T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

3 BOYUTLU NESNE ALGILAMASI İLE ÇALIŞAN ROBOT KOLLU OTONOM TASIMA SİSTEMI TASARIMI

YÜKSEK LİSANS TEZİ

Tichaona Jonathan MAKOMO

Enstitü Anabilim Dalı	**: MEKATRONİK MÜHENDİSLİĞİ**
Tez Danışmanı	**: Dr. Öğr. Üyesi Barış BORU**

Ocak 2020

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

3 BOYUTLU NESNE ALGILAMASI İLE ÇALIŞAN ROBOT KOLLU OTONOM TASIMA SİSTEMI TASARIMI

YÜKSEK LİSANS TEZİ

Tichaona Jonathan MAKOMO

Enstitü Anabilim Dalı : MEKATRONİK MÜHENDİSLİĞİ

Bu tez 14/01/2020 tarihinde aşağıdaki jüri tarafından oybirliği/oyçokluğu ile kabul edilmiştir.

Prof. Dr.	Dr. Öğr. Üyesi	Doç. Dr.
Durmuş KARAYEL	Barış BORU	Devrim AKGÜN
Jüri Başkanı	Üye	Üye

BEYAN

Tez içindeki tüm verilerin akademik kurallar çerçevesinde tarafımdan elde edildiğini, görsel ve yazılı tüm bilgi ve sonuçların akademik ve etik kurallara uygun şekilde sunulduğunu, kullanılan verilerde herhangi bir tahrifat yapılmadığını, başkalarının eserlerinden yararlanılması durumunda bilimsel normalara uygun olarak atıfta bulunulduğunu, tezde yer alan verilerin bu üniversite veya başka bir üniversitede herhangi bir tez çalışmasında kullanılmadığını beyan ederim.

Tichaona Jonathan MAKOMO

14.01.2020

TEŞEKKÜR

Kinect teknolojisiyle robot kontrolü yapmaya çalıştığım çalışmamın sonuna gelmiş bulunmaktayım. Bu konuda daha sonra yapılacak olan çalışmalara temel olmasını ve insanlığa yararlı olmasını dilerim. Tez konusunun belirlenmesinden sonuçların alındığı ana kadar tüm çalışmalarımda değerli yardımlarıyla tez çalışmasında yol gösteren, kıymetli danışmanım Dr. Öğr. Üyesi Barış BORU'ya sonsuz teşekkürlerimi sunarım. Çalışma boyunca desteğini benden esirgemeyen ve bana yönden destek olan Araştırma Görevlisi ve doktora öğrencisi Kenan Erin'e sonsuz teşekkürü bir borç bilirim.

IÇİNDEKİLER

SİMGE VE KISALTMALAR DİZİNİ

3B	: 3 Boyutlu
DOF	: Serbestlik derece
FMS	: Esnek Üretim Sistemleri
FOV	: Görüş alanı
GUI	: Grafiksel kullanıcı arayüzü
IR	: Infrared
Kinect	: Windows sensörü için Kinect
RGB	: Kırmızı, Yeşil, Mavi
SDK	: Yazılım Geliştirime Kiti
TCP	: Tool'un Merkez Noktası
TCP/IP	: İletim Kontrol Protokolü / İnternet Protokolü
TOF	: Uçus Zamanı
USB	: Universal Serial Bus

TABLOLAR LISTESI

ŞEKİLLER LİSTESİ

ÖZET

Anahtar Kelimeler: Kinect, nesne tanıma, 3d görme sistemi, MATLAB, RobotStudio

Bu tez kapsamında, 3D algıyıcıyı bir robotik manipülasyona entegre etme, bir nesneyi otomatik olarak algılama, izleme ve tutma ve başka bir yere yerleştirme işlemleri yapılmıştır. Robotu kontrol etmek için, çok yönlü ve güçlü bir programlama dili olan MATLAB programı kullanılmıştır. 6 eksenli robot manipülatörleri endüstride Montaj, paketleme, paletleme ve alma ve yerleştirme işlemlerinde yaygın olarak kullanılmaktadırlar. Bu çalışma kapsamında fabrikalarda kullanılan tut ve bırak işlemi uygulanmıştır.

Çoğu robotik sistem algılama yeteneğinden yoksundur ve bu nedenle iş parçalarını tanıma ve görev alanını algılama işlerinde çalışamaz. Bu tür sistemler, görevdeki değişiklikleri barındıracak şekilde otomatik değişiklik yapma esnekliği ve kapasitesinden yoksundur. 3B görüntü algılama sistemini bir robot sisteme entegre etmek esnekliğe neden olur ve değişiklik yapılmasını sağlar.

Bu çalışma için, bir gripper ve Windows kamera sensörü için 3D Kinect ile donatılmış bir ABB IRB120 robotundan oluşan bir robotik sistem kullanılmıştır. 3D veri toplama, görüntü işleme ve bazı farklı kamera parametreleri incelenmiştir. Kameradan elde edilen görüntülerden robotun çalışma alanını belirlemek ve iş parçalarını tanımak için kullanılır. Bu görüntüler daha sonra nesnelerin konumunu ve yönünü hesaplamak için kullanılır. Bu görüntüler kullanılarak, bir nesneyi kavramak için robot ile otomatik bir yol ve yörünge geliştirildi. İş parçalarını tespit edebilmek için, MATLAB'ın Bilgisayar Görme Araç Kutusu ve Görüntü Alma Araç Kutusu'ndaki mevcut algoritmalar kullanılarak nesne tanıma teknikleri uygulanmıştır. Bunlar nesnenin konumu ve ilgilenilen nesnenin yönelimi hakkında bilgi vermektedir. Bilgi daha sonra bir bilgisayar ağı üzerinden istemciden sunucuya bir bağlantı yoluyla yol planlaması için robota aktarılır.

Geliştirilen sistem daha sonra laboratuvar ortamında deneysel olarak uygulanmıştır ve projenin genel önemini bulmak için tüm çalışma, projenin bireysel hedeflerini bir dizi hedefe dayalı çalışma çerçevesine göre değerlendirerek değerlendirilmiştir. Bu nedenle test sonuçları, sistemin robotun çalışma alanına rastgele yerleştirilmiş nesneleri seçip yerleştirebildiğini göstermektedir. Elde edilen sonuçlarla yüksek bir başarı oranı tespit edilmiştir. Buna ek olarak, esnek bir işlem sağlamak için robotik bileşenleri birleştiren güvenilir ve basit bir iletişim geliştirilmiştir.

DESIGN OF AUTONOMOUS HANDLING ROBOTIC MANIPULATOR SYSTEM WITH 3D OBJECT DETECTİON

SUMMARY

Keywords: Kinect, object recognition, 3d vision system, MATLAB, RobotStudio

This thesis deals with the possibility of integrating a 3D sensor into a robotic manipulation, with an intention to automatically detect, track and grasp an object, placing it in another location. To enhance the flexibility and easy functionality of the robot, MATLAB, a versatile and powerful programming language is used to control the robot. These kind of 6 DOF robot manipulators are widely used in industry. They find common application in assembling, packaging, palletizing and pick and place operations. For this work, a common industrial task in many factories of pick and place is implemented.

Most robotic systems lacked sensory capabilities and therefore cannot work in an intelligent manner in terms of recognizing workpieces and perceiving task space. Systems of this kind lacked the flexibility and capacity for automatic modification in their path to accommodate changes in the task. A robotic system with a flexibility and allowance for modification is achieved through the integration of a 3D vision sensing system.

For this work, a robotic system consisting of an ABB IRB120 robot equipped with a gripper and a 3D Kinect for Windows camera sensor is used. The 3D data acquisition, image processing and some different camera parameters are investigated. The information in the image acquired from the camera is used to determine the robot's working space and to recognize workpieces. This information is then used to calculate the position and orientation of the objects. Using this information, an automatic path to grasp an object was designed and developed to compute the possible trajectory to an object. To be able to detect the workpieces, object recognition techniques are applied using available algorithms in MATLAB's Computer Vision Toolbox and Image Acquisition Toolbox. These give information about object position and orientation of the object of interest. The information is then transferred to the robot for path planning via a client-to-server connection over a computer network. The system is implemented for a laboratory scale experiment and to find the overall importance of the project, the entire work was evaluated by assessing the individual objectives of the project against a set of goal-based work frame. The test results therefore show that the system is able to pick and place objects randomly placed in the robot's working space in real time. A high success rate is shown by the obtained results. In addition to this a reliable and simple communication between the robotic components was developed.

BÖLÜM 1. GİRİŞ

1.1. Geçmiş

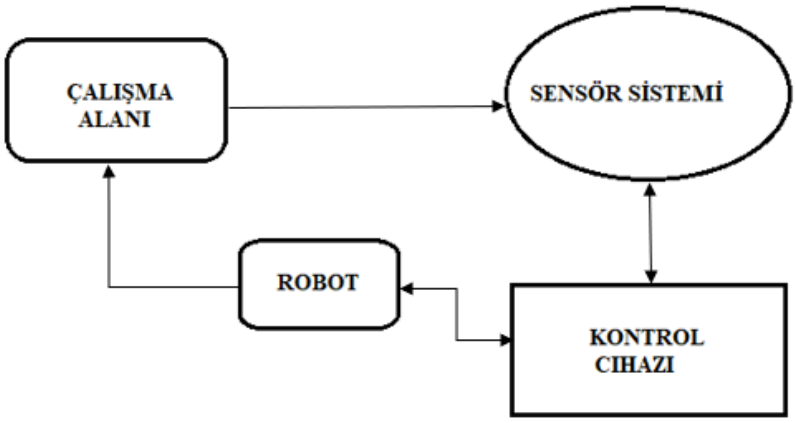
Bu çalışma, yüksek hassasiyetli bir robot manipülatör ile entegre edilmiş bir görüntü sisteminden görüntü bilgisini kullanarak tutma ve yerleştirme yöntemini uygulayarak nesneleri manipüle etmenin basit bir endüstriyel görevini göstermektedir. Bu tezin amacı, Kinect sensöründen gelen görüntü bilgilerini kullanarak başarılı bir yüksek hassasiyetli tutma ve bırakma işlemini geliştirmektir. Gelişmiş kontrol uygulamalarındaki artış, güçlü donanım platformları ve gelişmiş algılama yetenekleri nedeniyle, robotik sistemler navigasyon, keşif, eğlence, sanayi, insani refah vb. gibi çeşitli alanlarda yer bulmaya başlamıştır.

Son yıllarda, robotik sistemler birçok işleme ve üretim endüstrisinde gittikçe daha fazla kullanılmaktadır. Yarı otomatik olarak kullanılıp insanlar ile yan yana çalışırlar, oysa bazı durumlarda tam otomatiktirler yani bir görevi tamamlamak için robotlar birbirleriyle çalışmaktadır. Üzerinde çalışılan nesneler, renk, şekil, boyut ve benzeri gibi değişmez fiziksel parametrelerle farklılık göstermektedir. Bu nedenle, robotun kontrol sistemindeki her çalışma parçası hakkında anlık bilgileri kolaylaştırabilmek için gerçek zamanlı bir algılama sistemi gerekir [1]. Esnek üretim sistemlerinin (FMS) geliştirilmesinde bir takım teknolojik ilerlemeler kaydedilmiştir. Bir FMS, robot kontrolörleri ve makine parçaları ile birlikte robot manipülatörleri gibi bir veya daha fazla taşıma cihazından oluşan, tasarlandığı ve geliştirildiği farklı parça ailesini idare edebilecek şekilde düzenlenmiş bir sistem olarak tanımlanmaktadır [2]. Robotlar etkili algılama kabiliyetinin eksikliği nedeniyle, birçok üretim birimi ve montaj noktası, iş parçalarını tanımak ve çalışma alanını görmek için akıllıca hareket edemez. Bu sistemlerin birçoğunun manipülasyon için robota gönderilen önceden tanımlanmış konumlar ve oryantasyonlarla çalışması gerekir. Bununla birlikte, robotların yeni pozisyona gitmesinin zorlaştığı durumlarda ve oryantasyonda değişiklik müsaade

etmezler. Bu anormallikleri gidermek için robot sistemi, kesin olarak konumlandırılmış nesnelerle başa çıkabilmek ve çalışma ortamındaki belirsizlik ve varyasyonları ele almak için robotla gerçek zamanlı olarak iletişim kurabilen akıllı ve esnek bir görüş sistemi ile donatılmalıdır.

Ancak bu çalışma, değişen nesneye dinamik ayarlamanın şart olduğu uygulamalara gerçek zamanlı sensör tabanlı kontrol sistemi sağlamaya odaklanmaktadır. [1] ve [3], kameradan elde edilen görsel bilgilerin, servo sistemi olarak adlandırılan kapalı devre robot kontrolü için kullanıldığını söylemiştir. Servo sistemlerin çalışması, özellikleri ve zorlukları hakkında genel bir bilgi [4] ve [5] söylemiştir. Yaptıkları çalışmada, elde edilen görüntüden 2B formatında nesne bilgisi alınır ve robot kontrolü için nesnenin konumu ve yön bilgisi ile 3B poza dönüştürülür. Kontrol görevi 3B Kartezyen uzayda planlanmıştır ve verilerin 2B'den 3B'ye eşlenmesi için kamera modeli gereklidir. Görsel bir servo sistemi kurmak için, kinematik ve dinamikler, kontrol teorisi, görüntü işleme ve kamera kalibrasyonu dahil bilgisayarlı görü, bilgisayarlı görüntü sistemi ve kamera kalibrasyonu dahil olmak üzere robot modellemesi dahil farklı alanlardan gelen bilgiler kullanılmıştır [6], [7], [8].

Vizyon bilgisinin bir diğer önemli özelliği, robotun görüş alanını sürekli güncelleme yeteneğini arttırmasıdır. Bu yapılandırma normalde el ele veya el ele yapılandırma olarak adlandırılır. Tüm sistemin mimarisi Şekil 1.1.'de gösterilmektedir:



Şekil 1.1. Robotik Görme Tabanlı Kontrol Sistemi

Diyagram aşağıdaki gibi açıklanabilir:

- Çalışma alanı (Workspace): demirbaşlar, iş parçaları, iş parçaları ve aletler içerir
- Sensör sistemi (Sensory system): robotun çevresini algılamasına ve cisimleri tanımasına izin verme.
- Kontrol sistemi (Control system): sırasıyla görevleri ve robotu düzenlemek için sistem bilgisayar yazılımını ve robot kontrol ünitesini barındırır.
- Robot manipülatörü: Robot kontrol ünitesinin kontrolü altında eylem yapar.

Otomotiv endüstrisi geleneksel olarak itici güç ve otomatik robot manipülatörlerinin en büyük tüketicisi olduğu için, üretim hacmi önemli ölçüde düşük olan işletmeler, otomotiv endüstrisi bugün olduğu kadar birkaç yıl içinde de robotlardan faydalanacaktır [9]. Öte yandan, küçük seri üretimi olan işletmeler için, bu otomasyon büyük ölçüde yatırım maliyetleri ve programlama kabiliyeti önemli meselelerdir. Sürekli değişen üretim modelleriyle, yeniden programlama maliyetinin yatırım maliyetini büyük bir oranda artması muhtemeldir [10].

Bu çalışmada tutma ve yerleştirme işlemi, bir robotun kendi alanında bir nesneyi nasıl izleyebileceğini, algılayabileceğini ve kavrayabildiğini göstermektedir. Buradaki görev, masaya yerleştirilmiş renkli bir nesneyi kavramak ve görevi tamamlamak için bir görsel denetim sistemi, renkli nesneyi gerçek zamanlı olarak renk segmentasyonu algoritmaları kullanarak tespit etmektedir. Nesne daha sonra robot manipülatörü tarafından belirlenmiş bir yere taşınır. Bütün bunlar TCP / IP tarafından sağlanan iletişim protokolleri kullanılarak yapılmaktadır. Görüntü işleme algoritması, koordinat dönüşümleri, yol planlama algoritmaları ile robot kolu kontrolü gerçekleştirilmektedir. Sistemin geçerliliği tartışılmış ve sonuçlar sunulmuştur.

1.2. Hedefler

Bu tezin amacı aşağıdaki gibidir:

- Robotun çalışma ortamını algılamak ve manipüle edilecek iş parçalarını tanımak için robot 3B görme sistemini araştırmak ve uygulamak.

- Olası sabit kavramaları hesaplayan ve gripper duruşunu buna göre ayarlayan bir otomatik kavrama planlayıcısı geliştirmek ve uygulamak
- Ana bilgisayardaki robot ve programın manipülasyon için birbirleriyle iletişim kurmasını sağlayan bir iletişim protokolü geliştirmek
- Yukarıdaki tüm fonksiyonellikler, ilgili bir görsel algılama sistemi gerçekleştirmek.

1.3. Metod

Bu çalışmada sunulan hedeflerin teorik ve pratik bölümleri vardır. Teorik yönleri cevaplamak ve böylece problemi pratik olarak çözmeyi desteklemek ve hazırlamak için ilgili eserin ve birçok makalenin literatürü araştırılmıştır.

Hedef 1:

3B kamera ile robotik vizyon sunulmuştur. Bu proje, girdi için kullanılan Microsoft Kinect kullanımını önerilmiştir. Bu tür kameraların çalışma prensibini açıklamak önceliklidir. 3B görüntülerin yapısı, 3B görüntülerinin robot vizyonunda kullanılması için gereken ortak işlem adımları açıklanmıştır.

Hedef 2:

Robotu MATLAB fonksiyonlarıyla kontrol etmek için RAPID adlı robotik programlama dilinde kodlanmış bir yazılım geliştirilmiştir. Bu işlevlerden bazıları, yerel robot dili işlevleriyle uyuşur, ancak işlevselliğini arttırmak için yeni işlevler de geliştirilmelidir.

Hedef 3:

Bir nesnenin algılaması için bir program geliştirilmiştir ve daha sonra robotun manipülasyonu için nesnenin konumu ve yönelimi robota gönderilmiştir. Program, ABB robotları için kullanılan Robot Studio RAPID'de yazılmış başka bir programla uyumlu olmalıdır. Bilgisayar programı ile robot arasında istemci ve sunucu iletişimi için bir program geliştirilmiştir. 3B kameradan elde edilen verilere dayanarak gerekli düzeltmeleri hesaplayan program bu projenin ana kısmını oluşturmaktadır.

1.4. Tezin Yapısı

Bu tez kapsamında, 3 boyutlu görüntüleri kullanarak endüstriyel robotları otomatik olarak kontrol eden ve algılanan nesneleri işlemek için görüntü verilerinin işlenmesini sağlayan bir uygulamanın geliştirilmesi amaçlanmıştır. Bu çalışmanın deney düzeneği, derinlik veri toplama, nesne uyumu, dönüşüm tahmini ve bilgi akışı açıklanmıştır.

Bölüm 2, mevcut tekniklerin araştırılması ile ilgili geçmiş deneyimlerin kavramlarına genel bir bakış sunar. Bu bölüm aynı zamanda robot manipülatörü, bu proje için kullanılan kamera gibi çeşitli bileşenlerle ilgili tüm teknik bilgiler açıklanmış olup yapılan çalışma anlatılmıştır.

Bölüm 3, bu bölüm gelişmiş sistemi sunmaktadır. Deneysel kurulum ve sorunu çözmek için izlenen adımlar sunulmuştur.

Bölüm 4, deneyden elde edilen sonuçlar, sistem performansını iyileştirmeye yönelik bazı önerilerle birlikte sunulmuştur.

Bölüm 5, çalışmayı gelecekteki çalışmalar ve uygulamalar için tartışma, sonuç ve önerilerle özetlemektedir.

BÖLÜM 2. LİTERATÜR İNCELENMESİ

2.1. Ilgili İş

Bu bölümde, halihazırda var olan tekniklerin araştırılmasıyla ilgili kavramlara değinilmiştir. Bu bölümün temel amacı, yaptığımız çalışmada yer alan tüm bileşenlerle ilgili tüm teknik bilgiler açıklanmıştır. Bu bileşenlerden bazıları robotun kendisi, tutucu, kullanılan kamera sensörü, görüntü sistemi, yazılım ve donanım bileşenlerini içerir.

2.2. Önceki Eserler

[11]'e göre, çevreyi ve öngörülen girdileri dikkatli bir şekilde kontrol etmek fabrika otomasyonunun temelini oluşturur. Marvel, endüstrilerin uzun süredir vizyon algılamanın gerekli olduğu her süreçte 2B görsel algılamaya güvendiğini söylemeye devam etti. Ancak, bu daha az yapılandırılmış ortamlarda teknoloji ve robot kullanımındaki gelişmeler nedeniyle değişmektedir. Bu ortamlarda 3B algılamaya ve planlamaya ihtiyaç vardır.

2.3. Bilgisayar Görüşü

Bilgisayar görme sistemleri, bir ışık kaynağı ile birleştirilmiş iğne deliği kamerasından oluşmaktadır. Sistemin bir manipülasyon çerçevesi ile düzgün çalışması için, sistemin manipüle edilecek tüm nesneleri takip etmesi sağlanmalıdır. Ancak böyle bir sistemin çalışması iki alt sistemden oluşacaktır, Nesne tespiti ve pozisyon tespiti. Görme rehberli robotlar konsepti son yıllarda çok fazla ilgi gördü. Endüstriyel proseslerdeki robotların görmesine izin verilirse, sektördeki farklı prosesler için birçok robotik fonksiyonellik açar. Endüstriyel robotik sistemlerde görüntü sensörlerinin popüler olmasının nedeni budur. Vizyon veya duyusal geri bildirim olmadığında, endüstriyel

bir robotun çevresiyle akıllıca etkileşime girmesi zorlaşır. Bir robotun, çalıştığı çevre ile ve onunla iletişim kurmak için ihtiyaç duyduğu en temel anlam vizyondur [12]. Robot vizyonu ve endüstriyel uygulamalar için bilgisayar vizyonu ve görüntü tanıma teknikleri kullanılır. Birçok endüstriyel robotik operasyonda, alma ve yerleştirme işlemleri gibi birçok işin otomasyonuna ihtiyaç vardır [13].

Geçmişte, kullanımda olan görme sistemleri, “bak ve hareket” adı verilen açık döngü tekniğine dayanıyordu. 1970'lerin sonunda, dikiş kaynağı, tutma, 10Hz servolama ve izlemek için 3B pozisyon kontrolü yapabilen sistemler gösterilmiştir. Buna göre [14], Inoue ve Shirai, görsel geri bildirimi kullanarak görevin doğruluğunu artırmak için robot pozisyonunun nasıl düzeltilebileceğini gösterdi. Bunu kare kutuya kare bir prizma yerleştirmek için 7 serbestlik derece manipülatörü kullanarak başardılar.

Görüş geri bildirim robotu kontrolü için kullanılabilecek iki yöntem vardır: dinamik görünüm ve taşıma sistemi ve doğrudan görsel servo sistemi [15]. Dinamik görünüm ve hareket sisteminde, robotik manipülasyonun görüş geribildirim kontrolü hiyerarşik bir şekilde gerçekleştirilir. Hiyerarşi, daha yüksek seviyedeki görüş sisteminin daha düşük seviyeli bir robot kontrol ünitesine istenen bir referans girişi sağlaması anlamına gelir; bu, daha sonra robotu dahili olarak dengelemek için eklemlerden gelen geribildirimleri kullanır. Doğrudan görsel servo kontrol sistemlerinde, adının doğrudan olduğunu söylesede bir görsel kontrolör doğrudan robot bağlantılarına girdi hesaplar. Bununla birlikte, doğrudan görsel servo sistemi, yüksek hızlı görüş geri bildirim bilgisi talebinde bulunur ve sırayla hem sistemin yazılımı hem de donanımı üzerinde büyük bir yük oluşturur [16].

Son yıllarda manipülasyon algısı, yeni algılama protokolleri uygulayarak normal 2D görüntü işlemeden geçmiştir. [17] 'de, bir konteynere yığılmış nesnelerin derinlik kenarlarını görmek için çoklu flaşlı bir kamera sensörü uygulanmıştır.

Son teknolojiler çok çeşitli uygun fiyatlı yüksek çözünürlüklü RGB kameraları getirmiştir. Bununla birlikte, yalnızca 3B dünyasının bir yansımasını yakalamak için

tasarlanmış 2B görüntüler kullanırken, sağlam ve çok verimli görme güdümlü sistemler geliştirmek zordur.

Uçuş Süresi (ToF) derinlik kameralarının ve Microsoft Kinect^{TM} derinlik kamerasını ortaya çıkışı, 3B veri akışını video hızında kullanıma sunmuştur. Bu, robota yararlı ve akıllı kararlar alınmasında yardımcı olmak için yeni yöntemler gördü.

2.4. Kinect Kameralar

Kinect sensörü (Şekil 2.1.'de), PrimeSense adlı bir İsrail şirketi tarafından geliştirilmiştir ve Kasım 2010'da Microsoft tarafından piyasaya sürülmüştür. Başlangıçta Microsoft'un konsolu XBOX 360 için bir oyun aracı olarak üretildi, ancak derinlik yetenekleri ve satın alınabilirliği birçok uygulama için cazip hale getirdi. Şimdi Kinect sensörü robotik, sağlık, grafik, İnsan-Bilgisayar-Etkileşim, eğlence vb. alanlarda kullanılmaktadır.

2.4.1. XBOX kinect - Windows için Kinect

Xbox 360 oyun konsolu piyasaya sunulduğundan bu yana en çok popülerliği hızla artmaktadır. Cihazın popülerliği arttıkça, Microsoft, Şubat 2012'de resmi Microsoft Kinect SDK'yı piyasaya sürdü. Özel olarak Windows'taki uygulamaların geliştirilmesi için hedeflenen değiştirilmiş donanım ve ürün yazılımı içeren yeni bir sensörle birlikte geldi. Kinect Xbox 360 ve Windows Kinect arasında küçük bir fark bulunmaktadır. Microsoft Kinect, daha iyi iskelet takibi gibi bazı gelişmiş SDK özelliklerinin yanında 40 mm'ye kadar yakın derinlik bilgisi elde etmemizi sağlamaktadır (Şekil 2.2.'de). Bu fark, Windows Kinect'in ticari bir lisansa sahip olması ve Xbox Kinect ile mümkün olmayan ticari uygulamalar için kullanılmasını sağlamaktadır. [18].

Şekil 2.1. Microsoft Kinect Cihazi [18]

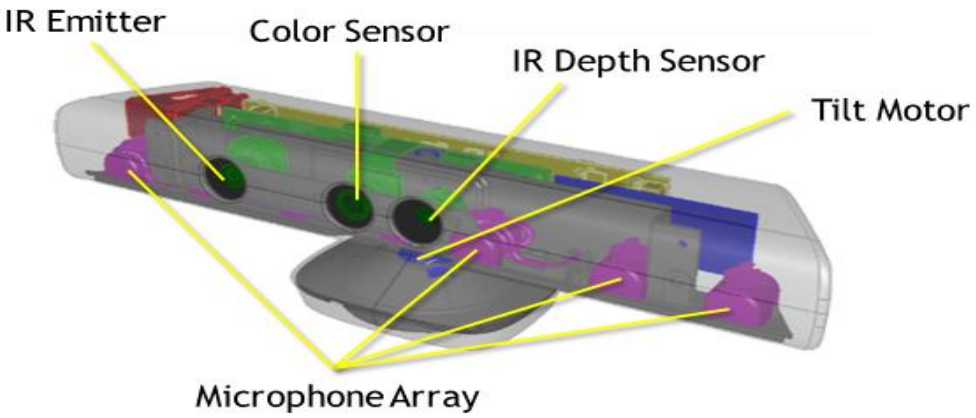
2.4.2. Donanım bileşenleri

Kinect'in derinlik sensörü sistemi, bir kızılötesi (IR) vericiden ve onu yakalayan bir kızılötesi kameradan oluşur. Derinlik algılama sisteminin yanı sıra, cihaz donanımında bazı bileşenler mevcuttur [18].

Xbox için Kinect donanım bileşenlerinin özellikleri Tablo 2.1.'de gösterilmektedir.

Tablo 2.1. Xbox için Kinect'in teknik özellikleri [18].

Bileşen	**Açıklama**
Kızılötesi projector	• Yansıtılan benekli deseni oluşturmak için camdan kırılan bir 830nm kızılötesi lazer yayar.
Kızılötesi kamera	• Kızılötesi projektör tarafından oluşturulan deseni kaydeder. • 1280 x 960 çözünürlüğe ve 57^{O} yatay ve 43^{O} dikey görüş alanına (FOV) sahip monokromatik bir CMOS sensördür.
Renkli kamera	• 640 x 480 çözünürlüğe ve 30FPS kare hızına sahip bir VGA renkli kamera (saniye başına kare). • Çözünürlük yeteneği 1280 x 960'a kadar çıkabilir, ancak kare hızında yaklaşık 10FPS'lik bir azalma olabilir. • İade edilen renk formatı RGB, YUV veya Bayer olabilir.
Mikrofon dizisi	• Gürültü bastırma ve yankı iptali ile yüksek kaliteli ses kaydı için dört mikrofon dizisi. • Eğim motoru -27 ila +27 aralığında, programlı olarak kontrol edilir. • Yerçekimi ile ilgili olarak oryantasyonun alınması için 3 eksenli akselerometre



Şekil 2.2. Microsoft Kinect bileşenleri [19]

2013 yılında, Xbox One için Kinect (Şekil 2.2.), Microsoft Xbox video oyun konsolu için ilk nesil sensörlerin devamı olarak geliştirilmiştir. Kinect'in bu sürümü 512 x 424 piksel derinlik çözünürlüğünde geniş açılı TOF kamera ile ileri bir teknolojiye sahiptir. Aynı zamanda yüksek çözünürlüklü 1920 x 1080 piksel RGB kamera ve 48kHz'de çalışan dört dizili bir kameradan oluşmaktadır. Kameranın görüş alanı 70^{o} yatayda ve 60^{o} dikeyde durur ve Sürekli Dalga Modülasyonu kullanır. Sensör, iki yarıya ayrılmış ve saat sürücüleri tarafından sürülen bir piksel dizisinden oluşur. Bu üst ve alt yarıların topladığı fotonlar karşılaştırıldığında, ışığın dalga süresi ölçülür ve mesafe hesaplanır. Bu, sensörün yüksek derinlik hassasiyetine sahip olmasını sağlamaktadır [19].

Şekil 2.3. Xbox One Kamera alanı koordinat sistemi [19]

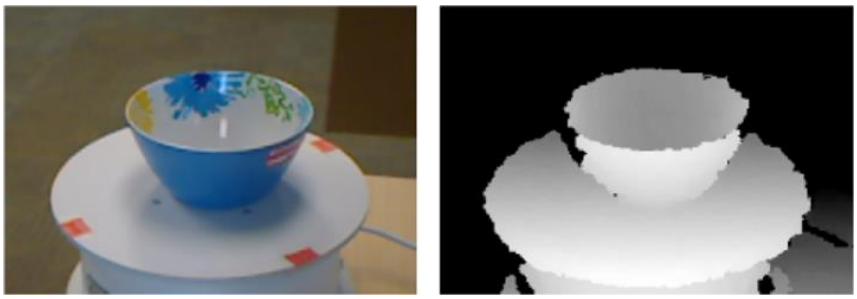
[20] 'e göre, ışık kaynağı modülasyon sinyali, her piksel saatinin fazı, modülasyon frekansı ve her piksel için kazanç hepsi programlanabilir. Bu özellik, sensörün endüstride kullanım için arzu edilen özelliklere sahip olmasını sağlar. Yüksek piksel çözünürlüğü ile birleştiğinde, sensör 0,8- 4,2 metre arasındaki mesafelerde yüksek

dinamik aralık çalışması sağlamaktadır. Aralığın %0,5'inin derinlik belirsizliğine ve %1'in altındaki doğruluk hatasına sahiptir. Çerçevelerin ortalaması, sensöre yakın bir mesafede alınabilir ve yüksek hassasiyet elde edilebilir. Mevcut tüm ToF sensörleri arasında, Kinect en yüksek piksel çözünürlüğüne sahiptir [20].

2.4.3. Windows için Kinect

Microsoft Windows için uyumlu olan Kinect Windows V2, 2014 yılında piyasaya sürülmüştür. Windows için yazılım geliştirme kiti SDK 2.0 ile birlikte geldi [19]. Kinect'in bu sürümü, Kinect Xbox'ta bulunmayan özelliklere sahip harici bir güç adaptörü ve bir USB 3.0 bağlantısıyla birlikte kullanılmaktadır. Bu gelişmiş sürüm, Windows 8, Windows 8.1 ve daha ileri sürümler için Kinect özellikli yazılımın geliştirilmesi içindir. Kinect Windows sensörü bu iş için kullanılmış olan ve bir Windows 10 ile uyumlu olan sensördür.

Görüntü elde etmek için, Windows Kinect sensörü kullanılmaktadır. Windows Kinect renkli bir kamera, IR kameraya sahiptir. Şekil 2.4.'de gösterildiği gibi hem RGB hem de derin görüntüler elde etme yeteneğine sahiptir.



Şekil 2.4. RGB ve Derinlik görüntüleri Kinect V2'yi oluşturur

RGB görüntüler renkli görüntülerdir ve kamera insan gözünün gördüğü gibi görmektedir. Derinlik kameradaki her pikselin derinliğini kaydeder. Bunu, bir nesneye doğru ışık yaymak için IR yayıcılar kullanarak yapar ve ışık, nesnenin yüzeyine ulaştığında ve IR kamera tarafından yakalandığında yansıtılır. Dolayısıyla, derinlik yukarıda açıklanan TOF prensibi kullanılarak hesaplanır. Nesnenin kameradan

uzaklığı daha sonra yayılan ışığın nesneye gitmesi ve kızılötesi kameraya geri sıçraması için geçen süre ölçülerek hesaplanır. Sabit bileşen ışık hızı olacaktır.

Windows Kinect V2 ToF sensörünün teknik özelliklerini özetleyen Tablo 2.2.'de verilmiştir.

Tablo 2.2. Kinect V2 sensörünün teknik özellikleri

Bileşen	**Açıklama**
IR kamera çözünürlüğü	512 x 424 piksel
RGB kamera çözünürlüğü	1920 x 1080 piksel
Görüş alanı	70 x 60 derece
Kare hızı	Saniyede 30 kare
Operasyonel ölçüm aralığı	0,5 – 4,5 metre
Nesne piksel boyutu	1,4mm (@ 0,5m) ila 12mm (@ 4,5m)

2.5. Obje Algılama

Nesne algılama, bir videodan veya görüntüden bir nesneyi bulma işlemidir. Nesne algılama işlemi, görüntüde ne tür nesnelerin olduğunu, konumlarını ve yönlerini belirlemeyi sağlamaktadır. Nesne tespiti için kullanılan birçok farklı yazılım ve algoritma vardır. Örneğin, tek başına MATLAB, özellik çıkartma, arka plan çıkarma, şekil algılama ve diğer algoritmalar arasında renk bölümlendirmesinden nesne algılama için birçok yol sunmaktadır. Bu konuyu ele almak için pek çok araştırma ve çalışma yapılmıştır. Nesne algılamanın arama penceresi boyutunu giderek arttırarak yapılan ve kayan pencere yaklaşımını kullanan bir yöntem tarafından araştırılmıştır [21].

Çalışmalarım zaman içerisinde robot vizyon sistemlerine yardımcı olmak amacıyla nesne izleme alanında yapıldı. Bu özellikle endüstride hassasiyet için hassas görüş gerektiren montaj işlemleri için yapılmıştır. Nesne tespiti için kullanılan yaygın veya

en yaygın yöntem, görüş sisteminin söz konusu nesnenin farklı görüntülerini içeren bir veri kümesi kullanılarak eğitildiği yöntemdir. Bu görüntüler daha sonra tespit edilmesi için gerçek görüntü ile eşleştirilebilir [22].

Ancak bu işlem, tüm işlemin yapılması için zaman alıcıdır ve zamanın çok önemli olduğu vizyon temelli manipülasyona uygulanamaz. Bunun ışığında, zamandan tasarruf etmek için başka algoritmalar geliştirilmiştir ve nesnenin başka bir özelliği bulunabilir, örnek olarak nesnenin centroid'i verilebilir [23]. Houshangi ve arkadaşları bu yaklaşımı çoklu nesnelerin tespiti için kullandılar. Nesne izlemesinde nesnenin centroidini bulmak için matematiksel algoritmalar tarafından önerilmiştir [24]. Öte yandan, görüntü çarpması ve Kalman filtrelemesi gibi nesne izleme yöntemleri de savunucudur [25].

[26], farklı nesne takip ve tespit yaklaşımlarını incelemiştir. Bu çalışma, nesne algılamanın yapılabileceği birçok farklı yol olduğunu göstermiştir. Metotlar, nesnenin modellenmesi için tespitin yapıldığı ortamdan düşünülen görüntü özelliklerine, nesnenin şekline ve görünümüne göre farklılık gösterir. Bunu, [27], [28] ve [29] 'de tartışılan insan vücudunun takibi izlenmiştir. Daha basit bir nesne sınıflandırma metodu devreye sokuldu [30] ve elde edilen sonuçlarla Kalman filtre algoritmasından daha iyi olduğu ortaya çıkmıştır. Alan'ın metodu, nesne takibinde doğru sonuçlar elde etmek için zamansal farklılık ve görüntü şablonu eşleşmesine dayanıyordu. Bundan daha önce, [31], Kalman filtresinden daha iyi performans gösteren başka bir stokastik algoritma tanıtmıştır. Bu algoritma, karışıklık ortamlarında dinç izleme için önerildi. Çalışmalarında çevre karışıklıktan uzaktı ancak nesne algılama sınıflandırması ünlü Kalman filtresini kullanmıştır. Nesne algılama, iç ve dış mekanlarda birçok çalışma ortamı içerdiğinden, nesnelerin mümkün olan en kısa sürede algılanmasını sağlamak amacıyla farklı yaklaşımlar farklı ortamlar için yapılmıştır. İç mekan ortamlarında nesne algılama için kullanılan yöntemler, dış mekanlarda kullanılanlara göre değişebilir. [32]'de, her pikselin bir Gauss karışımı olarak modellendiği bir yöntem önermiştir. Modelin güncellenmesi için çevrimiçi bir yaklaşım uygulanmış ve sabit dış mekân izlemesinden tatmin edici sonuçlar elde edilmiştir.

Bilinen bir başka nesne algılama yöntemi, Liu tarafından önerilen arka plan çıkarma işleminde segmentasyondur [33]. Bu, bir nesneyi tespit etmek için referans çerçevesi mevcut çerçeveden çıkarıldığında bulunan fark içinde olur. Bu keşifler ve araştırmaların tümü, özerk ve yardımcı navigasyona yardımcı olmak için Kalman filtreleme yönteminin bir uzantısıydı. Kalman metodu, Kalman tarafından, [34] gizli veri astarı filtreleme problemi için özyinelemeli bir çözüm elde ederek yazılmıştır.

2.6. Görüntü İşleme

Bu proje, gerçek zamanlı nesne tespiti için bir Kinect kamera sensörü kullanan robotlara yardımcı navigasyon hakkındadır. Sensör tarafından elde edilen görüntüler daha sonra görüntüyle ilgilenilen nesneyi tespit etmek için görüntü işleme işlevleri ve algoritmaları kullanılarak işlenir. Nesnenin rengi, şekli, boyutu gibi özellikler nesne algılama için kullanılmaktadır.

2.6.1. Görüntü alma araç kutusu

Görüntü toplama araç kutusu MATLAB'da bulunan bir araç kutusudur. MATLAB ve Simulink'te görüntü ve video çekmekten sorumludur [35]. Bu araç kutusu, MATLAB'da kameraların başlatılmasını ve durdurulmasını, görüntü ve video kareleri edinmesini sağlayan işlevleri içerir.

2.6.1.1. Görüntü bölümlendirme

Görüntü segmentasyonu, çoğunlukla nesne algılama amacıyla bir nesneyi farklı parçalara bölme işlemidir [36]. [37] 'ye göre, segmentasyon, her görüntü için daima nesne tespitiyle başlar. Normalde uygulanan yöntem, arka plan çıkarma veya ön plan algılamaya yöneliktir. İlke, referans çerçevesinin mevcut çerçeveden çıkarılması ve işlemde bazı piksellerin hareketsiz kalmasıdır. Sabit pikseller arka planı tanımlarken, sürekli değişen veya hareket edenler ön planı belirtir. Bu normalde hareket tabanlı algılama olarak adlandırılır. Her zaman olduğu gibi, bilgisayar bazı dış sesler nedeniyle hareketli pikselleri algılayabilir. MATLAB bu anormalliği engellemek için

bir işlev sunar ve buna blob analizi denir. Çevresel gürültünün tüm segmentasyon sürecine karışmasını önlemek veya azaltmak için eşikleme adı verilen bir işlem yapılır. Eşikleme, segmentasyonun gerçekleştirileceği değerin belirlendiği bir işlemdir. Bununla birlikte, hangi tür bölümlendirme seçildiğine bağlı olarak değişir. Örneğin renk segmentasyonu için farklı renklerin eşik değerleri farklılık gösterir. Eşik değeri kullanarak, etkili blob analizi elde edilir.

2.6.1.2. Blob analizi

Blob analizi, tutarlı görüntü bölgelerinin analizine dayanan makine vizyon tekniğidir [38]. Kabarcıklar, ön planda konumlarını değiştiren piksel kümeleridir. Çerçevelerden birtakım özellikler hesaplanabilir. Bu özelliklerden bazıları, blobun ana ekseni, alanı, boyutları, centroid yönü ve şeklidir. Bu özellikleri belirlemek için MATLAB'de regionprops adlı bir fonksiyon kullanılır. MATLAB'daki bilgisayar vizyonu araç kutusundan blobAnalyzer adlı bir işlevi vardır. Blob bölgesi özelliği, istenen blob'u bulmak için kullanılan özelliktir [37].

2.6.2. Görüntü işleme araç kutusu

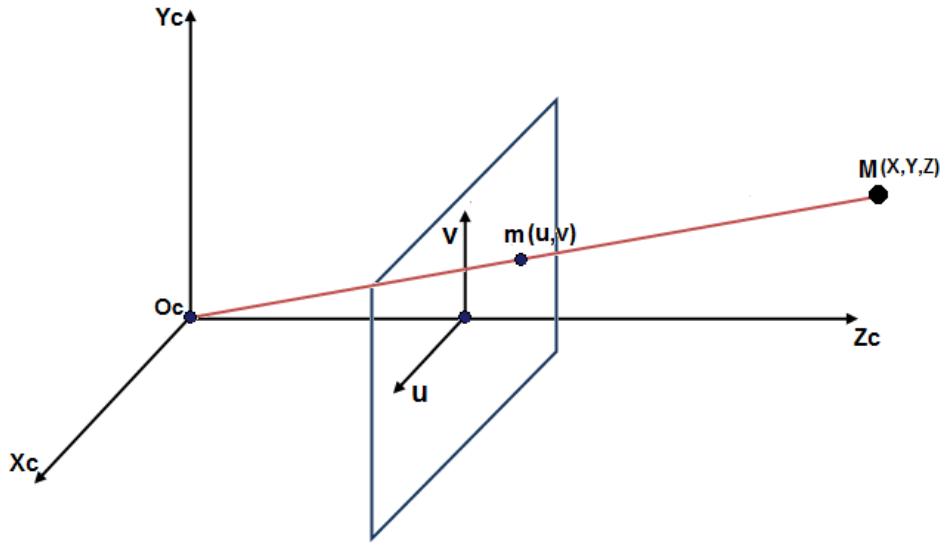
Görüntü işleme araç kutusu, MATLAB'ın bilgisayarla görülebilmesi için en temel araç kutularından biridir. Görüntü işleme ve analiz, görselleştirme ve algoritma geliştirme için çok çeşitli fonksiyonlar, uygulamalar ve algoritmalar ile kullanışlıdır (Mathworks, 2014). Bu, nesnenin algılanmasından, renginin ve yönünün tespit edilmesinden sorumlu olan araç kutusudur. Ayrıca görüntüdeki gürültüyü de azaltabilir. Renkli olarak, bir görüntü, her bir öğenin belirli bir pikselin RGB değeri olduğu bir M(satır) x N(sütun) x 3 matrisi olarak depolanır [39]. Her pikselin yoğunluğu 0 ile 255 arasındadır. Bir BW (Siyah beyaz) görüntü, her pikselin 0 (siyah) veya 1 (beyaz) olduğu bir 2B matris olarak depolanır.

[40] 'e göre, renk tabanlı segmentasyon, tek renkli bir nesnenin arka plandan ayrıldığı bir işlemdir. $L * a * b$ renk uzayına dönüştürüldükten sonra görüntü, nesnenin kırmızı-yeşil ve sarı-mavi bileşenleri arasındaki Öklid mesafesi hesaplanır. Asgari değerin,

ilgilenilen nesnenin en doğru tahminine yol açtığı belirtilmektedir. Köşeler, MATLAB'daki Harris köşe dedektörü işlevi kullanılarak hesaplanır ve ayrıca merkezciler köşegenlerin kesişimi ile de verilir.

2.6.3. Poz tahmini

Poz tahmini, bir görüntüdeki veya videodaki nesnenin pozunu bulma işlemidir. Proseslerde, poz tahmini için kullanılan duruma, ortama ve sensörlere bağlı olarak kullanılan birçok farklı yöntem ve algoritma vardır. Poz tahmini, nesne tespitinden sonra gelir. Bir nesne izlenip belirlendikten sonra uygun kavrama noktalarının bulunduğundan emin olmak için pozu hesaplanmalıdır. Bir nesnenin pozunu hesaplamak için bir monoküler görüntü veya stereo görüntü kullanılabilir. POSIT algoritması, tek bir iğne deliği kamera yöntemi kullanılarak (Şekil 2.5.), poz tahmini için çoğunlukla görüş tabanlı manipülasyonda kullanılır [41].



Şekil 2.5. İğne Deliği (Pinhole) Kamera Modeli [41]

Görmeye dayalı poz tahmininin alt tabakasını, kenarlar, köşeler ve eğriler gibi görüntü özellikleri oluşturur. Söz konusu nesnenin 3B pozisyonunun bilinmesi gerekmekte ve poz tahmininin gerçekleşmesi için takdir edilmelidir. Herhangi bir nesnedeki poz, kamera ile ilişkili olarak rotasyon ve çevirisinin bir birleşimidir. Dönme 3x3 matrisi temsil eder ve 3x4 birleştirilmiş matristeki 3x1 çevirisini matematiksel olarak [R | T] olarak temsil eder. İğne deliği ilkesi, herhangi bir 3D için ve herhangi bir nokta için,

kameranın odak uzaklığı ve ana odak noktası ile birlikte pozunu hesaplamak için kullanılan bir görüntü üzerinde 2B karşılık gelen projeksiyon olacak şekilde çalışır.

Odak uzaklığı ve ana odak noktası, önerilen bir kalibrasyon yöntemi kullanılarak hesaplanabilir [42]. Normal iğne deliği kamera modelini ve onun koordinatlarını yukarıdaki gösteren Şekil 2.5.'ten itibaren, 3B nokta M $[XYZ]^T$'nin m $[x, y]^T$ noktasında 2B görüntü düzlemine yansıdığı ve homojen bir matris ile temsil edildiği görülebilir. Şeklinde:

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}_m = \begin{pmatrix} f & 0 & \alpha_x & 0 \\ 0 & f & \alpha_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}_K \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}_M \tag{2.1}$$

Burada K, kameranın içsel matrisidir; f, kamera odak uzaklığıdır, λ, homojen ölçeklendirme faktörüdür, αx ve αy, kameranın ana noktasıdır. Tüm bu parametreler, 2d düzleminde bir M noktasının nokta projeksiyonunu hesaplamak için kullanılır;

$$\lambda m = \left[K \mid 0_3\right] \begin{bmatrix} R & 0 \\ 0_3 & 1 \end{bmatrix} \begin{bmatrix} 0_3^T & -T \\ 0 & 1 \end{bmatrix} M \tag{2.2}$$

Bu nedenle, 3B noktaları 2B düzlem görüntü noktalarına yansıtmak suretiyle poz tahmin edilir.

2.7. Kamera Kalibrasyonu

Kamera kalibrasyonu, aynı kamera tarafından alınan belirli bir görüntüden veya videodan bir iğne deliği kamera modelinin kamera parametrelerini alma işlemidir. Bu parametreler daha sonra mercek bozulmasını düzeltmek, gerçek dünya birimlerinde tespit edilen bir nesnenin boyutunu hesaplamak ve sahnedeki bir kameranın konumunu belirlemek için kullanılır [43]. Bu proje, IR ve RGB sensörlerinden oluşan Kinect kamera sensörünü kullanır. Daha iyi doğruluk ve daha iyi lokalizasyon için, iç parametrelerini ve birbirlerinin konumlarını belirlemek için bu iki kameranın uygun şekilde kalibre edilmesi gerekir. Her içsel matris, iki odak uzunluğu, iki temel nokta

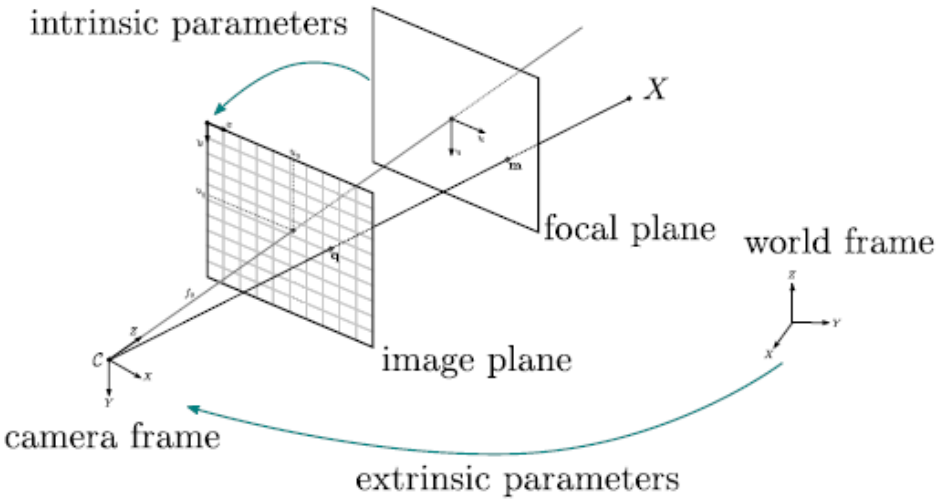
parametreleri ve 5 bozulma katsayısı ile toplam 9 parametre içerir. Ekstrinsik matris ise, 12 döndürme için 9 ve çeviri için 3 olmak üzere 12 parametreden oluşur. Birçok kamera üreticisi kameralarını fabrikada kalibre eder. Kinect için ayrıca fabrikada kalibre edilmiştir, ancak sürücüler IR değerleri ve RGB kameraların iç değerleri ve göreceli konumları hakkında hiçbir şey göstermezler. Bu kullanılmadan önce, kameraların herhangi bir operasyon alanında doğru sonuçlar almak için ve herhangi bir görev için kalibre edilmesine ihtiyaç duyulmasına neden olur [44].

Kamera kalibrasyonu, uzayda bir 3B noktasını 2B görüntü düzlemine yansıtan parametreleri belirlemeye çalışır. En yaygın yöntem sensörleri ayrı ayrı kalibre etmek ve aralarındaki pozu kalibre etmektir [45], [46]. Bu yöntem, kalibrasyonda her zaman yaygın olarak kullanılan yöntemdir, Bouguet'in MATLAB Kamera Kalibrasyon Araç Kutusu [47] ve OpenCV kütüphanesi [48]. Bu yöntemlerde, farklı pozlardan görüntüler çekmek için bir dama tahtası deseni kullanılır. Kalibrasyon modelinin köşelerini tespit etmek için kullanılabilecek birkaç algoritma vardır. Degrade arama alt piksel doğruluğunu optimize etmek için kullanılır. Kalibrasyonun doğruluğunu arttırmak için birçok pozun farklı pozlarda çekilmesi gerekir. Kalibrasyon işlemi, başarılı bir kalibrasyon için bir pikselden daha az olması gereken yeniden düzeltme hatasını azaltmak için bir teklifte olacaktır. Reprojeksiyon hatası, projeksiyon matrisini kullanarak gerçek 2B yansıtılan noktalar ile 3B noktaların ortalama farkıdır. Kalibrasyon için kullanılan popüler optimizasyon aracı Levenberg-Marquardt algoritmasıdır [49].

Kinect'in derinlik kamerası bağımsız kalibrasyonunda yüksek hassasiyet gerektirir. [50], Windows'un Kinect gibi ToF kameralar için çok kanallı bir yöntem önermiştir. Bir robot kolunda kullanıldığında, robotun kameranın pozunu bilmesi gerekecektir, çünkü yöntem çok sayıda parametre kullanır. [51], kameranın pozunu belirleyebilen yüksek çözünürlüklü bir kamera kullanarak bir robot ihtiyacını ortadan kaldırdı. (D. Lichti, 2008), lazer parametreli tarayıcıyı, tüm parametrelerin kalibrasyonunu yapan bir yöntem olan bir düzlemsel kalibrasyon nesnesi kullanarak bir kalibrasyon yapmanın bir yolu olarak tanıtmıştır. İç kamera parametreleri, piksel koordinatları ve

3d koordinatlar arasındaki ilişkiyi temsil ederken, diğer yandan dünya referans çerçevesine göre, dış parametreler kameranın konumunu ve yönünü belirler [52].

Daha önce de belirtildiği gibi, kalibrasyon, piksel koordinatlarının gerçek dünya koordinatlarına dönüştürülmesi ve ayrıca lens bozulmasının azaltılması amaçlıdır. Son zamanlarda kamera kalibrasyonu yapmak için birçok yöntem geliştirilmiştir. Zhang ve Tsai tarafından sunulan yöntemler en yaygın kullanılandır [53], [54]. Kalibrasyonun nedeni, içsel ve dışsal parametreleri bulmamızdır. Herhangi bir kameranın kendine özgü parametreleri değiştirilemez ve satın alındıktan sonra kameranın veri sayfasıyla birlikte verilir. Dışsal parametreleri kameranın konumunu ve oryantasyonu Şekil 2.6.'de gösterir.



Şekil 2.6. İğne deliği kamerasının genel bir modeli [55]

Yukarıdaki şekil, fikri 3D koordinatlar $[X, Y, Z]$ ve görüntü düzlem koordinatları (x, y) arasında bir ilişki kurmak olan normal bir iğne deliği kamerasını göstermektedir. Bu, aşağıdaki denklemin çözülmesiyle elde edilir [56], [57].

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} [T] \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.3}$$

İçsel parametreler fx ve fy olmak üzere sırasıyla yatay ve düşeyde fokal uzunluğu tanımlayan dört parametreden ve ana noktalar olan c_x ve c_y'den oluşur. T, dışsal parametreleri tanımlayan, dönüş ve çeviri parametrelerini belirten bir 4x4 matrisidir. Zhang'ın metodu tüm bu parametreleri çözüyor.

Dışsal parametreler, referans çerçevesine göre kameranın konumunu ve yönünü tanımlar. Elle kalibrasyon olarak iyi bilinir [58], [59]. Dışsal parametreler ayrıca 3x3 dönme matrisine ve 3x1 çeviri matrisine sahip 4x4 matris olarak da gösterilebilir.

$$L = \begin{pmatrix} r_{11} & r_{12} & r_{13} & -R_1^T T \\ r_{21} & r_{22} & r_{23} & -R_2^T T \\ r_{31} & r_{32} & r_{33} & -R_3^T T \end{pmatrix} \tag{2.4}$$

Ri, I = 1,2,3, R matrisinin satır tarafından oluşturulan 3-vektörüdür.

Genel olarak, sensör veya lens değiştirilmedikçe, bir kez kendinden içsel kalibrasyon yapılması yeterlidir. Bunu yapmak için test edilmiş ve onaylanmış yöntemler zaten mevcuttur [60], [61], [62]. Dışsal kalibrasyon için, ihtiyaca ve çevreye bağlı olarak, özellikle vizyon yardımcı robotik operasyonlarla, el ele yapılabilir [63]. Bununla birlikte, bazı araştırmacılar tarafından kullanılan tüm çoklu sensörler için ortak bir çalışma alanına yerleştirilen ısmarlama hedef nesnenin kullanılmasıyla çoklu kamera ve robot kalibrasyonu gerçekleştirilmiştir [64].

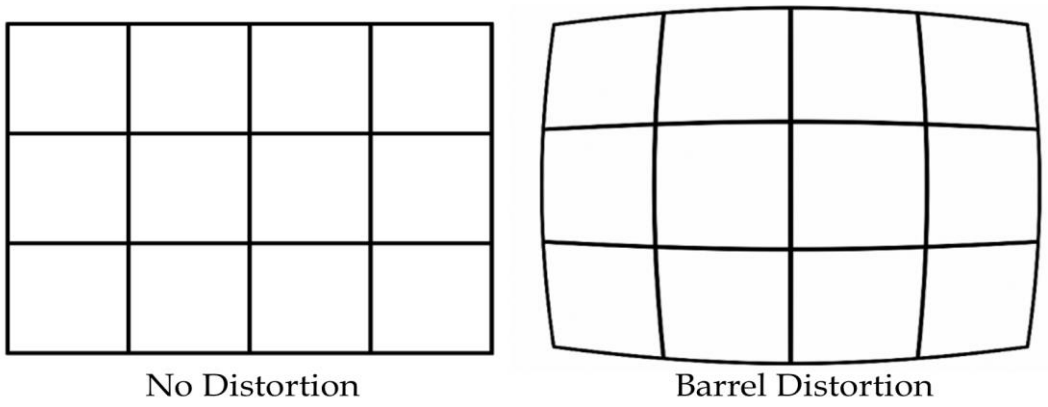
Dışsal kalibrasyonun yapılacağı sistemlerde, dışsal parametrelerin doğruluğunu belirlediği için koordinat oryantasyonunun çok önemli olduğu anlaşılmalıdır. Bazı yönelimler yanlış sonuçlar verecektir ve dönüş açıları hesaplanmalıdır. Nesne koordinatının yönlendirmesi yanlışsa, dışsal parametreleri tahmin etmek zor olacaktır [65].

Robot bilimindeki yanlışlık sorununu çözmek için, bazı araştırmacılar kalibrasyon yaparken daha iyi sonuçlar almak için farklı algoritmalar kullanırlar. Coste ve arkadaşları, Tsai'nin yakınında radyal bozulma olmadan bir yöntemi nasıl kullandığını

açıklar. Levenberg-Marquardt algoritmasını kullandığını, Tsai yönteminin daha iyi ve uygun olduğunu düşündüğünü yineledi. Bunun nedeni, süreci başlatmak için parametrelerin değer aralığı hakkında bazı temel bilgilere ihtiyaç duymasıdır. Yöntem kısıtlamaları azaltır ve hızlı bir şekilde çözüme ulaşmanıza yardımcı olur [65].

2.8. Lens Bozulması

Bu, radyal ve teğetsel bozulmaların bir sonucu olarak ortaya çıkan bir olgudur (Şekil 2.7.). Bununla birlikte, mercek bozulmasının temel olarak radyal bozulmaya neden olduğu kabul edilmiştir ve keşfedilmiştir. Radyal bozulma, görüntüyü bir bütün olarak deforme eden, düz çizgilerin kavisli görünmesine neden olan optik sapma olarak tanımlanabilir. Bu, lensin yanlış radyal eğriliğinin bir sonucudur. Endüstriyel görüntü sistemleri için görüntü büyütme, optik merkeze olan uzaklıkla azalır, görüntü üzerinde bir varil şekilli etkiye neden olur ve bu nedenle namlu bozulma olarak bilinir.



Şekil 2.7. Bozulmamış doğrusal görüntü (solda) ve namlu bozulma (sağda) [66]

2.9. Robot

Bu tezde kullanılan robot, aşağıdaki şekilde gösterilen bir ABB IRB120'dir. ABB'nin en yeni nesil 6 DOF manipülatöründen biri olan IRB 120, ABB'nin robot koleksiyonunun en küçüğü ve 3 kg'lık bir taşıma kapasitesiyle geliyor. Sadece 25 kg ağırlığındaki endüstriyel robot, özellikle esnek robot tabanlı otomasyonun gerekli olduğu imalat endüstrileri için tasarlanmıştır. Taşınabilir ve kullanıma uygun olan

robot, esnek işlemler için kullanılmasını sağlayan ve harici sistemlerle iyi iletişim kurabilen açık bir yapıya sahiptir. Robot, montaj işlemleri, malzeme taşıma için çok uygundur ve daha fazla kontrol ve yol doğruluğu ile kompakt, çevik ve hafif bir çözüm sunar. Bu proje için kullanılan ABB IRB120 robotu, Şekil 2.8.'de gösterilmiştir.

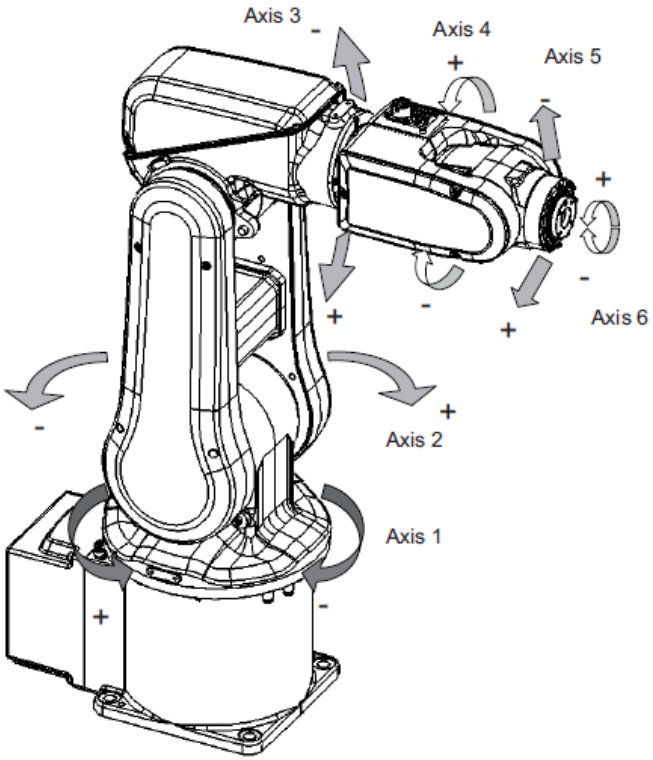
Şekil 2.8. ABB IRB120 robotik manipülatör [67]

ABB'nin en küçük robotu olan IRB120, ABB serisinin işlevselliğine ve uzmanlığına sahip ancak daha küçük bir pakettedir. Ağırlığı, neredeyse her yere ve herhangi bir yönde veya herhangi bir kısıtlama olmadan açılı olarak monte edilmesini sağlar [67].

2.9.1. Çok amaçlı bir robotu

IRB 120, çeşitli endüstrilerde, yiyecek ve içecek, elektronik, ilaç, tıbbi ve araştırma sektörlerini içeren farklı işlem türleri için kullanılabilen çok amaçlı bir robottur. Gıda sınıfı yağlama (NSF H1), yiyecek ve içecek uygulamaları için ödünsüz güvenlik ve hijyen sağlayan Temiz Oda ISO Sınıf 5'i içerir.

IRB 120'nin geometrisi, 580 mm'lik yatay bir erişim, sınıfının en iyisi strok, tabanının 112mm altına ulaşma ve çok kompakt bir dönüş yarıçapı ile birlikte gelir.



Şekil 2.9. ABB 120 robot mekanik yapı [67]

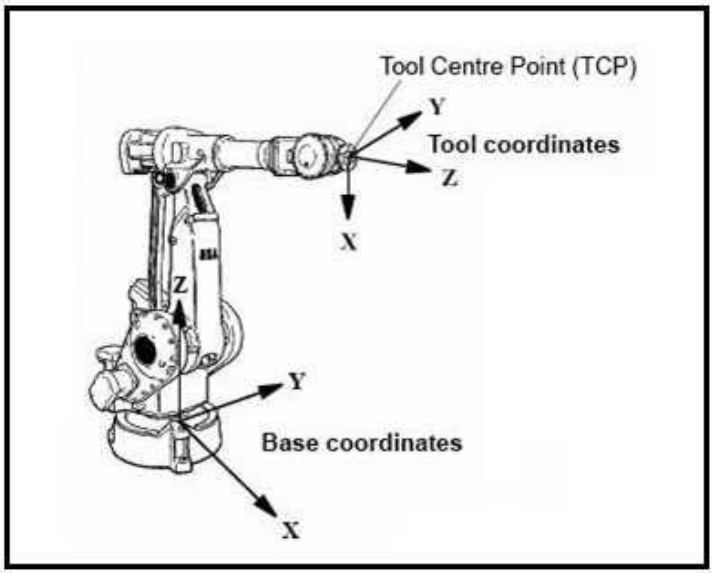
Yukarıdaki robot'un açıklaması, Tablo 2.3.'te gösterilmektedir.

Tablo 2.3. Robot ekseni özellikleri [67]

Hareketin yeri	**Hareket türü**	**Hareket aralığı**	**Hız**
Eksen 1	Dönme hareketi	+165° to -165°	250 °/s
Eksen 2	Kol hareketi	+110º to -110º	250 °/s
Eksen 3	Kol hareketi	+70º to -110º	250º/s
Eksen 4	Bilek hareketi	+160° to -160°	320 °/s
Eksen 5	Viraj Hareketi	+120º to -120º	320 °/s
Eksen 6	Dönüş hareketi	+ 400° to -400 ° (varsayılan)	420 °/s
		+242 devir -242 devir	
		maksimum	

2.9.2. Robot koordinat sistemi

Robotikte, her zaman robotun konumu ve hareketi TCP'e göre verilir. TCP, her zaman robotun kullandığı herhangi bir aracın merkezinde bulunur. Robotun tabanına göre ölçülür veya kendi koordinat sistemini kullanabilir. Aşağıdaki Şekil 2.10.'de, robot tabanını ve TCP koordinat sistemlerini göstermektedir.



Şekil 2.10. Robot tabanı ve takım koordinat sistemleri [67]

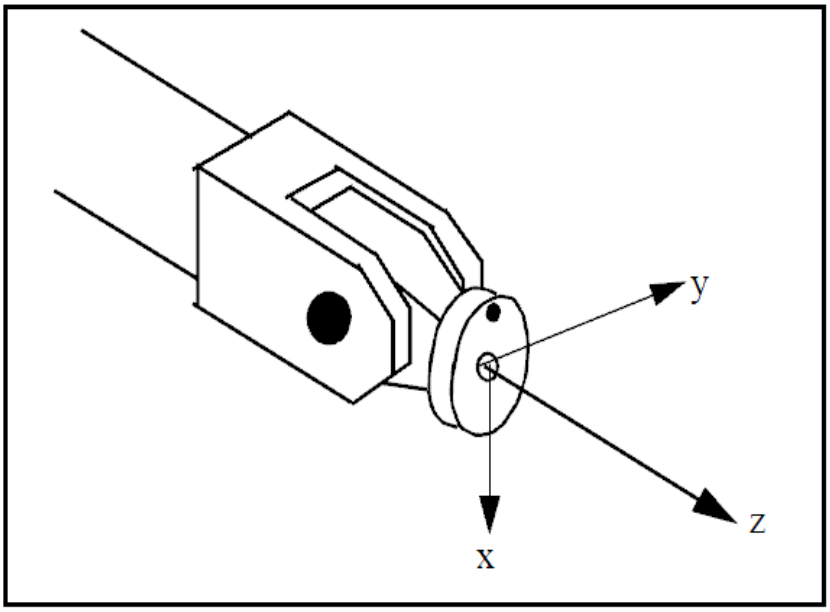
Robot taban sistemi aşağıdaki gibi tanımlanabilir:

- Robotun tabanında bulunur
- Kaynak, eksen 1'in merkezinde ve robotun taban montaj yüzeyinde bulunur
- xy düzlemi taban montaj yüzeyi ile çakışır, böylece x ekseni tabandan öne ve y ekseni sola bakar
- Z ekseni yukarı bakar ve robotun 1. ekseni ile çakışır [67].

2.9.3. Takım merkez noktası

Her zaman olduğu gibi, bir robot üzerinde uç efektör olarak kullanılan alet, robotun flanşına monte edilmiştir. TCP'nin tanımlanması gerekir ancak kendi koordinat sistemine sahip olması gerekir. Herhangi bir zamanda takımın pozisyonu, takım

koordinat sisteminin yönlendirilmesi ile tanımlanır. Bu, koşmaca sırasında robotun hareket yönlerini almak için kullanılan koordinat sistemidir. Bilek koordinat sistemi, araç koordinat sistemine referans verilen sistemdir. Bileğin koordinat sistemi Şekil 2.11.'de gösterilmiştir:



Şekil 2.11. Robot bilek koordinat sistemi [67]

- Bileklik koordinat sistemi montaj flanşlarından biridir.
- TCP kaynağı, montaj flanşının merkezinde bulunur
- Z ekseni daima montaj flanşından dışarıya doğru bakıyordur
- Bir kalibrasyon noktasında, x ekseni zıt doğrultuda, temel koordinat sisteminin orijini yönünde
- Y ekseni sola işaret eder ve temel koordinat sisteminin y eksenine paralel bir eksen olarak görülebilir

2.10. Robot Hareket Denetleyicisi

IRB 120 robotu, ABB tarafından tasarlanmış ve özel amaçlı denetleyici IRC5 tarafından kontrol edilir. Bu kompakt kontrolör, büyük kurulumlarda kullanılan uygulamalara güçlü bir fonksiyon ve hassasiyet getirir. Ayrıca, kontrol ünitesi yerden tasarruf sağlayan daha küçük bir boyuta gelir ve tek fazlı bir güç girişi, tüm sinyaller

için harici konektörler ve dâhili genişletilebilir bir 16 giriş / çıkış sistemi [67] aracılığıyla kolayca devreye alınabilir.

Kumanda, ABB robotlarını sanal olarak programlamak ve kontrol etmek için kullanılan bir grafik kullanıcı ara yüzü yazılımı olan RoboStudio ile birlikte gelen üst seviye programlama dili için destek sağlar. Ayrıca, FlexPendant veya TeachPendant adlı iyi tasarlanmış bir el arayüzü ve denetleyicisi ile de kullanışlıdır. FlexPendant, kontrol ünitesine entegre bir kablo ve konektörle bağlanır [68].

2.10.1. Robot yazılımı ve programlama

ABB robotu, robot sanal bir ortamda programlayabilen bir bilgisayar uygulama yazılımı kullanır. Uygulama RobotStudio denir. Çevrimdışı programlama, oluşturma, simülasyon veya robot istasyonlarına izin verir. RobotStudio'nun çevrimdışı programlama yeteneği, programcıların bir robotun en uygun pozisyonunu bulmasını sağlar ve bu özellik pahalı duruş sürelerini önler ve üretimde gecikmeleri önler [68].

Hem FlexPendant hem de RobotStudio, robotu çevrimdışı simüle etmek için ve robotu programlamak için kullanılabilir. Bununla birlikte, FlexPendant, robot pozisyonunun ve programdaki yol dizilerinin modifikasyonunda en iyi şekilde uygulanırken, RobotStudio çoğunlukla karmaşık programlama için uygundur [67].

BÖLÜM 3. GELİŞMİŞ SİSTEM

3.1. Giriş

Bu bölüm, beklenen sonuçları elde etmek için gerçekleştirilen prosedürün hakkında ayrıntı vermektedir. Deney düzeneğinin adım adım süreci, çözümün nasıl elde edildiğini ve anlaşılmasını sağlamak için ayrıntılı olarak açıklanmaktadır. Projede, sonuçların çıkarıldığı her şeyini açıklayan bir bölümdür.

3.2. Kamera Kalibrasyonu

Kamera kalibrasyonu, görüntü işleme ve bilgisayarlı görme işlemlerinde kaçınılmaz bir işlemdir. Kalibrasyon denilir çünkü bilindiği gibi üreticiden alınan fabrika kalibrasyon parametreleri, bazı uygulamalar için gerektiği kadar doğru değildir. Bu projede kullanılan Kinect sensörü daha sonra kullanıldığı amaç için doğruluğunu ve etkinliğini sağlamak üzere kalibre edildi. Literatürde açıkça vurgulandığı gibi, Kinect'in kalibrasyon parametrelerini hesaplamak için birçok yöntem kullanılabilir. Üreticinin distorsiyon parametreleri doğru değil, bu yüzden distorsiyonu düzeltmek için sensör kalibre edilmelidir.

3.2.1. İçsel kalibrasyon

İçsel kalibrasyon, kameranın dahili parametrelerini bulma işlemidir ve ayrıca distorsiyonu düzeltir. Bir kamera kalibre edilmediğinde, çıkan görüntüler bozulur ve bu nedenle işlem sırasında yanlış sonuçlar verir (Şekil 3.1.)

Şekil 3.1. Orijinal görüntü (solda) ve Düzeltilmiş görüntü (sağda) [43]

J. R Terven'in metodu bu projede Kinect kamerayı kalibre etmek için kullanılmaktadır. Kinect kamerada bir RGB kamera ve her ikisi de aynı anda Terven'in metodu kullanılarak kalibre edilen bir derin kamera bulunur. Terven, MATLAB'daki Kinect V2 kalibrasyonu için Kin2 Toolbox adlı bir araç kutusu MATLAB için tasarladı. Araç kutusu, Microsoft Kinect 2 SDK'yı içine alan sınıflardan ve işlevlerden oluşur. C ++ ile MATLAB çalıştırılabilir dosyalarına dayanan büyüklükler, bu proje için kullanılan sürüm, koordinat haritalama, iskelet izleme, 3B yeniden yapılandırma ve yüz hareketlerini tanıma gibi farklı özelliklere sahip 30 işlev içeriyor [69].

Öncelikle, mex dosyalarının çalışmakta olan kütüphaneye eklenmesi ve bu klasörden çalıştırılması gerekiyordu. Mex dosyaları başarıyla yürütüldükten sonra, aynı anda RGB ve derinlik kamerasının kalibrasyonu için m dosyasının çalıştırılması gerekir. MATLAB aşağıda gösterildiği gibi geri döner; RGB görüntüsü ile harmanlanmış derinlik geri beslenir. Şimdi kod, renkli kameranın kalibrasyonu için c harfine ve derinlik sensörünün kalibrasyonu için c harfine basmanızı ve işlemden çıkmak için d komutunu vermenizi sağlar.

D tuşuna bastığınızda derinlik kamerası parametreleri hesaplanır ve sonuçlar gösterilir. X ve y yönlerinde odak uzunluklarından oluşan iç parametreler, eğriltme parametresi, ana nokta ve 5 bozulma parametresi verilmiştir.

Derinlik kamerası kalibrasyonu hesaplandıktan sonra, derinlik kamerasının kalibrasyonu için c tuşuna basın. Aynı şekilde renkli kamera iç parametreleri de verilmiştir. Ayrıca, Terven'in araç kutusu, RGB kamera ile derinlik kamera ve çeviri vektörü arasındaki dönüşüm matrisini de hesaplar.

Kalibrasyon işleminin sonunda hem derinlik hem de kamera için içsel (kamera) matrislere, daha sonra bir dönüşüm matrisine ve RGB kameradan derinliğe kadar her noktayı haritalayan bir çeviri vektörüne sahibiz.

3.3. Deneysel Kurulum

Aşağıdaki şekilde, projede kullanılan deneysel kurulum Şekil 3.2.'de gösterilmektedir.

Şekil 3.2. İş hücresinin düzenlenmesi

Yukarıdaki şekil burada açıklanmıştır. Kinect kamera sensörü, robotun çalışma hücresinin veya çalışma alanının üzerine yerleştirilir. Oradan çalışma masasına veya bir konveyöre yerleştirilen nesnelerin resimlerini çekebilir. Bir resim çekildikten sonra, nesne algılama algoritması bir anlık görüntü alarak ve bilgisayarla görü ve görüntü işleme için MATLAB'ın araç kutularını kullanarak analiz ederek başlar. Her

çerçeve için renk segmentasyonu ve bölge özellikleri teknikleri kullanılır. Bu, tespit edilen cismin centroid koordinatlarının verilmesine neden olur (x, y). Ancak, Z ekseni, kamera dikey olarak sabitlendiğinden kamera çerçevesinden bilinir. Bu mesafe, deney boyunca ölçülür ve aynı kalır.

Bu projede, dönüşüm matrisinin kamera çerçeve koordinatlarını, robot çerçeve koordinatlarına dönüştürmek için kullanılmadığı bilinmelidir. Bununla birlikte, bunun ışığında, kamera çerçevesinden bulduğumuz koordinatlara robot çalışma alanında oluşturulan çalışma nesnesine atıfta bulunulur. Bu nedenle çalışma nesnesi referans noktası, bulunan koordinatları haritalandırmak için kullanılır, böylece robotun temelini çalışma nesnesi referans noktasına oturtur. Robotun çalışma alanında tespit edilen koordinatlara, robot tabanından değil, yaratılmış olan çalışma nesnesi referans koordinat sisteminden referans verilecektir.

Tespit edilen renkli nesnenin koordinatları daha sonra TCP / IP iletişimi ile MATLAB'dan gerçek robota gönderilir. Robot koordinatları soketlerine ulaştığında, algılanan nesneye hareket eder, onu alır ve başka bir yere yerleştirir. Bir süre döngü hem robotun hızlı programında oluşturulur, böylece robotun soketlerinde herhangi bir giriş varsa sürekli dinler. Soket sürekli olarak dinler ve aynı zamanda koordinatlar MATLAB'dan gönderilir. Kamera, çalışma alanının görüntülerini çekmeye ve koordinatları almak için işlemeye devam ederken, bunları TCP / IP üzerinden robota gönderirken ve robot sürekli gitmesi gereken koordinatları almayı aramaya devam ettikçe devam eder.

3.4. Obje Algılama

Kinect kamera kalibre edildikten ve distorsiyon düzeltildikten sonra, kameranın artık MATLAB'da bozulma olmadan düzgün bir şekilde işlenebilen doğru görüntüler alabileceği anlamına gelir. Nesne algılama, kamera sensörünün kalibrasyonunu takip eder. MATLAB bilgisayar vizyonunda kullanılan araç kutularına sahiptir. Bu çalışmada kullanılan en popüler araç kutuları, Görüntü işleme araç kutusu ve Görüntü toplama araç kutusudur.

Nesne tespiti, görüntü edinme ve işlemeyle başlar. Önce Görüntü edinme araç kutusunun MATLAB'da görüntü elde etmek için farklı tür ve özellikteki kameraların kullanılmasına izin veren fonksiyonları vardır. Bu çalışmada kullanılan Kinect, 'videoinput' işlevini kullanarak MATLAB'da kullanılmak üzere uyumludur. Bir görüntü çekildikten ve bir anlık görüntü alındıktan sonra görüntü işleme başlar.

3.5. MATLAB Seçimi

MATLAB, mühendislikte yaygın olarak kullanılan çok yönlü bir üst seviye programlama dilidir ve endüstride kullanılan pek çok modern disiplini kapsayan araç kutuları içerir [70]. Aşağıdaki nedenler bu projede MATLAB'ın neden seçildiğini açıklamaktadır.

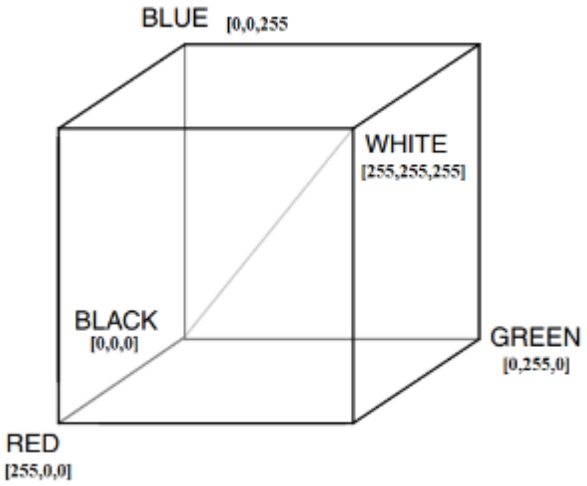
- Görüntü İşleme Araç Kutusu ve Görüntü Alma Araç Kutusu'nu kullanarak kolay görüntü ve video alma ve işleme.
- (GPIB, seri, TCP / IP ve UDP) gibi ana iletişim protokolleri ile harici cihazlarla kolay iletişim. Bu, Alet Kontrol Araç Kutusu fonksiyonları ile elde edilebilir.
- Aynı yazılımda kontrol, simülasyon ve görsel kontrol entegrasyonu,
- Robotun sanal bir modelini geliştirmek için Simulink kullanımı mümkündür
- GUI'lerin kolay uygulanması
- MATLAB Help, MathWorks çevrimiçi desteği ve internetteki mevcut birçok kod örneği farklı türden robotik uygulamaların programlanmasına yardımcı olur [70].

3.6. Görüntü İşleme

Görüntü işleme araç kutusu, algılanan nesnenin özelliklerini bulmak için kullanılabilecek çok sayıda algoritma getirir. Nesnenin bir anlık görüntüsü alındığında, bu görüntü, istediğimizi elde edebilmemiz için onun işlendiği bir dizi algoritmadan geçer. Alan, centroid, küçük eksen, ana eksen, sınırlayıcı kutu ve benzeri gibi özellikler, MATLAB'daki 'regionprops' özelliği kullanılarak belirlenebilir. Ancak bu projede, nesneleri tespit etmek için renk segmentasyonu kullanıldı.

3.6.1. Renk eşiği

Bu projede algılanacak nesne üzerinde bir ayrım yapmak için renk segmentasyonu kullanılır. Renkli nesneyi arka plandan ayırmak için kullanılır. Renk tonları arasındaki Öklid mesafeleri hesaplanır. Bilgisayar grafiğinde, RGB (Kırmızı, Yeşil, Mavi) renkler kullanılır ve Şekil 3.3.'de gösterildiği gibi bir 3D kartezyen sistemi ile temsil edilir:



Şekil 3.3. RGB renk uzayı [71]

[72] tarafından tarif edildiği gibi, bu projedeki renk bölümlendirme işlemi, tüm renkli nesnelerin matrisinin depolandığı bir prosedürü izler. Bu depodan belirli bir rengin eşiğinin bulunduğu görülüyor. Bu nedenle her renk, içinde düşeceği minimum ve maksimum RGB eşik değerlerine sahip olmalıdır. Bu değerler, belirli bir rengin özellikleri özelliklerini filtrelemek için kullanılan değerlerdir. Aşağıda renk maskesi oluşturmak için kullanılan bir denklem verilmiştir:

$$I(i,j)=\begin{Bmatrix}0, P(i,j)<T_{\min}\\ 1, T_{\min}\le P(i,j)\le T_{\max}\\ 0, P(i,j)>T_{\max}\end{Bmatrix} \tag{3.1}$$

Yukarıdaki denklem renk bölünmesi işlemini açıklar. Nesne kendi rengine göre ayıklanır. Farklı renkteki nesneler robotun çalışma alanına ve kameranın görüş açısına konur, ama bu fonksiyon eğer eşik değeri minimum ve maksimum değerlerinin

arasındaysa tek bir rengi seçer. Bu da diğer nesneler sıfır olarak işlenmesi ve böylelikle seçilen renk görülebilirken diğerlerinin görüntüde karanlık görünmeleri demektir.

3.6.2. RegionProps

Regionprops, görüntü işlemede çok faydalı bir fonksiyondur. Ne elde etmek istediğinize bağlı olarak oldukça fazla özellik sunar. 22 farklı özellik hesaplayabiliyor ancak bu proje için sadece üçe odaklanıyor. Çalışma alanındaki kırmızı bir nesneyi bulmak için görüntü işlemede kullanılan bir MATLAB programından bir alıntı, Şekil 3.4.'de bulunmaktadır.

```
redThresh = 0.25;
vid = videoinput('kinect',1);
vid.FramesPerTrigger=5;
vid.TriggerRepeat=2;

% set(vid, 'FramesPerTrigger', Inf);
set(vid, 'ReturnedColorspace', 'rgb')
vid.FrameGrabInterval = 1;

start(vid)
flushdata(vid,'triggers');
flushdata(vid);
fopen(tc);

while(vid.FramesAcquired<=100)
    s=getsnapshot(vid);
    red = s(:,:,1);
    diffFrame = imsubtract(s(:,:,1), rgb2gray(s));
    diffFrame = medfilt2(diffFrame, [3 3]);
    binFrame = imbinarize(diffFrame, redThresh);
    binFrame = bwareaopen(binFrame,300);
    bw = bwlabel(binFrame, 8);

    stats = regionprops((bw), 'BoundingBox', 'Centroid');
    centroids = cat(1, stats.Centroid);
    imshow(s)
```

Şekil 3.4. Görüntü işleme kodunun bir parçası

Yukarıdaki şekil, kırmızı bir nesnenin tespit edileceği görüntü işleme için bir alıntıdır. 0'dan 1'e kadar bir ölçekten kırmızı eşik, 0,25'e yerleştirilir. Bu nedenle, edinilen tüm kareler için kırmızı bir renk arıyoruz.

İlk olarak, "getsnapshot" özelliği kullanılarak bir anlık görüntü çekilir. Bir anlık görüntü sadece FOV kameranın tam renkli RGB görüntüsünü döndürür. Gözün normal olarak ne görebileceğini (Şekil 3.5.'de) gösterir.

Şekil 3.5. RGB ekran görüntüsü kamera tarafından alınmış

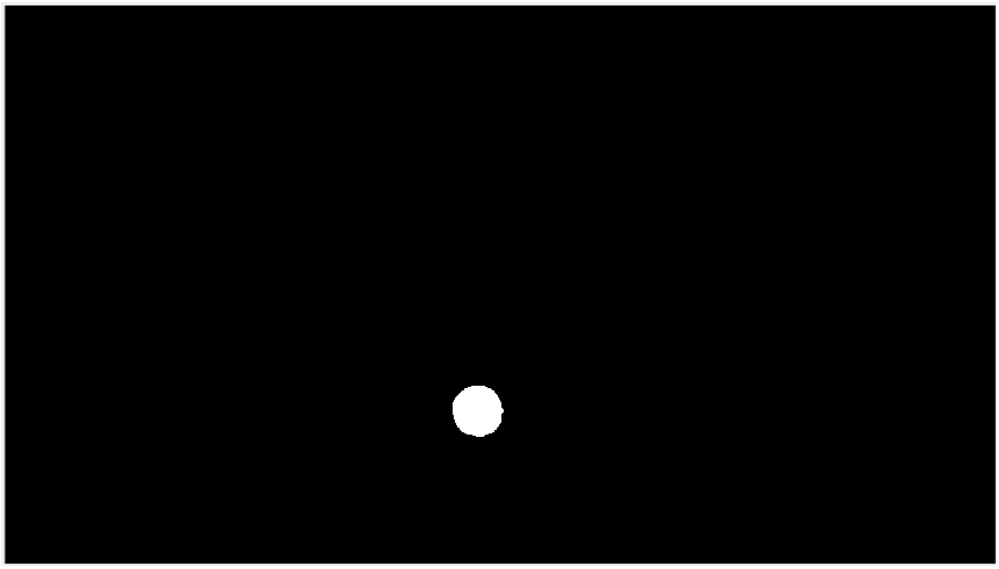
Bunu, görüntünün kırmızı bileşenini alan bir komut izler. Bu komut arka plandaki her şeyin karanlık olmasını sağlar ve görüntünün kırmızı bileşenini Şekil 3.6.'de gösterildiği gibi yalnızca parlak olarak bırakır:

Şekil 3.6. Resmin kırmızı bileşeni

Resmin kırmızı bileşeni alındıktan sonra, medyan filtreleme izler. Medyan filtreleme, görüntünün ‘salt and pepper’ gürültüsünü filtrelemek için kullanılan bir görüntü işleme işlevidir [73]. Görüntü pikselinde piksel tarafından hareket ettirilerek ve her değerin komşu piksellerin medyanı ile değiştirilmesiyle görüntülerden parazit giderme doğrusal olmayan bir yöntemdir. Medyan filtreleme aynı anda gürültüyü giderir ve görüntünün kenarlarını korur ve bu nedenle evrişimden daha etkilidir [73]. Yakından bakacak olursanız, Şekil 3.7.’nin görüntü daha belirgin, net ve gürültü kaldırılmıştır.

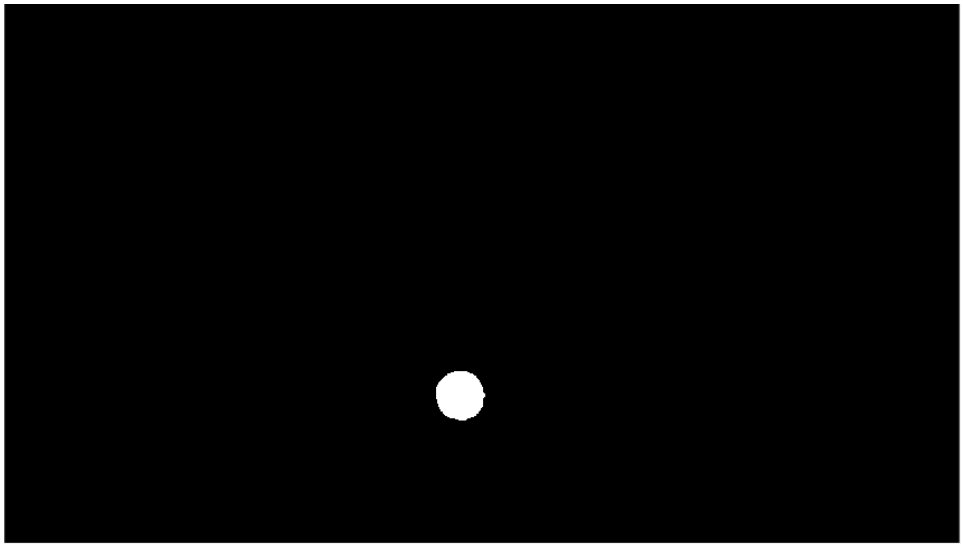
Sekil 3.7. Median filtreden sonra

Görüntü temizlendikten ve salt tuz ve karabiberden filtrelendikten sonra, görüntü ikili görüntüye dönüştürülür ve görüntüdeki diğer her şey siyah hale getirilirken kırmızı bileşen beyaz olarak döndürülür. 'İmbinarize' denilen bir özellik kullanılır. Görüntü Şekil 3.8.'de gibi gelir:



Şekil 3.8. Binarize sonrası görüntü, kırmızı bileşen beyaz renkte gösterilmiştir

Bundan sonra, “bwareaopen” işlevi kullanılır. (Şekil 3.9.)



Şekil 3.9. ‘bwareaopen’dan sonra

Bu işlev daha önce bulunan ikili görüntüden, 300 pikselden daha az olan tüm nesneleri kaldırır. Kodda verilen format binFrame = bwareaopen (binFrame, 300); 300'den daha az olan pikselleri kaldıracağı anlamına gelir. Sonuç, yukarıda gösterildiği gibi başka bir ikili görüntüdür. Bunu, bu proje için sadece iki özelliğe ihtiyaç duyulan bir 'bölge haritası' işlevi izler. Kırmızı nesnemizin büyüklüğüne bağlı olarak sınırlandırılacağı sınırlama kutusu ve robota gönderilecek olan bileşenin centroid'i. Özellikler, Şekil 3.10.’de gibi gösterilir:

Şekil 3.10. Bounding box ve centroid

3.6.3. Objenin centroid'i

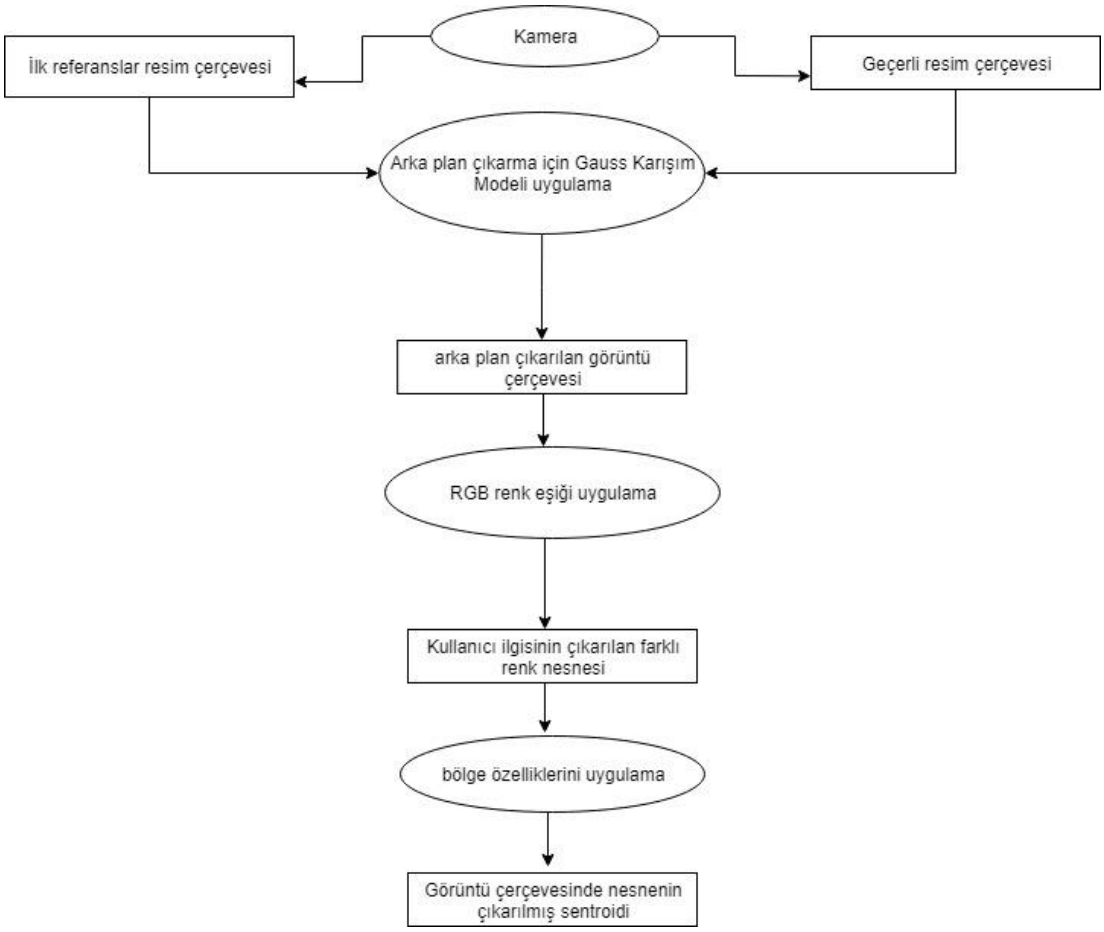
Centroid (x, y) nesnenin kütle merkezinin yatay koordinatıdır ve x koordinatı ile temsil edilir. Kütle merkezinin dikey bileşeni y koordinatıdır [73]. Bu proje için, centroid çok önemli bir özelliktir, çünkü nesnenin tam yerini belirlemede anahtardır. Centroid, aşağıdaki denklem ile hesaplanır;

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i, \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{3.2}$$

Sınırlayıcı Kutu, algılanan bileşenin bağlı olduğu bir dikdörtgendir. Q, görüntü boyutlarının sayısı olduğu ve [sol üst köşe… genişlik] olduğu, Q köşesinin [x, y, z…] şeklinde olduğu ve sol üst köşesini belirten 1x2 vektördür. Sınırlayıcı kutu, genişlik [x genişlik y genişlik…] biçimindedir ve sınırlayıcı kutunun her boyut boyunca genişliğini belirtir [73]. Yukarıdaki MATLAB programından görülebileceği gibi, 'istatistik'teki her nesne için tüm bu özellikler bir "if" ifadesinde doğrulanır. Kırmızı

nesnenin özelliklerinde centroid ve sınırlayıcı kutu bulunur. Sınırlama kutusu, görüntünün üstünde veya altında bir nesnenin noktasını bulmada kilit bir özelliktir.

Görüntü işleme algoritması MATLAB'da geliştirilmiştir ve Şekil 3.11.'de akış şemasında gösterilmiştir:



Şekil 3.11. Özellik çıkarımına dahil olan akış şeması

Objenin koordinatları bulunduktan hemen sonra, robotu nesneye yönlendirmek için şimdi robot kontrol ünitesine gönderilmeleri gerekir. Bu, TCP / IP iletişimi kullanılarak yapılır.

3.6.4. Centroid'ten derinlik almak

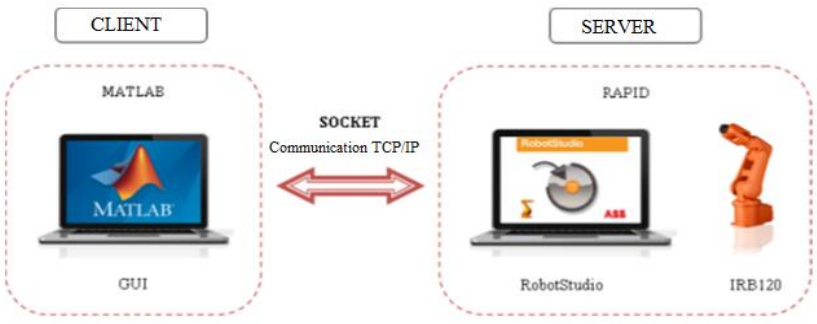
J. R. Terven'in Kinect araç kutusu, Kinect kameranın kalibrasyonu için kullanıldı. Derinlik ve renkli kamera için harici ve dahili kalibrasyon parametreleri bulundu. Üreticinin kalibrasyon parametreleriyle karşılaştırıldığında, bu yöntem neredeyse

üreticilerle aynı sonuçları verdi. Kalibrasyon parametreleri, pratik mesafeye dönüşümün belirlenmesinde büyük rol oynar. Derinliği yakaladıktan sonra nesnenin sensörden milimetre cinsinden bir mesafe olması gerekir. Bu mesafe OpenNI, derinlik (X_d, Y_d) fonksiyon biçiminde bulunur. Kinect için Kin2 araç kutusunda, JR Terven, RGB çerçevesindeki nesnenin centroidine karşılık gelen derinlik koordinatlarını içeren bu OpenNI fonksiyonunu içerir. Sensördeki bu nesnenin derinliğini veya mesafesini milimetre cinsinden bulmak için işlev derinliğinin derinliklerinde (x_d, y_d) kullanılan koordinatlar. Bu kameradan gelen Z koordinattır.

3.7. TCP/IP İletişim

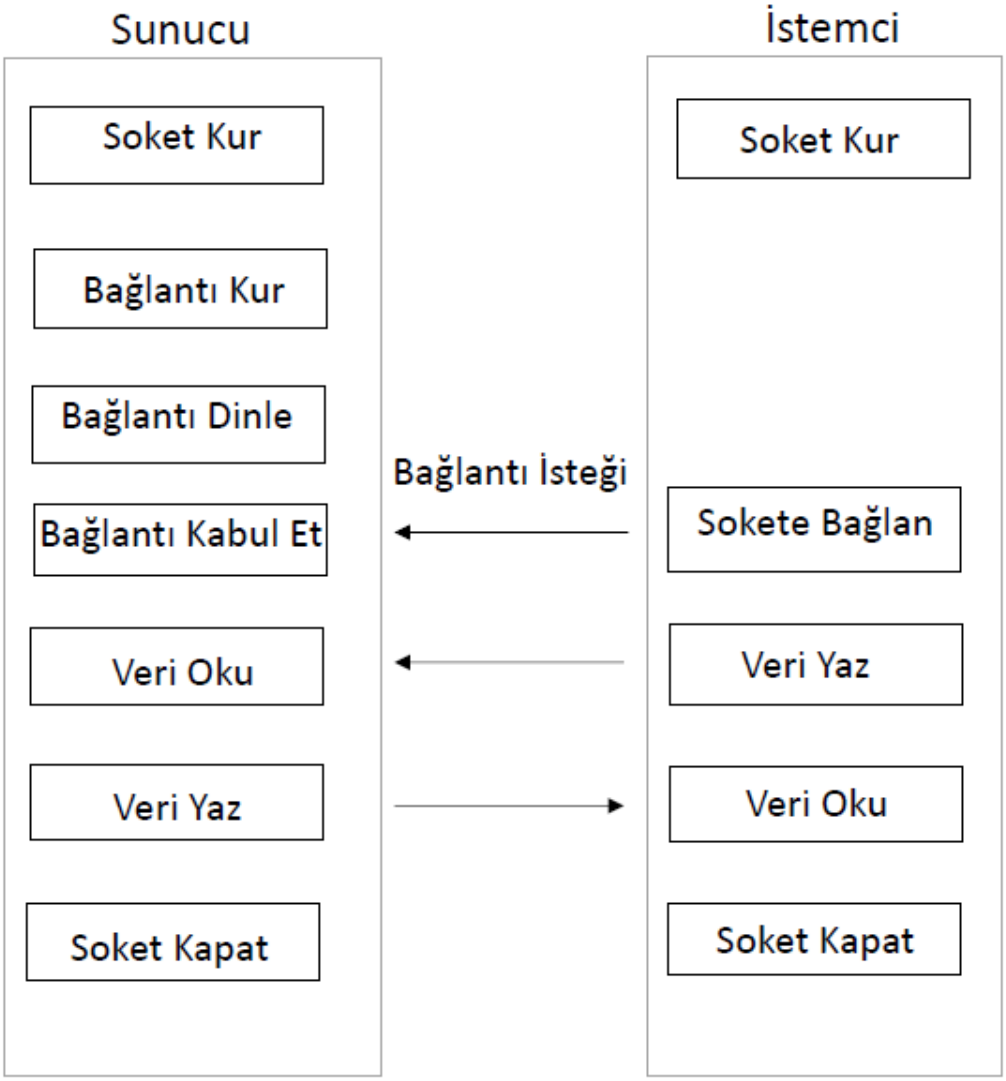
TCP / IP, bir ağda çalışan cihazlar arasında bağlantı kurmak için kullanılan bir İletim Kontrol Protokolü / İnternet Protokolüdür. Çoğunlukla, cihazların birbirlerini ileterek birlikte çalışması gereken endüstriyel otomasyonda kullanılır. Bununla birlikte, endüstride iletişim için kullanılabilecek birçok iletişim protokolü vardır ancak bu proje TCP / IP kullanır.

Bir bilgisayar ve robot arasında bağlantı kurulur. Robot, ABB robotları için Robotstudio adlı grafiksel kullanıcı arayüzünde RAPID adlı bir programlama dili tarafından kontrol edilir. Bağlantı, MATLAB'ın çalıştığı bir bilgisayar ile robotu çalıştıran robot kontrolörü arasındadır. TCP / IP bu proje için en iyi seçimdir, çünkü ABB robotları için kullanılan RAPID robotik programı soket iletişimi için diğer iletişim protokollerinden daha iyi destek sağlar. Getirdiği diğer bir avantaj ABB robotunun kontrolörünün Yerel Alan Ağı (LAN) için bir seçeneğe sahip olmasıdır.



Şekil 3.12. İstemci ve sunucu arasındaki ilişki [74]

Bununla birlikte, bu protokolün tek dezavantajı, RobotStudio'da izin verilen dizi uzunluğunun yalnızca 80 karaktere izin vermesidir. Bu durum, pozisyon ve oryantasyon gerekli olan dize uzunluğunu geçebileceğinden robota aktarılabilecek veri miktarını sınırlar. Dize satır üzerinde olmadığından emin olmak için, dizenin büyük bir paket gönderilmesi yerine küçük paketler halinde gönderilmesi ve sonra tek tek açılması çok önemlidir. Ancak bu programa gecikme getirir ve fazla kod satırı nedeniyle işlemi yavaşlatır. Bu nedenle uygun bir çözüm değildir.



Şekil 3.13. İstemci-sunucu modeli için sıralı diyagram

3.7.1. TCP / IP'ye temel yaklaşım

- Hem istemci hem de sunucuda bir soket yaratılmış. Bir robot kontrol cihazı istemci veya sunucu olabilir,
- Bir bağlantı isteğine hazırlamak için sunucudaki ‘SocketBind’ ve ‘SocketListen’ kullanın.
- Sunucuya gelen soket bağlantı isteklerini kabul etmesi talimatı verilir.
- İstemciden soket bağlantısı isteme
- İstemci ve sunucu arasında veri gönderme ve alma
- Veri aktarımı başarılı olduğunda her iki soket de kapalıdır.

3.7.2. Client olarak matlab

MATLAB'da görüntü işleme gerçekleştirilir ve tespit edilen bir nesnenin koordinatları bulunur. Bu anda, koordinatlar daha sonra TCP / IP üzerinden robota gönderilir. Koordinatlar verildikten sonra, MATLAB'da bir dizi formuna dönüştürülürler.

TCP / IP, İnternet ağı üzerinden iki cihaz arasında bağlantı kurmak için Alet Kontrolü araç kutusunda bulunur. Komut bir IP adresi ve bir frekans veya yerel ana bilgisayar ve uzak ana bilgisayarı giriş bağımsız değişkenleri olarak koyar. Yukarıda gösterildiği gibi, tc, bağlantının kurulduğu bir değişkendir. Bir komut fopen (tc), iletişimi açar. İletişimi açtığınızda, iki cihaz arasında bilgi okuma veya yazma imkânı vardır. Bilgi paylaşan iki cihaza istemci ve sunucu denir ve iletişim kurulduktan sonra aralarında bilgi gönderebilir veya alabilirsiniz.

X, Y, Z koordinatları bulunduğunda, ‘int2str” kullanılarak dizge değerine dönüştürülür ve bir değişkende saklanır. Bundan sonra “fwrite”, dizgiyi internet üzerinden uzak bir cihaza yazmak için kullanılır. İstemci, bu durumda MATLAB, robota bir dizi şeklinde konum bilgisi göndermekten sorumlu olan kişidir.

3.7.3. Server olarak rapid

Robot tarafı, bu projede sunucu olarak çalışan taraftır. Sunucu, istemciden bilgi alan ve onu robot üzerinde yürütendir. İstemci koordinatları dizi değişkeni olarak gönderir ve sunucu bunları dizge olarak alır. Ardından, dizge, robotun bir işlemi yürütmek için gönderildiği koordinatlar olarak anladığı kayan veya çift değerlere dönüştürülür ("str2val", yani dize-değer). Robot bu koordinatları aldıktan sonra, hızlı bir şekilde RAPID'de programlandığı şekilde hareket etmeye ve işlem yapmaya başlar.

Uzak istemciyle soket iletişimi için RAPID'de yazılmış programı, Şekil 3.14.'de göstermektedir.

```
VAR socketdev server;
VAR socketdev client;
VAR string coord;
VAR num i;

PROC main()

SocketCreate server;
SocketBind server, "192.168.125.1", 55000;
SocketListen server;
SocketAccept server, client;
ClearRawBytes data;
WHILE TRUE DO

    coord := "";
    ClearRawBytes data;
    SocketReceive client \Str:=coord;
    ok:= StrToVal(coord,p);
    TPWrite "coord="\pos:=p;
    cartesiantarget:= [p,[0.108861,0.000395,-0.167482,0.986507],[-1,-1,-2,1],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    Path_10;

ENDWHILE


SocketClose server;


ENDPROC
```

Şekil 3.14. Server'da Rapid kodu

3.8. Veri Tipleri

Bu durumda kullanılan veri tipi "socketdev" dir. Soket aygıtı olarak adlandırılır ve bir ağdaki diğer bilgisayarlarla iletişim kurmak için kullanılır.

3.8.1. İstemci için talimatlar

Uzak istemci ile iletişim kuran komutlar ve fonksiyonlar Rapid programlama dilinde yazılmıştır ve Tablo 3.1.'de gösterilmektedir.

Tablo 3.1. Soket programlama istemci komutları

Komut	Açıklama
SocketCreate	Yeni bir soket oluşturur ve Socketdev değişkenini kullanarak socket kurmak için kulanir
SocketConnect	İstemcinin sunucuya bağlanmasını sağlar.
SocketSend	Verileri soket iletişimi yoluyla uzak bir cihaza gönderir. Veriler karakter veya metin olması gerekmektedir
SocketReceive	Diger cihazdan very almak için kullanir. Alinan verilerin karakter veya metin olması gerekmektedir.
SocketClose	Socket kapatmak için kullanır.

Bununla birlikte, SocketClose'in SocketSend'den sonra kullanılmaması ZORUNLU çünkü soket kapatılmadan önce onay gerekmesi gerektiğine dikkat edilmelidir.

3.8.2. Server için talımatlar

RAPID'de, SocketConnect için hem istemci hem de sunucu tasarrufu için aynı talimatlar kullanılır. Ancak, sunucu aşağıdaki talimatları ekler.

- SocketBind: Soketi sunucudaki belirli bir bağlantı noktası numarasına bağlar. Sunucu, hangi bağlantıyı dinleyeceğinizi belirlemek için kullanır. IP adresi fiziksel bir bilgisayarı, port ise o bilgisayardaki bir programa mantıklı bir kanal tanımlar.
- SocketListen: Bilgisayarın sunucu olarak çalışmasını ve gelen bağlantıları kabul etmesini sağlar. "SocketBind" ile belirtilen bağlantı noktasındaki bağlantıyı dinleyecektir.

- SocketAccept: Gelen bir bağlantı isteğini kabul eder. Sunucu tarafından müşterinin isteğini kabul etmek için kullanılır.

İstemci uygulamasından önce başlatılması gereken sunucu uygulamasının, "SocketAccept" komutunun herhangi bir müşteri "SocketConnect" çalıştırmadan önce yürütülmesi gerektiği unutulmamalıdır.

İstemci ve sunucu arasındaki iletişim ilk önce sanal ortamda denenip test edilebilir ve robotun programlandığı şekilde hareket edip etmediğine bağlı olarak değişiklikler yapılabilir. İstemciden gönderilen mesajlar ve sunucudan gelen onay, program gerçek robota gönderilmeden önce ilk önce sanal ortamda test edilir.

Sanal robottan bazı farklılıklar ve farklılıklar nedeniyle gerçek robot üzerinde bazı değişiklikler yapılabileceği görülebilir, ancak yine de robotun verimliliğini ve doğruluğunu arttırmak için yapılması gerekir.

3.9. Robot Kontrolu

Bu projenin ana amacı, herhangi bir 6 DOF robotunu toplama ve yerleştirme işlemleri için kontrol eden, kullanımı kolay bir uygulama bulmaktır. Çalışmanın bu bölümünde sanal platformlardaki programların gerçek robota nasıl aktarıldığı ve bileşenlerin robotik sistem olarak çalışacak şekilde nasıl yapılandırıldığı açıklanmaktadır. Tüm bileşenler ve bunların genel çalışmaya katkısı açıklanmıştır. Ayrıca, robot yolu oluşturma ve yörünge haritalamanın kısa açıklamaları bulunabilir. Kullanılan robot tipi kısaca açıklanmıştır.

3.9.1. ABB 120

ABB IRB 120 robotu hakkında söylenebileceklerin çoğu, ikinci bölümde yapılan literatür taramasında belirtilmiştir. Bununla birlikte, kısaca ABB IRB 120 burada açıklanmaktadır. ABB'nin atölyesinde bulunan en küçük robot ve her işte kullanılabilecek kadar taşınabilir ve esnektir. Küçük hassasiyet ve hareket kabiliyeti

ve az ağırlık ve daha küçük boyutlarla birleştirilmiş mükemmel performans yetenekleri getiren antropomorfik bir 6DOF robotudur.

Bir serbestlik derecesi, bir bağlantının eklemdeki başka bir linke referans olarak yapabileceği bağımsız hareketlerin sayısı olarak tanımlanabilir. Bir robottaki toplam DOF sayısı, içindeki tüm eklemlerin DOF toplamı ile belirlenir. En yaygın kullanılan eklemler prizmatik ve dönerdir ve bu, bir robottaki DOF sayısının toplam eklem sayısıyla eşleştiği anlamına gelir [75].

IRB 120'nin bazı teknik özelliklerini Tablo 3.2.'de gösterilmektedir.

Tablo 3.2. IRB120 robot teknik özellikleri

Fiziksel	Boyutlu robot tabanı	180 x 180mm
	Boyutlu robot yüksekliği	700mm
	Ağırlık	25kg
Özellikler	Pozisyon tekrarlanabilirliği	0.01mm
	Robot montaji	Her acı
	İvme zamani 0-1m/s	0.07s
Varyantlar	Ulaşmak	580
	Yuık	3kg(dikey bilel ile 4kg)
	Armaload	3kg

3.9.2. IRC 5 compact kontrol cihazi

ABB 120, IRC5 beşinci nesil kompakt denetleyici tarafından kontrol edilir. Robotun kontrolünün arkasındaki beyindir. Yanında robotun hareket kontrolünü, kullanılabilirliği, programlanabilirliğini, esnekliği ve güvenliği PC araç desteği ile çoklu robot kontrol ünitesine getirir [75].

Şekil 3.15. IRC5 Kompact Kontroller

IRC5 denetleyicisi, renkli bir dokunmatik ekrana ve bir 3D joystick'e sahip bir kullanıcı arayüzü olan FlexPendant'a sahiptir. Robotun tüm özelliklerinin görülebileceği, yönetilebileceği ve programlanabileceği FlexPendant üzerindedir. Robotun yapmasını isteyebileceğiniz, koşudan programlamaya kadar her şey FlexPendant kullanılarak yapılır. RAPID programlama dili, robotun hareketlerini kontrol etmek için bazı iletişim protokollerini kullanan denetleyici ve diğer bilgisayarlarla iletişim kurmak için kullanılır.

Program, gelişmiş dinamik modelleme tarafından yönetilen bazı ilginç hareket kontrol özellikleri içerir. Kumanda, robotun performansını mümkün olan en kısa döngü süresi (QuickMove) ve hassas yol doğruluğu (TrueMove) açısından optimize eder. Aradaki programcıya danışmadan, robot otomatik olarak öngörülebilir ve yüksek performanslı davranış sergiler [75].

3.9.3. Rapid dil ve RobotStudio

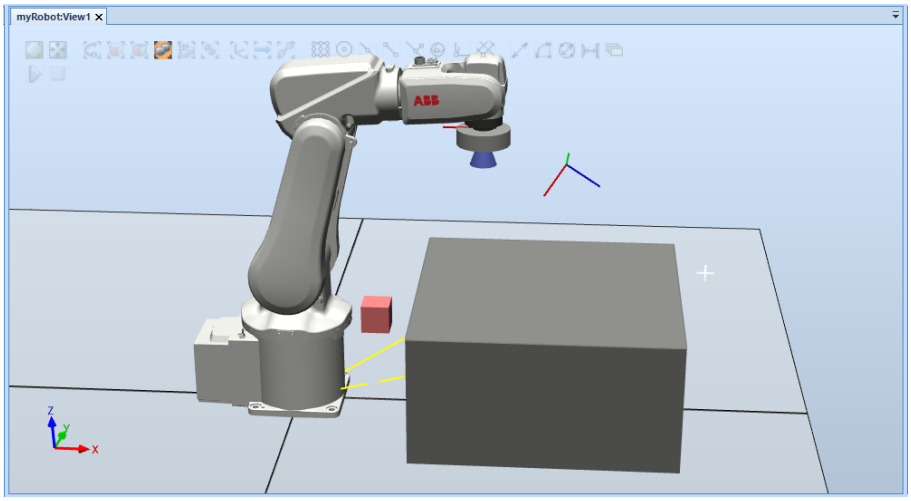
RobotStudio, ABB tarafından geliştirilen çevrimdışı bir programlama ve simülasyon yazılımıdır. Bir PC'ye kurulur ve kullanıcının robotu gerçek atölyede nasıl görünmesini istediğine bağlı olarak programlamasına izin verir. Bir bilgisayarda çalışan ancak gerçek IRC5 denetleyicisi ile aynı olan bir Sanal Denetleyicidir. Gerçek bir üretim hattında kullanılabilecek tüm ABB robotları ve diğer donanımları, kütüphaneleri ve geometrileriyle birlikte gelir. Ayrıca, modelleme sekmesini kullanarak üzerinde çalışmak isteyebileceğiniz herhangi bir nesneyi modellemenizi

sağlar ve SolidWorks gibi diğer modelleme yazılımlarında yapılan geometrileri de içe aktarabilir.

Bu proje için robot ilk olarak RobotStudio yazılımında tasarlandı. ABB IRB120 robotu sistemin veri tabanında bulunur ve çalışma alanına aktarılır. RobotStudio penceresinin araç çubuğundaki modelleme sekmesi kullanılarak dikdörtgen bir çalışma masası oluşturulmuştur. Manipüle edilecek bir nesne yaratıldı ve dikdörtgen tablonun üstüne yerleştirildi. RobotStudio'da pek çok şey yapılmadan önce, bir robot sistemi oluşturulur ve adlandırılır. Dosya sekmesinde, sisteminizin nasıl olacağına karar vereceğiniz bir seçenek seçimi vardır. Seçenekte, sisteminizde kullanmak istediklerinizi ve bu proje için iletişim seçeneğini seçebilirsiniz ve PC Arabirimi seçilmişti.

Neden? Çünkü sistem, bilgisayara da kurulmuş başka bir cihazla veya yazılımla iletişim kurmak için kullanılacaktı. Öyleydi, çünkü MATLAB, Robot stüdyosu ile iletişim kurmak için kullanılacaktı. Program gerçek robota gönderilmeden önce, RobotStudio'da robot istasyonunun oluşturulması gerekiyordu. RobotStudio'nun çalışmasının gerektiği şekilde hareket ettiğinden emin olmak için küresel değişkenler, sabit değişkenler, tasarım, yol oluşturma ve programlama RobotStudio'da yapıldı. Robot stüdyosunda robot istasyonu ve sistemi yarattıktan sonra, soket kullanarak RobotStudio ile iletişim kurmak için bir MATLAB kodu yazılmıştır. İletişim kurulduktan sonra, MATLAB'dan robota koordinatlar gönderildi ve MATLAB'dan gönderilen koordinatlara göre robot hareket ettirecek.

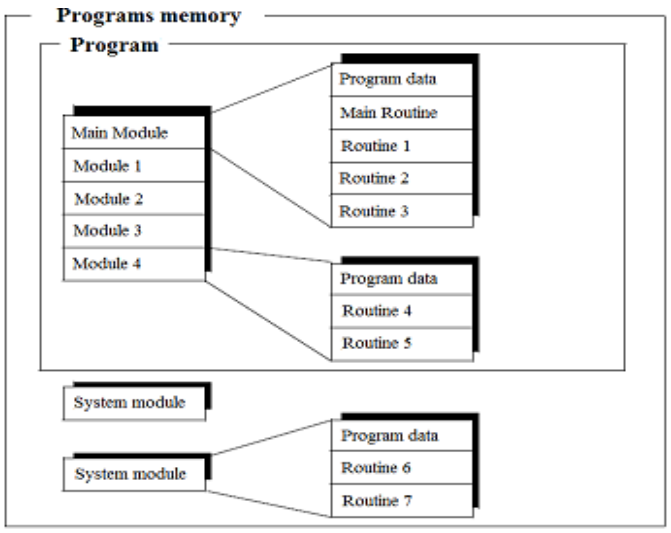
Aşağıdaki şema, RobotStudio platformunda geliştirilen bir IRB 120 şemasını, bir çalışma masasını ve manipüle edilecek nesneyi göstermektedir.



Şekil 3.16. RobotStudio robot sistemi

3.9.4. Rapid dili

RAPID, ABB'nin ABB robotlarını kontrol etmek için kullanılan üst düzey bir programlama dilidir. Herhangi bir diğer programlama dili gibi, robotu kontrol etmek için bilgili bir programcı tarafından kullanılması gereken kendi sözdizimini, talimatlarını, işlevlerini ve veri türlerini içerir. Kontrolörde depolanan modülleri ve kontrolör aşağıda gösterildiği gibi çalışırken çalıştırılması gereken modüller içindeki rutinleri içerir.



Şekil 3.17. Denetleyici dahili program belleğinin dağıtılması [75]

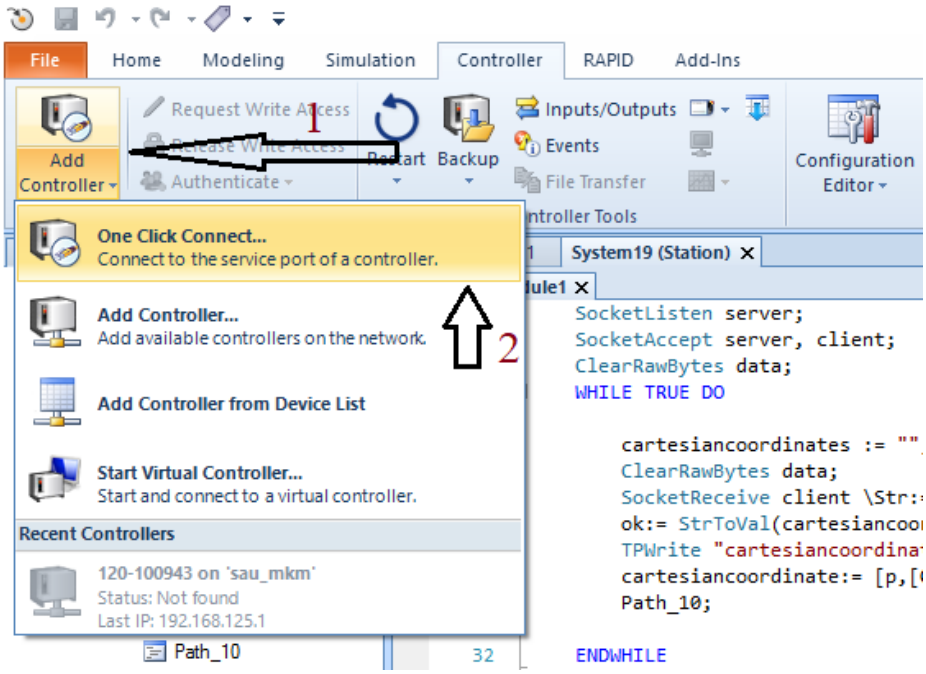
Robot RobotStudio'da geliştirildikten sonra RAPID ile senkronize etmeniz gerekir. Bu komut, grafiksel olarak programlama yaparken bildirdiğiniz tüm değişkenleri ve yarattığınız yolu RAPID'de otomatik olarak oluşturur. RAPID'de, her program belirli bir projenin herhangi bir gereksinimine uyacak şekilde değiştirilebilir. Eşitlemeden sonra soket iletişimi, TCP / IP protokolü kullanılarak MATLAB ile bağlantı oluşturmak üzere programlandı. Bu programın tamamı gerçek robota aktarılmaya hazırdı.

3.9.5. Transfer fonksiyonu işlemi

Bu, RobotStudio'daki bir programı bir bilgisayardaki RobotStudio'dan gerçek robot kontrol ünitesine aktarmak için kullanılan bir fonksiyondur. Bu fonksiyon bir kez kurulduktan sonra, bir programı RobotStudio'dan gerçek kontrol cihazına ve bunun tersi yönde aktarabilirsiniz. RobotStudio tarafından yapılan her değişiklik için kontrolöre güncellenmesi ve gerçek robotta yapılan her değişiklik için RobotStudio'ya aktarılması gerekir. Bu, her iki platformda da tekilliği korumak için yapılır.

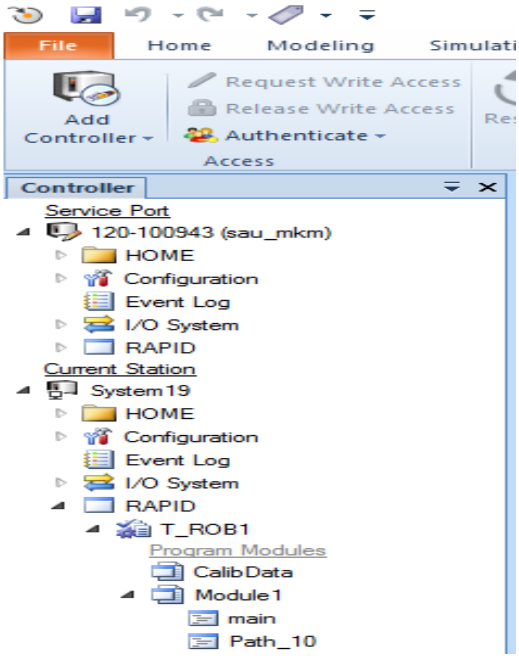
Aşağıda kodu robot stüdyosundan gerçek kontrol cihazına transfer etmek için bir işlem var.

- İlk şey, ethernet kablosunu IRC5 denetleyicisinden RobotStudio ve MATLAB çalıştıran bilgisayara bağlamaktır.
- İlk önce RobotStudio'nun ana sayfasına gitmelisiniz. Denetleyici sekmesi altında, Denetleyici Ekle'yi tıklattığınızda, kaydırma aşağıdaki gibi çıkar:



Şekil 3.18. Denetleyiciye bağlanma

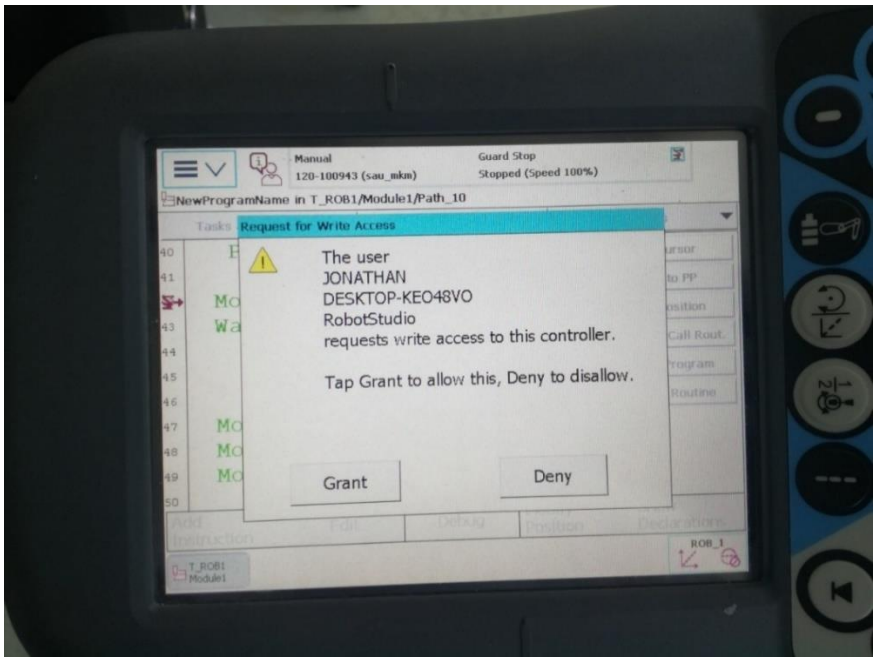
- Bir bağlantıya tıkladığınızda, denetleyici adı ve kullandığı IP adresi ile görünür olacaktır. Daha sonra hızlı bir şekilde IP adreslerinde bir ayar yapılır, böylece RobotStudio'daki IP adresi kontrol cihazında kullanılanla aynı olur. Aşağıda gösterildiği gibi gelir:



Şekil 3.19. Denetleyici bulundu

- İlişki oluşturun. Programcı, ikisini bağlayan bir ilişki yaratarak iki cihazın çalışma kurallarını tanımlar. Bunlardan hangisinin sanal denetleyici ve hangisinin gerçek denetleyici olduğunu tanımlamak için bir ilişki adı verilir.
- Bir ilişki oluşturduktan sonra, aşağıdaki ilişki sekmesi kaynak ve hedef gibi aktarım ayrıntılarını gösterir ve aktarımı yapılandırmanıza olanak tanır. İlişki sekmesi aşağıda gösterilmiştir:

Daha sonra gerçek robota tıklamanız gerekir ve 'request write access' yazılı bir simge vurgulanır. Tıkladığınızda, FlexPendant, kendisiyle iletişim kurmak ve denetleyicisine bir şeyler yazmak isteyen denetleyiciye erişim izni vermenizi isteyen bir mesaj verecektir.



Şekil 3.20. RobotStudio, robot kontrol ünitesine yazma erişimi istiyor

FlexPendant üzerindeki ‘GRANT’ düğmesini tıkladığınızda verileri aktarmaya hazır olacaksınız.

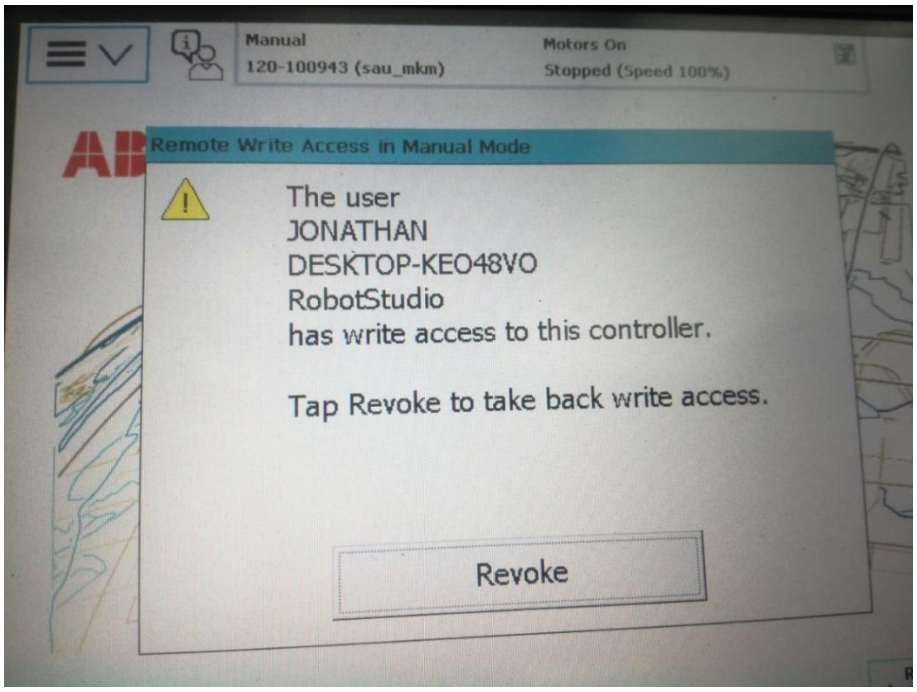
Aktarma konfigürasyonu bölümü size aktarmayı dahil etmeyi veya hariç tutmayı seçebileceğiniz modüllerin öğelerini gösterir. Bir öğenin onay kutusunu seçmek, onu aktarıma dahil eder ve bunun tersi de geçerlidir. Dahası, transferden önce, sanal kontrol

cihazında ve gerçek kontrol cihazında bulunan verileri karşılaştırma hedefini source komutuyla kullanarak karşılaştırabilirsiniz.

Aktarım konfigürasyon bölümündeki onay kutularını işaretleyerek veya işaretlerini kaldırarak aradaki farklara göz atabilir ve hangilerinin korunacağına karar verebilirsiniz. Şu anda gerçek kontrol cihazında mevcut olan verileri transfer sonucunun olası sonucu ile karşılaştır komutunu kullanarak karşılaştır hedefini kullanarak karşılaştırabilirsiniz.

- Yapılandırma tamamlandıktan sonra, aktarma işlemine devam etmek için aktarımı tıklayın. Dahil ettiğiniz veya hariç tuttuğunuz öğelere bağlı olarak transferin bir özeti gösterilecektir. Özeti kontrol ettikten sonra aktarımı tamamlarsınız ve çıkış penceresinde bir aktarım tamamlandı mesajı gösterilir.

FlexPendant, transfer yapıldıktan sonra aşağıdaki pencereyi gösterir.



Şekil 3.21. RobotStudio erişim izni Verdi

‘Revoke’ düğmesine tıkladıktan sonra FlexPendant üzerinde aktardığınız programı görebilirsiniz. Kontrolörlerin her birinde herhangi bir değişiklik yapılması durumunda,

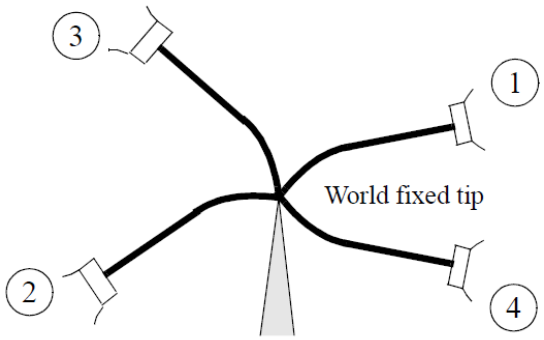
herhangi bir durumda aynı programla çalışabilmeleri için bilgileri birbirlerine hızlı bir şekilde güncellemek akıllıca olacaktır.

3.10. Tool'un Tanımı

Bu projede, önceden tasarlanmış bir vakum alet olarak kullanılmıştır. Normal olarak 6 DOF robotunun uç efektörüne bir alet monte edilir. Robotun düzgün çalışması için aletin tanımlanması gerekir. Takım koordinat sistemi doğru şekilde tanımlanmalıdır, çünkü robotun hareket etmesi için koordinatlar verildiğinde, robotun yönlendirildiği yere hareket eden takım sistemidir.

End efektöre bir takım monte edilecekse, kontrol cihazında manuel olarak yeni bir takım tanımlanmalıdır. Kullanılacak olan aracın oluşturulmasında FlexPendant kullanılarak izlenecek bir prosedür vardır. Aracınızı oluştururken izlenmesi gereken prosedür aşağıda verilmiştir:

- FlexPendant'da ABB sekmesine tıklayın
- Program Verileri penceresine tıkladığınızda, veri türlerinin listesi çıkar
- Takım verilerini seçin
- Yeni'yi kullanarak yeni aracı oluşturun. Bir aracın adını verebileceğiniz ve bazı parametreleri tanımlayabileceğiniz bir iletişim kutusu açılır. Aracı oluşturmayı tamamlamak için Tamam'ı tıklayın.
- Oluşturduğunuz aracın üzerine tıklayın ve aracı tanımlamak için DÜZENLE'ye tıklayın. Aracın tanımlanması için bir yöntem seçmelisiniz. Bu proje, aracı tanımlamak için beş noktanın kullanıldığı TCP ve Z yöntemini kullandı.
- Her nokta için robotu 4 farklı yere hareket ettirin ve Z ekseni yukarı gelecek şekilde sonlandırın. Bundan sonra Tamam'a tıklayın ve siz aracı yaratın.



Şekil 3.22. TCP için yaklaşım noktaları [76]

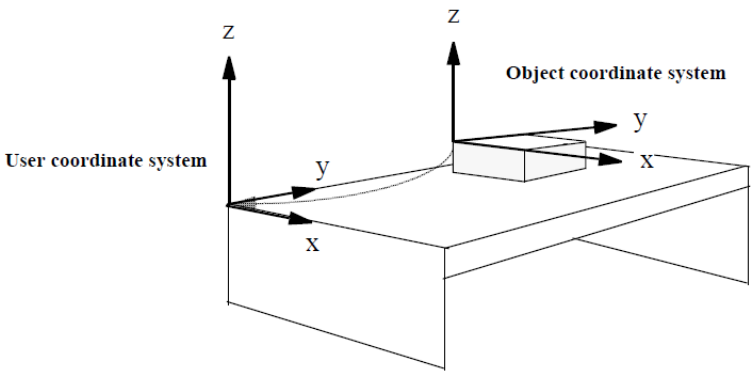
Tam bir oryantasyonun tanımlanması için, Z ekseninde istenen herhangi bir konuma dünya sabit ucuna gitmeniz gerekir. Bu noktalara uzatma noktaları denir. Bununla birlikte, uzama noktalarının, kullanılan son yaklaşma noktasıyla aynı yönelim ile tanımlanması gerektiği belirtilmelidir (ABB, FlexPendant User Guide, 2018). Araç tanımlandıktan ve listelendikten sonra, seçmek ve etkinleştirmek için jog penceresini kullanabilirsiniz.

3.11. Work Obje

Bir work objenin oluşturulması bu projenin en kritik kısımlarından biridir. MATLAB'da görüntü işleme ile robotun hareketi arasındaki bağlantı work objeye uzanır. Asıl itiş gücü, robotun kaynağını tabandan yaratılan work objenin kaynağına kaydırmaktı. Bu orijin daha sonra Kinect kameranın FOV orijini ile eşleştirildi, öyle ki work obje koordinat sisteminin (0,0,0) Kinect kameranın FOV orijini ile aynı noktada olacaktı.

3.11.1. Work objenin oluşturma süreci

Çalışma nesnesini yaratma süreci, neredeyse araç tanımını alma işlemiyle aynıdır. FlexPendant kullanım kılavuzuna göre, programlanan tüm pozisyonlar, nesne çerçevesi, kullanıcı çerçevesi ve dünya çerçevesi ile ilgili olan bir program yer değiştirme çerçevesi ile ilgilidir. Hem kullanıcı çerçevesi hem de nesne çerçevesi bir çalışma nesnesine dahil edilmiştir [76].



Şekil 3.23. Kullanıcı ve obje koordinat sistemi.

Bir work objeyi tanımlarken, aşağıdakilere bakılmalıdır:

- Program Verileri penceresini açın
- wobjdata veri türünü seçin ve verileri göster'i tıklayın
- NEW öğesini kullanarak, çalışma nesnesini tıklayıp adlandırın ve OK tuşuna basın
- Çalışma nesnesi oluşturulduğunda listelenir, ardından üzerine tıklayın ve düzenle'ye basın. Tanımla düğmesine tıkladığınızda, kullanıcı ve nesne çerçevelerini tanımlamanıza izin veren bir pencere açılır.
- Robotu X1 ve X2 pozitif yönlerinde iki nokta ve Y pozitif Y1 yönündeki bir noktadan Y1 döndürmeniz gereken 3 noktalı sistemi seçin. Her harekette, noktayı oluşturmak için Pozisyon Değiştir'i tıklayın. X1'de otomatik olarak bir başlangıç oluşturulur.
- Çalışma nesnesi tanımını bitirmek için Tamam'a tıklayın.

Work obje oluşturulduktan sonra, koşu penceresine geri dönmeli ve oluşturulan çalışma nesnesini seçmeli ve etkinleştirmelisiniz. Çalışma nesnesinin de değişebileceği ve gerektiğinde değerlerin değiştirilebileceği belirtilmelidir.

3.12. Matlab ve Robot

Projenin temel amacı, nesnelerin resmini çekmek için bir Kinect sensörü kullanmak, bazı araç kutularını ve algoritmaları kullanarak görüntüleri işlemek için MATLAB'ı kullanmak, tespit edilen nesnelerin koordinatlarını almak ve ardından koordinatları

robota TCP / IP yoluyla göndermek idi. Robot daha sonra yönlendirilmiş koordinatlara gider, nesneyi seçer ve başka bir yere yerleştirir.

Kinect görüntü elde etmek için kullanıldı ve MATLAB'a bağlandı. Bu projede nesne tespiti için renk segmentasyonu kullanıldı. Koordinatlar bulunduktan sonra başarılı bir şekilde dizgiye gönderildi. Sistemin çalışma nesnesi kameranın FOV'ı ile eşleştiği için, algılanan nesnenin koordinatlarının, X, Y ve Z eksenlerinde çalışma nesnesi kökenli aynı yer değiştirme ofsetleri olduğu anlamına gelir.

Bu projedeki Z bileşeninin sabit ve bilinen olduğu kabul edilir. Bu projede manipüle edilecek nesnelerin aynı yüksekliğe sahip olması nedeniyle aynı kabul edilir. Bulunan koordinatlar, MATLAB ve RAPID'de kurulan TCP / IP soket iletişimi ile robota başarıyla gönderildi. Genellikle beklenen sonuç elde edildi.

BÖLÜM 4. SONUÇLAR

Projenin sonuçları buraya konulacak, analiz edilecek ve sunulacaktır. Bulguların grafik ve çizelgeleri bu bölümde gösterilecektir.

4.1. Deneysel Kurulum

Aşağıdaki şekilde gösterildiği gibi, 50 mm çapında ve 45 mm yüksekliğinde silindirik bir nesne kullanılmıştır. Kameraya dik olarak düz bir yüzeye sıfır yükseklikle yerleştirildi. Kameradan farklı mesafelerde derinlik okumaları alındı. Derinlik ölçümü, robotun algılanan nesnenin doğru yüksekliğine taşındığından emin olmamız gereken ana parametredir. Gerçek mesafeler bir şerit metre kullanılarak ölçülmüştür.

Bu projede ölçülen Kinect derinlik mesafesi, üreticinin derinlik ve diğer her bir parametrenin doğru ölçümleri verdiğine inanılan Windows SDK için Microsoft Kinect ile karşılaştırıldı. SDK ölçümü diğer iki ölçümle karşılaştırıldı; gerçek mesafe ve bu durumda Kinect (proje) olarak adlandırılacak olan proje için tasarlanan Matlab programı.

4.2. Karşılaştırma Tablosu

Sdk, kinect ve gerçek mesafelerin karşılaştırması Tablo 4.1.'de gösterilmektedir.

Tablo 4.1. Comparisons of Actual distance, Kinect(project) with SDK

SDK (mm)	ACTUAL (mm)	KINECT(PROJECT) (mm)	DIFFERENCE (SDK/KINECT) (mm)	DIFFERENCE (SDK/ACTUAL) (mm)
505	501	504	1	4
533	530	532	1	3

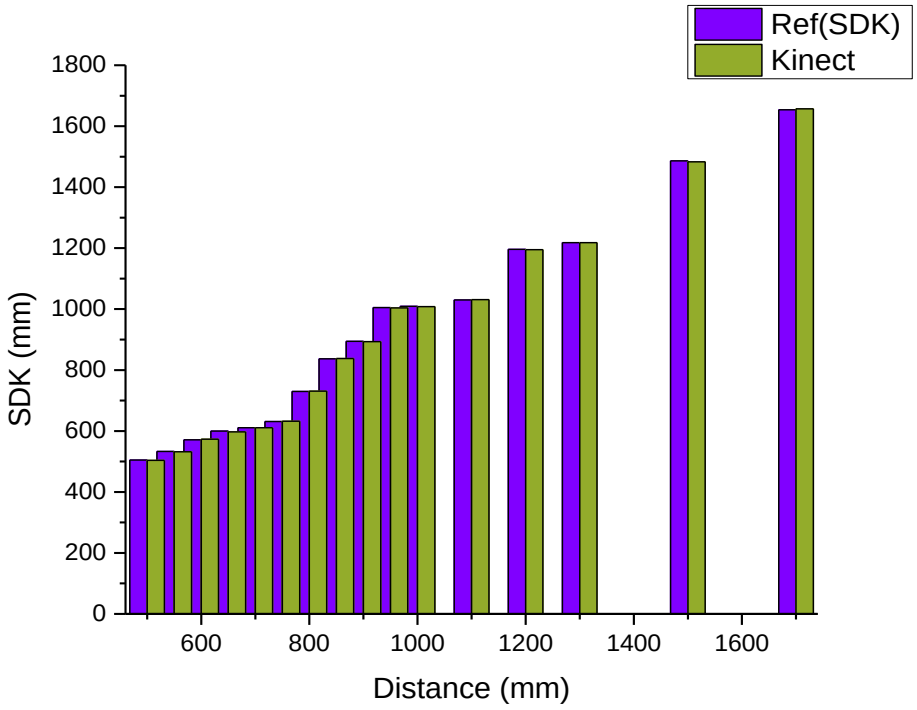
Tablo 4.1. Devamı

SDK (mm)	ACTUAL (mm)	KINECT(PROJECT) (mm)	DIFFERENCE (SDK/KINECT) (mm)	DIFFERENCE (SDK/ACTUAL) (mm)
571	568	573	2	3
600	595	597	3	3
611	606	611	0	5
631	629	632	1	2
730	727	731	1	3
837	835	838	1	2
894	891	893	1	3
1005	1003	1004	1	2
1009	1007	1008	1	2
1030	1027	1031	1	3
1196	1192	1195	1	2
1218	1214	1218	0	4
1482	1480	1483	1	2
1654	1652	1657	3	2

Yukarıdaki tablo SDK doğru ölçümü ve Kinect (Project) yöntemi arasında fazla bir fark olmadığını ortaya koymaktadır. Bu, bu projede kullanılan yöntemin SDK'nın doğruluğuna sahip olduğunu göstermektedir.

4.3. Sonuç Analizi

Kinect ve SDK derinlik ölçümleri Şekil 4.1.'de karşılaştırılmıştır.



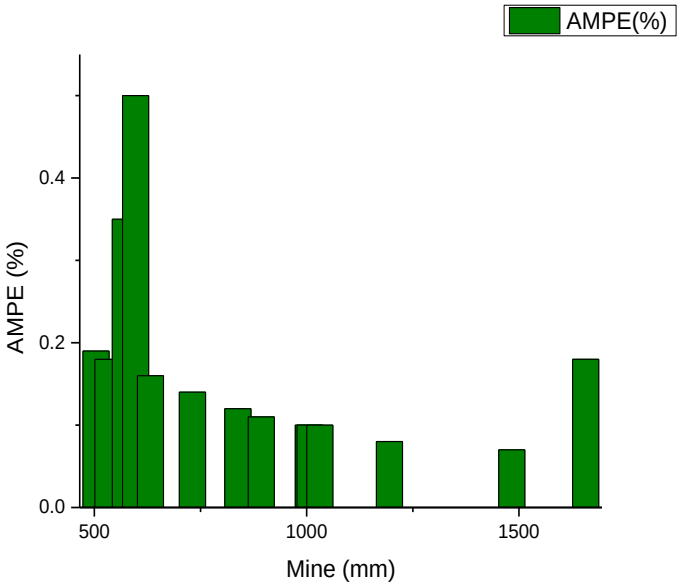
Şekil 4.1. SDK and Kinect (Project) depth measurements

Kinect ve SDK arasında 16 ölçümün ortalama 1.3 mm'si vardı. Şekil 4.2. grafikte, Kinect (Project) ve SDK arasındaki ortalama farkları göstermektedir.

Mutlak ortalama yüzde hatası (MAPE), aşağıdaki formül kullanılarak hesaplandı;

$$\text{fark(mm)} = \begin{pmatrix}\text{SDK}\\ \text{mesafe}\end{pmatrix} - \begin{pmatrix}\text{Kinect}\\ \text{derinlik}\\ \text{mesafe}\end{pmatrix} \tag{4.1}$$

$$\text{AMPE(\%)} = \left(\frac{|\text{fark}|}{\begin{matrix}\text{SDK}\\ \text{mesafe}\end{matrix}}\right) \text{x100\%} \tag{4.2}$$

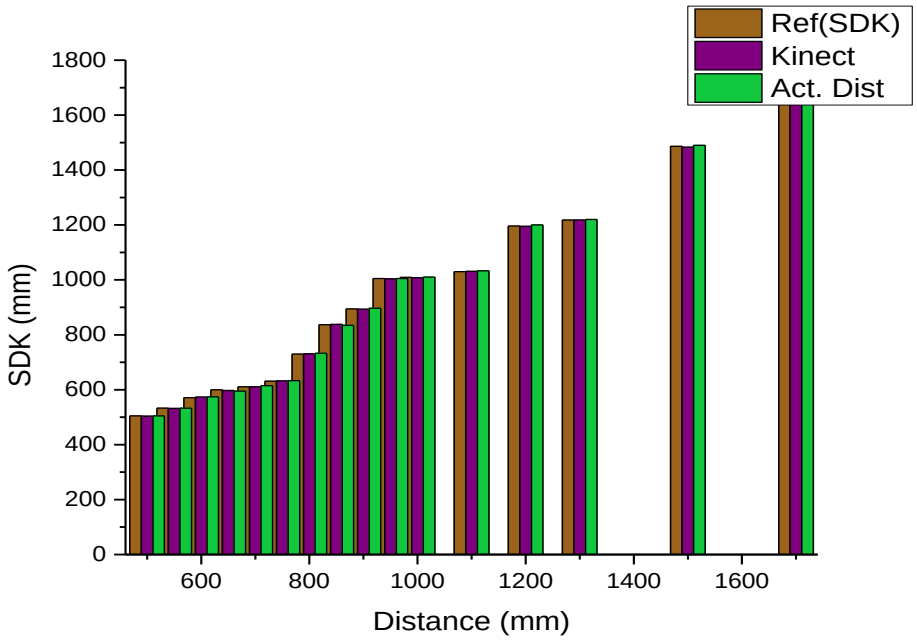


Şekil 4.2. Kinect Vs AMPE ile ölçülen mesafe

Projemizdeki SDK ile Kinect ölçülen derinlik arasındaki ölçümlerin mutlak ortalama yüzde hatası %0,15 idi. Bu neredeyse çok önemli bir hatadır, bu proje için geliştirilen kodun SDK standardına çok uygun olduğunu gösterir.

SDK, Kinect (Proje) ve gerçek mesafe arasındaki fark da analiz edildi. Şekil 4.3'te grafik üç arasında alınan ölçümü göstermektedir. Grafikten, bu üçünün ölçümünün hemen hemen aynı ölçüttüğü görülebilir. Bu nedenle, gerçek ölçümün derinlik ölçüme eşit olarak kullanılabileceği sonucuna kesinleştirebiliriz, çünkü bunlar neredeyse aynıdır.

SDK ile gerçek ölçüm arasındaki ortalama fark sadece 2,8 mm'dir. Bu, SDK ile fiili ölçüm için AMP'nin%0,3 olduğu anlamına gelir.



Şekil 4.3. SDK, Kinect ve gerçek ölçüm arasındaki karşılaştırma

4.4. Conclusion

Bu tezde, 3B nesneyi renk bölümlendirmesine göre bulma algoritması geliştirilmiştir. Ana odak, bir nesnenin derinliğinin kinect sensöründen nasıl bulunabileceği üzerine verildi. Nesnenin yüksekliği, nesnelerin bulunduğu düzlemin derinlikten sensöre olan uzaklığı çıkartarak bulundu. Bu yükseklik robota verildi ve robot başarıyla nesnenin bulunduğu yere taşındı, aldı ve sonra belirtilen bir yere yerleştirdi. Bir nesnenin algılanması, seçilmesi ve yerleştirilmesi için geçen ortalama süre 9 saniyeydi. Nesne tespit edildikten sonraki reaksiyon süresi 53 milisaniyeydi. Robot ekleminin maksimum hızı kullanılırsa, nesne tespitinden toplamaya kadar geçen toplam süre daha az olurdu. Kinect rengi doğru şekilde ayırt edebiliyordu ve robot algılanan nesneye doğrulukla gidebiliyordu. Geliştirilen algoritma başarılı oldu.

BÖLÜM 5. SONUÇ VE ÖNERİLER

5.1. Özeti

Bu çalışmada, endüstriyel bir robotun gerçek zamanlı olarak Windows Kinect V2 için Microsoft Kinect kullanarak kontrol edilmesine yönelik bir yöntem elde edildi. Çok önemli olan, farklı yükseklikteki nesneleri manipüle etmeniz durumunda, bir nesnenin sensörden uzaklaşmasına izin veren derinlik sensörü idi. Projenin hedeflerine, bir rgb ve derinlik sensörleri kullanarak vizyon destekli toplama ve yerleştirme işleminin uygulanması ile ulaşılmıştır. Geliştirilen sistem, birkaç kameranın kullanıldığı veya kameranın robotun son efektörüne yerleştirildiği diğer geleneksel sistemlere kıyasla daha esnek bir sistemdir. Kinect'in büyüklüğü el-göz olarak kullanılmaya uygun değildir çünkü robotun hareketi sensörü bir yerde yeterince tutamayabilir. Bu projenin hedefleri için düşük maliyetli sensör etkin çalışması, sabit bir konuma getirilmesini gerektiriyordu.

Kinect'in RGB HD kamerası kullanılarak nesne tespiti için MATLAB platformunda renk segmentasyonu algoritmaları kullanıldı. Nesnenin o noktadaki sensörden uzaklığı Kinect'in derinlik sensörü kullanılarak hesaplandı. Manipüle edilen nesneler doğada küresel olduğu için, oryantasyon konusu bir sorun olmayacağından tespit edilmesine izin vermek kolaydı.

IRC5 kompakt robot kontrol ünitesi ile el ele çalışan MATLAB ve RobotStudio'da robotik kolla iletişim kurmak için güvenilir bir iletişim sistemi kuruldu ve başarıyla uygulandı. Bu sistem, endüstrideki her türlü toplama ve yerleştirme ve / veya paketleme işlemi için kullanımı kolay ve basittir.

Genelde, hedefleri, sistemi değerlendirmek için kullanılan laboratuar ölçekli bir ortamda gerçekleştirildi. Görüntü işlemenin yol planlamasından robot hareket

kontrolüne entegrasyonundan geliştirilen sistem, endüstride ve benzer donanım platformlarında benzer işlemlerde kullanılabilir.

5.2. Sınırlamalar

Sistemin süreçte bazı eksiklikleri vardı. Diğer tüm ölçüm cihazları gibi, Kinect burada ve orada bazı düzensizliklere sahipti. Işık işleme, görüntü işlemede son derece önemlidir. Farklı aydınlatma koşullarında diğer parametreler ile aynı tutulduğu, farklı sonuçlara ve bazı yanlışlıklara sahip olduğumuz tespit edildi. Işık çoğunlukla derinlik sensörünün çıkışını etkiler. Bu nedenle, iç mekanda çalışırken, Kinect'in verimli çalışması için aydınlatma koşullarının uygun şekilde ayarlanması gerekir.

Sonuçlardan, etkin işlem için nesnenin kamera düzleminden çok uzakta olmaması gerektiği de anlaşılabilir. Bunun nedeni, nesnenin uzaklığı arttıkça düzlemden oluştuğu için, gerçek mesafe ile Kinect derinliği arasındaki fark da artar. Bu aynı zamanda buluntu derinliğindeki hatanın da arttığı anlamına gelir. Sensör bu nedenle verimli sonuçlar için manipüle edilecek nesnelerden uzak olmayan bir mesafeye yerleştirildi.

5.3. Gelecekteki İşler için Öneriler

Proje sırasında gerçekleştirilen eksiklikler için, bu konuda veya sürekli gelişim için bu sistemde yapılabilecek gelecekteki çalışmalar için aşağıdakiler önerilmektedir.

- Görme sistemi tarafından oluşturulan, manipüle edilecek nesnenin bir 3D modelini dahil ederek sistemin daha esnek olması sağlanabilir.
- Oluşturulacak sistemin işlevine bağlı olarak, Kinect sensörü, nesnenin yöneliminin farklı bakış açılarından tahmin edilmesini sağlamak için robota veya başka bir manipülatöre monte edilebilir. Bununla birlikte, Kinect, boyutundan dolayı doğruluk nedeni ile sıkıca monte edilmelidir. Bunu akılda tutarak, sensörün hareketsiz şekilde monte edilmiş konumda kaldığından emin olmak için kullanılacak hız optimum olmalıdır.

- Işığın değişmesi, görüş sisteminin tutarsız bir şekilde çalışmasına neden olduğundan, sabit veya sabit bir aydınlatma test ortamı oluşturulmalıdır. Bu da tekrarlanan kalibrasyon gerektirebilir.
- Sistem sadece toplama ve yerleştirme amaçlı endüstriyel amaçlar için kullanılamaz, aynı zamanda trafik denetimi, özerk yol bulma ve eşzamanlı lokalizasyon ve haritalama (SLAM) gibi alanlarda da uygulanabilir. Derinlik özelliklerini kullanarak nesnelerin izlenmesi, özerk görmeye dayalı yol bulma sistemleri ve askeri gözetim sistemleri geliştirmek için mobil robotlarda kullanılabilir.

KAYNAKLAR

[1] J. Hill, W. Park, "Real time control of a robot with a mobile camera," *9th International Symposium on Industrial Robots,* p. 233–246, 1979.

[2] K. Rezaie, S. Nazari Shirkouhi, S.M. Alem, "Evaluating and selecting flexible manufacturing systems by integrating data envelopment analysis and analytical hierarchy process model," *Asia International Conference on Modelling and Simulation,* pp. 460–464,, 2009.

[3] K. Hashimoto, "Visual Serving: Real Time Control of Robot Manipulators Based on Visual Sensory Feedback,," 1993.

[4] Hutchinson, F. , Chaumette, S., "Visual servo control basic approaches," *Robotics & Automation Magazine, IEEE, vol. 13,* pp. 82-90, 2006.

[5] Hutchinson, F., Chaumette S., "Visual servo control. ii. advanced approaches [tutorial]," *Robotics & Automation Magazine, IEEE, vol. 14,* p. 109–118, 2007.

[6] D. Kragic, and H. I. Christensen, et al.,, "Survey on visual servoing for manipulation," *Computational Vision and Active Perception Laboratory Fiskartorpsv, vol. 15,,* 2002.

[7] H. Wu, W. Tizzano, T. Andersen, N. Andersen, O. Ravn, "Hand-Eye Calibration and Inverse Kinematics of Robot Arm using Neural Network," *Springer,* p. 581–591, 2013.

[8] H. Wu, L. Lu, C.-C. Chen, S. Hirche, K. Khnlenz, "Cloud-based networked visual servo control," *I E E E Transactions on Industrial Electronics, vol. 60, no 2,* pp. 554 – 566,, 2013.

[9] Meyer, R. D. Schraft and C., "The need for an intuitive teaching method for small and medium enterprises," *ISR Robotik, Germany,* 2012.

[10] B. Akan, "Human Robot Interaction Solutions for Intuitive Industrial Robot Programming," *Västerås: Mälardalen University, 2012.,* 2012.

[11] J. A. Marvel, R. D. Eastman, G. S. Cheok, K. S. Saidi, E. R. Messina, B. Bollinger, P. Evans, J. Guthrie, E. Hershberger,C. Martinez, K. McNamara, and J. Wells,, "Technology readiness levels for randomized bin picking," *Performance Metrics for Intelligent Systems (PerMIS),*, 2012.

[12] N. Zucch, "Machine Vision," The fairmont Press Inc, 1987.

[13] P. Piccinini, A. Prati, R. Cucchiara, "Real-time object detection and localization with sift-based clustering," in *Image and Vision Computing*, 2012.

[14] Inoue, Y., Shirai H., "Guiding a robot by visual feedback in assembly tasks," *Pattern Recognition, 5*, pp. 99-108, 1973.

[15] Sanderson, A., Wess, L., "Image-based servo control using relational graph error signals," *Proc. IEEE 1980*, pp. 1074-1077, 1980.

[16] F. Conticelli and B. Allotta, "Two-level visual control of dynamic look-and-move systems," *Proceedings of the IEEE 2000 International Conference on Robotics and Automation, San Francisco, CA, USA*, p. 3784–3789, 24-28 April 2000.

[17] M.-Y. Liu, O. Tuzel, A. Veeraraghavan, Y. Taguchi, T. K. Marks, R. Chellappa,, "Fast object localization and pose estimation in heavy clutter for robotic bin picking," *The International Journal of Robotics Research 2012*, 2012.

[18] MSDN, "Kinect for Windows sensor components and speci," 2015., Erişim Tarihi: 22.02.2019.

[19] Microsoft, "Kinect forWindows SDK," *Microsoft*, 2014.

[20] A. Payne, A. Daniel, A. Mehta, B. Thompson, C. S. Bamji, D. Snow, H. Oshima, L. Prather, M. Fenton, L. Kordus, et al, " 3d time-of-flight image sensor with multifrequency photo-demodulation up to 130mhz and 2gs/s adc," *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International,*, pp. 134-135, 2014.

[21] D. Lowe, "Distinctive image features from scale invariant keypoints," *International Journal of Computer Vision (IJCV),*, pp. 91-110, 2004.

[22] T. A. Y. Y. Baştanlar Y, "Improved Sift Matching for Image Pairs with Scale Difference," *IET Electronic Letters*, 2010.

[23] K. N. Houshangi. A.J., "Real-time vision feedback for servoing robotic manipulator with self-tuning controller," *IEEE Transactions on Systems, Man and Cybernetics,* pp. 134 –142,, Jan/Feb 1991.

[24] Sanderson, Alison E. Hunt, Arthur C., "Vision-based predictive robotic tracking of a moving target," Pittsburgh, PA, 1982.

[25] Yu Huang, Thomas S. Huang,Heinrich Niemann, "Segmentation-based object tracking using image warping and kalman filtering," 2002.

[26] Alper Yilmaz, Omar Javed, Mubarak Shah, "Object track ing: A survey," *Acm Computing Surveys (CSUR),,* no. 38(4):13, 2006.

[27] Aggarwal, J. K., Cai, Q., "Human motion analysis: A review," *Comput. Vision Image Understand,* no. 73,3, pp. 428-440, 1999.

[28] D. M. Gavrila, "The visual analysis of human movement: A survey," *Comput. Vision Image Understand,* vol. 1, no. 73, pp. 82-98, 1999.

[29] Moeslund, T., Granum, E, "A survey of computer vision-based human motion capture," *Comput. Vision Image Understand,* vol. 3, no. 81, pp. 321-268, 2001.

[30] Alan J Lipton, Hironobu Fujiyoshi, Raju S Patil, "Moving target classification and tracking from real -time video," in *Fourth IEEE Workshop on Applications of Computer Vision,* 1998.

[31] M. I. A. Blake, "Contour track ing by stochastic propagation of conditional density," in *European Conference on Computer Vision*, 1996.

[32] Grimson, Chris Stauffer ,W Eric L, "Adaptive back ground mixture models for real -time track ing," *Vision and Pattern Recognition,* vol. 2, 1999.

[33] Ya Liu, Haizhou Ai, Guang-you Xu,, "Moving object detection and track ing based on background subtraction," *International Society for Optics and Photonics, 2001.,* pp. 62-66, 2001.

[34] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transaction of the ASME—Journal of Basic Engineering,* pp. 35-45, March 1960.

[35] Mathworks, "Image Processig Toolbox User's Guide," 2014. http://www.mathworks.com/help/pdf_doc/images/images_tb.pdf..,Erişim Tarihi:20.06.2019.

[36] Mathworks, "Image segmentation," 2019. http://www.mathsworks.com., Erişim Tarihi: 21.06.2019.

[37] Harika Nanduri, Dr. Manoj Soni, "VISION CONTROLLED PICK AND PLACE OF MOVING OBJECT BY 3R ROBOT," *International Journal of Advance Research and Innovative Ideas in Education,* vol. 2, no. 4, 2016.

[38] A. Vision, "Blob Analysis," 2019. https://docs.adaptive-vision.com., Erişim Tarihi: 23.07.2019.

[39] Mathworks, "MATLAB and Simulink for Technical Computing," 2015. http://www.mathworks.com/.., Erişim Tarihi: 23.09.2019.

[40] C. Yang, S.Amarjyoti, X. Wang, Z. Li, H. Ma, C.Y. Su, "Visual Servoing Control of Baxter Robot Arms with Obstacle Avoidance Using Kinematic Redundancy," *Researchgate,* 2015.

[41] Davis, Daniel F. Dementhon, Larry S., "Model-based object pose in 25 lines of code," *Int. J. Comput. Vision,* vol. 15, no. 1-2, p. 123–141, 1995.

[42] He, Aihua Chen, Bingwei, "A camera calibration technique based on planar geometry feature.," *Mechatronics and Machine Vision in Practice, M2VIP 2007.*, pp. 165 –169,, 2007.

[43] Mathworks, 2019: http://www.mathworks.com., Erişim Tarihi: 12.09.2019.

[44] C. C. Ntina, "Stereo Calibration of Depth Sensors with 3D Correspondences and Smooth Interpolants," 2013.

[45] Q. Zhang, R. Pless, "Extrinsic calibration of a camera and laser range finder (improves camera calibration)," *IROS,* vol. 3, p. 2301–2306., 2004.

[46] R. Unnikrishnan, M. Hebert,, "Fast extrinsic calibration of a laser rangefinder to a camera," Robotics Institute, Pittsburgh, 2005.

[47] J.Y.Bouguet, "Matlab calibration tool," http://www.vision.caltech.edu/bouguetj/calib_doc/.,Erişim Tarihi: 12.01.2019.

[48] OpenCV, 2019, http://www.opencv.org/..,Erişim Tarihi 14.01.2019.

[49] Argyros, M., Lourakis, A., "The design and implementation of a generic sparse bundle adjustment software package based on the Levenberg-Marquardt algorithm," Institute of Computer Science-FORTH, Crete, Greece, 2004.

[50] S. Fuchs, G. Hirzinger, "Extrinsic and depth calibration of ToF cameras," *CVPR, 2008*, pp. 1-6, 2008.

[51] M. Lindner, A. Kolb,, "Calibration of the intensity-related distance error of the PMD TOF-Camera," *Intelligent Robots and Computer Vision XXV*, vol. 6764, 2007.

[52] E. Trucco, A. Verri, Introductory Techniques for 3-D Computer Vision, 3rd ed., USA: Prentice-Hall Inc, 1998.

[53] J. J. Craig, Introduction to Robotics Mechanics and Control, 3rd ed., Pearson Prentice Hall, 2005.

[54] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, "Robotics Modelling, Planning and Control," *Springer*, 2010.

[55] S. R. Nicolas, E. D. Chaillet, Microrobotics for Micromanipulation, Wiley, 2013.

[56] C. Liang, F. Wang, B. Shi, Z. Huo, K. Zhou, Y. Tian, D. Zhang, "Design and control of a novel asymmetrical piezoelectric actuated microgripper for micromanipulation," *Sensors and Actuators A: Physical*, vol. 269, pp. 227-237, 2018.

[57] L. Wang, L. Ren, J. K. Mills, W. L. Cleghorn, "Automated 3-D Micrograsping Tasks Performed by Vision-Based Control," *IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING*, vol. 7, no. 3, pp. 417-426, 2010.

[58] Hu, S., Ma, Z., "Hand-eye calibration," *Springer: Computer Vision*, p. 355–358, 2014.

[59] R. Horaud, F. Dornaika, "Hand-eye calibration," *The International Journal of Robotics Research*, vol. 14, no. 3, p. 195–210, 1995.

[60] R. Y. Tsai, "A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses," *IEEE Journal of Robotics and Automation*, vol. 3, no. 4, pp. 323–344,, 1987.

[61] T. Wiedemeyer, "IAI Kinect2," *Institute for Artificial Intelligence, University Bremen*, 2015.

[62] C. Amon, F. Fuhrmann, F. Graf,, "Evaluation of the spatial resolution accuracy of the face tracking system for kinect for windows v1 and v2," in *Proceedings of the 6th Congress of the Alps Adria*, 2014.

[63] V. Lippiello, B. Siciliano, L. Villani, "Eye-in-hand/eye-to-hand multi-camera visual servoing," *44th IEEE Conference on Decision and Control, 2005 and 2005 European Control Conference,* p. 5354–5359., 2005.

[64] T. Heikkil¨a, M. Sallinen, T. Matsushita, F. Tomita, "Flexible handeye calibration for multi-camera systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000).,* 2000.

[65] A. Coste, "3D Computer Vision: Geometric Camera Models and Calibration," 2013.

[66] Yoo, Trong-Hop, Do, Myungsik, "An in-Depth Survey of Visible Light Communication Based Positioning Systems," *Sensors,* 2016.

[67] ABB, 2018. http://www.newabb.com., Erişim Tarihi: 14.07.2019.

[68] ABB, 2012. http://www.newabb.com., Erişim Tarihi 14.07.2019.

[69] Juan R. Terven, Diana M. Cordova, Kin2 User Guide, 2016.

[70] S. Attaway, MATLAB: a practical introduction to programming and problem solving, 2nd ed., Elsevier Inc,, 2012.

[71] Pramod Kumar Thotapalli, CH. R. Vikram Kumar, B.ChandraMohana Reddy, "Feature Extraction of Moving Object over a Belt Conveyor Using Background Subtraction Technique," in *Proceedings of the 10th International Conference on Precision, Meso, Micro and Nano Engineering,* Chennai, India, 2017.

[72] P.B.Vijayalaxmi, Rohan Putta, Gayatri Shinde, Punit Lohani, "Object Detection Using Image Processing For An Industrial Robot," *International Journal of Advanced Computational Engineering and Networking,* vol. 1, no. 7, pp. 21-26, 2013.

[73] MathWorks, "Image Processig Toolbox User's Guide,," 2014.

[74] Fernández, Nicolás Blanco, "“Generación de trayectorias y evitación de obstáculos para el robot IRB120 en entorno Matlab”," *UNIVERSIDAD DE ALCALÁ,* p. 47, 2015.

[75] I. T. Balmori, "VISION-BASED GUIDANCE SYSTEM FOR A 6-DOF ROBOT," *Hamk University of Applied Sciences,* 2014.

[76] ABB, FlexPendant User Guide, Vasteras, Sweden: ABB Robotics Products AB, 2018.

[77] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations.," in *Proc. 7th IEEE International Conference on Computer Vision*, 1999.

ÖZGEÇMİŞ

Tichaona Jonathan Makomo, 29.05.1991'da Seke, Zimbabve'de doğdu. İlk, orta ve lise eğitimini Zimbabve'de tamamladı. 2009 yılında St Alberts Yüksek Okulu'nden mezun oldu. 2010 yılında başladığı Chinhoyi Üniversitesi Mekatronik Mühendisliği Bölümü'nü 2014 yılında bitirdi. 2017 yılında Sakarya Üniversitesi Mekatronik Mühendisliği Bölümü'nde yüksek lisans eğitimine başladı ve 2020 yılında bitirdi.