BAŞKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

OPERATÖR TARAFINDAN KONTROL EDİLEBİLEN KUVVET GERİ BESLENİMLİ GEZGİN BİR ROBOT TASARIMI

EFE ANIL AKSÖZ

YÜKSEK LİSANS TEZİ
2011

OPERATÖR TARAFINDAN KONTROL EDİLEBİLEN KUVVET GERİ BESLENİMLİ GEZGİN BİR ROBOT TASARIMI

DESIGN OF A TELEOPERATED MOBILE ROBOT WITH FORCE FEEDBACK MECHANISM

EFE ANIL AKSÖZ

Başkent Üniversitesi
Lisansüstü Eğitim Öğretim ve Sınav Yönetmeliğinin
MAKİNA Mühendisliği Anabilim Dalı İçin Öngördüğü
YÜKSEK LİSANS TEZİ
olarak hazırlanmıştır.

2011

Fen Bilimleri Enstitüsü Müdürlüğü'ne,

Bu çalışma, jürimiz tarafından **MAKİNE MÜHENDİSLİĞİ ANABİLİM Dalı'nda YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Başkan :

Prof. Dr. Faruk Elaldı

Üye (Danışman) :

Yrd. Doç. Dr. Bedi Cenk Balçık

Üye :

Yrd. Doç. Dr. Mustafa Doğan

ONAY

Bu tez 18/02/2011 tarihinde, yukarıdaki jüri üyeleri tarafından kabul edilmiştir.

..../..../.....

Prof.Dr. Emin AKATA

FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRÜ

TEŞEKKÜR

Çalışma süresince yardımları ve verdiği bilgiler sayesinde tezin gelişiminde sonsuz bir katkısı olan ve zorlukların aşılmasında her zaman yardımcı olan Sayın ANDAÇ TÖRE ŞAMİLOĞLU ’na,

Çalışmanın yönlenmesinde karşılaşılacak olası hataların önceden tespiti ile yadsınamaz bir yardımı dokunan Sayın YRD. DOÇ. DR. B. CENK BALÇIK ’a,

Değerli yorumları ve her alandaki desteği ile her zaman yol gösterici olan Sayın PROF. DR. FARUK ELALDI ’ya,

Karşılaştığım problemlerin çözümünde deneyimlerinden yararlandığım Sayın YRD. DOÇ. DR. MUSTAFA DOĞAN’a

Ayrıca Desteklerini benden hiç esirgemeyip her zaman yanımda olan Günsu Özkarar’a ve maddi manevi yardımlarından dolayı Aileme,

İçtenlikle teşekkürlerimi sunarım…

ÖZ

OPERATÖR TARAFINDAN KONTROL EDİLEBİLEN KUVVET GERİ BESLENİMLİ GEZGİN BİR ROBOT TASARIMI

Efe Anıl AKSÖZ
Başkent Üniversitesi Fen Bilimleri Enstitüsü
Makine Mühendisliği Anabilim Dalı

Bu tez çalışması, kuvvet geri besleme mekanizmasına sahip, operatör tarafından kontrol edilebilen bir robotun tasarımı ve üretimini amaçlamaktadır. 5 serbestlik dereceli Master ve 4 serbestlik derecesine sahip Slave robot kolu tasarlanmıştır. Böyle bir kontrol için yüksek miktarlarda harcamalara ihtiyaç duyulduğundan yeni bir yöntem geliştirmek projenin ana amacı olmuştur. Bu alandaki çalışmalardan farklı olarak adım motorlar hareket mekanizması olarak kullanılmıştır. Motorların adım hareketleri yeni geliştiren algoritmanın temeli olarak alınmıştır. Pozisyon ve Kuvvet geribesleme kontrolü için Slave'den ölçülen açı değerleri Master koluna gönderilmiştir. Yeni tasarlanmış algoritma ile Slave kolundaki direnç kuvvetleri Master kolundan hissedilmiştir.

ANAHTAR SÖZCÜKLER: Kuvvet Geribesleme, Master – Slave Kontrol, Gezgin robotlar, Robot Kolları. Paralel Kontrol.

DANIŞMAN: Yrd. Doç. Dr. Bedi Cenk Balçık, Başkent Üniversitesi, Makine Mühendisliği Bölümü.

ABSTRACT

DESIGN OF A TELEOPERATED MOBILE ROBOT WITH FORCE FEEDBACK MECHANISM

Efe Anıl AKSÖZ
Baskent University Institute of Science
Department of Mechanical Engineering

This research work presents the design and the manufacture of a teleoperated mobile robot with force feedback mechanism. A 5 DOF “Master” and a 4 DOF “Slave” robot arm were designed. Due to high expenses are required to develop a system to control these kinds of mechanisms, designing an alternative method is the main goal of the project. Unlikely other applications on force feedback mechanisms, stepper motors are used as actuators. The step motion of the motors are taken as the basics of the new genarated algorithm. The measured angle values are sent from Slave to the Master robot arm for position and force feedback control. The resistive forces in slave arm can be sensed by the master arm with a new proposed algorithm.

KEYWORDS: Force Feedback Mechanism, Master – Slave Manipulation, Mobile Robots, Robot arms, Parallel Control.

ADVISOR: Asst. Prof. Bedi Cenk Balçık, Baskent University, Mechanical Engineering Departmant.

İÇİNDEKİLER

Sayfa

ŞEKİLLER LİSTESİ

ÇİZELGELER LİSTESİ

1. GİRİŞ

Robot teknolojisinin gelişmesiyle bu alandaki uygulamalar gittikçe genişlemeye başlamıştır. Artık robotlar sadece büyük montaj işler için değil, açık ve kapalı alanlardaki daha karmaşık görevleri yerine getirebilmektedir. Bir robotun durmaksızın gerçek zamanlı bir otonom hareketi için çevreden olabildiğince bilgi alınması gerekmektedir. Bunun içinde çeşitli sensörler kullanılmaktadır.

Robotlar insanoğlunun sonsuz hayal gücünün ve bilgisinin teknolojik bir ürünüdür. Robotik otomasyon teknolojisi çok geniş bir potansiyel markete sahip olmuştur. 1960'lardan beri robotlar işgücü yoğunluğunu azaltıp, prodüktiviteyi arttıran bir endüstriyel araç olmaya devam etmektedir. Ancak yüksek doğruluk ve hız oranına rağmen robotların konumlarının sabit oluşu uygulama alanlarını daraltmaktadır. Bu nedenle 1980'lerde birçok ülke gezgin robot teknolojisi alanında araştırma birimleri oluşturmaya başladı [25].

Günümüzde otonom gezgin robotların kullanımı günden güne artmaktadır. Gezgin Robotlar insanların ulaşamayacağı lağım, dar borular, ya da insanlar için tehlikeli bölgelerde, tıbbi veya nükleer atıkların toplanması gibi islerde insanların yerini alması için kullanılmaktadır. Bu robotlar demir çelik gibi ağır sanayide kullanılan hammaddeleri taşımak ya da yarı mamul ürünlerin yer değiştirmesi sırasında bu ürünlerin teslimatında kullanılabilirler [21].

2000'li yıllardan beri gezgin robotlar bomba tespit, kaldırma, taşıma ve imha etme görevlerinde insanların yerini aldı. Bulunduğu konumun şartlarından dolayı bombanın etkisiz hale getirilmesi zor olan durumlarda veya patlatılması çevreye ve insanlara zarar verme ihtimali taşıyan durumlarda operatör tarafından kontrol edilebilen robotlar devreye girmektedir. Operatörün kontrolünü daha verimli kılabilmek için görüntü sistemleri ile geliştirilmesi ve robot kolunda kuvvet geri besleme yapılması konularında birçok çalışma yapılmıştır. Yapılan bu çalışmalar sayesinde operatörün derinlik kavramı ve hissiyatı arttırılmış böylece hata riskleri ortadan mümkün olduğunca kaldırılmıştır. Master – Slave robot kolu kullanarak operatör gezgin robotun üstünde bulunan kolun bir eşiyle çalışmakta ve böylece

duruma daha fazla hakim olabilmektedir. Haptik sistemlerin de entegre edilmesiyle tam anlamıyla senaryo operatöre yaşatılabilmektedir. Bu eklentiler beraberinde yüksek miktarlarda bir maliyet getirmektedir. Haptik sistemler ve ileri görüntü sistemleri tehlikeli bölgelerde çalışan gezgin bir robota entegre edilmekte ve görev sonunda genellikle bu sistemlerin mekanik ve elektronik parçalarında deformasyon meydana gelmektedir. Bu parçaların değişimi veya tamiri ise her görevde maddi zararlara yol açmaktadır [11].

1.1. Robotların Tarihsel Gelişimi

Robotların gelişim tarihi 3 kuşakta incelenebilir. Birinci kuşak robotlar öğrenen robotlardır. Gerek operatör tarafından verilen komutlar, gerekse kontrolcü tarafından yönlendirilen bir robot kolunun hareketleri robotun hafıza elemanları tarafından saklanır. Çalışma esnasında alınan veriler ile hareket tekrarlanarak gerçekleşmeye başlar. Bu çeşit robotlar çevreden bilgi alamazlar. Bu geribeslemesin olmayışı robotun değişken çevresel etkilere adapte olmasını zorlaştırır.

İkinci kuşak robotlar ise kendi duyuları olan, dış çevredeki değişkenlere uyum sağlayabilen, duyma görme ve hissetme özellikleriyle seçim yapabilen robot türleridir. Çalışma esnasında sensörlerden alınan bilgilerin yardımıyla konumunu akıllıca ayarlayabilen bu robotlar ve çevresel değişimlerden etkilenmez. Alınan kararlar tamamen sensör bilgilerinin önceden oluşturulmuş bir algoritmaya göre yorumlanmasıyla elde edilir.

Üçüncü kuşak robotlar ise akıllı robotlardır. İkinci kuşakta olduğu gibi birçok sensörler ve ölçüm cihazları ile çevreden bilgiler alınır. Bu veriler yapay zekâ teknolojisi ile robotun bağımsız düşünme, öğrenme, tanıma akıl yürütme ve yargılama gibi ileri düzey özelliklerini geliştirir[25].

1.2. Gezgin Robotlar

Gezgin robotların hem gerek özel gerekse endüstriyel alanlarda kullanım sayısı gün geçtikçe artmaktadır. 1960larda kullanılan endüstriyel robotlar artık en bilinen robot türleri olmaktan çıkmıştır. Bunu günümüzdeki tüketici uygulamalarıyla da görüleceği gibi artık en yaygın kullanılan robotlar gezgin robotlar olmaya başlamıştır[19].

Gezgin robot kullanımındaki bu artışın nedeni, yansıtılan imajın orijinal ve lüks görünümden sıyrılıp, kullanımı kolay, pratik ve insan gücünü azaltan yapılara dönüşmüş olmasındır. İnsan gücünü ortadan kaldırmaya verilebilecek en güzel örneklerden biri iRobot ve Electrolux firmaların ürettiği gezgin elektrik süpürge robotudur. 2004 sonlarında UNECE'e (United Nation Economic Commissiin for Europe Survey) göre 1,2 Milyon gezgin robot özel amaçlar için kullanılmaktaydı ve bu sayının büyük bir kısmını elektrikli süpürge robotları oluşturmakta. 2004 yılı bu konudaki gelişmenin en çok görüldüğü yıl olmuştur. Bu yıl içersinde 550,000 robot üretilmiştir.

Gezgin robotların bu kadar fazla sayıda kullanılmakta olmasına rağmen, kapasiteleri limitlidir. Çoğu temizlik robotları, tekerlekli yapısıyla, önceden programlanmış davranışlarla sınırlı olmakla beraber yol bulma algoritmaların yardımıyla karşılaştığı engellere çarpmadan hareket eder. Kullanılan bu basit navigasyon metotları büyük sayıdaki problemlerin çözümü olsa da, ileri düzeydeki daha karmaşık navigasyon metotları kullanmadan yapılmak istenilen çözümler sınırlıdır. Örnek olarak kutu kanyonlar (Bkz: Şekil 1) adı verilen girişin ve çıkışın aynı alandan olan geometrilerde kullanılan bu yöntemler sonuçsuz kalmaktadır.

Şekil 1 – Kutu Kanyonlar (Box Canyons) [16]

Kişisel kullanım için piyasaya sürülecek robotların fiyatı çok önem taşımaktadır. Bu alanda tüketicinin dikkat ettiği en önemli unsur maliyettir. Bu unsur, robotlar için

kullanılabilecek sensörleri de kısıtlamaktadır. Gezgin robot tasarımındaki amaç ise pahalı sensörlerle bir uygulama olmaktan çok karşılanabilir uygun sensörler kullanılarak, yazılımın ön plana çıktığı ürünler üretmek olmaktadır.

Gezgin robotlarda karşılaşılan en büyük zorluklardan biri ise robotun karar verme yapısıdır. Algoritmalar ve önceden tanımlanan hareketler ile robotlar sınırlı görevleri gerçekleştirebilmektedir. Bu nedenle navigasyon sistemleri yerine daha ileri sistemler kullanarak robotun çevresini haritasını çıkarmasını sağlamak olmalıdır. Bu sayede daha zeki ve çevresine uyum sağlayarak verilen görevi daha başarılı bir biçimde gerçekleştirebilen robotlar yapılabilir [14].

Gezgin robotlar farklı çeşitlerde bulunmaktadır. Manüel olarak yada uzaktan kontrol edilebilen robotlar yönetme kolu ve ya benzeri kontrol üniteleri ile hareket ettirilebilen robotlardır. Kontrol üniteleri robota direk olarak bağlanabilir veya kontrol kablosuz iletişim kurabilen bir kontrol ünitesi ya da dizüstü bilgisayar yardımıyla yapılabilir. Uzaktan kontrol tamamen operatörün tehlikeden mümkün olduğunca uzak tutma amacıyla kullanılmaktadır. ANATROLLER ARI-100 ve ARI-50 (Robotics Design), Talon (Foster-Miller), PackBot (iRobot), and MK-705 Roosterbot (KumoTek) robotları bu tip robotlara örnek olarak gösterilebilir.

1.3. Bomba İmha(EOD/IEDD) Robotları

Bomba imha sırasındaki operatörün baskısını azaltıp, güvenliği ve verimliliği arttırmak çözülmesi gereken ana problemdir. Bazı araştırmacılar otonom kontrolün en önemli kısmını oluşturan görüş planlama yani robotun çevresini araştırma ve haritasını çıkarma alanına yoğunlaşmıştır [13]. Kullanılan mesafe ölçerlerden alınan bilgilere göre robotun dolaştığı çevrenin olabildiğince ayrıntılı bir haritası çıkartılır. Harita çıkartılırken odometredeki gürültü ve sensörlerdeki gürültü aşılması gereken 2 önemli problemdir. Literatürde bu problem "*Simultaneous Localization and Mapping Problem (SLAM)"* olarak adlandırılır.

Günümüzde değişik konfigürasyonda ve teknolojide üretilmiş birçok Bomba İmha Robotu bulunmaktadır. (Bkz: Tablo 1) Bomba imha görevi için kullanılan birçok aksesuar da mevcuttur. Bu aksesuarlar ve robotlar yardımıyla insan hayatını

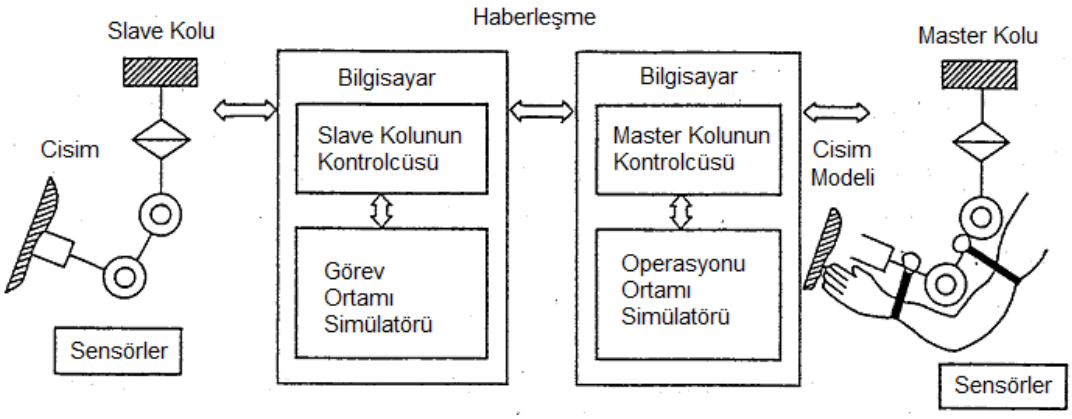
tehlikeye atmadan olası bir tehlike taşıyan bir cisim güvenli bir bölüme alınabilmekte veya imha edilebilmektedir.

Üretici	Robotun Adı	Ağılık(kg) Uzunluk - Genişlik (mm)	Hız(m/dk), Kontrol
AB Precision Ltd.	Cyclops L.E.	30, 875 - 397	27, Kablo
Allen Vanguard	Defender	250, 1470 - 700	53, RF veya Kablo
VMRobots	MR5	200, 1300 - 800	50, RF veya Kablo
QinetiQ	Ground Hog	40, 870 - 535	133, RF veya Kablo
Telerob	Teodor	380, 1302 - 680	50, RF veya Kablo
Remotec	Wolverine	299, 1452 - 702	45, RF veya Kablo
Yujin Robotic	Robhaz	145 , 690 - 910	166, RF veya Kablo

Tablo 1 – Bazı Bomba İmha Robotları

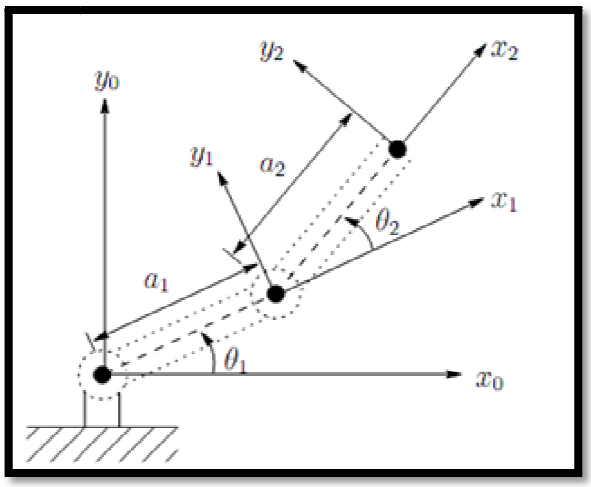
1.4. Master - Slave Kontrol

Gezgin robotların üzerinde bulunan robot kollarının kontrolü veya önemli bir parçanın montajı için kullanılan robot kollarının kontrolü için geliştirilen yöntemlerden biri ise Master – Slave kontroldür. Bu yöntemde robot kolunun hareketini sağlamak için operatör hareket ettirilmesi istenilen kol ile aynı konfigürasyondaki başka bir kolu eli ile yönlendirmesi ile gerçekleşir. Kullanılan bu yöntem ile operatör duruma daha çok hâkim olur ve hata yapma oranı daha azalır. Şekil 2'de Master Slave kontrol konseptine bir örnek görülebilmektedir.



Şekil 2 – Master - Slave Kontrol Konsepti [22]

Bu kontrolde robot kolun tutucusunun pozisyonu ve eklemlerin açılarını bulmak için ters kinematik denklemlerden ve düz kinematik denklemlerden yararlanılır. Master kolunun operatör tarafından hareket ettirilip belli bir konuma getirildikten sonra, her eklemde bulunan ayarlı dirençler veya enkoderlerden elde edilen açılar yardımıyla tutucunun koordinatları bulunabilir. Eklem açıları bilinen robot kolunun tutucusunun koordinatlarını bulmak için düz kinematik denklemler kullanılır. Şekil 3'de 2 boyutlu 2 serbestlik dereceli robot kolu ve düz kinematik denklemler görülebilmektedir.



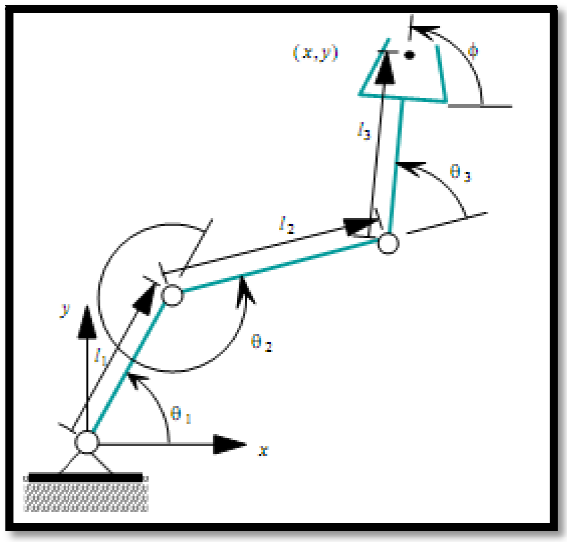
Şekil 3 – 2 serbestlik dereceli robot kolu [20]

$$x_2 = a_1 \times \cos(\theta_1) + a_2 \times \cos(\theta_1 + \theta_2) \quad (1.1)$$

$$y_2 = a_1 \times \sin(\theta_1) + a_2 \times \sin(\theta_1 + \theta_2) \quad (1.2)$$

$$\theta_{kol} = \theta_1 + \theta_2 \quad (1.3)$$

Ters kinematik denklemler ise koordinatları bilinen tutucuya sahip kolun eklem açılarını bulmak için kullanılırlar. Master kolunun hareketi sonucu tutucunun yeni koordinatları Slave koluna gönderilir. Slave kolu alınan bu koordinatlara gitmek için eklemlerinin açılarına ihtiyacı vardır. Ters kinematik denklemler ile bu açıları bulunabilmektedirler. Şekil 4'de 2 boyutlu 3 serbestlik dereceli robot kolu ve ters kinematik denklemler görülebilmektedir.



Şekil 4 – Ters Kinematik Denklemler [20]

$$\theta_1 = \gamma + \cos^{-1}\left(\frac{-(x'^2 + y'^2 + l_1^2 + l_2^2}{2l_1\sqrt{x'^2 + y'^2}}\right) \quad (1.4)$$

$$\theta_2 = a\tan 2\left(\frac{y' - l_1\sin\theta_1}{l_2}, \frac{x' - l_1\cos\theta_1}{l_2}\right) - \theta_1 \quad (1.5)$$

$$x' = x - l_3\cos\phi \quad (1.6)$$

$$y' = y - l_3\sin\phi \quad (1.7)$$

$$\theta_3 = \phi - (\theta_1 + \theta_2) \quad (1.8)$$

Master – Slave kontrolünde kinematik denklemlerin kullanılması sonucu bazı istenmeyen sonuçlar elde edilebilir. Bunun nedeni istenilen bir koordinata ulaşmak için birden fazla çözümün olmasıdır. Bu nedenle Master – Slave kontrolünde daha çok açılar ve belli bir zaman aralığındaki açısal değişimler kullanılarak birbirleri ile paralel çalışan 2 adet robot kol yapılabilir. Bu kollar daha sonra bir görevde kullanıldığında operatörün Master kolunu götürdüğü noktanın yanı sıra hareket sırasında çizdiği rota da önem kazanır.

1.5. Robot Kolları Ve Kuvvet Geribeslemesi

Gezgin robotların en önemli özelliği çevreyle olan ilişkisidir. Bu ilişki robot kollarıyla ve bu kolların üstünde bulunan kıskaçlarla sağlanır. Operatör bir kumanda yardımıyla robot kolunu hareket ettirir ve yapılması istenilen görev gerçekleştirilir. Verilen görev bomba imha robotlarında olduğu gibi bir cismin uzaklaştırılması, zararlı olan bir maddenin bulunduğu kapağın açılması gibi insan sağlığına zararlı veya tehlikeli görevler olabilir. Bu gibi görevlerde operatöre çok büyük sorumluluk düşmektedir. Operatörler görevi yerine getirebilmek için çalışma alanından uzakta olsa bile duruma hâkim olmalıdır. Bunun için robot kollarında kuvvet geribeslemesi alanında birçok çalışma bulunmaktadır.

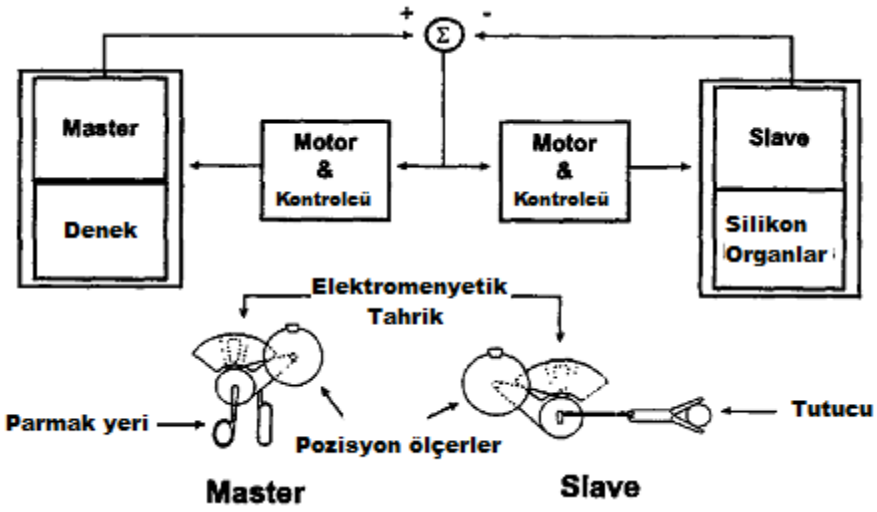
Görev alanındaki Slave kolunun maruz kaldığı kuvvetleri Master koluna ve operatöre hissettirmek için kullanılan yöntemlerden biri slave koluna kuvvet sensörleri takmak ve bu sensörlerden alınan veriler sayesinde kontrolü sağlamaktır. Kuvvet/Tork sensörlerinin bağlı bulundukları eleman bir kuvvete maruz kaldığında, elemanı oluşturan malzeme üzerinde gerilmeler meydana gelir. Bu gerilmeler yük hücreleri tarafından ölçülür. Elde edilen gerilme değerleri ile kuvvetler hesaplanabilmektedir. Piyasada 6 eksenli kuvvet/tork sensörleri mevcuttur. Bu sensörler F_x, F_y, F_z, T_x, T_y, T_z değerlerini ölçebilmektedir. Şekil 5'de farklı boyutlardaki 6 eksenli kuvvet/tork sensörleri görülebilmektedir.

Şekil 5 – 6 Eksenli Kuvvet/Tork Sensörleri [2]

Jean-Pierre MERLET'ın 2004 yılında yaptığı çalışmada[11] 2 adet paralel robot kolu kullanarak kuvvet geribeslemesi için düzenek kurmuştur. 6 adet bağlantı elemanı ile kıskaç ve taban birbirine bağlanmıştır. Bu mekanik tasarım ile kuvvet geri-beslenimi daha hassas sonuç verdiğinden dolayı bahsedilen çalışmada her bağlantı elemanı için kuvvet sensörleri kullanılmıştır. Lineer ayarlı direnç ise pozisyon kontrolü için kullanılmıştır. Sistemin pozisyon kontrolü, robot kolunun tabana göre yer değişimini temel alarak yapılmış ve INRIA adı verilen prototip tasarlanmıştır. Yapılan deneylerde robot kolunun montaj işlemlerindeki performansı incelenmiştir. Gezgin kol 1mm yanılmayla işlemi tekrarlayabilmiş ve pozisyon kontrolü ile birlikte kullanılan kuvvet kontrolünün verdiği sonuçlar kaydedilmiştir.

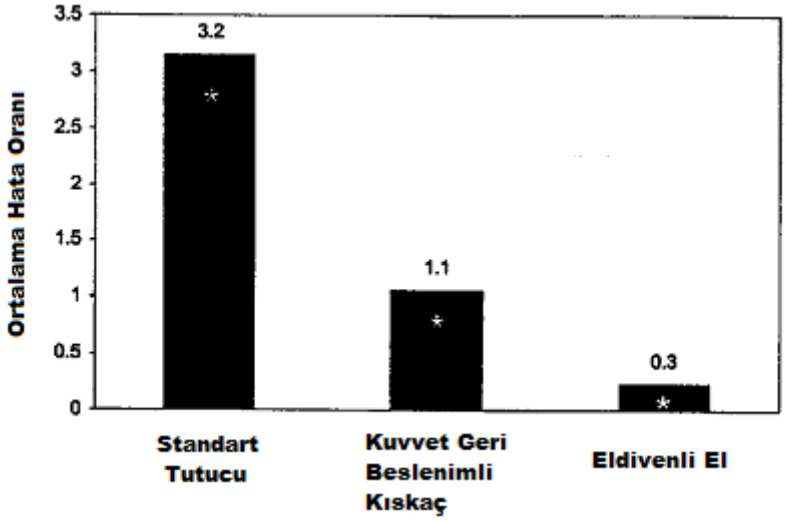
Kuvvet geribeslemesi için yapılan çalışmalarda genel olarak Haptik kollar kullanılmaktadır. Haptik kollar 6 serbest dereceli kuvvet ve tork sensörleri bulunan ağırlık kaldırmak için tasarlanmamış robot kollarıdır. Haptik, literatürde dokunuş teknolojisi olarak bilinir. Haptik sistemlerde kullanıcıya kuvvet, titreşim ve hareket geribeslemesi yaparak dokunma hissi verir.

Minimal invasif ameliyatlardaki kısıtlamalardan biri olan dokunma hissinin azalması konusunda yapılan çalışmada [12], kuvvet geri beslemeli kıskacın kullanılması ile operatöre dokunma hissi yaşatmak amaçlanmaktadır. Test amacıyla 10 denekten daha önceden hazırlanmış aynı boyutlarda fakat farklı esnekliğe sahip 6 adet silikon organ modelini en yumuşaktan en serte doğru sıralamaları istenilmiştir. Bu deney için denekler laboratuar eldivenli ellerini, ameliyatlarda kullanılan tutucu ve kuvvet geribeslenimli kıskaç kullanmışlardır. Sonuçlara göre, kuvvet geri beslenimli kıskaç standart tutucudan daha iyi sonuçlar vermiş ama el ile yapılan sıralama kadar başarılı olamamıştır.

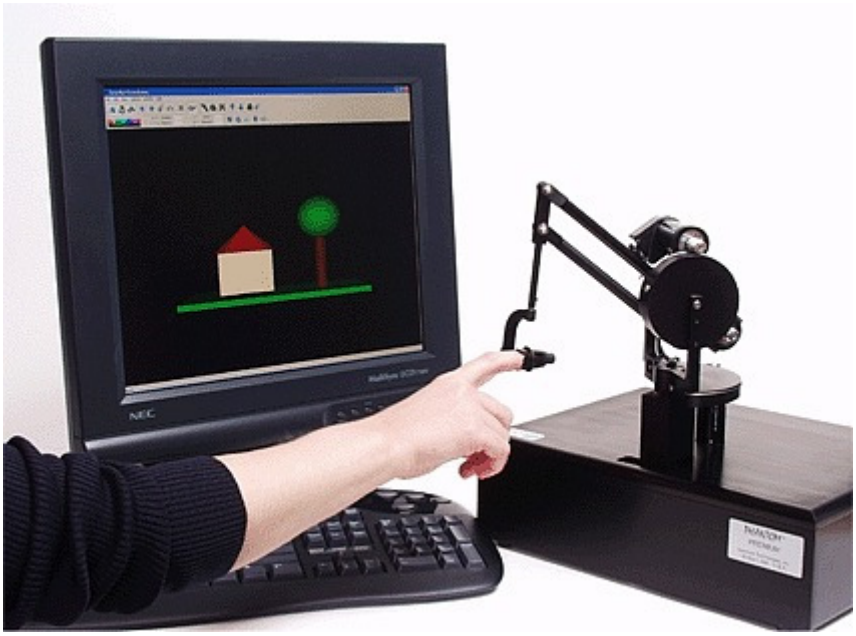


Şekil 6 – Master Slave Kontrolcü düzeneği [12]

Şekil 6'da görülen kontrolcü düzeneği Slave robot kolunun Silikon organlar ile yaptığı temas sonucu değişen pozisyonları Master koluna gönderir. Aradaki hata oranına göre Haptik kol operatöre dokunma hissini yaşatacak direnç kuvvetini uygular. Şekil 7'de deney sonuçlarından elde edilen hata değerlerini karşılaştıran grafik görülmektedir. Bu sonuçlara göre kuvvet geribeslemesin standart tutucuya göre daha verimli olduğu görülmektedir.

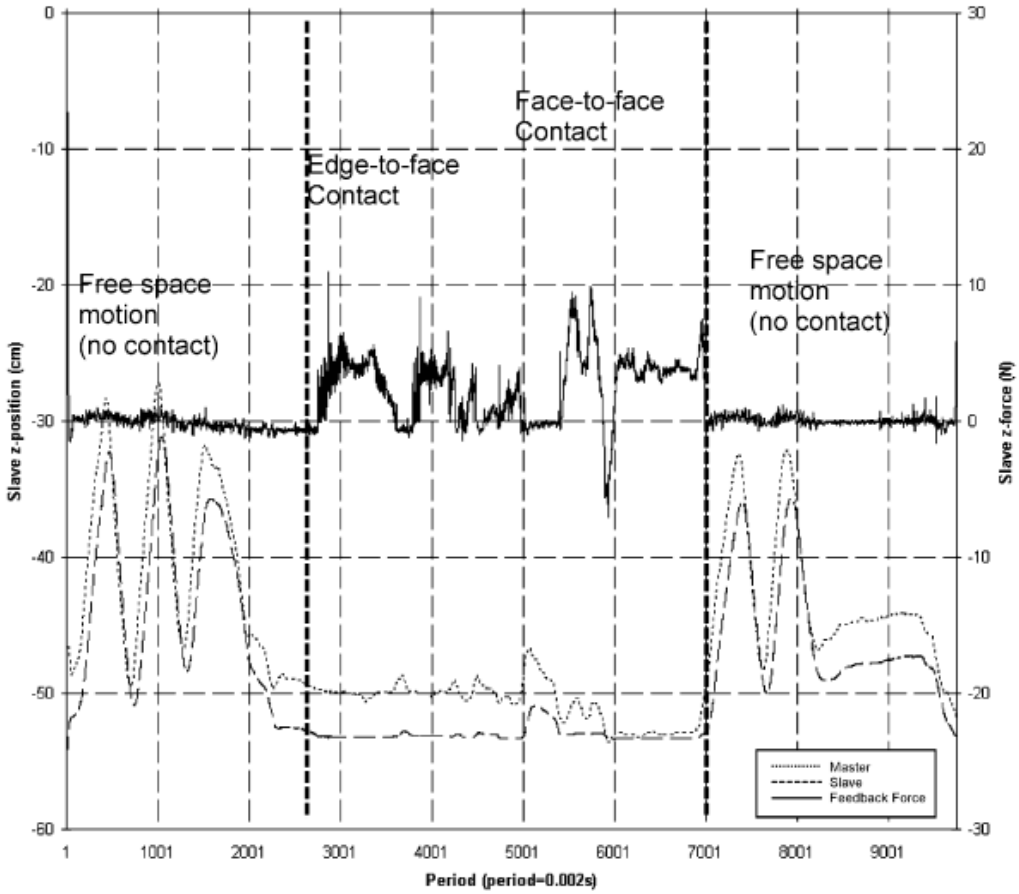


Şekil 7 – Ortalama Hata Oranları [12]



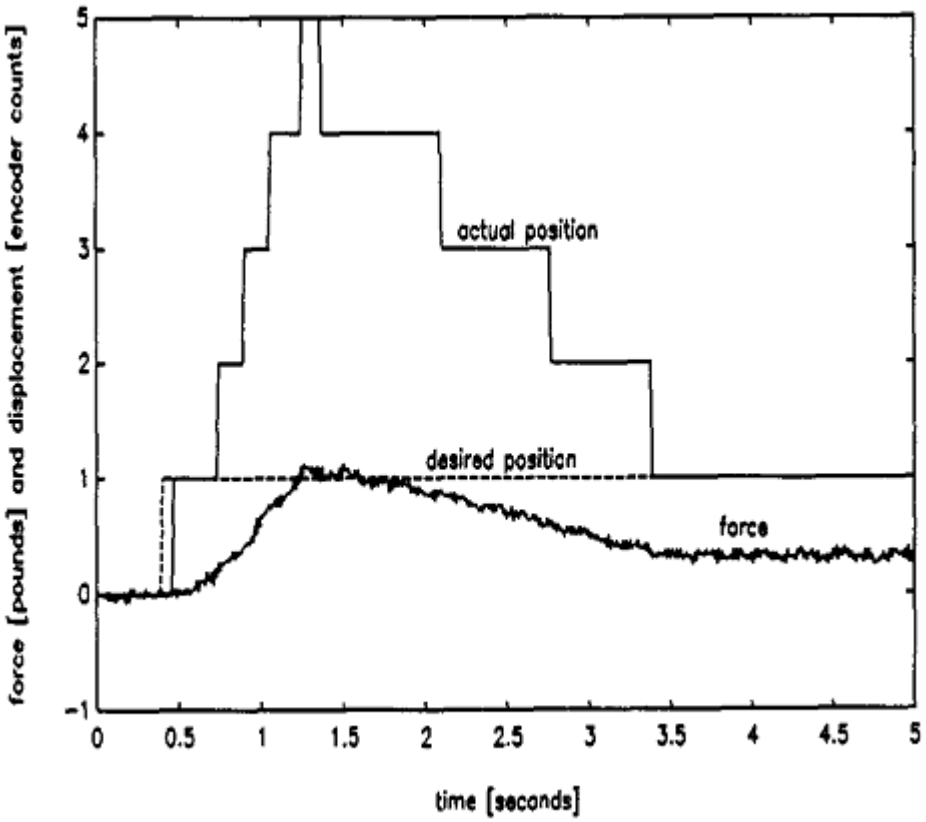
Şekil 8 – *Phantom* Haptik kol [24]

H.Y.K. Lau ve L.C.C. Wai yaptığı çalışmada [7] Master kolu olarak 6 serbestlik dereceli PHANTOM Haptik kolu (Bkz: Şekil 8) ve slave olarak 4 serbestlik dereceli WAM robot kolu kullanılmıştır. 2 adet farklı denetim algoritmasında yapılan çalışmada sonuçlar karşılaştırılmıştır. Pozisyon – Kuvvet kontrolünde Slave kolu pozisyon kontrolü ile kontrol edilirken eklemlerinde bulunan tork ölçerlerden elde edilen değerlerden türetilen kuvvet değerleri Master kolunun kuvvet kontrolünde kullanılmıştır. Sonuç olarak kuvvet geribeslemesi operatöre Haptik koldan verilmiştir. Pozisyon – Pozisyon kontrolünde birbirleri ile aynı mekanik tasarıma sahip olan 2 robot kolu da pozisyon kontrolü ile kontrol edilmiştir. Pozisyon farklarına göre geri beslenim uygulanmıştır. Şekil 9'daki grafikte Master ve Slave kolunun pozisyonları ve temas anındaki kuvvetler verilmiştir.



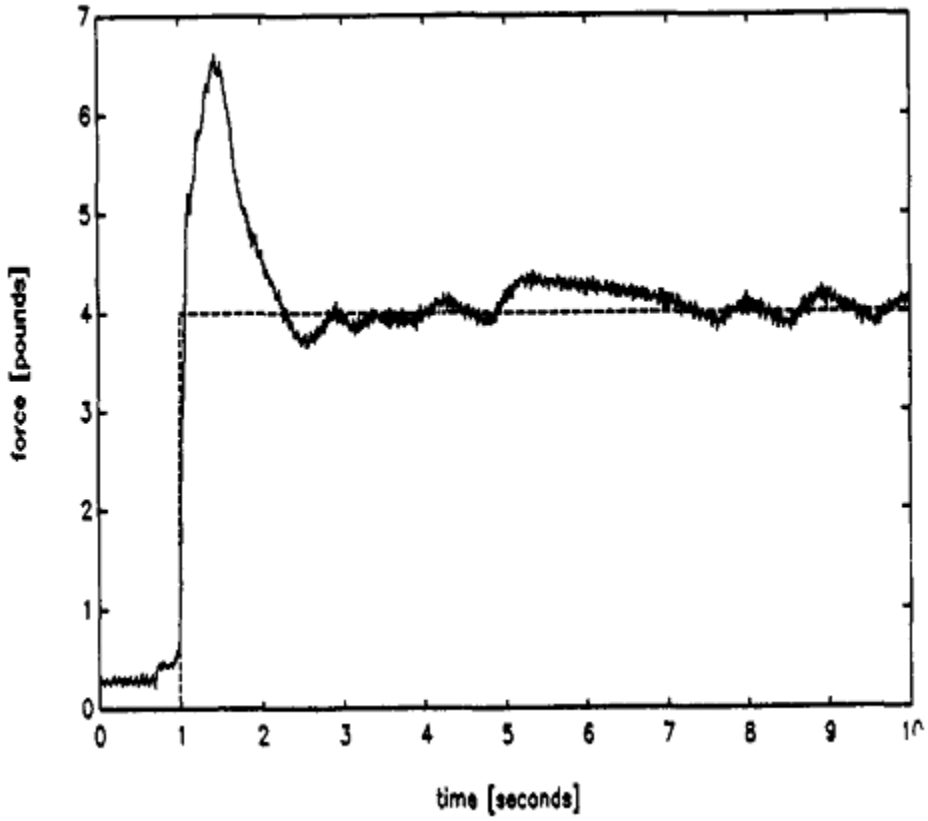
Şekil 9 – Master ve Slave kolunun Kartezyen rotaları ve temas kuvveti [7]

J. G. Hollinger, R. A. Bergstrom, ve J. S. Bay'ın yaptığı çalışmada[8] adım motorların kullanıldığı robot kolu üzerinde robotik uygulamalarda "sert kontak" sorununu çözmek için bulanık mantık denetleyici ile kuvvet kontrolü yapılmıştır. Sert bir yapıya sahip robot kolu esnekliği çok az olan bir cisim ile temas ettiği anda kuvvet kontrolünün gerçekleşmesi ve geribesleme ile sorunun çözülmesi hedeflenmiştir. Bulanık mantık kuvvet kontrolü MERLIN 6540 endüstriyel robot üzerine uygulanmış ve bu analiz için gerekeli mekanik modifikasyonlara gidilmiştir. Eklem pozisyonu kullanarak adım motorunun en küçük hareketi için kuvvet geribesleme cevapları elde edilmiştir. Kuvvet kontrolü 6 serbestlik derecesine sahip MERLIN 6540 robot koluna uygulanan eklem pozisyon kontrol yöntemi için optik enkoderler kullanılmıştır. Bir dönüş sırasında 0.0075 derecelik açı okuyabilen bu sistem sayesinde 0.127 mm bir hassasiyet ile ölçümler her 4ms'de bir alınabilmiştir. Sürtünme kuvvetini azaltmak için kuvvet tork sensörüne bilyalı bir sistem eklenmiştir. Bu sayede sürtünmeden kaynaklanan kuvvetler ortadan kaldırılmıştır.



Şekil 10 – Kuvvet ve pozisyon hatası [8]

Şekil 10'daki grafikte bir hareket esnasında sistemin açık döngü cevabı görülebilmektedir. 0.5 saniye ile 1 saniye arasındaki gerçek pozisyon ve istenilen pozisyon arasındaki hata oranının artması ile geri beslemesi yapılan kuvvet değeri arttırılmıştır. Bu arttırılan kuvvet değerini kontrol etmek için bulanık mantık denetleyicisi kullanılmıştır. Geliştirilen kuvvet yöntemi ile bir adımlık hareket için bulanık mantık denetleyicisinin cevabı Şekil 11'deki grafikte verilmiştir.



Şekil 11 – Bulanık Mantık denetleyicisinin bir adıma karşı cevabı [8]

1.6. ÇALIŞMANIN AMAÇLARI

Tehlikeli bölgelerde insanların yerini ajan ve bomba imha robotlarının alması için birçok çalışmalar yapılmaktadır. Robotu kullanan operatör kullanma ekipmanında bulunan düğmelerle tüm eklemlerin yapabileceği hareketi sağlamaya çalışır. Bu durum ise operatör için çok zor ve hataya açıktır. Operatörün robotun bulunduğu durumu daha iyi kavraması ve robot kolunu istenilen noktaya getirebilmesi ve gereken hareketlere daha iyi karar verebilmesi için düğmelerden daha etkili bir sistem kullanılması gerekmektedir [13].

En yakın teknolojide manüel kontrolü güçlendirmek için Haptik sistemler kullanılmaktadır. Fiyatları 2500$'dan başlayan bu teknoloji ile operatör dokunma hissi aktarılır. Bu sayede 2 boyutlu görüşün doğuracağı hatalar azaltılabilmektedir. Kullanılan bu teknoloji kuvvet geri beslemeli olarak çalışmaktadır.

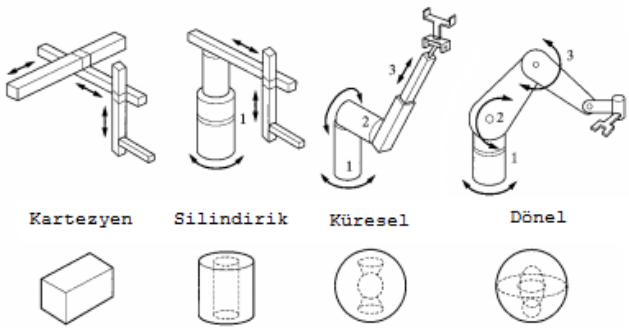
Bu çalışmanın amacı kuvvet geri beslemeli iletişim kurabilen master - slave robot kontrolü ile çalışabilen adım motorları ile çalışan robot kolları tasarlamaktır. Bu sayede Haptik sistemlerin çalışma prensibine çok yakın ve daha az maliyetli bir sistem geliştirmek istenmektedir. Bu sistemin performansına göre uygun algoritmaların ortaya çıkarılması amaçlanmaktadır. Ayrıca Haptik sistemlerin gezgin robot üzerindeki robot kolunun konfigürasyonuna benzememesinden doğabilecek olası hataları en aza indirgemeyi hedefleyen bu çalışma, yazılımı ön plana çıkarmayı hedeflemektedir.

Robot kol düzeneği ile operatör tarafından kullanılan master kolunun hareketini taklit eden slave kolu çevre ile etkileşime girmesi durumunda master kolu bu kez slave kolunu taklit etmesiyle operatör kuvvetleri hissedebilecektir.

2. ROBOT KOLUNUN MEKANİK TASARIMI

Tasarlanacak Robot kolları operatör tarafından kullanılacağı için mekanik tasarım yetişkin bir insan kolunun ölçülerine göre yapılmıştır. İnsan kolunun ortalama uzunluğu 63.5cm (25 in) olarak verilmiştir [10].Bu değeri aşmayacak bir robot kolu tasarlanması gerektiğinden kullanılan elemanların ölçülerini belirlemede referans değer olarak 60 cm kullanılmıştır.

Robot kolunun konfigürasyonu en fazla çalışma alanı kapasitesine göre seçilmiştir. 4 çeşit robot kolu konfigürasyonu mevcuttur. Kartezyen, Silindirik, Küresel ve Dönel yapıdaki robot kolları ve çalışma alanları Şekil 12'de görülebilmektedir.

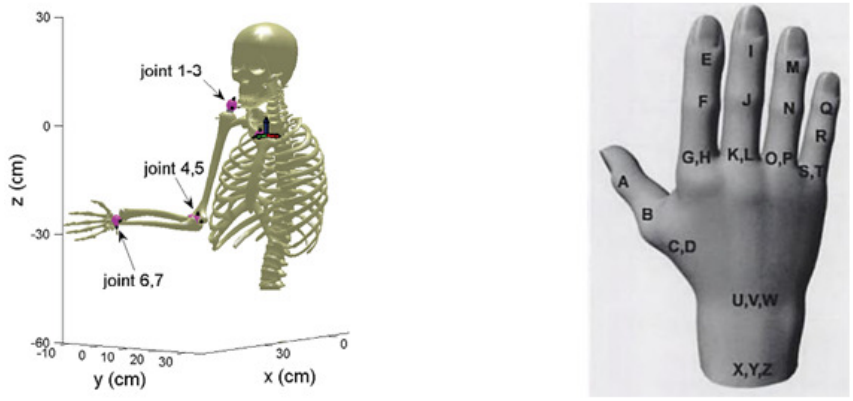


Şekil 12 – Robot kollarının Konfigürasyonları

Projede gezgin robotlarda çokça kullanılan dönel konfigürasyonlu robot kolunun tasarlanması uygun görülmüştür. Bu sayede maksimum çalışma alanı ve kolay

montaj imkânı sağlanmıştır. Çalışma alanı dışında robot kollarının serbestlik derecesi verimlilik açısından çok büyük önem taşır. Serbestlik derecesi (DOF – Degree of Freedom) bir cihazın yapabildiği bağımsız hareketlere verilen addır. 3 boyutlu uzayda 6 serbestlik derecesi mevcuttur. 3 adet öteleme (x,y,z) ve 3 adet dönme (raw, pitch, yaw).

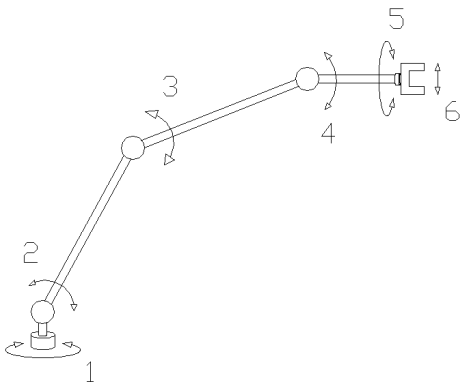
İnsan kolu 7, elimiz ise bilek hareketi dâhil olmak üzere 25 serbestlik dercesine sahiptir. Şekil 13'de insan kolunun ve elinin yapabileceği bağımsız hareketler görülebilmektedir.



(a) (b)

Şekil 13 – (a) İnsan kolunun [9] ve (b) insan elinin serbestlik dereceleri [4].

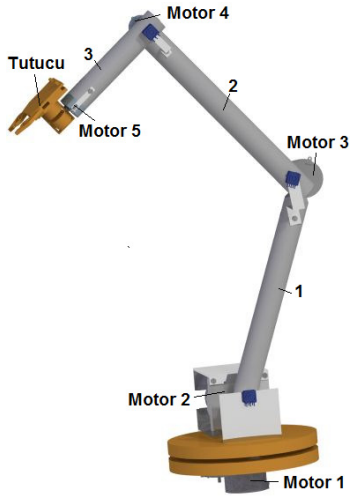
Robot kolları insan koluna çok benzer yapıda olabilir. 7 serbestlik dereceli robot kolları mevcuttur. Robot kollarında serbestlik dereceleri, bağlantı elemanları ve motorların yapıları ile belli olmaktadır. Projede tasarlanan robot kolu dönel bağlantı elemanları kullanılmış ve böylece her bir eklemde 1 serbestlik derecesi elde edilmiştir. Böylece tasarlanan robot kolu kıskaç hareketi dâhil toplamda 6 serbestlik derecesine sahiptir. Şekil 14'de robot kolunun toplam serbestlik derecesi görülebilmektedir. Bu tasarım 6 serbestlik derecesi ile çalışma alanındaki istenilen hareketi gerçekleştirebilecektedir.



Şekil 14 – Tasarımın serbestlik derecesi

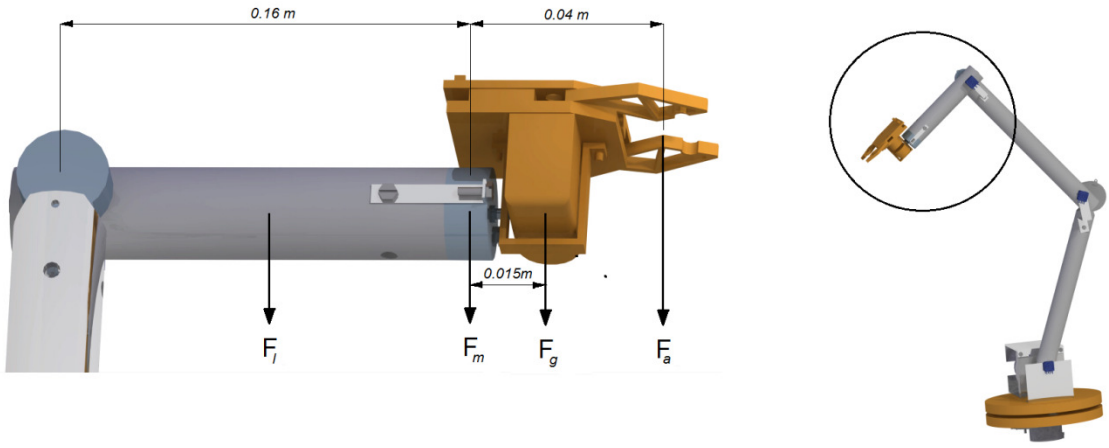
2.1. Tasarım Kriterleri

Seçilecek motorların kapasitesi referans boyutlara ve malzeme seçimlerine göre elde edilen ağırlıktan hesaplanan tork değerlerine bağlıdır. Başlangıç için bu değerler Şekil 15'de görülen 16cm boyutunda olan 3 numaralı bağlantı elemanına ve kaldırılacak ağırlığa göre hesaplanır.



Şekil 15 – Numaralandırılmış elemanlar

Şekil 15'de bağlantı elemanları gösterilmiş ve tork hesaplamaları bağlantı elemanlarının en son elemanı olan 3 numaralı elemandan başlanarak yapılmıştır.



Şekil 16 – 3 numaralı bağlantı elemanı ve tutucu boyutları

$$m_a = 2kg, m_m = 0.2kg, m_l = 0.0204kg, m_g = 0.35kg, l_a = 0.2m, l_m = 0.16m, l_l = 0.08m, l_g = 0.015m$$

$$T_3 = m_a \times l_a + m_m \times l_m + m_l \times l_l + m_g \times l \quad (2.1)$$

$$T_3 = 2kg \times 0.2m + 0.2kg \times 0.16m + 0.0204kg \times 0.08m + 0.35kg \times 0.015m \quad (2.2)$$

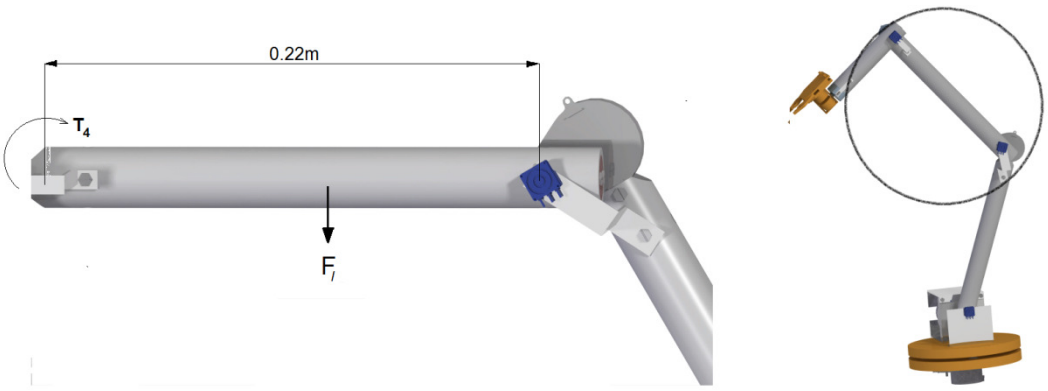
$$T_4 = 0.00022765kgm \quad (2.3)$$

$$T_4 = 0.00223Nm \quad (2.4)$$

$$T_4 = 0.223Ncm \quad (2.5)$$

Şekil 16'da görülen bağlantı elemanı boyutlarına ve kütlelerine göre harekete başlaması için gereken hesaplar denklem 2.1 ile 2.5 arasında verilmiştir. 4 numaralı motor 0.223 Ncm tork verebilecek kapasitede olması gerekmektedir. Motor robot kolunun en uç elemanı olacağı için ağırlığı en düşük motor seçilmelidir. Bu nedenle redüktörlü motora ihtiyaç duyulacaktır. Ancak redüktör oranının çok yüksek tutulması gücü arttırırken hız kaybına yol açacaktır.

Şekil 17'de görülen bağlantı elemanın boyutlarına ve kütlelerine göre 3 Numaralı motorun uygulaması gereken minimum tork değerlerinin için gereken hesaplamalar denklem 2.6 ile 2.10 arasında verilmiştir. Şekil 18'de görülen 1 numaralı motor için bulunan değerler kullanılarak yapılan hesaplamalar denklem 2.11 ile 2.15 arasında verilmiştir.



Şekil 17 – Numaralı bağlantı elemanı

$m_l = 0.0382923kg, l_l = 0.11m, T_4 = 0.00022765kgm$

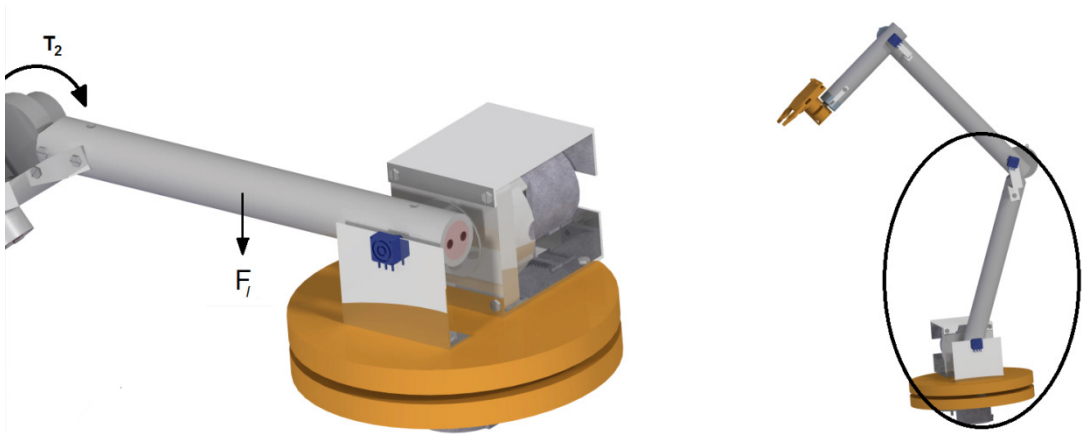
$$T_3 = T_4 + m_2 \times l_2 \quad (2.6)$$

$$T_3 = 0.0022765kgm + 0.038293 \times 0.11m \quad (2.7)$$

$$T_3 = 0.00648873kgm \quad (2.8)$$

$$T_3 = 0.0637Nm \quad (2.9)$$

$$T_3 = 6.37Ncm \quad (2.10)$$



Şekil 18 – 1 Numaralı bağlantı elemanı

$m_l = 0.0382923kg, l_l = 0.11m, T_3 = 0.00648873kgm$

$$T_2 = T_3 + m_1 \times l_1 \quad (2.11)$$

$$T_2 = 0.00648873kgm + 0.038293 \times 0.11m \quad (2.12)$$

$$T_2 = 0.010096kgm \quad (2.13)$$

$$T_2 = 0.105Nm \quad (2.14)$$

$$T_2 = 10.5Ncm \quad (2.15)$$

2.2. Kullanılan Elemanlar

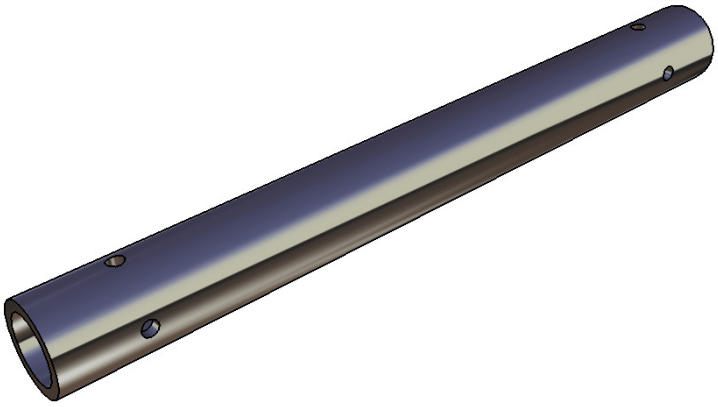
Tasarlanan robot kolları birbirleri ile aynı yapıda olup tamamen simetriktir. Aralarındaki tek fark Master robot kolunda bir kıskaç (gripper) bulunmamaktadır. Bunun nedeni ise operatörün kullandığı bir robot kolunun kıskaca değil kontrol kolu benzeri bir tutacağa ihtiyacı olmasıdır. Robot kollarında genelde servo motor kullanılmaktadır. Geliştirilen algoritma ve kontrol için bu projede step motorlar kullanılmıştır. Step motorlar birçok güç ve değer aralığında bulunabileceğinden bağlantı direkt olarak motora ve motor şaftına yapılabilmesi öngörülmüştür. Bu nedenle tüm bağlantı elemanları dayanıklı ve çok hafif olmalıdır.

2.2.1. Bağlantı elemanları

Robot kollarında hafif ve yumuşak yapılarından dolay işlemesi de kolay olan malzemeler aşağıda verilmiştir.

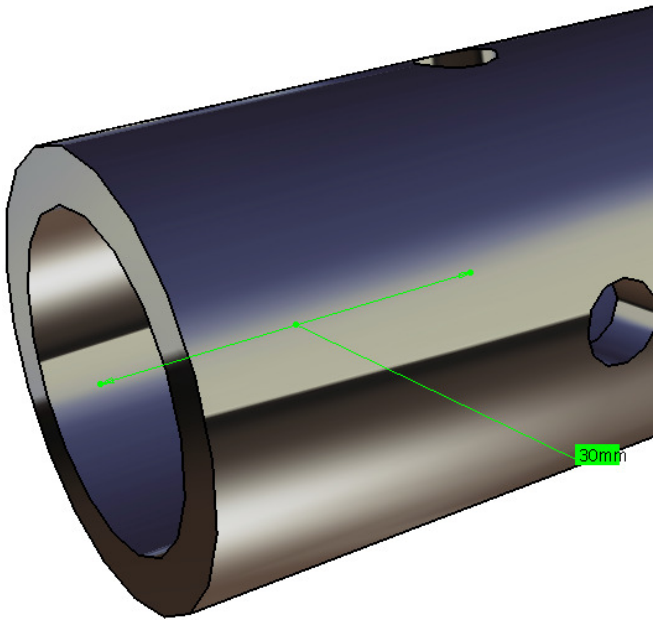
- Alüminyum
- Balsa Tahtası (Daha çok uçan robotlarda kullanılır.)
- Karbon fiber
- HDPE (High Density PolyEthylene)
- Styrofoam

Bağlantı elemanları kare çubuk halinde veya silindir halinde olabilir. Silindirik içi boş çubuk hem diğerlerine göre daha hafif hem de kolay bulunabilmektedir. Ayrıca içi boş silindir eylemsizlik momentinden dolayı daha çok tercih edilmektedir. Diğer malzemelere göre alüminyum silindirik çubuklar piyasada kolaylıkla ve çok ucuza bulunabilmektedir. Bağlantıdan dolayı boyutların daha uygun olması için 25cm'lik bağlantı çubukları kullanılmıştır. Şekil 19'da 3mm et kalınlığındaki ve 16mm dış çapa sahip bağlantı çubukları görülebilmektedir.



Şekil 19 – 16 mm dış çapa sahip bağlantı elemanı

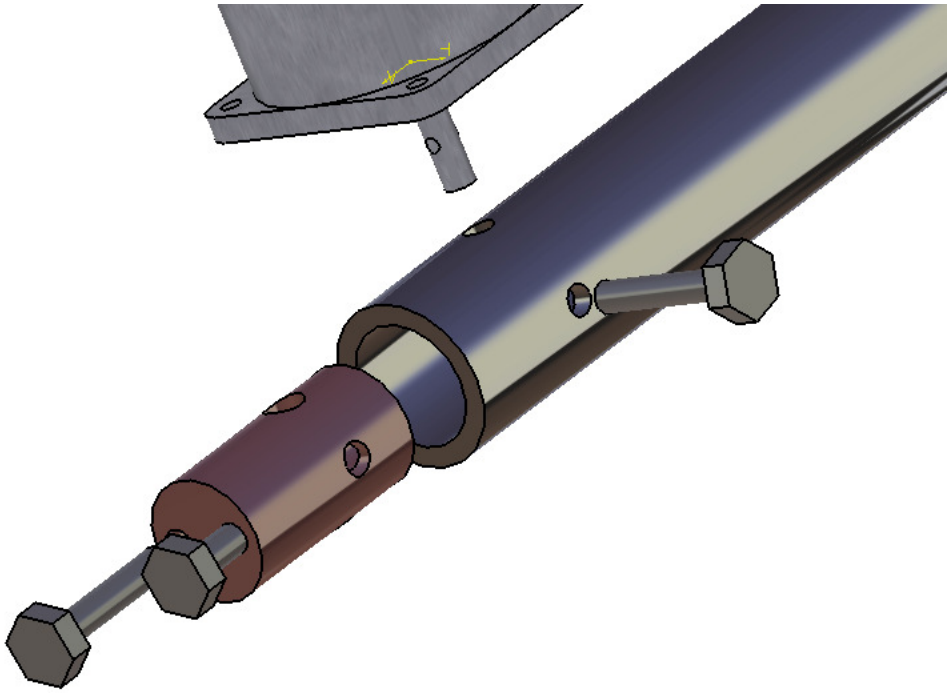
Şekil 20'de ise yataklama için deliklerinin konumu görülebilmektedir. Alüminyumun yoğunluğu 0,00271 kg/m^3 olan çok hafif bir malzemedir. Buna karşılık saf alüminyumun akma dayanamı 7–11 Mpa, Alüminyum alaşımlarının ise akma dayanımı 200 MPa ila 600 MPa aralığında değişebilmektedir [14]. Verilen ölçülerdeki alüminyum çubuğun ağırlığı 0,0383 kg dır.



Şekil 20 – Bağlantı elemanı üzerinde yataklama için deliklerinin konumu

Bağlantı elemanlarının başından ve sonundan 30 mm uzaklıkta açılan deliklerin çapı kullanılan step motorun şaftının 0.25 inch (6,35mm) olarak verilen çapa eşit olması için 6,5 mm olarak açılmıştır. Çubuk elemanın motora olan bağlantısı için bir diğer elemana ihtiyaç duyulmuştur. Bu eleman bağlantı çubuklarının iç çapına eşit boyuttadır. Elemanın konumu motor şaftında olacağı için bir moment oluşturmayacak

ve ağırlığı motorların kaldırdığı yüke çok fazla etki etmeyecektir. Bu nedenle çok yumuşak ve diş açıldığından kolay aşınan alüminyum malzemesi yerine bakır kullanılmıştır. Bağlantı 3 adet vida yardımıyla desteklenmiştir. Ayrıca şafta açılan 3mm çapındaki delikten yararlanılarak bağlantı güçlendirilmiştir. Bu bağlantının görüntüsü Şekil 21'de görülebilmektedir.



Şekil 21 – Motorla bağlantı

2.2.2. Ayarlı direnç

Eklemlerde açıları ölçmek için 10kΩ değerinde ayarlı direnç kullanılmıştır. Piyasada çok kolay bulunabilmesi ve ucuzluğundan dolayı tercih edilen bu ayarlı direnç'in çalışma aralığı 270° derecedir. Robot kollarında açısal değişimin 180° dereceden daha fazla olmaması nedeniyle çalışma aralığı bu değerden yüksek olan ve en hassas ölçüm elde edilebilen bir ayarlı direnç kullanılması en uygun çözüm olmaktadır.

2.2.3. Adım motorları

Robot kolunun hareketini sağlamak için adım motorları seçilmiştir. Bu, geliştirilen algoritmanın öngördüğü ölçüm değerleri ile en uygun çözüm getiren sonuç olarak düşünülmüştür. Adım motorları, 360 derecelik tam bir dönüşü birçok küçük adımda yapabilen DC motorlarıdır. Herhangi bir geribesleme mekanizması kullanılmadan adım motorlarının pozisyonu ayarlanabilmektedir. Adım motorlarının DC motorlardan farkı ise dişli benzeri bir demirin etrafındaki üzerinde birçok diş bulunan elektro-mıknatıslardır. Bu elektro-mıknatıslar bir kontrolcü tarafından uyarıldığı zaman birbirine yakın dişliler arasında çekim olur ve böylece bir adımlık hareket sağlanır. Adımlar arasında geçen zaman kontrolcü tarafından ayarlanır ve hız bu şekilde kontrol edilebilmektedir. Ayrıca uygun bir şekilde kurulan devre ile motorun sağlayacağı maksimum tork değerine ulaşılabilir. Elektro manyetik bobin sarım farkına göre adım motorları tek kutuplu ve çift kutuplu olarak ikiye ayrılır. Bu çalışmada farklı boyutlarda 10 adet tek kutuplu adım motoru kullanılmıştır. Tek kutuplu adım motorlarında 6 adet uç bulunur. Bunlardan ikisi ortak topraklama uçlarıdır. Diğerleri 4 uç ise 2 faz içindir.

Projede kullanılacak adım motorları hesaplamalarda bulunan tork değerlerine göre seçilmiştir. Şekil 22-25 arası gösterilen tablolarda adım motorlarının kapasiteleri görülebilmektedir.

a) Motor 1-2: Astrosyn STEPPER 23LM-C355-P9V

Motor çapı : 59.42cm
Motor uzunluğu : 49.5 mm
Sarım direnci : 2.2 Ohm
Tork : 6.2 Kg-cm
Rotor Interia : 0.110 Kg-cm2
Hareket esnasındaki Tork : 0.54 Ncm
Ağırlık : 450 gr

Şekil 22 – Astrosyn Adım Motoru

2 Numaralı motor, denklem 2.11'e göre yapılan hesaplamalarda bulunan 10.5 Ncm'lik tork değerine ulaşabilmesi için 1:20 oranında motora bağlı bir redüktöre ihtiyaç duyulmaktadır. Bu sayede 10.8 Ncm değerinde tork elde edilmiştir. Gücün artış oranına göre açısal hız değeri de aynı oranda azalmıştır.

Motor 3 – SMW3596 Mimbea Motor

Motor boyutları	SMW35 — 96
Bir Dönüşteki adım sayısı	96 (3.75° / Step)
Sürüş Metodu	2 — 2 2 — 2 PHASE
Devre çeşidi	BIPOLAR CONST. CURR.
Voltaj	24 [V]
Hareketli tork değeri	**0.085 Ncm**

Şekil 23 – Mimbea SMW3596 ve Özellikleri

3 Numaralı motor, denklem 2.6'ya göre yapılan hesaplamalarda bulunan 6.37Ncm'ilk tork değerine ulaşabilmesi için 1:75 oranında motora bağlı bir redüktöre ihtiyaç duyulmaktadır. Bu sayede 6.38 Ncm değerinde tork elde edilmiştir.

SMSSH5 - 20 Mimbea Motor

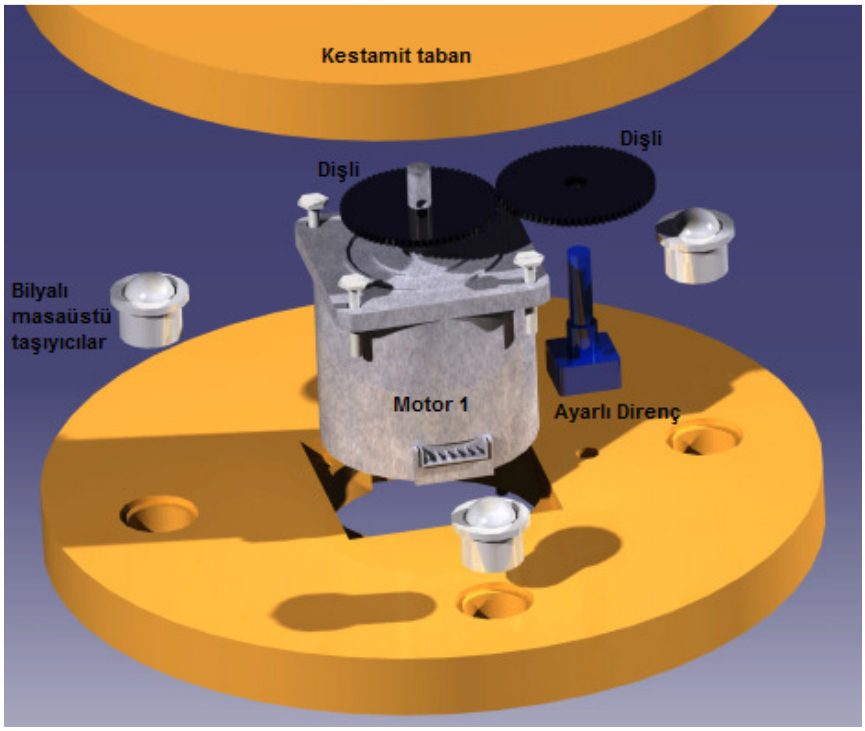
Motor Boyutları	SMSSH5 - 20
Bir Dönüşteki Adım Sayısı	20
Adım Aralığı	18°
Sürüş Metodu	2 - 2 PHASE
Voltaj	3V
Hareketli tork değeri	0.0025 Ncm

Şekil 24 – Mimbea SMSSH5 ve Özellikleri

4 Numaralı motor, denklem 2.1'e göre yapılan hesaplamalarda bulunan 0.223Ncm değerinde torka ulaşabilmesi için 1:100 oranında bir redüktöre ihtiyaç duyulmaktadır. Bu sayede 0.25 Ncm değerinde tork elde edilmiştir.

2.3. Montaj ve Cad Modelleme

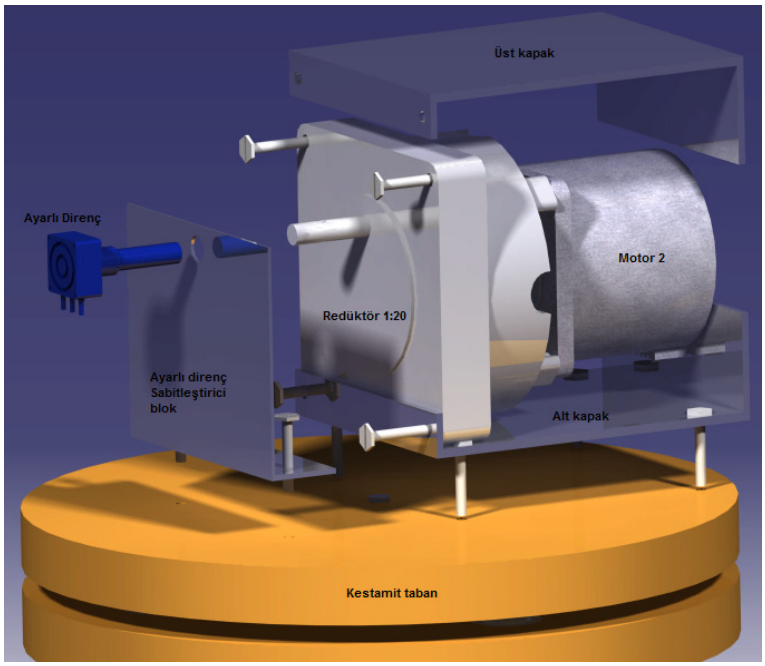
Robot kollarının montajı eklem açılarının 270 derece olacak şekilde ve ayarlı direnç ile açı ölçümüne imkân verecek bir biçimde olmalıdır. Z eksenindeki dönüş hareketi sağlayacak ilk motorun montajı 98mm yarıçapındaki kestamit silindirlere yapılmıştır. 2 silindir arasındaki dönüş esnasında oluşacak sürtünme kuvvetini en aza indirmek için bilyalı masaüstü taşıyıcılar kullanılmıştır. Açıyı ölçmek için ayarlı direncin montajı şaft yönünde tüm motor kolunun bulunmasından dolayı 2 adet dişli yardımıyla yapılmıştır. Bu dişlilerden biri motor şaftına diğeri ise kestamit tabana bağlanarak dönme hareketi ayarlı direnç üzerine aktarılmıştır. Bu sayede alınan ölçümler ile açı hesaplanabilmektedir. CAD modeli CATIA V5 programı ile hazırlanmıştır ve görüntüsü Şekil 25'de gösterilmektedir.



Şekil 25 – Taban montajı

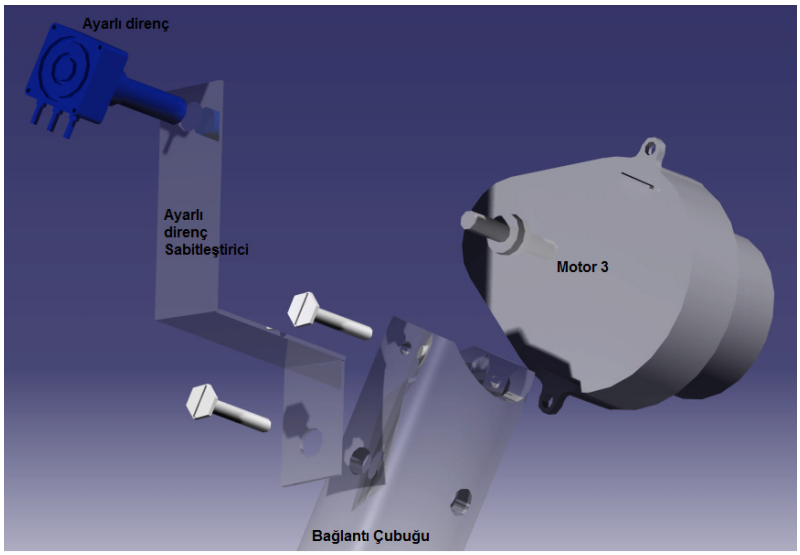
Tabanın üstüne montajı yapılacak ve robot kolunun tüm elemanlarının hareketini sağlayacak motor ve redüktörünün montajı alüminyum sac levha ile kestamit tabana yapılmıştır. Kestamit tabanın merkezinin hizasında 1. Bağlantı elemanı bulunmaktadır. Böylece hareket sırasında z ekseninde bir öteleme olmasından kaçınılmıştır. Bağlantı 4 adet metrik 3 vida ile yapılmıştır. Ayarlı direnç şafta

bağlanmış ve bir blok ile taban bağlanarak sabitlenmiştir. Sabitleme işlemi robot kolunu yatay pozisyona getirerek bacaklardaki direnç değeri ölçüp master koluyla beraber yapılmıştır. Böylece montaj hatalarından doğacak açı farkları engellenmiştir. Bağlantı elemanının şafta olan montajı (Bkz: Şekil 21) redüktör şaftına açılan 2mm çapındaki bir delik ile güçlendirilmiştir. Redüktör ile motor arasındaki mesafeyi azaltmak için redüktöre kulakçıklar yapılmış ve motor buraya 3 adet vida ila sabitlenmiştir. Motorun olası istenmeyen hareketlerini de önlemek amacıyla motor bloğunu sabit tutmak için redüktör ve motoru kapsayan bir sac levha ile üstü kapatılmış ve 96mm uzunluğundaki sac bükülerek motoru sabitleştirmiştir. 3 boyutlu modeli Şekil 26'de görülebilmektedir.

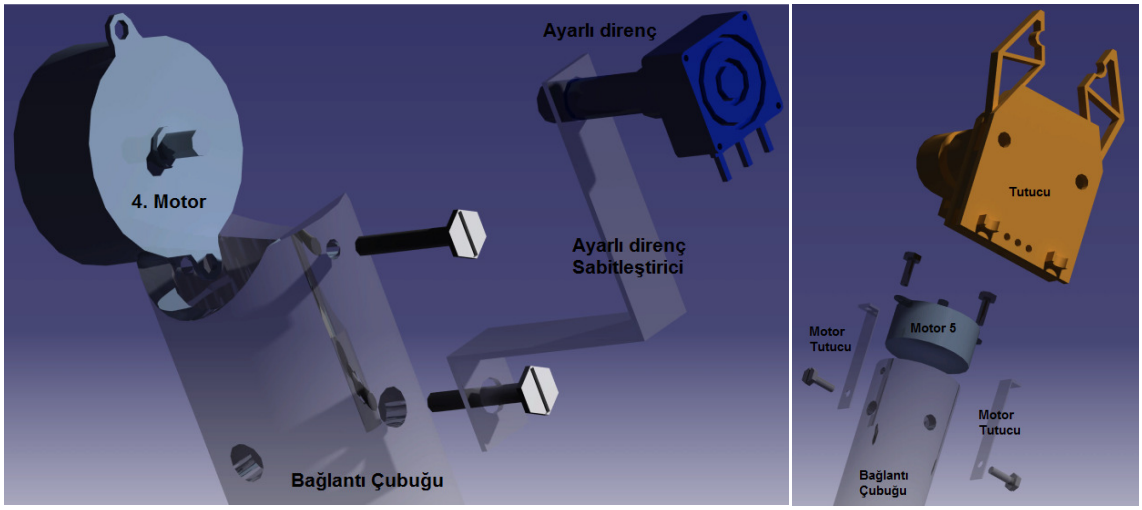


Şekil 26 – 2. Motor ve Redüktörün tabana montajı

Bağlantı çubuğunun ve 3 numaralı motorun montajı Şekil 27'de görülebilmektedir. Motorun kulakları silindirik çubuğa vida ile sabitlenirken ayarlı direnci sabitleştirmek için çubuk ile arasına bir parça yapılmıştır.

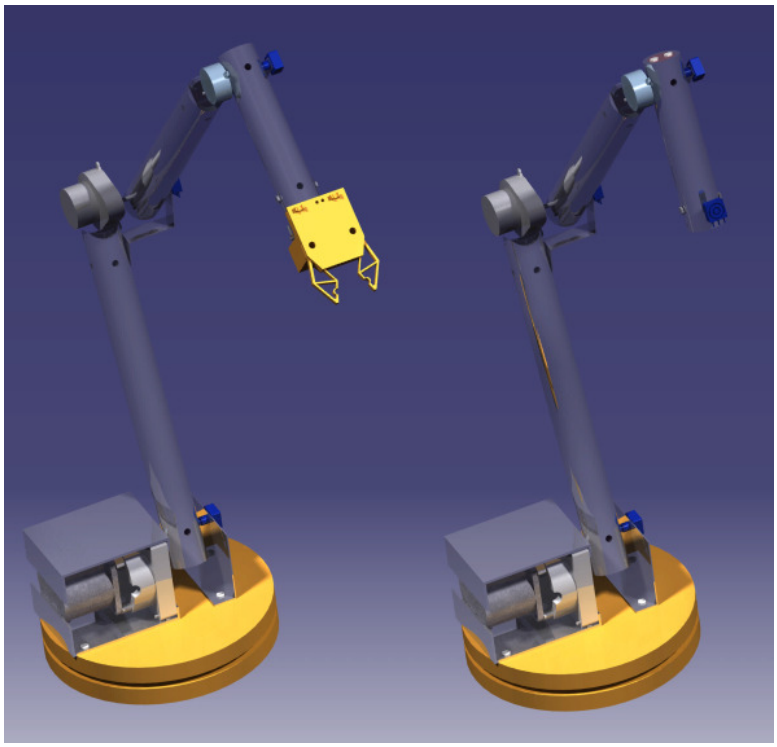


Şekil 27 – Bağlantı çubuğu ile 3 numaralı motorun montajı



Şekil 28 – 4. Motora olan bağlantı ve tutucunun robot koluna eklenmesi

4.Motora olan bağlantı ve tutucunun robot koluna eklenmesi için ayarlı direnç sabitleştiricisi ve robot tutucu parçalarının üretilmesinden sonra bağlantı M3 vidalarla yapılmıştır. Şekil 28'de görüldüğü gibi montaj motor şaftlarına yapılmış ve vidalarla güçlendirilmiştir. Sonuç olarak Şekil 29'de görülen biçimde master ve slave kollarının imalatı tamamlanmıştır. Ek-1'de CAD modellemesi yapılmış robot kollarının detaylı 3 boyutlu resimleri mevcuttur.



Şekil 29 – Slave (sağda) ve Master (solda) robot kolları

3. ELEKTRONİK TASARIM

Prototip robot kolunun denetiminin sağlanması ve hem adım motorlarını sürmek amacıyla hem de seri haberleşmeyi kolay bir şekilde kontrol edebilecek kapasiteye sahip elektronik devreler kullanılması gerekmektedir. Piyasada adım motorlarını kontrol eden birçok devre bulunmaktadır. Ancak bu çalışmada hem birbirleriyle haberleşebilen hem de farklı boyutlarda ve çalışma aralıklarındaki motorları sürebilen devreler gerekmektedir. Bu nedenle piyasadaki hazır devreler yerine tamamen özgün bir tasarıma sahip devrelerin tasarlanması ve üretilmesi gerekmektedir. Tasarlanan bu devrelerde motorların aynı anda hareket yapabilmesi için birbirinden bağımsız algoritmaların çalışabileceği farklı mikroişlemciler gerekmektedir. Bu sayede robot kolunun hareketi motorların aynı anda çalışmasıyla sağlanacak ve hareket sürekli bir şekilde olacaktır. Tek bir robot kolunda bulunan 5 adet motoru kontrol edebilen bir devrede, 5 adet mikrodenetleyici olacağından ve bu denetleyicilerin çalışacağı elektronik devre elemanlarını da içeren bir devrenin boyutları çok büyük olacağından. Her bir motor için ayrı ayrı devrelerin tasarlanması uygun görülmüştür. Böylece hem devrelerin karmaşıklığı azaltılacak hem de olası hataların belirlenmesi

kolaylaşacaktır. Ancak tasarlanacak bu devrelerin arasındaki bağlantıların daha da önem kazanmaktadır.

3.1. Elektronik Devre Elemanları

Tasarlanan devre elemanları adım motorlarını aldığı açı bilgisine göre hareket ettirebilecek ve bu esnadaki olası ısınma ve performans düşüklüğünü en aza indirmesi gerekmektedir. Bu amaçla kullanılacak parçaların mikro denetleyiciyi yüksek akımdan koruması ve devrenin aşırı ısınmasını önlemeleri gerekmektedir.

3.1.1. PIC16F877A

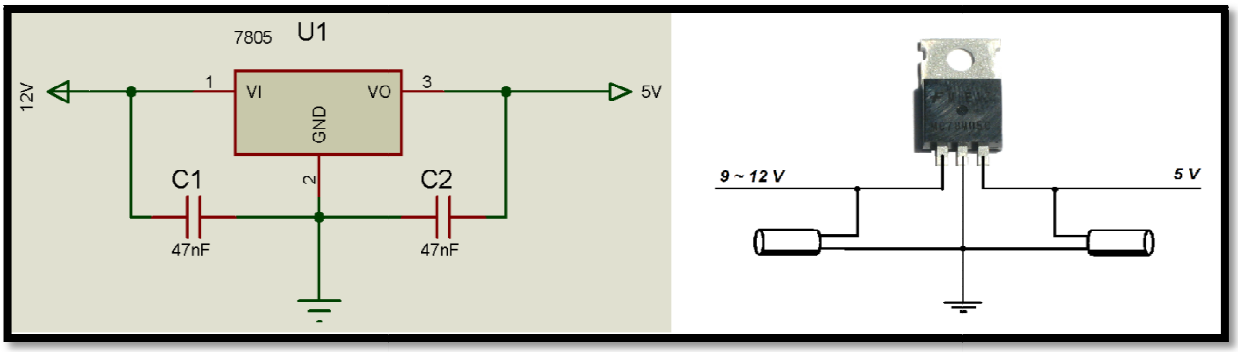
Günlük hayatta kullandığımız neredeyse tüm elektronik ve mekanik ürünlerde mikro kontrolcüler bulunmaktadır. Modern otomobillerde, mikrodalga fırınlarda buzdolaplarında çamaşır makineleri mikro kontrolcüler ile çalışmaktadır. Mikro kontrolcüler belli bir hafıza kapasitesine sahip, küçük bir işlemci çekirdeğine sahip, girdi ve çıktılara izin veren tek bir bütünleşmiş devre içeren denetleyicilerdir. Montajının ve programlamanın kolay olması nedeniyle çok tercih edilen bu yapılar çok farklı çeşitlerde oldukça ucuza bulunabilmektedir. Bu projede 40 bacaklı yapıda olan PIC16F877A(Şekil 30) kullanılmıştır. Ayrıntı özellikleri Tablo 2'de bulunabilir.

Şekil 30 – PIC16F877A

Parametrenin adı	Değeri
Program Hafıza Tipi	Flash
Program Hafıza Kapasitesi(KB)	14
CPU Hızı (MIPS)	5
RAM (Byte)	368
Data EEPROM (bytes)	256
Zamanlayıcı	2 x 8-bit, 1 x 16-bit
ADC	8 ch, 10-bit
Sıcaklık aralığı (C)	-40 to 125
Çalışma Voltaj aralığı (V)	2 to 5.5
Bacak Sayısı	40

Tablo 2 – PIC16F877A özellikleri

PIC16F877A 5 giriş/çıkış portuna sahiptir. Bunlar A, B, C, D ve E olarak adlandırtır. Bu kontrolcü 20 Mhz kristal ve 22pf kapasitörler ile kullanılmaktadır. Ayrıca bu projede olduğu gibi 12V'luk bir kaynak ile çalışılmak istenildiğinde voltaj düzenleyici devre ile çalışma aralığı olan 2 – 5.5V arasına indirilmelidir. Şekil 31'da 7805 ile kurulan bir devre görülebilmektedir.



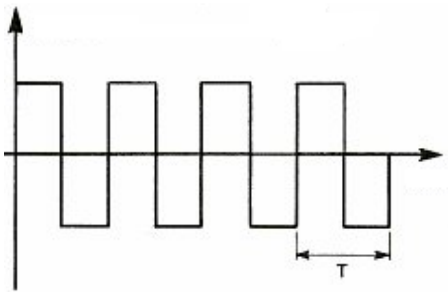
Şekil 31 – 7805 ile kurulan bir devre

Besleme voltajı 5V'a indirgendikten sonra devre 20 Mhz kristal ve kapasitörlerle tamamlanır. PIC16F877A programlandıktan sonra çıkış olarak seçilen port'dan gerekli çıkışlar verilerek yapılmak istenen işlem gerçekleştirilir. Kontrolcünün 2. Bacağı olan AN0 bacağından ise Analog sinyaller okunabilmektedir. Bu sayede bir ayarlı direnç ile açı ya da pozisyon ölçülebilir (Bkz: ELEKTRONİK TASARIM).

Okunan değerler 2 ila 5.5V aralığında olabilir ve bu değer Analog'dan dijital'e çevrilerek 0 ile 1024 (10bit) Aralığında dönüştürülür ve koda dâhil edilerek okunmak istenen değere göre işlemler devam ettirilebilmektedir (Bkz: YAZILIM). Ayrıca denetleyicinin 25 ve 26. bacakları olan RX/TX bacakları ile haberleşme sağlanabilmektedir. İstenildiğinde Max232 kullanılarak RS232 yardımıyla bilgisayar ile iletişim kurulabileceği gibi diğer bir kontrolcünün RX/TX bacaklarına ters bağlanarak 2 denetleyicinin haberleşmesi sağlanabilir. (Bkz: MAX232) Kontrolcü ile tasarlanan projeler ilk önce bir simülatör programıyla denenir. Böylece devrede karşılaşılacak olası hatalar azaltılmış olur. Devredeki hatalı bir bağlantı ile kontrolcü, besleme voltaj aralığı dışında bir değer ile beslenirse bu kalıcı hatalar yol açabilir. Simülatör ile denenen program en sonunda devreye aktarılır. Kontrolcü piyasada bulunan programlayıcı devreler ile programlanabilir. Ancak bu işlem için Denetleyici devreden bağımsız olmalıdır. Her seferinde tekrar çıkarmamak için Max232 ve *"bootloader"* kullanılarak seri haberleşme ile programlama işlemi hızlandırılabilir.

3.1.2. L297 Adım Motor Kontrolcü

L297 Motor kontrolcüsü mikro işlemci uygulamalarında, çift kutuplu 2 fazlı ve tek kutuplu 4 fazlı adım motorları için 4 fazlı sürücü sinyalleri oluşturur. Motor, yarım adım ve tam adımda sürülebilir. Bu elektronik elemanın en büyük özelliği ise sadece "clock", yön ve ayar için gerekli giriş sinyallerine ihtiyaç duymasıdır. Kontrolcünün "CLOCK" bacağına uygulanan bir sinyalin artan her köşesinde bir adım için gereken çıktılar verilmektedir. Şekil 32'de 4 adım için gereken sinyal ve 4 kere çalıştırıldığında mikroişlemcinin B1 bacağı kullanılarak bu sinyali oluşturan Ccs kodu görülebilmektedir.




```
#define CLOCK PIN_B1

void stepgit()
{
output_low(CLOCK);
delay_ms(100);
output_high(CLOCK);
}
```

Şekil 32 – 4 adımlık hareket için gereken sinyal ve örnek kod

Motorun hareketi için gereken sinyalin frekansı ile hız ayarlanabilir. Yön bilgisi ise L297 kontrolcüsünün CW/CCW bacağına 0 ya da 1 sinyalleri ile belirlenir. Tablo 3'de tüm bacakların işlevi görülebilmektedir.

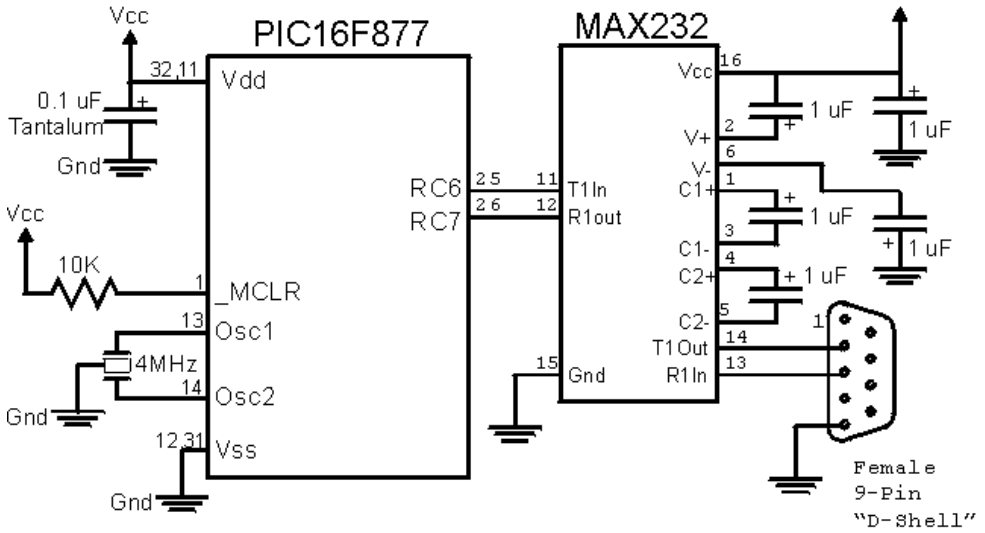
No	Adı	İşlevi
1	SYNC	Diğer L297'ler ile senkronize çalışma için kullanılır
2	GND	Toprak bağlantısı
3	HOME	Aktif sinyal aldığında Çıkışa başlangıç değerini (ABCD = 0101) uygular.
4	A	A çıkışı
5	INH1	A ve B'nin inhibe kullanımını sağlar
6	B	B çıkışı
7	C	C çıkışı
8	INH2	C ve D'nin inhibe kullanımını sağlar
9	D	D çıkışı
10	ENABLE	Aktive edilmediğinde A,B,C,D, INH1 ve INH2 pasif olur
11	CONTROL	Kontrol girdisi.
12	Vs	5V Besleme
13	SENS2	C ve D çıkışları için yük akım algılama Voltajı
14	SENS1	A ve B çıkışları için yük akım algılama Voltajı
15	Vref	Referans Tepe yük akımını belirler
16	OSC	Chopper hızını belirlemek için RC devresi kurulması gereken giriş
17	CW/CCW	Bu bacağa uygulanan sinyal motor yönünü belirler. 1 – CW, 0 - CCW
18	CLOCK	Uygulanan bir sinyale göre 1 adım için gereken çıktılar verilmektedir
19	HALF/FULL	1 – Yarım, 0 – Tam adım
20	RESET	Çıkışa başlangıç değeri uygulanır. (ABCD = 0101)

Tablo 3 – L297'nin özellikleri

3.1.3. MAX232

MAX232 RS232 seri portdan alınan sinyalleri TTL'ye uygun olarak dönüştürerek kullanılmasını sağlar. Bu sayede mikro kontrolcü ile bilgisayar arasındaki bağlantı sağlanmış olur. Bir MAX232 bütünleşmiş devre ile 2 adet mikro kontrolcü bilgisayar

ile haberleşebilmektedir. Şekil 33'de MAX232'nin genel yapısı ile bağlantı örneği görülmektedir. Detaylı bilgi için Max232'nin veri sayfası bakanız.

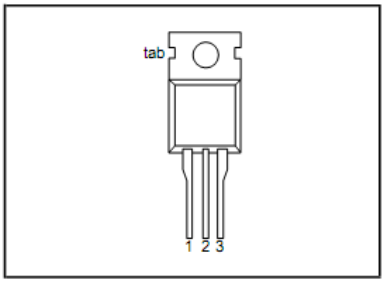


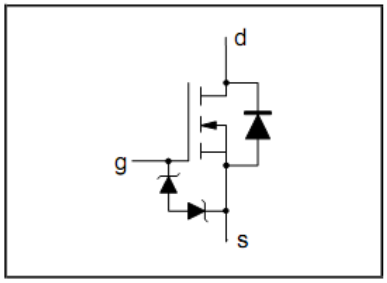
Şekil 33 – MAX232'nin genel yapısı ve bağlantı örneği

3.1.4. Mosfetler

Mosfetler elektronik sinyallerini yükseltmekte veya anahtarlamakta kullanılır. Kullanım alanı çok fazla ve çeşitli olan bu transistorlardan IRFZ44 projede anahtarlamak için kullanılmıştır. Mosfetin 2. Bacağına uygulanan bir voltaj devreyi tamamlar ve 1 ve 2. Bacaklar arasında bağlantı tamamlanır. Şekilde IRFZ44'nin bacakları ve yapısı görülmektedir. Ekte bulunan veri sayfasında detaylı bilgilere ulaşabilir.

PIN	DESCRIPTION
1	gate
2	drain
3	source
tab	drain





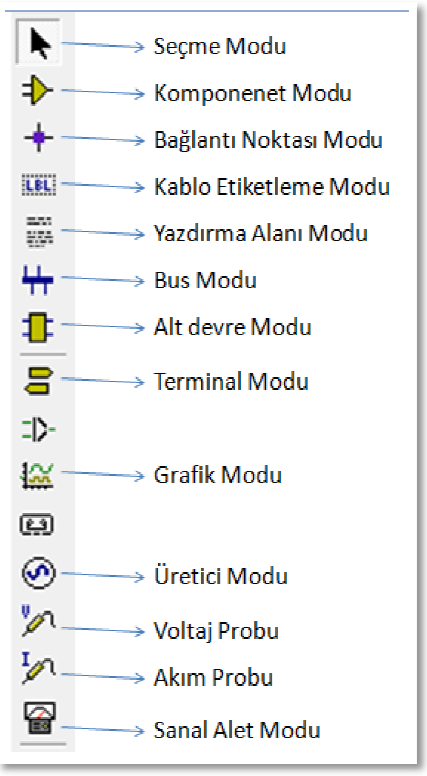
Şekil 34 – IRFZ44'nin bacakları ve yapısı

3.2. Proteus Programı İle Devre Tasarımı

3.2.1. ISIS

ISIS programı ile tasarlanan devrelerin simülasyonu yapılabilmektedir. Bu sayede yazılan programdaki hataların tüm devreyi etkilemesi engellenir. Proteus programında bir devre elemanı ya da bir kabloyu seçmek için sol tuşu kullanmanız gerekmektedir. Sağ tuş ile tıkladığında parçanın özelliklerini veya konumunu değiştirilebilir. Devreyi kurduktan sonra voltaj okuma ya da diğer okuma işlemlerin sonuçlarını görmek için sağ altta bulunan *"play"* düğmesine basmak gerekmektedir. "*Play"* düğmesine basılması devreye güç vermenizi sağlar ve hedeflenen verilere ancak bu sayede ulaşabilinir.

Tasarım araç çubuğu devre tasarımında kullanılan düğmeleri barındıran kısımdır. Bu çubukta en çok kullanılan düğmeler ve işlevleri Şekil 35'de görülmektedir.



Şekil 35 – ISIS ara yüz

- Seçme Modu: Devre elemanlarını seçmek için kullanılır.
- Bileşke Modu: Bu düğmeye basıldığında sağ menüde seçilen ve kullanılabilir durumdaki komponentler listelenir. Yeni komponent eklemek için P(pick devices) düğmesine basılıp çıkan listeden arama sonucu istediğiniz elemanı menüye ekleyebilirsiniz.
- Bağlantı Noktası Modu: Bu seçenek işaretli olduğunda kabloları birleştirip düğümler elde etmek etkinleşir.
- Terminal Modu: Bu kısım devreyi beslemek için *power*, topraklamak için *ground* gibi önemli bağlantıları yapabilmenize yarar.
- Üretici Modu: Alternatif akım üretmenizi sağlar.
- Voltaj ve Akım Probu: Üzerine koyacağınız kablo üzerindeki akımı veya gerilimi bağlama şeklinizi değiştirmeden okur.
- Sanal Alet Modu: Voltmetre Multimetre RS232 terminal gibi cihazları barındıran kısım.

3.2.2. ARES

ARES programı ISIS'de çizilen ve test edilen devreye gerekli yolları ve bağlantıları yapmaya yarayan programdır. ISIS'de bulunan "ARES" tuşu ile devre ARES'e aktarılır. Burada gereken boyut seçildikten sonra sol menüye otomatik olarak eklenen parçalar tek tek birbirine bağlanır. Devrenin karmaşıklığına göre baskı devre 2 katmanlı olabilir. Yolların genişliği genelde tasarımcıya bırakılır ancak hesaplamalar ile o yolun üzerinden geçen akım genişliğin belirlenmesinde önemli bir yol sağlar. ARES'de yapılan yollar istenilen şekilde ve uzunlukta olabilir. Ancak 90 derecelik keskin dönüşlerden kaçınılırsa devrenin performansı daha da artar ve olası hatalardan kaçınılır.

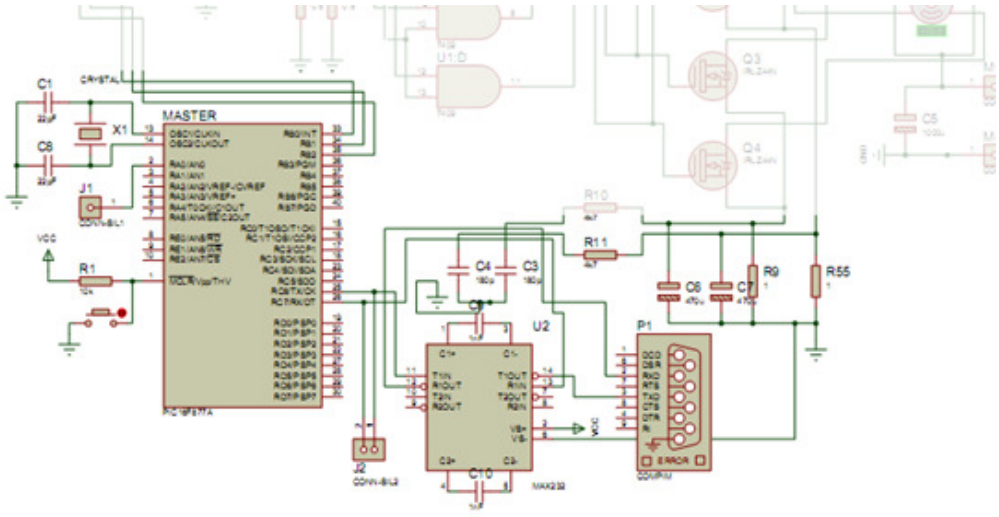
3.3. Tasarlanan Devre Ve İşlevi

Tasarlanan devre PIC16F877A ile L297 kullanılarak step motor sürebilen bir devredir. Devreyi tasarlarken step motorunun kontrolü önce L293 kullanılmıştır. Bu devre ile yapılan çalışmalarda PIC içerisinde çalıştırılan programın karmaşıklığı L293 ile kontrolü daha da zorlaştırmıştır. L293 ile yapılman kontrolde istenilen bacaklara 1

değerini doğru sırada vermek gerekmektedir. Bu işlem zaten karmaşık olan programı daha da karmaşıklaştırmıştır. İkinci olarak L297 – L298 çifti kullanılarak kontrol sağlanmaya çalışılmıştır. L297 ile bir step hareketin "clock" bacağına verilen tek bir giriş ile yapılabildiği için kontrol kolaydır. Bu sayede karışık bir programda kullanılması daha verimli olmuştur. Ancak robot kollarını +z ve –z yönlerinde hareketini sağlayan motorlar 1,5 amper çekmektedir. Bu değer ise L297 – L298 kullanımında aşırı bir ısınmaya sebep olmuştur. Bu nedenle son olarak Mosfet kullanılmasına karar verilmiş ve devre buna göre tasarlanmıştır. Devre 3 bölümde incelenebilir:

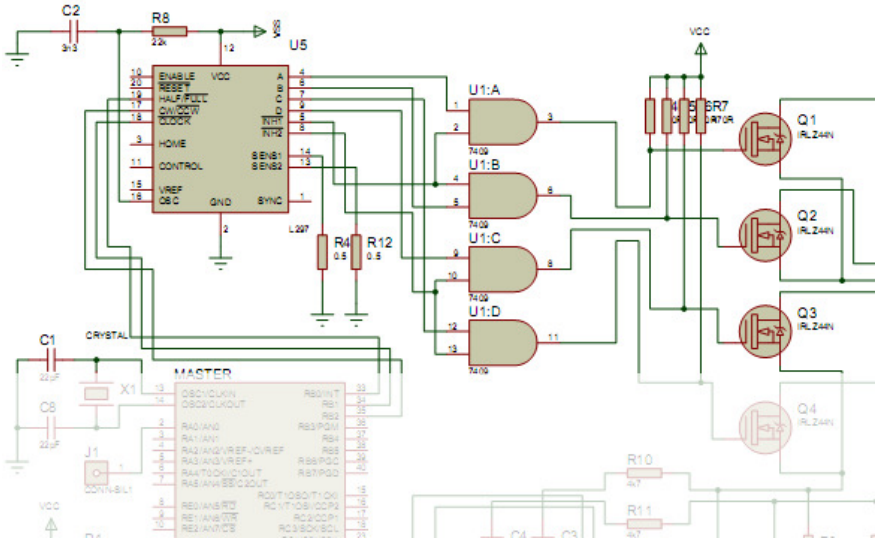
Birinci bölüm PIC16F877A için gereken devre elemanlarının bulunduğu ayarlı dirençlerden gelen bilgileri okuyan ve adım motorlarının hareketi için yön ve hız kontrolünün sağlandığı kısımdır(Bkz: Şekil 36). Bu bölümde 12V ile beslenen devreyi PIC16F877A'nın çalıştığı 5V'a indirgen voltaj regülatörü ve PIC'i çalışmasının sağlayan devre elemanları bulunmaktadır. Ayrıca deneyler sırasında bilgilerin alınabilmesi ve hataların daha kolay saptanabilmesi için gereken bilgisayarla RS232 ile bağlantı sağlayan max232 devre elemanı yine bu bölümde yer almaktadır. PIC16F877A 1 nolu bacaktan 5V ile beslenir. Bu bağlantıya bir adet anahtar konulmuştur. Bu anahtar sayesinde oluşan herhangi bir hata esnasında devre baştan başlatılabilir ve programlama sırasında kolaylık sağlamaktadır. Klemens bağlantıdan 12V ile beslenen devre 7805 voltaj regulatörü ile 5V 'a indirgenir ve bağlantı elemanları bu çıkış ile beslenir. Max232 PIC'in RX ve TX haberleşme bacaklarına bağlanır ve çıkışı RS232 bağlantının yapılabileceği dişi soketine verilir. Ayrıca RX ve Tx bacakları doğrudan klemens bağlantıyla birleştirilmiştir ve bu sayede devrelerin birbirleri ile haberleşmesi için gereken bağlantı sağlanmıştır. PIC 3 adet çıkış verir. Bu çıkışlar:

- Motorun adım hızını doğrudan etkileyen ve L297'nin "clock" bacağına verilen çıkış. Bu değer programda 2ms'de bir verilmektedir.
- Motorun döneceği yönü belirten çıkış. Eğer değer 1 ise motor saat yönünde, 0 ise saat yönünün tersine döner.
- L297'nin çalışmasını kontrol eden ve "enable" bacağına bağlanan çıkış.



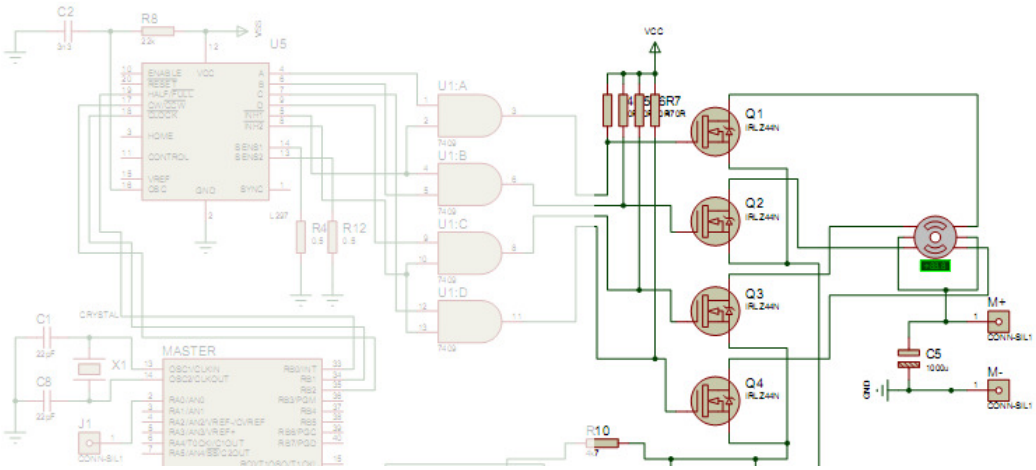
Şekil 36 – PIC16F877A için gereken devre elemanlarının bulunduğu bölüm

İkinci bölüm L297 motor sürücü ile "VE" kapısının bulunduğu kısımdır. Şekil 37'de görülen bu bölümde PIC'den alınan yön ve giriş sinyallerinin frekansına göre L297 motorun bir adım gidebilmesi için gereken sinyalleri üretir. Bu sinyaller L297'nin A, B, C ve D çıkış bacaklarından yapılır. Ancak bazı durumlarda INH1 ve INH2 bacakları çıkış için kullanılmaktadır. Bu nedenle L297'nin çıkış bacakları MOSFET'lere bağlanmadan önce VE kapısından geçirilir. Bu sayede 2 bacağı senkronize çalıştırmaya yarayan INH1 ve INH2 bacakları kullanılabilmektedir. Bu kısımda ayrıca motorun hareketini sağlayan 12V ile beslenen kısımlarda kullanılan devre elemanları bulunmaktadır. Kapasitörler ve direnç değerleri kullanılan en büyük motorun çekebileceği en yüksek amper değerine göre hesaplanmıştır.



Şekil 37 – Devrenin L297 motor sürücü ile "VE" kapısının bulunduğu kısım

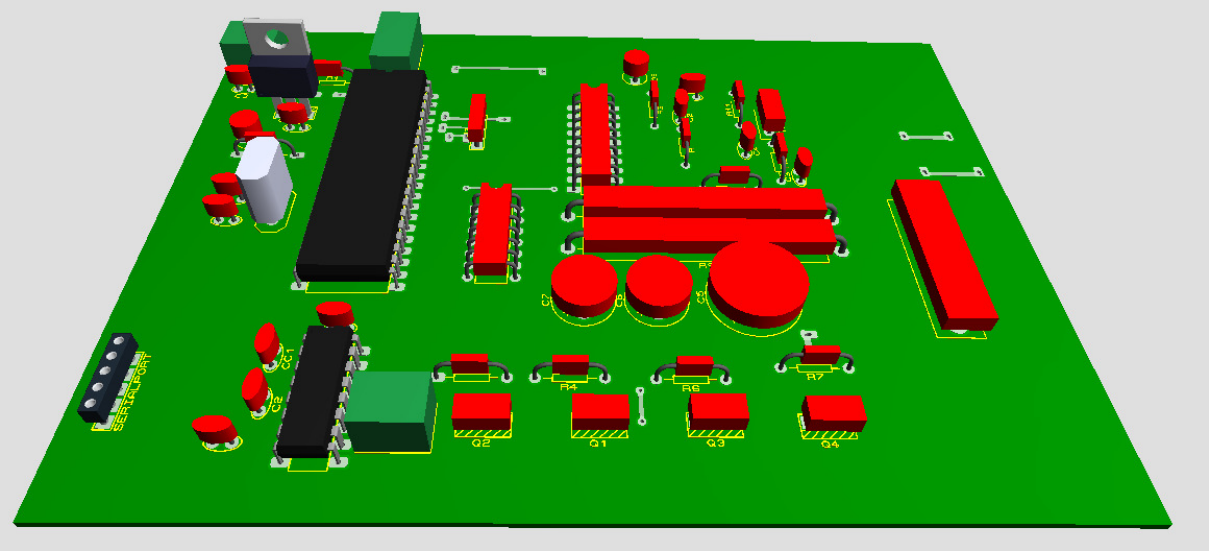
Şekil 38'de görülen bu kısımda devrenin fazla ısınmasını engelleyen MOSFET'ler ve motor çıkışları bulunmaktadır. Daha önce yapılan denemelerde motor doğrudan akımı L297 üzerinden çekmeye çalıştığı için ısınma olmuş ve sonuca ulaşılamamıştır. 4 adet MOSFET devre elemanlarını motoru besleyen ve akımın çekildiği doğrudan güç kaynağına bağlı kısımdan ayırmıştır. Bu sayede bir anahtar görevi görmektedir. L297'den oluşturulan çıkışlar "VE" kapısından geçtikten sonra mosfetlerin bacağına ulaşır. Giriş alan mosfet motor ile güç kaynağı arasındaki bağlantıyı kapatır. Bu sayede Motor devre elemanlarından bağımsız bir şekilde dönmeye başlar. Deneylerde en büyük motorun çalıştırılmasında MOSFET'lerin ısındığı gözlemlenmiş ancak bu ısı iletkenliği yüksek alüminyum soğutucu plakalar kullanılarak kontrol altına alınabilmiştir. Soğutucu plakalar 4 Mosfet için ayrı ayrı kullanılmıştır. 1 adet soğutucu 4 Mosfet'e birden takıldığında topraklar ortak olacağından devre istenilen bir şekilde çalışmayacaktır. Bu nedenle her devre için 8mm'lik soğutucu plakalar kullanılmıştır.



Şekil 38 – Mosfetler ve motor çıkışları

Üç kısımda incelebilecek olan bu devre önce Proteus programında denenmiştir. Ancak bilineceği üzere proteus programında dışarıdan müdahale olan bu çalışma ve benzeri çalışmalarda simülasyon yetersiz kalmaktadır. Bu nedenle sadece motorun hareketi ve iletişim doğruluğu incelebilmiştir. Bu deney için daha basit bir program yazılmış ve haberleşme kurularak test edilmiştir. 2. Aşamada ise bu devre elemanları ile birlikte breadboard 'a kurulmuştur. Bu sayede simülasyondan kaynaklanan sorunlar çözülmüş ve motorun dönüşü gözlenip programdaki değerler değiştirilmiştir. Örnek olarak, proteus simülasyonunda adımlar arası bekleme değerleri önem kazanmamaktadır. Ancak gerçekte deneyler sonucunda ulaşılan en düşük bekleme süreleri büyük motor için adım başı 2ms, diğerlerinde ise 1.5 ve 1 ms bekleme

süreleri olarak bulunmuştur. Ayrıca 2 devrenin haberleşmesi sırasında “printf” kullanımı proteus’da çalışmış ancak gerçek devre ile yapılan denemelerde bu çözüm sonuçsuz kalmıştır. Bu nedenle program algoritması bölümünde belirtilen haberleşme protokolü devrenin breadboard kısmında yapılan deneyler sonucu tasarlanmıştır. Son olarak devrenin çalışmasını hızlandırmak ve düzgün kılmak amacıyla devre baskı devreye geçirilmiş ve ARES ile yapılan yol haritaları basılmıştır. Sonuçta Şekil 39’da görülen devreden her motor için birer tane olmak üzere toplamda 9 adet kullanılmıştır. Baskı devrelerde 5 adet atlama kullanılmak zorunda kalınmıştır. Bu atlamalar PIC ile L297 arasındaki bağlantılarda ve devreyi çevreleyen 12V yollarında kullanılmıştır. Bu nedenle baskı devre 2 katmanlı olarak adlandırılabilir. (Bkz: Şekil 39)



Şekil 39 – Baskı devre

4. YAZILIM

Robotik programlama bilinen bilgisayar ile programlama çalışmalarından farklıdır. Robotun operatör tarafından verilen komutları yapabileceği uygun dillerde yazılmış bir yazılım gerekmektedir. Robotik programlama sadece verilen komuta göre robota iş yaptırmak kalmayıp aynı zamanda bu işin doğru yapılıp yapılmadığını kontrol etmesi gerekmektedir. Bu nedenle yazılım değişen çevresel etkilere adapte olabilmesi gerekmektedir. Sensörlerden alınan bilgilere göre yenilenebilen ve çok geniş çalışma alanlarına uyan bir yazılım, diğer programlama yöntemlerinden

oldukça karmaşık ve hassas olmalıdır. Yazılımda ne kadar değişen faktör olursa olsun durmaksızın yenilenen çevrenin etkileri daha önce düşünülmemiş olabilir. Bu nedenle robotik yazılımlar aşağıda listelenen özelliklere sahip olmalıdır.

- Eş zamanlı işlemci
- Çevre modelleme
- Hareket kontrol
- Giriş / Çıkış
- Sensörlerden veri okuyabilme
- Hataları fark edebilme
- Doğru hareketlerin tekrarlanabilmesi ve
- Spesifik dil yapısı [19]

4.1. CCS-C İle PIC Programlama

Mikro kontrolcülerin kullanımdaki artış, robotik uygulamalarda ileri düzey programlama dillerinden C diline olan ihtiyacı doğurmuştur. C dili, hem günlük kullanılan İngilizceye olan yakınlığı ile hem de ikilik sistemde olan makine diline kolaylıkla çevrilebilme özelliği sayesinde elektronik ürünler ve mikrodenetleyici programlama da oldukça yaygın olarak kullanılmaktadır. Diğer elektronik ürünlerde olduğu gibi Microchip firmasının PIC mikrodenetleyici ürünlerinde de C dili ile programlama popüler bir şekilde kullanılmaktadır [3].

PIC ürünleri için mevcut olan C derleyicileri içinde, PIC ürünlerinin neredeyse tümünü destekleyen, büyük bir oranla ANSI C uyumlu, esnek ve çok kolay bir şekilde mikrodenetleyici programlanmasına izin veren, birçok iletişim protokolü ve çevresel ürünler için hazır kütüphane dosyaları (kontrol fonksiyonları) içeren CCS (**C**ustom **C**omputer **S**ervices Inc.) firması ürünü **CCS C** PIC C Compiler programı oldukça popülerdir [2].

Kullanılan mikro kontrolcünün kütüphanesi kullanılan frekans ADC kanalları giriş ve çıkış olarak kullanılacak bacaklar ve haberleşmek için gerekli özellikler programın

başında yazılır. Şekil 40'de projede kullanılan PIC16F877 için yazılmış programın başında yazılan kısım ve bu kod parçacıklarının açıklamaları görülebilmektedir.

```
#include <16F877A.h>                          // 16877A kütüphanesi
#fuses HS,NOWDT,NOPROTECT,NOLVP               // Cihaz ayarları
#use delay(clock=20000000)                    // 20 MHz çalışma frekensı
#use rs232(baud=19200,xmit=PIN_C6,rcv=PIN_C7) // baudrate ve haberleşm için kullanılan pinler
#org 0x1F00,0x1FFF{}                          // ROM'da bellek ayırma.
```

Şekil 40 – PIC için başlangıç özellikleri

CCS-C Tablo 4'de görülen değişkenler ve uygun mikro kontrolcünü kütüphaneleri kullanarak giriş ve çıkışların kontrolüne dayanır. Bu değişkenlerin kullanılabilmesi için "*stdlib.h*" (standart library) ve "*input.c"* kütüphaneleri kullanılır. Standart kütüphane ile yazdırma okuma karşılaştırma gibi işlemler yapılabilmektedir. *"Input.c"* kütüphanesinde programın giriş ve çıkışlarını kontrol etmek için gereken işlemlerin gerçekleştirilmesini sağlayan komutlar mevcuttur.

Değişken Tipi	**CCS-PICC tipi**	**Boyut**	**Aralık**	
Bit	int1,short	1	0	1
Char	int8,int,char,signed char, short int	8	-128	127
unsigned char	unsigned int, unsigned char	8	0	255
signed char	int8, int char signed char, int, short int	8	-128	127
İnt	int16, long int	16	-32768	32767
short int		16	-32768	32767
unsigned int	unsigned long int	16	0	65535
signed int		16	-32768	32767
long int	int32	32	-2147483648	2147483647
unsigned long int	unsigned int32	32	0	42949667295
signed long int	signed int32	32	-2147483648	2147483647
float	Float	32	1,18E-35	3,40E+41
double		32	1,18E-35	3,40E+41

Tablo 4 – CCS'de kullanılan değişkenler ve büyüklükleri [18]

Program algoritmasına uygun değişkenler istenilirse *"global"* tanımlanabilir. Bu sayede program içindeki fonksiyonlar bu değişkenlere ulaşabilir. Ayrıca değişkenler *"define"* komutu ile tanımlanabilir. Bu yöntem genelde kontrolcü bacaklarını isimlendirmek için kullanılır. Kontrolcü bacakları *"PIN_X"* şeklinde tanımlıdır. Programda her seferinde *"PIN_X"* ifadesini kullanmak birçok hataya sebep olabilir ve

programın anlaşılmasını zorlaştırır. Şekil 41'de global tanımlanan değişkenler ve bacak tanımlamaları görülebilmektedir.

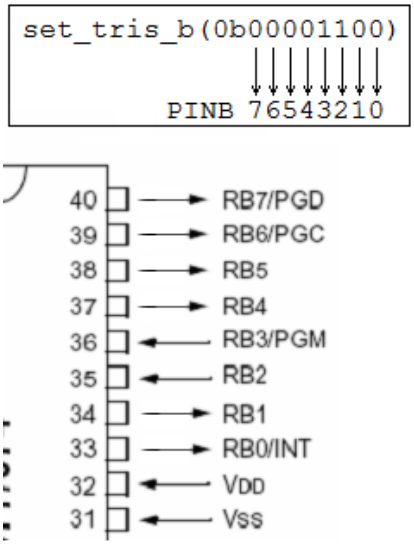
```
#include <stdlib.h>
#include <input.c>
#include <math.h>

#define HALFFULL PIN_B0
#define CWCCW PIN_B2
#define CLOCK PIN_B1

signed long sadval=0,dd=50,madval=0,kontrol,oldmadval=0,aci;
int stepcount=0,bol=0,force=0;
char towhom;
long b1,b2;
```

Şekil 41 – Global değişkenler ve bacak tanımlamaları

Değişkenlerin tanımından sonra ana fonksiyon içinde hangi bacakların giriş hangi bacakların çıkış olduğunu belirtmek için "*set_tris_x()*" fonksiyonu kullanılır. Bu fonksiyona *"0b"* bloğundan sonra gönderilen 2 tabanlı sayının içerisindeki "1" sayıları bulunduğu konumdaki bacakların giriş bacağı olacağını belirler, "0" ise çıkış olanları ifade eder. Sayının en sonunda bulunan basamak ikili sistemdeki gibi 0. Basamağa karşılık gelir böylece bacak numaraların konumu sağda sola artış gösterir. Örneğin b portundaki bacaklarda sadece B2 ve B3 bacaklarını giriş olarak kullanan bir program için *"set_tris_b(0b00001100)"* ifadesi kullanılmalıdır.(Bkz: Şekil 42)



Şekil 42 – Set_tris örneği

Giriş ve çıkış bacaklarının belirlenmesinden sonra programın asıl işlevini gerçekleştirdiği blok bir döngü içerisinden tanımlanır. PIC programlarının normal programlardan farkı ise bu döngüdür. Yazılan programın çalışıp sonlanması PIC programlarında istenilen bir özellik değildir. Program sensörlerden gelen bilgilere göre çalışmasını her seferinden değiştireceği için bu programın tekrarlanabilmesi gerekmektedir. Bu nedenle "*while(1)*", "*while(true)"* ifadelerinin içerisinde asıl program algoritması çalışmaktadır. CCS ile programlamada for, if, else if, komutları diğer C programlarında olduğu gibi kullanılır. Printf, getc, putc gibi komutlar ise ekrana yazdırmak için değil genelde haberleşme için kullanılır. CCS ile yazılan kodlar Ek-6'da bulunmaktadır.

4.2. Seri Haberleşme

PIC programlarında haberleşme PIN_C6 ve PIN_C7 bacaklarından yapılabilmektedir. Program içerisindeki printf() ve putc() komutu ile yazdırmak istenen değişken PIN_C6 bacağından gönderilir. Eğer PIC ile bilgisayar arasındaki haberleşme için MAX232 yardımıyla RS232 ile bağlantı kurulabilir(Bkz Max232). Bu sayede bilgisayarda yazılan başka bir programın port'u dinlemesi ile haberleşme sağlanır. Bu haberleşmede *"baudrate"* ve bilgisayarda kullanılan *"COM"* büyük ölçüde önem arz etmektedir. Haberleşme "printf()" komutuyla yapıldığında gönderilmek istenilen değişken değerinden sonra "\r" yazılması gerekmektedir. Seri port üzerinden okuma "\r" 'ye kadar olur ve eğer program bu değeri göremezse program işlemlere devam etmek yerine bu değeri beklemeye geçer. Putc() fonksiyonu kullanılarak haberleşme yapılmak istenildiğinde ise istenilen karakter direkt gönderilir ve getc() fonksiyonu ile karşılanır. Eğer bu yöntem ile bir sayı gönderilmek istenilirse bu sayı en fazla 255 olabilir (Bkz: Tablo 4). Projede olduğu gibi 0 ile 1024 arasında herhangi bir sayı gönderilmek isteniyorsa. Bu sayı ancak 2 karakter ile gönderilebilir. Projede kullanılan yöntem için program algoritması bölümüne bakınız. Eğer 2 kontrolcünün haberleşmesi isteniliyorsa 1. Kontrolcünün PIN_C6 bacağı 2. Kontrolcünün PIN_C7 bacağına ve 1. Kontrolcünün PIN_C7 bacağı 2. Kontrolcünün PIN_C6 bacağına bağlanır böylece birinden gönderilen bilgi 2. Kontrolcü tarafından alınır. Bu sayede haberleşme kolayca yapılabilir. Birçok kontrolcünün aynı anda haberleştiği bir sistemde gönderilen bilginin önüne gönderilen kontrolcünün kodu yazılabilir. Bu

sayede bilgiyi tüm kontrolcüler okur ancak sadece ilgili olan PIC kullanır. Projede Master ve Slave devreleri birbiriyle haberleşmektedir. Gönderilmek istenilen açı değerinin başına “m” ya da “s” karakterinden sonra yollanır. “m” gönderimin Master devresinde olduğunu, “s” ise gönderimin Slave devresine olduğu belirtir. “p” ise master’den slave’e gönderilen PWM değerindir. Bu sayede olası çakışmalarda bilgi kaybı önlenir. Daha karmaşık haberleşmelerde bilgini doğru alınıp alınmadığı da kontrol edilmesi gerekmektedir. Ancak bu projede yapılan deneyler sırasında bahsedilen yöntemden daha ileri bir haberleşme yönteminin kullanılmasına gerek kalmadığı düşünülmüştür.

4.3. Kesmeler

Bazı özel uygulamalarda yoklama (polling) eğer geçerli ve istenilen bir çözüm vermiyorsa kesme (interrputs) mekanizması kullanılabilir. Kesmeler kullanılan algoritmaya alternatif bir algoritma eklenebilmeyi sağlar. Programcı tarafından belirlenen bir işlem olduğunda kesmeler sayesinde algoritma dallanıp algoritma başka bir rotayı izleyebilir. Bu esnada gerçekleştirilen işlem kesme algoritmasında sürülen işlemi bekler ve sonunda program kaldığı yerden ana rotasına geri döner. Birçok çeşit kesmeler mevcuttur. Bunlardan zamanlayıcı kesmesi PWM/CCP kesmesi, ADC ve RDA kesmesi en çok kullanılanlardandır.

Yazılan programın seri portdan bir karakter gelene kadar normal bir şekilde ana döngüsünde devam etmesi isteniyorsa RDA kesmesi kullanılabilir. Bu sayede kesme rotasında gelen karakter tanımlanmış global bir değişkene atanıp, bu değer ana döngüde tekrar kullanılabilir. Ayrıca ana döngüden bayrak kullanılarak bloklar halinde çalışan ana döngüde seyreden program kesmelerle alınan değerler ile bu bayrak değişkeni değiştirilebilir ve böylece gelen karaktere göre program istenilen bloklara girebilir. Bazı özel uygulamalarda ve veri gönderiminin birkaç milisaniye alabileceği düşük hızlarda (<9600 baudrate) karakter gönderimi için de kesme kullanılabilir. Böylece seri portdan istenilen değişken Mikro denetleyiciyi engellemeden verici kesmesiyle haberleşme sağlanabilir.

CCS uygulamalarında kesme rotasında izlenecek fonksiyon *#int_xxx* ifadesinden sonra yazılır. RDA kesmesi için örnek bir kod görülebilmektedir.

```
#INT_RDA
void rs232_isr()
{
adval=getc();
}
```

Kesmeleri geçerli kılmak için bazı kesme bitleri ayarlanmalıdır.

```
enable_interrupts(INT_RDA);
enable_interrupts(GLOBAL);
```

Projede Master ve Slave devrelerinin birbirleriyle haberleşmesi RDA kesmesiyle yapılmaktadır. 192000 baudrate hızında çalışıldığından ve arka arkaya üç karakter gönderileceği için verici kesmesi kullanılmamıştır. Gönderim ana döngü içerisinde uygun anlarda yapılmaktadır. Ancak veri alınması için ana döngüden çıkılıp kesme rotasına girilmesi gerekmektedir.

4.4. Program Algoritması

Geliştirilen PIC programı, Master kolunda bulunan 4 adet step motoru, Slave kolunda bulunan 5 adet step motoru ve bu motorlar arasındaki haberleşmeyi kontrol eder. Akış Diyagramı Ek4 ve Ek5'de mevcuttur.

Akış diyagramlarından da görülebileceği gibi, sistemin çalışması master kolunun hareketiyle olur. Operatör Master kolunu hareket ettirdiğinde. Bir döngü içerisinde master motorlarına bağlı bulunan ayarlı dirençten veri okunur. Bu veriler 20ms'de bir güncellenir. Bu aşamada program okunan verinin üstüne tekrar okuma yapmaktan kaçınır ve açı ve eski açı olarak 2 adet değişken kullanır. Açı değişkeni o anda okunan açı değeridir. 20 ms'ye sonra açı değişkeni eski açı değişkenine atanır ve okuma baştan yapılır. Her seferinde açı ve eski açı arasındaki fark kontrol edilir. Eğer fark istenilen hassasiyet değerinden büyükse açısal değişim algılanır. Bu değişim

değerini slave kolundaki açısal değişimin fark edildiği ayarlı direnç'e a karşılık gelen devreye gönderir.

Açısal veriler 0 – 1024 değerleri arasında okunmaktadır. Bu değer PIC16F877A ile gönderebilmek için "*printf("%ld", aci);"* komutu yeterli olmaz. Çünkü "printf" ile gönderilmek istenilen veri 0 ila 256 değerli arasında olabilmektedir. Bu nedenle basit bir haberleşme protokolü yazılmıştır. Açısal değer binler – yüzler ve onlar – birler basamakları olarak ikiye ayrılır. Bu sayede 2 basamaklı 2 adet sayı elde edilir. Bu sayılara karşılık gelen karakter değerleri başına gönderilecek devrenin adının baş harfi ile birlikte ardı ardına yollanır. Slave alıcı devreleriyle alınan bu değerler tekrar birleştirilir. Bu sayede 1024 değerinin 256 ya indirgenmesiyle oluşacak veri kaybı önlenir ve "putc()" komutunun 2 kere kullanılmasıyla haberleşme sağlanabilmiş olur. Örneğin gönderilecek veri 987 olsun. Bu değerin mod 10'u onlar ve birler basamağından oluşan sayıyı verir. Bu sayı b=87. 100'e bölümü ise Varsa binler ve yüzler basamağından oluşan sayıyı verir. Bu sayı ise a = 9. A ve b değişkenlerine karşılık gelen karakterler.'TAB' + 'W' karakter olarak gönderilir. Bu gönderim sırasında daha sonra değinilecek olan geribesleme haberleşmesi ile karışmaması amacıyla başına slave'e gönderildiğini gösteren 's' karakteri konulur. Böylece 's' + 'TAB' ve 'W' karakteri slave devresine gönderilir.

RX portunu bir döngü içerisinde dinleyen slave motorlarının PIC programı ise porttan aldığı bir kesme ile okumaya geçer. İlk okuduğu karakter kendisine gönderildiğini gösteren 's' karakteri ise, ardından gelen 2 karakteri okur. Eğer değilse okuma modundan çıkıp tekrar port'u dinlemeye devam eder. Okunan 2 karakterin ilki 100 ile çarpılarak diğeri ile toplanır ve böylece gönderilen veri alınmış olur. Bu veri ilgili slave motorunun gitmesi gereken açıdır. Bu nedenle motor kontrol fonksiyonuna değer yollanır. Motor kendi açısını okur ve gitmesi istenilen açı ile karşılaştırır. Bu karşılaştırma sonucunda motorun yönü belli olur. O yöne dönmesi için gereken komut sırası ile verilir ve motor adım adım dönmeye başlar.

Slave motoru bir adım dönmeden önce bulunduğu açı değerini okur. Daha sonra 2ms'de bir adım dönen motorun her adımından sonraki açı değeri okunur. Bu okuma işlemi ayarlı dirençten yapılmaktadır ve bu okuma sırasında belli bir zamana ihtiyaç

duyulmaktadır. Denemeler ile yapılan sonuçlara göre 200 okumada bir yapılan karşılaştırmanın en iyi sonucu verdiği görülmüştür. Bu nedenle okuma sırasından arttırılan bir sayaç yardımı ile sayacın 0 ve 200 değerlerindeki veriler karşılaştırılır. Eğer bu veriler aynı ise, motorun hareket edemediği anlaşılır ve bulunduğu açı değeri b maddesinde bahsedilen yöntemle bu kez başına master'a gönderildiğini gösteren 'm' harfi ile beraber yollanır. İşte bu gönderim sırasında masterdan bir veri aynı anda alınmak isterse doğabilecek hatalardan kaçınmak amacıyla verinin gönderileceği robot kolunun baş harfe verinin başına konulmaktadır. Okunan veri kendi harfi ile gelmiyorsa gönderim sağlanmaz.

Master kolu a maddesinde belirtildiği gibi bir döngü içerisinde açı değerlerini okur ve bu değerleri karşılaştırarak bir fark olup olmadığını kontrol etmektedir. Bu esnada Slave kolunun istenilen konuma gidemeyip engellendiği açı değeri master koluna kesme olarak gelir. Bu kesmeyi önce kendisine gelip gelmediğini kontrol edilir. Kendisine gelen veri bu kez master kolunun gitmesi istenilen açı değerine eşitlenir. Kesme bayrak olarak kullanılan bir değişkenin değiştirilmesi ile master koluna gelen geribesleme fonksiyonuna girmesi sağlanır ve master kolu bu sefer slave kolu gibi hareket etmeye başlar. Operatörün tuttuğu master kolu kendisinin yönlendirdiği noktadan, slave kolunun engellenmiş olduğu açıya geri dönmek istediğinde ise operatöre karşı bir kuvvet uygular.

Operatöre uygulanan bu kuvvet esnasında slave kolu, karşılaştığı engeli aşmak için üretilen darbelerin (pulselerin) genişliklerinin kontrol edilerek (PWM - Pulse Width Modulation) motorun uyguladığı orantılı bir şekilde arttırılır. Normal bir kontrol sırasında program 512 PWM ile çalışmaktadır. 512'den doğru orantılı bir biçimde 1024'e artırım sırasında aynı artış master kolunda da olmaktadır. İstenilen hassasiyet oranlarına göre hesaplanan zamanlarda gerçekleşen bu artış sırasında operatör geribesleme orantılı olarak hisseder. Engel aşılana kadar bu işlemler devam eder. İşlemin sonlanması için Master kolunun slave kolunun gidemediği noktaya geri dönmesi veya slave kolunun karşılaştığı engeli aşması gerekmektedir. Bu iki durumdan herhangi biri olduğunda açılar eşitlenir. Master kolu ile Slave kolu tekrar aynı açıya geldiğinde her şey başa döner.

5. Kullanılan Robotun Özellikleri

Pioneer firması tarafından üretilen P3-DX olarak adlandırılan çok amaçlı gezgin robot bir çok üniversitedeki araştırmalar için kullanılmaktadır. Alüminyum gövdeye sahip bu robotun boyutları 44cm x 38cm x 22cm'dir. Diferansiyel sürüşe sahip gezgin robotun hareketini sağlayan, motorlara bağlı robotun ön kısmına monte edilmiş 16.5 cm çapında 2 adet tekerleği bulunmaktadır. Ayrıca arakada denge sağlamak için kullanılan 6 cm çapında bir küçük tekerleği vardır. Pioneer 3-DX ulaşabileceği en yüksek hız 1.6 m/s' dir. Gezgin robotun hareketini sağlayan iki adet motor bulunmaktadır. Bu motorların bağlı olduğu iki adet 38.3:1 dişli oranına sahip olan dişli kutuları bulunmaktadır. Ayrıca robotun kendi konumunu belirlemesi için konum bilgisini oluşturan 500 pulse'lı enkoder bulunmaktadır. Diferansiyel sürüş platformu ile iki tekerleği birlikte çevirir veya 32 cm yarıçapında duran tekerleğin etrafında dönme yapabilir. Robot %25 derece kalkarak 2.5 cm yüksekliğindeki eşiği atlayabilir. Gezgin robotun enerji ihtiyacını karşılamak için robotun içine monte edilmiş üç adet akü bulunmaktadır. Bunlara ilave olarak 180 derece önü çevreleyen 8 adet ön ultrasonik mesafe algılayıcısı ve 180 derece arkayı çevreleyen 8 adet arka ultrasonik mesafe algılayıcısı bulunmaktadır. Ultrasonik mesafe algılayıcıları yaklaşık 15 cm den 7m ye kadar okuyabilmektedir. Kullanılan robotun arka tarafında 5 adet tampon bulunmaktadır. Tampona 100 gram basınç uygulandığında bir girdi olarak algılanır. Tampon panelindeki her tamponun boyutu 10.0 cm x 2.5 cm x 1.0 cm dir.

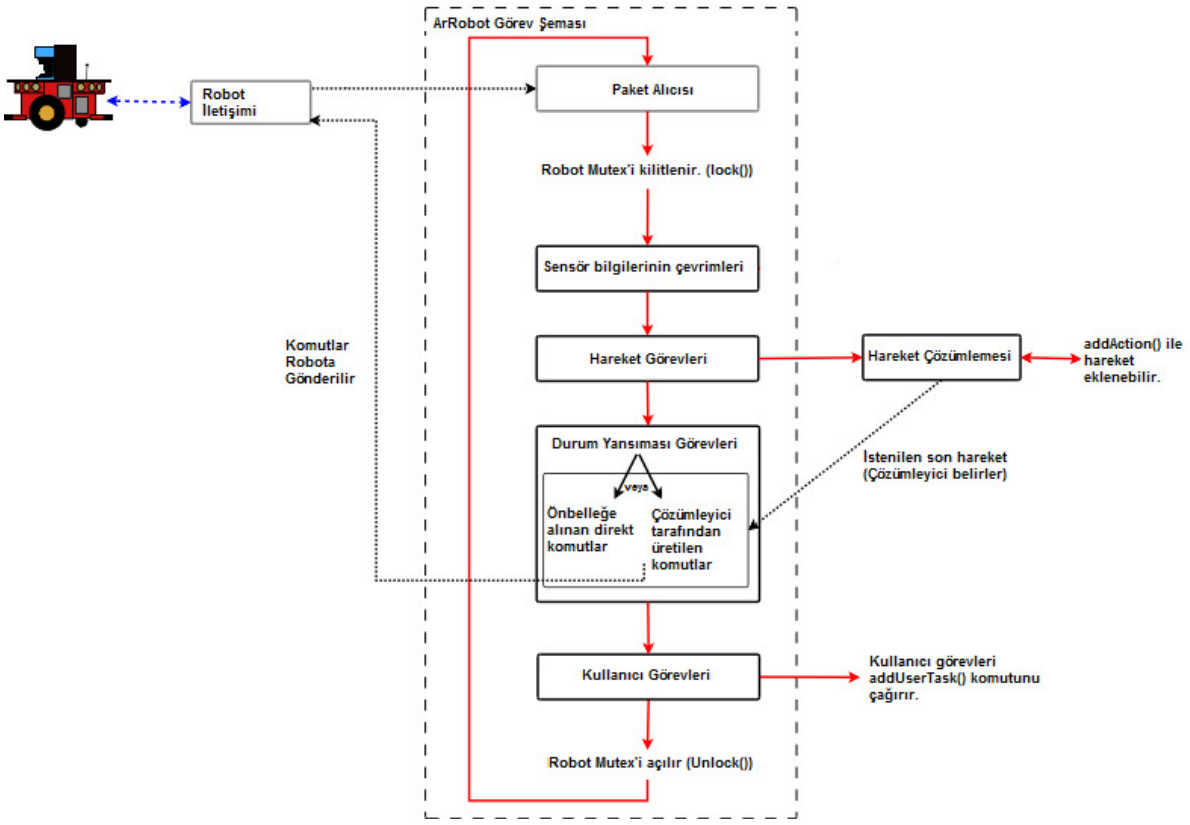
Operatör ile iletişim kablosuz modem ile sağlanır. Ayrıca ana makine ve gezgin robot arasında robota bazı komutları yükleyebilmek için TCP/IP protokolünü kullanan bir yazılım geliştirilmiştir [5].

5.1. YAZILIMLAR

5.1.1. ARIA

ARIA, Pioneer ActivMedia gezgin robotların kontrol uygulamaları için C++ dilinde yazılmış nesne yönelimli arayüz programıdır. Aria, robotun kontrolü sensörlerin kullanımı kolaylaştırmak amacıyla yazılmış birçok kullanışlı özelliğe sahip Linux ve Windows platformlarında kullanılabilen bir programdır. ARIA farklı düzeylerde kontrol sağlar. ArRobot sınıfı ile robota direkt komut gönderilebilir ya da "*Actions"* sınıfı ile daha karmaşık ve yüksek seviyedeki kontroller yapılabilir. (Bkz: Şekil 43)

ArNetworking yardımcı kütüphanesi ile bağlı bulunan ağlardan iletişim kurulabilmekte ve bu programlarla uzaktan kontrol sağlanabilmektedir. Bunun dışında ek olarak birçok kütüphane bulunmaktadır. Ses tanıma, kaydetme, video kaydetme, renk takibi gibi birçok konuda kontrolü kolaylaştırmak amacıyla kütüphaneler mevcuttur [15].



Şekil 43 – ArRobot Görev Şeması[15]

ARIA esnek bir yapıya sahiptir. Tek davranışı gerçekleştirebildiği gibi bir çok davranışı da gerçekleştirebilmektedir. ARIA davranışların öncelik sırasına göre karar verir.

ARIA'nın Anahtar Özellikleri

- Hız, yönelme, bağıl yönelme dinamik olarak kontrol edilir.
- Kullanıcı Giriş / Çıkış yüzeyini, tutucuları, kamerayı sağa-sola, yukarı aşağıya çevirmeyi ve diğer aksesuarları bütünleştirir
- Bütün mesafe araçlarını tek bir fonksiyonda sorgular.
- Yüksek esnekliğe sahiptir
- Çapraz veritabanı kullanır (Linux ve Win 32)
- Kaynak dokümanları içinde mevcuttur.
- Çok yönlü uygulamalar için davranış sistemli planlama kurar.
- Engelden kaçınma için davranış oluşturabilirler [5].

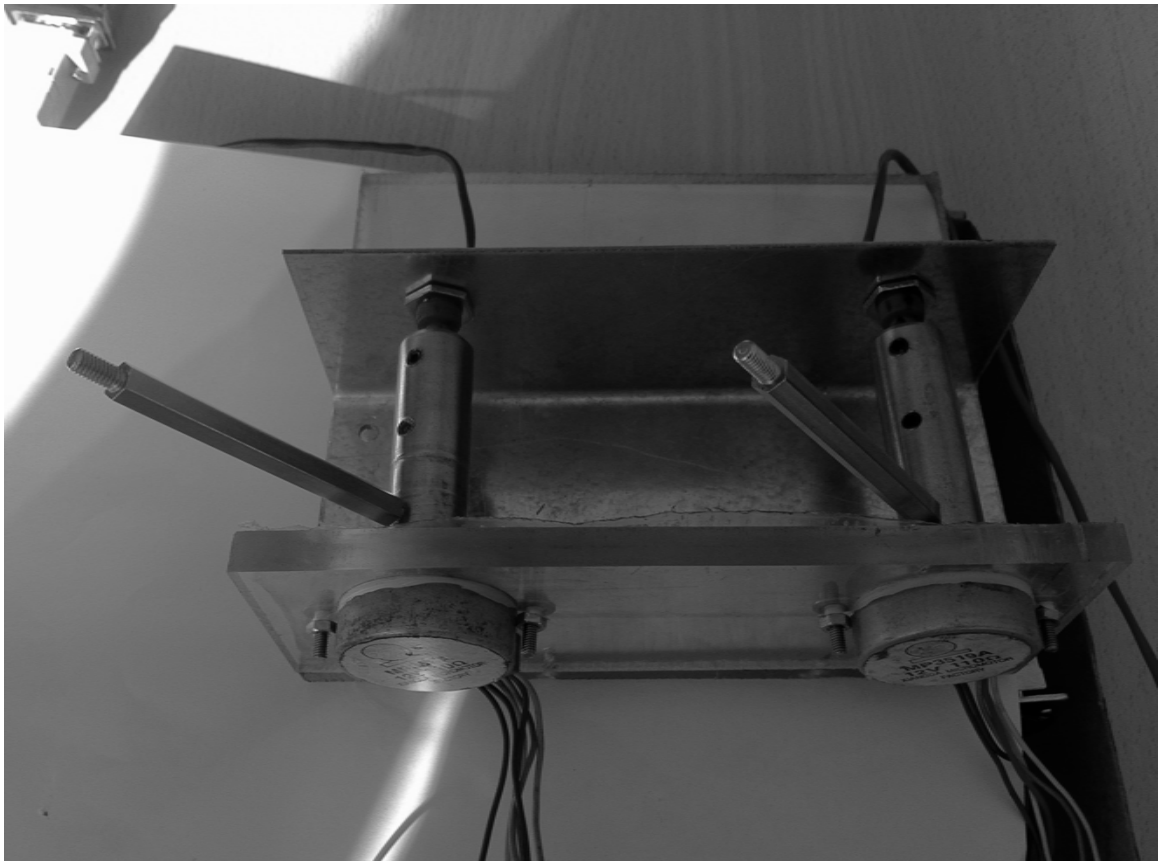
5.1.2. Simülatör

ARIA ile yazılan programlar robota yüklenip gerçek ortamda denenmeden önce hataları azaltmak amacıyla *MobileSim* adlı simülatör programı ile yazılan kodlar denenir. *SRIsim* de aynı görevi yapmaktadır fakat MobileSim simülatörü, SRIsim'e göre daha üstün özellikleri vardır. MobileSim, Richard Vaughan tarafından geliştirilen Stage simülatörü temeline dayanır. MobileSim çevredeki duvarları ve engelleri simule etmek için çizgi verilerini kullanır. "Mapper" veya "Mapper3-Basic" programı ile çizgi temelli harita oluşturulur. Mapper, SRIsim ile birlikte çalışan bir çizim programıdır. Robotun çalışma ortamı için dünya modeli yaratılır. ".wld" uzantılı dosyadaki çizgi verilerini kullanır. Mapper3 – Basic ise MobileSim ile birlikte çalışmaktadır, ".map" uzantılı dosyadaki çizgi verilerini kullanmaktadır.

6. ELDE EDİLEN VERİLERİN DEĞERLENDİRMELERİ

6.1. TEK EKLEM İLE YAPILAN DENEYLER

İlk olarak algoritmanın test edilmesi ve olası bağlantılardan kaynaklanabilecek veri kayıplarını aza indirgemek amacıyla tek eklem ile testler yapılmıştır. Bu deneyler için 2 adet adım motoru kullanılarak bir deney düzeneği tasarlanmıştır. Şekil 44'de görülen düzenekte, ayarlı direnç ile motor milinin bağlantısından kaynaklanan hataları önlemek için, motor mili ve ayarlı direnç bağlantıyı sağlayacak olan parçaya sıkı geçme ile bağlanmış ve ayrıca 2 adet civata ile sabitlik arttırılmıştır. 60mm uzunluğundaki bir eklem bağlantı elemanına sabitlenmiştir.

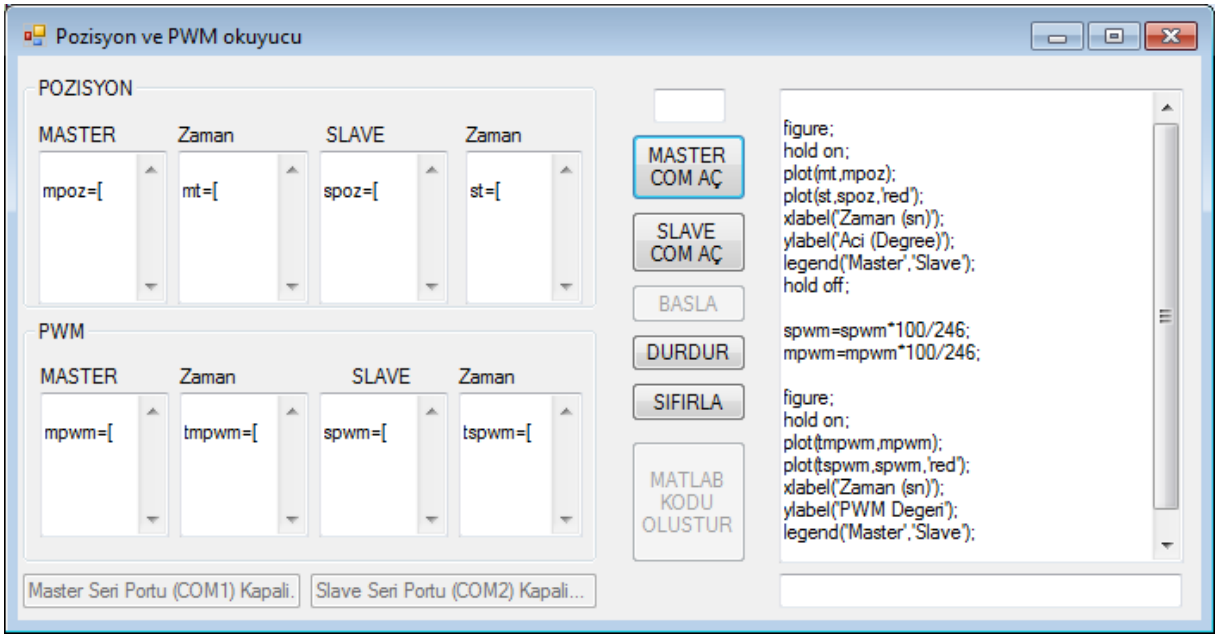


Şekil 44 – Tek Eklem için Deney Düzeneği

Hareket sırasında elde edilen verilerin bilgisayara kaydedilmesi için mikro denetleyicilerin birbirleri ile haberleşmesinde C6 ve C7 bacakları yerine D2 ve D3 bacakları kullanılmıştır. D2 ve D3 bacakları kullanılarak mikro denetleyiciler arasında gönderilen veriler aynı anda C6 ve C7 bacaklarından bilgisayara gönderilmiştir.

Ayrıca daha çok veri elde edebilmek konum ve PWM değerleri haberleşmenin olmadığı anlarda da belirli zaman aralıklarında bilgisayara gönderilmiştir. Veriler 0 ile 260 derece aralığında gönderilirken, PWM değerlerinin yüzde karşılıkları gönderilmektedir.

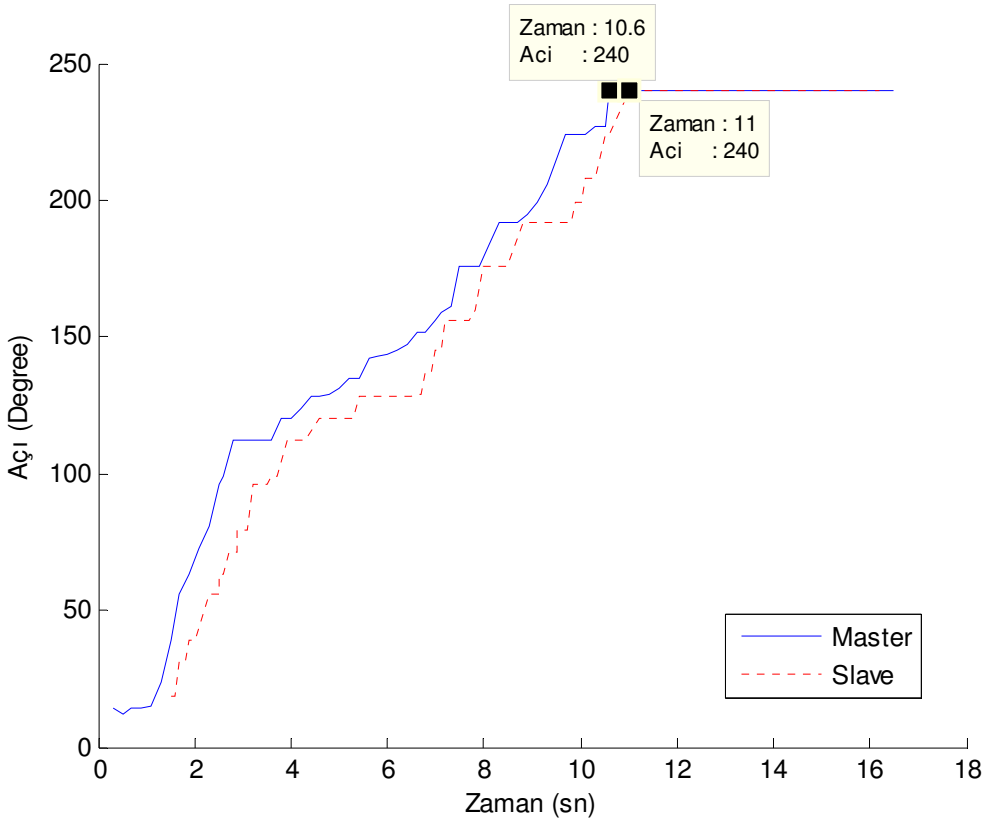
Mikro denetleyicilerin gönderdiği bu verileri gönderildikleri zaman ile birlikte kaydedebilmek için C# ile bir program yazılmıştır.



Şekil 45 – Pozisyon ve PWM okuyucu programı

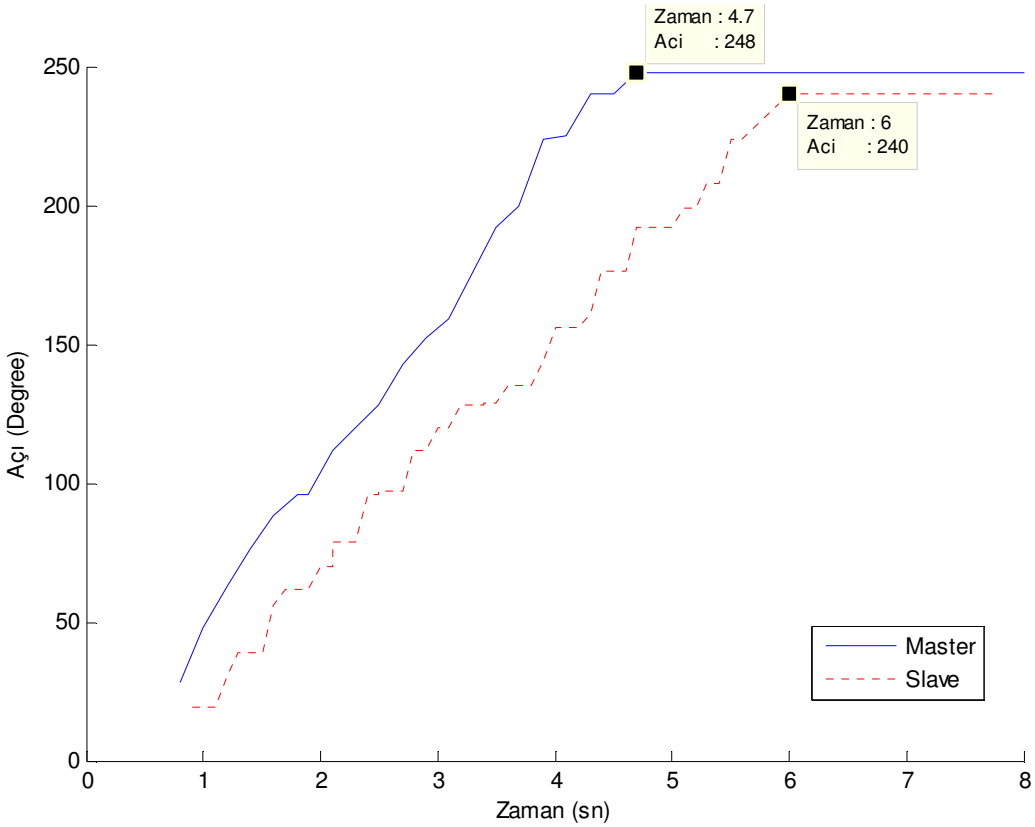
Şekil 45'de görülen ara yüze sahip bu program Master ve Slave kollarının pozisyon değerlerini kullanıcı tarafından başlatılabilen zaman sayacı sayesinde 100ms'lik aralıklar ile kaydedebilmektedir. Mikro denetleyicileri birbirleri ile yaptıkları haberleşmede olduğu gibi, açı değerlerinden önce "a", PWM değerlerinden önce ise "p" karakteri gönderir. Alındaan bu karaktere göre açı veya PWM değeri doğru alana yazılır. Deney sonuçlandıktan sonra, "DURDUR" düğmesi ile zaman durdurulur ve alınan değerler MATLAB programında kullanılmasını kolaylaştırmak için köşeli parantez ile kapanır. Kaydedilen veriler kullanılarak grafiklerin çizilmesi için MATLAB kodları sağ tarafta bulunun yazı bölümüne girilebilmektedir. Matlab kodu oluştur düğmesi ile kaydedilen veriler ve zaman değerleri matrisler halinde ".m" uzantılı

olarak bilgisayara kaydedilir. Daha sonra bu dosya çalıştırılarak grafikler elde edilebilir.



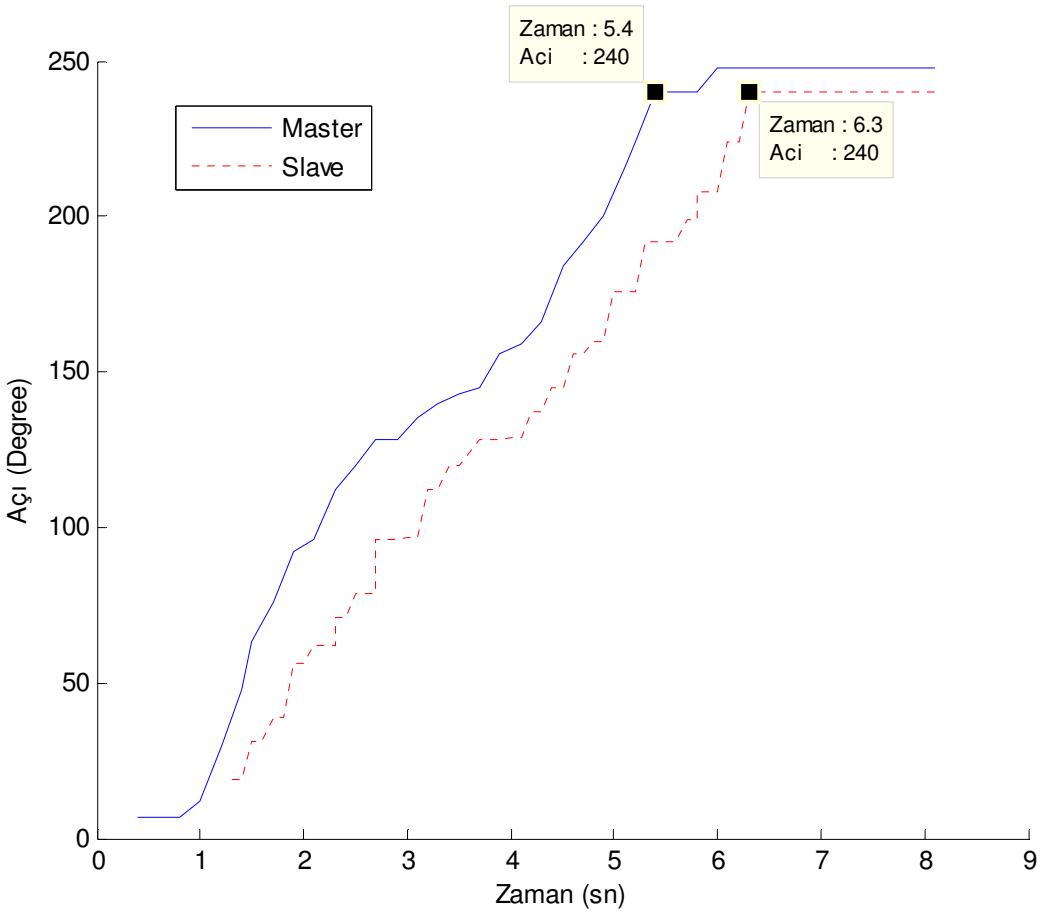
Şekil 46 - Tek eklemde pozisyon analizi 1

Yapılan deney esnasında Master ve Slave eklemlerinin açısal pozisyonları C# da yazılan program sayesinde kaydedilmiştir. Master ve Slave eklemleri ayrı COM portlarından bilgisayara bağlandığı için veriler birbirinden bağımsız olarak alınabilmiştir. Pozisyon değerleri alındıkları zaman değerleri ile birlikte kaydedilerek Matlab programı ile grafikleri çizilmiştir. Çizilen grafik Şekil 46'de görülebilmektedir. Deneyde, Master eklemi operatör tarafından ortalama 0.38 rad/sn'lik hız ile hareket ettirilmiş ve buna karşılık Slave ekleminin cevabı izlenilmiştir. Hareket esnasında Slave eklemi yavaşlatıcı herhangi bir engel ile karşılaşmamıştır. Master kolunun 10.3 saniyede yaptığı 226 derecelik açısal harekete karşılık Slave kolu bulunduğu konumdan harekete başlamış ve Master kolunun durduğu 240 derecedeki konumuna 0.4 saniye gecikme ile 11. saniyede ulaşmıştır. Bu hareket sırasında Slave kolunun ortalama hızı 0.35 rad/sn olarak ölçülmüştür.



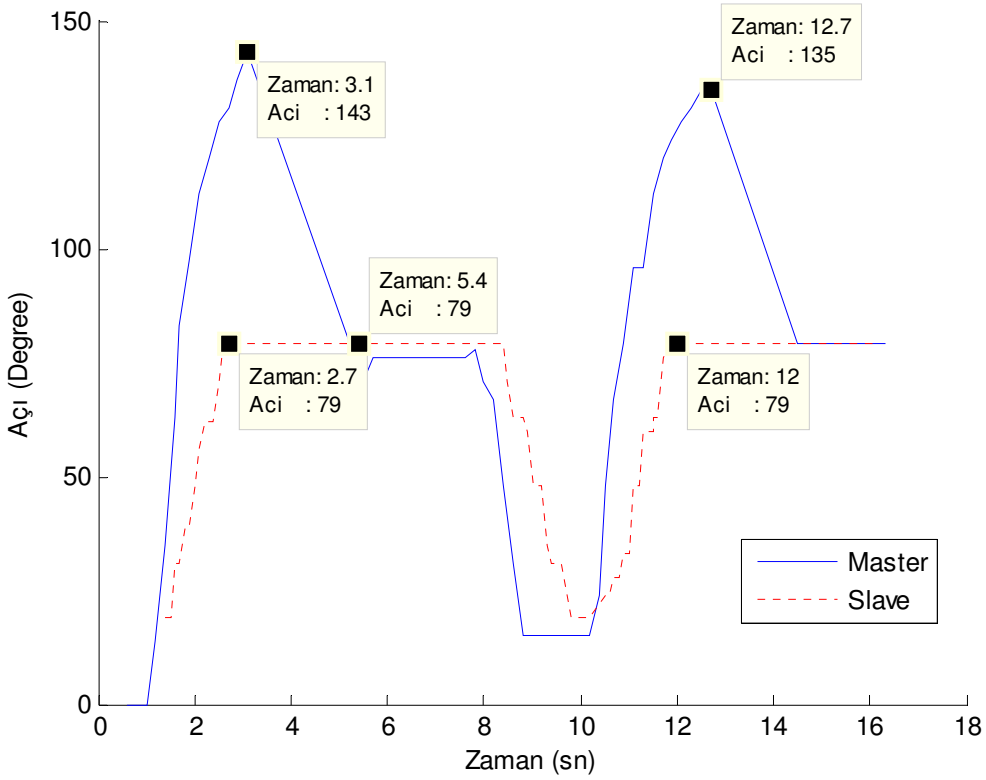
Şekil 47 - Tek eklemde Pozisyon analizi 2

Yapılan deney hızlandırılmış, Master eklemi operatör tarafından ortalama 0.82 rad/sn'lik hız ile hareket ettirilmiş ve buna karşılık Slave ekleminin cevabı izlenilmiştir. Hareket esnasında Slave eklemi yavaşlatıcı herhangi bir engel ile karşılaşmamıştır. Master kolunun 4.7 saniyede yaptığı 226 derecelik açısal harekete karşılık Slave kolu bulunduğu konumdan harekete başlamış ve Master kolunun durduğu 240 derecedeki konumuna 1.3 saniye gecikme ile 6. saniyede ulaşmıştır. Bu hareket sırasında Slave kolunun ortalama hızı 0.64 rad/sn olarak ölçülmüştür. Alınan veriler ile çizilen Şekil 47'de görülebilmektedir.



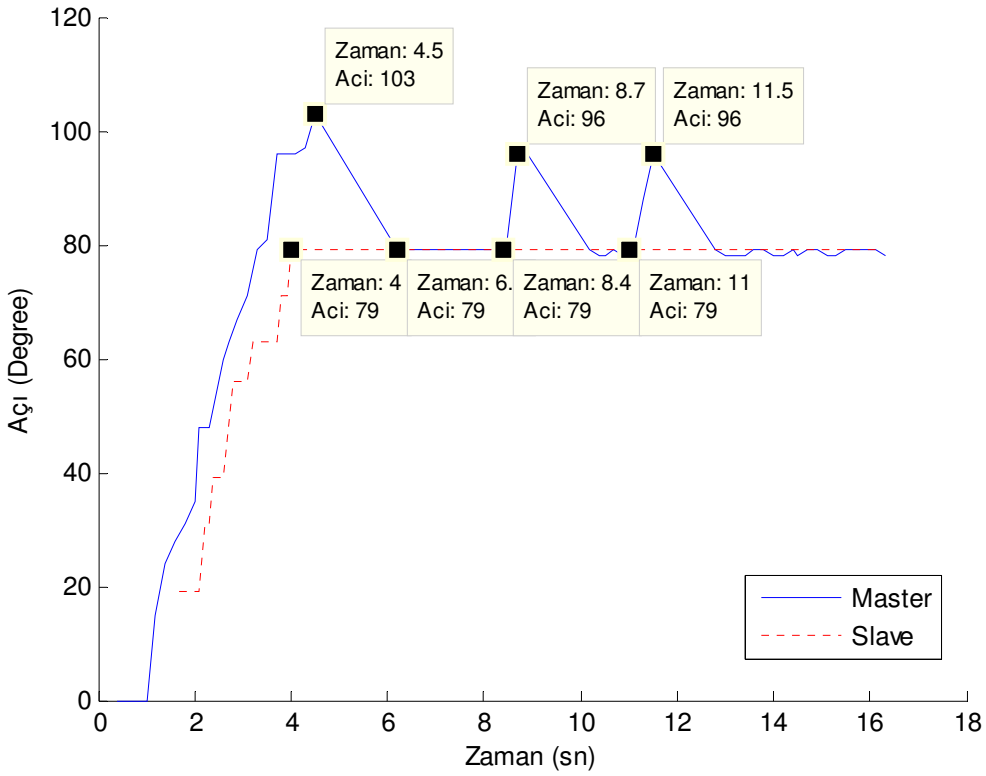
Şekil 48 - Tek eklemde Pozisyon analizi 3

Şekil 48'de yapılan deney esnasında pozisyon değerleri ve zaman değerleri ile çizilen grafik görülebilmektedir. Deneyde, Master kolu operatör tarafından ortalama 0.79 rad/sn'lik hız ile hareket ettirilmiş ve buna karşılık Slave kolunun cevabı izlenilmiştir. Hareket esnasında Slave kolu yavaşlatıcı herhangi bir engel ile karşılaşmamıştır. Master kolunun 5.4 saniyede yaptığı 226 derecelik açısal harekete karşılık Slave kolu bulunduğu konumdan harekete başlamış ve Master kolunun durduğu 240 derecedeki konumuna 0.9 saniye gecikme ile 6.3. saniyede ulaşmıştır. Bu hareket sırasında Slave kolunun ortalama hızı 0.60 rad/sn olarak ölçülmüştür.



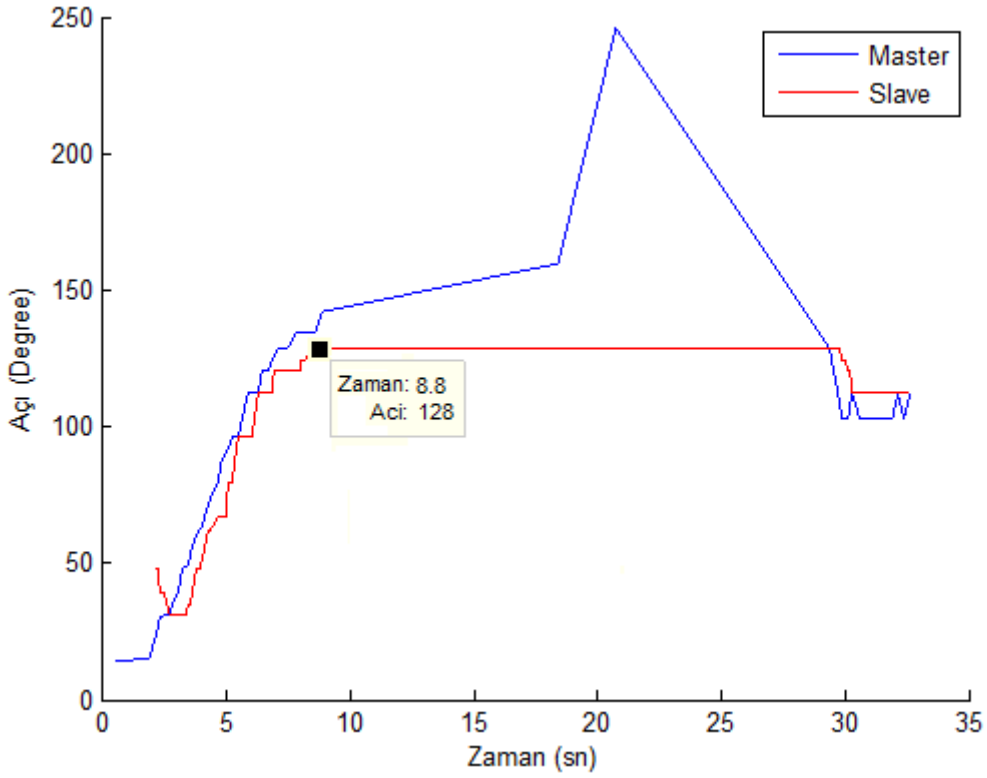
Şekil 49 - Engel ile yapılan Pozisyon analizi 1

Tek eklemde Engel ile pozisyon analizi için yapılan deney esnasında Master ve Slave kollarının açısal pozisyonları C# da yazılan program sayesinde kaydedilmiştir. Pozisyon değerleri alındıkları zaman değerleri ile birlikte kaydedilerek Matlab programı ile grafikleri çizilmiştir. Çizilen grafik Şekil 49'da görülebilmektedir. Deneyde, Master kolu operatör tarafından hareket ettirilmiş ve buna karşılık Slave kolunun cevabı izlenilmiştir. Hareket esnasında Slave kolunun hareket yörüngesinde kolunun hareketini tamamen durduracak bir engel yerleştirilmiştir. Master kolunun bulunduğu konuma giden Slave kolu, açısı 80 derece olduğunda engele takılmıştır. Takıldığı konum Master koluna yollanmış ve Master kolu serbest bırakılmıştır. Slave kolunun hareketinin engellenmesinden 0.5 saniye sonra Master kolu hareketi operatöre hissettirmek için, ters yöndeki hareketine başlamıştır. Engeli aşmak için bir kere daha bu hareket tekrarlanmış ve bu sefer Master kolunun cevabı 0.8 saniye sonra olmuştur.

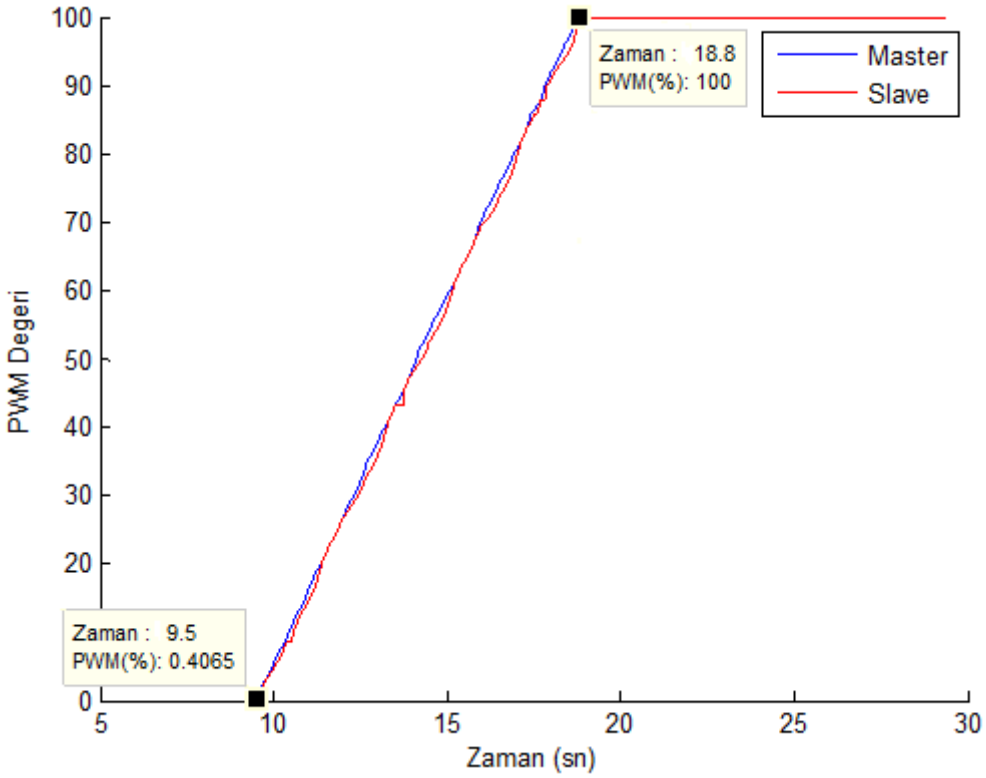


Şekil 50 - Engel ile yapılan Pozisyon analizi 2

Bir başka deneyde, Master kolu operatör tarafından hareket ettirilmiş ve buna karşılık Slave kolunun cevabı izlenilmiştir. Hareket esnasında Slave kolunun hareket yörüngesinde kolunun hareketini her 2 yönde tamamen durduracak bir engel yerleştirilmiştir. Master kolunun bulunduğu konuma giden Slave kolu, açısı 80 derece olduğunda engele takılmıştır. Takıldığı konum Master koluna yollanmış ve Master kolu serbest bırakılmıştır. Slave kolunun hareketinin engellenmesinden 0.5 saniye sonra Master kolu hareketi operatöre hissettirmek için, ters yöndeki hareketine başlamıştır. Eğer operatör master kolunu serbest bırakmamış olsaydı bu ters hareket esnasında slave kolunun bir cisme temas ettiğini algılayabilmiş olacaktı. Daha sonra Master kolu engeli aşmak için iki kere daha bu hareket tekrarlanmış ve Master kolunun cevabı ilk tekrarlamada 0.5 ve ikinci tekrarlamada ise 0.2 saniye sonra olmuştur. Bu hareket sırasındaki pozisyon değişim grafiği Şekil 50'de görülebilmektedir.

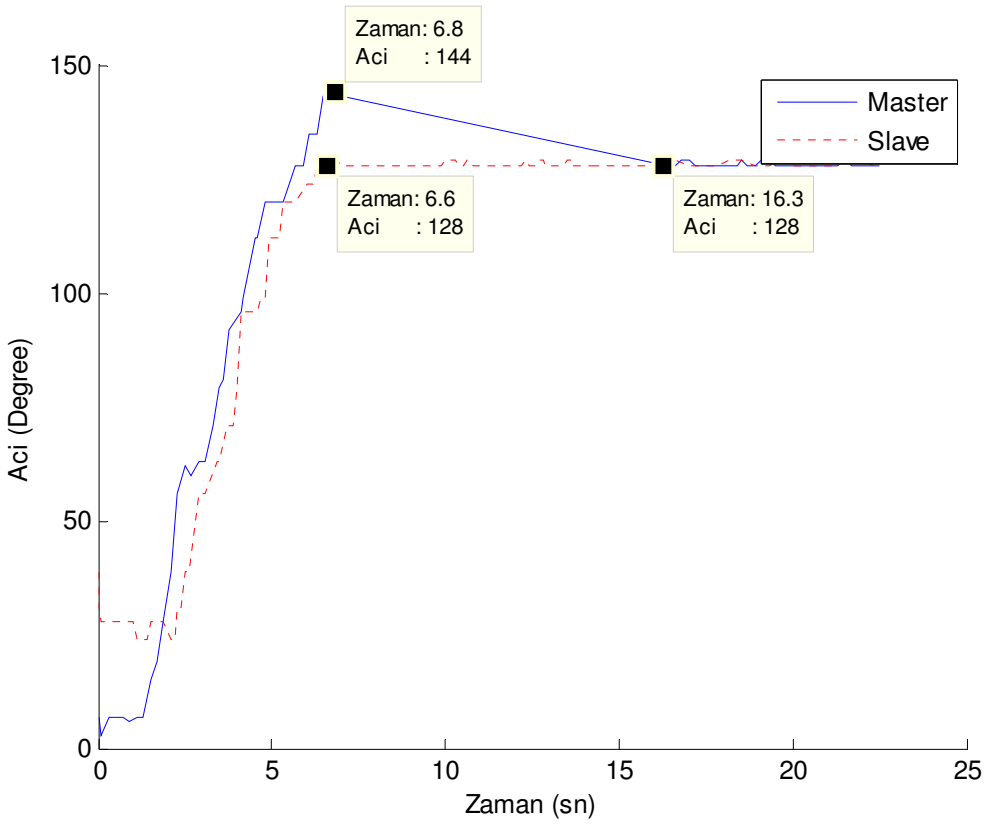


Şekil 51 – Tek Eklemde 1. Kuvvet Analizi Pozisyon Grafiği



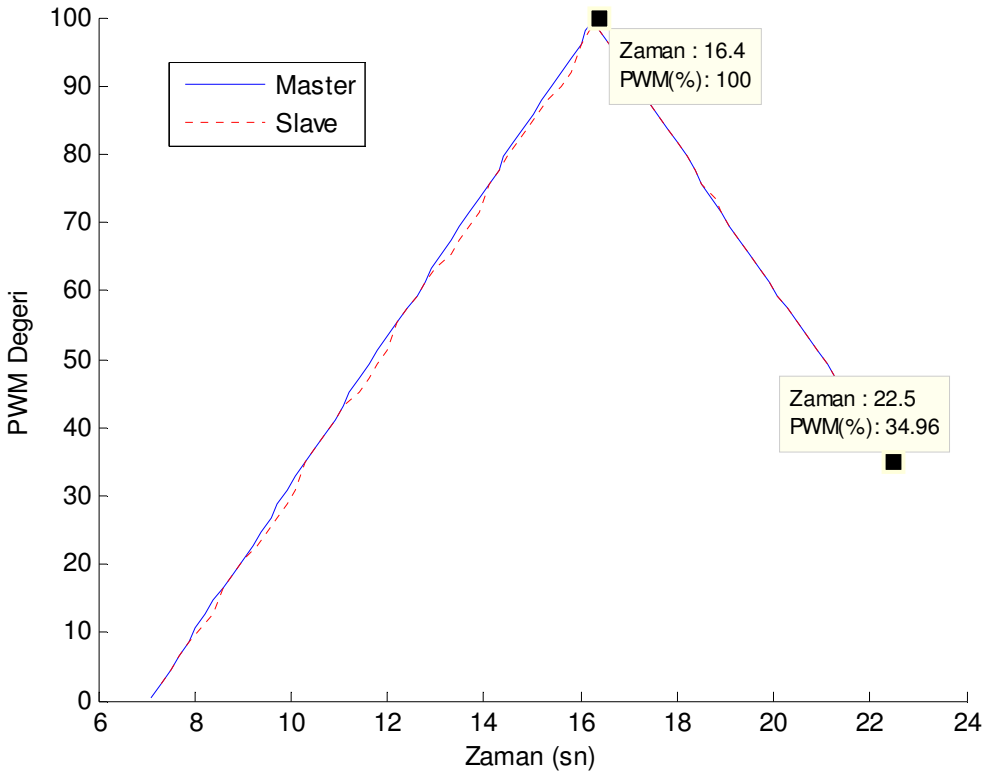
Şekil 52 – Tek Eklemde 1. Kuvvet Analizi PWM Grafiği

Kuvvet analizi için yapılan deneylerde ise pozisyon değerleri yanı sıra Slave kolunun bir cisme teması sırasındaki PWM değerleri de kaydedilmiş ve grafikleri çizilmiştir. Şekil 51'de görüldüğü gibi slave kolu 8.8 saniye sonra 128 derecede bir engele takılmış ve açısı sabitlenmiştir. Bu esnada hareket ettirilen master kolunun açısal hızında azalma gözlenmiştir. Şekil 52'da görülebileceği gibi Slave kolunun engele temas etmesinden 0.7 saniye sonra PWM değeri minimum değerden artmaya başlamıştır. PWM değeri Master ve Slave kollarında değişmesi ile Slave kolunda uygulanan ve Master kolunda ters yönde operatöre uygulanan kuvvet değeri eşit biçimde artmaya başlamıştır. 9.7 saniye sonra PWM değeri çıkabileceği en yüksek konuma erişmiş ve bu değerde sabitlenmiştir. PWM değeri en yüksek konumda olmasına rağmen Slave kolu aynı sürede engeli aşamamıştır. PWM değeri en yüksek değerinde olduğundan motor tarafından uygulanan kuvvet en fazla 2.06 N olarak ölçülmüştür. Bu ölçüm robot kolunun Dikomsan HT-300 hassas tartıda çıkabileceği maksimum ağırlık ile hesaplanmıştır.



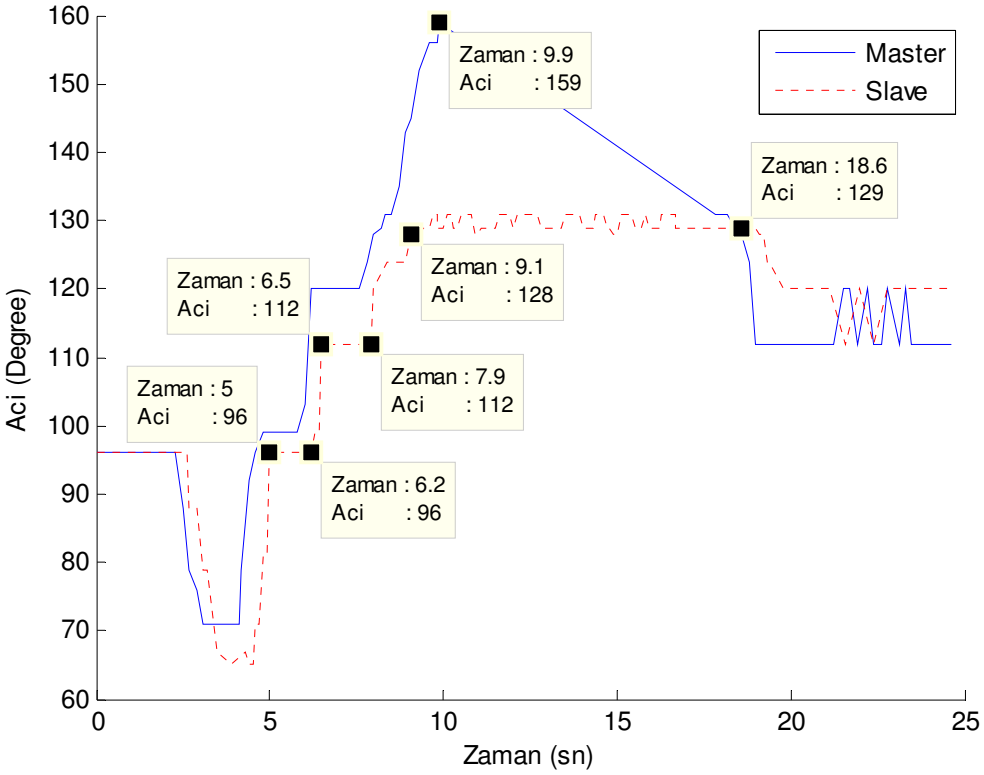
Şekil 53 – Tek Eklemde 2. Kuvvet Analizi Pozisyon Grafiği

Tek eklemde 2. Kuvvet analizi slave kolunun hareket yörüngesinde 128 dereceye gelen konuma yerleştirilen engel ile yapılmıştır. Şekil 53'de görüldüğü gibi slave kolu 6.6 saniye sonra 128 derecede bir engele takılmış ve açısı sabitlenmiştir. Bu esnada hareket ettirilen master kolunun açısal hızında azalma gözlenmiştir. Şekil 54'de görülebileceği gibi Slave kolunun engele temas etmesinden 0.5 saniye sonra PWM değeri minimum değerden artmaya başlamıştır. PWM değeri Master ve Slave kollarında değişmesi ile Slave kolunda uygulanan ve Master kolunda ters yönde operatöre uygulanan kuvvet değeri eşit biçimde artmaya başlamıştır. 9.3 saniye sonra PWM değeri çıkabileceği en yüksek konuma erişmiş ve bu anda master kolu slave kolunun pozisyonuna getirilerek kuvvet artışı durdurulmuştur. Böyle bir hareket esnasında robot kolu bir ağırlık taşıyabileceği düşünülerek PWM değeri bir anda azaltılmak yerine belli zaman aralıklarında kademeli olarak azaltılmaya başlamıştır. PWM değeri en yüksek değerinde olduğundan motor tarafından uygulanan tork değeri en fazla 0,1236 Nm olarak ölçülmüştür. Bu ölçüm robot kolunun Dikomsan HT-300 hassas tartıda çıkabileceği maksimum ağırlık ile hesaplanmıştır.



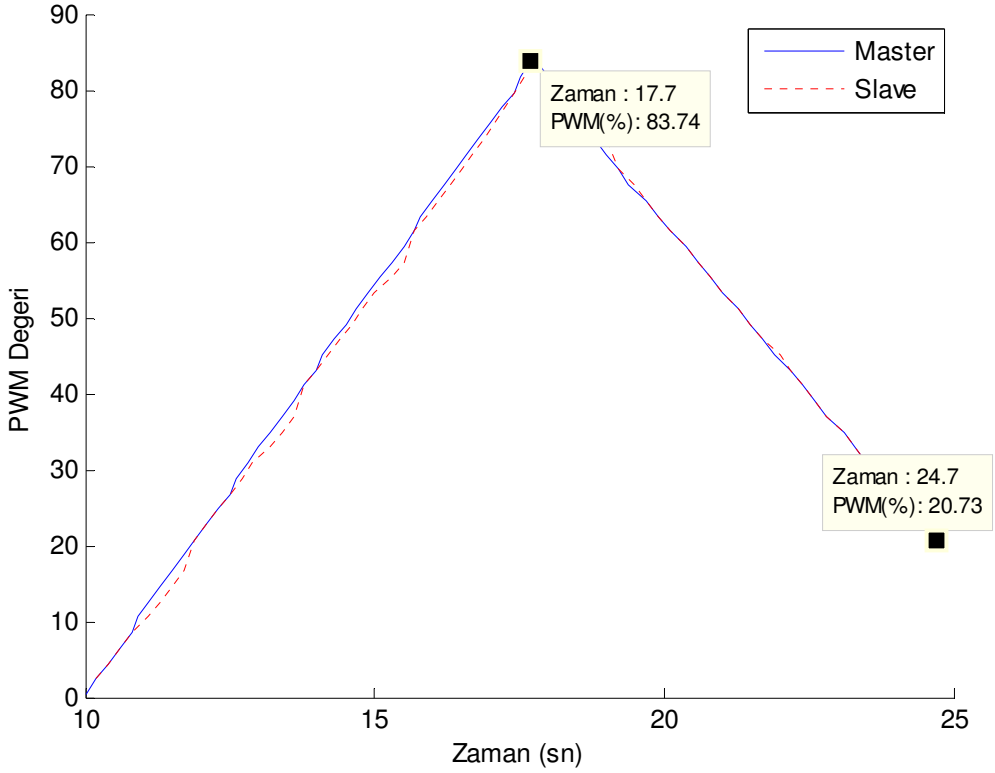
Şekil 54 – Tek Eklemde 2. Kuvvet Analizi PWM Grafiği

Tek eklemde yapılan 3. Kuvvet Analizi sırasında slave kolunu yavaşlatıcı 2 adet engel 95 derecedeki konuma ve 110 derecedeki konuma yerleştirilmiştir.

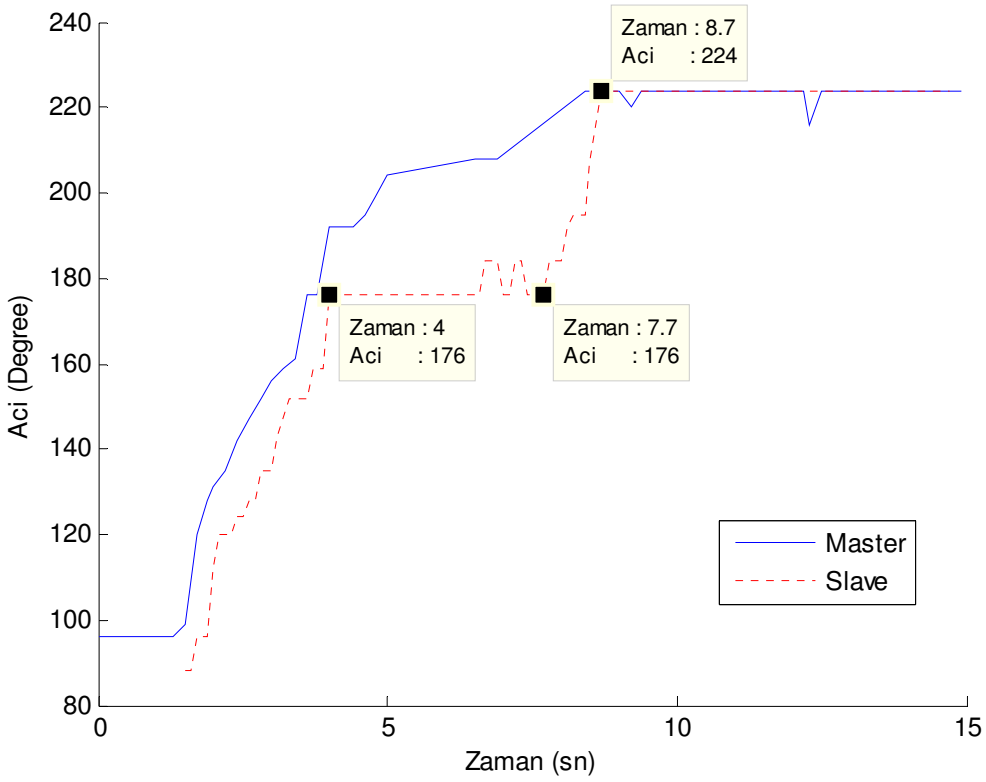


Şekil 55 – Tek Eklemde 3. Kuvvet Analizi Pozisyon Grafiği

Slave Kolunun hareketini tamamen durduracak 3. bir engel ise 128 dereceli konuma yerleştirilmiştir. Şekil 55'deki grafikte Slave kolu hareket ettirildiğinde 96 derecedeki engele 1.2 saniyelik bir sürede takılmış ancak engeli aşabildiği görülebilmektedir. Aynı şekilde 6.5 saniye sonra 110 derecedeki 2. Engele takılmış ancak bu engeli de 1.4 saniyede aşabilmiştir. Bu zamana kadar engele takılma süresi master kolunun cevap süresinden daha kısa olduğu için pozisyon değişimi minimum PWM değerinde sağlanmıştır. Slave kolu 9.6. saniyede 3. Engele takılmış ve açı değişimindeki kısıtlama 0.4 saniye sonra fark edilerek PWM değeri arttırılmaya başlanılmıştır. 7.8 saniye sonra PWM değeri %83 iken master kolunu kullanan operatör slave kolunu daha fazla zorlamak için master kolunun konumu değiştirmiştir. 17.8. saniyede PWM değeri azalmaya başlamıştır. PWM değişim grafiği Şekil 56'de görülebilmektedir.

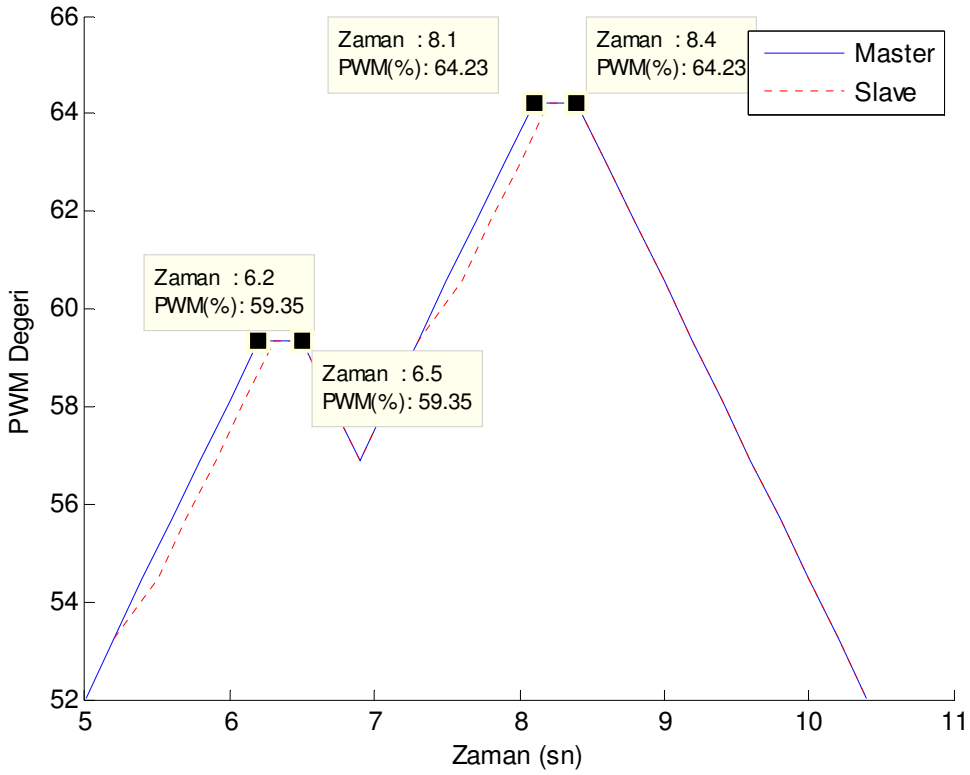


Şekil 56 - Tek Eklemde 3. Kuvvet Analizi PWM Grafiği



Şekil 57 - Tek Eklemde 4. Kuvvet Analizi Pozisyon Grafiği

Şekil 57'de görüldüğü gibi slave kolu 4 saniye sonra 176 derecede bir engele takılmış ve 3,7 saniye boyunca açısı sabitlenmiştir. Bu esnada hareket ettirilen master kolunun açısal hızında azalma gözlenmiştir. Şekil 58'de görülebileceği gibi Slave kolunun engele temas etmesinden 1 saniye sonra PWM değeri %50 den artmaya başlamıştır. PWM değeri Master ve Slave kollarında değişmesi ile Slave kolunda uygulanan ve Master kolunda ters yönde operatöre uygulanan kuvvet değeri eşit biçimde artmaya başlamıştır. 1.2 saniye sonra PWM değeri çıkabileceği en yüksek konuma erişmiş ve bu değerde sabitlenmiştir. PWM değeri %59 iken Slave kolu engeli aşabilmiştir. 6.9. saniyede konumlar birbirine eşit olmadığından PWM değeri %64'e kadar artmaya devam etmiştir. 8.4. saniyede ise konumlar eşit olduğundan PWM değeri çalışma performansı olan %50'ye düşmeye başlamıştır. Bu sırada PWM en yüksek değerinde olduğundan motor tarafından uygulanan tork değeri 0,0792 Nm olarak hesaplanmıştır. Bu ölçüm robot kolunun Dikomsan HT-300 hassas tartıda çıkabileceği maksimum ağırlık ile hesaplanmıştır.

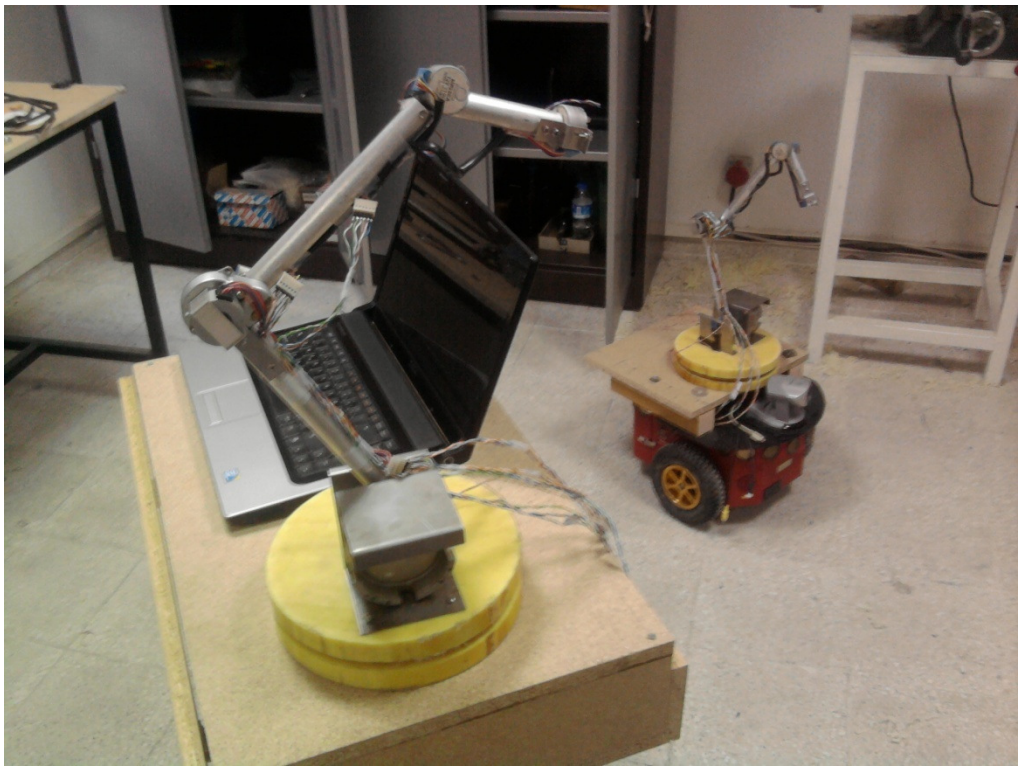


Şekil 58 - Tek Eklemde 4. Kuvvet Analizi PWM Grafiği

6.2. ROBOT KOLLARINDAN ALINAN VERİLER

5 adet adım motor ile 5 serbestlik derecesine sahip olan slave kolu ve 4 adet adım motor ile 4 serbestlik derecesine sahip master kolunun prototip tasarımı yapılmış ve kollar üretilmiştir. (Bkz: Şekil 59)

Bağlantı elemanları olarak 16mm çapındaki alüminyum borular kullanılmıştır. Açılar ise 10k ve 5kΩ'luk ayarlı dirençlerden okunmuştur. Ayarlı dirençlerden 0.7 derecelik bir hassasiyetle okunabilen veriler ile kontrol yapılmıştır.



Şekil 59 – İmalatı yapılan robot kolları

Devreler arasında açı ve PWM değerlerinin gönderimi ve alımı için gereken haberleşme PIC16F877A mikro kontrolcüsünün RX ve TX bacakları ile yapılmıştır. Algoritma ve haberleşme problemlerini aza indirgemek için tasarlanan devre ilk önce Proteus programında denenmiş daha sonra breadboard'a kurularak olası hataların giderilmesi için denenmiştir. Yer kazanımı ve hata izleme kolaylığı düşünülerek her motor için ayrı ayrı baskı devreleri yapılmış ve ısınma sorunu için modifikasyonlara

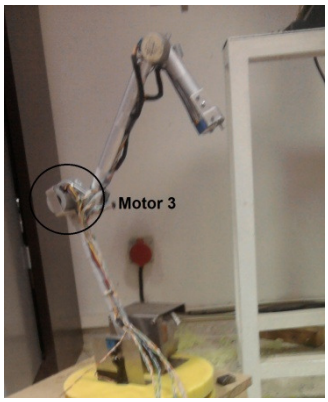
gidilmiştir. Tüm sorunların çözümünden sonra robot koluna bağlantıları tamamlanmıştır.

Kuvvet kontrolü için adım motorlarının küçük adımlar ile hareket edebilme özellikleri kullanılarak yeni bir algoritma geliştirilmiştir. Bu algoritma içerisindeki değerleri okumak için verilen bekleme süreleri göz önüne alınarak teorik grafikler çizilmiştir.

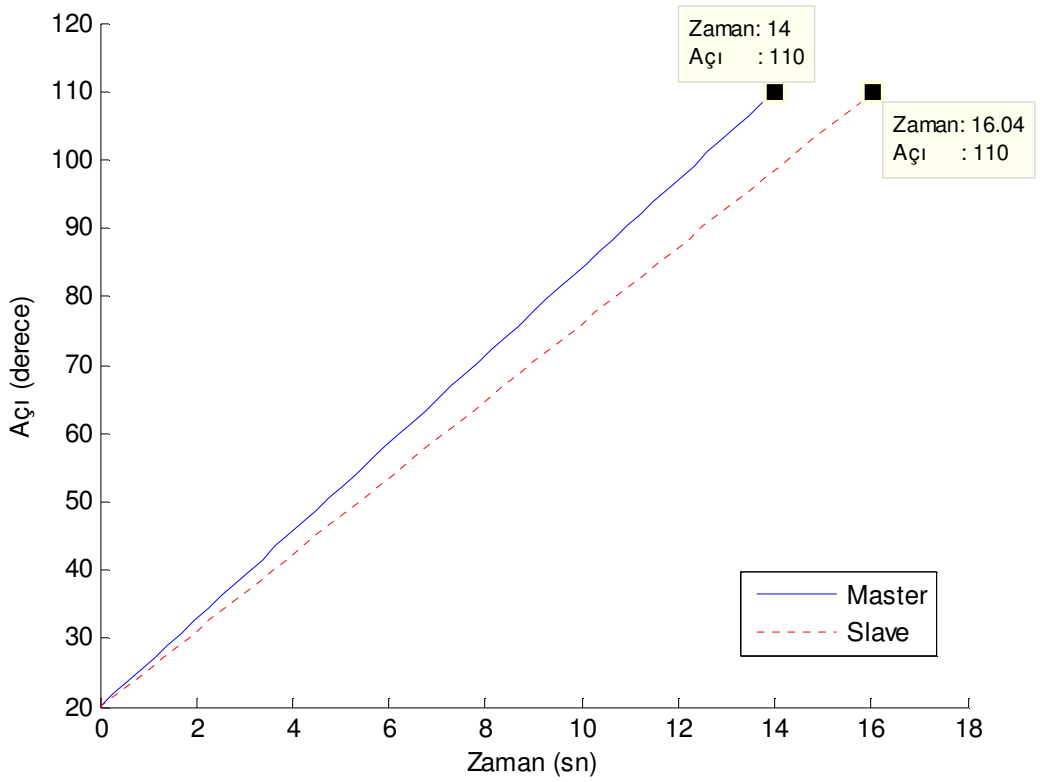
Yapılan Testler sonucunda 1. ve 2. Motor 25 W'lık bir direnç kullanılmasına rağmen motorların 2 Ampere yakın akım çekmesinden dolayı aşırı ısınan devre elemanları uygun gözlemler yapılması için gereken süre boyunca devrenin çalışmasını engellemiştir. 3. ve 4. Motorlarda ise boyutlarını ve çektiği akımların çok fazla olmamasından dolayı beklenildiği gibi performans alınmış ve kuvvet geribeslemesi hissedilmiştir. 4. Motor en son konumda olduğu dişli oranı yüksek tutulup boyut ve tork olarak en düşük model seçilmiştir. Hesaplamalara göre uygun olan motor kritik noktalarda tutucunun ağırlığını kaldıramamıştır. Bu nedenle sonuçlar 3. Motordan alınan değerler göre hesaplanmıştır. 3. Motor en yavaş motor olduğundan dolayı veri alımı daha kolay olmuştur.

6.2.1. 3 Numaralı Motor'dan alınan sonuçlar

3 numaralı motorun konumu Şekil 60'da görülmektedir. Kullanılan 1:75 oranındaki redüktör sayesinde 0.1121 rad/sn'lik açısal hız elde edilmektedir.

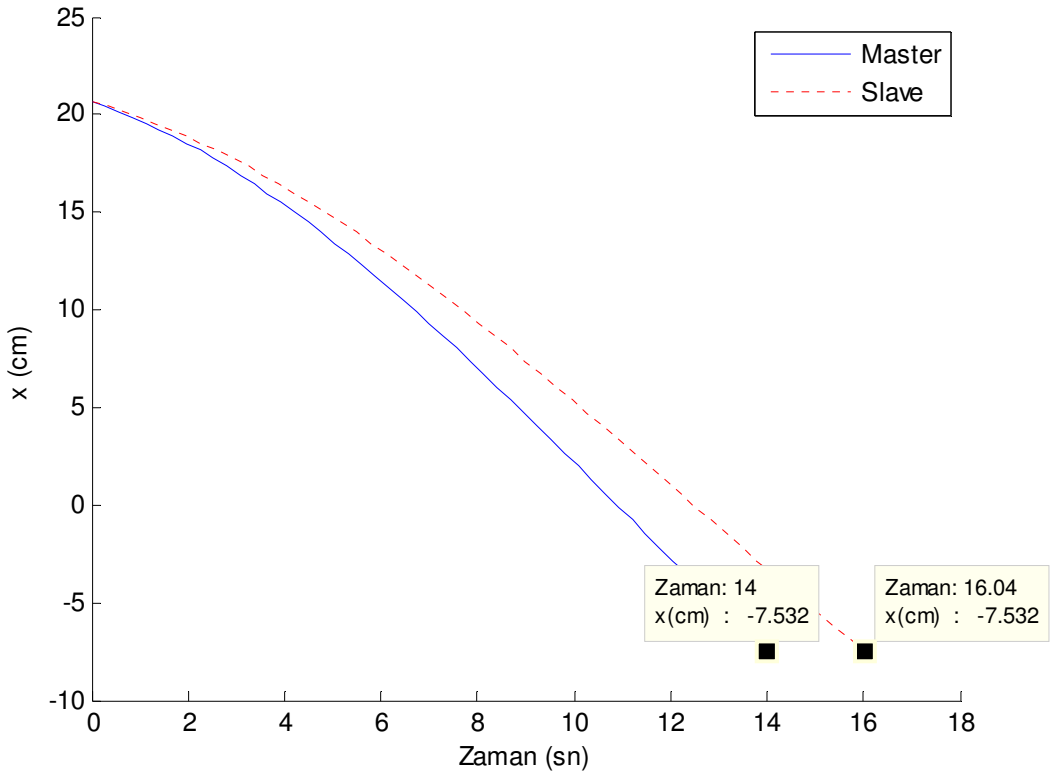


Şekil 60 – 3. Motorun konumu



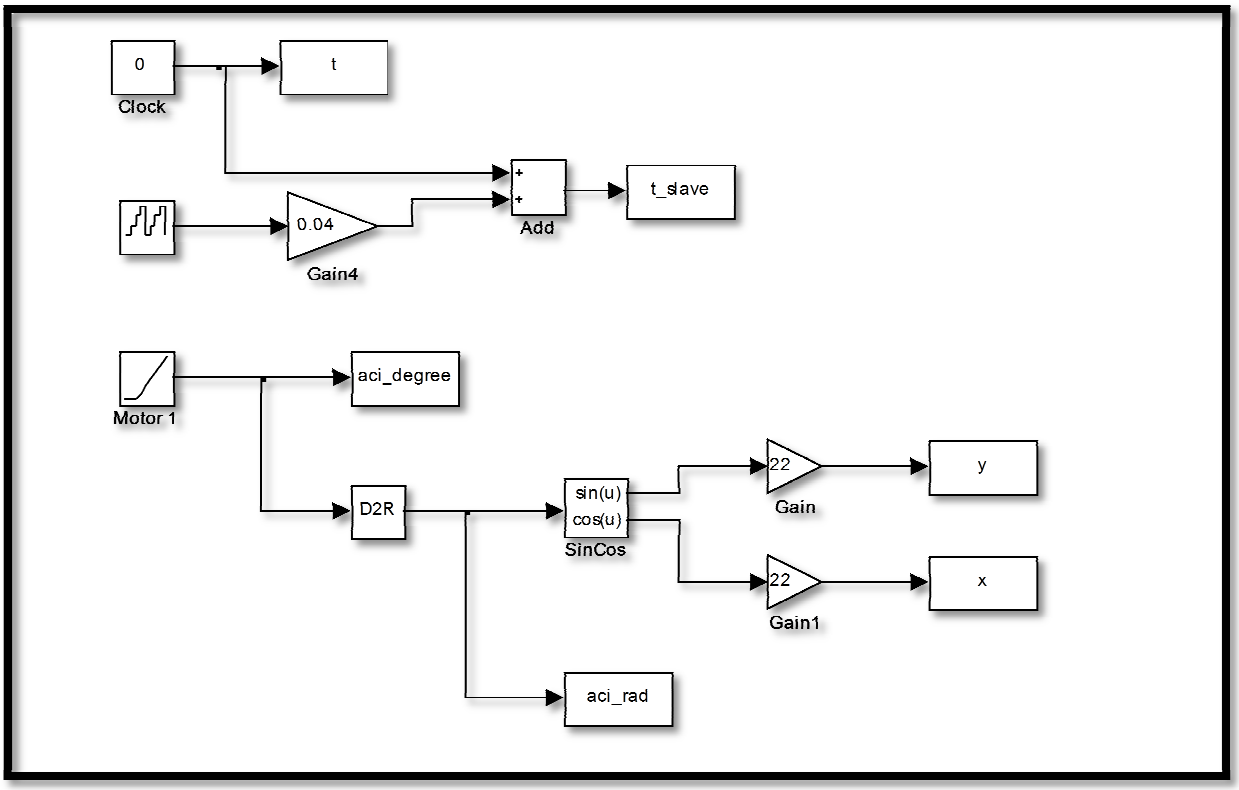
Şekil 61 – 3 Numaralı Motor Hareketlerinin sonucunda beklenen θ–t grafiği.

Şekil 61'de görülen grafikte bir dönüşte 96 adet adımlık hareket yapan motorun bir adımı 3.75^{o}'liktir. 1:75 dişli oranı ile ulaşabilecek en yüksek hız 0.1121 rad/sn'dir. Bu hızda hareket ettirilen master kolunun açısal değişimi slave koluna gönderildiğinde slave kolu π/2 radyanlık bir değişime cevabı şekilde görüldüğü gibi 2.04sn geç olması beklenmektedir.

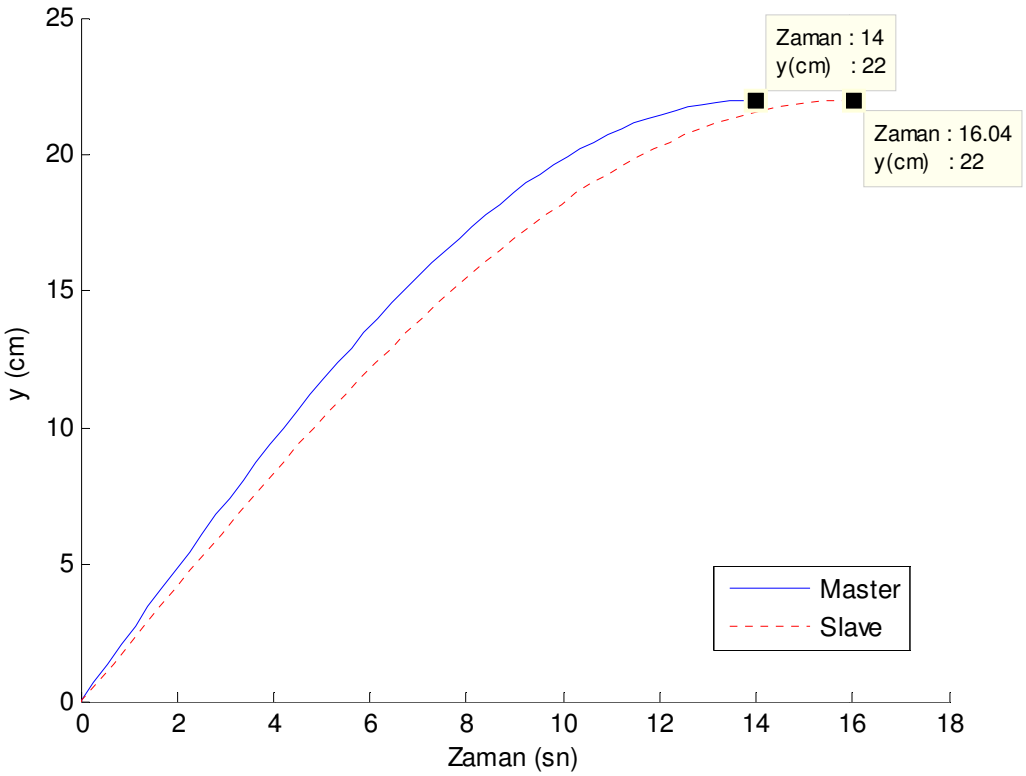


Şekil 62 – Slave kolunun π/2 radyanlık bir değişim karşısında x-t grafiği

Slave kolu, teorik hesaplamalarla elde edilen sonuçlara göre 16 saniyede π/2'lik açısal değişimi yapabilmektedir. Master kolundan 2.04sn sonra slave kolu da x ekseninde istenilen pozisyona gidebilmektedir. X konumundaki değişimin değeri radyan cinsinden elde edilip kosinüsü alınarak bulunur. Bu değer 3. Motorun bulunduğu ekleme göre alınır. Motor tabanına göre değişim düz kinematik denklemlere bulunabilir. Ancak, açısal değişim ile kontrol söz konusu olduğu için bu yönteme başvurulmamıştır. Her bir eklemdeki değişim kendi çalışma alanındaki koordinatlara göre hesaplanmıştır. Bu hesaplamalar Matlab/Simulink'de modellenip sonuçlar bu yolla elde edilmiştir. Matlab/Simulink Modeli Şekil 63'de görülebilmektedir. Motorun redüktörden dolayı değişen hız grafiği rampa fonksiyonu ile verilmiş (Bkz: Şekil 62) bu açıların kosinüs ve sinüs değerleri bağlı bulunan elemanın uzunluğu ile çarpılıp "x" ve "y" koordinatlarındaki değişim gözlenmiştir.

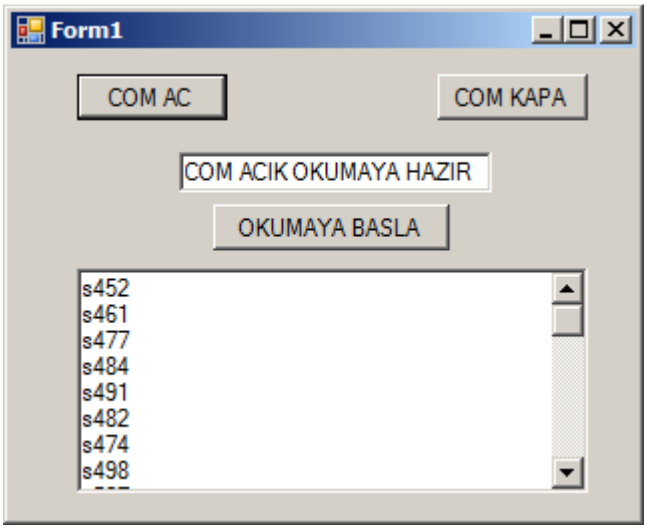


Şekil 63 – Teorik sonuçlar için Matlab / Simulink Modeli

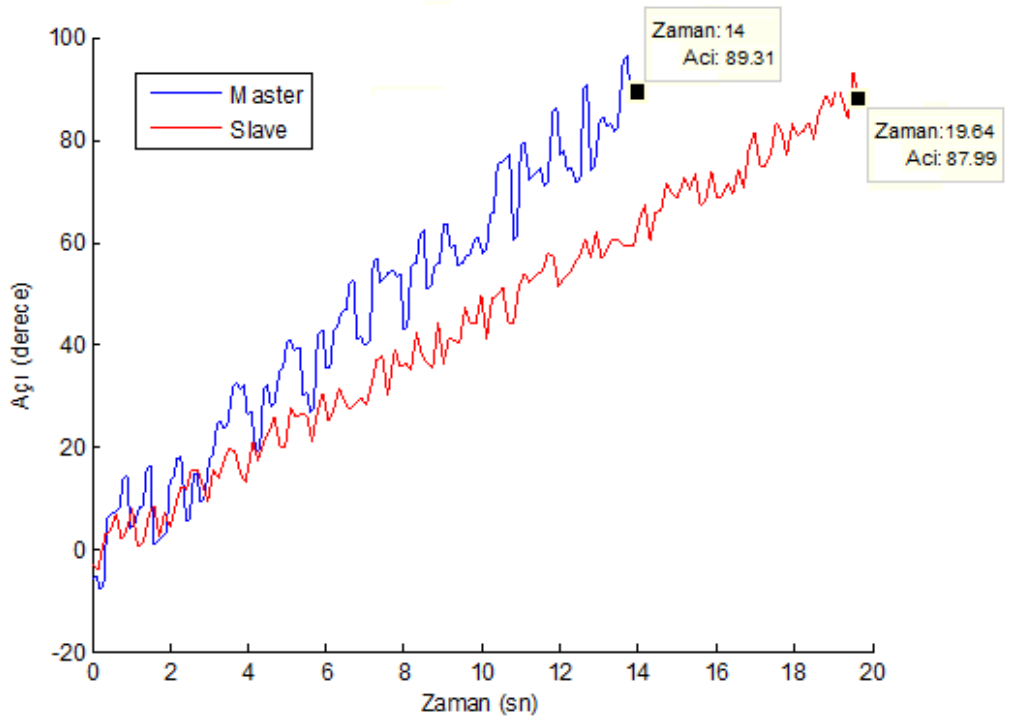


Şekil 64 - Slave kolunun π/2 radyanlık bir değişim karşısında y-t grafiği

3. Motorun açısal hızıyla hesaplanan bu teorik değerlerden sonra robot kolları çalıştırılarak mikro kontrolcüler arasındaki haberleşme izlenmiş ve pratik değerlere ulaşılmıştır. Açısal değişim olduğunda master'dan slave'e açı gönderilmektedir. Bu gönderim kodda yapılan ufak bir değişiklik sonucu her defasında bilgisayardan izlenebilmiş ve elde edilen veriler ile şekildeki grafik çizilmiştir. Şekil 65'de yazılan programın ara yüzü görülebilmektedir.



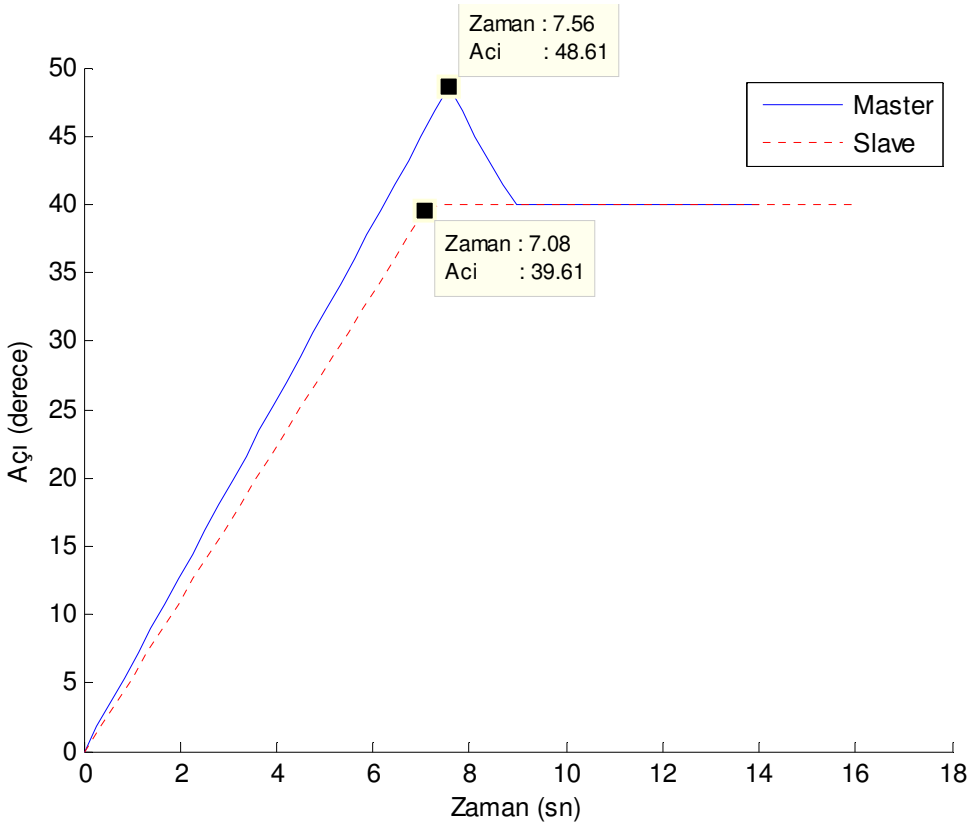
Şekil 65 – Pratik sonuçları elde etmek için C# yazılmış Program



Şekil 66 – 3 Numaralı Motordan elde edilen θ–t grafiği

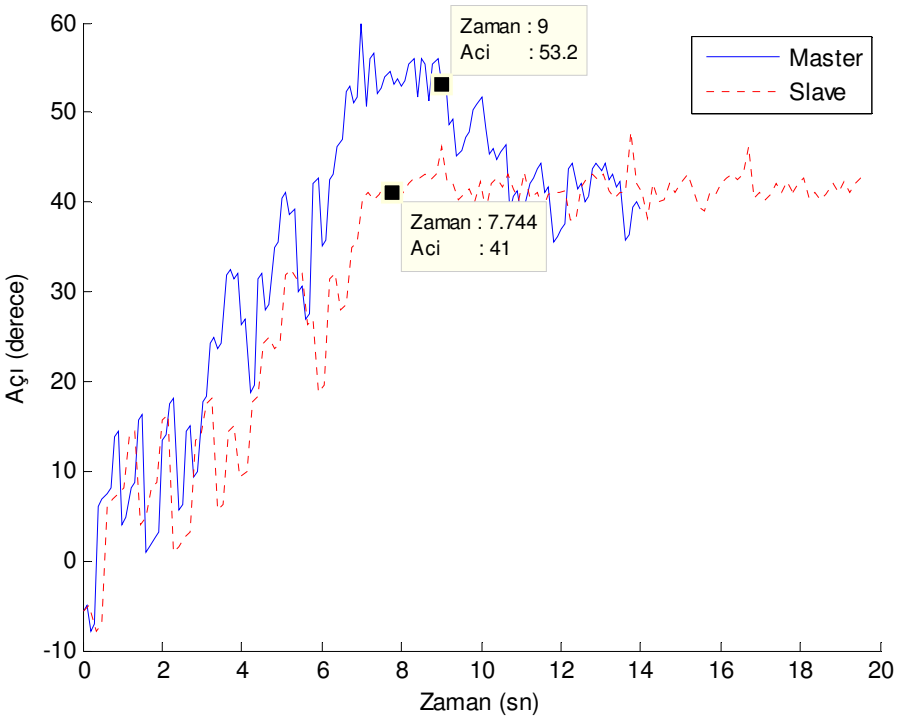
Şekil 66'da görüldüğü gibi slave kolunun istenilen açı değerine gelebilmesi için beklenen değer olan 2.04 den farklı olarak 5.64 sn 'lik bir gecikme elde edilmiştir. Bu gecikme çalışma esnasında mekanik sistemden veya devrenin çalışma performansından ötürü olabileceği düşünülmüştür. Bu verilerin kosinüs ve sinüs değerleri kullanılarak bulunan pozisyon zaman grafikleri bulunabilir.

Kuvvet geribeslemesi için 90 derecelik bir değişim yapılması istenilen slave kolunun 40 derecede maruz kaldığı bir engele Master'ın ve Slave'in cevabı Matlab / Simulink'de modellenmiştir. Buna göre 40 derecede karşılaştığı engel dolayısıyla açısı bu konumda sabit kalan slave kolu, takıldığı açıyı fark ettiğinde master koluna bu açıyı gönderir. Master kolu ise gitmesi gereken bu açıya geri dönüş yapar. Bu geri dönüş, 0.46 sn sonra olması beklenmektedir. (Bkz: Şekil 67)



Şekil 67 – Geribesleme esnasındaki θ–t grafiği

Yapılan test ile elde edilen veriler kullanılarak çizilen grafikte görüleceği gibi, beklenen değerden farklı olarak 1.26 sn sonra cevap alınmıştır. (Bkz: Şekil 68)

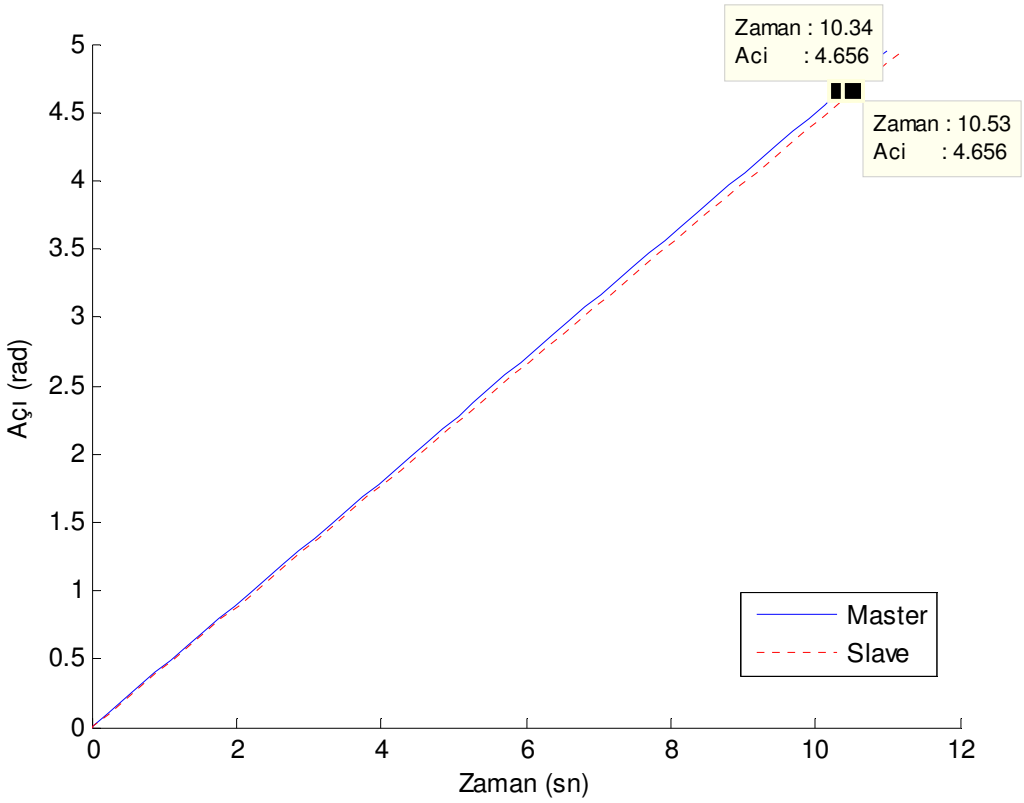


Şekil 68 – Kuvvet geribesleme cevabı

6.2.2. 4 Numaralı Motor

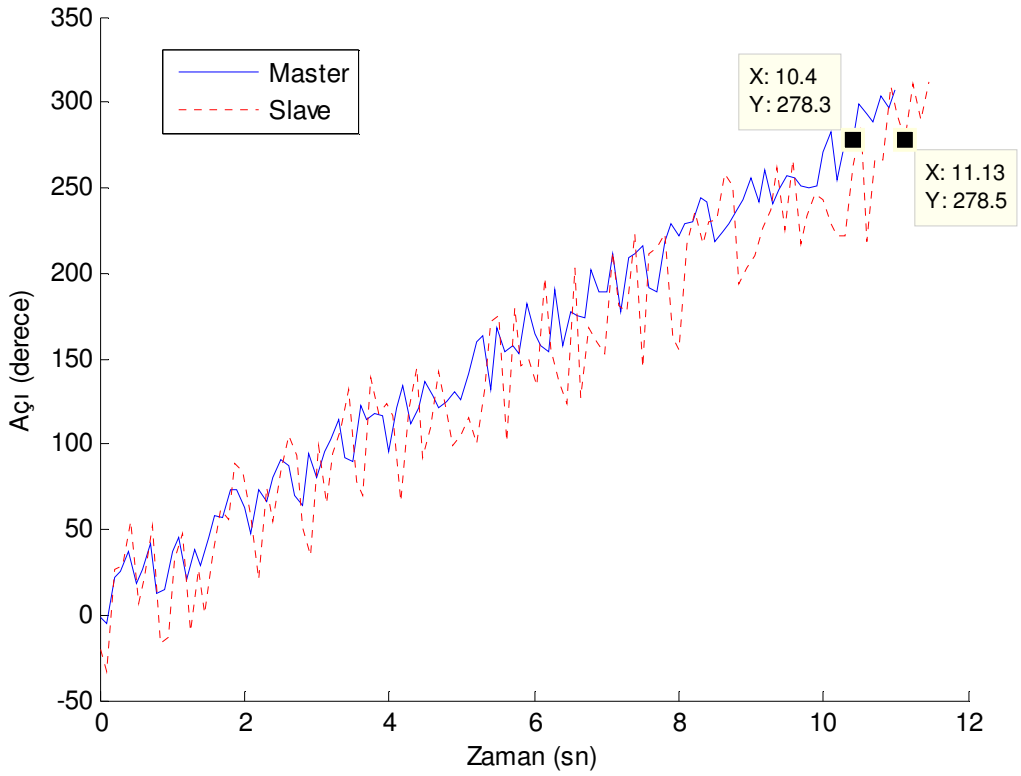
4 Numaralı motorun konumu Şekil 69'de görülmektedir. Çalışma esnasında haberleşme sırasında gönderilen veriler C#'da yazılan program ile izlenmiştir. Bu sayede pratik sonuçlar elde edilmiştir.

Şekil 69 – 4 Numaralı Motorun Konumu



Şekil 70 – 4 Numaralı Motorun Açı – Zaman grafiği

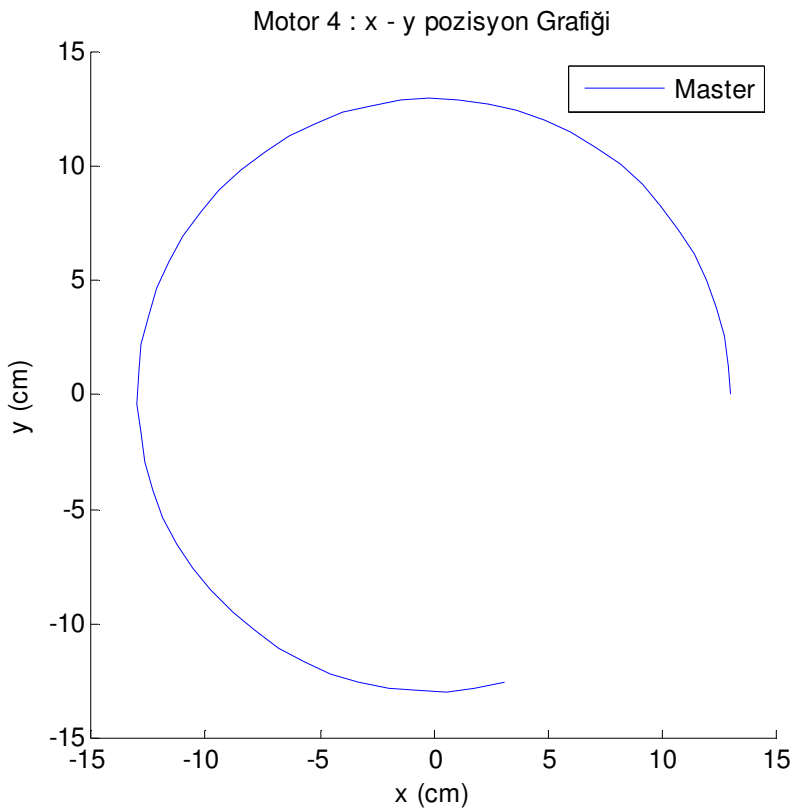
4 Numaralı motor ile yapılan teorik çalışmada 1:100 dişli oranına sahip bir redüktör ile kullanılan motordan elde edilebilecek en yüksek hız 0.45 rad/sn'dir. Şekil 70'de görüldüğü gibi 10.53 saniyede 266.8 derecelik hareketi yapabilmektedir. Operatör master kolunu bu hızda hareket ettirdiğinde slave kolu 0.19 sn'de bu hareketi takip edebilmelidir. Böyle bir hareket Şekil 70'de görülen x-y grafiğini vermektedir.



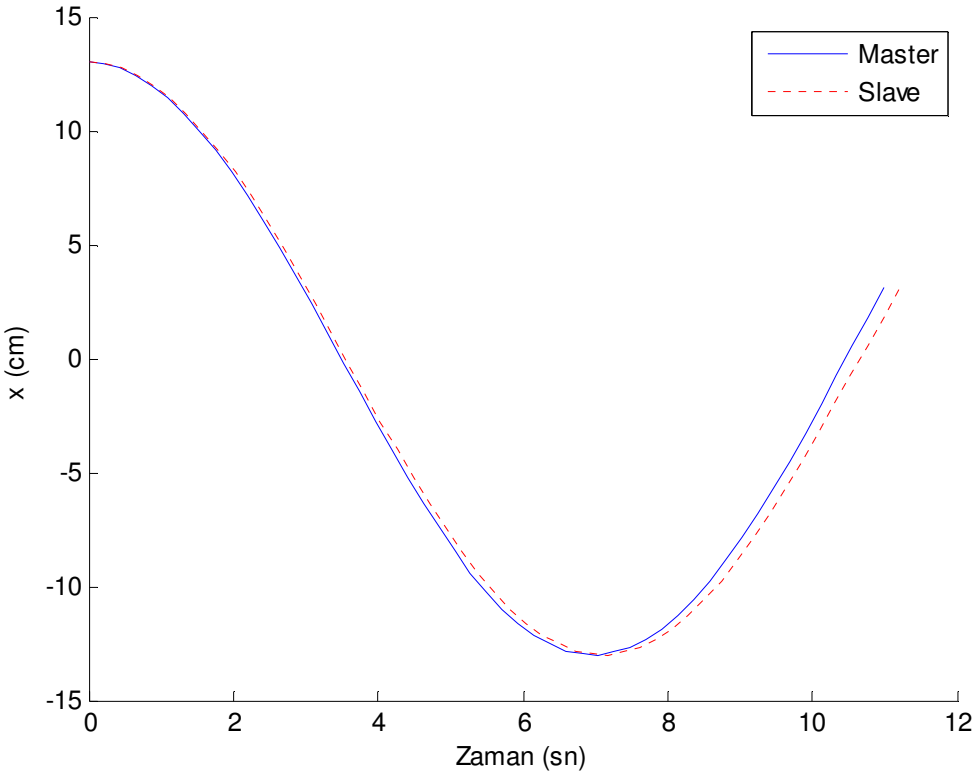
Şekil 71 – 4 Numaralı Motor Pratik Deney Açı – Zaman Grafiği

Şekil 71'de görüldüğü gibi yapılan deney sonucu alınan veriler sayesinde çizilen grafik Slave kolunun Master kolunu 1.09 saniyelik gecikmeyle takip ettiğini göstermektedir. Bu sürede, alınan verilerin değişken olmasından dolayı tam olarak aynı derecede olamamıştır. Bu nedenle algoritmada açı kontrolünde eşitlik yerine bir aralık kullanılmıştır.

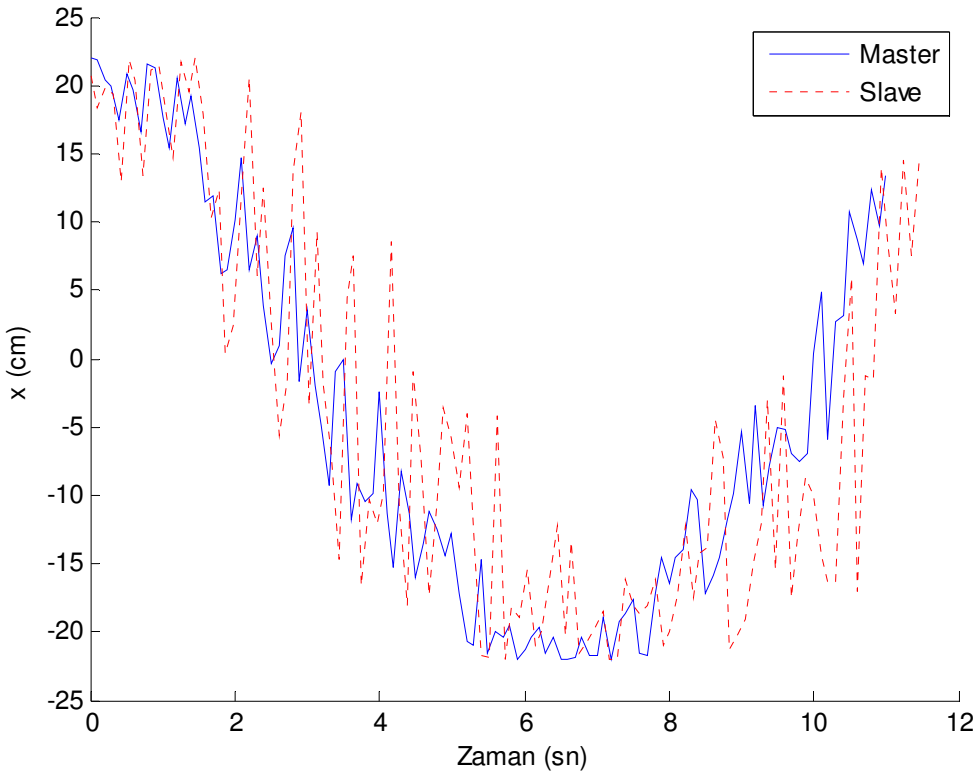
Şekil 72'de x-y grafiği 270 derecelik bir hareket esnasında takip edilen yörünge görülmektedir. Ayrıca Şekil 73'de 4. Motorun bulunduğu bağlantı elemanı için x yönündeki teorik olarak bulunmuş yer değişimi görülmektedir. Yapılan deneyler sonucu Şekil 74 ve Şekil 76'da alınan verilerin değişken olmasından dolayı lineer olmayan bir grafikler görülebilmektedir. Teorik sonuçlarda elde edilen sinüs hareketi pratik sonuçlarda da gözlemlenmiştir Ancak hareket sırasındaki titreşimden ve mekanik bağlantılardaki boşluklardan kaynaklanan değişken veriler grafiği etkilemiştir.



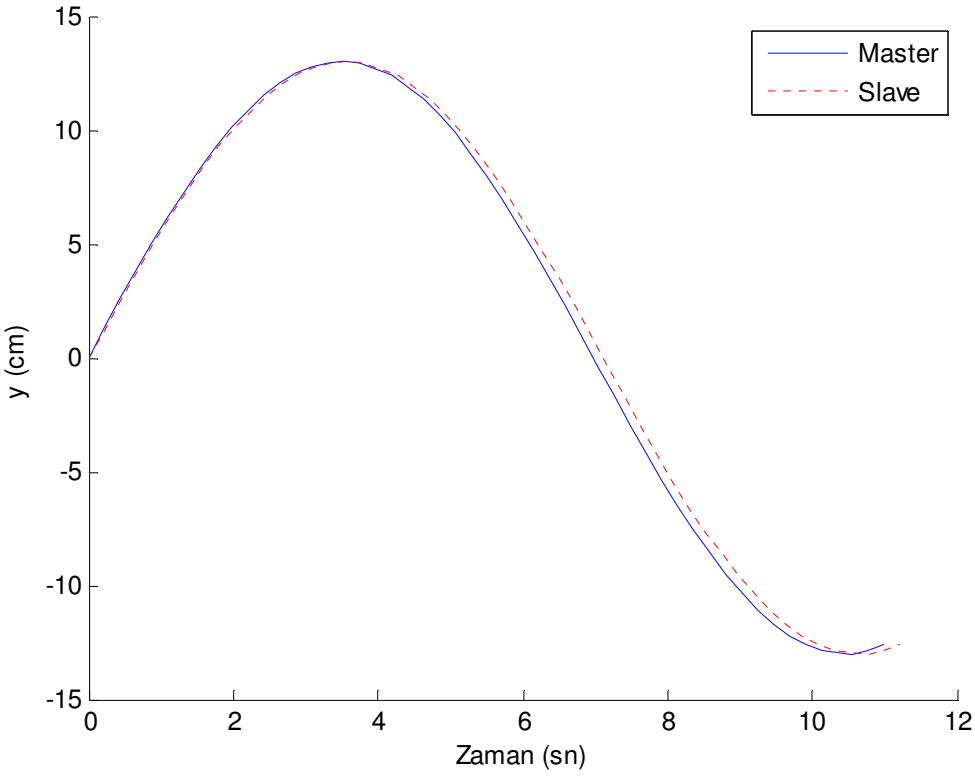
Şekil 72 – 4 Numaralı Motor Teorik x – y grafiği



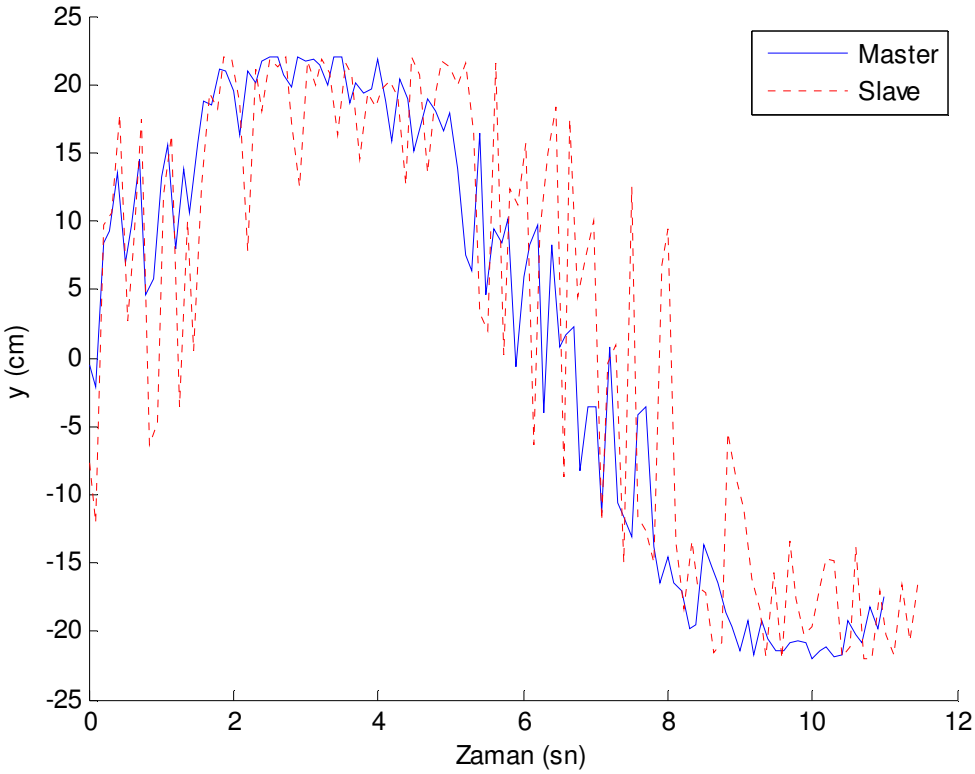
Şekil 73 – 4 Numaralı motor teorik x – zaman grafiği



Şekil 74 – 4 numaralı Motor Pratik deney x-t grafiği



Şekil 75 – 4 Numaralı Motor y – zaman grafiği



Şekil 76 – 4 Numaralı Motor Pratik Deney y-t Grafiği

7. SONUÇ VE YORUMLAR

Yapılan deneyler sırasında, 1. ve 2. Motorun 2 Ampere yakın akım çekmesi nedeniyle devrede ısınma tespit edilmiştir. Z yönünde hareketi sağlayan 1 numaralı motora eksenel yük etki etmektedir. Adım Motorlarının yüksek eksenel yük kapasitelerinden dolayı daha küçük boyutlarda bir motor kullanılabilir. Bu sayede ısınma problemi ortadan kalkacak ve kontrol sırasındaki hatalar önlenecektir.

3. ve 4. Motorlarda ise çekilen akımın çok fazla olmamasından dolayı beklenen bir performans alınmış ve PWM arttırılması ile motor torkunun 0,0618 Nm ile 0,1236 Nm arasında değişmesi ile kuvvet geribeslemesi yapılmıştır. Ancak 3. Motor bu kuvvet geribeslemeyi yaşatmak için en az 1.3 saniyeye ihtiyaç vardır. Slave kolu bir yere çarptıktan sonra yapılan testlerdeki master kolunun en hızlı cevabı 1.26 sn sonra

olmuştur. Bu hata geribesleme kontrolünde adım motorların ile ayarlı dirençlerin bağlantılarındaki boşluklardan dolayı kaynaklanmaktadır.

4. Motor en son konumda olduğundan dolayı dişli oranı yüksek tutulup boyut ve tork olarak en düşük model seçilmiştir. Hesaplamalara göre uygun olan motor kritik noktalarda tutucunun ağırlığını kaldıramamıştır. Bu sorun ise kullanılan adım motorun daha güçlü bir adım motoruyla değiştirilmesi ile çözülebilir. Daha güçlü bir motor daha çok ağırlık yapacağından dolayı bu modifikasyon diğer eklemlerdeki motorları ekleyecektir. Bu nedenle en kritik olarak 4. Motor düşünülmüştür. Tutucu ve tutucunun dönüşünü sağlayan 5. Motor ise master hareketi tekrarlamıştır. Tutucunun bir engel karşısında kısıtlı hareketi direkt olarak 3. ve 4. Motorları etkiyeceğinden bu kısımda uygulaması mekanik olarak çok zor olan kuvvet geribesleme kullanılmamıştır.

Robot kolları ile yapılan çoklu eklem deneylerinde ayarlı direnç ve motor mil bağlantısından, motor milleri ile bağlı olan redüktörlerdeki diş boşluklarından ve birden fazla motorun aynı anda hareketi sırasında oluşan titreşimden kaynaklanan olası mekanik hataların elde edilen verileri etkilediği öngörüldüğünden dolayı. Tek eklem ile testler yapılmıştır.

Hazırlanan deney düzeneği ve C# programı yardımıyla, yapılan testler sırasında veriler toplanmıştır. Tekrarlanarak yapılan birçok deneyden elde edilen veriler ile çizilen grafiklerde, kullanılan motorun ortalama açısal hızı 0.60 rad/sn olarak bulunmuştur. Operatör 0.60 rad/sn'lik açısal hızdan daha hızlı bir şekilde master kolunu hareket ettirdiğinde 0.4 ila 1.3 saniye arasında değişen gecikme süreleri ile master kolunun cevabı gözlemlenebilmiştir. Haptik robot kolları ile yapılan H.Y.K. Lau ve L.C.C. Wai'nin çalışmasında [5] bu gecikme süresi 1 ila 3 ms arasında değişim göstermektedir. Adım motorları ile yapılan bu çalışmadan gecikme süresi açı ölçmek için kullanılan sensörün hassasiyetiyle doğrudan orantılıdır. Geliştirilen algoritma adım motorlarının bir harekete karşılık açısal değişimini kullanmaktadır. Bu açısal değişim ne kadar hassas olursa o kadar hızlı bir şekilde kontrolcü çalışabilmektedir.

Motorların millerine bağlı olarak 60mm uzunluğunda kullanılan eklem ile yapılan testler sonucu 2.06 N'luk kuvvet elde edilebilmiştir. Bu değer PWM ayarıyla değiştirilerek kuvvet geribesleme yapılmıştır. J. G. Hollinger, R. A. Bergstrom, ve J. S. Bay'ın yaptığı çalışmada[22] adım motorları ile çalışan MERLIN 6540 endüstriyel robot üzerine uygulanan kontrolde ve 26.7 N'luk kuvvet elde edilmiştir. H.Y.K. Lau ve L.C.C. Wai yaptığı çalışmada[5] ise ulaşılan en yüksek değer 10N olmuştur.

Yapılan deneylerde PWM değeri minimum değerden en yüksek değere 9 saniyede çıkarılmış ve pozisyonlar tekrar eşitlendiğinde aynı hızda düşürülmüştür. Bu artış oranı ayarlanabilmektedir. Hassas cisimler ile çalışılıyorsa bu oran düşük tutularak PWM çok yavaşça arttırılabilir. Bu sayede hassas cisimler zarar görmez. Dayanıklı cisimler ile çalışılıyorsa ve hız önemli bir faktör ise bu oran yüksek tutularak PWM hızlıca arttırılabilir. Bu sayede tepki süresi artar ve engelleri aşmak için gerekli kuvveti elde etmek daha kısa sürer.

Kuvvet geribeslemesi literatürdeki gibi Haptik kollar veya kuvvet sensörleri dışında bir yaklaşımla hazırlanan bu proje, robot kollarında pek fazla kullanılmayan adım motorlarıyla yapılmıştır. Amaç pahalı sensörler ve hazır kollar kullanmak yerine daha ucuz olan adım motorlarla ve uygun algoritmalarla sonuca gitmekti. Adım motorları uyguladığı güç bakamından çok geniş bir yelpazede yer almasında karşın istenilen güçtekilerin fiyatı bakımından projenin asıl hedefi dışına çıkmaktadırlar. Geliştirilen bu algoritma uygun motorlar ve enkoder'lar kullanılarak daha iyi performans vereceği düşünülmektedir. Açısal değişimlerin ölçümünde kullanılan ayarlı dirençler çok uygun fiyatlara bulunmasına karşın böyle hassas çalışmalarda kullanımının uygun olmadığı düşünülmüştür. Ölçümler ayarlı dirençlerin hassasiyetlerine göre bulunan aralıklarda alındığında, sonuçlar istenilen kararlılıkta değildir. Bunun nedeni, ayarlı dirençlerden alınan ölçümlerin çok değişken olmasıdır. Enkoder kullanılan bir sistem tasarlandığında alınan veriler daha kararlı olacağından algoritmanın performansı artacaktır.

Kuvvet geribeslemesi ile ilgili yapılan çalışmalarda, Kuvvet/Tork sensörleri kullanılarak ve DC motorlarında akım kontrolü metodu kullanılmıştır. Bu yöntemle sonuca ulaşılmak istenildiğinde kullanılması gereken DC motorların ağırlıkları ve

boyutları çok önemli olduğundan istenilen tork değerlerinin sağlanabilmesi için çok pahalı cihazlar alınması gerekmektedir. Akım kontrolü için tasarlanması gereken kontrol ise çok daha karmaşık olduğundan, bu çalışmadaki algoritma ve adım motorları sayesinde kuvvet değer yaklaşımı yapılmıştır. Bu yaklaşım metodu ile çok daha ucuza temin edilebilecek cihazlarla kuvvet geribeslemesi prensibi olan slave kolundaki direnç kuvvetinin operatöre yansıtılması yaklaşım metodu ile sağlanmıştır.

Sonuç olarak bu çalışmada PWM artışı ile motor torkunun kontrol edilmesi ile çalışan sistem tasarlanmış, seçilen elemanlara uygun kontrol ve yazılım gerçekleşmiştir. Adım motorların ters kutuplama yöntemi kontrol edilir ve açı ölçmek için enkoder kullanılır ise bu şekilde tasarlanmış sistemlerin performansları daha da arttırılabilir. Böylece Haptik sistemlere alternatif olarak kullanılabilir sistemler haline getirilebilinirler.

KAYNAKLAR LİSTESİ

[1] Conceicao, A. S., Moreira, A.P., Costa, P. J., A nonlinear model predictive control strategy for trajectory tracking of a four-wheeled omnidirectional mobile robot, Optimal Control Applications and Methods, John Wiley and Sons, v.29, s.335-352, 2007.

[2] ATI Industrial Automation: Multi-Axis Force / Torque Sensors., http://www.ati-ia.com/products/ft/sensors.aspx

[3] Çiçek Serdar, CCS C İle PIC Programlama. Altaş Yayıncılık, s.14, 2009.

[4] Dan Liu, Emanuel Todorov, Hierarchical Optimal Control of a 7-DOF Arm Model Adaptive Dynamic Programming and Reinforcement Learning, ADPRL '09. IEEE Symposium on Computational Intelligence, 2009.

[5] Eroğlu Elif Gezgin Robotlarda Ultrasonik Mesafe Algılayıcılarla Robot Davranış-larının Kontrolü ve Çevre Haritalama Yüksek Lisans Tezi / Eskişehir Osmangazi Üniversitesi Elektrik Elektronik Mühendisliği Anabilim Dalı, 2006.

[6] H. Y. K Lau. and Wong, V. W. K. An immunity-based distributed multi-agent control framework, IEEE Transactions on System, Man and Cybernetics, Part A, 36(1), s. 91 – 108, 2006.

[7] H.Y.K. Lau, L.C.C. Wai Implementation of position–force and position–position teleoperator controllers with cable-driven mechanisms. Robotics and Computer-Integrated Manufacturing, s.145–152, 2004.

[8] J. G. Hollinger, R. A. Bergstrom, ve J. S. Bay "A Fuzzy Logic Force Controller For A Stepper Motor Robot", Virginia Polytechnic Institute and State University

[9] Julie A. Jacko Constantine Stephanidis Human-computer interaction: theory and practice., New Jersey : Lawrance Erlbaum Associates, 2003. - Cilt 2 : 2 : s. 572, 2006.

[10] Kulwiec Raymond A. Materials handling handbook: American Society of Mechanical Engineers, International Material Management Society, s. 565, 1985.

[11] L. A. Zuniga, A. J.C. Pedraza O., E. Gorrostieta, L. Garcia-Valdovinos, J. M. Ramos, C.A. Gonzalez. Design and Manufacture of a Mobile Robot applied to the Manipulation of Explosives IEE, s.84 - 89, 2008

[12] MacFarlane Mark, Rosen Jacob, Hannaford Blake, Pellegrini Carlos, Sinanan Mika, Force-Feedback Grasper Helps Restore Sense of Touch in Minimally Invasive Surgery: J GASTROINTEST SURG, vol. 3, s.278-285, 1999.

[13] MERLET Jean-Pierre, Force-feedback control of parallel manipulators, IEEE. Valbonne, France, s.1484 – 1489, 2004.

[14] Milford Michael John, Robot navigation from nature: simultaneous localisation, mapping, and path planning based on Hippocampal Models, Springer, s.3, 2008.

[15] MobileRobots Advanced Robotics Interface for Applications (ARIA) Developer's Reference Manual ARIA Overview, 2006. 24 Haziran 2010. http://www.ai.rug.nl/vakinformatie/pas/content/Aria-manual.

[16] NewScientist, Earth canyon hints at ancient megafloods on Mars 1994. http://www.newscientist.com/articleimages/dn13948/., 2008

[17] Polmear Iz. J. Light Alloys: Metallurgy of the Light Metals: Arnold, 1995.

[18] Richard H. Barnett Larry O'Cull,Sarah Cox,Sarah Alison Cox Embedded C programming and the microchip PIC : Cengage Learning, 2004.

[19] Saha Subir Kumar, Introduction to Robotics: Tata McGraw-Hill, 2008.

[20] Siegwart, R. and Nourbakhsh, I. R., Introduction to Autonomous Mobile Robots, The MIT Press, ch 6, 2004

[21] Souma Mahmoud, Alhaj A Technologies for autonomous navigation in unstructured outdoor environments: PhD Thesis University of Cincinnati P 239 - 2003.

[22] Susumu Tachi Taisuke Sakaki Impendance controlled master-slave manipulation system - Tokyo : JRSJ 9, 1989.

[23] T. Graham, Bombs and Bombings, Illinois, U.S.A., s.58-130. 2006.

[24] The Cutting Edge of Haptic Research., http://www.mechatronictips.com/ technology/ motioncontrol/the-cutting-edge-of-haptic-research/., 2008

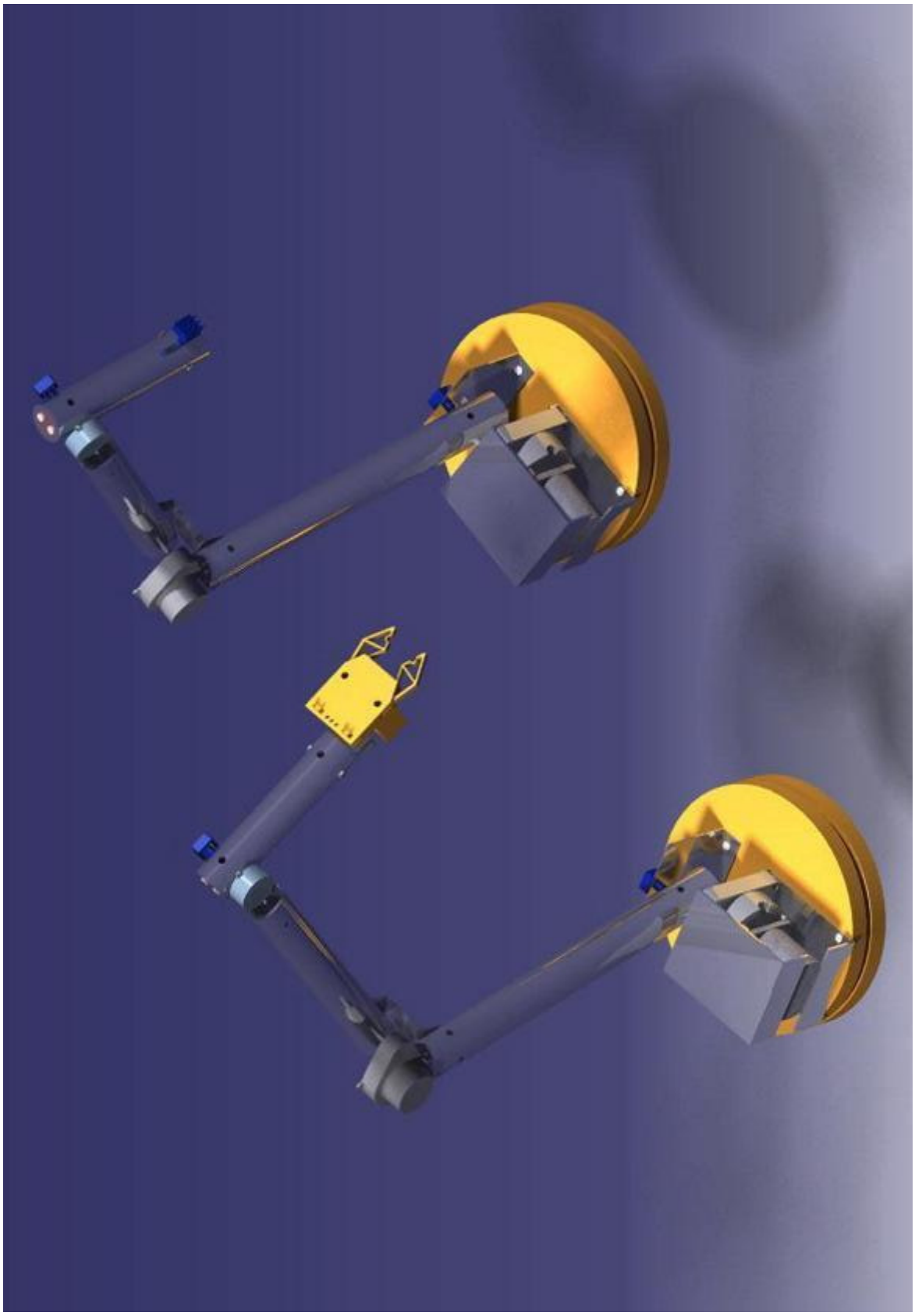
[25] WANG Jia, WEI Boyu, ZHANG Ye, CHEN Hu, Design of An Autonomous mobile Robot for Hospital IEEE, 2009.

[26] Y. Yamamoto and X. Yun, Coordinating Locomotion and Manipulation of a Mobile Manipulator, IEEE Transactions on Automatic Control, 6/a ed. vol. 39, s.1326-1332, 1994.

EKLER LİSTESİ

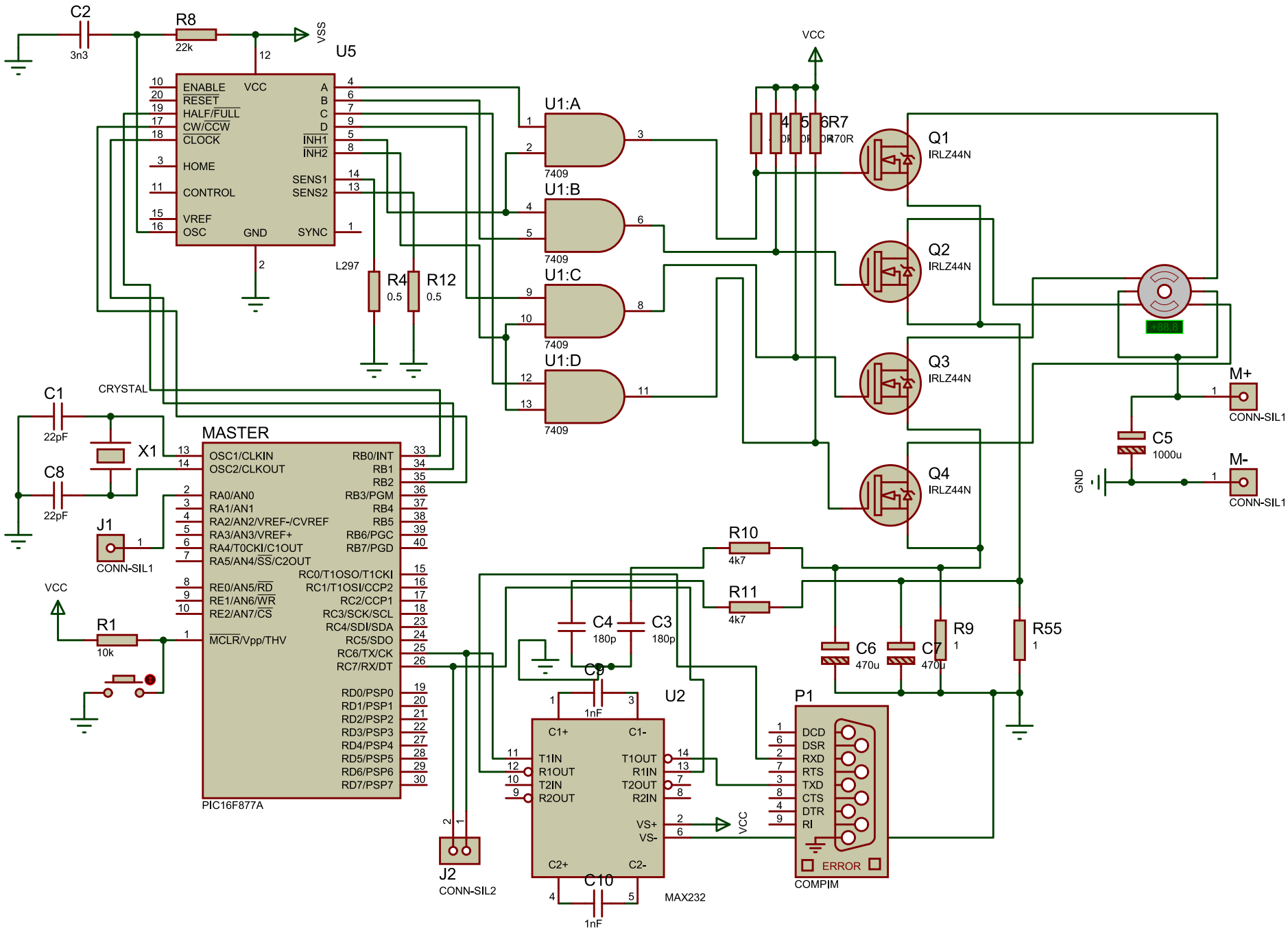
EK - 1

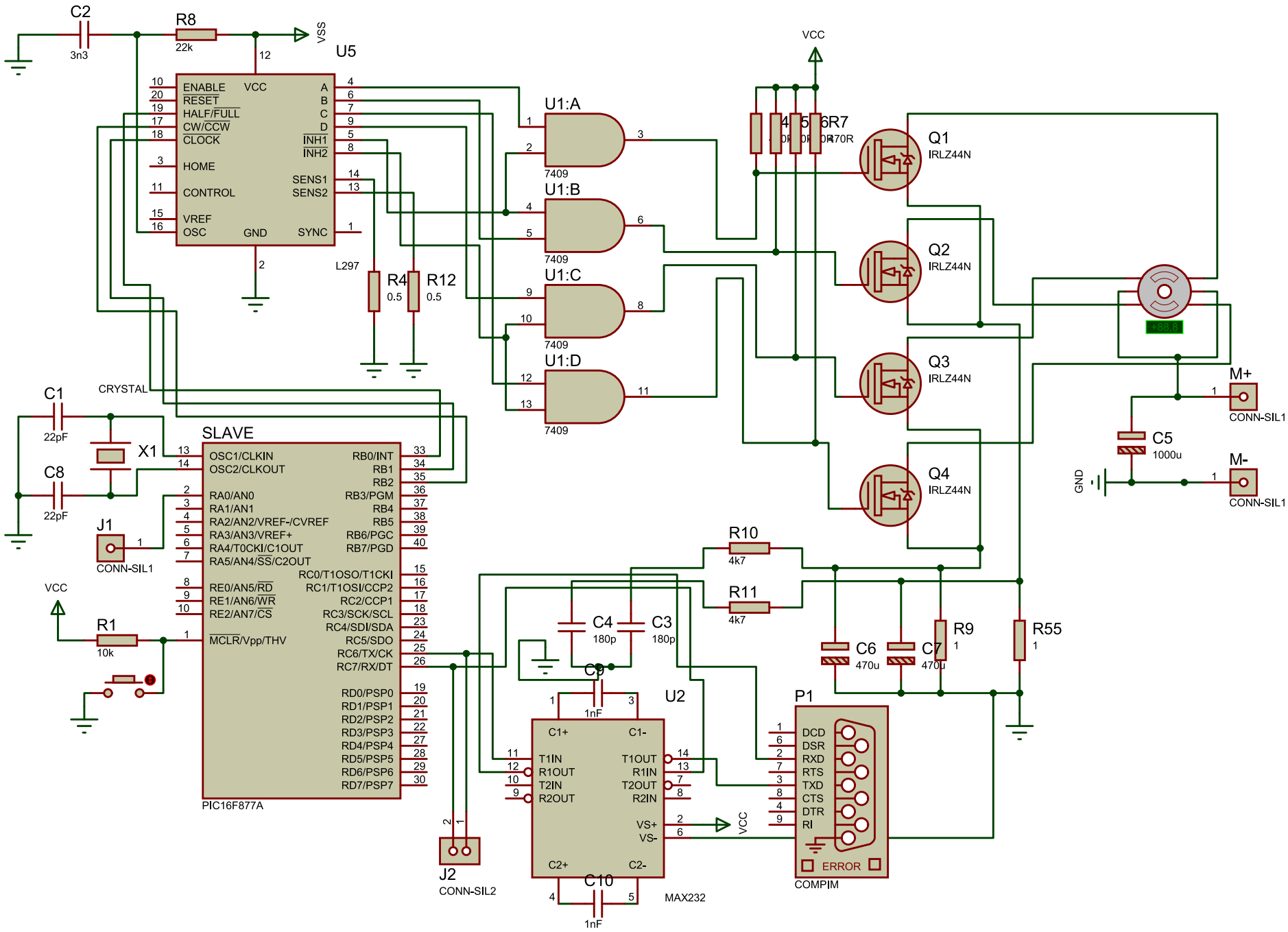
CAD MODELLEME

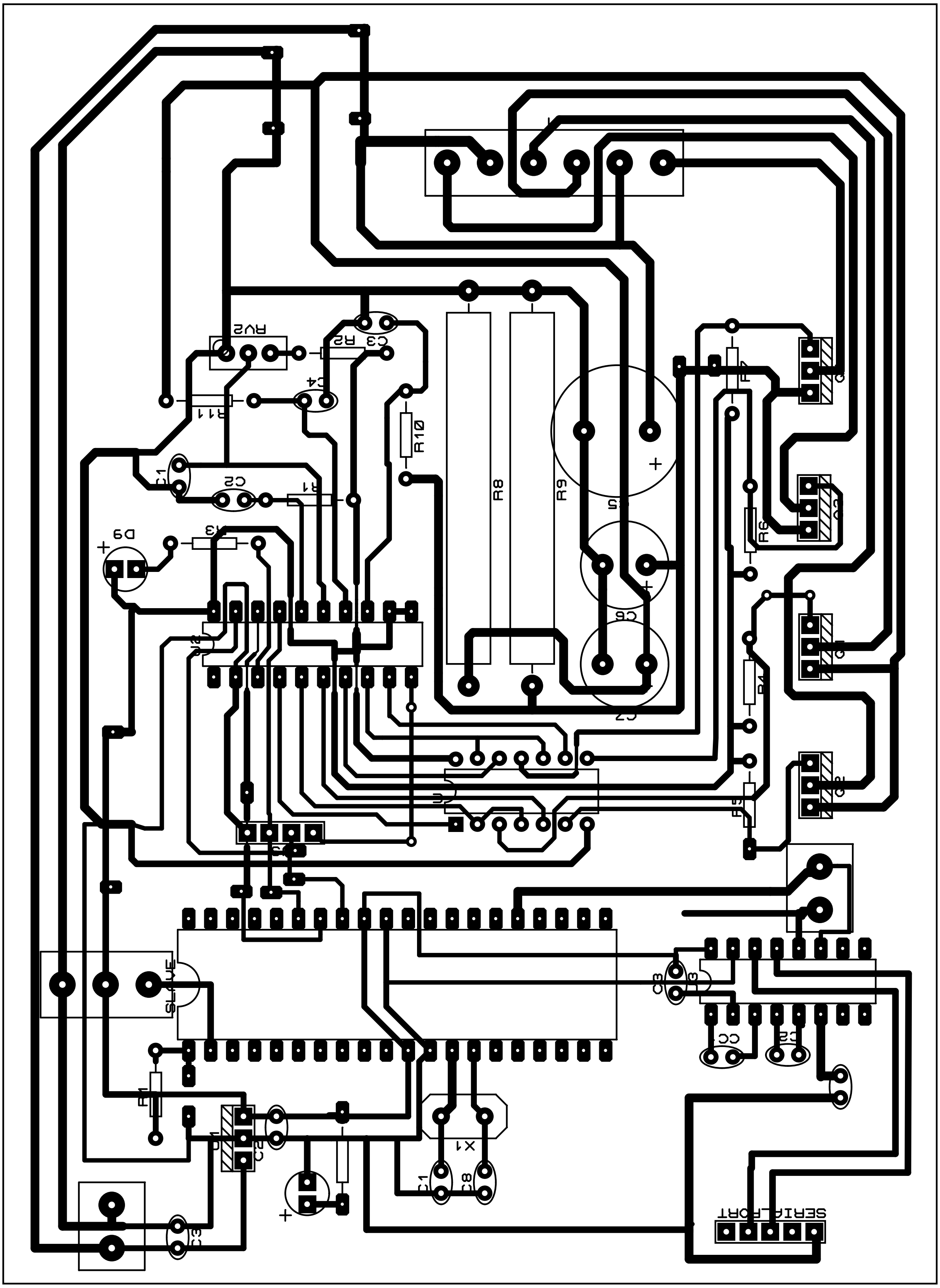


EK - 2

DEVRE ŞEMALARI


C2
3n3
R8
22k
VSS
U5
ENABLE
RESET
HALF/FULL
CW/CCW
CLOCK
HOME
CONTROL
VREF
OSC
VCC
GND
A
B
C
D
INH1
INH2
SENS1
SENS2
SYNC
L297
R4
0.5
R12
0.5
U1:A
U1:B
U1:C
U1:D
7409
VCC
R5
R6
R7
470R
Q1
IRLZ44N
Q2
IRLZ44N
Q3
IRLZ44N
Q4
IRLZ44N
M+
M-
CONN-SIL1
C5
1000u
GND
CRYSTAL
C1
22pF
C8
22pF
X1
MASTER
OSC1/CLKIN
OSC2/CLKOUT
RA0/AN0
RA1/AN1
RA2/AN2/VREF-/CVREF
RA3/AN3/VREF+
RA4/T0CKI/C1OUT
RA5/AN4/SS/C2OUT
RE0/AN5/RD
RE1/AN6/WR
RE2/AN7/CS
MCLR/Vpp/THV
RB0/INT
RB1
RB2
RB3/PGM
RB4
RB5
RB6/PGC
RB7/PGD
RC0/T1OSO/T1CKI
RC1/T1OSI/CCP2
RC2/CCP1
RC3/SCK/SCL
RC4/SDI/SDA
RC5/SDO
RC6/TX/CK
RC7/RX/DT
RD0/PSP0
RD1/PSP1
RD2/PSP2
RD3/PSP3
RD4/PSP4
RD5/PSP5
RD6/PSP6
RD7/PSP7
PIC16F877A
J1
CONN-SIL1
R1
10k
R10
4k7
R11
4k7
C4
180p
C3
180p
C9
1nF
U2
C1+
C1-
T1IN
R1OUT
T2IN
R2OUT
T1OUT
R1IN
T2OUT
R2IN
VS+
VS-
C2+
C2-
MAX232
C10
1nF
J2
CONN-SIL2
P1
DCD
DSR
RXD
RTS
TXD
CTS
DTR
RI
ERROR
COMPIM
C6
470u
C7
470u
R9
1
R55
1

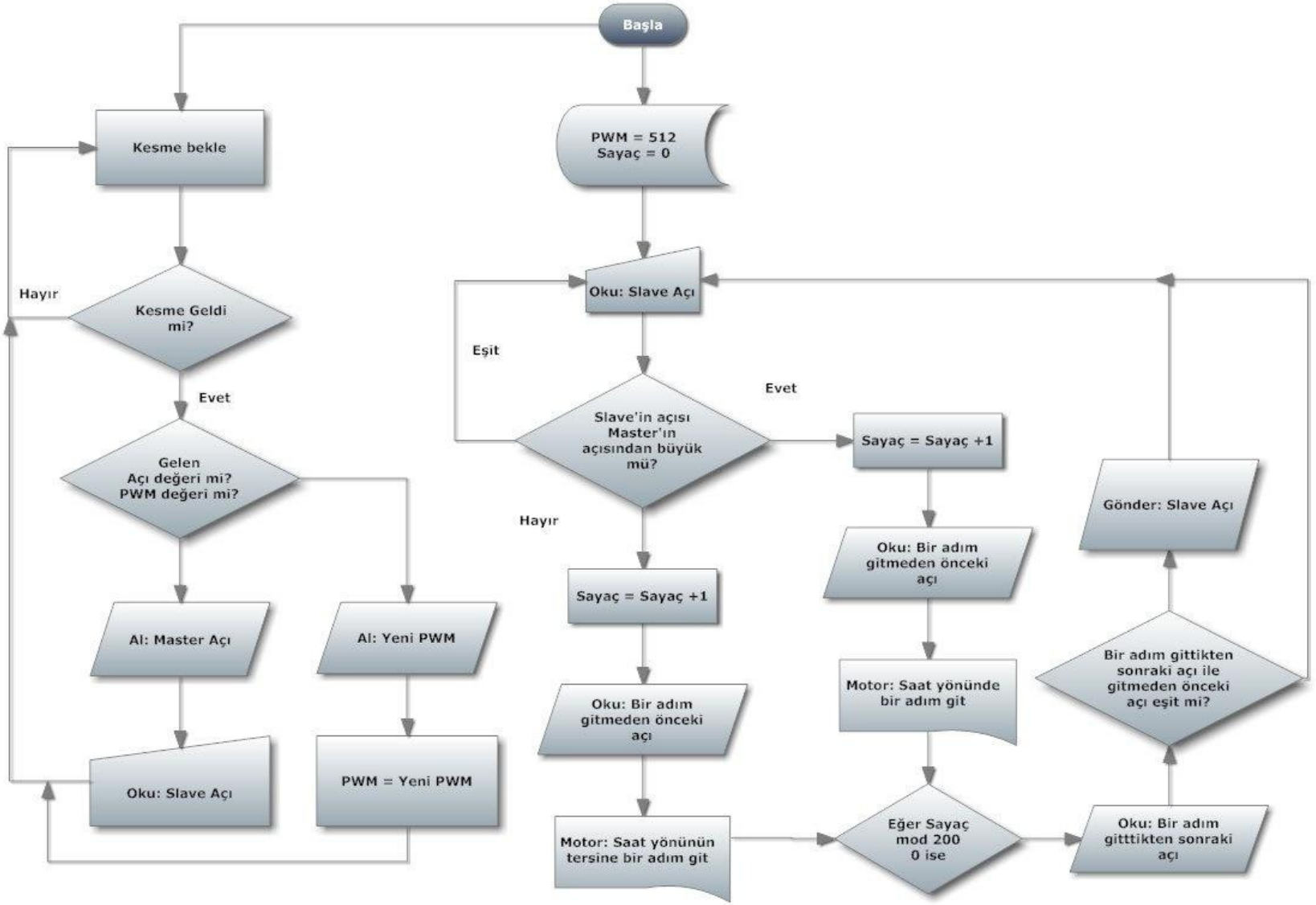

C2
3n3
R8
22k
VSS
U5
ENABLE
RESET
HALF/FULL
CW/CCW
CLOCK
HOME
CONTROL
VREF
OSC
VCC
GND
A
B
C
D
INH1
INH2
SENS1
SENS2
SYNC
L297
R4
0.5
R12
0.5
U1:A
U1:B
U1:C
U1:D
7409
VCC
R5
R6
R7
470R
Q1
IRLZ44N
Q2
IRLZ44N
Q3
IRLZ44N
Q4
IRLZ44N
M+
M-
CONN-SIL1
C5
1000u
GND
CRYSTAL
C1
22pF
C8
22pF
X1
SLAVE
OSC1/CLKIN
OSC2/CLKOUT
RA0/AN0
RA1/AN1
RA2/AN2/VREF-/CVREF
RA3/AN3/VREF+
RA4/T0CKI/C1OUT
RA5/AN4/SS/C2OUT
RE0/AN5/RD
RE1/AN6/WR
RE2/AN7/CS
MCLR/Vpp/THV
RB0/INT
RB1
RB2
RB3/PGM
RB4
RB5
RB6/PGC
RB7/PGD
RC0/T1OSO/T1CKI
RC1/T1OSI/CCP2
RC2/CCP1
RC3/SCK/SCL
RC4/SDI/SDA
RC5/SDO
RC6/TX/CK
RC7/RX/DT
RD0/PSP0
RD1/PSP1
RD2/PSP2
RD3/PSP3
RD4/PSP4
RD5/PSP5
RD6/PSP6
RD7/PSP7
PIC16F877A
J1
CONN-SIL1
VCC
R1
10k
R10
4k7
R11
4k7
C4
180p
C3
180p
C9
1nF
U2
C1+
C1-
T1IN
R1OUT
T2IN
R2OUT
T1OUT
R1IN
T2OUT
R2IN
VS+
VS-
C2+
C2-
MAX232
C10
1nF
VCC
J2
CONN-SIL2
P1
DCD
DSR
RXD
RTS
TXD
CTS
DTR
RI
ERROR
COMPIM
C6
470u
C7
470u
R9
1
R55
1



EK - 3

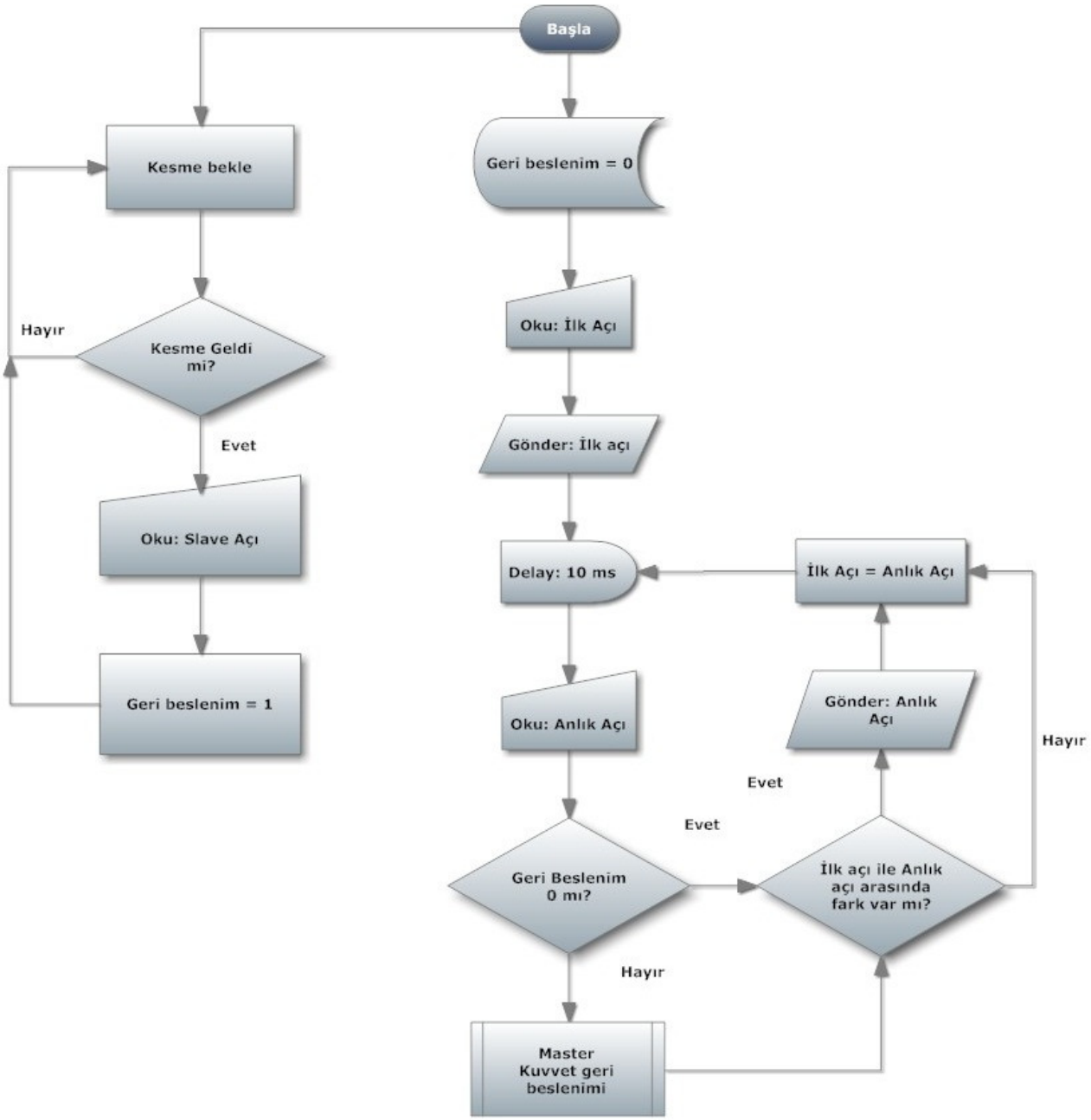
AKIŞ DİYAGRAMI (SLAVE)

SLAVE AKIŞ DİYAGRAMI

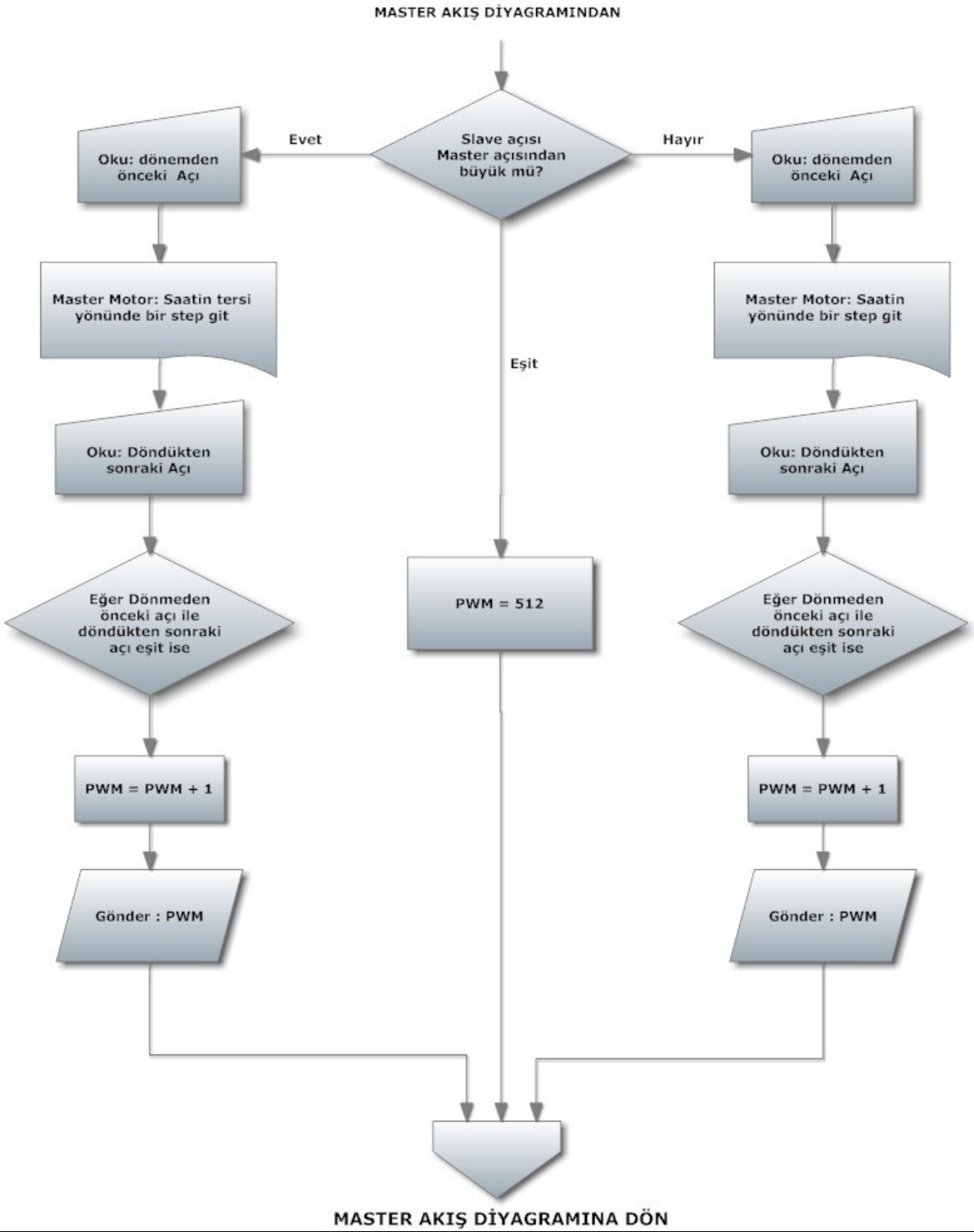


EK - 4
AKIŞ DİYAGRAMI
(MASTER)

MASTER ANA AKIŞ DİYAGRAMI



MASTER GERİ BESLENİM BLOĞU



EK - 5

CCS İLE YAZILMIS KODLAR

```
 1: #include <16F877A.h>
 2: #device ADC=10
 3: #fuses HS,NOWDT,NOPROTECT,NOLVP
 4: #use delay(clock=20000000)
 5: #use rs232(baud=19200,xmit=PIN_C6,rcv=PIN_C7)
 6: #use fast_io(B)
 7: #use fast_io(C)
 8: #use fast_io(D)
 9: #org 0x1F00,0x1FFF{}
10: #opt 9
11: #include <stdlib.h>
12: #include <input.c>
13: #include <math.h>
14:
15: #define HALFFULL PIN_B0
16: #define CWCCW PIN_B2
17: #define CLOCK PIN_B1
18:
19: signed long sadval,madval,pwmrec=512;
20: signed long k1,k2,k,ilk=0,son=100;
21: int x,count=0;
22: char towhom;
23: long b1,b2;
24:
25: void sendtomaster()
26: {
27: //if(!input(PIN_B7))
28: //{
29: output_high(PIN_B6);
30: sadval=read_ADC();
31: putc('m');
32: putc(sadval/100);
33: putc(sadval%100);
34: output_low(PIN_B6);
35: //}
36: }
37:
38: signed long kontrol() // CALISTIRILDIGINDA 1 STEP GIDER.
39: {
40: k=read_ADC();
41: return k;
42: }
43:
44: void sagagit() // CALISTIRILDIGINDA 1 STEP GIDER.
45: {
46: k1=kontrol();
47: output_low(CWCCW);
48:
49: output_low(CLOCK);
50: delay_ms(2);
51: output_high(CLOCK);
52: delay_ms(2);
53:
54: k2=kontrol();
55:
56: if(count==1)
57:    ilk=k1;
58:
59: if(count%200==0)
60:    {
61:    son=k2;
62:    if(ilk==son)
63:    sendtomaster();
64:    count=0;
65:    }
66:
67:
68:
```

```
 69: count++;
 70: }
 71:
 72: void solagit() // CALISTIRILDIGINDA 1 STEP GIDER.
 73: {
 74: k1=kontrol();
 75: output_high(CWCCW);
 76:
 77: output_low(CLOCK);
 78: delay_ms(2);
 79: output_high(CLOCK);
 80: delay_ms(2);
 81:
 82: k2=kontrol();
 83:
 84: if(count==1)
 85:    ilk=k1;
 86:
 87: if(count%200==0)
 88:    {
 89:    son=k2;
 90:    if(ilk==son)
 91:    sendtomaster();
 92:    count=0;
 93:    }
 94:
 95:    if(ilk==son)
 96:    sendtomaster();
 97:
 98: count++;
 99: }
100:
101: void stepgit() // CALISTIRILDIGINDA 1 STEP GIDER.
102: {
103: k1=kontrol();
104: output_low(CLOCK);
105: delay_ms(100);
106: output_high(CLOCK);
107: delay_ms(50);
108: }
109:
110: void stepturu(int x) // 0 - HALF // 1 - FULL STEP.
111: {
112: if(x==0)
113: output_high(HALFFULL);
114: if(x==1)
115: output_low(HALFFULL);
116: }
117:
118: void yon(int x)// 1.SAAT YONU // 0 SAATIN TERSI
119: {
120: if(x==0)
121: output_low(CWCCW);
122: if(x==1)
123: output_high(CWCCW);
124: }
125:
126: void piconfig()
127: {
128: set_tris_c(0b10000000);
129: set_tris_b(0b00000001);
130: setup_adc_ports( AN0 );
131: setup_adc(ADC_CLOCK_DIV_2);
132: setup_ccp1(CCP_PWM);
133: setup_ccp2(CCP_PWM);
134: set_pwm1_duty(512);
135: enable_interrupts(INT_RDA);
136: enable_interrupts(GLOBAL);
```

```
137:
138: stepturu(1);
139: set_adc_channel( 0 );
140: }
141:
142: void main()
143: {
144: piconfig();
145:
146: sadval=read_ADC();
147: madval=650;
148: output_low(PIN_B6);
149: delay_ms(50);
150: while (true)
151: {
152:    sadval=read_ADC();
153:
154:    if((sadval<madval) && (madval-sadval>10))
155:    set_pwm1_duty(512);
156:     sagagit();
157:
158:
159:    if((sadval>madval) && (sadval-madval>10))
160:
161:     solagit();
162:
163:    delay_ms(2);
164:
165: }
166:
167: }
168:
169: #INT_RDA
170: void rs232()
171: {
172: towhom=getc();
173: if(towhom=='s')
174: {
175: b1=getc();
176: b2=getc();
177: madval=b1*100+b2;
178: }
179: if(towhom=='p')
180: {
181: b1=getc();
182: b2=getc();
183: pwmrec=b1*100+b2;
184: }
185: sadval=read_ADC();
186: }
187:
188:
189:
```

```
 1: #include <16F877A.h>
 2: #device ADC=10
 3: #fuses HS,NOWDT,NOPROTECT,NOLVP
 4: #use delay(clock=20000000)
 5: #use rs232(baud=19200,xmit=PIN_C6,rcv=PIN_C7)
 6: #use fast_io(B)
 7: #org 0x1F00,0x1FFF{}
 8: #opt 9
 9: #include <stdlib.h>
10: #include <input.c>
11: #include <math.h>
12:
13: #define HALFFULL PIN_B0
14: #define CWCCW PIN_B2
15: #define CLOCK PIN_B1
16:
17:
18: signed long sadval,madval,oldmadval,pwmsend;
19: int force=0,count=0;
20: char towhom='0';
21: long b1,b2;
22:
23: void sendpwm() // CALISTIRILDIGINDA 1 STEP GIDER.
24: {
25:    putc('p');
26:    putc(pwmsend/100);
27:    putc(pwmsend%100);
28: }
29:
30: void sagagit() // CALISTIRILDIGINDA 1 STEP GIDER.
31: {
32:
33: output_low(CWCCW);
34: output_low(CLOCK);
35: delay_ms(2);
36: output_high(CLOCK);
37: delay_ms(2);
38: }
39:
40: void solagit() // CALISTIRILDIGINDA 1 STEP GIDER.
41: {
42: output_high(CWCCW);
43:
44: output_low(CLOCK);
45: delay_ms(2);
46: output_high(CLOCK);
47: delay_ms(2);
48: }
49:
50: void stepturu(int x) // 0.5 - HALF // 1 - FULL STEP.
51: {
52: if(x==5)
53: output_high(HALFFULL);
54: if(x==1)
55: output_low(HALFFULL);
56: }
57: void yon(int x)// 1.SAAT YONU // 0 SAATIN TERSI
58: {
59: if(x==0)
60: output_low(CWCCW);
61: if(x==1)
62: output_high(CWCCW);
63: }
64: void sendtoslave()
65: {
66:    //if(!input(PIN_B6))
67:    //{
68:    output_high(PIN_B7);
```

```
 69:    putc('s');
 70:    putc(madval/100);
 71:    putc(madval%100);
 72:    output_low(PIN_B7);
 73:    //}
 74: }
 75:
 76: void masterforce()
 77: {
 78:    if(sadval<madval)
 79:    {
 80:       sagagit();
 81:    }
 82:
 83:    if(sadval>madval)
 84:    {
 85:    solagit();
 86:    }
 87:
 88:    if(sadval==madval)
 89:    {
 90:    force=0;
 91:    pwmsend=512;
 92:    }
 93:
 94:    count++;
 95:
 96:    if(count%200==0 && pwmsend<1024)
 97:    {
 98:    pwmsend=pwmsend+1;
 99:    sendpwm();
100:    }
101:
102: }
103:
104: void main()
105: {
106: set_tris_c(0b10000000);
107: set_tris_b(0b01000000);
108: output_low(PIN_B5);
109: setup_adc_ports( AN0 );
110: setup_adc(ADC_CLOCK_DIV_2);
111: setup_ccp1(CCP_PWM);
112: setup_ccp2(CCP_PWM);
113: set_pwm1_duty(1023);
114: enable_interrupts(INT_RDA);
115: enable_interrupts(GLOBAL);
116: stepturu(1);
117:
118: set_adc_channel( 0 );
119: delay_ms(100);
120: //oldmadval=read_ADC();
121: oldmadval=512;
122: delay_ms(50);
123:
124:
125: while (true)
126: {
127:    madval=read_ADC();
128:
129:    if(force==1)
130:    masterforce();
131:
132:
133:    delay_us(100);
134:
135:    if((oldmadval<madval) && (madval-oldmadval>5))
136:    {
```

```
137:    sendtoslave();
138:    }
139:
140:    if((oldmadval>madval) && (oldmadval-madval>5))
141:    {
142:    sendtoslave();
143:    }
144:
145:
146:
147:    oldmadval=madval;
148:    delay_ms(2);
149:    }
150:
151: }
152:
153: #INT_RDA
154: void rs232_isr()
155: {
156: towhom=getc();
157:
158:    if(towhom=='m'){
159:    b1=getc();
160:    b2=getc();
161:    sadval=b1*100+b2;
162:
163:    }
164:    force=1;
165: }
166:
167:
168:
```

EK – 6
C# İLE YAZILMIŞ KODLAR

```
using System;
using System.IO;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        double sn = 0, saniye=0;
        string Saci_pwm = "";
        string Maci_pwm = "";
        string Saci = "";
        string Maci = "";
        string Spwm = "";
        string Mpwm = "";
        string MdataRead = "";
        string SdataRead = "";

        private void basla()
        {
            if (!serialPort_slave.IsOpen)
                MessageBox.Show("Slave Seri Portu (" +
serialPort_slave.PortName.ToString() + ") Kapali!");

            if (!serialPort_master.IsOpen)
                MessageBox.Show("Master Seri Portu (" +
serialPort_master.PortName.ToString() + ") Kapali!");

            if (serialPort_slave.IsOpen && serialPort_master.IsOpen)
                timer1.Start();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            if (!serialPort_slave.IsOpen)
            {
                textBox_Sstatus.Text = "Slave Seri Portu (" +
serialPort_slave.PortName.ToString() + ") Kapali...";
                button_basla.Enabled = false;
            }

            if (!serialPort_master.IsOpen)
            {
                textBox_Mstatus.Text = "Master Seri Portu (" +
serialPort_master.PortName.ToString() + ") Kapali...";
                button_basla.Enabled = false;
```

```
            }
        }

        private void button_basla_Click(object sender, EventArgs e)
        {
            basla();
        }

        private void timer1_Tick(object sender, EventArgs e)
        {
            sn += 1;
            saniye = sn/10;
            textBox1.Text = saniye.ToString();
        }

        private void button_dur_Click(object sender, EventArgs e)
        {
            timer1.Stop();
            textBox_mpoz.Text += "];";
            textBox_mzpoz.Text += "];";

            textBox_spoz.Text += "];";
            textBox_szpoz.Text += "];";

            textBox_spwm.Text += "];";
            textBox_mpwm.Text += "];";

            textBox_tspwm.Text += "];";
            textBox_tmpwm.Text += "];";

            button_mfile.Enabled = true;
         }

        private void masterdata(object sender,
System.IO.Ports.SerialDataReceivedEventArgs e)
        {
            Maci_pwm = serialPort_master.ReadLine();
            CheckForIllegalCrossThreadCalls = false;

            if (Maci_pwm == "a")
            {
                Maci = serialPort_master.ReadLine();
                long aci = long.Parse(Maci);
                textBox_mpoz.Text += aci.ToString() + Environment.NewLine;
                textBox_mzpoz.Text += saniye.ToString().Replace(",", ".") +
Environment.NewLine;
            }

            if (Maci_pwm == "p")
            {
                Mpwm = serialPort_master.ReadLine();
                long pwm = long.Parse(Mpwm);
                textBox_mpwm.Text += pwm.ToString() + Environment.NewLine;
                textBox_tmpwm.Text += saniye.ToString().Replace(",", ".") +
Environment.NewLine;
            }
```

```
        }

        private void slavedata(object sender,
System.IO.Ports.SerialDataReceivedEventArgs e)
        {
            Saci_pwm = serialPort_slave.ReadLine();
            CheckForIllegalCrossThreadCalls = false;

            if (Saci_pwm == "a")
            {
                Saci = serialPort_slave.ReadLine();
                long aci = long.Parse(Saci);
                textBox_spoz.Text += aci.ToString() + Environment.NewLine;
                textBox_szpoz.Text += saniye.ToString().Replace(",", ".") +
Environment.NewLine;
            }

            if (Saci_pwm == "p")
            {
                Spwm = serialPort_slave.ReadLine();
                long pwm = long.Parse(Spwm);
                textBox_spwm.Text += pwm.ToString() + Environment.NewLine;
                textBox_tspwm.Text += saniye.ToString().Replace(",", ".") +
Environment.NewLine;
            }

        }

        private void button_slavecom_Click(object sender, EventArgs e)
        {
            if (!serialPort_slave.IsOpen)
            {
                serialPort_slave.Open();
                textBox_Sstatus.Text = "Slave Seri Portu (" +
serialPort_slave.PortName.ToString() + ") acik...";
            }

            if (serialPort_master.IsOpen)
                button_basla.Enabled = true;
        }

        private void button_mastercom_Click(object sender, EventArgs e)
        {
            if (!serialPort_master.IsOpen)
            {
                serialPort_master.Open();
                textBox_Mstatus.Text = "Master Seri Portu (" +
serialPort_master.PortName.ToString() + ") acik...";
            }

            if (serialPort_slave.IsOpen)
                button_basla.Enabled = true;

        }

        private void button_sifirla_Click(object sender, EventArgs e)
        {
```

```
            timer1.Stop();
            sn = 0;
            saniye = 0;
            textBox_mpoz.Text = "";
            textBox_mpwm.Text = "";
            textBox_spoz.Text = "";
            textBox_spwm.Text = "";
            textBox_mzpoz.Text = "";
            textBox_tmpwm.Text = "";
            textBox_szpoz.Text = "";
            textBox_tspwm.Text = "";
            textBox1.Text = "";
        }

        private void button_mfile_Click(object sender, EventArgs e)
        {
            string simdi = DateTime.Now.ToString().Replace(":", "");
            simdi = simdi.Replace(".", "");
            simdi = simdi.Replace(" ", "");

            File.WriteAllText("C:\\Users\\EFE ANIL\\Desktop\\Mfiles\\" + "d" +
simdi + ".m", textBox_mpoz.Text);
            File.AppendAllText("C:\\Users\\EFE ANIL\\Desktop\\Mfiles\\" + "d"
+ simdi + ".m", textBox_mzpoz.Text);
            File.AppendAllText("C:\\Users\\EFE ANIL\\Desktop\\Mfiles\\" + "d"
+ simdi + ".m", textBox_spoz.Text);
            File.AppendAllText("C:\\Users\\EFE ANIL\\Desktop\\Mfiles\\" + "d"
+ simdi + ".m", textBox_szpoz.Text);
            File.AppendAllText("C:\\Users\\EFE ANIL\\Desktop\\Mfiles\\" + "d"
+ simdi + ".m", textBox_mpwm.Text);
            File.AppendAllText("C:\\Users\\EFE ANIL\\Desktop\\Mfiles\\" + "d"
+ simdi + ".m", textBox_tmpwm.Text);
            File.AppendAllText("C:\\Users\\EFE ANIL\\Desktop\\Mfiles\\" + "d"
+ simdi + ".m", textBox_spwm.Text);
            File.AppendAllText("C:\\Users\\EFE ANIL\\Desktop\\Mfiles\\" + "d"
+ simdi + ".m", textBox_tspwm.Text);
            File.AppendAllText("C:\\Users\\EFE ANIL\\Desktop\\Mfiles\\" + "d"
+ simdi + ".m", textBox_mfile.Text);

            textBox_matlab_done.Text = "d" + simdi + ".m olusturuldu...";
        }




    }
}
```