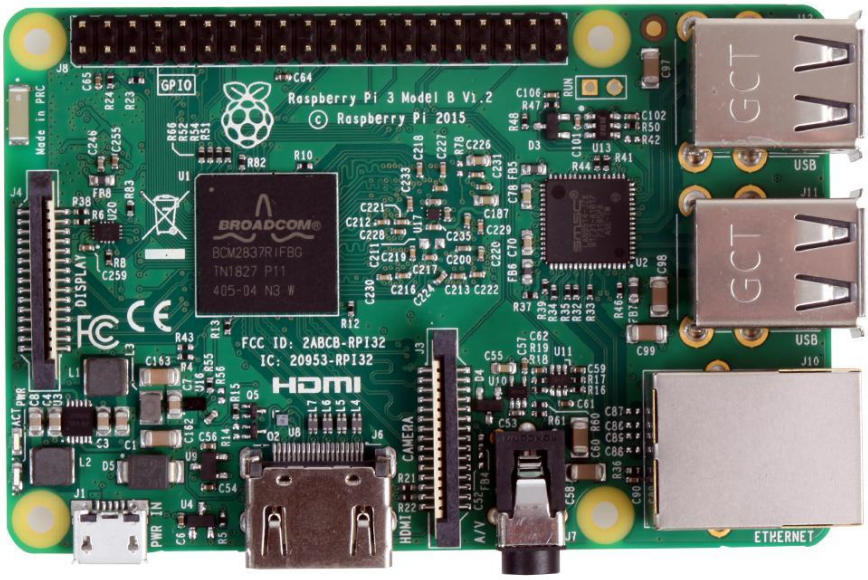
Elektrik-Elektronik Mühendisliğine Giriş I-II

OTONOM ARAÇ PROJESİ RAPORU

1.Raspberry Pi Kurulumu:

Bu labda Raspberry Pi'nin küçük bir bilgisayar olduğunu, projedeki programlamamız için neler yapabileceğimizi ve çalışma sistemini inceledik.



Bize verilen lab doğrultusunda, Raspberry Pi için 3B modelini tercih ettik. Raspberry Pi'ye Linux işletim sistemini kurarak genel kurulum işlemlerini tamamladık.

Raspberry Pi'a Linux İşletim Sistemi Kurulumu

Öncelikle SD kartımıza raspian işletim sistemini kurduk. İşletim sistemi kurulu olan SD kartımızı raspberry'e takıp, Raspberry HDMI ile ekrana bağladık. Bu işlemlerden sonra açılış ekranında kurulumu tamamladık.

2. Ubuntu Kurulumu

Sanal makine'yi (Virtual Box) bilgisayarımıza kurduk ve bu sanal makineye Ubuntu 16.04.7 LTS(Xenial Xerus) işletim sistemini kurduk. Ubuntu'nun bu versiyonunu kurmamızın sebebi, ROS Kinetic'i en efektif bir şekilde kullanabilmek içindir.

ubuntu

3.ROS

ROS, açılımı Robot Operating System olan ve robotları kontrol etmeyi sağlayan bir yazılımdır. İsminde işletim sistemi ifadesi geçse de insan ile robot arasında iletişimi sağlayan açık kaynak kodlu bir arayüz yazılımı denebilir.

ROS'un büyük bir avantajı da önceden geliştirilmiş algoritmaları robotun işlemcisine yükle-kullan formatında hızlı bir şekilde çalışabilir ve performans karşılaştırması, gözetimi yapabilme kolaylığı sağlamaktadır.

ROS Kurulumu

Bu projede kullanacağımız ROS versiyonu Kinetic'dir. Bunun kurulumunu bu site üzerinden adımları takip ederek bütün paketlerini kurduk. Bu adımdan sonra, çalışma ortamımızı yani **catkin_ws**'yi oluşturduk. Bu adımdan kodlarımızı yazmaya hazır bir ortam haline geldi

4.Ultrasonik Ses Sensörleri(HC-SR04):

Bu labda Ultrasonik ses sensörünün sonar iletişimini kullanarak karşısındaki nesneye olan mesafeyi hesaplayan bir kaynak olduğunu öğrendik. Sonar dediğimiz sistem ses dalgalarını kullanarak cismin uzaklığını hesaplamamıza yardımcı oldu.

Bize verilen lab doğrultusunda 4 adet HC-SR04 Ultrasonik ses sensörü kullandık. HC-SR04 ses sensörünün 4 bacağı bulunmaktadır. Bu lab ile 4 bacağın işlevini öğrenmiş olduk. Bunlar;

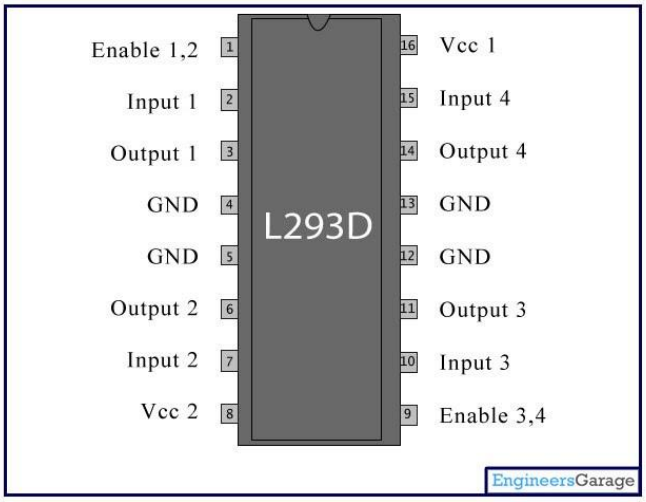
- Vcc = 5 Volt besleme bacağı.
- Gnd = Topraklama bacağı.
- Trig = Sensörün ses dalgası göndermesi için sistemi tetikleyen bacak.
- Echo = Gönderilen ses dalgasının yankısını almak için sistemi tetikleyen bacak.

HC-SR04 sensörümüz 5 volt elektrik ile çalışmaktadır. En verimli ölçüm yaptığı mesafe 2-200 cm arasındadır. 200 cm'den fazla mesafelerde verimli bir şekilde ölçüm yapamamaktadır.

Labda bizden istenilene göre HC-SR04 breadboard üstünde devresini kurarak raspberry pi'ya yazmış olduğumuz mesafe algılama algoritması ile belli mesafeleri ölçerek ekrana yazdırdık.

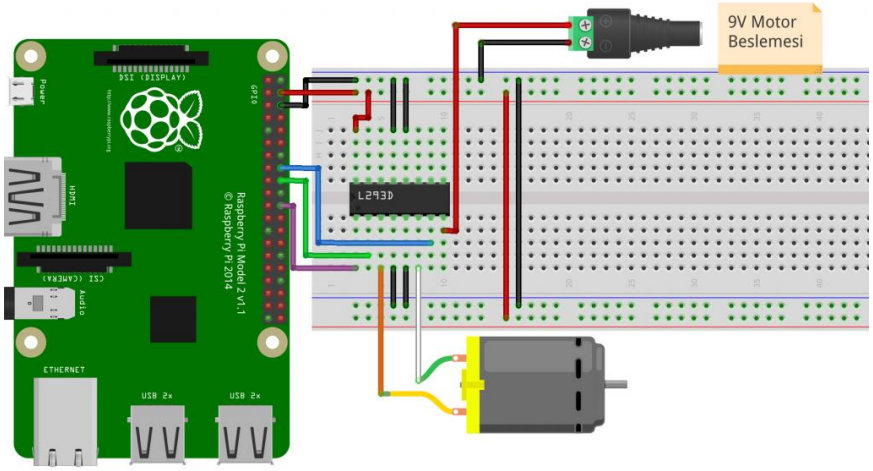
5. Motor ve Motor Sürücü:

Motor sürücü olarak L293D entegresini kullandık. L293D entegresini kullanmamızdaki temel sebep 2 tane DC motoru aynı anda sürebilmektir.



Bize verilen labta 6V enkoderli Dc motor istenildi fakat biz bunun yerine 12V enkoderli Dc motor kullandık. Buradaki tercihimiz temel sebebi böyle pilli sistemlerde pilden çekilen yüksek akımdan dolayı sistemin fazla ısınmasını ve pilin hızlı bitmesini engellemekti.

Breadboard üzerine devresini kurduğumuz motor sürücü ile tek bir motoru sürdüren kodu raspberry'e yazarak motoru sürdük.

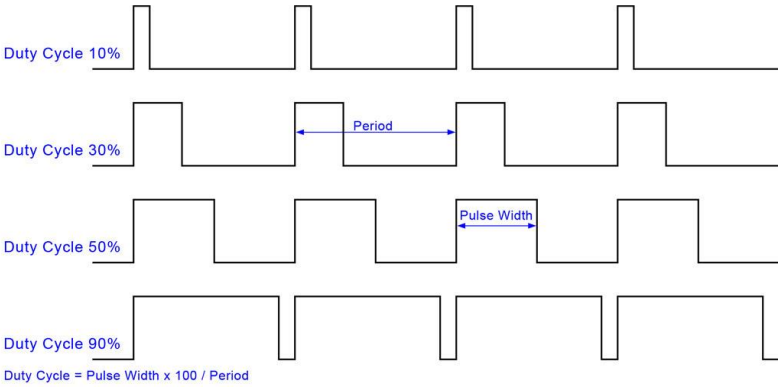


6.Mesafeye Göre PWM Kontrolü:

Daha önce yaptığımız ultrasonik mesafe sensörü devresini kullanarak, sensör ile engel arasındaki mesafeyle doğru orantılı olarak motorun hızının belirleyecek kodu yazarak labda istenilen sonuca ulaştık.

PWM Nedir?

Açılımı **Pulse Width Modulation** yani **Sinyal Genişlik Modülasyonu** olan bu teknik en basit haliyle bir **sinyal modülasyon tekniği** olarak tanımlanabilir. Sinyal bilgisinin aktarım için uygun hale çevrilmesi amacının yanı sıra güç kontrolü sağlamak ve elektrik makineleri, güneş pili şarj üniteleri gibi özel devrelere destek olmak amacı da taşır.



7. Enkoder Hesabı:

Enkoder, döner veya doğrusal hareketi dijital sinyale dönüştürür. Genellikle hız, yön, mesafe veya konum gibi hareket parametrelerinin izlenmesi veya kontrol edilmesi için kullanılır.

Lab'da istenilen doğrultuda kodumuzu yazarak enkoder okumaya ve kontrol etmeye çalıştık fakat motorların enkoderlerini okumak için Raspberry Pi yeterli özelliklere sahip olmadığından mikro işlemciler ile Motorun enkoder verisini ölçüp Tx/Rx pinleri ile seri haberleşme mantığıyla Robotun beyini olan Raspberry Pi ile haberleştirdik. Raspberry üzerindeki yükü azaltmak ve adım kaçırmamak için bir PIC devresi tasarlayıp kodlarını yazarak enkoder okuma ve yönetme işini tamamen PIC in yapmasını sağlamayı amaçladık. Bu işlemle enkoderi okuyup sonraki aşamalar için kontrol edilebilir bir hale getiriyoruz.

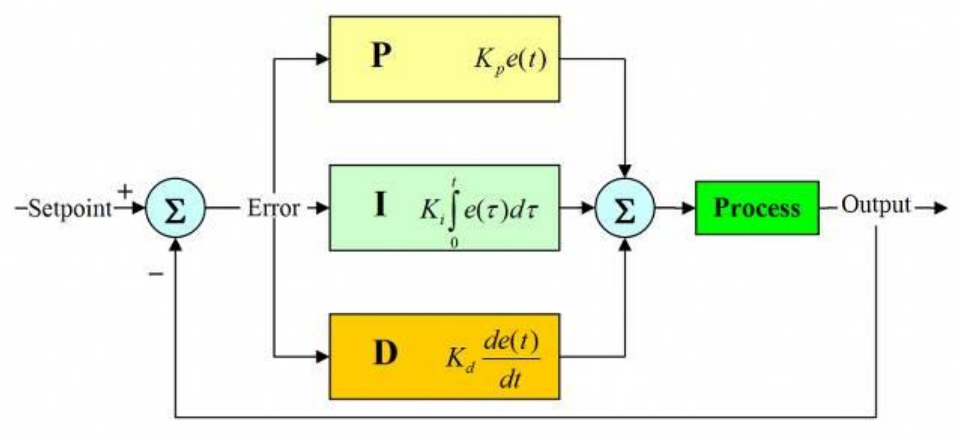
Araştırmalarımızın sonucuna göre tercih ettiğimiz mikroişlemci modeli PIC 18F46K22'den iki adet kullandık bunun nedeni daha hassas ve hızlı işlem yapabilmemizdir. İki mikroişlemci de farklı enkoderleri okuyor. Biri diğer PIC'e enkoder bilgisini gönderiyor, diğer PIC ise hem kendisine bağlı olan enkoderi okuyup hem diğer PIC'den gelen enkoder bilgisini alıyor hem de raspberry'e dataları gönderiyor.

8. Hız Hesabı (RPM)

Enkoder çıktımızın verdiği çıktılara göre enkoder verisini, yazdığımız kodlar ile bir sayaç belirledik ve tekerleğin içindeki milin ne kadar ilerlediğini bulduk, bu milin kaç saniyede kaç adım attığını hesaplayarak bunu RPM (Revolutions Per Minute) cinsinden hıza dönüştürdük. Bir while döngüsü içinde zaman değişkenini sürekli güncellediğimiz için milin her adımında anlık olarak hız çıktısı alabildik.

9. PID

PID kontrol elektronik cihazlar, mekanik cihazlar ve pnömatik sistemler gibi çok geniş alanlarda kullanılabilirler. PID, Proportional (Oransal), Integral (Integral) ve Derivative (Türev) terimlerini içeren bir kontrolcüdür. Her bir terimin sisteme etkisi farklıdır.

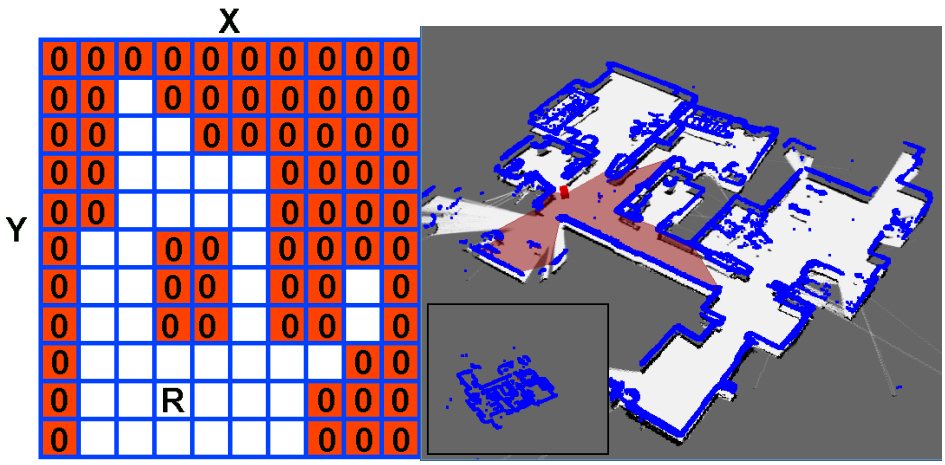


PIC devremizin bize verdiği enkoder çıktılarına göre, hataları en aza indirmek için PID kodumuzu yazdık ve bunu raspberry üzerinden sistemimize entegre ettik.

Her sistemde farklılık gösteren Kp, Ki ve Kd değerlerini deneyerek ve formülü kullanarak sistemi en stabil çalıştıran PID fonksiyonumuzu oluşturduk.

10. Mapping

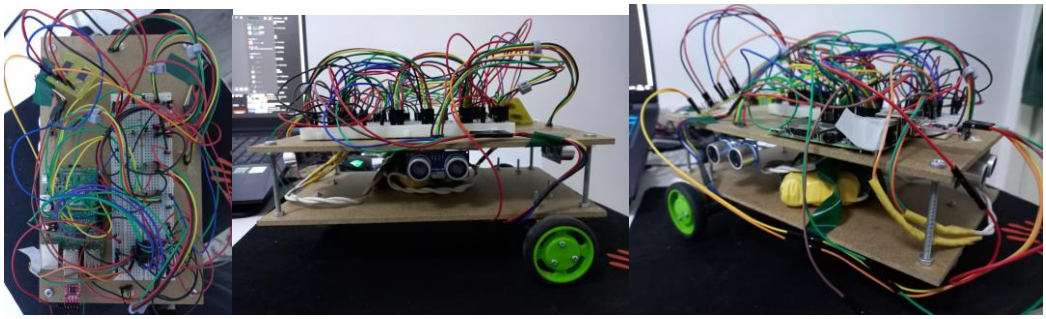
Doğrusal(lineer) veya doğrusal olmayan(nonlineer) problem çözümlemelerinde neyin nerede olduğunu tek tek hatırlamak yerine hatırlanacak her şeyi tek bir matrixden eşleştirme yöntemi
RAM'i ve işlemci yükünü hafifleterek programcıyı büyük zahmetlerden kurtarır. Hatta neredeyse programın kalbi mappingdir desek aşırı olmaz.



Mapping algoritmasını kafamızda kurup, çıkışa giden en kısa yolu bulan ve bu bulduğu yoldan çıkışa ilerlemesini sağlayan kod algoritmasını yazdık. Mapping algoritmamız her adımda öncelikli olarak solunu kontrol ederek eğer doluysa 1 boşsa 0 olarak işaretliyor. Önce sola gitmesinin sebebi labirent çözümünde daha hızlı çözüme ulaşmak için sol yön öncelik olarak alınır. Bu işlemin sonucunda labirentin tamamını gezerek matrise çevirmiş olur.
Sonrasında kurduğumuz algoritma sayesinde Deep search (DSS) yöntemiyle bulunduğu noktadan çıkışa olan en kısa yolu bularak bunun çıktısını verir ve sonuç olarak araç çıkışa doğru ilerler.

Rapor Özeti

Projemizde bugüne kadar istenilen labları başarılı bir şekilde yaptık. Şu an pratik olarak denemediğimiz sadece mapping kısmı kaldı ve birkaç gün içinde bunu da deneyeceğiz ve projemizi tamamlamaya çok yaklaşmış olacağız. Genel olarak baktığımızda; projemizin mesafe sensörleri, motorları, mesafeye göre PWM kontrolü ve enkoder okuyup kontrol etmesi çalışıyor. Mapping kısmı birkaç gün içinde test edilecek.



Kodlara bu PDF dosyasının yüklü olduğu klasör içindeki “Kodlar” klasöründen erişebilirsiniz.

Malzeme Listesi

- Raspberry Pi Model 3B x1
- HC-SR04 Ultrasonik Ses Sensörü x4
- L293D Motor Sürücü x1
- 12 V 290 RPM Enkoderli DC Motor ve Motor Tutucu(Pololu Marka) x2
- PIC 18F46K22 x2
- Lityum İyon Pil x3
- Sarhoş Teker x1
- 5 Cm Çaplı Tekerler x2
- Jumper Seti (Dişi-Dişi, Erkek-Erkek, Dişi-Dişi, Erkek-Dişi) x4
- Breadboard x1
- 2.2 kOhm Direnç x8
- USB-TTL Dönüştürücü x1

Kaynakça

- https://www.electronicwings.com/raspberry-pi/raspberry-pi-uart-communication-using-python-and-c
- https://stackoverflow.com/questions/25897723/pid-controller-for-dc-motor
- https://www.youtube.com/playlist?list=PLk51HrKSBQ8-jTgD0qgRp1vmQeVSJ5SQC
- https://github.com/richardw05/gopigo_ws/blob/master/src/diffdrive_controller/src/diffdrive_controller.py
- http://moorerobots.com/blog/post/4
- https://www.youtube.com/watch?v=u9l-8LZC2Dc&ab_channel=ImeshSachinda
- https://github.com/imeshsps/ros-navbot
- https://github.com/merose/diff_drive
- https://roboticsbackend.com/raspberry-pi-gpio-interrupts-tutorial/
- https://projects.raspberrypi.org/en/projects/robotPID/1
- https://www.projehocam.com/pid-kontrol-algoritmasi-nedir/
- https://www.elektrikport.com/teknik-kutuphane/pid-denetleyiciler/11787#ad-image-0

Projeyi Hazırlayanlar

İsmail Utku CAN (21803002)

Başak YALÇINER (21803016)

Furkan BAYDAR (21803017)

Emre ÖZKUL (21803012)