}_1$
  - $\tilde{\bar{\tau}}_1 = \hat{\bar{I}}_1^{-1} \hat{\bar{\tau}}_1$
  - $\bar{\eta}_1 = \bar{\Psi}_1 \hat{\bar{\tau}}_1$
- Modül 0 için:
  - $\tilde{\eta}_0 = \bar{A}_{1,0}^T \bar{\eta}_1 + \bar{A}_{2,0}^T \bar{\eta}_2$
  - $\hat{\tau}_0 = -P_0^T (\tilde{\eta}_0 + w_o^*)$
  - $\tilde{\tau}_0 = \hat{I}_0^{-1} \hat{\tau}_0$

Simülasyonda kullanılan ve daha az işleme sahip olan link seviyesi denklemleri ise şu şekilde aktarılabilir:

- Modül 2 için:
  - Link 3 için:
    - $\hat{\tau}_{3^2} = \tau_{3^2}$
    - $\tilde{\tau}_{3^2} = \hat{I}_{3^2}^{-1} \hat{\tau}_{3^2}$
    - $\eta_{3^2} = \Psi_{3^2} \hat{\tau}_{3^2}$
  - Link 2 için:
    - $\tilde{\eta}_{2^2} = A_{3^2,2^2}^T \eta_{3^2}$
    - $\hat{\tau}_{2^2} = \tau_{2^2} - p_{2^2}^T (\tilde{\eta}_{2^2} + w_{2^2}^*)$
    - $\tilde{\tau}_{2^2} = \hat{I}_{2^2}^{-1} \hat{\tau}_{2^2}$
    - $\eta_{2^2} = \Psi_{2^2} \hat{\tau}_{2^2}$
  - Link 1 için:
    - $\tilde{\eta}_{1^2} = A_{2^2,1^2}^T \eta_{2^2}$
    - $\hat{\tau}_{1^2} = \tau_{1^2} - p_{1^2}^T (\tilde{\eta}_{1^2} + w_{1^2}^*)$
    - $\tilde{\tau}_{1^2} = \hat{I}_{1^2}^{-1} \hat{\tau}_{1^2}$
    - $\eta_{1^2} = \Psi_{1^2} \hat{\tau}_{1^2}$
- Modül 1 için:
  - Link 3 için:
    - $\hat{\tau}_{3^1} = \tau_{3^1}$
    - $\tilde{\tau}_{3^1} = \hat{I}_{3^1}^{-1} \hat{\tau}_{3^1}$
    - $\eta_{3^1} = \Psi_{3^1} \hat{\tau}_{3^1}$

- Link 2 için:
  - $\tilde{\eta}_{2^1} = A^T_{3^1,2^1}\,\eta_{3^1}$
  - $\hat{\tau}_{2^1} = \tau_{2^1} - p^T_{2^1}\,(\tilde{\eta}_{2^1} + {w_{2^1}}^*)$
  - $\tilde{\tau}_{2^1} = {\hat{I}_{2^1}}^{-1}\,\hat{\tau}_{2^1}$
  - $\eta_{2^1} = \Psi_{2^1}\,\hat{\tau}_{2^1}$
- Link 1 için:
  - $\tilde{\eta}_{1^1} = A^T_{2^1,1^1}\,\eta_{2^1}$
  - $\hat{\tau}_{1^1} = \tau_{1^1} - p^T_{1^1}\,(\tilde{\eta}_{1^1} + {w_{1^1}}^*)$
  - $\tilde{\tau}_{1^1} = {\hat{I}_{1^1}}^{-1}\,\hat{\tau}_{1^1}$
  - $\eta_{1^1} = \Psi_{1^1}\,\hat{\tau}_{1^1}$

- Modül 0 için:
  - Link 1 için:
    - $\tilde{\eta}_0 = A^T_{1^1,0}\,\eta_{1^1} + A^T_{1^2,0}\,\eta_{1^2}$
    - $\hat{\tau}_0 = -p^T_0\,(\tilde{\eta}_0 + {w_0}^*)$
    - $\tilde{\tau}_0 = {\hat{I}_0}^{-1}\,\hat{\tau}_0$

Daha detaylı bir biçimde link seviyesinde aktarılan, serbest taban örneği için oluşturulmuş ileri dinamik geri özyineleme bölümünde serbest tabanda eklemlere uygulanan torklardan dolayı oluşan tork hesaplanmaktadır. Ayrıca eklemlerin birbirine olan etkileri de yine burada hesaplanmıştır.

İleri Özyineleme

Serbest taban etkisinin eklem ivmelerine eklendiği bu aşamada ilk olarak serbest taban ivmesi hesaplanmaktadır. Modül seviyesindeki denklemler şöyledir:

- Modül 0 için:
  - $\ddot{q}_0 = \tilde{\tau}_0$
  - $\mu_0 = p_0\,\ddot{q}_0$
- Modül 1 için:
  - $\tilde{\bar{\mu}}_1 = \bar{A}_{1,0}\,\mu_0$
  - $\ddot{\tilde{q}}_1 = \tilde{\bar{\tau}}_1 - \bar{\Psi}^T_1\,\tilde{\bar{\mu}}_1$
- Modül 2 için:
  - $\tilde{\bar{\mu}}_2 = \bar{A}_{2,0}\,\mu_0$
  - $\ddot{\tilde{q}}_2 = \tilde{\bar{\tau}}_2 - \bar{\Psi}^T_2\,\tilde{\bar{\mu}}_2$

Modül seviyesinin ardından link seviyesindeki denklemler şu şekilde aktarılabilir:

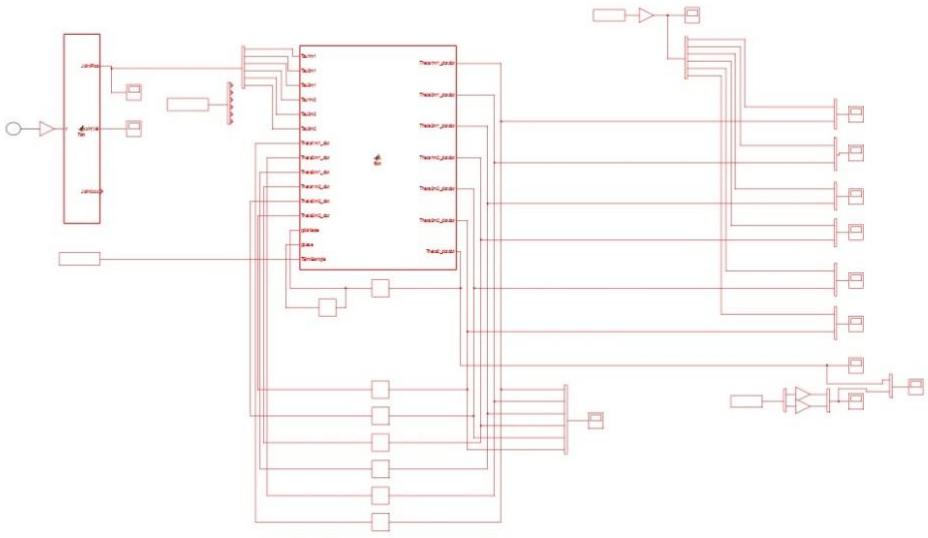
- Modül 0 için:
  - $\ddot{q}_0 = \tilde{\tau}_0$
  - $\mu_0 = p_0\, \ddot{q}_0$
- Modül 1 için:
  - Link 1 için:
    - $\tilde{\mu}_{1^1} = A_{1^1,0}\, \mu_0$
    - $\ddot{q}_{1^1} = \tilde{\tau}_{1^1} - \Psi_{1^1}^T\, \tilde{\mu}_{1^1}$
    - $\mu_{1^1} = p_{1^1}\, \ddot{q}_{1^1} + \tilde{\mu}_{1^1}$
  - Link 2 için:
    - $\tilde{\mu}_{2^1} = A_{2^1,1^1}\, \mu_{1^1}$
    - $\ddot{q}_{2^1} = \tilde{\tau}_{2^1} - \Psi_{2^1}^T\, \tilde{\mu}_{2^1}$
    - $\mu_{2^1} = p_{2^1}\, \ddot{q}_{2^1} + \tilde{\mu}_{2^1}$
  - Link 3 için:
    - $\tilde{\mu}_{3^1} = A_{3^1,2^1}\, \mu_{2^1}$
    - $\ddot{q}_{3^1} = \tilde{\tau}_{3^1} - \Psi_{3^1}^T\, \tilde{\mu}_{3^1}$
    - $\mu_{3^1} = p_{3^1}\, \ddot{q}_{3^1} + \tilde{\mu}_{3^1}$
- Modül 2 için:
  - Link 1 için:
    - $\tilde{\mu}_{1^2} = A_{1^2,0}\, \mu_0$
    - $\ddot{q}_{1^2} = \tilde{\tau}_{1^2} - \Psi_{1^2}^T\, \tilde{\mu}_{1^2}$
    - $\mu_{1^2} = p_{1^2}\, \ddot{q}_{1^2} + \tilde{\mu}_{1^2}$
  - Link 2 için:
    - $\tilde{\mu}_{2^2} = A_{2^2,1^2}\, \mu_{1^2}$
    - $\ddot{q}_{2^2} = \tilde{\tau}_{2^2} - \Psi_{2^2}^T\, \tilde{\mu}_{2^2}$
    - $\mu_{2^2} = p_{2^2}\, \ddot{q}_{2^2} + \tilde{\mu}_{2^2}$
  - Link 3 için:
    - $\tilde{\mu}_{3^2} = A_{3^2,2^2}\, \mu_{2^2}$
    - $\ddot{q}_{3^2} = \tilde{\tau}_{3^2} - \Psi_{3^2}^T\, \tilde{\mu}_{3^2}$
    - $\mu_{3^2} = p_{3^2}\, \ddot{q}_{3^2} + \tilde{\mu}_{3^2}$

Son ileri özyineleme ile serbest taban ivmeleri hesaplanmış ardından bu ivmelerin eklemler üzerindeki etkileri sırayla tabandan modül uçlarına doğru yayılmıştır.

Düzlemsel yürüyen robota ait hem modül seviyesinde hem de link seviyesinde ileri dinamik özyineleme şeması açıklanmıştır. Gerçekleştirilen simülasyonlarda link seviyesindeki eşitlikler kullanılmıştır.

Simülasyon

Yürüyen düzlemsel robota ilişkin olarak oluşturulmuş ileri dinamik algoritmasının MATLAB Simulink diyagramı genel şeması Şekil 3.14'de gösterilmiştir. Diyagramda bulunan gömülü fonksiyon algoritmanın tamamını m-file sentaksı şeklinde içermektedir.



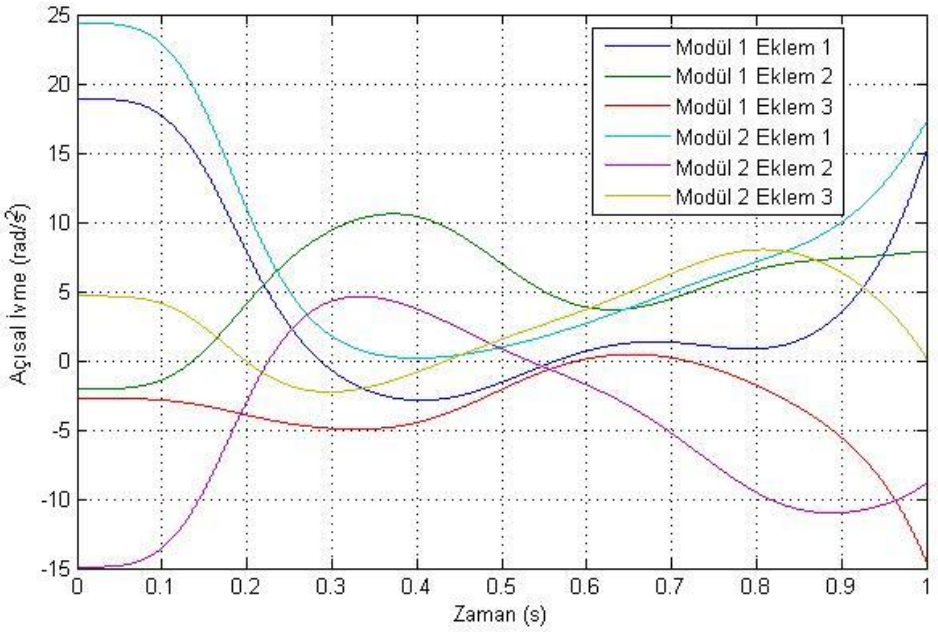
Şekil 3.14 : Yürüyen robot ileri dinamik MATLAB algoritması

Simülasyonlarda uygulanan eklem torklarına ilişkin veriler, Tablo 3.6'da gösterilmiştir.

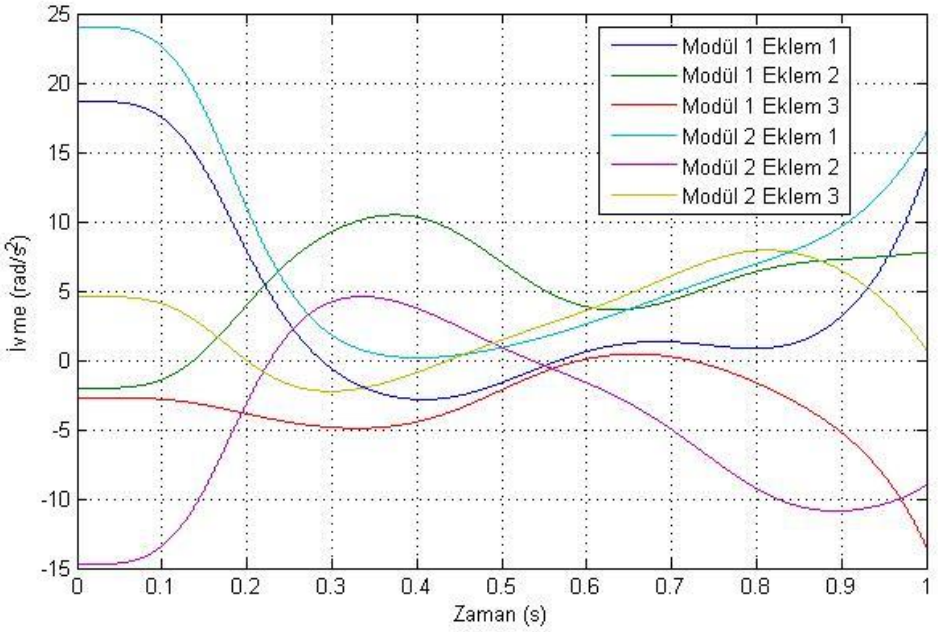
Tablo 3.6 : Eklem tork girdileri

	1^1	2^1	3^1	1^2	2^2	3^2
$\tau\ (Nm)$	0.082	0.082	0	0.082	0	0

Rastgele seçilen üç ekleme uygulanan sabit tork girdileri uygulanarak simülasyon gerçekleştirilmiştir. Takip eden grafiklerde (Şekil 3.15 ve Şekil 3.16) ADAMS ve MATLAB algoritmalarının çıktıları aktarılmıştır.



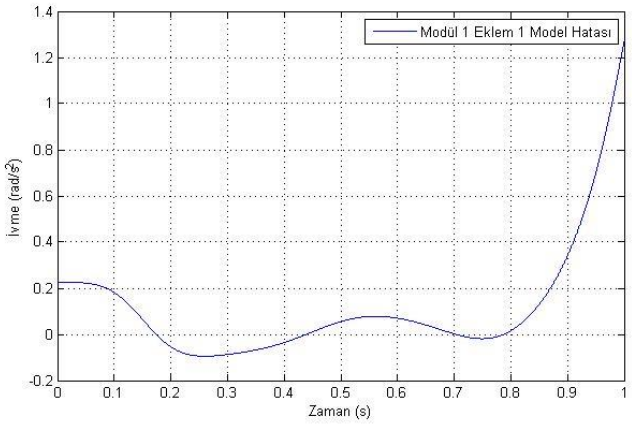
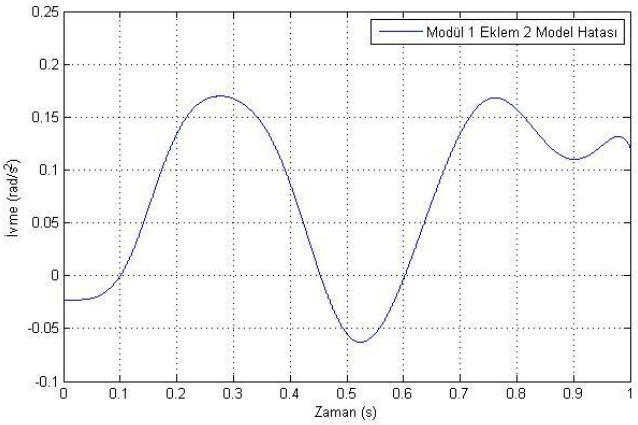
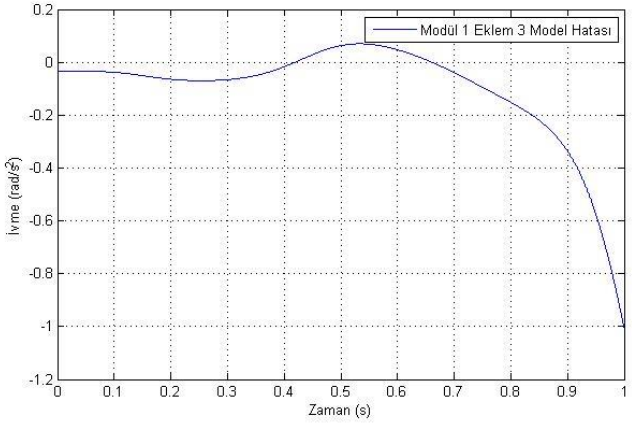
Şekil 3.15 : ADAMS ileri dinamik çıktısı

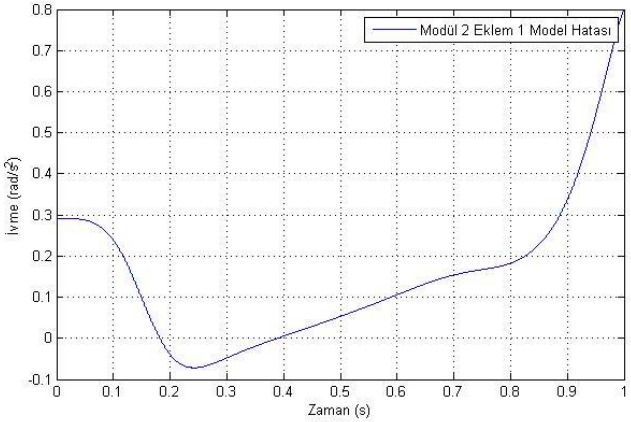
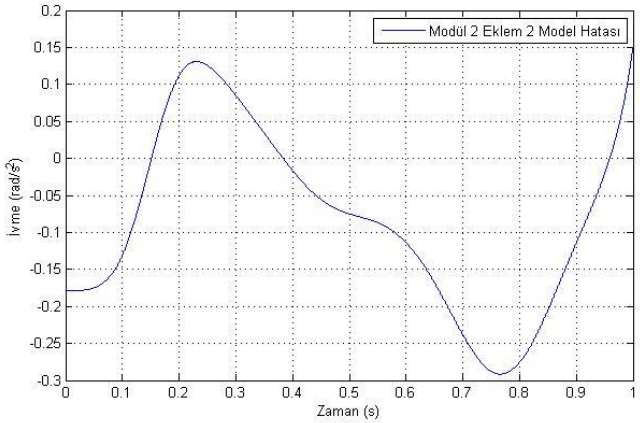
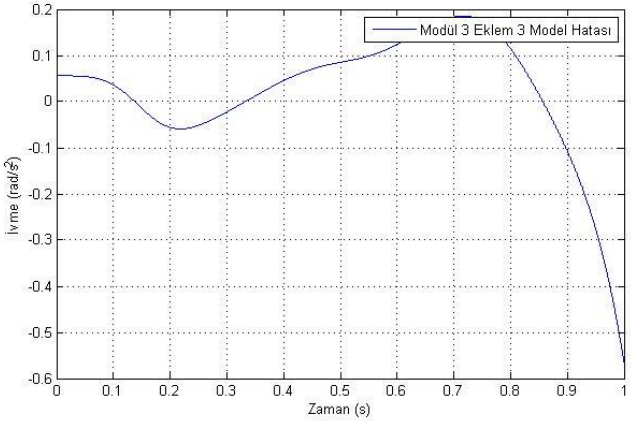


Şekil 3.16 : MATLAB algoritması ileri dinamik çıktısı

MATLAB algoritmasının doğruluğunu daha net bir şekilde görmek adına Tablo 3.7 oluşturulmuştur. Tablo 3.7'de eklem bazında ADAMS ile MATLAB algoritma arasındaki varyasyon grafik olarak verilmiştir.

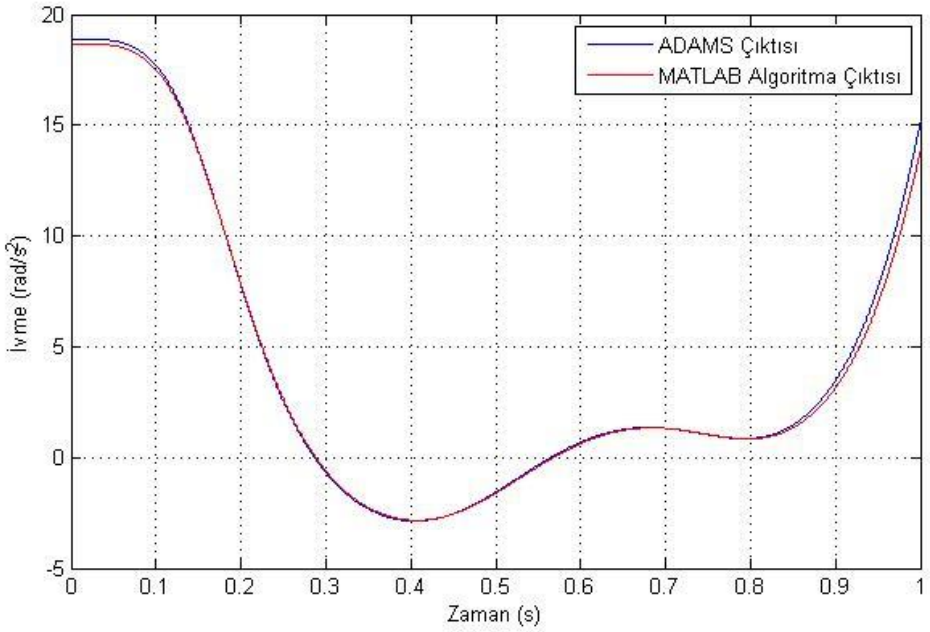
Tablo 3.7 : Yürüyen robot ileri dinamik eklem ivme karşılaştırma tablosu

Eklem No	Eklem İvme Hesaplama Hatası Grafiği
1^1	
2^1	
3^1	

1^2	
2^2	
3^2	

Tablo 3.7 incelendiğinde modelleme hatasının maksimum değeri 1 nolu modülün ilk linkinde ortaya çıkmıştır. Yaklaşık, anlık olarak $1{,}3\,rad/s^2$ ivme hatası oldukça yüksek görülse de uygulanan tork girdisi yüksek olduğundan eklem ivmesi de yüksek

çıkmıştır. Şekil 3.17'de 1 nolu modül 1 nolu ekleme ilişkin ADAMS ve MATLAB algoritma çıktıları üst üste çizdirilerek daha iyi anlaşılması amaçlanmıştır.



Şekil 3.17 : Modül 1 eklem 1 ivme çıktıları

Hem form olarak hem de değerlerin yakınlığı göz önünde bulundurulduğunda algoritmanın doğru çalıştığı söylenebilir.

4. ÜÇ BOYUTLU YÜRÜYEN ROBOT

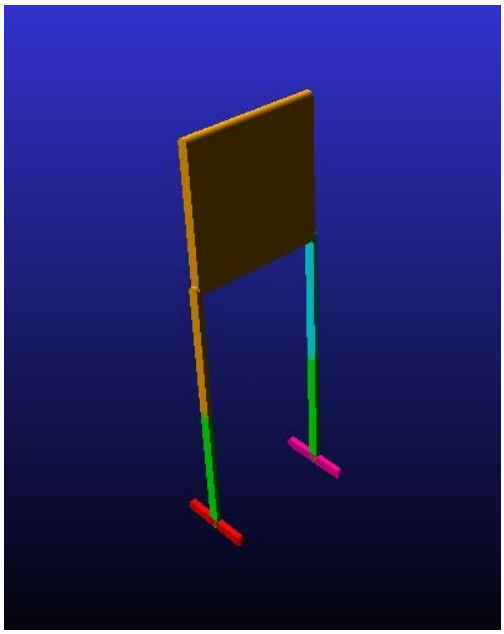
Bu bölüme kadar, karmaşık yapılı robotlara ilişkin kinematik ve dinamik analizler aktarılmaya çalışılmıştır. Basit örneklerle desteklenen çözümlerin, algoritma olarak pratiğe dökülmesi ve doğruluklarının testleri sağlanmıştır. İlk olarak sadece ileri kinematiği barındıran, kinematik bölümünde modül ve link seviyesinde özyinelemeli yapı akratılmış ve hemen ardından sabit ve sabit olmayan olmak üzere iki adet düzlemsel örnekle simülasyonlar gerçekleştirilmiştir.

Dinamik modelleme bölümü, karmaşık yapılı robotlar için ileri ve ters dinamik modellerini içermektedir. Teorik olarak aktarılan modül ve link seviyesindeki denklemler, kinematik bölümünde olduğu gibi basit düzlemsel yapılarla simüle edilmiştir. Kinematik bölümde kullanılan sabit ve sabit olmayan tabanlı robotik yapıların ters ve ileri dinamik modelleri oluşturulmuştur.

Düzlemsel sabit olmayan tabanlı örnekte, hareketli taban ve ayak yörüngeleri oluşturulmuş ancak detaylarından bahsedilmemişti. Bu bölümde analizleri gerçekleştirilen 3 boyutlu yürüyen robota ait, simüle edilen özyinelemeli yapıya ilişkin modül ve link seviyesinde denklemler, simülasyon sonuçları, simülasyonun doğruluk testi, gövde ve ayak yörüngelerinin oluşturulması ve birden fazla serbestlik derecesine sahip eklemlerin kinematik olarak tanımlanması başlıkları açıklanmıştır.

3 boyutlu yürüyen robota ilişkin dinamik modelleme bölümü, ters dinamik ve ileri dinamik modül ve link seviyesi eşitliklerinin verilmesi, link seviyesindeki denklemlerle simülasyonun gerçekleştirilmesi, simülasyonun farklı bir dinamik analiz programı ile karşılaştırılması aşamalarını barındırmaktadır.

Bu bölümde analizleri gerçekleştirilen robot yapısı Şekil 4.1'de görülmektedir.



Şekil 4.1 : 3 boyutlu yürüyen robot

Yürüyen robot, insan eklemleri baz alınarak oluşturulmuştur. Buna göre gövde ile bacaklar arasındaki eklemler 3 serbestlik dereceli küresel eklemlerden, diz eklemi tek serbestlik dereceli rotasyonel eklemden ve bilek eklemi iki serbestlik dereceli universal eklemden oluşturulmuştur. Toplamda 12 serbestlik derecesi barındırmaktadır.

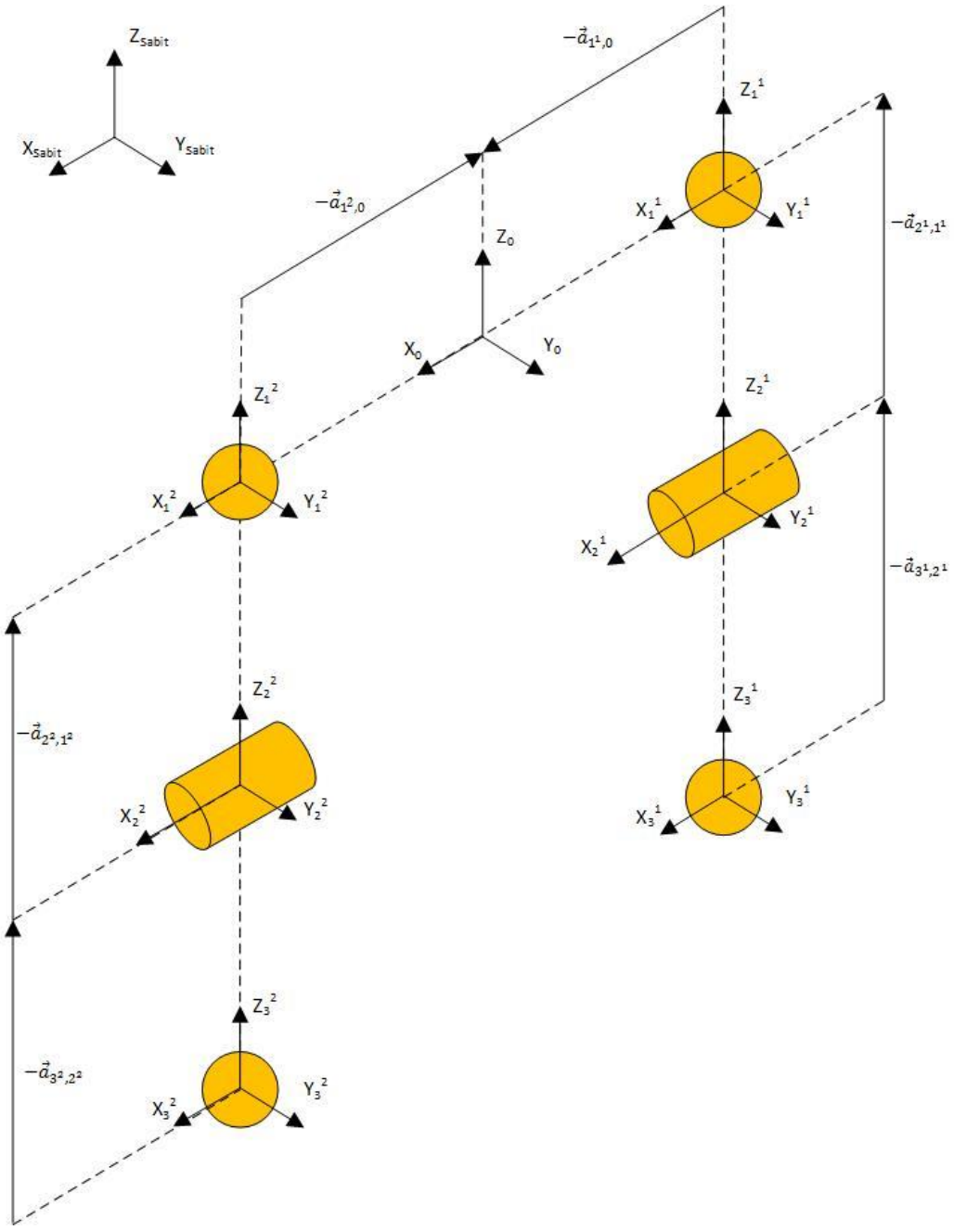
4.1. Kinematik Modelleme

3 boyutlu yürüyen robot, düzlemsel olarak ele alınan yürüyen robotun yapısıyla modül ve link tanımlaması olarak birebir aynıdır. Tabanı hareketli olarak kabul edilen ve 3 modülden oluşan bir yapıya sahiptir. Taban "0" nolu modül olarak isimlendirilmekte ve geriye kalan modüllerde robotun bacaklarını oluşturmaktadır. Her iki modülde üçer adet link içermektedir. Düzlemsel yürüyen robottan, bu örneği ayıran kısım eklem tanımlamalarıdır. Düzlemsel örnekte, tüm eklemler bir serbestlik derecesine sahipken, 3 boyutlu örnekte iki adet küresel, iki adet üniversal ve iki adet tek serbestlik dereceli rotasyonel eklem bulunmaktadır.

Küresel ve üniversal eklem barındırması, bu eklemlere ilişkin kinematik yapıların anlatılmasını gerektirmektedir. İlk olarak eklemlere ilişkin koordinat sistemlerinin ataması ve robotun başlangıç konfigürasyonunu belirleyen vektörlerin belirlenmesi açıklanmıştır. Ardından simülasyonda kullanılan koordinat eksenlerinin

güncellenmesine ilişkin bölümde kullanılacak olan küresel ve universal eklemlerin kinematik analizleri açıklanmıştır.

Robotun eksen atamaları ve linklere bağlı olarak oluşturulan ve kinematik analizde kullanılan vektörler Şekil 4.2'de görülmektedir.



Şekil 4.2 : 3 boyutlu yürüyen robot eksen atama şeması

Görüldüğü üzere eksen atamaları, sabit olarak tanımlı koordinat eksenlerine paralel olarak seçilmiştir. Eklemlerin dönme eksenleri bu koordinat eksenlerine göre

tanımlanmaktadır. Buna göre eklemlerin dönme eksenleri Tablo 4.1'deki gibi belirlenmiştir.

Tablo 4.1 : Dönme eksenleri

Eklem Numarası	Dönme Ekseni
1^1	$\boldsymbol{X_{1^1}}$, $\boldsymbol{Y_{1^1}}$, $\boldsymbol{Z_{1^1}}$
2^1	$\boldsymbol{X_{2^1}}$
3^1	$\boldsymbol{X_{3^1}}$, $\boldsymbol{Y_{3^1}}$
1^2	$\boldsymbol{X_{1^2}}$, $\boldsymbol{Y_{1^2}}$, $\boldsymbol{Z_{1^2}}$
2^2	$\boldsymbol{X_{2^2}}$
3^2	$\boldsymbol{X_{3^2}}$, $\boldsymbol{Y_{3^2}}$

Dönme eksenlerine ek olarak Twist yayılım matrislerinin bağlı olduğu vektörler de belirlenmiş ve yukarıdaki şekilde gösterilmiştir. Bu vektörler robotun yapısını belirlediğinden kinematik çözümde önemli yer kaplamaktadır. Vektörlerin başlangıç değerleri Tablo 4.2'de aktarılmıştır.

Tablo 4.2 : Vektör tanımlamaları

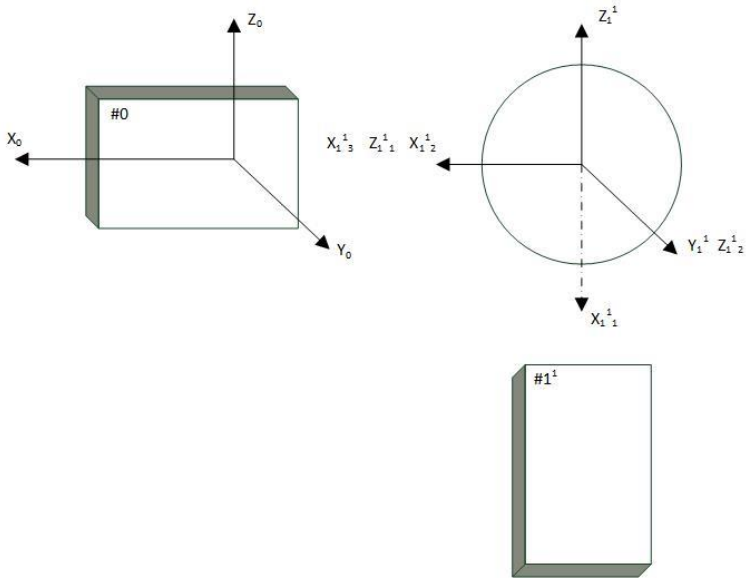
Link Numarası	Değeri
$-\boldsymbol{a_{1^1,0}}$	$+0{,}25 * \boldsymbol{X_0} + 0 * \boldsymbol{Y_0} + 0 * \boldsymbol{Z_0}$
$-\boldsymbol{a_{2^1,1^1}}$	$0 * \boldsymbol{X_{1^1}} + 0 * \boldsymbol{Y_{1^1}} + 0{,}5 * \boldsymbol{Z_{1^1}}$
$-\boldsymbol{a_{3^1,2^1}}$	$0 * \boldsymbol{X_{2^1}} + 0 * \boldsymbol{Y_{2^1}} + 0{,}5 * \boldsymbol{Z_{2^1}}$
$-\boldsymbol{a_{1^2,0}}$	$-0{,}25 * \boldsymbol{X_0} + 0 * \boldsymbol{Y_0} + 0 * \boldsymbol{Z_0}$
$-\boldsymbol{a_{2^2,1^2}}$	$0 * \boldsymbol{X_{1^2}} + 0 * \boldsymbol{Y_{1^2}} + 0{,}5 * \boldsymbol{Z_{1^2}}$
$-\boldsymbol{a_{3^2,2^2}}$	$0 * \boldsymbol{X_{2^2}} + 0 * \boldsymbol{Y_{2^2}} + 0{,}5 * \boldsymbol{Z_{2^2}}$

Bu vektörler ve eklemlere atanan koordinat eksenleri robotun eklem hareketlerine göre güncellenmektedir. Güncelleme işlemi temelde eksen takımlarının, eklem hızlarına bağlı olarak güncellenmesinden ibarettir. Tüm vektörler bu eksen takımlarına bağlı olarak oluşturulduğu için, otomatik olarak güncellenmektedir. Buna

ek olarak, 3 boyutlu yürüyen robotta üç serbestlik derecesine kadar eklemler bulunduğu için, bu eklemlere ilişkin rotasyon matrislerinin oluşturulması belirli bir temele bağlı kalarak oluşturulmuştur.

4.1.1. Euler açı eklemleri

İki ve daha fazla serbestlik derecesine sahip eklemlerin rotasyonunu belirtmek için bu eklemler birden fazla tek serbestlik dereceli eklem ile tanımlanmakta ve bu tanıma göre bir DH tablosu oluşturularak rotasyon hesaplanmaktadır. Diğer yandan bu eklemlere Euler açılarıyla yaklaşmak da mümkündür. Euler açıları, uzaydaki bir cismin yönelimini üç farklı eksende sıralı dönme olarak tanımlamaktadır. Ancak Euler açıları her eksende tanımlanabilir. Bu nedenle DH tablosu gibi sistematik yaklaşımlara uzak kalmaktadır[9]. DH tablosu oluşturulurken sabit rotasyonlar X ekseninde, değişken rotasyonlar Z ekseninde yapılmaktadır. Bu nedenle Euler açıları ile DH tablosu arasında bir bağlantı kurulması gerekmektedir. Bu bağlantı Saha'nın öne sürdüğü "Euler Eklem Açıları" konsepti ile açıklanmaktadır. Örnek olarak, Şekil 4.3'de görülen küresel eklemin rotasyon matrisi, XYZ "Euler Eklem Açıları" yöntemi ile oluşturulan DH tablosu ile açıklansın:



Şekil 4.3 : 3 serbestlik dereceli eklemin koordinat ekseni atamaları

Burada amaç, 3 serbestlik dereceli eklemin hareketine bağlı olarak, 1^1 nolu link ile 0 nolu link arasındaki rotasyon matrisini DH yöntemi kullanarak açıklamaktır. DH yöntemini bu eklem üzerinde kullanabilmek için, tüm serbestlik derecelerine ilişkin koordinat eksenlerinin tanımlanması gereklidir. Şekil 4.3'da görülen küresel eklemin

serbestlik derecelerine ilişkin koordinat eksenlerinin ataması, önceden belirlenen Euler açı sıralamasına göre yapılmıştır. Burada XYZ Euler açıları kullanıldığından, ilk serbestlik derecesine ilişkin "$Z_{1_1^1}$" ekseni X ekseni ile, ikinci serbestlik derecesine ilişkin "$Z_{1_2^1}$" ekseni Y ekseni ile ve son olarak üçüncü serbestlik derecesine ilişkin "$Z_{1_3^1}$" ekseni ise Z ekseni ile kesişecek şekilde belirlendi. X eksenleri, DH yöntemine göre atandı. Bu sonuca göre sadece rotasyon bölümünü ilgilendiren değişkenlere göre Tablo 4.3'de verilen DH parametreleri ortaya çıkmaktadır.

Tablo 4.3 : Küresel eklem XYZ Euler açıları için DH tablosu

Link No	α	θ
0	0	90
1_1^1	90	$\theta_1^* - 90$
1_2^1	-90	$\theta_2^* - 90$
1_3^1	90	θ_3^*

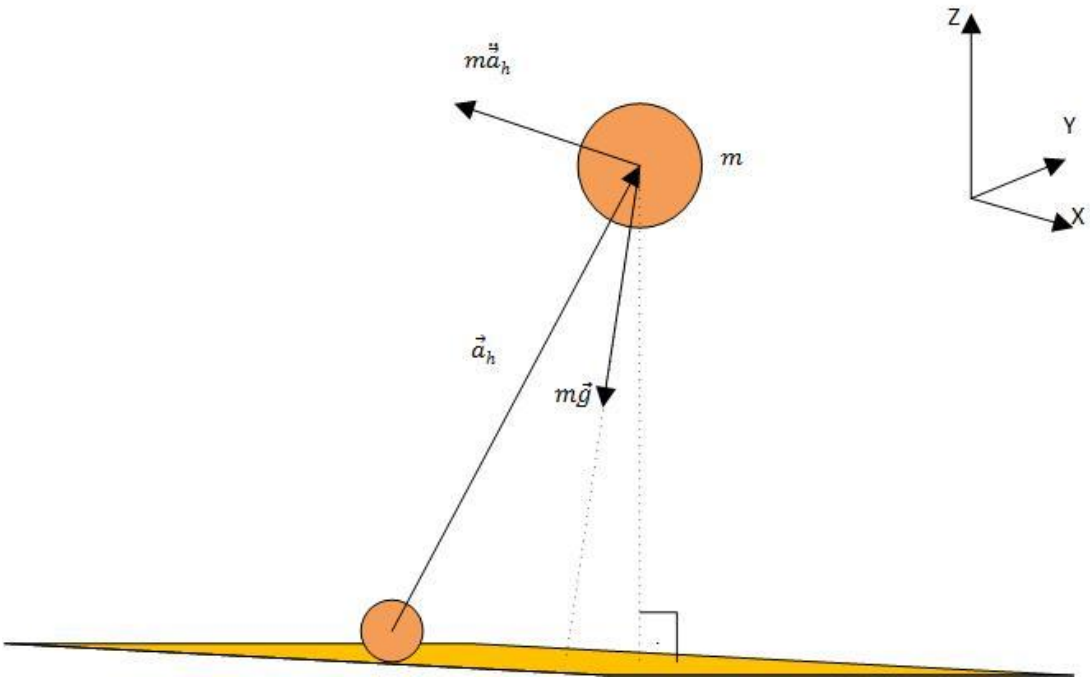
Tablo 4.3'de link olarak belirtilen serbestlik derecesine ilişkin linkler, aslında bu yöntemde sanal link olarak ele alınmaktadır. Bu tabloya göre iki gerçek link arasındaki rotasyon matrisi açıklanabilir.

$$Q_{XYZ} = Q_{+Z} Q_{1_1^1} Q_{1_2^1} Q_{1_3^1} \tag{4.1}$$

Elde edilen denklem 3.125'deki rotasyon matrisi önceden de bahsedildiği gibi küresel eklemin hareketi nedeniyle etkilenen vektörleri ve matrislerin hesaplanması için, eksen güncelleme kısmında kullanılmaktadır.

4.1.2. Gövde ve ayak yörüngeleri

Sabit olmayan yapıda olan 3 boyutlu yürüyen robota ilişkin olarak, insansı bir adım atmak için gövde ve ayak yörüngelerinin uygun bir biçimde belirlenmesi gereklidir. Kajita ve Toni'nin öne sürdüğü "Ters Sarkaç Modeli" tabanlı yörünge planlaması bu çalışmada seçilmiştir. Bu modele göre tüm kütle yürüyen robotun gövdesinde toplanmış olarak kabul edilmekte ve bu noktanın yürüme esnasında yere temas eden ayakta oluşturduğu moment üzerinden yörüngeler oluşturulmaktadır.



Şekil 4.4 : Ters sarkaç modeli

Temas eden ayak ile gövde arasını işaret eden ve sabit koordinat ekseninde (robottan bağımsız) tanımlı olan "$\boldsymbol{a}_h$" vektörü, robotun eylemsizlik kuvveti ve yerçekimi kuvveti kullanılarak robotu bu konumdaki dengesini sağlayan eşitlik denklem 4.2 ile verilmiştir.

$$\boldsymbol{a}_h \; x \; (m \; \ddot{\boldsymbol{a}}_h) - \; \boldsymbol{a}_h \; x \; (m \; \boldsymbol{g}) = 0 \tag{4.2}$$

Yerçekimi ve eylemsizlik kuvvetlerinin, temas eden ayakta meydana getirdiği momenti sıfırlayan "$\boldsymbol{a}_h$" vektörünün hesaplanması için bu vektörlerin daha detaylı incelenmesi gerekmektedir.

$$\begin{gathered} y_h \; \ddot{z}_h - z_h \; \ddot{y}_h + g \; y_h = 0 \\ z_h \; \ddot{x}_h - x_h \; \ddot{z}_h - g \; z_h = 0 \\ x_h \; \ddot{y}_h - y_h \; \ddot{x}_h = 0 \end{gathered} \tag{4.3}$$

Elde edilen denklem takımı 4.3'deki üç denklem, "z_h" yüksekliğinin sabit istenmesi ve düz adım atma için "y_h" yörüngesinin sıfır istenmesi durumları göz önüne alındığında denklem 4.4'deki eşitliğe ulaşılmaktadır.

$$\ddot{x}_h - \frac{g}{z_h} \; x_h = 0 \tag{4.4}$$

Elde edilen "X" yörüngesine yönelik diferansiyel denklem çözüldüğünde denklem 4.5 ile belirtilen çözüme ulaşılır.

$$x_h(t) = c_1\, e^{wt} + c_2\, e^{-wt} \tag{4.5}$$

Sonucuna varılmaktadır. Bu denklemin katsayıları başlangıç koşullarına göre denklem 4.6'daki şekilde elde edilir:

$$\begin{aligned} c_1 &= \frac{1}{2}\left(x_h(0) + \frac{1}{w}\,\dot{x}_h(0)\right), \quad w = \sqrt{\frac{g}{z_h}} \\ c_2 &= \frac{1}{2}\left(x_h(0) - \frac{1}{w}\,\dot{x}_h(0)\right), \quad w = \sqrt{\frac{g}{z_h}} \end{aligned} \tag{4.6}$$

Böylece gövde yörüngesi bulunmuştur. Periyodik adımlama için başlangıç koşulları uygun bir biçimde belirlenmelidir.

$$\begin{aligned} x_h(0) &= -x_h(T_{Final}) \\ \dot{x}_h(0) &= \frac{1 - e^{-wt}}{1 - e^{wt}}\, w\, x_h(0) \end{aligned} \tag{4.7}$$

Gövde yörüngesinin ardından, ayak yörüngesine sıra gelmektedir. İki adet ayaktan bir tanesi adım atma boyunca hareketsiz kaldığından sadece hareketli ayağa ilişkin yörünge oluşturulmaktadır. Tekrarlı adımlama durumunda ise, yörüngenin uygulandığı ayak değiştirilmektedir. Saha ayak yörüngelerini denklem 4.8'deki şekilde tanımlamaktadır.

$$\begin{aligned} x_a(t) &= -l_s \cos\left(\frac{\pi}{T}\, t\right) \\ z_a(t) &= \frac{h_f}{2}\left(1 - \cos\left(\frac{\pi}{T}\, t\right)\right) \end{aligned} \tag{4.8}$$

Ayağın hareketi de yine düz adımlama için y ekseninde 0 pozisyon değişimi olarak ortaya çıkmaktadır. X ve Z yörüngelerinde bulunan parametreler, "l_s" adım uzunluğu, "T" adım periyotu, "h_f" adım yüksekliği ve "t" geçen zaman olarak hesaba katılmaktadır.

4.1.3. Kinematik simülasyon

Robotun kinematik denklemleri modül seviyesinde düzlemsel örnek ile birebir aynıdır. Ancak birden fazla serbestlik derecesi içerdiği için, 3 boyutlu örnekte link seviyesindeki denklemler farklılaşmaktadır. Hatta serbestlik derecesi seviyesine inilmesi gerekmektedir.

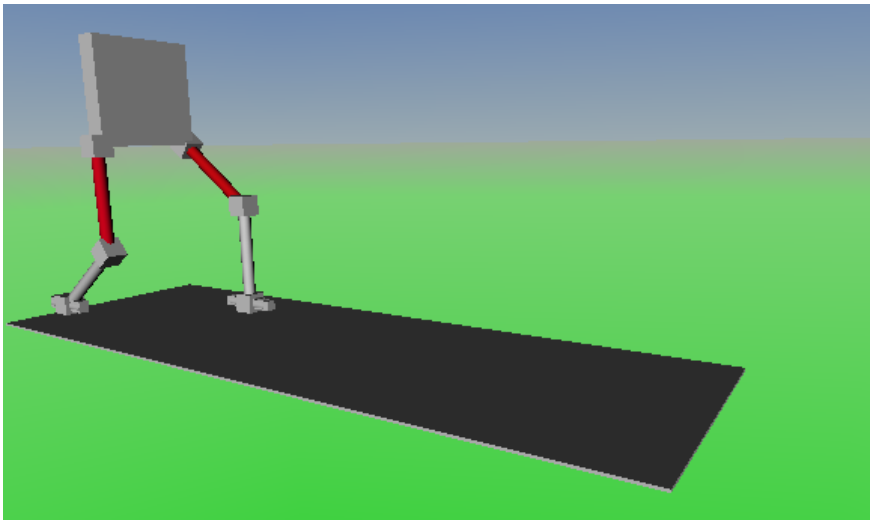
Bu durumda kinematik modeli veren denklemler serbestlik derecesi seviyesinde şu şekilde verilebilir.

- Modül 0 için:
  - $t_0 = P_0\, \dot{q_0}$
- Modül 1 için:
  - Link 1 için:
    - $t_{1_1^1} = A_{1^1\,0}\, t_0 + p_{1_1^1}\, \dot{\theta}_{1_1^1}$
    - $t_{1_2^1} = t_{1_1^1} + p_{1_2^1}\, \dot{\theta}_{1_2^1}$
    - $t_{1_3^1} = t_{1_2^1} + p_{1_3^1}\, \dot{\theta}_{1_3^1}$
  - Link 2 için:
    - $t_{2^1} = A_{2^1 1^1}\, t_{1_3^1} + p_{2^1}\, \dot{\theta}_{2^1}$
  - Link 3 için:
    - $t_{3_1^1} = A_{3^1\,2^1}\, t_{2^1} + p_{3_1^1}\, \dot{\theta}_{3_1^1}$
    - $t_{3_2^1} = t_{3_1^1} + p_{3_2^1}\, \dot{\theta}_{3_2^1}$
- Modül 2 için:
  - Link 1 için:
    - $t_{1_1^2} = A_{1^2\,0}\, t_0 + p_{1_1^2}\, \dot{\theta}_{1_1^2}$
    - $t_{1_2^2} = t_{1_1^2} + p_{1_2^2}\, \dot{\theta}_{1_2^2}$
    - $t_{1_3^2} = t_{1_2^2} + p_{1_3^2}\, \dot{\theta}_{1_3^2}$
  - Link 2 için:
    - $t_{2^2} = A_{2^2 1^2}\, t_{1_3^2} + p_{2^2}\, \dot{\theta}_{2^2}$
  - Link 3 için:
    - $t_{3_1^2} = A_{3^2\,2^2}\, t_{2^2} + p_{3_1^2}\, \dot{\theta}_{3_1^2}$
    - $t_{3_2^2} = t_{3_1^2} + p_{3_2^2}\, \dot{\theta}_{3_2^2}$

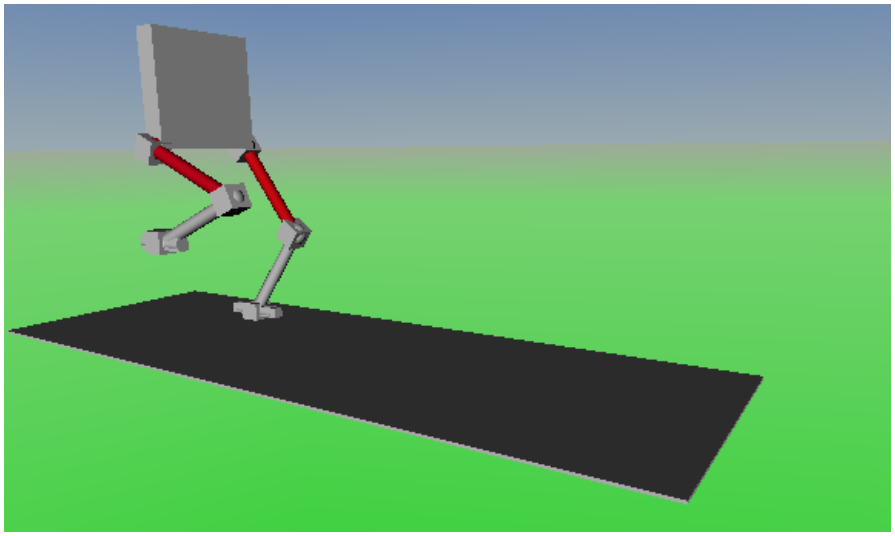
Aşağıda belirtilen adımlama değerlerine göre simülasyon gerçekleştirilmiştir.

- Maksimum Ayak Yüksekliği: 0.4 metre
- Adım Uzunluğu: 0.8 metre

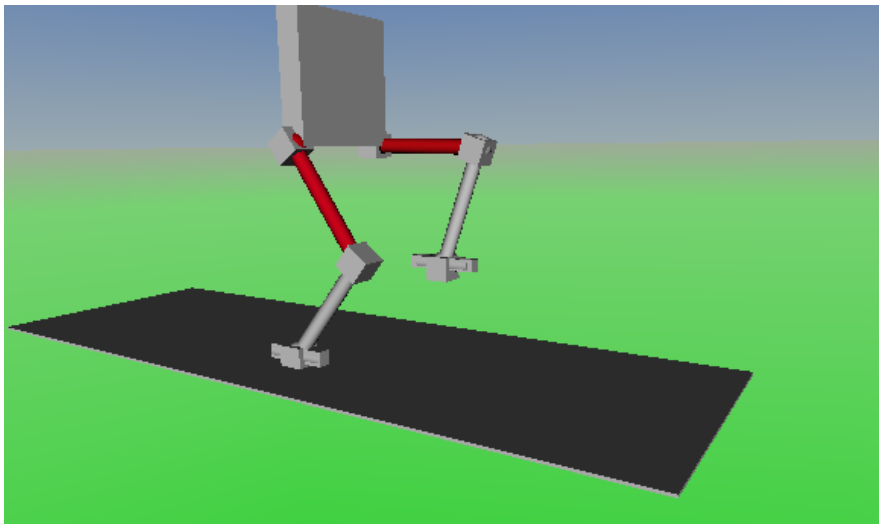
Kinematik algoritması çıktısı kullanılarak görsel arayüzde serbest halde bulunan küp şeklindeki cisimler hareket ettirilmiş ve eklemleri takip edip etmediği gözlenmiştir. Simülasyonun farklı anlarına ilişkin ekran görüntüleri Şekil 4.5, Şekil 4.6 ve Şekil 4.7 ile aktarılmıştır.



Şekil 4.5 : 3 boyutlu İki ayaklı yürüyen robot kinematik simülasyon görüntü 1



Şekil 4.6 : 3 boyutlu İki ayaklı yürüyen robot kinematik simülasyon görüntü 2



Şekil 4.7 : 3 boyutlu İki ayaklı yürüyen robot kinematik simülasyon görüntü 3

Şekil 4.5, Şekil 4.6 ve Şekil 4.7 ile verilen kinematik simülasyon görüntüleri ile küp şeklindeki cisimlerin hem lineer pozisyon hem de oryantasyon olarak ilgili oldukları robot eklemlerini takip ettikleri görülmektedir. Fakat simülasyon uzun süre çalıştırıldığında bazı eklemlerin ıraksadığı görülmektedir.

4.2. Ters Dinamik Modelleme

Düzlemsel yürüyen robota benzer biçimde burada da ters dinamik modelleme 3 aşamadan meydana gelmektedir. 3 boyutlu yürüyen robotta kinematik modelleme de olduğu gibi dinamik modelleme bölümünde de serbestlik derecesi seviyesine inilmesi gerekmektedir. Sırayla ileri özyineleme, geri özyineleme ve ikinci ileri özyineleme aşamaları işletilerek hesaplanan ters dinamik modele ilişkin sadece serbestlik derecesi seviyesindeki (simülasyonda kullanılan) denklemler aktarılmıştır.

İleri Özyineleme

"Twist", "Twist" türevi ve "Wrench" vektörlerinin hesaplanmasını içermektedir. Modül mertebesi düzlemsel iki ayaklı yürüyen robot ile aynı olduğu için burada tekrar edilmemiştir.

- Modül 0 için:
  - $t_0 = p_0\, \dot{q_0}$
  - $\dot{t_0} = \dot{p_0}\, \dot{q_0} + p_0\, \ddot{q_0}$
  - ${w_0}^* = M_0\, \dot{t_0} + \Omega_0\, M_0\, E_0\, t_0$

- Modül 1 için:
  - Link 1 için:
    - Serbestlik Derecesi 1 için:
      - $t_{1_1^1} = A_{1^1,0}\, t_0 + p_{1_1^1}\, \dot{q_{1_1^1}}$
      - $\dot{t_{1_1^1}} = \dot{A}_{1^1,0}\, t_0 + A_{1^1,0}\, \dot{t}_0 + \dot{p_{1_1^1}}\, \dot{q_{1_1^1}} + p_{1_1^1}\, \ddot{q_{1_1^1}}$
      - $w_{1_1^1}{}^* = [0\ 0\ 0\ 0\ 0\ 0]^T$
    - Serbestlik Derecesi 2 için:
      - $t_{1_2^1} = t_{1_1^1} + p_{1_2^1}\, \dot{q_{1_2^1}}$
      - $\dot{t_{1_2^1}} = \dot{t_{1_1^1}} + \dot{p_{1_2^1}}\, \dot{q_{1_2^1}} + p_{1_2^1}\, \ddot{q_{1_2^1}}$
      - $w_{1_2^1}{}^* = [0\ 0\ 0\ 0\ 0\ 0]^T$
    - Serbestlik Derecesi 3 için:
      - $t_{1_3^1} = t_{1_2^1} + p_{1_3^1}\, \dot{q_{1_3^1}}$
      - $\dot{t_{1_3^1}} = \dot{t_{1_2^1}} + \dot{p_{1_3^1}}\, \dot{q_{1_3^1}} + p_{1_3^1}\, \ddot{q_{1_3^1}}$
      - $w_{1_3^1}{}^* = M_{1^1}\, \dot{t_{1_3^1}} + \Omega_{1_3^1}\, M_{1^1}\, E_{1^1}\, t_{1_3^1}$
  - Link 2 için:
    - $t_{2^1} = A_{2^1,1^1} t_{1_3^1} + p_{2^1}\, \dot{q_{2^1}}$
    - $\dot{t_{2^1}} = \dot{A}_{2^1,1^1} t_{1_3^1} + A_{2^1,1^1} \dot{t_{1_3^1}} + \dot{p_{2^1}}\, \dot{q_{2^1}} + p_{2^1}\, \ddot{q_{2^1}}$
    - $w_{2^1}{}^* = M_{2^1}\, \dot{t_{2^1}} + \Omega_{2^1}\, M_{2^1}\, E_{2^1}\, t_{2^1}$
  - Link 3 için:
    - Serbestlik Derecesi 1 için:
      - $t_{3_1^1} = A_{3^1,2^1}\, t_{2^1} + p_{3_1^1}\, \dot{q_{3_1^1}}$
      - $\dot{t_{3_1^1}} = \dot{A}_{3^1,2^1}\, t_{2^1} + A_{3^1,2^1}\, \dot{t}_{2^1} + \dot{p_{3_1^1}}\, \dot{q_{3_1^1}} + p_{3_1^1}\, \ddot{q_{3_1^1}}$
      - $w_{3_1^1}{}^* = [0\ 0\ 0\ 0\ 0\ 0]^T$

- Serbestlik Derecesi 2 için:
  - $t_{3_2^1} = t_{3_1^1} + p_{3_2^1}\, q_{3_2^1}^{\cdot}$
  - $t_{3_2^1}^{\cdot} = t_{3_1^1}^{\cdot} + p_{3_2^1}^{\cdot}\, q_{3_2^1}^{\cdot} + p_{3_2^1}\, q_{3_2^1}^{\cdot\cdot}$
  - $w_{3_2^1}{}^{*} = M_{3^1}\, t_{3_2^1}^{\cdot} + \Omega_{3_2^1}\, M_{3^1}\, E_{3^1}\, t_{3_2^1}$

- Modül 2 için:
  - Link 1 için:
    - Serbestlik Derecesi 1 için:
      - $t_{1_1^2} = A_{1^2,0}\, t_0 + p_{1_1^2}\, q_{1_1^2}^{\cdot}$
      - $t_{1_1^2}^{\cdot} = \dot{A}_{1^2,0}\, t_0 + A_{1^2,0}\, \dot{t}_0 + p_{1_1^2}^{\cdot}\, q_{1_1^2}^{\cdot} + p_{1_1^2}\, q_{1_1^2}^{\cdot\cdot}$
      - $w_{1_1^2}{}^{*} = [0\ 0\ 0\ 0\ 0\ 0]^T$
    - Serbestlik Derecesi 2 için:
      - $t_{1_2^2} = t_{1_1^2} + p_{1_2^2}\, q_{1_2^2}^{\cdot}$
      - $t_{1_2^2}^{\cdot} = t_{1_1^2}^{\cdot} + p_{1_2^2}^{\cdot}\, q_{1_2^2}^{\cdot} + p_{1_2^2}\, q_{1_2^2}^{\cdot\cdot}$
      - $w_{1_2^2}{}^{*} = [0\ 0\ 0\ 0\ 0\ 0]^T$
    - Serbestlik Derecesi 3 için:
      - $t_{1_3^2} = t_{1_2^2} + p_{1_3^2}\, q_{1_3^2}^{\cdot}$
      - $t_{1_3^2}^{\cdot} = t_{1_2^2}^{\cdot} + p_{1_3^2}^{\cdot}\, q_{1_3^2}^{\cdot} + p_{1_3^2}\, q_{1_3^2}^{\cdot\cdot}$
      - $w_{1_3^2}{}^{*} = M_{1^2}\, t_{1_3^2}^{\cdot} + \Omega_{1_3^2}\, M_{1^2}\, E_{1^2}\, t_{1_3^2}$
  - Link 2 için:
    - $t_{2^2} = A_{2^2,1^2} t_{1_3^2} + p_{2^2}\, q_{2^2}^{\cdot}$
    - $t_{2^2}^{\cdot} = \dot{A}_{2^2,1^2} t_{1_3^2} + A_{2^2,1^2} t_{1_3^2}^{\cdot} + p_{2^2}^{\cdot}\, q_{2^2}^{\cdot} + p_{2^2}\, q_{2^2}^{\cdot\cdot}$
    - $w_{2^2}{}^{*} = M_{2^2}\, t_{2^2}^{\cdot} + \Omega_{2^2}\, M_{2^2}\, E_{2^2}\, t_{2^2}$

- Link 3 için:
  - Serbestlik Derecesi 1 için:
    - $t_{3_1^2} = A_{3^2,2^2}\, t_{2^2} + p_{3_1^2}\, \dot{q_{3_1^2}}$
    - $\dot{t_{3_1^2}} = \dot{A}_{3^2,2^2}\, t_{2^2} + A_{3^2,2^2}\, \dot{t}_{2^2} + \dot{p_{3_1^2}}\, \dot{q_{3_1^2}} + p_{3_1^2}\, \ddot{q_{3_1^2}}$
    - $w_{3_1^2}{}^{*} = [0\ 0\ 0\ 0\ 0\ 0]^T$
  - Serbestlik Derecesi 2 için:
    - $t_{3_2^2} = t_{3_1^2} + p_{3_2^2}\, \dot{q_{3_2^2}}$
    - $\dot{t_{3_2^2}} = \dot{t_{3_1^2}} + \dot{p_{3_2^2}}\, \dot{q_{3_2^2}} + p_{3_2^2}\, \ddot{q_{3_2^2}}$
    - $w_{3_2^2}{}^{*} = M_{3^2}\, \dot{t_{3_2^2}} + \Omega_{3_2^2}\, M_{3^2}\, E_{3^2}\, t_{3_2^2}$

"Wrench" ifadeleri link ve serbestlik derecesi seviyelerinde elde edilmiştir. Serbestlik derecesi seviyesinde, ilk serbestlik derecesi için "Twist" vektörü ve "Twist" türevi vektörü oluşturulurken, Twist yayılım matrisi bulunmakta ancak diğer serbestlik derecelerinde Twist yayılım matrisi bulunmamaktadır. Bunun nedeni birden fazla serbestlik derecesine sahip eklemlerin, dönme eksenlerinin bir noktada kesişmesinden kaynaklanmaktadır. Buna ek olarak "Wrench" vektörü son serbestlik derecesinde bir değer alırken, daha düşük serbestlik derecelerinde sıfır vektörüne eşitlenmiştir. Bunun nedeni, sonraki özyinelemelerde görüleceği üzere sadece son serbestlik derecelerine ilişkin "Wrench" vektörü kullanılmasındandır.

Geri Özyineleme

Üst eklemlerin etkileri ve tüm eklemlerle serbest taban arasındaki ilişkiyi tanımlayan eylemsizlik matrisleri hesaplanmaktadır. Sadece simülasyonda kullanılan link seviyesinde eşitlikler aktarılmıştır.

- Modül 2 için:
  - Link 3 için:
    - Serbestlik Derecesi 2 için:
      - $\widetilde{w}_{3_2^2} = w_{3_2^2}{}^{*}$
      - $h_{3_2^2} = p_{3_2^2}{}^{T}\, \widetilde{w}_{3_2^2}$
      - $\widetilde{M}_{3_2^2} = M_{3^2}$

- $K_{3_2^2} = \widetilde{M}_{3_2^2}\, p_{3_2^2}$
- $\widetilde{K}_{3_2^2} = {A_{3^2,0}}^T K_{3_2^2}$

▪ Serbestlik Derecesi 1 için:

- $\widetilde{w}_{3_1^2} = \widetilde{w}_{3_2^2}$
- $h_{3_1^2} = {p_{3_1^2}}^T \widetilde{w}_{3_1^2}$
- $\widetilde{M}_{3_1^2} = \widetilde{M}_{3_2^2}$
- $K_{3_1^2} = \widetilde{M}_{3_1^2}\, p_{3_1^2}$
- $\widetilde{K}_{3_1^2} = {A_{3^2,0}}^T K_{3_1^2}$

○ Link 2 için:

▪ $\widetilde{w}_{2^2} = {w_{2^2}}^* + A^T_{3^2 2^2}\, \widetilde{w}_{3_1^2}$

▪ $h_{2^2} = {p_{2^2}}^T \widetilde{w}_{2^2}$

▪ $\widetilde{M}_{2^2} = M_{2^2} + A^T_{3^2 2^2}\, \widetilde{M}_{3_1^2}\, A_{3^2 2^2}$

▪ $K_{2^2} = \widetilde{M}_{2^2}\, p_{2^2}$

▪ $\widetilde{K}_{2^2} = {A_{2^2,0}}^T K_{2^2}$

○ Link 1 için:

▪ Serbestlik Derecesi 3 için:

- $\widetilde{w}_{1_3^2} = {w_{1_3^2}}^* + A^T_{2^2 1^2}\, \widetilde{w}_{2^2}$
- $h_{1_3^2} = {p_{1_3^2}}^T \widetilde{w}_{1_3^2}$
- $\widetilde{M}_{1_3^2} = M_{2^2} + + A^T_{2^2 1^2}\, \widetilde{M}_{2^2}\, A_{2^2 1^2}$
- $K_{1_3^2} = \widetilde{M}_{1_3^2}\, p_{1_3^2}$
- $\widetilde{K}_{1_3^2} = {A_{1^2,0}}^T K_{1_3^2}$

▪ Serbestlik Derecesi 2 için:

- $\widetilde{w}_{1_2^2} = \widetilde{w}_{1_3^2}$
- $h_{1_2^2} = {p_{1_2^2}}^T \widetilde{w}_{1_2^2}$
- $\widetilde{M}_{1_2^2} = \widetilde{M}_{1_3^2}$
- $K_{1_2^2} = \widetilde{M}_{1_2^2}\, p_{1_2^2}$
- $\widetilde{K}_{1_2^2} = {A_{1^2,0}}^T K_{1_2^2}$

▪ Serbestlik Derecesi 1 için:

- $\widetilde{w}_{1_1^2} = \widetilde{w}_{1_2^2}$
- $h_{1_1^2} = {p_{1_1^2}}^T \widetilde{w}_{1_1^2}$

- $\widetilde{M}_{1_1^2} = \widetilde{M}_{1_2^2}$
- $K_{1_1^2} = \widetilde{M}_{1_1^2}\, p_{1_1^2}$
- $\widetilde{K}_{1_1^2} = {A_{1^2,0}}^T K_{1_1^2}$

- Modül 1 için:
  - Link 3 için:
    - Serbestlik Derecesi 2 için:
      - $\widetilde{w}_{3_2^1} = {w_{3_2^1}}^*$
      - $h_{3_2^1} = {p_{3_2^1}}^T \widetilde{w}_{3_2^1}$
      - $\widetilde{M}_{3_2^1} = M_{3^1}$
      - $K_{3_2^1} = \widetilde{M}_{3_2^1}\, p_{3_2^1}$
      - $\widetilde{K}_{3_2^1} = {A_{3^1,0}}^T K_{3_2^1}$
    - Serbestlik Derecesi 1 için:
      - $\widetilde{w}_{3_1^1} = \widetilde{w}_{3_2^1}$
      - $h_{3_1^1} = {p_{3_1^1}}^T \widetilde{w}_{3_1^1}$
      - $\widetilde{M}_{3_1^1} = \widetilde{M}_{3_2^1}$
      - $K_{3_1^1} = \widetilde{M}_{3_1^1}\, p_{3_1^1}$
      - $\widetilde{K}_{3_1^1} = {A_{3^1,0}}^T K_{3_1^1}$
  - Link 2 için:
    - $\widetilde{w}_{2^1} = {w_{2^1}}^* + A^T_{3^1 2^1}\, \widetilde{w}_{3_1^1}$
    - $h_{2^1} = {p_{2^1}}^T \widetilde{w}_{2^1}$
    - $\widetilde{M}_{2^1} = M_{2^1} + A^T_{3^1 2^1}\, \widetilde{M}_{3_1^1}\, A_{3^1 2^1}$
    - $K_{2^1} = \widetilde{M}_{2^1}\, p_{2^1}$
    - $\widetilde{K}_{2^1} = {A_{2^1,0}}^T K_{2^1}$
  - Link 1 için:
    - Serbestlik Derecesi 3 için:
      - $\widetilde{w}_{1_3^1} = {w_{1_3^1}}^* + A^T_{2^1 1^1}\, \widetilde{w}_{2^1}$
      - $h_{1_3^1} = {p_{1_3^1}}^T \widetilde{w}_{1_3^1}$
      - $\widetilde{M}_{1_3^1} = M_{2^1} + + A^T_{2^1 1^1}\, \widetilde{M}_{2^1}\, A_{2^1 1^1}$
      - $K_{1_3^1} = \widetilde{M}_{1_3^1}\, p_{1_3^1}$
      - $\widetilde{K}_{1_3^1} = {A_{1^1,0}}^T K_{1_3^1}$

- Serbestlik Derecesi 2 için:
  - $\widetilde{w}_{1_2^1} = \widetilde{w}_{1_3^1}$
  - $h_{1_2^1} = {p_{1_2^1}}^T \widetilde{w}_{1_2^1}$
  - $\widetilde{M}_{1_2^1} = \widetilde{M}_{1_3^1}$
  - $K_{1_2^1} = \widetilde{M}_{1_2^1} p_{1_2^1}$
  - $\widetilde{K}_{1_2^1} = {A_{1^1,0}}^T K_{1_2^1}$
- Serbestlik Derecesi 1 için:
  - $\widetilde{w}_{1_1^1} = \widetilde{w}_{1_2^1}$
  - $h_{1_1^1} = {p_{1_1^1}}^T \widetilde{w}_{1_1^1}$
  - $\widetilde{M}_{1_1^1} = \widetilde{M}_{1_2^1}$
  - $K_{1_1^1} = \widetilde{M}_{1_1^1} p_{1_1^1}$
  - $\widetilde{K}_{1_1^1} = {A_{1^1,0}}^T K_{1_1^1}$

- Modül 0 için:
  - Link 1 için:
    - $\widetilde{w}_0 = {w_0}^* + A_{1^1 0}^T \widetilde{w}_{1_1^1} + A_{1^2 0}^T \widetilde{w}_{1_1^2}$
    - $h_0 = {p_0}^T \widetilde{w}_0$
    - $\widetilde{M}_0 = M_0 + A_{1^1 0}^T \widetilde{M}_{1_1^1} A_{1^1 0} + A_{1^2 0}^T \widetilde{M}_{1_1^2} A_{1^2 0}$
    - $K_0 = \widetilde{M}_0 P_0$
    - $I_{0,0} = {K_0}^T P_0$

Robotun tip noktalarından, tabana doğru gerçekleştirilen bu özyineleme aşamasında, serbest taban etkisini içermeyen tork değerleri hesaplanırken, "$\widetilde{w}$" vektörü kullanılır. Bu vektörün hesaplanması, serbestlik derecesi sırasına göre değişmektedir. Son serbestlik derecesinde, üst eklem etkileri, Twist yayılım matrisi ile çarpılarak aktarılırken, diğer serbestlik derecelerinde direk olarak "$\widetilde{w}$" vektörü bir üst serbestlik derecesinin "$\widetilde{w}$" vektörüne eşitlenmektedir. Bu durum, "$\widetilde{M}$" ile belirtilen "Eklemli" kütle matrisi için de geçerlidir.

"Eklemli" kütle matrisi ve ilgili eklemden tabana olan Twist yayılım matrisi, eklemlerle, taban arasındaki eylemsizlik matrisinin hesaplanmasında kullanılmaktadır. Bu çıktı, bir sonraki özyinelemede gerçek eklem torklarının hesaplanmasında etkilidir.

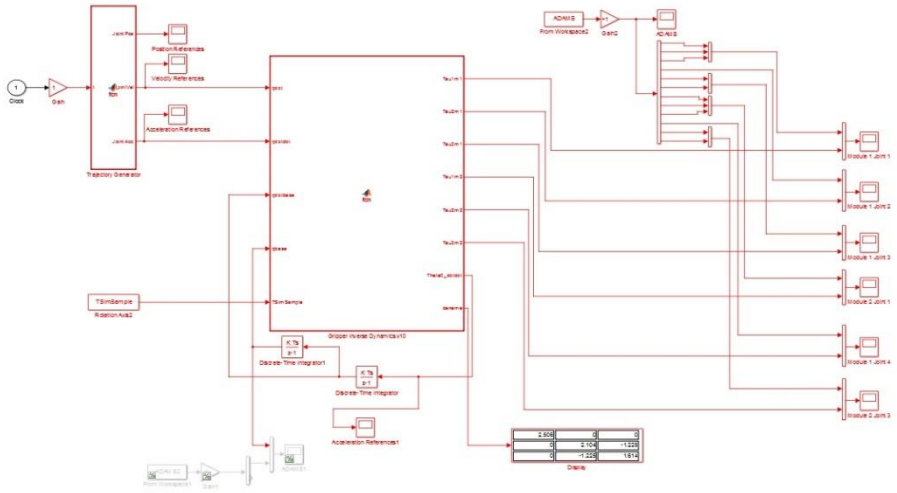
İleri Özyineleme

Son ileri özyinelemede serbest taban etkisini içeren tork değerleri ve tabanın ivmeleri hesaplanmaktadır.

- Modül 0 için:
  - Link 0 için:
    - $\ddot{q}_0 = -I_{0,0}^{-1}\,\tau_0$
    - $\tilde{q}_0 = P_0\,\ddot{q}_0$
- Modül 1 için:
  - Link 1 için:
    - $\tau_{1_1^1} = \widetilde{K}_{1_1^1}{}^T \tilde{q}_0 + h_{1_1^1}$
    - $\tau_{1_2^1} = \widetilde{K}_{1_2^1}{}^T \tilde{q}_0 + h_{1_2^1}$
    - $\tau_{1_3^1} = \widetilde{K}_{1_3^1}{}^T \tilde{q}_0 + h_{1_3^1}$
  - Link 2 için:
    - $\tau_{2^1} = \widetilde{K}_{2^1}{}^T \tilde{q}_0 + h_{2^1}$
  - Link 3 için:
    - $\tau_{3_1^1} = \widetilde{K}_{3_1^1}{}^T \tilde{q}_0 + h_{3_1^1}$
    - $\tau_{3_2^1} = \widetilde{K}_{3_2^1}{}^T \tilde{q}_0 + h_{3_2^1}$
- Modül 2 için:
  - Link 1 için:
    - $\tau_{1_1^2} = \widetilde{K}_{1_1^2}{}^T \tilde{q}_0 + h_{1_1^2}$
    - $\tau_{1_2^2} = \widetilde{K}_{1_2^2}{}^T \tilde{q}_0 + h_{1_2^2}$
    - $\tau_{1_3^2} = \widetilde{K}_{1_3^2}{}^T \tilde{q}_0 + h_{1_3^2}$
  - Link 2 için:
    - $\tau_{2^2} = \widetilde{K}_{2^2}{}^T \tilde{q}_0 + h_{2^2}$
  - Link 3 için:
    - $\tau_{3_1^2} = \widetilde{K}_{3_1^2}{}^T \tilde{q}_0 + h_{3_1^2}$
    - $\tau_{3_2^2} = \widetilde{K}_{3_2^2}{}^T \tilde{q}_0 + h_{3_2^2}$

4.2.1. Ters dinamik simülasyonu

3 boyutlu yürüyen robota ilişkin verilen link seviyesindeki özyineleme şeması MATLAB Simulink ortamında simüle edilmiştir. Genel şema Şekil 4.8'de görülmektedir.



Şekil 4.8 : 3 boyutlu yürüyen robot ters dinamik MATLAB algoritması

Simülasyonda girdi olarak S eğrisi biçiminde eklem açısal pozisyon referansları kullanılmıştır. Birinci ve ikinci türevi alınarak, hız ve ivme referansları da hesaplanmış ve girdi olarak uygulanmıştır.

$$\theta_i = \frac{2\,\pi}{T_{Periyot}^2}\left(\theta_{final_i} - \theta_{başlangıç_i}\right)\sin\left(\frac{2\,\pi\,t}{T_{Periyot}}\right) \tag{4.9}$$

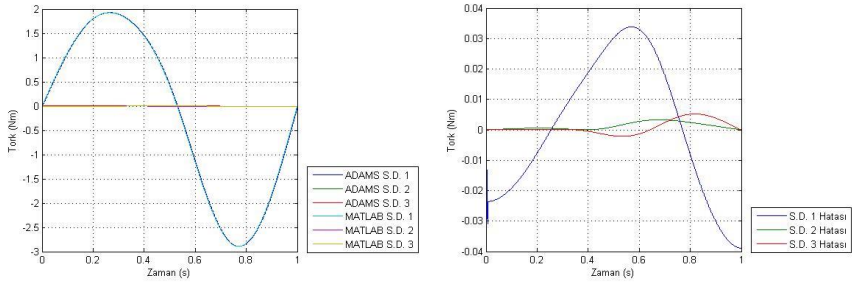
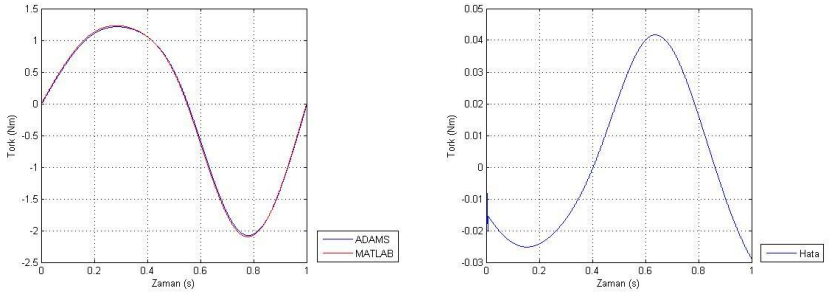
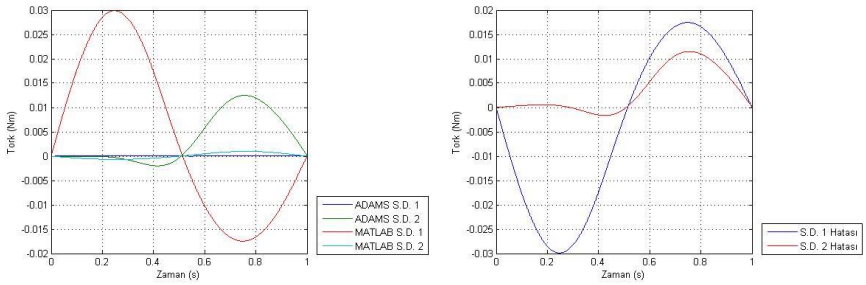
Tablo 4.4 : Eklem başlangıç ve final pozisyonları

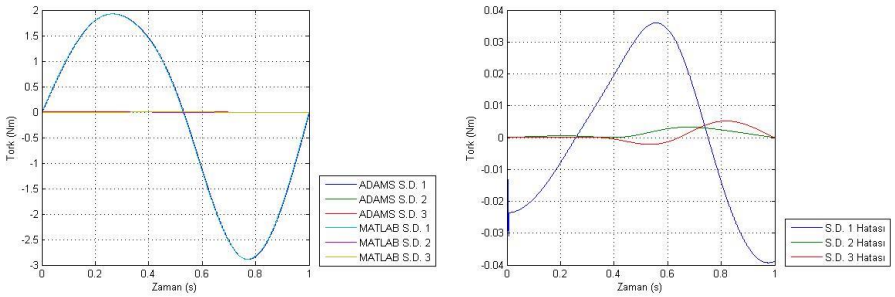
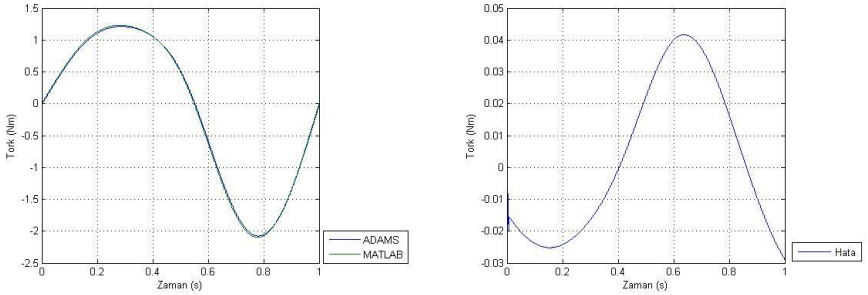
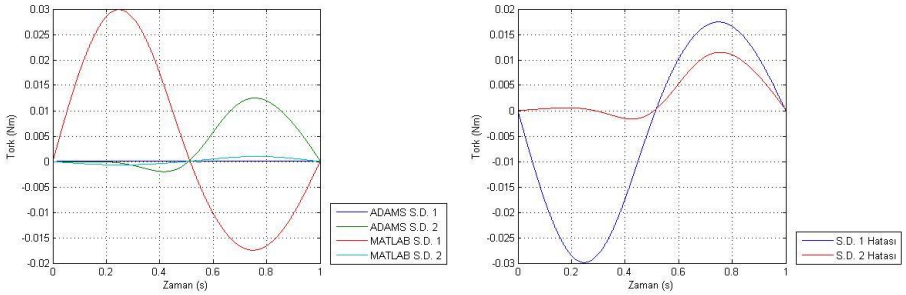
	1_1^1	1_2^1	1_3^1	2^1	3_1^1	3_2^1	1_1^2	1_2^2	1_3^2	2^2	3_1^2	3_2^2
$\theta_{başlangıç_i}$	0	0	0	0	0	0	0	0	0	0	0	0
θ_{final_i}	-60	0	0	-60	0	-60	-60	0	0	-60	0	-60

Tüm eklemlerin bir serbestlik derecesine aynı açı farkları verilmiştir. Bu referanslar hem MATLAB algoritmasında hem de ADAMS ortamında uygulanarak, tork ölçümleri karşılaştırılmıştır. Robot, yüksek sayıda serbestlik derecesine sahip olduğundan robot eklemlerine ilişkin ADAMS ve MATLAB çıktılarının

karşılaştırılması Tablo 4.5'de verilmiştir. Her satırda iki adet grafik verilerek, soldaki grafikle ADAMS ve MATLAB tork değerleri sağdaki grafikle aralarındaki hatalar gösterilmiştir. Böylece hatanın genliği daha iyi anlatılmaya çalışılmıştır.

Tablo 4.5 : Yürüyen robot ters dinamik eklem tork karşılaştırma tablosu

Eklem No	ADAMS ve MATLAB Tork Çıktıları (Solda) Eklem Tork Hesaplama Hatası Grafiği (Sağda)
1^1	
2^1	
3^1	

1^2	
2^2	
3^2	

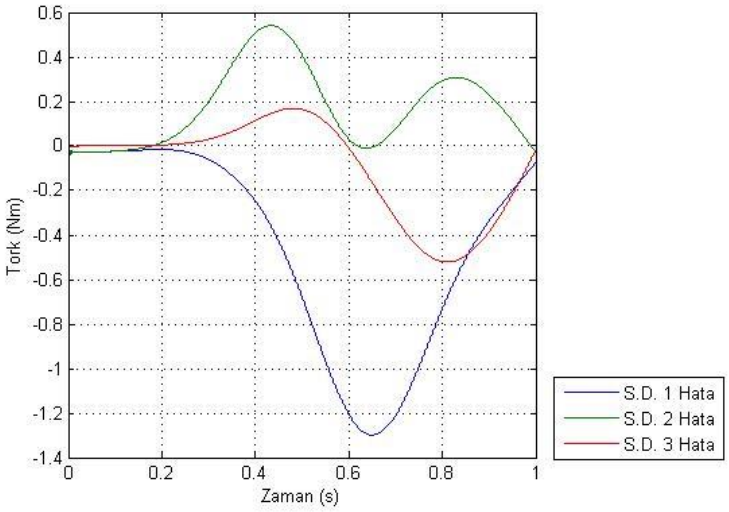
Tablo 4.5'ye göre modüllerin ilk iki eklemlerindeki hatalar düşük kalmakta ancak üçüncü eklemlere ilişkin hatalar oldukça yüksek kalmaktadır. Referans olarak her eklemden sadece tek serbestlik derecesine referans verilmesiyle elde edilen bu grafikler, birden fazla serbestlik derecesine aynı anda ivme referansı verildiğinde hataların daha da arttığı gözlenmektedir.

Örneğin Tablo 4.6'teki girdiler için tork çıktılar incelensin:

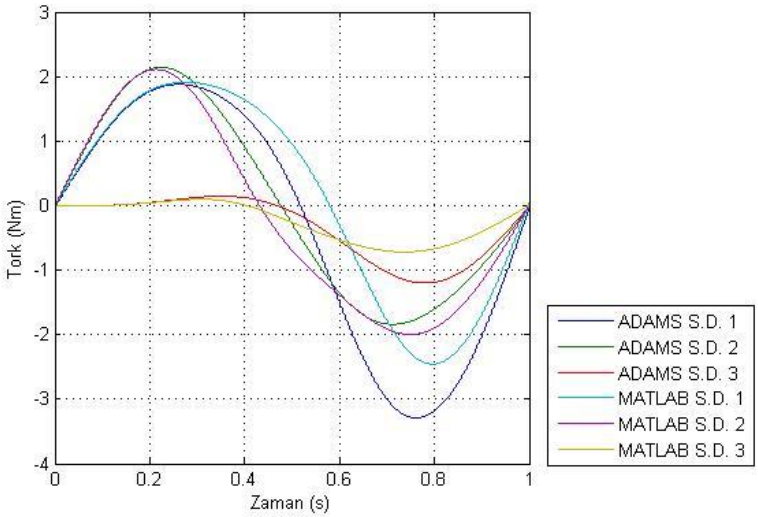
Tablo 4.6 : Eklem başlangıç ve final pozisyonları

	1_1^1	1_2^1	1_3^1	2^1	3_1^1	3_2^1	1_1^2	1_2^2	1_3^2	2^2	3_1^2	3_2^2
$\theta_{başlangıç_i}$	0	0	0	0	0	0	0	0	0	0	0	0
θ_{final_i}	-60	-60	0	-60	0	0	-60	-60	0	-60	0	0

Bu referanslara göre 1 nolu modülün ilk eklemine ilişkin tork çıktıları ve ADAMS ile MATLAB arasındaki hata, Şekil 4.9 ve Şekil 4.10'da belirtilmiştir.



Şekil 4.9 : Modül 1 eklem 1 ADAMS ile MATLAB arasındaki hata



Şekil 4.10 : Modül 1 eklem 1 ADAMS ve MATLAB tork çıktıları

İlk ekleme ilişkin tork çıktılarına bakıldığında hataların oldukça yüksek çıktığı söylenebilir. Şunu belirtmek gerekir ki, sadece tek serbestlik derecesine referans uygulandığında hatalar düşük kalmakta iken, aynı anda birden fazla serbestlik derecesine referans uygulandığında simülasyon başlangıcında hata sıfıra yakınken, simülasyon ilerledikçe hatalar artmaktadır.

4.3. İleri Dinamik Modelleme

Serbest tabanlı olan 3 boyutlu yürüyen robot için ileri dinamik modelleme üç aşamadan meydana gelmektedir. Ters dinamikle ortak noktaları bulunmaktadır. Verilen tork girdilerine göre eklem ivmeleri ve taban ivmeleri hesaplanmaktadır. Modül seviyesindeki denklemler aktarılmadan simülasyonda kullanılan link ve serbestlik seviyesi denklemleri aktarılmıştır.

İleri Özyineleme

İlk ileri özyineleme eklem ivmeleri sıfır alınarak çözülen ters dinamik ilk ileri özyineleme ile aynıdır.

Geri Özyineleme

İleri dinamiğe giriş olarak verilen eklem torklarına ve eklem hızları kullanılarak ilk ileri özyinelemede hesaplanan "Wrench" vektörlerine göre, eklemlerin birbirine olan etkilerini de içeren eklem tork değerlerini hesaplar. Ancak serbest tabanın hareketinden kaynaklı olan etkiler bu aşamada dahil edilmez. Link ve serbestlik derecesi seviyesindeki eşitlikler aşağıda sıralanmıştır.

- Modül 2 için:
  - Link 3 için:
    - Serbestlik Derecesi 2 için:
      - $\hat{\tau}_{3_2^2} = \tau_{3_2^2} - p_{3_2^2}^T \left(w_{3_2^2}{}^*\right)$
      - $\tilde{\tau}_{3_2^2} = \hat{I}_{3_2^2}{}^{-1} \hat{\tau}_{3_2^2}$
      - $\eta_{3_2^2} = \Psi_{3_2^2} \hat{\tau}_{3_2^2}$

- Serbestlik Derecesi 1 için:
  - $\tilde{\eta}_{3_1^2} = \eta_{3_2^2}$
  - $\hat{\tau}_{3_1^2} = \tau_{3_2^2} - p_{3_2^2}^T (\tilde{\eta}_{3_1^2})$
  - $\tilde{\tau}_{3_1^2} = \hat{I}_{3_1^2}{}^{-1} \hat{\tau}_{3_1^2}$
  - $\eta_{3_1^2} = \Psi_{3_1^2} \hat{\tau}_{3_1^2}$

- Link 2 için:
  - $\tilde{\eta}_{2^2} = A_{3^2,2^2}^T \eta_{3_1^2}$
  - $\hat{\tau}_{2^2} = \tau_{2^2} - p_{2^2}^T (\tilde{\eta}_{2^2} + w_{2^2}{}^*)$
  - $\tilde{\tau}_{2^2} = \hat{I}_{2^2}{}^{-1} \hat{\tau}_{2^2}$
  - $\eta_{2^2} = \Psi_{2^2} \hat{\tau}_{2^2}$

- Link 1 için:
  - Serbestlik Derecesi 3 için:
    - $\tilde{\eta}_{1_3^2} = A_{2^2,1^2}^T \eta_{2^2}$
    - $\hat{\tau}_{1_3^2} = \tau_{1_3^2} - p_{1_3^2}^T (\tilde{\eta}_{1_3^2} + w_{1_3^2}{}^*)$
    - $\tilde{\tau}_{1_3^2} = \hat{I}_{1_3^2}{}^{-1} \hat{\tau}_{1_3^2}$
    - $\eta_{1_3^2} = \Psi_{1_3^2} \hat{\tau}_{1_3^2}$
  - Serbestlik Derecesi 2 için:
    - $\tilde{\eta}_{1_2^2} = \tilde{\eta}_{1_3^2}$
    - $\hat{\tau}_{1_2^2} = \tau_{1_2^2} - p_{1_2^2}^T (\tilde{\eta}_{1_2^2} + w_{1_2^2}{}^*)$
    - $\tilde{\tau}_{1_2^2} = \hat{I}_{1_2^2}{}^{-1} \hat{\tau}_{1_2^2}$
    - $\eta_{1_2^2} = \Psi_{1_2^2} \hat{\tau}_{1_2^2}$
  - Serbestlik Derecesi 1 için:
    - $\tilde{\eta}_{1_1^2} = \eta_{1_2^2}$
    - $\hat{\tau}_{1_1^2} = \tau_{1_1^2} - p_{1_1^2}^T (\tilde{\eta}_{1_1^2} + w_{1_1^2}{}^*)$
    - $\tilde{\tau}_{1_1^2} = \hat{I}_{1_1^2}{}^{-1} \hat{\tau}_{1_1^2}$
    - $\eta_{1_1^2} = \Psi_{1_1^2} \hat{\tau}_{1_1^2}$

- Modül 1 için:
  - Link 3 için:
    - Serbestlik Derecesi 2 için:
      - $\hat{\tau}_{3_2^1} = \tau_{3_2^1} - p_{3_2^1}^T \left(w_{3_2^1}{}^*\right)$
      - $\tilde{\tau}_{3_2^1} = \hat{I}_{3_2^1}{}^{-1} \hat{\tau}_{3_2^1}$
      - $\eta_{3_2^1} = \Psi_{3_2^1} \hat{\tau}_{3_2^1}$
    - Serbestlik Derecesi 1 için:
      - $\tilde{\eta}_{3_1^1} = \eta_{3_2^1}$
      - $\hat{\tau}_{3_1^1} = \tau_{3_2^1} - p_{3_2^1}^T \left(\tilde{\eta}_{3_1^1}\right)$
      - $\tilde{\tau}_{3_1^1} = \hat{I}_{3_1^1}{}^{-1} \hat{\tau}_{3_1^1}$
      - $\eta_{3_1^1} = \Psi_{3_1^1} \hat{\tau}_{3_1^1}$
  - Link 2 için:
    - $\tilde{\eta}_{2^1} = A_{3^1,2^1}^T \eta_{3_1^1}$
    - $\hat{\tau}_{2^1} = \tau_{2^1} - p_{2^1}^T \left(\tilde{\eta}_{2^1} + w_{2^1}{}^*\right)$
    - $\tilde{\tau}_{2^1} = \hat{I}_{2^1}{}^{-1} \hat{\tau}_{2^1}$
    - $\eta_{2^1} = \Psi_{2^1} \hat{\tau}_{2^1}$
  - Link 1 için:
    - Serbestlik Derecesi 3 için:
      - $\tilde{\eta}_{1_3^1} = A_{2^1,1^1}^T \eta_{2^1}$
      - $\hat{\tau}_{1_3^1} = \tau_{1_3^1} - p_{1_3^1}^T \left(\tilde{\eta}_{1_3^1} + w_{1_3^1}{}^*\right)$
      - $\tilde{\tau}_{1_3^1} = \hat{I}_{1_3^1}{}^{-1} \hat{\tau}_{1_3^1}$
      - $\eta_{1_3^1} = \Psi_{1_3^1} \hat{\tau}_{1_3^1}$
    - Serbestlik Derecesi 2 için:
      - $\tilde{\eta}_{1_2^1} = \tilde{\eta}_{1_3^1}$
      - $\hat{\tau}_{1_2^1} = \tau_{1_2^1} - p_{1_2^1}^T \left(\tilde{\eta}_{1_2^1} + w_{1_2^1}{}^*\right)$
      - $\tilde{\tau}_{1_2^1} = \hat{I}_{1_2^1}{}^{-1} \hat{\tau}_{1_2^1}$
      - $\eta_{1_2^1} = \Psi_{1_2^1} \hat{\tau}_{1_2^1}$

- Serbestlik Derecesi 1 için:
  - $\tilde{\eta}_{1_1^1} = \eta_{1_2^1}$
  - $\hat{\tau}_{1_1^1} = \tau_{1_1^1} - p_{1_1^1}^T (\tilde{\eta}_{1_1^1} + w_{1_1^1}{}^*)$
  - $\tilde{\tau}_{1_1^1} = \hat{I}_{1_1^1}{}^{-1} \hat{\tau}_{1_1^1}$
  - $\eta_{1_1^1} = \Psi_{1_1^1} \hat{\tau}_{1_1^1}$
- Modül 0 için:
  - $\tilde{\eta}_0 = A_{1^1,0}^T \eta_{1_1^1} + A_{1^2,0}^T \eta_{1_1^2}$
  - $\hat{\tau}_0 = -p_0^T (\tilde{\eta}_0 + w_0{}^*)$
  - $\tilde{\tau}_0 = \hat{I}_0{}^{-1} \hat{\tau}_0$

Serbest taban etkisini içermeyen ancak eklemlerin birbirine olan etkilerini içeren eklem "Wrench" vektörlerinin hesaplanması link ve serbest taban düzeyinde verilmiştir.

İleri Özyineleme

Son aşama olan bu ileri özyineleme da taban ivmeleri, eklem torklarına göre hesaplanmakta ve bu ivmelerin etkileri tabandan tip noktalara doğru aktarılmaktadır. Link ve serbestlik derecesi seviyesinde verilen denklemler takip eden satırlarda 3 boyutlu yürüyen robot için sıralanmıştır.

- Modül 0 için:
  - $\ddot{q}_0 = \tilde{\tau}_0$
  - $\mu_0 = p_0 \ddot{q}_0$
- Modül 1 için:
  - Link 1 için:
    - Serbestlik Derecesi 1 için:
      - $\tilde{\mu}_{1_1^1} = A_{1^1,0} \mu_0$
      - $\ddot{q}_{1_1^1} = \tilde{\tau}_{1_1^1} - \Psi_{1_1^1}^T \tilde{\mu}_{1_1^1}$
      - $\mu_{1_1^1} = p_{1_1^1} \ddot{q}_{1_1^1} + \tilde{\mu}_{1_1^1}$

- Serbestlik Derecesi 2 için:
  - $\tilde{\mu}_{1_2^1} = \mu_{1_1^1}$
  - $\ddot{q}_{1_2^1} = \tilde{\tau}_{1_2^1} - \Psi_{1_2^1}^T \tilde{\mu}_{1_2^1}$
  - $\mu_{1_2^1} = p_{1_2^1} \ddot{q}_{1_2^1} + \tilde{\mu}_{1_2^1}$
- Serbestlik Derecesi 3 için:
  - $\tilde{\mu}_{1_3^1} = \mu_{1_2^1}$
  - $\ddot{q}_{1_3^1} = \tilde{\tau}_{1_3^1} - \Psi_{1_3^1}^T \tilde{\mu}_{1_3^1}$
  - $\mu_{1_3^1} = p_{1_3^1} \ddot{q}_{1_3^1} + \tilde{\mu}_{1_3^1}$

o Link 2 için:
- $\tilde{\mu}_{2^1} = A_{2^1,1^1} \mu_{1_3^1}$
- $\ddot{q}_{2^1} = \tilde{\tau}_{2^1} - \Psi_{2^1}^T \tilde{\mu}_{2^1}$
- $\mu_{2^1} = p_{2^1} \ddot{q}_{2^1} + \tilde{\mu}_{2^1}$

o Link 3 için:
- Serbestlik Derecesi 1 için:
  - $\tilde{\mu}_{3_1^1} = A_{3^1,2^1} \mu_{2^1}$
  - $\ddot{q}_{3_1^1} = \tilde{\tau}_{3_1^1} - \Psi_{3_1^1}^T \tilde{\mu}_{3_1^1}$
  - $\mu_{3_1^1} = p_{3_1^1} \ddot{q}_{3_1^1} + \tilde{\mu}_{3_1^1}$
- Serbestlik Derecesi 2 için:
  - $\tilde{\mu}_{3_2^1} = \mu_{3_1^1}$
  - $\ddot{q}_{3_2^1} = \tilde{\tau}_{3_2^1} - \Psi_{3_2^1}^T \tilde{\mu}_{3_2^1}$
  - $\mu_{3_2^1} = p_{3_2^1} \ddot{q}_{3_2^1} + \tilde{\mu}_{3_2^1}$

- Modül 2 için:

o Link 1 için:
- Serbestlik Derecesi 1 için:
  - $\tilde{\mu}_{1_1^2} = A_{1^2,0} \mu_0$
  - $\ddot{q}_{1_1^2} = \tilde{\tau}_{1_1^2} - \Psi_{1_1^2}^T \tilde{\mu}_{1_1^2}$
  - $\mu_{1_1^2} = p_{1_1^2} \ddot{q}_{1_1^2} + \tilde{\mu}_{1_1^2}$

- Serbestlik Derecesi 2 için:
  - $\tilde{\mu}_{1_2^2} = \mu_{1_1^2}$
  - $\ddot{q}_{1_2^2} = \tilde{\tau}_{1_2^2} - \Psi_{1_2^2}^T \tilde{\mu}_{1_2^2}$
  - $\mu_{1_2^2} = p_{1_2^2} \ddot{q}_{1_2^2} + \tilde{\mu}_{1_2^2}$
- Serbestlik Derecesi 3 için:
  - $\tilde{\mu}_{1_3^2} = \mu_{1_2^2}$
  - $\ddot{q}_{1_3^2} = \tilde{\tau}_{1_3^2} - \Psi_{1_3^2}^T \tilde{\mu}_{1_3^2}$
  - $\mu_{1_3^2} = p_{1_3^2} \ddot{q}_{1_3^2} + \tilde{\mu}_{1_3^2}$

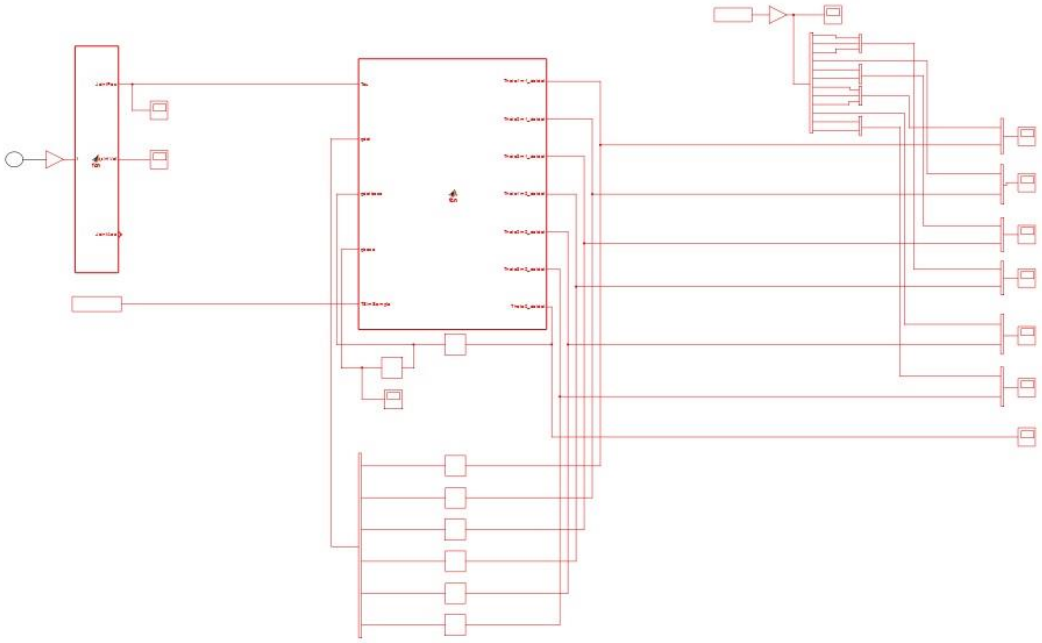
o Link 2 için:
- $\tilde{\mu}_{2^2} = A_{2^2,1^2} \mu_{1_3^2}$
- $\ddot{q}_{2^2} = \tilde{\tau}_{2^2} - \Psi_{2^2}^T \tilde{\mu}_{2^2}$
- $\mu_{2^2} = p_{2^2} \ddot{q}_{2^2} + \tilde{\mu}_{2^2}$

o Link 3 için:
- Serbestlik Derecesi 1 için:
  - $\tilde{\mu}_{3_1^2} = A_{3^2,2^2} \mu_{2^2}$
  - $\ddot{q}_{3_1^2} = \tilde{\tau}_{3_1^2} - \Psi_{3_1^2}^T \tilde{\mu}_{3_1^2}$
  - $\mu_{3_1^2} = p_{3_1^2} \ddot{q}_{3_1^2} + \tilde{\mu}_{3_1^2}$
- Serbestlik Derecesi 2 için:
  - $\tilde{\mu}_{3_2^2} = \mu_{3_1^2}$
  - $\ddot{q}_{3_2^2} = \tilde{\tau}_{3_2^2} - \Psi_{3_2^2}^T \tilde{\mu}_{3_2^2}$
  - $\mu_{3_2^2} = p_{3_2^2} \ddot{q}_{3_2^2} + \tilde{\mu}_{3_2^2}$

İvmelerin yayılması ile, serbest taban etkilerinin ve eklem etkilerinin tamamını içeren eklem ivmeleri hesaplanmıştır. Böylece 3 boyutlu yürüyen robota ait tüm ileri dinamik özyineleme şeması ortaya çıkartılmıştır.

4.3.1. İleri dinamik simülasyonu

Link seviyesinde verilen ileri dinamik özyineleme şeması ışığında "m-file" sentaksında algoritma oluşturulmuştur. Genel şema yine Simulink diyagramı olarak meydana getirilmiştir. Şekil 4.11'de bu şema görülmektedir.



Şekil 4.11 : 3 boyutlu yürüyen robot ileri dinamik MATLAB algoritması

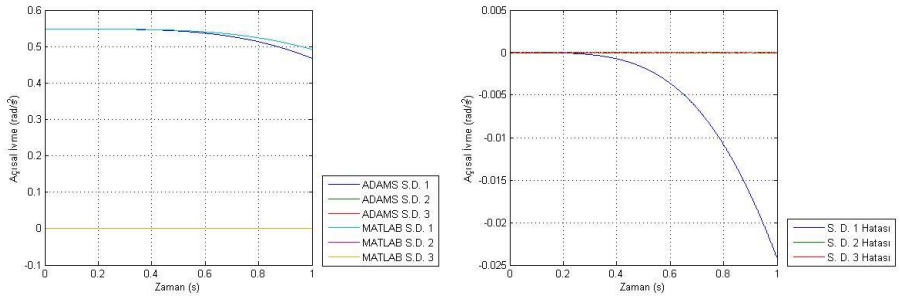
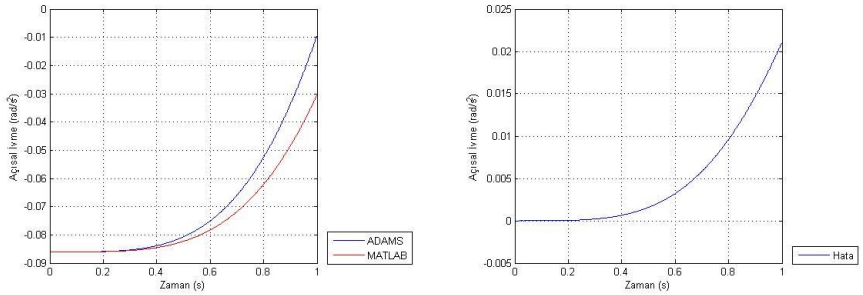
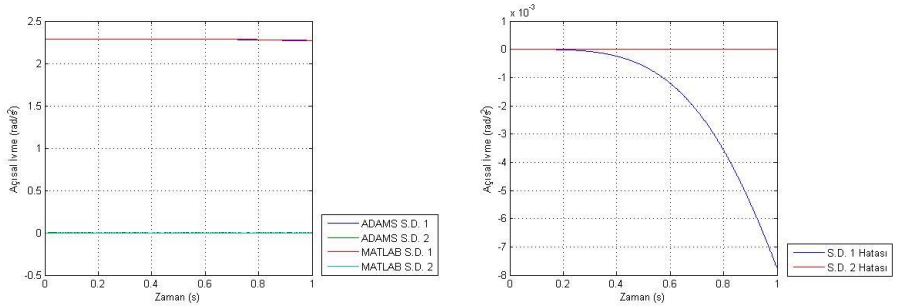
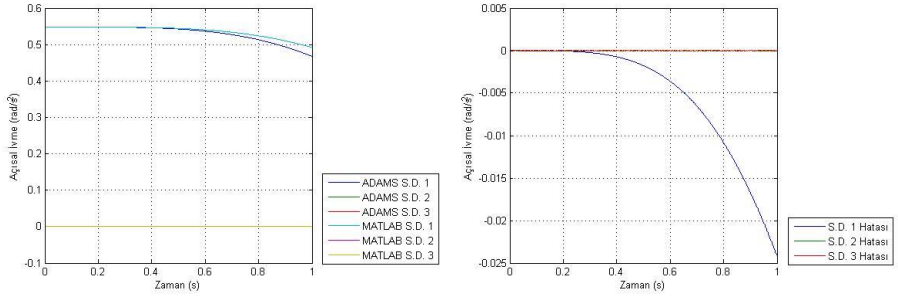
İleri dinamik algoritması denenirken eklemlere uygulanan tork değerleri Tablo 4.7 ile verilmiştir.

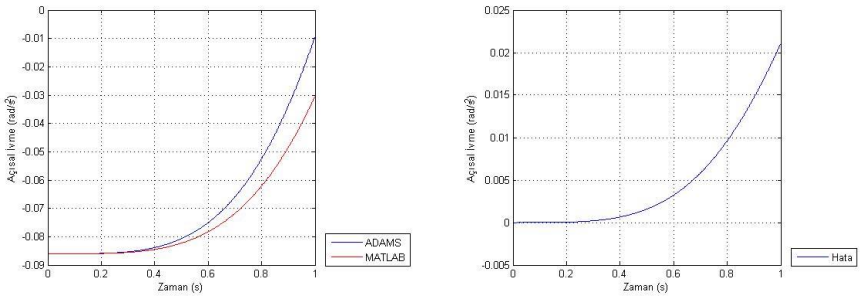
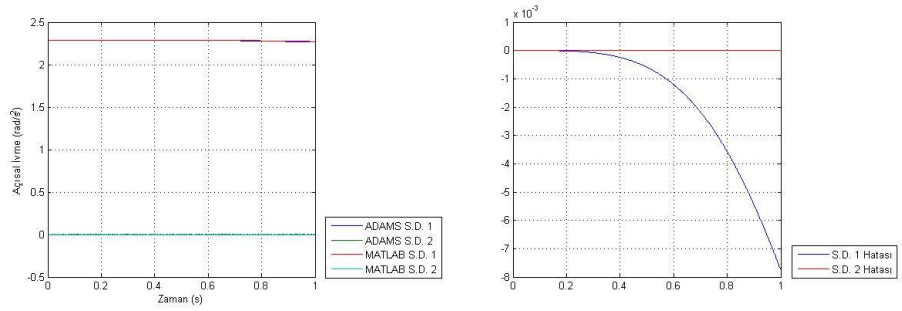
Tablo 4.7 : Eklem başlangıç ve final pozisyonları

	1_1^1	1_2^1	1_3^1	2^1	3_1^1	3_2^1	1_1^2	1_2^2	1_3^2	2^2	3_1^2	3_2^2
$\tau\ (Nm)$	0.1	0	0	0.052	0.001	0	0.1	0	0	0.052	0.001	0

Tüm eklemlerin ilk serbestlik derecelerine uygulanan tork değerleri, hem ADAMS ortamında hem de MATLAB algoritmasında kullanılmış ve bu değerlere karşılık olarak hesaplanan eklem ivmeleri bir tablo halinde karşılaştırmalı grafiklerle Tablo 4.8'de aktarılmıştır.

Tablo 4.8 : Yürüyen robot ileri dinamik eklem ivme karşılaştırma tablosu

Eklem No	ADAMS ve MATLAB İvme Çıktıları (Solda) Eklem İvme Hesaplama Hatası Grafiği (Sağda)
1^1	
2^1	
3^1	
1^2	

2^2	
3^2	

Tablo incelendiğinde MATLAB ile ADAMS arasındaki hatanın simülasyonun ilk anlarında çok çok küçük iken, simülasyon ilerledikçe logaritmik olarak hatanın arttığı görülmektedir. Burada algoritmanın hatasının logaritmik olarak artması, kısıtlı bir çalışma süresi altında simülasyonun doğru, uzun simülasyon sürelerinde hatalı çalıştığı sonucuna varılmasına neden olmaktadır.

5. SONUÇ VE ÖNERİLER

Bu çalışmada, yürüyen robotların kinematik ve dinamik analizleri modüler yöntem ile elde edilmiş ve örnekler üzerinden simülasyon ortamında algoritmalar oluşturulmuştur. Algoritmaların doğruluğu, yine karmaşık kinematik yapıların çözümünü yapabilen ve endüstride güvenilirliği kanıtlanmış bir yazılımla kontrol edilmiştir. Oluşturulan örnekler sabit ve sabit olmayan tabanlı örnekleri içermektedir. Tüm örnekler için modüler yöntemle oluşturulan MATLAB algoritmasıyla elde edilen ileri ve ters dinamik çıktılar, ADAMS çıktıları ile büyük oranda örtüşmüştür. Yürüyen robotların yörünge planlanmasına yönelik çalışma yapılmış ve robot MATLAB ortamında görselleştirilerek yörünge planlamasının uygunluğu gösterilmiştir.

Bu çalışma robotun dinamik ve kinematik analizlerini içermekte, kontrol ile ilgili herhangi bir bilgi içermemektedir. Bu çalışmada ters dinamik elde edildiğinden, kolaylıkla hesaplamalı tork kontrol yöntemi uygulanabilir. Daha ileri seviye ve literatürde az bulunan özyinelemeli dinamik model tabanlı adaptif veya robust kontrol konuları araştırmaya açık konular olarak bulunmaktadır.

KAYNAKLAR

1- **Ardema M.D.** (2005). *Newton-Euler Dynamics.* Springer Science + Business Media.

2- **Fearherstone R.** (1987). *Robot Dynamics Algorithm.* Springer Science+Business Media New York

3- **Hanlei W.** (2010). *On the Recursive Implementation of Adaptive Control for Robot Manipulators.* Chinese Control Conference

4- **Jain A., Rodriguez G.** (1993). *An Analysis of the Kinematics and Dynamics of Under-Actuated Manipulators.* IEEE Transaction on Robotic and Automation, vol 9, pp. 411-422.

5- **Kajita S., Tani K.,** (1996). *Experimental Study of Biped Dynamical Walking.* IEEE Control Systems

6- **Lambers J.** (2009). *Mat 610 Summer Session 2009-10 Lecture 4 Notes*. The University of Southern Mississippi.

7- **Saha S. K**. (1997). *A Decomposition of the Manipulator Inertia Matrix.* IEEE Transactions on Robotics and Automation, Vol. 13, No. 2

8- **Shih, C. L., Gruver, W. A.,& Lee, T. T.** (1993). *Inverse kinematics and inverse dynamics for control of a biped walking machine.* Journal of Robotic Systems, 10(4), 531–555.

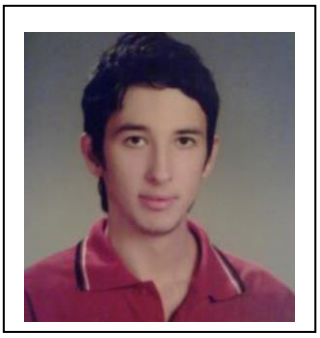
9- **Viyajkumar Shah S., Saha S. K., Dutt J.K**. (2013). *Dynamics of Tree-Type Robotic Systems*. Springer Netherlands.

10- **Vukobratovic M., Potkonjak V., Babkovic K., Borovac B.** (2007). *Simulation model of general human and humanoid motion*. Springer Science + Business Media B.V.

11- **Vukobratovic M., Stepanenko J.,** (1972), *On The Stability of Anthropomorphic Systems.* Mathematical Biosciences.

12- **Yeşiloğlu S. M., Temeltaş H.** (2010). *Dynamical Modelling of Cooperating Underactuated Manipulatos for Space Manipulation.* ADVANCED ROBOTICS Volume: 24, Issue: 3, Pages: 325-341.

ÖZGEÇMİŞ



Ad-Soyad **: Süleyman Baran HEPGÜVEN**

Doğum Tarihi ve Yeri **: 21.02.1991 – İnegöl/Bursa**

E-posta **:** hepguven@itu.edu.tr

ÖĞRENİM DURUMU:

- **Lisans** **:** 2013, İstanbul Teknik Üniversitesi, Elektrik Elektronik Fakültesi, Kontrol Mühendisliği
- **Yükseklisans** **:** Devam ediyor, İstanbul Teknik Üniversitesi, Kontrol ve Otomasyon Mühendisliği Anabilim Dalı, Kontrol ve Otomasyon Mühendisliği Programı

MESLEKİ DENEYİM VE ÖDÜLLER:

09/2010 – Staj (10 gün)

İstanbul Teknik Üniversitesi – İstanbul

Departman: Atölye

Yapılan Çalışma: AC/DC Güç Kaynağı Yapımı

07/2012 – Staj (30 gün)

ASELSAN - Ankara

Departman: Elektronik Tasarım

Yapılan Çalışma: Mikrokontrolör (DSP, FPGA) Yazılım Oluşturulması

09/2012 – Staj (20 gün)

SANKO A.Ş İsko Şubesi – İnegöl/Bursa

Departman: Elektrik Bakım Onarım

Yapılan Çalışma: PLC Yazılımı

İş (Yarı Zamanlı)

Baç Mühendislik Makina ve Otomasyon San. Tic. Ltd. Şti. – Sarıyer/İstanbul

Pozisyon: Kontrol Mühendisi

Giriş: 10/2012 - 06/2013

İş (Tam Zamanlı)

Baç Mühendislik Makina ve Otomasyon San. Tic. Ltd. Şti. – Sarıyer/İstanbul

Pozisyon: Kontrol Mühendisi

Giriş: 06/2013 - …..