KOCAELİ ÜNİVERSİTESİ*MÜHENDİSLİK FAKÜLTESİ

ÖRÜMCEK ROBOT İLE GÖRÜNTÜ İŞLEME VE YAPAY ZEKA UYGULAMALARI

LİSANS TEZİ

Berat Eren TERZİOĞLU

Bölümü: Elektronik ve Haberleşme Mühendisliği

Danışman: Doç. Dr. Serhat YILMAZ

KOCAELİ, 2021

ÖNSÖZ ve TEŞEKKÜR

İnsansız araçlar günümüzde çeşitli alanlarda yaygın bir şekilde kullanılmaktadır. İnsanların günlük işlerinde, endüstri uygulamalarında ve ekstrem koşullarda çalışma kabiliyetleri sebebiyle yaygın olarak kullanılır.

Bu tez çalışmasında otonom örümcek robot ile görüntü işleme kullanılarak nesne takibi uygulaması gerçekleştirilmiştir. Yapılan çalışmanın aynı alanda çalışacak araştırmacılara faydalı olmasını dilerim.

Bu projenin gerçekleştirilmesi sürecinde destek ve öğretilerinden ötürü değerli hocam Doç. Dr. Serhat Yılmaz'a teşekkür ederim.

Mayıs 2021, KOCAELİ Berat Eren TERZİOĞLU

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLO DİZİNİ

SİMGELER VE KISALTMALAR

GPIO : Genel Amaçlı Giriş-Çıkış

RPi : Raspberry Pi

VNC : Sanal Ağ Bağlantısı

RGB : Kırmızı-Yeşil-Mavi Renk Uzayı

HSV : Ton-Yoğunluk-Değer Uzayı

BGR : Mavi-Yeşil-Kırmızı Renk Uzayı

PWM : Darbe-Genlik Modülasyonu

ÖR : Örümcek Robot

SP : Spider Robot

FLC : Fuzzy Logic Controller

UART : Haberleşme Protokolü

AU : Arduino Uno

AS : Arduino Shield

SBC : Single Board Computer

ÖRÜMCEK ROBOT İLE GÖRÜNTÜ İŞLEME VE YAPAY ZEKA UYGULAMALARI

Berat Eren TERZİOĞLU

Anahtar Kelimeler: Görüntü işleme, OpenCV, Raspberry Pi, Arduino, PWM, Örümcek Robot, Servo motor, UART.

Özet: Günümüzde endüstri, askeri, ev aletleri gibi çok çeşitli alanlarda kullanılan insansız araçlar gelecekte daha da yaygın olarak kullanılacaktır. Bu proje kapsamında Örümcek Robot ile Görüntü İşleme ve Nesne Takibi Uygulamaları projesinin yapılması amaçlanmıştır. Amaç kapsamında robotun otonom hareketi, nesne tanıma ve nesne takibi uygulamalarıyla gerçekleştirilmiştir.

Görüntü işleme uygulaması, Raspberry Pi 3 B+ kartıyla yapılmıştır. Görüntü işleme algoritması Python dilinde OpenCV kütüphanesi kullanılarak yapılmıştır. Motor kontrolleri Arduino geliştirme kartı ile yapılmıştır. Hareket kontrolü, Raspberry Pi kartından gelen nesne tespit verilerinin Arduino kartına aktarılmasıyla yapılmıştır. İki kartın haberleşmesi için UART haberleşme protokolü kullanılmıştır. Robot iskeleti, 4 kol üzerinde bulunan servo motorların belli bir sırayla çalışmasıyla ileri veya geri PWM sinyalleriyle birlikte hareket eder. PWM sinyalleri Raspberry Pi kartından Arduino kartına aktarılacak verilerle oluşturulur. Nesneyi tespit eden program servo motorları konumlandırarak harekete başlar. Takip edilecek nesnenin, kamera ekranındaki konumuna göre motor hareketleri şekillendirilir. Proje kapsamında kırmızı bir nesnenin tespit edilmesi ve takip edilmesi amaçlanmıştır. Bu kapsamda görüntü işleme algoritması kırmızı renkte cisimlerin tespiti ve takibini yapacaktır.

IMAGE PROCESSING AND ARTIFICIAL INTELLIGENCE APPLICATIONS WITH SPIDER ROBOT

Berat Eren TERZİOĞLU

Keywords: Image processing, OpenCv, Raspberry Pi, Arduino, PWM, Spider Robot, Servo-motor, UART.

Abstract: Within the scope of this project, it is aimed to perform Image Processing and Object Tracking Applications with Spider Robot.Within the scope of the purpose, the autonomous movement of the robot was made with object recognition and object tracking applications. Image processing application was made with Raspberry Pi 3 B + board. The image processing algorithm was made using the OpenCV library in Python language. Motor controls are made with Arduino development board. The motion control is done by transferring the object detection data coming from Raspberry Pi board to Arduino board. UART communication protocol is used for the communication of the two cards. The robot skeleton moves forward or backward with PWM signals, with the servo motors on the 4 arms operating in a certain order. PWM signals are created with the data to be transferred from the Raspberry Pi board to the Arduino board.The program that detects the object starts the motion by positioning the servo motors. Motor movements are shaped according to the position of the object to be tracked on the camera screen. Within the scope of the project, it was aimed to detect and track a red object.

1. GİRİŞ

İnsansız araçlar, günümüzde endüstride ve günlük yaşamımızda yaygın bir şekilde kullanılmaktadır. Endüstride hızlı ve kesin sonuç almak için kullanılan insansız araçlar günlük yaşantımızı daha kolay bir hale getirmek için geliştirilmiştir. İnsan hayatını kolaylaştırmakla kalmayıp insanoğlu için çok elverişsiz ortamlarda bile yüksek verimle çalışmalarından ötürü insanlık adına pek çok görevi başarıyla yürütmüşlerdir.

İnsansız araçların hava, su ve karada kullanılacak şekilde üretim çeşitleri mevcuttur. Bu projede karada çalışan Örümcek Robot üzerinde çalışılmıştır. Bu araçların kontrolleri uzaktan manuel olarak veya otonom olarak yapılır. Uzaktan manuel olarak kontrol edilen araçlar operatör ile haberleşmeli ve sensörlerinden (kamera, sıcaklık sensörü, Ultrasonik sensör vb.) elde ettiği verileri operatör ile anlık olarak paylaşmalıdır. Otonom araçlarda ise operatöre ihtiyaç duyulmadan topladığı verileri kendisi işleyerek yönelim ve ölçüm yapabilirler.

Bu projede otonom çalışan 4 kollu Örümcek Robot kullanılmıştır. Görüntü işleme uygulamaları için, boyutu ve görüntü işleme uygulamalarında sağladığı imkanlar açısından Raspberry Pi 3 B+ mini bilgisayar kullanımı tercih edilmiştir.

Görüntü işleme uygulamaları için OpenCV kütüphanesi kullanılmıştır. OpenCV (Open Source Computer Vision Library) gerçek zamanlı bilgisayar görüsü uygulamalarında kullanılan açık kaynaklı kütüphanedir. İlk olarak Intel tarafından geliştirilmiş olup, halen daha geliştirilme süreci devam etmektedir. Bu kütüphane çoklu platform ve BSD lisansı altında açık kaynaklı bir yazılımdır.[1]

Raspberry Pi kurulumu için gerekli adımlar ve görüntü işleme için kullanacağımız açık kaynak kütüphane olan OpenCV kütüphanesinin de kurulumu için yapılması gereken adımlara detaylıca yer verilmiştir.

Projede amaç Raspberry Pi Camera'dan alınan görüntü verilerini işleyerek robotu hareket ettiren motorları harekete geçirmektir. Motorları sürmek için Arduino Uno geliştirme kartı kullanılmıştır. Raspberry Pi'dan gelen konum verileri Arduino'ya gönderilir. Arduino IDE'sinde yazılan kodlar Raspberry Pi'dan alınan verilere göre motorlara PWM sinyalleri gönderir. Gönderilen PWM sinyalleriyle birlikte nesne takibi sağlanır.

2. DONANIM TASARIMI

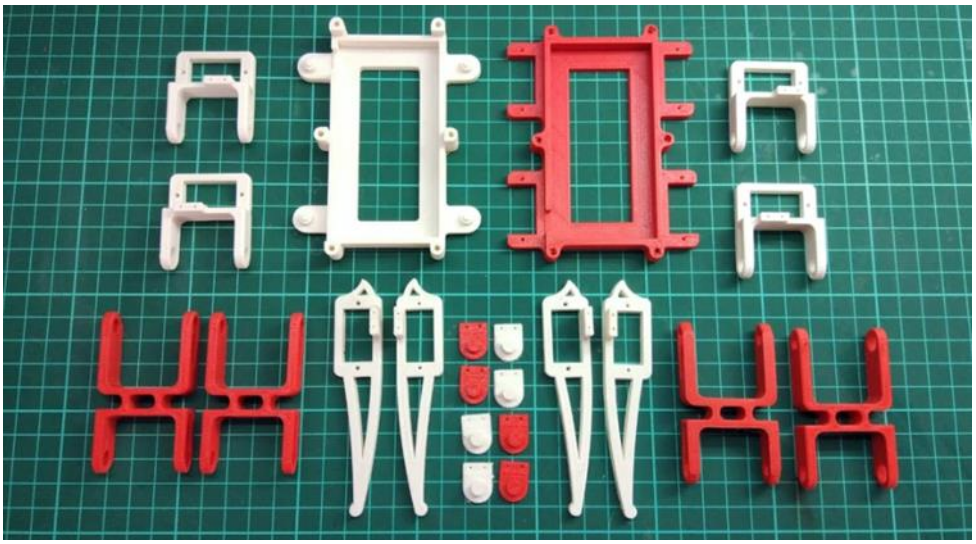
Otonom robotlar, bulundukları ortama göre taşıdıkları yazılım sayesinde yapacaklarına kendi karar verebilme yeteneği kazandırılmış veya tanımlanmış bir görevi sürekli olarak devam ettirebilen araçlardır. Günümüz endüstrisinde kullanılan robotların çoğu tanımlanan görevi sürekli olarak yerine getirecek şekilde programlanmıştır. İnsan gücüne göre daha az maliyetli ve riskli işleri daha düşük riskle gerçekleştirmeleri bakımından yaygın olarak kullanılmaktadırlar.

Bu proje kapsamında Örümcek Robot (SP) kullanılmıştır. Bu aracın güvenli ve verimli olarak çalışabilmesi için aracın donanımsal yapısının doğru tasarlanması gerekir.

Verimli bir çalışma elde edilebilmesi için doğru donanım parçaları seçilmeli ve bu parçaların kullanımına uygun bir ortamda deneylerin yapılması gerekmektedir.

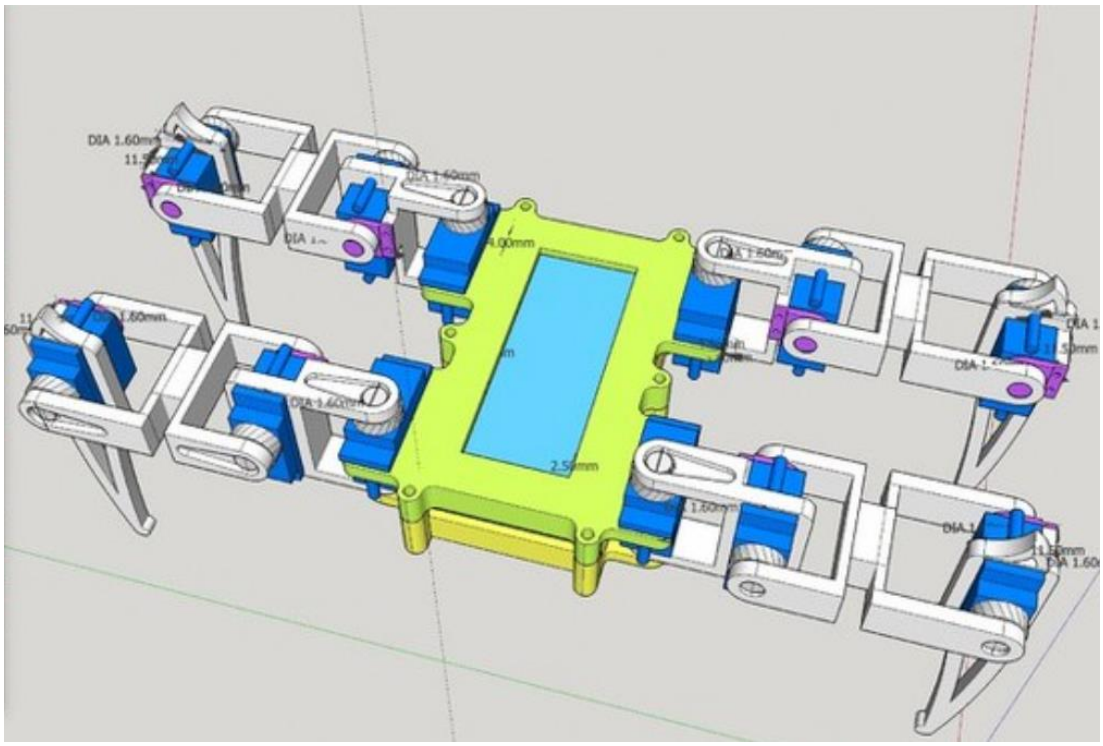
2.1 Gövde Tasarımı

Kullanılan aracın parçaları 3 boyutlu yazıcıdan çıktı alınan gövde, eklem ve bacak olacak şekilde parçalardan oluşmaktadır. Araç 12 adet servo motor ile hareket edecek şekilde tasarlanmıştır. Gövde tasarımı üzerinde servo motorların oturacağı şekilde boşluklar bulunur. Gövde içinde ise Li-on pillerin yerleştirilebileceği bir boşluk bulunmaktadır. İskelet parçaların demonte hali Şekil 2.1’ de verilmiştir.



Şekil 2.1: Örümcek Robot İskeleti [18]

Robot iskeletinin parçaları ister sipariş usulü istenirse de SkecthUp modelleme programında tasarlanarak temin edilebilir. Şekil 2.2'de modelleme programı üzerinde parça boyutlarını gösteren tasarım şeması bulunmaktadır. Uygulanan projede parçalar hazır olarak sipariş edilmiştir.



Şekil 2.2: Robot iskeleti modelleme [19]

2.1.1 Servo Motorlar

Örümcek Robot 12 adet motor içermektedir. Şekil 2.2.1'de de görüldüğü gibi motorların 4 tanesi ana gövdeye monte edilir. 4 tanesi eklem kısmına ve kalan 4 motor ise bacaklara monte edilir. Motorların hareketi saat yönünde ve saat yönünün tersi yönünde olacak şekilde tasarlanmıştır.

Servo motorların yapısında motorların hareket etmesini sağlayan birer DC motor bulunmaktadır. Bu DC motorun dış tarafında bir dişli mekanizması, potansiyometre ve bir motor sürücü devresi bulunmaktadır. Potansiyometre, motor milinin ne kadarlık bir açı ile dönüş yaptığını ölçmektedir. Servo motorun içerisinde bulunan DC motor hareket ettikçe potansiyometre döner ve kontrol devresi motorun bulunduğu pozisyon ile istenilen pozisyonu kıyaslar ve bu şekilde motor sürme işlemini yapar. Servo motorlar, diğer motorlar gibi harici bir motor sürücü devresine gereksinim duymadan çalışmaktadırlar. Çalışma açıları en fazla

180˚ olacak şekildedir fakat 360˚ çalışma aralığına sahip özel amaçlı Servo motorlar da vardır. Servo motorlar genellikle 4.8-6V gerilim ile çalışmaktadırlar. 7.4V ve daha yüksek gerilimle çalışan servo motorlar da üretilmektedir.[2]

2.1.2 Arduino Uno

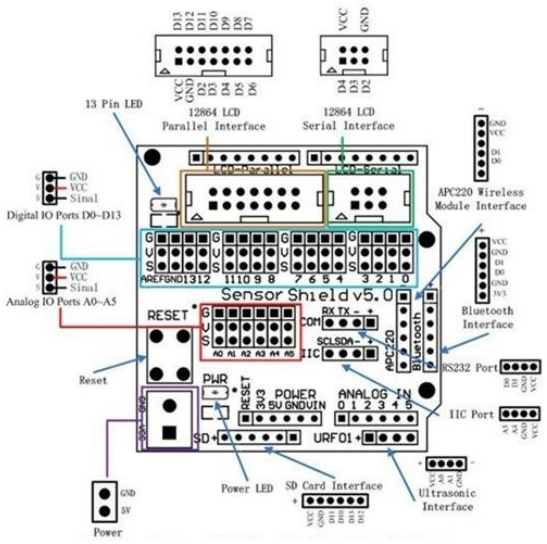
Şekil 2.3: Arduino Uno [20]

Bu projede motorları sürmek için Arduino Uno R3 klonu kullanılmıştır. R3 klonu üzerlerinde kullanılan elektronik devre elemanları ve tüm fonksiyonelliği orijinal kart ile aynıdır. Ancak orijinal Arduino Uno kartında DIP kılıfında bulunan mikroişlemci, bu üründe ise SMD kılıfında yer alır. Kartı beslemek için iki adet seri bağlı 3.7V Li-on pil kullanılmıştır.

2.1.3 Arduino Sensor Shield 5.0

Şekil 2.4: Arduino Sensor Shield [21]

Arduino Sensor Shield, Arduino Uno R3 kartı ile birlikte kullanılır. Kart üzerinde birçok giriş çıkış biriminin karta bağlanmasını sağlar. Kart üzerinde Arduino'nun tüm giriş çıkış pinleri DATA, VCC ve GND sırası ile 3 pin haline getirilerek kart üzerinde dağıtılmıştır. Servo Motor pinleri aşağıda verilen Arduino Sensor Shield Şekil 2.2.3.1'te verilen pin tablosunda Digital IO Portlarına Örümcek Robot kollarının kodlamadaki sırasına göre değişkenlik göstererek 2 ve 13 numaralı girişler arasına yerleştirilir.



Şekil 2.5: Arduino Sensor Shield Pin-out [22]

2.2 Örümcek Robot'un Çalışma Prensibi

Örümcek Robot dört adet bacak ve bacakların bağlandığı bir adet gövdeden oluşmaktadır. Her bir bacak üç adet parçadan oluşmaktadır. Bacağı oluşturan parçalar birbirlerine servo motorlar ile ve her bir bacak gövdeye bir adet servo motor ile bağlanmaktadır. Servo motorlar örümcek için eklem görevi görmekte ve Örümcek Robot'un hareketini sağlamaktadır.

Her bir bacak için Örümcek Robot gövdesine bağlı olan servo motora X, sonraki servo motora Y ve en uç kısımda bulunan servo motora Z isimleri verilmiştir. Örümcek Robota tepeden bakıldığı zaman sağ tarafta ve geride bulunan bacağa 1. Bacak, sağ tarafta ve ileride bulunan bacağa 2. Bacak olarak isimlendirilmiştir. Sol tarafta ve geride bulunan bacağa 3. Bacak ve sol tarafta ve ileride bulunan bacağa 4. Bacak olarak isimlendirilmiştir. Robot sensörün sol yanında bulunan 2. bacağını kaldırarak ileri hareket ettirir daha sonra bu hareketi diğer üç bacak tekrar eder. Bu hareket dizini tamamlandığında robot bir adım ilerlemiş olacaktır.

2.2.1 Örümcek Robot'un İleri Yönlü Hareketi

Bacakta bulunan Y isimli servo motorun açısı belirli bir miktar azaltılır, Z isimli servo morun açısı belirli bir miktar azaltılır ve Örümcek Robot'un bacağının yer ile olan teması kesilir. X isimli servo motorun açısı belirli bir miktar arttırılarak Örümcek Robot'un bacağı ileriye doğru çevrilir. Y ve Z isimli servo motorların açısı başlangıç konumuna getirilerek bacağın yer ile olan teması sağlanır. Böylece Örümcek Robot'un bir bacağı için bir adım tamamlanmış olur.

Örümcek Robot'un ileriye doğru hareket etmesi için bacakların koordineli bir şekilde adım atmaları gerekir. İleri yönlü hareket için bacakların hareket sırası; 2 numaralı bacak, 1 numaralı bacak, 3 numaralı bacak ve 4 numaralı bacak şeklindedir.

2.2.2 Örümcek Robot'un Geri Yönlü Hareketi

Bacakta bulunan Y isimli servo motorun açısı belirli bir miktar azaltılır, Z isimli servo morun açısı belirli bir miktar azaltılır ve Örümcek Robot'un bacağının yer ile olan teması kesilir. X isimli servo motorun açısı belirli bir miktar arttırılarak Örümcek Robot'un bacağı geriye doğru hareket ettirilir. Y ve Z isimli servo motorların açısı başlangıç konumuna getirilerek bacağın yer ile olan teması sağlanır. Böylece Örümcek Robot'un bir bacağı için bir adım tamamlanmış olur.

Örümcek Robot'un geriye doğru hareket etmesi için bacakların koordineli bir şekilde adım atmaları gerekir. Geri yönlü hareket için bacakların hareket sırası; 4 numaralı bacak, 3 numaralı bacak, 1 numaralı bacak ve 2 numaralı bacak şeklindedir.

2.2.3 Örümcek Robot'un Sağ Yönlü Hareketi

Bacakta bulunan Y isimli servo motorun açısı belirli bir miktar azaltılır, Z isimli servo motorun açısı belirli bir miktar azaltılır ve böylece bacağın yer ile olan bağlantısı kesilir. X isimli servo motorunun saat yönünde dönecek şekilde açısı arttırılır. Y ve Z isimli servo motorları başlangıç açılarına getirilir. Bu sayede bir bacağın sağa dönüşü tamamlanmış olur.

Örümcek Robot'un tam sağa doğru dönebilmesi için bu işlemler her bacak için üçer kez tekrarlanır. Sırasıyla hareket ettirilecek bacaklar; 2 numaralı bacak, 4 numaralı bacak, 3 numaralı bacak ve 1 numaralı bacaktır.

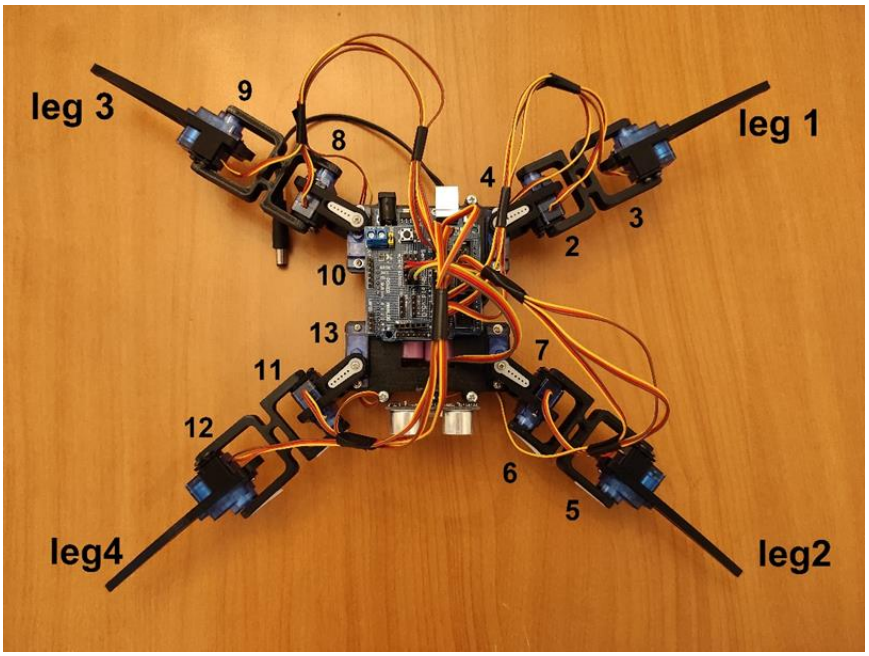
2.2.4 Örümcek Robot'un Sol Yönlü Hareketi

Bacakta bulunan Y isimli servo motorun açısı belirli bir miktar azaltılır, Z isimli servo motorun açısı belirli bir miktar azaltılır ve böylece bacağın yer ile olan bağlantısı kesilir. X isimli servo motorunun açısı saat yönünün tersine doğru azaltılır. Y ve Z isimli servo motorları başlangıç açılarına getirilir. Bu sayede bir bacağın sola dönüşü tamamlanmış olur.

Örümcek Robot'un tam sola doğru dönebilmesi için bu işlemler her bacak için üçer kez tekrarlanır. Sırasıyla hareket ettirilecek bacaklar; 4 numaralı bacak, 2 numaralı bacak, 1 numaralı bacak ve 3 numaralı bacaklardır.

2.3 Servo Motorların Arduino Sensor Shield Pin Girişleri

Şekil 6'da verilen pin bağlantıları Arduino Sensor Shield üzerinde aynı görseldeki gibi yapılmalıdır. Farklı pin girişleri yapıldığında senkronizasyon problemleri yaşanabilmektedir. Arduino kodlamaları da Şekil 2.4'te verilen şemaya göre yapılmıştır. İleri yön olarak Bacak 2 ve 4'ün bulunduğu taraf referans alınmıştır.



Şekil 2.6: Servo Motor Pin Girişleri

3. RASPBERRY Pİ KURULUMU

Raspberry Pi, Raspberry Pi Vakfı tarafından üretilen tek kart bilgisayarlardır. Bir bilgisayarda bulunan işlemci, RAM, giriş ve çıkış portları gibi tüm donanım birimleri tek bir devre kartı üzerinde birleştirilmiştir. Ufak tasarımı ve portatif yapısı sayesinde bu bilgisayarları robotik projelerde, akıllı ev sistemlerinde, gömülü sistemlerde kullanılır. Tercihe göre klavye, fare ve monitör gibi çevre donanımlarını ekleyerek masaüstü bilgisayar olarak da kullanılabilir. Raspberry Pi gibi tek kart bilgisayarlar, Arduino gibi mikro kontrol kartlarının yeterli gelmediği ve aynı anda birden fazla işlemin yapılması gereken durumlarda tercih edilirler. Raspberry Pi, Linux OS ve Windows 10 IoT Core gibi özel olarak geliştirilen bir işletim sistemini çalıştırabilmektedir.[3][4]

Proje kapsamında görüntü işleme uygulamalarını gerçekleştirmek için Raspberry Pi 3 Model B+ kullanılmıştır.

3.1 Raspberry Pi 3 B+ Teknik Özellikleri

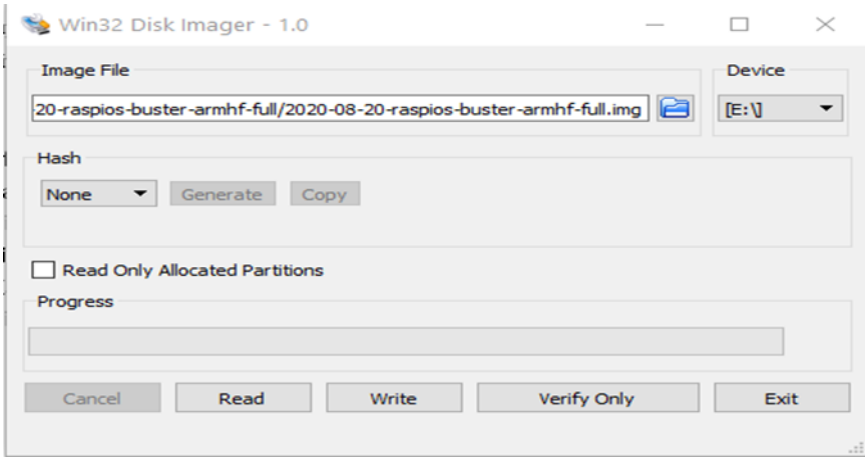
Şekil 3.1: Raspberry Pi 3 Model B+ [23]

Tablo 3.1: Raspberry Teknik Özellikler

Anakart	Broadcom BCM2837
CPU	1.2 GHz 64-bit 4 çekirdekli ARM Cortex-A53
GPU	Broadcom VideoCore IV @ 250 MHz
RAM	1 GB (Paylaşımlı)
USB 2.0 Portları	4
Video Giriş	15-pin MIPI kamera arayüzü (CSI) konnektörü
Video Çıkış	HDMI, 3.5mm TRS konnektörü
Ses Giriş	I^2C
Ses Çıkış	HDMI ile dijital
On-board depolama	MicroSDHC slotu
On-board ağ	10/10 Mbit/s Ethernet, 802.11n WiFi, Bluetooth 4.1
Çevre birimleri	17* GPIO
Güç Derecesi	800 mA
Güç Kaynağı	5V MicroUSB ya da GPIO başlığı
Boyut	85.60 mm x 56.5 mm

3.2 Raspbian İşletim Sisteminin Kurulması

Raspberry Pi üzerinde herhangi bir işletim sistemi kurulu olmadan satılmaktadır. Raspberry Pi kartının kullanılabilmesi için öncelikle bir SD karta veya USB bellek üzerine işletim sistemi kurulması gereklidir. İşletim sisteminin kurulması için Raspberry Pi internet sitesi üzerinden Raspbian işletim sistemi indirilmelidir. Raspbian işletim sistemi indirildikten sonra bir Disk Imager programı ile SD kart veya USB bellek içine yazdırılmalıdır. Bu yazdırma işlemi bittikten sonra RaspbianOS sistemi kullanıma hazır olacaktır. Kurulum tamamlandıktan sonra ise projenin yapımında kullanılacak şekilde ayarlanmalar yapılmalı, gerekli programların indirilmesi ve kurulumu yapılmalıdır.[5]



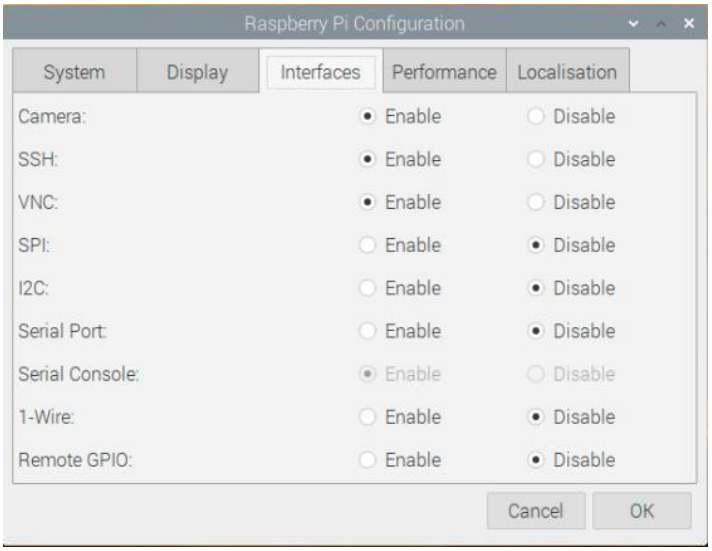
Şekil 3.2: RaspbianOS kurulum

3.2.1 Sistem Ayarlarının Yapılması

Projeye başlamadan önce Raspberry Pi üzerinde birkaç ayar yapılması gerekir. Proje kapsamında bu ayarların yapılması gerekir. Yapılmağı takdirde çeşitli sorunlarla karşılaşılabilmektedir.

3.2.1.1 Arayüz Ayarları

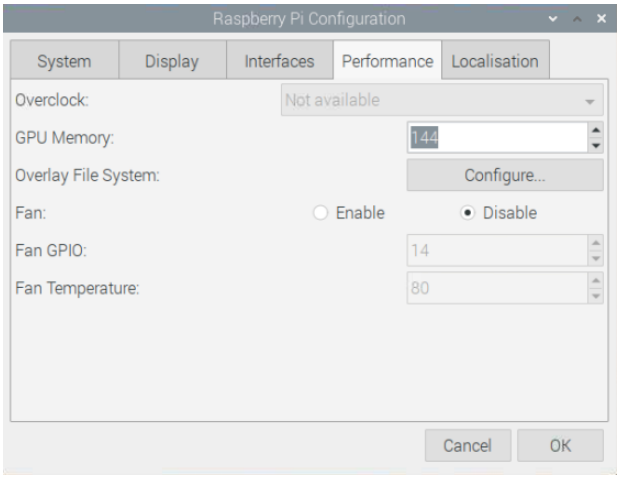
Arayüz ayarları düzenlemesi Kamera ve çeşitli fonksiyonların aktif hale getirilmesini içerir. Sistemin varsayılan ayarlarında çoğu fonksiyon "Disabled" olarak ayarlanmıştır. Bu fonksiyonların Şekilde gösterilen gibi "Enabled" olarak değiştirmemiz gerekir.



Şekil 3.3: Raspberry Configuration

3.2.1.2 Performans Artırma

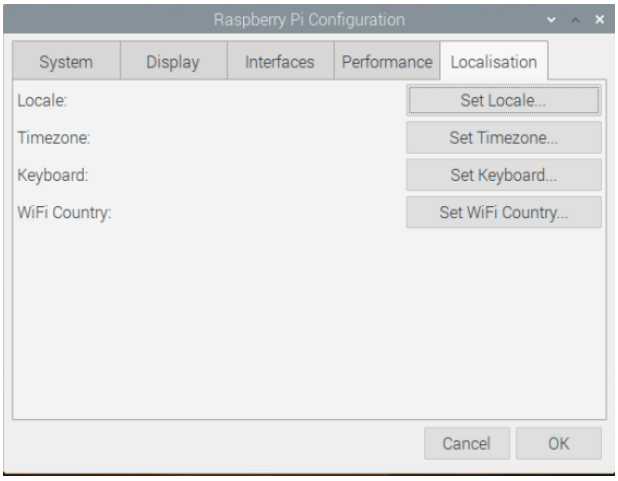
Bu ayarın yapılması Raspberry Pi'nin çalışma performansını artırmak amacıyla yapılmaktadır. Zorunlu değildir ancak hızın artması görüntü işleme uygulaması için daha uygun olacaktır. Ancak performans artırma ısınma sorununu beraberinde getirecektir. Kartın çok uzun süreler yüksek performansta çalıştırılması cihaza zarar verebilir. Isınma sorunu ise harici fan ile giderilebilir.



Şekil 3.4: Performans Ayarları

3.2.1.3 Bölgesel Ayarlar

Bu kısımda klavye ayarları, saat dilimi ayarları, Wi-Fi bağlantısı, bölge seçimi gibi ayarlara yer verilir.



Şekil 3.5: Bölgesel Ayarlar

3.3 OpenCV Kütüphanesi

OpenCV (Open Source Computer Vision) açık kaynak kodlu görüntü işleme kütüphanesidir. 1999 yılında Intel şirketi tarafından geliştirilmeye başlanmıştır. Gelişim süreci halen devam etmektedir. İlk olarak C programlama dili ile geliştirilmeye başlanmıştır. Hali hazırda bir çok programlama dilinde de kullanımına devam edilmektedir.

OpenCV kütüphanesi bünyesinde görüntü işleme ve makine öğrenmesi uygulamalarına yönelik 2500 civarında algoritma bulunur. Bu algoritmalar ile çok çeşitli yazılımlar üretilmektedir. Yüz tanıma, nesneleri tanıma, nesne sınıflandırma, görüntü karşılaştırma, görüntü iyileştirme gibi pek çok işlemi yapabilme imkânı sağlar.

Bu proje kapsamında görüntü işleme uygulamaları OpenCV kütüphanesi kullanılarak yapılmıştır. Açık kaynak kodlu olması ve bu alanda bulunan en geniş kütüphanelerden biri olması tercih sebeplerindendir.

3.3.1 OpenCV Kurulumu

Projede görüntü işleme uygulamalarının verimli bir şekilde uygulanabilmesi için öncelikle OpenCV kütüphanesi Raspberry Pi sistemine indirilmelidir. Bu işlem vakit alan bir işlem olduğundan internet kesintisine uğramamak adına Ethernet kablosuyla internete bağlanmanız tavsiye edilir.

İlk olarak Raspberry Pi Configuration ekranı açılmalı ve sırasıyla aşağıda verilen kodlar configuration ekranına yazılmalıdır. Her satırdan sonra enter tuşu ile devam edilmelidir. EK D kısmında Raspberry OpenCV kurulum kodlarına yer verilmiştir.

4. RASPBERRY Pİ – ARDUİNO HABERLEŞMESİ

4.1 Giriş

Proje kapsamında servo motorların sürülmesi işlemini Arduino Üzerinden yapacağız. Bunun için Raspberry kart içine kurmamız gereken Arduino dosyaları bulunmaktadır.

İki kartın haberleşmesi için Seri Haberleşme kullanılmıştır. Seri Haberleşme basitçe veri aktarmanın yollarından biridir. Veriler, aynı anda birçok bitin gönderildiği paralel haberleşmenin aksine, her seferinde 1 bit olmak üzere sırayla gönderilecektir.

Bu haberleşme "UART" protokolü kullanarak gerçekleştirilir. Temel olarak, iki kart arasında iletişim kurmamıza olanak sağlayan Seri İletişime dayalı bir asenkron çoklu ana protokoldür. [17]

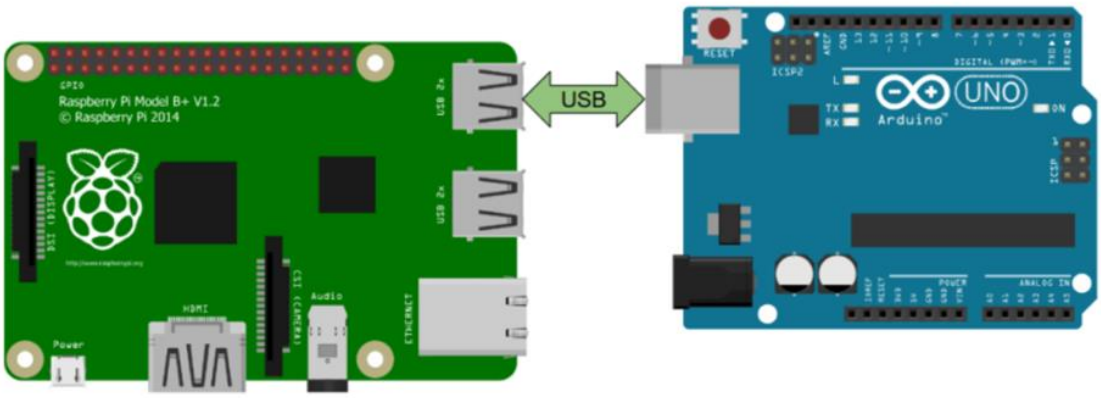
4.2 Seri Haberleşme Donanımsal Kurulum

Donanım kurulumu için iki yol bulunmaktadır:

- Serial via USB
- Serial via GPIO's

Bu projede "Serial via USB" kullanılmıştır.

Raspberry Pi tarafında USB konektörü ile kart üzerinde bulunan dört girişten birine yerleştirilir.



Şekil 4.1: Raspberry Pi- Arduino Uno Bağlantı [24]

4.3 Raspberry Pi Arduino IDE Kurulumu

Raspberry Pi terminal ekranı açılıp aşağıdaki kodlar yazılmalıdır:

- sudo apt-get install arduino
- sudo apt-get install python-serial mercurial
- python3 -m pip install pyserial
- sudo apt-get install python3-pip

Kodlar yazıldığında Raspberry üzerinde Arduino için gerekli yazılım kurulmuş olacaktır.

4.4 Raspberry Pi'den Arduino'ya Bilgi Gönderme

1) Raspberry Pi üzerinde kurmuş olduğumuz Arduino IDE'sinde aşağıdaki kodları derleyip çalıştırmamız gerekir.

```
1.  void setup() {
2.    Serial.begin(9600);
3.  }
4.
5.  void loop() {
6.    if (Serial.available() > 0) {
7.      String data = Serial.readStringUntil('\n');
8.      Serial.print("You sent me: ");
9.      Serial.println(data);
10.   }
11. }
```

Şekil 4.2: Arduino Haberleşme Kodları

2) Raspberry Pi Python IDE'sinde aşağıdaki kodların derlenip çalıştırılması gerekir.

```
1.  #!/usr/bin/env python3
2.  import serial
3.  import time
4.
5.  if __name__ == '__main__':
6.      ser = serial.Serial('/dev/ttyACM0', 9600, timeout=1)
7.      ser.flush()
8.
9.      while True:
10.         ser.write(b"Hello from Raspberry Pi!\n")
11.         line = ser.readline().decode('utf-8').rstrip()
12.         print(line)
13.         time.sleep(1)
```

Şekil 4.3: Raspberry Haberleşme Kodları

Kodun 6. Satırında 'ttyAM0' kısmı yerine 'ttyUSB0' yazılmalıdır.

Her iki kodu da kendi IDE'lerinde çalıştırdığımızda çıktı ekranında "Hello from Raspberry Pi!" yazısını gözlemleyeceğiz.

5. GÖRÜNTÜ İŞLEME UYGULAMALARI

5.1 Giriş

Yürütülecek proje kapsamında Örümcek Robot'un fiziksel özellikleri önceden tanıtılmış olan bir nesnenin takibini yapması amaçlanmıştır. Bu amaç doğrultusunda OpenCV kütüphanesi ile Python programlama dilinde görüntü işleme uygulamaları yürütülmüştür.

Görüntü işleme, kameradan alınan veya fotoğraf üzerinden alınan görüntüyü dijital model haline getirmek ve bazı işlemleri gerçekleştirmek için geliştirilmiştir. Görüntü spesifik görüntü elde etmek veya ondan birtakım yararlı bilgiler çıkarmak için kullanılan yöntemdir.

Görüntü işleme temel olarak üç adımda gerçekleştirilir.

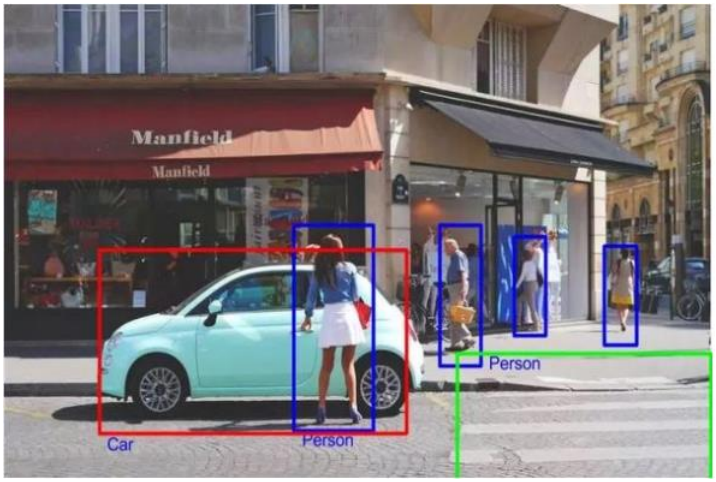
- Görüntünün optik tarayıcı ile veya dijital fotoğraflarla elde edilmesi.
- Veri sıkıştırma, görüntü iyileştirme ve uydu fotoğrafları gibi görüntüleri içeren görüntüyü analiz etme veya kullanma.
- Çıktı, sonuçların görüntü analizine dayalı olarak değiştirilmiş bir şekilde kullanıma sunulmasıdır.

5.2 Görüntü İşleme Kullanım Alanları

- **Görüntü İyileştirme**: Elde edilen görüntülerde gürültü olması durumu ve bozulmalara neden olan şeyleri ortadan kaldırarak daha net ve temiz bir görüntü elde etme işlemidir. [25]

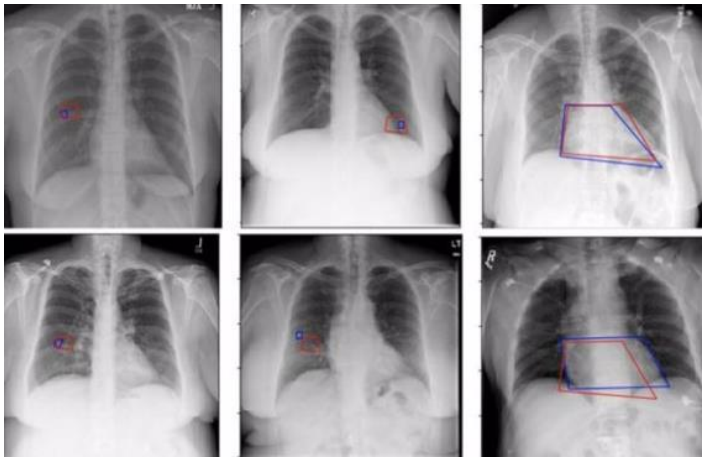
Şekil 5.1: Görüntü İyileştirme [25]

- **Cisim Tanıma:** Tespit edilmek istenen cisme göre farklı yöntemler ve algoritmaları kullanarak nesne tespiti ve nesne takibi uygulamaları gerçekleştirilebilmektedir. [25]



Şekil 5.2: Cisim Tanıma [25]

- **Sağlık Sektörü:** Görüntü işleme uygulamaları sayesinde çeşitli hastalıkların teşhisi kolaylıkla yapılabilmektedir. [25]



Şekil 5.3: Sağlık Sektörü Uygulamaları [25]

- **Savunma Sanayi:** İnsansız hava araçları, görüntü ile hedef takibi yapan roketler gibi araçların bünyesinde bulunan donanımlar, görüntü işleme sonucu elde edilen veriler doğrultusunda hareket gerçekleştirir.[25]

Proje kapsamında cisim tanıma ve takibinde kullanılan bir algoritma oluşturacağız.

5.3 Görüntü İşleme Algoritmasının Oluşturulması

Yazılım geliştirme aşamasında görüntü işleme uygulaması için takip edilmesi gereken adımlar aşağıda verilmiştir.

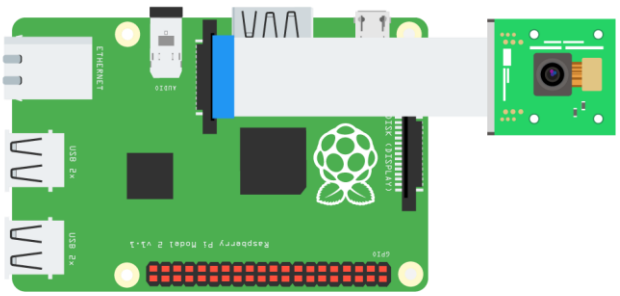
Örümcek Robot Projesi'nde amaç, renk skalası daha önceden belirlenmiş olan bir nesnenin kamera ekranındaki yerini tespit etmek ve bu nesnenin bulunduğu konuma göre servo motorlara PWM sinyalleri göndererek nesne takibini sağlamaktır.[14]

Algoritma akışı aşağıdaki şekilde ilerleyecektir.

- Görüntüyü elde etme
- Nitelik tespiti
- Nesne takibi

5.3.1 Görüntünün Elde Edilmesi

Görüntü işlemeye başlamadan önce ilk olarak görüntünün yakalanması gerekmektedir. Görüntünün elde edilmesi Raspberry Pi Camera ile gerçekleştirilir. Görüntünün kalitesi, görüntü işleme uygulamasının daha temiz ve problemsiz olmasını sağlayacaktır. Pi Camera modülü Şekil 18'de gösterildiği gibi bağlanmalıdır.



Şekil 5.4: Pi Camera Modülü Bağlantı [26]

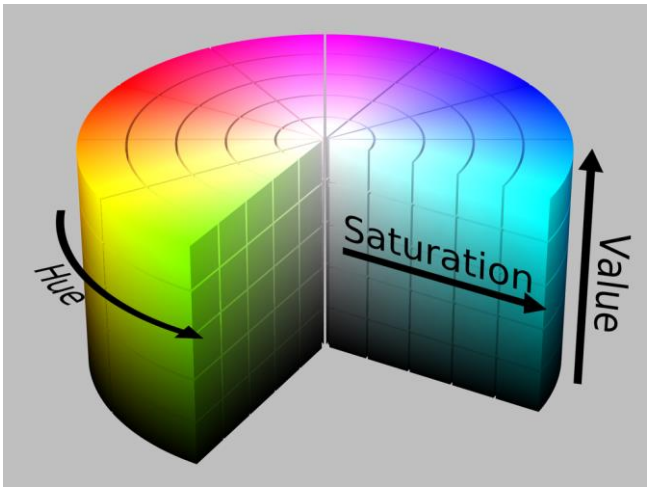
5.3.2 Nitelik (Renk) Tespiti

Elde edilen görüntü üzerinde istenen özelliklere sahip özelliğin belirgin hale getirilmesi işlemine nitelik tespiti denir. Önceden belirgin olmasını istediğimiz özelliğin otomatik olarak tespitini yapmak için bu işlemi uygularız. Bu proje kapsamında baktığımızda, robotun kırmızı renkli bir cismi tespit etmesi gerekmektedir. Kırmızı renkli olma durumu, burada cismin niteliği, özelliğidir. Kırmızı renk dışındaki bütün veriler göz ardı edilmiştir.

OpenCV ile Görüntü işleme uygulamasında RGB (Red-Green-Blue) renk skalası ve HSV modelleri kullanılır.[10]

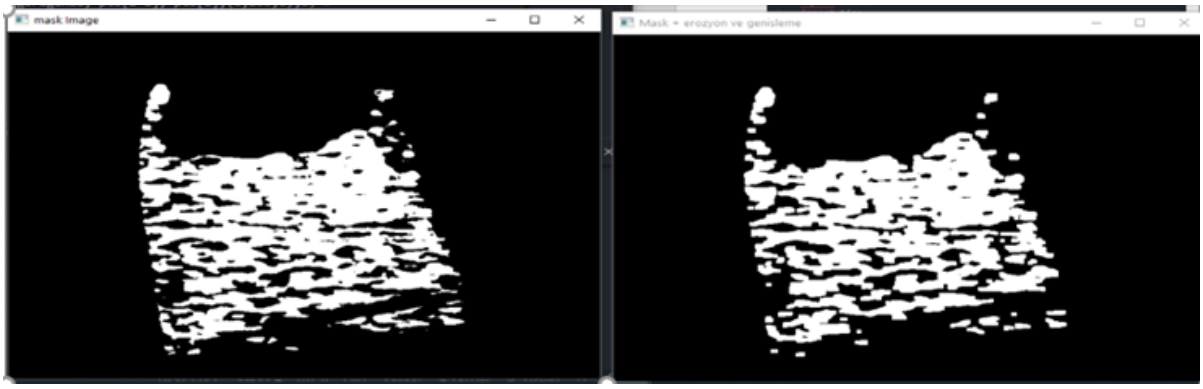
RGB skalasında ana renkler temsil edilir ve bu renk skalası 0 ve 255 değerleri arasında yer alır.

HSV ise piksellerin üç adet parametre değeri ile temsil edilmesidir. Bu parametreler ton, doygunluk ve değerdir. Renk tespitinde üst ve alt aralık olarak istenilen rengin parametreleri programa tanıtılır.



Şekil 5.5: HSV Renk Uzayı [27]

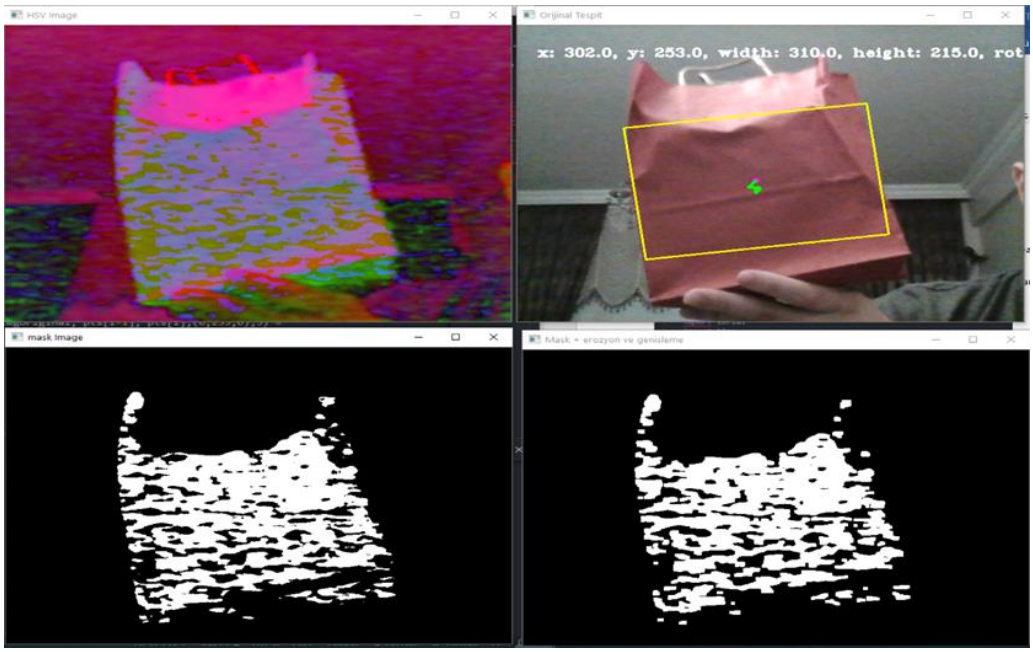
Renk tespitinde maskeleme ve gölgeleme işlemleri uygulanmalıdır. Bu işlemler tanınacak nesnenin daha net anlaşılır biçimde algılanmasını sağlar. Etrafta bulunan farklı cisimlerin siyah olarak gösterilmesini sağlar. Şekil 20'de görüleceği üzere sadece kırmızı renkli bir poşetin algılandığını ve ekrana yansıtıldığını anlayabilmekteyiz.



Şekil 5.6: Maskeleme İşlemi

5.3.3 Kamera ile Nesne Takibi

Kameradan alınan görüntülerde tespit edilen cismin takip edilmesi işleminde öncelikle tespit edilen cisme bir merkez tayin edilmesi gerekmektedir. Merkez tayin edilme işleminden sonra yapılması işlem gereken bu merkezin kamera ekranındaki konumunun değişimini hesaplayan bir momentum hesaplaması gerekir. Nesnenin kamera ekranındaki hareketini daha iyi anlamak için merkez noktasını takip edecek şekilde ekranda çizgi oluşturacak şekilde fonksiyonlar kullanılmalıdır. Konum bilgileri ekrana anlık olarak yazdırılmalıdır. Proje kapsamında Örümcek Robot'un fiziksel olarak takip sağlaması için bu cisim merkezi verileri anlık olarak Arduino'ya gönderilir.

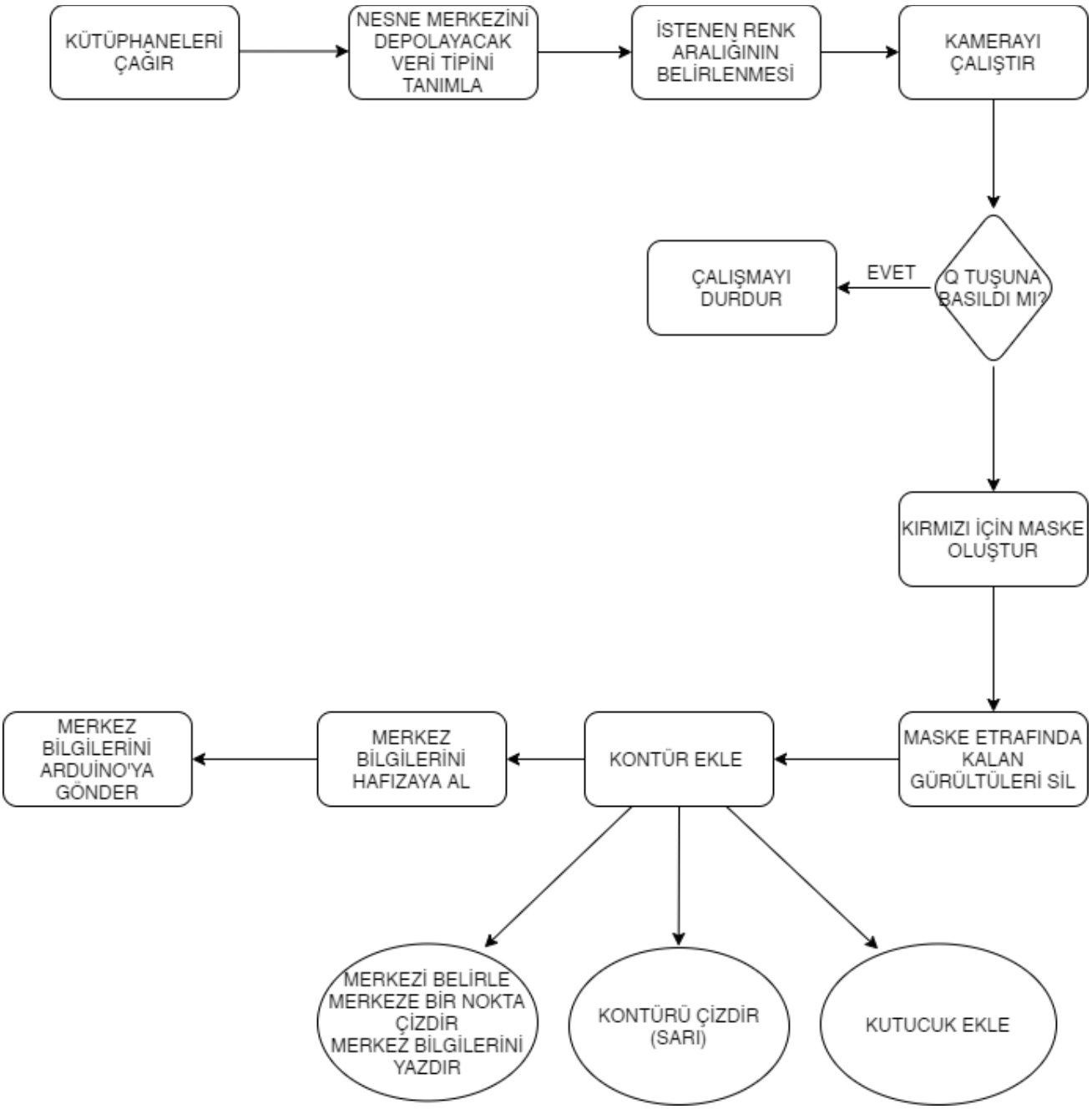


Şekil 5.7: Python ile Nesne Takibi

5.4 Görüntü İşleme Uygulamasının Python Dilinde Kodlanması

Görüntü İşleme Uygulamaları kodlama aşamasında algoritma oluşturulurken yukarıdaki adımlar izlenmiştir. Kod dizini OpenCV kütüphanesinin, numpy kütüphanesinin ve nesne merkezini depolayacak veri tipinin kütüphanelerinin import edilmesiyle başlar. Sonrasında nesne merkezini depolayacak veri tipi tanımlanır. Ardından kırmızı renk için RGB skalasında kırmızıya denk gelen en düşük ve en yüksek aralık tanımlanır. Kameradan görüntü alınması için OpenCV kütüphanesinden kamera kaydını başlatacak fonksiyon çağırılır. While döngüsü başlatılıp True değeri döndükçe while döngüsü içindeki işlemler yapılır. Bir if blok yapısı içinde blurlama, hsv, renk için maske oluşturma, maske etrafındaki görüntüleri silme işlemi yapılır. Sonrasında kontür adımına geçilir. Bu adıma geçildiğinde aynı if bloğu içinde kalınmalıdır. Kontür büyüklüğü sıfırdan büyük olduğunda cismi dikdörtgen olarak algılaması için OpenCV kütüphanesinden komut çağırılır. Cisim etrafına kutucuk oluşturması için kullanılan komut çağırılır. Cismin merkezini belli edecek komut çağırılır. Cismin merkezine pembe renkte bir nokta atanır. Kontür çizdirilir. Ve konum bilgisi output ekranına yazdırılır. Bu adımdan sonra cismin ekrandaki hareket bilgisini Arduino Uno kartına aktaracak kod bloğuna geçilir. Bu işlemlerden sonra uygulama Arduino üzerinden devam edecektir. Python kodlarına EK A' da yer verilmiştir.[6][7]

5.5 Görüntü İşleme Algoritma Akış Şeması



Şekil 5.8: Görüntü İşleme Algoritma Akış Şeması

6. ÖRÜMCEK ROBOT'UN HAREKET KONTROLÜ

6.1 Giriş

Örümcek Robot 4 kollu otonom bir robottur. Hareket mekanizması Bölüm 2.3'te verilmiştir. Bu bölümde hareket mekanizmasının belirli bir algoritma ile programlanması işlemine yer verilmiştir.

Algoritma oluştururken bulanık mantık kontrolör (FLC) ile cismin yatay eksendeki konumuna göre servo motorlara PWM (Darbe Genişlik Modülasyonu) sinyalleri gönderilir.

6.2 Hareket Algoritmasının Oluşturulması

Servo motorların sürülmesi Arduino Uno aracılığıyla sağlanmıştır. Öncelikle motorlara PWM sinyallerinin gönderilmesi için Raspberry Pi kartından nesne tespiti sinyallerinin gönderilmesi gerekmektedir. Öncelikle motorlar ve giriş pinleri tanımlanmalıdır. Sonrasında Raspberry Pi ile haberleşme için kullanılacak kodlar yazılmalıdır. Raspberry Pi'den gelen sayısal değerler bir değişkene atanmalı ve PWM sinyallerini bu değişkene göre oluşturacak fonksiyon tanımlamaları yapılmalıdır. Bu fonksiyonlar oluşturulurken dikkat edilmesi gereken hususlar Bölüm 2.3' te verilen hareket mekanizmasının uygulanması işlemidir. Yön kontrolleri farklı PWM sinyalleri ile sağlanacağından her yön için farklı bir fonksiyon oluşturulmalıdır. Yön fonksiyonları oluşturulurken motor hızları da cismi takip edebilecek miktarda olmalıdır. Tüm fonksiyon blokları oluşturulurken if-else bloğu içinde olmalıdır. Şekil 6.4'de verilen blokta algoritma akış şeması verilmiştir.

6.3 Bulanık Mantık ile Hareket Kontrolü

Örümcek Robot tespit edilen nesneyi sağlıklı bir şekilde takip edebilmesi için cismin kamera ekranındaki konumun yatay eksende belirlenmesi gerekmektedir. Yatay eksende konumun belirlenebilmesi içinse bulanık mantık kontrolörü algoritma içerinde önemli bir yere sahiptir. Şekil 6.1'de verilen sistemde Bulanık Mantık Kontrolör yapısı verilmiştir.

Şekil 6.1: FLC Sistemi

Nesnenin, Örümcek Robot'un kamera ekranındaki konum ve uzaklık bilgisine karar verirken çıkıştan gelen fark verisini bulanık mantık sistemine giriş olarak geri vermek amacıyla kurulmuştur.[8]

Motor kontrolü işlemi sonucu elde edilen, hedefe olan uzaklık ve hizalama çıkış verileri geri besleme olarak kullanılmaktadır.

Motor kontrol işlemi ile Örümcek Robot'un hangi motorunun ne kadarlık bir hız ve ivme ile kontrol edilmesi belirlenir. Motor kontrolü sonucu tespit edilen cismin yarıçapı ve yatay eksendeki konumu, hatayı azaltmak ve sıfıra yaklaştırmak amacıyla değiştirilmiştir. Bu işlem sırasında tespit edilen cismin konum ve yarıçap verisi geri besleme olarak sürekli döngü halinde kullanılmaktadır.

6.4 Bulanık Mantık ile Sonuç Çıkarma İşlemi

Bulanık Sonuç Çıkarım Sistemi (FIS) karar vermeyi sağlayan bulanık mantık sisteminin ana birimidir. Akış diyagramında mantık birimlerini kullanarak karar verme mekanizmasını inşa eder. Sistem yapısında sadeleşme amaçlanmıştır. Bulanık değerlerin net bir değere dönüşene kadar işlemler sürer.

FIS'de birden çok model mevcuttur. Bunlardan Mamdani, Tsukamoto ve Sugeno Takagi modelleri en yaygın olarak kullanılanlardır. Bu proje kapsamında Mamdani modeli kullanılarak sonuç üretilmiştir.[19]

6.4.1 Mamdani FIS Yöntemi

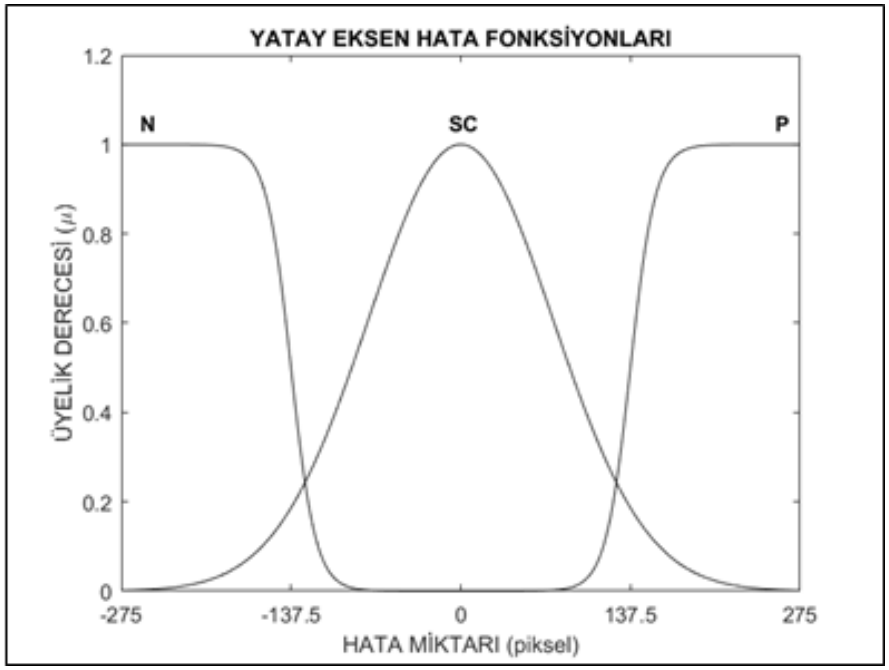
Mamdani, fonksiyonların giriş-çıkış üyelik derecelerini belirlemek için kullanılır. Belirlenen fonksiyonun kuralları ve geçerlilik dereceleri belirlenir. Sonrasında gerektirme işlemi uygulanarak önceden belirlenmiş olan kural dizinine ait sonuçlar ile bulanık kümeleri belirlenir. Birleştirme işlemi ile çıkışa verilecek bulanık küme elde edilir. Durulaştırma işlemi ile elde edilen nümerik çıkış belirlenir. Şekil 6.2'te Mamdani FIS akış diyagramı verilmiştir.[11]

Şekil 6.2: Mamdani FIS Akış Diyagramı

6.4.2 Yatay Eksende Mamdani FIS Çıkarımı

Yatay düzlemde, takibi yapılan cismin kamera ekranı ara yüzünde konum bilgileri Raspberry Pi tarafından gönderilmiştir. Cisim merkezi yatay eksende 0-550 piksel aralığında değer alır. Buradan yola çıkarak kamera ekranı ara yüzünün merkezi referans noktası 0 değeri olarak belirlenir. Üyelik fonksiyonu ise -275 ve 275 aralığında değerler alır. Anlık konum bilgisinin 275 değerinden çıkarılmasıyla hata fonksiyonunu üretiriz.[12]

Hata fonksiyonu değerinin negatif veya pozitif bir değer çıkmasıyla cismin Örümcek Robot'a göre yönü tayin edilir. Negatif bir değer üretildiğinde cisim robota göre sağda, pozitif bir değer üretildiğinde cisim robota göre soldadır.[8]



Şekil 6.3: Hata Fonksiyonları Grafiği

Tablo 6.1: FIS tablosu Yatay eksen

Hata Endeksi	FIS
Pozitif	Sola Dön
Sıfıra Yakın	Dur
Negatif	Sağa Dön

Tablo 2'de verilen komut haritası göz önünde bulundurulduğunda X olarak adlandırılan ve gövdeye direkt olarak monte edilen servo motorlar saat yönünde 90 derecelik hareketi sağlanır.

Bu işlemler bir döngü halinde devam eder. Mamdani FIS akış diyagramındaki adımlar takip edilerek elde edilen sonuçlar birleştirme işlemiyle çıkış bulanık mantık kümesi elde edilir. Bu işlem toplama işlemi ile gerçekleştirilir.

Sonuçta çıkış değerlerinin ağırlıklı ortalaması alınarak Durulaştırma işlemi uygulanarak elde edilen değer ile birlikte motorlar için oluşturulan döngü içerisinde kullanılır.

Elde edilen ağırlıklı ortalama değeri yatay eksende PWM değeri olarak kullanılır.

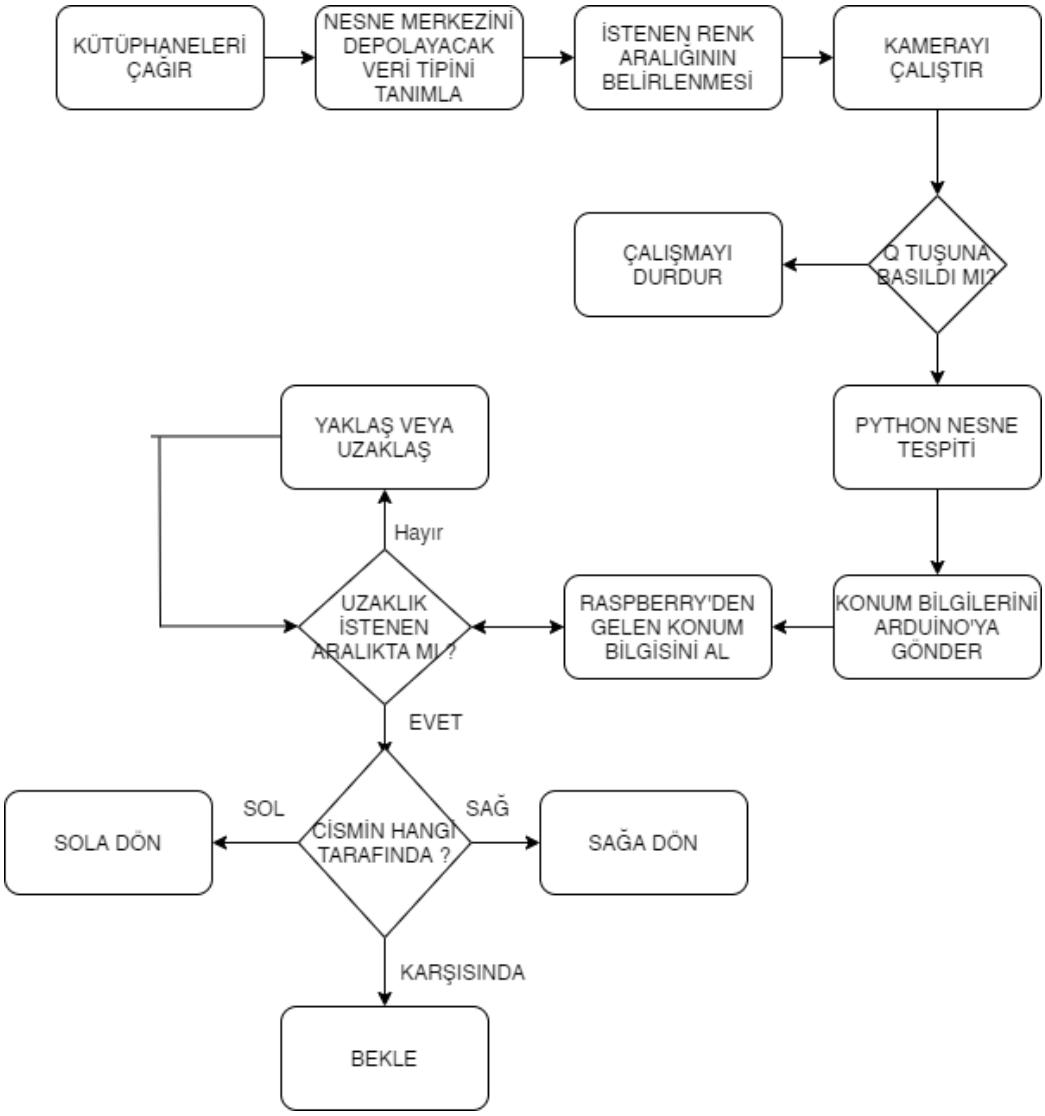
6.4.3 Mesafe Bilgisine Göre Dikey Düzlemde Basit Mantık Çıkarımı

Örümcek Robot'un cisme olan uzaklığı cismin genişlik bilgisinin belirlenmesi ile birlikte bu genişlik bilgisinin belirlenen referans değerden az ya da fazla oluşuna göre hareket algoritması oluşturulur. Algoritmada öncelik mesafe bilgisinde olmalıdır. Cismin robota olan uzaklığı istenilen mesafede ise yön kontrolleri yapılmalıdır. Öncelikle Şekil EK A.2' de verilen Python kodunun 55 ve 80. Satırları arasında algoritmaya göre yazılım oluşturulmuştur.

Tablo 6.2: Dikey Eksen Mesafe Komutları

Değer Aralığı	Komut
150'den Küçük	İlerle
150-300 Arası	İstenen Değer Aralığında, Yön Kontrollerini Yap
300'den Büyük	Gerile

Genel olarak Örümcek Robot'un kontrol şeması aşağıda verilen Şekil 6.4'de gösterilmiştir.



Şekil 6.4: Kontrol Algoritması

6.4.4 Motorların Hız Kontrolü

Motorların hız kontrolü FIS algoritmasında ortaya çıkan hata değerlerinin azaltılması işlemidir. Burada yapılacak işlemi Arduino kodları üzerinde tanımlanmış olan hız değerleri koşula bağlanarak belirli değer aralıklarında artırılıp azaltılmasıyla sağlanır.

Servo Motorların hız ayarları için tanımlı olan hız parametreleri PWM sinyallerinin Duty Cycle değerine göre şekillenir.

Bu proje kapsamında hız kontrolü ondalık sayı bir değişken atanarak hızlanma ve yavaşlama işlemleri o ondalık sayı değerin belli bir katsayıyla çarpılması veya bölünmesi işlemi ile yapılmıştır.

7. SONUÇLAR

Örümcek Robot donanım parçaları adım adım monte edilerek birleştirilmiş ve teste hazır hale getirilmiştir. Görüntü işleme uygulamasının daha sağlıklı olması için kamera açısındaki bölgeler beyaz karton ile kaplanmıştır. Beyaz bölgelerle kaplanan alan içine uzaktan kırmızı bir cisim tutularak nesne algılama ve takibi işlemleri gerçekleştirilmiştir.

Deney sırasında Raspberry Pi kartında yaşanılan ısınmadan ötürü çok zorlanmaması tavsiye edilir. Güç kaynağı olarak kullanılan 3.7V Li-on pillerin seri bağlanması ile elde edilen 7.4V'luk gerilim bu sorunun kaynağı olarak görülmektedir.

Nesne Tanıma ve nesne takibi uygulamaları başarıyla sonuçlanmıştır. Uygulanan FIS algoritmalarına göre nesneye yönelim hareketleri gözlenmiştir.

Nesne takibi sırasında servo motorların çalışmasında bazı aksaklıklar yaşanmıştır. Bu aksaklıkların Robot iskeleti üzerine düşen ağırlıktan ötürü olduğu düşünülmektedir. Deneyler az pürüzlü ortamda yapılmalıdır.

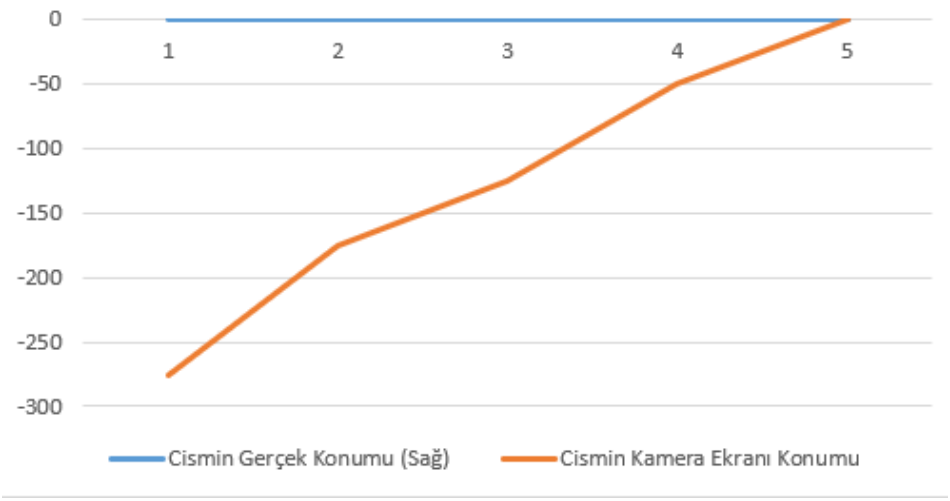
Görüntü işleme uygulaması Raspberry Pi üzerinde başarılı bir şekilde yürütülmüştür. Şekil 8.5'de deney aşaması verilmiştir. Görüntünün daha net algılanması adına arka planda beyaz bir karton kullanılmıştır.

Örümcek Robot'un tam monte edilmiş hali Şekil 8.6'de verilmiştir.

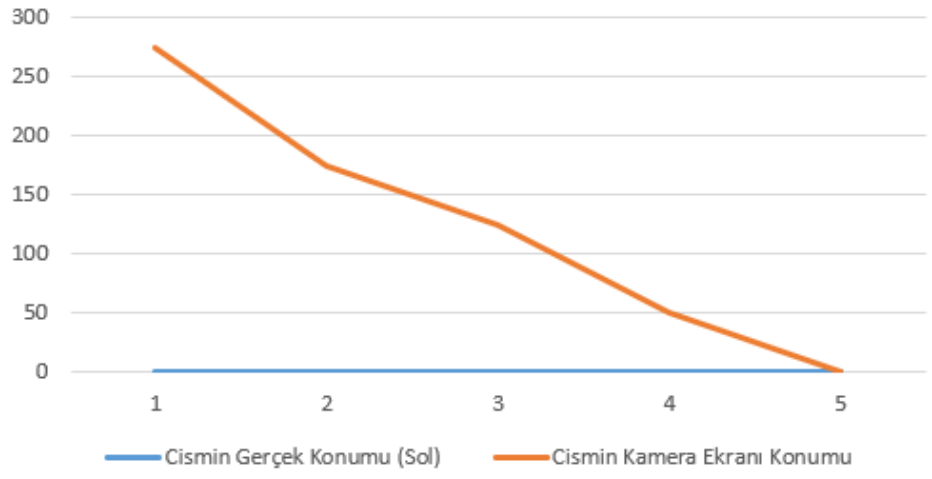
Örümcek Robot'un çalışma videosu ek dosyalarda mevcuttur.

8. DENEYSEL SONUÇLAR

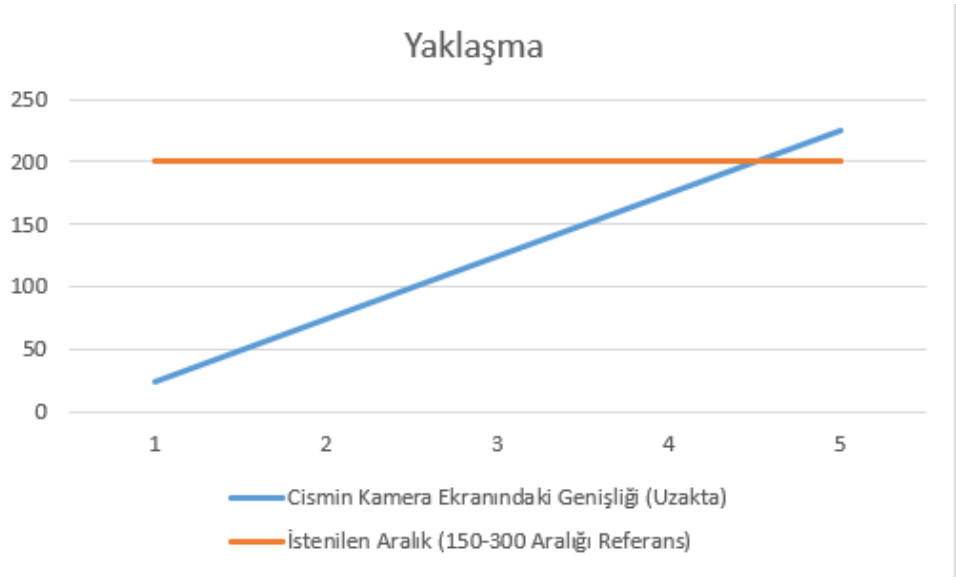
Raspberry Pi kartının çalışma yavaşlığından ötürü cismin yüksek hızda yer değiştirmesini takip etmekte zorluklar yaşanmıştır. Yavaş veya kesikli zaman aralıklarında yapılan yer değiştirmelerde takip başarılı bir şekilde yapılmıştır. Robot kasasının ve motorların ağırlıktan ötürü zorlandığı tespit edilmiştir. Şekil 8.1 ve Şekil 8.2'te Örümcek Robot'un Cisme Zamana göre yaklaşım grafikleri verilmiştir. Şekil 8.3 ve Şekil 8.4'te cisme olan uzaklığına göre yaklaşma ve uzaklaşma grafikleri verilmiştir.



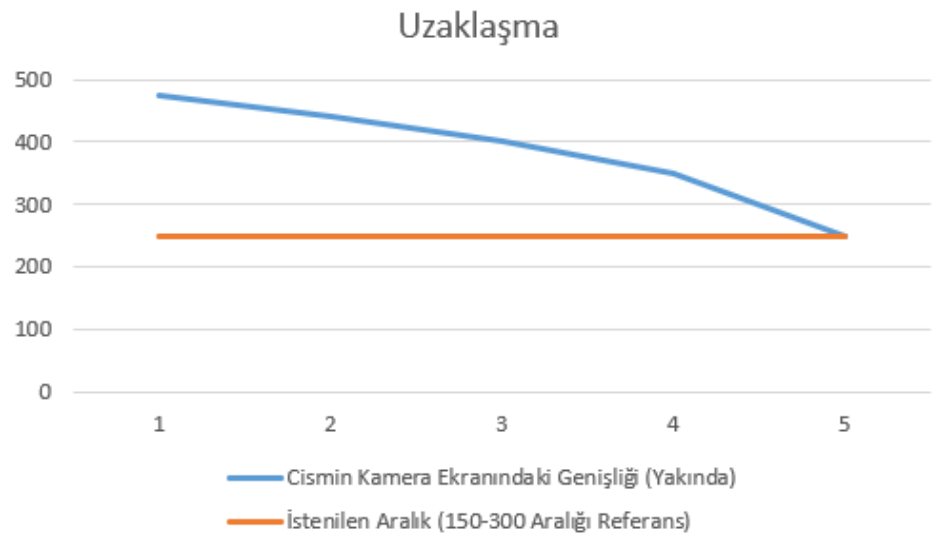
Şekil 8.1: Örümcek Robot'un Cisme (Sağda) Zamana Göre Yaklaşımı



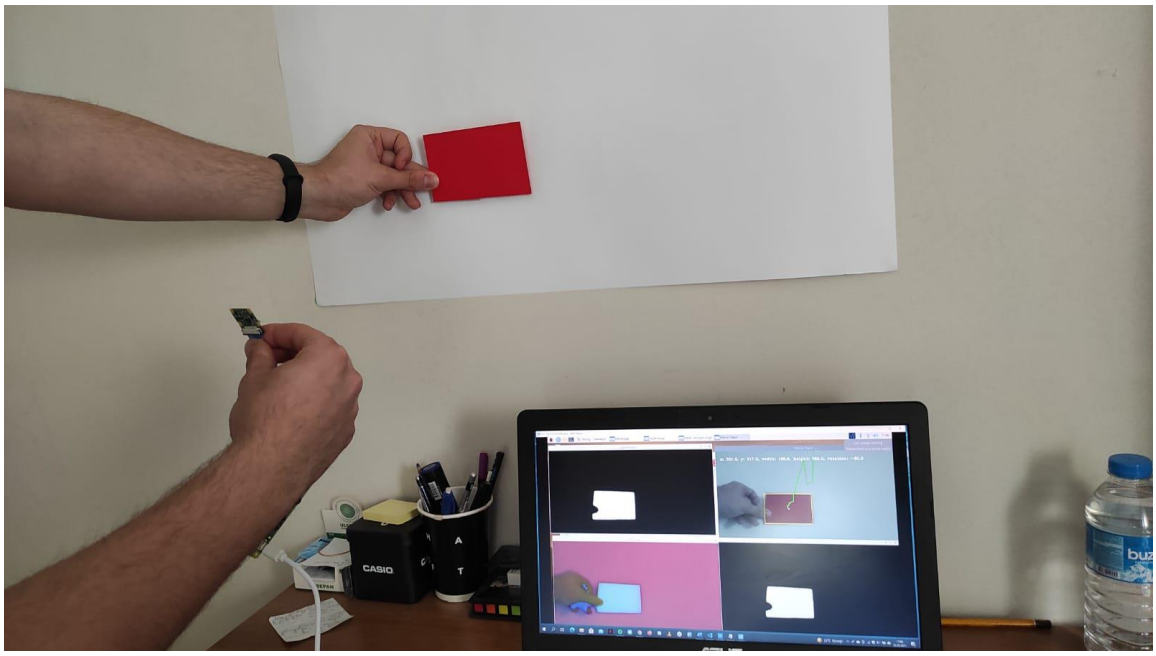
Şekil 8.2: Örümcek Robot'un Cisme (Solda) Zamana Göre Yaklaşımı



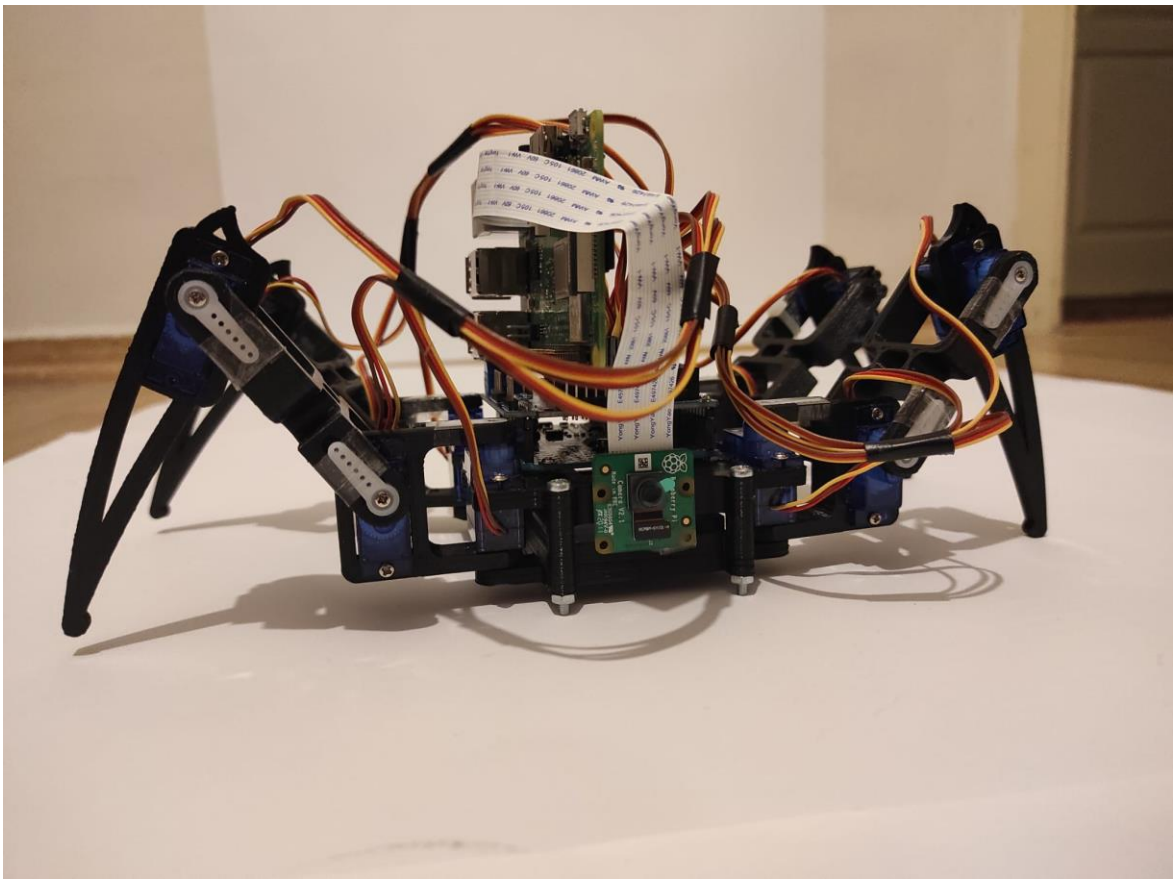
Şekil 8.3: Örümcek Robot'un Zamana Göre Cisme Yaklaşma Grafiği



Şekil 8.4: Örümcek Robot'un Zamana Göre Cisimden Uzaklaşma Grafiği



Şekil 8.5: Deney Aşaması



Şekil 8.6: Örümcek Robot

KAYNAKLAR

[1] OpenCv. 2016. https://mesutpiskin.com/blog/opencv-nedir.html (Erişim tarihi 16.05.2021.)

[2] Servo Motor. 2017. https://seferihisareml.meb.k12.tr/ (Erişim tarihi: 16.05.2021)

[3] Raspberry Pi ürün satın alım. 2020. https://urun.n11.com/arduino-urunleri-ve-setleri/raspberry-pi-4-4gb-P360380298 (Erişim tarihi : 04.04.2021)

[4] Raspberry Pi. 2021. https://www.raspberrypi.org/ (Erişim tarihi: 17.05.2021)

[5] Raspberry İşletim Sistemi kurulumu. 2020

https://www.raspi-tr.com/2012/08/03/debian-raspbian-kurulumu/ (Erişim tarihi: 15.12.2020)

[6] REN, S. (2015), HE, K. (2015), GIRSICK, R. (2015), SUN, J. (2015), Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, Cornell University

[7] MALLICK, S. (2016), Histogram of Oriented Gradients explained using OpenCV, 10 Mayıs 2021 tarihinde https://learnopencv.com/histogram-of-oriented-gradients/ adresinden alındı.

[8] YILMAZ S. (2007),” Bulanık Mantık ve Mühendislik Uygulamaları”, KOU Yayınları, Yayın No: 289

[9] YILMAZ, S. (2020), KILCI, B. (2020), 6 Serbestlik Dereceli Sualtı Aracı ve Manipülatör Sistemi ile Görüntü İşleme Uygulamaları, İleri Mühendislik Çalışmaları ve Teknolojileri Dergisi,1(2), 63-79

[10] Image Tresholding. https://docs.opencv.org/master/d7/d4d/tutorial_py_thresholding.html (Erişim tarihi: 15.03.2021)

[11] Yuanyuan Chai, Limin Jia, and Zundong Zhang(2009) , Mamdani Model based Adaptive Neural Fuzzy Inference System and its Application, *International Journal of Computational Intelligence 5:1 2009*

[12] Mamdani and Sugeno Fuzzy Inference Systems , https://www.mathworks.com/help/fuzzy/types-of-fuzzy-inference-systems.html , (Erişim tarihi: 29.05.2021)

[13] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg (2015), SSD: Single Shot MultiBox Detector, Cornell University

[14] AMBIKA CHOUDHURY, Top 8 Algorithms For Object Detection, https://analyticsindiamag.com/top-8-algorithms-for-object-detection/

[15] Raspberry Wireless Configuration, https://www.raspberrypi.org/documentation/configuration/wireless/headless.md

[17] Raspberry Pi-Arduino Serial Communication, https://roboticsbackend.com/raspberry-pi-arduino-serial-communication/

[18] Örümcek Robot 3D Baskı, https://urun.n11.com/arduino-urunleri-ve-setleri/orumcek-robot-3d-baskisi-elektronik-icermez-P440472575

[19] Örümcek Robot Tasarım, https://www.projehocam.com/arduino-android-kontrollu-orumcek-robot-yapimi/

[20] Arduino Uno, https://www.robotistan.com/arduino-uno-r3-klon-usb-kablo-hediyeli-usb-chip-ch340

[21] Arduino Shield, https://www.robotistan.com/arduino-io-genisleme-shieldi

[22] Arduino Shield Pinout , https://www.robotistan.com/arduino-io-genisleme-shieldi

[23] Raspberry Pi 3 B+ , https://market.samm.com/raspberry-pi-3-b-plus

[24] Raspberry Pi-Arduino Serial Communication Connect, https://roboticsbackend.com/raspberry-pi-arduino-serial-communication/

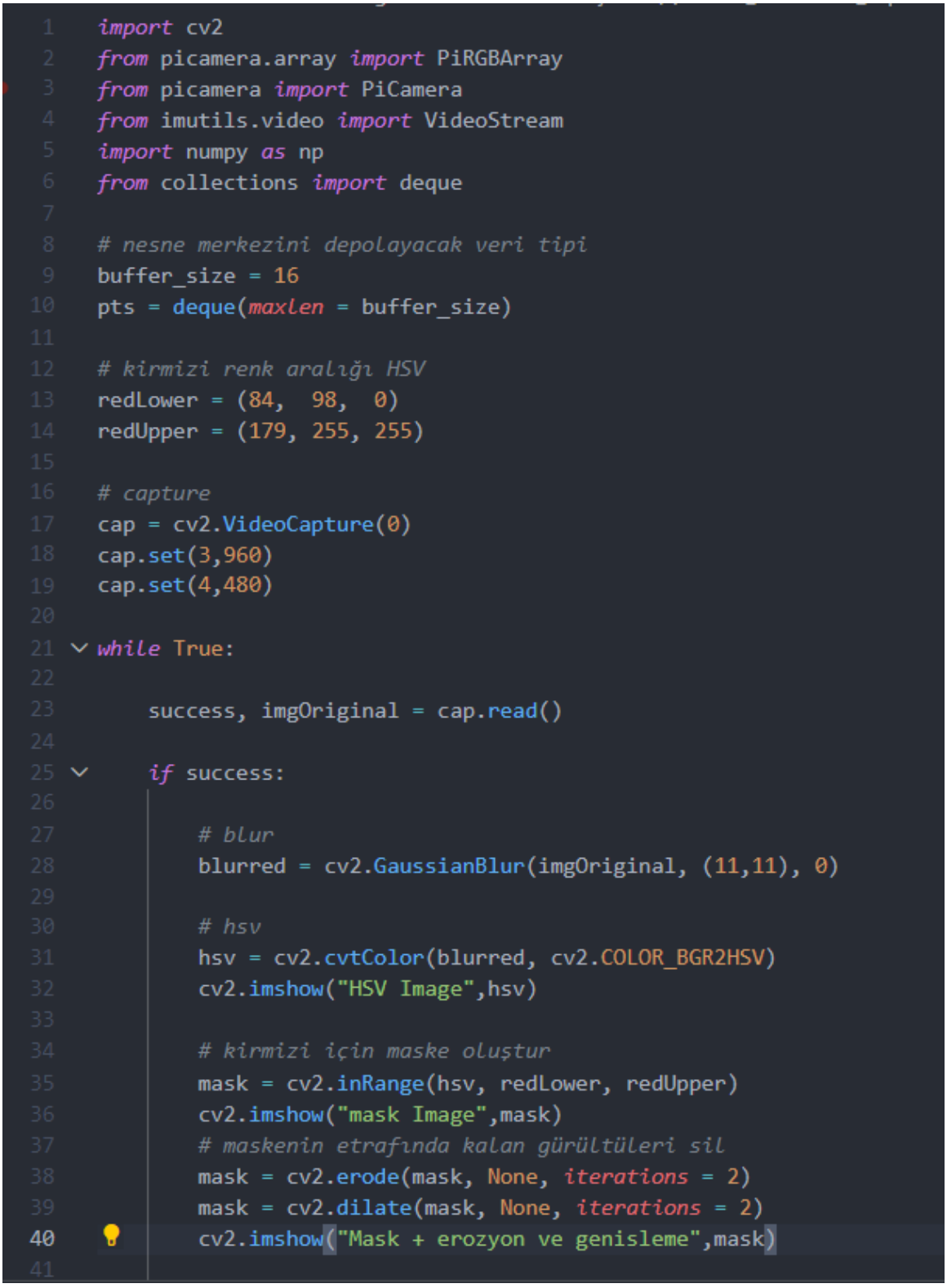
[25] https://www.ceyrekmuhendis.com/goruntu-isleme-nedir-ve-nerelerde-kullanilir/

[26] Pi Camera Bağlantı, https://www.allaboutcircuits.com/projects/how-to-build-a-picamera/

[27] HSV Renk Uzayı Gösterimi, https://tr.m.wikipedia.org/wiki/Dosya:HSV_color_solid_cylinder_saturation_gray.png

EK A

Bu bölümde görüntü işleme uygulamalarında kullanılmış olan Python kodlarına yer verilmiştir.



```python
import cv2
from picamera.array import PiRGBArray
from picamera import PiCamera
from imutils.video import VideoStream
import numpy as np
from collections import deque

# nesne merkezini depolayacak veri tipi
buffer_size = 16
pts = deque(maxlen = buffer_size)

# kirmizi renk aralığı HSV
redLower = (84,  98,  0)
redUpper = (179, 255, 255)

# capture
cap = cv2.VideoCapture(0)
cap.set(3,960)
cap.set(4,480)

while True:

    success, imgOriginal = cap.read()

    if success:

        # blur
        blurred = cv2.GaussianBlur(imgOriginal, (11,11), 0)

        # hsv
        hsv = cv2.cvtColor(blurred, cv2.COLOR_BGR2HSV)
        cv2.imshow("HSV Image",hsv)

        # kirmizi için maske oluştur
        mask = cv2.inRange(hsv, redLower, redUpper)
        cv2.imshow("mask Image",mask)
        # maskenin etrafında kalan gürültüleri sil
        mask = cv2.erode(mask, None, iterations = 2)
        mask = cv2.dilate(mask, None, iterations = 2)
        cv2.imshow("Mask + erozyon ve genisleme",mask)
```

Şekil EK A.1: Python Kod Kısım 1

```
        # kontur
        (contours,_) = cv2.findContours(mask.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
        center = None

        if len(contours) > 0:

            # en buyuk konturu al
            c = max(contours, key = cv2.contourArea)

            # dikdörtgene çevir
            rect = cv2.minAreaRect(c)

            ((x,y), (width,height), rotation) = rect

            s = "x: {}, y: {}, width: {}, height: {}, rotation: {}".format(np.round(x),np.round(y),np.round(width),np.round(height),np.round(rotation))
            print(s)

            # moment
            M = cv2.moments(c)
            center = (int(M["m10"]/M["m00"]), int(M["m01"]/M["m00"]))

            #haberleşme

            ser = serial.Serial('/dev/ttyUSB0',9600)
            kamera = x, g = width

            if g<150 :

                ser.write(b'1')

            elif g<300 and g>=150:

                if (kamera<200) :
                    ser.write(b'2')

                elif (kamera<=400 and kamera>200) :
                    ser.write(b'3')

                elif (kamera>400) :
                    ser.write(b'4')

            else:
                ser.write(b'5')
            # kutucuk
            box = cv2.boxPoints(rect)
            box = np.int64(box)
```

Şekil EK A.2: Python Kod Kısım 2

```
                # kutucuk
                box = cv2.boxPoints(rect)
                box = np.int64(box)

                # konturu çizdir: sarı
                cv2.drawContours(imgOriginal, [box], 0, (0,255,255),2)

                # merkeze bir tane nokta çizelim: pembe
                cv2.circle(imgOriginal, center, 5, (255,0,255),-1)

                # bilgileri ekrana yazdır
                cv2.putText(imgOriginal, s, (25,50), cv2.FONT_HERSHEY_COMPLEX_SMALL, 1, (255,255,255), 2)

                #b = "x: {}, y: {}".format(np.round( (int(M["m10"]/M["m00"]))),np.round(int(M["m01"]/M["m00"])))

            # deque
            pts.appendleft(center)

            for i in range(1, len(pts)):

                if pts[i-1] is None or pts[i] is None: continue

                cv2.line(imgOriginal, pts[i-1], pts[i],(0,255,0),3) #

            cv2.imshow("Orijinal Tespit",imgOriginal)

        if cv2.waitKey(1) & 0xFF == ord("q"): break

    cap.release()


    cv2.destroyAllWindows()
```

Şekil EK A.3: Python kodları kısım 3

Kodların amaç ve işlevleri yorum satırlarında verilmiştir.

EK B

Örümcek Robot'un hareket kontrol kodları ek bir belge olarak verilmiştir. Mantık Kodları aşağıdaki şekilde gösterilmiştir.

```
void loop()
{
    if(Serial.available())          //From RPi to Arduino
    {
      x = (Serial.read() - '0');    //conveting the value of chars to integer
      Serial.println(x);
    }

      if(x==1)
      {
        step_forward(4);
      }

    else if(x==3)
      {
        sit();

      }

    else if(x==4)
      {
        turn_left(3);
      }

    else if(x==2)
      {
        turn_right(3);
      }

    else if(x==5)
      {
        step_back(4);
      }
      delay(1000);
}
```

Şekil EK B.1: Arduino Hareket Mantık Kodları

EK C

Raspberry Pi VNC Viewer ve Raspbian Kurulumu

Raspberry Pi bilgisayarını monitör ve klavye olmaksızın kurulumuna ihtiyaç duyulduğunda aşağıdaki adımlar takip edilmelidir.

- Öncelikle Bölüm 3.3'te verilen talimatlar uygulanmalıdır.
- SD kart veya USB bellek içine kurulan işletim sistemi Raspberry pi'e takmadan önce linki verilen adresteki adımlar uygulanmalıdır. https://www.raspberrypi.org/documentation/configuration/wireless/headless.md [15]
- Bu adımların uygulanması verilen internet adresinde açılan sayfada bulunan kodları boş bir metin belgesine yapıştırıp kod içinde bulunan Modem ismi ve Şifre kısmı bağlanılacak olan ağın bilgileri ile doldurulur. Sonrasında metin belgesi uzantısı kaldırılarak "`wpa_supplicant.conf" ismi ile değiştirilip Raspbian işletim sisteminin kurulu olduğu dosyaya yapıştırılır.`
- Boş bir metin belgesi oluşturulup ismini uzatışını kaldırarak "ssh" olarak değiştirildikten sonra tekrar Raspbian işletim sisteminin bulunduğu konuma yapıştırılır.
- Bu adımlar sonrasında belleği Raspberry Pi'a takıp çalıştırdığımızda Modeme bağlanmış olacaktır.
- Modem ara yüzünü açıp Raspberry Pi'nin IP adresini öğrenmemiz gerekir.
- IP adresini öğrendiktinden sonra bilgisayarımıza PuTTY isimli programı indirmemiz gerekmektedir.
- Programı çalıştırdığımızda önceden öğrenmiş olduğumuz IP adresini boşluğa giriyoruz.
- Açılan terminal ekranında login as: pi , Password: Raspberry şeklinde doldurarak enter tuşuna basıyoruz.
- Giriş yaptığımızda Raspberry Pi komut istemi ile karışılaşırız.
- Komut istemi satırına sudo raspi-config yazarak enter tuşuna basıyoruz.
- Sistem ayarları ekranı karşımıza çıktığında Interfacing Options seçeneği seçilir.
- Açılan ekranda VNC seçeneği seçilir.
- Açılan ekranda VNC ayarlarını Enabled konuma alıyoruz.
- Bu işlemlerden Sonra bilgisayara VNC Viewer indirilip kurulur.
- Program çalıştırıldığında Raspberry Pi IP adresi ile birlikte girilir.

EK D

1) Öncelikle Raspberry yazılımının güncel olması gereklidir. Güncel olmama durumuna karşın güncellemeyi denetleyen ve güncel değilse güncelleme yapan kodlar girilir.

- $ sudo apt-get update
- $ sudo apt-get upgrade
- $ sudo reboot

2) Sırasıyla CMAKE geliştirici paketleri, Görüntü I/O paketleri, Video I/O paketleri, GTK geliştirme kütüphanesi ve optimizasyon paketleri kurulumu gerçekleştirilir.

- $ sudo apt-get install build-essential cmake pkg-config
- $ sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
- $ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev
- $ sudo apt-get install libxvidcore-dev libx264-dev
- $ sudo apt-get install libgtk2.0-dev libgtk-3-dev
- $ sudo apt-get install libatlas-base-dev gfortran

3) Python 3 ve Python ile bilgi işlem için kullanılan NumPy kütüphanesi kurulur.

- $ sudo apt-get install python3 python3-setuptools python3-dev
- $ sudo python3 get-pip.py
- $ sudo pip3 install numpy

4) OpenCV son versiyonu 3.4.14 kaynak kodları indirilir.

- $ cd ~
- $ wget -O opencv.zip https://github.com/opencv/opencv/archive/3.4.14.zip
- $ wget -O opencv_contrib.zip https://github.com/opencv/opencv_contrib/archive/3.4.14.zip
- $ unzip opencv.zip

- $ unzip opencv_contrib.zip

5) OpenCV derlenip yüklenmesi.

- $ cd ~/opencv-3.4.0/
- $ mkdir build
- $ cd build
- $ cmake -D CMAKE_BUILD_TYPE=RELEASE \

-D CMAKE_INSTALL_PREFIX=/usr/local \

-D BUILD_opencv_java=OFF \

-D BUILD_opencv_python2=OFF \

-D BUILD_opencv_python3=ON \

-D PYTHON_DEFAULT_EXECUTABLE=$(which python3) \

-D INSTALL_C_EXAMPLES=OFF \

-D INSTALL_PYTHON_EXAMPLES=ON \

-D BUILD_EXAMPLES=ON\

-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-3.4.0/modules \

-D WITH_CUDA=OFF \

-D BUILD_TESTS=OFF \

-D BUILD_P-RF_TESTS= OFF ..

6) Alan boyutlandırma

- $ sudo nano /etc/dphys-swapfile
- $ sudo /etc/init.d/dphys-swapfile stop
- $ sudo /etc/init.d/dphys-swapfile start

7) OpenCV derlendikten sonra oluşturma aşamasına geçilir. Oluşturma aşaması yaklaşık olarak 2 saat sürmektedir. Bu süre içerisinde Raspberry Pi ısınma sorunu yaratmaktadır. Bu sebeple Raspberry Pi'yi korumak amacıyla soğutucu alüminyum çubuklar veya DC fan gibi soğutma sistemleri kullanılmalıdır.

- $ make -j4
- $ sudo make install
- $ sudo ldconfig
- $ sudo reboot

Bu işlemler yapıldığında OpenCV kütüphanesi kullanıma hazır hale getirilmiştir.

Raspberry Pi yeniden başladıktan sonra konfigürasyon ekranında aşağıdaki kodlar yazıldığında OpenCV kütüphanesinin kurulduğunu göreceğiz.

ÖZGEÇMİŞ

1998 yılında Karabük şehrinde doğdu. İlk, Orta ve Lise öğrenimini Karabük'te tamamladı. 2017 yılında Kocaeli Üniversitesi Elektronik ve Haberleşme Mühendisliği Bölümüne yerleşmiştir. 2021 yılında bitmesi beklenen bölümde öğrenim hayatına devam etmektedir.