T. C.
İSTANBUL GELİŞİM ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

Mekatronik Mühendisliği Anabilim Dalı

POZİSYON VERİSİ İLE KONTROL EDİLEN 6 EKSEN ROBOT MAKİNE BESLEME UYGULAMASI

Yüksek Lisans Tezi

Berk Doğa Mutlu IŞILAK

Danışman
Dr. Öğr. Üyesi Safar POURABBAS

İstanbul – 2022

TEZ TANITIM FORMU

Yazar Adı Soyadı : Berk Doğa Mutlu IŞILAK

Tezin Dili : Türkçe

Tezin Adı : Pozisyon Verisi ile Kontrol Edilen 6 Eksen Robot Makine Besleme Uygulaması

Enstitü : İstanbul Gelişim Üniversitesi Lisansüstü Eğitim Enstitüsü

Anabilim Dalı : Mekatronik Mühendisliği

Tezin Türü : Yüksek Lisans

Tezin Tarihi : 26.01.2022

Sayfa Sayısı : 90

Tez Danışmanları : Dr. Öğr. Üyesi Safar POURABBAS

Dizin Terimleri : Altı eksen robotlar, robotlar, robotik otomasyon, otomasyon, endüstriyel haberleşme protokolleri

Türkçe Özet : Bu çalışmada bir endüstriyel robota pozisyon değerleri bilgisayar programı vasıtası ile gönderilerek, daha kolay güncellenebilen ve kaza ihtimalinin daha düşük olduğu bir programlama yöntemi geliştirilmesi amaçlanmıştır.

Dağıtım Listesi :
1. İstanbul Gelişim Üniversitesi Lisansüstü Eğitim Enstitüsüne
2. YÖK Ulusal Tez Merkezine

İmzası

Berk Doğa Mutlu IŞILAK

T. C.
İSTANBUL GELİŞİM ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

Mekatronik Mühendisliği Anabilim Dalı

POZİSYON VERİSİ İLE KONTROL EDİLEN 6 EKSEN ROBOT MAKİNE BESLEME UYGULAMASI

Yüksek Lisans Tezi

Berk Doğa Mutlu IŞILAK

Danışman
Dr. Öğr. Üyesi Safar POURABBAS

İstanbul – 2022

BEYAN

Bu tezin hazırlanmasında bilimsel ahlak kurallarına uyulduğu, başkalarının eserlerinden yararlanılması durumunda bilimsel normlara uygun olarak atıfta bulunulduğu, kullanılan verilerde herhangi tahrifat yapılmadığını, tezin herhangi bir kısmının bu üniversite veya başka bir üniversitedeki başka bir tez olarak sunulmadığını beyan ederim.

Berk Doğa Mutlu IŞILAK

…/…/2022

İSTANBUL GELİŞİM ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ MÜDÜRLÜĞÜNE

Berk Doğa Mutlu IŞILAK ’ın Pozisyon Verisi ile Kontrol Edilen 6 Eksen Robot Makine Besleme Uygulaması adlı tez çalışması, jürimiz tarafından Mekatronik Mühendisliği anabilim dalı, Mekatronik Mühendisliği bilim dalında YÜKSEK LİSANS tezi olarak kabul edilmiştir.

	İmza	
Başkan	*Dr. Öğr. Üyesi Safar POURABBAS* (Danışman)	
	İmza	
Üye	*Dr. Öğr. Üyesi Khalıd O.Moh. YAHYA*	
	İmza	
Üye	*Dr. Öğr. Üyesi Rıza İLHAN*	

ONAY

Yukarıdaki imzaların, adı geçen öğretim üyelerine ait olduğunu onaylarım.

... / ... / 2022

İmzası

Prof. Dr. İzzet GÜMÜŞ

Enstitü Müdürü

ÖZET

Robotik otomasyon projelerinde projenin farklı ürün ve durumlara adapte edilmesi oldukça zaman alır. Ayrıca bu değişiklikler bilinçli kişiler tarafından yapılmaz ise kazalara sebep olmaktadır. Bu tez çalışmasında değişken ürünlere kolay geçiş sağlanması için pozisyon değerlerinin dışarıdan gönderildiği bir robot projesi oluşturulmak istenmiştir. Böylece robot programları, fabrika değil ofis ortamında daha bilinçli kişiler tarafından oluşturulabilecektir. Çalışma için değişken pozisyon değerlerine göre çalışabilecek robot programları ve robotun dışarıdan kontrol edilmesi için gerekli olan alt yapı simülasyon ortamında kurulmuştur. Bu simülasyon Modbus protokolü üzerinden haberleşmeye uygun hale getirilmiştir. C# programlama dili kullanılarak Modbus protokolü üzerinden robota pozisyon değerlerini gönderen ve robotu kontrol eden bir arayüz oluşturulmuştur. Simülasyon ve arayüz programı gerçek dünyaya birebir uygun şekilde birbiri ile haberleştirilmiştir.

Anahtar Kelimeler: Altı eksen robotlar, robotlar, robotik otomasyon, otomasyon, endüstriyel haberleşme protokolleri

SUMMARY

In robotic automation projects, it takes quite some time to adapt the project to different products and situations. Also, if these changes are not made by conscious people, they can cause accidents. In this thesis work, it is aimed to create a robot project in which position values are sent from outside in order to provide easy transition to variable products. Thus, robot programs can be created by more conscious people in the office environment not in the factory. Robot programs that can operate according to variable position values for the study and the infrastructure required for external control of the robot have been set up in the simulation environment. This simulation is made suitable for communication via the Modbus protocol. Using the C # programming language, an interface has been created that sends position values to the robot via the Modbus protocol and controls the robot. The simulation and interface program are communicated with each other in a way that is exactly appropriate for the real world.

Keywords: Six axis robots, robots, robotics automation, automation, industrial communication protocols

İÇİNDEKİLER

ÖZET ..... **i**
SUMMARY ..... **ii**
İÇİNDEKİLER ..... **iii**
TABLOLAR LİSTESİ ..... **v**
ŞEKİLLER LİSTESİ ..... **vi**
EKLER LİSTESİ ..... **viii**
ÖNSÖZ ..... **ix**
GİRİŞ **1**

BİRİNCİ BÖLÜM
ROBOTLAR

1.1. Robotların Tarihçesi 3
1.2. Endüstriyel Robotlar 4
1.3. Endüstriyel Robotların Sınıflandırılması 6
1.3.1. Kartezyen Robotlar 6
1.3.2. Silindirik Robotlar 7
1.3.3. Scara Robotlar 8
1.3.4. Küresel Robotlar 8
1.3.5. Mafsallı Robotlar 9
1.4. Endüstriyel Robotun Programlanması 17
1.5. Robot Kinematiği 21
1.5.1. İleri Kinematik 21
1.5.2. Ters Kinematik 28

İKİNCİ BÖLÜM
ENDÜSTRİYEL HABERLEŞME PROTOKOLLERİ

2.1. Ethernet IP 29
2.2. DeviceNet (CAN Bus) 30
2.3. Modbus 30
2.3.1. Modbus RTU 32
2.3.2. ModbusTCP 32

ÜÇÜNCÜ BÖLÜM
BİLGİSAYAR PROGRAMI ÜZERİNDEN ROBOT KONTROLÜ

3.1. Robot Tarafında Haberleşme ve Sinyallerin Hazırlanması 34
3.2. Robot Programının Hazırlanması 51
3.3. Bilgisayar Programının Hazırlanması 53

SONUÇLAR VE ÖNERİLER **57**

KAYNAKÇA 58
EKLER..... 61

TABLOLAR LİSTESİ

Tablo 1. Fanuc R2000ic 165f robotu Denavit-Hartenberg parametreleri.....24

ŞEKİLLER LİSTESİ

Şekil 1. CNC torna besleme örneği....6
Şekil 2. Kartezyen robot yapısı....7
Şekil 3. Silindirik robot yapısı....8
Şekil 4. Scara robot yapısı....8
Şekil 5. Küresel robot yapısı....9
Şekil 6. 6 eksen mafsallı robot yapısı....10
Şekil 7. FANUC LR Mate 200iD robot manipülatör....11
Şekil 8. FANUC R-30iB Mate robot kontrolör....11
Şekil 9. FANUC robot kontrol paneli....12
Şekil 10. Pnömatik tutucu örneği....13
Şekil 11. Robotik kaynak torcu örneği....14
Şekil 12. Robotik freze örneği....15
Şekil 13. Doğrusal eksen örneği....16
Şekil 14. Dairesel eksenler....16
Şekil 15. Nachi FD Ondesk program görüntüsü....18
Şekil 16. Sağ el kuralı görseli....18
Şekil 17. Robot noktadan noktaya komutu....20
Şekil 18. Robot doğrusal hareket komutu....20
Şekil 19. Robot dairesel hareket komutu....21
Şekil 20. Fanuc R2000ic 165f katalog değerleri....22
Şekil 21. Fanuc R2000ic 165f kol uzunluk değerleri....23
Şekil 22. Fanuc robot eksenlerin açı gösterimi....23
Şekil 23. Fanuc robot ekranında Denavit-Hartenberg parametreleri....24
Şekil 24. Örnek ileri kinematik dönüşüm değerleri....25
Şekil 25. Örnek ileri kinematik dönüşümü robot pozisyonu....26
Şekil 26. Modbus data tipleri....31
Şekil 27. Modbus fonksiyon kodları....31
Şekil 28. Modbus RTU veri paketi içeriği....32
Şekil 29. Modbus TCP ve Modbus RTU kıyaslaması....33
Şekil 30. Fanuc ethernet ayarlarına giriş....35
Şekil 31. Fanuc TCP/IP ayarlarına giriş....35
Şekil 32. Fanuc TCP/IP ayarları örneği....36
Şekil 33. Fanuc Modbus ayarlarına giriş....37
Şekil 34. Fanuc Modbus ayarları örneği....37
Şekil 35. UOP çıkışları görüntüsü....39
Şekil 36. UOP girişleri görüntüsü....40
Şekil 37. UOP çıkışları atanması....41
Şekil 38. UOP girişleri atanması....42
Şekil 39. Atama yapıldıktan sonra flag görüntüsü....43
Şekil 40. Dijital çıkış atamaları....44

Şekil 41. Atama yapıldıktan sonra dijital çıkışların görüntüsü.....45
Şekil 42. Sayısal ve pozisyon değişkenleri.....46
Şekil 43. Gelen pozisyon değeri hesabı.....47
Şekil 44. Anlık açı değerleri aktarımı.....48
Şekil 45. Takım koordinat sistemi ekranı.....49
Şekil 46. Masa merkezi konumu.....50
Şekil 47. Kullanıcı koordinat sistemi ekranı.....50
Şekil 48. Değişken pozisyonlu program akış şeması.....52
Şekil 49. Bilgisayar programı arayüzü.....54
Şekil 50. Yazılım akış şeması.....55

EKLER LİSTESİ

EK-A DIŞARIDAN GELEN KOMUTLAR İLE EKSENLERİ HAREKET ETTİREN ROBOT PROGRAMININ KODLARI
EK-B POZİSYON DEĞERLERİNİ ALAN VE GÖNDEREN PROGRAM KODLARI
EK-C MAKİNE YÜKLEME BOŞALTMA PROGRAMINA AİT KODLAR
EK-Ç BİLGİSAYAR PROGRAMINA AİT KODLAR

ÖNSÖZ

Çalışmalarım boyunca yol gösteren, bilgi birikimini paylaşan danışman hocam sayın Dr. Öğr Üyesi Safar POURABBAS hocama ve Prof. Dr. Bedri YÜKSEL hocama teşekkürlerimi sunarım. Ayrıca süreç boyunca yardımlarını ve desteklerini esirgemeyen aileme ve Kübra DOĞAN ‘a ayrıca teşekkür ederim.

GİRİŞ

İnsanlar var olduğu zamandan beri işlerini kolaylaştırmak veya yapmak istediklerini gerçekleştirebilmek için yeni aletler ve makineler icat etmişlerdir. Geçmişten günümüze gelen bu süreçte makinelere olan ihtiyacımız artmıştır. Başlangıçta sadece insan kontrolünde olan bu makineler, artan kalite ve verimlilik ihtiyacı ile beraber otomasyon sistemlerinin doğmasına neden olmuştur. Otomasyon sistemleri neredeyse her alanda insanların yerini almakta veya insanlarla beraber çalışmaktadır. Otomasyon sayesinde tekrar eden işler insanlar için sıkıcı, tehlikeli veya ağır olmaktan çıkmıştır. Otomasyonun kullanıldığı işlerde insanın müdahale imkanı azaldıkça, kalite ve verimlilik artışı görülmektedir. İşler her zaman aynı kaliteye ve verimliliğe sahip olmaktadır.

İnsanların yaptığı fiziksel işlerin otomasyona dönüştürülmesi ihtiyacı robotların ortaya çıkmasına sebep olmuştur. Robotlar insan hareketlerini taklit edebilecek kabiliyetlerle imal edilirler. Sensörler ve haberleşme sistemleri sayesinde çevreleri ile etkileşime giren robotlar, çevrelerindeki ekipmanlar, makineler ve robotlarla birlikte çalışabilirler. Robotlar ekipmanları ve makineleri aynı operatörler gibi kullanarak insanların yaptığı işleri gerçekleştirebilirler.

Günümüzde rekabetten kaynaklı daha düşük maliyetlerle üretim ihtiyacı artmıştır. Daha düşük maliyetler ile üretim için daha yüksek adetlerde üretmek ve bu adetlerdeki hurda ürün miktarını azaltmak gerekmektedir. Robotlar programlandıkları işleri her zaman programlandığı şekilde yaparlar. Bu sebeple bütün ürünler aynı hızda ve aynı kalitede üretilirler. Robotların verimlilikleri sabittir ve her zaman aynı şekilde çalıştıkları için insandan kaynaklı olan hatalı üretimleri gerçekleştirmezler. Bu da rekabet etmek isteyen firmalarda robotik otomasyon sistemlerine olan ilgiyi arttırmıştır.

Üretimde robotların talep edilmesinin bir diğer sebebi de ortam koşulları veya işin kendisidir. Bazı ortamlar insan sağlığı açısından çalışmaya uygun olmayabilir veya bazı yapılan işler insanlar için tehlikeli olabilir. Bazı özel durumlarda robot sistemleri verim veya kalite artışı sağlamasa bile ortam koşullarından veya sağlık açısından tercih edilebilirler. Örneğin aşırı sıcak, aşırı soğuk veya zehirli ortamlarda robotlar sıklıkla kullanılırlar. Boya veya kumlama uygulamaları sağlık açısından

tehlikeli ortamlara örnek gösterilebileceği gibi metal döküm uygulamaları da yüksek sıcaklık içeren ortamlara örnek verilebilir.

Robotlar yeniden programlanabilir makineler olmalarına karşın sadece öğretilen işi yapabilme kapasitesine sahiptir. Bu tez çalışmasında makine besleme uygulaması gerçekleştirileceği için makine besleme üzerinden örnek vermek gerekirse; makine besleme uygulamalarında robotlar belli bir konumdan ham haldeki ürünleri alarak işlemi gerçekleştirecek makineye yükler. Robotun ürünü alacağı konum sabit ve hep aynı şekilde olmalıdır. Ayrıca robotun yükleyeceği ürünler de her zaman aynı ürün olmalıdır. Eğer farklı bir ürün yüklenmek istenirse veya aynı ürünün farklı bir bölgeden alınması istenirse bu işlemler robota ayrıca öğretilmelidir. Öğretme işlemleri robot kurulumunu gerçekleştiren kişiler tarafından veya robotun kullanıldığı firma içerisindeki yetkin kişiler tarafından yapılmaktadır. Bu da ek personel veya servis maliyetini beraberinde getirmektedir. Öğretme işlemleri daha az yetkinlikte insanlar tarafından üstün körü yapıldığı zaman çoğunlukla kazalar meydana gelmektedir. Robot kazalarında robot, çevre ekipmanlar, makineler ve işlenecek ürünler zarar gördüğü gibi insanlar da yaralanabilmektedir.

Robot kazalarının önüne geçmek ancak aynı zamanda da yeni ürün için robot programının hızlı bir şekilde güncellenebilmesi isteği, farklı bir robot otomasyon sistemi ihtiyacı doğurmuştur. Bu tez çalışmasında bu ihtiyaca yönelik bir robot sistemi oluşturulmuştur. Robotun programlanması ve simülasyonu için Fanuc Roboguide programı kullanılmıştır. Simülatör içerisinde malzeme yüklenmesi amacıyla bir cnc torna makinası yerleştirilmiştir. Robotun yükleyeceği ürünler bir masa üzerinde simülatör içerisine yerleştirilmiştir. Oluşturulan referans noktasına göre ürünlerin koordinatları bilgisayar yazılımı üzerinden yollanmaktadır. Bilgisayar yazılımının yazımı için kolay ara yüz oluşturulabilmesi ve yazılan yazılımın kolay çalıştırılabilmesi için C# programlama dili tercih edilmiştir. Robot ile bilgisayar yazılımı arasındaki haberleşmede gerçek uygulamalarda düşük maliyetli olması dolayısıyla Modbus TCP/IP haberleşmesi tercih edilmiştir.

BİRİNCİ BÖLÜM

ROBOTLAR

Robotlar dışarıdan kontrol edilebilen, kullanım sağlanacağı alana göre programlanabilen akıllı makinelerdir. Bu makineler, yapacağı işi, oluş ve hareketi nesnelere göre işleyen, gerekli görüldüğü her seferinde tekrardan programlanabilen ve yenilenen cihazlardır. Robotlar, günümüzde insanların hayatlarını birçok alanda kolaylaştırmaktadır.

Robotlar insanların ve bazı makinelerin hareketlerini taklit edebilecek şekilde tasarlanırlar. Robotlar yapacakları işe göre programlanarak onlar için özel olarak imal edilen ekipmanları kullanırlar ve insanlar için zorlayıcı, sıkıcı, tekrar eden işleri kolay bir şekilde gerçekleştirirler. İnsanlar için tehlikeli veya sağlıksız işler robotlar tarafından gerçekleştirilebilir.

1.1. Robotların Tarihçesi

Robot kelimesi ilk olarak 1921 yılında yazılan RUR isimli oyunda Karel Capek tarafından kullanılmıştır. Bu oyunda yer verilmek istenilen robotların insanlar gibi fabrikalarda çalışacaklarını ve ilerleyen zamanlarda ise dünyaya hükmedeceklerini anlatılmıştır. 13.yy da yaşamış olan Ebu El Cezari tasarlamış olduğu otomatik makineler ile tarihe robot kavramını gerçekçi olarak yansıtmıştır. Ebu El Cezari ‘ye ait olan kitaplar 19. ve 20. yüzyılda çevrilerek Avrupa' da birçok mühendislik fakültelerinde ders olarak gösterilmiştir (Ozan, 2020).

Robotların tarihi gelişimlerine bakacak olursak;

• ‘’ Satranç Oynayan Türk’’ Macar mekanikçi tarafından tasarlanmış fakat dönemin teknolojik gelişmeleri ile çözüme kavuşturulamadığından dolayı müzeye kaldırılmış ve tarihe karışmıştır. (1770)

• Westinghouse tarafından sigara içebilen bir robot tasarlanmış. (1937)

• Işığa duyarlı robot olarak bilinen Elsie ve Elmer tasarlanmış. (1949)

• Unimat adında bir robot üretilmiş ve ilk robot şirketi kurulmuş. (Unimation) (1954)

• Rancho adında 6 eksenli ve bilgisayar destekli robot kol tasarlanmıştır. (1963)

• USA ordusu için 4 bacaklı ve hareket kabiliyeti olan robot üretilmiştir. (1966)

• Hidrolik olmayan ve Alman üretimi olan KUKA ilk endüstriyel robot ve Japonya tarafından Wabot insansı robot yapılmıştır. (1973)

• Wabot 2 Japonya tarafından tasarlanmış ve yenilenmiş hali ile sunulmuştur.

• Honda, Asimo adında olan robot projesini başlatmıştır. (1986)

• Türk firması olan Altınay; 6 Eksene sahip robotu tasarlamıştır. (1990)

• İlk sosyal robot olan Aibo üretilmiştir. (1999)

• Görüntü işleme tabanlı İki farklı prototipi yapan Akınsof, yazılımsal gelişmeler üzerinde çalışarak insansı robot çalışmalarına başlamıştır. (2009)(Ozan, 2020).

1.2. Endüstriyel Robotlar

Endüstriyel robot ISO 8373'e göre:

Üç veya daha fazla eksene sahip, programlanabilir, birçok amaç için kullanılabilen, taşınabilen veya yere sabit olan manipülatöre robot denir (ISO, 2012).

Endüstrinin pek çok alanında robotlar kullanılmaktadır. Robotlar yaptıkları işe göre farklı çeşitlerde oldukları gibi farklı donanımlar da kullanmaktadırlar. Operatörlerin ve çalışanların hareket kabiliyetlerine sahip mekanik bir yapıda üretilen robotlar, operatörlerin hareketlerini taklit edecek şekilde programlanırlar. Bütün bunlara karşın robotlar sadece programlandıkları şekilde çalışabileceklerinden dolayı farklı durumlara karşı farklı tepkiler veremezler.

Endüstriyel robotların kullanılması ile beraber bazı avantajlar ve dezavantajlar oluşmaktadır. Avantajları şu şekilde sıralanabilir:

• İnsanlar için tehlikeli ve sağlıksız ortamlarda çalışabilirler.

• İşverenlerin çalışanlar için sağlamış olduğu yan ve sosyal haklar gibi giderlerinin olmaması ile ucuz iş gücü sağlayabilirler.

• İş ve ürün niteliğinde düşüş olmadan uzun süreli çalışma gücü gösterebilirler.

• İnsanlar için ağır ve zorlayıcı işlerde rahatlıkla çalışabilirler.

• İş yeri güvenliği ve insanların iş sağlığı ve güvenliği süreçlerine tabi olmadıkları için uzun süre kesintisiz çalışabilirler ve iş gücü kaybı yaşamazlar.

Avantajlarının yanı sıra dezavantajları da bulunmaktadır. Bunlar şu şekilde sıralanabilirler:

• Ucuz iş gücü sağladıklarından dolayı işsizlik oranını arttırabilirler ve insanların işsiz kalmasına sebep olabilirler.

• Süreklilik halinde kontrol ve bakıma ihtiyaç duyarak zaman kaybı yaratabilirler.

• İşlem sırasında yanlış veri işleme alındıysa yanlış ürün sürekliliğine neden olabilirler.

• İnsanlar arası sosyalleşme olanaklarını düşürme gücü vardır.

• Sadece programlandıkları işi yapabilirler, duruma göre farklı tepkiler veremezler (Azizi, 2020).

Endüstride robotlar avantajları dolayısıyla pek çok alanda çeşitli işlerde kullanılmaktadır. Bunlara kaynak, polisaj, taşıma, paketleme gibi pek çok alan sayılabilir. Bu tez çalışmasında makine besleme alanı üzerine çalışıldığı için makine besleme uygulamalarından bahsetmek gerekirse;

Endüstride kullanılan makinelere iş parçasını yüklemek ve boşaltmak için sıklıkla robotlar kullanılmaktadır. Plastik enjeksiyon, pres besleme gibi hız gerektiren uygulamalar olduğu gibi CNC torna ve freze besleme uygulamalarına da sık rastlanır. Robotlar, çok sık farklı parça modellerine geçilmeyen seri imalatlar için oldukça avantajlıdır.

Uygulamaya ve parça tipine göre farklı yöntemlerle robota işlenecek parçalar ulaştırılır. İşlendikten sonra robot uygulamaya göre farklı yöntemlerde istifleme yapar.

Robotların insanlara kıyasla hızlı olması, mola vermemesi, yorulmaması avantajları olduğu gibi bu makineleri kullanan operatörlerin yaralanmasının da önüne geçmeleri yarar sağlamış olur. Şekil 1. ‘de CNC torna makinasına malzeme yükleyip boşaltan robot örneği görülmektedir. Robotun üzerinde malzemeleri tutmasını sağlayan bir tutucu takım da bulunmaktadır. Bu uygulama da Fanuc marka bir robot kullanılmıştır ve robot burada operatörün yaptığı işi yapmaktadır.

Şekil 1. CNC torna besleme örneği

1.3. Endüstriyel Robotların Sınıflandırılması

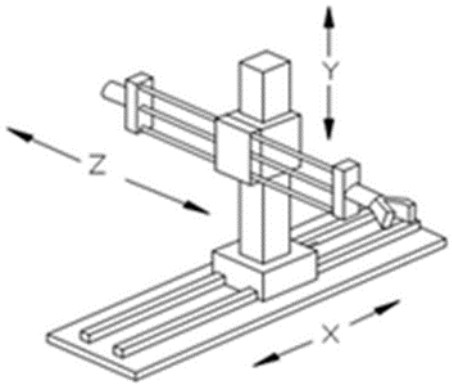
Endüstriyel robotlar sınıflara ayrılmıştır. Bu sınıfların oluşmasında özellikle koordinat sistemlerinin, çalışma uzaylarının yapısı ve mekanik tasarımları etkili olmuştur. Robotların birbirlerine göre hareket kabiliyetlerinde, erişimlerinde ve en üst yük kapasitelerinde farklar oluşur. Endüstride robotlar tercih edilirken bu kıstaslar önem arz eder (Yıldırım, 2019).

1.3.1. Kartezyen Robotlar

Kartezyen robotlar genellikle X,Y,Z eksenlerinde doğrusal hareket eden robotlardır. Robotun motorları doğrusal hareket sistemlerine bağlıdır ve eksenler doğrusal olarak hareket eder. Çalışma uzayları kübiktir. Açılı hareketler yapamazlar. Çalışma uzayları kendilerinden küçük olmak zorundadır. Taşıdıkları tutucu gibi ekipmanları kendi eksenlerindeki kızak vb. yapıların boyu kadar mesafelere, doğrusal olarak götürebilirler. Dolayısıyla çalışma alanları kendi içlerinde kalmaktadır. Büyük çalışma alanların için daha büyük bir taban alanına sahip olmak zorundadır. Yüksek enerji tüketirler. Doğrusal hareketi sağlamak için karmaşık mekanizmalara ihtiyaç

duyar. Ağır yük taşıma işlerinde avantajlıdır. Yüksek performans ve hıza sahiptirler (Alsabbagh, 2019).

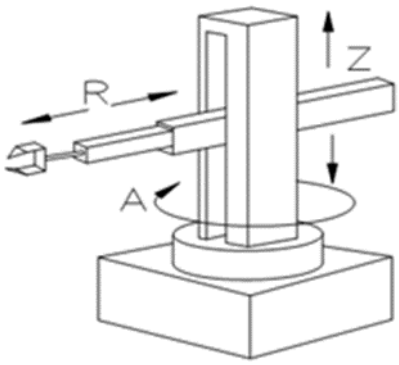
Şekil 2. ‘de kartezyen robotun yapısını ifade eden bir resim görülmektedir. Kartezyen robotlar gerçekte resimdekine benzer doğrusal eksenlere sahiptirler.



Şekil 2. Kartezyen robot yapısı
Kaynak: Thompson, P. (2019).

1.3.2. Silindirik Robotlar

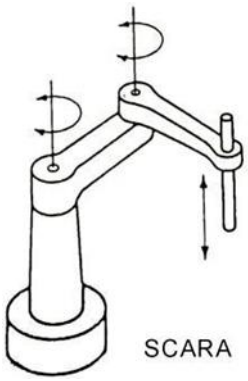
Silindirik robotlar üç eksenden oluşur. Bir eksen dairesel hareketi sağlarken diğeri yukarı aşağı hareketi sağlar. Üçüncü bir eksense yatay olarak ileri geri hareket sağlamaktadır. Robot silindirik bir çalışma alanı içerisinde hareket eder. Silindirik robotun yapısına ait görüntü Şekil 3. ‘te görülebilir. Robotun ulaşabileceği her nokta silindirik koordinatlar ile belirtilebilir. Basit bir yapıya sahiptir. Kartezyen robotlara göre daha düşük hassasiyete sahiptir (Yıldırım, 2019).



Şekil 3. Silindirik robot yapısı
Kaynak: Cylindrical coordinates assignment help. (t.y.).

1.3.3. Scara Robotlar

Scara robotlar 1960 'lı yıllarda Japonya 'da üretilmiştir. Scara ismi Selective Compliance Assembly Robot Arm 'ın kısaltmasıdır. İki dairesel eksen ve bir adet dikey doğrusal eksenden oluşur. Scara robot yapısına ait görsel Şekil 4. 'te görülebilir. Scara robotlar dairesel hareketleri çok hızlı gerçekleştirir ve hassasiyetleri çok yüksektir. Bu sebeple Scara robotlar hız gerektiren montaj işlerinde sıklıkla kullanılır (Yıldırım, 2019).

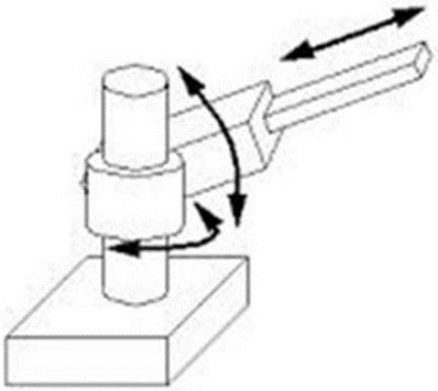


Şekil 4. Scara robot yapısı
Kaynak: Robotic arms. (t.y.).

1.3.4. Küresel Robotlar

Küresel robotlar büyük boyutlu teleskopik yapılı robotlardır. Kol tabanı etrafında dönebildiği gibi yukarı aşağı doğru da bir açısal hareket yapar. Ayrıca ileri geri hareket edebilen bir doğrusal eksen de kolun üzerindedir. Şekil 5. 'te küresel robot

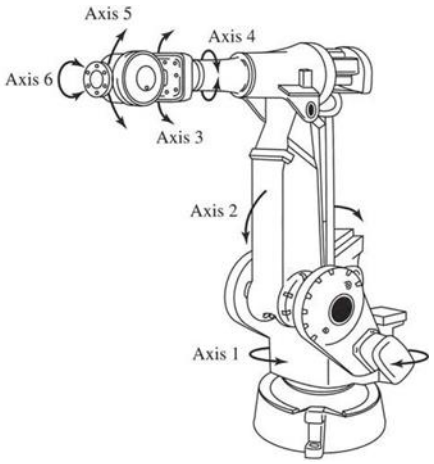
yapısına ait görsel görülebilir. Çalışma alanı küre biçimindedir. Genellikle büyük boyutlardadır. Hassasiyeti düşüktür (Yıldırım, 2019).



Şekil 5. Küresel robot yapısı
Kaynak: Robot types. (t.y.).

1.3.5. Mafsallı Robotlar

Mafsallı robotlar en az 3 eklemden oluşur. Bu eklemler insan koluna benzer bir yapı oluşmasını sağlar. Her bir mafsalın konumu motorlar üzerinden alınan bilgi ile takip edilir. Mafsal konumları ve kol boyu Jacobian matrisi ile işleme sokularak uç takımın bulunacağı koordinat hesaplanır. Mafsallı robot yapısı örneği Şekil 6. ‘da görülebilir. Endüstride kullanılan robotların çoğu mafsallı robotlardır. Hareket kabiliyetleri yüksek ve hızlıdırlar. Hassasiyetleri kartezyen robotlar kadar yüksek değildir. Çalışma alanları geniştir. Çalışma alanlarının genişliği ve insan kolu benzeri yapılarının verdiği hareket kabiliyetleri dolayısı ile kendilerine ve çevreye çarpabilirler (Yıldırım, 2019).



Şekil 6. 6 eksen mafsallı robot yapısı
Kaynak: Types of robots in automation robotics. (t.y.).

Robot sistemleri bazı temel bölümlerden oluşur. Manipülatör, kontrolör ve kontrol paneli temel bölümlerdir. Bunlara ilave olarak takımlar ve harici eksenler gelir (Ersöz, 2007).

Manipülatör nesneleri farklı serbestlik derecelerinde hareket ettirme amacıyla kullanılan, eklemlerden veya kayar eksenlerden oluşan mekanik yapıdır. Kontrol ünitesi tarafından kontrol edilir (ISO, 2012).

Genellikle 6 eksen robot kollar manipülatör olarak adlandırılır. Sistemde fiziksel hareketi gerçekleştiren yapıdır. 6 eksen robot kolların gövdesi genellikle alüminyum dökümden imal edilmektedir. Üzerinde eyleyici olarak servo motorlar bulunur. Kuvvet aktarma amaçlı dişli kutuları, kayış kasnak gibi ekipmanlar kullanılır. Motorların, kuvvet aktarma elemanlarının ve gövdenin boyutları robotun taşıması istenen ağırlığa ve istenen erişim mesafesine göre değişmektedir. Uç noktada flanş adı verilen, ekipmanların üzerine bağlanabilmesi için cıvata delikleri bulunan bir çelik parça vardır. Endüstride sıklıkla kullanılan Fanuc marka bir 6 eksen manipülatör Şekil 7. ‘de mevcuttur.

Şekil 7. FANUC LR Mate 200iD robot manipülatör
Kaynak: Fanuc. (t.y.). Lr mate 200id.

Manipülatörün kontrolünü sağlayan bilgisayar sistemine kontrolör denir. Motor sürücüler, kontrolörün üzerindedir. Giriş çıkış birimleri buraya bağlanır ve kontrolör üzerinden sistemle beraber çalışması sağlanır. Çoğu zaman programlanabilir bir PLC içerir. Acil stop, kapı vs. gibi güvenlik ekipmanlarının bağlanabilmesi için uygun girişlere sahiptir. Örnek bir kontrolör görüntüsü Şekil 8. ‘de mevcuttur.

Şekil 8. FANUC R-30iB Mate robot kontrolör
Kaynak: Fanuc. (t.y.). R-30ib plus.

Robotu hareket ettirmek, programlamak ve gerekli konfigürasyonları yapmak için ihtiyaç duyulan çevre birimine kontrol paneli denir. Üzerinde ekran bulunur. Robotun hareket ettirilmesi ve diğer işlemler için üzerinde tuşlar da bulunmaktadır. Kontrolör üzerinde yapılmak istenen işlemler kontrol paneli vasıtası ile yapılır. Örnek kontrol paneli görüntüsü Şekil 9. ‘da mevcuttur.

Şekil 9. FANUC robot kontrol paneli
Kaynak: Fanuc. (t.y.). FANUC. R-30ib plus controller.

İnsanlar endüstride çalışırken ellerini ve yaptıkları işler için geliştirilmiş özel aletleri kullanırlar. Robotların da programlandıkları işleri gerçekleştirebilmeleri için özel aletlere ihtiyaçları vardır. Robotlar için yapacakları işe göre geliştirilen özel aletlere takım veya uç işlevci denir. Takımlar robotların üzerine montajlanarak robot tarafından taşınır. Bazı durumlarda özel ara aparatlar kullanarak robotun takımları alıp bırakabilmesi sağlanabilir. Endüstride çok farklı türlerde robot takımları kullanılmaktadır. Bunlara taşıma işleri için kullanılan tutucular, kaynak torcları gibi çok sayıda örnek verilebilir. Ayrıca çok özel bir iş için özel takımlar da tasarlanıp kullanılabilir.

Takımlar sinyaller üzerinden veya endüstriyel haberleşme yöntemleri üzerinden robotun kontrolörü ile haberleşebilir. Pnömatik ekipmanlar, elektrik motorları gibi

donanımlar işin gerekliliğine göre takımın üzerinde bulunabilirler. Takımlar işe göre özel olarak tasarlanırlar (Efe, 2018).

Bu tez çalışmasında robot ham ürünleri alarak makineye yükleme yapacaktır. Bu iş için tutucu veya iş hayatında sıklıkla kullanılan İngilizce ismiyle gripper kullanılır. Tutucular mekanik olarak açılıp kapanarak robotların cisimleri tutmasını sağlar. Robotlar tutma işlemi sayesinde cisimleri bir noktadan başka bir noktaya taşıyabilirler. Tutucular pnömatik, hidrolik olabileceği gibi motorlar vasıtası ile de açılıp kapanabilirler. Bazı tutucular cisimleri vakumlayarak tutmaktadırlar. Sıklıkla pnömatik tutucular kullanılmaktadır. Pnömatik tutucu örneği Şekil 10. 'da görülmektedir. Kompresörden gelen hava valfler tarafından yönlendirilerek pnömatik tutucunun içerisindeki özel mekanizmanın hareketi sağlanır. Böylece tutucunun parmakları açılıp kapanmaktadır.

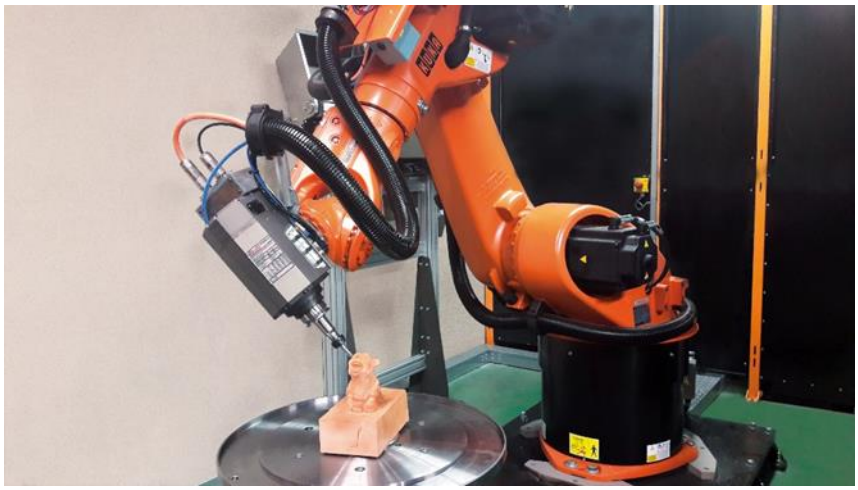
Şekil 10. Pnömatik tutucu örneği
Kaynak: Robotic equipment spotlight. (t.y.).

Endüstride robotlar, kaynak uygulamalarında çok sık kullanılır. Normalde operatörün kendi el becerileri ile gerçekleştirdiği kaynak uygulaması robotlar tarafında çok sağlıklı şekilde gerçekleştirilir. Robot ile kaynak yapılabilmesi için robota takım olarak kaynak torcu bağlanması gerekmektedir. Normalde operatörün elinde tuttuğu torc robot tarafından kullanılmış olur. Şekil 11. 'de kaynak torcu bağlı bir robot örneği mevcuttur. Kaynak teli torcun içerisinden geçer. Tel robot tarafından kontrol edilen bir tel sürme ünitesi vasıtası ile torcun ucuna itilir. Robotlar, kaynak uygulamalarını

gerçekleştirebilmek için kaynak makineleri ile haberleşir. Bu haberleşme sayesinde kaynak makinesi robot tarafından kontrol edilmiş olur.

Şekil 11. Robotik kaynak torcu örneği
Kaynak: Whitter, T. (2017, 2 Ağustos).

Robotlar ahşap, köpük vb. malzemeleri işlemek için de kullanılmaktadır. Bu tipte malzemelerin işlenmesinde robotlar 5 eksen cnc makinelere göre çok daha uygun fiyatlıdır. Ayrıca robotların hareket kabiliyeti 5 eksen cnc makinelere göre avantajlıdır. Bu işlem için robotlara freze motorları monte edilmektedir. Freze motorlu robot uygulaması örneği Şekil 12. ‘de görülmektedir. Freze ucuna takılan kesici takım ile robot malzemenin üzerinden talaş kaldırmaktadır. Ayakkabı vb. kalıpların, model heykellerin işlenmesinde sıklıkla robotlar kullanılır.



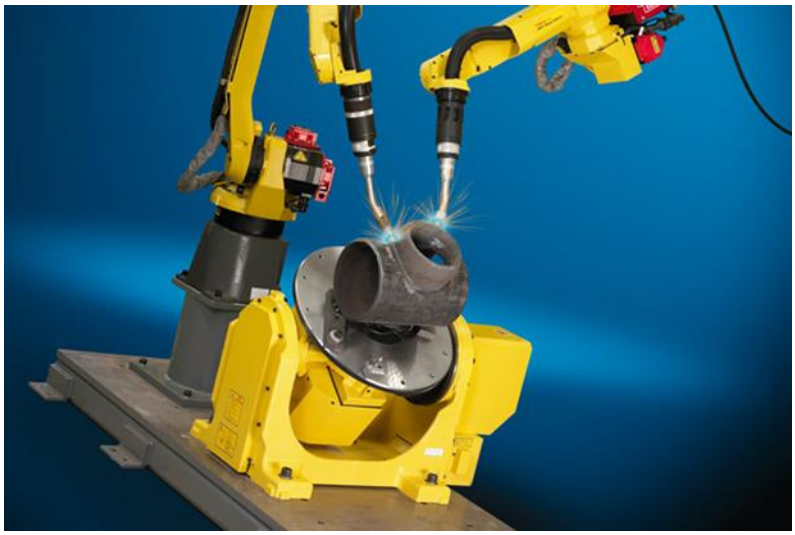
Şekil 12. Robotik freze örneği
Kaynak: Spindles for robotics. (t.y.).

Bazı endüstriyel robot projelerinde 6 eksenli robot kullanılsa bile, robotun hareket kabiliyetinin yetersiz kaldığı durumlar olabilir. Örneği robotun üzerinde işlem yaptığı malzemenin döndürülmesi gerekebilir veya robotun uzak bir mesafeye erişebiliyor olması gerekebilir. Bu gibi durumlara çözüm olarak robot sistemlerine harici eksenler eklenmektedir. Bu harici eksenler üzerinde servo motorlar bulunur. Harici eksenlerin üzerindeki motorlar robot kontrolörüne bağlıdır ve robotla beraber senkron olarak hareket edecek şekilde programlanabilir. Harici eksenler üzerlerinde dişli kutuları bulundurur ve motorlardan aldıkları hareketle doğrusal veya dairesel hareket edebilirler.

Doğrusal hareket sağlayan harici robotun erişim alanını artırmakta çok etkilidir. Kızaklı yapılardan oluşur. Şekil 13. ‘de doğrusal harici eksen örneği bulunmaktadır. Robot kızakların üzerindeki hareketli bölüme monte edilir. Hareketli bölümde bulunan harici eksen motoru, kremayer dişli sistemleri ile doğrusal hareketi sağlar. Doğrusal harici eksenler robot projelerinde sıklıkla kullanılırlar. Örneğin birden fazla makinenin beslendiği büyük sistemlerde robotun bir makineden diğerine geçiş yapabilmesi için kullanılabilirler.

Şekil 13. Doğrusal eksen örneği
Kaynak: Kuka. (t.y.). Kuka linear unit kl 4000.

Robot projelerinde dairesel hareket sağlayan harici eksenler de kullanılabilir. Örnek bir dairesel harici eksen görüntüsü Şekil 14. ‘de görülmektedir. İş parçasının döndürülmesi istenen robot projelerinde bu tip eksenler kullanılır. Örneğin kaynak yapılacak parçayı döndürmek için kullanılabilir. Freze ile heykel vs. gibi parçaların işlendiği durumlarda da sıklıkla kullanılır. Dairesel eksenler kullanıldığı mekanizmaya göre tam tur dönebilir veya belli bir açıda hareket edebilir. Dairesel eksenler bazı özel durumlarda sadece iş parçasını döndürmek için değil, robotun kendisini döndürmek içinde kullanılabilir.



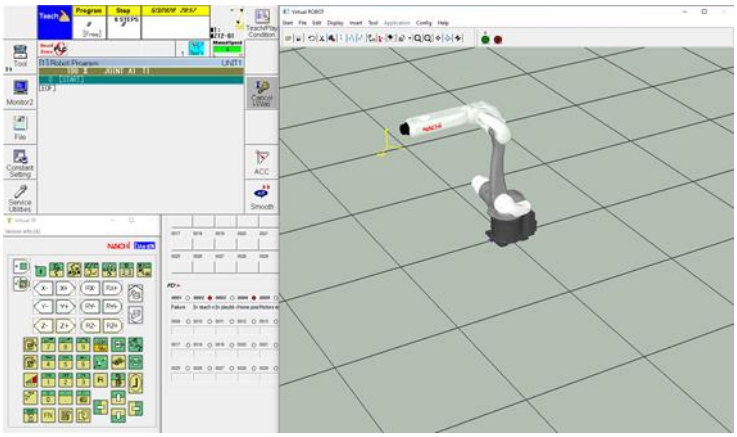
Şekil 14. Dairesel eksenler
Kaynak: Fanuc. (t.y.). Fanuc servo positioners.

1.4. Endüstriyel Robotun Programlanması

Endüstriyel robotlar standartlaşmış makinalardır. Programlanarak, yapacakları işe göre özelleşirler ve sisteme özel bir makine haline gelirler. Duruma göre tekrar ve tekrar programlanabilirler. Endüstriyel robotların programlanmasında iki yöntem kullanılır. Bu yöntemler çevrim içi programlama ve çevrim dışı programlamadır.

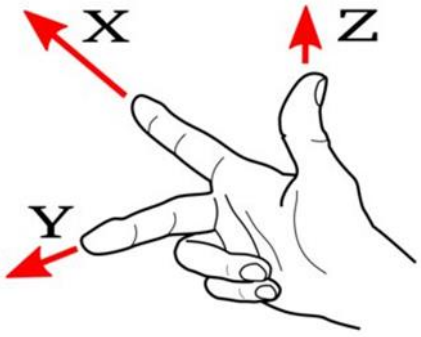
Robotun programlanması işleminin, robotun kontrol paneli (teach pendant) vasıtası ile yapılmasına çevrim içi programlama (online programming) denir. Kontrol paneli ile istenen hareket tipinde robot uzayda gerçekten hareket ettirilir ve istenen noktalar kaydedilir. Hareket komutlarının yanı sıra mantıksal işlemler ve döngüler de aynı programlar içerisine eklenebilirler. Ayrıca sinyallerin kontrolleri vb. komutlar kullanılabilir. Programlar yavaş hızda denendikten sonra otomatik olarak çalıştırılır. En sık başvurulan programlama yöntemidir.

Robotun programlanması işleminin fiziksel olarak robotun yanında bulunmadan bilgisayar programları üzerinden yapılmasına çevrim dışı programlama (offline programming) denir. Gerçek robot üzerinde yapılmak istenenler bu programlar üzerinde oluşturur. Programların içinde bulunan 3 boyutlu simülasyon araçları ile gerçek robot modelinin bir kopyası izlenir. Şekil 15. ‘de Nachi FD Ondesk çevrim içi robot programlama yazılmı görüntüsü mevcuttur. Robotların erişimleri denenerek gerçek robotun konulması gereken yer doğru şekilde tespit edilir. İşin yapılabilirliği kontrol edilir ve çevrim süresi hesaplanır. Fiziksel girişler ve diğer sistemlerle haberleşme gibi bazı özel durumlar dışında hemen hemen her konu bu programlar içerisinde oluşturulup test edilebilir. Oluşturulan bu programlar gerçek robota atılır.



Şekil 15. Nachi FD Ondesk program görüntüsü

Robotun programlanması için referans koordinat sistemlerine ihtiyaç duyulur. Robota uzaydaki konumlar öğretilirken bu koordinat sistemleri referans alınır ve onlara göre mesafeler hesaplanmış olur. Koordinat sistemleri ifade edilirken sağ el kuralı kullanılır. Sağ el kuralına ait görsel Şekil 16. ‘da bulunmaktadır. Pozisyonun koordinatı belirtilirken referans koordinat merkezine göre 3 doğrultuda ve bu doğrultulara göre dönüklük değerlerinde koordinat ifade edilir.



Şekil 16. Sağ el kuralı görseli

Koordinat sistemlerinden biri dünya koordinat sistemidir. Dünya koordinat sistemi, kullanıcı tarafından müdahale edilemeyen robot üzerinde tanımlı olarak gelen bir koordinat sistemidir. Merkez noktasının bulunduğu yer değişkenlik gösterebilir. Bazı robotlarda taban da bulunurken bazılarında 2. eksenin motorunun olduğu

konumdadır. Sağ el kuralı geçerlidir. Robotun baktığı eksen X+ yönünü ifade ederken Z+ yukarı doğrudur.

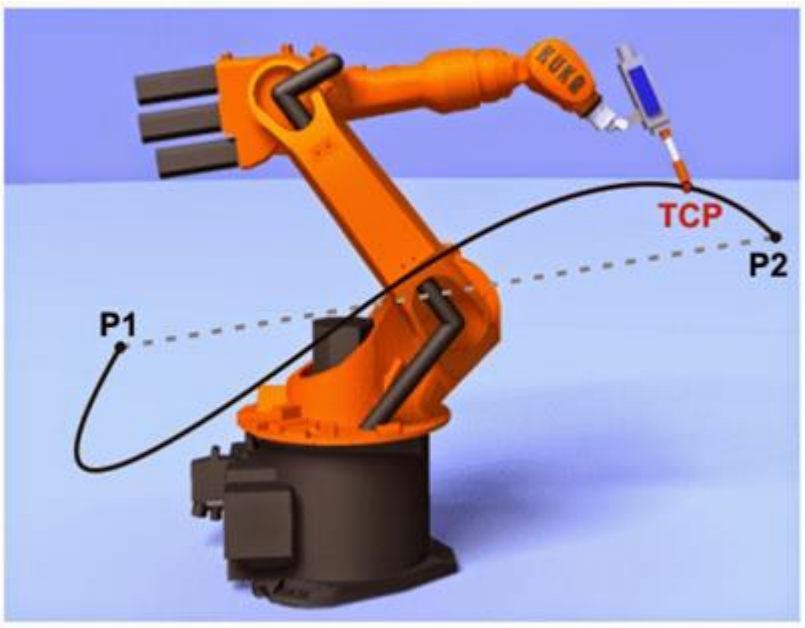
Dünya koordinat sistemine göre pozisyonlar kaydedilebilir veya bu pozisyonlar robota gönderilebilir. Bu pozisyon değerleri takım merkez noktasının dünya koordinat sisteminin merkez noktasına olan uzaklıklarını ve açısal değerlerini saklamaktadır. Robot üzerinde seçili olan kayıtlı takım değerlerine göre değişebilir. Eğer kayıtlı bir takım yoksa robotun flanşının ön yüzeyinin merkez noktası geçerli olmaktadır.

Robota bağlanan takıma göre tanımlanan koordinat sistemine takım koordinat sistemi denir. Takım merkez noktasının konumunu ifade eder. Koordinat sisteminin merkez noktası istenen yere tanımlanabilir. Değerleri flanş merkezine göre hesaplanarak girilir. Takım tanımlanırken değerler tasarıma göre el ile girilebileceği gibi bazı yöntemlerle öğretilerek robota da hesaplattırılabilir. Koordinat sisteminin Z+ yönü flanştan ileriye doğrudur. Sağ el kuralı geçerlidir. Robot programlanırken takım merkez noktasının diğer referans noktalara göre konumları kullanılır.

Robot programlanırken referans olarak kullanmak için uzayda istenilen bir noktaya koordinat sistemi tanımlanabilir. Özellikle açılı yüzeylerde işlem yapılırken avantaj sağlar. Robot programında öğretilen pozisyonlar tanımlanan koordinat sistemi merkezine göre oluşturulabilir. Koordinat değerleri öğretilen merkez noktasını referans alabilir. Bu sayede robot bir noktanın konumunu öğrenirken istenen referans noktasına göre öğrenebilir. Referans noktasının değeri güncellendiğinde bütün program kendini otomatik olarak düzenler.

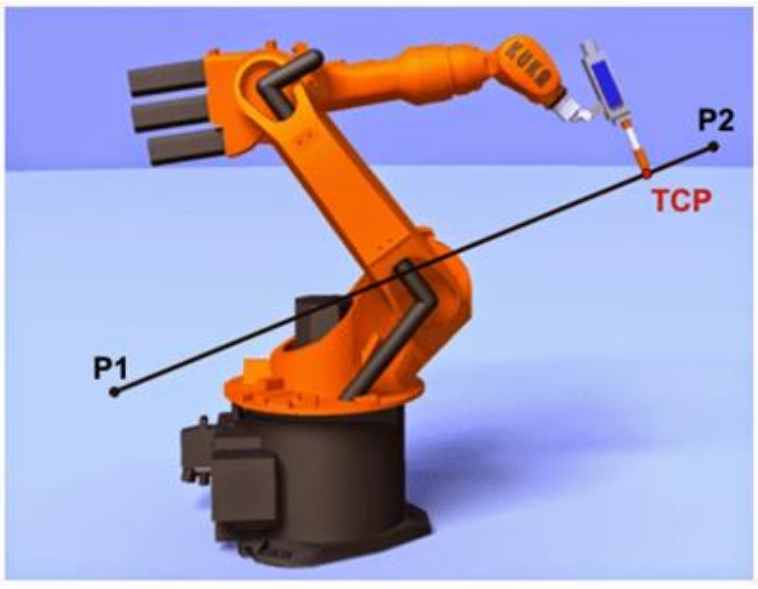
Robot programlarında temel olarak bulunan 3 tip hareket komutu vardır. Bunlar eklem veya noktadan noktaya hareket, doğrusal hareket ve dairesel harekettir.

Noktadan noktaya hareket komutunda robot, öğretilen noktaya en az enerji harcayacağı şekilde gitmektedir. En az enerji harcayacağı yol çoğu zaman doğrusal bir yol olmamakla beraber en kısa yol da olmayabilir. Robot yol boyunca eksenlerini hareket ettireceği açıları kendisi hesaplamaktadır. Dolayısıyla yol boyunca eksenlerin alacakları açıları tahmin etmek oldukça zordur. Noktadan noktaya hareket komutuna örnek Şekil 17’ de bulunaktadır.



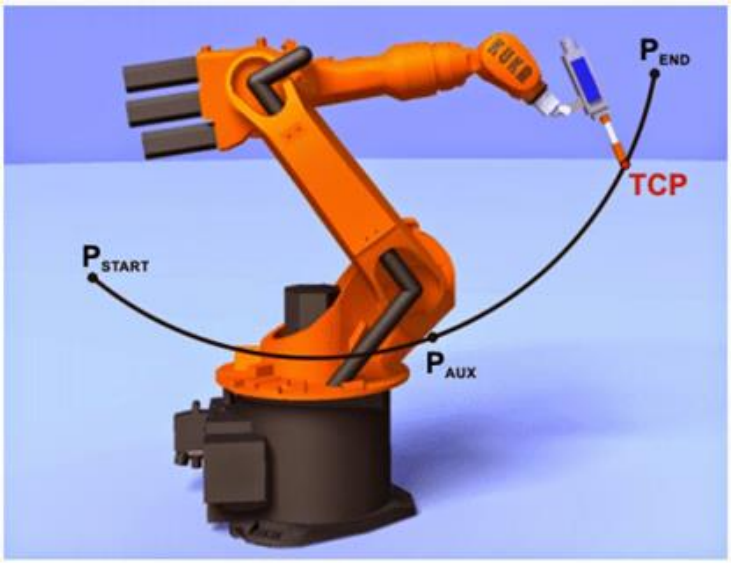
Şekil 17. Robot noktadan noktaya komutu
Kaynak: Meier, M. (2019, 21 Ocak).

Doğrusal hareket komutunda robot, öğretilen noktaya doğru hareket ederken takım merkez noktasını düz bir şekilde hareket ettirir. Yani takım merkez noktası, hedef noktaya giderken uzaydaki bir doğruyu takip eder. Şekil 18. ‘de doğrusal hareket komutu görülmektedir.



Şekil 18. Robot doğrusal hareket komutu
Kaynak: Meier, M. (2019, 21 Ocak).

Dairesel hareket komutunda ise robot hedef noktaya bir yay çizerek hareket eder. Bunun için robota bir hedef nokta ve bir yardımcı nokta öğretilir. Robot hedef noktaya giderken yardımcı noktadan geçen bir yayı otomatik olarak çizer. İki adet dairesel hareket komutu ile bir tam daire çizilebilir. Şekil 19. ‘da dairesel hareket komutu görülmektedir.



Şekil 19. Robot dairesel hareket komutu
Kaynak: Meier, M. (2019, 21 Ocak).

1.5. Robot Kinematiği

Kinematik, hareketin sebepleri ile ilgilenmeden cisimlerin hareketini kuvvet ve hız yönünden inceleyen bir alandır. Robot kinematiği robotun kuvvet ve hızının analizini gerçekleştirir. Uç işlevci ile robotun eklemleri arasındaki ilişkiyi robot kinematiği sağlar. Robot kinematiği ileri kinematik ve ters kinematik olmak üzere iki bölümde ele alınır (Yılmaz, 2010).

1.5.1. İleri Kinematik

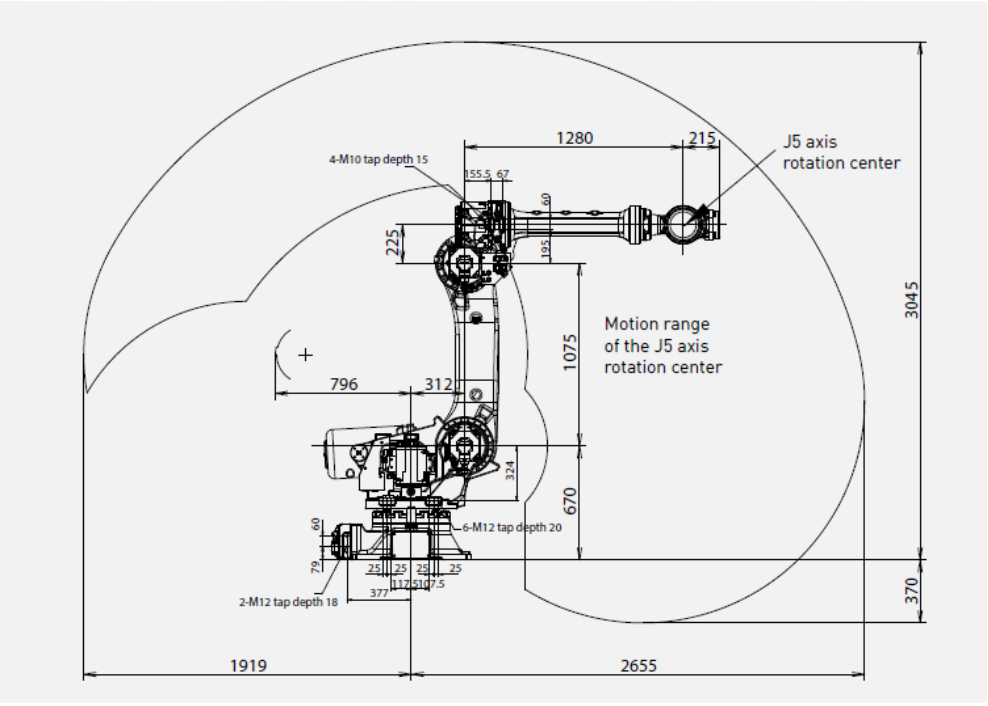
Robotun eklemleri arasındaki ilişki homojen dönüşüm matrisiyle açıklanır. Eklemler için dönüşüm matrislerinin art arda çarpılmasıyla, ana çerçeve ve araç çerçevesi ilişkisi tanımlanır. Böylece eklemlerin hareketine göre robotun uç işlevcisinin hareketi tespit edilir (Küçük, 2004).

İleri kinematik hesabında genellikle Denavit-Hartenberg değişkenlerini kullanan dönüşüm matrisi kullanılır. Denavit-Hartenberg parametreleri eksenler arasındaki kol boyu, eksenler birbirlerine olan açı farkı, birbiriyle çakışan eksenler arası mesafe ve eksen açılarıdır. Örnek olarak tez içerisinde kullanılacak olan Fanuc R2000ic 165f robotuna ait Denavit-Hartenberg parametreleri Tablo 1. ‘de gözükmektedir. Hesaplamalar için gerekli olacak olan robotun flanş kalınlığı da 215 minimetredir.

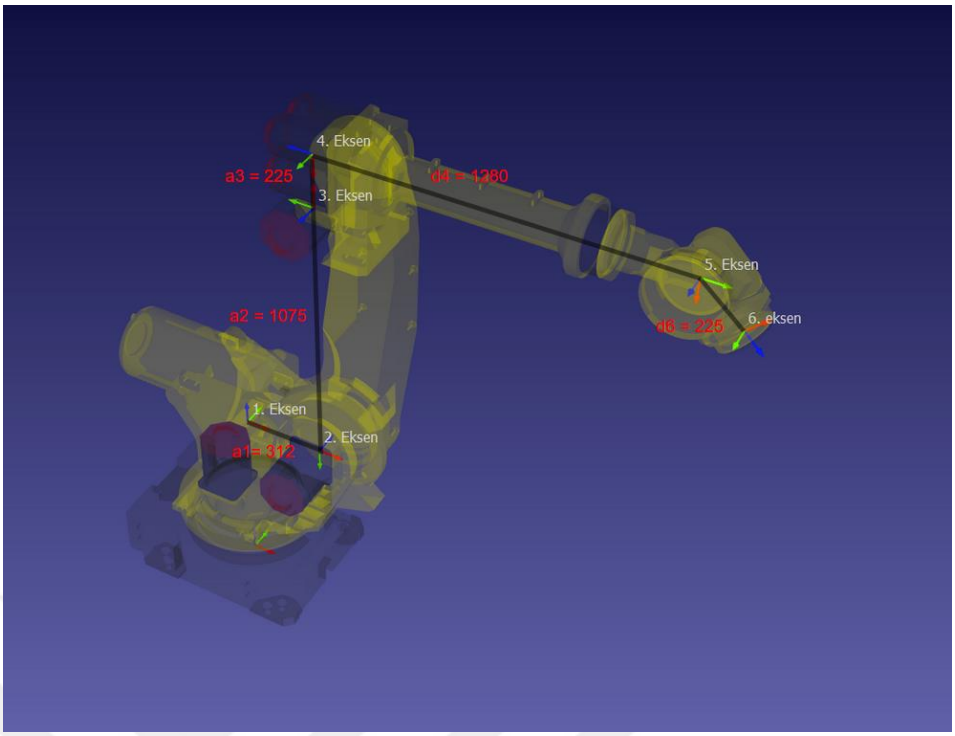
Bu bilgiler robota ait katalog üzerinden elde edilebilir. R2000ic 165f için katalog değerleri Şekil 20 ‘de mevcuttur. Katalog değerleri üzerinden eksenler arası mesafeler olan a ve d değerleri elde edilir. Robota ait mesafe değerleri Şekil 21 ‘de gösterilmiştir.

Tablodaki α değerleri eksenler üzerindeki koordinat merkezinin dönüklük değerleridir. Buna göre Şekil 21 ‘de her eksen için koordinat sistemleri yerleştirilmiştir. Eksenlerin Z ekseni etrafındaki açı değeri θ ile ifade edilir.

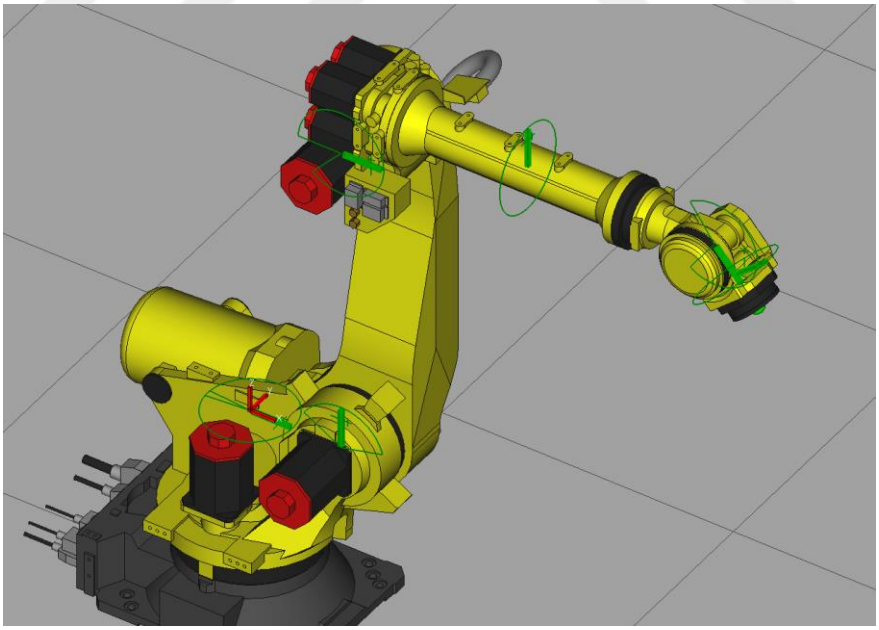
Robotun θ açıları Şekil 22 ‘de yeşil halkalar biçiminde sembolize edilmiştir. Görüntüde 1, 2, 3, 4 ve 6 numaralı eksenler 0 derecedeyken 5 numaralı eksen -45 derecede bulunmaktadır.



Şekil 20. Fanuc R2000ic 165f katalog değerleri
Kaynak: Fanuc (2019). Datasheet R2000iC 165F



Şekil 21. Fanuc R2000ic 165f kol uzunluk değerleri



Şekil 22. Fanuc robot eksenlerin açı gösterimi

Tablo 1. Fanuc R2000ic 165f robotu Denavit-Hartenberg parametreleri

	θ	α	a	d
1	θ_1	$\frac{\pi}{2}$	312	0
2	$\frac{\pi}{2} - \theta_2$	0	1075	0
3	$\theta_3 - \left(\frac{\pi}{2} - \theta_2\right) + \frac{\pi}{2}$	$\frac{\pi}{2}$	225	0
4	$-\theta_4$	$-\frac{\pi}{2}$	0	1280
5	θ_5	$\frac{\pi}{2}$	0	0
6	$-\theta_6$	0	0	0

Denavit-Hartenberg parametrelerine robot ekranından Şekil 23 ‘de olduğu gibi erişilebilir.

Şekil 23. Fanuc robot ekranında Denavit-Hartenberg parametreleri

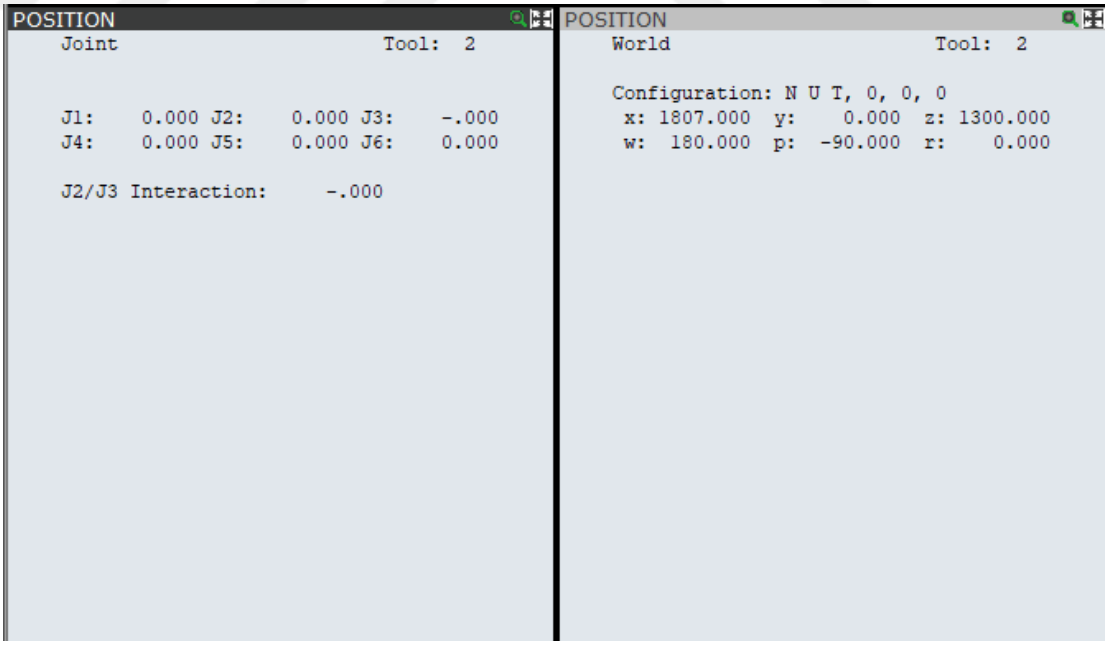
Parametreler her bir eksen için 1 numaralı denklemdeki matrise eklenmelidir ve 6 eksen için 6 matriste oluşturulmalıdır. Ayrıca robot uç işlevcisi de hesaba katılmalıdır. Uç işlevci olarak yalnızca robotun üstündeki bağlantı flanşı bulunduğu için 2 numaralı işlemdeki matrise flanş kalınlığı değeri yazılır. 3 numaralı işlem gerçekleştirildiğinde dönüşüm matrisi hesaplanmış olur.

$$T_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \,.\, cos\,\alpha_i & sin\,\theta_i \,.\, sin\,\alpha_i & a\,.cos\,\theta_i \\ \sin\theta_i & \cos\theta_i \,.\, \cos\alpha_i & -\cos\theta_i \,.\, \sin\alpha_i & a\,.\ \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

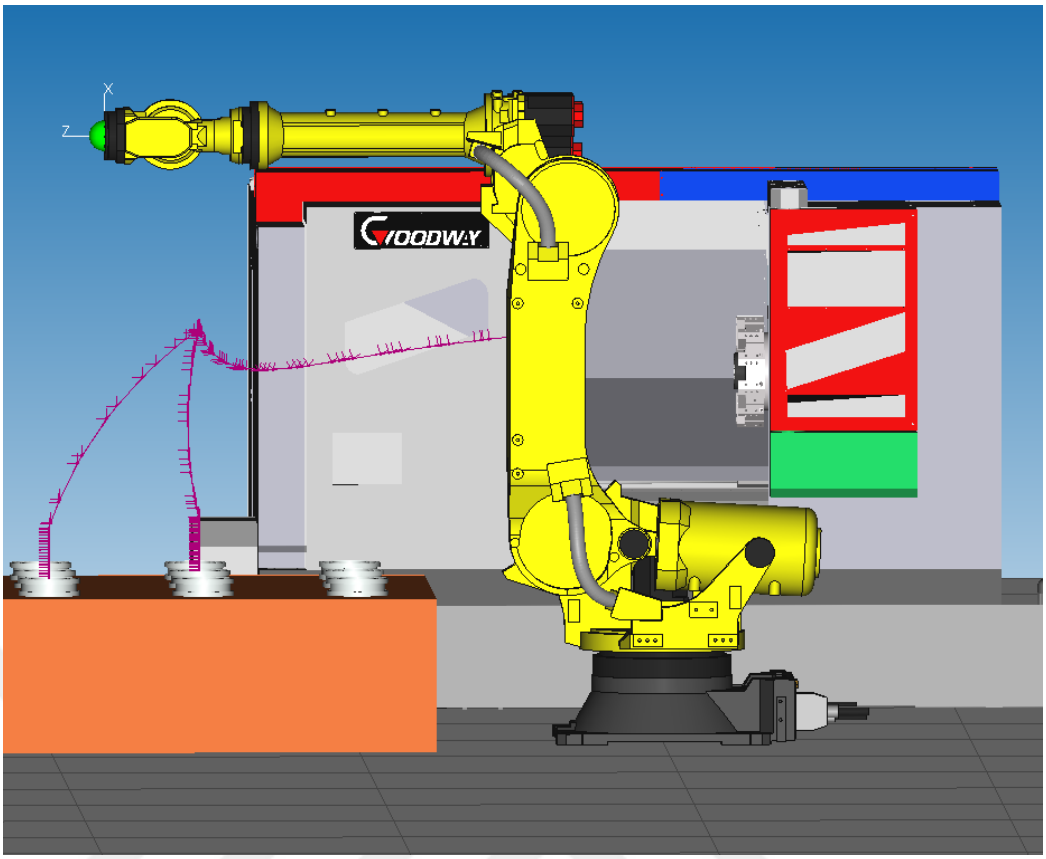
$$E = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & Flanş\ kalınlığı \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

$$T = (T_1.T_2.T_3.T_4.T_5.T_6)E \quad (3)$$

Örnek değerler Şekil 24 ‘de görülmektedir. Şeklin sol tarafında görüldüğü gibi J1 ‘den J6 ya eksenleri açı değerleri 0 iken, sağ tarafta kartezyen koordinat değerlerinde X 1807.0, Y 0.0 ve Z 1300.0 gözükmektedir. Bu esnada robotu bulunduğu fiziksel pozisyon Şekil 25 ‘de görülebilir.



Şekil 24. Örnek ileri kinematik dönüşüm değerleri



Şekil 25. Örnek ileri kinematik dönüşümü robot pozisyonu

$$T_1 = \begin{bmatrix} \cos\theta_1 & 0 & sin\,\theta_1 & 312\,.\,cos\,\theta_1 \\ \sin\theta_1 & 0 & -\cos\theta_1 & 312\,.\,\sin\theta_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

$$T_2 = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & 1075\,.\,cos\,\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & 1075\,.\,\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5)$$

$$T_3 = \begin{bmatrix} \cos\theta_3 & 0 & sin\,\theta_3 & 225\,.\,cos\,\theta_3 \\ \sin\theta_3 & 0 & -cos\,\theta_2 & 225\,.\,\sin\theta_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6)$$

$$T_4 = \begin{bmatrix} \cos\theta_4 & 0 & -sin\,\theta_4 & 0 \\ \sin\theta_4 & 0 & cos\,\theta_4 & 0 \\ 0 & -1 & 0 & 1280 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (7)$$

$$T_5 = \begin{bmatrix} \cos\theta_5 & 0 & sin\,\theta_5 & 0 \\ \sin\theta_5 & 0 & -cos\,\theta_5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (8)$$

$$T_6 = \begin{bmatrix} \cos\theta_6 & -sin\,\theta_6 & 0 & 0 \\ \sin\theta_6 & cos\,\theta_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (9)$$

$$E = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 215 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (10)$$

Fanuc R2000ic 165f robotu için eksenlere ait matrisler 4, 5, 6, 7, 8 ve 9 numaralı işlemlerde görülmektedir. Uç işlevciye ait matris 10 numaralı işlemde görülebilir. Eksenlerin o anki açı değerleri θ değerlerine yazıldığında, matrislerin çarpımından dönüşüm matrisi elde edilir ve robotun konumu tespit edilebilir.

Şekil 24 ve Şekil 25 ‘de görülen tüm eksen açı değerlerinin sıfır olduğu pozisyon için açı değerleri matrislere 11, 12, 13, 14, 15 ve 16 numaralı işlemlerde yazılmıştır. Bu matrislere göre dönüşüm matrisi 18 numaralı işlemde elde edilmiştir.

$$T_1 = \begin{bmatrix} 1 & 0 & 0 & 312 \\ 0 & 6{,}12323E-17 & -1 & 0 \\ 0 & 1 & 6{,}12323E-17 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (11)$$

$$T_2 = \begin{bmatrix} 6{,}12323E-17 & -1 & 0 & 6{,}58248E-14 \\ 1 & 6{,}12323E-17 & 0 & 1075 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (12)$$

$$T_3 = \begin{bmatrix} 1 & 0 & 0 & 225 \\ 0 & 6{,}12323E-17 & -1 & 0 \\ 0 & 1 & 6{,}12323E-17 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (13)$$

$$T_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 6{,}12323E-17 & 1 & 0 \\ 0 & -1 & 6{,}12323E-17 & 1280 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (14)$$

$$T_5 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 6{,}12323E-17 & -1 & 0 \\ 0 & 1 & 6{,}12323E-17 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (15)$$

$$T_6 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (16)$$

$$E = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 215 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (17)$$

$$T = \begin{bmatrix} 6{,}12323E-17 & -6{,}12323E-17 & 1 & 1807 \\ 6{,}12323E-17 & -1 & -6{,}12323E-17 & -1{,}19403E-14 \\ 1 & 6{,}12323E-17 & -6{,}12323E-17 & 1300 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (18)$$

1.5.2. Ters Kinematik

Ters kinematik araç çerçevesinin ana çerçeveye olan yönelimi ve konumu verildiğinde, robotun bu yönelim ve konuma ulaşabilmesi için gerekli olan açı setlerinin hesaplanmasıdır (Yılmaz, 2010).

Uç işlevcinin konumundan eklemlerin bulunması gereken açı değerleri hesaplanır. Ters kinematik hesaplanırken karşılaşılan denklemler birden fazla çözüm üretebilir. Dolayısıyla aynı noktanın birden fazla açıda yaklaşımı bulunabilir.

İKİNCİ BÖLÜM

ENDÜSTRİYEL HABERLEŞME PROTOKOLLERİ

Endüstriyel uygulamalarda makinelerin birbirleriyle ve hatta insanlarla iletişim halinde olması gerekmektedir. Ancak bu şekilde üretim gerçekleşebilir. Örneğin robotun kaynak makinası ile sürekli bağlantıda olması gerektiği gibi, bir makine besleme uygulaması yapıyorsa yükleme yapacağı makine ile de haberleşiyor olması gereklidir. Operatörlerinde sistemi kontrol edebilmesi veya gerekli değer girişlerini yapabilmesi sistemle haberleşen bir panel olması gereklidir. Bazı durumlarda bütün bunların uzaktan izlenebilmesi içinde sistemler kurulmaktadır. Bu aradaki veri alışverişi endüstriyel haberleşme protokolleri üzerinden geçmektedir (Asubay, 2018).

Pek çok sayıda haberleşme protokolü vardır. Bazıları standart protokoller olduğu gibi bazıları markaların kendilerine ait geliştirdikleri protokollerdir. Standart soket yapılarına sahip olabildikleri gibi kendilerine has tipte bağlantılarda olabilir. Firmalar farklı protokolleri kendi aralarında haberleştirmek içinde çözümler sağlamaktadır.

Robot tarafında da uygulamaya göre protokoller seçilerek kullanılır.

2.1. Ethernet IP

Ethernet IP, Rockwell Automation tarafından 2001 yılında geliştirilen bir haberleşme protokolüdür. ODVA (Open DeviceNet Vendor Association) tarafından desteklenmektedir. Ethernetin TCP/IP alt yapısı kullanılarak endüstriyelleştirilmiş bir versiyonudur (Alessandria, Senio ve Vitturi, 2007).

Standart Ethernet TCP/IP yapısını CIP (Common Industrial Protocol) ile entegre ederek gerçek zamanlı endüstriyel uygulamalara imkan tanır. Bu uygulamaların internet ağına açılması da imkan dahilindedir (Liu, Tang. ve Jiang, 2009).

CIP gerçek zamanlı veri aktarımı için geliştirilmiş bir protokoldür. DeviceNet, ControlNet gibi bazı başka haberleşme protokolleri üzerinde de çalışabilir. Giriş çıkış, teşhis ve konfigürasyon bilgilerinin aktarılmasını sağlar. CIP bağlantısında belirlenmiş giriş çıkışlar özel olarak kontrol edilebildiği gibi, bu bağlantı açık bir mesajlaşma için de kullanılabilir (Alessandria vd., 2007).

2.2. DeviceNet (CAN Bus)

CAN bir seri iletişim protokolüdür. Özelikle otomotiv sektöründeki uygulamalar için geliştirilmiştir ancak iyi bir performans sunması dolayısıyla diğer endüstriyel uygulamalar için de kullanılır. Kısa mesajlar için optimize edilmiş bir protokoldür. Mesaj odaklı bir protokoldür ve her bir mesajın kendine özel bir önceliği vardır. Mesaj parçalara ayrılır ve her bir parça veri yolu boş olana kadar bekler. Sırası geldikçe parça parça veriler iletilir.

CAN tabanlı bir ağda veriler bütün alıcılara gönderilir. Hedef veya kaynak veri içerisinde belirtilmez. Her mesaja özel bir etiketleme yapılır. Alıcılar bu etikete göre mesajları filtreleyerek kullanır. Ağdaki tüm alıcı cihazlar bu mantıkta mesajı kabul eder veya reddeder. Buna çoklu yayın (multicast) denir.

DeviceNet haberleşme için nispeten düşük maliyetli bir çözümdür. Cihaz düzeyindeki uygulamalarda önemli ölçüde kabul görmüştür (Lian, Moyne ve Tilbury, 2001).

2.3. Modbus

MODICON firması tarafından ortaya çıkan Modbus protokolü en çok kullanılan protokollerden biridir. Kullanımı kolaydır. efendi – köle (master - slave) mantığına sahiptir. Bilgisayarlar ile haberleşmede ve verileri depolamada büyük kolaylık sağlar. Kullanıma açık ve lisans gerektirmeyen bir protokoldür.

1979 yılında ortaya çıkmıştır. En yaygın ve en eski protokollerden biridir. Efendi ve köle cihazlar arasında basit bir istek ve yanıt mekanizması ile çalışır. Efendi cihaz yaz veya oku mesajları göndererek kontrolleri sağlayabilir. Belli bir köle yani istemci adresine yapılan yayın bir istek ve geri dönen bir yanıt içerir. Modbus haberleşme Modbus RTU ve Modbus TCP/IP olmak üzere iki şekilde gerçekleştirilir. Modbus RTU seri hattı kullanırken Modbus TCP/IP ethernet hattı üzerinden çalışır (Huitsing, Chandia, Papa ve Shenoi, 2008).

Modbus verileri adres, fonksiyon kodu, veri ve hata kontrolü bölümlerinden oluşur. Sorgulama, cevap ve hata olmak üzere 3 tip veri vardır. Veriler cihazlarda yazmaç adı verilen bölümlerde tutulur. Yazmaçlar 1 bitlik veriler saklayabileceği gibi 2 bayt veri de saklayabilir. Her yazmacın bir adres bilgisi vardır. Yazmaçların türlerine

göre farklı fonksiyonlar ile sorgulamalar yapılır. Hata durumlarında hata kodları ile yanıt alınır (Bakır, 2019).

Data Type	Absolute Addresses	Relative Addresses	Description
Coils	00001–09999	0–9998	Read coil status
Coils	00001–09999	0–9998	Force single coil
Coils	00001–09999	0–9998	Force multiple coils
Discrete inputs	10001–19999	0–9998	Read input status
Input registers	30001–39999	0–9998	Read input registers
Holding registers	40001–49999	0–9998	Read holding register
Holding registers	40001–49999	0–9998	Preset single register
Holding registers	40001–49999	0–9998	Preset multiple registers
Holding registers	40001–49999	0–9998	Read exception status
Holding registers	40001–49999	0–9998	Loopback diagnostic test

Şekil 26. Modbus data tipleri
Kaynak: Shahzad, A., Lee, M., Lee, Y. K., Kim, S. (2015).

Function Name	Function Code	Description
Read coil or digital output status	01	The field device responds to the logical coil(s) ON/OFF status.
Read digital input status	02	Read discrete inputs from the field device.
Read holding registers	03	Retrieves the contents of the holding register(s) from field device.
Reading input registers	04	Retrieves the contents of input register(s) from the field device.
Force single coil	05	The ON/OFF status of single logic coil is changed from the field device.
Preset single register	06	To change the content of a single holding register.
Read exception status	07	To retrieve the status of eight digital points as a short message request from the field device.
Loopback test	08	Employs diagnostic features including CRC errors and reports according to exceptions to test the operation of the system.
Force multiple coils or digital outputs	0F	To manage the ON/OFF status of the coils (or group of coils).
Force multiple registers	10	To change the content of a single register and to manage a group of coils

Şekil 27. Modbus fonksiyon kodları
Kaynak: Shahzad, A., Lee, M., Lee, Y. K., Kim, S. (2015).

Modbus yazmaçları Şekil 26 ‘da görüldüğü gibi açık kapalı değerleri alan bobinlerden ve sayısal değer alan yazmaçlardan oluşur. Şekil 27 ‘de bulunan fonksiyon kodları ile okuma ve yazma gibi işlemler gerçekleştirilebilir.

2.3.1. Modbus RTU

Modbus protokolünün seri haberleşme üzerinden kullanılan bir versiyonudur. RS-232 veya RS-485 portları kullanılır. Haberleşme efendi cihaz tarafından başlatılmaktadır. Köle cihazlar sürekli olarak gelen mesajlarda kendi adresini kontrol eder. Kendisine gelen bir mesaj olursa işlemlere başlar. Gönderilen mesaj 4 bölümden oluşmaktadır. Bunlar mesajın gönderildiği köle cihazın adresi, yapılacak işlemin türü, mesajın içerdiği veri ve verilerden elde edilen özel CRC kodudur. CRC kodu verinin doru şekilde ulaşıp ulaşmadığının kontrolü amacıyla üretilir. Köle cihaz da efendi cihaza aynı formatta veri göndererek cevap verir (Demir, 2012).

Modbus RTU mesajlarının paket içeri Şekil 28 ‘de görülebilir.

Slave Address	Function Code	Data	CRC
1 byte	1 byte	0 up to 252 byte(s)	2 bytes CRC Low | CRC Hi

Şekil 28. Modbus RTU veri paketi içeriği
Kaynak: Weis, O. (2020, 23 Mart).

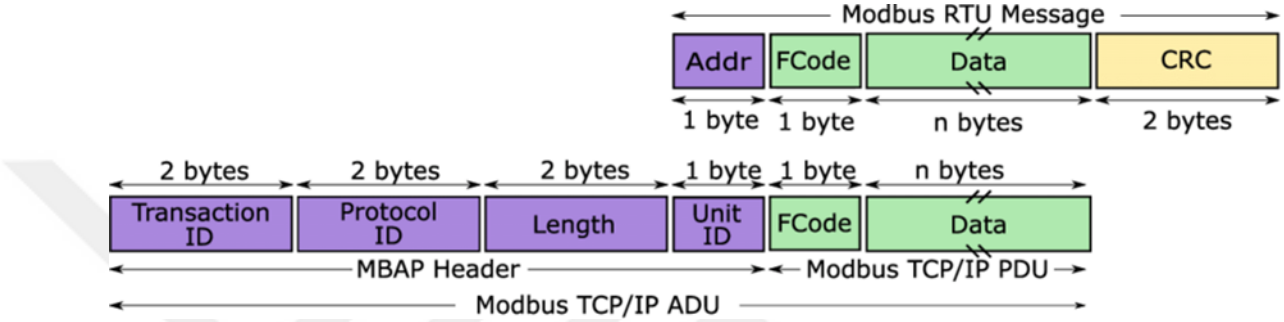
Veriler ASCII olarak gönderilir. İstenen yazmaç adresine fonksiyon kodunun ait olduğu işlem uygulanır. Hata kontrolü amacıyla 2 bayt kontrol verisi en sona eklenir. Bu veri CRC (Cyclic redundancy check) yani döngüsel artıklık kontrolünün sonucudur (Süzer, 2006).

2.3.2. ModbusTCP

Modbus TCP, modbus protokolü üzerindeki haberleşmenin ethernet vasıtası ile gerçekleştirilebilen bir çeşididir. Haberleşme istemci cihazın sunucu cihaza bağlantı kurması ile başlar. Bu işlemler için 502 numaralı port kullanılır. Hem talepler hem de cevaplar adres bilgisi içermektedir. Veri MBAP adında bir başlık birimiyle gönderilir. Bu birim içerisinde işlem tanımlayıcı, protokol tanımlayıcı, uzunluk ve birim tanımlayıcı bulunur. Gönderilen veri içerisinde TCP Checksum bölümünde hata kontrolleri yapılır (Akkaya, 2015).

Gönderilen mesajda bulunan ilk alan adres bilgisini içermektedir. Hedef cihazın IP adresi bulunur. Efendi cihaza 248 köle cihaz bağlanabilir. İkinci alan uygulanan fonksiyona ait bilgidir. Üçüncü alan veriye aittir. Son iki bayt hata denetimi için ayrılmıştır (Irmak ve Erkek, 2018).

Modbus TCP ve Modbus RTU kıyaslamasına ait görsel Şekil 29 ‘da bulunmaktadır.



Şekil 29. Modbus TCP ve Modbus RTU kıyaslaması

Kaynak: Gamess, E., Smith, B., Iii, G. A. F. (2020).

ÜÇÜNCÜ BÖLÜM

BİLGİSAYAR PROGRAMI ÜZERİNDEN ROBOT KONTROLÜ

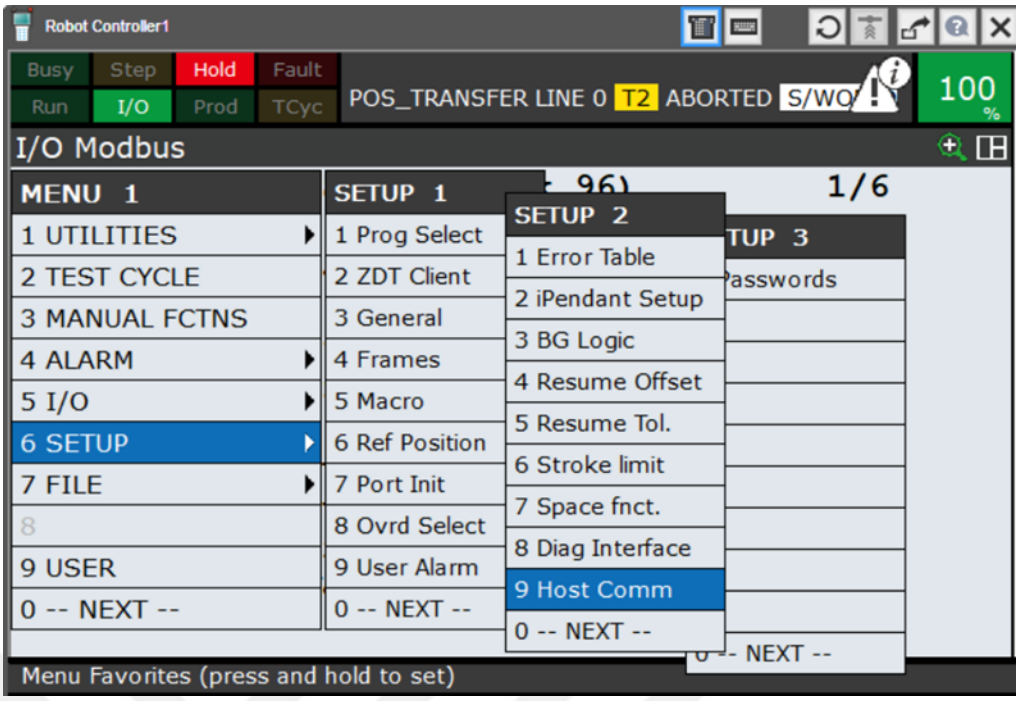
Endüstriyel robotlar makine besleme uygulamalarında genellikle sabit noktalardan ürün alarak çalışırlar. Örneğin üzerinde operatörün hep aynı şekilde malzemeleri dizebileceği gözler bulunan tepsiler kullanılmaktadır. Bu tepsilere dizilen ürünler robot tarafından alınır ve makineye yüklenir. Alınacak ürünlerin pozisyon ve açı bilgileri pozisyon değişkenleri içerisinde tutularak robot tarafından kullanılır. Ürünlerin bulunduğu pozisyonlar robota kontrolcü üzerinden robot kontrol edilerek öğretilir. Her ürün ve tepsi için bu öğretme işlemi ayrı ayrı yapılmaktadır.

Bu bölümde robotun bir bilgisayar yazılımıyla haberleşerek pozisyon değişkenlerinin güncellenebileceği bir haberleşme alt yapısı kurulacak ve bu değişken pozisyonlara göre ürünleri alıp yükleyen robot programları oluşturulacaktır. Böylece robot kontrolcüsü kullanılmadan yeni ürün öğretme işlemi gerçekleştirilmek istenmektedir. Ayrıca farklı bir robot programı üzerinden kontrolcü kullanılmadan robot fiziksel olarak hareket ettirilmek istenmektedir.

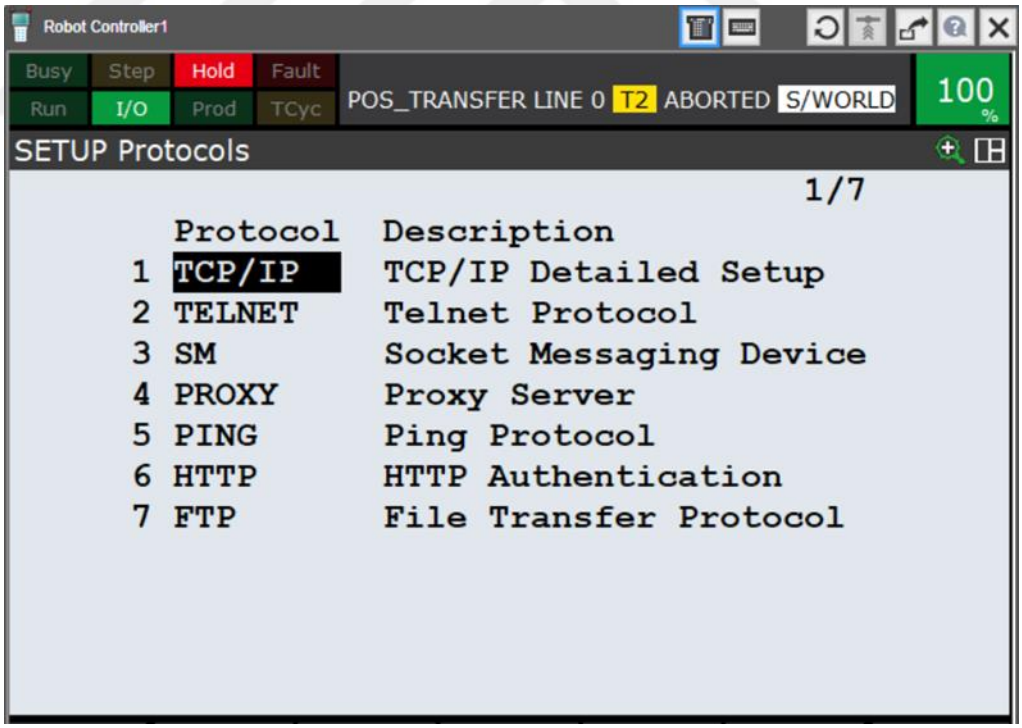
3.1. Robot Tarafında Haberleşme ve Sinyallerin Hazırlanması

Harici bir noktadan robotun kontrol edebilmesi ve değer alabilmesi için uygun bir haberleşme yapısı kurulması gerekmektedir. Modbus TCP haberleşme protokolünün düşük maliyetli olması ve robot simülatörü ile bilgisayar yazılımı arasında haberleşmenin daha sağlıklı çalışacak olması sebebiyle Modbus TCP haberleşme protokolü tercih edilmiştir.

Fanuc robot ekranında haberleşme ayarlanmıştır. Haberleşme ayarlarına giriş sırasıyla Şekil 30 ve Şekil 31 ‘de gösterilmiştir.

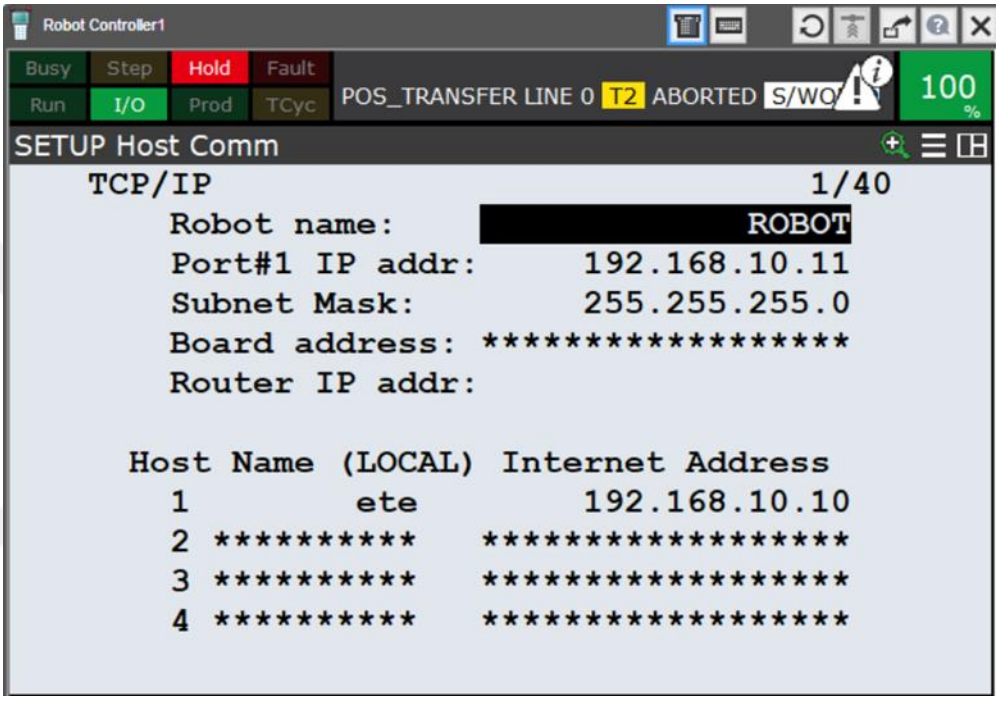


Şekil 30. Fanuc ethernet ayarlarına giriş



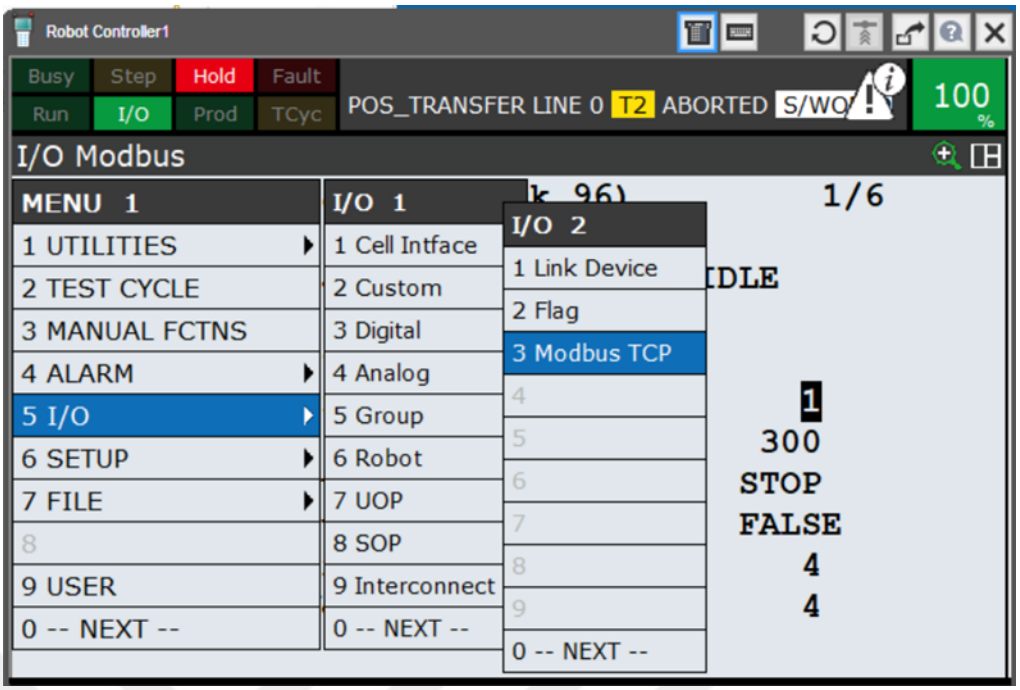
Şekil 31. Fanuc TCP/IP ayarlarına giriş

Haberleşmede 1 numaralı ethernet portu kullanılmıştır. Ethernet ile alakalı olan ayarlamalar Şekil 32 ‘de gözükmektedir. Burada robot için IP adresi değeri olarak 192.168.10.11 girilmiştir. Robota bağlanacak olacak yazılımın içinde bu sayfada IP adresi girilmesi gerekmektedir. Yazılım için 192.168.10.10 IP adresi tanımlanmıştır. Robot simülasyon içerisinde çalıştığı için bilgisayar yazılımı tarafında robot IP adresi olarak 127.0.0.1 kullanılmıştır.

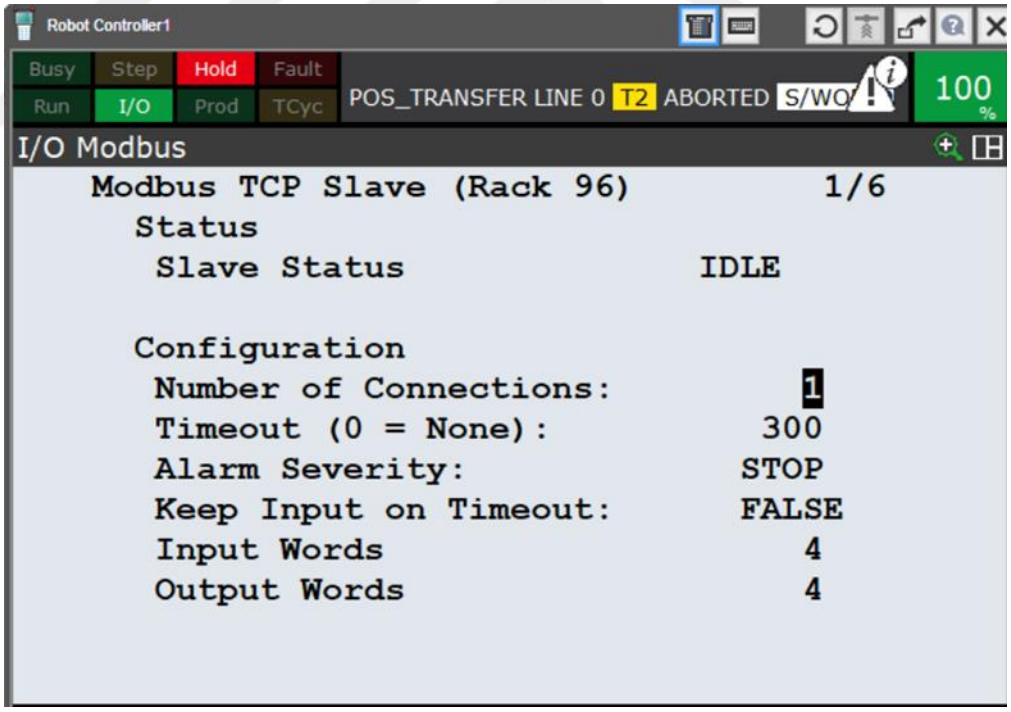


Şekil 32. Fanuc TCP/IP ayarları örneği

Robotun uzaktan kontrolü için sinyaller kullanılması gerekmektedir. Bu sinyaller Modbus TCP için tanımlanmıştır. Bu tanımlamaların yapıldığı ayar sayfasına giriş Şekil 33 ‘de gösterilmiştir. Sinyal olarak 1 “Word” değeri 16 adet sinyale tekabül etmektedir. Haberleşmede 4 * 16 yani 64 giriş, 64 çıkış sinyali tanımlanmıştır. Ayrıca haberleşme herhangi bir bağlantı problemi durumunda 300 saniye sonra hata verecek şekilde ayarlanmıştır. Bu ayarlara ilişkin görüntü Şekil 34 ‘de gözükmektedir.



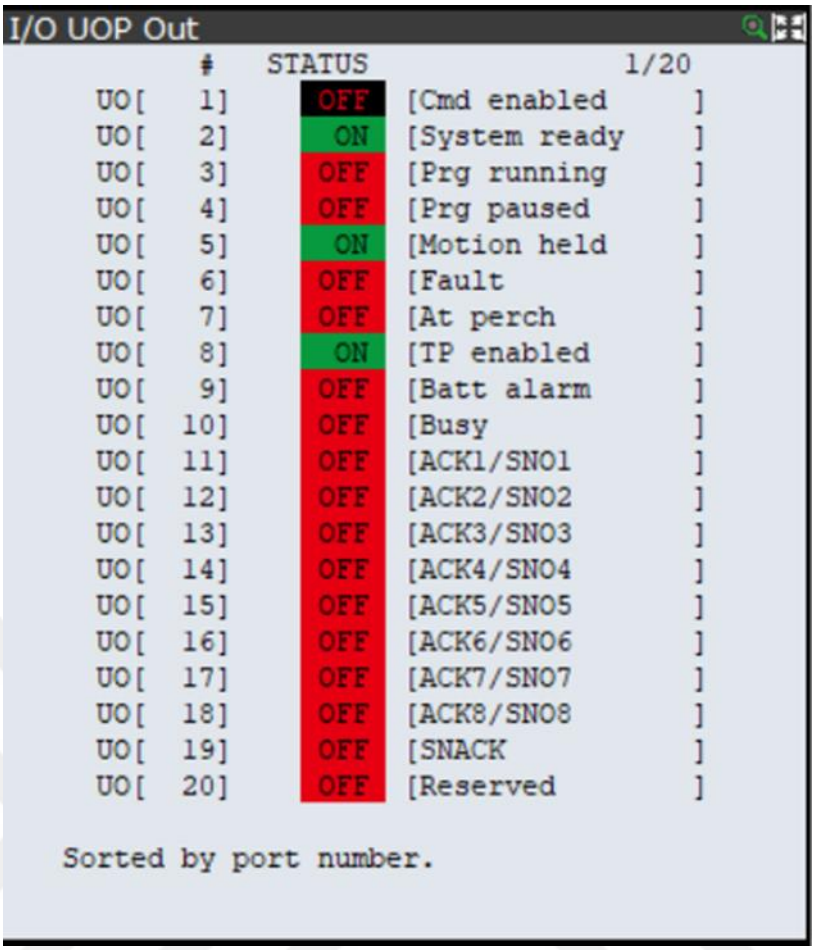
Şekil 33. Fanuc Modbus ayarlarına giriş



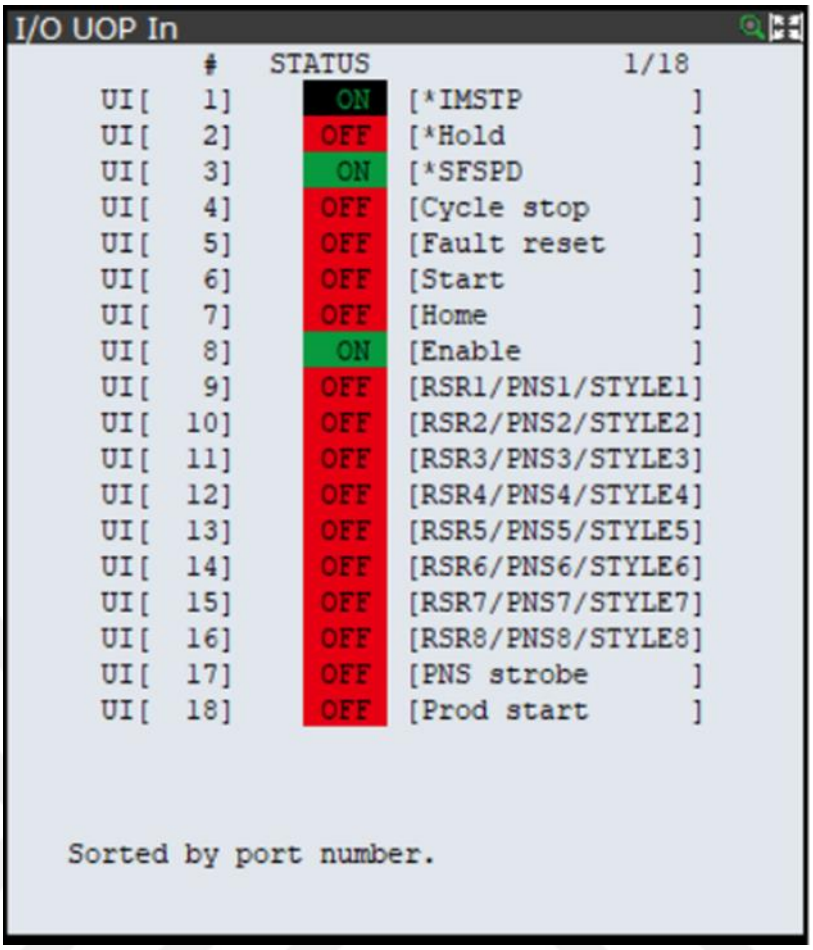
Şekil 34. Fanuc Modbus ayarları örneği

Endüstriyel robotlarda dışarıdan çalıştırma, programlar üzerinde kontrol sağlama veya durumlarla alakalı geri dönüşler alma gibi amaçlarla oluşturulmuş özel

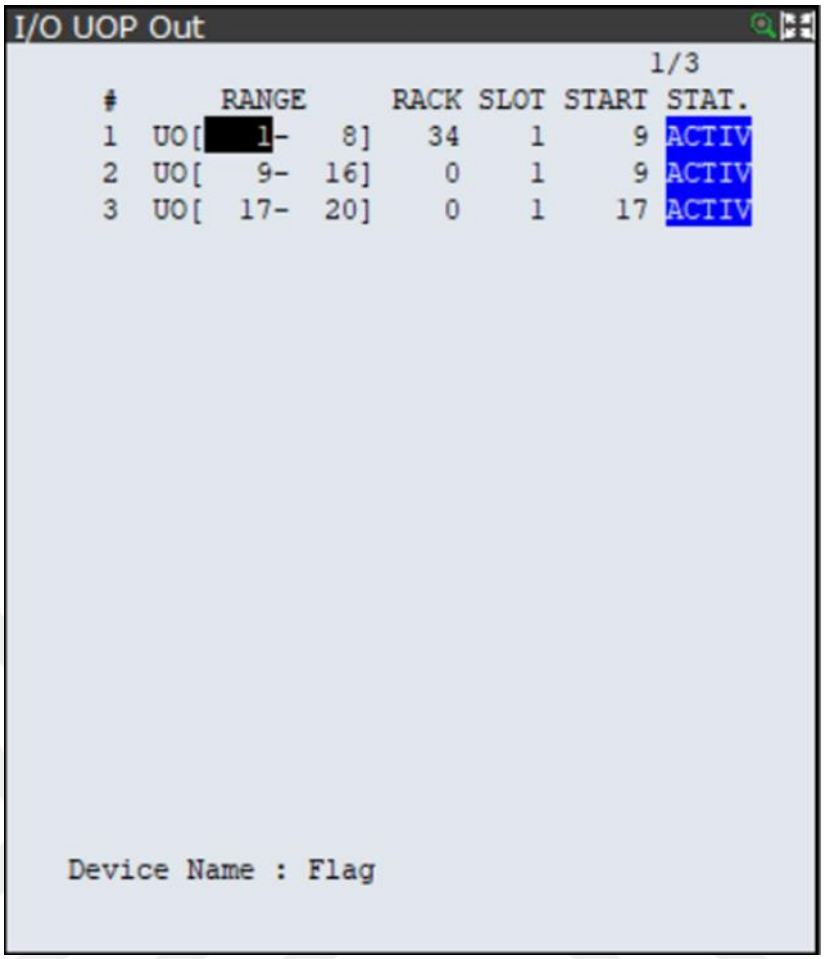
sinyaller bulunur. İsimlendirmeleri, içerikleri ve kullanım yöntemleri markadan markaya değişse de bütün robotlarda benzer mantıktadır. Fanuc marka robotlarda bu sinyaller “UOP” ismindedirler. Şekil 35 ‘de UOP çıkış sinyalleri, Şekil 36 ‘da UOP giriş sinyalleri görülmektedir. UOP çıkışları üzerinden robot programının ve robotun kendisine ait bilgiler alınmaktadır. UOP giriş sinyalleri ile ise robotun ve robot programlarının kontrolü sağlanmaktadır. UOP sinyallerinin doğrudan fiziksel giriş çıkış karşılıkları vardır. İstenirse haricen eklenen fiziksel giriş çıkışlara atanabilirler. Modbus haberleşmesinde tanımlanan giriş çıkışlardan robotun kontrol edilebilmesi için, bu tanımlanan giriş çıkışlar UOP sinyalleri ile eşlenmiştir. Bu işlem için UOP sinyalleri “FLAG” ismi verilen bir sinyal grubuna eşlenmiştir. Bu işleme ait ayarlar Şekil 37 ve Şekil 38 ‘de görülmektedir. İşlem sonrasında flag sinyallerinin UOP sinyallerine eşlenmiş görüntüsü Şekil 38 ‘de bulunmaktadır. Giriş çıkış sinyalleri de aynı flag sinyallerine eşlenerek UOP sinyalleri ile Modbus giriş çıkışları arasında bağlantı kurulmuştur. Bu işlem için yapılan ayarlamalara ait görüntü Şekil 39 ‘da bulunmaktadır. Sinyallerin eşlenmiş görüntüsü Şekil 40 ‘da görülebilir.



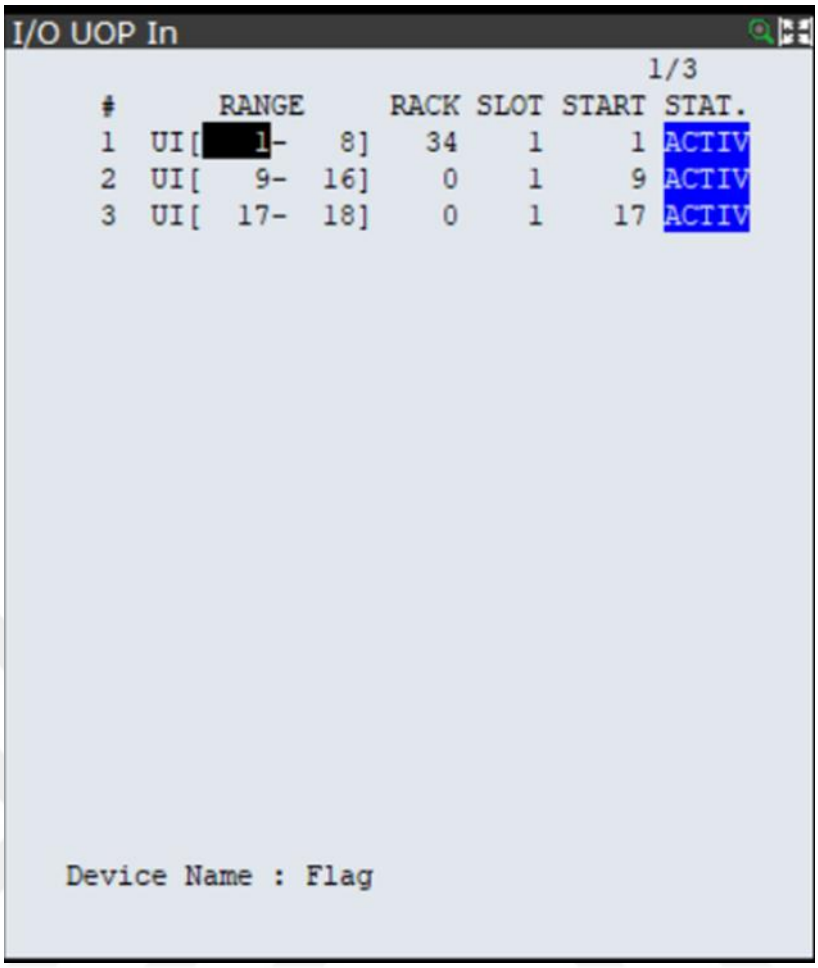
Şekil 35. UOP çıkışları görüntüsü



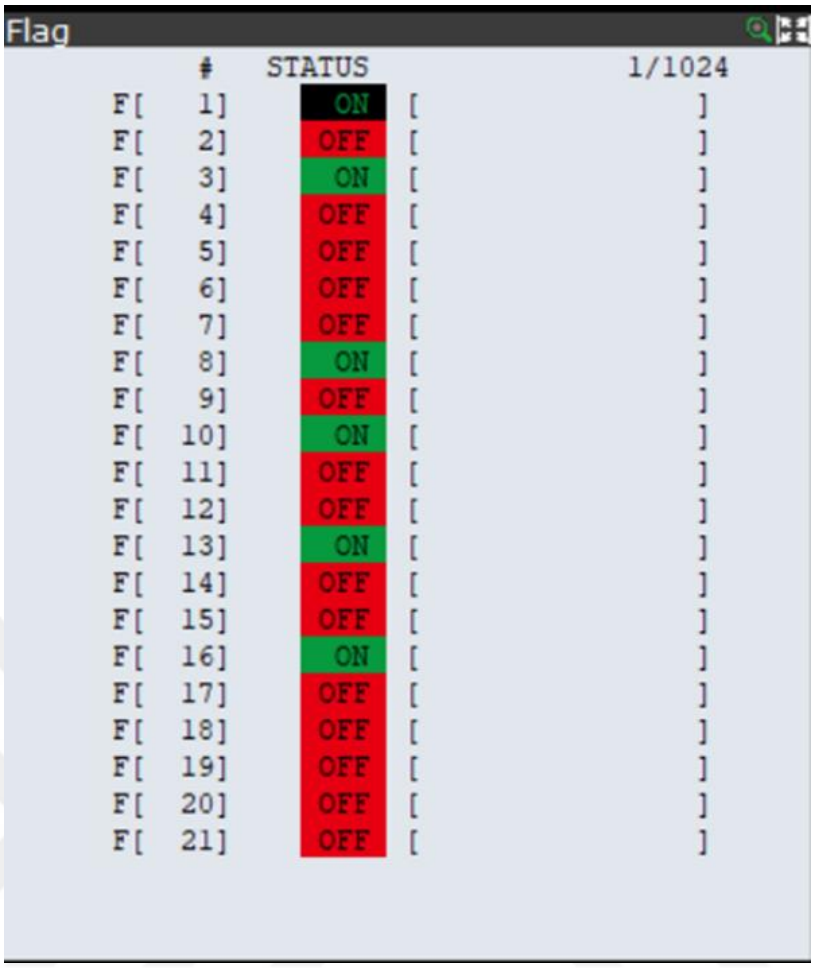
Şekil 36. UOP girişleri görüntüsü



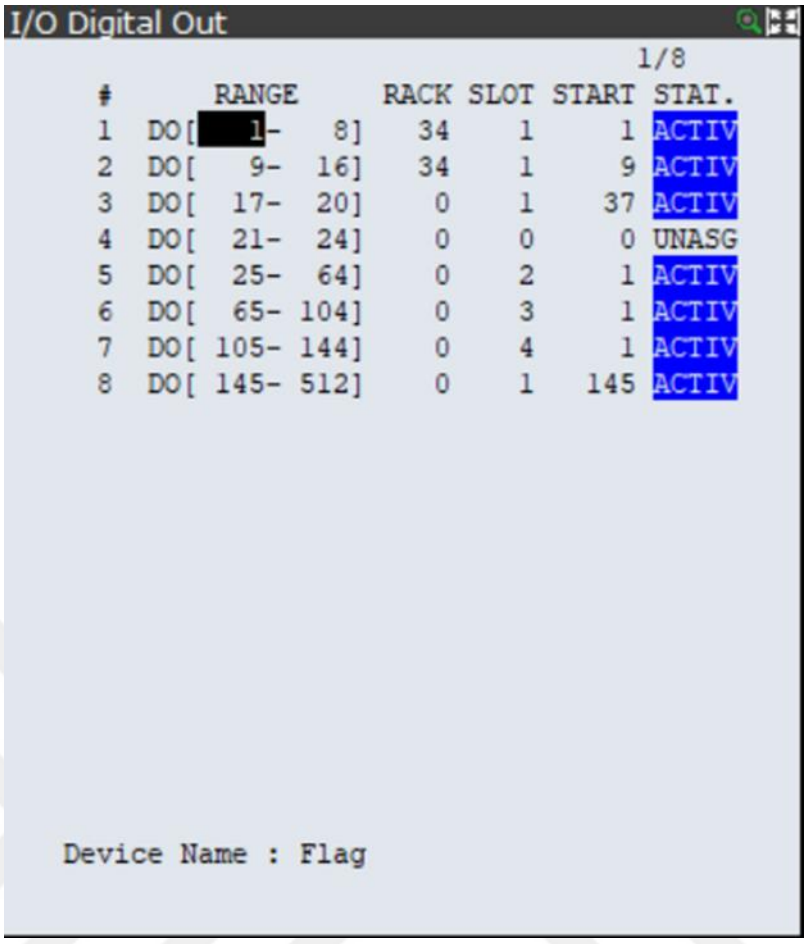
Şekil 37. UOP çıkışları atanması



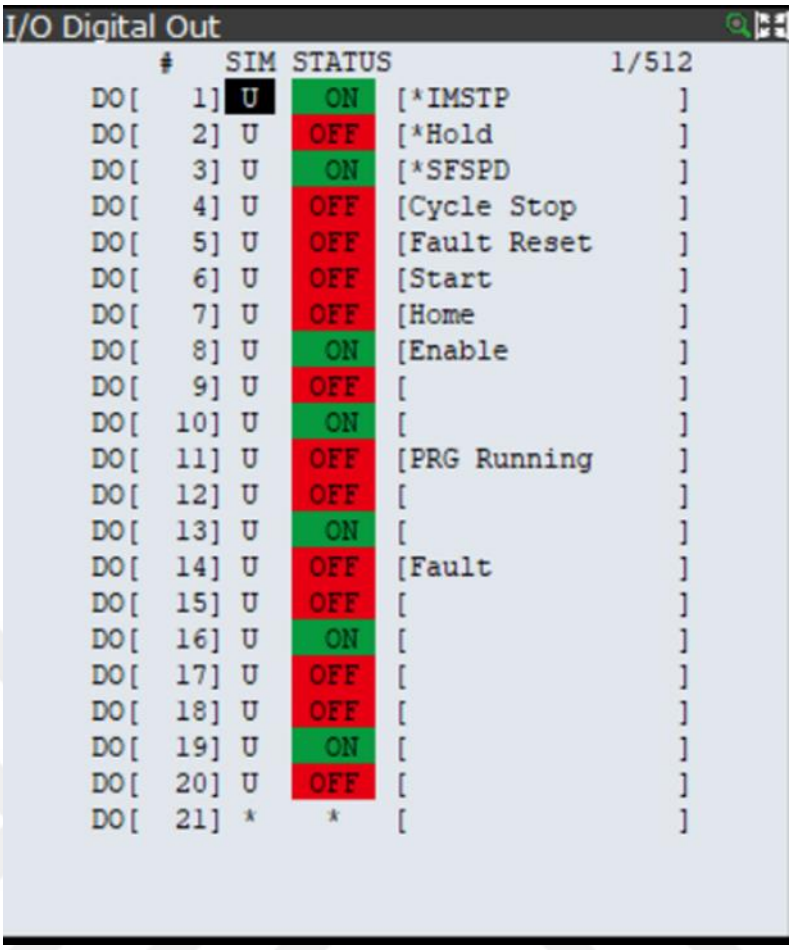
Şekil 38. UOP girişleri atanması



Şekil 39. Atama yapıldıktan sonra flag görüntüsü



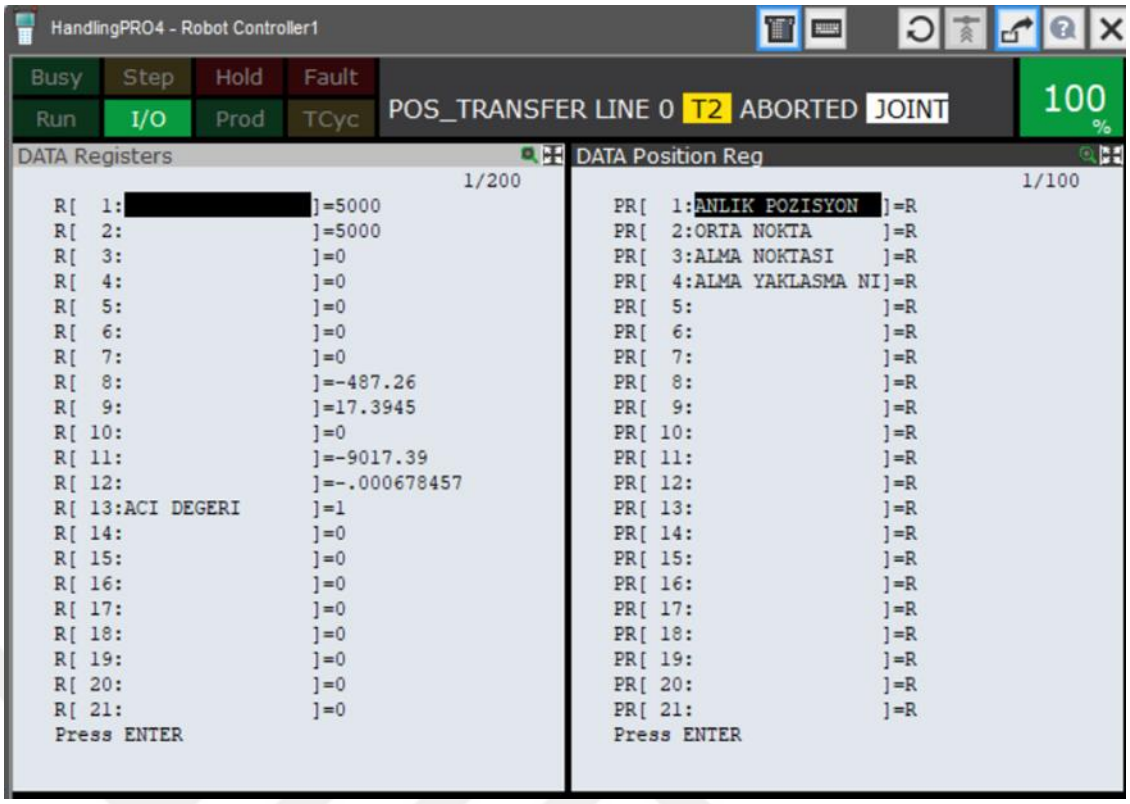
Şekil 40. Dijital çıkış atamaları



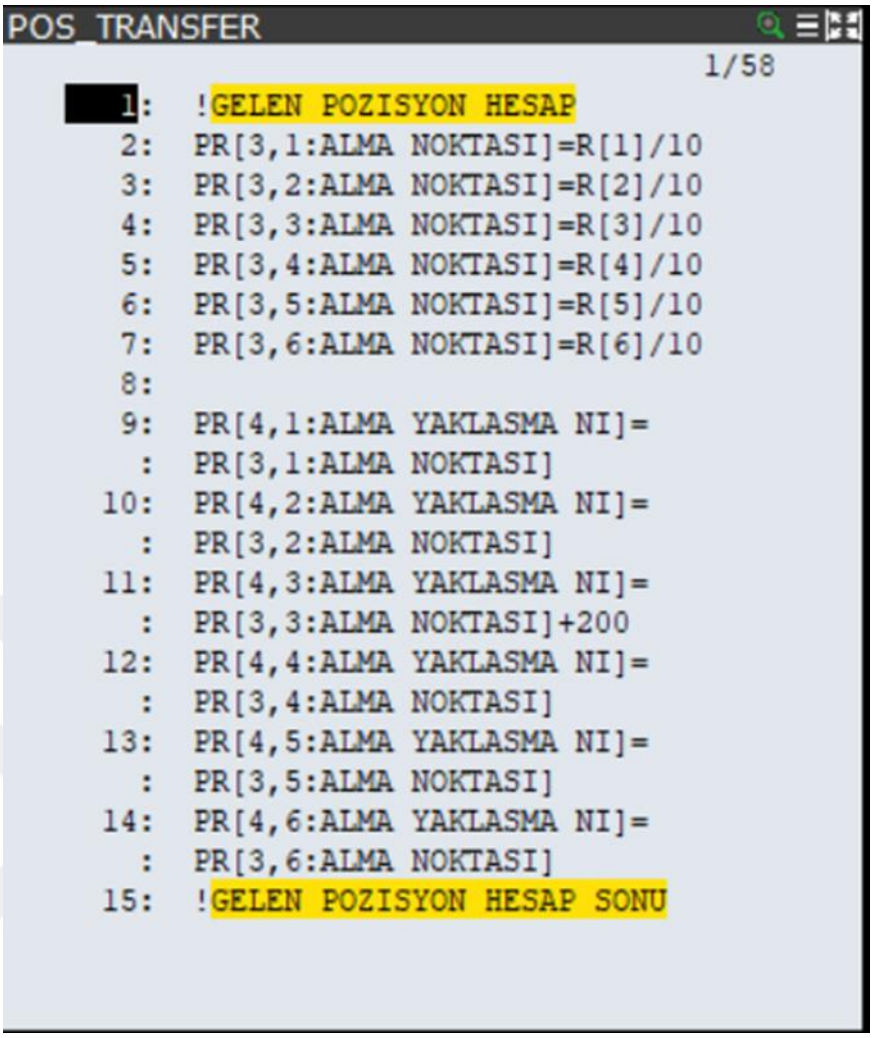
Şekil 41. Atama yapıldıktan sonra dijital çıkışların görüntüsü

Modbus haberleşme robotun dijital çıkışlarına, “Numerik Register” isimli ve “R[X]” kısaltmalı değişkenlere erişilmesini sağlar. Buradaki “X” ifadesi yerine değişken numarası yazılmaktadır.

Gelen pozisyon değerlerini almak için bir arka plan programı yapılma ihtiyacı duyulmuştur. X,Y,Z,W,P,R ifadelerinin her biri için ayrı bir değişken kullanarak bunlar bir pozisyon değişkeninin değerlerine atanmıştır. Ondalıklı sayıları ifade edebilmek için değerler bilgisayar yazılımından 10 ile çarpılarak gönderilmiştir. Dolayısıyla arka plan programında değerler bu sebeple 10 ‘a bölünmüştür. Değişkenlere ait görüntü Şekil 42 ‘de bulunmaktadır. Şekil 43 ‘de ise arka plan programına ait kodlar görülmektedir.



Şekil 42. Sayısal ve pozisyon değişkenleri

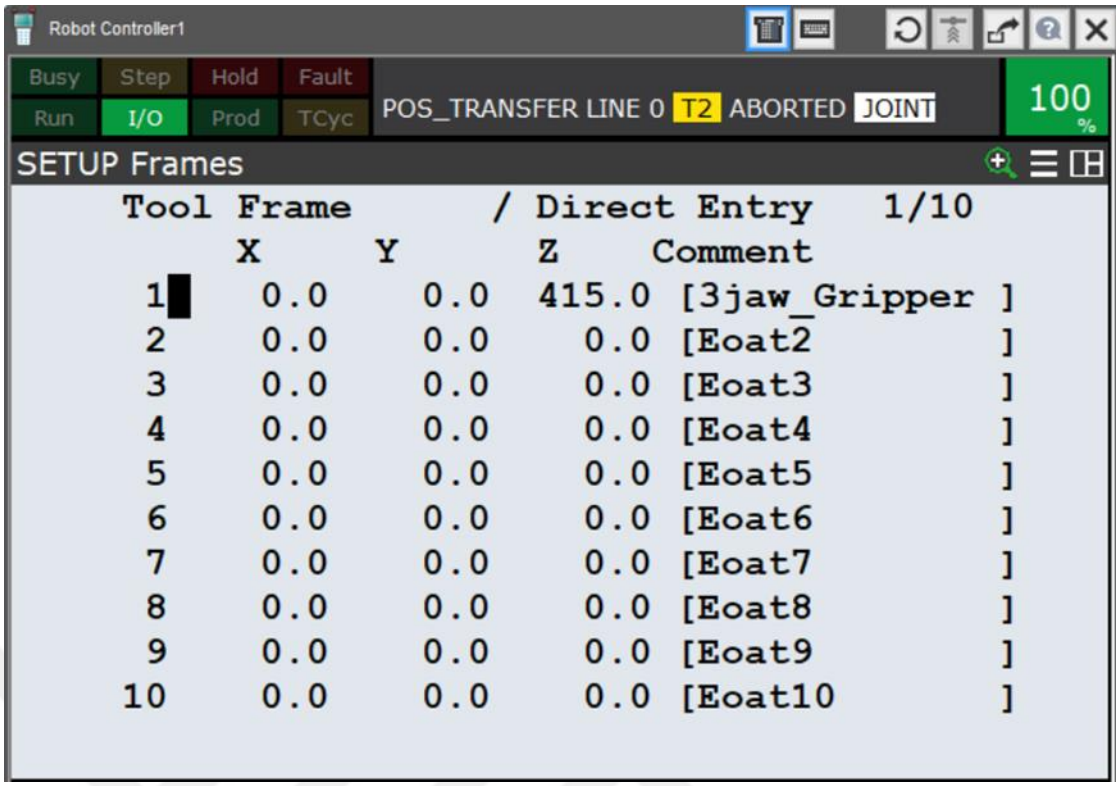


Şekil 43. Gelen pozisyon değeri hesabı

Robotlarda sistem değişkeni ismiyle ifade edilen üreticinin açık olarak sunduğu ve bazı özelliklerin kontrol edilmesini okunmasını sağlayan özel değişkenler bulunmaktadır. Örneğin bir numaralı eksenin anlık açı değerini öğrenilmek istenirse $SCR_GRP[1].$MCH_ANG[1] değişkeni bir sayısal değişkene eşitlenerek bu yapabilir. Bu şekilde açı değerleri dışarıya aktarılabilir. Bu projede de bu değişkenler robotun anlık pozisyonunun bilgisayar yazılımına gönderilmesi için kullanılmıştır. Robotun anlık pozisyon değerlerinin değişkenlere atanmasına ait görüntü Şekil 44 ‘de bulunmaktadır.

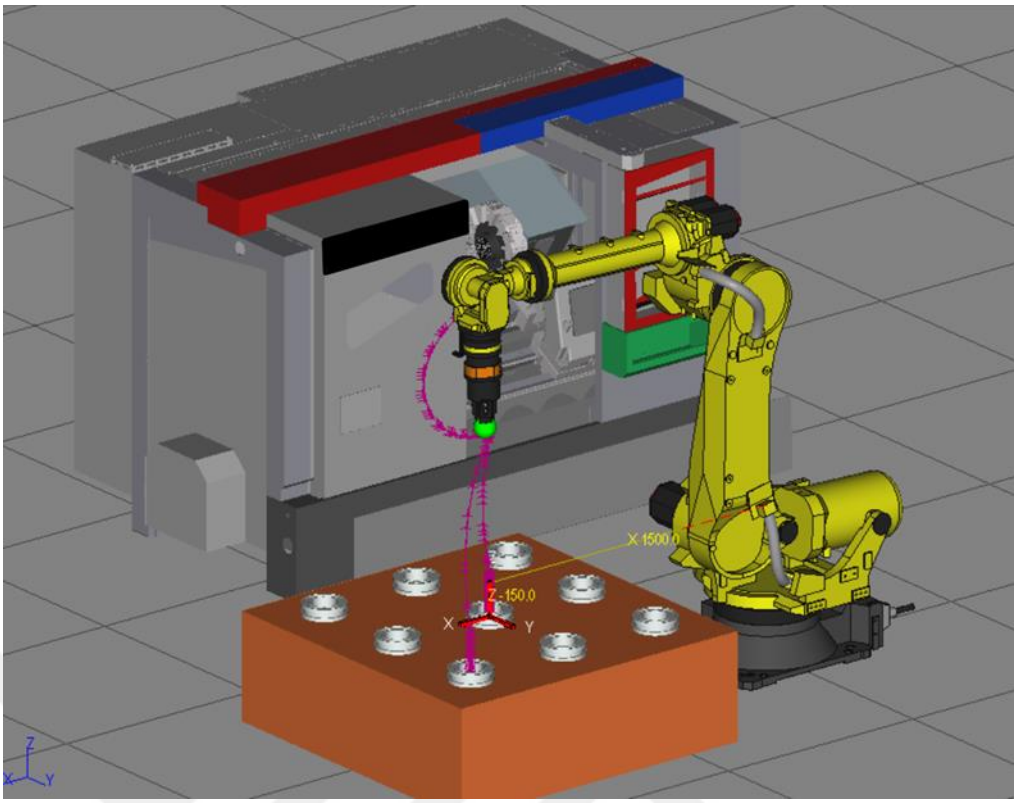
Şekil 44. Anlık açı değerleri aktarımı

Robotun uzaktan gerekli koordinata gönderilmesi için takım değerlerinin girilmesi gerekmiştir.. Flanş üzerine eş merkezli olarak bağlanan bir tutucuda X ve Y değerleri sıfıra eşittir. Tutucu takım robotun flanşına merkezli şekilde montajlandığı için sadece Z değeri girilmiştir. Robotun takım değerlerine ilişkin görüntü Şekil 45 ‘de görülmektedir.

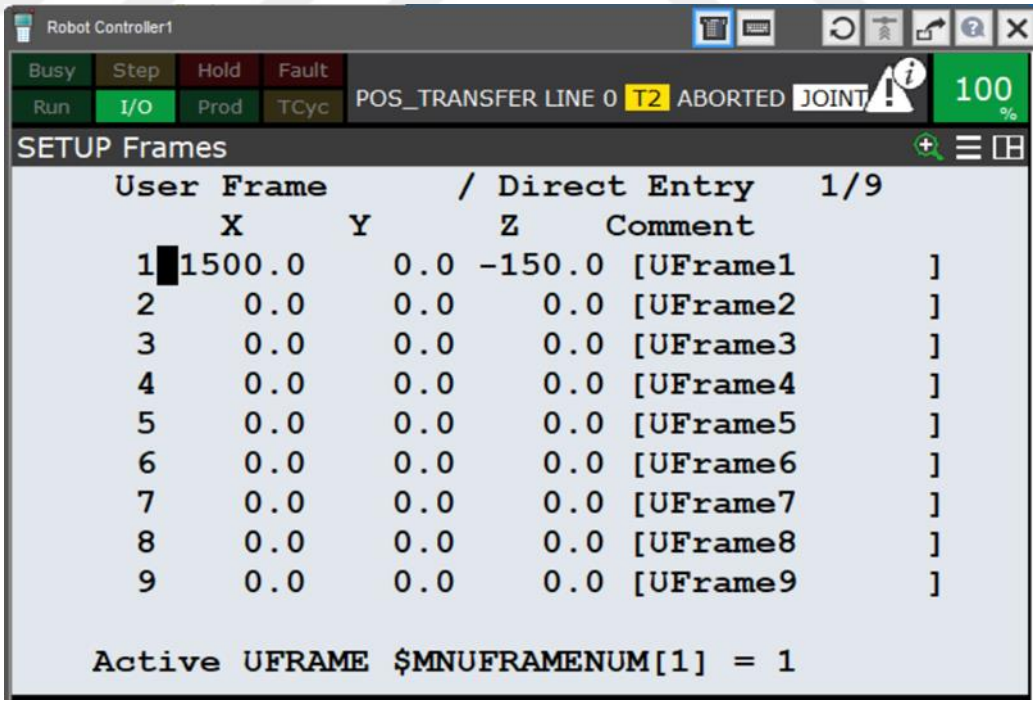


Şekil 45. Takım koordinat sistemi ekranı

Bilgisayar yazılımı üzerinden gelen pozisyon değerlerine göre robotun hareket edilebilmesi için referans bir koordinat merkezi olması gereklidir. Bu sebeple projede kullanıcı koordinat sistemi tanımlanmıştır. Kullanıcı koordinat sistemleri de aynı takım koordinatları gibi doğrudan robot üzerinden ölçü olarak girilebilir. Robotun ham ürünleri alacağı masanın merkezine bir kullanıcı koordinat merkezi tanımlanmıştır. Robotun 1. eksen merkezine olan uzaklıklar girilerek tanımlama yapılmıştır. Herhangi bir dönüklük olmadığı için W,P,R değerleri sıfır girilmiştir. Masanın robota göre konumunun görüntüsü Şekil 46 ‘da görülmektedir. Şekil 47 ‘de kullanıcı koordinat sisteminin tanımlanması için girilen değerler görülmektedir.



Şekil 46. Masa merkezi konumu



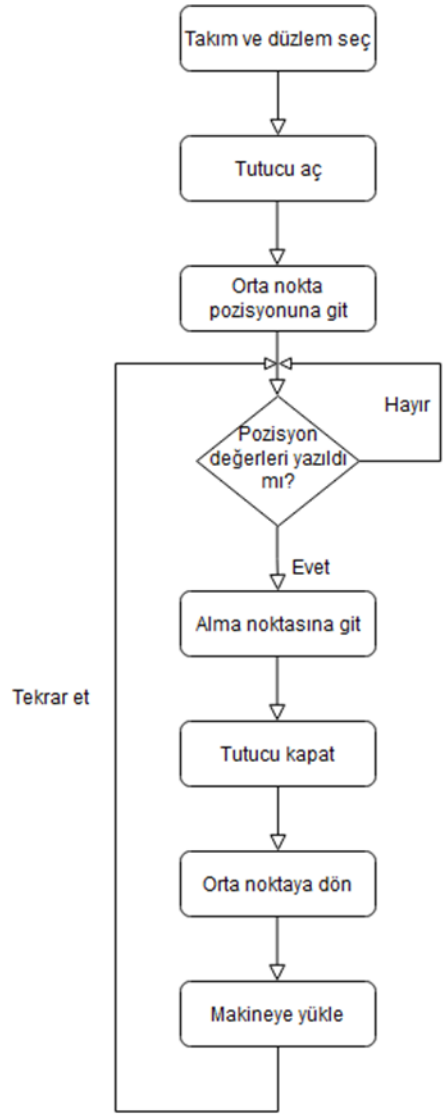
Şekil 47. Kullanıcı koordinat sistemi ekranı

3.2. Robot Programının Hazırlanması

Bir tepsi üzerinden ürünleri alabilmek için her bir ürünün pozisyonunun öğretilmesi gerekmektedir. Ayrıca bu pozisyonlara doğrusal olarak ulaşabilmek için ürünlerin güvenli bir mesafe üzerinde her bir ürün için birer pozisyon daha öğretilmesi gerekmektedir.

Bu çalışmada iki pozisyon değeri de dışarıdan güncellenmek istendiği için her bir ürün için farklı değişkenlere ihtiyaç yoktur. Sadece 1 ürün varmış gibi değişkenler oluşturulup robot bir döngüye sokularak sürekli olarak farklı pozisyonlardan ürün aldırılmıştır. Robot değerler yazıldı bilgisi aldığı zaman bu işlemleri gerçekleştirmektedir.

Dışarıdan gelen pozisyon değerleri arka plan programı vasıtası ile pozisyon değişkenlerine otomatik olarak yazılmaktadır. Bu robot arka plan programına ait kodlar EK-B ‘de görülebilir. Alma pozisyonuna ait değerler dışarıdan her bir eksen ve dönüklük değeri için ayrı ayrı sayısal değişkenlere yazılmaktadır. Bu sayısal değişkenler 10 değerine bölünerek “PR3” adresli ve “Alma Noktası” isimli pozisyon değişkenine eşitlenmektedir. 10 değerine bölünmesindeki amaç virgülden sonraki 1 haneyi kullanabilmek içindir. Dışarıdan bu değerler yazılırken 10 ile çarpılarak yazılmaktadır. Robotun ürün alma noktasına doğrusal olarak gidebilmesi için ihtiyaç duyduğu yaklaşma noktası X,Y ve dönüklük değerleri olarak alma noktası ile aynıdır. Z ekseninde bir güvenli yaklaşma mesafesi eklenmiştir. Bu pozisyon değişkenini oluşturmak için alma noktasının değerleri “PR4” adresli ve “Alma Yaklasma Noktası” isimli pozisyon değişkenine eşitlenmiştir. Alma yaklaşma noktasının Z ekseni değerlerine 200mm eklenmiştir. Değişken pozisyon verilerine göre sürekli bir döngü halinde ürün alan robot programına ait akış şeması Şekil 48 ‘de görülebilir.



Şekil 48. Değişken pozisyonlu program akış şeması

Pozisyon değerlerine göre ürün alıp makine besleyen robot programına ait kodlar EK-C ‘de görülebilir. Programda önceden öğretilen referans düzlem numarası ve malzemeyi alacak olan uç takıma ait numara seçilmiştir ve tutucu açık konuma alınmıştır. Robot bu noktadan itibaren sürekli olarak dışarıdan yeni değer yazılıp yazılmadığını kontrol eden bir döngüye girmektedir. Bu kontrolü bir sinyal üzerinden yapmaktadır. Bilgisayar programı üzerinden değer yazıldı bilgisi geldikten sonra robot

güncel değerlere göre ürün alma ve makine besleme işlemini yapar ve tekrar yeni değerleri beklemeye başlar.

Başka bir sistem üzerinden robot kontrolü sağlamak insanın anlık karar vererek yapmak istediği hareketleri robotta gerçekleştirmesine imkân vermektedir. Böylelikle kontrol paneli kullanılmadan veya çok uzak bir noktadan izlenerek robot hareket ettirilebilir.

Bu programlar kontrol panelini kullanmayı operatöre öğretip kullandırmak yerine belli kısıtlı bir mantık çerçevesinde özel tasarlanan panelin öğretilmesini sağlar. Operatörün manuel harekette de hata yapma şansı düşmüş olur.

Tehlikeli bölgelerde de anlık olarak karar verilip yapılması gereken özel görevlerde robot uzaktan kontrol edilerek böyle işlemler yaptırılabilir. Örneğin soğuk bir deponun içinde sabit bir programla çözülemeyen bir işlem varsa insanın yapacağı fiziksel müdahaleler yerine robot kollar kullanılabilir.

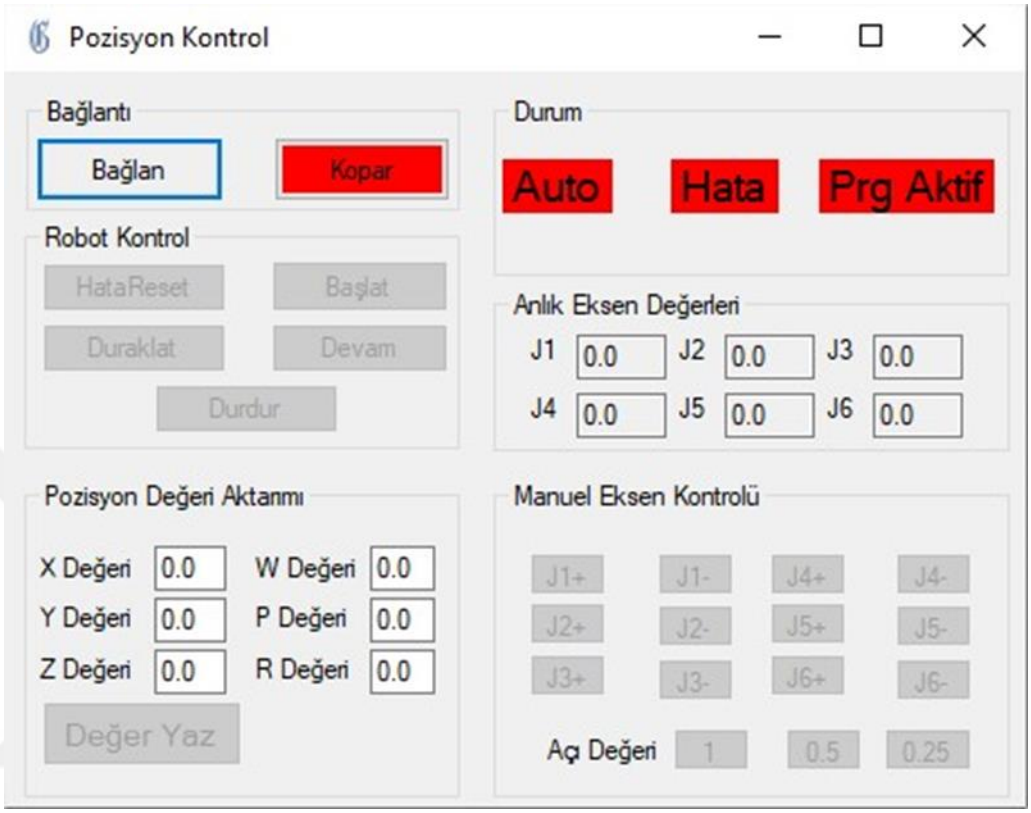
Bu çalışmada ilave olarak robotu uzaktan fiziksel olarak hareket ettirecek bir program yazılmıştır. Bunun için anlık eksen değerlerini kaydedilmiş ve o pozisyona açı farkları eklenerek robot o noktaya gönderilmiştir. Eksen hareketleri için her bir eksenin pozitif ve negatif yöne gitmesi ayrı ayrı belirtilmektedir ve bu yöne ne kadarlık bir açı hareketi yapması gerektiğini gönderilmektedir.

Bu programa ait kodlar EK-A ‘da görülebilir. Robot bir döngü içerisinde bilgisayar programındaki eksen hareket tuşlarından gelecek sinyalleri beklemektedir. Gelen sinyale göre “PR1” pozisyon değişkenindeki açı değerleri seçilen miktar kadar azaltılmakta veya arttırılmaktadır. Miktar seçimi EK-B ‘deki program vasıtası ile yapılmaktadır.

3.3. Bilgisayar Programının Hazırlanması

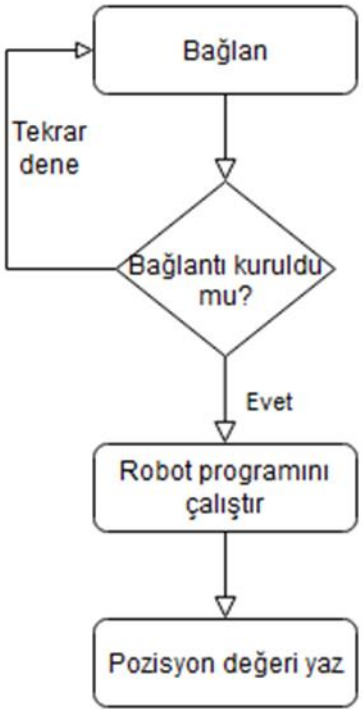
Tez çalışmasında pozisyon değerleri robota tek tek yazılıp yollanmaktadır. Bunun için bir ara yüz oluşturulmuştur. Bu ara yüz üzerinden robot ile modbus haberleşmesi kurulmuş ve pozisyon değerleri aktarılmıştır. Robot programının kontrolü için ise “Robot Kontrol” isminde bir panel oluşturulmuştur. “Durum” isimli bölümden robotun anlık durumu takip edilebilir. Ayrıca ara yüz üzerinden robotun fiziksel olarak hareket ettirilmesi için eksen kontrol tuşları ve güncel eksen

değerlerinin görülebileceği bir alan eklenmiştir. Bilgisayar programına ait ara yüz Şekil 49 ‘ da görülmektedir.



Şekil 49. Bilgisayar programı arayüzü

Programın yazılması için C# dili tercih edilmiştir. Tercih edilmesinde ara yüz oluşturmanın kolay oluşu ve modbus haberleşmesi için kütüphanelerin zengin oluşu etkili olmuştur. Yazılım modbus istemci modunda çalışmaktadır. Robot modbus sunucu konumundadır. Yazılım üzerinden robottaki sayısal değişkenler ve sinyaller kontrol edilebilmektedir. Bunun için modbus kütüphanesinin tekli ve çoklu sinyal yazma - okuma ve ayrıca tekli ve çoklu değişken yazma - okuma fonksiyonları kullanılmaktadır. Yazılımın pozisyon değişkenlerine değer yazan bölümüne ait akış şeması Şekil 50 ‘de görülmektedir.



Şekil 50. Yazılım akış şeması

Sunucuya bağlantının kurulabilmesi için öncelikle modbus nesnesinin oluşturulmuştur. Bu işlem için gerekli kod EK-Ç ‘de görülebilir. Sunucu robot, aynı bilgisayardaki bir simülatör olduğu için IP adresi olarak “127.0.0.1” yazılmıştır. Modbus TCP/IP haberleşmesi için 502 numaralı port kullanılmaktadır. Nesne oluşturulduktan sonra bağlantı kurulmuştur. Bağlantı kurma işlemi EK-Ç ‘de görülebilir.

Robot program kontrol işlemleri için kullanılan butonlara ait kodlar EK-Ç ‘de paylaşılmıştır. Bu işlemler için tek sinyal yazma fonksiyonu kullanılmıştır. Robot tarafında robotun kontrolünü saplayan UOP sinyalleri bu fonksiyon tarafından kontrol edilerek robot programımın çalıştırılmasına imkan sağlanmıştır.

Pozisyon değerlerinin yazılması işlemi için çoklu değişken yazma fonksiyonu kullanılmıştır. Değer yazıldı bilgisi robota sinyal üzerinden ulaştırılmıştır. Küsuratlarının robot tarafında kullanılabilmesi için değerler önce 10 sayısı ile

çarpılmaktadır. Robot tarafında bölünerek küsuratlar tekrar elde edilecektir. Bu işlemlere ait kodlar EK-Ç ‘de görülebilir.

SONUÇLAR VE ÖNERİLER

Tez çalışmasında robotlar, robotik otomasyonun oluşturulması için gerekli olan ekipmanlar ve endüstriyel haberleşme protokolleri hakkında detaylı bilgi verilmiştir.

Uzaydaki noktaların pozisyon değerlerinin girilebileceği ve robotun kontrolü için gerekli olan sinyallerin gönderilip alınabileceği bir ara yüz program oluşturulmuştur. Ara yüze bağlı girilen değerleri ve sinyalleri Modbus protokolü üzerinden gönderip alabilen bir yazılım yazılmıştır.

Gerçeğine uygun bir robot simülasyon ortamı oluşturularak Modbus haberleşme için gerekli olan haberleşme alt yapısı ayarlanmıştır. Robotun kontrolü için gerekli olan sinyal alt yapısı hazırlanmıştır. Dışarıdan gelen pozisyon değerlerini ve sinyalleri işleyip robot programlarının kullanımına hazırlayan bir arka plan program robot içerisinde yazılmıştır. Robotun fiziksel hareketlerini uzaktan kontrol eden bir robot programı yazılmıştır. Robot için temel bir makine yükleme ve boşaltma programı yazılmış ve bu program dışarıdan gelen pozisyon değerleriyle çalışacak şekilde ayarlanmıştır.

Ara yüz program üzerinden gönderilen değerler vasıtası ile robot bir referans noktaya göre farklı pozisyonlardan ürün alarak makine yükleme işlemini yapmaktadır. Robot yazılımları, bilgisayar yazılımı ve haberleşme alt yapısı taslak bir otomasyon sistemi özelliği taşımaktadır ve gerçek ortamda çalışabilecek özelliktedir. Gerçek ürünlere göre uyarlanır ve birden fazla pozisyon değeri bir liste içerisinden yollanırsa sahada çalışabilecek bir projeye dönüştürülebilir.

Çalışma sonucunda değişken pozisyon değerlerine göre farklı rotalar kullanan bir robotik otomasyon projesi alt yapısı oluşturulmuş. Bu sayede iş parçaları arası geçişlerin bilgisayar ortamında koordinatlar içeren listeler oluşturularak yapılabileceği insana bağımlılığı daha az ve insandan kaynaklı hata oranı daha düşük bir otomasyon sistemi oluşturulabilecektir. Çalışmadan elde edilen bu sonuçlar ile gerçek bir sistem alt yapısı kurulabilir. Pozisyon değerlerinin bir listeden gönderilmesi sağlanarak komple bir robot otomasyon sistemi oluşturulmuş olur. Yeni parçalarda referans noktaya göre bu pozisyon listeleri değiştirilerek parçalar arası kolay ve hızlı geçişe olanak sağlayan gerçek bir otomasyon sistemi oluşturulabilir.

KAYNAKÇA

Akbulut, M. A. (2007). *Modelling, identification and passivity-based control of 6 dof industrial robot* (Yayımlanmamış yüksek lisans tezi). İstanbul Teknik Üniversitesi, İstanbul.

Akkaya, Ş. (2015). *Fpga tabanlı modbus ağ geçidi tasarımı* (Yayımlanmamış yüksek lisans tezi). İstanbul Teknik Üniversitesi, İstanbul.

Alessandria, E., Senio, L., Vitturi, S. (2007). Performance analysis of ethernet/ip networks. IFAC Proceedings Volumes, 40(22), 391-398. doi:10.3182/20071107-3-FR-3907.00054

Alsabbagh, A. (2019). *Data analysis and force control of a 6-dof industrial robot* (Yüksek lisans tezi). University of Debrecen. Erişim adresi: http://dx.doi.org/10.13140/RG.2.2.20176.66567

Asubay, M. B. (2018). *Scada sistemlerinin tanıtımı ve kullanılan haberleşme protokolleri* (Yayımlanmamış yüksek lisans tezi). Bitlis Eren Üniversitesi, Bitlis.

Azizi, U. (2020). *Altı serbestlik dereceli robot kolu tasarım, modelleme ve imalatı* (Yayımlanmamış yüksek lisans tezi). İstanbul Aydın Üniversitesi, İstanbul.

Bakır, S. (2019). *Modbus rtu otomasyon protokolünün rs485 seri hat üzerinde çalışan haberleşme katmanında uygulanacak bir yöntemle hızının arttırılması* (Yayımlanmamış yüksek lisans tezi). Gebze Teknik Üniversitesi, Kocaeli.

Cylindrical coordinates assignment help. (t.y.). Erişim adresi: http://www.expertsmind.com/topic/robot-types/cylindrical-coordinates-910384.aspx

Demir, B. (2012). *Endüstriyel uygulamar için linux tabanlı 3g haberleşme özellikli akıllı modbus mesaj önceliklendirme mekanizmalı modbus gateway tasarımı* (Yayımlanmamış yüksek lisans tezi). Gebze Yüksek Teknoloji Enstitüsü, Kocaeli.

Efe, E. (2018). *Endüstriyel robot ve plc entegrasyonuyla talaşlı imalat üretim işleminin gerçekleştirilmesi* (Yayımlanmamış yüksek lisans tezi). Necmettin Erbakan Üniversitesi, Konya.

Ersöz, H. (2007). *Endüstriyel robotlar ve uygulama alanları* (Yayımlanmamış yüksek lisans tezi). Gazi Üniversitesi, Ankara.

Fanuc. (t.y.). Fanuc servo positioners. Erişim adresi: https://www.fanuc.eu/tr/tr/robotlar/aksesuarlar/hareket/konumland%C4%B1r%C4%B1c%C4%B1lar

Fanuc. (2019). Datasheet r2000ic 165f. Erişim adresi: https://www.fanuc.eu/tr/tr/robotlar/robot-filtre-sayfas%C4%B1/r-2000-serisi/r-2000ic-165f

Fanuc. (t.y.). FANUC. R-30ib plus controller. Erişim adresi: https://www.fanucamerica.com/products/robots/controllers

Fanuc. (t.y.). Lr mate 200id. Erişim adresi: https://www.fanuc.eu/tr/tr/robotlar/robot-filtre-sayfas%C4%B1/lrmate-serisi/lrmate-200-id

Fanuc. (t.y.). R-30ib plus. Erişim adresi: https://www.fanuc.eu/es/ue_zuk/robots/accesorios/robot-controller-r-30ib-plus

Gamess, E., Smith, B., Iii, G. A. F. (2020). Performance Evaluation of Modbus TCP in Normal Operation and Under A Distributed Denial of Service Attack. International Journal of Computer Networks and Communications, 12(2):1-21.

Huitsing, P., Chandia, R., Papa, M., Shenoi, S. (2008). Attack taxonomies for the modbus protocols. International Journal of Critical Infrastructure Protection, 1, 37-44. doi:10.1016/j.ijcip.2008.08.003

International Organization for Standardization. (2012). *Robots and robotic devices — Vocabulary* (ISO 8373: 2012). Erişim adresi: https://www.iso.org/obp/ui/#iso:std:iso:8373:ed-2:v1:en

Irmak, E., Erkek, İ. (2018). Endüstriyel kontrol sistemleri ve scada uygulamalarının siber güvenliği: modbus tcp protokolü örneği. Gazi Üniversitesi Fen Bilimleri Dergisi, 6(1), 1-16. doi:10.29109/http-gujsc-gazi-edu-tr.364411

Küçük, S. (2004). *Endüstriyel robotların modellenmesi ve çevrimdışı programlanması* (Doktora tezi). Kocaeli Üniversitesi Fen Bilimleri Enstitüsü, Kocaeli.

Kuka. (t.y.). Kuka linear unit kl 4000. Erişim adresi: https://www.kuka.com/en-de/products/robot-systems/robot-periphery/linear-units/kl-4000

Lian, F., Moyne, J. R., Tilbury, D. M.. (2001). Performance evaluation of control networks: ethernet, controlnet, and devicenet. IEEE Control Systems Magazine, 21(1), 66-83. doi:10.1109/37.898793

Liu, T., Tang, W., Jiang, H. (2009, Temmuz). The design of industrial ethernet adapter based on ethernet/ip. 2009 4th International Conference on Computer Science & Education, IEEE, Nanning. doi:10.1109/ICCSE.2009.5228400

Meier, M. (2019, 21 Ocak). Robot programming with kuka prc v3 [Blog yazısı]. Erişim adresi: http://mkmra2.blogspot.com/2019/01/robot-programming-with-kukaprc-v3.html

Ozan, E. (2020). *Robotlar ve uygulamaları* (Yayımlanmamış yüksek lisans tezi). Batman Üniversitesi Fen Bilimleri Enstitüsü, Batman.

Robot types. (t.y.). Erişim adresi: http://engineeronadisk.com/V2/book_integration/engineeronadisk-113.html

Robotic arms. (t.y.). Erişim adresi: http://www.massmind.org/techref/robot/arms.htm

Robotic equipment spotlight. (t.y.). Erişim adresi: https://www.robots.com/blogs/robotic-equipment-spotlight-grippers

Shahzad, A., Lee, M., Lee, Y. K., Kim, S. (2015). Real time modbus transmissions and cryptography security designs and enhancements of protocol sensitive information. Applied Cryptography and Security Concerns based on Symmetry for the Future Cyber World, 7(3), 1176-1210. doi:10.3390/sym7031176

Spindles for robotics. (t.y.). Erişim adresi: https://spindlerepair.com/robotics-spindles/

Süzer, E. S. (2006). *Uzaktan sayaç okuma teknikleri ve modbus-rtu, iec 61107 mod c protokolleri ile örnek yazılım* (Yayımlanmamış yüksek lisans tezi). İstanbul Teknik Üniversitesi, İstanbul.

Thompson, P. (2021, 14 Eylül). An easy-to-understand glossary of common robotics terms [Blog yazısı]. Erişim adresi: https://learn.g2.com/robot-terms

Types of robots in automation robotics. (t.y.). Erişim adresi: http://www.arbotist.com/types-of-robots.html

Weis, O. (2020, 23 Mart). How to read modbus data [Blog yazısı]. Erişim adresi: https://www.com-port-monitoring.com/how-to-read-modbus-data/

Whitter, T. (2017, 2 Ağustos). What to know about robotic welding gun liners. Erişim adresi: https://www.thefabricator.com/thefabricator/article/consumables/what-to-know-about-robotic-welding-gun-liners

Yıldırım, M. Y. (2019). *Gerçek zamanlı görüntü işleme temelli al – bırak yapabilen endüstriyel robot kol* (Yayımlanmamış yüksek lisans tezi). Karabük Üniversitesi, Karabük.

Yılmaz, D. (2010). *Bir robot kolunun bilgisayar destekli kinematik analizi* (Yayımlanmamış yüsek lisans tezi). Sakarya üniversitesi Fen Bilimleri Enstitüsü, Sakarya.

EKLER

EK-A

DIŞARIDAN GELEN KOMUTLAR İLE EKSENLERİ HAREKET ETTİREN ROBOT PROGRAMININ KODLARI

```
/PROG  JOG
/ATTR
OWNER           = MNEDITOR;
COMMENT         = "";
PROG_SIZE       = 1589;
CREATE          = DATE 20-08-27  TIME 22:01:50;
MODIFIED        = DATE 20-08-30  TIME 22:00:06;
FILE_NAME       = ;
VERSION         = 0;
LINE_COUNT      = 87;
MEMORY_SIZE     = 2001;
PROTECT         = READ_WRITE;
TCD:  STACK_SIZE        = 0,
      TASK_PRIORITY     = 50,
      TIME_SLICE        = 0,
      BUSY_LAMP_OFF     = 0,
      ABORT_REQUEST     = 0,
      PAUSE_REQUEST     = 0;
DEFAULT_GROUP   = 1,*,*,*,*;
CONTROL_CODE    = 00000000 00000000;
/APPL

AUTO_SINGULARITY_HEADER;
  ENABLE_SINGULARITY_AVOIDANCE    : TRUE;
/MN
   1:  LBL[1] ;
   2:  IF (DO[25:J1+]=ON) THEN ;
   3:  PR[1:ANLIK POZISYON]=JPOS    ;
   4:  PR[1,1:ANLIK POZISYON]=PR[1,1:ANLIK POZISYON]+R[13:ACI
DEGERI]    ;
   5:J PR[1:ANLIK POZISYON] 100% FINE    ;
   6:  DO[25:J1+]=OFF ;
   7:  ENDIF ;
   8:   ;
   9:  IF (DO[26:J1-]=ON) THEN ;
  10:  PR[1:ANLIK POZISYON]=JPOS    ;
  11:  PR[1,1:ANLIK POZISYON]=PR[1,1:ANLIK POZISYON]-R[13:ACI
DEGERI]    ;
  12:J PR[1:ANLIK POZISYON] 100% FINE    ;
  13:  DO[26:J1-]=OFF ;
```

```
  14:  ENDIF ;
  15:   ;
  16:  IF (DO[27:J2+]=ON) THEN ;
  17:  PR[1:ANLIK POZISYON]=JPOS    ;
  18:  PR[1,2:ANLIK POZISYON]=PR[1,2:ANLIK POZISYON]+R[13:ACI
DEGERI]    ;
  19:J PR[1:ANLIK POZISYON] 100% FINE    ;
  20:  DO[27:J2+]=OFF ;
  21:  ENDIF ;
  22:   ;
  23:  IF (DO[28:J2-]=ON) THEN ;
  24:  PR[1:ANLIK POZISYON]=JPOS    ;
  25:  PR[1,2:ANLIK POZISYON]=PR[1,1:ANLIK POZISYON]-R[13:ACI
DEGERI]    ;
  26:J PR[1:ANLIK POZISYON] 100% FINE    ;
  27:  DO[28:J2-]=OFF ;
  28:  ENDIF ;
  29:   ;
  30:  IF (DO[29:J3+]=ON) THEN ;
  31:  PR[1:ANLIK POZISYON]=JPOS    ;
  32:  PR[1,3:ANLIK POZISYON]=PR[1,3:ANLIK POZISYON]+R[13:ACI
DEGERI]    ;
  33:J PR[1:ANLIK POZISYON] 100% FINE    ;
  34:  DO[29:J3+]=OFF ;
  35:  ENDIF ;
  36:   ;
  37:  IF (DO[30:J3-]=ON) THEN ;
  38:  PR[1:ANLIK POZISYON]=JPOS    ;
  39:  PR[1,3:ANLIK POZISYON]=PR[1,3:ANLIK POZISYON]-R[13:ACI
DEGERI]    ;
  40:J PR[1:ANLIK POZISYON] 100% FINE    ;
  41:  DO[30:J3-]=OFF ;
  42:  ENDIF ;
  43:   ;
  44:  IF (DO[31:J4+]=ON) THEN ;
  45:  PR[1:ANLIK POZISYON]=JPOS    ;
  46:  PR[1,4:ANLIK POZISYON]=PR[1,4:ANLIK POZISYON]+R[13:ACI
DEGERI]    ;
  47:J PR[1:ANLIK POZISYON] 100% FINE    ;
  48:  DO[31:J4+]=OFF ;
  49:  ENDIF ;
  50:   ;
  51:  IF (DO[32:J4-]=ON) THEN ;
  52:  PR[1:ANLIK POZISYON]=JPOS    ;
  53:  PR[1,4:ANLIK POZISYON]=PR[1,4:ANLIK POZISYON]-R[13:ACI
DEGERI]    ;
  54:J PR[1:ANLIK POZISYON] 100% FINE    ;
  55:  DO[32:J4-]=OFF ;
```

```
  56:  ENDIF ;
  57:   ;
  58:  IF (DO[33:J5+]=ON) THEN ;
  59:  PR[1:ANLIK POZISYON]=JPOS    ;
  60:  PR[1,5:ANLIK POZISYON]=PR[1,5:ANLIK POZISYON]+R[13:ACI
DEGERI]    ;
  61:J PR[1:ANLIK POZISYON] 100% FINE    ;
  62:  DO[33:J5+]=OFF ;
  63:  ENDIF ;
  64:   ;
  65:  IF (DO[34:J5-]=ON) THEN ;
  66:  PR[1:ANLIK POZISYON]=JPOS    ;
  67:  PR[1,5:ANLIK POZISYON]=PR[1,5:ANLIK POZISYON]-R[13:ACI
DEGERI]    ;
  68:J PR[1:ANLIK POZISYON] 100% FINE    ;
  69:  DO[34:J5-]=OFF ;
  70:  ENDIF ;
  71:   ;
  72:  IF (DO[35:J6+]=ON) THEN ;
  73:  PR[1:ANLIK POZISYON]=JPOS    ;
  74:  PR[1,6:ANLIK POZISYON]=PR[1,6:ANLIK POZISYON]+R[13:ACI
DEGERI]    ;
  75:J PR[1:ANLIK POZISYON] 100% FINE    ;
  76:  DO[35:J6+]=OFF ;
  77:  ENDIF ;
  78:   ;
  79:  IF (DO[36:J6-]=ON) THEN ;
  80:  PR[1:ANLIK POZISYON]=JPOS    ;
  81:  PR[1,6:ANLIK POZISYON]=PR[1,6:ANLIK POZISYON]-R[13:ACI
DEGERI]    ;
  82:J PR[1:ANLIK POZISYON] 100% FINE    ;
  83:  DO[36:J6-]=OFF ;
  84:  ENDIF ;
  85:   ;
  86:  JMP LBL[1] ;
  87:   ;
/POS

/END
```

EK-B

POZİSYON DEĞERLERİNİ ALAN VE GÖNDEREN PROGRAM KODLARI

```
/PROG  POS_TRANSFER
/ATTR
OWNER           = MNEDITOR;
COMMENT         = "";
PROG_SIZE       = 1413;
CREATE          = DATE 20-08-24  TIME 20:42:52;
MODIFIED        = DATE 20-08-30  TIME 19:17:12;
FILE_NAME       = ;
VERSION         = 0;
LINE_COUNT      = 57;
MEMORY_SIZE     = 1749;
PROTECT         = READ_WRITE;
TCD:  STACK_SIZE        = 0,
      TASK_PRIORITY     = 50,
      TIME_SLICE        = 0,
      BUSY_LAMP_OFF     = 0,
      ABORT_REQUEST     = 0,
      PAUSE_REQUEST     = 0;
DEFAULT_GROUP   = 1,*,*,*,*;
CONTROL_CODE    = 00000000 00000000;
/APPL

AUTO_SINGULARITY_HEADER;
  ENABLE_SINGULARITY_AVOIDANCE    : TRUE;
/MN
   1:  !GELEN POZISYON HESAP ;
   2:  PR[3,1:ALMA NOKTASI]=R[1]/10    ;
   3:  PR[3,2:ALMA NOKTASI]=R[2]/10    ;
   4:  PR[3,3:ALMA NOKTASI]=R[3]/10    ;
   5:  PR[3,4:ALMA NOKTASI]=R[4]/10    ;
   6:  PR[3,5:ALMA NOKTASI]=R[5]/10    ;
   7:  PR[3,6:ALMA NOKTASI]=R[6]/10    ;
   8:   ;
   9:  PR[4,1:ALMA YAKLASMA NI]=PR[3,1:ALMA NOKTASI]    ;
  10:  PR[4,2:ALMA YAKLASMA NI]=PR[3,2:ALMA NOKTASI]    ;
  11:  PR[4,3:ALMA YAKLASMA NI]=PR[3,3:ALMA NOKTASI]+200    ;
  12:  PR[4,4:ALMA YAKLASMA NI]=PR[3,4:ALMA NOKTASI]    ;
  13:  PR[4,5:ALMA YAKLASMA NI]=PR[3,5:ALMA NOKTASI]    ;
  14:  PR[4,6:ALMA YAKLASMA NI]=PR[3,6:ALMA NOKTASI]    ;
  15:  !GELEN POZISYON HESAP SONU ;
  16:   ;
  17:  !ANLIK ACISAL POZISYON ;
  18:  R[7]=($SCR_GRP[1].$MCH_ANG[1]) ;
```

```
  19:  R[8]=($SCR_GRP[1].$MCH_ANG[2]) ;
  20:  R[9]=($SCR_GRP[1].$MCH_ANG[3]) ;
  21:  R[10]=($SCR_GRP[1].$MCH_ANG[4]) ;
  22:  R[11]=($SCR_GRP[1].$MCH_ANG[5]) ;
  23:  R[12]=($SCR_GRP[1].$MCH_ANG[6]) ;
  24:   ;
  25:  R[7]=R[7]*100    ;
  26:  R[8]=R[8]*100    ;
  27:  R[9]=R[9]*100    ;
  28:  R[10]=R[10]*100    ;
  29:  R[11]=R[11]*100    ;
  30:  R[12]=R[12]*100    ;
  31:  !ANLIK ACISAL POZISYON SONU ;
  32:   ;
  33:  !UZAKTAN KONTROL ACI DEGERI ;
  34:  IF (DO[37:1]=ON) THEN ;
  35:  DO[37:1]=OFF ;
  36:  DO[38:0.5]=OFF ;
  37:  DO[39:0.25]=OFF ;
  38:  R[13:ACI DEGERI]=1    ;
  39:  ENDIF ;
  40:   ;
  41:  IF (DO[38:0.5]=ON) THEN ;
  42:  DO[37:1]=OFF ;
  43:  DO[38:0.5]=OFF ;
  44:  DO[39:0.25]=OFF ;
  45:  R[13:ACI DEGERI]=.5    ;
  46:  ENDIF ;
  47:   ;
  48:  IF (DO[39:0.25]=ON) THEN ;
  49:  DO[37:1]=OFF ;
  50:  DO[38:0.5]=OFF ;
  51:  DO[39:0.25]=OFF ;
  52:  R[13:ACI DEGERI]=.25    ;
  53:  ENDIF ;
  54:  !UZAKTAN KONTROL ACI DEGERI SONU ;
  55:   ;
  56:   ;
  57:   ;
/POS
/END
```

EK-C

MAKİNE YÜKLEME BOŞALTMA PROGRAMINA AİT KODLAR

```
/PROG  IMALAT
/ATTR
OWNER           = MNEDITOR;
COMMENT         = "";
PROG_SIZE       = 429;
CREATE          = DATE 20-08-30  TIME 18:38:44;
MODIFIED        = DATE 20-08-30  TIME 21:54:58;
FILE_NAME       = ;
VERSION         = 0;
LINE_COUNT      = 16;
MEMORY_SIZE     = 893;
PROTECT         = READ_WRITE;
TCD:  STACK_SIZE        = 0,
      TASK_PRIORITY     = 50,
      TIME_SLICE        = 0,
      BUSY_LAMP_OFF     = 0,
      ABORT_REQUEST     = 0,
      PAUSE_REQUEST     = 0;
DEFAULT_GROUP   = 1,*,*,*,*;
CONTROL_CODE    = 00000000 00000000;
/APPL
AUTO_SINGULARITY_HEADER;
  ENABLE_SINGULARITY_AVOIDANCE    : TRUE;
/MN
   1:  UTOOL_NUM=1 ;
   2:  UFRAME_NUM=1 ;
   3:  CALL TUTUCU_AC    ;
   4:  LBL[1] ;
   5:J PR[2:ORTA NOKTA] 100% FINE    ;
   6:  IF (DO[40:DEGER YAZILDI]=ON) THEN ;
   7:  WAIT    .50(sec) ;
   8:J PR[4:ALMA YAKLASMA NI] 100% FINE    ;
   9:L PR[3:ALMA NOKTASI] 100mm/sec FINE    ;
  10:  CALL TUTUCU_KAPAT    ;
  11:L PR[4:ALMA YAKLASMA NI] 100mm/sec FINE    ;
  12:J PR[2:ORTA NOKTA] 100% FINE    ;
  13:  CALL YUKLEME    ;
  14:  DO[40:DEGER YAZILDI]=OFF ;
  15:  ENDIF ;
  16:  JMP LBL[1] ;
/POS
/END
```

EK-Ç

BİLGİSAYAR PROGRAMINA AİT KODLAR

Modbus nesnesinin oluşturulması:

```
ModbusClient modbusClient = new ModbusClient("127.0.0.1", 502);
```

Modbus bağlantısının kurulması:

```
private void button1_Click(object sender, EventArgs e)  //bağlan
        {
            try
            {
                modbusClient.Connect();

                if (modbusClient.Connected == true)
                {
                    button1.UseVisualStyleBackColor = false;
                    button1.BackColor = Color.LightGreen;
                }

                timer1.Enabled = true;

                button3.Enabled = true;
                button4.Enabled = true;
                button5.Enabled = true;
                button6.Enabled = true;
                button7.Enabled = true;
                button8.Enabled = true;
                button9.Enabled = true;
                button10.Enabled = true;
                button11.Enabled = true;
                button12.Enabled = true;
                button13.Enabled = true;
                button14.Enabled = true;
                button15.Enabled = true;
                button16.Enabled = true;
                button17.Enabled = true;
                button18.Enabled = true;
                button19.Enabled = true;
                button20.Enabled = true;
                button21.Enabled = true;
                button22.Enabled = true;
                button23.Enabled = true;
```

```
            }
            catch (Exception)
            {
                button1.UseVisualStyleBackColor = true;
                throw;
            }
        }
```

Modbus bağlantısının koparılması:

```
private void button2_Click(object sender, EventArgs e)  //kopar
        {
            try
            {
                modbusClient.Disconnect();
                button1.UseVisualStyleBackColor = true;
                timer1.Enabled = false;

                textBox7.Text = "0.0";
                textBox8.Text = "0.0";
                textBox9.Text = "0.0";
                textBox10.Text = "0.0";
                textBox11.Text = "0.0";
                textBox12.Text = "0.0";

                label13.BackColor = Color.Red;
                label14.BackColor = Color.Red;
                label15.BackColor = Color.Red;

                button3.Enabled = false;
                button4.Enabled = false;
                button5.Enabled = false;
                button6.Enabled = false;
                button7.Enabled = false;
                button8.Enabled = false;
                button9.Enabled = false;
                button10.Enabled = false;
                button11.Enabled = false;
                button12.Enabled = false;
                button13.Enabled = false;
                button14.Enabled = false;
                button15.Enabled = false;
                button16.Enabled = false;
                button17.Enabled = false;
                button18.Enabled = false;
```

```
                button19.Enabled = false;
                button20.Enabled = false;
                button21.Enabled = false;
                button22.Enabled = false;
                button23.Enabled = false;

            }
            catch (Exception)
            {

                //throw;
            }

        }
```

Program kontrol butonlarının oluşturulması:

```
private void button4_Click(object sender, EventArgs e)
//hatareset
        {
            modbusClient.WriteSingleCoil(4, true);
            Thread.Sleep(50);
            modbusClient.WriteSingleCoil(4, false);
        }

        private void button6_Click(object sender, EventArgs e)
//durakla
        {
            modbusClient.WriteSingleCoil(1, false);
        }

        private void button7_Click(object sender, EventArgs e)
//devam
        {
            modbusClient.WriteSingleCoil(1, true);
            Thread.Sleep(50);
            modbusClient.WriteSingleCoil(5, true);
            Thread.Sleep(50);
            modbusClient.WriteSingleCoil(5, false);
        }

        private void button5_Click(object sender, EventArgs e)
//başlat
        {
            modbusClient.WriteSingleCoil(5, true);
            Thread.Sleep(1000);
```

```
            modbusClient.WriteSingleCoil(5, false);
        }

        private void button8_Click(object sender, EventArgs e)
//durdur
        {
            modbusClient.WriteSingleCoil(3, true);
            Thread.Sleep(50);
            modbusClient.WriteSingleCoil(3, false);

        }
```

Koordinat değerlerini yazan fonksiyonun oluşturulması:

```
private void button3_Click(object sender, EventArgs e)  //değeryaz
        {
            try
            {
                float xValue =
(float)Convert.ToDouble(textBox2.Text);
                xValue = xValue * 10;
                int xValueInt = (int)Convert.ToInt32(xValue);

                float yValue =
(float)Convert.ToDouble(textBox3.Text);
                yValue = yValue * 10;
                int yValueInt = (int)Convert.ToInt32(yValue);

                float zValue =
(float)Convert.ToDouble(textBox4.Text);
                zValue = zValue * 10;
                int zValueInt = (int)Convert.ToInt32(zValue);

                float wValue =
(float)Convert.ToDouble(textBox6.Text);
                wValue = wValue * 10;
                int wValueInt = (int)Convert.ToInt32(wValue);

                float pValue =
(float)Convert.ToDouble(textBox5.Text);
                pValue = pValue * 10;
                int pValueInt = (int)Convert.ToInt32(pValue);

                float rValue =
(float)Convert.ToDouble(textBox1.Text);
                rValue = rValue * 10;
```

```
                int rValueInt = (int)Convert.ToInt32(rValue);

                modbusClient.WriteMultipleRegisters(0, new int[6]
{ xValueInt, yValueInt, zValueInt, wValueInt, pValueInt,
rValueInt});

                modbusClient.WriteSingleCoil(39, true);
//değeryazıldı

            }
            catch (Exception)
            {

                //throw;
            }

        }
```

Robot anlık pozisyon açı değerlerinin okunması:

```
private void readAngles()
        {
            try
            {
                int[] readHoldingRegisters =
modbusClient.ReadHoldingRegisters(6, 6);

                float angle1 = (float)(readHoldingRegisters[0]);
                float angle2 = (float)(readHoldingRegisters[1]);
                float angle3 = (float)(readHoldingRegisters[2]);
                float angle4 = (float)(readHoldingRegisters[3]);
                float angle5 = (float)(readHoldingRegisters[4]);
                float angle6 = (float)(readHoldingRegisters[5]);


                textBox7.Text = (angle1 / 100).ToString();
                textBox8.Text = (angle2 / 100).ToString();
                textBox9.Text = (angle3 / 100).ToString();
                textBox10.Text = (angle4 / 100).ToString();
                textBox11.Text = (angle5 / 100).ToString();
                textBox12.Text = (angle6 / 100).ToString();
            }
            catch (Exception)
            {
```

```
                //throw;
            }


        }
```

Robotun anlık durumunu kontrol eden fonksiyonun hazırlanması:

```
private void statusCheck()
        {
            try
            {
                bool[] readCoils = modbusClient.ReadCoils(10, 7);
                if (readCoils[0] == true)    //prg running
                {
                    label15.BackColor = Color.Green;
                }
                else
                {
                    label15.BackColor = Color.Red;
                }

                if (readCoils[3] == true)    //fault
                {
                    label14.BackColor = Color.Green;
                }
                else
                {
                    label14.BackColor = Color.Red;
                }

                if (readCoils[6] == true)    //auto
                {
                    label13.BackColor = Color.Green;
                }
                else
                {
                    label13.BackColor = Color.Red;
                }
            }
            catch (Exception)
            {

                //throw;
            }
}
```

Fonksiyonların çağrılması:

```
private void timer1_Tick(object sender, EventArgs e)
        {
            readAngles();
            statusCheck();

        }
```

Robotu hareket ettiren butonların ve hareket miktarı belirleyen butonların hazırlanması:

```
private void button9_Click(object sender, EventArgs e)  //j1+
        {
            modbusClient.WriteSingleCoil(24, true);
        }

        private void button10_Click(object sender, EventArgs e)
//j1-
        {
            modbusClient.WriteSingleCoil(25, true);
        }

        private void button12_Click(object sender, EventArgs e)
//j2+
        {
            modbusClient.WriteSingleCoil(26, true);
        }

        private void button11_Click(object sender, EventArgs e)
//j2-
        {
            modbusClient.WriteSingleCoil(27, true);
        }

        private void button14_Click(object sender, EventArgs e)
//j3+
        {
            modbusClient.WriteSingleCoil(28, true);
        }

        private void button13_Click(object sender, EventArgs e)
//j3-
        {
            modbusClient.WriteSingleCoil(29, true);
        }
```

```
        private void button20_Click(object sender, EventArgs e)
//j4+
        {
            modbusClient.WriteSingleCoil(30, true);
        }
        private void button19_Click(object sender, EventArgs e)
//j4-
        {
            modbusClient.WriteSingleCoil(31, true);
        }
        private void button18_Click(object sender, EventArgs e)
//j5+
        {
            modbusClient.WriteSingleCoil(32, true);
        }
        private void button17_Click(object sender, EventArgs e)
//j5-
        {
            modbusClient.WriteSingleCoil(33, true);
        }
        private void button16_Click(object sender, EventArgs e)
//j6+
        {
            modbusClient.WriteSingleCoil(34, true);
        }
        private void button15_Click(object sender, EventArgs e)
//j6-
        {
            modbusClient.WriteSingleCoil(35, true);
        }
        private void button21_Click(object sender, EventArgs e)
//1degree
        {
            modbusClient.WriteSingleCoil(36, true);
        }
        private void button22_Click(object sender, EventArgs e)
//0.5degree
        {
            modbusClient.WriteSingleCoil(37, true);
        }
        private void button23_Click(object sender, EventArgs e)
//0.25degree
        {
            modbusClient.WriteSingleCoil(38, true);

        }
```