T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

FABRİKA ÜRETİM OTOMASYONUNDA ROBOTLU SİMÜLASYON UYGULAMALARI VE OFFLİNE PROGRAMLAMA DA VERİMLİLİĞİ ARTTIRMA

YAŞAR BAYKAL
YÜKSEK LİSANS TEZİ
ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

GEBZE
2023

T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

FABRİKA ÜRETİM OTOMASYONUNDA ROBOTLU SİMÜLASYON UYGULAMALARI VE OFFLİNE PROGRAMLAMA DA VERİMLİLİĞİ ARTTIRMA

YAŞAR BAYKAL

YÜKSEK LİSANS TEZİ
ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMANI
PROF. DR. FUAD ALİEW

GEBZE
2023

T.R.

GEBZE TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

ROBOTIC SIMULATION APPLICATIONS IN FACTORY PRODUCTION AUTOMATION AND INCREASING PRODUCTIVITY IN OFFLINE PROGRAMMING

YAŞAR BAYKAL

A THESIS SUBMITTED FOR THE DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF ELECTRONIC ENGINEERING

THESIS SUPERVISOR

PROF. DR. FUAD ALİEW

GEBZE

2023

GEBZE TEKNİK ÜNİVERSİTESİ	**YÜKSEK LİSANS JÜRİ ONAY FORMU**

GTÜ Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 13/07/2023 tarih ve 2023/39 sayılı kararıyla oluşturulan jüri tarafından 09/10/2023 tarihinde tez savunma sınavı yapılan Yaşar BAYKAL'ın tez çalışması Elektronik Mühendisliği Anabilim Dalında YÜKSEK LİSANS tezi olarak kabul edilmiştir.

JÜRİ

ÜYE
(TEZ DANIŞMANI) : Prof. Dr. Fuad ALIEW

ÜYE : Doç. Dr. Serdar Süer ERDEM

ÜYE : Dr. Kutluk Bilge ARIKAN

ONAY

Gebze Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü Yönetim Kurulu'nun ……/……/…… tarih ve ……/…….. sayılı kararı.

İMZA/MÜHÜR

ÖZET

Fabrika üretim hatlarının kurulum süresi ne kadar uzun olursa üretimin başlaması da o kadar gecikir, bu sürelerin azalması ve kurulumda çıkabilecek hataları en aza indirgemek için simülasyon programları kullanılmaktadır.

Bu çalışmada simülasyon programında simülasyonu gerçekleştirilen tek hücreli hattın kurulumu yapıldı ve robot offline yörünge dosyaları yüklendi. Ara nokta ekleme metodu kullanılarak yörünge optimizasyonu yapıldı. Uygulama sonrası kaçıklık hesabı yapılarak yörüngelerin düzenlenmesi gerçekleştirildi. Endüstriyel uygulamalarda genellikle robot programları insanlar tarafından hat kurulduktan sonra manuel Öğretme El Terminali (TP) üzerinden yapılır. Fakat biz Process Simulate uygulamasında robotu offline programladık. Robot back-up dosyasını bilgisayar üzerinden USB (Universal Serial Bus) ile alarak kullandığımız FANUC ARC Mate 100iC model robota gene USB ile bağlanarak offline programı yükledik. Robotların isimleri değişse de çalışma prensibi aynıdır bizim elimizde FANUC marka robot olduğu için biz bu marka üzerinden ilerledik isteğe göre diğer marka robotlara da evirilebilir. Yüklenen back-up çalıştırıldığında sistemin kurulumundan kaynaklı kaçıklıklar mevcut bizde olması gereken bir nokta ile olduğu yer ile farkı bulmak amacı ile ölçüm cihazı kullandık. Farkı hazırlamış olduğumuz programa hem back-up hem de X, Y, Z, RX, RY, RZ koordinat farklarını girip çıkış aldığımız zaman yeni program sahaya uygun bir şekilde gelmesi sağlatılmaktadır.

Anahtar Kelimeler: Simülasyon (PS), Robot (Fanuc), Offline Programlama Verimliliği, Otomasyon Sistemleri.

SUMMARY

The longer the installation period of the factory production lines, the delayed the start of production. Simulation programs are used to reduce these times and to minimize the errors that may occur during the installation.

In this study, the single-cell line simulated in the simulation program was installed and the robot offline trajectory files were loaded. The orbit optimization was conducted using the method of adding midpoints. After the application, the misalignment calculation was made and the arrangement of the trajectories was carried out. In industrial applications, robot programs are usually made by humans after the line is set up, via the manual Teach Hand Terminal (TP). But we programmed the robot offline in Process Simulate. We took the robot backup file from the computer via USB (Universal Serial Bus) and connected the FANUC ARC Mate 100iC model robot, which we used, via USB, and loaded the offline program. Although the names of the robots have changed, the working principle is the same. When the loaded backup is run, there are glitches caused by the installation of the system, we used a measuring device to find the difference between a point where it should be and where it is. When we enter both the back-up and X, Y, Z, RX, RY, RZ coordinate differences to the program we have prepared the difference, and exit, the new program is ensured to come to the field in a suitable way.

Key Words: Simulation (PS), Robot (Fanuc), Offline Programming Efficiency, Automation Systems.

TEŞEKKÜR

Başta, yüksek lisans eğitimimde ve akademik hayatımda desteğini ve yardımlarını hiçbir zaman esirgemeyip bilgisi ile bu çalışmanın oluşmasının yolunu açan danışmanım Prof. Dr. Fuad ALIEW'e,

Deney çalışmalarımı destekleyen kendimi geliştirmem için elinden geleni yapan Ercan ERPOLAT abime,

Hep yanımda olan kardeşim can yoldaşım Yunus Emre SADIK'a,

Tüm hayatım boyunca her zaman yanımda olan, hiçbir fedakârlıktan kaçınmadan sonsuz sevgi ve anlayışlarıyla her türlü duygu durumunu benimle göğüsleyen sevgili Annem Münevver BAYKAL ve Babam Hamza BAYKAL'a, ve kıymetli Kardeşlerim Cafer BAYKAL ve Muhammed Said BAYKAL'a,

Son olarak bana her zaman destek olan, bütün zorlukları benimle göğüsleyen sevgili eşim Esma BAYKAL'a ve dünyaya gelmek üzere olan geliş haberiyle motivasyonumu arttıran sevgili Kızım Hazel BAYKAL'a en içten teşekkürlerimi sunarım.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR DİZİNİ

Simgeler ve Kısaltmalar		**Açıklamalar**
ARC	:	Endüstriyel Kaynak Tipi
CAD	:	Bilgisayar Destekli Tasarım
CATIA	:	Eşgüdümlü Üç Boyutlu Etkileşimli Uygulama
CMM	:	Koordinat Ölçüm Makineleri
DOF	:	Serbestlik Derecesi
DSM	:	Tasarım Yapısı Matrisi
ESRC	:	Taklit Edilmiş Özel Robot Kontrolcüsü
GTU	:	Gebze Teknik Üniversitesi
JT	:	Üç Boyutlu Veri Formatı
LS	:	Fanuc Robot Back-up Dosya Formatı
OLE	:	Nesne Bağlama ve Yerleştirme
OLP	:	Çevrimdışı Programlama
OPC	:	Process Kontrolü İçin OLE
PLM	:	Ürün Yaşam Döngüsü Yönetimi
PS	:	Process Simulate Programı
ROS	:	Robot İşletim Sistemi
SQL	:	Yapılandırılmış Sorgu Dili
TCP	:	Robot Araç Merkez Noktası
TP	:	Robot El Terminali
USB	:	Evrensel Seri Veri yolu
XML	:	Genişletilebilir İşaretleme Dili
3D	:	Üç Boyutlu

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

1. GİRİŞ

Fabrika otomasyonu üretimde büyük rol oynamaktadır ve özel olarak tasarlanmış robotik uygulamalarını içermektedir. Üretim süreçleri için farklı ihtiyaçlar olduğundan, teslim edilen sistemler de farklılık gösterir. Hem tedarik edilen hem de kendileri tarafından yapılan robotik manipülatörlerin kullanımı, müşteriye bir sistem planlama ve teslim etme için gereken beceri tabanını genişletir. Farklı tedarikçilerin robot sistemlerindeki çeşitlilik, çeşitli yerel programlama dilleri ve sistem ortamlarında destek ve bilgi talebini artırır.

Simülasyon, mühendislik ve sistem tasarımında uzun süredir kullanılan bir araçtır. Matematiksel veya fiziksel bir modelle gerçek dünya fenomenini taklit ederek, projenin veya sürecin gerçekleştirilmeden önceki tasarım hatalarının üstesinden gelmek mümkün olur. Bir ürünün veya hizmetin bir simülasyon modeli üreterek, işletmeler planlama maliyetlerini azaltabilmiştir çünkü kullanılan zaman azaltılmış olur. Bu tez, farklı amaçlar için bireysel olarak tasarlanması amaçlanan robotik sistemlerin değişen kurulumlarını test etmek ve bu sistemlerin teslimat sürecini iyileştirmek için uygun bir simülasyon ve çevrimdışı programlama yazılımı bulmaya odaklanmaktadır.

Teslimat verimliliğini ve robot donanım tasarım aşamasını iyileştirmek için bir simülasyon ve çevrimdışı programlama platformu, hem yerel olarak özel olarak tasarlanmış robotlar hem de piyasada bulunan robot manipülatörlerle kullanılmak üzere uygundur. Özellikler için gereksinimler belirlemek, mevcut araçların karşılaştırılması ve değerlendirmesi için sistemli bir yaklaşım sağlar. Bu şekilde, robot sistem teslimatının erken aşamalarında robot sisteminin sanal modelini test etmek için uygun bir araç bulmanın mümkün olabileceği düşünülmektedir.

1.1. Tezin Amacı, Katkısı ve İçeriği

Önerilen bu tez çalışması kapsamında, temel amaç kurulumu gerçekleştirilecek olan robotlu hatların kurulumunda kullanılmak üzere uygun bir çevrimdışı

programlama ve simülasyon aracı kullanmaktır. Kullanılan araçların kullanılabilirliğini ve performansını bir program yardımıyla arttırmaktır.

Çevrimdışı programlamanın ve simülasyonun faydaları ve kullanımına yönelik çalışma, çevrimdışı programlama süreçleri üzerine literatür taraması yapılarak sonuçlandırılmıştır. Amaç, özellikle büyük tesislerin kurulumu için çevrimdışı programlama ile yapılan yörünge ve programların verimliliğini arttırma konusunda kanıtlar sunmaktır. Literatürdeki örnekler, bu uygulamaların kazanılan değer üzerindeki bilimsel arka planı ve somut kanıtları sağlamak amacıyla, simülasyon ve çevrimdışı programlamadan faydalanarak örnek robotlu hat ile çalışma yapılarak yansıtılmıştır.

Robot manipülatörlerin kontrolü, modellemesi ve dolayısıyla simülasyonu temel alan karmaşık matematiksel teorilere dayanan simülasyon yöntemlerinin, çevrimdışı programlamanın ve robot manipülatörlerin ile ilgili simülasyon ve geliştirme araçları literatür taraması yapılmıştır. Robotların kontrolü, modellenmesi ve dolayısıyla simülasyonu karmaşık matematiksel teorilere dayandığından, tezin bilimsel arka planında robot sisteminin kinematiği ve hareket kontrolüne yer verilmiştir.

Simülasyon aracının kullanışlılık ve uygunluğu, mevcut bir robotlu hat simülasyon ve kontrol ortamının test edilmesi yoluyla pratikte test edilir. Var olan robotlu hattın simüle edilmesinin temel nedeni, simülasyon aracının robot geometrileri ve programlama için geçerliliğini göstermektir. Seçilen simülasyon aracının robot sistemi simüle etme yeteneğini ve genel simülasyon sürecindeki performansını gözden geçirmektir.

Bu tez çalışması sonucunda, simülasyonu gerçekleştirilen ve sahaya kurulan robotlu üretim hatlarının, kurulum ve devreye alma süreleri kısalarak verimlilik artırılmaktadır.

2. LİTERATÜR ÖZETİ

2.1. Simülasyon

Simülasyon terimi, bir model (CAD) kullanarak gerçek bir olguyu taklit etmek amacıyla yapılan bir görev veya süreci genelleştirerek çeşitli anlamlara sahiptir. Model matematiksel, fiziksel veya sanal olabilir. Matematiksel simülasyon, bir sistem davranışını tahmin etmek için matematiksel formüllerin kullanılmasıyla gerçekleştirilir ve uygun bir tahmin aracı olarak işlev gösteren matematiksel olarak modellenmiş simülasyonlar genellikle bilgisayar simülasyonlarıdır. Bilgisayar simülasyonları, belirli bir değişikliğin etkisini tahmin etmek için farklı endüstrilerde kullanılan bir araç olarak kullanılır. Ekonomistler, ekonomi politikalarının uygulanmasıyla ilgili farklı sonuçları tahmin etmek ve incelemek için bilgisayar simülasyonlarını kullanırlar [1]. Uygulamalı becerileri eğitmek için kullanılan bazı bilgisayar oyunu simülatörleri, profesyonel düzeyde insanla yapılan simülasyonun yerini alabilir veya genişletebilir. Bu, insan güvenliği ve hatalı uygulama riski gibi konuları içeren tıbbi personelin eğitiminde kullanılmasıyla açıkça görülebilir [2].

2.1.1. Robotik Sistemlerin Simülasyonu

Birçok mühendislik alanı, çeşitli matematiksel model tabanlı analizler yaparak tasarımın uygunluğunu kontrol etmek için simülasyonları kullanır. Bazı simülasyonlar bir tasarımın geleceğini tahmin etmek için kullanılırken, birçok simülasyon sadece belirli bir durumda sistemin görselleştirilmesi olarak bile işlev görebilir. Bir sistemin tüm verilerine sahip olmak yararlı olabilirken, bazı durumlarda sınırlı bir model tamamen yeterli olabilir. Örneğin, bir robot manipülatörünün görselleştirilmesi, sistemin fizik tabanlı bir modelini gerektirmeyebilir [3].

Diğer alanlardaki simülasyona çok benzer şekilde, eğitim ve sistem kullanımı eğitimi, robot teknolojisi ve dinamik makinelerle ilgili teknik konuları öğretmek için etkileşimli simülasyon ortamlarından büyük ölçüde fayda sağlar. Daha önceki yapılan bir çalışmaya göre, Wuhan Bilim ve Teknoloji Üniversitesi, robot çerçeve çalışması olarak ROS'u ve simülasyon ortamı olarak Gazebo'yu kullanarak robotlarla ilgili

uygulamaların öğretim yöntemlerini büyük ölçüde geliştirmeyi başardı. Simülasyon yapmanın çeşitli konularla ilgili olarak faydalı olduğunu belirttiği gibi, robotlar aynı zamanda simülasyon için bir araç olarak da kullanılabilir [4]. Bir başka çalışmada ise, robotik ve simülasyon arasındaki benzersiz bir kesişim yöntemini biyorobotik olarak tanımlar ve biyolojik varlıkların, örneğin hayvan türlerinin davranışlarını incelemek için robotik cihazları kullanarak gerçek tür davranışlarını taklit etme yöntemi olarak açıklar [5].

Robotikteki son gelişmeler genellikle çeşitli amaçlar için yapay zekâ uygulamalarına odaklanmaktadır. Robotik alanında yapılan en büyük araştırmaların odak noktası olan hareketli robotlar, kentsel trafiğin ortasında olduğu gibi sürekli değişen ve yoğun alanlarda otonom navigasyona odaklanmaktadır. Rastgele bir ortamın haritalanması, robot ortamına yerleştirilebilecek sabit ve hareketli engelleri algılamak için yapay zekâ, algoritmalar ve sensörler gerektirir. Bu tür algoritmaların test edilmesi zaman alıcı, maliyetli ve tekrar edilmesi zor olabilir. Simülasyonlar hem rastgele hem de yarı rastgele engel hareket desenlerini modelleyerek farklı tarama türlerini ve çevreyi test etmeyi mümkün kılar. Robot hareket etme yeteneği ile etkileşimli olan fiziksel kısım da robotun gerçek dünya robotuyla benzer şekilde çalışmasını sağlamak için modellenebilir [3], [6].

Aynı durum, robot manipülatörleri için de geçerlidir. Manipülatörler, kavrama, kaynak yapma, görüntüleme, problama, ekstrüzyon gibi çeşitli görevleri içerir ve esnek görevler için kavrama çözümü üretme gibi daha belirgin olmayan görevler gibi bazı tanımlanması zor görevleri de içerir. Bu görevlerin her biri yapay zekâ, sensörler ve makine görüşü uygulamaları ile geliştirilebilir. Mobil robot platformlarına kıyasla manipülatörler, daha yüksek hassasiyetle görev nesnelerini manipüle etmeleri gerektiğinden, nesnelerin algılanması mobil platformlardakilere göre daha küçüktür. Ayrıca, değişen şekillere, rijit veya deforme olanlara uyum sağlayan kavrama yöntemlerinin gereksinimleri, bölge tarama görüntüleme ve nesne algılama yeteneklerinin yanı sıra esnek görevler için bir kavrama çözümü üretme tasarımı için daha fazla gereksinimler ortaya koyar. Manipülatörler, mobil platformların uzaysal esnekliğini ve manipülatörlerin hassasiyetini birleştirerek entegre edilir. Manipülatörlerin simülasyonu, mobil robotların simülasyonuna kıyasla daha basittir, çünkü manipülatörlerin hareketle ilgili rastgele faktörleri daha azdır. Bu, kapsamlı

fizik tabanlı simülasyonun manipülatörlerin modellemesi için gerekli olmadığı nedenlerden biridir.

Robotikteki tüm bu alt alanlar, robot sistemlerinin çeşitli özelliklerini simüle edebilen bir simülatör gerektirir. Robotlar için genel olarak kullanılabilen simülatörler bulunmasına rağmen, robotik uygulama alanlarına odaklanan bazı simülasyon yazılımları da mevcuttur. Bu alanlarda fiziksel olayların simülasyonu için kullanılırlar. 2016-2020 yılları arasında, kullanılabilir simülatör sayısının artmasıyla birlikte robotik simülatörlerle ilgili yaklaşık 5000 alıntı yapılmıştır. Robotik simülatörler için genel bir gereklilik, ROS (Robot İşletim Sistemi) veya NVidia Isaac gibi diğer robotik teknolojilerin kullanımına izin veren işlevsel eklentilerin kullanılabilmesidir. Bu, farklı teknolojilerin entegrasyonu için büyük destek sunan popüler çerçeveler olduğu için geniş kapsamlı robotik sistemlerin geliştirilmesi için kullanılırlar [3].

2.2. Çevrimdışı Programlama

Çevrimdışı programlama ve çevrimiçi programlama, robotik sistemlerin programlanması için iki ana yöntemdir. Çevrimiçi programlama, fiziksel robot cihazını hareket ettirerek veya öğreterek pozisyonları ve hareketleri programlamayı içerir. Öte yandan, çevrimdışı programlama, robot yazılım aracını veya gerçek robotik cihazı içermeyen diğer uygun yöntemleri kullanarak programlama yapmaktır. Genel olarak, çevrimdışı ve çevrimiçi programlama ortamları arasındaki karşılaştırma, sistemin sanal modelinin doğruluğuna bağlıdır. Sistem ne kadar doğru bir şekilde modellendi ise, tasarım aşamasında sanal tasarıma konu olan özellikler o kadar fazla olabilir. Çevrimdışı programlama aşamasında yakalanan en küçük sorunlar, çevrimdışı kodun çevrimiçi ortama geçirilmesi sırasında yetersiz erişim ve manipülatör hareket programlama ile ilgili tehlikelerdir [7].

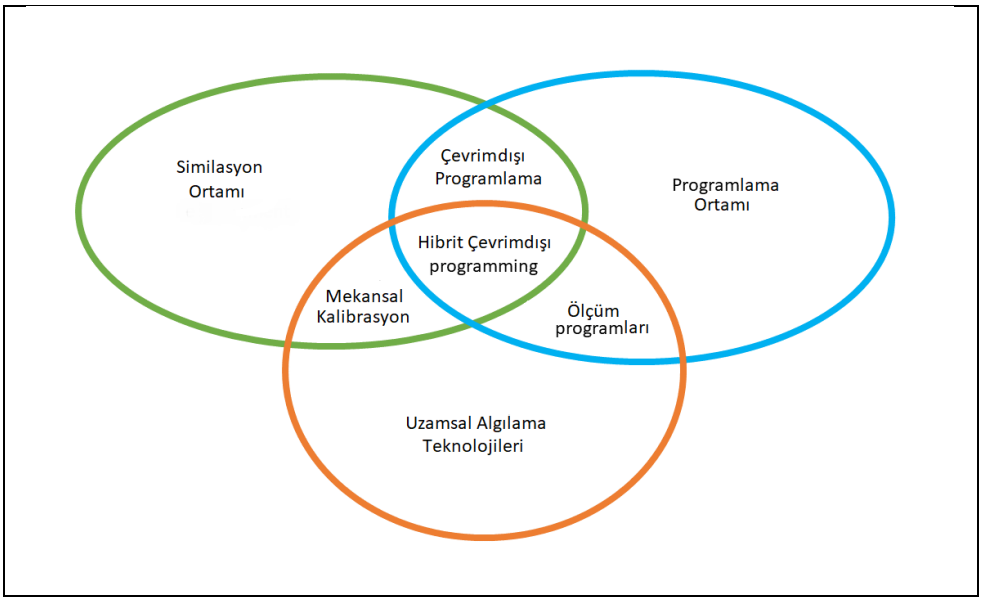
2.2.1. Çevrimdışı Programlamanın ve Simülasyonun Rolleri

Çevrimdışı programlama (OLP), hem müşteri hem de robot sistemini sunan kuruluş tarafından kullanılabilir. Bir robot sisteminin devreye alınma sürecini inceleyerek, değer eklemek için tamamlanması gereken iki görev vardır: mantıksal programlama ve hareket programlama. Mantıksal programlama, robot sürecinin

omurgasıdır, çünkü robotun nasıl çalıştığını tanımlar. Mantıksal programlama, robot sistemini kurulum ortamının fiziksel çevresinden etkilenmediği için, robot sisteminin teslimat sürecinin erken aşamalarında başlatılabilir. Bu nedenle, çevrimdışı programlama, mantıksal programlamaya önemli bir değer katmaz.

Bununla birlikte, hareket programlama, robotun çevresindeki ortamdan ciddi şekilde etkilenir ve henüz kurulmamış bir robot için doğru hareket komutlarını programlamak zordur. Temel sorunlar, gerçek dünya nesnelerinin konumları ile sanal ortamda kullanılan 3B modellerin kökeni arasındaki farklardır [8]. Hem özel hem de evrensel çevrimdışı programlama yazılımlarını kullanarak, doğru kinematik zincire sahip bir robot modeli, kurulum ortamını taklit eden bir sanal 3B ortamda manipüle edilebilir [9]. Bu nedenle, robot sisteminin simülasyon modeli, hareket komutlarını planlamak ve çevrimdışı durumdan sonra hareketleri ve pozisyon set noktalarını ayarlanabilir hale getirmek için büyük bir değer sağlar.

Kaynaklarda belirtildiği gibi, kaynaklanan avantajlar çevrimdışı programlamanın ve simülasyonun kullanıldığı kaynaklama işlemleri için zaman açısından verimli yöntemlerdir. Özellikle, kaynaklama endüstrisi, sanayi robotlarının daha tipik kullanımlarından biridir ve otomatik olarak kaynaklama yolları oluşturma ihtiyacı, zaman açısından verimli kaynak işlemleri için gelişmiş yöntemleri teşvik etmiştir. Bu tür bir yöntem, robot sisteminin çevrimdışı programlama modelini ve çalışma parçasını görsel veya fiziksel olarak algılanan verilerle birleştirerek, insansız hareket programlaması için hareket komutları ve torç kontrol programları otomatik olarak oluşturur. Bu şekilde, çevrimdışı programlama uygulamaları, varsayılan 3B konum verileri ve ölçülen verilerle otomatik hareket programlamasıyla işleme programlarının verimli bir şekilde oluşturulmasını sağlar [10], [11].



Şekil 2.1: Venn diyagramı formatında offline programlamanın gösterimi

Literatür referanslarına [9], [10], [11] dayanarak, Şekil 2.1'de gösterilen programlama, simülasyon ve sensörlerin kapsamlarını ve bu kapsamların birleşimini temsil eden bir Venn diyagramı olarak oluşturulabilir. Venn diyagramının aşağıdaki gibi tanımlanan bir liste haline getirilmesi mümkündür:

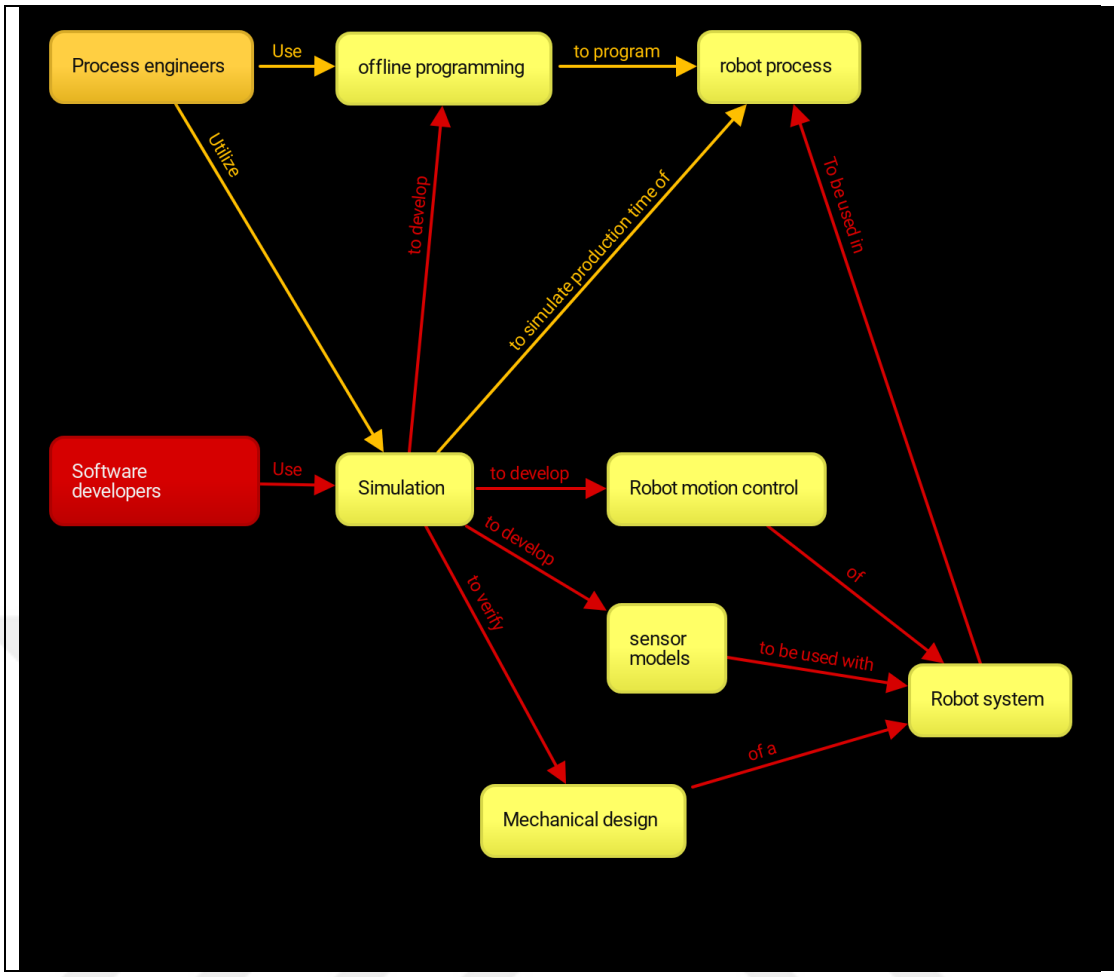
- Simülasyon ortamı, 3 boyutlu bileşenleri de içeren sistemlerin sanal modelini temsil eder.
- Programlama ortamı, programlama süreçlerinde kullanılan teknolojileri temsil eder.
- Uzaysal algılama teknolojileri, gerçek dünya nesnelerinin konumlarını ölçmek için kullanılan sensörleri temsil eder.
- Uzaysal kalibrasyon, nesnelerin gerçek dünya konumlarının sanal ortama çevrilmesini ifade eder.
- Offline programlama, robot sisteminin simülasyon modelini kullanarak hareketleri programlama pratiğini ifade eder.

- Ölçüm programları, ölçüm süreçlerini kontrol eder ve ölçülen verileri kullanılabilir bilgiye dönüştürür.

- Hibrit offline programlama, yukarıda bahsedilen tüm teknolojilerin ve uygulamaların birleşimini kullanarak süreç programlarının otomatik olarak oluşturulmasını sağlar.

Tez kapsamı dikkate alındığında, simülasyon ve offline programlama arasındaki ayrımı belirlemek için kullanım durumu farklılıkları açıklanmalıdır. Offline programlama, gerçek bir robota ihtiyaç duymadan hareket programlama yöntemidir, simülasyon ise bir robotu ve sistem ortamında bazı fiziksel davranışları görselleştirmek olarak anlaşılabilir [3]. Bu nedenle simülasyon ortamı, robot sistemlerinin tasarımı için bir test ortamı ve aracı olarak daha çok anlaşılabilir, program kodu oluşturma aracı değildir. Aşağıdaki zihin haritası, offline programlama ve simülasyon kullanım durumlarındaki farkları tanımlar.

Literatürdeki bulgulara [10], [3] dayanarak, Şekil 2.2, yazılım mühendisi ve süreç mühendisinin robot sistemine ilişkin rollerini göstermektedir, ancak sunulan kapsamla sınırlı değildir, çünkü mühendislik rolleri organizasyonlar arasında farklılık gösterir. Yazılım geliştiricinin rolü, robot sistemini teslim etme ve entegrasyondan önce geliştirmeden sorumludur. Süreç mühendisinin rolü ise değer yaratmayı mümkün kılan sistemin üretimini planlamaktır. Hem yazılım geliştirici hem de süreç mühendisi için simülasyon ve offline programlamanın rolleri, kullanımını açıklamak için kırmızı renkli oklarla (yazılım geliştirici için) ve turuncu renkli oklarla (süreç mühendisi için) tasvir edilmiştir. Simülasyon ve offline programlama arasındaki farklılaşmanın sonucu, offline programlamanın süreç programlarını tasarlama pratiği olduğu, simülasyonun ise teslimat sürecinden önce ve sonra sistemin incelenmesi için araçlar sağlayan bir araç olduğudur.

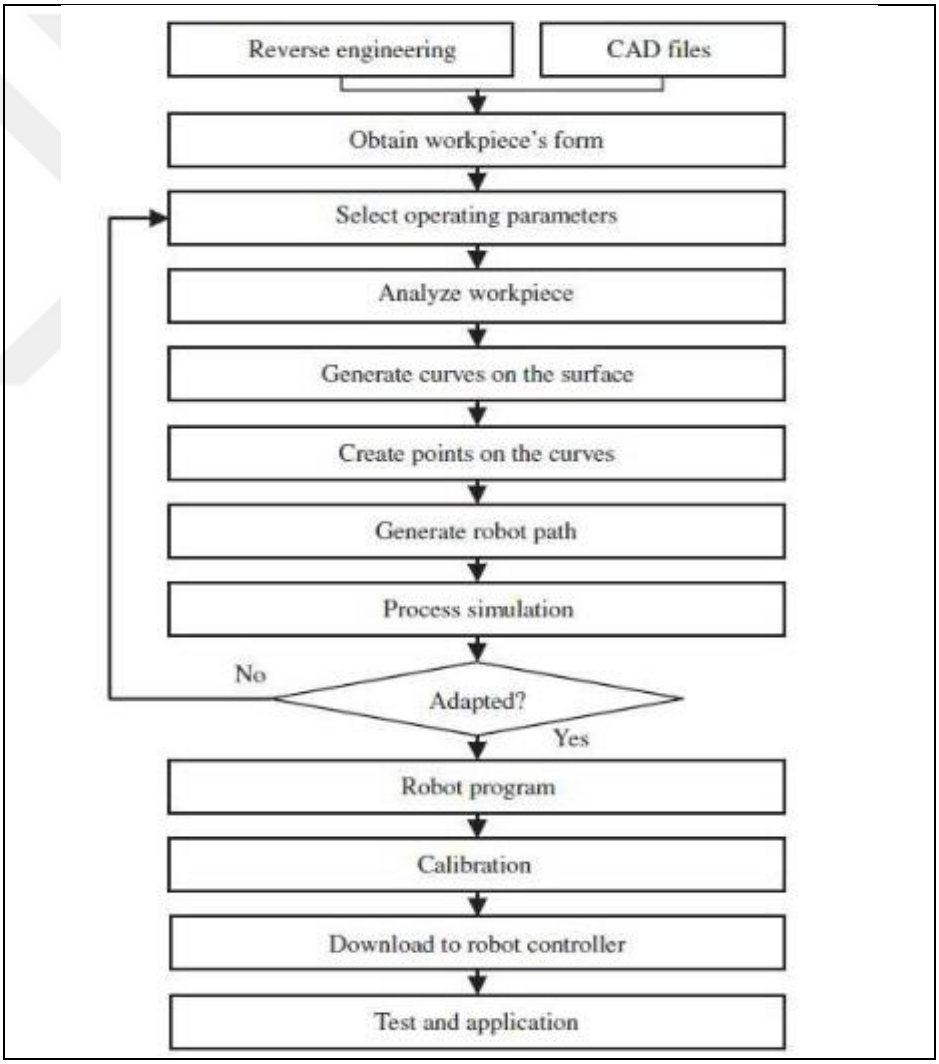


Şekil 2.2: Offline programlama ve simülasyon için kullanım senaryoları

2.2.2. Genel İş Akışı

Bir robot sistemi uygulanırken, offline programlama yöntemini kullanarak genel bir sistemli ve tekrarlayan bir yaklaşım izlenebilir, ancak bu, hem yazılım hem de donanım tasarım personelinin taahhüdünü gerektirir. Offline programlamanın kullanılması gereken en erken aşama, kalıcı tasarım seçimlerinin henüz kilitlemediği kavram aşamasıdır. Bu şekilde, 3D ortamda kaba bir düzeyde doğrulanabilen kavram düzeyi tasarımlar, CAD modelleri şeklinde donanım tasarımları ve hareket kontrolü için yazılımı içerebilir. Offline programlama, robot hareket modeli ile ilgili sorunları test etme imkânı sağlar ve böylece donanım ile ilgili tasarım hatalarını yakalama olasılığı yüksektir [12].

Bir çalışan kavram tasarımı birkaç iterasyon sonrasında uygulandığında, tasarımın geri kalanı başlayabilir, ancak offline programlama modelinin, mekanik tasarım yükseltmeleriyle eşzamanlı olarak paralel olarak yazılım geliştirme ilerledikçe her büyük donanım tasarım değişikliği uygulamasında güncellenmesi önemlidir. ABB RobotStudio, Fanuc Roboguide gibi birçok özel robot programlama aracı, sistem geliştirildikçe doğrulama için kullanılabilen yerel yazılım ortamı sağlar. Robot sürecinin karmaşıklığına, olası parça işleme ve diğer fiziksel temaslarla ilişkili manipülasyonlara bağlı olarak, nesne modelleri ve doğru araç mekanizma hareketi kullanılarak doğrulanmalıdır, çünkü offline programlama modelinin bitmiş fiziksel uygulamadan çok farklı olmaması yardımcı olur [12].



Şekil 2.3: Endüstriyel robot sisteminin entegrasyonu için tipik offline programlama iş akışı

Donanım kurulumunun hazır duruma geldiği noktada, robot sistemi sistem yapılandırmasını beklerken robot programı tamamlanmalı ve programı test etmek ve fiziksel nesneleri robot koordinat sistemiyle bağlamak için planlar hazır olmalıdır. Bu, robot programının sistem devreye alındıktan sonra değişmeyeceği anlamına gelmez, ancak işlem düzeyinde, program bazı konum değişkenlerinde küçük değişikliklerle işlevsel olmalıdır. Bu durum aynı zamanda robot sisteminin offline programlama modelinin artık kullanışlı olmayacağı anlamına gelmez. Sanal model hala, tanımlanmamış ve olası dış nedenlerden dolayı ertelenen geç tasarım değişikliklerini doğrulamak için kullanılabilir. Offline programlama modelini kullanmanın diğer nedenleri, teslim edilecek sistemlerin çoğu durumda teslimat fabrikasında test edildiği ve ilk kabul testlerinden sonra sistem gönderildiği ve nihai durumuna yeniden monte edildiği gerçeğidir. Değişikliklerin sanal ortam kullanılarak her zaman güvenli bir şekilde uygulanabileceği anlamına gelir [12].

Fiziksel nesnelerin yerleştirilmesi genellikle montaj çizimleri ile kesinlikle uyuşmaz. Bu nedenle, sistem için sanal model aynı tasarıma dayandığından, doğruluk uygun seviyede değildir. Offline programlama modelinin kullanılabilirliğini ve doğruluğunu artırmak için sanal 3D ortamın kalibrasyonu gerçekleştirilmelidir. Bu, sanal nesnelerin gerçek dünya nesneleriyle eşleştirilmesini içerir. Robot sistemi kinematik sistemini yerleştirmek veya başka bir doğru yöntem kullanarak gerçekleştirilebilir [12].

2.2.3. Katma Değer

Offline programlamanın sağladığı ana değerlerden biri, teslimat süresine etkisidir. Yalnızca çevrimiçi programlama yöntemlerini kullanarak, hareket yazılımıyla ilgili çalışmalar, robot sistemi sistemin bir parçası olarak kurulduktan sonra başlayabilir. Offline programlama, mekanik tasarım ve yazılım tasarımının paralel olarak yapılmasını sağlayarak planlama süresini kısaltmaya olanak tanır. Bu nedenle, offline programlama, sistem tasarımında olası yeniden tasarımların ve cihaz yeniden siparişlerinin maliyetlerini azaltmada, sistemi simülasyon ortamında test ederek doğrulamada etkilidir. Genel olarak, robot süreçlerinin performansı, offline programlama kullanılarak iyileştirilebilir ve daha verimli hale getirilebilir [10].

Ek olarak, çeşitli avantajlarının yanı sıra, bazı dünya olayları da kuruluşları offline programlama süreçlerini benimsemeye yönlendirebilir. En son böyle bir olay 2020 yılında gerçekleşti, dünya genelinde Corona virüsü pandemisi yayıldı ve ülkelerin çoğu, Covid-19 hastalığının yayılmasını önlemek için kilitlenmeler veya daha hafif kısıtlamalar gibi sosyal davranışlara güvenmek zorunda kaldı. Bu durum, robot programlaması gibi atölye operasyonlarına erişimi kısıtlayan uzaktan çalışma politikalarının artmasına neden oldu, bu da offline programlama araçlarını robot sistemleri teslim eden kuruluşlar için hayati hale getirdi [13].

2.3. Koordinat Sistemleri

Konum ve hareket için matematiksel gösterim çeşitli koordinat sistemlerinde modellenebilir. Kartezyen koordinat sistemi, 3 boyutlu uzayda konumu göstermek için yaygın bir gösterim şeklidir. Ek olarak, dönme hareketi de dönme açılarına bağlı olarak birden fazla gösterim şeklinde tanımlanabilir.

Basitlik açısından, kartezyen koordinat sisteminde bir konum veya nokta bir vektör olarak tanımlanır. Bir manipülatörde, taban koordinat sistemi, aynı zamanda global referans çerçevesi olarak da adlandırılır, manipülatörün taban bağlantısına sabit bir referans noktasıdır. Tüm konum ve dönme ile ilgili tanımlamalar, robotun uç noktasının konumu ve alt koordinat sistemlerinin global referans çerçevesine göre tanımlanır.

2.3.1. Oryantasyon

Oryantasyon, uzamsal konumun döndürme eşdeğeridir. İki kartezyen koordinat çerçevesi arasındaki eğimi anlamak ve göstermek için dönme teorisi gereklidir. Dönme matrisi dışında birden fazla dönme gösterimi vardır ve dönme matrisi, bir nesnenin uzamsal koordinat sistemindeki yönlendirmesini tanımlamak için düşük seviye gösterim olarak anlaşılabilir.

$$^{\alpha}R_b = \begin{vmatrix} R_{x,x} & R_{x,y} & R_{x,z} \\ R_{y,x} & R_{y,y} & R_{y,z} \\ R_{z,x} & R_{z,y} & R_{z,z} \end{vmatrix}$$

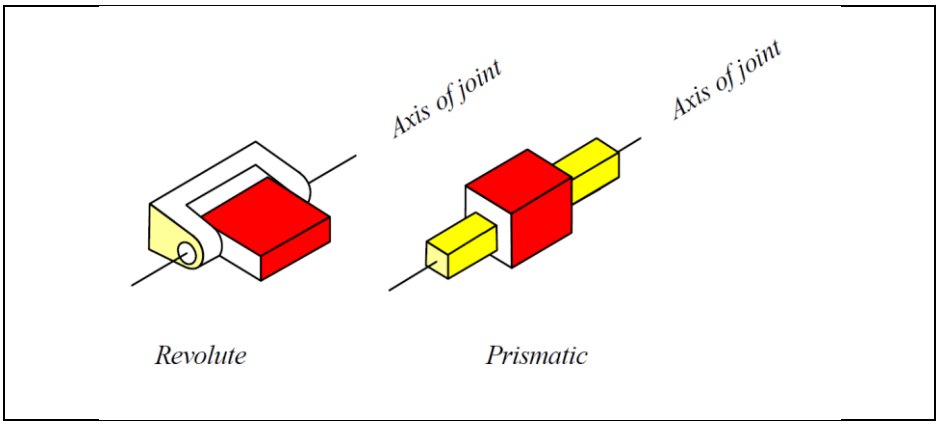
Şekil 2.4: Uzaysal koordinat sisteminde bazı nesnelerin yönelimini tanımlamak için kullanılan düşük düzeyli gösterim

2.3.2. Alt Koordinat Sistemleri

Koordinat sistemleri, pozisyon verilerini ana koordinat sistemi içinde bir gruba bağlamak için alt koordinat sistemine sahip olabilir. Robotik alanında Alt koordinat sistemlerinin tipik kullanımı, fiziksel çevreyi robot taban koordinat sistemiyle ilişkilendirmektir. Sistemde harici mekanizmalar varsa, alt koordinat sistemleri bu mekanizmaların hareketini ve konumunu taban koordinat sistemiyle ilişkilendirmek için kullanılabilir. Alt koordinat sistemleri, makine görüşü temelli uygulamalarda algılanan özellikleri robot taban koordinat sistemine bağlamak için dinamik bir yöntem olarak da kullanılır.

2.4. Robot Kinematik Modeli

Kinematik, manipülatörü matematiksel denklemler kullanarak modellenerek hareket planlaması ve robot tool pozisyon, hız ve ivme hesaplamasıyla ilgili problemleri çözmek için kullanılır. Literatürde genel yaklaşım, linkler ve eklemlerden oluşan bir kinematik zincirini manipülatör olarak tanımlar. Link, manipülatörün rijit bir bileşeni olarak kabul edilir. Tipik bir robot manipülatörü, komşu linklere göre hareket eden birkaç link içerir. Sabit manipülatörlerde ilk link baz link olarak adlandırılır çünkü ek kinematik sistemlere olanak sağlayan bir sınır görevi görür ve son link ise uç etkileyici olarak adlandırılır. Eklemler, bağlantılı linkler arasında sabit bir noktadan veya eksen üzerinden bağlantı sağlar ve bağlı linkler arasında tek veya çoklu serbestlik derecesi (DOF) sağlar. İki basit ve en yaygın eklem türü dairesel (revolute) ve doğrusal (prismatic) eklemlerdir. Dairesel eklemler, linkler arasında dönel hareket izin verirken, doğrusal eklemler linkler arasında translasyonel hareket izin verir [14].



Şekil 2.5: En yaygın iki eklem tipi

Robot manipülatörünün kontrolü ve hareketlendirilmesi göz önüne alındığında, eklemlerin kontrol edilerek istenen konuma hareket ettirilmesi için aktüatörler kullanılır. Prizmatik eklemler translasyonel olduğundan, tercih edilen, hassasiyeti, temizliği ve sessizliği nedeniyle döner elektrik motorlar tarafından hareket ettirilir [14]. Genellikle matematiksel modelleme, kalibrasyon ve mekanik konumlandırma gibi çeşitli nedenlerle robot manipülatörü, tüm eklem pozisyon değerleri sıfır olan bir "0-pozisyonu"na sahiptir. Bu pozisyon nadiren, son efektörün Temel Kartezyen koordinat sistemindeki sıfır konumuyla eşleşir, ancak istisnalar olabilir.

2.4.1. Tool Merkez Noktası

Robotun uç efektörünün uzay koordinat sistemindeki konumunun incelenmesi, son linkin koordinat sisteminin başlangıç linkin koordinat sistemiyle bağlanan bir referans noktasını gerektirir. Bu nokta, araştırılan noktanın son linkin koordinat sisteminde tanımlandığı TCP (Tool Center Point) olarak adlandırılır ve onu başlangıç koordinat sistemiyle ve olası alt koordinat sistemleriyle ilişkilendirir. Manipülatörlerde genellikle birden çok araç veya birden çok ilgi noktasına sahip bir araç olarak uç efektör bulunur. Bu genellikle birden çok tanımlanmış TCP gerektirir, bunlardan biri bir seferde etkin olabilir. Ancak, bağlantılı hareketin bir formu olarak birden çok TCP kullanarak hareket planlamak mümkündür.

2.4.2. İleri Kinematik

Kısacası, ileri kinematik, mevcut eklem değerleri ve linklerin boyutsal çevirilerini kullanarak, TCP'nin temel koordinat sistemine olan konumunu ve yönelimini çözmek için kullanılır. Linkler bir kinematik zincir oluşturduğundan, TCP'nin konumunu parçalar halinde çözmek mümkündür. Tanımlı koordinat sistemine göre TCP'nin konumunu ve yönelimini hesaplama ve modelleme, robotik sistemlerin birçok kullanım durumunda temel bir problemdir. Her robot manipülatör eklemi, link zinciri için bir pozisyonu tanımlamak için karşılık gelen bir değere sahiptir. Bu nedenle, bir robotun pozisyonunu boyutsal birimler, açısal birimler veya encoder değeri veya motor darbeleri gibi donanım özel birimlerde bir eklem değerleri listesi olarak göstermek mümkündür. Poz, bir robot manipülatörü için sabit bir modeldir. Eklem pozisyon değerleri, kinematik hesaplamaların temelidir, ancak bunlar TCP için bir Kartezyen pozisyonu belirtmek için kullanılamaz. Bunun yerine, Kartezyen koordinat sistemleri, 3 boyutlu uzayda uç efektör konumunu temsil etmek için kullanılır, çünkü Kartezyen koordinat sistemi ayrıca link geometrilerinden türetilen mesafeleri de hesaba katar [14], [15].

2.4.2.1. Denavit–Hartenberg Gösterimi

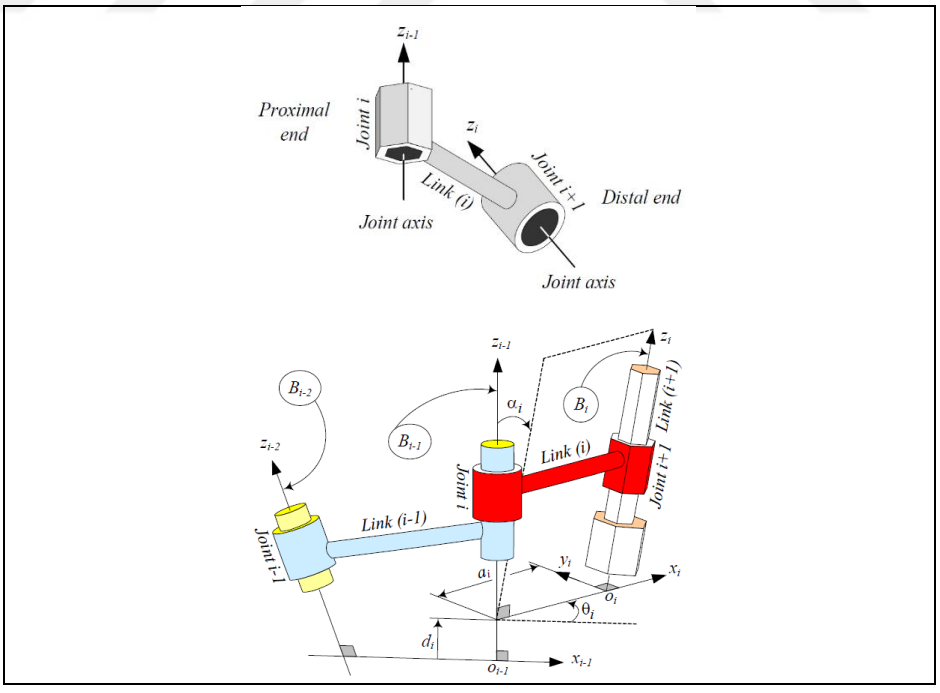
Denavit-Hartenberg (DH) gösterimi, kinematik zincir eklem açılarının pozisyona dönüştürülmesiyle ilgili birden fazla teoriden biridir ve robotikte yaygın olarak kullanılan kinematik teorilerden biridir. DH yöntemi, kinematik zinciri, bir önceki link ve bir sonraki link olmak üzere iki uç noktası olan linklere böler. Linkler arasındaki hareket serbestliği, eklem değerleri olarak tanımlanır. Her linkin bir yerel kartezik koordinat çerçevesi vardır ve bu çerçevede bir sonraki linke olan translasyon ve dönüşüm ofseti tanımlanır. DH yöntemi, linkin yerel başlangıcını, önceki linkle paylaşılan eklem merkezine yerleştirir, böylece çerçevenin z-ekseni pozitif translasyon yönünü tanımlar ve z-ekseni etrafındaki dönme pozitif dönme yönünü tanımlar. İki eksen arasındaki mesafe, her iki eksenin normali boyunca olan bir çizgiyle tanımlanan ek bir kural belirler. Bu iki kural, tek bir link için DH parametrelerini verir [16], [14], [15].

- a: x-eksenleri arasındaki en kısa mesafeyle bir sonraki linke olan translasyon,
- α: yerel x-ekseni etrafındaki dönme ofseti,
- d: z-eksenleri arasındaki en kısa mesafeyle bir sonraki linke olan translasyon,
- θ: yerel z-ekseni etrafındaki dönme ofseti.

Bu parametrelerin temel amacı, manipülatörün her linki için standart bir model tanımlamaktır. Link işlevine bağlı olarak, bu DH parametrelerinden biri, linkler arasında serbestlik derecesine izin vermek için bazı sınırlayıcılarla birlikte isteğe bağlı olarak seçilebilir. Şekil 2.5, DH parametrelerinin birbirine bağlı basit bir çift linki revolute eklemle nasıl modellediğini göstermektedir.

Bir sonraki linkle paylaşılan eklem türüne bağlı olarak, d veya θ veya her ikisi de isteğe bağlı eklem değerleridir. Bu parametreleri listeleyerek, herhangi bir seri manipülatör için bir tablo gösterimi oluşturmak mümkündür.

Tabana göre hem uzamsal hem de dönme ofsetlerini içeren bir konum vektörü oluşturulabilir.



Şekil 2.6: DH parametreleriyle belirtilen link özellikleri

Tablo 2.1: Link sayısı n olan manipülatör için Genel DH tablosu.

Link	a_i	α_i	d_i	θ_i
1	a_1	α_1	d_1	θ_1
2	a_2	α_2	d_2	θ_2
...	...	...	...	...
n	a_n	α_n	d_n	θ_n

Çerçeve, tüm kinematik zincir boyunca dönüşümün hesaplanmasını gerektirir. Her biri için önceki bağlantı koordinat çerçevesi B_{i-1}'den geçerli bağlantı koordinatına tek dönüşüm çerçeve B_i aşağıdaki denklemi uygular.

$$i-1_{Ti} = D_{zi-1,di} * R_{zi-1,\theta i} * D_{xi-1,ai} * R_{xi-1,\alpha i} \tag{2.1}$$

Dönüşüm, z eksenleri arasındaki uzamsal mesafe $D_{zi-1,di}$, x eksenindeki dönme $R_{zi-1,\theta i,}$ x eksenleri arasındaki uzamsal mesafe $D_{xi-1,ai}$ ve z eksenindeki dönme $R_{xi-1,\alpha i}$ içeren çarpımların yapıldığı çoklu matrisi ifade eder. Her bir matris aşağıdaki gibi oluşturulur:

$$D_{z_{i-1,d_i}} = \begin{vmatrix} 1 & 0 & 0 & \alpha_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{2.2}$$

$$R_{x_{i-1,\alpha_i}} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\alpha_i & -sin\alpha_i & 0 \\ 0 & sin\alpha_i & cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{2.3}$$

$$D_{x_{i-1,a_i}} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \alpha_i \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{2.4}$$

$$R_{z_{i-1,\theta_i}} = \begin{vmatrix} cos\theta_i & -sin\theta_i & 0 & 0 \\ sin\theta_i & cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad (2.5)$$

Tek matris formunda, çeviri formda not edilebilir.

$$i-1_{T_i} = \begin{vmatrix} cos\theta_i & -sin\theta_i cos\alpha_i & sin\theta_i sin\alpha_i & \alpha_i cos\theta_i \\ sin\theta_i & cos\theta_i sin\alpha_i & -cos\theta_i sin\alpha_i & \alpha_i sin\theta_i \\ 0 & sin\alpha_i & cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad (2.6)$$

Tek matris formunda, çeviri formda belirtilebilir Tüm kinematik zinciri çözmek, tüm bağlantı dönüşümlerinin matris çarpımı ile mümkündür.

B_1'den B_n'ye. Böylece dönüşüm için ${}^1T_n$ aşağıdaki denklemi uygular, sonuç

DH açı değeriyle tanımlanan pozdaki manipülatörün tam çevirisidir.

Her bağlantıdaki parametreler.

$${}^0T_n = {}^0T_1 * {}^1T_2 * \ldots * {}^{n-1}T_n \quad (2.7)$$

Ortaya çıkan matris, temsil eden RTCP döndürme matrisini çıkarmak için parçalara bölünebilir. TCP konumunu temsil eden TCP yönü ve konum vektörü PTCP.

$${}^0T_n = \begin{vmatrix} R_{x,x} & R_{x,y} & R_{x,z} & P_x \\ R_{y,z} & R_{y,y} & R_{y,z} & P_y \\ R_{z,x} & R_{z,y} & R_{z,z} & P_z \\ 0 & 0 & 0 & 1 \end{vmatrix} = \begin{vmatrix} R_{TCP} & P_{TCP} \\ 0 & 1 \end{vmatrix} \quad (2.8)$$

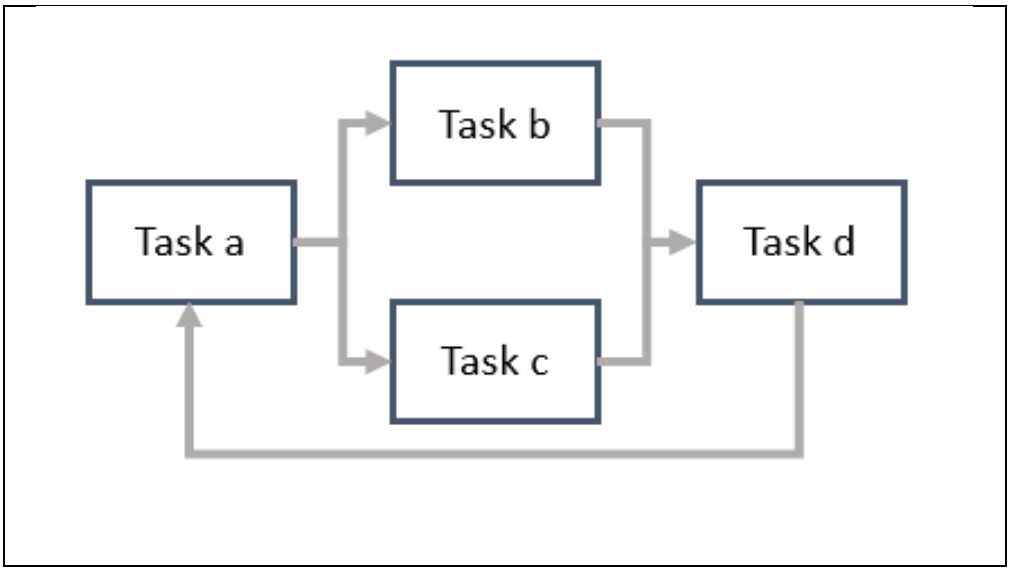
2.4.3. Ters Kinematik

Ters kinematik, ileri kinematiğin tam tersi işlemdir. İleri kinematikte, manipülatörün eklemlerinin değerlerinden yola çıkarak, TCP'nin uzaydaki konumu ve rotasyonu hesaplanırken, ters kinematikte eklemlerin değerleri, TCP'nin uzaydaki konumu ve rotasyonundan hesaplanır. Manipülatörün aynı TCP konumuna birden fazla pozisyonda ulaşabilme olasılığı nedeniyle, uygun koşullar altında tüm bu pozisyonlar doğru eklemler değerlerini verebilir. Ancak, bir konumdan diğerine geçiş yaparken, iki pozisyon birbirine yakın değilse hareket düzgün bir şekilde gerçekleşmeyebilir.

Yalnızca prizmatik eklemlerden oluşan robot geometrilerinde, ters kinematik hesaplaması, dönme gibi yeteneklere sahip 6 serbestlik dereceli (DOF) karmaşık geometrilere kıyasla daha basittir. Birbirine dik üç prizmatik eklem içeren 3 DOF'lu portal geometri manipülatörü örneğinde, TCP'nin konumu, her bir eklemin referans alınan temel koordinat sistemi üzerindeki translasyon değerlerinin önceden tanımlanmış bir ofsetidir. Aynı doğrusallık, ters kinematikte hesaplanabilir, sonuç ise eklemlerin baz koordinat sistemi üzerindeki ters ofsetin translasyon değerleri olacaktır. Bu tez kapsamında, ters kinematik probleminin sadece portal geometriyle sınırlı olması nedeniyle, karmaşık geometrilerin zorlukları sunulmamıştır.

2.4.4. Tasarım Yapısı Matrisi

Tasarım yapısı matrisi (Design Structure Matrix - DSM), bir takım içindeki neden-sonuç ilişkileri veya gereksinimler gibi bağlantıların temsil edilmesi için kullanılan kare matristir. DSM, sistem tasarımı ve süreç geliştirme gibi alanlarda yaygın olarak kullanılan bir araçtır [17], [18], [19]. Her dikey satırda bir tek öğe bulunur ve her öğe için sütunlarda karşılığı vardır. Aynı öğelerin aynı dizinde yer aldığı matris birleştirilerek DSM oluşturulur. Aşağıdaki görüntüler, bir süreçteki görevlerin bağımlılığını ve matrisin yapısını göstermektedir. DSM tablosunda 2.2 numaralı görevler, sütunlarda 1 ile işaretlenen görevlere bağımlıdır. Başlangıçta bir görev kendi üzerinde bekleyemez, bu nedenle diyagonal, görev kimliğiyle işaretlenir [3].



Şekil 2.7: İteratif sürecin genel işlem grafiği

Tablo 2.2: Şekil 2.7'deki bağımlılıkları gösteren DSM tablosu

	task a	task b	task c	task d
task a	a			1
task b	1	b		
task c	1		c	
task d		1		d

DSM'nin kullanımı standartlaştırılmamıştır, bu nedenle sütun ve satırların rolleri değişebilir ve matris oldukça basit olduğundan kullanım alanları geniş kapsamlı olarak uygulanabilir ve özelleştirilebilir. DSM hem küçük sistemlerin hem de birden fazla sistem içeren büyük kümelerin incelenmesi için ölçeklenebilir bir araçtır. Matrisin işlevselliğini genişleten operasyonlar, incelenen sisteme ilişkin daha fazla veri çıkarmak için matrisin yeniden yapılandırılmasında kullanılabilir [20].

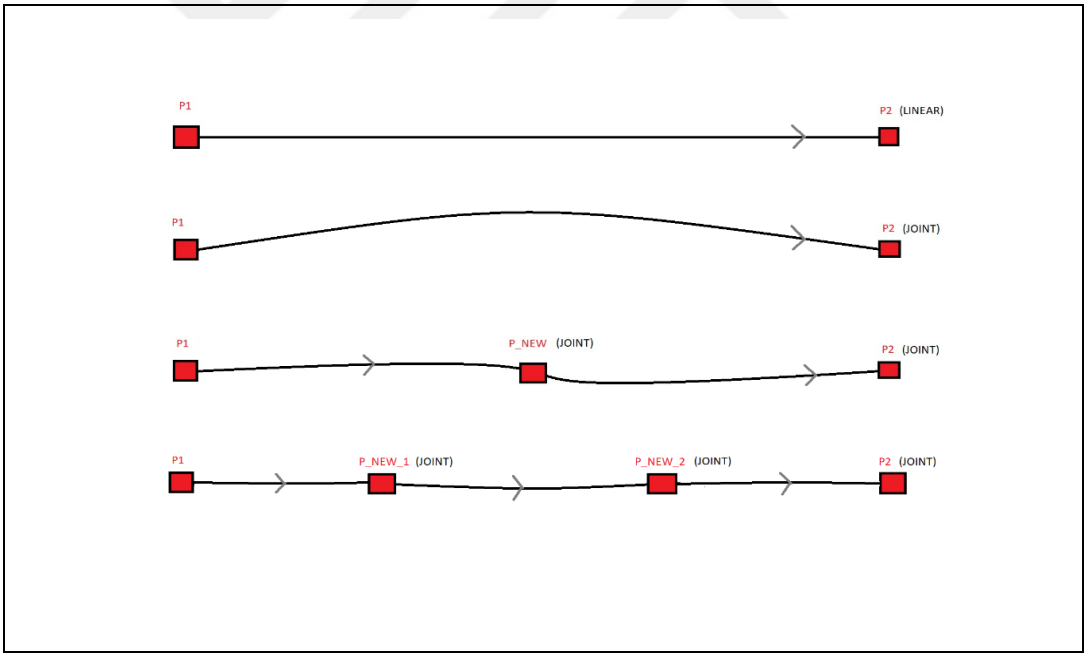
2.5. Ara Nokta Ekleme Metodu

Robotlar, günümüzde birçok endüstriyel, ticari ve tıbbi uygulamada kritik bir rol oynamaktadır. Robotların hassaslık, doğruluk ve verimlilik gereksinimleri, onların belirli rotalarda hareket ederken düzgünlüklerini korumalarını zorunlu kılar. Ancak, bazen doğrudan bir noktadan diğerine gitmek yeterli değildir ve bir rota boyunca

hareketin düzgün ve akıcı olması gerekebilir. Ara nokta ekleme metodu robot yörüngelerinin verimliliğini arttırırken düzgünlüğünü korumak için kullanılır [21].

Ara nokta ekleme metodu, robotun belirli bir rota üzerinde hareket ederken rotanın düzgünlüğünü korumak için kullanılan bir stratejidir. Temel prensip, robotun rotasına eklenen ara noktalar aracılığıyla hareketini düzeltmesini sağlamaktır. Bu sayede, robotun hareketi daha akıcı ve düzgün hale gelir ve sonuç olarak daha iyi bir performans elde edilir.

Ara nokta ekleme metodu, özellikle çok eklemli endüstriyel robotlar için önemlidir. Çok eklemli endüstriyel robotların her bir ekleminin koordinasyonu zor olabilir, ancak ara noktalar, eklem hareketlerini daha uyumlu hale getirir. Sonuç olarak, robotlar daha hassas ve düzgün bir şekilde hareket edebilir [22].



Şekil 2.8: Ara nokta ekleme metodunun uygulanması

Robotların rotalarını daha düzgün ve hassas bir şekilde takip etmelerine yardımcı olan bu strateji, endüstriyel ve ticari robotik uygulamalarda önemli bir rol oynamaktadır. Bu metot, robotlar için daha akıcı, doğru ve verimli hareketlerin sağlanmasına katkı sağlamaktadır [23].

3. DENEYSEL ÇALIŞMALAR

Sistem üç ana bölümde ele alınabilir; bunlardan birincisi simülasyon, ikincisi ölçüm, üçüncüsü ise robot manipülatörü hareketi ve kontrolüdür.

3.1. Sistem Yapısı

3.1.1. Process Simulate

Deneyde Process Simulate simülasyon programı kullanılmıştır. Son yıllarda, robotik simülasyon yazılımları geliştiricileri arasında ürünlerinin kalitesini ve performansını artırmak, daha karmaşık özellikler ve araçlar oluşturmak ve daha kapsamlı, kesin bir dijital modelin yararına olacak şekilde çalışmalar büyük bir ilgi görmüştür. Bu çabalar, Sanayi 4.0 kavramının temelinde durmakta ve bir üretim tesisi geliştirme sürecinde olmazsa olmaz bir araç olan Sanal Devreye Alma'yı (Virtual Commissioning) ortaya çıkarmıştır.

Farklı yazılım çözümleri üzerinde çalışan yazılım geliştiricileri olsa da en gelişmiş uygulamalardan biri Siemens PLM bölümüne ait Tecnomatix paketinin bir parçası olan Process Simulate'dir.

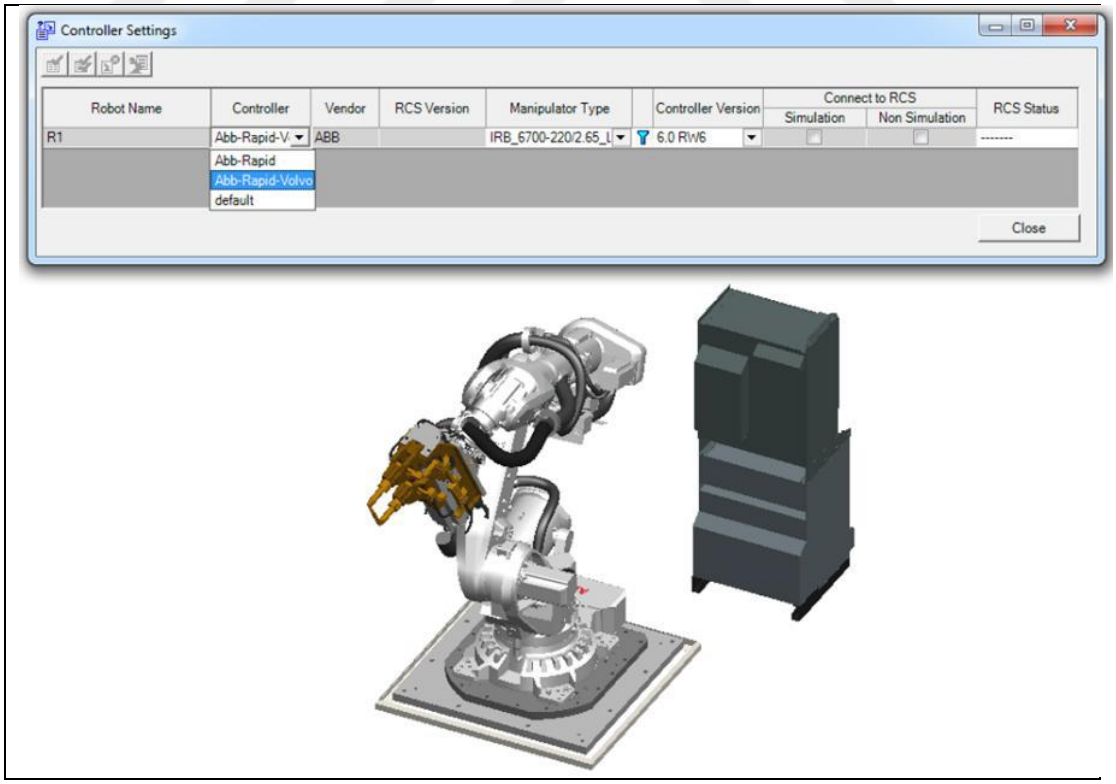
Process Simulate, Robcad yazılım çözümüyle birçok ortak özelliğe sahip olsa da akıllı nesneler oluşturma ve bu nesneleri sinyallerle kontrol etme konusunda çok daha fazla odaklanmaktadır. Bu yetenek, kullanıcının ya bir sanal Siemens PLC üzerinde PLC kodunu çalıştırmasına ve doğrudan Process Simulate ile bağlantı kurmasına ya da mantığı gerçek bir PLC üzerinde (herhangi bir markadan) çalıştırmasına ve Process Simulate ile OPC Sunucusu aracılığıyla bağlantı kurmasına olanak sağlar.

Process Simulate ile uygulanan Sanal Devreye Alma, tamamen kanıtlanmış Robot Sanal Denetleyicileri, PLC programları ile çalışarak gerçek devreye alma sürecindeki maliyetlerin, zamanın ve diğer kaynakların azaltılmasına büyük ölçüde yardımcı olur ve üretim, simülasyon ve kontrol mühendisleri arasında sorunsuz bilgi alışverişine katkı sağlar. Ayrıca, Process Simulate, özellikle Teamcenter ile çok iyi entegre olmuş olan Tecnomatix Yazılım Paketi içinde yer aldığından avantaj sağlar.

Bu önemli özelliklerinden dolayı Process Simulate en kapsamlı yazılım çözümü olmuştur ve dünya genelinde otomotiv endüstrisinde bir endüstri standardı haline gelmiştir.

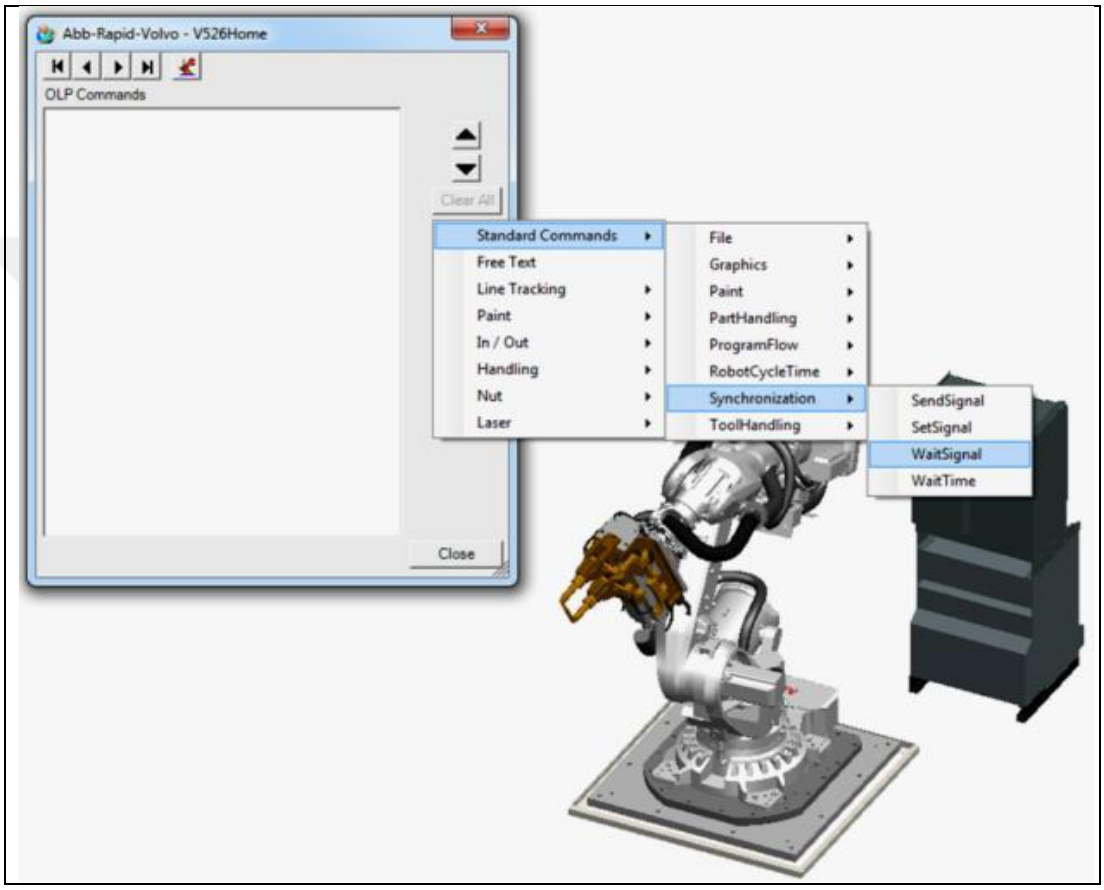
3.1.1.1 Taklit Edilmiş Özel Robot Kontrolcüsü (ESRC)

Belirli bir robot marka ve model için bir Robot Kontrolcüsü'nü taklit eder. Process Simulate, herhangi bir robot olarak tanımlanan herhangi bir kaynağı çalıştırabilen bir Varsayılan Kontrolcüye sahip olmasına rağmen, işlevselliği sınırlıdır ve genel bir araç olması amaçlanmıştır. Bir Robot Üreticisi tarafından kendi ürünü için yapılan bir Kontrolcü kullanarak, kullanıcı robota daha hassas hareket eğrilerine ve gerçek kontrolcünün gerçekleştirebileceği belirli komutlara erişim sağlar. Sonuç olarak, kullanıcı, Robot Kontrolcüsü ile simülasyonu çalıştırdığında, ESRC'ye sahip robotun çok daha hızlı olduğunu fark eder. Bu bilgiye sahip olmak, simülasyonu doğru bir şekilde doğrulamak için doğru Robot Kontrolcüsüne sahip olmanın son derece önemli olduğunu göstermektedir (Şekil 3.1).



Şekil 3.1: Bir robot için kontrolcü ayarlarının yapılması

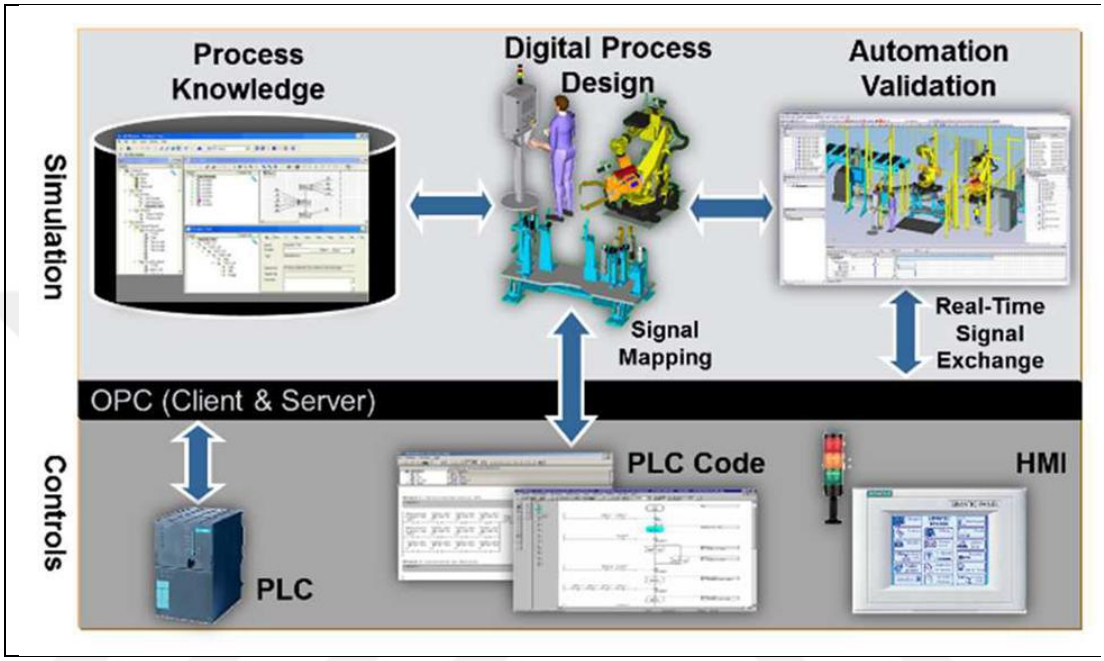
Hassas hareketin yanı sıra, kullanıcı ayrıca OLP (Çevrimdışı Programlama) komutları adı verilen daha gelişmiş talimatlara da erişebilir. Bu komutlar, "XML Özelleştirme" adı verilen bir süreçle her robot için özel olarak oluşturulur. Özelleştirme temel olarak, gerçek robot komutunu belirli kontrolcü dilinde içeren ve aynı zamanda Process Simulate için daha kullanıcı dostu bir görünümde bir temsil içeren bir ".xml" dosyasıdır.



Şekil 3.2: OLP (Çevrimdışı Programlama) komutlarının kullanımı

OLP komutları, robotik işlemin herhangi bir konumuna eklenebilir ve robot o konuma geldikten sonra, kontrolcü önce her komutu geçer ve alt program gerçekleştirildikten sonra robot bir sonraki konuma hareket eder. Bu işlevsellik, daha önce bahsedilen özelleştirme profilleriyle birlikte, kullanıcıya Robotik İşlemler ve OLP Komutlarından gelen bilgileri kullanma ve gerçek kontrolcü ve robot ile çalışacak bir programı dışa aktarma yeteneği verir. Dahası, kullanıcı, fiziksel kontrolcüde çalışan gerçek programları içe aktarabilir.

Process Simulate, tüm OLP komutlarını doğru konumlarda içeren bir işlemi otomatik olarak oluşturur. Bu, bir şirketin mevcut bir fabrikayı dijitalleştirmek istediğinde son derece kullanışlıdır. Bu şekilde, sıfırdan yeniden oluşturma ihtiyacını aşarak, doğru ve kesin robot programlarına, tüm parametreleriyle birlikte sahip olabilirler.

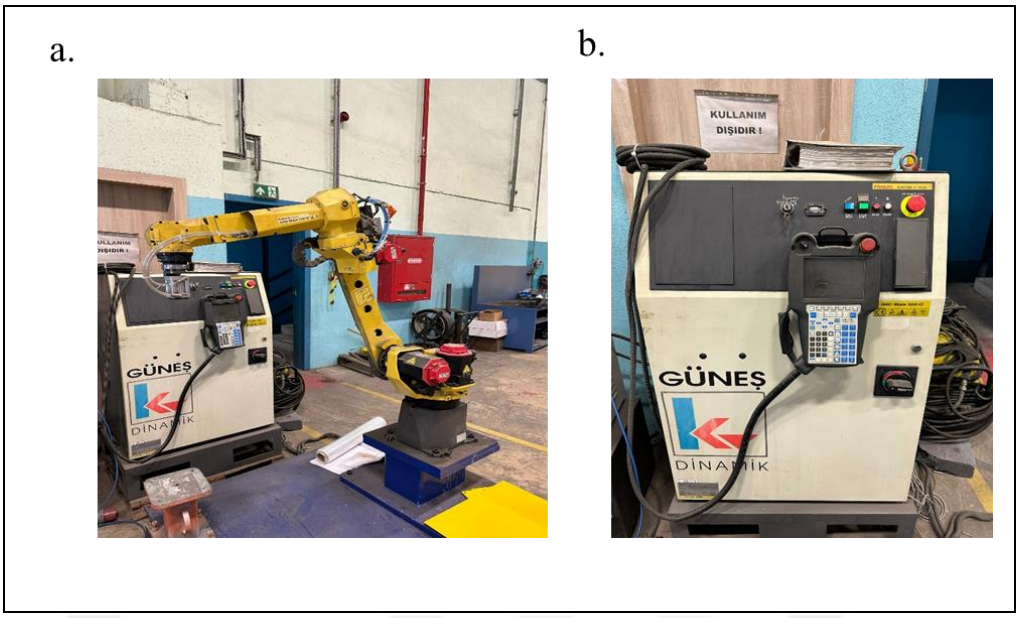


Şekil 3.3: Sanal devreye alma yapılandırması

3.1.2. Manipülatör Hareketi ve Kontrolü

3.1.2.1 Robot Kolu (FANUC ARC Mate 100iC)

Sistemde kullanılan endüstriyel robot Fanuc markasının R-2000iB/125L modelidir (Şekil 3.4). Robot 6 serbestlik derecesine sahiptir. Tasıma kapasitesi 12 kg'dır. Erişim mesafesi 1420 mm'dir.

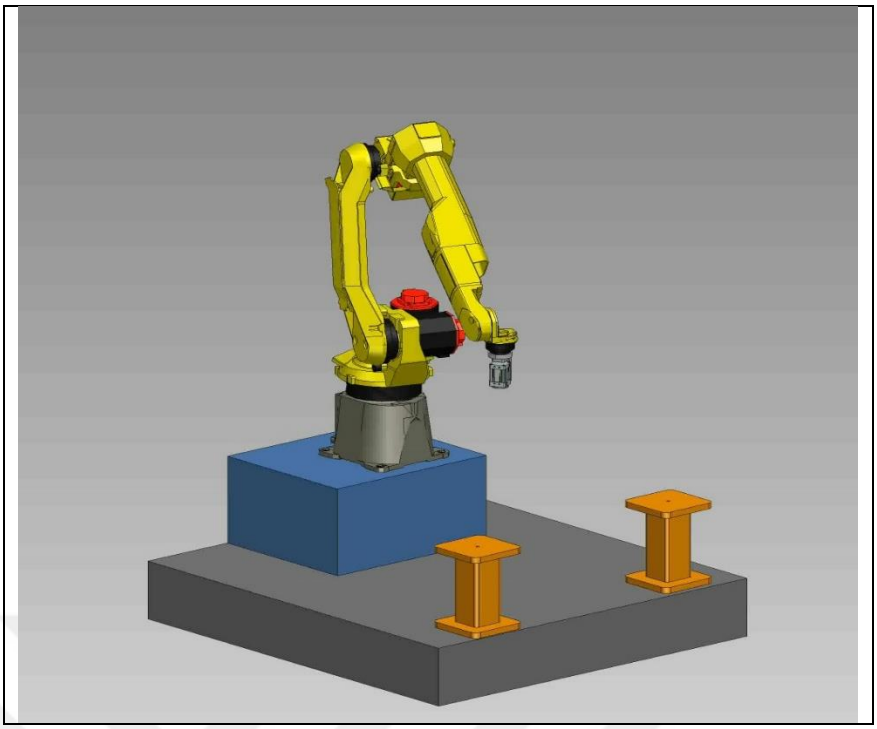


Şekil 3.4: Örnek endüstriyel robot gösterimi. a. Manipülatör kısmı. b. Kontrol kabini

3.2. Deneyin Yapılışı

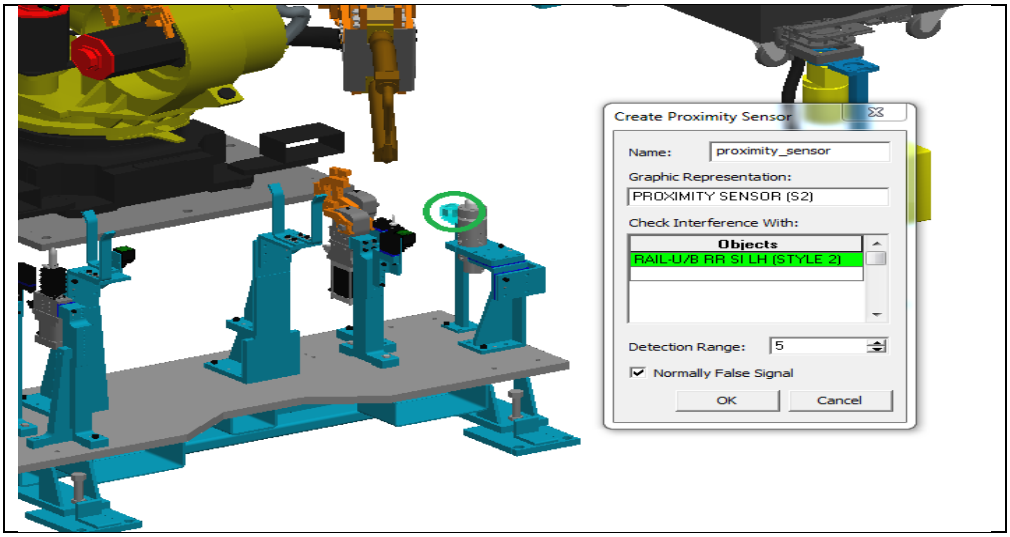
3.2.1. Simülasyon

Process Simulate programını kullanarak uygulamasını gerçekleyeceğimiz robot çalışma alanını simüle etmek için, kullanacağımız ekipmanların CAD datalarının standart olanlarını Robot gibi üretici firmalardan alarak, standart olmayanlar ise CATIA programında çizilerek elde edildi. Process simulate programı "JT" formatında dosya kabul ettiği için çevirme programı yardımıyla datalar çevrildi. Oluşturduğumuz simülasyon projesine datalar yüklenerek hat simüle edildi (Şekil 3.5).



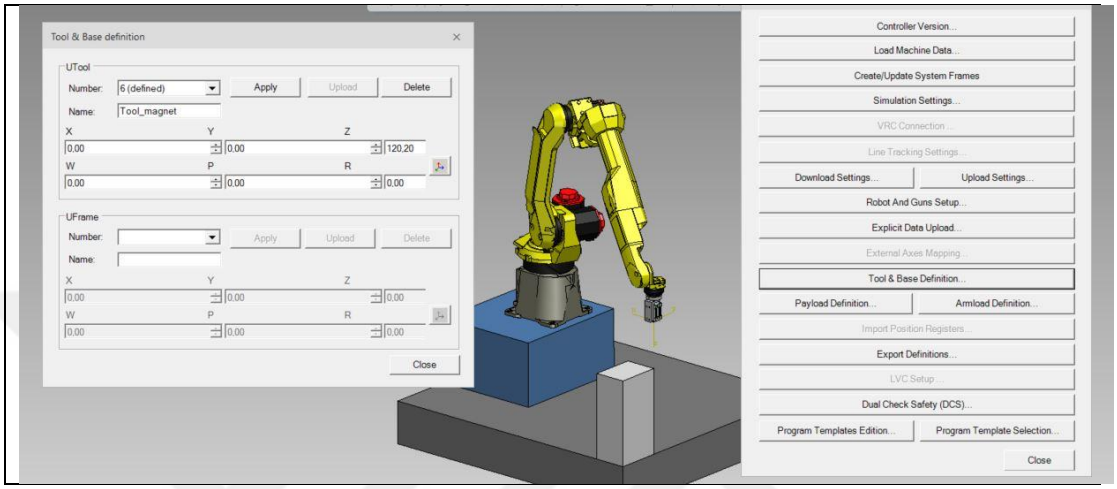
Şekil 3.5: Process Simulate projesi genel hat görüntüsü

Simülasyon programlarına yüklenen CAD datalarının saha uygulamaları kullanımında farklı görevleri mevcuttur. Bu görevler program içerisinde fikstür, parça, sensör gibi tanımlamalarla girildi ve yüklemiş olduğumuz ekipmanlar programda belirtildi (Şekil 3.6).



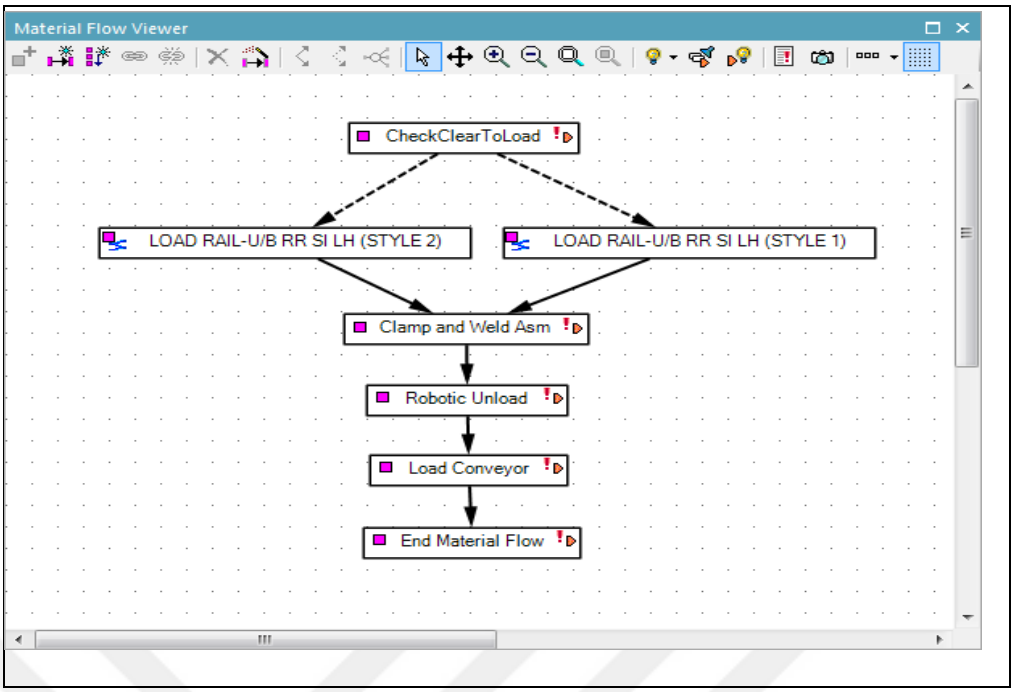
Şekil 3.6: Process Simulate programında örnek yakınlık sensörü tanımlama.

Robotların hareketlerindeki Servo motorların kinematik hesaplamaları oldukça önemlidir ve bu hesaplamalar, robotun marka ve modeline göre değişen bir yazılım tarafından kontrol edilir. Bu nedenle, doğru verilerin girildiği çalışma alanı ve robot detayları önemlidir. Hareketlerin simülasyonunu gerçekleştiren projede, robot özellikleri ayarlandı (Şekil 3.7).



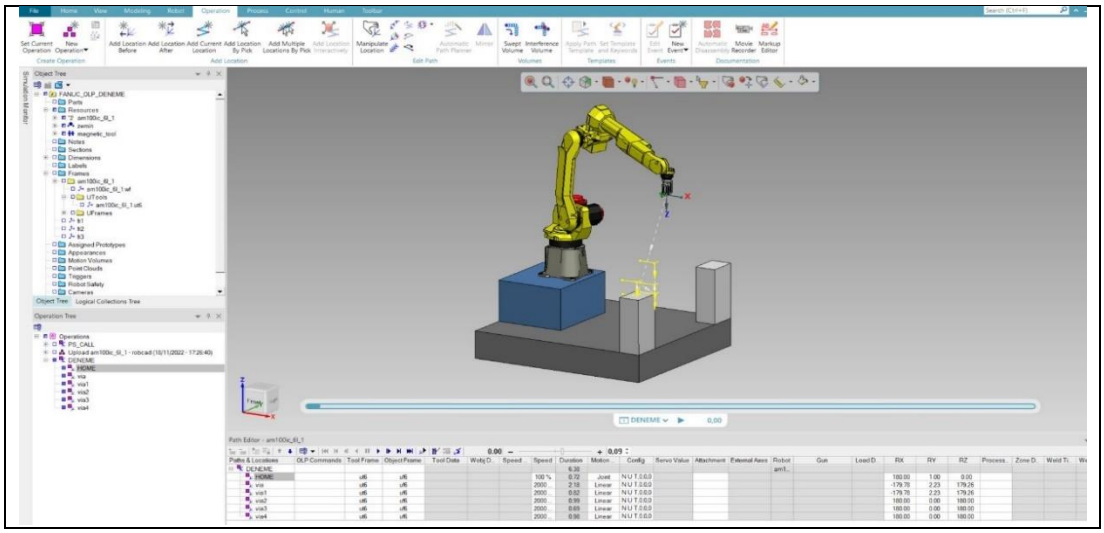
Şekil 3.7: Process Simulate programında tool base tanımlama

Sahada uygulanacak olan hareketler öncelikle simülasyon programında gerçekleştirildi. Her hareketin gerçeklemesi sahada meydana gelecek durumlara göre oluşturuldu ve bunun için simülasyon üzerinde akış şeması oluşturuldu (Şekil 3.8). Günümüzde gelirin etkilendiği bir ortamda her saniyenin değeri büyüktür. Bu nedenle, robot kurulumunu gerçekleştiren firma, süreç akışında iyileştirmeler yaparak, öngördüğü çevrim süresini (cycle time) yakalamayı hedefler ve bunun kontrolünü sağlar.



Şekil 3.8: Process Simulate programında akış şeması

Deneyimiz için tasarlanan sistemde, robotun önünde 750 mm aralıkla yerleştirilmiş mevcut 2 adet ayak bulunmaktadır. Hareketleri kısıtlayacak herhangi bir engel bulunmamaktadır ve robotun parçalar üzerindeki hareket yörüngesini gerçekleştirirken, önündeki ayaklara kolayca erişim sağlandı. Bu durum, simülasyon programında tam olarak simüle edildi. Oluşturduğumuz yörüngenin, robot için yedek dosyası alındı (Şekil 3.9).



Şekil 3.9: Robot yörüngesinin oluşturulması

3.2.2. Ölçüm

Ölçüm iki yöntem kullanılarak yapılabilir, bu yöntemler; ölçüm cihazı kullanımı aracılığıyla ve yetkin robot programcısı yöntemidir. Bu aşamada kullanıcı karar vermelidir. Her iki yöntemde de sonuç hemen hemen aynı olmasına rağmen hassasiyette farklılıklar olabilir. Bu durum ekipman, saha ortamı, zaman, yetkin personel gibi koşullara göre alınması gereken karardır.

3.2.2.1 Ölçüm Cihaz ve Programı

Bu cihazlar pahalı ve hassasiyeti yüksek cihazlardır. Hexagon ölçüm cihazlarının bazı örneklerini bulabilirsiniz:

- Koordinat Ölçüm Makineleri (CMM): Hexagon, 3 boyutlu parça ölçümlerini gerçekleştirmek için hassas CMM'ler üretmektedir. Bu cihazlar, parçaların geometrik özelliklerini ve toleranslarını kontrol etmek için kullanılır.

- Taşınabilir Ölçüm Cihazları: Hexagon, taşınabilir el tipi lazer tarayıcılar ve optik takip sistemleri gibi portatif ölçüm cihazları da üretmektedir. Bu cihazlar, büyük parçaların veya karmaşık geometrilere sahip parçaların saha ölçümlerinde kullanılır.

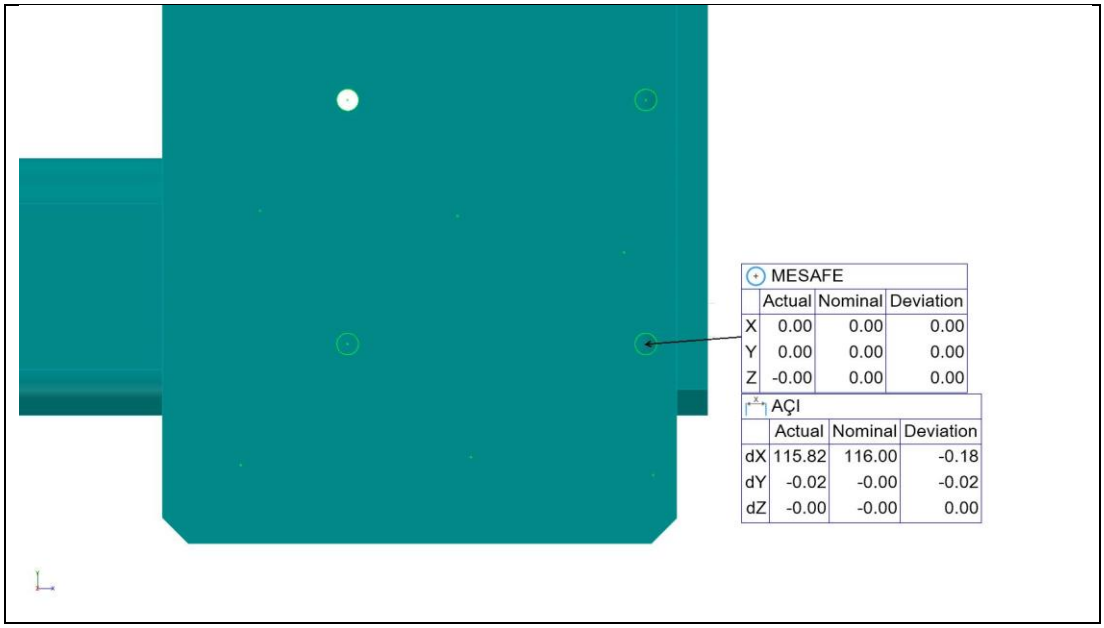
- Optik Ölçüm Sistemleri: Hexagon, optik sensörler ve kameralar kullanarak hızlı ve doğru optik ölçümler gerçekleştiren sistemler üretir. Bu sistemler, yüzey taraması, profil ölçümü ve çok noktalı ölçüm gibi uygulamalarda kullanılabilir.

- Diğer Ölçüm Cihazları: Hexagon, ölçüm gereksinimlerine göre çeşitli diğer cihazlar da üretmektedir. Örneğin, şekil ve yüzey analiz sistemleri, diş ölçüm cihazları, hava yastığı yataklı ölçüm cihazları gibi çeşitli ürünler mevcuttur.

Hexagon ölçüm cihazları genellikle yüksek hassasiyet ve doğruluk sağlar. Bunlar, otomotiv, havacılık, savunma, medikal ve genel imalat gibi birçok endüstriyel sektörde yaygın olarak kullanılan ölçüm araçlarıdır. Her bir cihazın özellikleri ve yetenekleri ürüne bağlı olarak değişiklik gösterebilir (Şekil 3.10).

Şekil 3.10: Örnek ölçüm cihazları

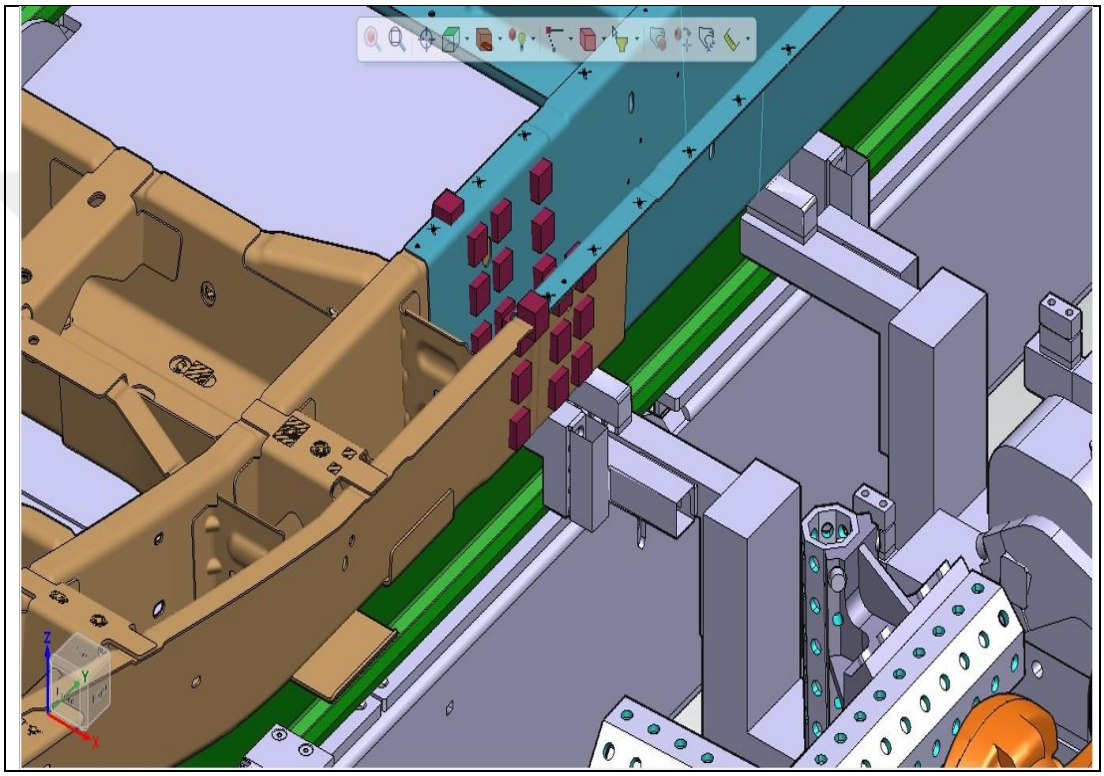
Deneyde Koordinat Ölçüm Makineleri (CMM) tipi ölçüm cihazı olan LEICA-AT960 model 3 Boyutlu Ölçüm Cihazı kullanılarak verilerde alındı (Şekil 3.11). Hassasiyet oranı yüksek cihazlar olmasına rağmen dezavantajı bulunmaktadır. Bu cihazların kurulumu ve kalibrasyonu vakit almaktadır ve saha içerisindeki sıcaklık değeri gibi çevresel faktörler bile önemlidir. Burada seçim yaparken zaman ve ölçümün çok hassas olması isteği bu yöntemi mantıklı kılmaktadır.



Şekil 3.11: Metrolog programı ölçüm değerleri

3.2.2.2 Yetkin Robot Programcısı

Bu yöntem daha basit fakat hassasiyeti düşük yöntemdir. Hat kurulmadan önce, işlem yapılacak parçanın işlem noktaları tasarımcı ve robot programcısı ile paylaşıldı (Şekil 3.12). Robotun ucunda, işlem için bir araç (Tool) veya parçanın yerleştirileceği bir pnömatik veya hidrolik sistem bulunabilmektedir. Fikstür, bu parça ve noktalara göre seçilmiştir. Bu yöntem basit olmakla birlikte hassasiyeti düşük bir yöntemdir.



Şekil 3.12: Örnek parça kaynak noktası (Punta) dağılımı

Offline back-up yörüngesi Robot programcısı tarafından sahaya kurulmuş olan hücredeki robota yüklendi, yükleme işlemi bittikten sonra robot hızı düşük seviyede yörünge çalıştırıldı (Şekil 3.13). Bu işlem sırasında birden fazla robot çalıştırılabilmektedir. Program ilk defa çalıştırılarak deneme yapıldığı için güvenlik önlemleri, çevre kontrolü ve dikkat çok önemlidir.

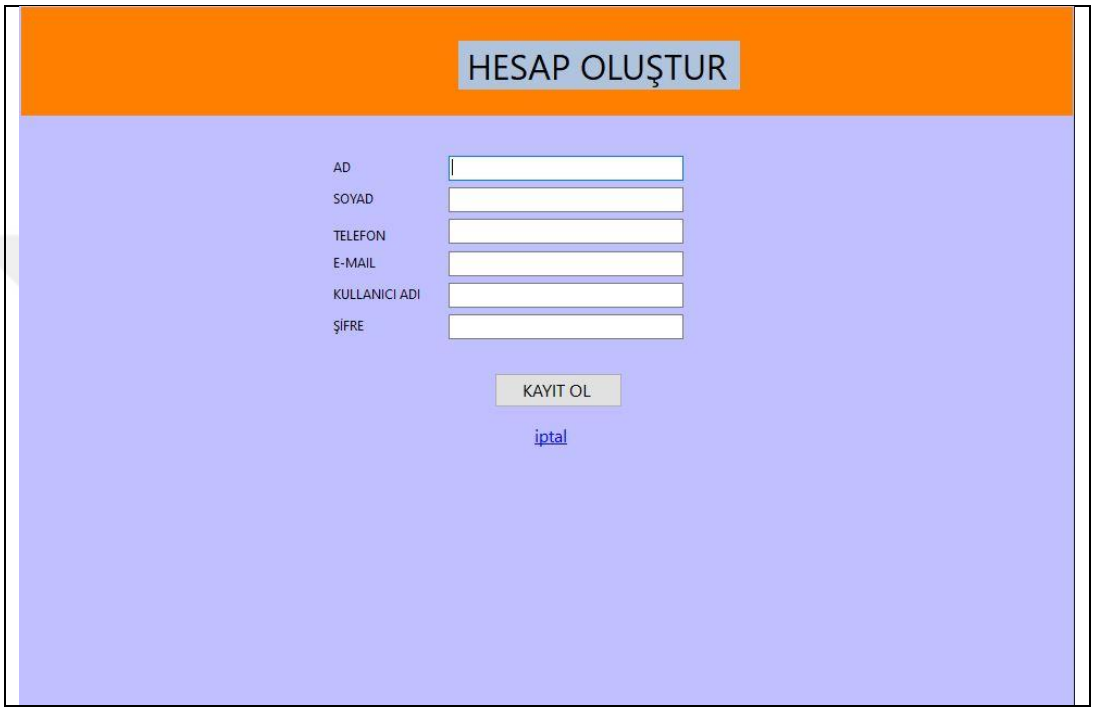
Şekil 3.13: Örnek offline back-up robot yörünge kontrolü

Simülasyon programında, robotun Tool'u tam istenen noktaya gelirken, teoride her şeyin tam yerinde olması gerektiği durumda bile kurulum veya minimal kaymalar nedeniyle parça üzerindeki hedef noktadan sapmalar oluşabilmektedir (Şekil 3.14). Robot programcısı, robotu hedef noktaya getirir ve ardından yeni nokta ile offline yedek dosyası içindeki koordinat farkını kaydederek genel sapmayı belirler. Bu sayede, sistemdeki kaymaları takip edilir ve gerekli düzeltmeleri yapılır.

Şekil 3.14: Deney aşamasında oluşan kaçıklık

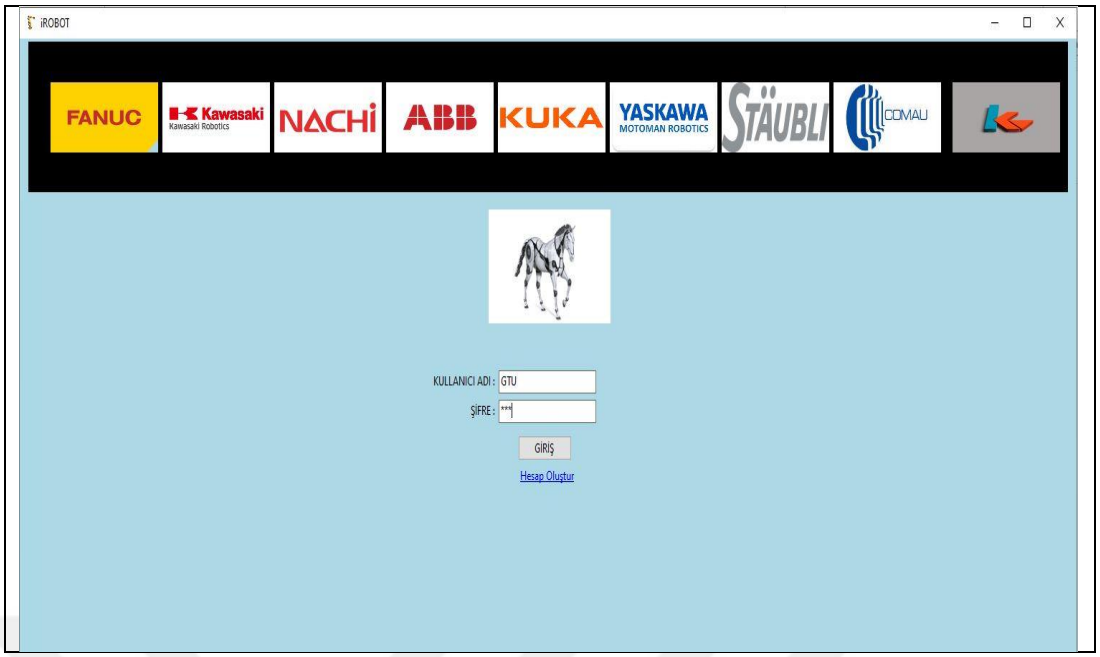
3.2.3. IROBOT (Visual Studio/C#)

Irobot bu tezin amacını gerçekleştirmek için Visual Studio programında C# yazılım dilinde yazılmış olan bir programdır. Programda öncelikle kayıt ekranı oluşturuldu (Şekil 3.15) ve kullanacak kişilerin verileri SQL Database kullanılarak kaydedildi.



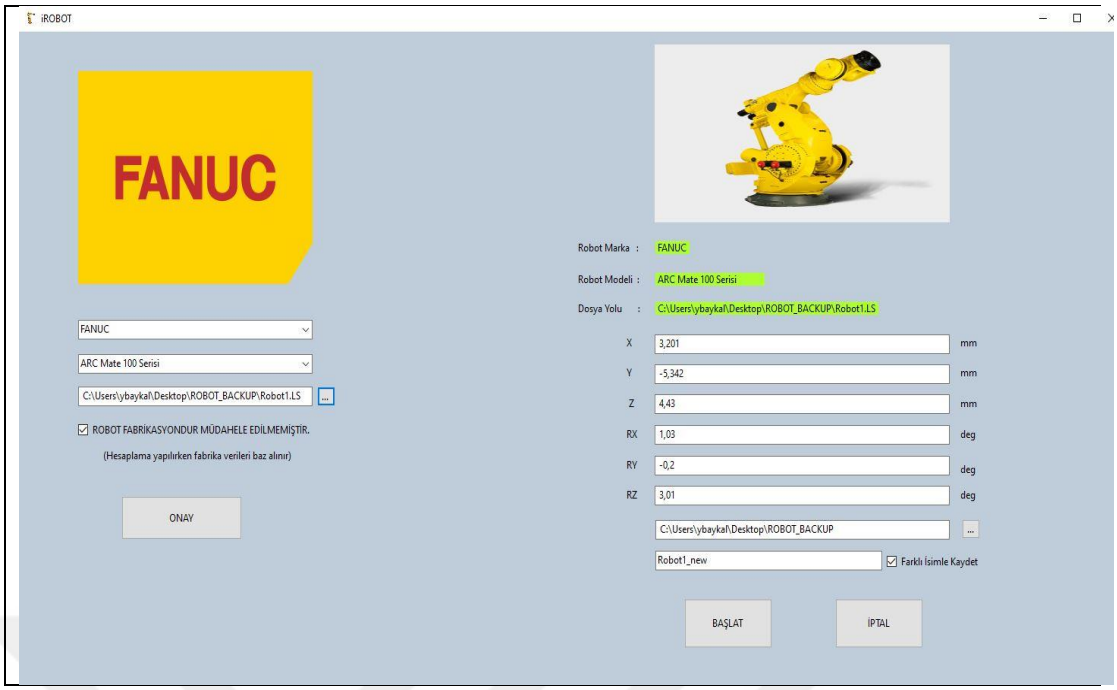
Şekil 3.15: IROBOT hesap oluşturma sayfası

Program kişilerin çalışmalarını özelleştirebilmesi için kullanıcı adı ve şifre ile girilmektedir. Program kurulumu gerçekleştirilip açıldığı zaman Ana Sayfa ekranı açılmaktadır (Şekil 3.16). Program uyumlu markaların logoları Ana ekranda bulunmaktadır. Saha uygulamalarında kullanım oranı yüksek markalar çalışmada öncelik olarak seçildi. Bu düşünce doğrultusunda programın kullanım popülaritesinin artması da amaçlandı. Kullanıcı bu sayfadan kayıt ekranına ulaşabilmekte ve eğer kayıtlı ise yine bu ekrandan giriş yaparak işlemlerini gerçekleştirebilmektedir.



Şekil 3.16: IROBOT ana sayfa

Programın kullanılabilmesi için, IROBOT programının desteklediği robot markalarının yedek (back-up) dosyalarına sahip olunması gerekmektedir. Sahada, hat devreye alınmadan önce offline olarak optimize edilen robot yörüngesi, en hızlı şekilde optimize edilir. Bu optimizasyon için, ölçüm sonuçlarından elde edilen yörünge noktalarının sapmaları, program ara yüzünde girilmelidir (Şekil 3.17). Böylece, yörüngeyi sahada optimize etmek ve hattı verimli bir şekilde devreye almak mümkün olur.

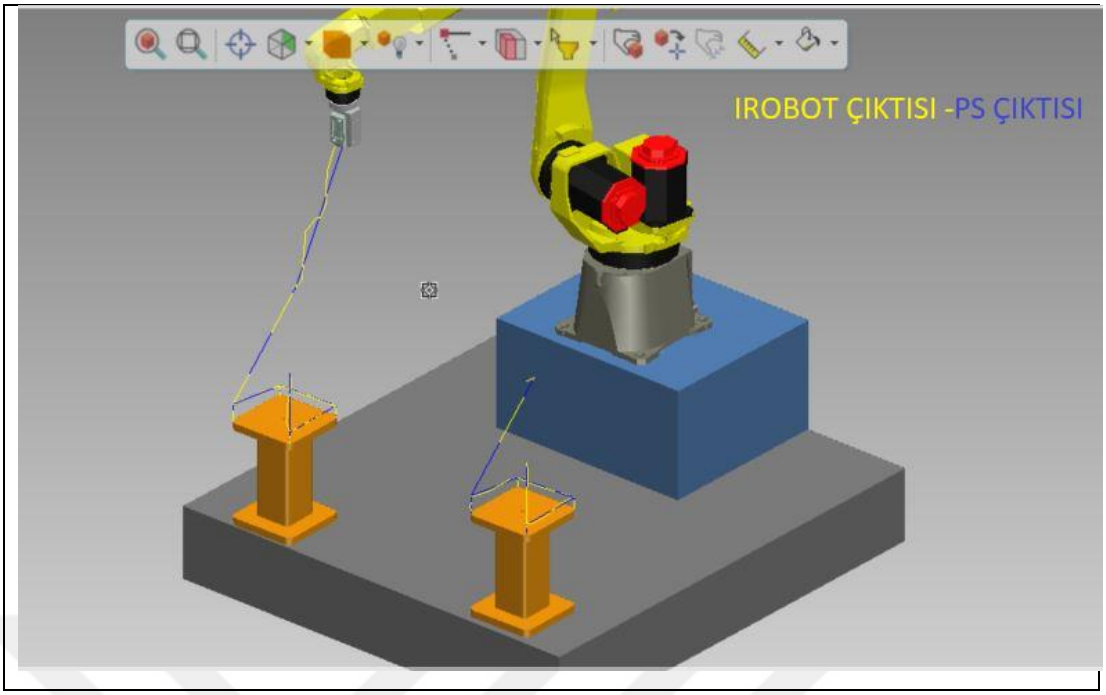


Şekil 3.17: IROBOT veri giriş sayfası

Bu aşamada ölçümün hassasiyetine göre çıkacak back-up dosyası ve bu dosyanın sahaya uygulamasında program yapısı değişiklik göstermektedir, bu nedenle çıkacak olan yeni back-up dosyasının uygulama aşamasında dikkatli olmak çok önemlidir. Teorik olarak yeni back-up dosyası kullanılabilir yeni robot yörüngesi olmaktadır.

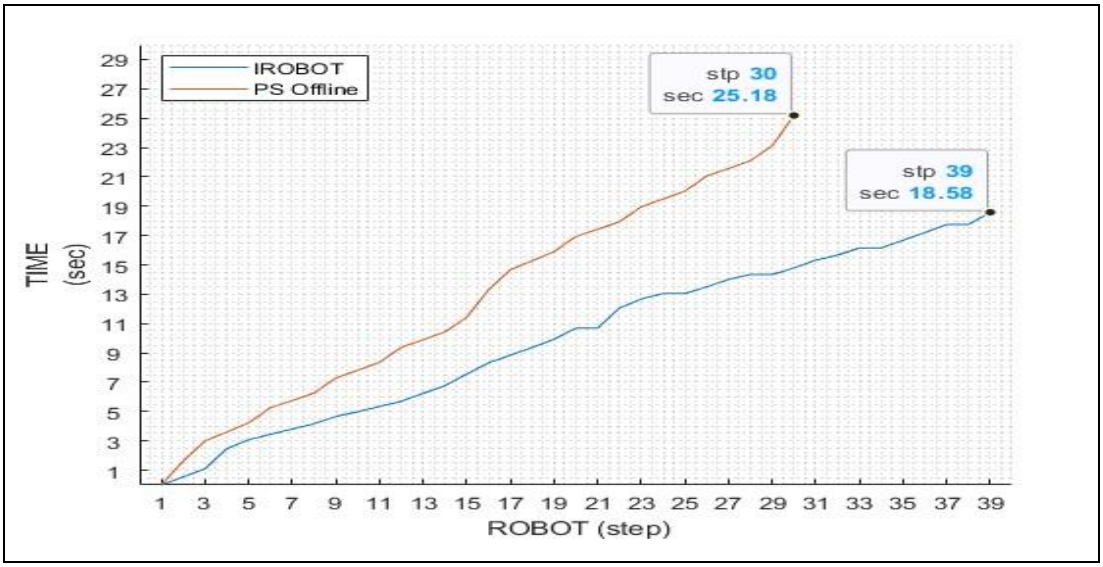
3.2.4. Deneysel Sonuçlar

Deneyimizin amacı doğrultusunda robotlu hücreler simüle edildi. Kullanılan Process Simulate programında offline back-up dosyası alındı. Tez çalışmasında yazılımı yapılan IROBOT programına yüklenerek yörünge optimizasyonu çıktısı alındı. Alınan çıktı tekrar simülasyon programında çarpma ve çevrim zamanı kontrolü yapıldı. Program optimizasyon işlemi yaparken yörünge içerisinde belirtilen işlem noktalarına müdahale yapmamaktadır bu da işlem noktaların belirtildiği gibi istenen yere gitmesini engellemeden optimizasyon işlemini sağladı (Şekil 3.18).



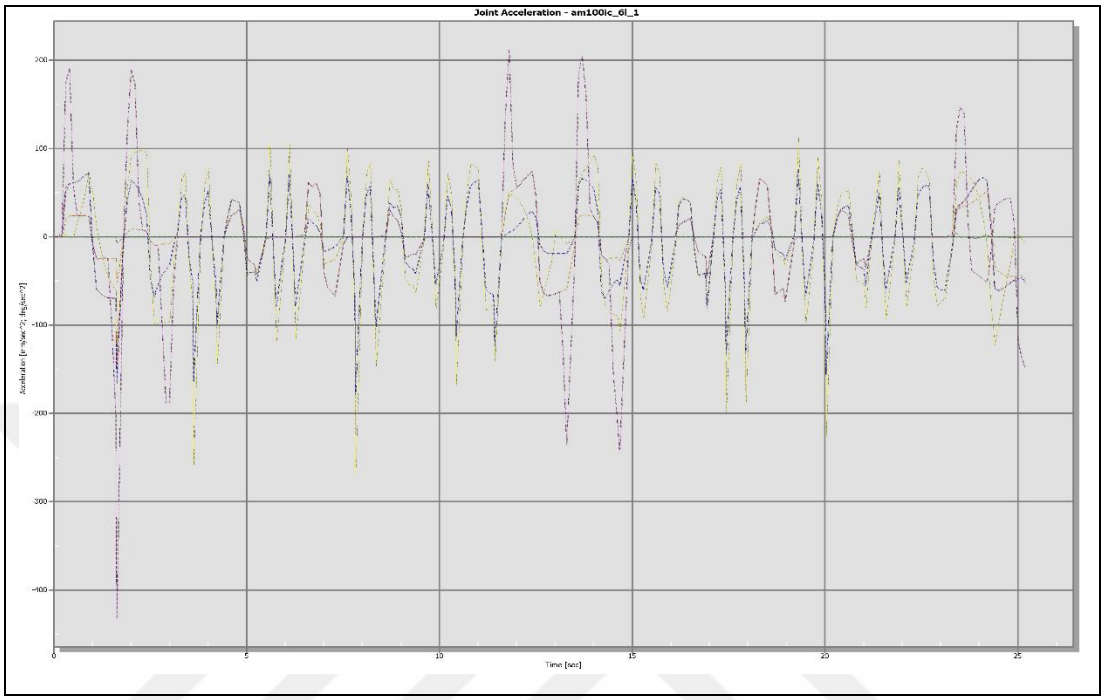
Şekil 3.18: PS yörüngesi ve IROBOT yörüngesinin karşılaştırılması

Yeni yörünge ile ilk yörünge arasında program optimizasyonu sonrası %30'dan fazla verim artışı olduğu gözlendi (Şekil 3.19). Bu yörüngenin ve nokta sayılarının fazlalığına göre değişim gösterebilir.

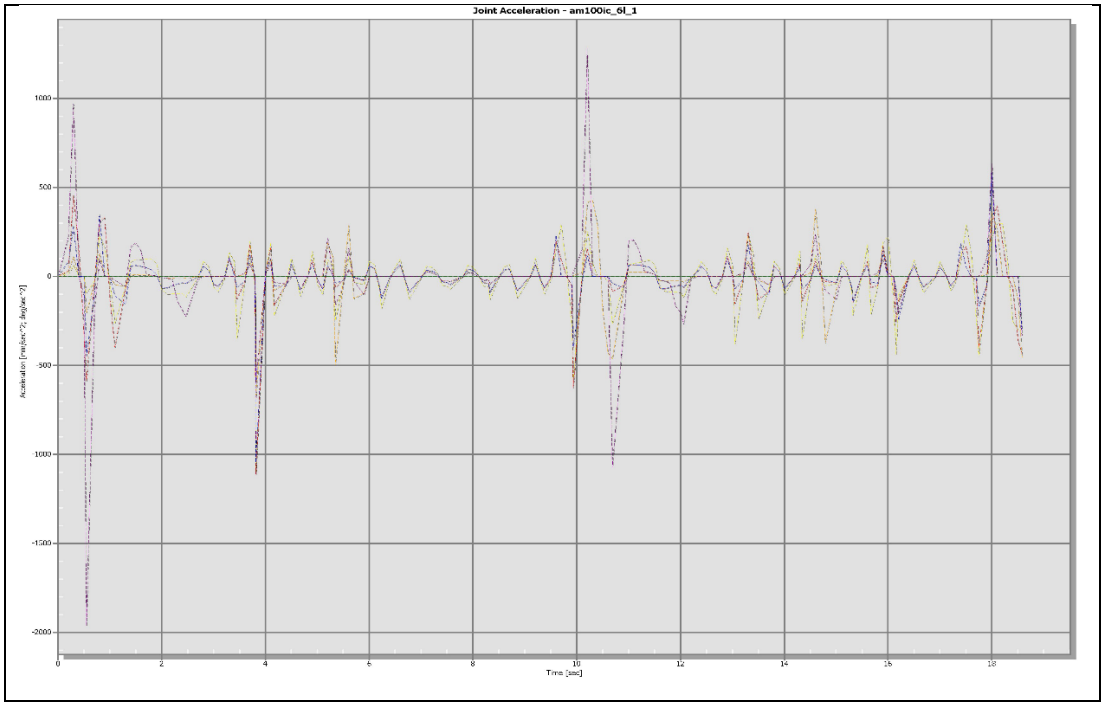


Şekil 3.19: PS yörünge çıktısı ve IROBOT yörünge çıktısı çevrim zamanının karşılaştırması grafiği

Robotun optimizasyon işlemi gerçekleştirilmeyen PS yörüngesi eksen hızlanması normal (Şekil 3. 20). Optimizasyon işlemi yapılan IROBOT yörüngesi hızlanması daha hızlıdır (Şekil 3. 21).

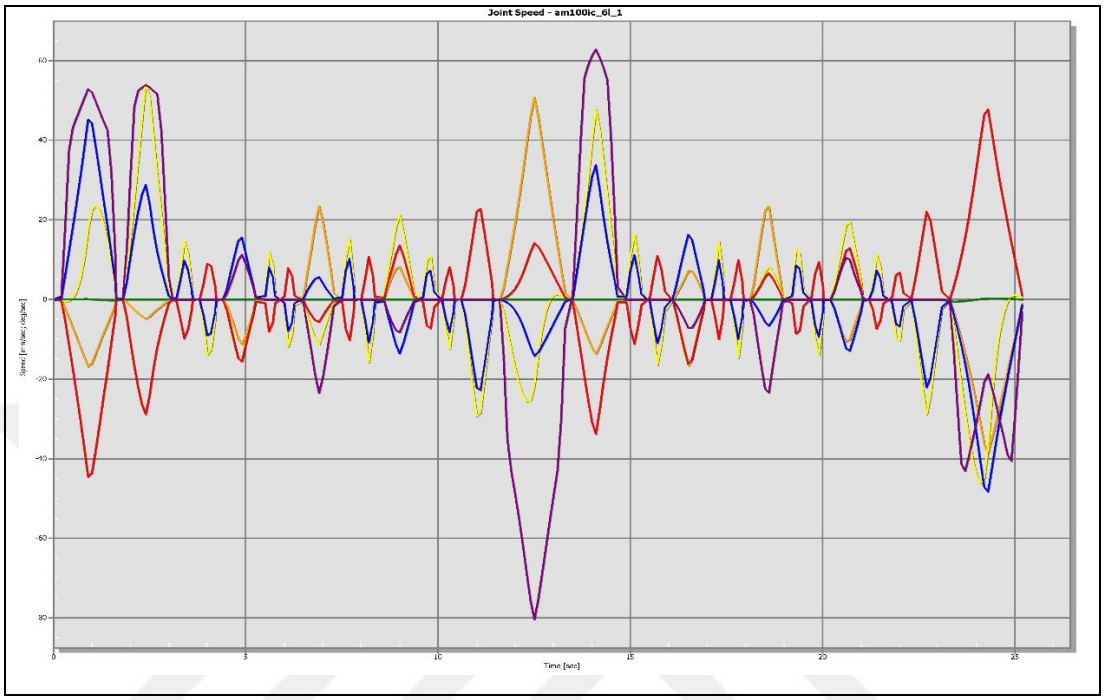


Şekil 3.20: PS yörünge eksen hızlanmaları grafiği

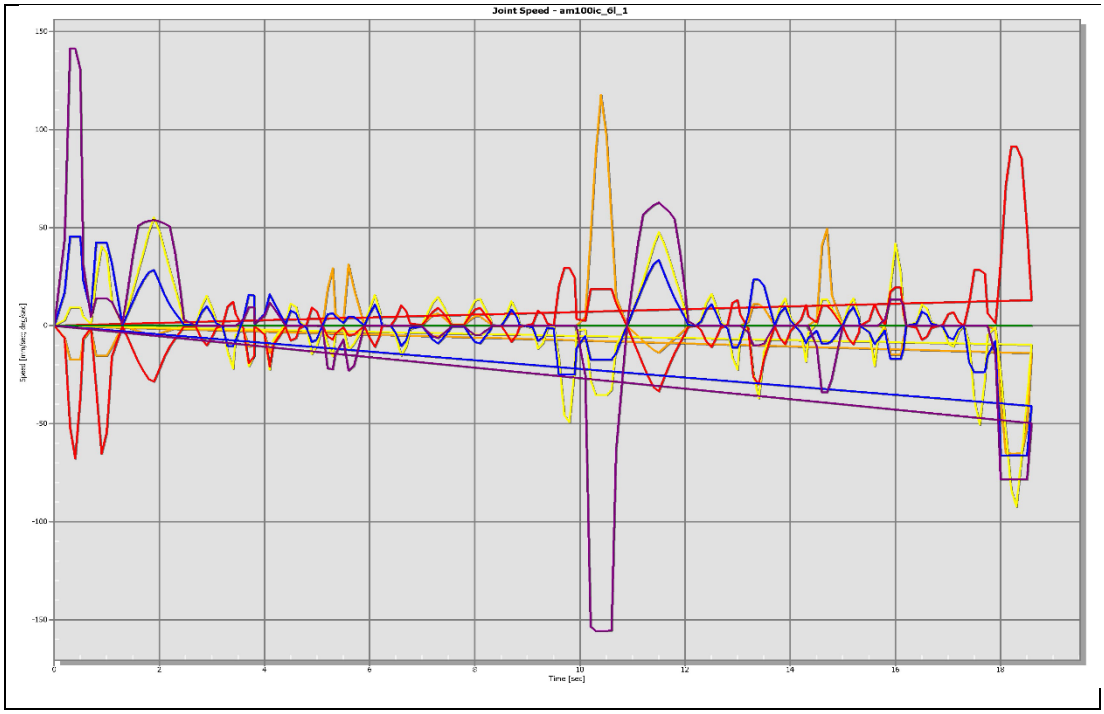


Şekil 3.21: IROBOT yörünge eksen hızlanmaları grafiği

Robotun optimizasyon işlemi gerçekleştirilmeyen PS yörüngesi eksen hızları normal (Şekil 3.22). Optimizasyon işlemi yapılan IROBOT yörüngesi eksenleri daha hızlıdır (Şekil 3. 23).

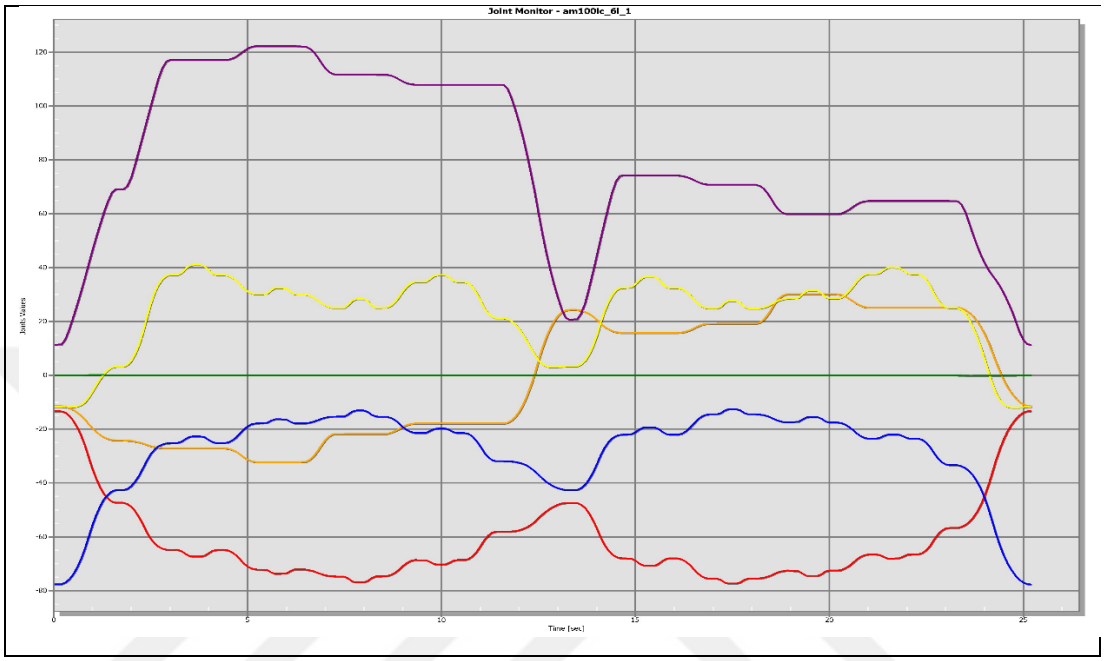


Şekil 3.22: PS yörünge eksen hız grafiği

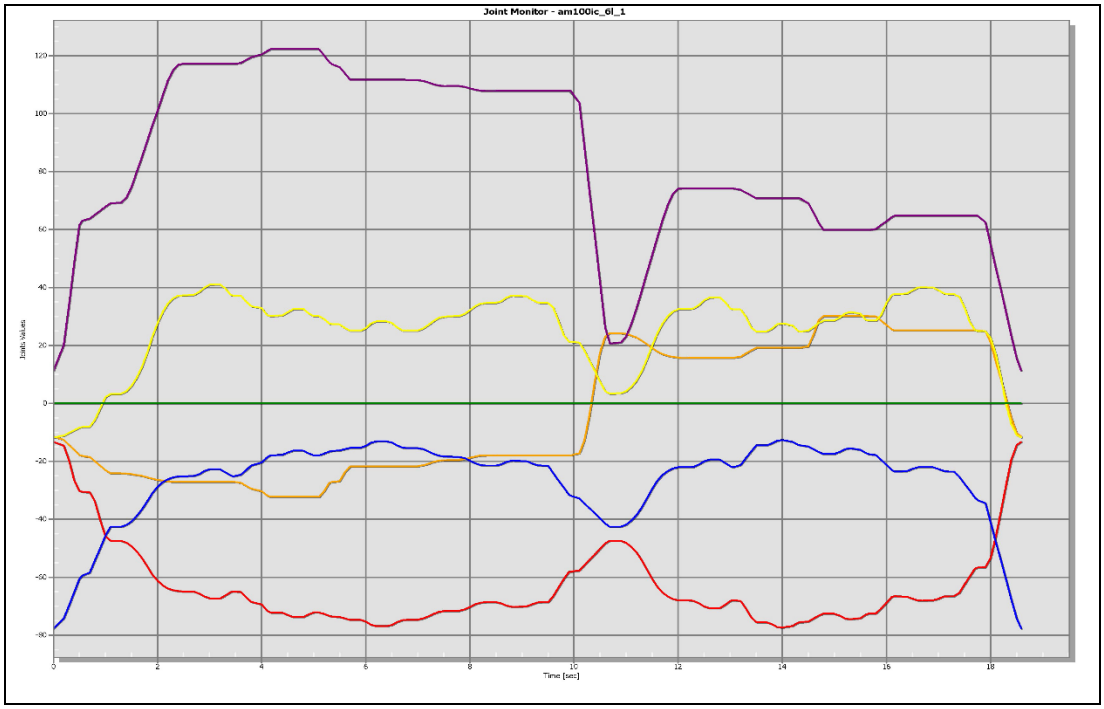


Şekil 3.23: IROBOT yörünge eksen hız grafiği

Robotun optimizasyon işlemi gerçekleştirilmeyen PS yörüngesi eksen değerleri ne bakıldığı zaman daha keskin geçişler mevcut (Şekil 3.24). Optimizasyon işlemi yapılan IROBOT yörüngesi eksen değerleri ne bakıldığı zaman keskin geçişlerin daha yumuşak geçişlere döndüğü gözükmektedir (Şekil 3.25).

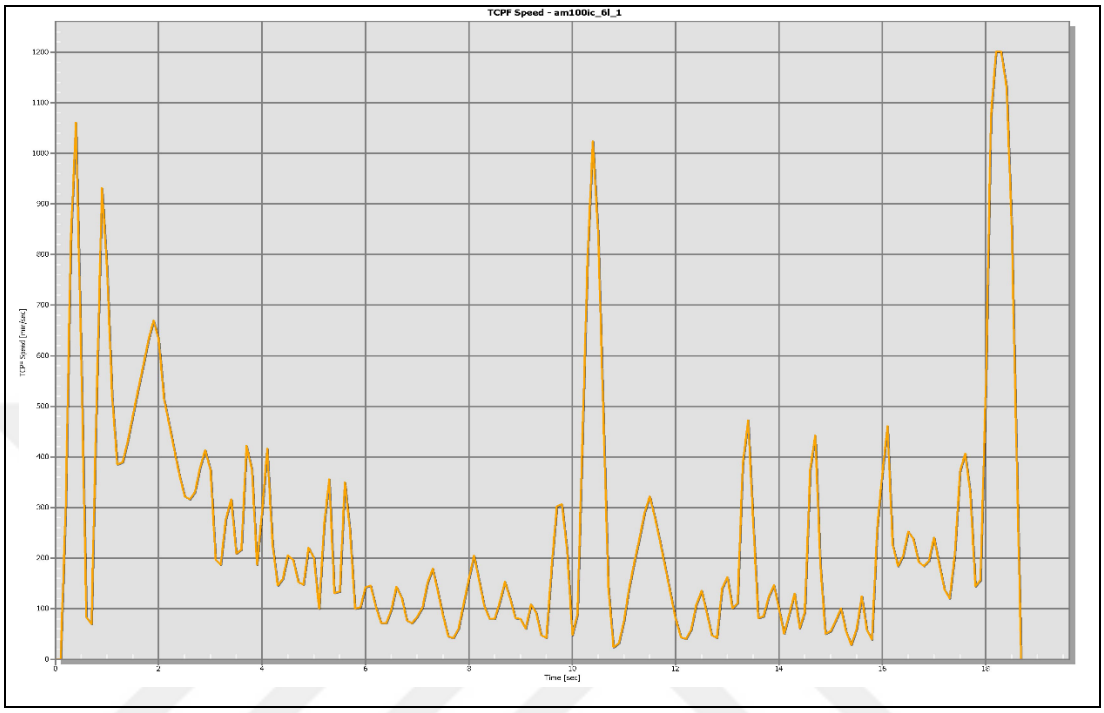


Şekil 3.24: PS yörünge eksen değerleri

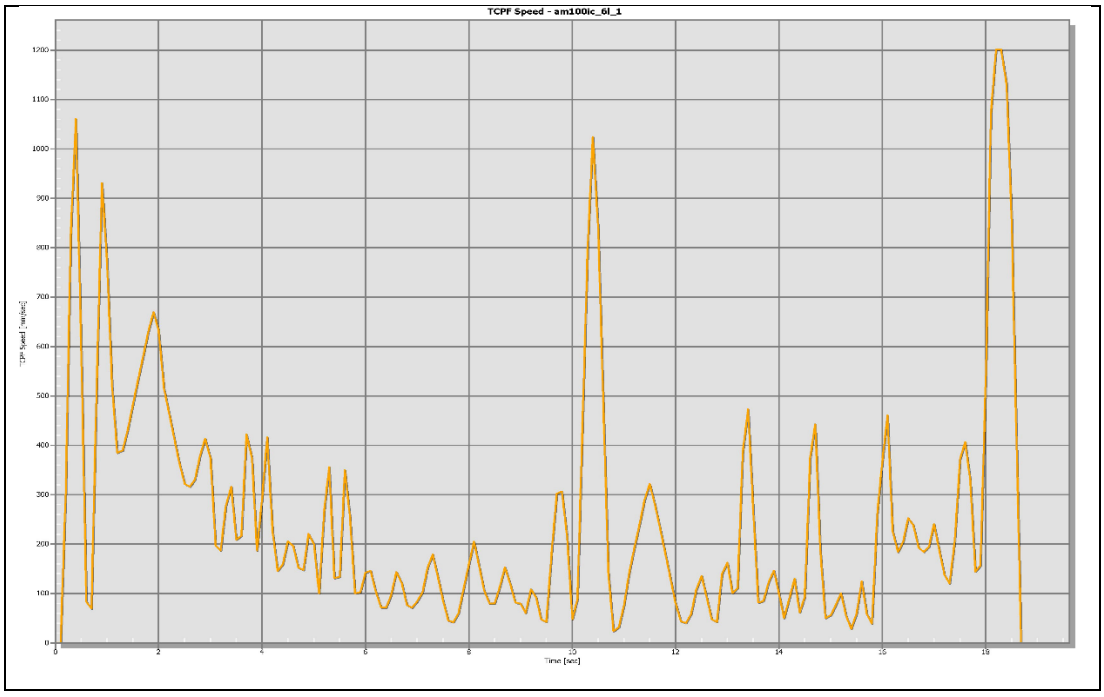


Şekil 3.25: IROBOT yörünge eksen değerleri

Robotun optimizasyon işlemi gerçekleştirilmeyen PS yörüngesi robot TCP hızı normal (Şekil 3.26). Optimizasyon işlemi yapılan IROBOT yörüngesi robot TCP hızı daha hızlıdır (Şekil 3.27).



Şekil 3.26: PS yörünge eksen değerleri



Şekil 3.27: IROBOT yörünge eksen değerleri

IROBOT back-up dosyasının son hali Fanuc marka robota yüklendi. Yörüngenin olması gereken nokta ile gittiği nokta arasındaki kaçıklık yetkin robot programcısı yöntemiyle ölçüldü. IROBOT programına ölçtüğümüz yörünge kayıklık verileri girilerek yeni back-up dosyası oluşturuldu. Yeni yörünge çalıştırıldığı zaman robotun başarılı bir şekilde istenen noktalara gittiği tespit edildi (Şekil 3.28).

Şekil 3.28: Yeni yörünge sonrası robot doğru noktada

4. TARTIŞMA ve SONUÇ

Yapılan tüm bu çalışmalar başarıyla gerçekleştirildi. Process Simulate programından alınan offline back-up dosyaları programda işlendi ve robota yeni back-up dosyası yüklendi. Robot yeni back-up dosyası doğrultusunda yörüngesini kaydırıp gitmesi gereken noktalara gitmektedir. Kazanılan çevrim zamanı süresi grafikler ve uygulamalarla test edildi sonuçlar alındır. Sistem 1 saat boyunca 40 kez otomatik bir döngüde çalıştırılarak performansı test edildi. Yapılan incelemeler sonucu robotun gitmesi gereken noktalara göre pozisyon hassasiyeti 2-10 mm arası değiştiği tespit edildi. Hassasiyetin bu kadar fazla olmasının sebeplerinden biri kullanılan ölçüm yöntemidir. Yetkin robot programcısı yöntemi sahada kullanılması daha kolay olan yöntemdir. Bu yöntem insan kaynaklı olması nedeniyle küçük hassas kaçıkları oluşabilmektedir. Uygulama ortamında homojen bir aydınlatma olması gerekmektedir. Gidilmek istenen noktanın görüşü ne kadar net olur ise, program o ölçüde hassas bir şekilde ayarlanabilmektedir. Simülasyon programında oluşturduğumuz back-up dosyası uzantısı .LS uzantılı dosyadır. Bu dosya direkt robota yüklenebilmektedir fakat bazı Fanuc modellerinde yüklenebilmesi için RoboGuide programı kullanılarak .TP uzantılı back-up dosyası alınması gerekebilmektedir. IROBOT programının en verimli ve en çok kullanılan saha uygulaması ise parçasının robot tarafından Punta kaynak attığı uygulamalardır. Robotun gitmesi gereken hedef noktaların alanlarının küçük olası kayıklığın hesaplamasındaki hassasiyeti arttırdığı için konum bilgisi almak kolay ve hızlıdır. Punta kaynaklı atan robotlu sistemler seri bir sekilde çalışılabilmektedir ve durmaksızın üretimini gerçekleştirebilir. Kurulan hattın ne kadar hızlı devreye alınması maliyeti düşürürken üretime başlama süresinin artması verimi artırmaktadır.

KAYNAKLAR

[1] Marimon R.,Scott A., (2001), "Computational Methods for the Study of Dynamic Economies", OUP Catalogue, Oxford University Press, number 9780199248278.

[2] Koivisto J.M., Haavisto E., Niemi H., Katajisto J., Multisilta J., (2016), "Elements Explaining Learning Clinical Reasoning Using Simulation Games", International Journal of Serious Games, 3 (4), 29-43.

[3] Collins J., Chand S., Vanderkop A., Howard D., (2021), "A review of physics simulators for robotic applications", IEEE Access, 9, 51416–51431.

[4] Cheng L., Guo S., Wu J., (2021), "Innovation Education Research Based on Crossobject Interactive Robot Simulation Platform", SHS Web of Conferences 123, 01020, Wuhan, China, 2261-2424, September.

[5] Datteri E., Schiaffonati V., (2019), "Robotic Simulations, Simulations of Robots", Minds & Machines, 29, 109–125.

[6] Elmquist A., Negrut D., (2021), "Modeling Cameras for Autonomous Vehicle and Robot Simulation: An Overview", IEEE Sensors Journal, 21 (22), 25547-25560.

[7] Pan Z., Polden J., Larkin N., van Duin S., Norrish J., (2012), "Recent progress on programming methods for industrial robots", Robotics and Computer Integrated Manufacturing, 28 (2), 87-94.

[8] Hong L., Wang B., Yang X., Wang Y., Lin Z., (2020), "Offline programming method and implementation of industrial robot grinding based on VTK", Industrial Robot, 47 (4), 547–557.

[9] Deng S., Cai Z., Fang D., Liao H., Montavon G., (2012), "Application of robot offline programming in thermal spraying", Surface Coatings Technology, 206 (19-20), pp. 3875–3882.

[10] Bedaka A.K., Vidal J., Lin C.Y., (2019), "Automatic robot path integration using three-dimensional vision and offline programming", International Journal of Advanced Manufacturing Technology, 102 (5-8), 1935–1950.

[11] Zheng C., An Y., Wang Z., Wu H., Qin X., Eynard B., Zhang Y., (2022), "Hybrid offline programming method for robotic welding systems", Robotics and Computer-Integrated Manufacturing, 73, 102238.

[12] Holubek R., Delgado Sobrino D. R., Košťál P., Ružarovský R., (2014), "Offline programming of an ABB robot using imported CAD models in the RobotStudio software environment", Applied Mechanics and Materials, 693, 62-67.

[13] Castor M., (2020), “Robotic offline programming in a COVID-19 world: Offline robot programming platforms are increasingly being used during the COVID-19 pandemic. See four scenarios and programming features”, Control Engineering, 67 (12), M8-M8.

[14] Jazar R. N., (2010), “Theory of applied robotics: Kinematics, dynamics, and control”, 2nd Edition, Springer US, 1–883.

[15] Siciliano B., Khatib O., (2016), “Springer Handbook of Robotics”, Ed. by B. Siciliano and O. Khatib.

[16] Hartenberg R. S., (1964), “Kinematic synthesis of linkages”, Ed. by J. Denavit.

[17] Konstantinidis E. I., Katsavounis S., Botsaris P. N., (2020), “Design structure matrix (DSM) method application to issue of modeling and analyzing the fault tree of a wind energy asset” Wind Energy (Chichester, England), 23 (3), 731–748.

[18] Qiu L., Liu X., Zhang S., Sun L., (2014), “Disassemblability modeling technology of configurable product based on disassembly constraint relation weighted design structure matrix (DSM)”, Chinese Journal of Mechanical Engineering, 27 (3), 511–519.

[19] Reza C. M., Dachyar M., Nurcahyo R., (2019), “Project Scheduling of New Product Development Process in Automotive Industry in Indonesia Using Design Structure Matrix (DSM)”, IOP conference series. Materials Science and Engineering, 598 (1), 12048.

[20] Eppinger S. D., (2012), “Design structure matrix methods and applications”, Ed. by T. R. Browning.

[21] Khatib O., Burdick J., (1987), “Motion of robotic manipulators along specified paths”, International Journal of Robotics Research, 6(3), 3-17.

[22] Siciliano B., Sciavicco L., Villani L., Oriolo, G., (2010), “Robotics: Modelling, Planning and Control”, Springer Science & Business Media.

[23] Craig J.J., (2005), “Introduction to Robotics: Mechanics and Control”, Pearson Education.

ÖZGEÇMİŞ

Yaşar BAYKAL, 2018 yılında Gebze Teknik Üniversitesi Elektronik Mühendisliği Bölümü'nden mezun oldu. 2019 yılında Gebze Teknik Üniversitesi Elektronik Mühendisliği Anabilim Dalı'nda Prof. Dr. Fuad ALIEW danışmanlığında yüksek lisans eğitimine başladı. 2018 yılından itibaren Güneş Dinamik Şirketinde Otomasyon Mühendisi olarak görev yapmaktadır.

EKLER

Ek A: Tez Çalışması Kapsamında Yapılan Yayınlar

Baykal Y., Aliew F., "Fabrika Üretim Otomasyonunda Robotlu Simülasyon Uygulamaları ve Offline Programlamada Verimliliği Arttırma", 2023, 7. GTÜ Lisansüstü Araştırmalar Sempozyumu 2023.