NECMETTİN ERBAKAN ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

MEKATRONİK MÜHENDİSLİĞİ

MEKATRONİK MÜHENDİSLİĞİ UYGULAMALARI

PROJE RAPORU

AD: Mevlüt

SOYAD: KOÇ

İÇİNDEKİLER

PROJENİN AMACI

Projenin amacı, kablosuz iletişim ve hareket algılama teknolojilerini bir araya getirerek hem manuel hem de otonom olarak çalışabilen bir otomasyon aracı geliştirmektir. Bu proje, robotik sistemlerin kontrolü ve otomasyonu konusunda teorik bilgiyi pratiğe dönüştürmeyi hedefler. Aynı zamanda, sensör verilerinin kullanımı, veri kaydı, oynatma mekanizmaları ve engelden kaçınma algoritmaları gibi önemli teknolojik yaklaşımların uygulanmasını içerir.

Projenin temel amaçları şunlardır:

1. **Kablosuz İletişim:** Verici ve alıcı modülleri (NRF24L01) kullanılarak araç ile kontrolcü arasında güvenilir bir iletişim altyapısı oluşturmak.
2. **Hareket Algılama:** MPU6050 sensörü aracılığıyla el hareketlerini algılamak ve bu hareketlere karşılık gelen komutları araca ileterek araç hareketlerini kontrol etmek.
3. **Engelden Kaçınma:** HC-SR04 ultrasonik mesafe sensörü ile aracın çevresindeki engelleri algılamasını ve bu engellerden kaçınmasını sağlamak.
4. **Kayıt ve Oynatma:** Araç hareketlerini kaydedip, gerektiğinde bu hareketlerin tekrar oynatılmasını mümkün kılan bir mekanizma geliştirmek.
5. **Otonom ve Manuel Modlar:** Araç, hem manuel kontrol hem de otonom bir şekilde çalışabilecek şekilde tasarlanmıştır. Bu sayede kullanıcı hem doğrudan kontrol sağlayabilir hem de aracın belirli görevleri bağımsız olarak yerine getirmesini izleyebilir.

Bu proje, robotik sistemlerin gerçek dünya problemlerine uygulanabilirliğini artırmak için tasarlanmıştır. Aynı zamanda, öğrencilere ve geliştiricilere elektronik devre tasarımı, kablosuz iletişim, sensör entegrasyonu ve yazılım geliştirme konularında kapsamlı bir öğrenme deneyimi sunar.

GİRİŞ

Kullanılan Teknoloji ve Ekipmanlar

Bu projede, modern elektronik, sensör teknolojisi, kablosuz iletişim ve mikrodenetleyici tabanlı sistemler bir araya getirilerek işlevsel bir otomasyon aracı tasarlanmıştır. Kullanılan teknoloji ve ekipmanlar, projenin amacına uygun olarak seçilmiş ve entegre edilmiştir. Aşağıda bu teknolojiler ve ekipmanların detaylı açıklamaları yer almaktadır:

1. Kablosuz İletişim: NRF24L01 Modülü

- **Özellikleri:** NRF24L01, düşük güç tüketimi ve yüksek veri aktarım hızıyla bilinen bir 2.4 GHz kablosuz iletişim modülüdür. Verici ve alıcı arasında güvenilir bir veri iletimi sağlar.
- **Kullanımı:** Bu projede, verici birimi (Arduino Nano) ile alıcı birimi (Arduino Uno) arasında kablosuz bağlantıyı sağlamak için kullanılmıştır. Hareket komutları ve sensör verileri bu modül aracılığıyla iletilmiştir.

2. Hareket Algılama: MPU6050 İvmeölçer ve Jiroskop Sensörü

- **Özellikleri:** MPU6050, hem ivmeölçer hem de jiroskop sensörlerini bir arada bulunduran bir modüldür. X, Y ve Z eksenlerinde hareket ve dönüş bilgilerini algılar.
- **Kullanımı:** Projede, kullanıcı el hareketlerini algılamak ve bu hareketlere karşılık gelen komutları üretmek için kullanılmıştır. Eldiven üzerine yerleştirilen bu sensör, aracın manuel kontrolünde önemli bir rol oynamaktadır.

3. Engelden Kaçınma: HC-SR04 Ultrasonik Mesafe Sensörü

- **Özellikleri:** HC-SR04, ses dalgalarını kullanarak mesafe ölçümü yapan bir sensördür. Yüksek doğruluk ve düşük maliyet avantajı sunar.
- **Kullanımı:** Araç üzerindeki engelleri algılamak ve engellerden kaçınma algoritmasını çalıştırmak için kullanılmıştır. 8 cm'nin altındaki mesafelerde aracın durması ve geri hareket etmesi sağlanmıştır.

4. Mikrodenetleyiciler: Arduino Nano ve Arduino Uno

- **Özellikleri:** Arduino platformu, açık kaynaklı bir elektronik geliştirme platformudur. Nano, kompakt yapısı ile verici modülünde; Uno ise daha geniş giriş/çıkış seçenekleri ile alıcı modülünde kullanılmıştır.
- **Kullanımı:** Arduino Nano, hareket algılama ve komut gönderimi görevlerini yerine getirirken; Arduino Uno, araç üzerindeki motor kontrolü ve engelden kaçınma gibi işlemleri gerçekleştirmiştir.

5. Motor Kontrol: L298N Motor Sürücü

- **Özellikleri:** L298N, iki DC motoru bağımsız olarak kontrol edebilen bir motor sürücü modülüdür. Yüksek akım kapasitesi ve kolay kullanım sağlar.
- **Kullanımı:** Araçta kullanılan motorların hız ve yön kontrolü bu sürücü ile gerçekleştirilmiştir.

6. Hafıza Yönetimi: EEPROM AT24C32

- **Özellikleri:** EEPROM, mikrodenetleyicinin enerjisi kesildiğinde bile verileri saklayabilen bir hafıza birimidir.
- **Kullanımı:** Araç hareketlerinin kaydedilmesi ve oynatılması için hareket verileri EEPROM'da saklanmıştır.

7. Araç Gövdesi ve Motorlar

- **Gövde:** Araç, dayanıklı bir şasi ve bir tekerlek sistemi üzerine inşa edilmiştir. Bir yönlendirici tekerlek (caster wheel) kullanılmıştır.
- **Motorlar:** Araçta, L298N ile kontrol edilen iki DC motor kullanılmıştır. Bu motorlar, aracın ileri, geri, sağa ve sola hareketlerini sağlar.

8. LED ve Butonlar

- **LED:** Kullanıcıya kayıt ve oynatma durumlarını göstermek için bir LED kullanılmıştır.
- **Butonlar:** Bir buton kayıt işlemini başlatırken, diğeri kaydedilen hareketlerin oynatılmasını sağlar.

9. Güç Kaynağı

- Araç ve kontrol sistemleri, 9V bataryalarla beslenmiştir. Yeterli enerji sağlamak ve sistem stabilitesini korumak için uygun kapasitörler de eklenmiştir.

10. Yazılım ve Algoritmalar

- **Algoritmalar:** Mesafe algılama, engelden kaçınma, kayıt ve oynatma gibi işlevler için özel algoritmalar geliştirilmiştir.
- **Yazılım:** Arduino IDE kullanılarak tüm sistem için gerekli yazılımlar yazılmıştır. Verici ve alıcı birimleri arasında iletişim kurmak, sensör verilerini işlemek ve motorları kontrol etmek için kodlar optimize edilmiştir.

Projenin Kapsamı

Bu proje, kullanıcı tarafından kontrol edilen ve belirli düzeyde otonom hareket yeteneğine sahip bir araç geliştirilmesini kapsamaktadır. Sistem, el hareketleriyle manuel kontrol ve belirli hareketlerin kaydedilip tekrar oynatılması gibi işlevleri içermektedir. Aynı zamanda engelden kaçınma özelliği sayesinde güvenli bir şekilde çalışabilmektedir.

Proje aşağıdaki alanları kapsamaktadır:

1. **Manuel Kontrol:**
    - Kullanıcının eline takılan bir eldiven üzerinden MPU6050 sensörü yardımıyla el hareketleri algılanır.
    - Bu hareketler, araç üzerinde motorların hız ve yönünü belirlemek için komutlara dönüştürülür.
    - Kullanıcı, aracı ileri, geri, sağa ve sola hareket ettirebilir.
2. **Hareket Kaydı ve Oynatma:**
    - Araç, kullanıcı tarafından yapılan hareketleri kaydedebilir.
    - Kaydedilen hareketler daha sonra tekrar oynatılabilir, böylece araç belirli bir rotayı otonom şekilde izleyebilir.
    - Kayıt ve oynatma işlemleri, butonlarla kontrol edilir ve durum göstergesi olarak bir LED kullanılır.
3. **Engelden Kaçınma:**
    - Araç, önündeki engelleri algılamak için HC-SR04 ultrasonik mesafe sensörünü kullanır.
    - Sensör, 8 cm'den daha kısa bir mesafede bir engel algılarsa, araç durur ve geri hareket eder.
4. **Kablosuz İletişim:**
    - Verici ve alıcı birimleri arasında veri aktarımı, NRF24L01 modülleriyle sağlanır.
    - Bu modüller, eldiven ve araç arasındaki kablosuz bağlantıyı kurarak kullanıcı komutlarını hızlı ve güvenilir bir şekilde iletir.
5. **Motor Kontrol:**
    - L298N motor sürücü modülü ile araç üzerindeki motorların hız ve yönü kontrol edilir.
    - Bu kontrol, hem manuel hareketler hem de oynatma sırasında gerçekleşir.
6. **Güç Yönetimi ve Dayanıklılık:**
    - Sistem, 9V bataryalarla beslenir ve enerji kesintilerinde veri kaybını önlemek için EEPROM kullanılır.
    - Kapasitörler, enerji stabilitesini sağlamak için eklenmiştir.
7. **Geliştirilebilirlik:**
    - Sistem, ek sensörler veya işlevler eklenerek daha da genişletilebilir.
    - Bu kapsam, hem eğitim hem de endüstriyel uygulamalarda kullanılabilir bir prototip sunar.

YÖNTEM

Kullanılan Yazılımlar ve Kütüphaneler

Alıcı Kodu:

```
#include <nRF24L01.h>

#include <RF24.h>

RF24 radio(8, 7);  // CE, CSN pinleri
const byte addresses[][6] = {"00001", "00002"};
char gelenKomut;

const int motorA_IN1 = 3;
const int motorA_IN2 = 4;
const int motorB_IN1 = 5;
const int motorB_IN2 = 6;

const int highSpeed = 250;  // Hızlı hareket için hız
const int turnSpeed = 150;  // Dönüş sırasında düşük hız

const int trigPin = 10;
const int echoPin = 9;
const int ledPin = 2;

unsigned long lastSensorCheck = 0;  // Son sensör kontrol zamanı
const unsigned long sensorInterval = 20;  // Sensör kontrol aralığını daha
kısa yap (ms)

bool engelVarMi = false;

void setup() {
  pinMode(motorA_IN1, OUTPUT);
  pinMode(motorA_IN2, OUTPUT);
  pinMode(motorB_IN1, OUTPUT);
  pinMode(motorB_IN2, OUTPUT);

  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(ledPin, OUTPUT);

  Serial.begin(115200);
  radio.begin();
  radio.openWritingPipe(addresses[0]);  // Yazma için adres
  radio.openReadingPipe(1, addresses[1]);  // Okuma için adres
  radio.setPALevel(RF24_PA_MIN);
  radio.startListening();  // Alıcı moduna geç
```

```
}

void loop() {
  // Mesafe ölçümünü belirli aralıklarla yap
  if (millis() - lastSensorCheck >= sensorInterval) {
    lastSensorCheck = millis();
    long distance = mesafeOlc();
    engelVarMi = (distance < 8);

    if (engelVarMi) {
      digitalWrite(ledPin, HIGH);
      dur();  // Engelle karşılaşıldığında dur
      geriGit();  // 0.3 saniye geri git
      delay(30);  // 0.3 saniye bekle
    } else {
      digitalWrite(ledPin, LOW);
    }
  }

  // Komutları oku
  if (radio.available()) {
    radio.read(&gelenKomut, sizeof(gelenKomut));  // Komutu al
    Serial.print("Gelen komut: ");
    Serial.println(gelenKomut);

    if (!engelVarMi) {  // Engel yoksa hareket et
      switch (gelenKomut) {
        case 'I': ileriGit(); break;
        case 'G': geriGit(); break;
        case 'R': ileriSaga(); break;
        case 'L': ileriSola(); break;
        case 'D': duzSaga(); break;
        case 'Z': duzSola(); break;
        default: dur(); break;
      }
    } else {
      dur();  // Engel varken hareketi durdur
    }
  }
}

long mesafeOlc() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
```

```
long duration = pulseIn(echoPin, HIGH, 30000);  // 30ms timeout
  return (duration == 0) ? 500 : (duration * 0.034 / 2);  // Ölçüm hatası
durumunda max mesafe döndür
}

void ileriGit() {
  analogWrite(motorA_IN1, highSpeed);
  digitalWrite(motorA_IN2, LOW);
  analogWrite(motorB_IN1, highSpeed);
  digitalWrite(motorB_IN2, LOW);
}

void geriGit() {
  digitalWrite(motorA_IN1, LOW);
  analogWrite(motorA_IN2, highSpeed);
  digitalWrite(motorB_IN1, LOW);
  analogWrite(motorB_IN2, highSpeed);
}

void ileriSaga() {
  analogWrite(motorA_IN1, highSpeed);  // Sol motor tam hız
  digitalWrite(motorA_IN2, LOW);
  analogWrite(motorB_IN1, turnSpeed);  // Sağ motor düşük hız
  digitalWrite(motorB_IN2, LOW);
}

void ileriSola() {
  analogWrite(motorA_IN1, turnSpeed);  // Sol motor düşük hız
  digitalWrite(motorA_IN2, LOW);
  analogWrite(motorB_IN1, highSpeed);  // Sağ motor tam hız
  digitalWrite(motorB_IN2, LOW);
}

void duzSaga() {
  analogWrite(motorA_IN1, highSpeed);  // Sol motor tam hız
  digitalWrite(motorA_IN2, LOW);
  digitalWrite(motorB_IN1, LOW);  // Sağ motor durur
  digitalWrite(motorB_IN2, LOW);
}

void duzSola() {
  digitalWrite(motorA_IN1, LOW);  // Sol motor durur
  digitalWrite(motorA_IN2, LOW);
  analogWrite(motorB_IN1, highSpeed);  // Sağ motor tam hız
  digitalWrite(motorB_IN2, LOW);
}

void dur() {
```

```
analogWrite(motorA_IN1, 0);  // Sol motor durur
  digitalWrite(motorA_IN2, LOW);
  analogWrite(motorB_IN1, 0);  // Sağ motor durur
  digitalWrite(motorB_IN2, LOW);
}
```

Verici Kodu:

```
#include <EEPROM.h>
#include <Wire.h>
#include <nRF24L01.h>
#include <RF24.h>
#include <MPU6050.h>

#define BUTTON1_PIN 2  // Kayıt butonu
#define BUTTON2_PIN 3  // Oynatma butonu
#define LED_PIN 4      // Kayıt sırasında yanacak LED

#define MAX_HAREKET 1365  // 4096 byte / 3 byte (zaman ve hareket) = ~1365
hareket
#define KAYIT_SURESI 60000  // 1 dakika

RF24 radio(8, 7);  // CE, CSN pinleri
const byte addresses[][6] = {"00001", "00002"};
MPU6050 mpu;

char hareketKomut;
int hareketIndex = 0;
bool kayitAktif = false;
unsigned long kayitBaslangicZamani = 0;

void setup() {
  pinMode(BUTTON1_PIN, INPUT_PULLUP);
  pinMode(BUTTON2_PIN, INPUT_PULLUP);
  pinMode(LED_PIN, OUTPUT);

  Serial.begin(115200);
  Wire.begin();
  mpu.initialize();

  radio.begin();
  radio.openWritingPipe(addresses[1]);  // Alıcı adresi
  radio.openReadingPipe(1, addresses[0]);  // Verici adresi
  radio.setPALevel(RF24_PA_MIN);
  radio.stopListening();
}

void loop() {
```

```
  if (digitalRead(BUTTON1_PIN) == LOW) {
    delay(50);  // Debounce
    if (!kayitAktif) {
      Serial.println("Kayıt başladı...");
      kayitAktif = true;
      kayitBaslangicZamani = millis();
      digitalWrite(LED_PIN, HIGH);  // LED'i yak
    } else {
      Serial.println("Kayıt durduruldu.");
      kayitAktif = false;
      digitalWrite(LED_PIN, LOW);  // LED'i söndür
      kayitTamamla();
    }
    while (digitalRead(BUTTON1_PIN) == LOW);  // Buton bırakılana kadar bekle
  }

  // Eğer kayıt aktifse hareketleri kaydet
  if (kayitAktif) {
    if (millis() - kayitBaslangicZamani >= KAYIT_SURESI) {  // 1 dakikayı
geçtiyse
      Serial.println("Kayıt otomatik olarak durduruldu.");
      kayitAktif = false;
      digitalWrite(LED_PIN, LOW);  // LED'i söndür
      kayitTamamla();
    } else {
      kayitYap();
    }
  }

  // Oynatma butonuna basıldığında
  if (digitalRead(BUTTON2_PIN) == LOW) {
    Serial.println("Kayıt oynatılıyor...");
    digitalWrite(LED_PIN, HIGH);  // LED'i yak
    kayitOynat();
    digitalWrite(LED_PIN, LOW);  // LED'i söndür
    Serial.println("Oynatma tamamlandı.");
    delay(50);  // Debounce
  }

  // Normal hareket gönderme
  hareketKomut = eldivenHareketiAl();  // Eldivenden hareket komutu al
  Serial.print("Hareket: ");
  Serial.println(hareketKomut);  // Seri monitörde hareketi yazdır
  radio.write(&hareketKomut, sizeof(hareketKomut));  // Komutu alıcıya gönder
}

// **Kayıt Fonksiyonu**
void kayitYap() {
```

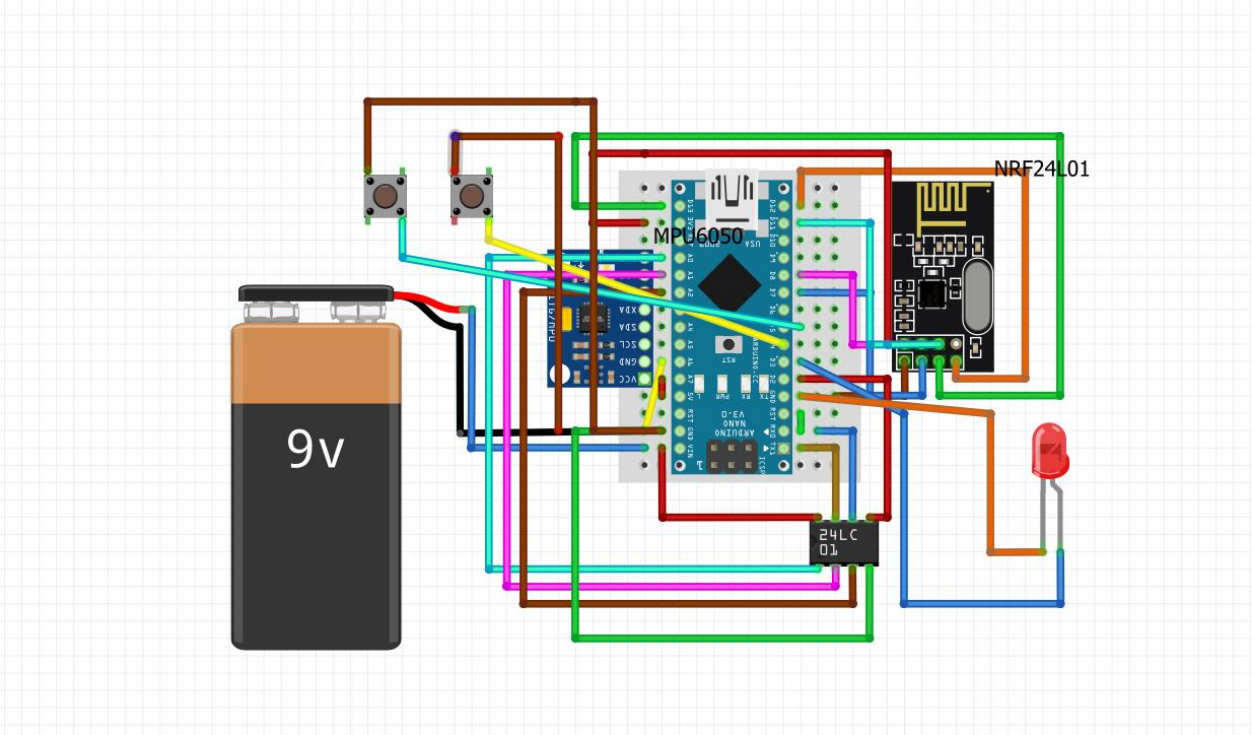
```
hareketKomut = eldivenHareketiAl();  // Eldivenden hareket komutu al
  if (hareketIndex < MAX_HAREKET) {
    Wire.beginTransmission(0x50);  // AT24C32 I2C adresi
    Wire.write((int)(hareketIndex * 2) >> 8);  // Yüksek byte adres
    Wire.write((int)(hareketIndex * 2) & 0xFF);  // Düşük byte adres
    Wire.write(hareketKomut);  // Hareket komutunu yaz
    Wire.endTransmission();
    delay(5);  // Yazma işlemi için bekleme süresi

    Serial.print("Kaydedilen: ");
    Serial.println(hareketKomut);
    hareketIndex++;
  }

  // Komutu anlık olarak araca gönder
  radio.write(&hareketKomut, sizeof(hareketKomut));

  delay(30);  // Her hareket arasında 30 ms gecikme
}

// **Kayıt Tamamlama Fonksiyonu**
void kayitTamamla() {
  // Kayıt tamamlandıktan sonra son işaretleyici ekle
  Wire.beginTransmission(0x50);
  Wire.write((int)(hareketIndex * 2) >> 8);
  Wire.write((int)(hareketIndex * 2) & 0xFF);
  Wire.write('\0');
  Wire.endTransmission();

  // Araç 2 saniye duracak
  char durKomutu = 'S';
  radio.write(&durKomutu, sizeof(durKomutu));
  delay(2000);  // 2 saniye bekle

  // LED 2 saniye yanıp söner
  manuelModGecis();
}

// **Oynatma Fonksiyonu**
void kayitOynat() {
  hareketIndex = 0;
  char komut;

  while (true) {
    Wire.beginTransmission(0x50);
    Wire.write((int)(hareketIndex * 2) >> 8);
    Wire.write((int)(hareketIndex * 2) & 0xFF);
    Wire.endTransmission();
```

```
Wire.requestFrom(0x50, 1);

    if (Wire.available()) {
      komut = Wire.read();
    }

    if (komut == '\0') {
      break;
    }

    Serial.print("Oynatılan: ");
    Serial.println(komut);

    radio.write(&komut, sizeof(komut));
    hareketIndex++;
    delay(30);  // Hareketler arasında 30 ms gecikme
  }

  // Oynatma bitince manuel moda geçiş
  manuelModGecis();
}

// **Manuel Mod Geçişi İçin LED Yanıp Sönmesi**
void manuelModGecis() {
  delay(2000);  // 2 saniye bekle
  for (int i = 0; i < 2; i++) {
    digitalWrite(LED_PIN, HIGH);
    delay(500);
    digitalWrite(LED_PIN, LOW);
    delay(500);
  }
}
// **Eldiven Hareket Algılama**
char eldivenHareketiAl() {
  int16_t ax, ay, az, gx, gy, gz;
  mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);

  const int threshold = 7000;  // Hassasiyet eşiği
  if (ax > threshold && abs(ay) < threshold) return 'D';  // Düz sağ
  if (ax < -threshold && abs(ay) < threshold) return 'Z';  // Düz sol
  if (ay > threshold) {
    if (ax > threshold) return 'R';  // İleri sağ
    if (ax < -threshold) return 'L';  // İleri sol
    return 'I';  // İleri
  }
  if (ay < -threshold) return 'G';  // Geri
  return 'S';  // Dur
}
```

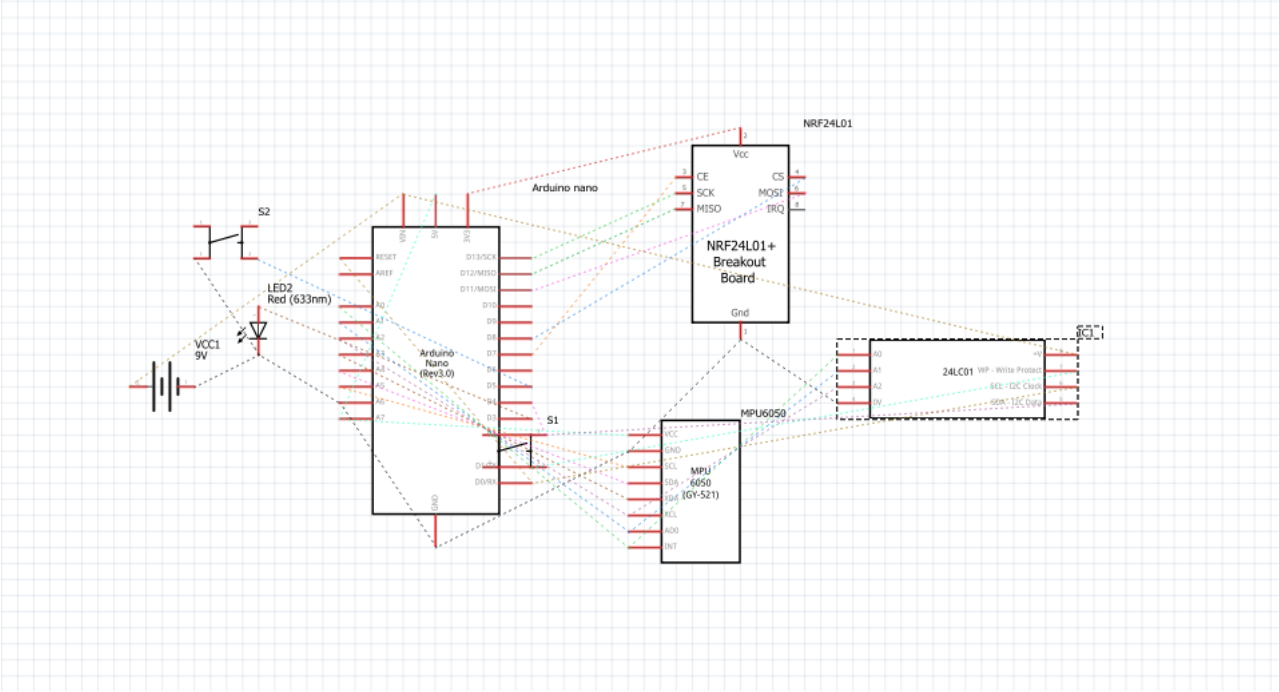
Projenin Tasarımı

Eldiven Üstü:

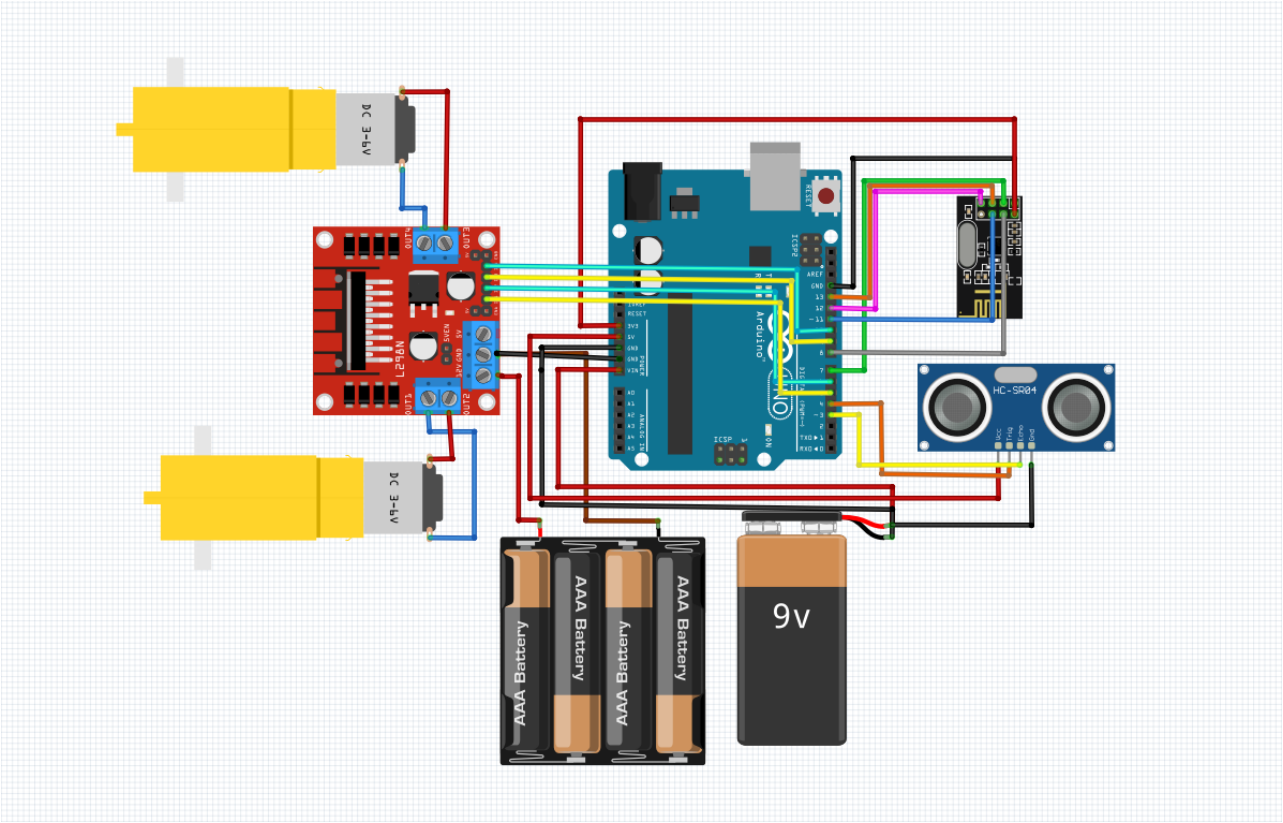
Tasarım kısmı Fritzing üzerinden yapılmıştır. GÖRSEL 1.1



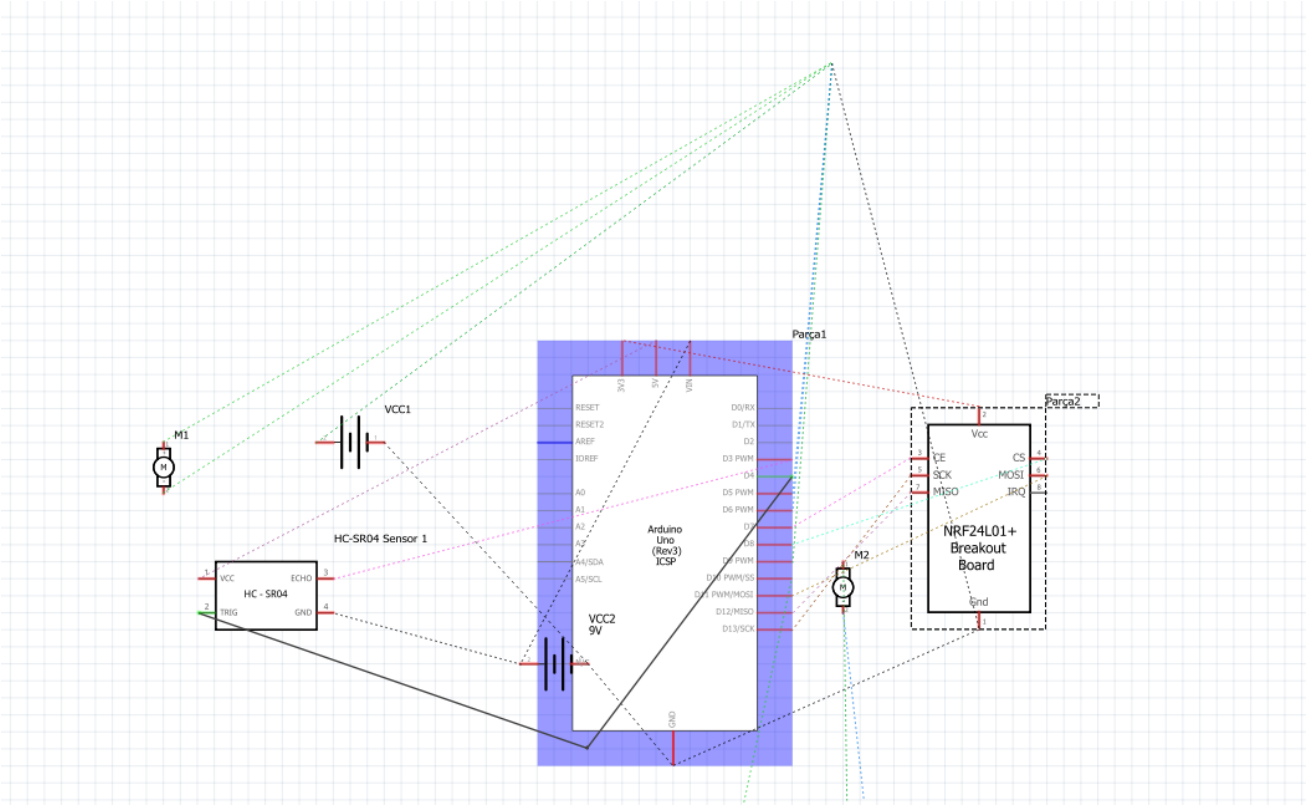
Şematiği:



Araç Üstü:



Şematiği:



13.

Sonuçlar ve Testler

Projede, manuel kontrol, hareket kaydı, kayıt oynatma, engelden kaçınma gibi özelliklerin başarıyla çalışıp çalışmadığını değerlendirmek için aşağıdaki testler yapılmıştır. Sonuçlar her bir özelliğin işlevselliğini doğrulamaya yönelik olarak incelenmiştir.

1. Manuel Kontrol Testi

- **Amaç:** Kullanıcının el hareketleriyle aracın yönlendirilip yönlendirilemediğini kontrol etmek.
- **Yöntem:** Verici modülü aktif hale getirildi. Eldiven takılarak farklı yönlerde el hareketleri yapıldı: İleri, ileri sağ, ileri sol, geri, sağ, sol, durma. Vericiden gelen verilerin alıcı tarafından doğru işlenip işlenmediği gözlemlendi.
- **Sonuç:** Araç, kullanıcı hareketlerini doğru bir şekilde takip etti. Hareket gecikmesi, kablosuz iletişim nedeniyle ihmal edilebilir düzeydeydi.

2. Hareket Kaydı Testi

- **Amaç:** Kullanıcının hareketlerini kaydedip daha sonra tekrar edebilme özelliğini test etmek.
- **Yöntem:** Kayıt butonuna basılarak belirli bir hareket dizisi kaydedildi (ileri → ileri sağ→ ileri sol→ sağ → geri → sol → dur). Kayıt tamamlandıktan sonra kayıt oynatma butonuna basıldı. Araç, hareketleri sırayla tekrar etti.
- **Sonuç:** Hareket dizisi başarıyla kaydedildi ve oynatıldı. LED, kayıt oynatma sırasında doğru şekilde yanıp söndü.

3. Engel Algılama ve Kaçınma Testi

- **Amaç:** HC-SR04 mesafe sensörünün engelleri algılayıp kaçınma manevralarını doğru bir şekilde gerçekleştirdiğini doğrulamak.
- **Yöntem:** Araç manuel modda hareket ettirildi. Yoluna farklı mesafelerde engeller yerleştirildi. Engel algılandığında aracın durup, geri çekilip, yön değiştirme tepkisi gözlemlendi.
- **Sonuç:** Mesafe sensörü, engelleri 20 cm mesafeden başarıyla algıladı. Araç, geri hareket ederek alternatif bir yön seçti ve engelden kaçındı.

4. Kablosuz İletişim Testi

- **Amaç:** NRF24L01 modülleri arasındaki iletişimin kesintisiz ve stabil olup olmadığını kontrol etmek.
- **Yöntem:** Verici ve alıcı modüller arasındaki mesafe artırılarak sinyal gücü test edildi. Farklı ortamlarda (kapalı alan, açık alan) testler yapıldı.
- **Sonuç:** Kapalı alanda 15 metreye kadar, açık alanda 25 metreye kadar iletişim stabil kaldı. Sinyal gücü, metal engeller nedeniyle zayıfladı ancak performansı etkilemedi.

5. Güç Kaynağı Stabilitesi

- **Amaç:** Güç kaynağının sistemi kesintisiz çalıştırıp çalıştırmadığını test etmek.
- **Yöntem:** Araç üzerindeki bataryalar tamamen şarj edildi. Uzun süreli kullanımda (30 dakika) sistem performansı gözlemlendi.
- **Sonuç:** Güç kaynağı, motorlar ve sensörler için yeterli stabilite sağladı. Daha büyük kapasitörler eklenerek sistemdeki voltaj dalgalanmaları minimize edildi.

Genel Sonuçlar

1. **Manuel kontrol:** Sorunsuz çalıştı.
2. **Hareket kaydı ve oynatma:** Hatasız bir şekilde çalıştı.
3. **Engelden kaçınma:** Mesafe sensörü başarılı bir şekilde işlev gördü.
4. **Kablosuz iletişim:** Beklenen mesafelerde kararlı performans sergiledi.
5. **Güç stabilitesi:** Uzun süreli kullanım için yeterliydi.

Öneriler

1. Araç Hızı ve Hassasiyeti

- **Öneri:** Araç hareketlerinin hızını kontrol edebilmek için PWM sinyalleriyle motor hız kontrolü ekleyebilirsiniz.
- **Fayda:** Araç farklı zeminlerde ve durumlarda daha hassas hareket edebilir.

2. Çoklu Engel Algılama

- **Öneri:** Daha kapsamlı engel algılama için birden fazla HC-SR04 sensörü ekleyerek aracın çevresel farkındalığını artırabilirsiniz.
- **Fayda:** Araç yalnızca öndeki değil, yanlardaki ve arkadaki engelleri de algılayabilir.

3. Bluetooth veya Wi-Fi Entegrasyonu

- **Öneri:** Kablosuz kontrol sistemine NRF24L01 yerine Bluetooth (HC-05) veya Wi-Fi (ESP8266) modülü ekleyerek, aracı bir mobil uygulama veya web arayüzü üzerinden kontrol edebilirsiniz.
- **Fayda:** Kullanıcı deneyimini artırır ve daha geniş kontrol imkânı sunar.

4. Enerji Toplama (Solar Panel)

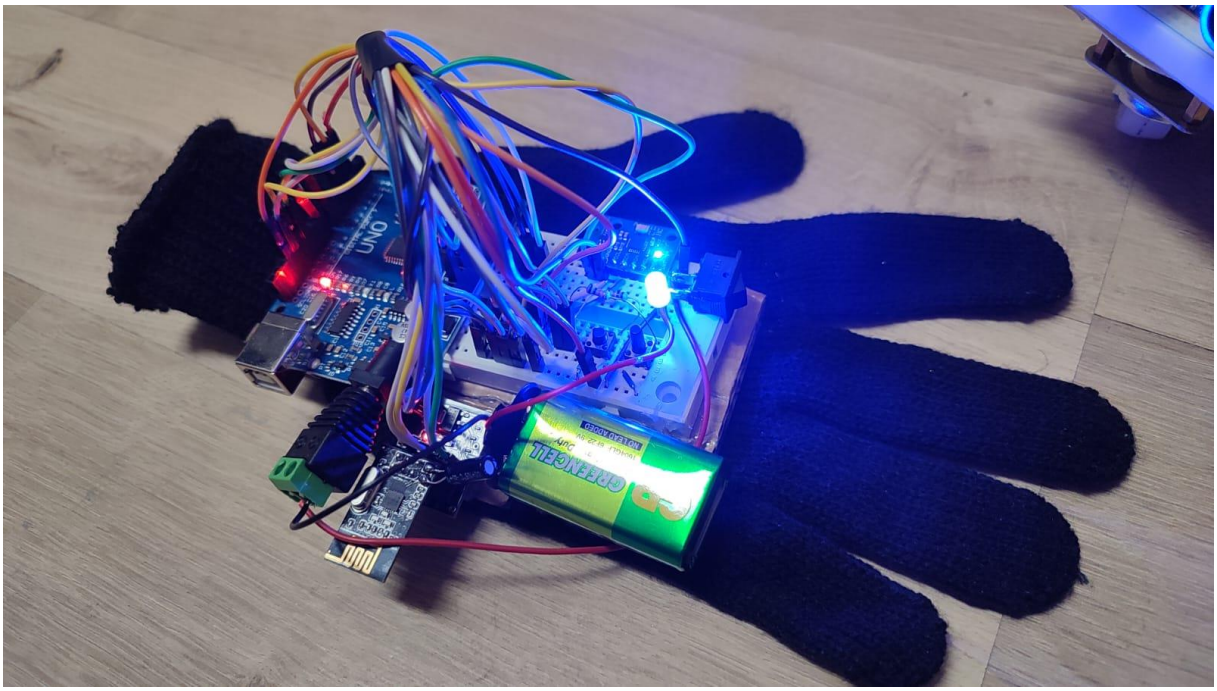
- **Öneri:** Araç üzerine küçük güneş panelleri ekleyerek pilin şarj süresini uzatabilirsiniz.
- **Fayda:** Çevre dostu bir proje olur ve enerji verimliliği artar.

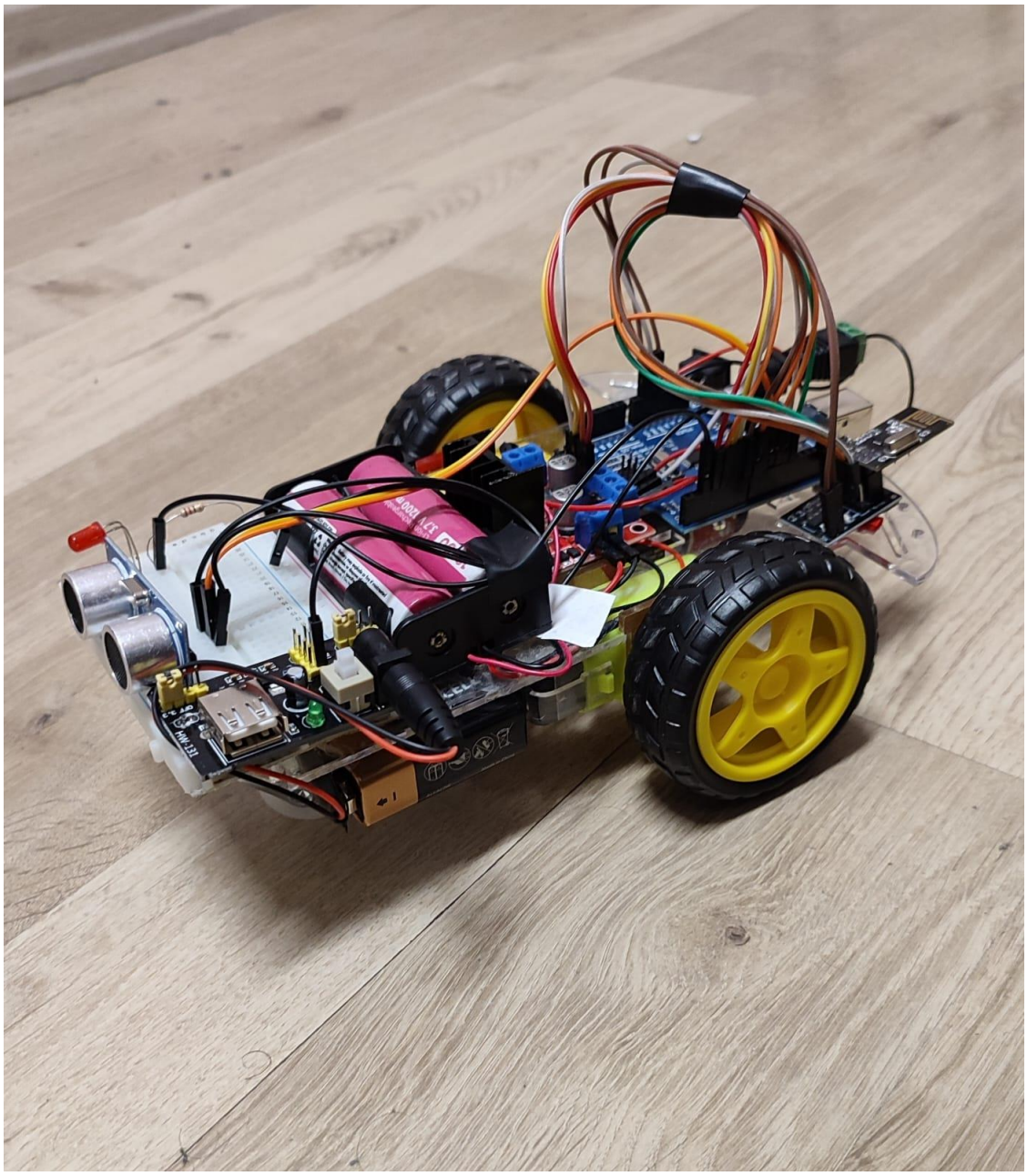
Projenin Genel Değerlendirilmesi

Projem genel anlamda oldukça güçlü ve iyi düşünülmüş bir yapıya sahip. MPU6050 sensörü kullanılarak el hareketlerine dayalı kontrol sistemi oluşturulması, kullanıcı dostu bir deneyim sunuyor. Bu, sistemin yenilikçi ve pratik bir yaklaşım sergilemesini sağlıyor. NRF24L01 ile kablosuz haberleşme, araç kontrolünü daha esnek hale getiriyor ve kullanıcının hareket alanını genişletiyor. Aracınızın ileri, geri, sağa, sola, düz sağ ve düz sol gibi çeşitli hareket kabiliyetlerine sahip olması, çok yönlü kullanım avantajı sunuyor. Ayrıca, projeye eklediğiniz LED ve HC-SR04 sensörü, güvenlik ve görsel geri bildirim açısından işlevselliği artırıyor.

Projenin modüler yapısı da dikkat çekici bir özellik. Arduino Nano ve Uno gibi kartların kullanılması, sisteminizi genişletmeyi ve güncellemeyi kolaylaştırıyor. Bu, ileride projeye eklemeler yapmayı veya sistemi geliştirmeyi daha da basitleştirebilir. Bu unsurlar, projenizi hem işlevsellik hem de inovasyon açısından güçlü bir konuma yerleştiriyor.

Projenin Son Hali





KAYNAKÇA

https://www.arduino.cc

https://github.com/nRF24/RF24

https://github.com/ElectronicCats/mpu6050

https://www.electronics-tutorials.ws/