T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

RASPBERRY Pİ KARTI KULLANARAK NESNE TESPİT VE TAKİP ROBOTUNUN TASARIMI VE MODELLENMESİ

Taner YILMAZ

YÜKSEK LİSANS TEZİ

Mekatronik Mühendisliği Anabilim Dalı
Danışman : Doç. Dr. Mehmet ÇAVAŞ
İkinci Danışman: Dr. Öğr. Üyesi Muzaffer ASLAN

OCAK 2019

T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

RASPBERRY Pİ KARTI KULLANARAK NESNE TESPİT VE TAKİP ROBOTUNUN TASARIMI VE MODELLENMESİ

YÜKSEK LİSANS TEZİ

Taner YILMAZ

161134101

Tez Kitapçığının Enstitüye Verildiği Tarih: 18.12.2018
Tez Teslim Tarihi: 23.01.2019

Tez Danışmanı : Doç. Dr. Mehmet ÇAVAŞ (Fırat Üniversitesi)

Diğer Jüri Üyeleri : Doç. Dr. Ömür AYDOĞMUŞ (Fırat Üniversitesi)

Dr. Öğr. Üyesi Mehmet ÜSTÜNDAĞ (Bingöl Üniversitesi)

OCAK-2019

ÖNSÖZ

Bu tez çalışmasında literatür taraması yapılarak robotların gelişim süreci araştırılmış, nesne takip robotunun tasarımı ve modellemesi yapılarak farklı alanlarda kullanımı amaçlanmıştır.

Tez konusu seçiminde ve yaptığım çalışmalarda desteklerini esirgemeyerek her konuda bana yardımcı olan tez danışmanlarım Doç. Dr. Mehmet ÇAVAŞ ve Dr. Öğr. Üyesi Muzaffer ASLAN ‘a teşekkürlerimi sunarım.

Taner YILMAZ
ELAZIĞ-2019

İÇİNDEKİLER

ÖZET

İnsan yeteneklerini taklit eden ve onun gibi davranan makineler yapmak bilim insanları için her zaman heyecan verici bir araştırma konusu olmuştur. Başlangıçta, bilgisayarların hesap yapabilme yetenekleri dikkate alınarak, düşünme, işitme, konuşma, tanıma, görme gibi yeteneklerinde geliştirebileceği düşünülmüş ve bu düşüncenin temelinde, insan gibi dış dünyayı algılaması, elde ettiği verileri akıllı bir biçimde işleyerek çeşitli problemlere çözüm üretmesini sağlamak hedeflenmiştir.

Bilgisayarlar duyduklarımızın görüntüsünü çizebilir veya bir kokunun şiddetine göre bir görsel şekil oluşturabilmektedir. Bu özellikler bilgisayar görmesi konusunun temelini oluşturmaktadır. Bu bağlamda bilgisayarlı görmenin temel amacı, algılanan verilere göre kabul edilebilir, makul ve mantıklı kararlar verebilmesidir. Nesne olarak belirtilen ve görsel olarak algılanabilen her şeyin tespiti, takibi vb. özellikleri kapsamaktadır. Bilgisayarın nesne üstünde yaptığı görmeye ait bilgiler ışığında belirli bir algoritma ile nesneleri tanıma, ayırt etme, sınıflandırma, belirleme, bulup getirme, değiştirme, doğrulama gibi durumları ifade eden bütün işlemler yapılabilmektedir.

Son yıllarda robotik sistemler alanındaki çalışmalara bakıldığında, insanlarla satranç oynayan, fabrikalarda vida sıkan, imla hatalarını düzelten, parfüm koklayan, hastalıkları teşhis edebilen vb. birçok alanda farklı işlemleri gerçekleştirebilen sistemler geliştirilmiş ve bu sistemler güvenle kullanılmakla birlikte yakın gelecekte de daha geniş bir yelpazede kullanılacağı ön görülmektedir.

Video akışında, hareketli bir nesnenin gerçek zamanlı algılanması ve takibi oldukça güçtür. Bu tez ile, Raspberry Pi kartı ve Python yazılımı ile OpenCV görüntü işleme kütüphanesinden, anlık video görüntüsü ile renk uzayından faydalanılarak nesne tespit işlemi gerçekleştirilmiştir.

Anahtar Kelimeler: Görüntü işleme, Nesne Tespit, Raspberry Pi, Nesne Takip Robotu

SUMMARY

Design And Modeling Of Object Detection And Tracking Robot With Using Raspberry Pi Card

Scientists have always wanted to make machines that mimic human abilities and act like it. This has been an exciting research topic for scientists. Initially, computers were able to use their computing capabilities. Then, it was thought that it could improve in the abilities of thinking, hearing, speaking, recognition and vision. At the basis of this idea, it is aimed to be able to perceive the external world as a human being and to provide the solution of various problems by processing the obtained data in a clever way.

The computer can draw the image of what we hear or create a visual shape according to the severity of a smell. These features form the basis of computer vision. In this context, the main purpose of computer vision is to make reasonable and reasonable decisions that can be made according to the perceived data. It is anything that can be visually perceived as an object. It includes features such as detection, tracking. Computer vision consists of features such as recognizing, distinguishing, classifying, identifying, finding, changing, and verifying objects. In this direction, all operations are performed according to a certain algorithm.

In recent years, when we look at the studies in the field of robotic systems, playing chess with people, tightening screws in factories, correcting spelling mistakes, smelling perfumes, diagnosing diseases etc. systems that can perform different operations in many areas have been developed.

Key words: Image Processing, Object Detection, Raspberry Pi, Object Tracking Robot

ŞEKİLLER LİSTESİ

Sayfa No

SEMBOLLER LİSTESİ

KISALTMALAR

GUI	: Grafik Kullanıcı Arabirimi
HSV	: Hue, Saturation, Value
RGB	: Red, Green, Blue
VG	: Jeneratör Voltajı
GPU	: Grafik İşlemci
USB	: Evrensel Seri Veriyolu
MHZ	: MegaHertz
GHZ	: GigaHertz

1. GİRİŞ

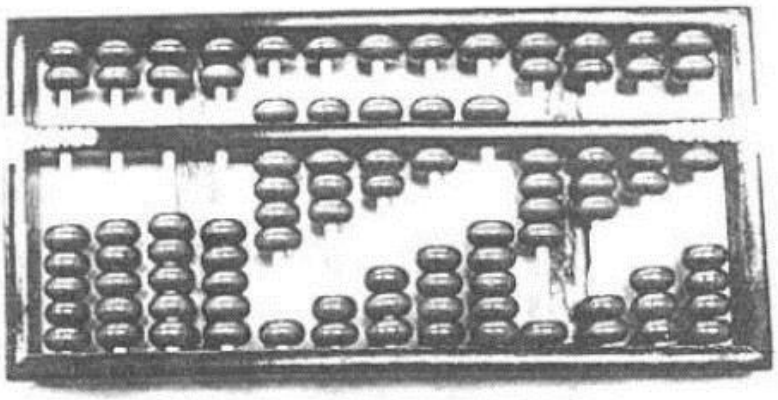
1.1.Robotların Tarihi Gelişimi

20. yüzyılda kullanılmaya başlanılan robot kelimesi, kelime anlamı olarak mekanik sistemleri, kontrol ve algılama sistemleri ile belirli bilgisayar algoritmalarına bağlı olarak işlem yapabilen makineler olmakla birlikte, sahip oldukları özelliklere bakıldığında çeşitli maddeleri, parçaları, aletleri vb. cisimleri kullanılan bir program ile özelliklerine göre istenilen işlemleri gerçekleştiren fonksiyonel makineler olarak da tanımlanmaktadırlar.

Robotlar, ortamdan topladığı verileri, sahip olduğu program bilgisiyle sentezleyerek, anlamlı ve kullanım amaçlarına yönelik hareket edebilen ardışık işlemleri güvenli bir şekilde yapabilen makineler iken, bu tanımlamalar değerlendirildiğinde, bir mekanizmanın veya sistemin robot olarak kabul edilebilmesi için çevresel ünitelerden gelen verileri algılaması ve bu verileri kullanarak istenilen devre ve sistemlerin kontrol ve kumandası için gerekli olan ardışık işlemleri yapabilmesi gerekmektedir. Bir robottan yapılması beklenen matematiksel ve mantıksal işlemlerin yapılabilmesi için oluşturulacak modeller için bir yazılıma ihtiyaç duyulmaktadır. Bu yazılım sayesinde elde edilen veriler kullanılarak gerekli işlemleri yapmak için karar verebilen bir elektro-mekanik sistem olarak kullanılmaktadır.

Tarihi gelişime bakıldığında, robotlar ve benzeri makinalara ait ilk bilgiler M.Ö. 3000 yıllarına kadar dayanmaktadır. Eski Mısır ve Yunan gibi medeniyetlerinde otomatik su saatleri vb. makinalar geliştirdikleri bilinmektedir. Aynı zamanda 13. Yüzyılda yaşayan fakat batı tarafından pek bilinmeyen sibernetiğin babası kabul edilen Ebu'l İz El Cezari'nin otomatik makineleri dışında 20.yy ortalarına kadar robotik alanda elle tutulur bir gelişme olmamıştır. Robotik sistemler ile ilgili olarak bu sürede ancak teori, fikir ve hayali tasarımlardan öteye gidilememiştir. Daha sonra 1920 yılında Karel Capek tarafından yazılan "R.U.R. (Rossum's Universal Robots)" adlı oyunda ilk defa robot kelimesi kullanılmış ve kelimenin kökeni çek dilindeki hizmet eden anlamında kullanılan "Robota" kelimesinden türetilmiştir. Daha sonraki süreçte robotik sistemler hayatımızda yer almaya başlamış, günümüzde farklı alanlarda, farklı uygulamalar için yaygın bir şekilde kullanılmaktadır [1]. Robotik sistemlerin kronolojik gelişim sürecine bakıldığında;

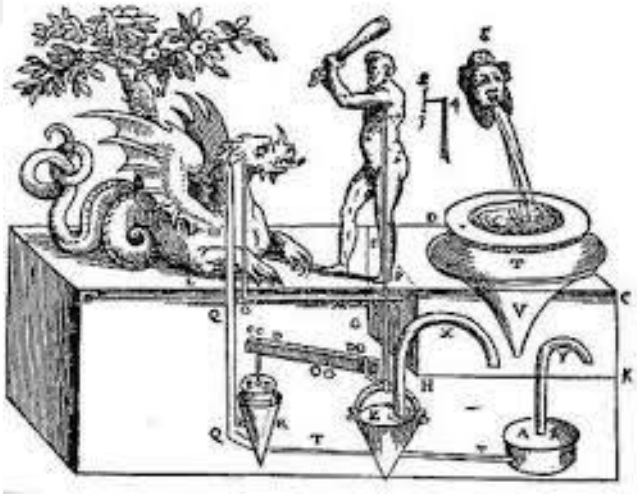
-Şekil 1.1 de gösterildiği gibi ilk dijital bilgisayar ve günümüzdeki ilk abaküs M.Ö. 1000 yıllarında Hindistan bölgesinde üretilmiştir.



Şekil 1.1 İlk abaküs

-Aristo'ya göre "Eğer her araç kendi işini görebilseydi, insan eline ihtiyaç duymadan kendi mekik dokuyabilseydi, kendi çalabilseydi, yöneticilerin elemanlara ihtiyacı kalmazdı". Bu açıklama ile otomasyon sistemlerinin temellerinin atıldığı kabul edilmektedir.

-M.Ö. 300 yıllarında su ile kendi kendine hareket eden makine ve Herkül tarafından okla öldürülen ejderha uygulamaları ile ilk otomasyon sistemi uygulamaları yapılmıştır.



Şekil 1.2. Su ile çalışan otomat ve Herkül otomatı

M.Ö. 250' de mucit Ctesibius suyla çalışan bir saat mekanizması yapmıştır.

1350 yılında Fransa'daki bir katedralin tepesine otomatik bir horoz yerleştirilmiş ve her gün öğlen saatinde kanatlarını çırparak ötmesi sağlanmıştır.

1700 yıllarında kullanılan otomatlar ile, 40 harf uzunluğunda bir mesaj kalem ile yazılmış ve 12 melodiye kadar da çeşitli müzikler çalınabilmiştir.

1800’lü yıllarda Marie Jacquard tarafından numerik olarak kontrol edilebilen bir mekanik dokuma tezgâhı icat edilmiş, aynı zamanda Thomas Edison daha önce ürettiği ve “Fonograf” olarak adlandırdığı icadından esinlenerek konuşabilen bir bebek tasarlamıştır.

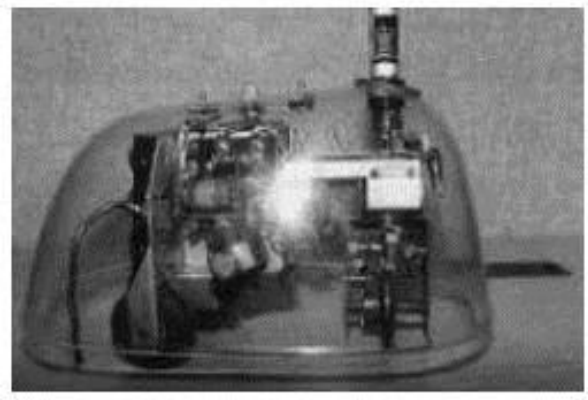
1890 yılında Nikola Tesla yaptığı çalışma ile ilk uzaktan kumandalı aracı geliştirmiş,

1928 yılında ise Londra’da elektrikle çalışan bir robot tasarlanıp üretilmiş olsa da elektrik motoru, elektromıknatıslar, makaralar, çarklar içermesine rağmen robot sadece kendi içerisinde hareket özelliğine sahip olmasından dolayı erişim sahasının sınırlı olması bir dezavantaj olarak görülmüştür.

1930’lu yıllara gelindiğinde uçak tasarımcıları uçaklar için otomatik pilot tasarlamış ve robot pilot olarak isimlendirmişlerdir. Aynı dönemde ilk defa sprey boya kullanılarak duvar boyayan endüstriyel robotlar tasarlanmış ve kullanılmıştır.

1940’lı yıllarda Westinghouse ise, yatay düzlemde bağımsız hareket eden iki robot üretmiş ve bu dönemde basit bir öğrenme algoritması ile çalışan labirent çözebilen bir fare robot tasarlanarak üretilmiştir.

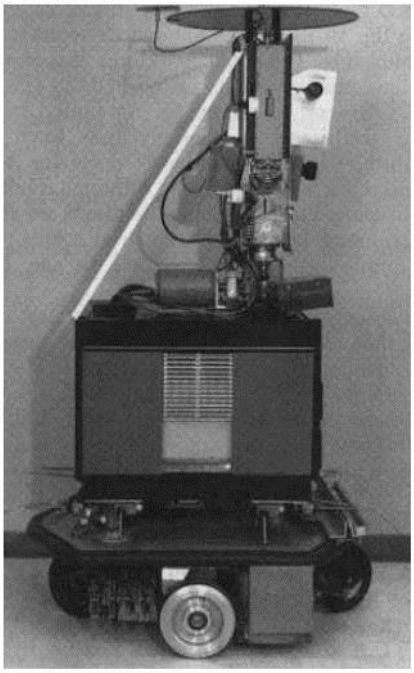
1953 yılında Grey Walter yaptığı çalışma ile Şekil 1.3’deki robot bir kaplumbağa geliştirmiş ve aynı zamanda Japon firması Seiko farklı tipte üretilen birçok saatin parçalarının montajını yapabilen minyatür bir robot geliştirerek kullanmaya başlamıştır.



Şekil 1.3. Kaplumbağa robotu *[2]*.

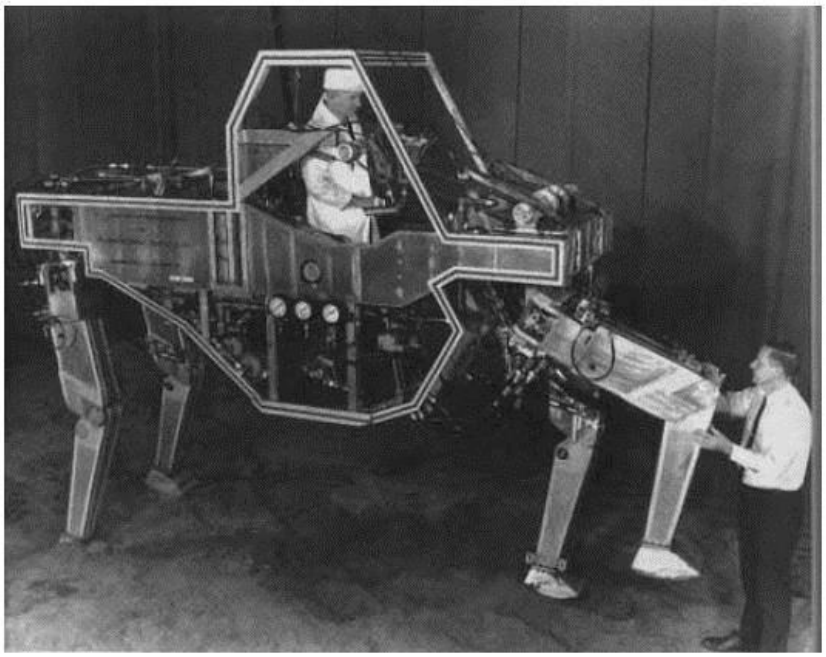
1954’te George Devol yaptığı çalışmalar sonucunda bilgisayar kontrollü endüstriyel bir robot üretmiş ve patentini almıştır.

1960’lı yıllarda Şekil 1.4’deki “Shakey” isimli bilgisayar kontrollü bir robot geliştirilmiş ve kullanılmaya başlanmıştır.



Şekil 1.4.Shakey robotu *[2]*.

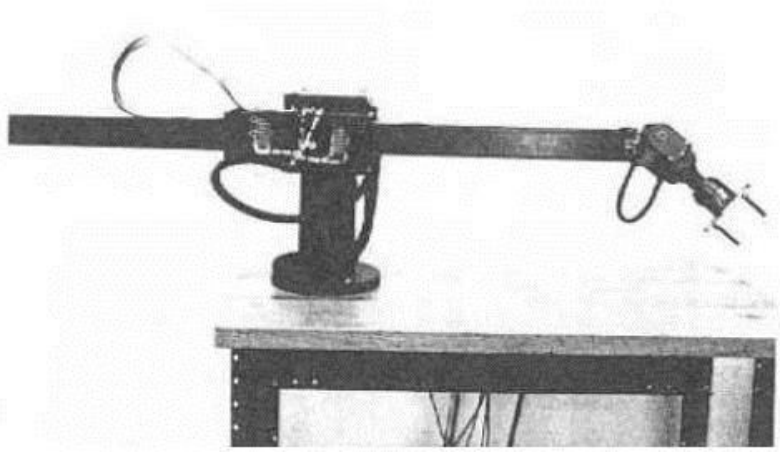
1960'lı yıllarda Şekil 1.5'de Hughes Aircraft şirketi "Mobot" adlı uzaktan kumandalı makineler üretmiş ve kullanmıştır. Ayrıca bu dönemde General Electric firması tarafından da bilgisayar kontrollü ve yürüyebilen ilk ayaklı bir araç üretilmiş ve kullanılmıştır.



Şekil 1.5. Bilgisayar kontrollü araç

1973’de Richard Hohn’un yaptığı çalışmalar sonuç vermiş ve ilk ticari bilgisayar kontrollü robot geliştirilerek kullanılmaya başlanmıştır. Bu robotta hidrolik hareket mekanizması kullanılmış ve 100 kg kadar ağırlık kaldırılmıştır.

1976 yılına gelindiğinde ise Şekil 1.6’da Stanford Üniversitesi’nde yapılan çalışmalar sonucunda “Stanford kolu” olarak adlandırılan ve elektrik enerjisi ile çalışan bir robot kol geliştirilmiştir.



Şekil 1.6. Stanford kolu *[2]*.

1977 yılına gelindiğinde Stanford araştırma enstitüsü, bir robot görme sistemi geliştirerek bu alandaki çalışmaları bir adım ileriye taşımıştır.

1980’lı yıllara gelindiğinde ise o dönemde, piyasada robot üreten dokuz Japon, dokuz Avrupa ve dört Amerikan şirketi olmak üzere toplam 22 şirket bu alanda ciddi çalışmalar yapmaya başlamış ve robot endüstrisi hızlı bir gelişim sürecine girmiştir.

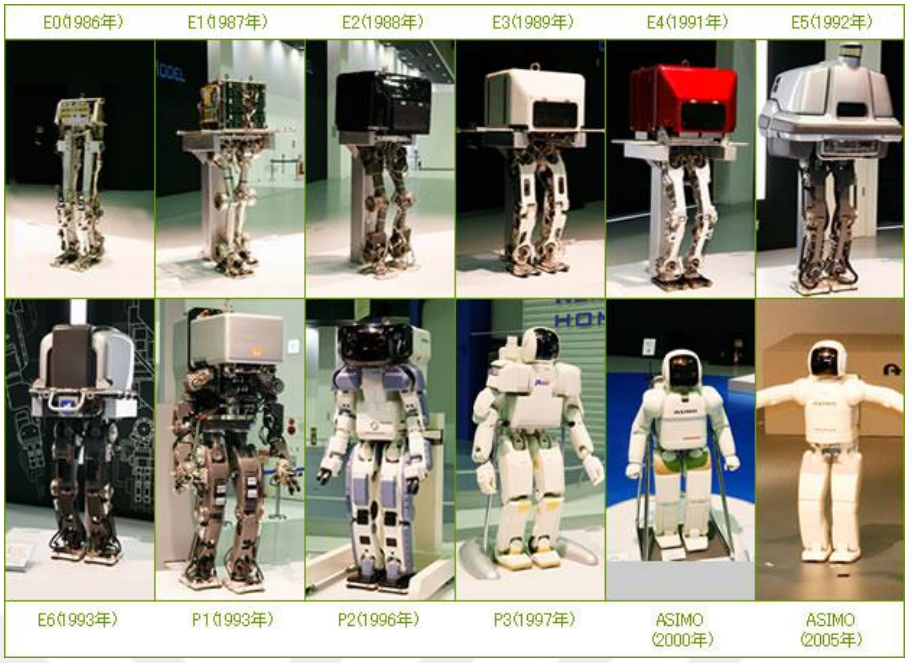
1983 Odetics adlı şirket, 6 bacaklı yürüyen bir robot üreterek piyasaya sürerken,

1986 yılında Honda firması insansı bir robot geliştirmek amacıyla bir proje başlattığını açıklamıştır.

1988 yılında bir hastanede ilk defa yardımcı bir robot kullanılmıştır.

1994’de Dante II olarak adlandırılan yürüyen bir robot geliştirilmiştir.

1996 yılına gelindiğinde Honda firmasının yürüttüğü proje kapsamında üretmiş olduğu insansı robot dünyaya tanıtılmıştır ve sonraki yıllarda Şekil 1.7’ de görüldüğü gibi geliştirilmiştir.



Şekil 1.7. Yıllara göre insansı robotların gelişimi

1997 yılına gelindiğinde Robotlar arası ilk futbol turnuvası Japonya'da düzenlenmiş ve Mars gezegenine gönderilen "Sojourner" ise Mars yüzeyinde keşif görevi yaparak Mars hakkında veri toplamıştır.

2003 yılında Nasa tarafından Mars yüzeyine iki robot daha gönderilirmiştir. Osaka üniversitesi ise "Actroid" adı verilen ve insan davranışlarını taklit edebilen, yeni bir robot geliştirdiğini açıklamıştır.

2004 yılında Irobot tarafından geliştirilen Mars keşif robotları sayesinde, Mars hakkında veri toplanmış ve bu veriler sayesinde Mars yüzeyinin yapısı tanımlanmıştır. Günümüzde de robot sektöründeki gelişmeler devam etmekte birçok ülke robotik sistemlerin geliştirilmesi için ciddi bütçeler ayırmaktadır. Örneğin, sadece kuzey Amerika'da robot sektörü 1,06 milyar dolarlık iş hacmine ulaşmıştır.

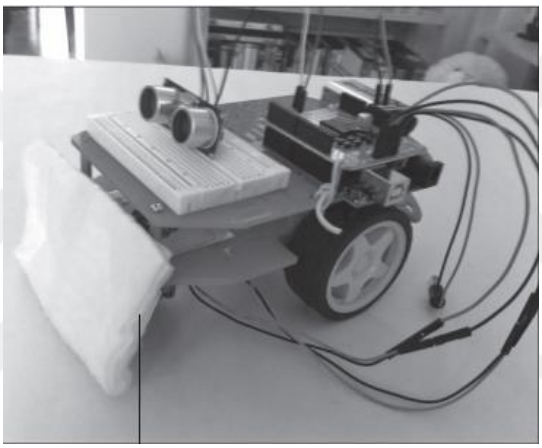
2006 yılında ise Şekil 1.8'deki dünyanın ilk yer yıkama robotu olan "Scooba" üretilmiş ve temizlik sektöründe kullanılmaya başlanmıştır.

Robot ve robot sistemlerinin tarihsel gelişimi genel olarak bu şekilde olmakla birlikte, günümüzde de farklı alanlarda farklı amaçlar için kullanılmak üzere tasarlanan, geliştirilen ve araştırılan birçok robotik sistem bulunmaktadır. Robotik sistemlerde gelinen nokta dikkate alındığında, robotik sistemlerden alınan verimin ve kalitenin yüksek olması nedeniyle yakın gelecekte birçok alanda yapılacak birçok işlemin robotlar tarafından yapılacağı öngörülmektedir.

Şekil 1.8.Scooba robotu *[3]*.

1.1. Robotik Sistemler ve Uygulama Alanları

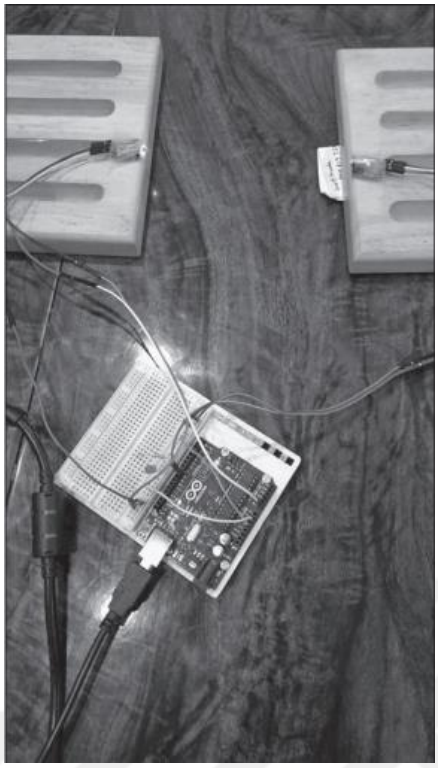
Robotik sistemler günlük yaşamın her alanında aktif olarak rol almakla birlikte özellikle sanayide ticari amaçlı kullanılmak üzere üretilmiş birçok robotik sistem bulunmaktadır. Yüzey temizleyici robot olan "Roomba" robotik sistemlerin ticari uygulamasına verilebilecek örneklerden biridir. Etkili bir robotik sisteminin tasarımı, ancak gelişmiş yapay zekâ algoritmaları, canlıların davranış biçimleri gibi önemli parametreler dikkate alınarak robotik sistemin tasarımı ve üretimi ile mümkündür. Bu nedenle tasarlanacak robotik sistemin nerede, nasıl ve ne şekilde çalışacağının dikkate alınması üretilecek robotik sistem açısından önemlidir. Şekil 1.9'da bir süpürücü robot örneği gösterilmiştir.



Şekil 1.9. Süpürücü robot

Robotik sistemlerin uygulama alanlarının gelişimi incelendiğinde, her geçen gün kullanım ve uygulama alanlarının arttığı görülmektedir. Robotik sistemler, askeri alanlar başta olmak üzere, sağlık sektöründe, rehabilitasyon çalışmalarında, endüstriyel ve sanayi uygulamalarında, evlerimizde, vb. birçok alanda güvenle kullanılmaktadır.

Özellikle askeri alanda robotik sistemlerin rolü önem kazanmış ve bu alanda robotik sistemler başarılı bir şekilde kullanılmaya başlanmıştır. Özellikle de yerden kumandalı hava araçları bu alandaki uygulamalara örnek verilebilir. Bu sistemler genellikle görüntü alma saldırı düzenleme vb. görevler için kullanılmaktadır. Bu sistemler müdahaleleri algılayabilen, alarm sistemlerine sahip, optik, yakınlık ve kızıl ötesi algılayıcılar gibi çeşitli sensörlerin kullanıldığı ve başarılı sonuçların alındığı robotik sistemlerdir. Şekil 1.10'da alarm tespit sistemlerinin bir örneği gösterilmiştir.

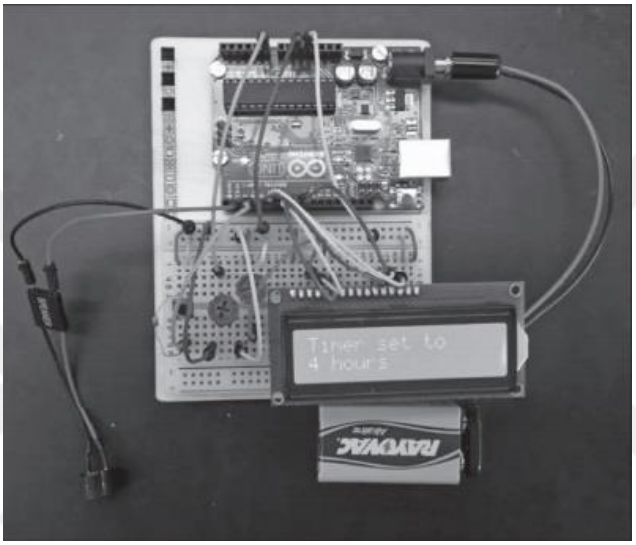


Şekil 1.10. Ardunio ile hazırlanmış alarm tespit sistemi

Robotik sistemlerin temel özelliklerinden biri de hareketlilik, özerklik, güç, iletişim ve algılayıcılar için kablosuz bağlantıların kullanılmasıdır. Çeşitli teknolojiler kullanılarak modern robotik sistemlerde kablosuz kontrol ve kumanda yapılabilmektedir. Wi-Fi teknolojisi bu alanda kullanılan önemli iletişim yöntemlerinden biridir. Şekil 1.11'de hazırlanmış olan sistemde robotik sistemleri kontrol ve kumanda etmek için, sisteme basit bir sunucu kurulur ve Wi-Fi üzerinden gönderilen kontrol ve kumanda komutları ile istenilen işlemler yapılmaktadır [4].

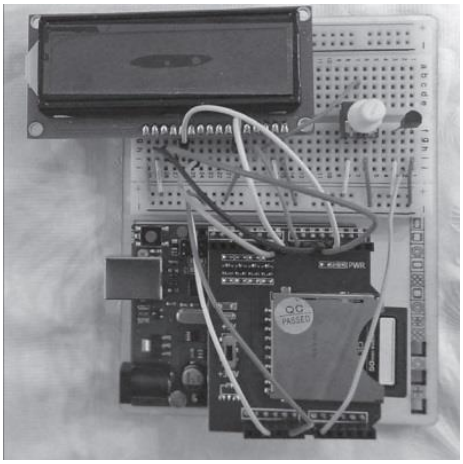
Şekil 1.11.Wi-Fi kontrollü robotik sistem

Son dönemlerde robotik sistemlerin yaygın kullanıldığı alanlardan biri de tıp alanıdır. Bu alanda hem hastaların sıra alması hem de hasta takip sistemlerinde (hasta asistan robotu) veri takibi alanında, yüksek yetenek ve hassasiyet gerektiren ameliyatlarda vb. birçok farklı uygulama için kullanılmaktadır. Hasta asistanı robotlar, özellikle hastanın vücut ısısını, şeker seviyelerini, tansiyonunu vb. hastaya ait birçok parametrenin ölçülmesi ve izlenmesi için kullanılmaktadır. Şekil 1.12'de bu amaçlar doğrultusunda hazırlanmış bir sistem örneği gösterilmektedir.



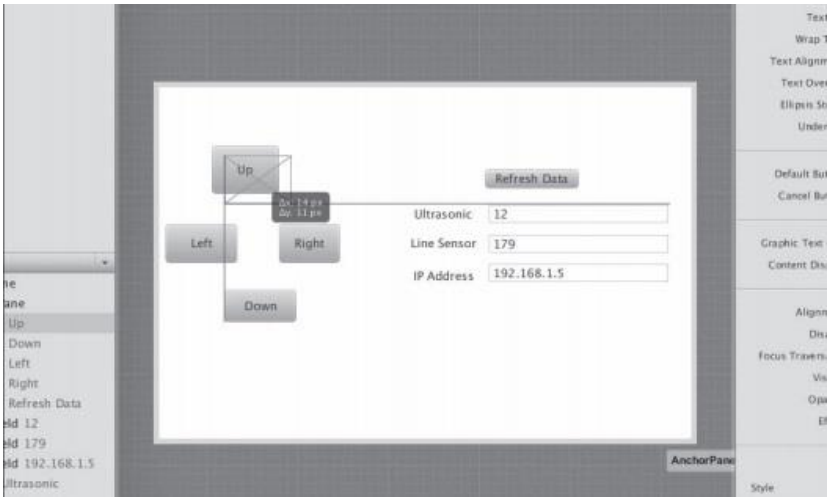
Şekil 1.12. İlaç hatırlatıcı robot

Ayrıca hava durumu takip sistemlerinde de robotik sistemler kullanılmaktadır. Bu sayede, verilerin hesaplanması, analiz edilmesi ve tahminlerde bulunulması vb. konularda etkili sonuçlar alınmaktadır. Meteorologlar, hava koşullarıyla ilgili problemlerde geniş verilerle çalışıp, hava modelleri hazırlayıp, kasırga gibi hava koşullarını ve genel hava durumlarını tahmin etmeye çalışırken, bunların en önemli yardımcıları robotik sistemlerdir [5]. Bütün bu kullanım alanları dışında sıcaklık sensörlerinin kullanıldığı uygulama alanları için de Şekil 1.13' te sıcaklık sensörlü bir sistem örneği gösterilmiştir.



Şekil 1.13. Veri işleyen sıcaklık sensörlü robot

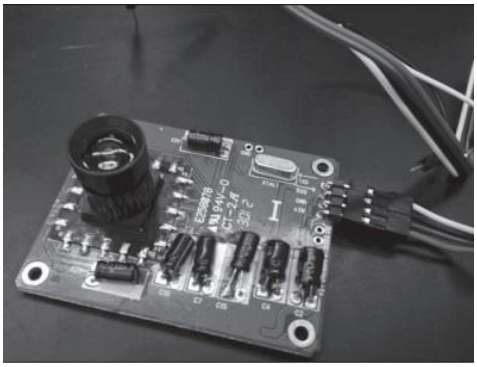
Günümüzde, robotik uygulamalar için kullanılan metin tabanlı veriler (program ara yüzleri) görsel anlamda daha kolay ve anlaşılır hale dönüştürülmüştür. Bu sayede kullanıcının robotik sistemleri kullanabilmesi daha da kolaylaştırılmıştır. Tüketici uygulamasını, çözümünü kolay ve eğlenceli hale getirmek için bir grafik kullanıcı arabirimi (GUI) kullanılmaktadır. Bu arabirimler, robotik sistemlerde kullanıcı tarafından verilen komutların dijital bir ekranda takibini sağlamak için tasarlanmış iken aynı zamanda robot için geliştirilen Wi-Fi sunucusuna bağlanarak gerektiğinde kullanılan butonlar aracılığı ile verilen komutları yürüterek gerekli işlemleri yapabilmektedir.



Şekil 1.14. Kullanıcı arabirimi olan robot örneği

Robotik sistemlerin bir diğer kullanım alanı güvenlik sistemleridir. Gelişen teknolojiyle birlikte güvenlik sistemlerinde kullanılan kameralar sayesinde elde edilen görüntünün görüntü işleme teknikleri kullanılarak sisteme adapte edilmesi robotik sistemler tarafından yapılmaktadır. Şekil 1.14'te görülen robot, önceden oluşturulan arabirim üzerinden, aldığı

görüntüleri Wi-Fi şebekesi üzerinden kullanıcıya göndermek suretiyle izlenen ortamda ne olup bittiği hakkında kullanıcıyı bilgilendirmektedir.



Şekil 1.15. Kameralı robot

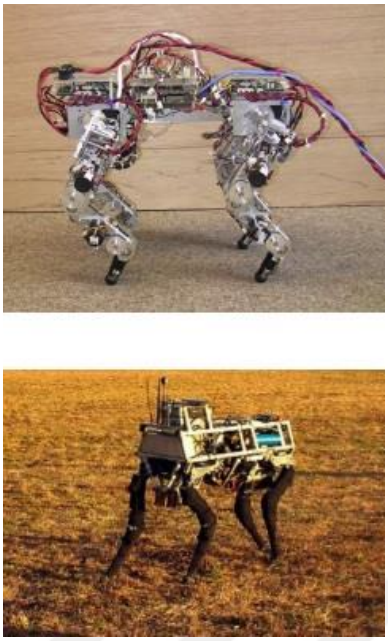
Robotların tasarımı ve dizaynı için gerekli özellikler ve kullanılabilir ekipmanlar, robotik sistemlerin gelişim sürecinde insan tarafından yapılması gereken tehlikeli ve zor işlerde robotik sistemlerin kullanılması tercih edilmiştir. Bu ekipmanlardan yaygın olarak kullanılan kamera sistemleri Şekil 1.15'de gösterilmiştir. Robotlar uzun bir süre sadece üretim bantlarında kullanılmış, insan ve makine etkileşimi sadece insanların, robotların hareketlerini belirleyen komutlar veren algoritmaların geliştirilmesinden ibaret olmuştur [6]. Yaygın olarak kullanılan sabit robotlardan zamanla insanoğlunun ihtiyaçlarının artması ile mobil robotlara geçiş sağlanmış ve farklı alanlarda mobil robotlar kullanılmaya başlanmıştır.

1.2. Robotlar ve Uygulamaları

Mobil robotlar, bir yerden başka bir yere operatörlerin yardımı olmadan hareket edebilen robotlardır. Yalnızca belirli bir çalışma alanına taşınabilen endüstriyel robotların çoğundan farklı olarak, önceden tanımlanmış bir çalışma alanında arzulanan hedeflerini gerçekleştirmek için serbestçe dolaşma özelliğine sahiptirler. Bu hareket kabiliyeti, yapılandırılmış veya yapılandırılmamış ortamlardaki geniş bir uygulama alanında kullanılmalarını sağlamaktadır. Mobil zemin robotları, tekerlekli mobil robotlar (WMR) ve bacaklı mobil robotlar (LMR) olarak iki gruba ayrılmaktadır. Ayrıca mobil robotlar insansız hava araçları ve özerk su altı araçlarında da (AUV) kullanılmaktadır. Tekerlekli robotlar diğer mobil robotlara göre daha popülerdir. Bunun nedeni, daha düşük mekanik karmaşıklığa sahip olmaları ve enerji tüketimlerinin daha az olması vb. avantajları birçok uygulama için bu robotları daha cazip hale getirmektedir. Şekil 1.16' da bu tarz robotların örnekleri gösterilmiştir.

Şekil 1.16. Tekerlekli robotlar

Bacaklı robotlar ise, standart olmayan merdivenler, moloz yığınları vb. ortamlar için daha uygundur. Genel olarak iki, üç, dört veya altı bacaklı olan sistemler tercih edilse de diğerleri de kullanılmaktadır. Şekil 1.17'de dört bacaklı robot örnek olarak gösterilmiştir. Tek bacaklı robotlar, yalnızca atlamayla hareket edebilme kabiliyetine sahip olduklarından dolayı uygulama alanları sınırlıdır. Mobil robotlar ayrıca mobil manipülatörlerde çeşitli görevleri gerçekleştirmek için bir veya daha fazla ışık manipülatörü ile donatılmış tekerlekli veya bacaklı robotlarda içermektedirler.



Şekil 1.17. Bacaklı robotlar

Mobil robotların gelişmesi insan-robot etkileşimine yeni bir yaklaşım getirmiş özellikle insan oğlu için tehlike oluşturabilecek birçok alanda kullanılmaya başlanması mal ve can kaybının önüne geçmiştir. Bütün bu kullanımlar ile beraber aynı zamanda acil kurtarma, askeri keşif, havacılık, jeolojik ve yeraltı keşifleri gibi alanlarda da güvenle kullanılmakta ve başarılı sonuçlar alınmaktadır [7].

Bu robotlar tehlikeli alanlarda başarı ile kullanılırken engebeli ve karmaşık arazilerdeki uygulamaları ise sınırlı olmuş ve bu durum dezavantaj olarak görülse de [5,6] sahip oldukları engel tanıma, denge ve görüntü işleme gibi özelliklere sahip olmalarından dolayı daha çok tercih edilmektedir.

Ayrıca mobil robotlar uzay araştırmaları, sualtı araştırmaları, elektrik ile çalışan araçlar, lojistik ve tarım alanları vb. farklı uygulamalar içinde güvenle kullanılmaktadır. Gelişen teknolojiyle birlikte, merak uyandıran uzay araştırmaları robotik sistemlerle hız kazanmış ve uzay robotiği alanındaki gelişmeler daha çok gezegenler arası gezi için geliştirilen akıllı sistemlerin tasarımı ve modellenmesi üzerine yoğunlaşılmıştır. Bu sistemlerde, biyolojik olarak insan hareketlerinden ilham alan, yenilikçi, hareket becerileri iyi olan, her türlü hava şartlarında çalışabilen ve sarp arazi yapılarına uygun, kullanıldığı alanlarda inceleme ve araştırma yapabilen, fonksiyonel keşif ve navigasyon özelliklerine sahip robotik sistemler olması başarılı sonuçlar alınmasında son derece önemli olmuştur.

Sualtı robotlarının tasarımı ve modellenmesinde de yapay zekâ yöntemleri kullanılmaktadır. Su altı inceleme ve araştırmalarda; sanal daldırma yöntemleri kullanan uzaktan kumandalı sualtı araçlarında kullanıcı desteği için sistemlerin geliştirilmesi, su altı uygulamalarda robot kollarının özerk manipülasyonu ve görev planlaması için yeni yöntemlerin tasarımı ve geliştirilmesi, su altı kameraları ile görüntü alınarak nesne tanıma yöntemlerinin geliştirilmesi vb. işlemleri kolaylıkla yapabilecek sistemlerin tasarlanarak kullanılması hedeflenmektedir.

Lojistik alanında da robotik sistemler kullanılmaktadır. Kullanılan bu sistemler sayesinde, sektörel bazda işlemlerin daha güvenli ve daha hızlı yapılarak olası mağduriyetlerin önlenmesi hedeflenmektedir.

Robotik sistemler kurtarma ve güvenlik alanında da kullanılmaktadır. Bu kapsamda iç ve dış mekanlar için uygun mobil platformların geliştirilmesi, potansiyel mağdurların yerlerinin tespit edilmesi ve kurtarılması, nesne ve kişilerin tespiti için radar sistemleri, lazer sistemleri vb. sensor uygulamalarının mobil robotlara entegrasyonu sonucunda oluşturulan sistemler ile başarılı uygulamalar yapılmaktadır.

Tarım alanında kullanılan mobil robotlar ise kullanılan tarım makinelerinin performansını arttırırken bu makinelerin harcadığı kaynak tüketimini azaltarak, çevresel tanıma yöntemleri kullanılmak suretiyle, birden fazla tarım makinesi arasında iş birliği sağlamak suretiyle, tarım süreçleri hakkında veri toplamakta ve bilgi işleme uygulamaları ile tarım sektörüne ciddi katkılar sağlamaktadır.

2. MATERYAL ve METOT

2.1.Raspberry Pi Kartı ile Nesne Takip Robotu Modellemesi

Hareketli nesnelerin takibi ve pozisyon denetimi, önceki bölümlerde de bahsedildiği gibi başta silah sistemleri, otomasyon sistemleri, haberleşme sistemleri, robotlar ve hava araçlar olmak üzere birçok alanda kullanılmaktadır. Robotik sistemlerin tasarımı ve istenilen sonuçları elde etmek zor ve karmaşıktır. Hareketli nesneleri izleyen ve pozisyon denetimi yapan sistemlerde hassasiyet ve cevap hızı önemlidir. Bu sistemlerin hassasiyetini artırmak için kullanılan donanım ve yazılımın uygun olması sistemin kararlı çalışmasını sağlarken doğru sonuçların alınmasını ve sistemin maksimum verimle çalışmasına imkân sağlamaktadır. Bu çalışmada, Raspberry uygulama kartı ile, OpenCV görüntü işleme kütüphanesi kullanılarak, Python programı aracılığıyla yapılacak kodlama ile oluşturulacak program, robot gövdesine yerleştirilen kamera yardımı ile nesnelerin takibi yapılacaktır. Bu sayede tanımlı renk uzayındaki nesnenin tespiti yapılarak robot ile takibi sağlanacaktır.

2.1.1. Bilgisayarlı Görme

İnsan yeteneklerini taklit eden ve onun gibi davranan makineler yapmak bilim insanları için her zaman heyecan verici bir araştırma konusu olmuştur. Başlangıçta, bilgisayarların hesap yapabilme yetenekleri dikkate alınarak, düşünme, işitme, konuşma, tanıma, görme gibi yeteneklerinde geliştirebileceği düşünülmüş ve bu düşüncenin temelinde, insan gibi dış dünyayı algılaması ve elde ettiği verileri akıllı bir biçimde işleyerek çeşitli problemlere çözüm üretmesi hedeflenmiştir.

Stockman ve Shapiro bilgisayarlı görmeyi; “Herhangi bir duyuyla elde edilen datadan görsel bilgi çıkarma işlemi” olarak tanımlamışlardır. Bilgisayar duyduklarımızın görüntüsünü çizebilir veya bir kokunun şiddetine göre bir görsel şekil oluşturabilir. Bu bağlamda bilgisayarlı görmenin temel amacı, algılanan verilere göre kabul edilebilir makul ve mantıklı kararlar verebilmesini sağlamaktır. Nesne olarak belirtilen şeyler görsel olarak algılanabilen cisimdir. Manzara, nesnelerin bir araya gelerek oluşturduğu bir bütünlüktür. Karar, nesneleri tanıma, ayırt etme, sınıflandırma, belirleme, bulup getirme, değiştirme,

doğrulama gibi durumları ifade eden ve belirli bir algoritmaya göre ardışıl olarak yapılan işlemlerdir.

Bilgisayarlı görme alanındaki çalışmalar 1970'lere dayanmaktadır. İlk dönemlerde sadece yazılar gibi çok basit şekillerin algılanması yapılmış ve bu temel üzerine farklı uygulamalar geliştirilmeye çalışılmıştır. 1980'li yıllara gelindiğinde bir nesnenin şekline ve sınırlarına bakılarak nesneler tanımlanmaya çalışılmış ve tanımlanan bu nesneler üzerinden kararlar verebilen sistemler geliştirilerek kullanılmaya başlanmıştır. Kullanılan algoritmalar sayesinde nesnelerde köşe saptama, gölgelendirerek şekil elde etme gibi verilerin elde edilmesi fiziksel tabanlı modellemelerin yapılması prensibinden yola çıkılarak gerçekleştirilmiştir. 1990'lı yıllara gelindiğinde ise bir görüntünün değil, farklı açılardan elde edilen birçok görüntünün işlenmesi başarılmış ve yüz tanıma sistemi kullanılarak hareketten faktorizasyon tabanlı yapı çıkarma, görüntü bölümlendirme gibi yöntemler kullanılarak başarılı sonuçlar elde edilmiştir. 2000'li yıllardan itibaren konunun önemi kavranmış ve üç boyutlu modelleme, birçok bilim insanının üzerinde çalıştığı konulardan biri haline gelmiştir.

Son yıllarda robotik sistemler alanındaki çalışmalara bakıldığında, insanlarla satranç oynayan, fabrikalarda vida sıkan, imla hatalarını düzelten, parfüm koklayan, hastalıkları teşhis edebilen vb. birçok alanda farklı işlemleri gerçekleştirebilen sistemler geliştirilmiş ve bu sistemler güvenle kullanılmakla birlikte yakın gelecekte daha geniş bir yelpazede kullanılacağı ön görülmektedir.

İnsanoğlunun görme sistemi, ışık enerjisinin cisimlerden yansıyarak göz üzerinde bulunan retinaya düşmesi sonucunda gerçekleşir. Üç boyutlu evrenden algıladığı bu verileri iki boyutlu veri haline dönüştürerek işler ve yaşamı boyunca oluşturduğu bilgi bankasını kullanarak değerlendirir ve yeni algılamaları anlamlandırarak yaşamını idame ettirir.

Bilgisayarlı görme sistemleri karmaşık görünse de hepsi için geçerli olan birkaç teknik vardır. Kameradan gelen bir görüntünün kullanışlı bir şekilde işlemeden önce, bu görüntünün bilgisayar tarafından anlaşılacak formata dönüştürülmesi gerekmektedir. Bu dönüşüm işlemine görüntünün sayısallaştırılması denir ki burada, görüntünün her bir piksel değeri bir sayı olarak hafızada depolanacak olan karelere bölünür. Her piksel noktasında görüntünün parlaklığını ve koyuluğunu temsil eden bir tam sayı bulunur. Bütün piksel değerleri için bu işlem gerçekleştirildiğinde görüntü tamsayılardan oluşmuş bir matrise dönüştürülmüştür. Resim bilgisi bu formata getirildikten sonra, artık uygun yazılımlar kullanılarak işlenebilmektedir. Bu kapsamda resim işleme, görüntü işleme, örüntü tanıma,

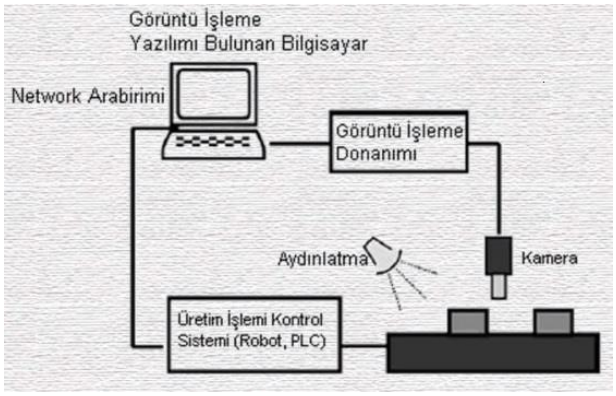
durum analizi, optik işleme, video işleme, görüntü yorumlama vb. algılamalar ve tanımlamalar bilgisayar görmesini oluşturmaktadır.

2.1.2. Bilgisayarlı Görmenin Basamakları ve Uygulama Alanları

Bilgisayarlı görme sistemleri genel olarak üç aşamada elde edilen görüntülerin işlenmesi ile elde edilmektedir. Şekil 2.1' de görüldüğü gibi ilk önce çeşitli görüntü işleme teknikleriyle, resimden bilgileri daha kolay çıkarmamızı sağlayacak başka bir resim elde edilir ki daha sonra keskinleştirme, dağıtma, köşelerini elde etme, eşik değer uygulama, histogramını dağıtma vb. gibi işlemlerden geçirilerek ikinci evreye hazırlanmaktadır. Bu aşamada resimden resmin karakteristik özellikleri çıkarılarak farklı bölgelerin renkleri, çembersel yaylar ve çizgiler, birleşik elemanlar gibi resmin öğeleri çıktı olarak alınırken, bir yandan da bu karakteristikler veri yapılarında tutulmaktadır. Son aşamada elde edilen özelliklerden görüntünün içindeki nesneler bulunup, bilgisayarın verdiği nesnelerin tanınması, ayırt etme, sınıflandırma, sınırlandırma, belirleme, bulup getirme, değiştirme, doğrulama vb. kararlar ile işlemler gerçekleştirilmektedir.

Şekil 2.1. Bilgisayar görmesi örneği

Bilgisayarlı görmenin uygulama alanlarını incelediğimizde, kameralar aracılığıyla aldığı görüntüleri değerlendiren ve bu değerlendirmelere göre iş yapan robotlar, robot görme sistemini oluşturmaktadır. Bu robotlar üretim aşamasında kameralar yardımı ile ürün görüntüsünü elde ederek kullanılan çeşitli yazılımlar aracılığıyla Şekil 2.2'de görüldüğü gibi kalite kontrolü yapmaktadır. Ayrıca robot işçi uygulamaları ile bir malzemenin bir yerden başka bir yere taşınma işlemleri de bu robotik sistemler ile yapılmaktadır.



Şekil 2.2. Parça denetim sistemi

Bilgisayarlı görme sistemleri tıbbi uygulamalarda da kullanılmaktadır. Şekil 2.3 ve Şekil 2.4' de bu bilgisayarlı görme sistemlerinin tıbbi uygulamalarına örnekler gösterilmiştir. Hastanelerdeki bilgisayarlı görüntü arşivleri, çekilen röntgen, tomografi ve ultrason görüntülerinin bilgisayar ortamında saklanmasını ve istenen bilgiye erişimi sağlayan önemli veri tabanlarıdır. Çekilen her türlü tıbbi görüntü, bilgisayar ortamında sayısal veri olarak saklanabilmektedir. İhtiyaç halinde bu görüntülere, görüntü veri tabanından ulaşıla bilinmektedir. Bu kullanımların dışında doktorlar için teşhis destek sistemlerinde de kullanılmaktadır.

Şekil 2.3. Röntgen cihazı *[8]*.

Şekil 2.4. Tomografi cihazı *[8]*.

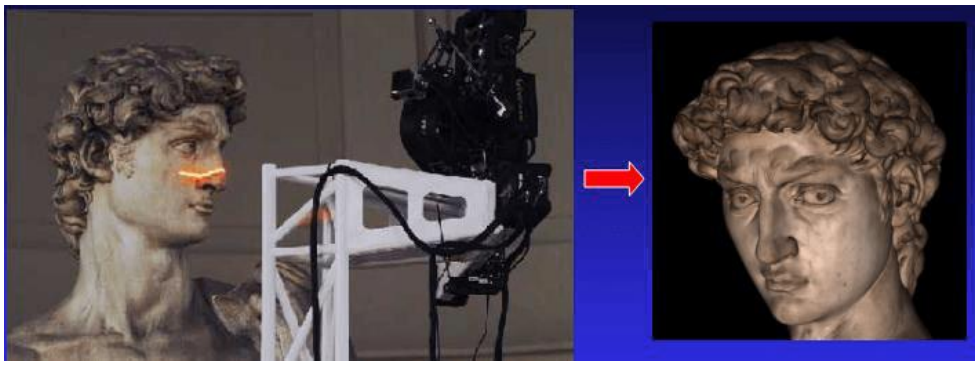
Bilgisayarlı görme sistemlerinin bir diğer yaygın kullanım alanlarından birisi de askeri ve sivil amaçlı güvenlik sistemleridir. Bu sistemlerin dışında işyerlerinde personel ve misafirlerin giriş ve çıkışlarının denetimi için geliştirilen ve kullanılan farklı sistemlerde mevcuttur. Plaka tanıma sistemleri bu uygulamalara örnektir.

Ayrıca güvenlik açısından önemli olan alanlarda kameralar aracılığı ile gözlenir ve bu alanda herhangi bir hareketlilik olduğunda bu durumu değerlendiren yazılımlar, alarm sistemlerini harekete geçirerek bu bölgelerin güvenliğini sağlamaktadır. Gece süresince güvenlik sistemlerinde kamera görüntüleri genelde durağandır. Hareket eden cisimler olmadığı için görüntü zaman içinde değişim göstermez. Çekilen görüntüler arasındaki fark hesaplanarak oluşan farka ve değişim büyüklüğüne göre sistem çalışmaya başlamaktadır. Bir diğer uygulama alanı da parmak izi tanıma güvenlik sistemleridir. Biometrik sistemlerin kullanılmaya başlanılmasından bu yana, Şekil 2.5'de bir örneği gösterilen parmak izi tanımlama sistemi en yaygın kullanılan güvenlik sistemlerinden biridir. Parmak izi yüzeyindeki birçok halka, ilmek, kemer ve adalardan oluşur. Hat bitimlerine minutiae denir ve bunların karşılaştırılmasına göre net sonuçlar alınmaktadır. Parmak sensöre yerleştirildiğinde sistem elektrik dalgaları göndererek elde ettiği parmak izi verisini kaydeder. Temasta kalma miktarı, yüzeye temas biçimi, basınç, ısı, elektrik dalgaları parmak izi tanımlamalarını etkileyen önemli parametrelerdir [9].

Şekil 2.5. Parmak izi tanıma sistemi

Havacılık sektöründe uzay ve hava fotoğraflarının değerlendirilmesi için bu alanda kullanılmak üzere önemli uygulamalar geliştirilmiştir. Dünyanın çevresinde dönen yüzlerce uydu dünyanın ve diğer gök cisimlerinin çeşitli fotoğraflarını çekerek yeryüzündeki kontrol merkezine göndermekte ve bu uydulardan alınan fotoğraflar sayesinde çeşitli ölçeklerde coğrafi ve jeolojik haritalar hatasız bir şekilde çizilebilmektedir. Aynı zamanda meteorolojik tahminlerin yapılabilmesi için hava hareketlerinin algılandığı ve görüntü işleme tekniklerinin başarılı bir şekilde uygulandığı önemli bir alandır. Meteoroloji uydularından elde edilen fotoğrafların değerlendirilmesi ve kestirim yöntemleri ile günlük hava tahmin raporları hazırlanmakta günlük, haftalık ve on beş günlük tahminler yapılabilmektedir.

Bütün bu kullanım alanlarının dışında sokaklarda, trafikte yaya ve araç tanımlamada, destekli sürüş sistemlerinde, rota belirleme, vb. alanlarda da kullanılmaktadır.



Şekil 2.6. Sanal albüm için alınan görüntü örneği

Beyin yapısının incelenmesinde benzerliklerin bulunup getirilmesi, ekolojik çalışmalar için böceklerde kimlik saptama işlemlerinin yapılması, doküman analizleri, kalite kontrol denetimleri, çekilen videolarda nesne tanınması, üç boyutlu modeller oluşturulması vb. daha birçok alan bilgisayarlı görmenin uygulama alanları olarak görülmektedir. Şekil 2.6'da bilgisayarlı görme için alınmış görüntü örneği gösterilmiştir.

2.1.3. Görüntü İşleme ve Yöntemleri

Günümüzde resimler ve videolar hayatımızın her alanında karşılaşabileceğimiz verilerdir. Görüntü işleme araştırmaları makine öğrenmesi ve istatistik bilimi tarafından domine edilmiştir. Gerçek dünya sahnesini anlamak için nesneleri veya olayları tespit etmek, onları sınıflandırmak, resimler ve videolar ile yapılmaktadır. Bilgisayar programları ile bu görüntülerde neyin olduğunu anlamak için algoritmalar tasarlanmakta ve bu algoritmalar kullanılarak işlemler gerçekleştirilmektedir. Bilgisayar görmesinde görüntü ilgilenilen nesnelerden ve görüntüdeki her şeyin temsil ettiği bir arka plandan oluşmaktadır. Bu nesneler arasındaki ilişkiler ve etkileşimler bu sahne anlayışı için anahtar faktörlerdir. Nesne algılama ve tanıma, bilgisayar görmesinin iki önemli öğesidir. Nesne algılama, bir nesnenin varlığını, kapsamını ve görüntüdeki konumunu belirlerken, nesne tanıma ise, nesnenin ait olduğu eğitim veri tabanındaki nesne sınıfını tanımlamaktadır. Nesne algılama genellikle nesne tanımadan önce gelir. Nesne algılama alanı tipik olarak fotometrik veya geometrik özellikleri eğitim veri tabanındaki hedef nesneninkilerle eşleşen kısımları yerelleştirmek için bir görüntünün her bir parçasını arayarak gerçekleştirmektedir. Şablon ile görüntü arasındaki benzerlik yeterince yüksek ise ölçekler, dönüşler ve bir saptama belirlenmektedir. Bir şablon ve görüntü bölgesi arasındaki benzerlik korelasyonları (SSD) ile ölçülmektedir. Son yıllarda, görüntü bazlı nesne detektörlerinin eğitim verilerine duyarlı olduğu da bilinmektedir **[10]**.

Görüntü işleme, görüntüyü dönüştürüp gelişmiş bir görüntü elde etmek veya ondan bazı yararlı bilgiler çıkarmak için görüntüleri dijital formata dönüştürüp, üzerinde işlemler yapılmasını sağlayan bir yöntemdir. Genellikle görüntü işleme sistemi, önceden ayarlanmış sinyal işleme yöntemlerini uygularken, görüntüleri iki boyutlu sinyal olarak işlemden geçirmeyi içermektedir. Görüntü işleme temel görüntünün optik tarayıcı ile alınması, veri sıkıştırma, görüntü iyileştirme ve uydu fotoğrafları gibi insan gözü olmayan lekelenme kalıplarını içeren görüntüyü analiz etme yöntemidir. Alınan sonuçların görüntü analizine dayalı olarak değiştirilmiş, kullanıma hazır hale getirme adımlarını içermektedir. Bu kapsamda görüntü işlemenin temel amaçları aşağıdaki gibi sıralanabilir;

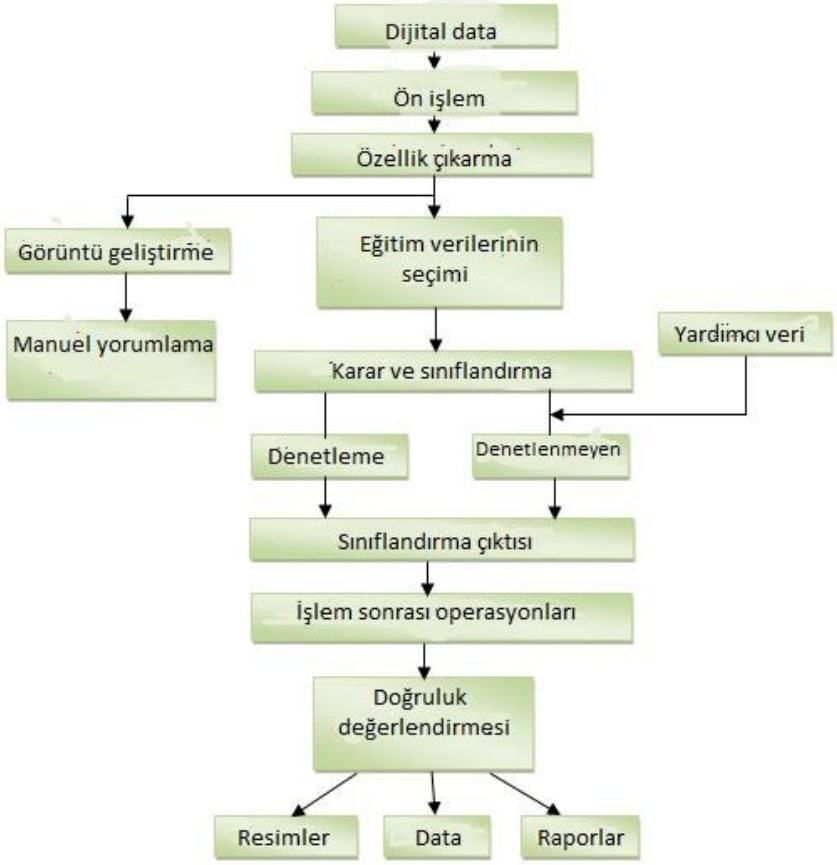
a. Görselleştirme ile görünmesi zor nesneleri gözlemleme,
b. Görüntü keskinleştirme ve restorasyon ile gürültülü görüntüleri iyileştirme,
c. Görüntü alımı ile ilgi çekici ve yüksek çözünürlüklü görüntü arama,
d. Desen tanıma ile bir görüntüdeki çeşitli nesneleri tanımlama,

e. Görüntü tanıma ile bir görüntüdeki nesneleri ayırt etme,

Görüntü işleme için iki yöntem kullanılır. Bunlar analog görüntü işleme ve dijital görüntü işlemedir. Fotokopiler ve fotoğraflar gibi basılı kopyalar için analog veya görsel görüntü işleme teknikleri kullanılırken ham veri alarak geçmişte toplanmış ve işlenmemiş görüntüler kullanılmaktadır. Görüntü analistleri tanımlamak istedikleri ürünler ile ilgili geçmiş işlemleri sisteme öğretirken bir derin öğrenme kolu olarak görüntü işleme, geçmiş verilerin işlenmesi temelinde çalışmaktadır.

Dijital işleme teknikleri dijital görüntülerin bilgisayarlarla manipüle edilmesine yardımcı olmaktadır. Şekil 2.7'de dijital görüntü işleme aşamaları gösterilmiştir. Uydu platformundan alınan görüntüler, algılayıcı hatası nedeniyle eksiklik içermektedir. Her türlü veri için dijital teknik kullanılırken bu işlem üç aşamada yapılmaktadır. Bu aşamalar;

a. Ön işleme,
b. Geliştirme görüntüleme,
c. Bilgi çıkarımıdır.



Şekil 2.7. Dijital görüntü işleme aşamaları

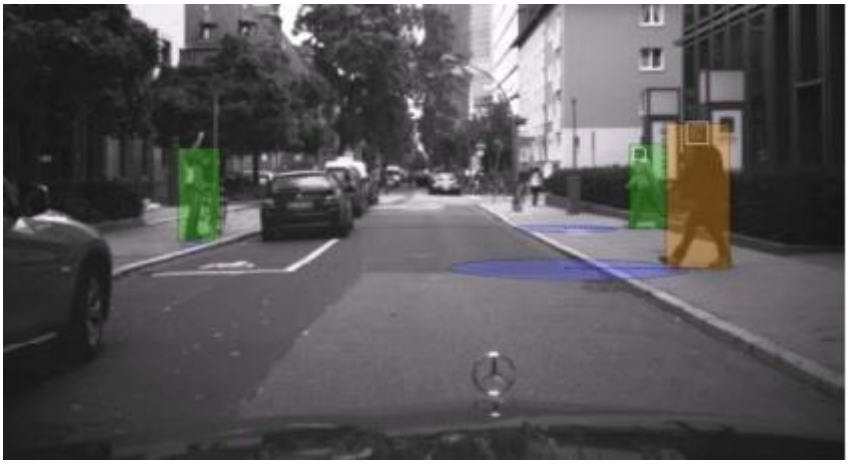
2.1.4. Nesne Takip

1960'lı yılların başından itibaren havacılık uygulamalarındaki gelişmeler ile başlayan, modern nesne takip sistemlerinin araştırmaları günümüze kadar devam etmiştir [11]. En basit haliyle, nesne izleme bir sensörden alınan gürültülü ölçümler göz önüne alındığında bir nesnenin yörüngesini sürekli olarak tahmin ve takip etme görevi olarak tanımlanmaktadır. Bununla birlikte son birkaç yıldır, farklı kullanım durumları ve senaryoların çok miktarda kullanılmasından dolayı nesne takibi çeşitli biçimlere dönüşmüştür [12].

Literatürde yaygın olarak kullanılan öznitelikler ile nesne takibi gerçekleştirilmektedir. Renk, kenar, doku, derinlik, süper piksel, hareket ve optik akıştır. Nesne sınıflandırma algoritmaları 4 temel başlıkta toplanmıştır [13]. Bunlar;

a. Hareket tabanlı,
b. Doku tabanlı,
c. Şekil tabanlı,
d. Renk tabanlı algoritmalardır.

Hareket tabanlı sınıflandırmada, nesne hareket bilgileri kullanılarak yapılır diğer sınıflandırma çeşitlerinde özniteliklerden faydalanılmaktadır. Son zamanlarda yapılan çalışmalarda tespit-izleme yaklaşımı popüler olmuştur. Bir nesne algılayıcısının sonuçlarına dayanarak, algılamalar ve önceden izlenen hedefler arasındaki uyumu bularak, mekânsal ve zamansal izleme problemini çözmeye çalışmaktadır. Görüntü çerçevelerini izleme görevi için girdi olarak kullanmak, aydınlatma değişiklikleri, kayıtsız nesneler, değişen görünüm ve/veya pozlama ve tıkanıklık gibi yeni zorluklar ortaya çıkarmaktadır. Bu nedenle, bu alandaki bazı yaklaşımlar, izlenen nesnenin görünümünü çevrimiçi bir şekilde öğrenmeyi amaçlamaktadır. Güçlü bir izleme yönteminin kullanımı özellikle akıllı araçlarda özerk sürüş bağlamında çok önemlidir. Böylece, nesne dedektörlerinin çıktısı sağlamlaştırılırken, bu da çevrenin daha güvenilir bir şekilde algılanmasını sağlamaktadır. Zaman içinde hedeflerin takibi, doğrudan gözlemlenebilir değişkenlerin tahmin edilmesini mümkün kılmaktadır. Görme temelli yaya tespiti ve izleme sistemi durumunda, hız ve ivme sadece tespitlerinden dolayı zaman içinde çıkarılabilmekte ve bu kinematik bilgi, özellikle Şekil 2.8'de gösterildiği gibi, gelecekteki hareketin tahmini için hayati önem taşımaktadır.



Şekil 2.8. Yayaların tespit edildiği sahne örneği

Son zamanlarda, derin öğrenme temelli yaklaşımların kullanılmaya başlanmış olması, görüntü sınıflandırılmasında görme temelli görevlerde büyük bir performans artışı sağlamıştır [14].

Nesne takibi, bilgisayar görmesinde önemli bir araştırma koludur. Video gözetiminde, güvenlik izlemede, video analizi ve diğer alanlarda yaygın olarak bu yöntem kullanılmaktadır. Geleneksel izleme algoritmaları, hızlı hareket ve tıkanıklık gibi nedenlerden belirli sahnelerde yetersiz performans göstermektedir. Aday örnekler hızlı hareketinden dolayı gerçek hedefi kaybedebilmektedir. Dahası, hedefin görünümü hareket ile değişebilmektedir. Bu durumda belirli sahnelerde izlemenin doğruluğunu ve sağlamlığını geliştirmek önemli bir sorun olarak değerlendirilmektedir.

Günümüzde nesne takibi endüstriyel uygulamalarda, robot ve savunma sanayi uygulamalarında vb. birçok alanda önemli bir yere sahiptir. Özellikle endüstriyel uygulamalarda üretilen ürünün kalitesi ve üretim sisteminin doğruluğu önemlidir. Bu nedenle üretilen ürünün takibi veya pozisyonunun bilinmesi uygulamanın kalitesini ve doğruluğunu artırırken hata olasılığını da düşürmektedir. Bu durum hatasız ve doğru üretim yapma imkânı sağlarken üretimde verimi artırmak suretiyle üretim maliyetlerini de düşürmektedir [15].

Nesne ve konum takibi ile konum bilgisi savunma sanayinde de geniş bir uygulama alanına sahiptir. Örneğin; askeri bir aracın takibi sağlanıp konumu belirlenerek haberleşme sistemleri ile ilgili birimin bilgilendirilmesi vb. kullanımlar dışında diğer güvenlik sistemleri, trafik kontrol sistemleri, medikal görüntüleme gibi birçok alanda güvenle kullanıldığı görülmektedir.

2.2.Çalışmada Kullanılan Yöntemler

2.2.1. Takip Sistemleri

Robotik sistemlerde, hareket halindeki cisimleri gözlemlemek ve pozisyon verisini sonraki bir işlem için zamanında, sıralı bir şekilde temin etmek için izleme sistemleri kullanılmaktadır. Araç takibi, sonar izleme, video izleme, hava trafik kontrol takibi vb. takip sistemlerine örnek olarak gösterilmektedir.

Mekanik algılama: Hedef ve çevre arasındaki fiziksel bağlantıları içermektedir. Enkoder ve potansiyometrelerin kullanımında, nesnelerin pozisyonu veya bir motorun dönüşü gibi uygulamalarda kullanılmaktadır.

Sonarlar: Ses dalgaları yardımı ile bir cismin ne kadar uzakta olduğunu, boyutlarının ne olduğu, hızı, şekli vb. bazı önemli verileri elde etmemizi sağlayan sistemlerdir. Uzun menzilli ama gürültü duyarlıdırlar.

Optik sensörler: Bu sistem yayılan veya yansıyan ışığın ölçülmesine bağlıdır. Genellikle, izleme sistemi, bir veya daha fazla optik sensör ve bir veya daha fazla ışık sensöründen oluşan optiklere bağlıdır. Yaygın optik sensör tipleri, ultraviyole ışık gibi belirli bir ışık spektrumunu yakalamak için sabit filtreli kameraları içermektedir. Işık kaynağı yapay olarak üretilmiş ışık veya ortam ışığı olabilmektedir.

İnersiyal sensör: Cihazın veya aracın hızını, dönme açısı ve dönüklük miktarlarını tespit edebilmektedir. Jiroskoplar ve ivme ölçerler gibi devrelere entegre edilebilmektedir. Düşük gecikme süresi ve gürültüye karşı daha az duyarlı olmasına rağmen, doğruluk hassasiyeti zayıftır [16].

Bu projemizde takip sistemlerinden 3 boyutlu nesnenin takibi için video tabanlı nesne izleme yöntemi kullanılarak koordinatları, konumu ve yönelimleri izlenerek takip edilmek istenmiştir. Bu tarz sistemler robot görmesi, insan bilgisayar etkileşimi vb. uygulamalarda kullanılmaktadır.

Video izleme sistemlerinin iki farklı çeşidi bulunmaktadır.

a. Model tabanlı,
b. Görünürlüğe dayalı.

Model tabanlı sistem, çevrenin kısmen veya tam bir modeli mevcut olduğunda, poz ve gözlemcinin konumunu elde etmek için kullanılır. Sistem, sürekli olarak oluşabilecek sorunlara hızlı bir şekilde çözüm sağlamaktadır. Dezavantajları pahalı olmasıdır. Genellikle çok kameralı sistemler tarafından sağlanan iyi görsel bilgilere bağlı olarak çalışmaktadır.

Görünüm tabanlı sistemler ise daha düşük veri işleme gücüne ve donanım karmaşıklığına bağlı olmakla birlikte, ayrı ve sınırlı sayıda koordinat veya algılanan nesnenin konumunu sağlamaktadır [17].

Bu tez çalışmasında görünüm tabanlı sistem esas alınarak tanımlı renk, şekil ve boyut gibi özellikler ile nesnenin konumu ve hareketlerinin izlenmesi hedeflenmiştir. Kullanılan uygulama kartı ve seçilen programlama dilinde hazırlanan yazılım ile düşük veri işleme gücü gibi olumsuzlukların etkilerinin ortadan kaldırılması, yapılan donanımsal modelleme ile birlikte donanımsal karmaşıklığın önüne geçilerek, sistemin düşük maliyetli olması sağlanmıştır. Böylece sistem hem efektif hem de düşük maliyet ile daha tercih edilebilir bir hal almaktadır. Bunun yanı sıra seçilen ekipmanlar ve yazılımların geliştirilebilir olması, bu çalışma aracılığıyla ihtiyaç duyulabilecek sağlık, tarım, askeri, lojistik, otomotiv vb. sektörlerde yeni uygulama alanları oluşturacağı kanaatindeyim.

2.2.2. Nesne Tespit ve İzleme Yöntemleri

Robot görmesinde, nesne takibi çeşitli faktörlerden dolayı oldukça zordur. Örneğin rijit olmayan nesne yapıları, oklüzyonlar, kamera hareketi ve hem nesnedeki hem de sahnedeki hızlı değişiklikler vb.

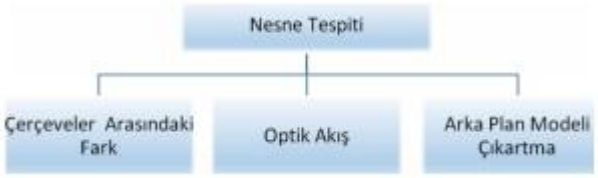
Tek hedefli izleme, gürültülü ölçümlere dayanarak hedeflerin durumunun kesin tahminine odaklanırken, birden fazla hedefin eş zamanlı izlenmesi, ek bir zorluk sağlamaktadır. Çoklu nesne takibi sırasında, yeni ölçümler ile izlenen hedeflerin bütünü arasındaki yazışmaların tanımlanması gerekmektedir. Aksi takdirde problemler meydana gelebilmektedir [18]. Bu problemler genellikle bilinmeyen sayıda hedef, algılayıcıdan gelen yanlış veriler, potansiyel olarak yeni oluşan nesneler gürültü ile daha karmaşık hale gelmektedir. Video verilerindeki nesnelerin izlenmesi, bilgisayar görmesi ve makine öğrenmesinden ek teknikler içeren görsel hedef takibinin araştırma alanına yol açmıştır [19].

Literatür çalışmaları incelendiğinde nesne takibi yapan çalışmalar genellikle 4 aşamadan oluşur, Şekil 2.9'da bu aşamalar gösterilmiştir.

a. Video ön işleme,
b. Nesne tespiti,
c. Nesne sınıflandırılması
d. Nesne takibi olarak gruplandırılmaktadır.

Şekil 2.9. Nesne takibi aşamaları

Bir video veya görüntüdeki nesnenin tespiti için iki temel bilgi kullanılır. Birincisi görsel öznitelik (renk, doku ve şekil gibi) ikincisi ise hareket bilgisidir. Bunlar dışında nesne tespitinde kullanılan yöntemler Şekil 2.10'da gösterilmiştir.



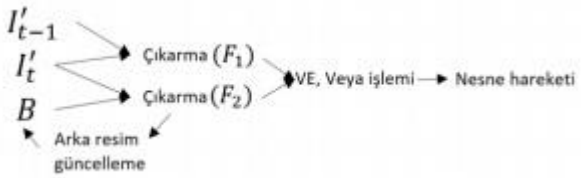
Şekil 2.10. Nesne tespit yöntemleri

Hareketli nesne tespiti için kullanılan yöntemlerden biri, iki görüntü arasındaki geçici değişiklikleri bulma yöntemidir. Bir video içerisinde arka arkaya gelen iki video çerçevesinin çıkartılması ile elde edilmektedir. Kullanılan bir diğer yöntemde ise belirlenmiş nesnenin olduğu görüntüden sadece arka plan görüntüsünün bulunduğu çerçevenin çıkarılması ile elde edilmektedir [20]. Daha sonra elde edilen fark görüntüsü belirli bir eşik değerden geçirilerek ve istenmeyen nesneler elenir. Oluşan görüntüden gürültüler temizlenerek istenen nesnelerin görüntüsü elde edilmektedir.

Şekil 2.11. *Arka plan çıkarma yöntemi [21].*

Şekil 2.11'deki yöntem 3 adımda sıralana bilir [22].

a. Yeni video çerçevesiyle bir önceki çerçevenin birbirinden çıkarılması.
b. Arka plan görüntüsüyle yeni çerçevenin birbirinden çıkarılması.
c. Birinci ve ikinci adımlarla elde edilen iki görüntünün birbiriyle mantıksal kapı işlemine tabii tutulması.



Şekil 2.12. Nesne tespitine yönelik algoritma diyagramı *[23]*.

Şekil 2.12'deki çalışma modeli için tek bir arka plan görüntüsü kullanıldığında inşa edilen bu model ışık değişimine karşı hassastır. Yaklaşık medyan metodu yöntemiyle referans alınan tek bir görüntü yerine n sayıda ve hafızada saklanan arka plan görüntüleri ile elimizdeki görüntünün bir algoritma kullanılarak kıyaslanıp hedef nesne ortaya çıkarılmaktadır [22]. Bu yöntemde daha büyük bir hafızaya ihtiyaç duyulmaktadır. Gaussian ortalama arka görüntü çıkarma yönteminde arka plan görüntüsünü çıkartmak için diğer yöntemde olduğu gibi mevcut arka plan görüntüsü çıkarılarak her bir pikselin olasılık yoğunluk dağılımları bulunmaktadır. Pikselin ilgili olasılık yoğunluğuna bakılarak bu

görüntünün arka plana mı yoksa nesneye mi ait olduğu tespit edilmektedir. Bu yöntemde işlemler 3 aşamada yapılmaktadır [21].

a. Arka planın olasılık görüntüsünün oluşturulması.
b. Yeni görüntüdeki arka plan ve nesnenin birbirinden ayrılması.
c. Arka plan görüntü modelinin güncellenmesi.

Nesne tespitinde arka planın güncellenmemesi, kullanılan yöntemlerin ışık gibi oluşacak dış ortam değişikliklerine karşı oldukça hassas olmalarına neden olmaktadır. Bu yüzden arka plan güncellenmesi nesne tespitinde önemli yer tutmaktadır. Arka plan ve nesnenin ortaya çıkarılması ile takip etme yöntemlerinin dışında kullanılan başka bir yöntem ise optik akış yöntemidir. Bu yöntemde arka arkaya gelen görüntülerden piksel hareketliliğine bakılarak arka plan çıkarılmakta ve bu şekilde nesne takibi yapılabilmektedir. Sınıflandırma; bir veri kümesindeki belirli nesnelerin özelliklerine göre gruplara ayrılmasıdır. Bu ayrıştırma işlemi yapılırken nesnenin kendine has öznitelikleri kullanılmaktadır. Bu öznitelikler nesne tanımlanması için kullanılan sayısal değerlerden (piksel değerleri, görüntüdeki ortalama yoğunluk değerleri vb.) oluşmaktadır. Öznitelik seçimleri ve bunların kullanımı performansı doğrudan etkilemektedir.

Nesne takip uygulamalarında birçok problem ile karşılaşma ihtimali yüksektir. İstenen bir nesneyi bulmak zor olabilmekte ve hareketli bir nesnenin yörüngesini tahmin etmede de zorluklar yaşanabilmektedir. Bu nedenle izleme nesne kaynaklı nedenlerden dolayı zorlaşmaktadır. Bu duruma;

a. 3 boyutlu dünyadan alınan verinin 2 boyutlu görüntüye aktarımı sırasında oluşabilecek bilgi kaybı,
b. Karmaşık nesne hareketi,
c. Karmaşık nesne şekli,
d. Görüntülerde gürültü oluşumu,
e. Sahne aydınlatma değişimleri,
f. Bir nesnenin karmaşık veya sert olmayan nesne gibi karmaşık niteliği,
g. Gerçek zamanlı işlem gereksinimleri vb. örnek verilebilmektedir.

Bir nesneyi izleme özelliği tipik olarak, işlevlerini yerine getirmek için birleştirme özelliğini kullanarak nesneleri izleyen algoritmadır. Kullandığı özellikler;

Renk: Nesnenin yüzey yansıtma özelliği ve ışık kaynağının spektral güç dağılımı bir nesnenin rengini etkileyen iki fiziksel faktördür.

RGB renk spektrumu, görüntü işlemede rengi göstermek için kullanılmaktadır. RGB renk spektrumu, nesnenin rengini göstermede optimal değildir çünkü algısal olarak homojen değildir. RGB renk spektrumundaki renkler, ortalama insan gözü tarafından renk algısı ile uyuşmaz. Dahası, RGB boyutları korelasyonludur. Bütün bunların aksine HSV ton, doygunluk, değer gibi olguları temsil etmesine rağmen, algısal olarak renk spektrumları vardır. Burada sunulan renk spektrumları gürültüye karşı duyarlıdır. Yapılan analizler incelendiğinde farklı renk uzaylarından hiçbiri diğerinden daha verimli olmadığından, kullanımları arasında ciddi farklar bulunmamaktadır [24]. Yalnızca HSV, renk bilgisini şiddetten veya aydınlatmadan ayırmasıyla öne çıkmakta ve kolaylık sağlamaktadır. Çünkü değer ayrıldığından, yalnızca doygunluk ve renk tonu kullanarak bir histogram veya eşikleme kuralları oluşturulabilmektedir. Sadece tonu söndürerek bile, RGB'den çok daha iyi çalışacak bir temel rengi anlamlı bir şekilde temsil edilebilmektedir. Sonuç olarak, daha güçlü bir renk eşiği daha basit parametreler ile aşılmaktadır. Bu bilgiler ışığında bu çalışmada hazırlanan sistemde, renk uzaylarından HSV renk uzayı daha ağırlıklı olarak kullanılmıştır.

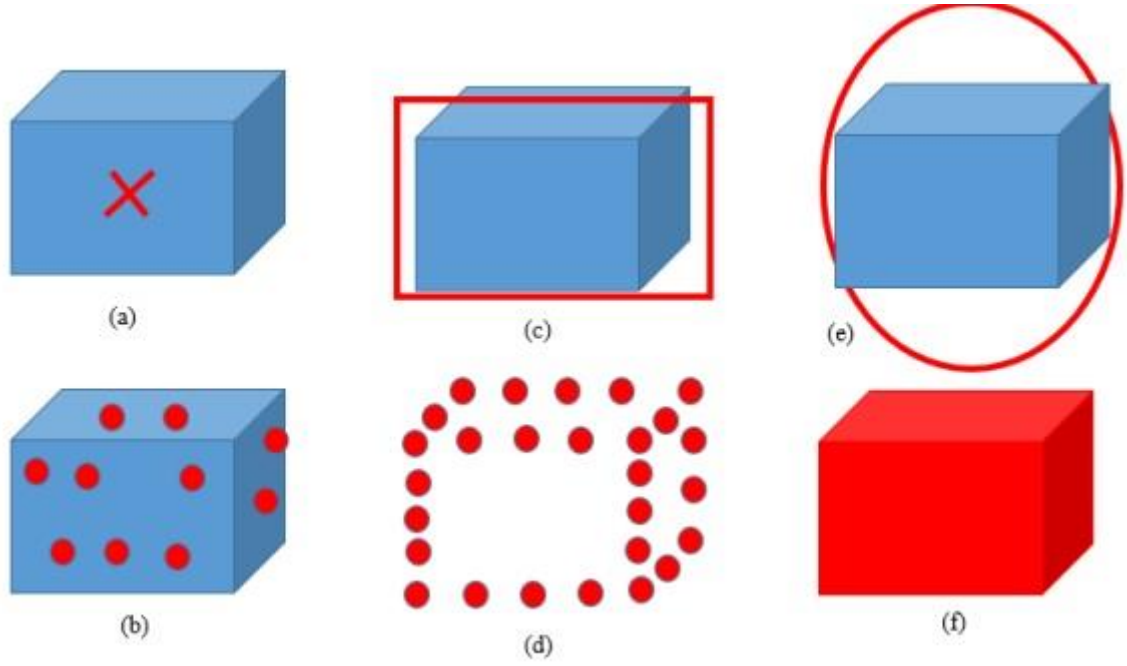
Bu çalışmada grafiksel bir kullanıcı arabirimi veya kontrol paneli gibi bir şey içermeyen bir program oluşturmak yanlıştır, çünkü bu tür sistemlerin çalıştığı ortamlar içinde çalışma süresi boyunca çok şey değişebilmektedir. Örneğin, sabit aydınlatmalı bir oda içinde kullanılmadığı zaman, kullanıcı tarafından girilen parametreler, bilgisayarların piksel değerlerine insan gözlerinden çok daha duyarlı olduğu için kullanıcı girdilerinden farklı değerler ortaya çıkabilmektedir. Bu nedenle OpenCV'nin kullanışlı bir özelliği, kontrol paneli penceresine eklenebilen ve çalışma zamanı sırasında çeşitli değerleri değiştirmek için kullanılan yerleşik izleme çubuklarına sahip olmasıdır. Böylece ortam aydınlatmasına göre renk uzayındaki parametreler değiştirilebilmektedir.

Kenarlar: Birçok nesnenin kenarları, görüntü yoğunluğundan dalgalanmalara neden olma eğilimindedir. Bu değişiklikler kenar algılama kullanılarak takip edilebilmektedir. Kenarların aydınlatmadaki değişikliklere daha az duyarlı olduğu bilinmektedir. Ayrıca düz zeminde hareketinin kolaylığından kavisli nesne kullanılması seçeneğini ön plana çıkarmaktadır. Bu çalışmada Canny kenar detektörü tercih edilmektedir. Optimal detektör olarak da bilinen Canny kenar detektörünün algoritması üç temel kriter hedefler.

a. Düşük hata oranı,
b. İyi lokalizasyon (tespit edilen kenar pikselleri ve gerçek kenar pikselleri arasındaki mesafe en aza indirilmesi),

c. Minimum yanıt (her kenarda sadece bir detektör tepkisi) olması.

Canny detektörü alt ve üst olmak üzere iki sınır kullanmaktadır. Bir piksel gradyanı üst sınırdan(eşikten) yüksekse, piksel bir kenar olarak kabul edilmektedir. Bir piksel değeri alt eşiğin altındaysa, kenar reddedilmektedir. Piksel gradyanı iki sınır arasındaysa, sadece üst sınırın üzerindeki bir piksele bağlı kenar kabul edilmektedir [25]. Şekil 2.13'de nesneler için kullanılan farklı kenar belirleme metotları gösterilmiştir.



Şekil 2.13. Farklı şekillerde nesne türleri *[26]*.

Renk modeli, bir görüntü çerçevesinden izlenmesi ve hedeflenmesi için en kolay ve hızlı yollardan biri olup, belirli bir rengin bir karedeki ayrılmasını sağlayarak yönünü ve konumunu döndürmektedir. Düşük maliyetli olmaları mütevazı sistemler için uygundur [27].

Temel olarak, her bir renk, her bir kanalın belirli bir renge karıştırılma derecesinin anlaşılmasına yardımcı olan mavi, kırmızı ve yeşil değerden oluşmaktadır. Bu nedenle, izleme gereksinimlerini gerçekleştirmek için kanalların her biri için minimum ve maksimum değerlerin zorunlu olması, parlaklığın nesnenin rengi kadar güvenilir olmadığının göz önüne alınması gerekmektedir. Tespit edilecek olan renkler belirlendikten sonra algılanan objelerin tespiti için çeşitli algoritmalar veya yöntemler uygulamak mümkündür. Görüntüyü kolayca okunabilir hale getirmenin ilk adımı, görüntüyü eşiğe daha kolay bir şekilde HSV renk uzayına dönüştürmektir.

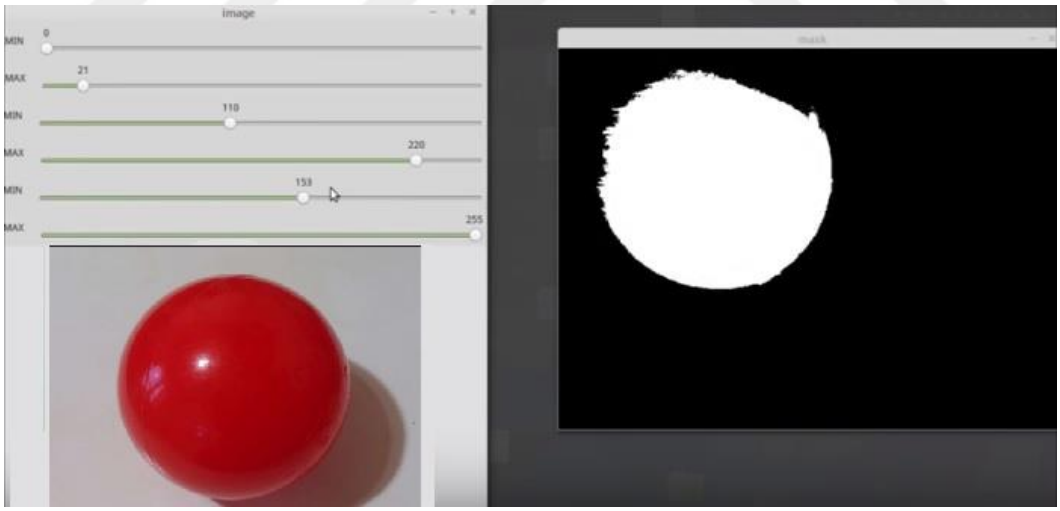
İkinci adım, HSV değerlerini içeren görüntünün yakalanması ve bir piksel dizisi boyunca yinelemek değerlerin bir alt sınırın altında olması durumunda da bu piksellerin değerlerinin değerlerin üstündeyken ise 255 olduğu eşiktir. Etkili bir görüntü yaratma, daha yüksek sınır aralığı ile sağlanmaktadır. Sınırlar, güncel olarak ayarlanmış ton, doygunluk ve değer sınırlarını içeren skalerler ile belirtilmektedir.

$$dst(I) = lowerb \leq src(I)_{hue} \leq upperb \cap \quad \textbf{(2.1)}$$

$$\cap\, lowerb \leq src(I)_{sat} \leq upperb \cap \quad \textbf{(2.2)}$$

$$\cap\, lowerb \leq src(I)_{val} \leq upperb, \quad \textbf{(2.3)}$$

Dst denklemin sonucunu depolayacak olan hedef çerçeve üzerindeki alt ve üst HSV aralıklarını içeren mevcut piksellerdir. Src, kameradan çekilen görüntü karesinde geçerli piksellerdir [28]. Bu görüntü daha sonra kontur bulma algoritmaları ile okunarak analiz edilmektedir.



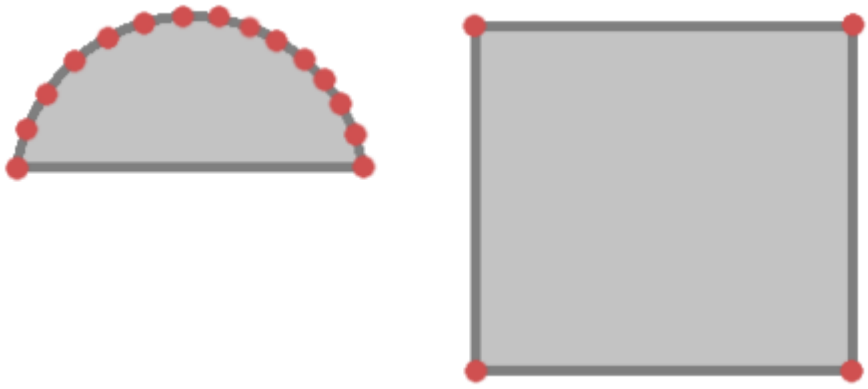
Şekil 2.14. Kontrol paneli ve görüntülenen nesne verisi

Bu çalışmada kamera yardımıyla alınan görüntüdeki piksellerin istenilen renk ile tanımlanması için görüntünün her bir pikselinin taranması prensibine dayalı bir sistem hazırlanmıştır. Şekil 2.14'de kontrol paneli yardımı ile nesnenin renk uzayındaki parametreleri gösterilmiştir. Ayrıca hata payının minimize edilmesi için şekil ve boyut gibi özelliklerden yararlanılmıştır. Böylece nesnenin doğru bir şekilde tanınması için, birkaç

parametrenin gerektirdiği özelliği sağlaması sistemin doğruluk hassasiyetini de artırmaktadır. Ayrıca sistemde kullanılan renk tanımlaması uygulaması için HSV ve RGB renk uzaylarından faydalanılarak, Python programlama dilinde hazırlanan bir kod ile nesne tespitinin doğruluk hassasiyetinin artırılması hedeflenmiştir. Nesne tespit aşamasında alınan görüntülerde oluşabilecek gürültüler için de filtre kullanılmıştır.

Kamera aracılığı ile alınan eşik(threshold) görüntüden nesneler bulunurken, yakalanan görüntü ile eşik görüntü eşleştirildikten sonra hazırlanan program, nesnenin olduğu alanı beyaz piksel bölgelerini aramaya ve nesne olarak nitelendirdikleri taktirde bu bölgeler bellekte saklanmaktadır. Söz konusu bölgelerin bulunabilmesi için OpenCv kütüphaneleri findContours adlı bir işleve sahiptir. FindContours'u kullanmak için, eşlenmiş görüntü önce 8 bitlik tek kanallı bir görüntüye dönüştürülmelidir (HSV veya RGB görüntülerde üç adet 8 bitlik kanal bulunur). Dönüşüm elzem olmasada herhangi bir değişiklik yapılmasını önlemek için yine de bir tampon görüntü oluşturulmalıdır [29]. Ardından program görüntüyü, tek bir renk alanını çevreleyen noktaların bir dizisini oluşturan, temelde vektörler olan konturlar için aramaktadır.

Kavisli nesneler söz konusu olduğunda, bu vektörler pürüzsüz şekiller yanılsaması yaratacak şekilde büyük miktarlarda noktalar içermekte ve işlev ile kaynak görüntüyü değiştirme özelliğine sahip olmaktadır. Bu sayede daha az noktanın elde edilmesi sağlanmış olacaktır. Şekil 2.15'de kavisli olan ve olmayan nesnelerin tespitindeki açıklayıcı noktaları gösterilmiştir. Bu çalışmada kavisli nesneler referans alınmıştır. Tersine, dikdörtgen bir nesnenin konturunu bulurken ise sadece dört nokta gerekmektedir. Bu şekilde dikdörtgen oluşturulmaktadır.



Şekil 2.15. Nesnelerin açıklayıcı noktaları örneği

Görüntüdeki tüm lekeler hakkında bilgi topladıktan sonra program, her bir kontur içindeki alanlar için alan boyutunu ve merkezi koordinatları hesaplayabilmektedir.
Onların alan değeri varsayılan veya kullanıcı tanımlı minimum değerleri aşar ve en yüksek tutarın altında kalırsa bu konture bağlı alanların sadece nesneleri olarak kabul edilmektedir. Kullanıcı bunu yürütme zamanında belirtmemişse, varsayılan nesne boyutu sınırları hesaplanır ve böylece görüntü en küçük nesnelerden 100 veya en büyük nesnelerden 4 enine 100 sığdırılabilir. Örneğin, varsayılan bir görüntü yakalama görüntüsü boyutuyla (256x256), en küçük nesneler en az 656 piksel içerirken en büyük nesne 16.384'e kadar piksel içerebilmektedir.
Bu yöntem ve metotlar ile nesne tespiti ve takibi için tanımlı nesnenin doğruluğu sağlanmış bir sistem oluşturulmuştur.

2.2.3. DC Motorlar ve Kontrolleri

Bu çalışmada dört tekerlek için sol ve sağ tekerlekler olmak üzere birer adet dc motorun bağlı olduğu robot gövdesi kullanılmıştır. Tekerleklerin hareketi için kullanılan motorların takibi için bu motorları saat yönünde veya tersi istikamet ve hareketlerinin sağlanması için H köprüsü kullanılmıştır.

DC motor, elektrik enerjisini (direkt akım sistemini) mekanik enerjiye dönüştüren bir cihazdır. Bir DC motorun temel olarak yapısı, komütatör segmentleri ve fırçalar üzerinden besleme ucuna bağlanan bir akım taşıma armatürü içermektedir. Armatür, sabit olarak veya elektromıknatısın kuzey güney kutupları arasına yerleştirilmektedir. Armatürde doğru akım sağlandıkça, mıknatısın elektromanyetik etkisinden dolayı mekanik bir kuvvet oluşmaktadır. DC motorun armatür iletkenleri üzerinde etkili olan kuvvetin yönünü belirlemek için Fleming'in sol el kuralı kullanılmaktadır. Akım taşıyan bir iletken, manyetik alana dik olarak yerleştirilirse, iletken hem alanın yönüne hem de akım taşıma iletkenine karşılıklı olarak dik yönde bir kuvvet ile karşılaşmaktadır. Fleming'in Sol El Kuralı, motorun dönüş yönünü belirlemektedir. Sol el işaret parmağı, orta parmağı ve baş parmağı birbirine dik olarak uzatıldığında orta parmak iletkendeki akımın yönünü gösterirken işaret parmağı manyetik alanın yönünü baş parmak ise mekanik kuvvetin yönünü göstermektedir. Bir elektrik alanın (E) ve manyetik alanın (B) etkisi altında bir "v" hızda etki için sonsuz derecede küçük bir şarj dq'si yapıldığı zaman, Lorentz Force (dF),

$$dF = dq(E + vB) \quad \textbf{(2.4)}$$

DC motorun çalışması için, E = 0 dikkate alınarak düşünecek olursak;

$$dF = dq \times v \times B \quad \textbf{(2.5)}$$

Örnek olarak; dq v ve manyetik alan B'nin vektör çarpımıdır;

$$dF = dq \times (dL \div dt) \times B \; [V = dL \div dt] \quad \textbf{(2.6)}$$

Burada, dL iletken taşıma yükünün q uzunluğudur.

$$dF = dq \times (dL \div dt) \times B \; ya \; da \; dF = IdL \times B \;\; burada \; I = dq \div dt \; ya \; da$$

$$F = B \times I \times L \times \sin\theta \ \ olacaktır. \tag{2.7}$$

Bir DC motorun yapısı, armatür iletkeni boyunca akımın doğrultusunda alana dik olacak şekildedir. Bu nedenle kuvvet, her iki eşit alana dik doğrultuda armatür iletkeni üzerinde hareket ederken akım sabit kalmaktadır.

$Örnek: \theta = 90°$

Bu durumda, armatür iletkeninin sol tarafındaki akımı '+I' olarak alacak ve armatür iletkeninin sağ tarafındaki akım '-I' olacaktır, çünkü akımlar birbirinin zıt yönünde akmaktadır.

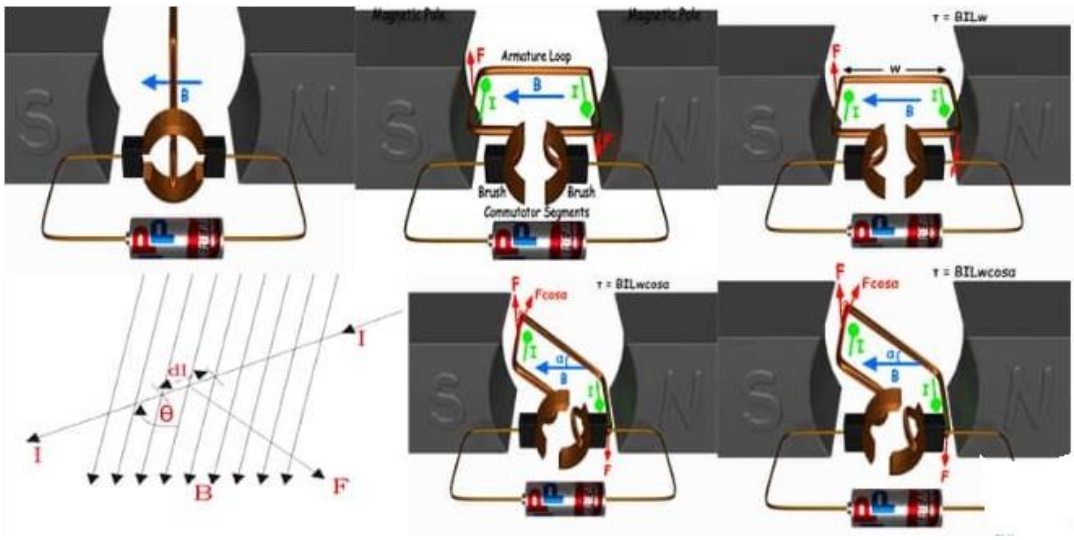
Ardından sol taraftaki armatür iletkeni üzerindeki kuvvet,

$$Fi = B \times I \times L \times \sin 90 = B \times I \times L \tag{2.8}$$

Benzer şekilde, sağ taraftaki iletken üzerindeki kuvvet,

$$Fr = B \times (-I) \times L \times \sin 90 = -B \times I \times L \tag{2.9}$$

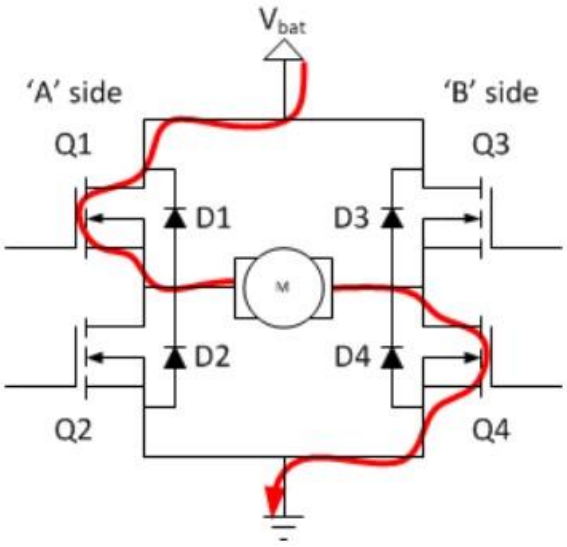
Bu nedenle, bu pozisyonda her iki taraftaki kuvvetin büyüklükte fakat tersi yönde ise eşit olmaktadır. İki iletken, w = genişlik armatür dönüşü ile ayrıldıklarından iki zıt kuvvet, dönme kuvvetinin veya armatür iletkeninin dönmesine neden olan bir tork üretmektedir [30].



Şekil 2.16. DC Motor *[30]*.

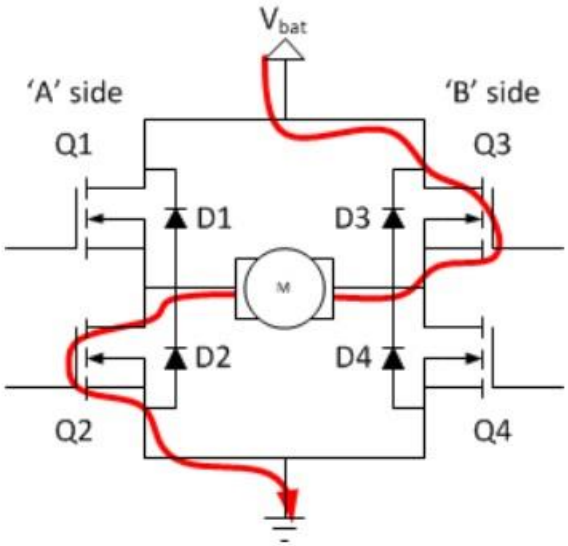
Bütün bu bilgiler ışığında bir DC motorun çalışma prensibi Şekil 2.16'da gösterilmiştir. Motor sürücüler ise motorlar ile sistem arasındaki kontrolü, bağlantıları sağlayan elektronik kartlardır. Bu çalışmada H köprüsü L298 motor sürücü kartı kullanılmıştır.

H-Köprü DC Motorun uçlarını bir gerilim kaynağına bağlarsak motor bir yönde dönmeye başlamaktadır. Fakat daha sonra gerilim kaynağının uçları yer değiştirildiğinde motor ters yönde hareket etmektedir. DC Motorların yön kontrolünü sağlayabilmek için H-Bridge (H-Köprü) yöntemi geliştirilmiştir. H-Bridge genel olarak 4 adet transistor, diyot ya da MOSFET ile gerçekleştirilen motorun iki yönlü dönebilmesini sağlayan bir yöntemdir. Şekil 2.17' de Q1 ve Q4 anahtarları kapanmış ve akım akmaya başlayarak motor dönüşü başlamıştır.



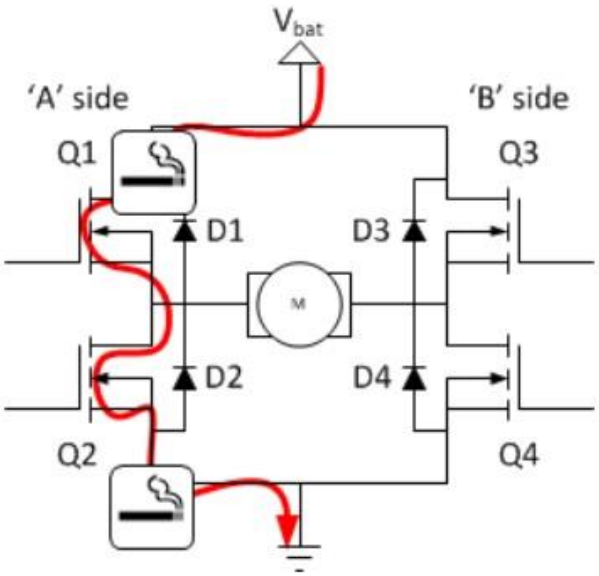
Şekil 2.17. H köprüsü

Motorun diğer yöne dönmesini sağlamak için, Q2 ve Q3 anahtarlama elemanlarının kapatılması gerekmektedir. Bu sefer Şekil 2.18' de görüldüğü gibi akım ters yönden akmaya başlar bu nedenle motor ters yönde dönmeye başlayacaktır.



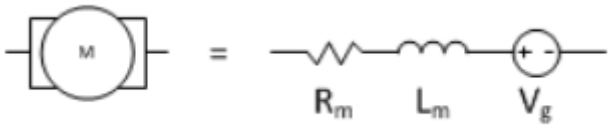
Şekil 2.18. Ters yönde dönen H köprüsü

H-bridge devresinde kesinlikle yapılmaması gereken anahtarlama kombinasyonu; Q1 ve Q2 ya da Q3 ve Q4 anahtarlarını aynı anda kapatmaktır. Bu durumda güç kaynağı ile topraklama kısa devre olacağından sistem zarar görecektir. Şekil 2.19' da devrede akımın nasıl aktığı ve kısa devre durumu görülmektedir.



Şekil 2.19. H köprüsü kısa devre

Elektriksel olarak bakıldığında, motor temel olarak manyetik alanda hareket eden bobinler içermektedir. Bobinler endüktans ve iç dirence sahiptirler. Manyetik alandaki hareketleri voltaj üretir ve buna jeneratör voltajı denirken Vg ile gösterilmektedir. Motor modeli Şekil 2.20'de görülmektedir.



Şekil 2.20. Dc motor modeli

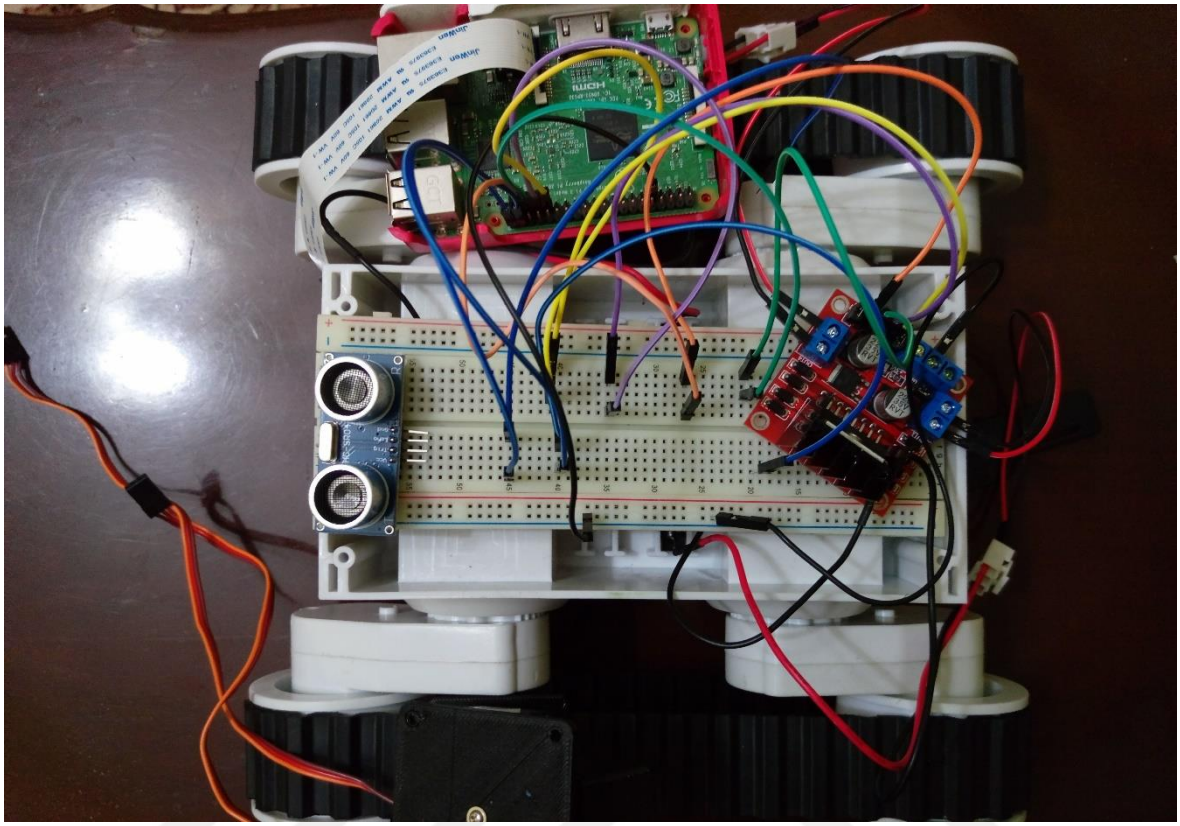
Şekil 2.20' de görülen iç direnç genellikle ihmal edilmektedir. Jeneratör voltajı (Vg), manyetik alanda hareket eden bobinin hızına bağlıdır.

Eğer motorun dönme hızı değiştirilmek istenirse, uygulanan voltajı azaltıp çoğaltılarak motorun hızı değiştirilmektedir. Böylelikle, H-bridge ile motorun hem yönü hem de hızı kontrol edilebilmektedir [31].

Bu çalışmada L298 motor sürücüsünün kullanım nedeni H köprüsü prensibine sahip olma ve maliyetinin çok düşük olmasıdır.

Bu şekilde oluşturulan sistem ile alınan anlık görüntüler görüntü işleme teknikleri ile analiz edilerek nesne tespiti sağlanmaktadır. Yazılan program sayesinde, tespit edilen nesnenin hareketi algılandığında, sistemdeki dc motorlar motor sürücü yardımıyla kontrol edilerek sistemin nesneyi takibi için gerekli hareketi oluşturmaktadır. Nesnenin tespiti motorların çalışmasını tetiklemektedir. Sistemde oluşabilecek nesnenin kamera açısından çıkması vb. durumlar için sistemin olduğu yerde 360° dönüş yapması sağlanarak kameranın görüntü açısından çıkan nesnenin yeniden tespiti sağlanmaktadır. Sistem için gerekli olan güç kaynağı uygulama kartı için taşınabilir güç kaynağı kullanılırken, motorlar için gerekli enerji 9V şarj edilebilir pil ile sağlanmıştır.

2.3. Robot Gövdesi ve Ekipmanlar



Şekil 2.21. Raspberry Pi kontrollü nesne takip robotu modeli

Çalışmada kullanılan ekipmanlar Şekil 2.21'de gösterilmiştir. Buna göre çalışma için;

- 1 adet Rover 5 motor robot gövdesi,
- Nesne takibi sağlanması için Pi kamera,
- 1 adeti Raspberry pi model 3,
- 1 adet L298N motor sürücüsü,
- 1 adet 9V pil,
- Robot ile nesneler arası mesafelerin takibi için 3 adet ultrasonik sensor,
- 1 adet taşınabilir güç kaynağı,
- Devrede kullanılmak için dirençler ve kablolar,

2.4. Çalışmada Kullanılan Yazılımlar

2.4.1. Görüntü İşleme Kütüphaneleri ve OpenCV

Görüntü işleme projeleri için kullanılacak kütüphane yapılacak projenin amacına uygun seçilmelidir. Çünkü farklı uygulamalar için kullanılan birçok görüntü işleme kütüphanesi mevcuttur. Bunlar Matlab, Halcon, OpenFrameworks, CIMG, Fiji, EndrovImagej, Leadtools, Pink, Boost ve OpenCV bu kütüphanelerden bazılarıdır. Bu çalışmada OpenCV kullanılmıştır. OpenCV (Open Source Computer Vision) açık kaynak kodlu görüntü işleme kütüphanesi olup 1999 yılında Intel firması tarafından geliştirilmiş ve ilk sürümü 2000 yılında BSD lisansı altında piyasaya sürülmüştür. BSD lisansına sahip olması, bu kütüphanenin ücretsiz olarak projelerde kullanabileceğini ifade etmektedir. OpenCV Windows, Linux, FreeBSD, Android, Mac OS, IOS platformlarında çalışabilmektedir. C, C++, Python, Java, Matlab, EmguCV programları ile de kullanılabilmektedir. OpenCV kütüphanesinde görüntü işlemeye ve makine öğrenmesine yönelik 2500'den fazla algoritma bulunmaktadır. Bu algoritmalar ile yüz tanıma, nesneleri ayırt etme, insan hareketlerini tespit edebilme, nesneleri sınıflandırma, plaka tanıma, üç boyutlu görüntü üzerinde işlem yapabilme, görüntü karşılaştırma, optik karakter tanımlama gibi işlemler yapılabilmektedir. Ayrıca OpenCV kütüphanesini oluşturan birçok bileşenler mevcuttur. Bunlar;

Core: OpenCV'nin temel fonksiyonları ve matris, point gibi veri yapılarını bulundurmaktadır.

HighGui: Resim görüntüleme, pencereleri yönetme ve grafiksel kullanıcı arabirimleri için gerekli olabilecek metotları barındırmaktadır.

Imgproc: Filtreleme operatörleri, kenar bulma, nesne belirleme, renk uzayı yöntemi, renk yönetimi gibi tüm fonksiyonları barındırmaktadır.

Imgcodecs: Dosya sistemi üzerinden resim ve video okuma yazma işlemlerini yerine getiren metotlar içermektedir.

Videoio: Kameralara ve video cihazlarına erişip görüntü almak ve yazmak için gerekli metotları barındırmaktadır. Bu çalışmada OpenCV'nin bütün bileşenlerinden faydalanılarak Python programlama dili kullanılmıştır.

OpenCV tüm belleği otomatik olarak işlemektedir. Her şeyden önce, std:: vector, cv :: Mat ve işlev ve yöntemlerin kullandığı diğer veri yapıları, gerektiğinde altta yatan bellek arabelleklerini ayıran yıkıcılara sahiptir. Bu, yıkıcılar, Mat durumunda olduğu gibi tamponları her zaman ayırmadıkları anlamına gelmektedir. Olası veri paylaşımını dikkate

almaktadırlar. Bir yıkıcı, matris veri arabelleği ile ilişkili referans sayacını azaltmaktadır. Tampon, sadece ve eğer referans sayacı sıfıra ulaşırsa, yani başka bir yapı aynı tampona atıfta bulunmazsa tahsis edilmektedir. Benzer şekilde, bir Mat örneği kopyalandığında, gerçek veriler gerçekten kopyalanmazken, bunun yerine, referans sayacı, aynı verinin bir başka sahibinin olduğunu ezberlemek için artırılmaktadır. Matris verilerinin tam bir kopyasını oluşturan Mat :: clone yöntemi de bulunmaktadır.

OpenCV hafızayı otomatik olarak ayırırken ayrıca hafızayı otomatik olarak çıkış fonksiyonu parametrelerine de otomatik ayırmaktadır. Yani, eğer bir fonksiyon bir veya daha fazla giriş dizisine (cv :: Mat örneği) ve bazı çıkış dizilerine sahipse, çıkış dizileri otomatik olarak veya yeniden tahsis edilmektedir. Çıkış dizilerinin büyüklüğü ve tipi, giriş dizilerinin boyutundan ve türünden belirlenmektedir. Gerekirse, işlevler çıkış dizisi özelliklerini anlamaya yardımcı olan fazladan parametreler de almaktadır.

Video karesi çözünürlüğü ve bit derinliği video yakalama modülünden bilindiğinden, dizi çerçevesi >> operatör tarafından otomatik olarak tahsis edilmektedir. Dizi kenarları cvtColor işlevi tarafından otomatik olarak ayrılmaktadır. Ayrıca giriş dizisi ile aynı boyuta ve bit derinliğine sahiptir. Kanal sayısı 1'dir, çünkü cv :: COLOR_BGR2GRAY renk dönüştürme kodu geçirilir, bu da gri tonlama dönüşümüne bir renk anlamına gelmektedir. Çerçeve ve kenarların, sonraki bütün video karelerinin aynı çözünürlüğe sahip olması nedeniyle döngü gövdesinin ilk çalıştırılması sırasında yalnızca bir kez ayrılmaktadır. Bir şekilde video çözünürlüğünün değişmesi durumunda, diziler otomatik olarak yeniden tahsis edilmektedir.

Bir bilgisayar görmesi kütüphanesi olarak, OpenCV, çoğu zaman bir kanal başına, 8 veya 16 bitlik bir kanal biçiminde kodlanan ve dolayısıyla sınırlı bir değer aralığına sahip olan görüntü pikselleriyle ilgilenmektedir. Ayrıca, renk uzay dönüşümleri, parlaklık / kontrast ayarlamaları, keskinleştirme, karmaşık enterpolasyon (çift kübik, Lanczos) gibi görüntüler üzerindeki bazı işlemler, kullanılabilir aralıktaki değerler üretebilmektedir. Sonucun en düşük 8 (16) biti kadar bilgi saklanırsa, bu görsel yapaylıklarla sonuçlanır ve başka bir görüntü analizini etkileyebilmektedir. Bu problemi çözmek için uygulanan yöntem, satürasyon aritmetiğidir. Örneğin, bir işlemin sonucunu 8 bitlik bir görüntüye kaydetmek için, 0..255 aralığındaki en yakın değeri bulunmalıdır:

$$l(x, y) = min(max(round(r), 0), 255) \quad \textbf{(2.11)}$$

Benzer kurallar 8 bit imzalı, 16 bit imzalı ve imzasız türlerde de uygulanmaktadır. Bu semantik, kütüphanede her yerde kullanılmaktadır.

Birçok OpenCV işlevi yoğun 2 boyutlu veya çok boyutlu sayısal dizileri işlemektedir. Genellikle, bu işlevler cppMat'i parametre olarak almakta ancak bazı durumlarda std :: vector<> (örneğin bir nokta kümesi için) veya cv :: Matx<> (3x3 homografi matrisi ve benzeri için) kullanmak daha uygun olmaktadır. API'da çoğaltmaları önlemek için özel "proxy" sınıfları tanıtılmıştır. Temel "proxy" sınıfı, cv :: InputArray'dir. Bir işlev girişindeki salt okunur dizileri geçirmede kullanılmaktadır. InputArray sınıfından cv :: OutputArray türetilmiş bir işlev için bir çıkış dizisi belirtmek için kullanılmaktadır. Normal olarak, bu ara türlere dikkat edilmemelidir, çünkü her şey otomatik olarak çalışmaktadır. InputArray / OutputArray yerine Mat, std :: vector<>, cv :: Matx<>, cv :: Vec<> veya cv :: Scalar 'ı kullanılabilir. Bir işlev isteğe bağlı bir girdi veya çıktı dizisine sahipse, cv :: noArray () öğesi es geçilmelidir.

OpenCV hata kontrolü konusunda ise, kritik hataları işaretlemek için istisnalar kullanılmaktadır. Giriş verisi doğru bir formata sahipse ve belirtilen değer aralığına aitse, ancak algoritma bir nedenden dolayı başarılı olamaz ise (örneğin, optimizasyon algoritması yakınsamadı), özel bir hata koduna (tipik olarak sadece bir boole değişkeni) döndürmektedir.

İstisnalar, cv :: Exception sınıfı veya türevlerinin örnekleri olabilmektedir. Bu yüzden diğer standart C ++ kütüphane bileşenlerini kullanarak kod içinde incelikle ele alınabilmektedir.

Özel durum genellikle ya CV_Error (errcode, description) makrosu ya da printf benzeri CV_Error_ (errcode, (printf-spec, printf-args)) varyantı kullanılarak ya da koşulu kontrol eden CV_Assert (koşul) makrosu kullanılarak atılmaktadır. Performans açısından kritik kod için, yalnızca Debug yapılandırmasında tutulan CV_DbgAssert (koşul) var olmaktadır. Otomatik bellek yönetimi sayesinde, tüm ara tamponlar ani bir hata durumunda otomatik olarak tahsis edilmektedir. Gerekirse istisnaları yakalamak için yalnızca bir try ifadesi eklenmesi gerekmektedir.

Mevcut OpenCV uygulaması tamamen yeniden girilebilmekte, aynı işlev veya farklı sınıf örneklerinin aynı yöntemleri farklı iş parçacıklarından çağrılabilmektedir. Ayrıca, aynı Mat, farklı durumlar içinde kullanılabilir, çünkü referans sayma işlemleri, mimariye özgü talimatları kullanılmaktadır [37].

Bu kütüphanenin anlatılmaya çalışılan bütün özellikleri ve Python programlama dili ile kombine çalışabilme özelliğinden dolayı tercih edilmiştir.

2.4.2. Python Programlama Dili

Python, dinamik semantik ile yorumlanmış, nesne yönelimli, yüksek seviyeli bir programlama dilidir. Dinamik yazım ve dinamik ciltleme ile birleştirilen yüksek düzeydeki veri yapıları, hızlı uygulama geliştirme yapabilmek için uygulamayı çok çekici hale getirmektedir. Mevcut bileşenleri birbirine bağlamak için bir komut dosyası veya yapıştırıcı dili olarak kullanılmasını sağlamaktadır. Python'un basit, öğrenmesi kolay sözdizimi okunabilirliği vurgular ve bu nedenle program bakım maliyetini azaltmaktadır. Python, program modülerliği ve yeniden kod kullanımını teşvik eden modülleri ve paketleri de desteklemektedir. Python yorumlayıcısı ve geniş standart kütüphanesi, tüm büyük platformlar için ücretsiz kaynak olarak kullanılabilir ve serbestçe dağıtılabilir olma özelliğine de sahiptir.

Çoğu zaman, programcılar sağladığı yüksek üretkenlik nedeniyle Python'u tercih etmektedirler. Derleme adımı olmadığından, düzenleme testi hata ayıklama döngüsü inanılmaz derecede hızlıdır. Python programında hata ayıklama sistemi oldukça kolaydır bir hata veya olması gerekenden farklı kod girişi hiçbir zaman bölümleme hatasına neden olmamaktadır. Bunun yerine, yorumlayıcı bir hata bulduğunda, bir istisna durum ortaya çıkarmaktadır. Program istisnayı yakalamadığında, tercüman yığın dizini yazdırmaktadır. Kaynak seviyesinde bir hata ayıklayıcı, yerel ve global değişkenlerin incelenmesine, keyfi ifadelerin değerlendirilmesine, kesme noktalarının ayarlanmasına, kod boyunca bir satırda adım atılmasına vb. izin vermektedir. Hata ayıklayıcı, Python'un introspektif gücüne işaret ederek Python'da yazılmıştır. Diğer taraftan, genellikle bir programda hata ayıklamanın en hızlı yolu, kaynağa birkaç yazdırma ifadesi eklemektir, hızlı düzenleme testi hata ayıklama döngüsü bu basit yaklaşımı çok etkili kılmaktadır.

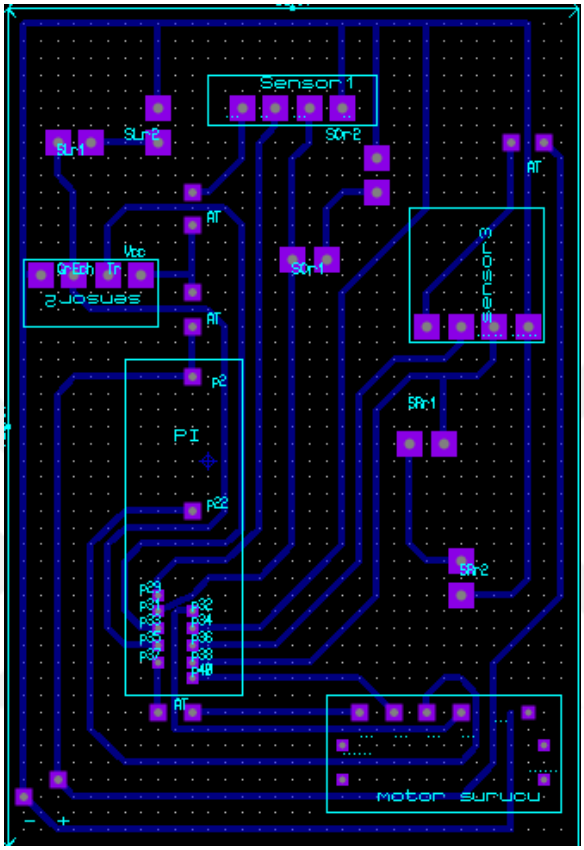
Python; C ve C++ gibi programlama dilleri ile kıyaslandığında,

a. Daha kolay öğrenilmekte.
b. Program geliştirme sürecini kısaltarak daha hızlı program yazma imkanı sağlamakta.
c. Yukarıdaki verilen programlama dillerine aksine ayrı bir derleyiciye ihtiyaç duymamakta.
d. Hem daha okunaklı, hem de temiz kodsal söz dizimine sahip olmaktadır.

Python'un bu ve buna benzer özelliklerinden dolayı, dünya çapında büyük kuruluşlar (Google, Yahoo, Dropbox) bünyelerinde Python programcıları çalıştırmaktadırlar [32].

2.5.Pnp Yöntemi ile Baskı Devre Hazırlanması

Breadboard ile hazırlanan devrenin Proteus Ares programı kullanılarak baskı devresi çizilmiştir.



Şekil 2.22. Ares programında hazırlanan baskı devre

Şekil 2.22' deki gibi programda hazırlanan baskı devre, lazer yazıcı ile kuşe kağıda çıkartıldı. Alınan çıktı bakırlı plakete yerleştirilerek ütüleme işlemine tabii tutularak kağıttaki baskı devresinin bakırlı plakete geçirildi. Kâğıt bakır plaketin üstünden çıkarıldıktan sonra bakır plaket tuz ruhu ve perhidrol karışımının bulunduğu bir kaba atılarak bakır yüzeyler karışım içerisinde eritildikten sonra, bakır plaket temizlenerek baskı devre çizimi tamamlandı. Daha sonra devre elemanlarının montajı için baskı devredeki pin noktaları ince uçlu matkap ile delinerek elektronik devre elemanları ve diğer bağlantılar lehimlenerek elektronik kart tamamlandı.

Baskı devre ve diğer ekipmanların montaj ve robot üstüne yerleştirmeleri yapıldıktan sonra bilgisayar ve Raspberry Pi arasında internet bağlantısı üzerinden kablosuz bağlantı kurularak, Python programında hazırlanan yazılım çalıştırılarak sistem test edildi ve başarılı sonuçlar alındı.

2.6. OpenCV-Python

OpenCV-Python bilgisayar görme problemlerini çözmek için tasarlanmış bir Python bağlamaları kütüphanesidir. Python, basitliği ve kod okunabilirliği nedeniyle çok hızlı bir şekilde popüler hale gelen Guido van Rossum tarafından başlatılan genel amaçlı bir programlama dilidir. Programcının okunabilirliği azaltmaksızın daha az kod satırında fikir ifade etmesini sağlamaktadır. C / C ++ gibi dillerle karşılaştırıldığında, Python daha yavaştır. Bu, Python'un C / C ++ ile kolayca genişletilebildiğini ve bu sayede C / C ++ 'da hesaplamalı olarak yoğun kod yazmamızı ve Python modülleri olarak kullanılabilen Python sarmalayıcıları oluşturmamızı sağlar. Bu bize iki avantaj sağlar. Birincisi, kod orijinal C / C ++ kodu kadar hızlıdır ve ikinci olarak, C / C ++ 'dan Python kodlaması daha kolaydır. OpenCV-Python, orijinal OpenCV C ++ uygulaması için bir Python sarıcısıdır.

OpenCV-Python, MATLAB tarzı bir sözdizimi ile sayısal işlemler için yüksek düzeyde optimize edilmiş bir kütüphane olan Numpy'yi kullanır. Tüm OpenCV dizi yapıları, Numpy dizilerine dönüştürülür. Bu ayrıca, Scipy ve Matplotlib gibi Numpy'yi kullanan diğer kitaplıklarla bütünleşmeyi kolaylaştırır.

OpenCV'de, tüm algoritmalar C ++ uygulamasında uygulanır. Ancak bu algoritmalar Python, Java vb. Gibi farklı dillerden kullanılabilir. Bu, bağlayıcı üreticilerince mümkündür. Bu jeneratörler, kullanıcıların C ++ işlevlerini Python'dan çağırmasını sağlayan C ++ ve Python arasında bir köprü oluşturur. Arka planda olup bitenlerin tam bir resmini elde etmek için, iyi bir Python / C API bilgisi gereklidir. Bu yüzden OpenCV'de ki tüm fonksiyonları, sarıcı işlevlerini elle yazarak Python'a genişletmek zaman alıcı bir görevdir. OpenCV daha akıllı bir şekilde bu işlemleri hazır şekilde yapar. OpenCV, bu sarmalayıcı işlevlerini, C ++ başlıklarından otomatik olarak, / Python / src2 modüllerinde bulunan bazı Python komut dosyalarını kullanarak üretir.

İlk olarak, modüller / python / CMakeFiles.txt, modüllerin Python'a genişletilmesini denetleyen bir CMake betiğidir. Uzatılacak olan tüm modülleri otomatik olarak kontrol edecek ve başlık dosyalarını yakalayacaktır. Bu başlık dosyaları, söz konusu modüller için tüm sınıfların, işlevlerin, sabitlerin vb. listesini içerir.

İkincisi, bu başlık dosyaları bir Python betiğine, / python / src2 / gen2.py modüllerine aktarılır. Bu Python bağlamaları jeneratör komut dosyasıdır. Başka bir Python script modülleri / python / src2 / hdr_parser.py çağırır. Bu, başlık ayrıştırıcı betiğidir. Bu başlık ayrıştırıcı, tüm üstbilgi dosyasını küçük Python listelerine ayırır. Yani bu listeler belirli bir

fonksiyon, sınıf vb. İle ilgili tüm detayları içerir. Örneğin, fonksiyon ismini, dönüş tipini, giriş argümanlarını, argüman türlerini içeren bir liste almak için bir fonksiyon ayrıştırılacaktır. Final listesi tüm fonksiyonların detaylarını, yapıları içerir. Bu başlık dosyasındaki sınıflar vb. olarak söyleyebiliriz. Ancak başlık ayrıştırıcı, başlık dosyasındaki tüm işlevleri / sınıfları ayrıştırmaz. Geliştirici, hangi fonksiyonların Python'a aktarılması gerektiğini belirlemelidir. Bunun için, başlık ayrıştırıcısının ayrıştırılacak işlevleri tanımlamasını sağlayan, bu bildirimlerin başına eklenen belirli makrolar vardır. Bu makrolar, belirli işlevi programlayan geliştirici tarafından eklenir. Kısacası, geliştirici, hangi fonksiyonların Python'a genişletileceğine ve bulunmayacağına karar vermektedir. Böylece başlık ayrıştırıcı, ayrıştırılmış işlevlerin son büyük bir listesini döndürür. Jeneratör komut dosyası (gen2.py), başlık ayrıştırıcı tarafından ayrıştırılan tüm işlevler / sınıflar / enumlar / yapılar için sarıcı işlevler oluşturacaktır (Bu üstbilgi dosyalarını derleme sırasında / modules / python / klasöründe py opencv_generated _ *. H dosyaları olarak bulunabilir.). Ancak Mat, Vec4i, kullanıcıya bazı temel OpenCV veri türleri olabilir. Manuel olarak genişletilmeleri gerekir. Örneğin, bir Mat tipi Numpy dizisine genişletilmelidir, Boyut iki tamsayıdan oluşan bir tüple uzatılmalıdır. Benzer şekilde, elle genişletilmesi gereken bazı karmaşık yapılar / sınıflar / fonksiyonlar vs. olabilir. Bu tür manuel sarıcı işlevleri, modüller / python / src2 / cv2.cpp dizinlerine yerleştirilir. Ardından, cv2 modülünü veren bu sarıcı dosyalarının derlenmesidir. Yani bir işlevi çağırdığımızda, Python'da res = equalizeHist (img1, img2) denilip, iki numpy dizisi geçirilirse çıktı olarak başka bir numpy dizisi beklenir. Böylece bu numpy dizileri cv :: Mat'a dönüştürülür ve daha sonra C ++ 'da equalizeHist () işlevini çağırır. Son sonuç, resNumpy dizisine dönüştürülür. Kısacası, hemen hemen tüm operasyonlar C ++ ile yapılır, bu da bize C ++ ile neredeyse aynı hızı verir.

2.7. Python'u Diğer Programlama Dilleri ile Karşılaştırma

Python genellikle Java, JavaScript, Perl, Tcl veya Smalltalk gibi diğer yorumlanmış dillerle karşılaştırılır. C ++, CommonLisp ve Scheme ile karşılaştırmalar da aydınlatıcı olabilmektedir. Pratikte, bir programlama dilinin seçimi genellikle maliyet, kullanılabilirlik, eğitim ve önceki yatırımlar vs. tarafından belirlenir [33]. Bu yönler oldukça değişken olduğundan, genel hatları ile incelediğimizde oluşan farklılıklar her bir programlama dili için şunlardır;

Java: Python programlarının genellikle Java programlarından daha yavaş çalışması beklenir, ancak hazırlanması daha az zaman almaktadır. Python programları, eşdeğer Java programlarından tipik olarak 3-5 kat daha kısadır. Bu fark Python'un yerleşik üst düzey veri tiplerine ve dinamik yazımına bağlanabilir. Örneğin, bir Python programcısı argüman veya değişken türlerini bildiren zaman harcamaz ve Python'un güçlü sözdizimsel desteği dil için doğrudan yapılandırılmış güçlü polimorfik liste ve sözlük türleri hemen hemen her Python programında bir kullanım bulmaktadır. Çalışma zamanı yazım farklılıklarından dolayı, Python'un çalışma zamanı Java'dan daha fazla çalışmalıdır. Örneğin, a + b ifadesini değerlendirirken, önce derleme zamanında bilinmeyen türlerini bulmak için a ve b nesnelerini incelemelidir. Daha sonra aşırı yüklenmiş kullanıcı tanımlı bir yöntem olabilecek uygun ekleme işlemini başlatır. Öte yandan, Java, etkin bir tamsayı veya kayan nokta ekleme gerçekleştirebilir, ancak a ve b için değişken bildirimler gerektirir ve kullanıcı tanımlı sınıfların örnekleri için + işlecinin aşırı yüklenmesine izin vermez.

Bu nedenlerden dolayı, Python bir programlama dili olarak çok daha iyi, Java ise düşük seviyeli bir uygulama dili olarak daha iyi karakterize edilmektedir. Aslında, ikisi birlikte mükemmel bir kombinasyon oluşturmaktadır. Bileşenler Java'da geliştirilebilir ve Python'daki uygulamaları oluşturmak için birleştirilebilir. Python, bir Java uygulamasında tasarımlarının "sertleştirilinceye" kadar bileşenleri prototiplemek için de kullanılabilir. Bu tür bir gelişmeyi desteklemek için, Java'da yazılmış bir Python uygulaması geliştirilmekte olup, bu da Python kodunun Java'dan çağrılmasına olanak tanımaktadır ve bunun tersi de geçerlidir. Bu uygulamada, Python kaynak kodu, Java bayt koduna çevrilir (Python'un dinamik semantiğini desteklemek için bir çalışma zamanı kitaplığından yardım alarak).

JavaScript: Python'un "nesne tabanlı" alt kümesi, kabaca JavaScript'e eşdeğerdir. JavaScript gibi (Java'nın aksine), Python, sınıf tanımlarına girmeden basit fonksiyonları ve değişkenleri kullanan bir programlama stilini destekler. Öte yandan, Python, sınıfların ve

öncesinin önemli bir rol oynadığı gerçek bir nesne yönelimli programlama stili ile çok daha büyük programların yazılmasını ve daha iyi kodların yeniden kullanılmasını desteklemektedir.

Perl: Python ve Perl, benzer bir arka plandan (her ikisi de uzun zamandır kullanılan Unix ’den sağlanmakta) geliyor ve benzer birçok özelliği barındırmalarına rağmen farklı bir felsefeye sahiptirler. Perl, yaygın uygulama odaklı görevler için desteği vurgular. Yerleşik normal ifadeler, dosya tarama ve rapor oluşturma özellikleri ile öne çıkar. Python ise, veri yapı tasarımı ve nesne yönelimli programlama gibi yaygın programlama metodolojileri için desteği vurgulamakta ve programcıları zarif ama aşırı derecede şifreli bir notasyon sunarak okunabilir (ve dolayısıyla sürdürülebilen) kod yazmaya teşvik etmektedir. Sonuç olarak, Python Perl'e yakın bir uygulama olarak gözükmektedir, ancak orijinal uygulama alanında nadiren üstündür. Ancak Python, Perl'in işinin ötesinde bir uygulanabilirliğe sahiptir.

Tcl: Python gibi, Tcl bir uygulama uzantısı dili ve tek başına bir programlama dili olarak kullanılabilir. Ancak, tüm verileri geleneksel olarak dizge olarak saklayan Tcl, veri yapıları üzerinde zayıftır ve tipik kodu Python'dan çok daha yavaş yürütür. Tcl ayrıca modüler ad alanları gibi büyük programlar yazmak için gereken özelliklere sahip değildir. Bu nedenle, Tcl'yi kullanan "tipik" büyük bir uygulama, genellikle C veya C ++ ile yazılmış, bu uygulamaya özgü Tcl uzantılarını içerirken, eşdeğer bir Python uygulaması genellikle "saf Python" ile yazılabilir. Elbette, saf Python gelişimi bir C veya C ++ bileşenini yazmaktan ve hata ayıklamaktan çok daha hızlıdır. Tcl'nin tek kullanımlık kalitesinin Tk araç takımı olduğu söylenmiştir. Python, standart GUI bileşen kitaplığı olarak Tk'ye bir arabirim benimsemektedir.

Tcl 8.0, sınırlı veri tipi desteğine sahip bir bayt kod derleyici sağlayarak hıza hitap eder ve ad alanları ekler. Ancak, yine de çok daha hantal bir programlama dilidir.

Smalltalk: Python ve Smalltalk arasındaki belki de en büyük fark Python'un daha "ana akım" sözdizimidir. Smalltalk gibi, Python da dinamik yazım ve ciltleme işlevi görür ve Python'daki her şey bir nesnedir. Ancak, Python yerleşik nesne türlerini kullanıcı tanımlı sınıflardan ayırır. Smalltalk'ın standart koleksiyon veri türleri kütüphanesi daha rafine bir tasarıma sahipken, Python'un kütüphanesi Internet, WWW, e-posta, HTML ve FTP gibi gerçeklerle ilgilenmek için daha fazla imkan sunmaktadır.

Python, geliştirme ortamı ve kod dağıtımı ile ilgili farklı bir felsefeye sahiptir. Smalltalk'ın geleneksel olarak, hem ortamı hem de kullanıcının programını içeren bir monolitik "sistem görüntüsü" olduğu yerlerde, Python, hem standart modülleri hem de kullanıcı modüllerini,

sistem dışında kolaylıkla yeniden düzenlenebilen veya dağıtılabilen bireysel dosyalarda depolar. Bunun bir sonucu, Grafik Kullanıcı Arabiriminin (GUI) bir Python programına eklenmesi için birden fazla seçeneğin bulunmasıdır, çünkü GUI sistemde yerleşik değildir.
C ++ :Java için hemen hemen her şey C ++ için de geçerlidir. Python kodunun eşdeğer Java kodundan tipik olarak 3-5 kat daha kısa olduğu durumlarda, eşdeğer C ++ kodundan 5-10 kat daha kısadır. Veriler gösteriyor ki, bir Python programcısının iki yılda iki C ++ programcısının bir yılda tamamlayamadığı bir zamanda bitirebileceğini ortaya koymaktadır [33]. Python, C ++ ile yazılmış bileşenleri birleştirmek için kullanılan bir dil olarak da ön plana çıkmaktadır.

3. BULGULAR

3.1. Devrenin Tasarımı ve Çalışması

Devre için gerekli elemanların tespiti ve tedarik süreci tamamlandıktan sonra Raspberry Pi, Rover 5 robot gövdesi, Motor sürücü, Pi kamera ve ultrasonik sensörler arasında bağlantılar breadboard üzerine kurulan devrede yapıldı. Hazırlanan elektronik devre ile sistemin çalışması denendi. Raspberry Pi modülünün güç beslemesi mobil bir kaynak ile sağlanarak Wi-Fi üzerinden bilgisayar ile kablosuz bağlantısı gerçekleştirildi. Kullanılan taşınabilir güç kaynağının robot gövdesinin alt kısmına monte edilmesi için 3D yazıcı kullanılarak iki adet destek parçası hazırlanıp yerleştirilerek, güç kaynağının sabit bir şekilde robot gövdesinin altına monte edildi. Breadboard üzerine kurulan devrenin baskı devresi bakır plaket üzerine hazırlandı. Bu şekilde baskı devre, motor sürücü, 9V pil, Raspberry Pi gibi ekipmanlar robot gövdesi içerisine yerleştirilerek devrenin dizaynı tamamlanarak sistem çalıştırıldı. Ultrasonik sensörler robotun sağ, sol ve ön kısmına yerleştirilerek robotun karşılaşacağı engelleri algılaması sağlandı.

Sistem aldığı görüntülerde görüntü çerçevesini algılarken önemli olan, eğer robot birkaç çerçeve algılama yaptıktan sonra topun yönünü tahmin edemezse, ne yapacağını bilememesi durumudur. O zaman nesne kameranın kapsamı dışına çıktığında bu bir belirsizlik durumuna neden olacağından dolayı bu olumsuzluğun giderilmesi, sistemin nesneyi takip ve algılamasını sağlamak için kendi etrafında 360 derece dönüşü yazılım ile sağlanarak bu problem çözüldü.

3.2. Kamera ile görüntü yakalama

Günümüzde birçok alanda belirlenen bölgeleri sürekli izleyebilen, istenmeyen veya daha önce tahmin edilemeyen verileri belirlerken kararlar veren ve buna göre yanıt veren sistemlere ihtiyaç duyulmaktadır. Bu tarz sistemlerde, otomatik gözetlemenin gerçekleştirilmesi için bilgisayar görmesi kullanarak nesne takibinin yapılması önemlidir.

Bu projede anlık video kaydıyla, görüntüleri çekmek ve nesneyi izleyerek görüntü işlemek için kamera kullanılmıştır. Bu çalışmada tercih edilebilecek birçok kamera ve kamera modülleri olmasına rağmen, çalışmadaki uygulama kartı olan Raspberry Pi kartının kamera modülü Pi kamera tercih edilmiştir. Bunun en büyük nedeni, hem robotun donanımsal dizaynında ebatlarından dolayı küçük ve efektif olmasını sağlamak hem de esnek yapısı, kullanım kolaylığı ile birlikte, maliyetinin de düşük olmasıdır.

Raspberry Pi kamera modülünden doğrudan video çekilmesi OpenCV ile basittir ve bu noktada herhangi bir ek sürücü veya başka bir yazılımda ihtiyaç yoktur. VideoCapture sınıfını çağırarak ve bir ad atamak suretiyle işlem kolayca başlatılmaktadır. Sınıf kopyası daha sonra, her döngü başlangıcında kameradan çerçeveler almak için kullanılmaktadır. Kamerayı başlattıktan sonra, gerçekten başarılı olup olmadığını kontrol etmek gerekir. Bu, hazırlanan yazılımda basit bir if döngüsü ile yapılabilmektedir. Çünkü tanımlı VideoCapture örneği, kamera çalışırken doğru dönen bir if döngüsü ile açılacaktır. Video yakalama ayarları kod içinde değiştirilmektedir. Herhangi bir grafik öğeyi görüntüleyen bir program hazırlanırken veya daha ayrıntılı olarak, sürekli değişen görüntüler görüntülenirken sonsuz döngü herhangi bir çizim isteğini işlemek ve yeterli zaman kazanmak için bir gecikme içermektedir. OpenCV için waitKey kodu kullanıcının belirli bir komut tuşuna basarak bu işlemi yapmasını sağlamaktadır [34].

4. SONUÇ ve ÖNERİLER

Bu tez çalışmasında, fiziksel özellikleri tanımlanan bir renk nesnesinin koordinatını (x ve y) konumlandırabilen, anlık alınan görüntü verilerinden, nesnenin pozisyonunu hesaplayıp yerini tespit edip takibini yapabilen bir robot geliştirilmiştir. Ayrıca benzeri çalışmalarda oluşan kompleks donanımsal dizaynlar yerine sade ve basit bir sistem dizaynı tercih edilmiştir.

Geliştirilen sistem bir stereo görüş sisteminden oluşmaktadır. Sistem, bir kamera, sensörler, motorlar, motor sürücüleri ve tespit edilen nesnenin 3D konumunu hesaplayarak nesneyi takip edebilmesi için yazılım uygulaması kullanılmıştır. Sistem başarılı bir şekilde tasarlanarak çalıştırılmıştır.

Bir nesne izleme sisteminin renklendirme yoluyla geliştirilmesi, bu çalışmanın ana hedefi olmuş ve bu kapsamda sistem geliştirilmiştir. Tasarlanan sistem objektifin, objenin yerleştirildiği çalışma tezgahına monte edildiği yerden 1.5m mesafede algıladığı ve objeyi yakaladığı görülmüştür. Fakat obje ile mesafe 2 metreyi aşınca lensin nesneyi doğru olarak algılamasında problemler yaşandığı gözlemlenmiştir. Nesnenin nerede olduğunu algılamak için kamera lensinin değiştirilmesi ve IR işaretleme aygıtının kullanılması, gelecekte buna bir çözüm olabilir kanaatindeyim.

Yapılan bu çalışma birden fazla görüntüyü eşleştirmek için kullanılan eşzamanlı renk aralıklarına sahip nesnelerin takibi için de geliştirilerek kullanılabilir.

Sonuç olarak geliştirilen prototip anlık görüntü işleme ve bilgi aktarımının ihtiyaç duyduğu insansız hava araçları, otonom otomobil , tarım için kullanılabilecek robotlar, askeri uygulamalar, lojistik, endüstriyel uygulamalar vb. birçok alanda kullanılabileceği gibi ihtiyaç duyulan farklı alanlar içinde geliştirilebilir.

KAYNAKLAR

[1] R. D. Inc, "History of Robotics: Timeline," [Online]. Available: https://www.robotshop.com/media/files/PDF/timeline.pdf. [Accessed 21 02 2018].

[2] «Robot nedir,» [Çevrimiçi]. Available: http://yapbenzet.kocaeli.edu.tr/robot-nedir/. [Erişildi: 10 11 2018].

[3] iRobot«ScoobaRobot,»[Çevrimiçi]. Available: https://store.irobot.com/default/scooba-floor-scrubbing/irobot-scooba-450/S450020.html?cgid=us. [Erişildi: 10 11 2018].

[4] J. Wiley, Handbook of Industrial Robotics, New York, 1999.

[5] L. Prayaga, «Introduction_to_Robotics,»*Robotics: A Project-Based Approach* , pp. 1-10.

[6] R. O. L. B. G. Quaglia, «A modular approach for a family of ground mobile robots,» *Int. J. Adv. Robot. Syst. 10 (7)*, 2013, p. 296.

[7] S. G. Tzafestas, «Mobile Robots: General Concepts,» *Introduction to Mobile Robot Control*, 2013, pp. 1-10.

[8] Siemens. [Çevrimiçi]. Available: https://www.healthcare.siemens.com/laboratory-automation/advanced-robotic-solutions. [Erişildi: 11 11 2018].

[9] G. N. N. Rezaeian, *Detecting near-duplicates in russian documents through using fingerprint algorithm Simhash*, Moskova, 2016.

[10] I. V. Fares Jalled, *Object Detection Using Image Processing*, Moskova, 2016.

[11] R. P. A. L. J. M. L. C. M. Kristan, «The visual object tracking VOT2014 challenge results,» *European Conference on Computer Vision (ECCV)*, 2014.

[12] D. M. C. R. C. S. C. A. D. A. W. M. Smeulders, *Visual tracking: An experimental survey*, 2014.

[13] K. Balaji S.R., «,A survey on moving object tracking using image processing,» *International Conference on Intelligent Systems and Control (ISCO), IEEE*, 2017.

[14] I. S. a. G. E. H. A. Krizhevsky, *ImageNet Classification with Deep Convolutional Neural Networks,* 2012.

[15] J. Kay, «The use of infrared viewing systems in electrical control equipment,» *Pulp and Paper Industry Technical Conference*, Jacksonville, 2005.

[16] E. B. Tuba Kurban, *İnersiyal Algılayıcı Tabanlı Hareket Yakalama.*

[17] N. K. A. A. A. Iason Oikonomidis, «Efficient Model-based 3D Tracking of Hand Articulations using Kinect,» [Çevrimiçi]. Available: http://www.bmva.org/bmvc/2011/proceedings/paper101/paper101.pdf. [Erişildi: 22 10 2018].

[18] S. S. A. D. a. M. C. N. Vo, «"Multi-Target Tracking,",» *Wiley Encyclopedia of Electrical and Electronics Engineering*, 2015.

[19] Z. Z. A. D. a. A. v. d. H. C. Shen, «A Survey of Appearance Models in Visual Object Tracking,» *Transactions on Intelligent Systems and Technology*, 2013, pp. 1-42.

[20] M. C, «Background Subtraction Using Running Gaussian Average: a Color Channel Comparison,In:,» *Seminar aus Bildverarbeitung und Mustererkennung, ,* 2014.

[21] K. Balaji S.R., «A survey on moving object tracking using image processing,,» *International Conference on Intelligent Systems and Control (ISCO), IEEE,* 2017.

[22] A. A.R, «Detection and Classification of a Moving Object in a Video stream,» *Proc. of the Intl. Conf. on Advances in Computing and Information Technology-ACIT,* 2014.

[23] W. Y. S. W. Li G., «Real-Time Moving Object Detection for Video Monitoring Systems,,» *Second International Symposium on, IEEE,* 2008.

[24] P. Alizadeh, *Object Distance Measurement Using a Single Camera for Robotic Applications,* Sudbury, Ontario, Canada : Laurentian University , 2015.

[25] «https://docs.opencv.org/3.4/da/d5c/tutorial_canny_detector.html,» opencv. [Çevrimiçi]. [Erişildi: 22 10 2018].

[26] G. M. Omosekeji, *Industrial Vision Robot with Raspberry Pi using Pixy Cam*, 2018.

[27] C.-M. Ivask, *Raspberry Pi based System for Visual Object Detection and Tracking*, Talinn: TALLINN UNIVERSITY OF TECHNOLOGY, 2015.

[28] opencv, «https://docs.opencv.org/2.4/modules/core/doc/operations_on_arrays.html#inrange, »opencv,[Çevrimiçi].Available: https://docs.opencv.org/2.4/modules/core/doc/operations_on_arrays.html#inrange. [Erişildi: 23 11 2018].

[29] «OpenCV API Reference : Structural Analysis and Shape Descriptors,» opencv, [Çevrimiçi].Available: https://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html. [Erişildi: 23 11 2018].

[30] [Çevrimiçi]. Available: http://www.derstagram.com/dc-motor-calisma-prensibi/. [Erişildi: 24 10 2018].

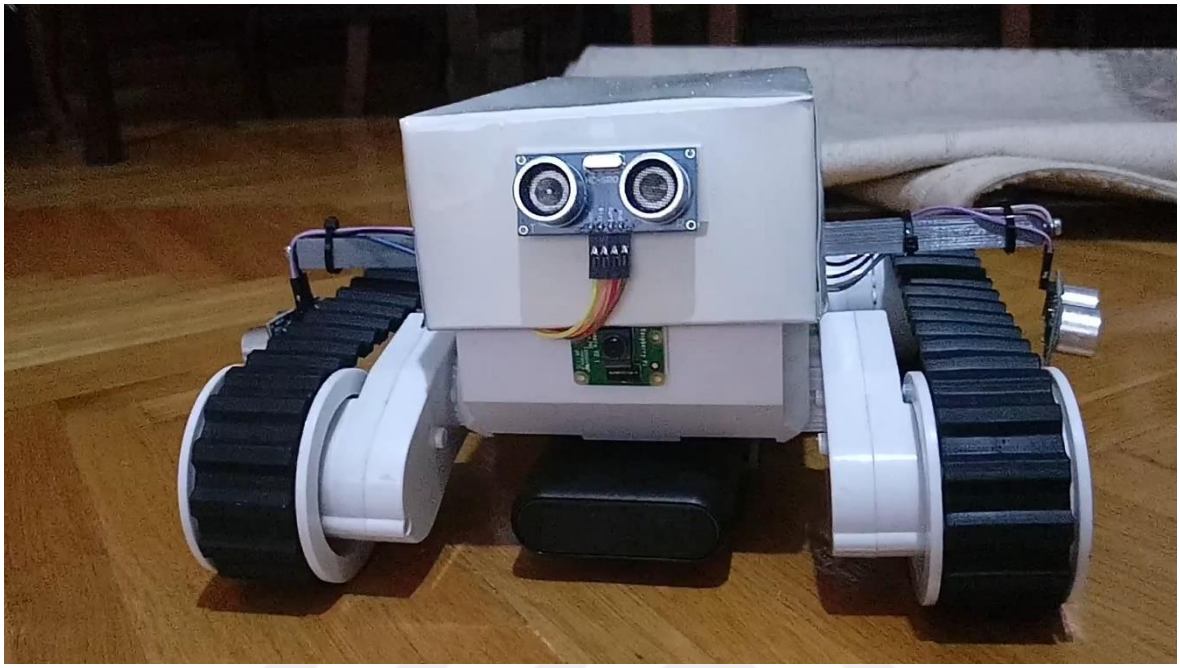
[31] «h köprüsü nedir,» [Çevrimiçi]. Available: https://www.elektrikport.com/universite/h-koprusu-nedir/17187#ad-image-0. [Erişildi: 10 10 2018].

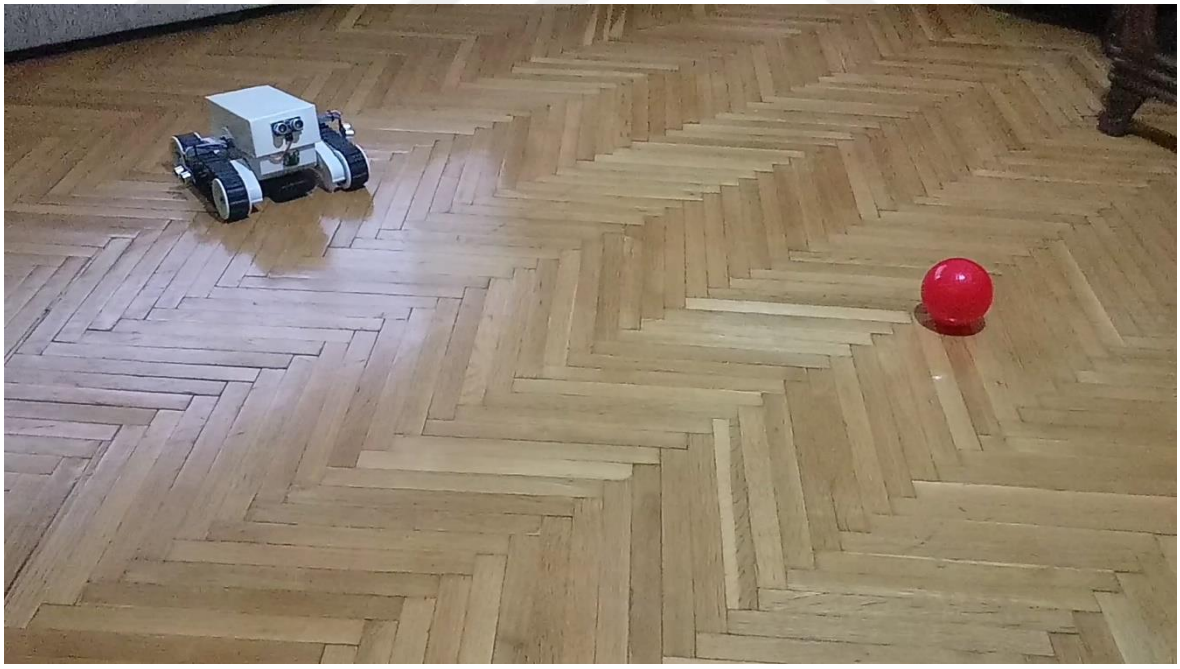
[32] «Python Programlama Dili Nedir,» [Çevrimiçi]. Available: https://www.python.tc/python-nedir/. [Erişildi: 28 10 2018].

[33] Python, «Comparisons,» Python, [Çevrimiçi]. Available: https://www.python.org/doc/essays/comparisons/. [Erişildi: 1 11 2018].

[34] «OpenCV API Reference : User Interface,» [Çevrimiçi]. Available: https://docs.opencv.org/3.4.4/d7/dfc/group__highgui.html#ga5628525ad33f52eab17feebcfba38bd7. [Erişildi: 3 10 2018].

EKLER





ÖZGEÇMİŞ

Adı ve Soyadı: Taner YILMAZ	
KİŞİSEL BİLGİLER	
ADRES : TELEFON : EPOSTA : DOĞUM YERİ/TARİHİ: MEDENİ HALİ:	Rüstempaşa Mahallesi Şeyh Şamil Bulvarı No:9/1 Merkez 23100 – Elazığ +90 (5366246600) tnrylmz@live.com Elazığ / 01 07 1991 Bekar
EĞİTİM BİLGİLERİ	
LİSANSÜSTÜ LİSANS : LİSE :	2019, Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Mekatronik Mühendisliği, Elektronik Sistemler, Elazığ 2015, Bahçeşehir Üniversitesi, Mühendislik Fakültesi, Mekatronik Mühendisliği, İstanbul 2008, Elazığ Merkez Anadolu Lisesi, Elazığ
İŞ DENEYİMİ	
2013 2015	Keban Hes Barajı, ElazığKurum/Kuruluş Adı, Şehir Stajyer (1 Ay) Şeker Fabrikası, Elazığ Stajyer (1 Ay)
BİLDİĞİ YABANCI DİLLER	
	İngilizce
ARAŞTIRMA DENEYİMİ	
Comsol Multiphysics, Solidworks Python, C, Ares, Matlab, Simulink Raspberry Pi, Ardunio uygulamaları	
İLGİLENDİĞİ DİĞER BİLİMLER	
Fitness, Vücut Geliştirme, Basketbol, Müzik, Bilim kurgu kitap ve filmler, Astroloji, Mitoloji	