**OTOMATİK ARABA AYNA KONTROLÜ**

**Giriş**

Bu projede şöför koltuğunun konumuna göre yan aynaların  otomatik olarak ayarlanmasını düşünülmüştür. Karşılaşılan sorun , şöför koltuğuna oturan her kişi koltuğun direksiyona olan mesafesini kendine göre ayarlıyor buna bağlı olarakta yan aynanın ayarı yapmak zorunda kalıyor. Kimi zamanda ayna ayarını unutarak yola çıkıyor ve sonradan seyir halindeyken düzeltiyor . Yapılan bu projede hem ayna ayarlama zahmetinden kurtarıcak hemde ayna ayarının unutulmasının önüne geçilecektir. Bu projede iki tane servo motor, bir tane mesafe sensörü, bir tanede raspberry pi 3 kullanılmıştır.

Bu projede tasarlanan mekanizma üzerinde konum kontrolünü sağlayacak 2 servo motor kullanılmıştır. Ultrasonik sensör koltuk konumunu algılayarak raspberry pi 3 aracılığıyla koltuk konum bilgisini motorlara gönderir. Ve motorlar uygun açıya giderler.

**Gerekli Donanım Bileşenleri**

1. 1 adet raspberry pi 3
2. 2 adet servo motor
3. 1 adet ultrasonik sensör

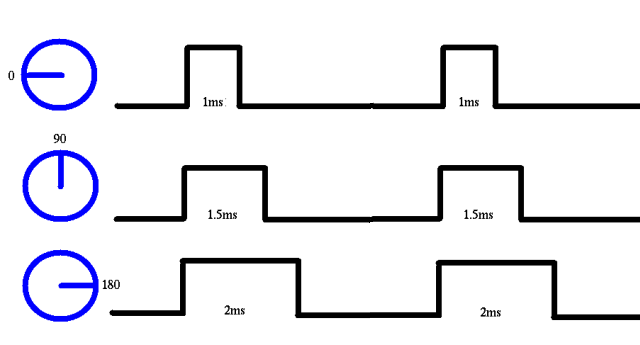
**Gerekli Yazılım Bileşenleri**

1. Python

**Kullanılan Bileşeklerin Özellikleri**

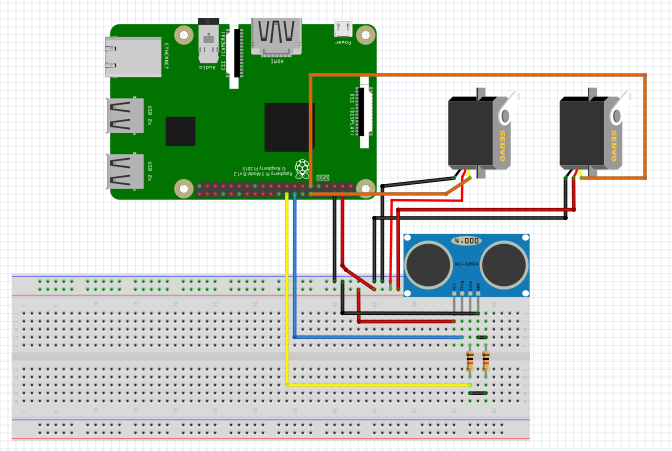
**Raspberry Pi 3:** Foundation tarafından 2009’da geliştirilmeye başlanmış kredi kartı büyüklüğündeki tek board’dan oluşmuş tam donanımlı bir mini-bilgisayardır.Raspberry Pi 2'de olduğu gibi Raspberry Pi 3'de de Broadcom marka Soc (System -On-Chip) kullanılmış durumda. Raspberry Pi 2'de 32Bit BCM2836'ın yerini Raspberry Pi 3'de 64Bit BCM2837 alıyor. 1.2GHz ARMv8 dört çekirdek işlemciye sahip Raspberry Pi 3, Raspberry Pi 2'ye oranla artık çok daha hızlıdır.

**Servo Motor:** Servo motorlar, bildiğimiz üzere PWM ile çalışırlar. Çoğunlukla 0-180 derece arası açılarda çalışırlar. RC arabamızın direksiyonunda, helikopterlerin pervanelerine açı vermede ve uçakların kontrol yüzeylerini hareket ettirmede kullanılırlar.



**Ultrasonik Sensör:** Ultrasonik sensörler, mesafe ölçümlerinde yaygın olarak kullanılmaktadır. TRIG pininden sinyal verildiğinde 40kHz frekansında bir ses dalgası sensör tarafından üretilir ve bu ses dalgası bir cisme çarpıp geri döndüğünde ECHO pini aktif hale gelir. Sesin havada yayılma hızını bildiğimiz için (deniz seviyesinde ve 21 °C sıcaklıkta 343.2 m/s), TRIG pinine verilen sinyalden sonra ECHO pininin aktif olduğu zamana kadarki süreyi ölçerek aradaki mesafeyi kolayca hesaplayabiliriz.

**Şematik Çizimi:**

****

**Nasıl Kullanılır:** Şöför koltuğunun direksiyona olan mesafesinin ayarlanmasıyla ayna otomatik olarak koltuk pozisyonuna göre ayarlanır.

**Yapım Aşamaları:**

1.Mekanik tasarımın tasarlanması.

2.Kullanılacak Malzemelerin Belirlenmesi ; Mekanik tasarım, Ayna.

3.Elektronik kısmının tamamlanması ; Raspberry pi 3 , ultrasonik sensör, servo motor.

4.Yazılımın tamamlanması ; Python kodlarının yazılması.

**Python Kodu:**

import RPi.GPIO as GPIO # GPIO kütüphanesi dahil edilir.

import time # time kütüphanesi dahil edilir.

GPIO.setmode(GPIO.BCM) # BCM pin numaralandırmısa kullanılır

GPIO.setwarnings(False) # GPIO uyarı mesajları kapatılır

GPIO.setup(18, GPIO.OUT) # 18 numaralı pin çıkış pini olarak tanımlanır

GPIO.setup(17, GPIO.OUT)

pwm = GPIO.PWM(18, 100) # 18 numaralı pinden 100 Hz frekansında pwm sinyali alınır

pwm2= GPIO.PWM(17, 100)

pwm.start(5)

pwm2.start(5) # %5 duty sycle seçilir

GPIO.setwarnings(False)

TRIG = 23 # Raspberry pi’nin 23 numaralı GPIO pinine sensörümüzün TRIG pini bağlanır

ECHO = 24

print ("HC-SR04 mesafe sensoru")

GPIO.setup(TRIG,GPIO.OUT) # TRIG pini çıkış pini olarak tanımlanır

GPIO.setup(ECHO,GPIO.IN) # ECHO pini giriş pini olarak tanımlanır

while True:

GPIO.output(TRIG, False) # TRIG pini logic 0 yapılır

print ("Olculuyor...")

time.sleep(5)

GPIO.output(TRIG, True) # TRIG pini logic 1 yapılır

time.sleep(0.00001)

GPIO.output(TRIG, False)

while GPIO.input(ECHO)==0:

pulse\_start = time.time() # ECHO pininin logic 0 olduğu süre

while GPIO.input(ECHO)==1:

pulse\_end = time.time() # ECHO pininin logic 1 olduğu süre

pulse\_duration = pulse\_end - pulse\_start #darbe süresi hesabı

distance = pulse\_duration \* 17150 # mesafe hesabı

distance = round(distance, 2) # virgülden sonra 2 basamak olucak şekilde yuvarlama

if distance > 1 and distance < 8: # direksiyona en uzak mesafe

duty = float(160) / 10.0 + 2.5 # Kare dalganın boyutu belirlenir

pwm.ChangeDutyCycle(duty) # Gerekli pwm sinyali üretilir

duty2 = float(170) / 30.0 + 2.5

pwm2.ChangeDutyCycle(duty2)

print ("1. servo : 0 DERECE")

print ("Mesafe:",distance - 0.5,"cm")

print ("2. servo : 0 DERECE")

elif distance >=8 and distance <16:

duty = float(164) / 10.0 + 2.5

pwm.ChangeDutyCycle(duty)

duty2 = float(160) / 30.0 + 2.5

pwm2.ChangeDutyCycle(duty2)

print ("1. servo : 10 DERECE")

print ("Mesafe:",distance - 0.5,"cm")

print ("2. servo : 5 DERECE")

elif distance >=16 and distance < 24:

duty = float(168) / 10.0 + 2.5

pwm.ChangeDutyCycle(duty)

duty2 = float(150) / 30.0 + 2.5

pwm2.ChangeDutyCycle(duty2)

print ("1. servo : 20 DERECE")

print ("Mesafe:",distance - 0.5,"cm")

print ("2. servo : 10 DERECE")

elif distance >=24 and distance < 32:

duty = float(172) / 10.0 + 2.5

pwm.ChangeDutyCycle(duty)

duty2 = float(140) / 30.0 + 2.5

pwm2.ChangeDutyCycle(duty2)

print ("1. servo : 30 DERECE")

print ("Mesafe:",distance - 0.5,"cm")

print ("2. servo : 15 DERECE")

else:

duty = float(176) / 10.0 + 2.5

pwm.ChangeDutyCycle(duty)

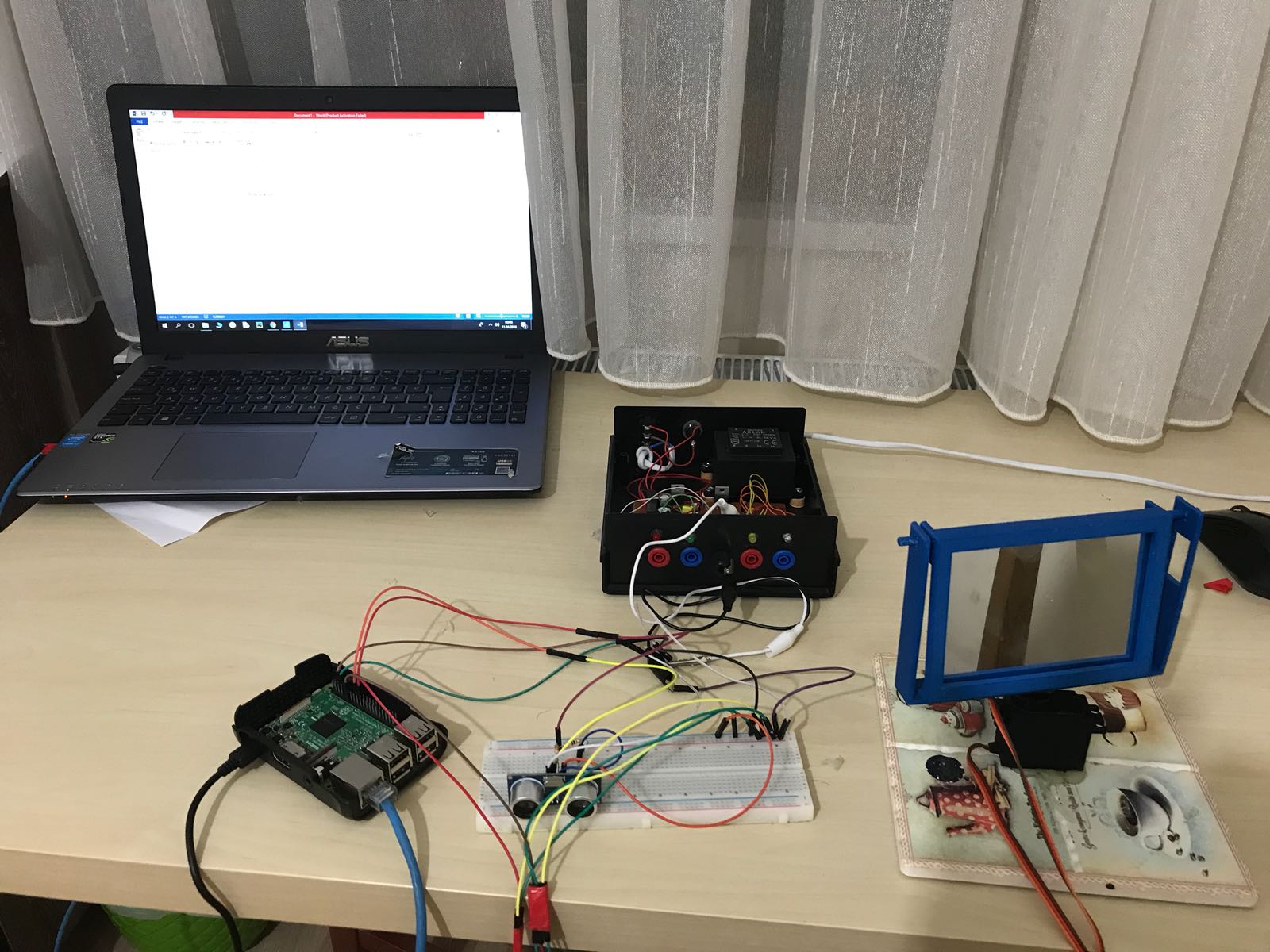
duty2 = float(130) / 30.0 + 2.5

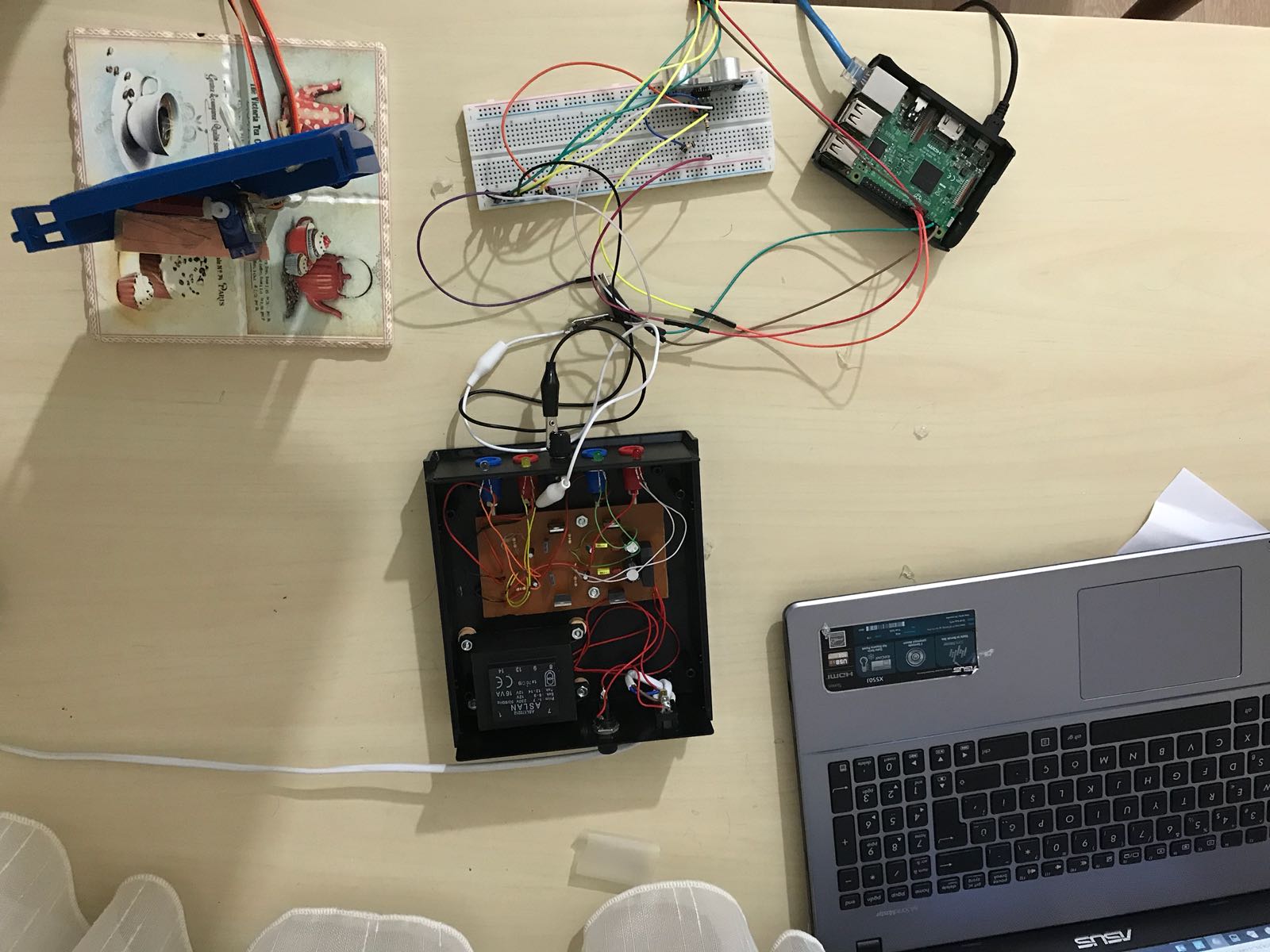
pwm2.ChangeDutyCycle(duty2)

print ("1. servo : 45 DERECE")

print ("İlk Konum")

print ("2. servo : 20 DERECE")

**Proje Resimleri:**

****

**Kaynak Kodu**

Kaynak kod[**http://maker.robotistan.com/**](http://maker.robotistan.com/)adresinden temin edilerek derlenmiştir.