KARADENİZ TEKNİK ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

TASARIM PROJESİ

ARDUNİO İLE ENGELDEN KAÇAN ROBOT YAPIMI

ÖZGÜR BEKAROĞLU

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

TRABZON

BAHAR 2014

KARADENİZ TEKNİK ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

TASARIM PROJESİ

ARDUNİO İLE ENGELDEN KAÇAN ROBOT YAPIMI

ÖZGÜR BEKAROĞLU

210968

DANIŞMAN: Doç. Dr. Mustafa ULUTAŞ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

TRABZON

BAHAR 2014

ÖNSÖZ

Ardunio ile engelden kaçan robot konulu bu çalışma Karadeniz Teknik Üniversitesi Bilgisayar Mühendisliği Bölümü'nde 'Tasarım Projesi' olarak yapılmıştır.

Lisans öğrenim süresince; tüm bilgi birikimlerini sakınmadan bizimle paylaşan hocalarımıza, değerli bölüm başkanımız Doç. Dr. Cemal KÖSE hocamıza, tez danışmanımız Doç. Dr. Mustafa ULUTAŞ hocamıza ve her türlü destekleriyle beni hiçbir zaman yalnız bırakmayan aileme, arkadaşlarıma teşekkürlerimi ve saygılarımı sunarım.

İÇİNDEKİLER

ÖZET

.

Teknolojinin ilerlemesiyle robotik sistemler günlük yaşantımızın bir parçası haline gelmiştir. Bununla birlikte ortaya çıkan fonksiyonellik sorunlarından biride engel algılama, engelden kaçma veya engel aşma özellikleridir.

Engelden kaçan robot sitemi günümüzde birçok alanda kullanılmaktadır. Bunu motorlu taşıtlarda örneğin seyir halinde engel algılama, günlük yaşantımızda örneğin mobilyalara çarpmadan gezinen elektrik süpürgesi veya askeri alanda arazi şartlarında gezinen robotun engel(mayın, taş blok vb.) algılamasında kullanılmaktadır.

Ardunio ile engelden kaçan robot tasarım projesi iki aşamadan oluşmuştur. Donanım kısmı; servo motorlar, sensör ve mikrodenetleyici ve yazılım kısmı, robotun karşılaştığı engellerle belirlenen algoritmalarla karşılık vermesi için ardunionun yazılımının yazılması olarak ayrılmıştır.

SEMBOLLER VE KISALTMALAR

AVR	Alf (Egil Bogen) and Vegard (Wollan)'s RISC processor
IDE	Tümleşik geliştirme ortamı
GPLv2	2.seviye Genel Kamu Lisansı
GNU	Genel Kamu Lisansı

1. GİRİŞ

İlk bölümde;

- Android,
- Servo motor,
- Ultrosonic sensör,

hakkında kısaca bilgi verilecektir.

1.1. Ardunio

Arduino bir G/Ç kartı ve Processing/Wiring dilinin bir uygulamasını içeren geliştirme ortamından oluşan bir fiziksel programlama platformudur. Arduino tek başına çalışan interaktif nesneler geliştirmek için kullanılabileceği gibi bilgisayar üzerinde çalışan yazılımlara da (Macromedia Flash, Processing, Max/MSP, Pure Data, SuperCollider gibi) bağlanabilir. Hazır üretilmiş kartlar satın alınabilir veya kendileri üretmek isteyenler için donanım tasarımı ile ilgili bilgiler mevcuttur.

Donanımsal açıdan arduino kartları bir Atmel AVR mikrodenetleyici (Eski kartlarda ATmega8 veya ATmega168, yenilerinde ATmega328) ve programlama ve diğer devrelere bağlantı için gerekli yan elemanlardan oluşur. Her kartta en azından bir 5 voltluk regüle entegresi ve bir 16 MHZ kristal osilator (bazılarında seramik rezonatör) bulunur. Mikrodenetleyiciye önceden bir bootloader programı yazılı olduğundan programlama için harici bir programlayıcıya ihtiyaç duyulmaz

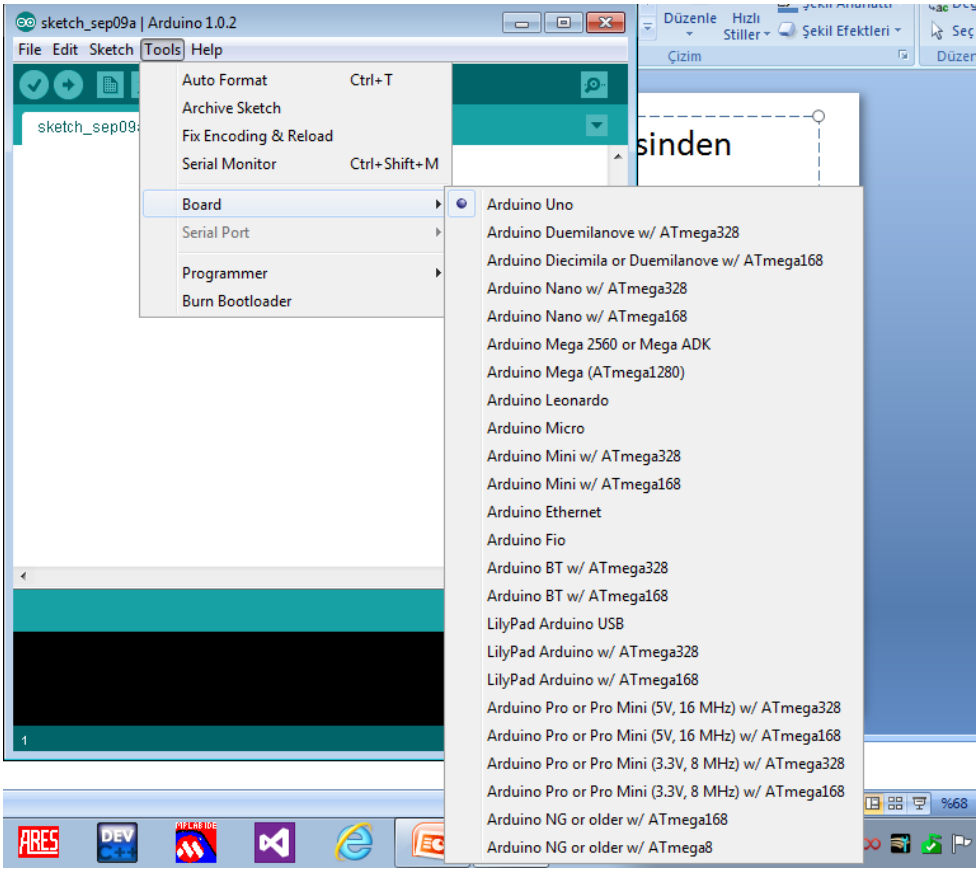
Yazılım açısından Arduino IDE kod editörü ve derleyici olarak görev yapan, aynı zamanda derlenen programı karta yükleme işlemini de yapabilen, her platformda çalışabilen Java programlama dilinde yazılmış bir uygulamadır. Geliştirme ortamı, sanatçıları programlamayla tanıştırmak için geliştirilmiş Processing yazılımından yola çıkılarak geliştirilmiştir.

Arduino donanım referans tasarımları Creative Commons Attribution Share-Alike 2.5 lisansı ile dağıtılmaktadır ve Arduino web sitesinden indirilebilir. Bazı Arduino donanımları için yerleşim ve üretim dosyaları da mevcuttur. Geliştirme ortamının kaynak kodu ve Arduino kütüphane kodları GPLv2 lisansıyla lisanslanmıştır.

Arduino'nun günümüzde çok fazla kullanılmasının nedenini ve örnek ardunio uygulamaları hakkında bilgi vermek gerekirse.

- Arduino açık kaynak kodlu bir mikrodenetleyici kartıdır.

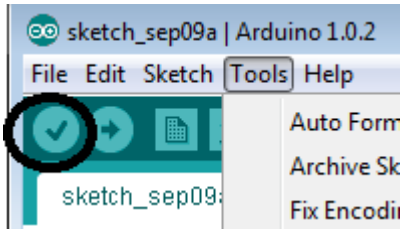
- Mikroişlemci bilgisi gerektirmez.
- Açık kaynaklı demek kullanıcı ile kaynak kodlarının paylaşıldığı ve değişiklik hakkının kullanıcıya verildiği anlamına gelir.
- Arduino'nun bu kadar popüler olmasının nedenlerinden biri de programlamasının kolay olmasıdır.
- Arduino geliştirme ortamı kendi sitesinden ücretsiz şekilde indirilebilir.
- Birçok çeşidi ve donanım eklentileri mevcuttur.(Shield)
- Arduino geliştirme ortamında tools menüsünden bilgisayara taktığınız arduino modelini kontrol etmekte fayda var.
- En önemli özelliklerinden birisi de zengin kütüphane desteğidir.



ŞEKİL1.1.1 Arduino kütüphaneleri

Programlama dili

- Arduino programlama dilinde 2 temel fonksiyon bulunur.
- 1-) setup () : Bu fonksiyon Arduino çalışmaya başladıktan sonra ya da reset butonuna basıldıktan sonra 1 kere çalıştırılır. Bu fonksiyonda tek seferlik fonksiyonlar çalıştırılır. Örneğin pin ayarlaması,seri haberleşme başlatılması gibi.
- 2-) loop() : Bu fonksiyon sonsuz döngü fonksiyonlarıdır. setup() fonksiyonunun hemen ardından çalıştırılır ve arduino çalıştığı sürece devam eder.
- Birçok programlama dilinde olduğu gibi arduino programlama dili de case-sensitive(büyük küçük harf duyarlı) bir dildir.
- Arduino da yazdığımız programları Verify butonu ile derleriz.



ŞEKİL1.1.2. Verify butonu

- Verify butonu yukarıda gösterilmiştir.
- Yanında ki buton ise upload butonudur. Bu buton ile yazdığımız programı Arduino'ya yükleriz.

Dijital Giriş Çıkış Fonksiyonları

- Dijital giriş/çıkış , 1 veya 0 bilgisini okumak yada göndermek için kullanılıyor.
- pinMode(pin no,giris yada çıkış modu); Bu fonksiyon pinlerin nasıl kullanılacağını ayarlar.eğer çıkış olarak kullanılacaksa OUTPUT giriş olarak kullanılacaksa INPUT yazılır. Örnek,
- pinMode(13,INPUT);
- pinMode(13,OUTPUT);

- digitalWrite(pin no, HIGH or LOW); Dijital olarak çıkış ayarlanmış pinlere 1 ya da 0 verilmesini sağlayan fonksiyondur. HIGH ise 5v LOW ise 0 volt çıkış verir.

- digitalRead(pin no); Dijital olarak giriş olarak ayarlanmış pinlerdeki değerin 1 ya da 0 olduğu değerini gösterir.

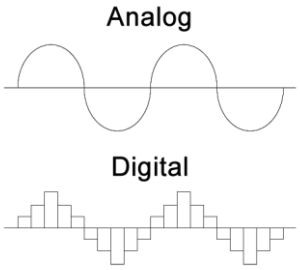
- NOT: Çıkış olarak ayarladığımız pinler 5v verse de maksimum verebileceği akım değeri 40mA dir. Yüksek akım isteyen elemanlarla çalışırken yükselteç kullanılmalıdır. (opamp,transistor)

Gecikme fonksiyonları

- delay(); Bu fonksiyonun içine yazdığımız kadar fonksiyonumuz o noktada o kadar milisaniye cinsinden durur.

- delayMicroseconds(); Bu fonksiyon ise Microsaniye cinsinden fonksiyonu durdurur.

Analog Giriş Çıkış İşlemleri



ŞEKİL1.1.3. Analog ve digital gösterim

- Arduino üzerinde bulunan mikrodenetleyicide 10 bitlik ADC bulunuyor. Bunun anlamı şudur ki 0-5v arası gerilimler 0 ile 1023 arasındaki sayılar olarak okunur. (1024 farklı değer)

- Eğer okuduğumuz analog değerin kaç volt olduğunu öğrenmek istersek deger*5/1023 işlemini yapmamız gerekir.

- Arduino'nun çeşidine göre üzerindeki analog giriş sayıları farklılık gösterir.

- Analog giriş den değer okumak için kullanacağımız fonksiyon analogRead(pin no); fonksiyonudur. Pin numarası olarak A0 , A1, ... yazılır.

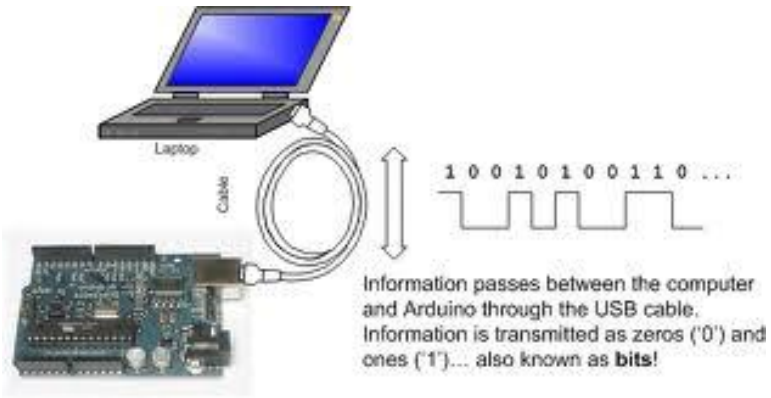
- Analog çıkış olarak PWM tekniği kullanılır. Bu teknikle dijital yöntemle analog çıkış değerleri üretebiliyoruz

- analogWrite(pin no,duty cycle); Bu fonksiyonla analog çıkış verebiliyoruz. 0 ile 255 arasında bir değer verilebilir. 255 değeri 5 volta denk gelir.

- Burada dikkat edilmesi gereken nokta bütün dijital çıkış pinlerinden analog çıkış veremiyoruz sadece yanında (~) işareti olanlardan analog çıkış verebiliyoruz.

Seri haberleşme

- Elektronik birimler bazı projelerde birbirleriyle iletişim kurmaları gerekebilir. Dijital haberleşmede 2 yöntem var seri ve paralel.



ŞEKİL 1.1.4 Arduino bilgisayar bağlantısı

- Seri haberleşmede veriler tek bir hat üzerinden sıra ile gönderilir.

- Seri haberleşmede daha az veri hattı gerekmektedir. Bu yüzden sıkça kullanılır. Günümüzde en çok kullanılan USB buna en iyi örnektir. Derlediğimiz programları arduino kartına yükleme işlemi de aslında seri haberleşme ile olur (USB ile).

- Seri haberleşme 2 ayrı hat üzerinden (RX ve TX) gerçekleşir.

- Arduino üzerinde bulunan seri haberleşme ünitesine UART (Universal asynchronous receiver/transmitter: Evrensel asenkron alıcı/verici) adı verilir. Arduino modeline göre 1 ya da daha fazla haberleşme ünitesi bulunabilir.

  TX ve RX in bağlı olduğu pinler seri haberleşme esnasında dijital olarak giriş ya da çıkış olarak kullanılamaz.

- **available**() ➔ Kaç tane okunmayı bekleyen veri (bayt) var?

- **begin**() ➔ Seri İletişimi başlatma
- **end**() ➔ Seri iletişimi sonlandırma
- **print**() ➔ Seri iletişim üzerinden veri gönderme (text)
- **println**() ➔ Veri gönderme (satır sonu karakteri eklenir)
- **read**() ➔ Gelen veriden okuma
- **readBytes**() ➔ Gelen verileri topluca okuma
- **write**() ➔ Veri gönderme (binary)

Dijital Giriş Çıkış Fonksiyonları

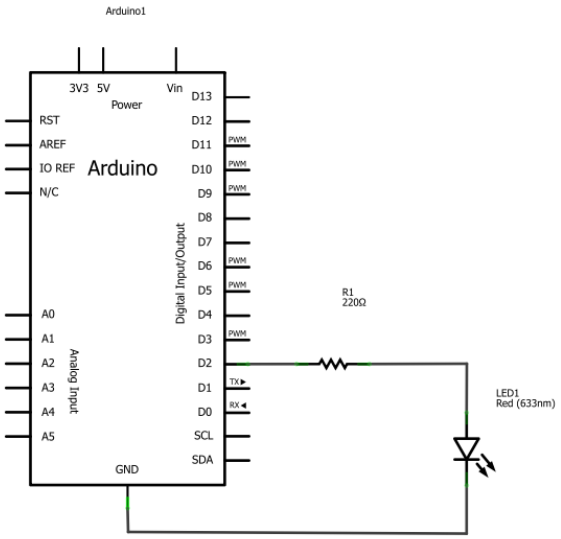
- Dijital giriş/çıkış 1 veya 0 bilgisini okumak ya da göndermek için kullanılıyor.
- pinMode(pin no,giris yada çıkış modu); Bu fonksiyon pinlerin nasıl kullanılacağını ayarlar. Eğer çıkış olarak kullanılacaksa OUTPUT giriş olarak kullanılacaksa INPUT yazılır. Örnek,
- pinMode(13,INPUT);
- pinMode(13,OUTPUT);
- digitalWrite(pin no, HIGH or LOW); Dijital olarak çıkış ayarlanmış pinlere 1 ya da 0 verilmesini sağlayan fonksiyondur. HIGH ise 5v LOW ise 0 volt çıkış verir.
- digitalRead(pin no); Dijital olarak giriş olarak ayarlanmış pinlerdeki değerin 1 ya da 0 olduğu değerini gösterir.
- NOT: Çıkış olarak ayarladığımız pinler 5v verse de maksimum verebileceği akım değeri 40mA dir. Yüksek akım isteyen elemanlarla çalışırken yükselteç kullanılmalıdır. (opamp,transistor, vb.)

Gecikme fonksiyonları

- delay(); Bu fonksiyonun içine yazdığımız kadar fonksiyonumuz o noktada o kadar milisaniye cinsinden durur.
- delayMicroseconds(); Bu fonksiyon ise microsaniye cinsinden fonksiyonu durdurur.

Örnek Projeler

1. Led Yakma

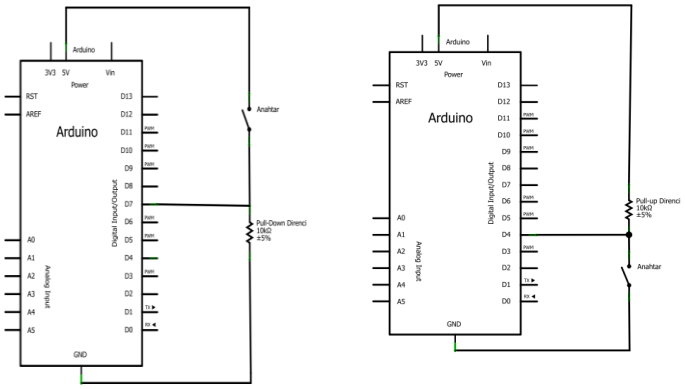


ŞEKİL 1.1.5 devre tasarımı

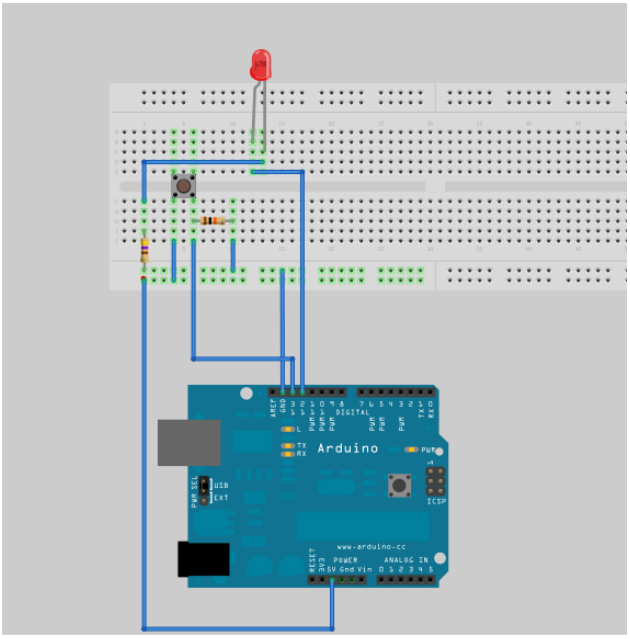
Program:

```
void setup() {

pinMode(2, OUTPUT);

}

void loop() {

digitalWrite(2, HIGH);

delay(1000);

digitalWrite(2, LOW);

delay(1000 );

}
```

2. Buton Girişi Okuma



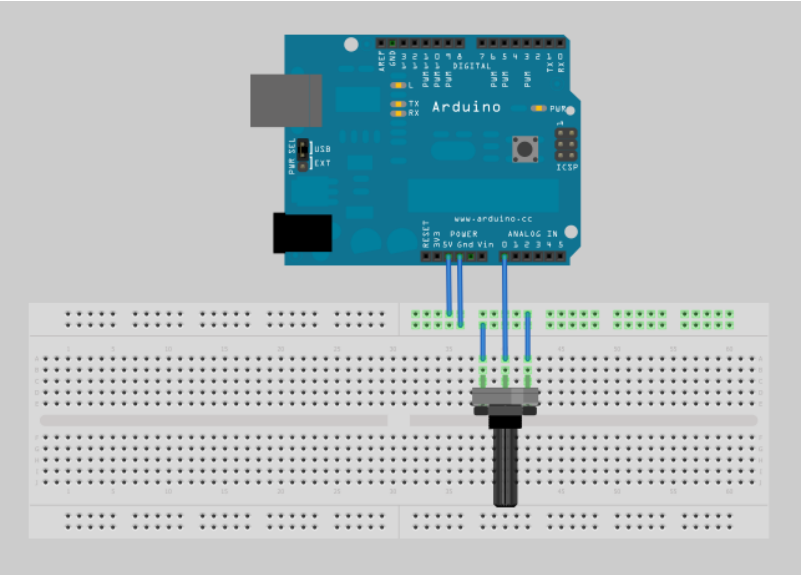
ŞELİK 1.1.6. devre tasarımları



ŞEKİL 1.1.7 program ile çizim

```
İnt ledPin=12;
İnt butonPin=13;
void setup()
{
pinMode(ledPin,OUTPUT);
pinMode(butonPin,LOW);
}
void loop() {
  // Buton durumunu oku
  buttonDurumu = digitalRead(butonPin);
  /* Butona basıldığında butonun durumu HIGH olacaktır.
   Bu durumda LED çıkışını HIGH yapıyoruz. Tersi durumda
   ise LOW yapıyoruz */
  if (buttonDurumu == HIGH) {
    digitalWrite(ledPin, HIGH);
  }
  else {
    digitalWrite(ledPin, LOW);
  }
}
```

3. Analog Giriş Okuma



ŞEKİL1.1.8 program ile çizim

```
void setup() {
  Serial.begin(9600);
}
void loop() {
    analogGiris = analogRead(A0);
  gerilimDegeri = (analogGiris / 1023.0)*5.0;
  Serial.print("Analog deger:");
  Serial.println(analogGiris);
  Serial.print("Gerilim degeri:");
  Serial.println(gerilimDegeri);
  delay(1000); // Bir saniye bekle}
```

1.2. Servo Motor

Servo, herhangi bir mekanizmanın işleyişini hatayı algılayarak yan bir geri besleme düzeneğinin yardımıyla denetleyen ve hatayı gideren otomatik aygıttır. Robot teknolojisinde en çok kullanılan motor çeşididir. Bu sistemler mekanik olabileceği gibi elektronik, hidrolik-pnömatik veya başka alanlarda da kullanılabilmektedir Servo motorlar da çıkış; mekaniksel konum, hız veya ivme gibi parametrelerin kontrol edildiği bir düzenektir. Servo motor içerisinde herhangi bir motor AC, DC veya Step motor bulunmaktadır. Ayrıca sürücü ve kontrol devresini de içerisinde barındırmaktadır.

D.C servo motorları, genel olarak bir D.C. motoru olup, motora gerekli D.C. aşağıdaki metotlardan elde edilir.

1- Bir elektrik yükselteçten.

2- A.C. akımın doyumlu reaktörden geçirilmesinden.

3- A.C. akımın tristörden geçirilmesinden.

4- Amplidin, retotrol, regüleks gibi dönel yükselteçlerden elde edilir.

D.C. servo motorlar çok küçük güçlerden çok büyük güçlere kadar imal edilirler(0,05 Hp'den 1000 Hp'ye kadar). Bu motorlar klasik D.C. motorlar gibi imal edilirler. Bu motorlar küçük yapılıdır ve endüvileri (yükseklik, uzunluk / Çap oranıyla) kutup atalet momentini minimum yapacak şekilde tasarlanırlar. Küçük çaplı ve genellikle içerisinde kompanzasyon sargısı olan, kuvvetli manyetik alanı boyu uzun doğru akım motorlarına da servo motor denir. D.C. servi motor çalışma prensibi açısından aslında, Statoru Daimi Mıknatıs bir D.C. motordur. Manyetik alan ile içinden akım geçirilen iletkenler arasındaki etkileşim nedeniyle bir döndürme momenti meydana gelir. Bu döndürme momenti manyetik alan vektörü ile sargı akım vektörü arasındaki açı 90° olduğunda maksimum değerin alır. Bir D.C. sevro motorda fırçaların konumları, her iki dönüş yönü için de döndürme momenti açısının 90° olmasını sağlayacak şekilde belirlenmiştir. Kolektör

segmentlerinin fazla olması neticesinde momentin sıfır bir noktada rotorun hareketsiz kalması engellenmiş olur.

Sanayide kullanılan çeşitli doğru akım motorları vardır. Servo sistemlerde kullanılan doğru akım motorlarına ise D.C. servo motorlar adı verilir. D.C. servo motorlarda rotor eylemsizlik momenti çok küçüktür. Bu sebepten piyasada çıkış momentinin eylemsizlik momentine oranı çok büyük olan motorlar bulunur.

Bazı D.C. servo motorların çok küçük zaman sabitleri vardır. Düşük güçlü D.C. servo motorlar piyasada genellikle bilgisayar kontrollü cihazlarda (disket sürücüler, teyp sürücüleri, yazıcılar, kelime işlemciler, tarayıcılar vs.) kullanılırlar. Orta ve büyük güçlü servo motorlar ise sanayide genellikle robot sistemleri ile sayısal denetimli hassas diş açma tezgâhlarında kullanılır. D.C. motorlarda alan sargıları rotor sargılarına seri veya paralel bağlanır. Endüvi sargılarından bağımsız olarak uyartılan alan sargılarının akısı endüvi sargılarından geçen akımın fonksiyonu değildir. Bazı D.C. motorlarda manyetik akı sabittir. Uyarma sargıları endüviden bağımsız olan veya sabit mıknatısla uyartılan motorlarda hız kontrolü endüvi gerilimi ile yapılabilir. Bu tip kontrol yöntemine endüvi kontrol yöntemi denir.

Uyarma sargılarının yarattığı akı ile yapılan denetlemede ise endüvi akımı sabit tutulur. Statorda bulunan uyartım sargılarının yarattığı akının kontrolü ile hız ayarlanır. Bu tip motorlara alan kontrollü motorlar denir. Fakat rotor sargılarından geçen akımın sabit tutulabilmesi ciddi bir problemdir. Zira rotor akımı yükün ve kaynağın birer fonksiyonudur. Endüvi kontrollü motorlara göre alan kontrollü motorların alan sabitleri daha büyüktür. Büyük aralıklarda değişen hız ayarlarında rotor geriliminin değiştirilmesi; buna karşılık küçük aralıklarda hassas hız ayarı gereken yerlerde ise alan sargılarının yaratmış olduğu manyetik akı hız kontrolü yöntemi tercih edilir.

D.C. servo motorlar genellikle “elektronik hareketli denetleyiciler ” adı verilen servo sürücüler ile kontrol edilirler. Servo sürücüler servo motorun hareketini kontrol ederler. Kontrol edilen büyüklükler çoğu zaman noktadan noktaya konum kontrolü, hız kontrolü ve ivme programlamasıdır. PWM tekniği adı verilen darbe genişlik modülasyonu genellikle robot kontrol sistemlerinde, sayısal kontrol sistemlerinde ve diğer konum denetleyicilerinde kullanılırlar.

DC Servo motor ve AC Servo motorun karşılaştırılması

Fırçasız servo motorlar D.C. servo motorların bakım gereksinimlerini ortadan kaldırmak amacıyla getirilmiştir. Modern servo sistemlerde kullanılan fırçasız servo motorların en önemli üstünlüğü fırça ve komütatör elemanlarının bulunmasıdır. Bu nedenle fırçaların bakımı diye bir olaydan bahsedilemez ve fırçalardan birçok problem önlenmiş olur.

Kolektörlü D.C. servo motorlarda oluşan problemler bazen çok açık bir şekilde belli olmaz. Bazen fırçalarda olan kirlenme bile problem oluşturabilir. Fırçaların performansı ve ömrü atmosferlik şartlarla bile değiştiğinden dolayı değişik ortam koşullarında değişik yapılı fırçalar kullanılabilmektedir. Fırçasız servo motorlarda verim, eş ölçüdeki bir D.C. servo motora oranla daha yüksektir ve fırçaların sürtünme etkisi olmadığından dolayı sürtünme kuvveti verime katkıda

bulunur. Kolektör ve fırça aksamının yokluğu motor boyunu düşürür. Bu sadece motor hacmini düşürmekle kalmaz rotor destek rulmanları arasındaki mesafe ve rotor boyunun kısalması dolayısıyla rotorun yanal rijitliği de arttırılmış olmaktadır. Bu özellik hız/eylemsizlik oranına gereksinim duyulan uygulamalarda önemlidir.

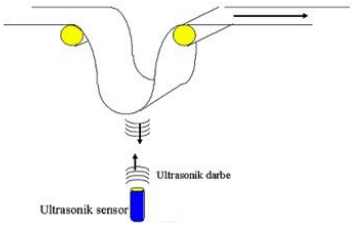
Fırçasız konfigürasyon da sarımların sabit stator içine sarılması sebebi ile ısı yalıtımı için daha fazla en kesit alanı sağlanabilmekte ve sargılarda oluşabilmek ısı artışı algılama elemanları vasıtasıyla kolayca algılanabilmektedir. Modern servo sistemlerde pozisyon sinyalinin belirlenmesi amacı ile bir kodlayıcı (encoder) veya çözümleyici (resolver) kullanılır. Kodlayıcı ve motorun tek bir ana iskelet üzerinde toplanması ile sistem daha kompakt bir yapıda olmaktadır. Bu motor yapısında manyetik akıyı üretmek için gerekli olan mıknatıs rotora monte edildiğinden dolayı döner-alan tipli motor yapısındadır. Senkron motor tipli fırçasız servo motorların yapıları doğru akım servo motorlarından farklı olması nedeniyle bu tipteki servo motorlar fırçasız D.C. servo motor olarak adlandırılır.

D.C. servo motorlardaki kolektörün aksine Fırçasız D.C. servo motorlar akımı yarı iletken güç elektroniği elemanları ile doğrulturlar. Diğer yönden rotor manyetik alanının kodlayıcı vasıtası ile algılanıp, algılanan bu pozisyona uygun düşecek şekilde stator sarımlarına üç fazlı alternatif akım verilmesi dolayısı ile kalıcı mıknatıslı senkron motor tipindeki fırçasız servo motorlar aynı zamanda A.C. servo motorlar olarak da adlandırılır. Fırçasız servo motorlarda rotor manyetik alanı ile statora verilen akımlar dikey şekilde kontrol edildiği taktirde D.C. servo motorlarla aynı olan hız-moment karakteristikleri elde edilir. Servo motorlar kullanımları gereği çok sık şekilde ivmelenme ve yavaşlama işlemlerine maruz kaldıklarından dolayı, maksimum moment değeri anma momentlerinin katlarca fazlası olmalıdır. D.C. servo motorlarda anma momentlerinin aşılması durumunda komütatör aksamında kıvılcımlaşma olayı meydana gelir. Aynı şekilde hız arttıkça moment değeri de çok hızlı bir şekilde düşer.

1.3. Ultrasonic Sensor

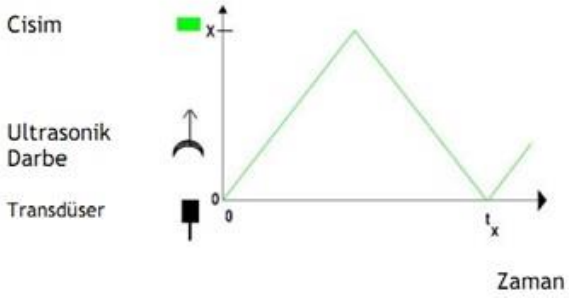
Ses dalgaları sınıflandırılmasında 20Khz-1Ghz aralığındaki ses sinyalleri ultrasonic ses olarak tanımlanmıştır. Birçok ultrasonic sensör 40Khz frekansında ultrasonic ses üretmektedir. Burada önemli olan sesin yüksekliğinde belirleyici olan etken frekanstır. Ses yüksekse frekansta yüksektir. Ultrasonic ses sinyallerini insan kulağı algılayamaz.

Sensörün çalışmasını incelersek.



ŞELİK1.3.1. Sensörlerin çalışma prensibi

Transdüser ultrasonik darbeyi iletir. Darbe sehimden yansır ve transdüser tarafından alınır. Darbenin gidiş geliş zamanı sensörle sehimin mesafesine göre orantılıdır.

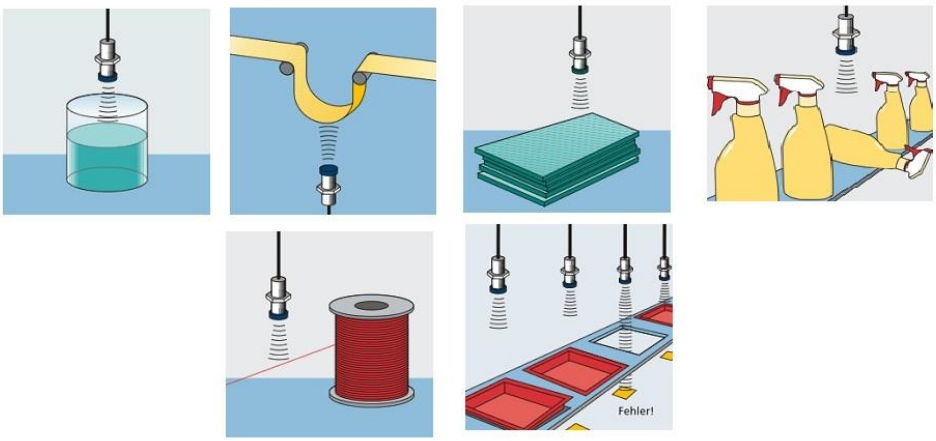


ŞEKİL 1.3.2. Darbe zaman grafiği

Ultrasonik darbe t=0 zamanında transdüser tarafından iletiliyor. X pozisyonundaki hedef tarafından yansıtıldıktan sonra t= tx zamanında darbe alınıyor. tx ; X mesafesi ile orantılıdır.

T=0 zamanında darbe iletilir (ultrasonic ses sinyali), cisimden yansır, transdüser tarafından algılanır ve tekrar gönderilir. Sonraki darbe ilk darbenin ultrasonik enerjisinin hepsi absorbe edildiğinde iletilmelidir. Bu yüzden sensöre bir pals gönderilir sensör okunur ve sensörün datasheetinde yazan süre kadar sensöre tekrar pals gönderilmez. Eğer bekleme yapmaksak sensör saçma değerler döndürür. Çünkü ilk yolladığımız sinyal bir yerden yansıyarak sensöre geri dönmeye devam eder.

Tüm katı ve sıvı cisimler ultrasonik dalgayı çok iyi oranda yansıtırlar. Hem katı hem de sıvı cisimlerden ultrasonik enerjinin %99u yansıtılır. Çok ufak oranlardaki enerji miktarı cisim tarafından emilir. Bundan dolayı sensörü çok çeşitli uygulamalarla da sorunsuz kullanabilmemiz mümkündür. Ayrıca robotlarda da sıkça kullanılmaktadır. Aşağıdaki resim bu tarz uygulamalara güzel bir örnektir.

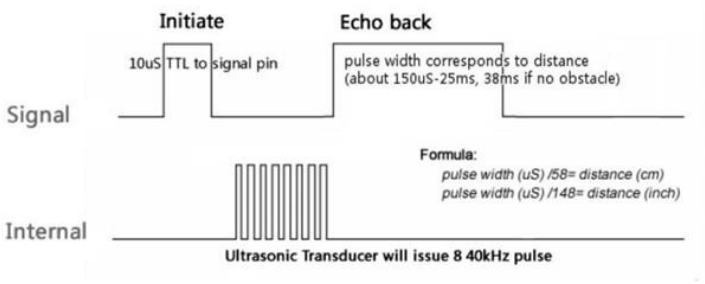


ŞEKİL 1.3.3. Sensör kullanım alanlarına örnek

HC-SR04 Sensörü

ŞELİK 1.3.4. ön taraf görünüm

HC-SR04 sensör üzerinde 4 adet pin mevcut. Bunlar; vcc, gnd, trig, echo pinleri. Sensörü kullanmak için trig pininden yaklaşık 10us'lik bir pals gönderiyoruz. Sensör kendi içerisinde 40khz frekansında bir sinyal üretip 8 pals verici transdüsere gönderiyor. Bu ses dalgası havada, deniz seviyesinde ve 15 °C sıcaklıkta 340 m/s bir hızla yol alır. Bir cisme çarpar ve geri sensöre yansır. Cismin sensörden uzaklığı ile doğru orantılı olarak echo pini bir süre lojik 1 seviyesinde kalır ve tekrar lojik 0 olur. mesafeyi ölçmek için tek yapmamız gereken echo pininin ne kadar lojik1 olduğunun süresini bulmaktır. Bu yapı aşağıdaki ŞEKİL 1.3.5. 'te de verilmiştir.



ŞEKİL 1.3.5. Lojik-1 gelme süresi

2. YAPILAN ÇALIŞMALAR

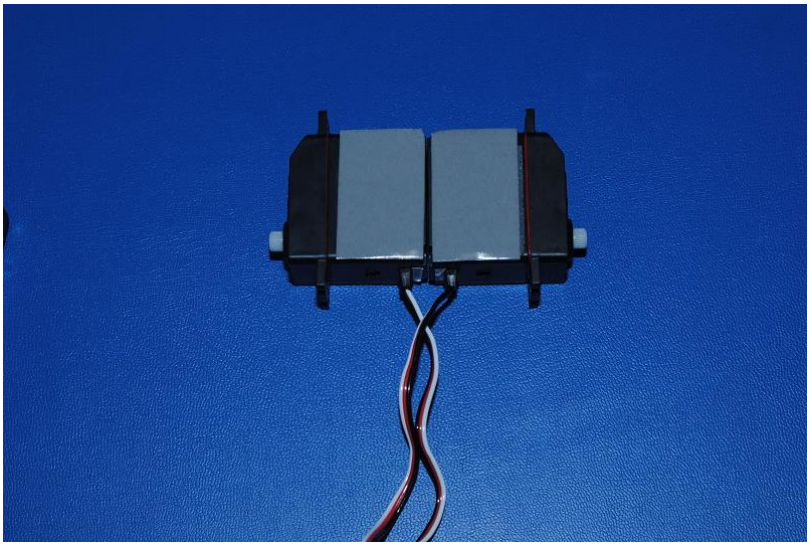
Yapılan tez çalışmasında engelden kaçan robot tasarımı gerçekleştirilmeye çalışılmıştır. Tasarımı iki kısım olarak görebiliriz. 1.kısım donanım ve tasarım aşaması. 2.kısım ise yazılım aşamasıdır. Donanım ve yazılım aşaması sırasında her bir parça ve kodun nasıl kullanıldığı paylaşılmıştır.

2.1. Robot Malzemeleri Montaj

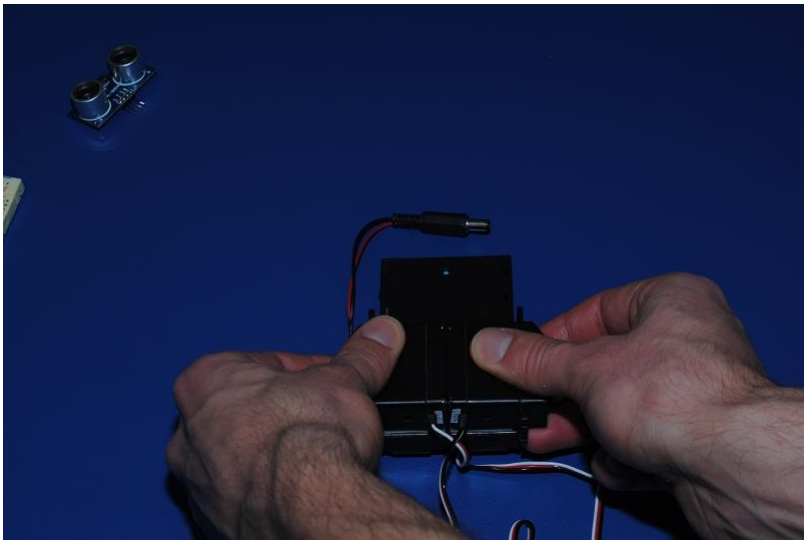
Robotumuzda pil yatağını alt yüzey olarak kullanıyoruz.

ŞEKİL 2.1.1. Pil yatağı

Servoların ürezine çift taraflı güçlü bantı yapıştırıyoruz.

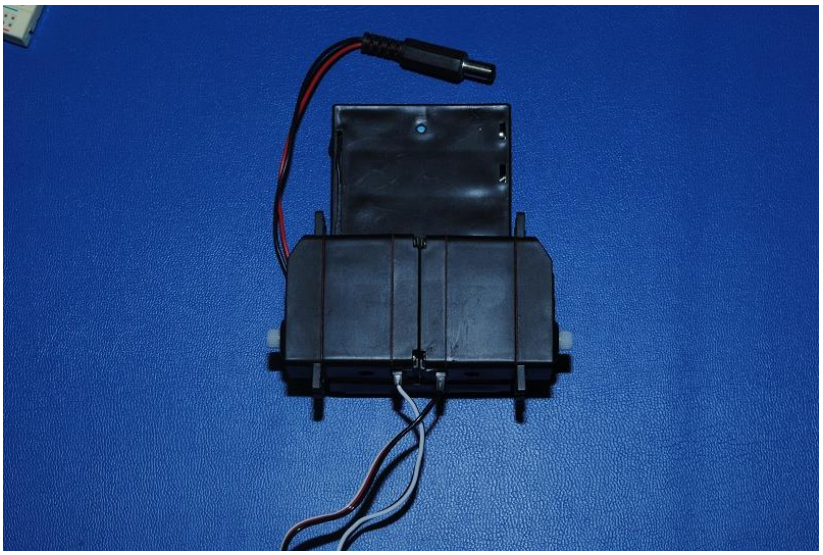


ŞEKİL 2.1.2. Servo motorlar



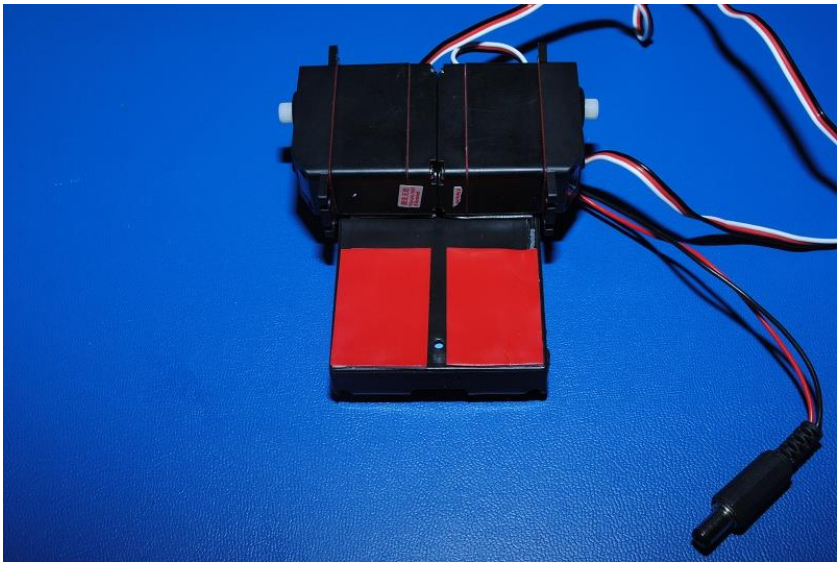
ŞEKİL2.1.3. Servo motorların alt kısmı ile pil yatağının bağlantısı

Servoları pil yatağının üstüne yapıştırıyoruz.

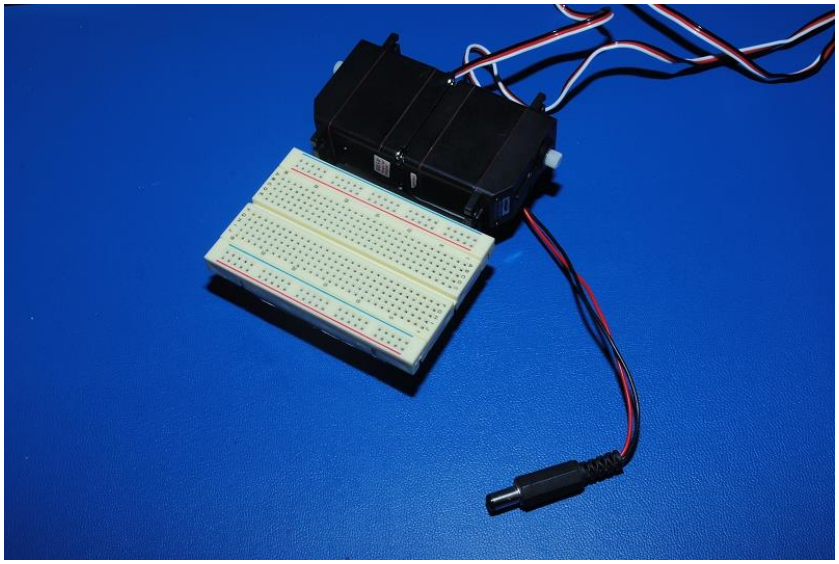


ŞEKİL.2.1.4. Pil yatağı ve servo motorlar

Pil yatağının diğer ucuna breadboard yerleştirirken yine çift taraflı güçlü bant kullanılmıştır.

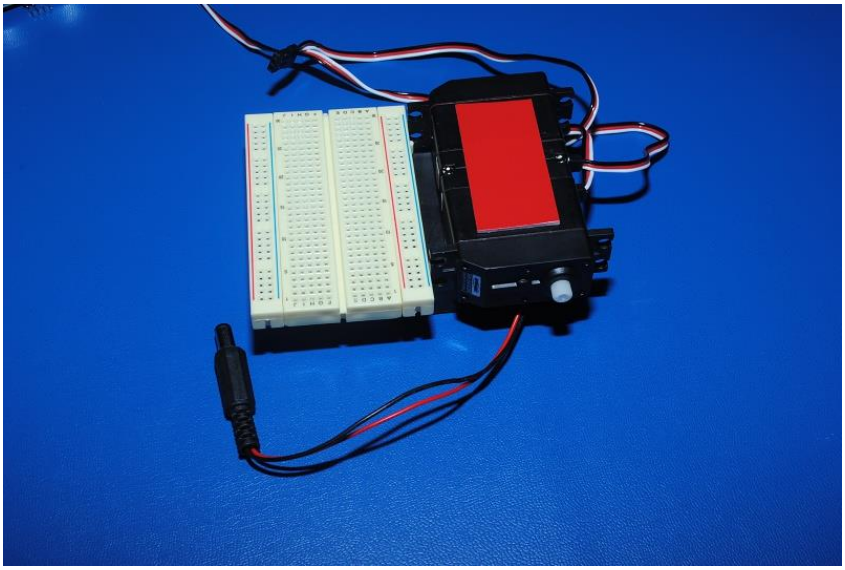


ŞEKİL2.1.5. Pil yatağına yapıştırılan güçlü çift taraflı yapıştırıcı

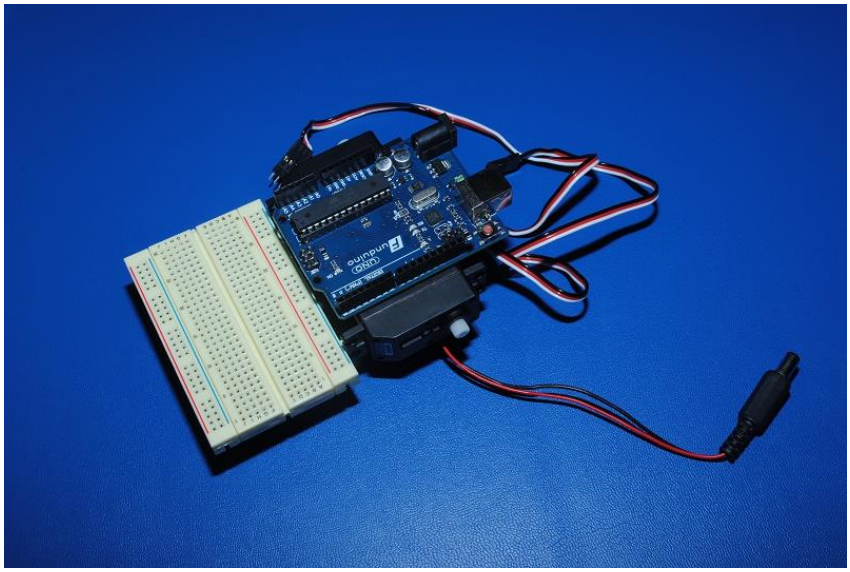


ŞEKİL 2.1.6 Breadboard

Arduinoyu robotta sabitlemek için bir yere ihtiyacımız olduğundun ve robotu olabildiğince kompakt yapmak istediğimizde servoların üstüne bantla yapıştırıyoruz.

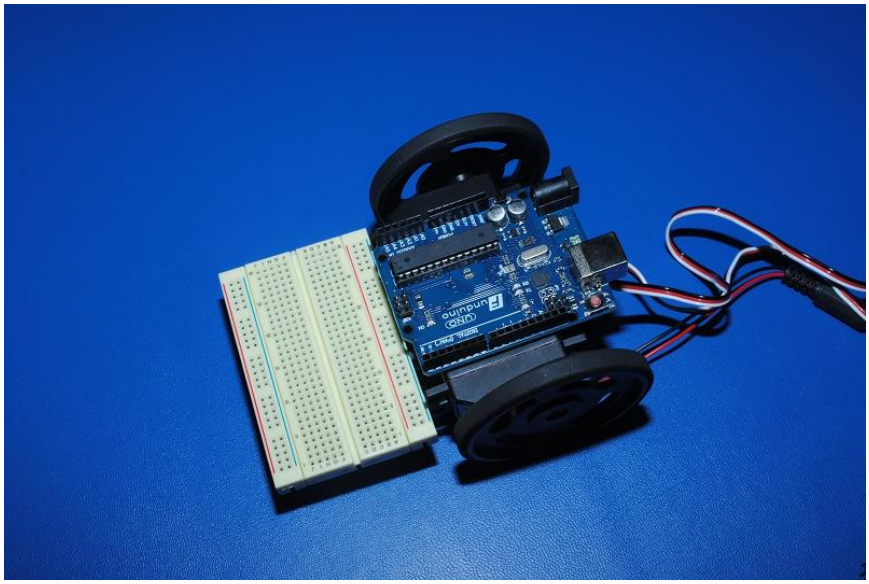


ŞEKİL 2.1.7 Arduinonun koyulacağı yerin belirlenmesi

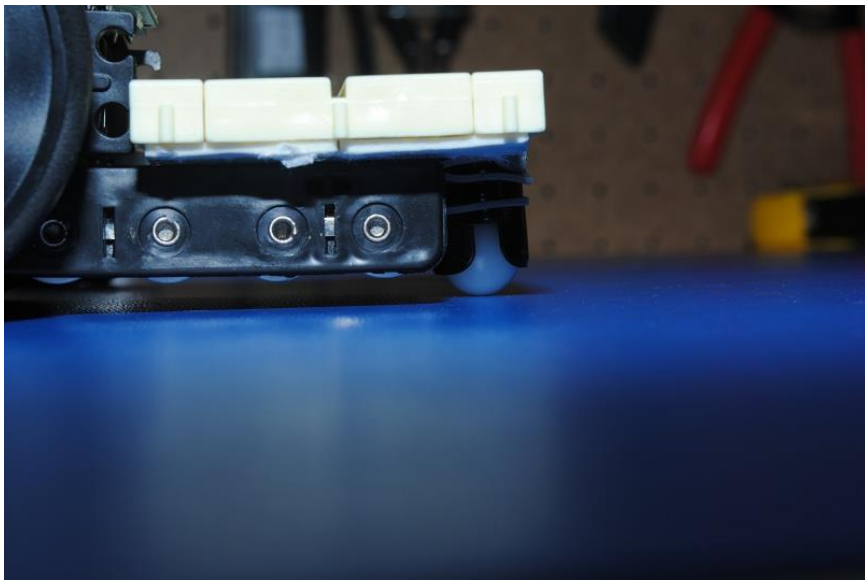


ŞEKİL 2.1.8 Ardunio yerleştirilmesi

Lastikleri servo motorlara takıyoruz ve sarhoş tekeri yerleştiriyoruz.

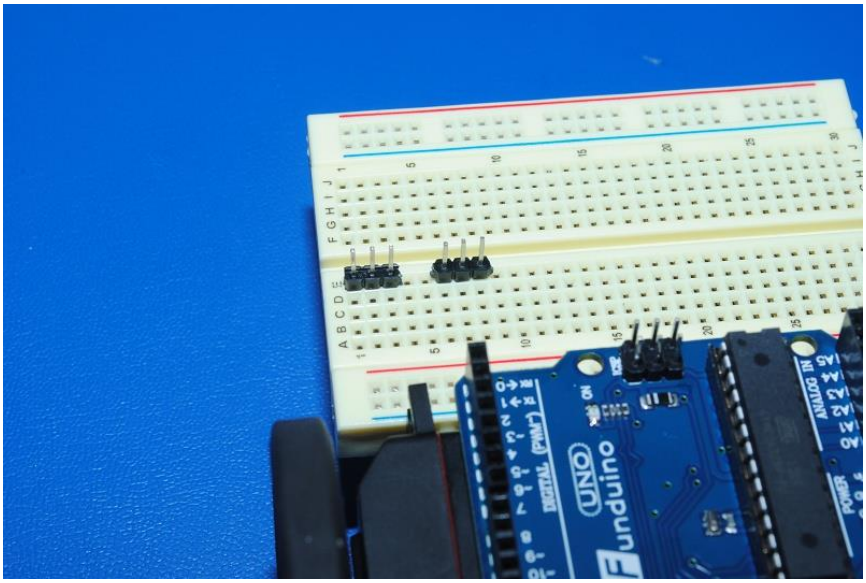


ŞEKİL 2.1.9. Lastik tekerleklerin takılmış halinin görünümü

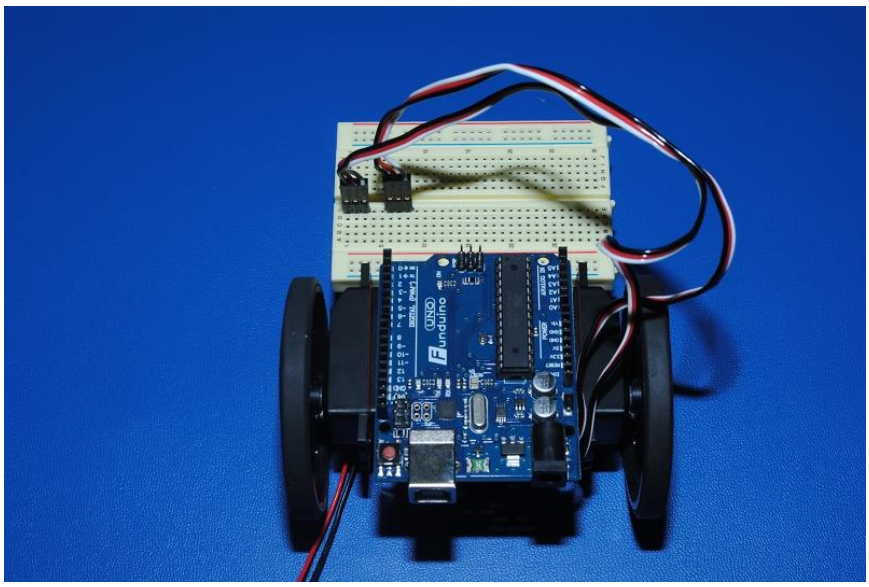


2.1.14 Sarhoş teker

2.2 Elektriksel Bağlantılar

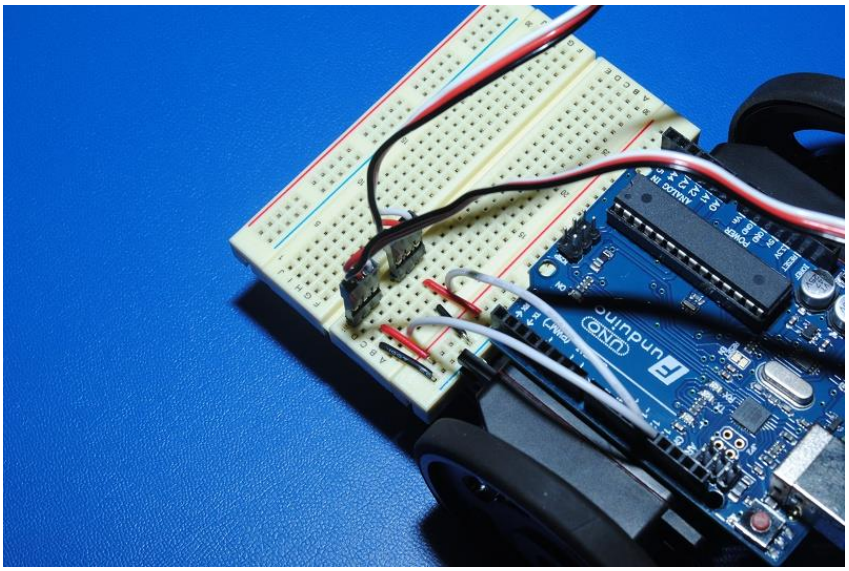


ŞEKİL 2.2.1. Pinlerin çakılması

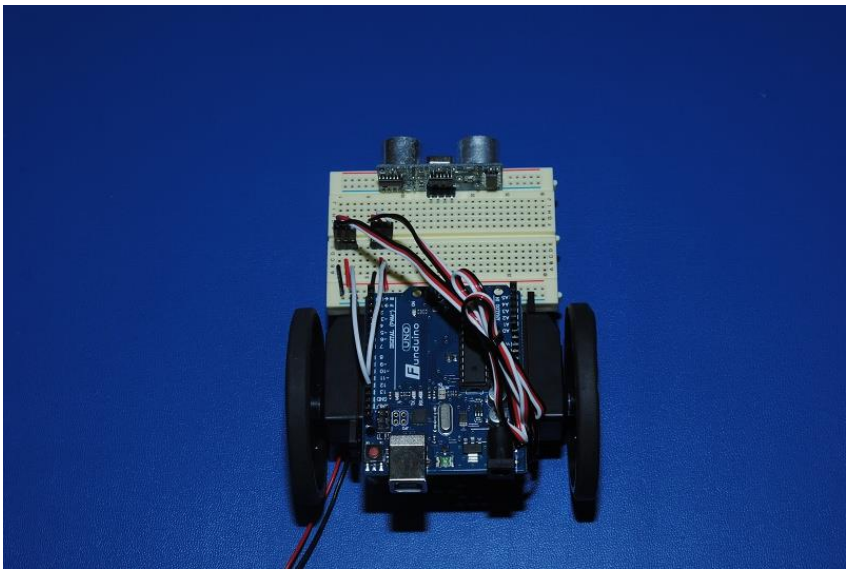


ŞEKİL 2.2.2. Ardunio bağlantıları

Breadboard arka tarafındaki pozitif (kırmızı) rayına servolar üzerinden kırmızı teller, breadboard arka tarafında GND(mavi) raya siyah teller ve arduino pimleri 12. & 13. Pimlere bağlandı

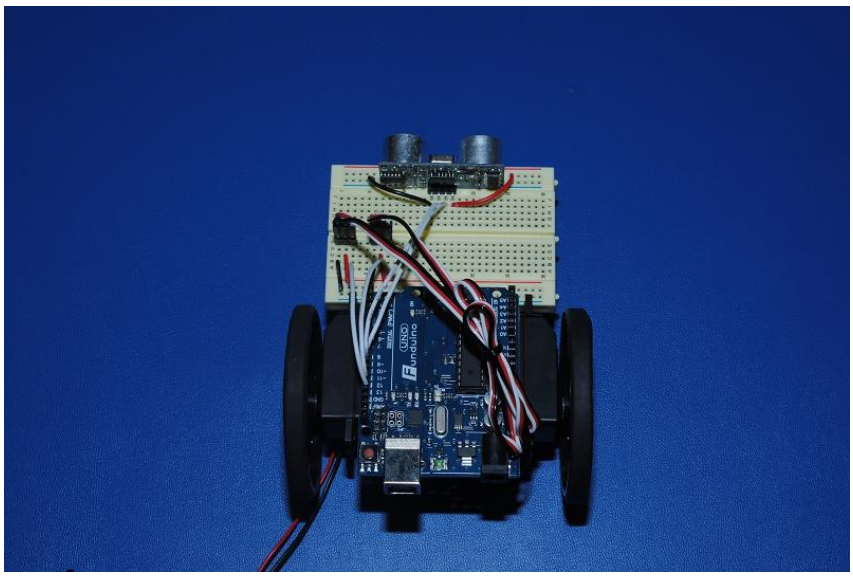


ŞEKİL 2.2.3. Breadboard üzerindeki bağlantı noktaları

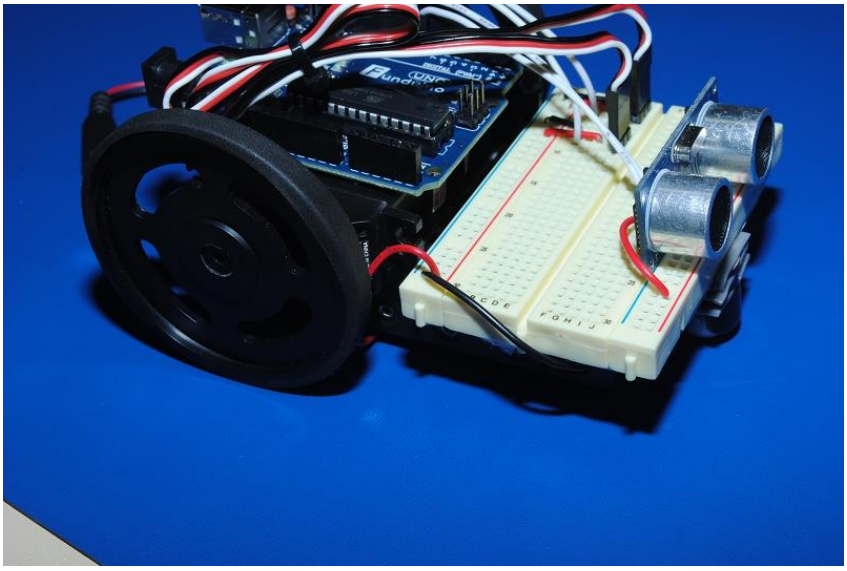


ŞEKİL 2.2.4. Ardunio bağlantıları

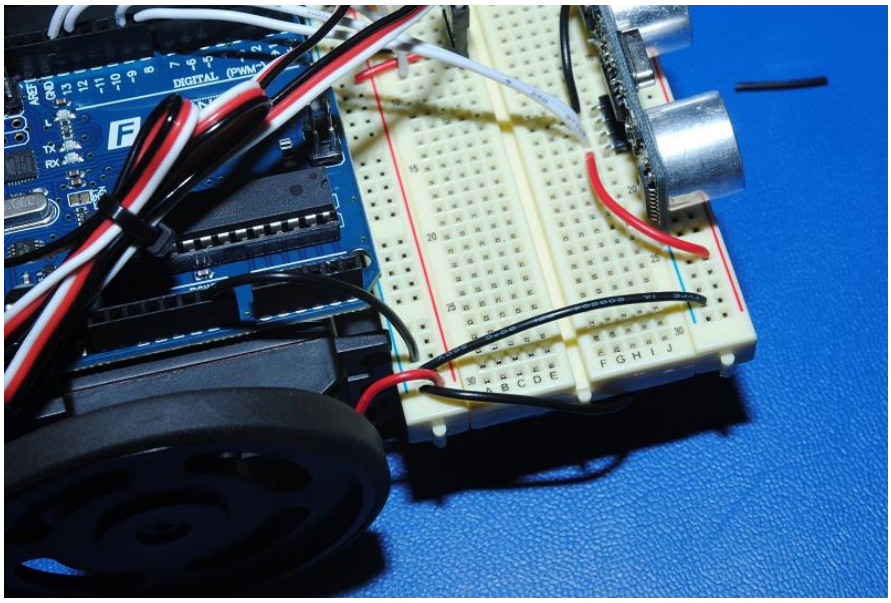
Sensörün ön kısmında, kart üzerindeki parlaklıklar pimleri gösterir. VCC bağlamak için kırmızı tel VCC güç rayına, GND'yi bağlanmak için siyah tel GND (mavi) güç rayına ve ardunino 8. (Trig) ve 9. (Echo) Pimleri için beyaz tel kullanılmıştır.



ŞEKİL 2.2.5. Sensürün yerleştirilmesi

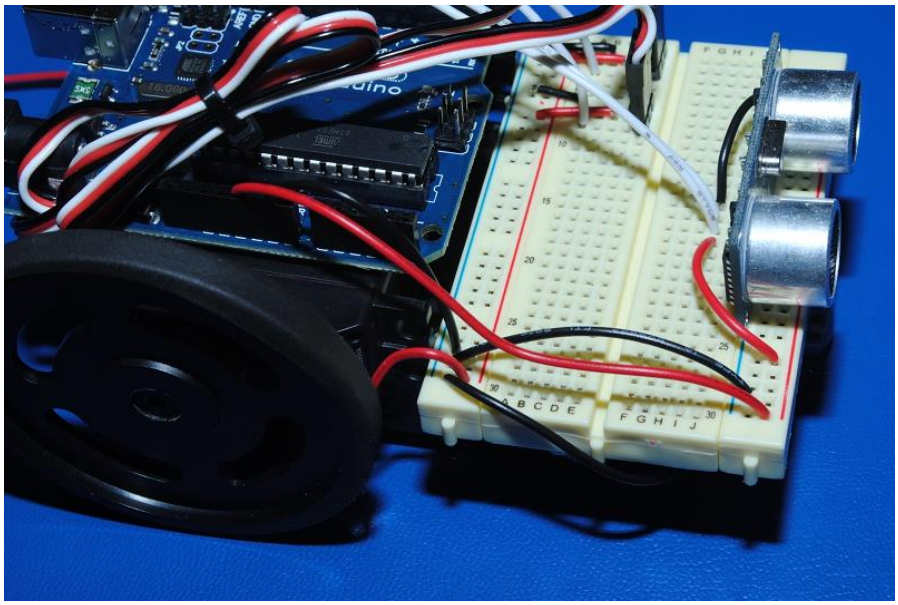


ŞEKİL 2.2.6. Sensör bağlantı noktaları



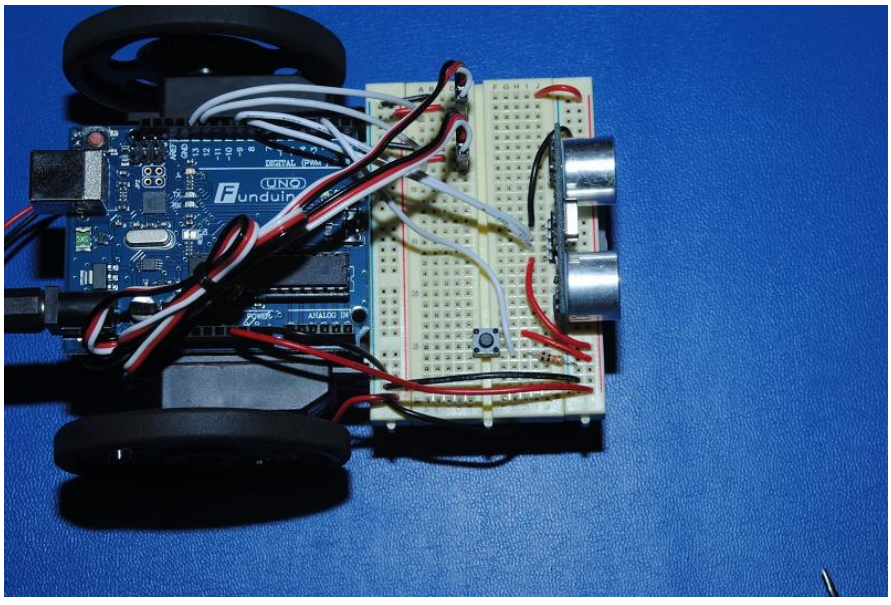
ŞEKİL 2.2.7 Sensör bağlantı noktaları

Şimdi, kırmızı teli Arduino 5V pinine bağlıyoruz.



ŞEKİL 2.2.8 Enerji bağlantıları

Son olarak robota kill switch ekleyerek robotun güç kaynağını sökmeden durdurulmasını sağlıyoruz.



ŞEKİL 2.2.9. Anahtarın yerleştirilmesi

2.3. YAZILIM

Robotun tasarlanan algoritmasını Ardunio IDE kullanarak robota aktarma kısmında kullanılan fonksiyonlar arduinonun sunduğu eğitim setlerinden öğrenilerek örnek fonksiyonlar üzerinden yazılmıştır.

2.3.1. "Hello World!"

```
void setup ()
{
Serial.begin(9600);
}
void loop ()
{
Serial.println ("Hello World!");
delay (500);
}
void setup ()
{
(9600) Serial.begin;
}
// loop () :
void loop ()
{
Serial.println ("Hello World!");
delay (500);
}
```

Döngü () Arduino açık olduğu sirece çağrı tutan özel bir fonksiyondur . Bu programın gövdesi gerçekten basit ve sadece iki satırdır. İlk satır (geri bilgisayara, bu durumda) seri bağlantı noktası üzerinden metin göndermek için Serial println() komutu, ikinci satır her döngüye yarım saniye gecikme sağlar

2.3.2 Servo Motorlar

Robot üzerinde servo motor kullanmak için yazılmış program verilmiştir.

```
#include <Servo.h>

/ / Servo nesnelerini yaratıyoruz
Servo leftMotor;
Servo rightMotor;

void setup ()
{
   leftMotor.attach (13);
   rightMotor.attach (12);
}
void loop ()
{
   leftMotor.write (180);
   rightMotor.write (180);
}
```

2.3.2. Kill Switch

Her seferinde Bataryayı çıkarmak sıkıntı yaratacağı için kill swich ekleyerek robotun çalışmasını beklemeye alacak program verilmiştir.bu işlemi döngü ile gerçekleştiriyoruz.

```
if(digitalRead(2) == HIGH) // anahtara basıldı mı?
{
   while(1)
   {
      leftMotor.write(90);
      rightMotor.write(90);
   }
}
```

Servo programının içine ekleyerek robotun istenildiğinde kolaylıkla durdurulmasını sağlıyoruz.

```
#include<Servo.h>
```

```
/ / Servo nesneleri yaratıyoruz
Servo leftMotor;
Servo rightMotor;
void setup ()
{
   leftMotor.attach (13);
   rightMotor.attach (12);
}
void loop ()
{
   (digitalRead (2) == HIGH) / / buton anahtarı basıldığında
   {
      while (1)
      {
         leftMotor.write (90);
         rightMotor.write (90);
      }
   }
   leftMotor.write (180);
   rightMotor.write (180);
}
```

3.3.4. Ultrasonik sensör

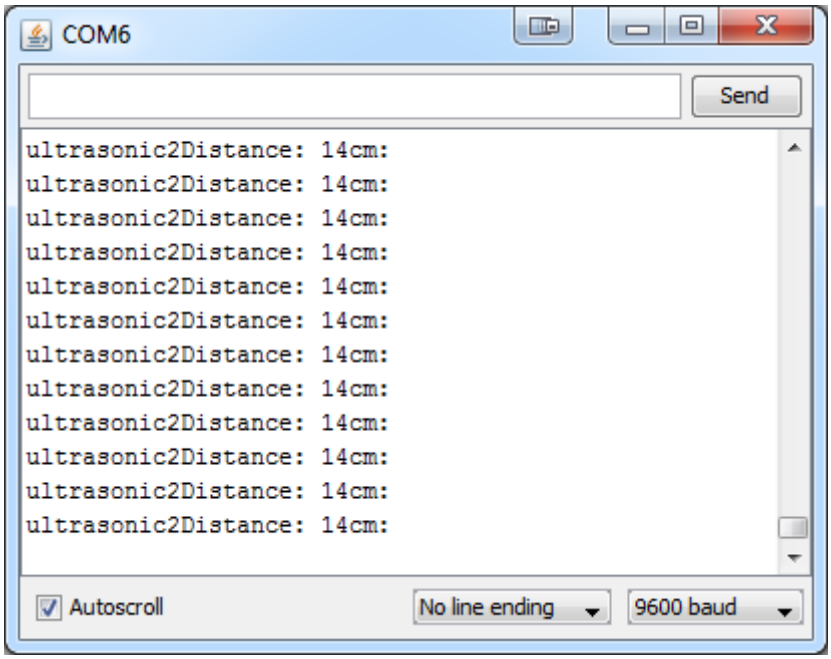
Arduino ile bir HC-SR04 ultrasonik sensör kullanarak sensörün çalışması için örnek program aşağıda verilmiştir.

```
const int serialPeriod = 250; / / Sadece her 1/4 saniye seri konsola yazdırmak
unsigned long timeSerialDelay=0;
const int loopPeriod = 20; / / 20ms = bir süre 50Hz frekans
unsigned long timeoopDelay=0;
/ / Ultrasonik sensörler için kullanılan trigonometrik ve yankı işaretçilerine belirtin
const int ultrasonic2TrigPin=8;
cons int ultrasonic2EchoPin=9;
int ultrasonic2Distance;
int ultrasonic2Duration;


void setup ()
{
   Serial.begin(9600);
```

```
  / / Ultrasonik sensör pin konfigürasyonları
  pinMode (ultrasonic2TrigPin, OUTPUT);
  pinMode (ultrasonic2EchoPin, INPUT);
}
void loop ()
{
  debugOutput ();// Seri konsol hata ayıklama iletileri

  if (millis () - timeLoopDelay> = loopPeriod)
  {
    readUltrasonicSensors (); / / Okuma ve ölçülen mesafeleri saklamak için

    timeLoopDelay = millis ();
  }
}
void readUltrasonicSensors ()
{
  / / Ultrasonik 2
  digitalWrite (ultrasonic2TrigPin, HIGH);
  delayMicroseconds (10); / / En az 10us için trigonometrik pini yüksek tutmalıyız
  digitalWrite (ultrasonic2TrigPin, LOW);

  ultrasonic2Duration = pulseIn (ultrasonic2EchoPin, HIGH);
  ultrasonic2Distance = (ultrasonic2Duration / 2) / 29;
}

void debugOutput ()
{
  if ((millis () - timeSerialDelay)> serialPeriod)
  {
    Serial.print ("ultrasonic2Distance:");
    Serial.print (ultrasonic2Distance);
    Serial.print ("cm: ");
    Serial.println ();
    timeSerialDelay = millis ();
  }
}
```

2.3.5. Ultrasonik Sensör ile Gezinme

Bu aşamadan sonra elde edilen programları birleştirerek sensörden gelen değerlere karşılık olarak robotun belirlenmiş hareketlerini programa ekliyoruz.



#include <Servo.h>

/ / Servo nesneler oluşturucaz

Servo leftMotor;

Servo rightMotor;

kurulum () fonksiyonu servo nesneleri yapılandırılacak:

leftMotor.attach (13);

rightMotor.attach (12);

bu noktadan sonra aşağıdaki gibi devam edicez

```
# include<Servo.h>
/ / Servo nesneler oluşturulacak
Servo leftMotor;
Servo rightMotor;
const int serialPeriod = 250; / / Sadece her 1/4 saniyede bir seri konsola yazdır
unsigned lon timeSerialDelay=0;
const int loopPeriod = 20; / / 20ms = bir süre 50Hz frekans
unsigned long timeLopDelay=0;
/ / Ultrasonik sensörler için kullanılan trigonometrik ve yankı işaretçilerini belirticez
const int ultrasonic2TrigPin=8;
const int ultrasonic2EchoPin=9;
int ultrasonic2Distance;
int ultrasonic2Duration;
void setup ()
{
   Serial.begin(9600);
   / / Ultrasonik sensör pin konfigürasyonları
   pinMode (ultrasonic2TrigPin, OUTPUT);
   pinMode (ultrasonic2EchoPin, İNPUT);
   leftMotor.attach (13);
   rightMotor.attach (12);
}
void loop ()
{
   (digitalRead (2) == HIGH) / / buton anahtarı basıldığında ise eğer
```

```
  {
    while (1)
    {
      leftMotor.write (90);
      rightMotor.write (90);
    }
  }
  debugOutput (); Seri konsol hata ayıklama iletileri
  if (millis () - timeLoopDelay> = loopPeriod)
  {
    readUltrasonicSensors (); / / Okuma ve ölçülen mesafeleri saklanolacak
    timeLoopDelay = millis ();
  }
}
void readUltrasonicSensors ()
{
  / / Ultrasonik 2
  digitalWrite (ultrasonic2TrigPin, HIGH);
  delayMicroseconds (10); / / En az 10US için trigonometrik pin yüksek tutmalıyız
  digitalWrite (ultrasonic2TrigPin, LOW);
  ultrasonic2Duration = pulseIn (ultrasonic2EchoPin, HIGH);
  ultrasonic2Distance = (ultrasonic2Duration / 2) / 29;
}
void debugOutput ()
{
```

```
  if ((millis () - timeSerialDelay)> serialPeriod)
  {
    Serial.print ("ultrasonic2Distance:");
    Serial.print (ultrasonic2Distance);
    Serial.print ("cm");
    Serial.println ();
    timeSerialDelay = millis ();
  }
}
```

Şimdi robotun hareketlerini ekliyoruz:

- Önünde engel yoksa ileri git
- Robotun önünde bir engel varsa, sola dönün
- Bunu gerçekleştirmek için sonlu durum makinesi kullanacağız.
-

```
/ / Durumlarını tanımlıyoruz
#define DRIVE_FORWARD 0
#define TURN_LEFT 1
 İn state= DRIVE_FORWARD; / / 0 = (DEFAULT) ileri sür, 1 = sola çevir
if (state == DRIVE_FORWARD) / / herhangi bir engel algılanırsa
{
}
else if (state == TURN_LEFT) / / engel algılandı - sola dönüyor
{
}
```

DRIVE_FORWARD fonksiyonu ile robotun hareket değişkenlerini tanımlıyoruz. Bu fonksiyonla robot ileri hareket etmektedir.

```
if (ultrasonic2Distance> 6 | | ultrasonic2Distance <0) / / önümüzde bir şey yoksa (not: bazı sensörler için  aralıkta hiçbir şey olmaması ultrasonicDistance=0 olumsuz sonuçlar doğurabilir)
{
   / / Ileri sürme
   rightMotor.write (180);
   leftMotor.write (0);
}
else / / önümüzde bir nesne var
{
   state = TURN_LEFT;
}
//TURN_LEFT  için kod:
Unsigned long  timeToTurnLeft=500; / / 90 derece dönüş 0,5 saniye sürer
Unsigned long turnStartTime = millis (); / / dönüş başladığında zamandan tasarruf
While ((millis ()-turnStartTime) <timeToTurnLeft)
 / /  timeToTurnLeft kadar (1.1 saniye) kadar döngüde kal
{
   / / Sola dönüş
   rightMotor.write (180);
   leftMotor.write (180);
}
state = DRIVE_FORWARD;
```

Bu aşamadan sonra tüm kodları birleştirerek robotumuzun yazılım kısmını tamamlayacağız.

```
#include <Servo.h>
/ / Servo nesneleri oluşturuyoruz
Servo leftMotor;
Servo rightMotor;
const int serialPeriod = 250; / / Sadece her 1/4 saniyede  seri konsola yazdır
unsigned long timeSerialDelay=0;
const int loopPeriod = 20; / / 20ms = bir süre 50Hz frekans
unsigned long timeLoopDelay =0;
/ / Ultrasonik sensörler için kullanılan trigonometrik ve yankı işaretçilerini belirtiyoruz
const int ultrasonic2TrigPin=8;
const int ultrasonic2EchoPin=9;
int ultrasonic2Distance;
int ultrasonic2Duration;
/ / Durumlarını tanımlıyoruz
# define DRIVE_FORWARD   0
#define  TURN_LEFT   1
İnt state = DRIVE_FORWARD ; / / 0 = ileri sür(DEFAULT) , 1 = sola çevirin
void setup ()
{
   Serial.begin(9600);
   / / Ultrasonik sensör pin konfigürasyonları
   pinMode (ultrasonic2TrigPin, OUTPUT);
   pinMode (ultrasonic2EchoPin, INPUT);
```

```
    leftMotor.attach (13);
    rightMotor.attach (12);
}
void loop ()
{
    İf (digitalRead (2) == HIGH) / / buton anahtarı basıldığında
    {
        while (1)
        {
            leftMotor.write (90);
            rightMotor.write (90);
        }
    }
    debugOutput ();// Seri konsol hata ayıklama iletileri
    if (millis () - timeLoopDelay> = loopPeriod)
    {
        readUltrasonicSensors (); / / Okuma ve ölçülen mesafeleri sakla
        Statemachine ();
        timeLoopDelay = millis ();
    }
}
void Statemachine ()
{
    if (state == DRIVE_FORWARD) / / herhangi bir engel algılanırsa
    {
```

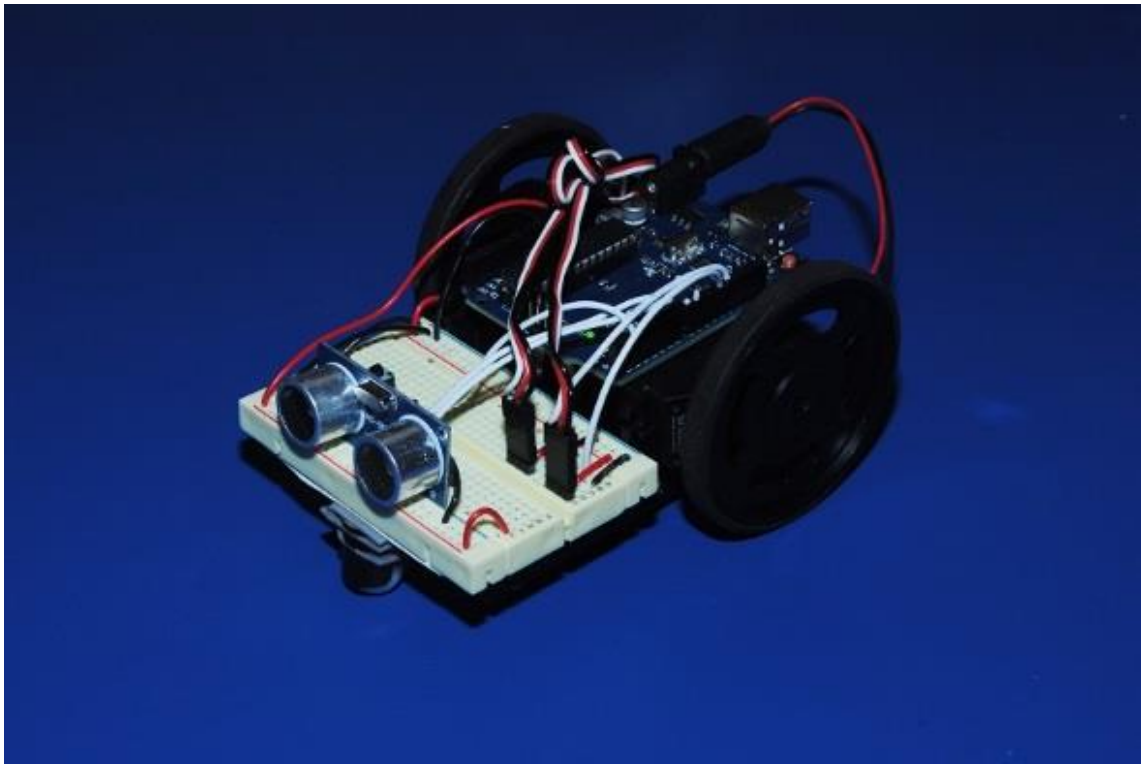
```
      if (ultrasonic2Distance> 6 | | ultrasonic2Distance <0) / / hiçbir şey önümüzde yoksa
      {
         / / Ileri sür
         rightMotor.write (180);
         leftMotor.write (0);
      }
      else / / önümüzde bir nesne var
      {
         state = TURN_LEFT;
      }
   }
   else if (state == TURN_LEFT) / / engel algılandı - turn left
   {
      Unsigned long timeToTurnLeft=500; / / 90 derece dönmesi 0,5 sn sürer
      unsigned long turnStartTime = millis (); / / dönüş başladığında zamandan tasarruf
      while((millis ()-turnStartTime) <timeToTurnLeft)  / / timeToTurnLeft kadar (0,5 saniye)
//döngüde kal
      {
         / / Sola dönüş
         rightMotor.write (180);
         leftMotor.write (180);
      }
      state = DRIVE_FORWARD;
   }
}
void readUltrasonicSensors ()
```

```
{
  / / Ultrasonik 2
  digitalWrite (ultrasonic2TrigPin, HIGH);
  delayMicroseconds (10); / / En az 10US için trigonometrik pin yüksek tutmalıyız
  digitalWrite (ultrasonic2TrigPin, LOW);
  ultrasonic2Duration = pulseIn (ultrasonic2EchoPin, HIGH);
  ultrasonic2Distance = (ultrasonic2Duration / 2) / 29;
}
/ /
void debugOutput ()
{
  if ((millis () - timeSerialDelay)> serialPeriod)
  {
    Serial.print ("ultrasonic2Distance:");
    Serial.print (ultrasonic2Distance);
    Serial.print ("cm");
    Serial.println ();
    timeSerialDelay = millis ();
  }
}
```

Tüm çalışma ve örnek programlarını birleştirerek Ardunio ile engelden kaçan robot programını tamamlanmıştır.

3. SONUÇLAR

Projede, ardunio ile engelden kaçan robot gerçekleştirilmiş oldu. Projenin amacı herhangi bir engelle karşılaştığında yön değiştirerek engelden kaçan bir robot tasarlanmasıdır.



KAYNAKLAR

http://letsmakerobot...node/list/robot

http://forums.trosse.com/robots.php