Gerçek Zamanlı Sistemler

Uzaklık Algılayıcı Eldiven

Hüseyin Can Ercan

12011009

Projenin Amacı

Proje görme yetisi olmayan kişiler için çevreyi algılamayı destekleyici, giyilebilir sistem tasarlanmasını amaçlamaktadır. Görme yetisinin kaybı hareket kabiliyetinin kısıtlanmasına neden olmaktadır. Çünkü kişi çevresi hakkında bilgi sahibi olamaz, bu durumun aşılması için alternatif kaynaklardan bilgi sağlanabilir. Elinizle çevrede gezdirebileceğiniz ve fiziksel geri besleme alabileceğiniz bir baston veya eğitim almış hayvanlar bu alanda kullanılan çözümlerdir.

Proje esnasında maliyet ve erişilebilir teknoloji dikkate alınarak geliştirilen çözüm uzaklığa göre titreşen mikro denetleyici tabanlı eldiven olmuştur. Eldiven en basit ifadeyle el ayasındaki uzaklık sensöründen alınan veriyi istenen parmağa parmağa yerleştirilmiş titreşim motorunun gücünü ayarlamak için kullanmaktadır. Mesafenin titreşime çevirimi söz konusudur. Böylece kullanıcı el ayasını istediği yöne doğrultarak titreşimden uzaklığı anlayabilmektedir. Sistemin kontrolü ise elin yer çekimine göre açısal konumu ile sağlanmaktadır, kullanıcı elini aşağı indirdiğinde sistem titreşim üretmeyi durdurmaktadır.

Basit bir baston üzerinde gezdirilen yüzey konusunda fikir verebilir, sabit menzil içindeki engelleri tespit etmek için kullanılabilir. Tüm bu verinin sağlanması için sadece kullanıma uygun bir el yeterlidir, güç tüketmez veya çalışır durumda tutulabilmesi için fazladan çaba gerektirmez. Fakat kullanıcı bilgi elde edebilmek için fiziksel olarak aktif rol oynamalıdır ve menzil sadece bastonun uzunluğu kadardır.

Eğer sopadan daha cazip bir çözüm üretilmek isteniyorsa öncelikle mesafe problemi çözülmelidir. Kullanıcıya sopadan daha uzun yarıçapta çalışacak sistem sunulmalıdır. Bu sistem konforlu olmalıdır, ağırlık veya boyut gibi problemleri bulunmamalıdır. Ayrıca sistem kullanıcıya anlaşılabilir şekilde geri bildirimde bulunmalıdır. Kullanıcının sağlıklı çalışan duyuları bu verinin aktarımı için uygundur fakat çözünürlük ve anlık aktarılabilecek veri limiti gibi ciddi problemler ortaya çıkaracaktır. Aktarım için titreşim, ses, sıcaklık gibi aday kanallar mevcuttur.

Projenin Gerçekleştirilmesi

Uzaklığı titreşime dönüştürecek sistemin gereksinimleri basit olarak uzaklık sensörü, titreşim motoru ve bu birimleri kontrol edecek mikro denetleyici olarak sıralanır. Titreşim motorunun insan tarafından ayırt edilebilir titreşimler üretebilmesi için gerekli güç mikro denetleyici üzerinden elde edilemeyebilir, bu problem yükseltici devre kullanımı ile çözülür.

Kullanıcının kontrol ara yüzünü sağlayacak olan açısal konum ise ivmeölçer sensörlerle rahatlıkla ölçülebilir. Yer çekimi ivmesinin yönüne göre elin hangi konumda durduğu ayırt edilecektir. Sistem elin farklı konumlarını farklı komutlar olarak değerlendirir.

Planlama aşamasında testlerin rahatlıkla yapılabilmesi için donanıma lcd panel de eklenmiştir. Ölçülen mesafe, motor titreşimi gibi değerler bu ara yüzden takip edilebilir.

Donanım Seçimi

Sistemin tümü donanım olarak mikro denetleyici, titreşim motoru, titreşim motorunun yükseltici devresi, uzaklık sensörü, ivmeölçer sensör, lcd panel ve hepsine güç verecek bir güç kartından oluşmaktadır.

Mikro denetleyici olarak Arduino Uno seçilmiştir. Maliyet, erişilebilirlik, uyumlu çevre birimleri açısından oldukça ideal.

Titreşim motoru olarak şaftına dengesiz ağırlık monte edilmiş 6V maksimum gerilimle çalışan doğru akım motoru seçilmiştir. Motor modelden bağımsız olarak gerekli voltaj aralığına göre belirlenmiştir. Sistemin genelinde sağlanabilecek maksimum voltaj seçilen güç kartı üzerinden sağlanan 5 volt ’tur.

Titreşim motorunun yükseltici devresi ise mikro denetleyici üzerinden sağlanacak PWM sinyali ile motorun kontrolünü sağlamaktadır. Kullanıcıya uzaklık ile ilgili bilgi vermek istiyorsak mesafeyi kabul edilebilir çözünürlükle kullanıcıya bildirmemiz gerekli. Engel var, engel yok demek yeterli değil. PWM sinyali ile doğru orantılı değiştirilecek motor titreşimi bu ihtiyacı bir noktaya kadar karşılayacaktır.

Uzaklık sensörü olarak HC-SR04 seçilmiştir. Sensörün maliyeti ve basit kullanım ara yüzü avantajdır. 4 metre maksimum menzili, ultrasonik ses dalgalarıyla ölçüm yapıyor olması, 4 metreyi aşan mesafelerden gelen yankıların mesafe bilgisinde gürültü yaratıyor olması dezavantajdır. Kullanıcı Trig pinine 10 mikro saniyelik kare dalga verdikten sonra Echo pininden yankıyı okuyarak arada geçen süreden mesafeyi hesaplamaktadır.

İvme ölçümü için maliyeti ve erişim kolaylığı dikkate alınarak MPU-6050 seçilmiştir. I2C ile iletişim kurulan sensörün hâlihazırda Arduino topluluğunca kullanılan kütüphanelerinin olması bir diğer avantajdır.

Lcd panel için 2 sıralı 16 karakterli ekran seçildi, ekranın doğrudan kontrol edilmemesi için I2C dönüştürücü kullanıldı. Projede temel birimlerden biri olmadığı için lcd panelin proje içerisindeki önceliği düşüktür, bu nedenle Arduino’nun lcd paneli I2C ara yüzünden kontrol

etmesi ve ileride kolayca donamımdan çıkarılabilmesi gözetilmiştir. Ayrıca lcd panelin ara eleman olmadan kontrolü pin kullanımı açısından diğer elemanlara göre çok daha fazladır.

Güç kartı olarak MB-V2 kullanılmıştır. 3.3 ve 5 volt çıkış veriyor olması, 6.5-12 volt aralığında gerilimle çalışabilmesi, 700 ma maksimum akım kapasitesi parçanın avantajlarıdır. Sistem içerisindeki güç kullanımını daha güvenli hale getirmektedir, Arduino'nun motor sürerken karşılaştığı kısıtlamaları ve riskleri ortadan kaldırmaktadır. Ayrıca giriş voltajının aralığı güç kaynağı olarak kullanıcıya seçme şansı vermektedir. Dezavantaj olarak getirdiği fazladan boyut sayılabilir.

Parçaların hepsi Arduino üzerine oturan tek yüzü lehimlenebilir petrinaks ile birleştirilmiştir. Parçalar JST uçlu kablolar yardımı ile petrinaksa bağlanarak parçaların eldiven üzerinde istenilen şekilde konumlandırılması mümkün kılınmıştır.

Yazılım Tasarımı

Sistemin sorumlulukları 3 başlıkta toplanmıştır. Biri ivme sensörünün okunmasıdır, yani kullanıcı girdisinin alınmasıdır. İkinci görev uzaklık sensörünün okunarak motor titreşimine yansıtılmasıdır. Uzaklık sensörünün okunması ve motor titreşiminin değiştirilmesi ayrık işler olarak değerlendirilmemiştir. Son görev olarak da lcd panelin güncellenmesi belirlenmiştir. Parçaların donanımsal hızların farklı olması, uzaklık ölçümü için geçen sürenin uzaklığa göre değişiklik gösteriyor olması, lcd panelin çok sık güncellendiğinde çıktı gösterememesi sistem içerisinde zamana göre bu işlemlerin düzenlenmesini zorunlu kılmaktadır.

Arduino Uno bir mikro denetleyicidir. Kaynak yönetimi sağlayacak bir işletim sistemi yoktur. İşlerin planlanması için kullanıcının kendisi işleri zamansal olarak planlayacak mekanizmayı gerçekleştirmelidir.

Sistemdeki işlerden en önemli olan kullanıcı girdisini alan ivme ölçümüdür, bu işlem önceden tanımlanmış tekar zamanı tamamlandıkça koşulsuz çalıştırılmaktadır. Önem sırasında ikinci motor ve uzaklık sensörünün görevidir. Bu işlem bloğunun çalıştırılması için koşul belirlenmiştir, kullanıcı komutu ile aktifleşmektedir. Son iş lcd güncellemesi diğerlerinden daha büyük bir periyod süresi ile çalışmaktadır, koşulu yoktur.

İşlerin zaman maliyeti tablosu:

*1000 tekrar	ms
Kullanıcı girdisinin alınması	1951
Uzaklık ve motor	55606
Lcd	19834

Uzaklık sensörü 200 cm mesafe ile test edilmiştir. Yankının geri dönüş süresi göz ardı edilemeyecek orana sahiptir.

Yapılan test ile işlerin tek başlarına yarattıkları zaman maliyeti tespit edilmiştir. Girdi 2 ms, uzaklık ölçümü ve motor kontrolü 55 ms, lcd 20 ms zaman almaktadır.

Uzaklık sensörü en maliyetli olandır çünkü 55 ms sürenin çoğunu bekleyerek geçirmektedir, 50 ms bekleme süresine sahiptir. HC-SR04 için datasheetin önerdiği doğrultuda 50 ms'yi aşan ölçüm sıklığı bariyeri oluşturulmuştur. Sistem istese de daha sık ölçüm yapamamaktadır. Ses ile gerçekleştirilen ölçümlerin sağlıklı olması açısından bu bir gerekliliktir.

Lcd öncelik olarak düşük olması sebebiyle yarattığı yük uzun vadede etkisizleştirilebilir. Diğer işlerin periyodlarına göre daha uzun periyot ayarlanması ile zaman maliyeti düşürülebilir.

Sistem zamanı ek küçük parça olarak 10 ms belirlenmiştir. İşlerin zamanlamasının takibi 10 ms aralıklarla yapılmaktadır. Herhangi bir iş için uygun koşul oluşmamış olsa bile algoritma 10 ms aralıklarla koşulları kontrol etmeye devam edecektir. Bu şekilde mikro boyutlu zaman kaymalarının yaratacağı bozulma en fazla 10 ms gecikme olarak yansıyacaktır. Zamanından erken çalıştırılmaya çalışılan uzaklık ölçüm bloğu en fazla bir sonraki iterasyona kadar gecikebilir.

Eğer durumu iyimser olarak ifade edersek yukarıdaki grafik ortaya çıkacaktır. Kırmızı, çizgi şeklini almış alan kullanıcı girdisinin maliyetini ifade ediyor, mavi alan uzaklık ölçümünü ve motor kontrolünü, yeşil alan lcd güncellemesini gösteriyor.

Kullanıcı girişi 100 ms, uzaklık okuma ve motor kontrolü 100 ms, lcd 300 ms aralıklarla çalıştırılmaktadır.

Sistemin deadline kısıtı kullanıcıya 1 sn içinde güncel geri besleme sağlayacak şekilde kabul edilmiştir. Bilimsel olarak görme engelli bir insanın ne ölçüde veri gereksinimi olduğu, saniyede kaç adım attığı ve kaç ayrı noktadan ölçüm yapmak istediği bilinmediği için sınır 1 sn olarak kabul edilmiştir. Seçilen mevcut durumda periyot uzunlukları kullanıcıya saniyede 10 ölçüm ile cevap vermektedir.

Sistem offline scheduling ile soft deadlineları olan işlere sahip olarak çalışmaktadır. Uzun vadede en az 1 sn de bir kullanıcıya doğru veri aktarımı hedeflenmektedir.

Filtreleme

Sistem 2 sensör bulundurmakta, biri ivme diğeri uzaklık sensörü. İki sensörün de sağlıklı değerler için filtrelenmesi gerekmektedir. İvme ölçümü kullanıcı kontrolünü sağladığı için çok aktif değişen, stabil olmayan değerler kontrolü zorlaştıracaktır. En ufak harekette bile sistemin motoru çalıştırıp kapaması veya kullanıcı bunu ifade etmese bile motoru durdurması

istenmeyen durumlardır. Bu durumları engellemek için örnekler 15'li ortalama filtresinden geçirilmektedir. Filtrenin uzunluğu değişkendir.

Çok uzun filtrenin kullanıcı komutlarının gecikmesine sebep olacağı unutulmamalıdır.

Uzaklık sensörünün filtrelenmesi için boyutu 5 olan ortalama filtresi kullanılmıştır. Uzaklık ölçümleri menzil içindeki ölçümlerde gürültü oranı düşük sonuçlar vermektedir. Yankının menzil dışındaki zeminden dönmesi veya açılı yüzeylerden hiç yankı dönmemesi gürültüye sebep olmaktadır. Fakat asıl sorum menzil dışındaki ölçümlerden gelmektedir. Bu durumda yankının dönüyor olması dönmemesinden daha büyük bir problem ortaya çıkarmaktadır. HCSR-04 menzil dışından yapılan ölçümleri bazı seferlerde 4-5 cm gibi oldukça kısa mesafeler olarak okumaktadır.

Bu problemin çözümü için sensörün menzil dışı ölçümleri için daha iyi test ortamlarında sensörün davranışı incelenerek ölçüm karakteristiği Arduino üzerinde bu tip hataları filtreleyecek şekilde düzenlenebilir.

Motor Kontrolü

Uzaklık ve motor titreşimi arasındaki ilişki doğrusal seçilirse kullanıcının uyarılması konusunda çözünürlük problemi ortaya çıkılmaktadır.

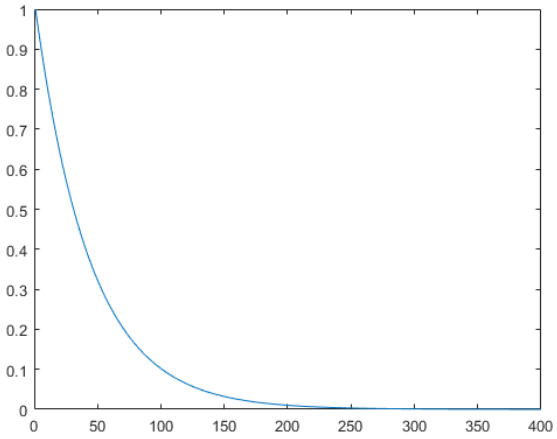
Bilimsel olmamakla beraber yapılan gözlemler motor titreşiminin ayırt edilebilmesi için PWM sinyalinin yüksek oranlarda değiştirilmesi gerektiğini göstermiştir. Örneğin 400 cm uzaklık %0, 0 cm %100 olacak şekilde PWM sinyali için değer aralığı belirlenirse aradaki fonksiyonun tipi çevrenin algılanmasını büyük oranda değiştirecektir.

10 cm ile 50 cm %10 titreşim farkı oluşturmaktadır. Fakat motor titreşiminden çıkarım yapan insan bu farkı ayırt edilemeyebilir. Bu durumda aktarılacak mesafe bilgisi motorun titreşim çözünürlüğü ve insan elinin algılama çözünürlüğü arasında sıkışıp kalacaktır. İnsan elinin farklı bölgelerine farklı sayıda motor yerleştirilmesi veya daha geniş titreşim kuvveti aralığına sahip motor kullanılması aktarılacak bilgi miktarını arttırabilir.

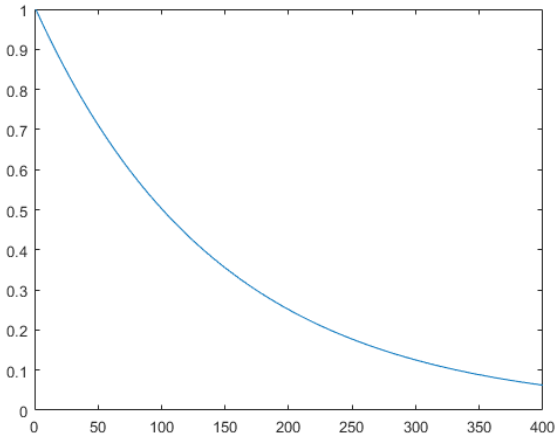
Eğer elimizdeki donanımı iyi kullanmak istiyorsak uzaklığı değerlendirirken farklı yaklaşımlar kullanabiliriz. Sistemin belirli bir menzildeki çözünürlüğünü arttırmak bu yaklaşımlardan biridir. Eğer kullanıcını için yol bulma odaklı düşünüyorsak 50 cm uzaklık kullanıcıyı en çok ilgilendiren yarıçap olacaktır. Bu yaklaşık olarak bir adım mesafesidir. Basitçe kullanıcı adım atabileceği ve atamayacağı alanları ayırt edebilecektir. Kullanıcının isteğine göre belirli uzaklık bantlarında sistemin çözünürlüğü arttırılabilir.

Eğer yürümeye odaklı tasarım yaparsak yakın mesafedeki değişimlerden daha anlaşılabilir çözünürlükte haberdar olmak için titreşimi üssel azaltabiliriz. Azalmasını sağlamak için 0.5 taban kabul edilip 400 cm 0-4 değer aralığına indirilip üs olarak kullanılabilir. Böyle bir denklem 1 cm için %100'e yakın PWM üretirken, 50 cm için %70, 100 cm için %50 PWM üretecektir. Menzilin devam değerlerinde fark edilemeyecek farklar oluşacaktır. 200 cm için %25, 300 cm için % 12.5 PWM üretilecektir.

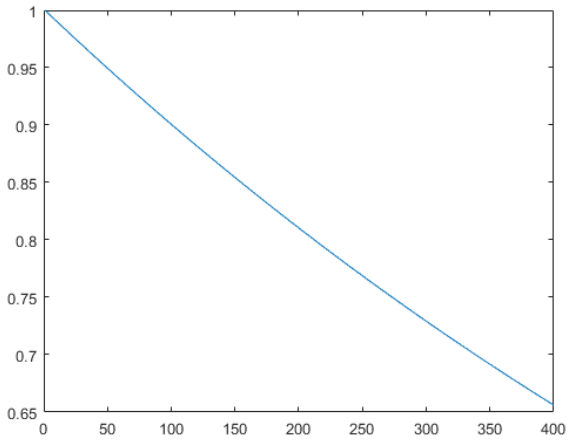
Kullanıcının isteğine göre taban değiştirilerek grafik örneklerde olduğu gibi değiştirilebilir. Taban sayısı arttıkça daha doğrusal azalma gözlenecektir.



$(0.1)^x$



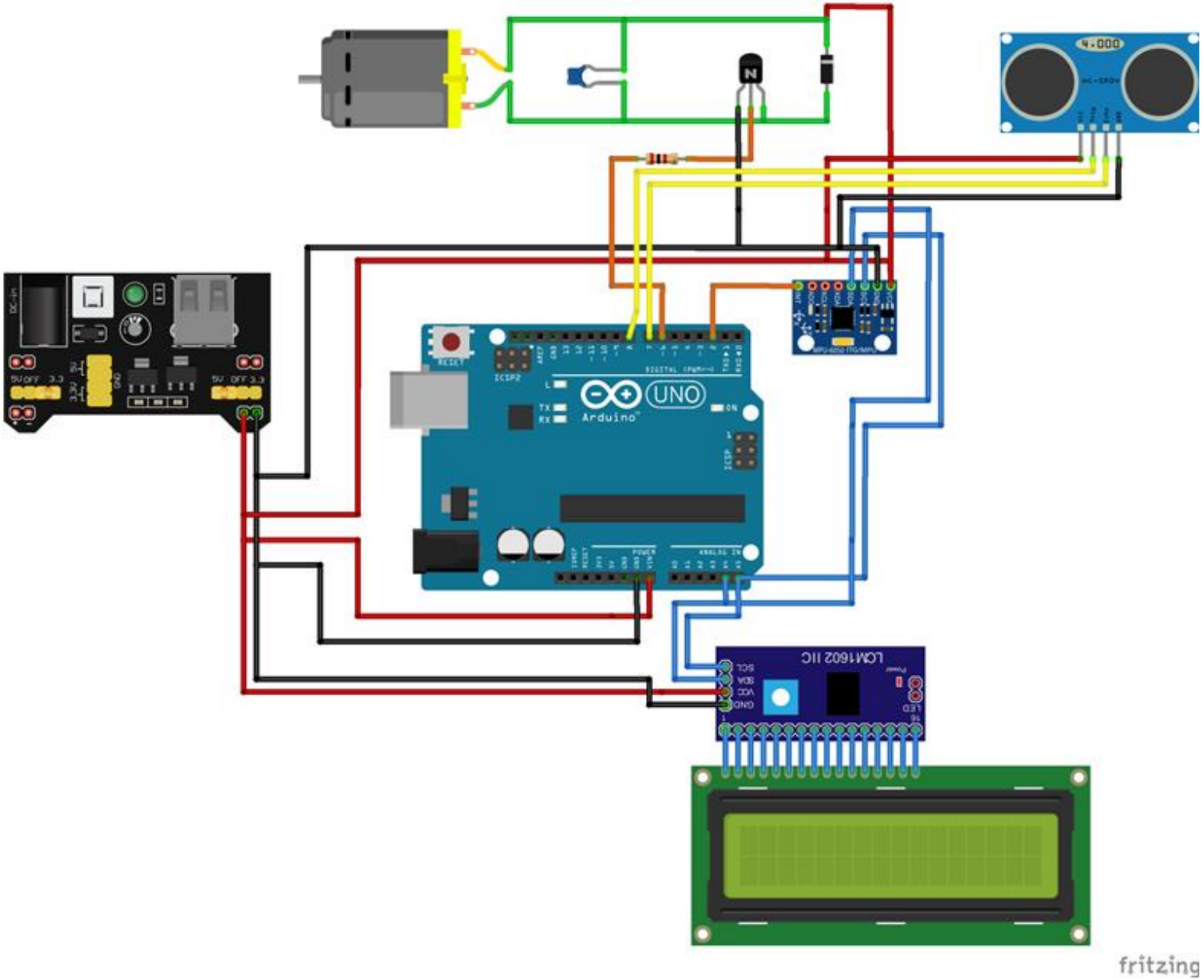
$(0.5)^x$

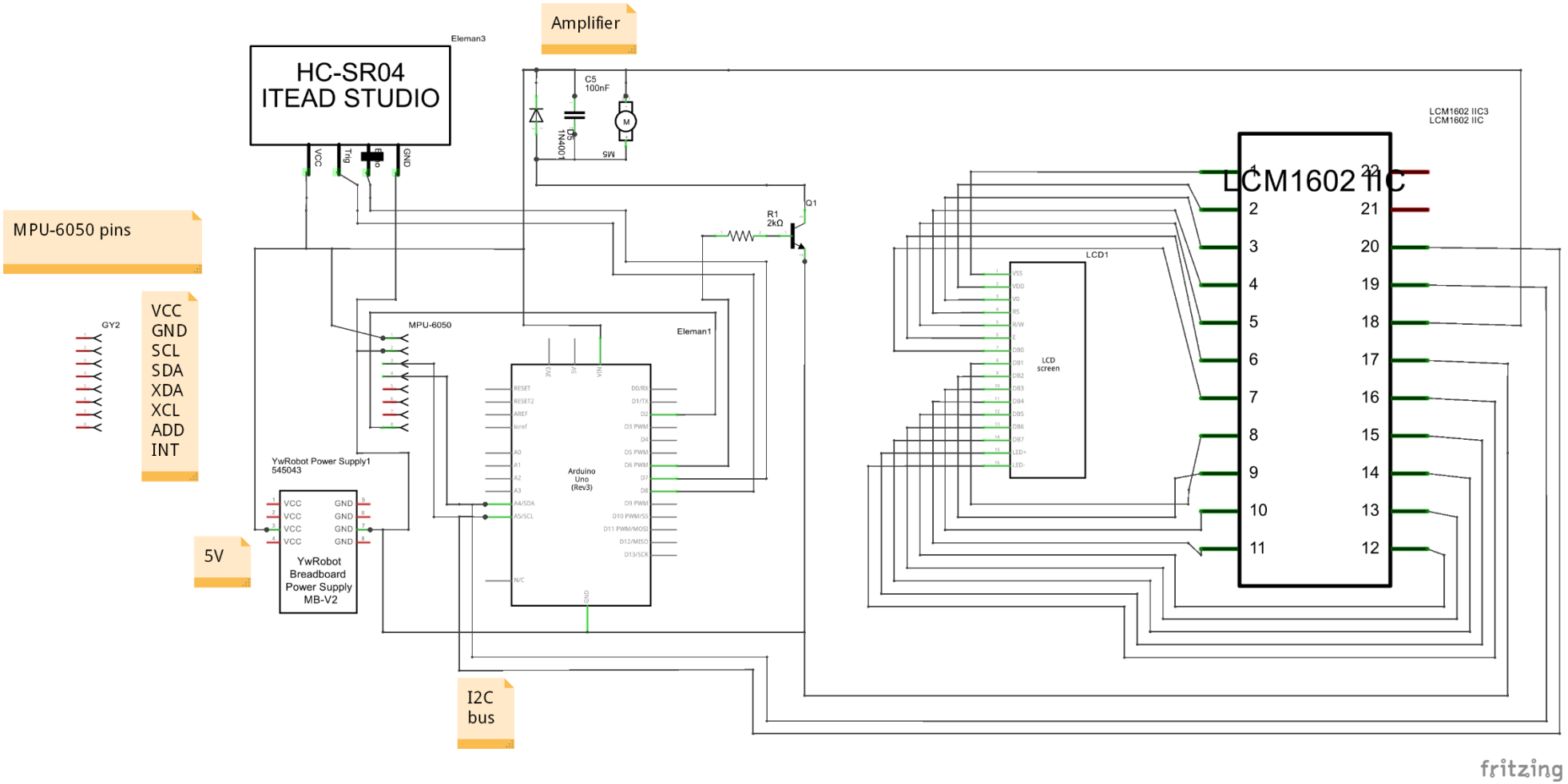


$(0.9)^x$

Yükseltici Devre

Arduino Uno'nun doğrudan yüksek akım tüketen cihazları kontrol etme şansı yoktur. Bu nedenle yükseltici devre kullanılmıştır. Yükseltici devre ayrıca Arduino'yu koruyucu tedbirler de içermektedir. Motorun indüktör gibi davranması durumunda akımın potansiyeli arttırma etkisini yok etmek adına ters bağlı bir diode, motor fırçalarındaki arklara karşı kapasitör, Arduino'nun pin çıkışından geçebilecek maksimum akımı sınırlandırabilmek adına direnç kullanılmıştır.


4.000
UNO
Arduino
LCM1602 IIC
fritzing


Amplifier
Eleman3
HC-SR04
ITEAD STUDIO
VCC
Trig
Echo
GND
C5
100nF
D5
1N4001
M
SW
R1
2kΩ
Q1
LCM1602 IIC3
LCM1602 IIC
LCM1602 IIC
LCD1
LCD
screen
MPU-6050 pins
GY2
VCC
GND
SCL
SDA
XDA
XCL
ADD
INT
MPU-6050
Eleman1
Arduino
Uno
(Rev3)
YwRobot Power Supply1
545043
YwRobot
Breadboard
Power Supply
MB-V2
5V
I2C
bus
fritzing

Kaynakça

http://learningaboutelectronics.com/Articles/Vibration-motor-circuit.php

https://arduino-info.wikispaces.com/LCD-Blue-I2C

http://www.instructables.com/id/Simple-Arduino-and-HC-SR04-Example/

http://playground.arduino.cc/Main/MPU-6050

Kod

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include "I2Cdev.h"
#include "MPU6050.h"
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE//Arduino IDE version
problem covarage, wire library bug
#include "Wire.h"
#endif

#define ACCELOREMETERFILTERSIZE 15
#define DISTANCEFILTERSIZE 5
#define TRIGGERPIN 8//HCSR-04 trigger pin
#define ECHOPIN 7//HCSR-04 echo pin
#define MOTORCONTROL 6//motor PWM output

#define FUNCTIONBASE 0.4

#define TIMEQUANTUM 10//ms
#define GYROTASKPERIOD 100//ms
#define MOTORTASKPERIOD 100//ms
#define LCDTASKPERIOD 300//ms

int16_t ax, ay, az;
int16_t gx, gy, gz;

//Signal filtering scturcture
struct Signal {
  int16_t *signalSeq;
  uint16_t seqSize;
  uint16_t seqCursor;
  int16_t sum;
  int16_t avarage;
};
//Filters' pre defined signal arrays
int16_t acceloremeterSeqX[ACCELOREMETERFILTERSIZE];
int16_t acceloremeterSeqY[ACCELOREMETERFILTERSIZE];
int16_t acceloremeterSeqZ[ACCELOREMETERFILTERSIZE];
int16_t distanceSeq[DISTANCEFILTERSIZE];

int16_t  insertMeasurement(Signal signalA, int16_t value);
void initializeSignal(Signal *signalA);
int16_t getDistance(uint8_t trPin, uint8_t ecPin);

LiquidCrystal_I2C lcd(0x27, 20, 4); // set the LCD address to 0x27 for a 16
chars and 2 line display
MPU6050 accelgyro;

Signal signalX;//three axis signals
Signal signalY;
Signal signalZ;

Signal distanceSignal;

void setup()
{
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
  Wire.begin();
#elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE//wire bug fix,
without fix Arduino can't find LCD
```

```
  Fastwire::setup(400, true);
#endif

  Serial.println("Initializing..");
  accelgyro.initialize();
  //signals initialization
  signalX.signalSeq = acceloremeterSeqX;
  signalY.signalSeq = acceloremeterSeqY;
  signalZ.signalSeq = acceloremeterSeqZ;
  distanceSignal.signalSeq = distanceSeq;
  signalX.seqSize = ACCELOREMETERFILTERSIZE;
  signalY.seqSize = ACCELOREMETERFILTERSIZE;
  signalZ.seqSize = ACCELOREMETERFILTERSIZE;
  distanceSignal.seqSize = DISTANCEFILTERSIZE;
  initializeSignal(&signalX);
  initializeSignal(&signalY);
  initializeSignal(&signalZ);
  initializeSignal(&distanceSignal);

  lcd.init();
  lcd.backlight();
  Serial.begin(9600);
}

void loop()
{
  int16_t dispX;//filter avarages of the acceloremeter axis measurements
  int16_t dispY;
  int16_t dispZ;

  pinMode(TRIGGERPIN, OUTPUT);
  pinMode(ECHOPIN, INPUT);
  pinMode(MOTORCONTROL, OUTPUT);
  int16_t distance = 0;
  int16_t avrDistance = 0;
  double duty = 0;//PWM duty value
  unsigned long systemTime = 0;//common time
  unsigned long gyroTaskLastTime = 0;//basic scheduling values, last
execution times
  unsigned long lcdTaskLastTime = 0;
  unsigned long motorTaskLastTime = 0;

  while (1)//endless loop, user can power off the system or reset with the
button
  {
    systemTime =  millis();
    //GYRO TASK BLOCK
    if ( (systemTime - gyroTaskLastTime) > GYROTASKPERIOD )
    {
      gyroTaskLastTime =  millis();
      accelgyro.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
      dispX = insertMeasurement(&signalX, ax);
      dispY = insertMeasurement(&signalY, ay);
      dispZ = insertMeasurement(&signalZ, az);
      //gyro measured and filtered
    }
    systemTime = millis();
    //MOTOR TASK BLOCK
    if ( (systemTime - motorTaskLastTime) > MOTORTASKPERIOD )
    {
      motorTaskLastTime = millis();
```

```
      if (-10000 > dispY)// palm looking forward
      {
        distance =  getDistance(TRIGGERPIN, ECHOPIN);
        if (distance > 400)
        {
          distance = 400;
        }
        avrDistance = insertMeasurement(&distanceSignal, distance);

        duty = pow(FUNCTIONBASE, float(float(avrDistance) / 100)); //duty
calculation via selected base value
        uint8_t dDuty = uint8_t(duty * 255);//analog output value of
vibration
        analogWrite(MOTORCONTROL, dDuty);
      }
      else
      {
        analogWrite(MOTORCONTROL, 0);// palm not looking forward
      }
    }
    systemTime = millis();
    //LCD TASK BLOCK
    if ( (systemTime - lcdTaskLastTime) > LCDTASKPERIOD )
    {
      lcdTaskLastTime = millis();
      if (-10000 > dispY)// palm looking forward
      {
        //print motor vibraiton and distance
        lcd.clear();
        lcd.print(avrDistance); lcd.print(" cm");
        lcd.setCursor(0, 1);
        lcd.print("%"); lcd.print(duty * 100); lcd.print(" ");
      }
      else
      {
        lcd.clear();
        lcd.print("NOT ACTIVE ");
      }
    }
    delay(TIMEQUANTUM);
  }
}

void initializeSignal(Signal *signalA)
{
  //all filter values are zero
  signalA->avarage = 0;
  signalA->seqCursor = 0;
  signalA->sum = 0;
  for (int i = 0; i < signalA->seqSize; i++)
  {
    signalA->signalSeq[i] = 0;
  }
}
int16_t  insertMeasurement(Signal *signalA, int16_t value)
{
  //calculate the avarage with only changing values, do not repeat the
whole calculation
  int removeSlot = (signalA->seqCursor + 1) % signalA->seqSize;
  signalA->avarage -= signalA->signalSeq[removeSlot] / signalA->seqSize;
  //cyclic memory usage
```

```
  signalA->signalSeq[removeSlot] = value;
  signalA->avarage += signalA->signalSeq[removeSlot] / signalA->seqSize;
  signalA->seqCursor = removeSlot;
  return signalA->avarage;
}
int16_t getDistance(uint8_t trPin, uint8_t ecPin)
{
  unsigned long realDuration = 0;
  uint16_t duration;
  uint16_t distance;
  digitalWrite(trPin, LOW);
  delayMicroseconds(4);
  digitalWrite(trPin, HIGH);
  delayMicroseconds(10);//trigger the pin
  digitalWrite(trPin, LOW);
  realDuration = pulseIn(ecPin, HIGH);//wait for response
  duration = realDuration;
  distance = (duration / 2) / 29, 41;//calculate the distance in cm form
  delay(50);//wait for reliable sensor read
  return distance;
}
```