T.C.

BİLECİK ŞEYH EDEBALİ ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

GÖRÜNTÜ İŞLEME DESTEKLİ YÜZ TAKİP EDEN ARAÇ

Efecan ALTAY

BİTİRME ÇALIŞMASI

DANIŞMANI : Arş.Gör.HAKAN ÜÇGÜN

BİLECİK

8 Haziran 2017

T.C.

BİLECİK ŞEYH EDEBALİ ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

GÖRÜNTÜ İŞLEME DESTEKLİ YÜZ TAKİP EDEN ARAÇ

Efecan ALTAY

BİTİRME ÇALIŞMASI

DANIŞMANI : Arş.Gör.HAKAN ÜÇGÜN

BİLECİK

8 Haziran 2017

BİLDİRİM

Bu kitaptaki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

İmza

Efecan ALTAY

Tarih:

ÖZET

BİTİRME ÇALIŞMASI

GÖRÜNTÜ İŞLEME DESTEKLİ YÜZ TAKİB EDEN ARAÇ

Efecan ALTAY

Bilecik Şeyh Edebali Üniversitesi
Mühendislik Fakültesi
Bilgisayar Mühendisliği Bölümü

Danışman: Arş.Gör.Hakan ÜÇGÜN

2017, 64 Sayfa

Jüri Üyeleri	**İmza**
.....	.....
.....	.....
.....	.....

Görüntü İşleme Destekli Yüz Takip Eden Araç, Raspberry Pi gibi gömülü kartların görüntü işlemede kullanılmasının ve görüntü işleme ile birlikte gömülü sistemin çalıştırılmasıdır. Görüntü işleme ile yüz algılanarak, özel tasarlanmış aracın kontrol edilmesidir.

Anahtar Kelimeler: Raspberry Pi, Görüntü işleme, Yüz Algılama, Yüz Takibi, Gömülü Sistem, Motor Kontrolü

ABSTRACT

THESIS

FACE-TRACKING CAR WITH IMAGE PROCESSING SUPPORT.

Efecan ALTAY

Bilecik Şeyh Edebali University
Engineering Faculty
Department of Computer Engineering

Advisor: Res.Ast.Hakan ÜÇGÜN

2017, 64 Pages

Jury	**Sign**
. . . . .	. . . . .
. . . . .	. . . . .
. . . . .	. . . . .

Face-tracking car with image processing support,Embedded cards such as Raspberry Pi are used in image processing and the embedded system is run with image processing. Controlling a specially designed vehicle by detecting the face by image processing.

Keywords: Raspberry Pi, Image Processing, Face Detecting, Face Tracking, Embedded System,Motor Control

ÖNSÖZ

Bitirme çalışmamın başından sonuna kadar emeği geçen ve beni bu konuya yönlendiren saygı değer hocamlarım ve danışmanım Sayın Arş.Gör.Hakan ÜÇGÜN'e ve Doc.Dr.Metin KESLER'e tüm katkılarından ve hiç eksiltmediği desteğinden dolayı teşekkür ederim.

Efecan ALTAY

8 Haziran 2017

İÇİNDEKİLER

ÖNSÖZ **v**

ŞEKİLLER TABLOSU **viii**

ÇİZELGELER TABLOSU **ix**

1 GİRİŞ **1**

2 PROJE TASARIMI **3**

3 MATERYAL VE YÖNTEM **5**

3.1 Projedeki Yüz Takibi Aşamaları: 5

3.2 Projeye Eklenen Aracın Tasarım Aşamaları: 6

3.3 Proje Malzemeleri ve Maliyeti 7

3.3.1 Projede Kullanılan Malzemeler 7

4 PROJEDE KULLANILAN DONANIMLAR VE YAZILIMLAR **8**

4.1 Raspberry Pi 8

4.1.1 Genel Sistem Özellikleri 8

4.1.2 Raspberry Pi Serileri ve Özellikleri 9

4.1.3 Sisteme Uyumlu İşletim Sistemleri 10

4.1.4 İşletim Sistemi Kurulumu 10

4.1.5 Windowstan Rasbiana Putty ile Bağlanmak 11

4.1.6 İnternete Çıkartmak 12

4.2 Raspberry Pi'DE C++ Kodunun Derlenmesi 13

4.2.1 MakeFile 13

4.2.2 CMake 13

4.2.3 Raspberry Pi CMake kurulumu 13

4.2.4 WiringPi Modülünün Projeye Eklenmesi 14

4.3 Servo Motor 15

4.4 Pan/Tilt Kit 15

4.5 Step Motor 16
4.5.1 Step motor nasıl çalışır ? 17
4.5.2 C++ step motor modülü 18
4.6 DC Motor ve L298N Motor Sürücü Devresi 19
4.7 RGB Led 20
4.8 Buzzer 20
4.9 Lipo Pil 21
4.10 OPENCV 22
4.10.1 Opencv kütüphanesinin Linux'ta derlenme işlemi 23
4.10.2 OpenCV kütüphanesinin kullanılması 24
4.10.3 OpenCV'de görüntü işleme ve tanıma 25
4.10.4 Kameranın OpenCV'de açılması 25
4.10.5 CascadeClassifier örnekleme dosyasının yüklenmesi 26
4.10.6 Görüntüdeki yüzün algılanması 27
4.11 Projenin Akış Diyagramı 29
4.12 Proje İçin Yazılan Cpp Modülleri 30
4.12.1 StepMotor sınıfı 30
4.12.2 L298N sınıfı 31

5 SONUÇLAR VE ÖNERİLER **33**

6 EKLER **34**
6.1 FindWiringPi.cmake 34
6.2 CMakeLists.txt 34
6.3 faceTracker.cpp 35
6.4 StepMotor.cpp 46
6.5 L298N.cpp 49

KAYNAKLAR **52**

ÖZGEÇMİŞ **54**

ŞEKİLLER TABLOSU

Şekil 2.1 Kişi Takip Sistemi 1 3
Şekil 2.2 Kişi Takip Sistemi 2 4
Şekil 4.1 Raspberry Pi Versiyonları ve Özellikleri [1]. 9
Şekil 4.2 Raspberry Pi Putty ile bağlanmak. 11
Şekil 4.3 İnternete çıkartma. 12
Şekil 4.4 Pan/tilt kit ve servo motor 15
Şekil 4.5 Step motor ve sürücü devresi [2] 16
Şekil 4.6 Step motor çalışma mantığı [10] 16
Şekil 4.7 Step motor çalışma mantığı [3] 17
Şekil 4.8 Step motor modülü fonksiyonları 18
Şekil 4.9 Webcam ve step motorlarla birlikte pan/tilt 18
Şekil 4.10 L298N motor sürücü devresi [4] 19
Şekil 4.11 Dc 250 rpm motor [5] 19
Şekil 4.12 RGB Led [6] 20
Şekil 4.13 Buzzer [7] 20
Şekil 4.14 11.1v 1050mAh Lipo [9] 21
Şekil 4.15 CMakeFileLists.txt 24
Şekil 4.16 OpenCV Yüz Tanıma Çıktısı 26
Şekil 4.17 OpenCV Yüz Tanıma Çıktısı 27
Şekil 4.18 Projenin Akış Diyagramı 29
Şekil 4.19 Projenin son durumu 32

ÇİZELGELER TABLOSU

Tablo 3.1 Malzeme listesi 7
Tablo 3.2 Projenin yapılmasında harcanan ek malzeme giderleri 7

1 GİRİŞ

Bu projede yüz algılama ve takip sisteminin devamı olarak, arac üzerine yerleştirilmiş yüz takip eden kamera ile birlikte bir araca yön verilmesi sağlanmıştır.

Projenin Amacı

Görüntü işleme destekli yüz takip eden araç, hava, kara veya deniz araçlarının, akıllı hale gelmesinin bir adımı olarak Raspberry Pi gibi kartların görüntü işlemede kullanılmasını yaygınlaştırmak.

Projenin Kapsamı

Raspberry Pi'de OpenCV Görüntü işleme ile Cpp kodu yazılarak servo yada step motorlarla yüz takibi yapılacaktır. Yüz takibine göre arac hareket edecektir. Kullanımı basitleştimek amacıyla modüler şekilde cpp kodları yazılacaktır.

Sonuçlar

Raspberry Pi, yüz algılamada yüksek çözünürlü kameralarda yeterince verimli olmamakla birlikte işlemci hızı yeterlidir. Pin kontrolü mikroişlemcilerde yapabildiğimiz tüm verimliliğiyle işlemlerimizi gerçekleştirmiştir.

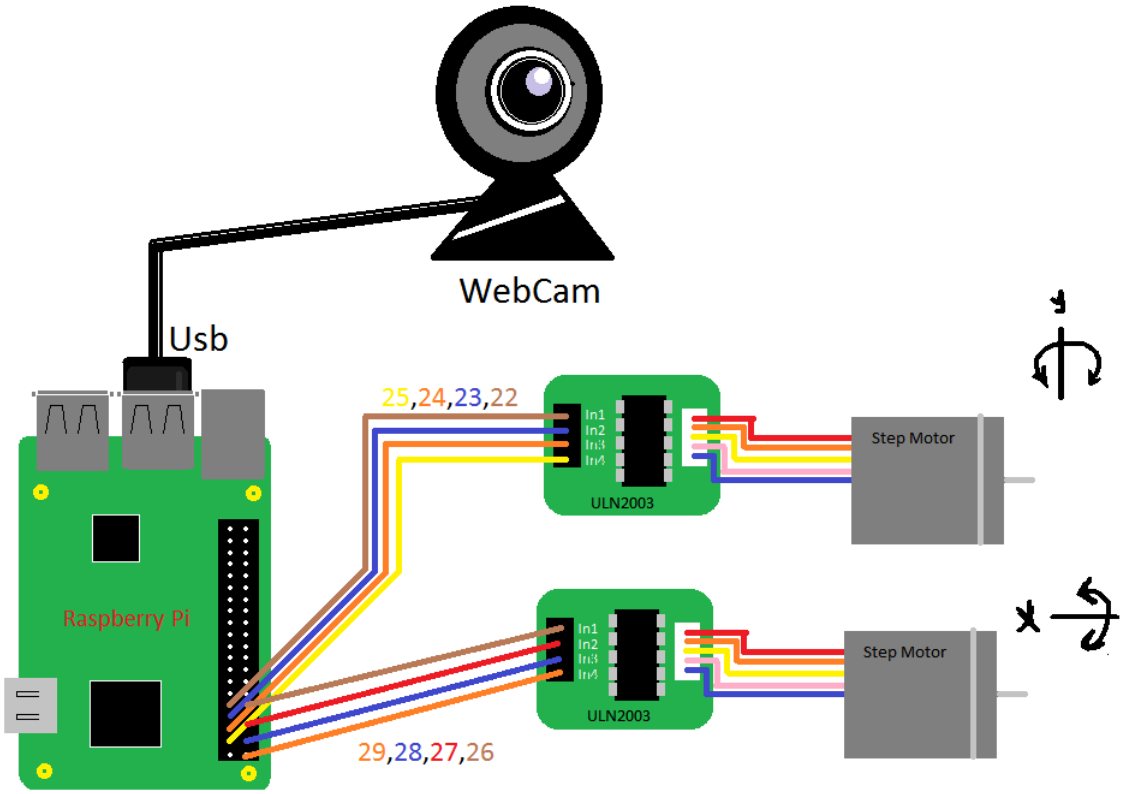
Yüz algılama sistemleri günümüzde, akıllı cihazların,gelişmiş dijital kameraların ve bir çok elektronik cihazın kullandığı sistemlerdir.Yüz algılama ve takip sistemi projesi Raspberry Pi'de uygulamarında çalışma başarımını görmek ve modüllerin Raspberry Pi'ye entegresini hızlandırmak amaçıyla yapılmıştır.

Benzer uygulamalarda görülen, matlabla görüntünün yüksek işlemciye sahip bilgisayarlarda işlenip gömülü sistemin uzaktan kontrol edilmesine karşılık, Raspberry Pi işlemcisi ve kontrol edilebilir GPIO pinleri sayesinde iki işi tek kartta yapma olanağı sağlanmıştır.

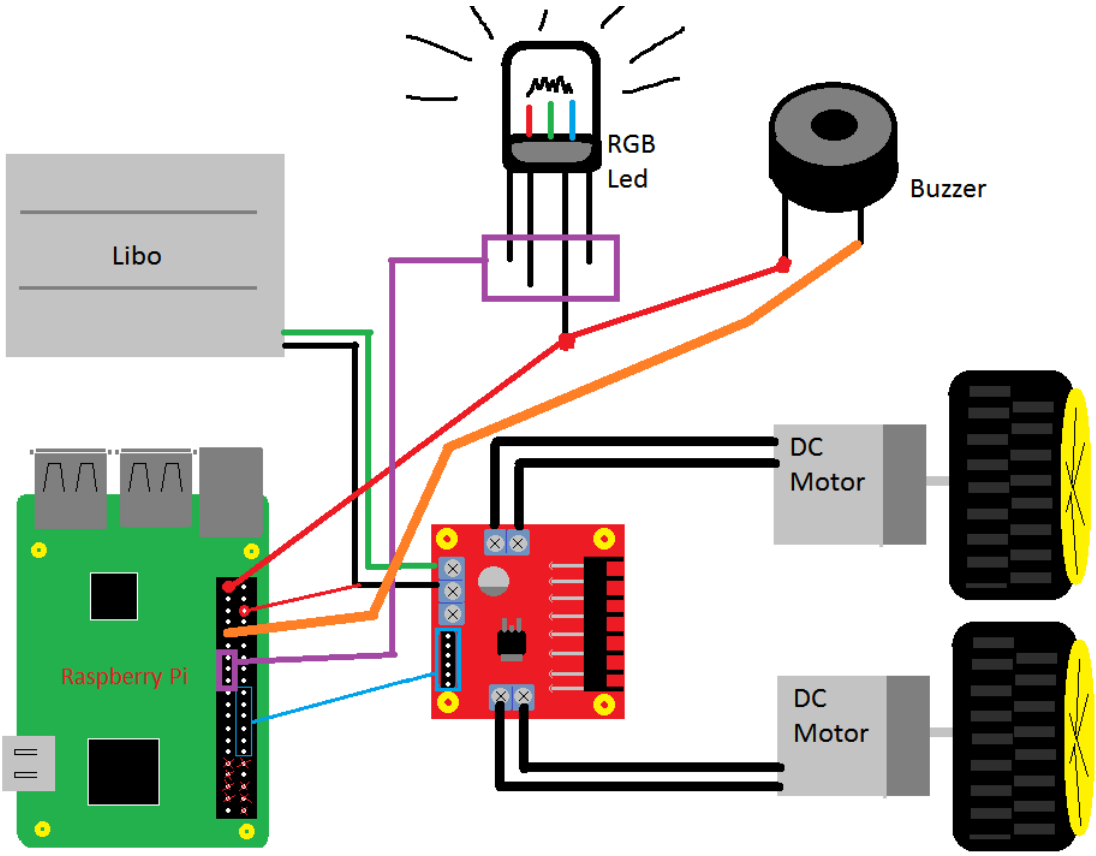
Nesne takibiyle birlikte akıllı arabalarda gelişim göstermekte. Akıllı arabaların görüntü işleme ile hareket etmesini temsil etmek için bu projede yapılacak işlemler basit bir protatip olarak yapılmıştır.

2 PROJE TASARIMI

Bu bölümde projenin tasarımına ait bilgiler yer almaktadır.Şekil 2.1'de yüz algılama ve takip sistemi projesinin tasarım şeması verilmiştir. Yüz algılama ve takip sistemine ek olarak Şekil 2.2'de araç tasarımı verilmiştir.



Şekil 2.1: Kişi Takip Sistemi 1



Şekil 2.2: Kişi Takip Sistemi 2

3 MATERYAL VE YÖNTEM

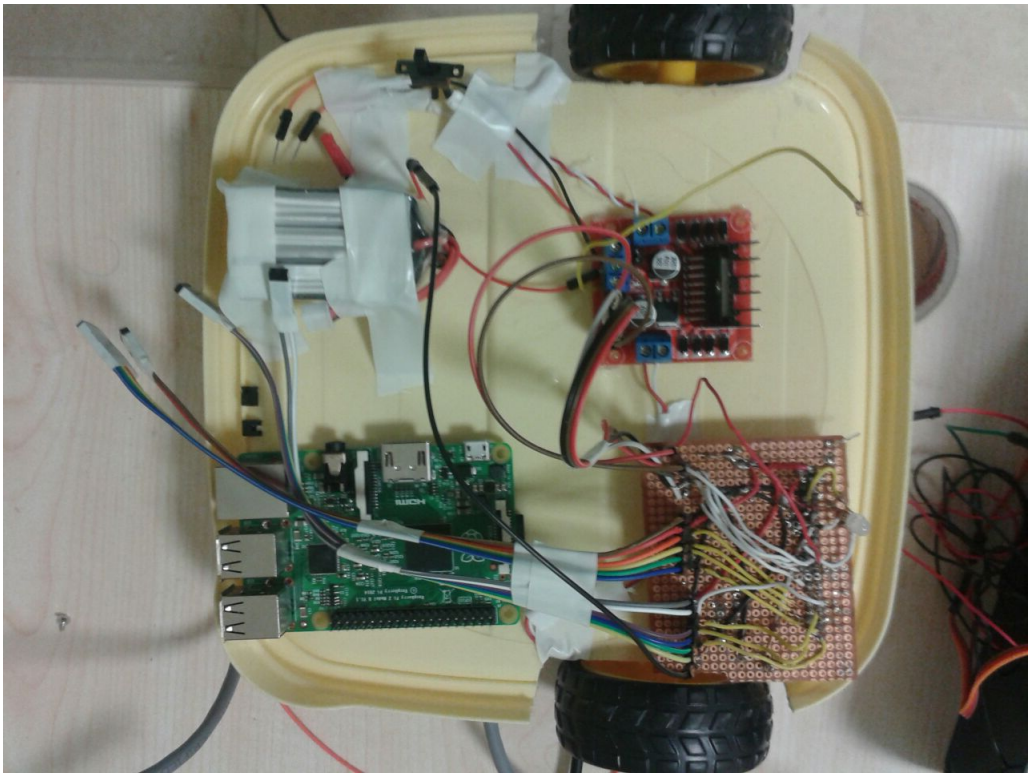
Bu kısımada projede kullanılan materyaller ve izlenilen yol anlatılacaktır.

3.1 Projedeki Yüz Takibi Aşamaları:

- Projede kullanılacak malzemeler ve listesi belirlenip siparişi yapıldı.
- Görüntü işlemede kullanılan OpenCV Kütüphanesi araştırması yapıldı.
- OpenCV kütüphanesinin Raspberry Pi kartındaki işletim sisteminde derlenip çalıştırılması sağlandı.
- Yüz algılaması için gereken program kodu yazıldı ve geliştirildi.
- Yüz algılamadaki performans test edildi.
- RaspberryPi GPIO kontrolü üzerine çalışma yapıldı.
- Kullanılacak modüller için cpp servoMotor ve stepMotor modülleri yazıldı.
- Test aşamasıda servoMotorun ve stepMotorun performansları test edildi.
- Uygun gelen servo Motor kullanılması seçildi.
- Test aşaması olarak Raspberry Pi görüntü işleme test aşaması belirlendi.

3.2 Projeye Eklenen Aracın Tasarım Aşamaları:

- Yüz takibi sistemine eklenecek parçalar belirlenip sipariş yapıldı.
- Gelen parçaların kesim ve montaj işlemleri yapıldı.
- Raspberry Pi'yi rahat sistemden çıkartıp takabilmek için devre(shield) geliştirildi.
- Sistemin, platformun üzerine yerleşmesi için tasarım ayarlandı.
- Kurulan sistem için ek olarak cpp kodları yazıldı. Yüz takibi sistemine dahil edildi.
- Test aşaması gerçekleştirildi.



3.3 Proje Malzemeleri ve Maliyeti

Bu bölümde projenin gerekli malzeme listesi ve maliyeti anlatılacaktır.

3.3.1 Projede Kullanılan Malzemeler

Raspberry Pi 2	130 tl
2*(Step motor + ULN2003A step motor sürücü devresi)	2*18 tl
Parm/Tilt Kit	30 tl
WebCam Camera	10 tl
2*(DC Motor + tekerlek)	40 tl
L298N Motor Sürücüsü	30 tl
11,1V 1050mAh lipo pil	60 tl
Plastik Kap	5 tl
jumper Kablolar	5 tl
.....	.....
Toplam	356 tl

Tablo 3.1: Malzeme listesi

Projenin yapılması için ek gerekli malzemeler aşağıda verilmiştir.

-Ethernet portu olan ve kablosuz internete çıkabilen bilgisayar
-Ethernet Kablosu

Üzerinde çalışılan ek olarak maaliyet olan malzeme listesi Tablo 3.1'de verilmiştir.

2x Mini Servo Motor	2*10 tl
.....	.....
Genel Toplam	356 + 20 = 376 tl

Tablo 3.2: Projenin yapılmasında harcanan ek malzeme giderleri

4 PROJEDE KULLANILAN DONANIMLAR VE YAZILIMLAR

Bu bölümde projede kullanılan donanımlar ve yazılımlar hakkında açıklamalar yapılacaktır.

4.1 Raspberry Pi

Birleşik Krallık'ta Raspberry Pi Vakfı tarafından okullarda bilgisayar bilimini öğretmek amacılığıyla geliştirilmiş kredi kartıbüyüklüğünde tek kartlı bir bilgisayardır.

Raspberry Pi bilgisayarı Element 14/Premier Farnell, RS Components ve Egoman firmaları tarafından imal edilmektedir. Bu firmalarRaspberry Pi bilgisayarını Internet üzerinde satmaktadırlar. Egoman, Çin ve Tayvan'da satılan bir sürümünü satmaktadır ve rengi kırmızı olup FCC/CE etiketlerini taşımaz, bunun dışında diğerlerinden hiçbir farkı bulunmamaktadır. [1]

4.1.1 Genel Sistem Özellikleri

Raspberry Pi, ilk modellerinde ARM1176JZF-S 700 MHz merkezi işlem birimini içeren Broadcom BCM2835 mikroçipi üzerine kurulmuştur. Daha sonra piyasaya çıkan Raspberry Pi 2 modelinde Broadcom BCM2836 kullanmıştır. VideoCore IV GPU grafik işlem birimine sahiptir.Booting ve veri depolaması için SD kart kullanır. Üzerinde USB 2.0 portları, HDMI video çıkışı, ses çıkışı, MIPI kamera girişi, GPIO arayüzü ve 5V MicroUSB güç girişi bulunmaktadır. [1]

4.1.2 Raspberry Pi Serileri ve Özellikleri

Aşağıdaki tablolarda Raspberry Pi özellikleri verilmiştir.

	Raspberry Pi 1 Model A	Raspberry Pi 1 Model A+	Raspberry Pi 1 Model B	Raspberry Pi 1 Model B+
Çıkış tarihi	Şubat 2013	Kasım 2014	Nisan-Haziran 2012	Temmuz 2014
Fiyatı	$25	$20	$35	$25
Anakart	Broadcom BCM2835			
CPU	700 MHz tek çekirdekli ARM1176JZF-S			
GPU	Broadcom VideoCore IV @ 250 MHz			
Bellek	256 MB (Paylaşımlı)		512 MB (Paylaşımlı)	
USB 2.0 portları	1	2	4	
Video giriş	15-pin MIPI camera arayüzü (CSI) bağlantısı			
Video çıkış	HDMI, RCA konnektörü	HDMI, 3.5 mm TRS konnektörü	HDMI, RCA konnektörü	HDMI, 3.5 mm TRS konnektörü
Ses girişi	I²S (revizyon 2 itibarıyla)			
Ses çıkışı	3.5 mm telefon jakıyla analog; HDMI ile dijital ve I²S (revizyon 2 itibarıyla)			
On-board bellek	SD / MMC / SDIO cart slotu	MicroSDHC slotu	SD / MMC / SDIO kart slotu	MicroSDHC slotu
On-board ağ	Yok		10/100 Mbit/s Ethernet (8P8C)	
Çevre birimleri	8× GPIO	17× GPIO	8× GPIO	17× GPIO
Güç derecesi	300 mA (1.5 W)	200 mA (1 W)	700 mA (3.5 W)	600 mA (3.0 W)
Güç kaynağı	5V MicroUSB ya da GPIO başlığı			
Boyut	85.60 mm × 56.5 mm	65 mm × 56.5 mm × 10 mm	85.60 mm × 56.5 mm	

	Raspberry Pi 2 Model B	Raspberry Pi 3 Model B	Compute Modulü	Raspberry Pi Zero
Çıkış tarihi	Şubat 2015	Şubat 2016	Nisan 2014	Kasım 2015
Fiyatı	$35	$35	$30	$5
Anakart	Broadcom BCM2836	Broadcom BCM2837	Broadcom BCM2835	
CPU	900 MHz dört çekirdekli ARM Cortex-A7	1.2 GHz 64-bit dört çekirdekli ARM Cortex-A53	700 MHz tek çekirdekli ARM1176JZF-S	1 GHz tek çekirdekli ARM1176JZF-S
GPU	Broadcom VideoCore IV @ 250 MHz			
Bellek	1 GB (Paylaşımlı)		512 MB (Paylaşımlı)	
USB 2.0 portları	4		1	1 (MicroUSB)
Video giriş	15-pin MIPI kamera arayüzü (CSI) konnektörü		2× MIPI kamera arayüzü	
Video çıkış	HDMI, 3.5 mm TRS konnektörü		HDMI, 2× MIPI görüntü arayüzü	Mini-HDMI
Ses girişi	I²S (revizyon 2 itibarıyla)			
Ses çıkışı	3.5 mm telefon jakıyla analog; HDMI ile dijital ve I²S (revizyon 2 itibarıyla)		Analog, HDMI, I²S	Mini-HDMI, PWM içindeki stereo ses
On-board bellek	MicroSDHC slotu		4 GB eMMC flash bellek çip	MicroSDHC
On-board ağ	10/100 Mbit/s Ethernet (8P8C)	10/100 Mbit/s Ethernet, 802.11n WiFi, Bluetooth 4.1		Yok
Çevre birimleri	17× GPIO		46× GPIO	40× GPIO
Güç derecesi	800 mA (4.0 W)		200 mA (1 W)	~160 mA (0.8 W)
Güç kaynağı	5V MicroUSB ya da GPIO başlığı			
Boyut	85.60 mm × 56.5 mm		67.6 mm × 30 mm	65 mm × 30 mm × 5 mm

Şekil 4.1: Raspberry Pi Versiyonları ve Özellikleri [1].

4.1.3 Sisteme Uyumlu İşletim Sistemleri

Raspberry Pi ARM mimarisinde tasarladığı için linux tabanlı işletim sistemlerinin çoğu çalıştılabilir fakat sistemine en uyumlu işletim sistemleri kendi resmi web sitesinin download kısmından ulaşabiliriz. Bunlar ; Raspian,Ubuntu Mate,Snapy Ubuntu Core,Windows 10 IOT Core... gibi.
Projede kullanılan işletim sistem Raspbiandır.

4.1.4 İşletim Sistemi Kurulumu

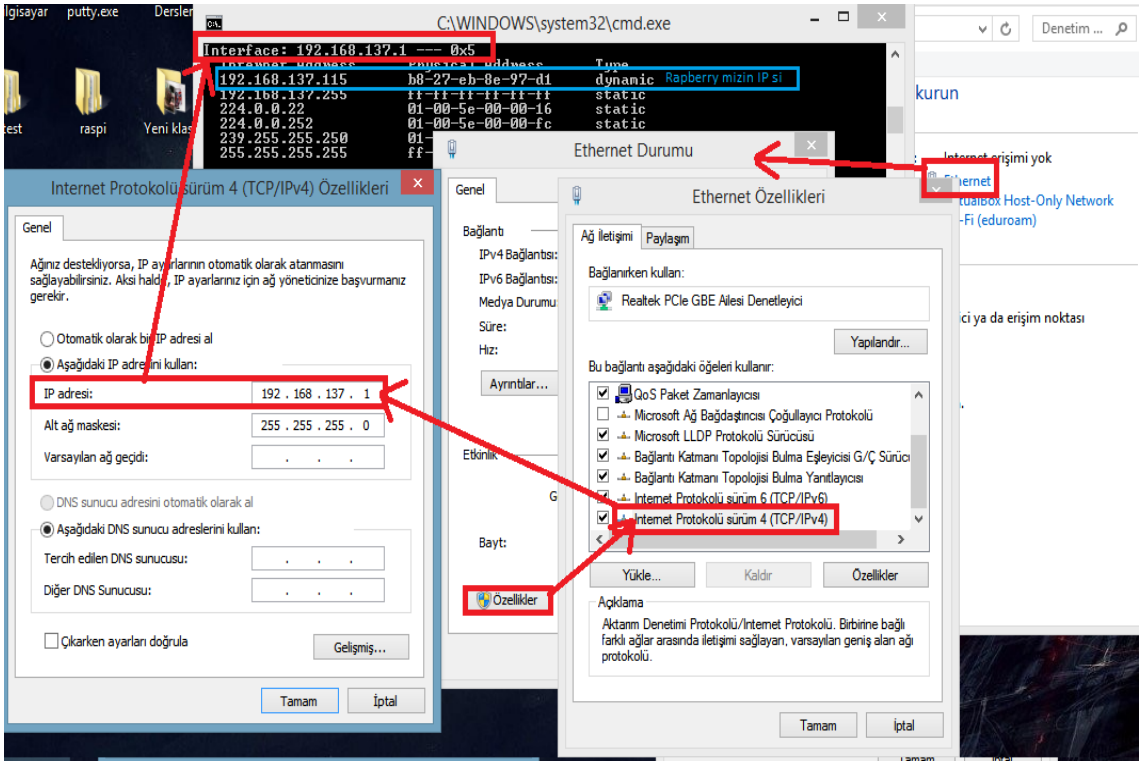
Raspberry pi bildiğimiz gibi SdCard ile çalışmaktadır. İşletim sistemi kurmak içinde bu sd cartın içine imajlamak gerekir bu işlem için raspberry pi nin resmi sitesinden imaj dosyası indirilip imaj programı yardımıyla sd kartımıza yükleriz ve Raspberry Pi da çalıştırmak üzere img dosyamız çalışır.
Win32 Disk imager : `https://sourceforge.net/projects/win32diskimager/`
Raspian işletim sistemi : `https://www.raspberrypi.org/downloads/raspbian/`
Eğer sdmizi tekrar kullanmak üzere geri formatlamak istiyorsak sd formatter programı kullanmalıyız.
SDCard formatter : `https://www.sdcard.org/downloads/formatter_4/`

4.1.5 Windowstan Rasbiana Putty ile Bağlanmak

Rapberry Pi'yi internete çıkartmak için interneti Raspberry için paylaşıma açmamız gerekir. Raspberry Pi'deki işletim sistemini kurduk ve işletim sistemine putty ile erişmek için,puttyi windowa kurmamız gerekir.

Putty : `http://www.putty.org/`

Putty kurulumu sonrası Raspberry Pi'nin konsoluna bağlanmak için ethernet kablosuyla bilgisayara bağlantı kurmamız gerekir. Bu olaydan sonra Raspberry Pi'ye bilgisayarımız otomatik olarak ip atayacaktır.Atanan IP'yi windows konsoldan arp -a yazarak bulabiliriz.

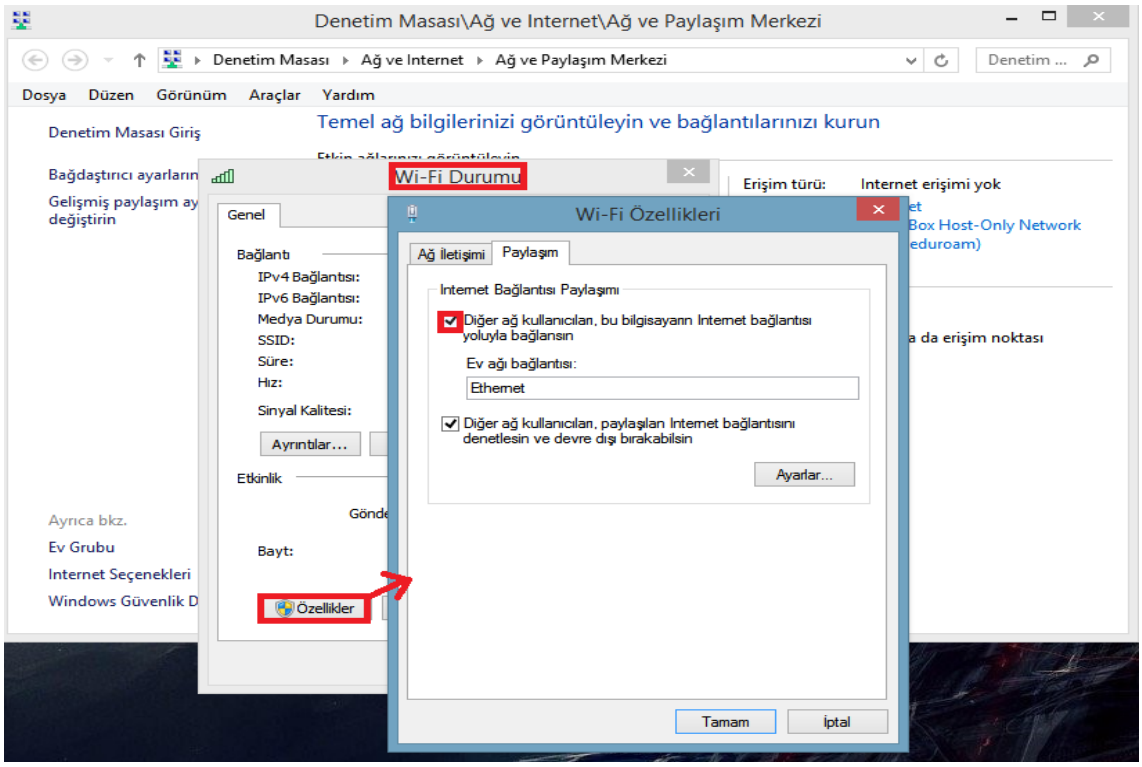


Şekil 4.2: Raspberry Pi Putty ile bağlanmak.

Atanan IP ile putty ile bağlanabiliriz. Ve Rasbian konsoluna ulaşabiliriz. Komutlar bildiğimiz linux işletim sistemi komutlarıdır.

4.1.6 İnternete Çıkartmak

Raspberry Pi'yi internete çıkartmak için interneti Raspberry için paylaşıma açmamız gerekir.



Şekil 4.3: İnternete çıkartma.

4.2 Raspberry Pi'DE C++ Kodunun Derlenmesi

Bu bölümde Raspberry Pi'de cpp kodları derlenmesi ve çalıştırılması anlatılacaktır.

4.2.1 MakeFile

C ile yazdığımız kodu derlerken gcc derleyicisine parametreler göndeririz ve derleriz.

kodlarımız fazla ise, örnek:

```
gcc -o deneme deneme.c -ldenemeler
gcc -o deneme2 deneme2.c -ldenemeler
```

gibi makefile ile komutlar oluştururuz ve bu komutlar hepsini derlemede çalıştırmada yardımcı olur.

4.2.2 CMake

CMake ise Makefile içeriklerini oluşturur. Makefile yazıcağımıza biz cmake yardımıyla derleyeceğimiz kodları yazarız, cmake ise basit halde çıkartır ve makefile oluşturur sonrasında bize sadece derleme işlemi için make komutunu çalıştırmak kalır.

4.2.3 Raspberry Pi CMake kurulumu

Raspberry Pi'de CMake yüklü değildir ama apt reposunda mevcuttur. Basit şekilde yükleyebiliriz. Local repoda linkleri olmayabilir. bu yüzden Update ederek işimiz daha garantiye alabiliriz.

```
$ sudo apt-get update
$ sudo apt-get install cmake
```

4.2.4 WiringPi Modülünün Projeye Eklenmesi

WiringPi kütüphanesi ve CMake sistemde yüklü ise, CMake'in sistemdeki yüklü konumuna erişim içine modül yazmamız gerekiyor, bu sayede CMake'e WiringPi kütüphanesini tanıtıyoruz.

Raspbian sisteminde repodan yüklediğimiz cmake

```
Usr/share/cmake
```

konumuna gelmektedir. Bu konumda cmake'e modul yazmamız gerekiyor. Bu yüzden

```
cmake/Module
```

klasörünün içine giriyoruz. Buraya modül eklemek için root kulanıcı olarak erişmemiz gerekecektir aksi taktirde kayıt işlemi yaptırmayacaktır. Root kullanıcısı olmak için

```
$ sudo su
```

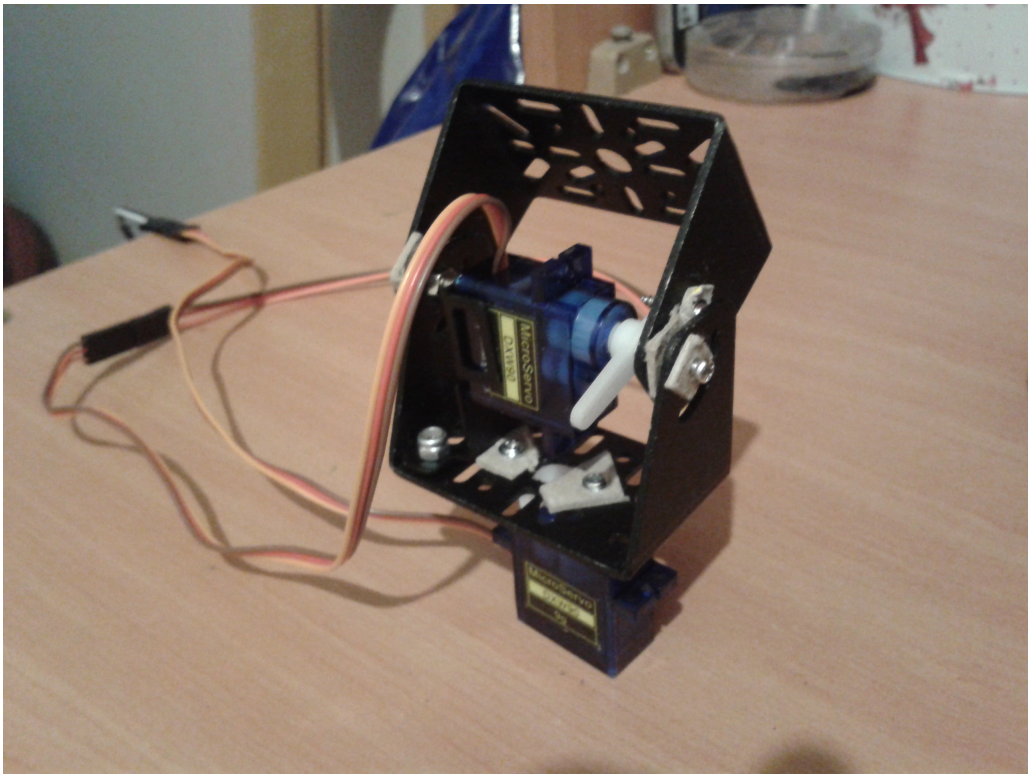
komutunu çalıştırmamız yeterlidir. Module klasörünün altına eklenecek cmake modulu FindWiringPi.cmake ile kaydedilecektir ve ekte kodları verilmiştir.

4.3 Servo Motor

Servo motorlar, pwm kanalı sayesinde istedimiz açı kadar döndürmeye yarayan motorlardır.Raspberry Pi'de pwm kanalını yazılımsal olarak oluşturan wiringPi kütüphanesinden kullanırız.

4.4 Pan/Tilt Kit

Pan/Tilt Kitler, 2 veya 3 eksenli olmak üzere dönüş mekanizmalarıdır.bu mekanizmalara step veya servo motorlar ile dönüş hareketleri sağlanır.

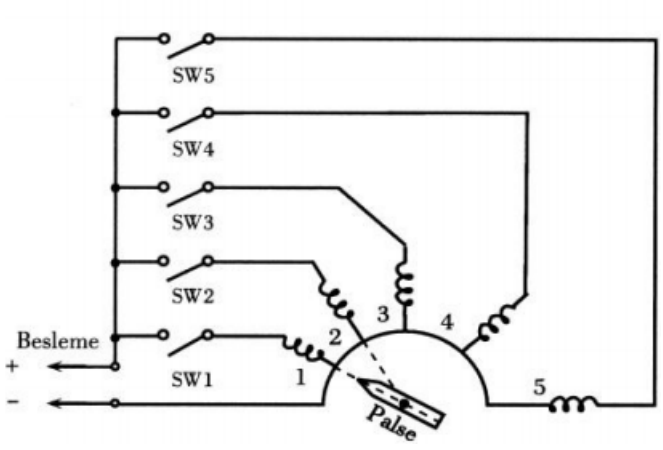


Şekil 4.4: Pan/tilt kit ve servo motor

4.5 Step Motor

Elektrik enerjisini dönme hareketine çeviren elektro-mekanik cihaza step motor denir. Step motorlara aynı zamanda adım motorları adı verilir. Açısal konumu adımlar halinde değiştiren, çok hassas sinyaller ile sürülen motorlara adım motorları denir. [2]

Şekil 4.5: Step motor ve sürücü devresi [2]



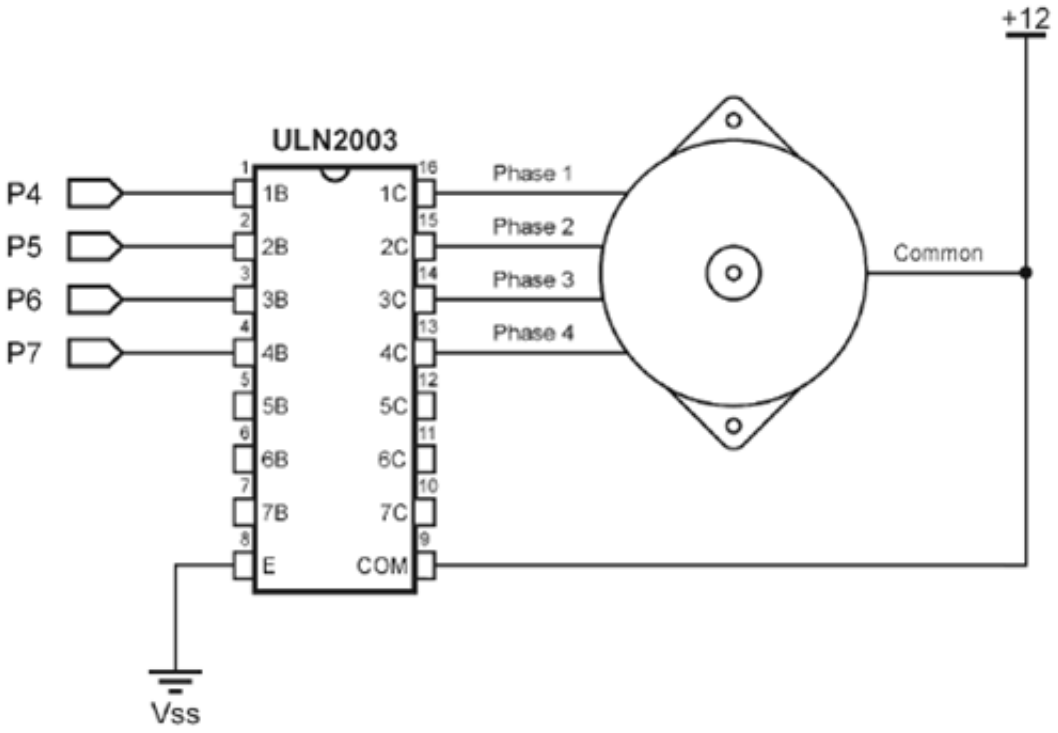
Şekil 4.6: Step motor çalışma mantığı [10]

4.5.1 Step motor nasıl çalışır ?

Step motoru çalıştırabilmek için step motor sürücü devresine ihtiyaç vardır. Projede kullanılan ULN2003 step motor sürücü devresidir.

ULN2003 devresinin çalışma mantığı P4,P5,P6 ve P7 girişlerine teker teker sırayla 5v giriş verip sonra 0v vermektir.Bu sayede step motorumuz bir step ilerlemiş olacaktır.

Saat yönünde dönmesi için P4,P5,P6 ve P7 sırası ile 5v verilip 0v verilir.saatin tersi yönünde dönmesini istiyorsak p7,p6,p5 ve p4 sırası ile 5v verilip 0v verilir.



Şekil 4.7: Step motor çalışma mantığı [3]

4.5.2 C++ step motor modülü

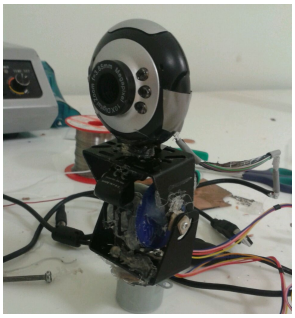
C++ da yazılan step Motor modülüne ait fonksiyonları görmektesiniz.

`https://github.com/EfecanAltay/RaspberryPi_Modules`

Adresinden RaspberryPi için cpp yazdığım modülleri kullanabilirsiniz.

```
void setup(void);// kurulum fonksiyonudur.bu fonksiyonla
    wiringPi kütüphanesi tekrar çağrılır.
void clearPins(void);//tüm giriş pinleri temizleyen
   fonksiyondur.
void turnRight(int);//motoru sağ döndürmeye yarayan
   fonksiyondur.Giriş parametresine göre motor o
   milisaniye araklıklarıyla sağa döner.(giriş minimum 5
    ms olabilir)
void turnLeft(int);//motoru sola döndürmeye yarayan
   fonksiyondur.Giriş parametresine göre motor o
   milisaniye araklıklarıyla sola döner.(giriş minimum 5
    ms olabilir)
void turnFastLeft();//Motorun sola dönmesi için minimum
   5 ms ayarlanmış olarak çalıştıran fonksiyondur.
void turnFastRight();//Motorun sağa dönmesi için minimum
    5 ms ayarlanmış olarak çalıştıran fonksiyondur.
StepMotor(int,int,int,int);//Motoru raspberrye
   bağladığımız pinleri giriş parametresi olarak kurulum
    yaptığımız kurucu fonksiyonumuz.
void clear();//Motorun tüm pinlerini temizleyen
   fonksiyon.
```

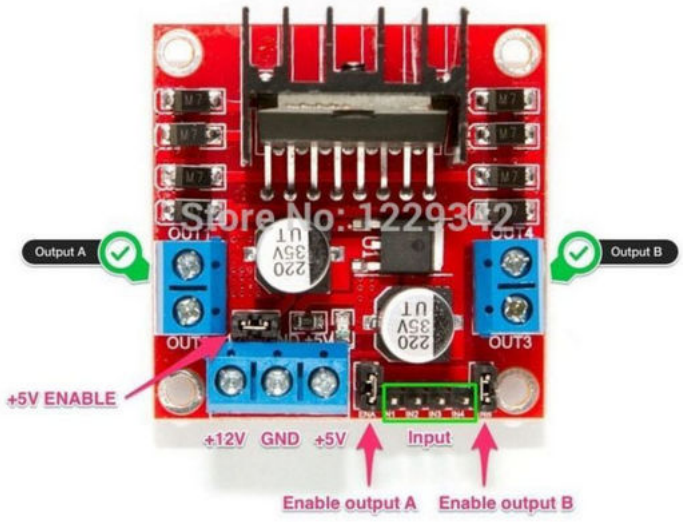
Şekil 4.8: Step motor modülü fonksiyonları



Şekil 4.9: Webcam ve step motorlarla birlikte pan/tilt

4.6 DC Motor ve L298N Motor Sürücü Devresi

L298N motor sürücü modülü bize 2 tane motorun pwm ile ayarlamayı sağlayarak 9-12 volt arasında beslemeyi sağlayan karttır.6 giriş pini bulunur bunlardan 2'si pwm girişi, 4'ü motorların dönüş yönlerini belirlemek içindir.

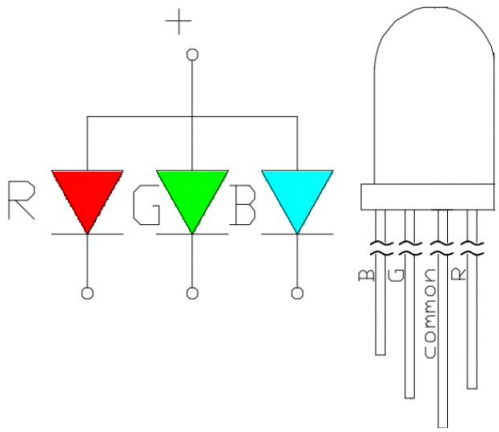


Şekil 4.10: L298N motor sürücü devresi [4]

Şekil 4.11: Dc 250 rpm motor [5]

4.7 RGB Led

3 adet ledin(kırmızı,yeşil,mavi) birleşimi şeklinde çalışmaktadır. 4 pini bulunur. Bir pin besleme diğer üç pini kırmızı, yeşil ve mavi ledlerin toplaklarıdır.



Şekil 4.12: RGB Led [6]

4.8 Buzzer

5v ile beslediğimizde sesle uyarı sinyali veren basit bir cihazdır.5v ve gnd olmak üzere 2 pini bulunur.

Şekil 4.13: Buzzer [7]

4.9 Lipo Pil

Lipo Pil nedir ? İçerisinde Lityum ve Polimer Bulunduran pillere kısaca LiPo pil enir. Günümüzde mp3 çalarlar, modelcilik, navigasyon cihazları gibi elektronik cihazlarda kullanılmaya başlamıştır. Avantajları : Lipo piller Nikel-Kadmiyum (NiCd) ve Nikel-Metal Hidrit (NiMH) pillere göre daha fazla akım üretebilirler. Kıyaslama yapıldığında diğer pillere oranla daha hafiftirler bu da modelcilikte kullanımı açısından bir avantajdır. NiCd ve NiMH pillere göre kullanım süreleri daha uzundur. İstenilen şekilde ve ölçüde üretilebilirler. Dezavantajları : Kullanılmaları ve şarj edilmeleri özel dikkat gerektirir. Özel şarj cihazı gerektirirler. Lipo pillerin içlerini açmamak, içeriğinin hava veya su ile temas etmemesini sağlamak gerekir. Koruyucu kabı dolayısı ile zaten koruma altına alınmış olsalar dahi dikkatli kullanmak gerekir. Lipo piller, verebileceklerinden fazla akım çekildiğinde çok ısınır ve şişerler. Bu sebeple kullanırken pilin kapasitesi aşılmaması yerinde olur. Aksi halde pilin ömrü biter ve kullanılamaz hale gelebilir. Çok fazla şişmiş piller veya darbe almış şekli bozulmuş, dolayısı ile koruyucu kabının yırtılma riski olan piller kullanılmamalı ve uygun yöntemle imha edilmelidirler. Lipo piller kesinlikle kısa devre yaptırılmamalıdır! Aksi halde hem pili kaybetme hem de güvenlik tehlikesi vardır. Saklama koşullarına gelince, mümkünse serin, kuru ve karanlık bir ortamda pilleri muhafaza etmek gerekir. Eğer pilleri uzun süre kullanmayacaksanız yarım şarjlı değerlerde saklamanız pillerinizin ömrünü uzatacaktır. [8]

Şekil 4.14: 11.1v 1050mAh Lipo [9]

4.10 OPENCV

OpenCV, Intel tarafından geliştirilerek BSD lisansı ile lisanslanmış, "Bilgisayarla Görü/Görme" kütüphanesidir. Özellikle gerçek zamanlı uygulamalar hedef alınarak geliştirilmiş olması, ticari kullanımı dahil ücretsiz olması ve Windows, Linux, MacOS X gibi farklı platformlarda kullanılabilmesi bu kütüphaneyi diğer görüntü işleme araçlarından bir adım öne çıkarmaktadır.

CvAux bileşeni, şablon eşleştirme(template-matching), şekil eşleştirme (shape matching), bir objenin ana hatlarını bulma (finding skeletons), yüz tanıma (face-recognition), ağız hareketleri izleme (mouth-tracking), vü- cut hareketlerini tanıma (gesture recognition) ve kamera kalibrasyonu gibi daha pek çok deneysel algoritmaları bünyesinde barındıran kütüphanedir.

OpenCV kütüphanesi, BSD lisansı ile lisanslanmıştır. Özgür lisanslar içinde en özgürü olarak bilinen bu lisansta kodu alan kişi, istediği gibi kullanma özgürlüğüne sahiptir [2]. Akademik ve ticari kullanımı ücretsiz olan bu kü- tüphane Windows, Linux, MacOS X gibi farklı platformlarda kullanılabilir

4.10.1 Opencv kütüphanesinin Linux'ta derlenme işlemi

Opencv Kütüphanesini kullanabilmek için opencv projesini işletim sistemimizde derlememiz gerekmektedir.Derlediğimizde opencv kütüphanesi projemize eklenmiş olucaktır. Raspbian işletim sisteminde derlenmesi biraz sürmektedir.

Derleme aşamaları aşadaki şekilde verilmiştir:

- Raspberry pi'ye opencv proje dosyalarının git ile yüklenmesi.

  -ilk olarak git sisteminin işletim sistemimize yüklenmesi gerekir.

```
$ sudo apt-get install git-all
```

  -sonrasında git ile uzak repodan opencvyi indirmemiz(clonelamamız) gerekiyor.

```
$ cd ~/<opencvyi nereye indirmek istiyorsak o konuma
   gelmeliyiz.>
$ git clone https://github.com/opencv/opencv.git
$ git clone https://github.com/opencv/opencv_contrib.
   git
```

- Opencv Kütüphanesinin cmake ile derlenip sistemimize yüklenmesi.

  -Aşağıdaki komuttaki gibi indirdiğimiz projenin içine girmeliyiz.

```
$ cd ~/opencv
```

  -Devamında build diye bir klasör oluşturuyoruz.

```
$ mkdir build
```

  -Son olarak o klasörün içine girip aşağıdaki komutlarla build ediyoruz.

```
$ cd build
```

```
$ cmake -D CMAKE_BUILD_TYPE=Release -D
   CMAKE_INSTALL_PREFIX=/usr/local ..
$ sudo make install
```

4.10.2 OpenCV kütüphanesinin kullanılması

Opencv Kütüphanesi kurulduktan sonra kütüphaneyi kullanabilmek için hangi aşamalar yapılacağı aşağıda anlatılmaktadır.

Cmake ile c++ kodlarımızı yazıp opencvde de c++ kütüphanesini kullanıcağız bu yüzden projemizdeki CMakeLists.txt 'e opencv kütüphanesini çağırmamız gerekecektir. aşağıdaki şekilde projeye ait CMakeLists.txt verilmiştir.

Not: Kütüphaneyi kullanabilmek için WiringPi Kütüphaneside eklenmiştir.Ekstradan json data ile işlemler yapabilmek için json kütüphanesi mevcuttur.

```
cmake_minimum_required(VERSION 2.8)
project( FaceTracing )

find_package(WiringPi REQUIRED)
find_package( OpenCV REQUIRED )
find_package(PkgConfig REQUIRED)

pkg_check_modules(JSONCPP jsoncpp)

link_libraries(${JSONCPP_LIBRARIES})

include_directories(${WIRINGPI_INCLUDE_DIRS})

add_executable( FaceTracing faceTracker.cpp stepMotor.cpp
   ioToTxt.cpp)

target_link_libraries(FaceTracing ${WIRINGPI_LIBRARIES})
target_link_libraries( FaceTracing ${OpenCV_LIBS} )
target_link_libraries( FaceTracing ${JSONCPP_LIBRARIES} )
```

Şekil 4.15: CMakeFileLists.txt

4.10.3 OpenCV'de görüntü işleme ve tanıma

Opencv'de görüntü işlemek Matlab'ta görüntü işlemeye çok benzer. ilk aşamada giriş resimleri yüklenir sonrasında resime filtreler veya marfolojik işlemler uygulanarak çıkış resim veya yorum verileri elde edilir.

Opencv'de görüntü işlemek için hazır fonksiyonlar kullanılır.Görüntüde nesne tanımak için de hazır fonksiyonlar bulunmaktadır.Bu projede bu fonksiyonları kullarak kameradan alınan görüntüdeki yüzleri tanıma üzerine aşağıdaki proje kodu verilmiştir.

4.10.4 Kameranın OpenCV'de açılması

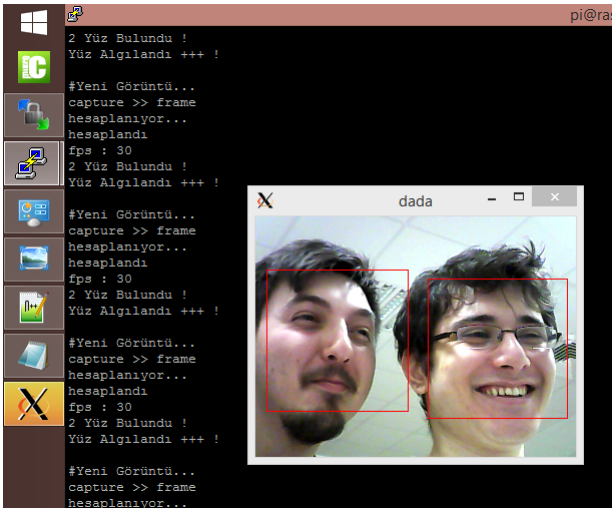
```
        //Kameradan görüntü alabilmek için OpenCV classı
           VideoCapture kullanılıyor.
        VideoCapture capture(0);
        //namedWindow("camera 1",WINDOW_AUTOSIZE);

        //Alınan görüntülerin genişlikle yüksekliği kırpılarak
           alınıyor.
        capture.set(CV_CAP_PROP_FRAME_WIDTH, 300);
        capture.set(CV_CAP_PROP_FRAME_HEIGHT, 300);

        Point captureSize = Point(capture.get(
           CV_CAP_PROP_FRAME_WIDTH),capture.get(
           CV_CAP_PROP_FRAME_HEIGHT));
        Point captureCenter = Point(captureSize.x/2,captureSize.
           y/2);

        //eğer kamera açık değilse hata fonksiyonu çalıştırılıp
           program kapatılıyor.
    if(!capture.isOpened())
    {
        printf("Video file could not be opened !\n");
                return -1;
    }
```

4.10.5 CascadeClassifier örnekleme dosyasının yüklenmesi

Aşağdaki kodta CascadeClassifier classı xml veri dosyasındaki verilerin yüklenmesi gösterilmiştir.

CascadeClassifier classı görüntüdeki nesneleri tanımak için hazırlanmış opencv c++ classıdır.

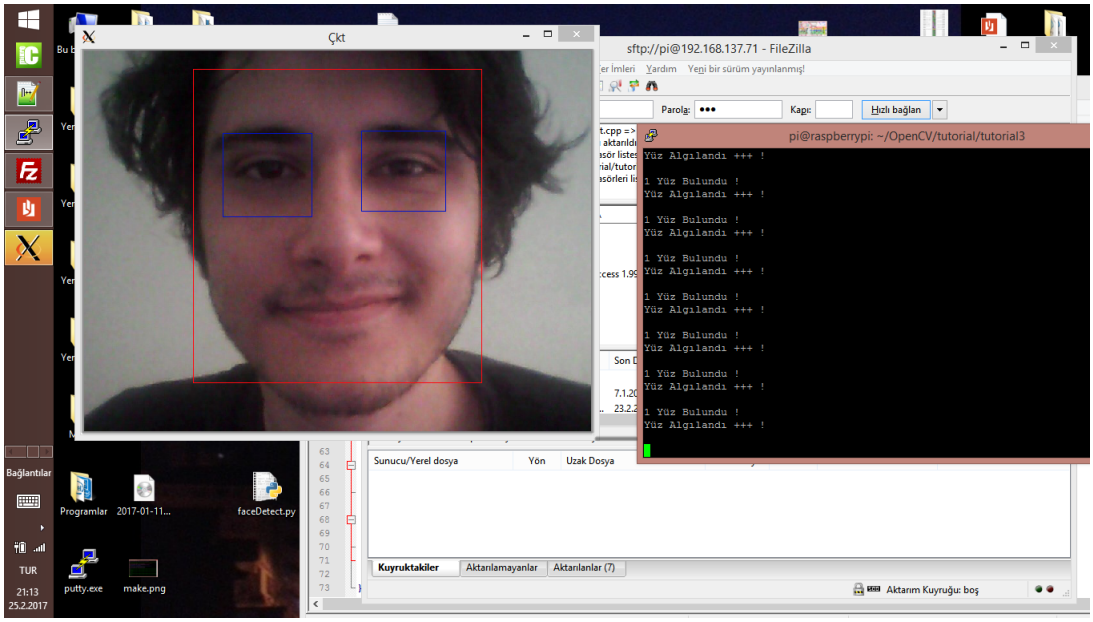
```
        //CascadeClassifier classı cisimleri tanımak için
           yazılmış veri sistemidir.
        //bu class sayesinde yüklenen örnek verilerle nesne
           tespiti yapabiliyoruz.
    CascadeClassifier cascade1,cascade2;
        //Opencv'nin içinde hazır yüz örnek dataları bulunmakta.
        //bu dataları aşağıdaki load fonksiyonu ile
           CascadeClassifier nesnesine yüklüyoruz.
    //eğer yoldaki haarcascade_frontalface_default.xml veri
       dosyası yoksa
        //hata fonksiyonu çalışıp program kapatılıcaktır.
        if( !cascade1.load( "./OpenCV/opencv/data/haarcascades/
           haarcascade_frontalface_default.xml" ) )
    {
        cerr << "Hata : Yüz oran dosyası bulunamadı !!" << endl;
                return -1;
    }
```



Şekil 4.16: OpenCV Yüz Tanıma Çıktısı

4.10.6 Görüntüdeki yüzün algılanması

Kameradan alınan görüntü sürekli bir akış içinde olduğundan while ile sürekli camera-daki resmi alıp,"*yüz var mı ?*" kontrolünü çalıştırıyoruz ve sonrasında bulunan yüzlere eni ve boyuna göre resim üzerine çizdiremeler yapıyoruz.sonuç olarak çıktısını yazdırıyoruz.



Şekil 4.17: OpenCV Yüz Tanıma Çıktısı

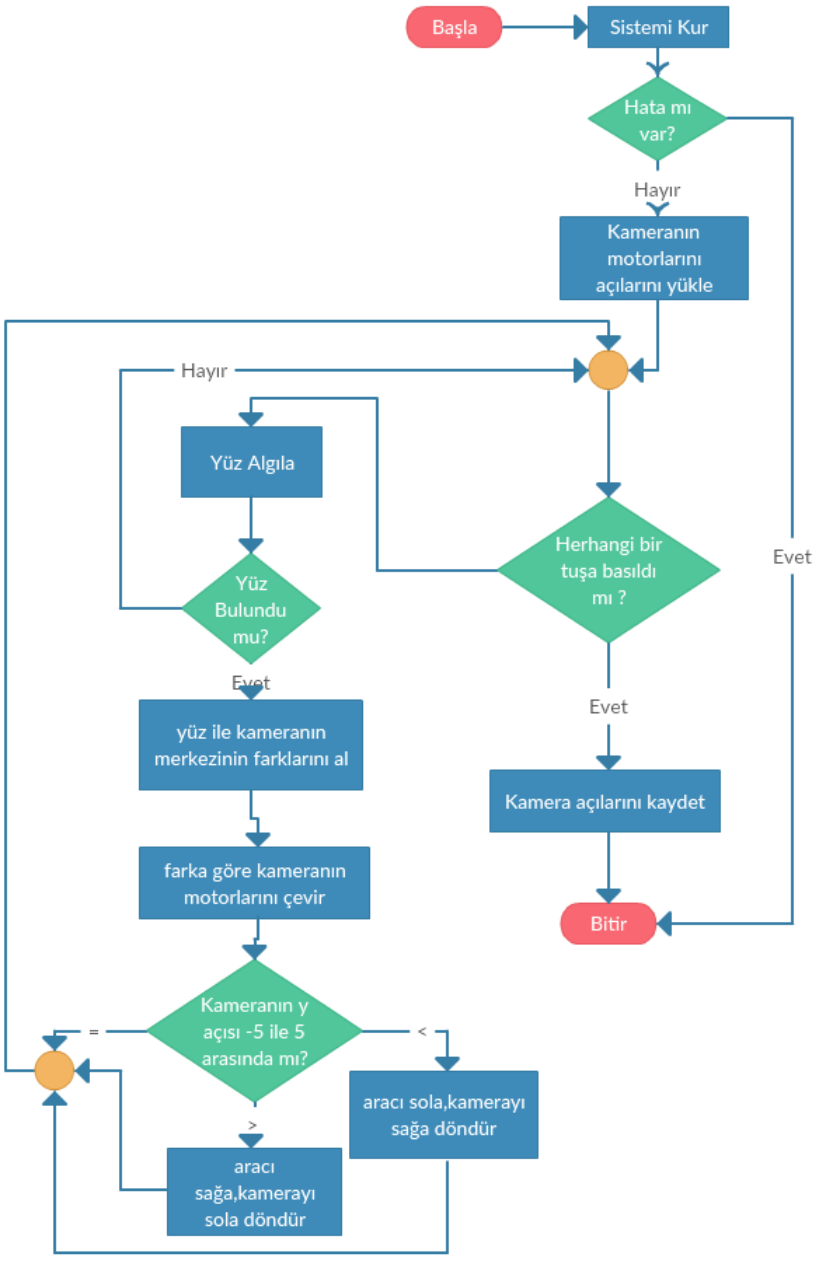
Aşağıdaki kod bu işlemi yapmaktadır.

```
    //Sonsuz bir döngü içinte biz nesne tanımaları
       gerçekleştiriyoruz.
while (true)
{
            printf("#Yeni␣Görüntü...\n");
    capture >> frame;
            printf("capture␣>>␣frame␣\n");
    if (frame.empty()){
                    errorAlarm();
        break;
            }
    //yüz tanıma isteniyorsa
```

```
        if(faceTracking){
                printf("hesaplanıyor...\n");
        vector<Rect> faces;
                vector<int> numbs;
        cascade1.detectMultiScale(frame, faces,1.1, 1, 0,Size
            (40,50));
                digitalWrite(3,LOW);
                printf("hesaplandı\n");
        if(faces.size() > 0){
                        digitalWrite(3,HIGH);
                        rectangle(frame, faces[0], Scalar(0, 0,
                            255));
                        Point faceCenter = Point(faces[0].x +
                            faces[0].width/2 ,faces[0].y  + faces
                            [0].height/2);
                        line(frame,faceCenter,captureCenter,
                            Scalar(255, 0, 0),1,8,0);
                        distance = captureCenter - faceCenter;
                        string strDistance =  "("+ NumToStr(
                            distance.x) + "," + NumToStr(distance
                            .y) + ")" ;
                        putText(frame,strDistance,faceCenter,
                            FONT_HERSHEY_SIMPLEX,.5f,Scalar(255,
                            0, 0),1, 8, false );
                }

                imshow("Çıktı", frame);

                fps = capture.get(CV_CAP_PROP_FPS);

                cout << "fps␣:␣" << fps << endl ;
                printf("%d␣Yüz␣Bulundu␣!\n", faces.size());
        }
```

4.11 Projenin Akış Diyagramı

Projenin tasarımı gereği algortima aşağıda verilmiştir.



Şekil 4.18: Projenin Akış Diyagramı

4.12 Proje İçin Yazılan Cpp Modülleri

WiringPi kütüphanesi kullarak cpp yazılan modüller anlatılacaktır. [Not: Kodlara ait sadece headerlar verilmiştir.Eklerde içeriklerini bulabilirsiniz.]

4.12.1 StepMotor sınıfı

Step motorların Raspberry Pi için entegresini basitleştirmek amaçlı yazılmıştır.

```
//----------------------StepMotor.h----------------------
#ifndef StepMotor_H
#define StepMotor_H
class StepMotor{
        public :
                void setup(void);//stepMotoru kurmak için
                   kullanırız. burada wiringPi pinleri
                   tanımlıdır.
                void clearPins(void);//motorun 5v olan Pinlerini
                    sıfırlar.
                void turnRight(int);//motoru giriş parametresi
                   bekletme aralığına göre sağ döndürür.
                void turnLeft(int);//motoru giriş parametresi
                   bekletme aralığına göre sola döndürür.
                StepMotor(int,int,int,int); //kurucu sınıftır.
                   stepMotor sürücü devresinin pinlerini gireriz
                   .
        private :
                int in0,in1,in2,in3;
};
#endif
```

4.12.2 L298N sınıfı

DC motor Sürücü devresi olan L298N ,Raspberry Pi de kullanımını kolaylaştırmak için sınıf yazılmıştır.

```
#ifndef _L298N_H_INCLUDED_
#define _L298N_H_INCLUDED_
#include<wiringPi.h>
#include <softPwm.h>
#include <iostream>
using namespace std;
class L298N{
        public :
                L298N(int,int,int,int);
                   //L298N(motor1_IN1,motor1_IN2,motor2_IN1,
                   motor2_IN2)
                L298N(int,int,int,int,int,int);          //L298N(
                   motor1_IN1,motor1_IN2,motor2_IN1,motor2_IN2,
                   motor1_pwm,motor2_pwm)
                void setPwm1(int);
                           //Eğer pwm ile kurulum yaptıysan
                   motor1 için burdan pwm verebilirsin.
                void setPwm2(int);
                           //Eğer pwm ile kurulum yaptıysan
                   motor2 için burdan pwm verebilirsin.
                void forwardMotor1();
                   //motor1'i ileri döndürmek için
                void backMotor1();
                           //motor1'i geri döndürmek için
                void forwardMotor2();
                   //motor1'i ileri döndürmek için
                void backMotor2();
                           //motor1'i geri döndürmek için
        private:
```

```
                int m11,m12,m21,m22;
                int pwm1,pwm2;
                bool usePwm;
};
#endif
```

Şekil 4.19: Projenin son durumu

5 SONUÇLAR VE ÖNERİLER

Kamerada mini servo motorların kullanımı da test edildi ve mini servo motorların kamera sabitlemesinde titremeler nedeniyle sorunlarla karşılaşıldı fakat iyi bir servo motor kullanılması durumunda kameranın dönüşünü ayarlamak için yeterli geleceği düşünüldü.

Projenin amaçı olan insan vücudunu algılama işlemi OpenCV'nin fonksiyonları verimli çalışamadı. Projenin devamında görüntü işleme konusunda iyi bir bilgi sahibi olup, projede geliştirmeler yapılabilir.

Projede kullanılan lipo bataryaların verimliliğin düşmesi lipo bataryaya zarar verdiği öğrenildi. Bu aşamada projedeki lipo bataryasının bitme aşamasında olduğunu söyleyen bir Ardunio gibi kartla haberleştirme sağlanabilir.

Sonuç olarak, Raspberry Pi kartı üzerinde yapmak isteğimiz kişi takip sistemi başarılı bir şekilde çalıştı fakat kullanılan malzemelerin mali değeri düşük olduğundan verimli başarım elde edilmedi.

Kişi Takip sistemlerine gelişen teknoloji ile ihtiyaç duyulacak ve bu alanda çalışmalar yapılacaktır. Bu projenin devamında görüntü işleme ve tanıma algoritlarını bulunduran opencv, daha hızlı çalışacak şekilde projeye uygun olarak optimize edilebilir. Raspberry Pi IoT sistemlere en uygun cihazdır. Android telefonlar gibi küçük ve üzerinde gömülü sistem çalışabilicek şekilde tasarlanmıştır. Bu projeyle ulaşılmak istenen performansa ulaşılamasa da uygun işlemcili sistemler kullanılarak ulaşılabilir.

İleride bu projeyi geliştirme olarak yüz tanıma kullanılabilir fakat çalışma verimliliği konusunda sorunlar yaratabilir. Bu yüzden iyi bir grafik işlemcisi olan 950mb/s işlemci hızından daha yüksek kartlarda test edilip uygulanabilir. Kişi takibi için geliştirilmiş olan araç daha iyi motorlar ve kasa ile güçlendirilebilir. Tekerlek sayısıs artırılabilir böylelikle hareketi daha verimli sağlanabilir. Ardunio gibi kartlar ile iletişim kurdurulabilir.

6 EKLER

6.1 FindWiringPi.cmake

```
find_library(WIRINGPI_LIBRARIES NAMES wiringPi)
find_path(WIRINGPI_INCLUDE_DIRS NAMES wiringPi.h)

include(FindPackageHandleStandardArgs)
find_package_handle_standard_args(wiringPi DEFAULT_MSG
WIRINGPI_LIBRARIES WIRINGPI_INCLUDE_DIRS)
```

6.2 CMakeLists.txt

```
cmake_minimum_required(VERSION 2.8)
project( FaceTracing )

find_package(WiringPi REQUIRED)
find_package( OpenCV REQUIRED )
find_package(PkgConfig REQUIRED)

pkg_check_modules(JSONCPP jsoncpp)

link_libraries(${JSONCPP_LIBRARIES})

include_directories(${WIRINGPI_INCLUDE_DIRS})

add_executable( FaceTracing faceTracker.cpp stepMotor.cpp
   ioToTxt.cpp)

target_link_libraries(FaceTracing ${WIRINGPI_LIBRARIES})
target_link_libraries( FaceTracing ${OpenCV_LIBS} )
target_link_libraries( FaceTracing ${JSONCPP_LIBRARIES} )
```

6.3 faceTracker.cpp

```
#!/usr/bin/python
#--------------------------------------
# lcd_16x2.py
#
# Author : Efecan Altay
# Date   : 18/10/2016
#
# Copyright 2017 Efecan Altay
#
# This program is free software: you can redistribute it and/or
   modify
# it under the terms of the GNU General Public License as
   published by
# the Free Software Foundation, either version 3 of the License,
    or
# (at your option) any later version.

#include <opencv2/opencv.hpp>
#include "./stepMotor.h"
#include "ioToTxt.h"

#include <iostream>
#include <string>
#include <unistd.h>
#include <signal.h>

#include<wiringPi.h>

using namespace cv;
using namespace std;
```

```
int yRadius = 0;//y motorun açısı
int xRadius = 0;//x motorun açısı
string dataDoc = "./data/motorsData.txt"; //motorların açıları
   dosyaya kaydediliyor.
int isAlarm = 1;//uyarı sesleri açıksa 1 olucak

int redLed = 1 ;//kırmızı led pin Numarası
int greenLed = 2 ;
int blueLed = 3 ;

ioToTxt dataIO; //dosyada kayıt tutmak için yazdığım modül

//StringDönüşümü----------------------------------------
//sstream kütüphanesi yüklü olaması gerek.
#include <sstream>
//konu linki: http://www.cplusplus.com/articles/D9j2Nwbp/

StepMotor yStep(25,24,23,22);//step Modüllerinin nesneleri
   oluşturuluyor.
StepMotor xStep(29,28,27,26);//
   ----------------------------------------

//int bir sayıyı string dönüşümü için kullanılan fonksiyon
template <typename T>
  string NumToStr ( T Number )
  {
     ostringstream ss;
     ss << Number;
     return ss.str();
  }
//---------------------------------------
void startAlarm(){ //start alarmı cihaz başlarken bir kez
   yapılan kurulum bölümü
```

```
        if(isAlarm){
                wiringPiSetup(); //wiringPi kütüphanesinin
                    kurulumu
                pinMode(1,OUTPUT);      //--Buzer için
                    kullanılan pin 1
                pinMode(0,OUTPUT);      //--RGB lediçin pinler
                    ayırılıyor
                pinMode(2,OUTPUT);      //- 2
                pinMode(3,OUTPUT);      //- 3
                digitalWrite(0,HIGH); //-Pinlere 5V giriş
                    veriliyor HIGH=1/LOW=0
                digitalWrite(2,HIGH); //---
                digitalWrite(3,HIGH); //---
                for(int i= 0;i < 3;i++){
                        digitalWrite(1,HIGH);
                        digitalWrite(2,LOW);
                        delay(50);
                        digitalWrite(1,LOW);
                        digitalWrite(2,HIGH);
                        delay(50);
                }
        }
}
void errorAlarm(){ //Bir Hata meydana geldiğinde çalışan fonk.
        if(isAlarm){
                for(int i= 0;i < 3;i++){
                        digitalWrite(1,HIGH);
                        delay(500);
                        digitalWrite(1,LOW);
                        delay(500);
                }
        }
}
```

```
void stopAlarm(){        //program durdurulduğunda çalışan alarm
   fonksiyonu.
        if(isAlarm){
                for(int i= 1;i <= 2;i++){
                        digitalWrite(1,HIGH);
                        delay(100/i);
                        digitalWrite(1,LOW);
                        delay(100/i);
                }
        }
}
void LoadData(){ //Cihaz açılırken motor dönüş açılarını
   yüklemek için kullanılan fonk.
        jsonData data = dataIO.LoadToFile(dataDoc);
        if(data == 0)
                data = dataIO.SaveToFile(dataDoc,xRadius,yRadius
                   );
        xRadius = data["xMotorRadius"].asInt();
        yRadius = data["yMotorRadius"].asInt();
}
void SaveData(){ //Cihazda motor açılarını dosyaya yazmak için
   kulanılan fonk.
        dataIO.SaveToFile(dataDoc,xRadius,yRadius);
}
void closingProgram(){ //program kapatıldığında çalışan fonk.
   stopAlarm();
   yStep.clearPins();
   xStep.clearPins();
   digitalWrite(3,HIGH);
   digitalWrite(3,HIGH);
   digitalWrite(3,HIGH);
   SaveData();
}
```

```
//programdan çıkıldığında kapatıldığında çalışan callback
   fonksiyonu
void signal_callback_handler(int signum)
{
   printf("Caught signal %d\n",signum);
   closingProgram();
   exit(signum);
}
//Step Motorları merkeze çekmek(motor verilerine bakarak) için
   kullanılan fonk.
//Sadece program başlarken kullanılır.
void setPosToOrigin(){
        while(1)
        {
                if(xRadius > 0){
                        xStep.turnLeft(5);
                        xRadius--;
                }else if(xRadius < 0){
                        xStep.turnRight(5);
                        xRadius++;
                }
                if(yRadius > 0){
                        yStep.turnRight(5);
                        yRadius--;
                }else if(yRadius < 0){
                        yStep.turnLeft(5);
                        yRadius++;
                }
                if(xRadius == 0 && yRadius == 0){
                        break;
                }
        }
}
```

```
//Ana fonksiyon.
int main(int argc, char** argv)
{
        startAlarm();
        LoadData();

        //program kapatıldığında callback fonksiyonu çağırmak
           için kullanılır.
        signal(SIGINT, signal_callback_handler);

        //algılanan cisimle cameranın merkezini tutan data.
        Point distance ;

        //yetki komutu sudo ile yetki vermeksizin progeyi
           çalıştırabiliyoruz.
        setenv("WIRINGPI_GPIOMEM", "1", 1);

        yStep.setup();//step Motorların kurulum fonksiyonları
        xStep.setup();//-------------------------------------

        setPosToOrigin();//başalarken kamera merkeze çekiliyor.(
           kaydedilen verilere bakılarak)

        //Kameradan görüntü alabilmek için OpenCV classı
           VideoCapture kullanılıyor.
        VideoCapture capture(0);
        //namedWindow("camera 1",WINDOW_AUTOSIZE);

        //Alınan görüntülerin genişlikle yüksekliği kırpılarak
           alınıyor.
        capture.set(CV_CAP_PROP_FRAME_WIDTH, 300);
        capture.set(CV_CAP_PROP_FRAME_HEIGHT, 300);
```

```
    Point captureSize = Point(capture.get(
        CV_CAP_PROP_FRAME_WIDTH),capture.get(
        CV_CAP_PROP_FRAME_HEIGHT));
    Point captureCenter = Point(captureSize.x/2,captureSize.
        y/2);

    //eğer kamera açık değilse hata fonksiyonu çalıştırılıp
        program kapatılıyor.
if(!capture.isOpened())
{
    printf("Video␣file␣could␣not␣be␣opened␣!\n");
    errorAlarm();
            return -1;
}
    double fps = 0 ;
Mat frame;
int frame_count = 0;
    //CascadeClassifier classı cisimleri tanımak için
        yazılmış veri sistemidir.
    //bu class sayesinde yüklenen örnek verilerle nesne
        tespiti yapabiliyoruz.
CascadeClassifier cascade1,cascade2;
    //Opencv'nin içinde hazır yüz örnek dataları bulunmakta.
    //bu dataları aşağıdaki load fonksiyonu ile
        CascadeClassifier nesnesine yüklüyoruz.
//eğer yoldaki haarcascade_frontalface_default.xml veri
   dosyası yoksa
    //hata fonksiyonu çalışıp program kapatılıcaktır.
    if( !cascade1.load( "./OpenCV/opencv/data/haarcascades/
        haarcascade_frontalface_default.xml" ) )
{
    cerr << "Hata␣:␣Yüz␣oran␣dosyası␣bulunamadı␣!!" << endl;
    errorAlarm();
```

```
            return -1;
}
    //Sonsuz bir döngü içinte biz nesne tanımaları
       gerçekleştiriyoruz.
while (true)
{
            printf("#Yeni␣Görüntü...\n");
    capture >> frame;
            printf("capture␣>>␣frame␣\n");
    if (frame.empty()){
                    errorAlarm();
        break;
            }
            printf("hesaplanıyor...\n");
    vector<Rect> faces;
            vector<int> numbs;
    cascade1.detectMultiScale(frame, faces,1.1, 1, 0,Size
       (40,50));
            digitalWrite(3,LOW);
            printf("hesaplandı\n");
    if(faces.size() > 0){
                    digitalWrite(3,HIGH);
                    //yüzlere dikdörtgen çizdiriliyor.
                    rectangle(frame, faces[0], Scalar(0, 0,
                       255));
                    //yüzlerin merkezleri hesaplanıp
                       faceCenter noktasında tutuluyor.
                    Point faceCenter = Point(faces[0].x +
                       faces[0].width/2 ,faces[0].y  + faces
                       [0].height/2);
                    //yüzün merkeziyle cameranın merkezi
                       arasında çizgi çizdiriliyor.
```

```
line(frame,faceCenter,captureCenter,
   Scalar(255, 0, 0),1,8,0);
//yüzün merkeziyle kameranın arasındaki
   mesafe hesaplanıyor.
distance = captureCenter - faceCenter;
//mesafenin x ve y si string ile
   tutuluyor.
string strDistance =  "("+ NumToStr(
   distance.x) + "," + NumToStr(distance
   .y) + ")" ;
//mesafe görüntüye yazdırılıyor.
putText(frame,strDistance,faceCenter,
   FONT_HERSHEY_SIMPLEX,.5f,Scalar(255,
   0, 0),1, 8, false );
//mesafelere göre motorlar döndürülüyor.
if(distance.x > 10){
        yStep.turnLeft(5);
        yRadius++;
        putText(frame,"zmotor␣:on␣<-",
           Point(0,captureSize.y-10),
           FONT_HERSHEY_SIMPLEX,.5f,
           Scalar(255, 0, 0),1, 8, false
            );
}
else if(distance.x < -10){
        yStep.turnRight(5);
        yRadius--;
        putText(frame,"zmotor␣:on␣->",
           Point(0,captureSize.y-10),
           FONT_HERSHEY_SIMPLEX,.5f,
           Scalar(255, 0, 0),1, 8, false
            );
}else{
```

```
                    putText(frame,"zmotor␣:on␣-",
                        Point(0,captureSize.y-10),
                        FONT_HERSHEY_SIMPLEX,.5f,
                        Scalar(255, 0, 0),1, 8, false
                         );
                }
                if(distance.y > 10){
                        xStep.turnRight(5);
                        xRadius++;
                        putText(frame,"xmotor␣:on␣^",
                        Point(0,captureSize.y-20),
                        FONT_HERSHEY_SIMPLEX,.5f,
                        Scalar(255, 0, 0),1, 8, false
                         );
                }
                else if(distance.y < -10){
                        xStep.turnLeft(5);
                        xRadius--;
                        putText(frame,"xmotor␣:on␣v",
                        Point(0,captureSize.y-20),
                        FONT_HERSHEY_SIMPLEX,.5f,
                        Scalar(255, 0, 0),1, 8, false
                         );
                }
                else{
                        putText(frame,"xmotor␣:on␣-",
                        Point(0,captureSize.y-20),
                        FONT_HERSHEY_SIMPLEX,.5f,
                        Scalar(255, 0, 0),1, 8, false
                         );
                }

            }else{
```

```
                        //hiç bir yüz bulunmadıysa motorların
                            pinleri kapatılıyor.
                        yStep.clearPins();
                        xStep.clearPins();
                        putText(frame,"zmotor␣:off",Point(0,
                            captureSize.y-10),
                            FONT_HERSHEY_SIMPLEX,.5f,Scalar(255,
                            0, 0),1, 8, false );
                        putText(frame,"xmotor␣:off",Point(0,
                            captureSize.y-20),
                            FONT_HERSHEY_SIMPLEX,.5f,Scalar(255,
                            0, 0),1, 8, false );
                }
                //Çıktı ekranda görüntüleniyor.
                imshow("Çıktı", frame);

                fps = capture.get(CV_CAP_PROP_FPS);

                cout << "fps␣:␣" << fps << endl ;
                printf("%d␣Yüz␣Bulundu␣!\n", faces.size());
                printf("\n␣X:␣%d␣" , xRadius);
                printf("Y:␣%d␣\n" , yRadius);
                //herhangi bir tuşa basıldıysa programı kapat
        if (waitKey(1) >= 0){
                    closingProgram();
           break;
                }
    }//while sonu
    return (0);
}//main sonu
```

6.4 StepMotor.cpp

```
#include "wiringPi.h"



class StepMotor{
        public :
                void setup(void);
                void clearPins(void);
                void turnRight(int);
                void turnLeft(int);
                void turnFastLeft(int);
                void turnFastRight(int);
                StepMotor(int,int,int,int);

        private :
                int in0,in1,in2,in3;

};
StepMotor::StepMotor(int _in0,int _in1 ,int _in2,int _in3 ){
        in0 = _in0;
        in1 = _in1;
        in2 = _in2;
        in3 = _in3;
}

void StepMotor::setup(){
        //setenv("WIRINGPI_GPIOMEM", "1", 1);
        wiringPiSetup();
        //Birinci Motorun girişleri Z eksen dönüşü
        pinMode(in0,OUTPUT);
        pinMode(in1,OUTPUT);
        pinMode(in2,OUTPUT);
```

```
        pinMode(in3,OUTPUT);
        }
void StepMotor::turnRight(int speed_milisec){
                digitalWrite(in0,HIGH);
                digitalWrite(in1,LOW);
                digitalWrite(in2,LOW);
                digitalWrite(in3,LOW);
                delay(speed_milisec);
                digitalWrite(in0,LOW);
                digitalWrite(in1,HIGH);
                digitalWrite(in2,LOW);
                digitalWrite(in3,LOW);
                delay(speed_milisec);
                digitalWrite(in0,LOW);
                digitalWrite(in1,LOW);
                digitalWrite(in2,HIGH);
                digitalWrite(in3,LOW);
                delay(speed_milisec);
                digitalWrite(in0,LOW);
                digitalWrite(in1,LOW);
                digitalWrite(in2,LOW);
                digitalWrite(in3,HIGH);
                delay(speed_milisec);
}
void StepMotor::turnLeft(int speed_milisec){
                digitalWrite(in0,HIGH);
                digitalWrite(in1,LOW);
                digitalWrite(in2,LOW);
                digitalWrite(in3,LOW);
                delay(speed_milisec);
                digitalWrite(in0,LOW);
                digitalWrite(in1,LOW);
                digitalWrite(in2,LOW);
```

```
                digitalWrite(in3,HIGH);
                delay(speed_milisec);
                digitalWrite(in0,LOW);
                digitalWrite(in1,LOW);
                digitalWrite(in2,HIGH);
                digitalWrite(in3,LOW);
                delay(speed_milisec);
                digitalWrite(in0,LOW);
                digitalWrite(in1,HIGH);
                digitalWrite(in2,LOW);
                digitalWrite(in3,LOW);
                delay(speed_milisec);
}
void StepMotor::turnFastRight(int speed_milisec){
                        digitalWrite(in0,LOW);
                        digitalWrite(in1,LOW);
                        digitalWrite(in2,HIGH);
                        digitalWrite(in3,LOW);
                        delay(speed_milisec);
                        digitalWrite(in0,HIGH);
                        digitalWrite(in1,LOW);
                        digitalWrite(in2,LOW);
                        digitalWrite(in3,LOW);
                        delay(speed_milisec);
}
void StepMotor::turnFastLeft(int speed_milisec){
                        digitalWrite(in0,LOW);
                        digitalWrite(in1,HIGH);
                        digitalWrite(in2,LOW);
                        digitalWrite(in3,LOW);
                        delay(speed_milisec);
                        digitalWrite(in0,LOW);
                        digitalWrite(in1,LOW);
```

```
                        digitalWrite(in2,LOW);
                        digitalWrite(in3,HIGH);
                        delay(speed_milisec);
}
void StepMotor::clearPins(){
        digitalWrite(in0,LOW);
        digitalWrite(in1,LOW);
        digitalWrite(in2,LOW);
        digitalWrite(in3,LOW);
}
```

6.5 L298N.cpp

```
#include "l298n.h"

L298N::L298N(int _m11,int _m12,int _m21,int _m22){
        m11 = _m11;m12 = _m12;m21 = _m21;m22 = _m22;
        usePwm = false ;
        wiringPiSetup();
        pinMode(m11,OUTPUT);pinMode(m12,OUTPUT);pinMode(m21,
           OUTPUT);pinMode(m22,OUTPUT);

}
L298N::L298N(int _m11,int _m12,int _m21,int _m22,int _pwm1,int
   _pwm2){
        m11 = _m11;m12 = _m12;m21 = _m21;m22 = _m22;
        pwm1 = _pwm1;pwm2 = _pwm2;
        usePwm = true ;
        wiringPiSetup();
        pinMode(m11,OUTPUT);pinMode(m12,OUTPUT);pinMode(m21,
           OUTPUT);pinMode(m22,OUTPUT);
        if(softPwmCreate(_pwm1,0,100) != 0){
```

```
                cerr << "Pwm1 Kurulamadı"<< endl;
        }
        if(softPwmCreate(_pwm2,0,100) != 0){
                cerr << "Pwm2 Kurulamadı"<< endl;
        }
}
void L298N::setPwm1(int duty){
        if(usePwm){
                 softPwmWrite(pwm1,duty);
        }else{
                cout << "Pwm Kullanmadın" << endl;
        }
}
void L298N::setPwm2(int duty){
        if(usePwm){
                softPwmWrite(pwm1,duty);
        }else{
                cout << "Pwm Kullanmadın" << endl;
        }
}
void L298N::forwardMotor1(){
        digitalWrite (m11, 1) ;
        digitalWrite (m12, 0) ;
}
void L298N::backMotor1(){
        digitalWrite (m11, 0) ;
        digitalWrite (m12, 1) ;
}
void L298N::forwardMotor2(){
        digitalWrite (m21, 1) ;
        digitalWrite (m22, 0) ;
}
void L298N::backMotor2(){
```

```
        digitalWrite (m21, 0) ;
        digitalWrite (m22, 1) ;
}
```

KAYNAKLAR

[1] Wikipedia,
https://tr.wikipedia.org/wiki/Raspberry_Pi [Ziyaret Tarihi 3 Mayıs 2017]

[2] StepMotor,
http://www.elektrikrehberiniz.com/elektrik-motorlari/step-motor-nedir-543/ [Ziyaret Tarihi: 21 Mayıs 2017]

[3] StepMotor3,http://www.instructables.com/id/Tracked-Robot-IR-Remote-Control-by-Arduino/ [Ziyaret Tarihi 3 Mayıs 2017]

[4] MotorSürücüDevresi,
http://www.instructables.com/id/Tracked-Robot-IR-Remote-Control-
[Ziyaret Tarihi 3 Mayıs 2017]

[5] MotorSürücüDevresi,
http://www.robotistan.com/6v-250-rpm-motor-ve-tekerlek-seti-321
jpg [Ziyaret Tarihi 3 Mayıs 2017]

[6] RgbLed,
https://cdn.solarbotics.com/products/photos/84728eb26b0947ef6547884cea645502/LED-RGB-8CD5kCA.jpg
[Ziyaret Tarihi 21 Mayıs 2017]

[7] Buzzer,
https://www.fabtolab.com/5V-piezo-buzzer [Ziyaret Tarihi 21 Mayıs 2017] https://kaankandemir.wordpress.com/2013/04/20/arduino-ile-step-motor-kontrolu/ [Ziyaret Tarihi: 21 Mayıs 2017]

[8] 5v1050mAhLipo,
https://www.youtube.com/watch?v=UVA_7d6zyyI[Ziyaret Tarihi: 30 Mayıs 2017]

[9] 5v1050mAhLipo,
http://www.robotistan.com/11lv-lipo-batarya-1050mah-25c-15762-1.jpg [Ziyaret Tarihi: 21 Mayıs 2017]

[10] StepMotor2,
http://www.merakliyizbiz.com/2016/12/step-motor-nedir.html [Ziyaret Tarihi: 21 Mayıs 2017]

[11] OpenCVKurulumu,
http://docs.opencv.org/trunk/d7/d9f/tutorial_linux_install.html [Ziyaret Tarihi : 22 Mayıs 2017]

[12] EfecanAltay,
https://github.com/EfecanAltay/RaspberryPi_Modules [Ziyaret Tarihi : 30 Mayıs 2017]

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı :Efecan Altay

Uyruğu :T.C.

Doğum Yeri ve Tarihi:İstanbul 31/07/1995

Adres :

Telefon :05459690960

e-mail :efecan95.gs@gmail.com

EĞİTİM DURUMU

Lisans Öğrenimi : BŞEÜ Bilgisayar Mühendisliği Bölümü

Bitirme Yılı :2017

Lise :Hacı Hatice Bayraktar Lisesi,Kartal/İstanbul

İLGİ ALANLARI:

Uygulama Geliştirme :Mobil Programlama,Oyun Programlama

Veritabanları :Sql,MongoDB

Gömülü Sistemler :Ardunio,RaspberryPi

İşletim Sistemleri :Linux,Windows

YABANCI DİLLER:

ingilizce