T.C.

BİLECİK ŞEYH EDEBALİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

AKILLI GÜVENLİK İÇİN UZAKTAN ETKİLEŞİMLİ ROBOT ARAÇ TASARIMI VE GERÇEKLENMESİ

YÜKSEK LİSANS TEZİ

SEZER KEÇELİ

TEZ DANIŞMANI

PROF. DR. CİHAN KARAKUZU

BİLECİK, 2021

10426960

T.C.

BİLECİK ŞEYH EDEBALİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

AKILLI GÜVENLİK İÇİN UZAKTAN ETKİLEŞİMLİ ROBOT ARAÇ TASARIMI VE GERÇEKLENMESİ

YÜKSEK LİSANS TEZİ

SEZER KEÇELİ

TEZ DANIŞMANI

PROF. DR. CİHAN KARAKUZU

BİLECİK, 2021

10426960

BEYAN

"Akıllı Güvenlik için Uzaktan Etkileşimli Robot Araç Tasarımı ve Gerçeklenmesi" adlı yüksek lisans/doktora/sanatta yeterlik tezi/dönem projesinin hazırlık ve yazımı sırasında bilimsel araştırma ve etik kurallarına uyduğumu, başkalarının eserlerinden yararlandığım bölümlerde bilimsel kurallara uygun olarak atıfta bulunduğumu, kullandığım verilerde herhangi bir tahrifat yapmadığımı, tezin herhangi bir kısmının Bilecik Şeyh Edebali Üniversitesi veya başka bir üniversitede başka bir tez çalışması olarak sunulmadığını, aksinin tespit edileceği muhtemel durumlarda doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi ve vermiş olduğum bilgilerin doğru olduğunu beyan ederim.

Bu çalışmanın, Bilimsel Araştırma Projeleri (BAP), TÜBİTAK veya benzeri kuruluşlarca desteklenmesi durumunda; projenin ve destekleyen kurumun adı proje numarası ile birlikte, ETİK KURUL onayı alınması durumunda ise ETİK KURUL tarih karar ve sayı bilgilerinin beyan edilmesi gerekmektedir.			
DESTEK ALINMIŞTIR		**DESTEK ALINMAMIŞTIR**	x
Destek alındı ise;			
Destekleyen kurum;			
Desteğin Türü		**Proje Numarası**	
1- BAP (Bilimsel Araştırma Projesi)			
2- TÜBİTAK			
Diğer;……			
ETİK KURUL onayı var ise;			
ETİK KURUL karar tarih/sayı:		**……/……**	

Sezer Keçeli

Tarih

……

İmza

……

ÖNSÖZ

Bu tez çalışmasının yazılmasında, çalışmamı sahiplenerek takip eden danışmanım Sayın Prof. Dr. Cihan KARAKUZU'ya değerli katkı ve emekleri için teşekkürlerimi ve saygılarımı sunarım.

Savunma sınavı sunumu sırasında değerli jüri üyeleri ve danışmanım Sayın Prof. Dr. Cihan KARAKUZU'ya çalışmamın son haline gelmesindeki değerli katkıları için teşekkürlerimi ve saygılarımı sunarım.

Son olarak bu günlere ulaşmamdaki emekleri adına değerli aileme teşekkür ederim.

Sezer Keçeli

2021

ÖZET

AKILLI GÜVENLİK İÇİN UZAKTAN ETKİLEŞİMLİ ROBOT ARAÇ TASARIMI VE GERÇEKLENMESİ

Bu çalışmada, akıllı güvenlik için uzaktan etkileşimli robot araç geliştirilmiştir. Park halinde robot araca gelebilecek zararları önlemek amaçlı görüntü işleme tekniğini kullanarak kamera yardımı ile uzaktan uyarı sistemi gerçeklenmiştir. Robot aracın engele çarpma ihtimalinde sesli uyarı vermesi sağlanmış, elektronik göstergede gerekli yazılı bildirim vermesi temin edilmiştir. Ayrıca internet üzerinden e-posta ile mesaj gönderilerek araç sahibinin uzaktan bilgi edinmesi hedeflenmiştir. Robotun geliştirilmesinde Raspberry Pi ve Arduino mini bilgisayar kartları, HC-SR04 ultrasonik mesafe sensörü, L298N motor sürücü kartı, Buzzer, HC05 Bluetooth sensörü, ESP-01 Wi-Fi modülü, 16x2 LCD ekran, Raspberry Pi kamera modülü ve enerji temini için LiPo pil kullanılmıştır.

Araca olası tehlike algılanması için hareketli görüntülerin, belirli rengin, kameradan alınan görüntü üzerindeki karakterlerin okunması amacıyla Raspberry Pi kartına Python OpenCV kütüphanesi kullanılarak Python programlama dili ile oluşturulan yazılım yüklenmiştir. Hareketli görüntü ve istenilen renk tespit edildiğinde led ile ışıklı uyarı sağlanmış LCD ekrana mesaj yazdırılmıştır. Bu işlemlerle birlikte aynı zamanda internet üzerinden istenilen yere mesaj gönderilmesi de sağlanmıştır. Tesseract programı kullanılarak resim üzerindeki karakterlerin okunması sağlanarak özellikle plaka bilgileri kaydedilmiştir. Ayrıca kaydedilen dosyanın internet üzerinden e-posta ile istenilen yere gönderilmesi de sağlanmıştır. Özetle bu çalışmada güvenlik isterleri için uzaktan etkileşim sağlayan ve denetlenebilen akıllı bir robot araç geliştirilmiştir.

Anahtar Kelimeler: Uzak Denetim, Akıllı Güvenlik, Görüntü İşleme, Robot Araç.

ABSTRACT

REMOTE INTERACTIVE ROBOT VEHICLE DESIGN AND IMPLEMENTATION FOR SMART SECURITY

In this study, a remote interactive robot vehicle was developed for smart security. A remote warning system was implemented with help of a camera by using the image processing technique in order to prevent any damage to the robot vehicle in park. In the event of the robot vehicle hitting any obstacle, it was ensured that it gave an audible warning, and it was ensured that it gave the necessary written notification on the electronic display. In addition, it is aimed that the vehicle owner can obtain information remotely by sending a message via e-mail over the internet. In the development of the robot, Raspberry Pi and Arduino mini computer boards, HC-SR04 ultrasonic distance sensor, L298N motor driver board, Buzzer, HC05 Bluetooth sensor, ESP-01 Wi-Fi module, 16x2 LCD screen, Raspberry Pi camera module and LiPo battery for energy supply were used.

A software created with Python programming language using Python OpenCV library was loaded on the Raspberry Pi card in order to read moving images, certain colors and the characters on the image taken from the camera in order to detect possible danger to the vehicle. When a moving image and a desired color are detected, a light warning was formed with the LED and a message is printed on the LCD screen. Along with these processes, it is also possible to send messages to any desired place over the internet. By using the Tesseract program, the characters on the picture were read and especially license plate information was recorded. In addition, it is also provided to send the recorded file to any desired location via e-mail over the internet. In summary, in this study, an intelligent robot vehicle that provides remote interaction and control for security requirements has been developed.

Keywords: Remote Control, Smart Security, Image Processing, Robot Vehicle.

İÇİNDEKİLER

Sayfa

ŞEKİLLER LİSTESİ

Sayfa

KISALTMALAR VE SİMGELER LİSTESİ

PWM: Sinyal Genişlik Modülasyonu (Pulse Width Modulation)

Rx: Almak (Receive X)

Tx: İletmek (Transmit X)

I/O: Giriş/Çıkış (Input/Output)

IDE: Tümleşik Geliştirme Ortamı (Integrated Development Environment)

ADC: Analog Dijital Dönüştürücü (Analog to Digital Converter)

GND: Toprak **(**Ground)

OCR: Optik Karakter Tanıma (Optical Character Recognition)

LBP: Yerel İkili Örüntüler Operatörü (Local Binary Patterns)

HOG: Yönlü Gradyan Histogramı (Histogram of Oriented Gradient)

LCD: Likit Kristal Görüntü (Liquid Crystal Display)

IoT: Nesnelerin İnterneti (Internet of Things)

LED: Işık yayan diyot (Light Emitting Diode)

1. GİRİŞ

Günümüzde gelişen teknoloji ile üretilen araçlara her geçen gün yeni özellikler eklenmekte bu eklenen özellikler ile araç kullanımı kolaylaşmakta ve güvenlik sistemleri daha iyi hale gelmektedir. Araç trafikte seyir halinde iken kaza yaptığında sesli uyarı verilerek veya istenilen yere mesaj gönderilerek bir an önce harekete geçilmesini sağlayıp olası bir can kaybı önlenebilir. Gelişen teknolojinin sunduğu imkanlarla güvenlik önlemleri de eklenerek akıllı araç uyarı ve denetleme sistemi geliştirmek mümkün hale gelmiştir. Alarm sistemleri ve kameralar olası tehlikelere karşı ev, işyeri güvenliği alanında hırsızlık ve diğer olumsuzluklara karşı uyarıcı sesli ikaz, yanıp sönen ışık ve internet üzerinden mesaj yoluyla acil durum çağrısı yapan elektronik güvenlik sistemleri bulunmaktadır.

Arduino kartı**,** elektronik devre elemanları ile bilgi alışverişi yaparak denetim sağlayan kolay programlanabilen açık kaynak kodlu mini bir bilgisayardır. Arduino kartına bağlanan sensörler ile etkileşimli sinyaller elde edilip bu sinyallerin incelenmesiyle akıllı denetim ve/veya gözetim yapmak mümkündür. Bu bağlamda çeşitli robotlar ve elektronik sistemler tasarlanabilir. Bu çalışmada, ilk örneği (prototipi) gerçeklenen bir robot araçta Arduino kartı kullanılarak HC05 modülü ile cep telefonu üzerinden DC motorların hareketi, engele olan mesafesinin hesaplanarak LCD ekran üzerine yazılması, engele 30 cm kaldığında buzzerdan sesli uyarı vermesi 15 cm kaldığında ise aracın durması sağlanmıştır.

Raspberry Pi kredi kartı büyüklüğünde tek bir karttan oluşan mini bir bilgisayardır. Bu mini bilgisayar üzerinde yer alan CSI kamera portuna takılan kamera ile OpenCV kütüphanesinin görüntü işleme komutları kullanılarak elde edilen görüntünün sayısallaştırılmasıyla üzerinde farklı işlemler uygulanarak anlamlı yorumlanabilir sonuçlar elde edilebilir. Elde edilen bu sonuçlar ile sesli uyarı, ışıklandırma, internet üzerinden bilgilendirme ile güvenlik sağlanabilir. Yapılan çalışmada Raspberry Pi kartı ve Python programı ile görüntü işleme kullanılarak hareketli görüntülerin elde edilmesi, istenilen rengin tespit edilmesi, resim üzerindeki yazıların okunması ile araç üzerindeki plakanın tespit edilmesi sağlanmıştır. Elde edilen bu bilgilerin Raspberry Pi mini bilgisayarın Wi-Fi özelliği kullanılarak internet üzerinden istenilen yere gönderilmesi sağlanmıştır.

Literatürde Arduino kartı, Raspberry Pi mini bilgisayarı, robot araç tasarımı, Arduino ve Raspberry Pi ile kullanılan sensörler ve kartlar, Python programı ile görüntü işleme konuları ile ilgili yapılan bazı çalışmalara ait bilgiler aşağıda verilmektedir.

Balcı, yapmış olduğu çalışmada Raspberry Pi mini bilgisayarını kullanılarak uzaktan verilerin anlık olarak takip edilmesiyle ortam izleme sistemi geliştirmiştir. Bu çalışmada akıllı ev otomasyon sistemleri ve nesnelerin interneti ile insanlar evlerinden uzak oldukları zamanlarda kurulan sistem ile ev ortamının sürekli kontrol edilerek istenmeyen bir durum meydana geldiğinde kullanıcının internet üzerinden bilgilendirilmesi sağlanmaktadır. Böylece meydana gelebilecek olumsuz durumlardan bir an önce haberdar olunarak bu tür durumlarda oluşabilecek maddi ve manevi zararların etkisinin ortadan kaldırılması veya en aza indirilmesi sağlanabilir. Bu çalışmada meydana gelebilecek yangının tespit edilmesi, hırsızlık durumuna karşı hareketli görüntülerin elde edilmesi, parlaklık ölçümünün yapılarak gece gündüz tespiti, ortamdaki nem ve sıcaklık değerleri, gaz ölçümü, su basmaları gibi veriler hakkında bilgi sahibi olunması sağlanmaktadır. (Balcı, 2019).

Göçerler, yapmış olduğu çalışmada yapay zeka tabanlı plaka tanıma ile bariyer kontrolü gerçekleştirilmiştir. Kontrol ve güvenlik amacıyla plakanın tespit edilmesi sağlanmıştır. Bu sistem otoyol giriş ve çıkışlarında bulunan gişeler de, otopark giriş ve çıkışlarında, trafik kurallarına uymayan araçların tespit edilmesi gibi durumlarda kullanılmaktadır. Tasarlanan sistemde plakanın tespiti için elde edilen görüntüye morfolojik işlemler uygulanarak görüntüdeki nesnelerin kenar tespiti yapılmaktadır. Plaka alanı tespit edilerek çıkarılan plaka alanındaki istenmeyen görüntüler temizlendikten sonra eğitilmiş OCR motoru kullanılarak plaka tanıma işlemi gerçekleştirilmektedir(Göçerler, 2019).

Yılmaz, yapmış olduğu çalışmada Raspberry Pi kartı kullanılarak nesne tespit ve takip robotunun tasarımı ve modellenmesi gerçekleştirilmiştir. Raspberry Pi kartı ile Python programı ve OpenCV kütüphanesi kullanılarak kameradan alınan anlık video görüntüsü ile video akışında, nesnenin tespit edilmesi, hareketli görüntünün algılanması ve nesnenin koordinatlarını hesaplayıp yerini tespit ederek takibini yapabilen bir robot gerçekleştirilmiştir (Yılmaz, 2019).

Aksu, yapmış olduğu çalışmada Arduino ile çalışan çok fonksiyonlu robot tasarımı gerçekleştirilmiştir. Arduino tabanlı bir telefon ile kullanıcının Arduino kartı kullanılarak yapılan robotla insanların ulaşamayacağı ve giremeyecek kadar küçük yerlere erişilmesi gibi durumlarda Bluetooth üzerinden haberleşerek bilgi alış verişi sağlanmıştır. Tasarlanan robot araca verilen ses komutları ile robotun istenilen yöne hareketi, ortamdaki gaz ölçüm değerleri ve sıcaklık bilgilerinin alınması sağlanmıştır (Aksu 2020).

Uysal, yapmış olduğu çalışmada Arduino kartı ve Bluetooth modülü kullanılarak uzaktan denetimli robot araç tasarımı gerçekleştirmiştir. Çalışmada oluşturulan araç platformu ile Android işletim sistemli bir telefon ile Bluetooth modülü üzerinden robot araç kontrol edilmiştir. Gerçeklenen robot araçla Bluetooth modülünün, servo ve DC motorların, sensörlerin kullanımı sağlanmıştır (Uysal, 2019).

Karacı ve Erdemir, yapmış oldukları çalışmada Arduino kartı kullanarak çok sensörlü ve düşük maliyetli gezgin hizmet robotu geliştirdiler. Bu robot ile sensörler kullanılarak insanın giremeyeceği yerlerdeki gaz, sıcaklık ve nem gibi istenilen bilgiler elde edilmiştir. Böylece gaz kaynaklı sorunların engellenmesi hedeflenmiştir. Mesafe sensörü kullanılarak robot aracın engele olan mesafesi hesaplanmış, engeller tespit edilerek engellere çarpmaması, kapalı bir ortamda bir açıklık varsa o açıklığı tespit ederek çıkması sağlanmıştır. Ayrıca üzerine kamera monte edilerek uzaktan kontrol edilmesi sağlanmıştır (Karacı ve Erdemir, 2017).

Anh ve Song, tutucusuna bağlı kamera ile hareketli görüntüleri bulmak için bir robot kol geliştirmişlerdir. Robot kolun eklem hareketleri servo motorlar ile sağlanmıştır. Arka arkaya görüntüler alınarak birbirleri ile karşılaştırılmıştır. Hareketli görüntü bulunduğu zaman, görüntünün konumu tespit edilmiştir (Anh ve Song, 2010).

Çetinkaya, yapmış olduğu çalışmada ten ve yüz bulma teknolojisine dayalı insan sayısını bulmayı gerçekleştirmiştir. Yerleştirilen kameraların yardımıyla eğitim ortamlarında bulunan panoları izleyen öğrencilerin sayısının bulunması hedeflenmiştir. Ten rengi tespitinde HSV ve YCbCr renk uzayları kullanılmış bu renk uzaylarından HSV renk uzayının daha başarılı sonuçlar verdiği görülmüştür. Bu sistemin tasarımında OpenCV kütüphanesi kullanılmıştır (Çetinkaya, 2012).

2. KULLANILAN DONANIM BİRİMLERİ

Bu tez çalışmasında Raspberry Pi mini bilgisayarı ve Arduino UNO programlanabilir kartları kullanılmıştır. Yapılan çalışmanın mekanik tasarımının bağlantıları ve kontrol edilmesi Arduino UNO kartı ile OpenCV kütüphanesinin kullanımı ile görüntü işleme kısmı ise Raspberry Pi mini bilgisayarı kullanılarak tasarlanmıştır.

Tasarlanan robot aracın DC motor ile tekerlek hareketi, Bluetooth modülü ile cep telefonu üzerinden kontrolü, ESP8266 modülü ile internet üzerinden bilgi alış verişi, HC-SR04 sensörü ile mesafe hesaplaması, Buzzer ile sesli uyarı verilmesi, led ile ışıklı uyarı verilmesi, 16x2 LCD ekran ile istenilen bilgilerin görüntülenmesi sağlanmış ve bu donanım birimlerinin bağlantılarının yapılarak denetiminin sağlanması için Arduino UNO kartı kullanılmıştır.

Raspberry Pi mini bilgisayarı ve CSI arabirimini kullanan kamera ile hareketli görüntülerin tespit edilmesi, elde edilen görüntü üzerindeki plaka bilgisinin okunması, belirlenen bir nesnenin ve yabani hayvan gibi belirli bir renge sahip nesnelerin tespit edilmesi için istenilen rengin okunması gibi işlemlerin yapılması sağlanmıştır.

2.1. Raspberry Pi

Raspberry Pi, üzerinde mikroişlemci, RAM, GPIO pinleri ve bir bilgisayar için gerekli tüm özelliklere sahip olan tek bir devre kartı üzerinde oluşturulmuş mini bir bilgisayardır. Raspberry Pi Şekil 2.1'de görüleceği üzere küçük boyutlu, üzerinde CPU, RAM, Ethernet kartı, Bluetooth, Wireless, Micro HDMI, 40 adet GPIO pin , Display ve Camera port bulunan programlanabilir bir karttır. Masaüstü bilgisayarların yapabildiği birçok işlemi yapabilmektedir. İşletim sistemi Raspberry Pi üzerine takılan micro-SD kart içerisine yüklenmektedir. (Vikipedi, 2019)

Şekil 2. 1. Raspberry Pi mini bilgisayarı

Raspberry Pi için birçok işletim sistemi kullanılmaktadır. Bu işletim sistemleri içerisinde Raspberry Pi için geliştirilmiş ve varsayılan işletim sistemi olan Raspbiandır. Bu

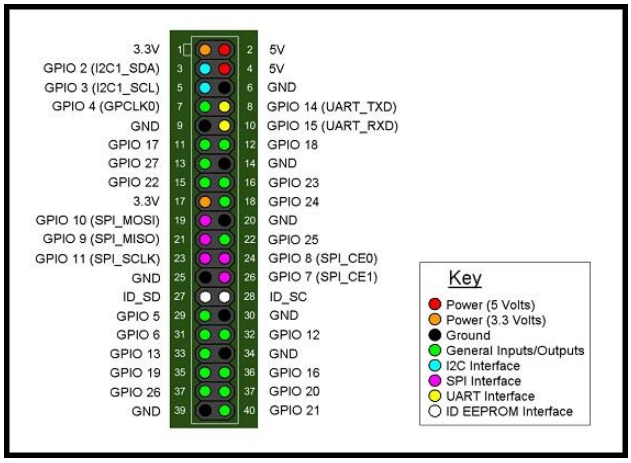
işletim sistemi kullanılarak Raspberry Pi'ye klavye, fare ve monitör bağlanıp masaüstü bilgisayar gibi kullanım mümkündür (Erşahin, 2015:7-18). Yapılan çalışmada Raspberry Pi mini bilgisayar kartına Raspbian işletim sistemi kurulmuş ve Python programı ile tez çalışmasındaki yazılımlar geliştirilmiştir.

2.1.1. Raspberry Pi otomatik başlatma

Raspberry Pi kartına monitör bağlamadan pil ile güç verildiğinde programın kendiliğinden otomatik olarak başlaması için birden fazla yol bulunmaktadır. Yapılan çalışmada Python dosyalarının kendiliğinden otomatik olarak çalışması için Raspbian işletim sisteminin kurulmasıyla sistem içerisinde bulunan *rc.local* dosyası kullanılmıştır. Çalıştırılacak dosya *rc.local* dosyasının içerisinde *exit0* komut satırının üzerine eklenerek otomatik çalıştırılması sağlanır. Dosyanın otomatik başlaması için *rc.local* dosyasının yanı sıra *.bashrc* dosyasına ekleme yaparak, çalıştırılacak programı *init.d* klasörüne ekleyerek, *systemd* dosyalarını kullanarak ve *crontab* kullanarak istenilen programın otomatik çalıştırılması sağlanır (Dexterindustries, 2020).

2.1.2. Raspberry Pi GPIO pinleri

Mini bir bilgisayar olan Raspberry Pi masaüstü bilgisayarın yapmış olduğu birçok görevin yanı sıra, elektronik devrelerde kontrol kartı olarak da kullanılmaktadır. Bu işlem Raspberry Pi üzerinde bulunan Şekil 2.2'de gösterilen GPIO pinleri programlanarak sağlanır.



Şekil 2. 2. Raspberry Pi GPIO pinleri
Kaynak: (Bektaş, 2017)

Raspberry Pi kartı üzerinde elektronik sistemlerle haberleşmek ve kontrolü sağlamak için 40 adet GPIO pini bulunmaktadır. Bu pinlerden 5 volt çıkış sağlamak için 2 adet, 3,3 volt çıkış sağlamak için 2 adet, devrenin tamamlanması için 8 adet GND ve EEPROM işlemleri

için 2 adet pin bulunmaktadır. Geriye kalan 26 pin giriş/çıkış pinidir. Bu giriş/çıkış pinlerinden 2 tanesi I2C seri haberleşmesini, 5 tanesi SPI seri haberleşmesini, ve 2 tanesi de UART seri haberleşmesini desteklemektedir.

Raspberry Pi GPIO pinlerini isimlendirirken iki farklı dizilim kullanılmaktadır. Bu dizilimler BCM ve BOARD dizilimidir. BCM dizilimi sıralı olmayan GPIO numaralarından oluşmaktadır. BOARD dizilimi ise pinlerin fiziksel numaralandırılmasıyla 1'den 40'a kadar devam eden sıralı sayılardan oluşur.

GPIO pinlerini programlarken öncelikle GPIO pin dizilimi ayarlanmalıdır. Yapılan çalışmada Raspberry Pi üzerinde bulunan GPIO pinlerini programlamak için Raspbian işletim sistemi içerisinde bulunan Python programı kullanıldı.

Raspberry Pi GPIO pinleri programlanırken zaman komutlarını kullanmak için *time* kütüphanesi kullanılır. Kütüphane içerisinde bulunan *time.sleep()* fonksiyonu ile programa gecikme verilebilir. Genel giriş/çıkış GPIO pinleri Arduino kartı üzerinde bulunan PWM pinleri gibi kullanılarak 0 volt ile 5 volt arasında istenilen gerilim değeri oluşturulabilir. Bu şekilde Raspberry Pi mini bilgisayarının genel giriş/çıkış pinleri istenildiğinde Arduino kartının dijital pinleri gibi istenildiğinde Arduino kartının PWM pinleri gibi kullanılabilir (Bektaş, 2017).

2.1.3. Raspberry Pi kurulum

Normal bir bilgisayardan pek bir farkı olmayan Raspberry Pi mini bilgisayara işletim sistemi kurmak için micro-SD kart kullanılır**.** Raspberry Pi Imager, Noobs gibi programlar kullanılarak micro-SD karta işletim sistemi yüklenir. Yapılan çalışmada iki ayrı mikro SD kart kullanılarak birisine Raspberry Pi Imager programı kullanılarak, diğerine ise Noobs programı kullanılarak işletim sistemi yüklenmiştir.

2.2. Arduino Uno Kartı

Arduino İtalyan elektronik mühendisleri tarafından geliştirilen kolay programlanabilir, ucuz bir donanıma sahip, açık kaynak kodlu esnek, kolay kullanımlı donanım ve yazılım tabanlı mini bilgisayardır. Kartın bir görseli Şekil 2.3'te verilmiştir.

Şekil 2. 3. Arduino UNO kartı

Arduino kartların çok çeşitli modelleri vardır ve bunlar kullanım amacına göre farklılık göstermektedir. Bu çalışmada Arduino UNO modeli kullanıldı. Arduino UNO ile birçok haberleşme işlemi gerçekleştirilir. Bu kartın 14 adet dijital giriş / çıkışı mevcuttur. Bu çıkış pinlerinden 6 tanesi 0-5 volt arasında istenilen gerilim değerini üretmek için PWM çıkışı olarak kullanılırken, 6 tanesi de analog giriş olarak kullanılmaktadır. RX ve TX pinleri kullanılarak seri haberleşme yapılmaktadır. Arduino IDE içerisinde bulunan seri port ekranı Arduino kartı ve bilgisayar arasında metin tabanlı bilgilerin gönderilip alınmasını sağlanmaktadır. Arduino kartı ile bilgisayar arasında USB üzerinden bir haberleşme algılandığında, Arduino üzerinde yer alan RX ve TX yazan ledlerin yandığı görülür. Arduino UNO kartında bir adet seri port bulunurken çeşitli kütüphaneler kullanılarak bu sayı yazılımsal olarak arttırılabilmektedir (Baykal ve Yücelen, 2016).

2.2.1. Arduino UNO pinleri

Arduino UNO üzerinde 14 adet dijital giriş,çıkış pini ve 6 adet analog giriş,çıkış pini bulunmaktadır**.** Digital pinler 0 veya 5 volt okuma veya güç çıkışı sağlayarak giriş, çıkış işlemi yapılmasını sağlayan pinlerdir. Bu pinlerden bazılarının birden fazla görevi bulunmaktadır. Örneğin D13 pininin diğer dijital pinler gibi kullanılmasına ilave olarak *serial clock* pini olarak da kullanılır yani SPI seri haberleşmede 2 cihazın saat hızlarını eşitleyerek eşzamanlı çalışmasını sağlamak için kullanılır. Bunun yanı sıra dijital pinler PWM sinyal genişlik modülasyonu olarak da kullanılabilir yani normalde digital pinler 0 veya 5 volt gerilim verebilirken PWM özelliği bulunan dijital pinleri *analogWrite* komutu kullanılarak 0-5 volt arası istenilen bir gerilim değeri elde edilebilir. Bu şekilde PWM pinleri kullanılarak DC motorun dönme hızı, ledin parlaklık miktarı gibi uygulamalar yapılabilir. (Ekmekci, 2017:54).

Analog pinler analog giriş / çıkış amaçlı kullanılan A0-A5 pinleridir. Sensörlerden gelen sinyaller okunabilir ve analog olarak 0-5 volt arası istenilen gerilim değeri elde

edilebilir. Analog giriş pinlerinin hassasiyeti 10 bittir. Bu değer 0-5 volt arası gerilimin 2^{10} yani 1024 parçaya bölünerek yaklaşık 0,0049 volt hassasiyetle gerilim değeri okunabilmektedir. Analog giriş çıkış pinleri potansiyometre, sensörler gibi elektronik devre elemanları tarafından elde edilen değerlerin dijitale çevrilerek Arduino kartına yüklenen yazılım ile kullanılması sağlanır.

Voltage In pini Arduinoya güç vermek için kullanılan bir giriştir. Arduino IDE programında yazılan kodlar USB bağlantısı ile bilgisayardan Arduino kartına aktarılır. Arduinoyu bilgisayar bağlantısının olmadığı durumlarda gerekli olan gerilimin sağlanabilmesi için *Voltage In* pini kullanılır. Yapılan çalışmada *Voltage In* pini kullanılarak LiPo pil ile gerekli olan güç sağlanmıştır. *Reset* pini Arduino içerisinde yüklü olan yazılımın çalışması devam ederken sıfırlanarak kodların baştan çalıştırılması için kullanılır.

Seri haberleşme de RX pini giriş için, TX pini ise çıkış için kullanılır. Bu pinler kullanımda iken bilgisayarda yazılan kodlar Arduino UNO'ya USB üzerinden aktarılamaz. USB üzerinden Arduino'ya kodların atılabilmesi için RX ve TX bağlantılarının çıkarılması gerekir. I2C seri haberleşmenin yapılması için Analog4 ve Analog5 pinleri kullanılır. Analog4 pini SDA, Analog5 ise SCL pini olarak kullanılır. Bu pinlerle seri haberleşme sağlanarak, iki cihaz arasında bilgi alışverişi yapılır. Seri haberleşmeyi sağlamanın diğer bir yoluda SPI pinleridir. Arduino üzerinde 10, 11, 12, 13 numaralı pinler SPI pin olarak kullanılır. Yapılan çalışmada 16X2 LCD ekrana yazı yazdırmak için I2C seri haberleşmesi, ESP8266 Wi-Fi modülünün ve HC-05 modülünün bağlantılarını sağlamak için UART seri haberleşmesi kullanılmıştır (Robotik Sistem, 2020).

2.2.2. Arduino kartı kurulumu

Arduino IDE programı Arduino uyumlu kartlara program yazmak ve yüklemek için kullanılır. C ve C++ dilleri ile yazılmış bir platformlar arası uygulamadır. Orijinal Arduino üzerinde bulunan ATmega16u2 çipi USB bağlantısını UART bağlantısına dönüştürmek için kullanılır. Fakat klon kartlarda ATmega16u2 çipinin yerine CH340 çipi kullanılmaktadır. Bu iki çip işlev olarak aynı görevi yerine getirmektedir. ATmega16u2 çipi bulunan Orijinal Arduinoda Arduino IDE programının kurulmasıyla gerekli olan sürücü (driver) dosyaları otomatik olarak kurulmaktadır. CH340 çipi bulunan klon Arduino'nun kullanılabilmesi için Arduino IDE programının haricinde sürücü dosyalarının kurulması gereklidir.

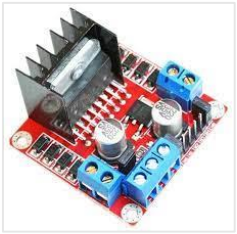
2.3. DC Motor

DC motor, doğru akım elektrik enerjisini mekanik enerjiye dönüştüren bir donanım birimidir. Motorun içerisinde bulunan sargılara elektrik akımı uygulanarak, motor içerisinde bulunan sabit mıknatıslara zıt yönde oluşan manyetik kuvvetin etkisi ile hareket oluşması sağlanır. Redüktör, yüksek dönme kuvveti üretir, elektrik motorlarının çıkışından düşük dönme kuvveti ve düşük hız elde edilebilir. Dişli kutu olarak da isimlendirilen redüktör motorlar ile hız düşürülerek yüksek tork ile farklı yönlerde dönme elde edilir.

Arduino Uno kartı ile 50 mA akım üretilmektedir. Bu akım DC motoru sürmek için yeterli değildir. Bu nedenle DC motoru doğrudan Arduino kartına bağlamak yerine motor sürücü kullanılır. Yapılan çalışmada DC motoru sürmek için L298N motor sürücü kartı kullanıldı.

2.4. L298N Motor Sürücü Kartı

L298N DC Motor sürücüsü, iki motoru sürmek için hazırlanmış bir çok Arduino projelerinde kullanılan bir motor sürücü kartıdır. Kartın bir resmi Şekil 2.5'de verilmiştir.



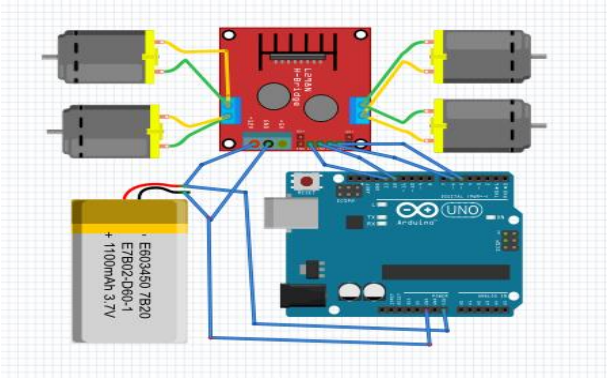
Şekil 2. 4. L298N motor sürücü

L298N DC motor sürücü Bluetooth çok çeşitli motor kontrol uygulamalarında kullanılabilir. Sürücünün bazı tipik özellikleri aşağıda listelenmiştir.

- İki ayrı motor aynı anda kontrol edilebilir
- Kanal başına 2A akım verebilir.
- Motor sürücü üzerinde dahili regülatörü vardır.
- Kısa devre ve yüksek sıcaklık koruması vardır.
- Kart üzerinde dahili soğutucu vardır.
- Kartın kenarlarında istenilen yüzeye sabitleyebileceğiniz vida delikleri bulunmaktadır.

L298N motor sürücü üzerinde in1, in2, in3, in4 girişleri DC motorun istenilen yönde hareketini EnA, EnB girişleri ise hız kontrolünü sağlamak için kullanılır. Hız kontrolü

yapılmayacaksa her bir kanal için etkinleştirme(enable) pinleri 5 V'a yani lojik 1 sinyaline bağlanır. Bu bağlantının kolay bir şekilde yapılması için 2 adet jumper bulunur. Şekil 2.6'da DC motorların Arduino kartı ve L298 motor sürücü kartına bağlantıları görülmektedir (Tezel, 2017:14)



Şekil 2. 5. DC motorların motor sürücü ve Arduino kartlarına bağlantıları

2.5. HC-SR04 Ultrasonik Mesafe Sensörü

HC-SR04 ultrasonik mesafe sensörü ses dalgalarını kullanarak karşısındaki nesneye olan mesafesini bulan 4 pinli elektronik devre elemanıdır. Bu sensörün ölçüm yaptığı mesafe aralığı 2 cm ile 400 cm arasında olup 5 volt gerilim ile çalışmaktadır. Sensörün bir görseli Şekil 2.6'da verilmiştir.

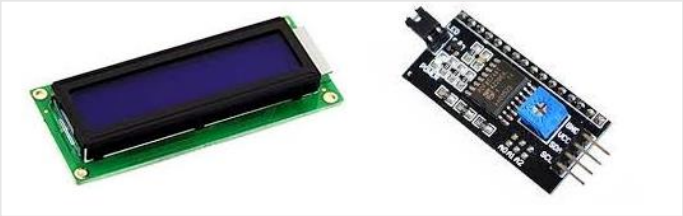
Şekil 2. 6. Ultrasonik mesafe sensörü

Bu sensörler insan kulağının duyamayacağı kadar yüksek frekansta ses dalgası üreterek robotların engele olan mesafesini hesaplayıp yönlerinin belirlenmesinde sıkça kullanılmaktadır. Mesafenin hesaplanmasında sensörün Trig ve Echo pinleri kullanılır. Bu pinlerden Echo pini ses dalgalarını gönderen Trig pini ise ses dalgalarını alan kısımdır (Daştan, 2019:13).

Sensörün çalışma prensibi Trig pininden uygulanan bir sinyal ile ultrasonik ses dalgasının yayılması sağlanır. Bu ses dalgası herhangi bir nesneye çarpıp geri döndüğünde Echo pini aktif hale gelir. İki sinyal arasındaki süre bulunarak, nesnenin sensörden olan uzaklığı belirlenir (Sailteknoloji, 2020).

2.6. 16x2 LCD Ekran

Genel görünümü Şekil 2.7'de verilen LCD (Licuit Crystal Display), sıvı kristal ekran elektrikle kutuplanan sıvının ışığı tek fazlı geçirmesi ve önüne eklenen bir kutuplanma filtresi ile gözle görülebilmesi ilkesine dayanan bir görüntü teknolojisidir.



Şekil 2. 7. 16X2 LCD ekran üst ve alt görünümleri

Mikrodenetleyicilerle birlikte kullanılan LCD ekranlar genellikle karakterlerin gösterilmesi için kullanıldığından çözünürlükleri düşüktür ve tek renkten oluşurlar. LCD ekranlar, yazılan kodlarla denetimlerin yapılarak istenilen bilgilerin elde edilmesi ile kullanıcıya büyük kolaylıklar sağlamaktadır. Ayrıca fiyatları düşük olduğu için Arduino, Raspberry Pi uygulamalarında yaygın bir şekilde kullanılmaktadır. Karakter LCD ekranlar, sadece karakterlerin görüntülenmesi için yapılmış ve karakter sayısı kadar özel hücre içeren ekranlardır. Ekranda yer alan her hücre yatayda 5, düşeyde 8 pikselden oluşur.

LCD ekranın 16×2 olarak ifade edilmesi LCD'nin 2 satırdan ve 16 sütundan oluştuğunu göstermektedir. Yani ekran her satırda 16 karakter olmak üzere iki satırda toplam 32 karakter görüntüleyebilir. Bunun dışında kaydırma yaparak 32 karakterden fazlası da görüntülenebilir. 16x2 LCD ekranlarda 16 adet pin vardır. Bu pinler ekranın üstünde, altında veya her iki tarafında da bulunabilir. Arka aydınlatma ışığı bulunmayan LCD ekranlarda 14 adet pin yer almaktadır. Ekran aydınlatması bulunan 15 ve 16 numaralı pinler, ekranlarda arka aydınlatma ışığını yakmak için kullanılır. LCD ekranı, Arduino ile kullanmak için *LiquidCrystal I2C* kütüphanesi kullanılır.

LCD ekranların Arduino gibi mini kartlarla bağlantılarını yapmak için fazla kabloya ve pine ihtiyaç duyulmaktadır. I2C modülü ile bu ihtiyaç 4 pin ile çözülmektedir. Bu pinler GND, VCC, SDA ve SCL pinleridir. I2C modülü üzerinde bulunan ekran parlaklık ayarı ve kontrast ayarının yapılabilmesiyle ekranının bu ayarlarını kontrol etmek için fazladan bağlantı yapılmasına gerek kalmaz.

2.6.1. LCD 16x2 kullanımı

LCD ekran bağlantıları yapılarak *LiquidCrystal_I2C* kütüphanesi eklenir. Kütüphanenin desteklediği fonksiyonlar kullanılarak LCD ekrana yazı yazdırılır. *LiquidCrystal_I2C* kütüphanesi içerisinde yer alan fonksiyonların bir bölümü aşağıda görülmektedir (Elektrobot, 2019).

init(): LCD ekranı başlatmak için kullanılır.

clear(): LCD ekranını temizlemek için kullanılır.

home(): İmleci ekranın sol üst köşesine getirmek için kullanılır.

setCursor(): İmlecin istenilen satır ve sütuna gelmesi sağlanır.

cursor(): Alt çizgili imlecin açılmasını sağlar.

noCursor(): Alt çizgi imlecini kapatmak için kullanılır.

blink(): Yanıp sönen imlecin açılmasını sağlar.

noBlink(): Yanıp sönen imlecin kapanmasını sağlar.

display(): Ekranı açmak için kullanılır.

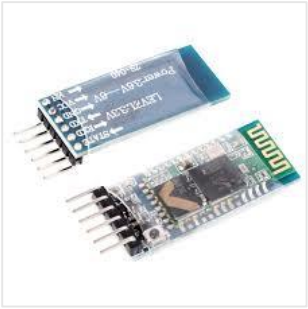
noDisplay(): Ekranı kapatmak için kullanılır.

backlight(): Arka plan ışığını açmak için kullanılır.

noBacklight(): Arka plan ışığını kapatmak için kullanılır.

2.7. HC-05 Bluetooth Modülü

Bluetooth, radyo frekansını kullanarak kablo bağlantıları olmadan kısa mesafede kablosuz iletişimi sağlayan teknolojisinin adıdır. Bluetooth modüllerinin HC-03, HC-04, HC-05, HC-06 (Şekil 2.8) gibi modelleri bulunmaktadır. Ancak ülkemizde yaygın olarak kullanılan modeleri HC-05 ve HC-06′ dır.



Şekil 2. 8. HC-05 modülü

Bluetooth modüllerinin içersinde adı, şifresi ve iletim hızı(baud Rate) hızı bilgileri varsayılan olarak ayarlanmış bir şekilde gelir. Varsayılan olarak gelen bu bilgiler adı HC-05, şifresi 0000 veya 1234 ve iletim hızı ise 9600 olarak ayarlanmıştır. Varsayılan olarak gelen bu bilgiler istenildiği zaman değiştirilebilir ve bu değiştirme işleminin yapılması için AT komutları kullanılır. Birbiri ile haberleşmenin sağlanacağı alıcı ve verici olarak kullanılan devre elemanlarının iletim hızları aynı olmalıdır.

HC-05 ana(master) ve bağımlı(slave) modunda çalışırken, HC-06 sadece bağımlı modda çalışmaktadır, Bluetooth bağımlı modda iken modüle başka bir Bluetooth cihazı tarafından bağlantı yapılabilirken, modül ile başka bir Bluetooth cihaza ilk bağlantı yapılamaz. Ana modda kullanılan Bluetooth modülü ile diğer Bluetooth cihazlarına doğrudan ilk bağlantı yapılabilir. HC-05 hem ana hem bağımlı modda çalışırken HC-06 sadece bağımlı modda çalışmaktadır. HC-05 modülünün bağımlı olarak kullanıldığı ve diğer modüllere ilk bağlantıyı sağlayabildiği için HC-06 ya göre daha çok tercih edilmektedir.

Bluetooh modülü ile seri haberleşme kullanılarak bilgi alış verişi sağlanır. Tx pini verici Rx pini ise alıcı olarak kullanılır. Bu modül veri alış verişini 3.3V ile yapmaktadır. Verici ucundan gönderilen veri mikrodenetleyici tarafından algılanır. Mikrodetleyicinin çıkış gerilimi 5V olduğu zaman Bluetooth modülü zarar görebilir. Bu nedenle mikrodenetleyici çıkışına gerilim bölücü bağlanarak istenilen gerilim elde edilmelidir (İkizoğlu, 2020). Robot aracın cep telefonu ile denetiminin sağlanması için HC-05 modülü kullanılmıştır.

2.8. ESP8266 Wi-Fi Modülü

ESP8266 Şekil 2.9'da verilen görünümde bir Wi-Fi modülüdür. Bu modül ile kablosuz erişim noktası kurulabilir ve kablosuz ağlara bağlantı sağlanabilir. Modülün kendi işlemcisi bulunduğu için üzerinde bulunan I/O (giriş/çıkış) pinleri kullanılabilmektedir. Böylece Arduino ve buna benzer herhangi bir mikrodenetleyici kartı bulunmasa da modül ile çalışabilir.

Şekil 2. 9. ESP8266 modülü

Bu modül Wi-Fi bağlantı noktası istemci(client) olarak, bir Wi-Fi erişim noktası (AccessPoint) olarak veya hem istemci hem de erişim noktası olarak kullanılabilir. İlk defa 2014 yılında AI-Thinker firması tarafından IoT modül haline getirilen çipin ESP8266-01'den ESP8266-14'e kadar çeşitli modelleri bulunmaktadır.

ESP8266-01 kablosuz ağlara bağlanabilen, kablosuz ağ erişim noktası kurulabilen bir modüldür. Bilgisayara bağlayıp programlamak için ek adaptöre ihtiyaç vardır. Modülde varsayılan olarak Arduino'da kolay kullanım için AT komutlarını destekleyen yazılım mevcuttur.

ESP8266-12 kablosuz ağlara bağlanabilen, kablosuz ağ erişim noktası kurulabilen ESP8266-01 modül özelliklerinin yanı sıra Arduino kartı gibi programlanabilme özelliğindedir. Diğer adı NodeMCU olan ESP8266-12 modülüne Arduino kartı ve Wi-Fi modülünün birleştirilmiş hali denilebilir. NodeMCU'yu programlamak için Lua dili ve Arduino IDE kullanılır. NodeMCU kartı ile uzaktan çiçek sulama, uzaktan sıcaklık ve nem kontrolü ya da uzaktan RGB led kontrolü gibi uygulamalar yapılabilir. (Unal, 2020)

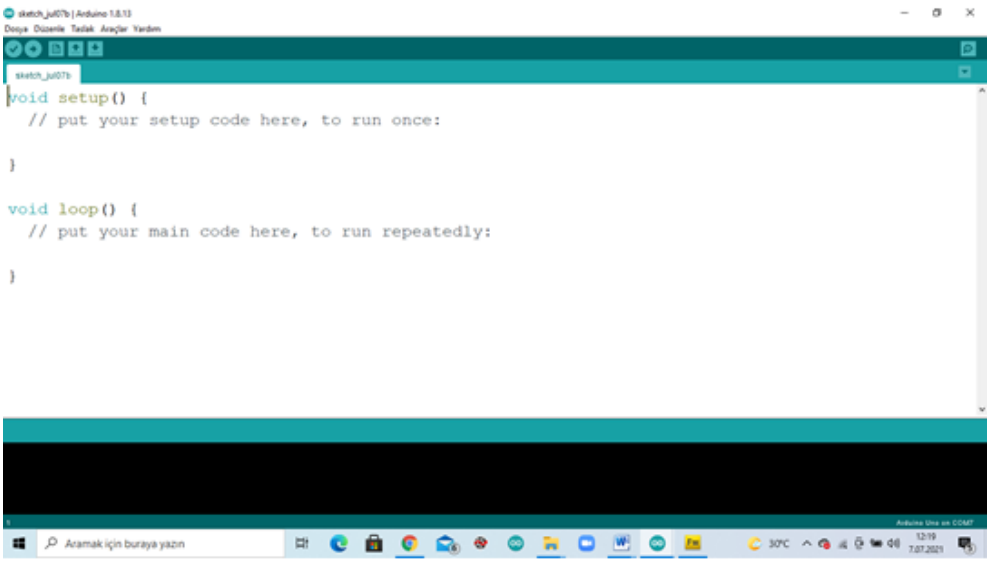
IoT(Internet of Things) nesnelerin internet bağlantısını sağlayarak birbirleriyle iletişim kurup bilgi alışverişini sağlayan teknolojidir. Bu teknoloji ile güvenlik sistemleri, ev aletleri, bilgisayarlar, robotlar, telefonlar birbirleriyle uyumlu bir şekilde çalıştırılabilmektedir. IoT cihazları birbirleriyle haberleşmesinin yanısıra insanlarla iletişime geçerek istenilen konuda bilgi sahibi olunması sağlanır. ESP8266 modülü ile Thingspeak, Blynk, Remotexy gibi platformlar aracılığıyla nesnelerin interneti teknolojisi kullanılabilir. Bu şekilde ESP8266 modülü ile verilerin cihazlardan toplanması, analiz edilmesi ve kullanıcıya bu analiz ve değerlerin sunulması için kullanılır. ESP8266 modülü ve IoT tekniği kullanılarak tatilde iken internet üzerinden çiçek sulama, oda sıcaklığını ve nem değerini öğrenme gibi uygulamalar yapılır. (Dönmez, 2020)

3. KULLANILAN PROGRAMLAR

Raspberry Pi mini bilgisayarı ve Arduino Kartı ile robot araç tasarımı, sensörlerin kontrol edilmesi ve internet üzerinden denetim ve toplama gibi birçok uygulama için bu kartların programlanması gerekir. Yapılan çalışmada Arduino UNO kartının programlanması için Arduino IDE yazılımı, Raspberry Pi mini bilgisayarının programlanması için Python programlama dili kullanılmıştır. Arduino kartına kodların yüklenmesi bilgisayar üzerinden USB bağlantısı ile yapılmaktadır. Raspberry Pi mini bilgisayarına kodların yüklenmesi ise hem bilgisayar üzerinden hem de mikro SD karta işletim sistemi yüklenip karta fare, klavye ve monitör eklenmesiyle bilgisayar gibi kullanılarak herhangi bir bilgisayar bağlantısı olmadan kendi üzerinden kodlar yüklenir.

3.1. Arduino IDE

Arduino yazılımı bir geliştirme ortamı (IDE) ve kütüphanelerden oluşur. IDE, Java dilinde yazılmış, C/C++ dillerinde yazılan kütüphaneleri kullanarak Arduino kartını programlamak için kullanılan açık kaynak kodlu, C programlama diline benzeyen tümleşik geliştirme ortamıdır. Arduino kütüphaneleri sayesinde mikrodenetleyici konusunda detaylı bilgi sahibi olmayı gerektirmeden Arduino kodlaması yapılır. Bilgisayarda Arduino IDE ile yazılan kodlar USB kablosu ile Arduino kartına aktarılır. Arduino IDE editörü ilk açıldığında karşımıza Şekil 3.1'de bulunan görüntü gelmektedir.



Şekil 3. 1. Arduino IDE editörü ekran görüntüsü

Arduino IDE editörü ilk açıldığında karşımıza gelen editör içerisinde *setup*() ve *loop*() olmak üzere 2 adet fonksiyon yer almaktadır. Bunlar Arduino IDE

programı çalıştırıldığında otomatik olarak ekrana gelmektedir. Arduino kodlarını yazarken en çok kullanılan metotlar bunlardır.

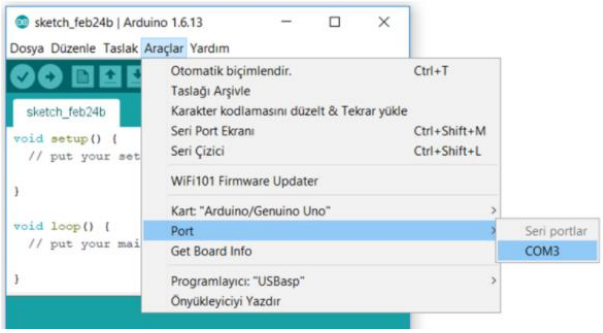
setup() fonksiyonu Arduino IDE editöründe oluşturulan sayfada varsayılan olarak bulunan fonksiyondur. Arduino üzerinde bulunan pinlerin yapılandırma işlemleri gibi proje için uygulama öncesi gerek duyulan işlemler bu fonksiyon içerisinde belirtilir. Bu fonksiyon program çalıştırıldığında sadece bir defa çalışır ve daha sonra *loop*() fonksiyonu içerisindeki kodlar çalıştırılır. Arduino kartının gücü kesilip geri verildiğinde tekrar çalışır.

loop() fonksiyonu da *setup*() fonksiyonu gibi Arduino IDE editöründe oluşturulan sayfada varsayılan olarak bulunan fonksiyondur. Setup fonksiyonu içerisindeki kodların çalıştırılmasından hemen sonra çalıştırılır ve programın çalıştırılması durdurulana kadar kodların çalışması devam eder. Bu şekilde sonsuz bir döngü oluşturularak sürekli tekrar edilmesi gereken işlemlerin yapılması sağlanır (Ateş, 2017).

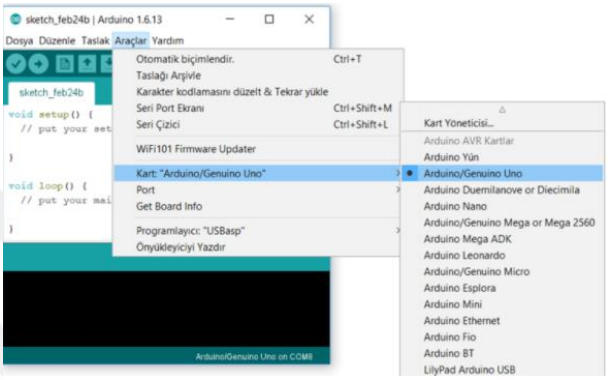
3.1.1. Arduino IDE ayarları

Bilgisayarda birden fazla kartı programlamak için bağlantı kurulması durumunda çalışmak istenilen kartın bağlı olduğu COM port bilgisi öğrenilmelidir. Arduino üzerinde USB bağlantısını seri UART bağlantısına dönüştüren bir çip bulunur. Klon Arduino kullanıldığı durumlarda CH340 isimli bir çip kullanıldığından COM port bilgisinin öğrenilmesi için sürücü yüklemesi yapılmalıdır. Orijinal Arduino UNO'da sürücü yükleme işlemi, Arduino IDE programının yüklenmesiyle otomatik olarak ATmega16u2 çipi tarafından sağlandığı için sürücü yükleme ihtiyacı yoktur.

Sürücüler yüklendikten sonra Arduino kattın COM portu öğrenilir. Arduino IDE editörünün araçlar menüsünden portlar bölümünden COM portu ve kartlar bölümünden kullanılan Arduino kartın modeli seçilir. Bu şekilde yazılan kodların Arduino karta yüklenmesi hazır hale gelir. Aşağıda yer alan Şekil 3.2'de Arduino IDE editöründe port şeçiminin nasıl yapıldığı gösterilmiştir (Semiz, 2018). Şekil 3.3'te kullanılan Arduino kartının seçiminin nasıl yapıldığı gösterilmiştir.



Şekil 3. 2. Arduino IDE port kontrolü

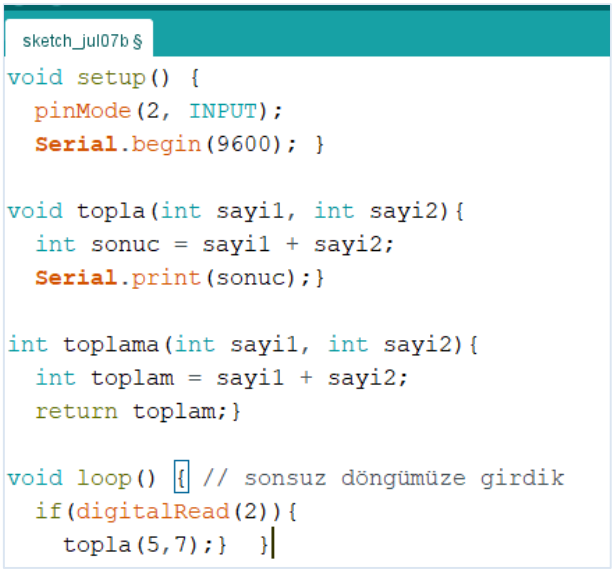


Şekil 3. 3. Arduino IDE kart seçimi

3.1.2. Arduino IDE fonksiyonlar

Fonksiyonlar, bir program içerisinde aynı işlemin defalarca yapılması durumunda, aynı kodların tekrar tekrar yazılması yerine yapılması gereken işlem için bir fonksiyon yazılarak gerekli görülen zamanlarda çağırmak için kullanılır. Fonksiyonlar parametre alabilir ve geriye değer döndürebilir. Arduino IDE editöründe *loop*() ve *setup*() fonksiyonu varsayılan olarak bulunmaktadır.

Geriye değer döndürmeyen fonksiyonların başında *void* ifadesi kullanılır. Daha sonra fonksiyon ismi yazılır. Aşağıda verilen Şekil 3.4'te *topla* adında geriye değer döndürmeyen fonksiyon oluşturulmuştur. Geriye değer döndüren fonksiyonlarda *"void"* deyimi yerine fonksiyonun *int, string* gibi geri dönüş türü ve fonksiyon adı yazılır. Fonksiyondan geri dönüş değeri almak için sonunda *return* deyimi kullanılır. Yine Şekil 3.4'te *toplama* adında geriye değer gönderen fonksiyon oluşturulmuştur.



Şekil 3. 4. Arduino IDE fonksiyon gösterimi

3.1.3. Kullanılan Arduino IDE komutları

Arduino kartının programlanmasında oluşturulan kodlar üç ana bölümden oluşur. Bu bölümler değişkenler, yapı elemanları ve fonksiyonlardır. Yapı elemanları, birden fazla kod satırının bir koşula bağlı olarak çalıştıran kontrol ifadeleri, mantıksal ve matematiksel işlemlerin yapılması için kullanılan operatörlerden ve programın çalıştırılmasında varsayılan olarak bulunan *loop()* ve *setup()* bloklarından oluşur. Değişkenler, program çalıştırıldığında istenilen değerlerin atandığı veri tutucularıdır. Fonksiyonlar ise bir program parçasını tekrar tekrar yazmamak için kullanılan yapılardır.

3.1.3.1. AnalogWrite() fonksiyonu

Arduino kartından 0 volt ile 5 volt arasında analog bir değer üretilmesini sağlayarak DC motorun dönüş hızını ayarlamak veya ledin parlaklığını artırıp azaltmak gibi durumlarda kullanılan fonksiyondur. Elde edilen gerilim PWM kare dalga şekildedir. *AnalogWrite()* fonksiyonu PWM pinleri ile kullanılabilir.

Kullanımı: analogWrite(pin,deger)

Yukarıda yer alan ifadede *pin* değişkeni PWM çıkışı almak istenilen pin numarasıdır. Kullanılan deger değişkeni ise 0 ile 255 arasında değerler alarak 0-5 volt arasında değer elde etmek için kullanılır. *deger* değişkenine 0 yazdığımızda bir çıkışın olmadığı, 255'e doğru ilerledikçe pinde 0 ile 5 volt arasında gerilim değerinin oluştuğu görülür. *deger* değişkenine 127 girildiğinde 2,5 volt gerilim 255 girildiğinde 5 volt gerilim gönderilir. Bu durumda 127 değeri ile led az parlak yanarken 255 değeri ile led daha parlak yanacaktır. Arduino kartı

üzerinde bulunan her pin *AnalogWrite()* fonksiyonu ile kullanılmaz. PWM pinleri *AnalogWrite()* fonksiyonu ile kullanılabilir. PWM pinleri Arduino kart üzerinde "~" simgesi ile gösterilmektedir. Arduino Uno üzerinde "~" simgesinin bulunduğu 6 adet PWM pini bulunmaktadır. Bu pinlere *analogWrite()* işlevi kullanılarak ara değerler gönderilebilir. Bir ledin parlaklık ayarının yapılması veya bir motorun çalışma hızının ayarlanması gibi uygulamalarda kullanılabilir.

3.1.3.2. AnalogRead() fonksiyonu

AnalogRead() fonksiyonu belirlenen pin üzerindeki analog sinyalin okunmasını sağlar. Arduino kartlar üzerinde 10 bit çözünürlüğünde ADC (Analog digital converter) bulunur ve bu dönüştürücü ile 10 bit çözünürlüğünde dönüştürme işlemi gerçekleştirilir. Arduino üzerindeki analog pin girişlerine 0 volt ile 5 volt arasında gerilim uygulanarak 0,0049 volt hassasiyet ile analog sinyal okunabilir. Sensör veya potansiyometreden okunan değerin led veya motorun çalıştırılması için gönderilirken okunan değerin yaklaşık 4'e bölünmesi gerekir. Bunun nedeni analog pinler 10 bitlik 1024'e kadar değer alabilirken dijital pinler 8 bitlik 255'e kadar değer alabilir. Analog sinyallerin okunması dijital sinyallerin okunmasından daha zor olduğu için okuma süresi dijitale göre daha yavaş olur. *AnalogRead*() fonksiyonu ile değer okunabilmesi için potansiyometre, sensör gibi bir donanım elemanı gereklidir. Arduino Uno kartında A0-A6 aralığında 6 adet analog pin bulunmaktadır.

AnalogRead() fonfsiyonu ile elde edilecek gerilim 0-5 volt arasındadır. Bu gerilim 5 volttan daha düşük bir gerilime ayarlanabilir. Bu ayarlama için Arduino kartının Aref pini kullanılır. Analog pinlerin giriş işlerinde kullanılan referans gerilimi *analogReferance*() fonksiyonu ile ayarlanır. Bu işlem sayesinde daha hassas ölçümler yapılabilir. AnalogRead fonksiyonu ile potansiyometre, LDR ışık sensörü gibi donanım birimlerinden değer okunur ve okunan değere göre 0-5 volt aralığında gerilim elde edilir. (Köse, 2014).

3.1.3.3. DigitalWrite() fonksiyonu

DigitalWrite() fonksiyonu çıkış olarak tanımlanan pine güç vermek için veya verilen gücü kesmek için kullanılır. Bu işlev iki parametre değeri alır. İlk parametre hangi pin numarasının kullanıldığını, ikinci parametre ise gerilimin durumunu belirlemek için kullanılır.

Kullanımı: digitalWrite(pinno, deger)

Yukarıda yazılan kodda ikinci parametreye HIGH değeri verilirse, Arduino kartının referans gerilim değeri belirtilen pinden çıkış gerilimi olarak ayarlanır. Referans geriliminde değişiklik

yapılmamış ise çıkış gerilimi 5 volt olacaktır. İkinci parametreye HIGH değeri yerine "1" veya True değerleri de kullanılabilir.

Yazılan kodlarda ikinci parametreye LOW değeri girilirse, Arduino kartında belirtilen pinde çıkış gerilimi olarak 0 volt verilerek güç kesilmiş olacaktır. İkinci parametreye LOW değeri yerine 0 veya *false* değerleri de kullanılabilir. LOW değeri pini 0 volt seviyesine çekerek gücü keser, HIGH değeri ise gerilimi 5 volt'a yükseltir. Ancak bu değer Arduino' nun bazı çeşitleri 3.3 volt ile çalıştığı için çıkışta 3.3 volt alınır.

3.1.3.4. DigitalRead() fonksiyonu

Dijital giriş olarak tanımlanan *digitalRead*() fonksiyonu belirtilen pindeki giriş değerine göre TRUE veya FALSE değeri döndürür. TRUE değeri dönerse 5 volt FALSE değeri dönerse 0 volt değeri elde edilir. Arduino kartına bağlı buton devre elemanını kullanılarak *digitalRead*() fonksiyonu ile değer okunabilir. Butona basıldığında *digitalRead*() fonksiyonu bize HIGH değeri döndürerek ledin yanması sağlanır. Butona basılı değilken LOW değeri döndürülerek ledin sönmesi sağlanır.

Kullanımı**:** digitalRead(deger)

Yukarıda yer alan değer değişkeni pin numarasıdır. Geriye dönen değer okunan bilgi için HIGH veya LOW olarak iki değer elde edilebilir.

3.1.3.5. PinMode() fonksiyonu

PinMode() fonksiyonu belirlenen pinin giriş veya çıkış işlemleri için kullanılmasına karar vermeyi sağlar. İki parametre değeri alır. İlk parametresi kullanılan pin numarasının hangisi olduğunu belirlerken, ikinci parametresi ise INPUT, OUTPUT**,** INPUT_PULLUP değerlerinden birini alarak pinin modunu belirlemek için kullanılır.

INPUT: Belirlenen pinin giriş olarak ayarlanması sağlanır.

OUTPUT: Belirlenen pinin çıkış olarak ayarlanması sağlanır.

INPUT_PULLUP, Belirlenen pinin giriş olarak ayarlanması sağlanarak dahili pull-up direncinin aktif edilmesi sağlanır. Böylece devre elemanlarına harici direnç bağlanmasına gerek kalmaz.

3.1.3.6. Delay() fonksiyonu

Delay() fonksiyonu programın kullanıldığı satırında belirlenen süre kadar beklenilmesini sağlar. Fonksiyonun içerisine yazılan değerin birimi milisaniyedir. (Yalçın, 2015).

Örnek kullanım**:** Delay(2000); # 2000 milisaniye yani 2 saniye bekler.

delayMicrosecond() fonksiyonu *delay()* fonksiyonunda olduğu gibi programın kullanıldığı satırında belirlenen süre kadar beklenilmesini sağlar. Fonksiyonun içerisine yazılan değerin birimi mikrosaniyedir.

Örnek: delayMicrosecond (20000); # 20000 mikrosaniye bekler.

3.2. Python Programlama Dili

Python, nesne yönelimli, yorumlamalı ve etkileşimli yüksek seviyeli bir programlama dilidir. Geliştirilmeye 1990 yılında Guido Van Rossum tarafından Amsterdam'da başlanmıştır. Unix, Linux, Mac, Windows, gibi hemen hemen her türlü platformda çalışabilir. Python ile sistem programlama, robotik kodlama, ağ programlama, web programlama ve görüntü işleme gibi birçok alanda yazılım geliştirilebilir.

Raspberry Pi mini bilgisayarı ile hareketli görüntülerin elde edilmesi, istenilen rengin tespit edilmesi, belirlenen nesnenin tespit edilmesi, araç üzerinde yer alan plakanın tespit edilmesi gibi görüntü işleme işlemlerinde Python programı kullanılmıştır.

3.2.1. Python editörleri

IDE, programcıların kodlama yapabilmesi için oluşturulmuş grafik arayüzü sunan bir yazılımdır. IDE içerisinde, kaynak kod düzenleyicisi oluşturma araçları ve bir hata ayıklayıcı bulunur. Python programında kod yazmak için Idle, Thony, Spyder, Pycharm gibi birçok IDE yazılımı bulunmaktadır.

Idle Python kurulduğunda varsayılan olarak gelen açık kaynak kodlu bir yazılımdır. PyCharm, özellikle Python programlama dili için kullanılan ve en çok tercih edilen geliştirme ortamlarından birisidir. PyCharm Python yazılımları geliştirmek için tek başına yeterli değildir, daha önceden Python'un bilgisayarda kurulu olması gerekir. Pycharm programının hem ücretli hem de ücretsiz sürümü bulunmaktadır. PyCharm Community Edition sürümü ücretsiz iken Professional Edition sürümü ise ücretlidir. Anaconda Python programlama dilini kullanmak isteyenler için hazırlanmış tümleşik bir Python dağıtımıdır. Veri bilimi, yapay zeka gibi konularda sıkça kullanılan kütüphanelerin yanı sıra Jupiter Notebook ve

Spyder gibi araçları da barındırır. Spyder, Python geliştirme için kullanabileceğiniz, en çok tercih edilen açık kaynak kodlu yazılımlardan birisidir. Yapılan çalışmada Python dağıtımı olan Anaconda programının içerisinde yer alan araçlardan açık kaynak kodlu olan Spyder IDE'si ve Raspberry Pi mini bilgisayarına kurulan Rasbian işletim sistemi içerisinde bulunan Thonny IDE'si kullanılmıştır.

4. GÖRÜNTÜ İŞLEME

Görüntü işleme, görüntünün sayısallaştırılarak dijital hale getirilip görüntüden anlamlı bilgiler çıkarılmasını sağlayan işlemler bütünüdür. Bu işlemler, görüntüyü oluşturan pikseller üzerinde matematiksel işlemler yapılarak gerçekleştirilir. Görüntü işleme teknikleri kullanılarak görüntü üzerinde bulunan gürültülerin yok edilerek temiz bir görüntü elde edilmesi, görüntü üzerinde bulunan ve insan algısının görmekte zorlandığı nesnelerin tespiti, görüntü üzerindeki farklı nesnelerin birbirinden ayırt edilmesi gibi işlemler yapılır. Görüntü işleme teknikleri ile bir görüntüden faydalı bir bilgi çıkarılarak yorumlanması sağlanır. İşlenecek görüntüler, fotoğraf makinası, tarayıcı ve kameralar yardımıyla elde edilebilir.

Kamera ve fotoğraf makinası üzerinden alınan görüntülerin işlenerek yararlı bilgilerin elde edilmesi için bazı işlemlerin yapılması gerekmektedir. Bu konuda geliştirilmiş birçok kütüphane bulunmaktadır. Bu kütüphanelerden OpenCV açık kaynak kodlu ve çok tercih edilen bir kütüphanedir. Yapılan çalışmada Python programı içerisinde OpenCV kütüphanesi kullanılarak görüntü işleme ile hareketli görüntülerin elde edilmesi, resim üzerindeki karakterlerin okunması, istenilen rengin tespit edilmesi ve Template Matching (Şablon Eşleştirme) yöntemi ile nesne tespiti yapılmıştır (Samtaş ve Gülesin, 2012:84-93).

4.1. Kullanılan Görüntü İşleme Komutları

Görüntü işleme matrisler üzerinde yapılan işlemler bütünüdür. Matrisler üzerinde işlem yapabilmek için görüntünün çalışma ortamına aktarılıp alınan görüntünün işlenerek analiz edilmesi ve görüntülenmesi ile görüntü işlemenin temel adımları gerçekleştirilir. Bu şekilde görüntü işleme ile görüntüden anlamlı bilgiler çıkarmak için programlama dillerinin dahili metodlarından ve kütüphanelerden yararlanılır. OpenCV kütüphanesi Python, C, C++, Java, Matlab programlama dilleri ile kullanılabilmektedir. Aşağıdaki alt başlıklarda, bu çalışmada kullanılan OpenCV fonksiyonları kısaca tanıtılmıştır.

4.1.1.VideoCapture() fonksiyonu

Görüntü işleme çalışmaları daha önceden elde edilmiş kaydedilen görüntüler üzerinde yapıldığı gibi canlı olarak elde edilen anlık kamera görüntüleri üzerinde de yapılabilir. Bu şekilde daha önceden kaydedilen görüntüler ve kamera ile elde edilen anlık görüntüler üzerinde çeşitli görüntü işleme yöntemleri uygulanarak analiz yapılabilmektedir.

VideoCapture() fonksiyonu kameradan görüntü almak veya elimizde var olan herhangi bir video kaydını okumak için kullanılır. Bu işlemin yapılabilmesi için OpenCV

kütüphanesi eklenmelidir. Bu fonksiyon ile dahili web kamerasından veya harici web kamerasından görüntü okunabilir. Kameranın haricinde daha önceden elde edilmiş video görüntüleri de okunabilir (Aydın, 2020).

Kullanımı: kamera = cv2.VideoCapture(0)

Yukarıda yer alan komut ile kameradan görüntü alma işlemi yapılır. *VideoCapture*() fonksiyonun içerisinde yer alan parametre değeri ile birden fazla kamera olması durumunda hangi kameradan görüntünün okunacağını belirlemek için kullanılır. Bilgisayarın kamerasını kullanmak için 0, harici kamerayı kullanmak için ise 1 değerleri kullanılır. Birden fazla kamera bağlantısı var ise bu sayıyı artırarak sırasıyla kullanılır.

Kullanımı: okunanVideo = cv2.VideoCapture('adana.mp4')

Yukarıda yer alan komut ile elimizde var olan adana.mp4 dosyası okunur. Video dosyasının sadece adını belirterek okunması için Python programında yazılan kod ile video dosyamızın aynı klasörde olması gerekir. Aynı klasörde değilse dosyanın yoluda belirtilmelidir.

4.1.2. CreateBackgroundSubtractorMOG2() fonksiyonu

CreateBackgroundSubtractorMOG2() arka planı çıkararak sabit bir zemin üzerindeki hareketli nesneleri yakalamak ve takip etmek için kullanılır. Bu fonksiyon ile arka plan yani sabit kalan kısımlar tespit edilir ve üzerindeki değişiklikler tespit edilerek hareketli görüntüler elde edilmiş olur. Şekil 4.1'de hareketli görüntülerin tespit edilmiş hali görülmektedir (Pişkin, 2017).

Şekil 4. 1. Hareketli görüntülerin belirlenerek arka planın çıkarılması örneği

4.1.3. FindContours() fonksiyonu

Konturlar, aynı renk veya yoğunluğa sahip tüm sürekli noktaları birleştirerek sınır boyunca bir eğri elde etmek için kullanılır. Bu şekilde konturlar sayesinde nesnenin sınır bölgeleri elde edilerek, aynı yoğunluktaki bir şeklin sınırlarının elde edilmesiyle şekil analizi, nesne algılama ve tanıma gibi işlemler yapılabilir. Konturların bulunması için *findContours*()

fonksiyonu kullanılır. Konturların elde edilmesinde daha iyi sonuç elde etmek için ikili yani siyah-beyaz görüntüler kullanılmalıdır. Bu nedenle, konturları bulmadan önce eşik veya keskin kenar algılama işlemi uygulanır. OpenCV'de kontur bulmak, siyah arka plandan beyaz nesne bulmak gibidir. Bu durumda bulunan hareketli nesneler beyaz ve hareket etmeyen nesneler ise siyah olacaktır. Şekil 4.2'de konturların elde edilmesine ait kodlar görülmektedir (Macit, 2020).

```
import cv2
import numpy as np
image = cv2.imread('kare2.png')
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
ret, thresh = cv2.threshold(gray, 127, 255, 0)
contours, hierarchy = cv2.findContours(thresh,cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
resim1=cv2.drawContours(image, contours, -1, (0, 255, 0), 3)
```

Şekil 4. 2. Konturların elde edilmesi için kullanılan kodlar

Konturların bulunması için giriş olarak verilen görüntünün ikili görüntüye çevrilmesi gerekir. Görüntüleri ikili görüntüye dönüştürmek için *threshold*() fonksiyonu kullanılır. İkili görüntü, görüntünün siyah ve beyaz olarak tanımlanmasıdır. Elde edilen konturların çizilmesi için *drawContours*() fonksiyonu kullanılır. Bu fonksiyonun kullanımın'da ilk argüman kaynak görüntü, ikinci argüman görüntüdeki konturların Python listesi, üçüncü argüman konturların indeksi, dördüncü argüman çizimin rengi ve beşinci argüman çizimin kalınlığıdır.

4.1.4. Treshhold() fonksiyonu

İstenilen görüntüyü ikili görüntüye çevirerek siyah-beyaz görüntünün elde edilmesi için kullanılan bir yöntemdir. Bu fonksiyon genellikle gri tonlu bir görüntüden, siyah ve beyaz renkten oluşan ikili görüntü elde etmek için kullanılır. Renkli görüntülerde çok fazla tercih edilmese de kullanılabilir. Bu şekilde kaynak görüntünün piksel değerleri belirlenen eşik değerine göre filtrelenmesi sağlanır.

Eşikleme işlemi morfolojik operatörler gibi görüntü üzerinde yer alan gürültülerin azaltılması veya nesnelerin belirlenmesini sağlamak gibi farklı amaçlar için kullanılır. Belirlenen görüntü üzerinde uygulanan eşikleme işleminin tipine bağlı olarak, belirlenen eşik değerine göre gri resme dönüştürülen renkli görüntüyü siyah ya da beyaz olarak günceller. Görüntü gri tona dönüştürüldükten sonra ikili görüntü elde edilebilir.

OpenCV içerisindeki sık kullanılan eşikleme tipleri aşağıda verilmiştir.

- THRESH_BINARY
- THRESH_BINARY_INV
- THRESH_TRUNC
- THRESH_TOZERO
- THRESH_TOZERO_INV

THRESH_BINARY eşiklemede kaynak olarak alınan gri tonlu görüntü üzerinde yer alan piksel değeri, belirlenen eşik değerinden büyükse maksimum değer olarak belirlenen değere atanır. THRESH_BINARY_INV tipinde ise kaynak olarak alınan gri tonlu görüntü üzerinde yer alan piksel değeri, belirlenen eşik değerinden küçükse maksimum değer olarak belirlenen değere atanır. THRESH_BINARY_INV eşikleme tipi THRESH_BINARY tipinin tersi olarak kullanılır. THRESH_TRUNCH tipinde kaynak olarak alınan gri tonlu görüntü üzerinde yer alan piksel değeri, belirlenen eşik değerinden büyükse görüntünün piksel değeri eşik değerine eşit olacak, eşik değerinden küçük olması durumunda ise piksel değeri değişmeyecektir.THRESH_TOZERO tipinde kaynak olarak alınan gri tonlu görüntü üzerinde yer alan piksel değeri, belirlenen eşik değerinden büyükse görüntünün piksel değerinde bir değişiklik olmayacak, eşik değerinden küçük olması durumunda ise piksel değeri siyah olarak değiştirilecektir. THRESH_TOZERO_INV tipinde kaynak olarak alınan gri tonlu görüntü üzerinde yer alan piksel değeri, belirlenen eşik değerinden küçükse görüntünün piksel değerinde bir değişiklik olmayacak, eşik değerinden büyük olması durumunda ise piksel değeri siyah olarak değiştirilecektir. (OpenCV, 2018).

4.1.5. CvtColor() fonksiyonu

Herhangi bir renk uzayından istenilen bir renk uzayına dönüşüm yapmak için *cvtColor()* fonksiyonu kullanılır. Renk uzayları birbirine dönüştürülürken görüntü kayıpları oluşmaktadır. Bu görüntü kayıpları renk uzayları arasında farklılık göstermektedir. Kameradan elde ettiğimiz RGB formatındaki görüntüde istenilen rengin tespit edilmesi için HSV renk uzayına dönüştürülmesi, RGB formatındaki görüntülerin yazıcıdan çıktısının alınması için CMYK renk uzayına dönüştürülerek işlem yapılması daha iyi sonuçlar verir. Bu nedenle yapılan işleme göre renk uzayları arasında dönüştürme işlemi yapılır.

Örnek kullanım:

```
img = cv2.imread('fare.jpg')
```

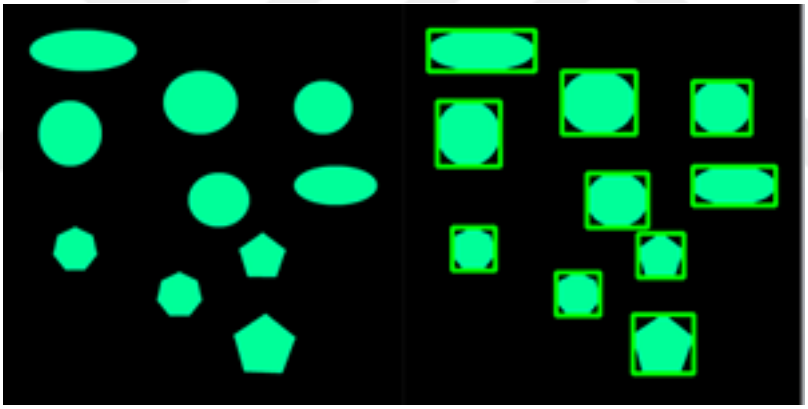
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

hsvrenk = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

Yukarıda yer alan kodlarda renkli görüntünün gri tonlu bir resme ve HSV renk uzayına dönüştürülmesi işlemi yapılmaktadır.

4.1.6. BoundingRect fonksiyonu()

Bir görüntü içerisinde oluşturulan konturların elde edildiği bölgeyi vurgulamak ve ikili olarak elde edilen nesnenin çevresine çizgi çizilmesini sağlamak için kullanılır. Çizilen bu çizgilerle dikdörtgen şekli oluşturulur. Bu işlev ile oluşturulacak dikdörtgenin sol üst köşesinin koordinatları ile dikdörtgenin genişliği ve yüksekliği elde edilir. *boundingRect*() işlevi ile elde edilen değerlere göre *rectangle*() işlevi ile çizgiler çizilir (OpenCV, 2021). Şekil 4.3'de nesnenin sınırlarının *boundingRect()* fonksiyonu ile belirlenerek sınırları belirlenen nesnenin *rectangle()* fonksiyonu ile çerçevesinin çizilmiş hali görülmektedir.



Şekil 4. 3. Nesnelerin çerçeve ile vurgulanması

4.1.7. Resim okuma yazma komutları

Resimleri dosyadan okumak için *imread*() fonksiyonunu kullanılır. Bu fonksiyon ile belirlenen resmin, adı ve uzantısını yazarak okuma yapılır. Okuması yapılacak görüntü ile Python programında yazılan program aynı ortamda bulunduğunda dosya yolunun belirtilmesine gerek yoktur. Resim okuma işleminde *imread*() fonksiyonuna sıfır değeri gönderildiğinde resim gri tonda elde edilir.

Okutulan resmin ekranda görüntülenmesi için *imshow*() fonksiyonu kullanılır. Fonksiyona iki parametre değeri girilerek kullanılır. Bu parametrelerden biri açılan pencerenin adı diğeri ise resmin atandığı değişkendir. Resim okuduktan sonra açılır ve çok

kısa bir süre sonra kendiliğinden kapanır. Pencere açıldıktan sonra kendiliğinden kapanmadan sabit kalması ve herhangi bir tuşa basılana kadar pencerenin kapanmaması için *waitKey*() ve *destroyAllWindows*() fonksiyonları kullanılır.

cv2.*waitKey*() fonksiyonu programın klavye ile kontrol edilmesini sağlar. Bu fonksiyon herhangi bir klavye olayı için belirtilen milisaniye kadar bekler. O süre zarfında bir tuşa basarsanız program bir sonraki işleme geçerek devam eder. Belirlenen süre içerisinde bir tuşa basılmazsa süre tamamlandıktan sonra program bir sonraki işlemden devam eder. Eğer fonksiyona sıfır değeri girilirse bir tuşa basılmadığı sürece sonsuza kadar bekler. Oluşturulan bütün pencerelerin kapanması için cv2.*destroyAllWindows*() fonksiyonu kullanılır. Sadece İstenilen pencerenin kapanması isteniyorsa pencere adı parantez içine yazılmalıdır.

Imwrite() fonksiyonu ile elde edilen görüntüleri dosya olarak istenilen yere kaydetmek için kullanılır. Bu fonksiyon resimleri kaydetmek için iki parametre kullanır. Bu parametreler dosya adı ve görüntüyü temsil eden değişken ismidir. Bu şekilde güvenlik amacıyla elde edilen görüntülerin istenildiği zaman ulaşılması için kaydedilmesi sağlanır. Ayrıca elde edilen bu görüntüler internetten istenilen yere gönderilmesi işleminin sağlanması için de kullanılır. Resim okuma ve yazma işlemlerine ait kodlar Şekil 4.8'de görülmektedir.

```
import cv2
resim = cv2.imread("bayrak.jpg") ; resim1 = cv2.imread("bayrak.jpg",0)
cv2.imshow("Renkli resim",resim) ;  cv2.imshow("Gri tonlu resim",resim1)
cv2.imwrite('kaydedilenresim.png' , resim1)
cv2.waitKey(0) # Bir tuşa basılana kadar bekler
cv2.destroyAllWindows()
```

Şekil 4. 4. Resim okuma yazma için kullanılan kod kümesi

4.1.8. InRange() fonksiyonu

Bu foksiyon istenilen rengin tespit edilmesi için kullanılır. İstenilen rengin elde edilmesi fonksiyona belirlenen rengin taban ve tavan değerlerinin girilmesiyle bulunur. İstenilen rengin tespit edilmesi için doğru renk uzayı seçilmeli ve renk verilerine eşikleme uygulanmalıdır. Renk verisi için iki eşik değeri kullanılır. *inRange*() fonksiyonu ile belirlenen iki eşik değer maskeleme yapılır. Yapılan maskeleme sonucunda iki eşik değeri arasında kalan yerler beyaz diğer yerler ise siyah olur. İstenilen renge ait nesne tespit edildikten sonra bazı işlemler yapılarak konumu, alanı gibi özellikleri hakkında bilgi sahibi olunması sağlanır. Örneğin kontur işlemleri kullanılarak rengi bulunan nesnenin etrafını tam olarak kaplayan

dikdörtgen veya daire şekli bulunabilir. Ayrıca konturlar ile istenilen renkteki nesne sayısı, nesnenin alanı bulunabilir.

Bilgisayarda görüntü işleme uygulamasında istenilen rengin tespiti için HSV renk uzayı kullanıldığında daha iyi sonuçlar elde edilir (Bosnali, 2019). Şekil 4.5'de RGB görüntü formatındaki resim HSV renk uzayına dönüştürülüp elde edilen HSV görüntüsüne *inRange*() fonksiyonu uygulanarak kırmızı renkler elde edilmiş hali gösterilmiştir.

Şekil 4. 5. Kırmızı renk tespiti

4.1.9. NumPy dizileri

NumPy, Python programlama dili ile kullanılan dizi ve matrisleri kullanarak ileri düzey matematiksel işlemler ve bilimsel hesaplamalar yapılabilen açık kaynak kodlu bir kütüphanedir.

Python programlama dilinde NumPy dizilerinin yerine listeler kullanılabilir. NumPy dizilerinin tercih edilmesinin en büyük nedenlerinden birisi listelere göre çok daha hızlı olmalarıdır. Yapılan çalışmada istenilen rengin tespit edilmesinde belirlenen rengin alt ve üst değerlerini belirlemede NumPy dizileri kullanılmıştır. Veri bilimi, makine öğrenimi, derin öğrenme gibi alanlarda sıklıkla kullanılmaktadır. (Ekren, 2020). Aşağıda NumPy işlem fonksiyonları listelenmiştir.

Ndim: Dizinin kaç boyutlu olduğunu bulmak için kullanılır.

Shape: Numpy dizisinin satır ve sütun sayısını bulmak için kullanılır.

Arange: Başlangıç ve bitiş değerlerinin belirtilerek her defasında kaçar kaçar artırılacağı belirtilip başlangıç değerinden başlayarak bitiş değerine kadar olan sayıların bulunmasını sağlayan Numpy dizisi oluşturmak için kullanılır

Linspace: Başlangıç ve bitiş değerleri belirtilerek eşit aralıklarda sayılar elde etmek için kullanılır.

Size: Dizinin eleman sayısını bulmak için kullanılır.

Reshape: Dizinin boyutunu değiştirmek için kullanılır.

4.2. Renk Uzayları

Renkleri tanımlamak için kullanılan matematiksel modellerdir. Renk çeşitliliğinin fazla olması nedeniyle renkleri gruplandırmak ve standartlaştırmak için renk uzayları kullanılır. Renk, ışığın değişik dalga boyları ile nesnelere çarparak gözümüze yansıması sonucu ortaya çıkan bir algılamadır. Bu algılama, ışığın nesneler üzerine çarpması ile bir bölümünün soğurulup bir bölümünün yansıması nedeniyle çeşitlilik göstermesi renk tonu veya renk olarak isimlendirilir. Tüm dalga boylarının tamamı gözümüze ulaşırsa bunu beyaz renk olarak, hiçbir dalga boyu gözümüze ulaşmazsa siyah renk olarak algılanır.

Renkler ışığın nesnelere çarparak gözümüze yansıma yapması sonucu oluşur. Güneşli günlerde renklerin daha parlak ve daha canlı olması, kapalı bir havada ise renklerin parlaklığının azalması ile daha koyu görünmeleri, renklerin ışığa bağlı olarak değiştiğini gösterir. Işığın olmadığı yerde nesneler ve renkler karanlıkta görünmez olur. Nesne üzerine yansıyan ışığın dalgaboylarının bir bölümü emilir bir bölümü yansıyarak parlar. Doğadaki nesnelerin açık renkli veya koyu renkli görünmeleri tutmuş oldukları ışığın azlığı veya çokluğuna bağlıdır. Örneğin yeşil olan nesnenin bu renkte görünmesinin nedeni, o nesnenin güneş ışığından sadece yeşil olan ışığı yansıtması diğer renkleri soğurmasıdır.(Güngör, 2019).

Görüntüler için farklı renk uzayları kullanılmaktadır. Gri tonlamalı bir görüntüde her piksel için 8 bit yani bir byte'lık alan gerekir . Gri tonlu bir resimde her piksel için 8 bitlik alanda 0 ile 255 arasında bir değer depolanır ve bu aralıktaki değerler ile gri rengin tüm tonları elde edilir. Böylece, gri tonlamalı bir görüntü iki boyutlu bir dizi kullanılarak oluşturulur. Dizinin boyutu görüntü için kullanılan matrisin satır ve sütun sayısına eşittir. Elde edilen bu diziyle tek kanallı bir görüntü oluşturulmuş olur. Gri tonlamalı bir görüntü sadece bir kanala sahip olmakta ve bu kanal beyazların yoğunluğunu göstermektedir. Görüntüye renk eklendiğinde görüntünün her bir pikseli için bellekte 3 byte yani 24 bitlik bir alana ihtiyaç vardır. Renkli görüntüde 24 bitlik bir bilgi ile yaklaşık 16,7 milyon renk elde edilir. Bu renkler görüntülerin işlenmesi için sınıflandırılıp numaralandırılarak düzenlenmesi ile renk uzayları oluşturulur.

4.2.1. RGB renk uzayı

19. yüzyılda geliştirilen RGB renk modeli üç rengin referans alınarak tüm renklerin oluşturulduğu bir renk uzayıdır. Referans olarak alınan bu üç renk Red (kırmızı), Green (yeşil) ve Blue (mavi) renklerinden oluşmaktadır. RGB renk uzayı red, green, blue

kelimelerinin baş harflerinden oluşan bir kısaltmadır. RGB, renkli görüntüler elde edebilmek için kullanılan standart renk uzaylarından birisidir. RGB renkler ışığa bağlı olduklarından televizyon, bilgisayar gibi dijital ekranlarda renkli görüntü elde etmek için kullanılmaktadır. Dijital ekranlarda görüntülenecek bir tasarımda RGB renkleri seçildiğinde daha canlı görüntüler elde edilir.

RGB renk modeli kırmızı, yeşil ve mavi renkten oluşmuş olsa da söz konusu renklerin birbirleriyle kombinasyonları ile milyonlarca farklı renk de elde edilir. RGB renklerin her bir kanalına tam yoğunlukla 255 değerleri verilerek birbiri ile karıştırıldığında beyaz, bu üç rengin her bir kanalına sıfır değerinin verilmesi durumunda siyah renk elde edilir (Bengu, 2020).

RGB görüntülerde renklerin oluşumunu sağlamak için üç renk yani üç kanal kullanılır. Kanal başına 8 bitlik bir bilgi ile görüntülerde her bir piksel için 24 bitlik (8 bit x 3 kanal) renk bilgisi elde edilir. Piksel başına üç kanalın kullanıldığı 24 bitlik görüntülerle yaklaşık 16,7 milyon farklı renk üretebilir. RGB modelinde üç renk bileşeninin değerlerinin aynı olduğu durumlarda gri tonlu bir renk elde edilir. Bu renk uzayında her bir kanalın değerinin 255 olduğu durumda beyaz renk elde edilir, her bir kanalın değerinin sıfır olduğu durumda siyah renk elde edilir.

4.2.2. HSV renk uzayı

HSV, renklerin renk özü, doygunluk ve parlaklık değerlerine göre ifade edilmesiyle oluşturulan renk uzayıdır. RGB'de renklerin karışımı kullanılmasına karşın HSV'de renk özü, doygunluk ve parlaklık değerleri kullanılır.

Renk özü (Hue), rengin mavi, sarı ve yeşil gibi baskın dalga boyunu belirlemek için kullanılır. Doygunluk (Saturation), rengin canlılığını belirlemek için kullanılır, doygunluk değerinin yüksek olması canlı renklerin elde edilmesine neden olurken, düşük olması gri tonlu renklerin elde edilmesine neden olur. Parlaklık (Value) ise rengin aydınlığını yani içinde bulunan beyaz oranını belirlemek için kullanılır. HSV renk uzayında parlaklık değeri sıfır girildiğinde siyah renk elde edilir. Beyaz rengin elde edilebilmesi için parlaklık değeri 255 girilmelidir.

Bilgisayarlı görme ile görüntü işleme uygulamalarında belirli bir renkteki nesnenin tespit edilmesi istenildiğinde HSV renk uzayının kullanılması daha iyi sonuçlar verir. HSV renk modeli kullanılarak sadece renk özü değerinin kullanılmasıyla eşik değer uygulanarak

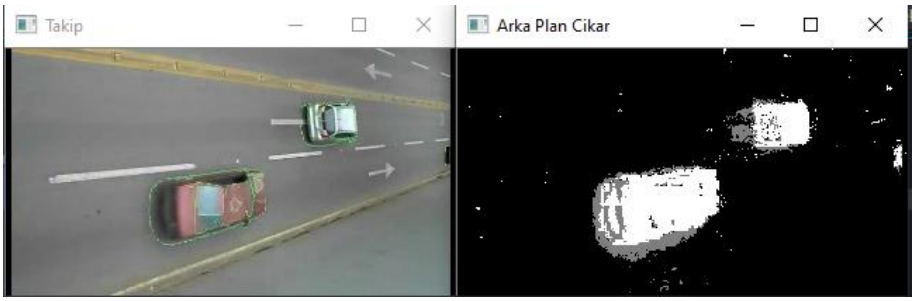
istenilen renklerin belirlenmesi sağlanabilir. OpenCV kütüphanesinde renk özü değerleri 0-179 arasında, doygunluk ve parlaklık ise 0-255 arası değerler alır.

4.3. Hareketli Görüntülerin Elde Edilmesi

İnsanların güvenlik istekleri gün geçtikçe artmakta, dünyanın birçok yerinde hırsızlık gibi olumsuz durumlar için ev, işyeri, okul gibi yerlerde güvenliğin sağlanması ihtiyaç haline gelmiştir. Güvenlik sistemlerinde bulunması gereken en temel bileşen insan, köpek gibi bir canlı tarafından veya elektronik devre elemanları ile gözetleme yapılmasıdır. Elektronik devre elemanlarıyla insanın olmadığı zamanlarda görüntü işleme teknikleri kullanılarak istenilen bölge gözetlenebilmekte ve hareketli nesneler tespit edilerek takibi yapılabilmektedir.

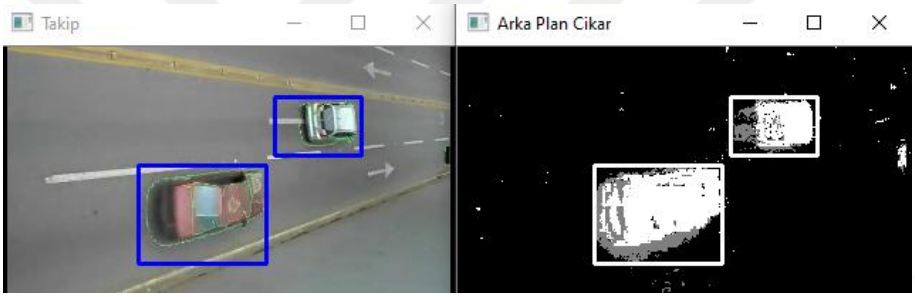
Sabit bir zemin üzerinde bulunan insan, hayvan, araç gibi hareketli görüntülerin elde edilerek takibinin yapılması için görüntü üzerinden hareketsiz görüntüler çıkarılır. Görüntü üzerinde hareketsiz bulunan arka plan çıkarılarak hareketli görüntülerin elde edilmesi sağlanmaktadır. Görüntü işleme algoritmaları görüntünün sabit kalan arka plan kısmını tespit ederek hareketli görüntüleri yakalayabilir. OpenCV'de arka plan çıkarma yöntemlerinden birisi *absdiff()* metodunu kullanmaktır. *absdiff()* metodu ile farklı zamanlarda elde edilen iki görüntünün matris değerlerinin birbirinden çıkarılarak elde edilen işlemin sonucunda hareketli kısımlar bulunur.

createBackgroundSubtractorMOG2() fonksiyonu, görüntü üzerinde yer alan arka planı çıkararak sabit bir zemin üzerinde yer alan hareketli görüntüleri bulmak ve takibini yapmak için kullanılır. Bu işlev ile arka plan yani sabit kalan kısımlar tespit edilir ve üzerindeki değişiklikler tespit edilerek hareketli görüntüler elde edilmiş olur. Bu işlev ile kameranın sabit olmadığı durumlarda rüzgar gibi nedenlerden dolayı kamera hareket ediyorsa, yağmurlu havalarda ve zeminin çok iyi seçilmediği kalabalık ortamlarda görüntünün arka planının tespit edilmesinin zor olacağı için verimli sonuçlar elde edilemez. Şekil 4.6'da yer alan resimde hareketli görüntülerin elde edilmiş hali görülmektedir.



Şekil 4. 6. Hareketli görüntülerin elde edilmesi

Hareketli görüntülerin seçilerek daha belirgin hale getirmek, etrafına çerçeve çizmek için *boundingRect*() ve *rectangle*() işlevleri kullanılır. Şekil 4.7'de hareketli görüntülerin etrafına çerçeve çizilerek daha belirgin hale getirildiği görülmektedir.



Şekil 4. 7. Hareketli görüntülerin dikdörtgen ile vurgulanması

4.4. OpenCv ile Nesne Tespiti

Görüntüdeki nesnenin renk, şekil gibi öznitelik bilgilerinden veya hareket etmesiyle elde edilen bilgilerden yararlanılarak görüntü işleme teknikleriyle insan, hayvan ve ev eşyaları gibi nesnelerin tespit edilmesi sağlanır. Görüntü işleme komutları kullanılarak tespit edilen nesnenin koordinatları bulunabilir, çerçeve içerisine alınabilir ve nesnenin özellikleri hakkında bilgi elde edilebilir.

OpenCV içerisinde nesneleri tespit etmek ve tanımak için kullanılan birçok algoritma bulunmaktadır. Template Matching (Şablon Eşleştirme) , HAAR Cascade, LBP (Local Binary Pattern), HOG (Histogram of Oriented Gradients) nesnelerin tespit edilmesi için kullanılan algoritmalardan bazılarıdır. Şablon eşleştirme yöntemi dışında kullanılan algoritmalar belirli özelliklerin temel alınmasıyla birbirinden farklı nesnelerin ayırt edilmesi için kullanılan makine öğrenmesi algoritmalarıdır. Bu yöntemler öğretilmiş bir nesnenin tanınması için kullanılır. Sınıflandırıcı ile nesnenin tespit edilmesinde aranan nesnenin

bulunduğu pozitif resimlere ve aranan nesnenin bulunmadığı negatif resimlere ihtiyaç vardır. Pozitif ve negatif resimlerin çerçeve içerisinde taranarak derin öğrenme yöntemleri ile öğretilerek nesnenin tespit edilmesi sağlanır. Şablon eşleştirme yöntemi kullanılarak aranan görüntünün kaynak görüntü üzerinde dolaştırılarak eşleştirme işlemi yapılmasıyla nesne tespit edilir(Pişkin, 2016).

4.4.1. Şablon Eşleştirme yöntemi

Şablon eşleme tekniği, belirlenen bir görüntü içerisinde şablon olarak ifade edilen küçük parçaların bulunması için kullanılan bir yöntemdir. Bulunması istenen şablon görüntüsü kaynak görüntü üzerinde (0,0) koordinatlarına yerleştirilir ve soldan sağa ve yukarıdan aşağıya doğru tüm matris elemanları kontrol edilip her pikselde eşleştirme yapılarak benzerlik olup olmadığı kontrol edilir ve benzerlik oranı oluşturulur. Karşılaştırma sonucu şablona benzeyen görüntünün piksel değeri elde edilir. Benzer şekillerin piksel koordinatları kaydedilerek Template Matching yöntemi ile kaynak görüntü üzerinde aranan şablon kayan pencere yöntemi ile aranır. Elde edilen sonuçlar üzerinde işlemler yapılarak nesnenin elde edilmesi sağlanır (Tuncer, 2019:56-60).

4.4.2. HAAR cascade yöntemi

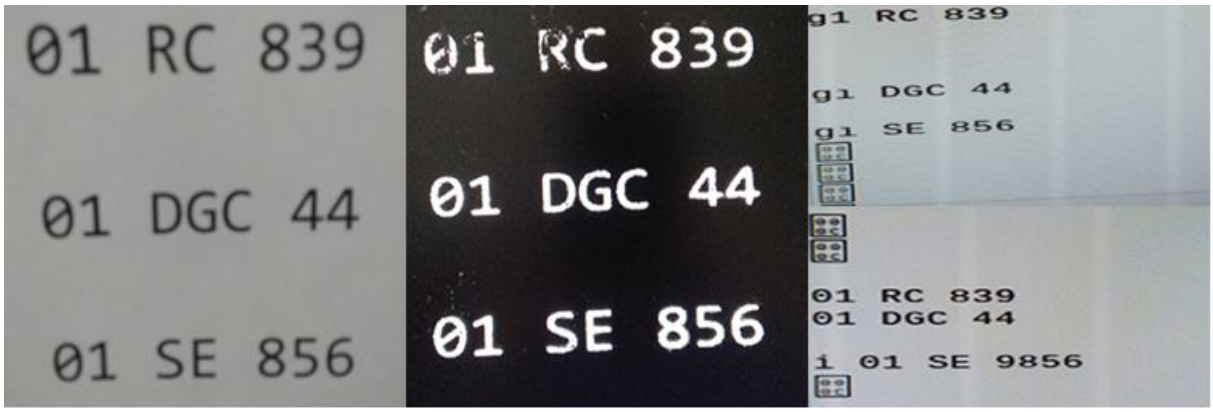
Sınıflandırıcı olarak tanımlanan öğretilmiş nesneleri bulmak için kullanılan bir yöntemdir. Bu yöntemle bulunması istenen nesneler aranan nesnenin bulunduğu pozitif resimler ile aranan nesnenin bulunmadığı negatif resimler bilgisayara tanıtılır ve daha sonra ona benzeyen görüntülerin bulunduğu resimler taranarak nesnenin bulunması sağlanır. Sınıflandırıcı kullanılarak görüntü üzerinde nesne bulmak için kullanılan bir yöntemdir. Bu yöntemle sınıflandırıcı eğitilir ve eğitilmiş görüntülerle nesnenin tespit edilmesi sağlanır. Nesnenin tespit edilmesinde Haar-like özelliği olan kenar, çizgi ve dört kare özellikleri kullanılarak çizgi, yüz, göz, kenar gibi birçok nesne tespit edilebilir.

Nesnenin tespit edilmesi için sisteme tanıtılması ve daha sonra bu tanımlanmış görüntülerin kullanılarak görüntü üzerinde karşılaştırma yapılarak nesnenin tespit edilmesi sağlanır. Haar cascade sınıflandırıcısı xml uzantılı bir dosya içerisinde nesnenin binlerce negatif ve pozitif görüntülerinin hazırlanmış verileri yer almaktadır. Pozitif görüntüler aranan nesnenin bulunduğu negatif görüntüler aranan nesnenin bulunmadığı görüntülerdir. OpenCV içerisinde birçok görüntü eğitilmiş olarak gelir. Haar cascade ile nesne tespitinde örneğin yüz görüntüsüne önden bakıldığında başarılı sonuçlar elde edilmekte iken yandan bakıldığında yüz tanımada başarılı sonuçlar elde edilmemektedir. (Kaplan, 2018:21-25).

4.5. Tesseract ile Yazı Karakteri Tanıma

OCR yöntemi ile optik karakter tanıma, tarayıcı ile taranan belgeler, PDF dosyaları veya dijital kamerayla elde edilen görüntüler gibi resim formatındaki dosyaların piksel piksel incelenerek el ile yazılmış gibi içerisinde bulunan karakterlerin elde edilmesidir. OCR yazılımları ile resim üzerinde yer alan harfler, rakamlar, semboller gibi ifadelerin okunması sağlanır. OCR programının veritabanında her bir karakter için tanımlanmış birçok parametre bulunmaktadır. Görüntü üzerinde yer alan pikseller program içerisinde yer alan parametrelerden birisine uygun ise, karakterin ortaya çıkması sağlanır.

Tesseract, tarayıcı ile taranan belgeler, kamera ile elde edilen görüntüler gibi resim dosyalarının üzerinde yer alan karakteri tanımak için kullanılan bir OCR yazılımıdır. İlk olarak 1985 ile 1995 yılları arasında Hewlett-Packard tarafından kapalı kaynak bir yazılım olarak geliştirilmiştir. 2005 yılında Hewlett Packard ve UNLV tarafından açık kaynak kodlu yazılım olarak kullanılmaya başlanmıştır. Hızlı ve ücretsiz olan bu yazılım Türkçe dil desteği olmakla birlikte birçok dil ile birlikte kullanılabilmektedir. Şekil 4.8'de yer alan resimde Python programı ve Tesseract kullanılarak görüntü üzerinde yer alan plaka bilgilerinin okunması sağlanmıştır (Çelik, 2020:2328-2340)

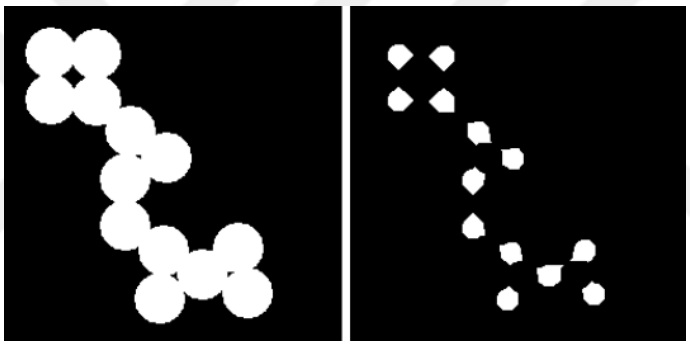


Şekil 4. 8. Resim üzerindeki karakterlerin okunması

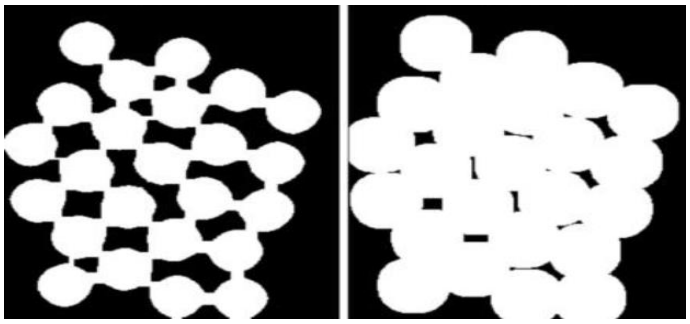
Rasbian işletim sistemi yüklü Raspberry Pi mini bilgisayarında Python programını kullanarak Tesseract ile resim üzerindeki karakterleri okuma işlemi gerçekleştirilerek robot aracın bulunduğu bölgede araçların plakalarının okunması sağlanmış ve plaka bilgileri belirlenen yere kaydedilmiştir. Aynı zamanda plaka bilgilerinin internet üzerinden istenilen yere gönderilmesi sağlanmıştır.

4.6. Morfolojik İşlemler

Morfoloji farklı organizmaların yapısal özelliklerini ve şekillerini inceleyen bilim dalıdır. Belirlenen görüntü üzerinde şekilsel bozukluk olduğu zaman, morfolojik işlemler uygulanarak bu durum giderilmeye çalışılır. Matematiksel morfoloji, siyah-beyaz ve gri tonlu görüntülerden şekilsel bilgi elde edilmesini sağlayan bir araçtır. Görüntü işlendikten sonra, ikili görüntüde istenmeyen lekeler ve parazitler olabilir. Morfolojik işlemler bu gürültüyü görüntüden çıkarmaya yardımcı olur. Morfoloji ile görüntü içerisinden istenilen nesnenin çıkarılması veya ayırt edilmesi sağlanır. Görüntü işlemede temel olarak kullanılan genişletme ve aşındırma olmak üzere iki temel morfolojik işlem vardır. Açma ve kapama işlemleri ise bu iki işlem kullanılarak elde edilir. Şekil 4.9'da morfolojik kapama işlemi, Şekil 4.10'da morfolojik açma işlemine ait resim görülmektedir.



Şekil 4. 9. Morfolojik kapama işlemi



Şekil 4. 10. Morfolojik açma işlemi

Görüntü işlemede kullanılan morfolojik işlemler ile resim üzerinde yer alan şekilleri birbirinden ayırmak ve şekillerin gruplandırılması için kullanılır. Morfolojik işlemler gri tonlu

görüntüler üzerinde kullanılsa da genellikle siyah-beyaz görüntüler üzerinde kullanılır. Genişletme, tekniği ile siyah-beyaz görüntü üzerinde bulunan nesneyi büyütmek veya kalınlaştırarak belirginleştirmek için kullanılan morfolojik işlemdir. Aşındırma tekniği ise, siyah-beyaz görüntü üzerinde bulunan nesneyi küçültmeye veya inceltmeye yarayan morfolojik işlemdir. Aşındırma işlemi ile nesneler küçültülür, küçülen nesneler birbirine bağlı ise küçülerek incelmesi ile birbirinden ayrılmaları sağlanır (Arslan, 2011).

Morfolojik işlemlerde kullanılan açma ve kapama işlemleri genişletme ve aşındırma işlemlerinin kullanılması ile gerçekleştirilir. Açma işlemi, görüntüye genişletme ve aşındırma işleminin sırayla uygulanmasıyla elde edilir. Bu işlemle birbirine yakın iki nesne görüntüde çok fazla bir değişim olmadan birbirinden ayrılması sağlanır. Kapama işlemi aşındırma ve genişletme işleminin sırasıyla uygulanmasıyla elde edilir. Bu şekilde birbirine yakın iki nesnenin görüntüde fazla değişiklik yapılmadan birleştirilmesi sağlanır.

4.7. Görüntülerde Filtreleme İşlemleri

Dijital görüntülerdeki gürültü, görüntünün elde edilmesi esnasındaki hareket, hava durumundan kaynaklanan olumsuzluklar veya resim çekimi esnasında yanlış ışık etkisi, görüntüler üzerinde ortaya çıkan leke, bozukluk gibi istenmeyen işaretlerdir. Bu gürültüler görüntünün kalitesini düşürdüğü gibi görüntüdeki anlamlı kısımların da fark edilmesine engel olabilmektedir. Genellikle görüntü elde edilirken, işlenirken veya iletilmesi sırasında oluşur. Görüntü işleme ile anlamlı bilgiler elde etmek için görüntü üzerinde yer alan gürültünün azaltılması gerekir. Görüntünün oluşumu ve iletilmesi esnasında oluşabilen veri kayıplarının veya bozulmaların azaltması için filtreler kullanılır. Filtreler görüntüleri yumuşatabilir, keskinleştirebilir ve ayrıntıları belirginleştirip, yok edebilirler filtreleme işlemleri genellikle görüntünün piksel değerlerinin uygun başka bir matrisle çarpılması ile gerçekleştirilir. Bu filtreleme işlemi için kullanılan matrislere kernel adı verilir.

Filtreleme işlemleri için kullanılan kernel matrisler genelde 3x3 boyutundadır. Filtreleme işlemi kernel matrislerin görüntü üzerinde gezdirilerek, görüntü matrisi ile kernel matrisin kesiştiği değerlerin çarpımının toplamları ile kernel matris elemanlarının toplamına bölünmesi ile yeni matris değerleri bulunur. Filtreleme işleminde sadece kernel matrisin ortadaki merkez pikselin değeri değiştirilir. Bu işlemler sırayla görüntüdeki tüm matris değerleri için yapılarak filtreleme işlemi gerçekleşir.

Filtreleme yapılırken karşılaşılabilecek sorunlardan birisi filtrenin görüntü dışına taşmasıdır. Görüntü üzerinde gezdirilen kernel matrisin, sınırlara gelmemesi için kenar

pikseller kullanılmaz. Görüntünün tüm matris değerlerine filtre uygulanmak isteniyorsa, kenar piksel değerleri merkez olduğunda, görüntü üzerinde işlenen 3x3'lük kernel matrisin üçüncü sütunu bulunmadığı için, tekrar ikinci sütunu üçüncü sütun yerine yerleştirilerek işlemlere devam edilir. Filtreleme esnasında piksellerin çarpma ve bölme işlemleri sonucunda maksimum veya minimum değerlerden daha büyük veya daha küçük bir değer elde edilmesi durumunda pikselin maksimum veya minimum renk değeri kullanılır. Filtreleme işleminde renkli bir görüntünün kullanılması durumunda, tüm renklerin (kırmızı, yeşil, mavi) ayrı ayrı, filtre matrisi ile işleme girmesi sağlanmalıdır. Yumuşatma filtrelerinin kullanılmasıyla bir piksel ile diğer piksel arasında değişim miktarının azaltılması sağlanır. Keskinleştirme filtrelerinde ise görüntü netleştirilir ve kenarların bulunması sağlanır.

4.7.1. Resim yumuşatma filtreleri

Yumuşatma filtreleri kullanılarak pikseller arasındaki değişim miktarı azaltılır. Filtreleme işlemi, görüntü matrisi ile filtre matrisinin kaydırılarak tüm piksellerle çarpılması ile filtreleme işlemi yapılır. Bu şekilde bir piksel ile diğer pikseller arasındaki değişim miktarı azaltılarak gürültü azaltılmış olur.

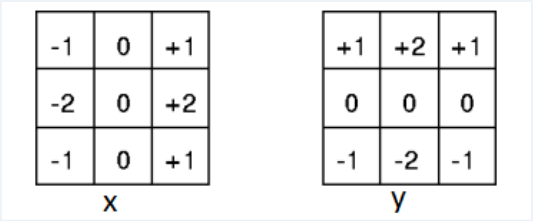
Ortalama filtresi ile bir görüntünün tüm matris değerinin komşularının ve kendisinin dahil olduğu ortalama değer ile değiştirilmesi ile sağlanır. Ortalama filtresi ile filtreler arasındaki değişim miktarı azaltılarak görüntüde yer alan gürültüler azaltılır. Kullanılan filtre matrisinin boyutu artırılırsa yumuşatma oranı da artırılmış olacaktır. Filtre görüntüdeki kenar üzerinde kullanılırsa kenarların her iki tarafında yeni değer oluşacağı için bu işlem kenarın bulanıklaşması nedeniyle keskin kenarın kaybolmasına sebep olabilir.

Medyan filtre, görüntü matrisinin değerlerinin hesaplanması için yakınındaki komşularına bakar. Medyan filtresinde piksel değeri komşu pikselleri sıralayıp sıranın ortasındaki değeri alarak filtreleme işlemi yapılır. Resim üzerindeki detayların kaybolmaması konusunda medyan filtresi mean filtresine göre daha iyi sonuçlar verir.

4.7.2. Kenar bulma filtreleri

Kenar bulma, bir görüntüdeki nesnelerin sınırlarını bulmak için kullanılır ve sınırların bulunmasıyla elde edilen görüntü siyah-beyaz görüntüdür. Kenarların elde edildiği görüntüde arka plan siyah nesnelerin sınır çizgileri beyaz renkle ifade edilir. Görüntünün matris değerleri arasındaki farkın yüksek olduğu yerler kenarların belirlenmesini sağlamaktadır. Kenar tespitinde elde edilen kenarların haricinde kalan diğer görüntüler siyah renge dönüştürülerek resimden çıkarılır.

Sobel kenar bulma filtresi görüntüyü siyah beyaza çevirdikten sonra Şekil 4.11'de kullanılan çekirdek matrisler ile dikey, yatay ve köşegen şeklindeki kenarların elde edilmesi için kullanılır. Sobel filtresi bir resmin kenarlarına karşılık gelen yüksek frekans bölgelerini tespit ederek keskin kenarların elde edilmesini sağlar (Aslan, 2018).



Şekil 4. 11. Sobel filtre matrisi

Canny algoritması görüntü üzerindeki piksellerin renk değerlerinin birbirlerinden farklılaşması ile belirlenerek kenar tespiti yapılır.

Örnek kullanım: kenar=cv2.canny(img,100,210)

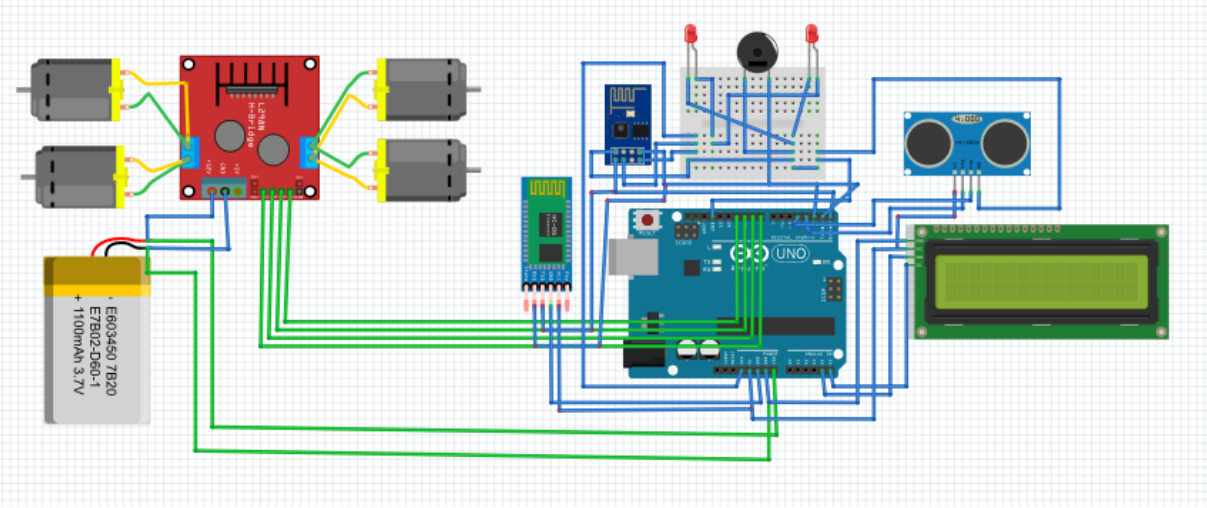
Yukarıda görüldüğü gibi kullanılan iki sayı, eşik değerlerini belirtir. Görüntü üzerinde yer alan matris değeri yüksek eşik değerinden fazla ise, bu noktadan kenar elde edilmeye başlanır ve matris değeri düşük eşik değerinin altına düşene kadar süreç devam ettirilerek kenar elde edilir.

5. TASARIMIN GERÇEKLENMESİ

Yapılan çalışma iki ana bölümden oluşmaktadır. Arduino UNO kartı kullanılarak DC motorlar ile tekerlek hareketi, HC-05 Bluetooth modülü ile cep telefonu üzerinden kontrol edilmesi, HC-SR04 modülü ile engele olan mesafesinin bulunması gibi işlemler gerçekleştirilmiştir. Raspberry Pi mini bilgisayarı ve kamera modülü kullanılarak hareketli görüntülerin, istenilen rengin, araç plakalarının tespiti gibi görüntü işleme işlemleri gerçekleştirilmiştir.

5.1. Arduino UNO Bağlantıları

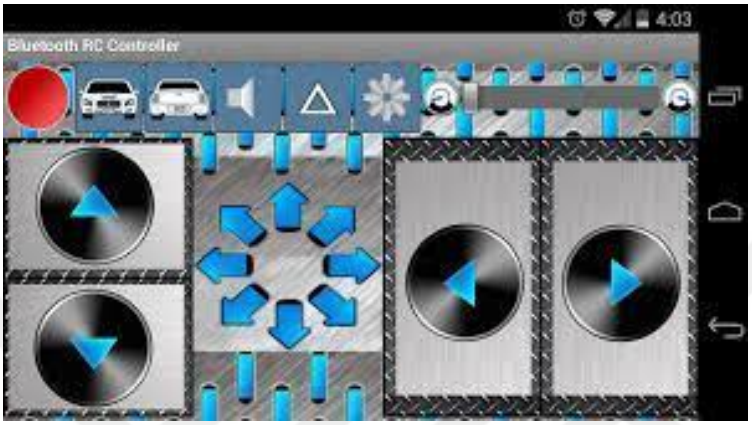
Yapılan çalışmada Arduino UNO ile kontrol edilen devre elemanlarının bağlantıları için çok amaçlı mobil robot platformu kullanıldı. Bağlantı şeması Şekil 5.1'de görülmektedir. Tasarımın güç beslemesi için Lipo pil kullanılmıştır. Lipo pilin artı ucu motor sürücü üzerinde yer alan 12 voltluk klemense, eksi uç ise GND klemensine bağlanmıştır.



Şekil 5. 1. Arduino UNO kartı bağlantı şeması

Robot aracın tekerlek hareketi için DC motor ve bu motorların sürülebilmesi için L298N motor sürücü kullanılmıştır. Kullanılan 4 DC motorun 2 tanesinin çıkışları L298N motor sürücünün solunda bulunan ikili klemense diğer ikisinin çıkışları ise sağında bulunan ikili klemense bağlanmıştır. Motor sürücü üzerinde bulunan in1, in2, in3, in4 pinleri Arduino üzerinde dijital pinlere bağlanarak DC motorların dönüş yönü ayarlanmıştır. Aynı zamanda motor sürücünün 12 voltluk ucu Arduino kartının Vin girişine ve motor sürücünün GND ucunu Arduino kartının GND ucuna bağlayarak Arduino kartının güç beslemesi sağlandı.

Robot aracın hareketini HC-05 Bluetooth modülünü kullanarak cep telefonu üzerinden sağlamak için Arduino Bluetooth Rc Car programı kullanıldı. Bu programın kullanılabilmesi için cep telefonunun Bluetooth özelliği ve Android işletim sistemine sahip olması gerekir. Kullanılan programın arayüzü Şekil 5.2'de görülmektedir. Arduino Uno kartına kodlama yapılırken kullanılan uygulama programında hangi klavye tuşunun hangi görevi yerine getirdiği program arayüzünde ayarlar bölümünde yer almaktadır.



Şekil 5. 2. Arduino Bluetooth RC car arayüzü

Robot aracın hareketi için in1, in2, in3, in4 pinlerinin Arduino'nun PWM pinlerine bağlanıp *Analogwrite* komutu kullanılarak 0-5 volt arası istenilen gerilimin verilmesiyle DC motorların istenilen yöne hareketi Arduino IDE programı ile sağlandı.

HC-SR04 mesafe sensörü bağlantısında Trig ve Echo pin bağlantıları için Arduino kartında 2 dijital pin kullanılmıştır. HC-SR04 mesafe sensörünün Trig ve Echo pinleri kullanılarak ses dalgaları yardımıyla engele olan mesafenin hesaplanması sağlandı. Sürenin hesaplanmasında bir pinin ne kadar süre HIGH ve LOW durumunda kaldığını belirlemek için *PulseIn()* fonksiyonu kullanıldı. Robot aracın engele olan mesafesinin LCD ekrana yazdırılması Şekil 5.3'te gösterilmiştir.

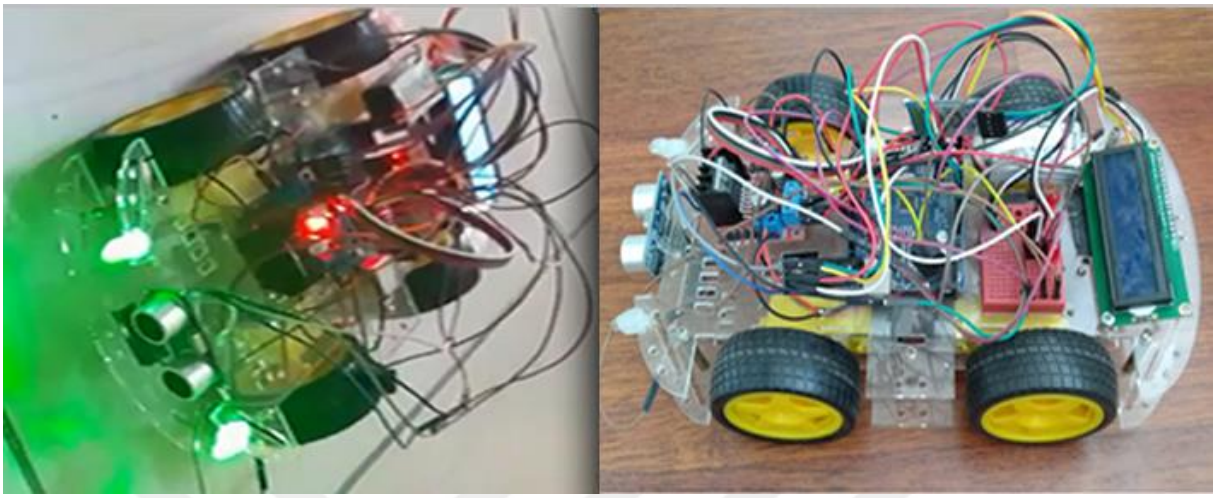
Şekil 5. 3. Robot aracın engele olan mesafesinin yazdırılması

LCD ekran bağlantısında I2C seri haberleşmesi kullanılarak SDA pin bağlantısı için Arduino kartında A4 girişi SCL pin bağlantısı için A5 girişi kullanıldı. Robot aracın engele olan mesafesi 30 cm den daha az kaldığında LCD ekran üzerinde "ARAÇ DURMALI" mesajının yazdırılması ve buzzerdan sesli uyarı verilmesi sağlandı. Engele olan mesafenin hesaplanarak LCD ekrana yazdırılması işlemlerinin kod kümesi Şekil 5.4'te gösterilmiştir.

```
long sure, mesafe;
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
sure = pulseIn(echoPin, HIGH);
mesafe = (sure/2) / 29.1;
 if (mesafe==31) lcd.clear();
 lcd.home();lcd.print(mesafe);lcd.print("cm");
 if (mesafe<100) { lcd.clear();lcd.home();lcd.print(mesafe);lcd.print("cm");}
 if (mesafe<=30) tone(buzzerPin, 440);//melodi(mesafe);
 if (mesafe>30) noTone(buzzerPin);
 if (mesafe<=30) { lcd.setCursor(4,1);lcd.print("ARAÇ DURMALI");}
```

Şekil 5. 4. Engele olan mesafenin yazdırılması için kullanılan kod kümesi

Tasarlanan çalışma çok amaçlı mobil robot platformu kullanılarak devre elemanları ile Arduino kartı bağlantıları yapılarak ilk örnek araç fotoğrafı Şekil 5.5’ te görülmektedir.

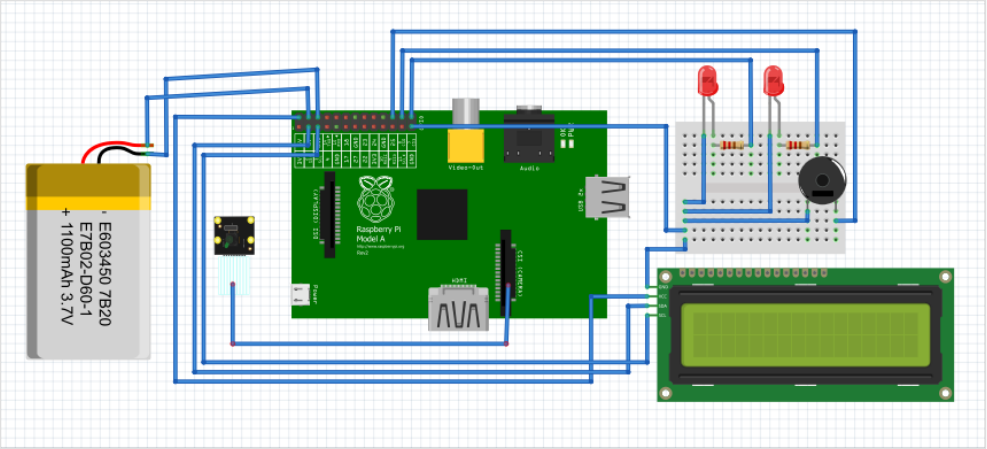


Şekil 5. 5. Tasarlanan ve gerçeklenen robot aracın görüntüsü

Bluetooth ve ESP8266 Wi-Fi modülü bağlantısında UART seri haberleşmesi kullanılarak Rx ve Tx pinlerine bağlantı yapılmıştır. Arduino UNO kartında bir çift Rx ve Tx pini bulunduğu için *softwareserial.h* kütüphanesi kullanılarak dijital pinlerin Rx ve Tx pini gibi kullanılması sağlanmıştır. Bluetooth modülü bağlantısında Rx ve Tx pinleri kullanılırken ESP8266 modülü için Rx ve Tx bağlantısı için dijital pinler kullanılmıştır. Bu bağlantılar yapılırken modül üzerinde yer alan Rx pininin Arduino kartı üzerinde Tx ucuna, modül üzerinde yer alan Tx pininin Arduino kartı üzerinde Rx ucuna ters olarak bağlanması sağlandı.

5.2. Raspberry Pi Bağlantıları

Yapılan çalışmada Raspberry Pi ile kontrol edilen devre elemanlarının bağlantıları Şekil 5.6’da görülmektedir. Raspberry Pi kameranın 15 pinli flex kablosu ile CSI arabirimi kullanılarak bağlantısı sağlanmıştır.



Şekil 5. 6. Raspberry Pi kartı bağlantı şeması

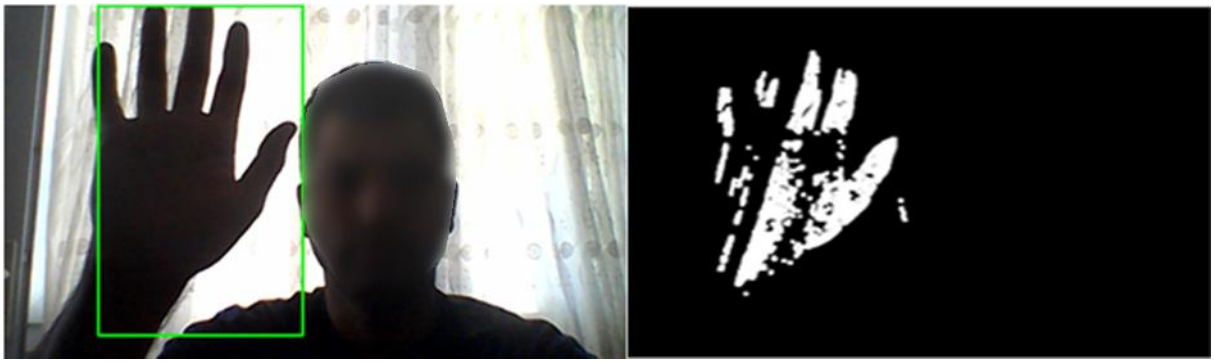
Hareketli görüntülerin elde edilmesi için *CreateBackgroundSubtractorMOG2*() fonksiyonu kullanılmıştır. Bu fonksiyon ile sabit arka planı referans alınarak üzerindeki değişikliklerin yakalanması sağlanmaktadır.

```
capture = cv2.VideoCapture(0)
maske = cv2.createBackgroundSubtractorMOG2()
kernel=np.ones((5,5),np.uint8)
while(1):
    ret, frame = capture.read()
    if not ret:
        break
    frame_yeni = frame
    yuzey_maske = maske.apply(frame_yeni)
    yuzey_maske=cv2.morphologyEx(yuzey_maske,cv2.MORPH_OPEN,kernel)
    pixelsayaci = np.count_nonzero(yuzey_maske)
    x, y, w, h = cv2.boundingRect(yuzey_maske)
    if (pixelsayaci > 3000):
        frame_yeni = cv2.rectangle(frame_yeni, (x, y), (x + w, y + h), (0, 250, 0), 2)
        pixelsayaci=0
        cv2.imshow('Kamera', frame_yeni)
        cv2.imshow('Maske', yuzey_maske)
        cv2.imwrite('maske.png',frame_yeni)
        cv2.imwrite('maske1.png',yuzey_maske)
    k = cv2.waitKey(1) & 0xff
    if k == 27:
        break
```

Şekil 5. 7. Hareketli görüntülerin elde edilmesi ile ilgili kodlar

Elde edilen hareketli görüntünün boyutu, sol üst ve sağ alt köşesinin koordinatları, hareketli görüntüler bulunarak 3000 pikselden büyük olan görüntülerin çerçeve içerisine alınması ve kaydedilmesi sağlandı. Hareketli görüntülerin elde edilmesi ile ilgili kodlar Şekil 5.7'de görülmektedir. Kameradan alınan görüntünün hareketli olan bölümleri çerçeve

içerisine alınması ve hareketli olan yerler beyaz renkte hareketsiz olan yerlerin siyah renkte gösterilmesi Şekil 5.8'de görülmektedir.



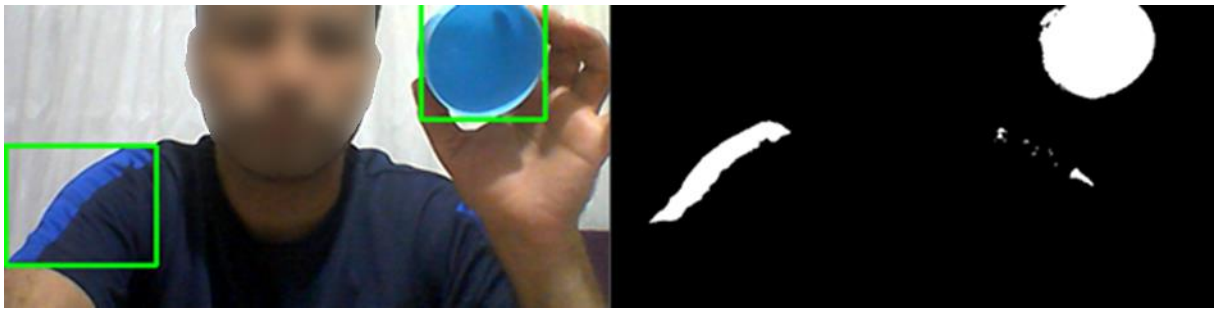
Şekil 5. 8. Hareketli görüntülerin elde edilmesi

İstenilen rengin tespit edilmesi için OpenCV kütüphanesinin *inrange()* fonksiyonu kullanıldı. Bu fonksiyon ile tespit edilecek rengin bulunması için alt ve üst limit değerleri belirlenir. Kameradan alınan görüntü HSV renk uzayına dönüştürülüp *inrange()* fonksiyonu ile maskeleme yapılarak renk tespit edilir. Açık mavi rengin tespiti ile ilgili kodlar Şekil 5.9'da görülmektedir.

```
capture = cv2.VideoCapture(0)
dusuk = np.array([75,100,100])
yuksek = np.array([130,255,255])   # Açık mavi renk aralıkları
while True:
    ret,goruntu = capture.read()
    hsv = cv2.cvtColor(goruntu,cv2.COLOR_BGR2HSV)
    mask = cv2.inRange(hsv,dusuk,yuksek)
    contours,hierarchy=cv2.findContours(mask,cv2.RETR_TREE,cv2.CHAIN_APPROX_NONE)
    for cnt in contours:
        if (cv2.contourArea(cnt)>2000):
            renk=1
            x,y,w,h=cv2.boundingRect(cnt)
            cv2.rectangle(goruntu,(x,y),(x+w,y+h),(0,255,0),thickness=4)
    cv2.imshow("BGR(BLU, GREEN, RED) GORUNTUSU",goruntu)
    cv2.imshow("HSV(HUE,STRATION,VALUE) GORUNTUSU",hsv)
    cv2.imshow("Mask Yapilmis Goruntu",mask)
```

Şekil 5. 9. Renk tespit etme kodları

Tespit edilen açık mavi renkli görüntünün boyutu 2000 pikselden büyük ise çerçeve içerisine alınması sağlandı. İstenilen rengin tespit edilmesine ait görüntü Şekil 5.10'da görülmektedir.



Şekil 5. 10. İstenilen rengin tespit edilmesi

Robot araç üzerinde yer alan kamera ile resim üzerinde yer alan karakterlerin okunması için Tesseract programı kullanıldı. Bu şekilde araçlar üzerinde yer alan plaka bilgilerinin okunması sağlandı. Okunan bilgilerin kaydedilmesi ve istenilen mail adresine gönderilmesi sağlandı. Resim üzerinde yer alan karakterlerin okunarak kaydedilmesi ve istenilen mail adresine gönderilmesi ile ilgili kodlar Şekil 5.11'de görülmektedir.

```
ret,goruntu1=capture.read()
    cv2.imwrite('plaka.png',goruntu1)
    filename = "plaka.png"
    text = pytesseract.image_to_string(Image.open(filename),lang="tur")
    dosya=open('plakalar.txt','a+')
    dosya.write(text)
    dosya.close()
    GPIO.setwarnings(False)
    print(text)
text1=""
dosya=open('plakalar.txt','r')
for satir in dosya:
    text1=text1+satir
mail.sendmail("kecelisezer001@gmail.com","sezer_keceli@hotmail.com",text1.encode("utf-8"))
```

Şekil 5. 11. Resim üzerinde yer alan karakterlerin okunması için kullanılan kodlar

Resim üzerinde yer alan karakterlerin okunmasında güneş ışığı, uzaklık gibi durumlar etkili olmaktadır. Yapılan çalışmada resim üzerinde yer alan karakterler okunup *plakalar.txt* dosyası oluşturularak kaydedilmesi sağlandı. Şekil 5.12'de *plakalar.txt* dosyasının içeriğinden bir bölüm görülmektedir. Bu şekilde resim üzerindeki karakterlerin okunması sağlanarak özellikle plaka bilgilerinin elde edilmesi sağlandı.

Şekil 5. 12. Resim üzerindeki karakterlerin okunması

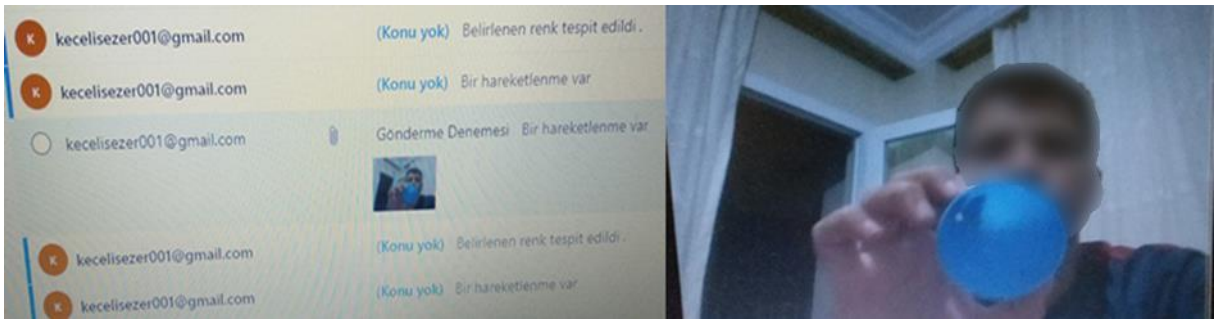
Hareketli görüntünün elde edilerek belirlenen metin ile birlikte gönderilmesi için *smtplib* ve *EmailMessage* kütüphaneleri kullanılmıştır. Kamera ile elde edilen hareketli görüntülerin kaydedilerek istenilen mail adresine görüntülerle birlikte istenilen metnin gönderilmesini sağlayan kodlar Şekil 5.13'de görülmektedir.

```
mesaj=EmailMessage()
mesaj["Subject"]="Hareketli Görüntü"
mesaj["From"]="kecelisezer001@gmail.com"
mesaj["To"]="sezer_keceli@hotmail.com"
mesaj.set_content("Hareketli görüntü tespit edildi")
while(1)
.# Burada hareketli görüntünün tespit edilmesi ile ilgili
.#kodlar bulunmaktadır
.
    if (pixelsayaci > 5000):
        cv2.imwrite("adana.png",frame)
        with open("adana.png","rb") as f:
            image_data=f.read()
            image_name=f.name
            image_type=imghdr.what(f.name)
        mesaj.add_attachment(image_data,maintype="image",subtype=image_type,filename=image_name)
        with smtplib.SMTP_SSL("smtp.gmail.com",465) as smtp:
            smtp.login("kecelisezer001@gmail.com","sk.9862046")
            smtp.send_message(mesaj)
```

Şekil 5. 13. Kameradan alınan hareketli görüntülerin kaydedilmesi ve e-posta gönderme kodları

Yapılan çalışmada belirlenen rengin bulunduğu bir hareketli görüntü tespit edildiğinde "Bir hareketlenme var" ve "Belirlenen renk tespit edildi" şeklinde iki ayrı mesajın ayrıca Belirlenen renkteki hareketli resmin kaydedilerek istenilen mail adresine gönderilmesi sağlandı. Belirlenen rengin bulunmadığı hareketli görüntülerde ise "Bir hareketlenme var" mesajının ve kaydedilen hareketli resmin iki ayrı mesaj olarak istenilen mail adresine gönderilmesi sağlandı. Belirlenen rengin bulunduğu bir hareketli görüntü tespit edildiğinde

gönderilen mesaj Şekil 5.14'te görülmektedir. Belirlenen rengin bulunmadığı bir hareketli görüntü tespit edildiğinde gönderilen mesaj Şekil 5.15'te görülmektedir.



Şekil 5. 14. Belirlenen rengin bulunduğu hareketli resmin mail adresine gönderilmesi

Şekil 5. 15. Belirlenen rengin bulunmadığı hareketli resmin mail adresine gönderilmesi

LCD ekran bağlantısı GPIO pinlerinin BOARD dizilimine göre LCD ekranın SDA pini için Raspbery Pi'nin 3 numaralı girişi SCL pini için 5 numaralı girişi kullanıldı. Arduino UNO kartı ve Raspberry Pi mini bilgisayarında elde edilen sonuçların LCD ekrana yazdırılması için her bir kart için ayrı LCD ekran kullanıldı. Kamera yardımıyla hareketli görüntü tespit edildiğinde 37 numaralı GPIO pinine bağlı ledin yanması ve LCD ekrana yazı yazdırılması sağlandı. İstenilen renk tespit edildiğinde 38 numaralı GPIO pinine bağlı ledin yanması ve LCD ekrana yazı yazdırılması sağlandı.

Yapılan çalışmada hareketli görüntüler ve belirlenen renkte hareketli görüntüler hakkında görsel uyarı için iki ayrı led kullanılmıştır. Belirlenen rengin bulunduğu hareketli görüntü tespit edildiğinde ledlerin her ikisinin yanması sağlandı, ayrıca LCD ekrana "HAREKET VAR" ve "RENK. TES. EDİLDİ (RENK TESPİT EDİLDİ)" yazdırıldı (Şekil 5.16). Belirlenen rengin bulunmadığı hareketli görüntü tespit edildiğinde ledlerin birisinin yanması sağlandı, ayrıca LCD ekrana "HAREKET VAR" ve "R. TES. EDİLMEDİ (RENK TESPİT EDİLMEDİ)" yazdırıldı (Şekil 5.17). Hareketli görüntünün tespit edilemediği

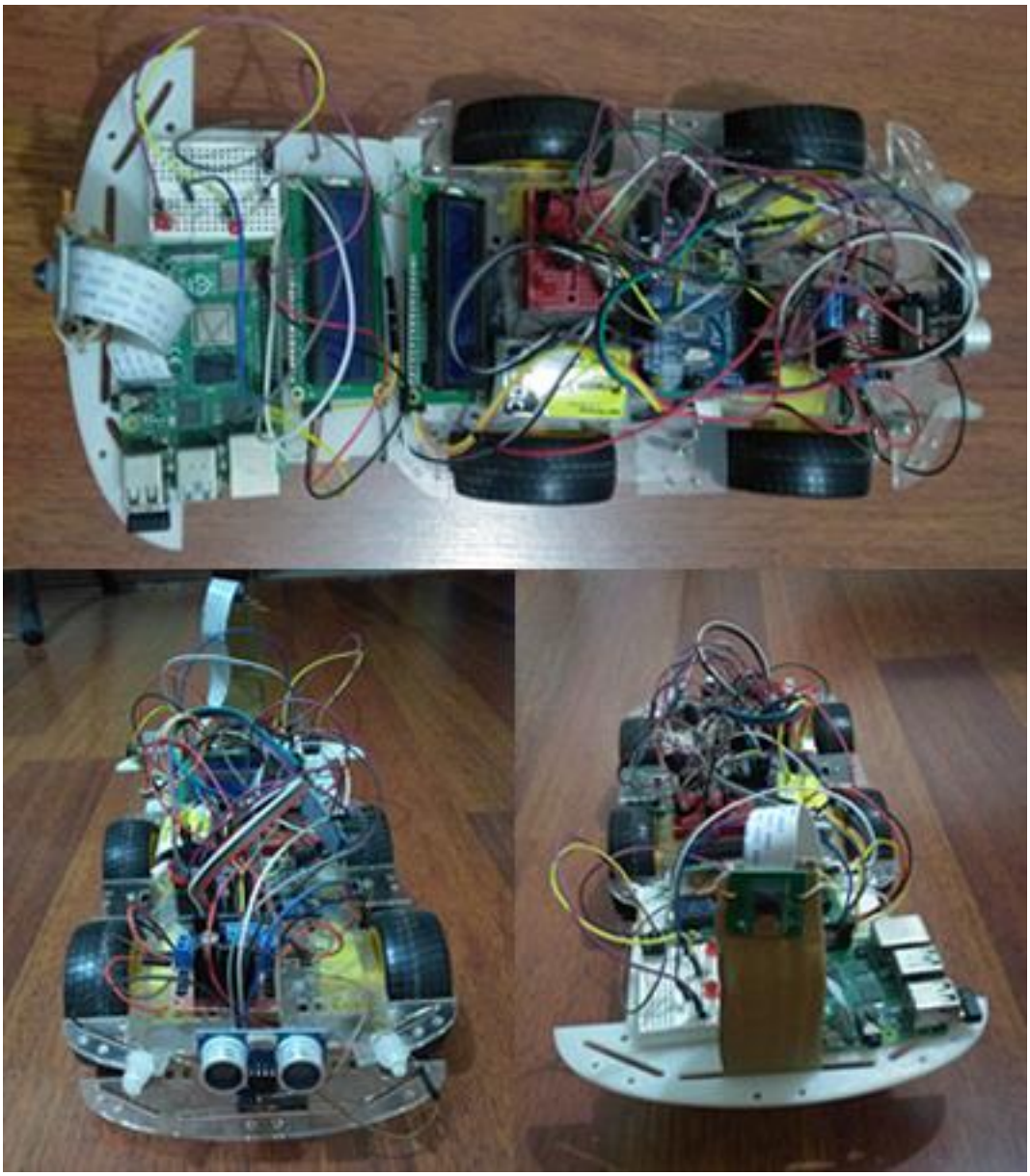
durumda ledlerin her ikisinin de sönük kalması sağlandı, ayrıca LCD ekrana “HAREKET YOK” ve “R. TES. EDİLMEDİ (RENK TESPİT EDİLMEDİ)” yazdırıldı (Şekil 5.18).

Şekil 5. 16. Belirlenen renk tespit edildiğinde yapılan yazılı bilgilendirme

Şekil 5. 17. Hareketli görüntü olduğunda yapılan yazılı bilgilendirme

Şekil 5. 18. Hareketli görüntü bulunamadığında yapılan yazılı bilgilendirme

Elektronik devre elemanlarının bir araya getirilerek çalışmanın gerçeklenmiş hali Şekil 5.19'de görülmektedir.



Şekil 5. 19. Tasarlanan ve gerçeklenen robot aracın fotoğrafları

6. SONUÇ

Bu çalışmada Arduino kartı kullanılarak cep telefonu ile denetlenebilen, engele olan mesafesi ölçülen ve engele çarptığında sesli uyarı mesajı veren aynı zamanda istenilen yere internet üzerinden mesaj gönderebilen bir robot araç tasarlanmış ve gerçeklenmiştir. Bu tasarım çerçevesinde araç ile uzaktan etkileşim mümkün kılınmıştır.

Gerçeklenen robot araç üzerindeki Raspberry Pi mini bilgisayar üzerinde Python ile yazılan programın koşturulması ile araç park halinde iken kamera ile hareketli görüntülerin algılanması, belirli bir renkteki harici bir tehlike (yabani hayvan vb. gibi) tespiti için görüntü işleme tabanlı uzaktan güvenlik uyarı imkanı da sağlanmıştır. Bu program komutlarıyla istenilen renkteki cismin tespit edilmesi kameranın görüş alanında görüntülenen diğer araçların plakalarının elde edilmesi için resim üzerindeki karakterlerin okunması sağlanmıştır. Görüntü işleme komutlarıyla bu sonuçlar elde edildiği anda güvenliğin sağlanmasında uyarı verilmesi için led yardımıyla görsel uyarı verilmesi 16x2 LCD ekrana istenilen mesajın yazdırılması, buzzer ile sesli uyarı verilmesi, internet üzerinden cep telefonuna mesaj gönderilmesi gerçeklenmiştir.

Yapılan çalışma geliştirilerek belirlenen nesnelerin sisteme tanıtılıp, bu nesnelerden herhangi birisi tespit edildiğinde tespit edilen nesneye göre farklı uyarı mesajları ve farklı kişilere mesaj gönderilmesi sağlanabilir. İnsanların giremeyeceği dar yerlerde robot aracın hareketi sağlanıp engelleri algılayarak otonom bir şekilde hareket etmesi ve ortam bilgilerinin elde edilmesi sağlanabilir.

KAYNAKÇA

Abaküs (2020). *Arduino ile Buzzerdan Ses Çıkarmak.* [Erişim:11.01.2021, https://www.abakuskitap.com/blog/icerik/buzzerdan-ses-cikarma]

Aksu, C. (2020). *Arduino ile Çalışan Çok Fonksiyonlu Robot*. (Yüksek Lisans Tezi). Haliç Üniversitesi, Lisansüstü Eğitim Enstitüsü, İstanbul.

Anh, L. T. and Song, J. B. 2010. Object tracking and visual servoing using features computed from local feature descriptor*, International Conference on Control, Automation and Systems (ICCAS),* Korea, 1044-1048.

Arslan, E. (2011). *Hücresel Sinir Ağı Sistemleri Kullanarak Hareketli Nesnelerin Görüntü İşleme Uygulamaları.* (Doktora Tezi). İstanbul Üniversitesi, Fen Bilimleri Enstitüsü,İstanbul.

Aslan, M. F. (2018). *OPENCL Ortamında Görüntü İyileştirme İşlemlerinin Paralel Programlama Yöntemiyle Gerçekleştirilmesi*. (Yüksek Lisans Tezi). Süleyman Demirel Üniversitesi, Fen Bilimleri Enstitüsü, Isparta.

Ateş, E. (2017). *Setup ve loop fonksiyonlari.* [Erişim:08.10.2019, https://www.mobilhanem .com/arduino-setup-loop-fonksiyonlari-ve-merhaba-dunya/]

Aydın, E. (2020). *OpenCv'de Temel Kamera ve Video İşlemleri.* [Erişim:04.06.2020, https://www.mobilhanem.com/opencvde-temel-video-ve-kamera-islemleri/]

Balcı, M. (2019). *Raspberry Pi Mini Bilgisayarı ile Ortam İzleme Sistemi Geliştirilmesi*. (Yüksek Lisans Tezi). Bilecik Şeyh Edebali Üniversitesi, Fen Bilimleri Enstitüsü, Bilecik.

Baykal, A. & Yücelen M. (2016). *Raspberry Pi Kurulumu ve Kullanımı*. [Erişim: 16.3.2020, https://ab.org.tr/ab16/bildiri/230.pdf]

Bengu. (2020). *RGB Renk Uzayları Hakkında Herşey*. [Erişim:13.02.2020, https://www. matbuu.com/blog/rgb-nedir.html]

Bosnali, C. (2019). *OpenCV ile Görüntü İşleme*. [Erişim: 2.01.2021, https://cihanbosnali. com/posts/opencv_goruntu_isleme_3.html]

Çelik, A. (2020). Optik Karakter Tanımada Hata Yayılım Algoritmalarının Performans Kıyaslaması. *Iğdır Üniversitesi Fen Bilimleri Enstitüsü Dergisi,* 10(4), 2328-2340.

Çetinkaya, H. (2012). *Ten Rengi ve Yüz Bulma Teknolojisine Dayalı İnsan Sayma Sistemi.* (Yüksek Lisans Tezi). Bilecik Üniversitesi, Fen Bilimleri Enstitüsü, Bilecik.

Daştan, B. (2019). *Bulanık Mantık Yöntemi ile Otonom Bot Optimizasyonu.* (Yüksek Lisans Tezi). İstanbul Gelişim Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul, s. 13-14.

Dexterindustries (2020). *Run a Program On Your Raspberry Pi At Startup*. [Erişim: 4.10. 2020, https://www.dexterindustries.com/howto/run-a-program-on-your-raspberry-pi-at-startup /]

Dönmez, M. (2020). *Esp8266 modülü nedir? Ne işe yarar*. [Erişim: 11.12.2019, https://www.muratdonmez.com.tr/esp8266-modulu-nedir-ne-ise-yarar/]

Ekmekci, E. (2017). *e-Health Kalkan ve Arduino Kullanarak Çoklu Fizyolojik İşaretlerin Bilgisayar Ortamında Görüntülenmesi.* (Yüksek Lisans Tezi). Afyon Kocatepe Üniversitesi, Fen Bilimleri Enstitüsü, Afyonkarahisar, s. 54-58.

Ekren, G. (2020). *Python Eğitim NumPy*. [Erişim: 16.10.2020, https://www.datasciencearth. com/numpya-giris-bolum-6/]

Bektaş, M. (2017). *Raspberry Pi GPIO İşlemleri*. [Erişim: 4.8.2019, ttps:// elektrikelektronik projeleri.blogspot.com/2017/03/raspberry-pi-gpio-islemleri. html]

Elektrobot (2019). *Serial I2C 16x2 Karakter LCD Modül Kullanımı*. [Erişim: 13.12.2019, http://www.elektrobot.net/serial-i2c-16x2-karakter-lcd-modul-kullanimi/]

Erşahin, G. (2015). *Tablet Bilgisayar ile Kablosuz Gezgin Robot Kontrolü*. (Yüksek Lisans Tezi). Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul, s.7-18.

Göçerler, B. (2015). *Yapay Zeka Tabanlı Plaka Tanıma ile Bariyer Kontrolü*. (Yüksek Lisans Tezi). Trakya Üniversitesi, Fen Bilimleri Enstitüsü, Edirne.

Güngör, H. (2019). *OpenCv Görüntü İşleme ve Renk Uzayı*. [Erişim:11.12.2020, https://medium.com/@hafizegungor/opencv-görüntü-işleme-ve-renk-uzayı-1a76562ff715]

İkizoğlu, K. (2020). *Arduino HC-05 Bluetooth Kullanımı*. [Erişim: 13.06.2020, http://blog. ikizoglu.com/2018/05/arduino-hc-05-bluetooth-kullanimi]

İzgöl, K. (2015). *DC Motor ve Çeşitleri*. [Erişim:14.01.2021, https://maker.robotistan. com/dc-motor-cesitleri-nelerdir/]

Kaplan, A. (2018). *Gerçek ve Yarı Zamanlı Yüz Tespit Etme*. (Yüksek Lisans Tezi). Fırat Üniversitesi, Fen Bilimleri Enstitüsü, s. 21-25.

Karacı, A. & Erdemir M. (2017). Arduino ve Wifi Temelli Çok Sensörlü Robot Tasarımı ve Denetimi. *Bilişim Teknolojileri Dergisi*, 10(4), 435-441.

Köse, O. (2014). *Arduino Analog Giriş ve Çıkış Komutları*. [Erişim: 16.03.2020, http://www.mikrocore.com/genel/arduino-analog]

Macit, H. (2020). *Python ve OpenCV ile konturları Kullanarak hareket Tespiti ve İzleme*. [Erişim: 27.09.2020, https://www.hbmacit.com/2020/09/21/python-ve-opencv-ile-iki-goruntu yu-harmanlama/]

Pişkin, M. (2016). *OpenCV Nesne Tespit ve Tanıma Yöntemleri*. [Erişim:12.03.2021, https://mesutpiskin.com/blog/opencv-nesne-tespiti.html**]**

Pişkin, M. (2017). *OpenCV Arka Plan Çıkarma*. [Erişim:11.02.2021, https://mesut piskin.com/blog/opencv-arka-plan-cikarma-background-subtraction.html

Robotik Sistem (2020). *Arduino Uno.* [Erişim: 13.11.2020, http://www.robotiksistem .com/arduino_uno_ ozellikleri.html]

OpenCV. (2021). *Contours Features.* [Erişim:12.04.2021, https://docs.opencv.org/4.5.2/ dd/d49/tutorial_py_contour_features.html]

OpenCV. (2018). *Image Tresholding*. [Erişim:12.08.2020, https://docs.opencv.org/3.4.3/d7/ d4d/tutorial_py_thresholding.html]

Sailteknoloji (2020). *HC-SR04 Ultrasonik Mesafe Ölçüm Sensörü Özellikleri nedir? Nasıl Çalışır?.* [Erişim:12.01.2021, https://www.sailteknoloji.com/blog/hc-sr04-ultrasonik-mesafe-olcum-sensoru-ozelliikleri-nedir-nasil-calisir-b35.html**]**

Samtaş, G. & Gülesin, M. (2012). Sayısal Görüntü İşleme ve Farklı Alanlardaki Uygulamaları. *Electronic Journal of Vocational Colleges*, 84-93.

Semiz, T. (2018). *Arduino Nedir? Detaylı Arduino Kurulumu*. [Erişim:08.07.2019, https://maker.robotistan.com/arduino-yazilim-kurulum/]

Tezel, C. (2017). *Elektrik Tahrikli Mobil Kontrollü Tank Robotun Tasarımı ve Gerçekleştirilmesi.* (Yüksek Lisans Tezi). Fen Bilimleri Enstitüsü, İstanbul, s. 14,15

Tuncer, S. A. (2019). Optic Disc Segmentation based on Template Matching and Active Contour Method, *Balkan Journal of Electrical & Computer Engineering*, 7(1), 56-60.

Unal, E. (2020). *IOT Dünyasından ESP8266 Wi-fi MCU incelemesi*. [Erişim:11.01.2021, https://enginunal.medium.com/iot-dunyasından-esp8266-wi-fi-mcu-incelemesi-ve-ornek-proje-1]

Uysal, B. (2019). *Arduino ve bluetooth kullanılarak uzaktan kontrollü robot tasarım, yazılım ve uygulamasının yapılması*. (Tüksek Lisans Tezi). Fen Bilimleri Enstitüsü, Yalova.

Vikipedi (2020). *Raspberry Pi*. [Erişim: 11.04.2020, https://tr.wikipedia.org/wiki/ Raspberry_Pi]

Yalçın, M. (2015). *Arduino Komutları*. [Erişim:19.02.2020, http://www.dudigan.com /2015/12/arduino-komutlari/]

Yılmaz, T. (2019*). Raspberry Pi kartı kullanarak nesne tespit ve takip robotunun tasarımı ve modellenmesi*. (Yüksek Lisans Tezi). Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Elazığ.