

**TC.  
KARABÜK ÜNİVERSİTESİ  
MÜHENDİSLİK FAKÜLTESİ  
MEKATRONİK MÜHENDİSLİĞİ BÖLÜMÜ**

**MATLAB GUI VE ARDUINO İLE MESAFE ÖLÇÜMÜ**

**MEKATRONİK PROJE UYGULAMASI-1 FİNAL TESLİM RAPORU**

**HAZIRLAYAN**

**Furkan GÜNTÜRKÜN**

**2014010226051**

**Semih HINIK**

**2014010226057**

**DANIŞMAN**

**Dr. Öğr. Üyesi Cihan MIZRAK**

**KARABÜK-2020**

## İÇİNDEKİLER

ÖNSÖZ.....	I
ÖZET.....	II
1.GİRİŞ .....	1
2.KULLANILAN MALZEMELER.....	2
2.1 ULTRASONİK SENSÖR (HC_SR04 ULTRASONİK MESAFE SENSÖRÜ).....	2
2.1.1. ULTRASONİK SENSÖR ÖZELLİKLERİ.....	5
2.1.2.ULTRASONİK SENSÖR NASIL ÇALIŞIR?.....	5
2.2. ARDUİNO UNO R3.....	7
3. SERİ HABERLEŞME.....	8
3.1. SERİ HABERLEŞME YOLUYLA ARDUİNO'YA GELEN VERİLERİN OKUNMASI.....	10
3.2. ÖRNEK PROGRAM İLE FONKSİYONLARIN KULLANIMI.....	11
3.3. NEWPİNG KÜTÜPHANESİ İLE MESAFE HESAPLAMA KODLARI.....	12
3.4 . PROGRAMIN MATLAB KODLARI.....	14
KAYNAKÇA.....	21

## ÖNSÖZ

Sensör, fiziksel ortamda algıladıkları olayları sonucunda elektriksel olarak geri dönüş veren elektronik malzemedir. Sensörler kullanım amaçlarına göre günümüzde birçok teknolojik alanda kullanılmaktadır. Sensörler uygulama ve çalışma ortamlarına göre farklılık göstermektedir. Bu projede ultrasonik mesafe sensörünün çalışma prensibi, Arduino UNO ile programlanması ve Matlab GUI ile kişiye mesafe bilgisi gösterme adımları detaylı olarak anlatılmıştır.

Furkan GÜNTÜRKÜN

Semih HINIK

KARABÜK-2020

## ÖZET

Bu projede ultrasonik mesafe sensöründen sürekli olarak aldığımız dijital verileri Arduino UNO ile bilgisayara aktarıp, Matlab GUI üzerinde oluşturduğumuz program üzerinden görüntüleme amaçlanmıştır. Kod editörü ve derleyici olarak kullanılan, aynı zamanda programı karta yükleme işlemini de gerçekleştiren Arduino IDE kullanılmıştır.

Sensör yardımıyla alınan mesafe bilgisi Arduino'ya verilmiştir. Arduino çıkışlarının elektronik devreyle bağlantısı gerçekleştirilmiştir. Mesafe bilgisine göre arduino, ultrasonik mesafe sensörü aracılığıyla cisimleri algılayarak verileri bilgisayara aktarmış ve yazılan basit bir programda kullanıcıya sunmuştur.

## 1. GİRİŞ

Sensör, fiziksel ortamda algıladıkları olayları sonucunda elektriksel olarak geri dönüş veren elektronik malzemedir. Sensörler kullanım alanlarına göre günümüzde birçok teknolojik alanda bulunmaktadır. Dijital ve Analog olmak üzere 2 çeşit sinyal çıkışı veren sensör vardır.

Sensörler çalışma alanlarına göre pek çok çeşitte bulunur. Birçok markada ve özelliklerde sensör temin etmek mümkündür.

Sensör farklı kullanım alanlarına göre günlük hayatta elektronik cihazlarda kullanılmaktadır. Basit bir örnek olarak sensörlü lamba olarak bilinen merdiven boşluklarında kullanılan lamba verilebilir.

Bir gaz sensörü ortamdaki gaz kaçağını belirler. Gaz kaçağının olabilme ihtimali bulunan her yerde kullanılabilir.

Kızılötesi algılama Sensörü (IR Sensörü) Nesne ve mesafe algılama gibi çeşitli farklı uygulamalarda kullanılan ışık bazlı sensörlerdendir. Renk tahmin etme, cisim algılama, uzaklık ölçümü ve robotik uygulamalarda kullanılmaktadır.

Endüstriyel uygulamalarda ultrasonik sensörler, sıvı seviyesi belirleme, akışkanlık belirleme, fabrikalarda ürün algılama ve daha birçok farklı işlemlerde kullanılırlar.

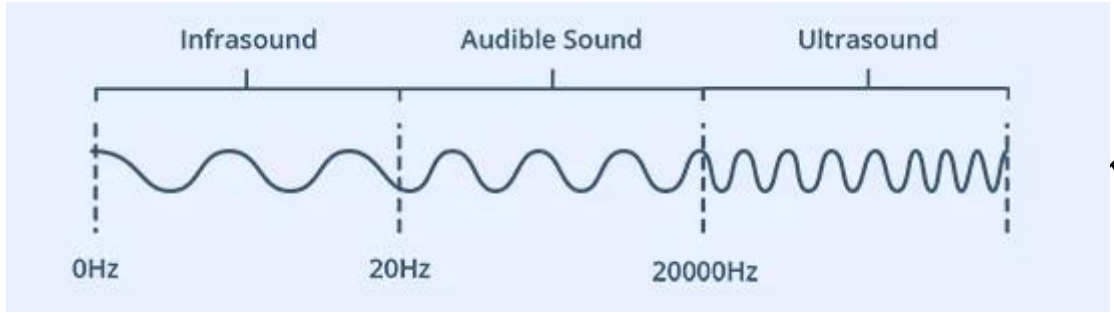
Bu projede ise ultrasonik mesafe sensörünün nasıl çalıştığı ile ilgili bilgi verilmiştir.

## 2. KULLANILAN MALZEMELER

### 2.1. ULTRASONİK SENSÖR (HC\_SR04 ULTRASONİK MESAFE SENSÖRÜ)

20KHz – 1GHz arasındaki ses sinyalleri ultrasonik ses olarak tanımlanır. Ultrasonik cihazlar, mesafe ve seviye ölçümü gerektiren çok çeşitli endüstriyel alanda kullanıldığı gibi güvenlik alarm sistemleri ve robot sistemlerinde de sıkça kullanılmaktadır.

Ultrasonik mesafe sensörü, insan kulağının duyabileceğinden daha yüksek frekanslara sahip yüksek genlikli ses dalgaları gönderen, geri gelen dalganın dönüş süresini hesaplayıp cisme olan uzaklık mesafesini belirleyen cihazdır.



Şekil 2.1.1 Ses dalga spektrumu

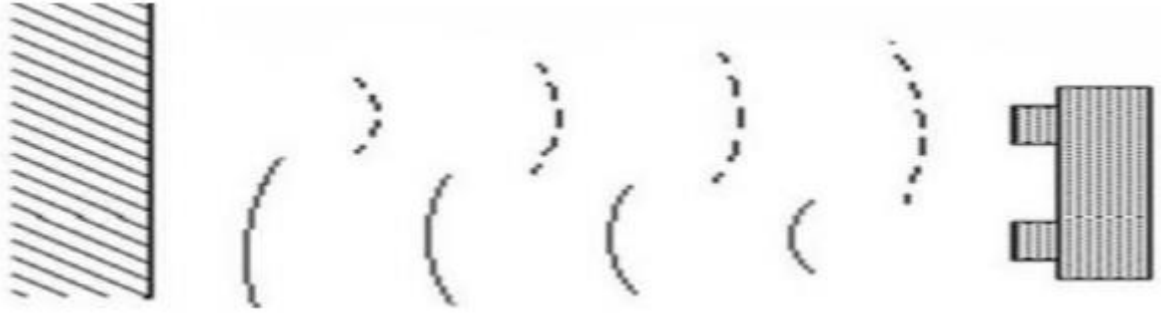
Ultrasonik ses dalga boyları 20.000 Hz'den daha yüksek frekanslara sahiptir. Ancak 20.000'den yüksek frekansları insan kulağı işitemez.

HC-SR04 Ultrasonik mesafe sensörü iki ultrasonik dönüştürücüden oluşur. Radar ve sonara benzer şekilde, ultrasonik dönüştürücüler, geri dönen sinyallerin süresini hesaplayarak mesafe tespiti yapmaktadır.

Pasif ultrasonik sensörler belirli koşullarda gelen ultrasonik sesi algılayan mikrofonlardır.

Dönüştürücüler (alıcı-verici), elektrik sinyalini 40 KHz ultrasonik ses darbelerine dönüştüren bir verici görevi görür. Alıcı, iletilen darbeleri dinler. Gelen dalgaların geri dönüş süresini hesaplayarak mesafeyi ölçer.

Ultrasonik sesi ileten verici kısımların ve ultrasonik sesi dinleyen alıcı kısımların yapısında ince piezoelektrik seramikler bulunmaktadır. Ultrasonik uzaklık sensörü piezoelektrik transducer (hoparlör)den gelen 40 kHz'lik ultrasonik sesin kısa darbelerini yayarak çalışmaktadır. Ses enerjisinin küçük bir kısmı sensörün önündeki maddelerden geri yansıyor diğer piezoelektronik transducere (mikrofon) gelir. Ultrasonik hoparlörden ses dalgalarının çıkış anı ile bu ses dalgalarının cisme çarpıp geri yansıyor ultrasonik mikrofon tarafından algılanması arasında geçen zaman hesaplanır. Ölçülen zamanın ikiye bölünüp ses hızıyla çarpması ile cisim ile ultrasonik sensör arasındaki mesafe bulunmuş olur.



Şekil 2.1.2 Ses dalgalarının engele çarpıp yansıması



Şekil 2.1.3 HC\_SR04 ULTRASONİK MESAFE SENSÖRÜ

**HC\_SR04 Ultrasonik mesafe sensörünün 5 adet pini var:**

VCC: Arduino'ya 5V pin bağladığımız HC-SR04 Ultrasonik mesafe sensörü için güç kaynağıdır.

Trigg (Trigger): Ultrasonik ses sinyallerini tetiklemek için kullanılır.

ECHO: yansıtılan sinyal alındığında bir BPM üretir. Pulse uzunluğu, iletilen sinyalin algılanması için geçen süre ile orantılıdır.

GND: Arduino topraklarına bağlanmalıdır.



### 2.1.1. Özellikleri:

Bu ultrasonik mesafe sensörü 2cm'den 400 cm'ye kadar 3mm hassasiyetle ölçüm yapabilmektedir

Çalışma Gerilimi	DC 5V
Çalışma Akımı	15mA
Çalışma Frekansı	40KHz
Maksimum Ölçüm	4m
Minimum Ölçüm	2cm
Ölçüm Aralığı	3mm
Ölçüm Açısı	15°
Tetikleyici giriş Sinyali	10µS TTL

### 2.1.2. HC-SR04 Ultrasonik Mesafe Sensörü Nasıl Çalışır?

Trigger pinine en az 10 µS (10 mikrosaniye) süreli bir darbe uygulandığında süreç başlar. Buna cevap olarak, sensör 40 KHz'de 8 darbeden oluşan bir ses patlaması gönderir. Bu 8 darbe, cihazın benzersiz ses imzası ile çıkar ve alıcının gelen özel ses dalgalarını, ortam gürültüsünden ayırt etmesini sağlar.

8 ultrasonik ses darbesi vericiden çıktıktan ve cisimden yansıdıktan sonra ECHO pinine ulaşır. ECHO pini sinyalin başlangıcını oluşturmaya başlamak için HIGH durumuna geçer.

Giden ses dalgaları geri yansımazsa, ECHO sinyali 38 mS (38 milisaniye) sonra zaman aşımına uğrar ve azalır. Böylece 38 mS'lik bir darbe sensörün önünde herhangi bir cisim olmadığını gösterir.

Darbeler cisme çarpıp geri yansır, sinyal alınır alınmaz ECHO: LOW olur. Bu olay, sinyalin alınması için geçen süreyle orantılı olarak genişliği 150  $\mu$ S ile 25 mS arasında değişen bir darbe üretir.

Alınan darbenin çıkış ve alınış süresi arasında geçen zaman, nesneye olan mesafeyi hesaplamak için kullanılır. Bu hesaplama basit mesafe-hız-zaman denklemi kullanılarak çözülebilir.

Sensörün önünde bilinmeyen bir mesafede bir cisim olduğunu ve ECHO pininde 500  $\mu$ S genişliğinde bir darbe alındığı düşünölsün. Nesnenin sensörden ne kadar uzak olduğunu hesaplınsın. Aşağıdaki denklem kullanılır.

$$\text{Mesafe} = \text{Hız} \times \text{Zaman}$$

Burada Zaman değeri yani 500  $\mu$ s değeri sahibiz ve hızı biliniyor. Sesin hızı 340 m / s. Mesafeyi hesaplamak için ses hızı cm /  $\mu$ s'ye dönüştürölmeli. "Mikrosaniye başına santimetre cinsinden ses hızı": 0,034 cm /  $\mu$ s`dir.

$$\text{Mesafe} = 0,034 \text{ cm} / \mu\text{s} \times 500 \mu\text{s}$$

Pulse, sinyalin gönderilmesi ve geri yansıtılması için geçen süreyi gösteriyor, böylece mesafeyi elde etmek için sonucu ikiye bölmek gerekiyor.

$$\text{Mesafe} = (0,034 \text{ cm} / \mu\text{s} \times 500 \mu\text{s}) / 2$$

$$\text{Mesafe} = 8,5 \text{ cm}$$

Nesnenin sensörden **8,5 santimetre** uzakta olduğu anlaşılıyor.

## 2.2. ARDUİNO UNO R3

Arduino, bilgisayar aracılığıyla programlanarak çeşitli projeler yapılabilen bir mikrokontrolcü platformudur. Arduino Uno, USB girişine ve adaptör girişine sahiptir. USB girişi sayesinde bilgisayara bağlanıp kolay bir şekilde kod atılabilir ya da bilgisayar ile haberleşmesi sağlanabilir.

Adaptör girişiyle ise adaptör ya da pil sayesinde arduino için gereken gücü alması sağlanır ancak usb ile bilgisayara bağlıyken ya da başka kaynaktan güç alırken bu girişi kullanmak zorunda değildir.

Arduino Uno üzerinde 3 adet led bulunur, bunların ikisi RX ve TX ledleridir, seri haberleşme sırasında çalışır diğer led ise 13. pine bağlıdır bu sayede led uyarısı istenilen durumlarda harici led bağlamadan bu led kullanılabilir.

Arduino Uno'nun bulundurduğu ATmega 328, 32 Kb'lık bir hafızaya sahiptir. Fakat bunun 0.5 Kb'lık bölümü Arduino bootloader'ı tarafından kullanılmaktadır. Bunun yanında Arduino Uno 2 Kb RAM ve 1 Kb EEPROM içerir. Arduino Uno bilgisayar ile, başka bir Arduino ile veya diğer mikrodenetleyiciler ile haberleşme sağlar.

ATmega328, RX ve TX pinlerinden erişilen UART TTL seri haberleşmeyi destekler. Kart üzerindeki ATmega16U2, seri haberleşmeyi USB üzerinden sağlar ve bilgisayardaki yazılımda sanal bir com portu olarak görünür. 16U2 standart USB com sürücülerini kullanır ve harici sürücü gerektirmez.

Kart üzerindeki RX ve TX ledleri veri alış-verişi sırasında yanıp söner. SoftwareSerial kütüphanesi Arduino Uno'nun dijital pinlerinden herhangi biri üzerinden seri haberleşmeye olanak sağlar.

Matlab ile seri porttan sensör verileri okunabilmesi için arduino'nun bu değerleri kullanıcıya seri port üzerinden geri göndermesi gerekir. İlk olarak arduino'nun kendi kod arayüzü kullanılarak sensör verilerinin seri porta gönderilmesi gerekiyor.

### **3. Seri Haberleşme**

Seri iletişim, dijital bilginin (1 ve 0) tek bir hat üzerinden sırayla iletilmesi demektir. Arduino'nun ve benzeri birçok mikrodenetleyicinin üzerinde seri iletişim birimleri bulunur. Bu birimlerle seri iletişim sağlanır. Bu birime UART / USART adı verilir.

Seri iletişimde belirli bir format kullanılır ve bu formatta veri baytlar halinde iletilir. Her bir bayt için belirli bir başlangıç ve bitiş bitleri bulunur. En çok kullanılan 1 bit start, 8 bit veri ve 1 bit stop bitidir, yani 1 bayt veri iletimi için 10 bit gönderilir.

Bilgisayardan Arduino'yu programlarken de seri iletişim kullanılır. Yazılan program derlendikten sonra seri iletişim yoluyla Arduino'nun hafızasına gönderilir.

Arduino sketch programında seri haberleşme örneği için basit bir program yazılabilir.

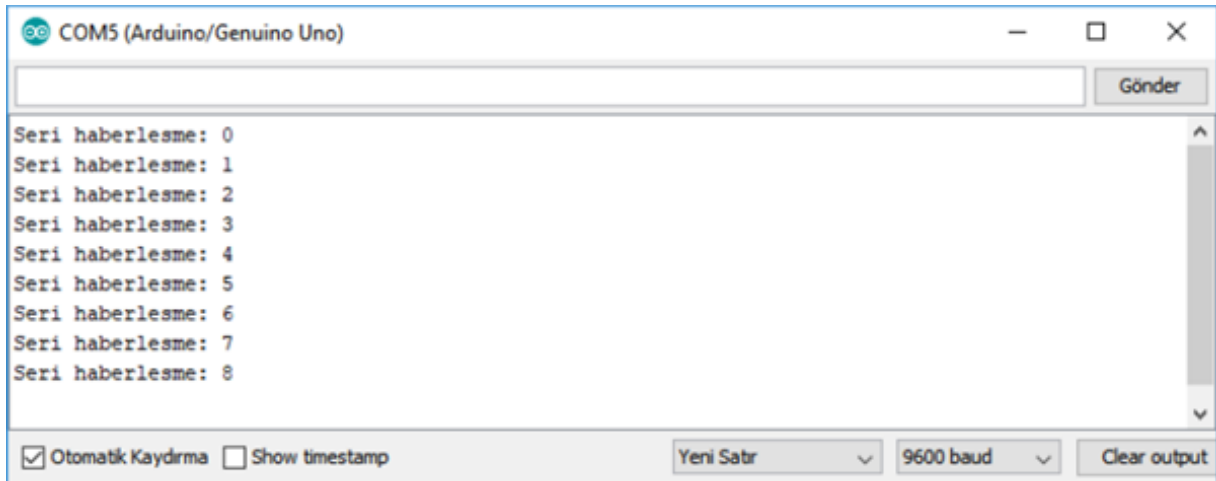
```
void setup()
{
  Serial.begin(9600); //Seri haberleşme hızı
}

int sayac = 0;

void loop()
```

```
{  
  
  Serial.print("Seri haberlesme: ");  
  
  Serial.println(sayac);  
  
  sayac++;  
  
  delay(1000);  
  
}
```

### Programın Serial Monitor çıktısı;



Programda setup() içerisinde Serial.begin(9600) şeklinde çağırılan fonksiyon, iletişim hızını ayarlar. Seri haberleşmede iletişim hızı “baud” (bits per second) adı verilen bir değerle ifade edilir ve bu değer saniyede gönderilen bit sayısını ifade eder. Serial monitor’ün sağ alt köşesindeki değerle programdaki değerlerin aynı olması gerekmektedir. Arduino’den gönderilen verileri düzgün görüntüleyebilmek için bu değerler aynı olmalıdır.

Serial.print() ve Serial.println() fonksiyonları ile string’ler doğrudan yollanıyor. İki fonksiyon arasındaki fark; println() fonksiyonunun string sonuna bir de satır sonu

karakteri (alt satıra geçmek için) eklemesidir. Bu fonksiyonlara int, float türünde sayılar da verilebilir.

Serial Monitor ASCII formatında gelen verileri görüntülemeyi sağlar. ASCII tablosunda her bir karaktere karşılık gelen bir sayısal değer mevcuttur. Seri haberleşmede bu değer gönderilir ve Serial Monitor ya da benzeri başka programlarda bu sayılara karşı gelen karakterler ekrana getirilir. Yani gelen veriler 0-255 arasında (8 bit) değerlerden oluşur.

### **3.1. Seri Haberleşme yoluyla Arduino'ya gelen Verilerin Okunması**

Arduino'ya seri iletişim üzerinden gelen veriler bir bellekte (128 bayt) saklanır. Buna “tampon bellek” de denilebilir, veriler program içerisinde okunana kadar bellekte saklanır. Bu belleğin kapasitesi sınırlı olduğu için veriler belirli bir süre içerisinde okunmazsa, yeni gelen veriler eskisinin üzerine yazılacaktır.

Belirli bir anda bellekte okunmayı bekleyen verilerin sayısı Serial.Avaliable() fonksiyonu ile öğrenilebilir: Serial.Avaliable() fonksiyonunun değeri sıfırdan farklı ise okunmayı bekleyen veriler var anlamına gelir. Gelen veriler Serial.read() ile okunabilir. Serial.read() fonksiyonu, tampon bellekte bekleyen verilerden ilk gelmiş olanı bize gösterir. Tampon belleğe ilk gelen veri ilk okunur, buna “ilk gelen ilk okunur (first in first out – FIFO) bellek” denilir. Bu fonksiyon ile her seferinde bir bayt okunur ve bu bayt daha sonra tampon bellekten silinir.

### 3.2. Örnek program ile fonksiyonların kullanımı:

```
int gelenVeri = 0; // genel veriyi tutacak değişken

char gelenKarakter

void setup()

{
    Serial.begin(9600); // Seri haberleşmeyi başlatılır (9600 bps hızında)
}

void loop()

{
    // Veri gelmiş mi?
    if (Serial.available() > 0)
    {
        // gelen veriyi oku
        gelenVeri = Serial.read();
        gelenKarakter = gelenVeri;
        Serial.print("Gelen Veri: ");
        Serial.println(gelenVeri);
        Serial.print("Gelen Karakter: ");
        Serial.println(gelenKarakter);
    }
}
```

### 3.3. Newping kütüphanesi ile mesafe hesaplama kodları

```
#include "NewPing.h" // NewPing kütüphanesi tanımlanır

#define TRIGGER_PIN 9 //Trig pini, Arduinodaki 9. Pine bağlanır

#define ECHO_PIN 10 //Echo pini Arduinodaki 10. Pine bağlanır


#define MAX_DISTANCE 400 //Maksimum mesafe programa belirtilir
(santimetre cinsinden)


NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE); //Tanımlanan
pinler ve maksimum mesafe kütüphaneye yüklenir.

float duration, distance;


void setup()

{

    Serial.begin(9600); //Seri haberleşme başlatılır.

}

void loop()

{

    distance = sonar.ping_cm(); //Ses gönderilir ve geri dönme süresine göre
çıkan değer santimetre cinsine çevrilir

    Serial.print("Mesafe = "); //Sonuçlar serial monitör'e gönderilir

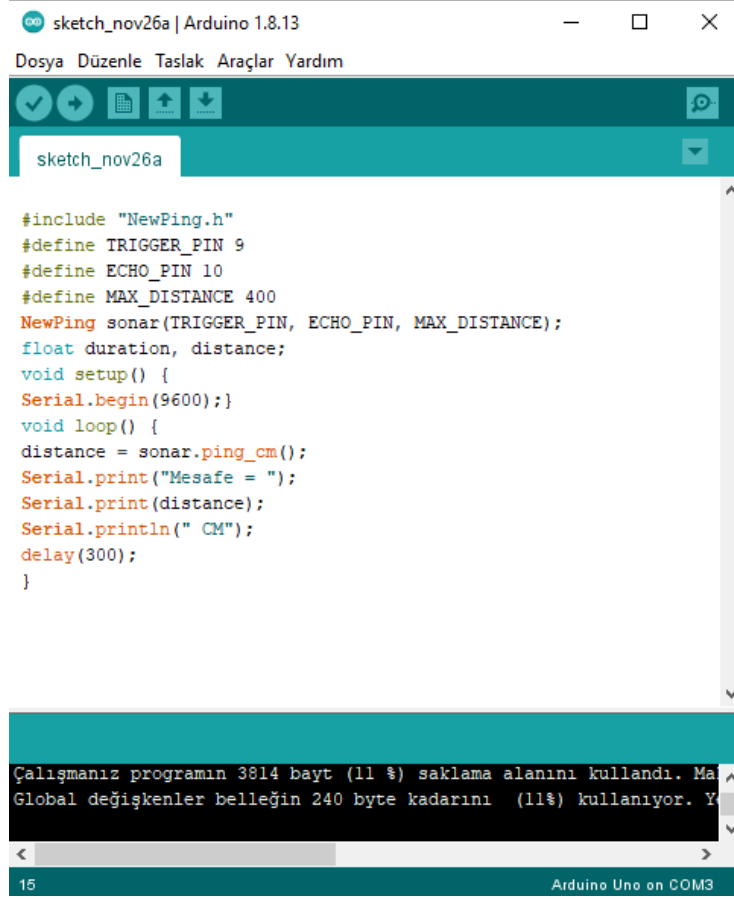
    Serial.print(distance);

    Serial.println(" CM");

    delay(500);

}
```



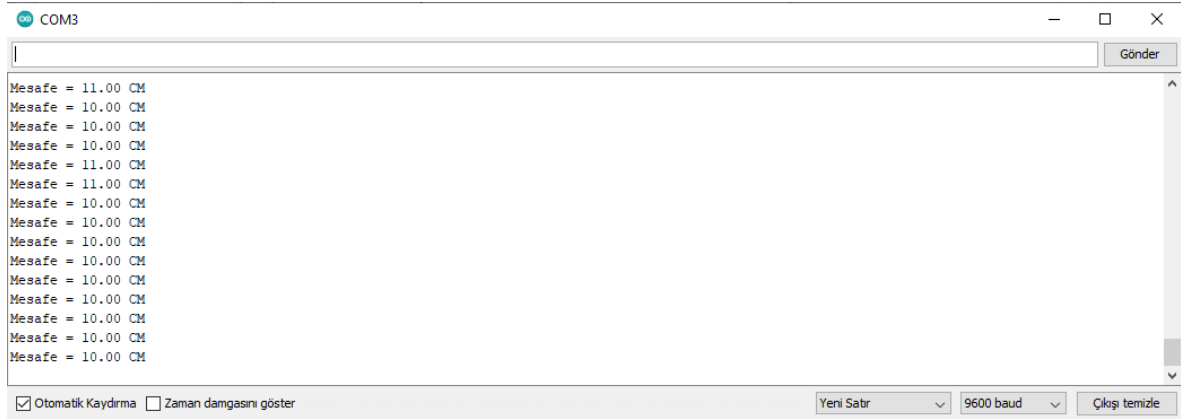


```
#include "NewPing.h"
#define TRIGGER_PIN 9
#define ECHO_PIN 10
#define MAX_DISTANCE 400
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);
float duration, distance;
void setup() {
  Serial.begin(9600);}
void loop() {
  distance = sonar.ping_cm();
  Serial.print("Mesafe = ");
  Serial.print(distance);
  Serial.println(" CM");
  delay(300);
}
```

Çalışmanız programın 3814 bayt (11 %) saklama alanını kullandı. Ma  
Global değişkenler belleğin 240 byte kadarını (11%) kullanıyor. Y

15 Arduino Uno on COM3

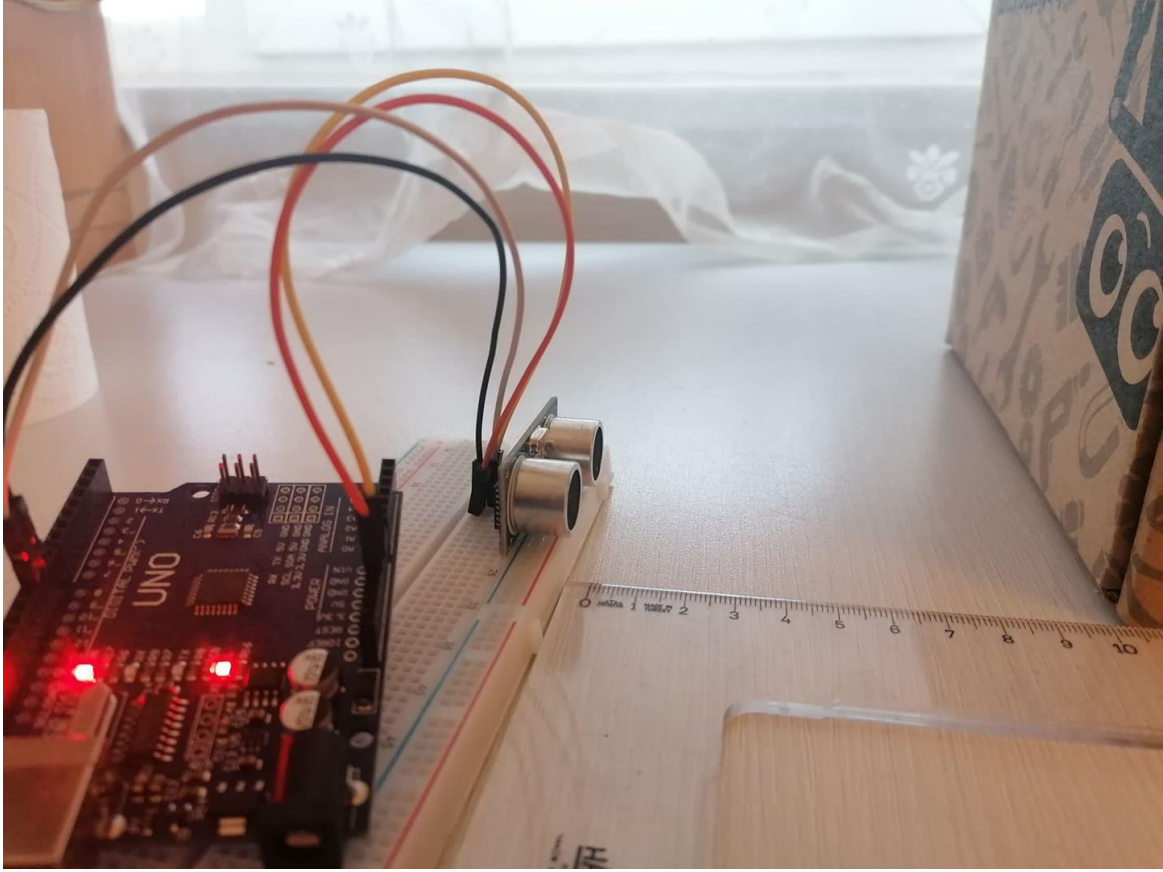
Programın kodlarla birlikte ekran görüntüsü.



```
Mesafe = 11.00 CM
Mesafe = 10.00 CM
Mesafe = 10.00 CM
Mesafe = 10.00 CM
Mesafe = 11.00 CM
Mesafe = 11.00 CM
Mesafe = 10.00 CM
Mesafe = 10.00 CM
Mesafe = 10.00 CM
Mesafe = 10.00 CM
Mesafe = 10.00 CM
Mesafe = 10.00 CM
Mesafe = 10.00 CM
Mesafe = 10.00 CM
Mesafe = 10.00 CM
Mesafe = 10.00 CM
Mesafe = 10.00 CM
```

☒ Otomatik Kaydırma ☐ Zaman damgasını göster Yeni Satır 9600 baud Çıkış temizle

Programın seri port ekranı görüntüsü.



Seri port ekran görüntüsü çekildiği sırada devre bu şekilde bulunmaktadır.

### 3.4. Programın MATLAB kodları

```
function varargout = SENSOR(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @SENSOR_OpeningFcn, ...
    'gui_OutputFcn',  @SENSOR_OutputFcn, ...
    'gui/LayoutFcn',  [] , ...
    'gui_Callback',   []);

if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargin
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function SENSOR_OpeningFcn(hObject, eventdata, handles, varargin)
```

```

guidata(hObject, handles);
handles.output = hObject;

guidata(hObject, handles);

function varargout = SENSOR_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;


function basla_Callback(hObject, eventdata, handles)

com=get(handles.port, 'value');
com=num2str(com);

com=strcat('COM',com);
haberlesme=serial(com, 'BaudRate', 9600);

set(haberlesme, 'InputBufferSize',256);
set(haberlesme, 'OutputBufferSize',256);
set(haberlesme, 'RequestToSend', 'off');

set(haberlesme, 'BaudRate', 9600);
set(haberlesme, 'Timeout',2);

try
    fopen(haberlesme);
catch
    fclose(instrfind('Port',com));
    try
        fopen(haberlesme);
    catch
        at=strcat(com,'Port Acilamiyor !');
        errordlg(at);
        set(handles.status,'string','Waiting');
        set(handles.status,'BackgroundColor','red');
        return;
    end
end
set(handles.status,'BackgroundColor','green');
set(handles.status,'string','Calisiyor');

durum=get(handles.status,'String');

while strcmp(durum,'Calisiyor')

    durum=get(handles.status,'String');
    pause(0.01);

```

```

gelenveri = fscanf(haberlesme);

boyut=length(gelenveri);

gelenveri=gelenveri(boyut-9:boyut-5);
disp(gelenveri)

yazdir=str2double(gelenveri);

set(handles.mesafe, 'String', yazdir);

if yazdir<5
    set(handles.yakinlik, 'String', 'Dur!')
    set(handles.yakinlik, 'BackgroundColor', 'r');
end

if yazdir>=5&&yazdir<10
    set(handles.yakinlik, 'String', 'Cok Yakin!')
    set(handles.yakinlik, 'BackgroundColor', 'y');
end

if yazdir>=10&&yazdir<20
    set(handles.yakinlik, 'String', 'Yakin!')
    set(handles.yakinlik, 'BackgroundColor', 'b');
end

if yazdir>=20&&yazdir<30
    set(handles.yakinlik, 'String', 'Uzak!')
    set(handles.yakinlik, 'BackgroundColor', 'g');
end

if yazdir>=30
    set(handles.yakinlik, 'String', '')
    set(handles.yakinlik, 'BackgroundColor', 'w');
end

end

fclose(haberlesme);
delete(haberlesme)

```

```

set(handles.status, 'BackgroundColor', 'red');

function dur_Callback(hObject, eventdata, handles)

    set(handles.status, 'string', 'Bekleniyor');


function mesafe_Callback(hObject, eventdata, handles)


function mesafe_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function distancia_CreateFcn(hObject, eventdata, handles)

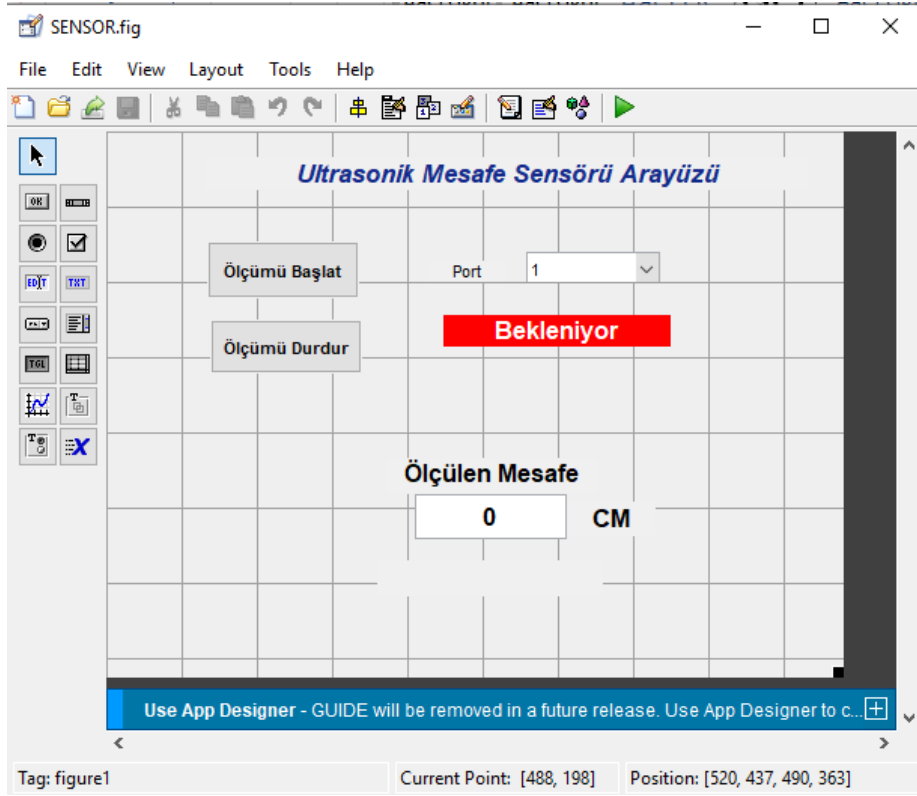

function port_Callback(hObject, eventdata, handles)


function port_CreateFcn(hObject, eventdata, handles)

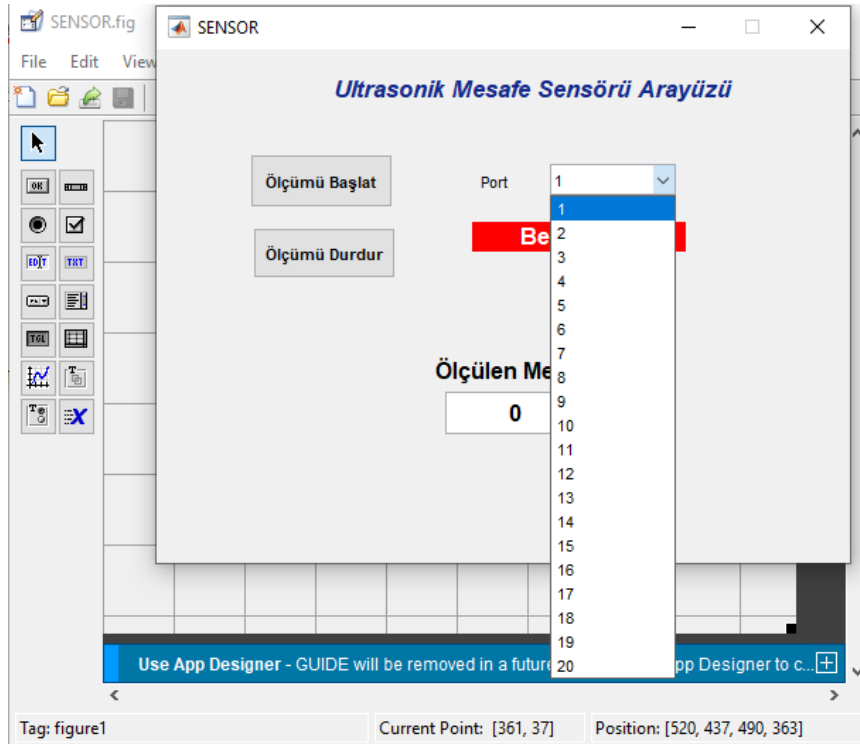
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUicontrolBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function status_CreateFcn(hObject, eventdata, handles)

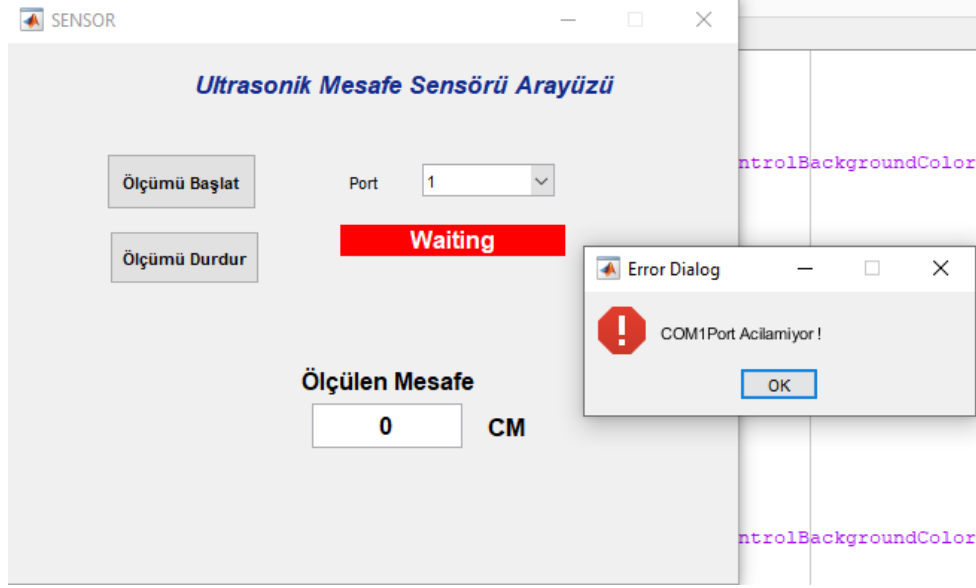
```



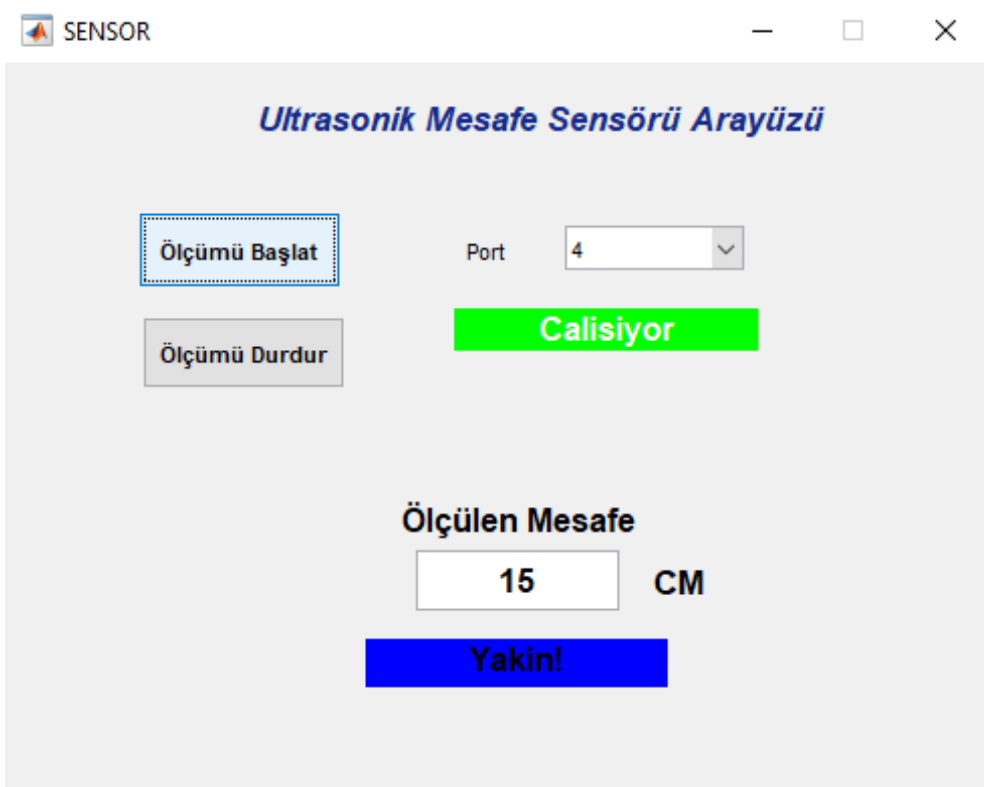
GUI ekranı görüntüsü.



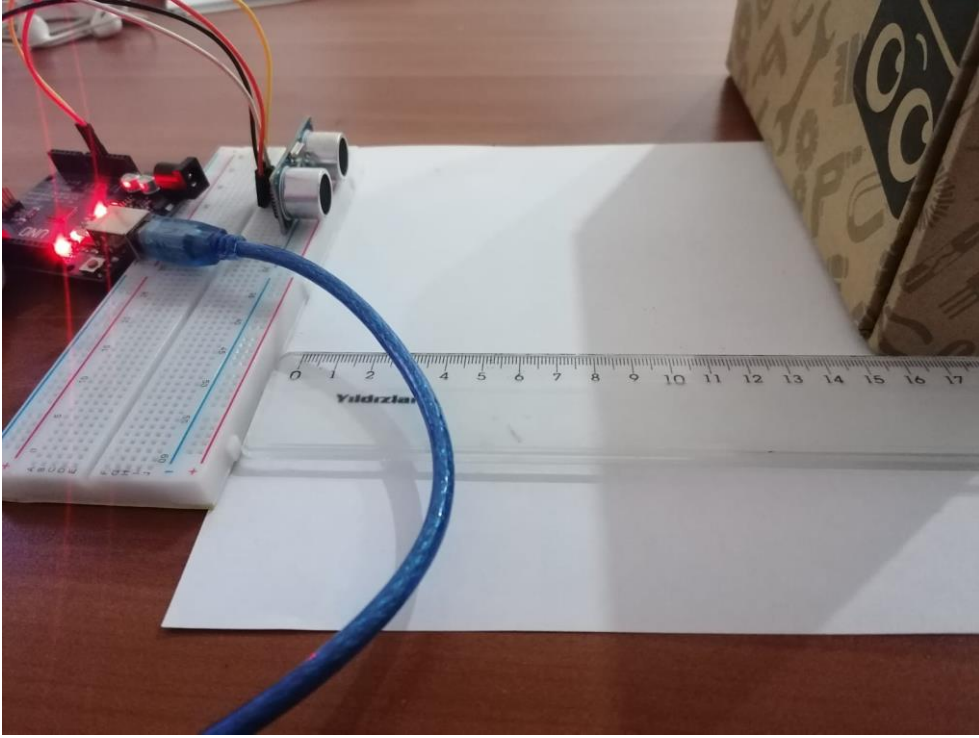
Port seçim menüsü.



Hatalı port seçim uyarısı.



Ölçüm yapılıyor.



Ölçüm yapıldığı sırada sensör ve arduino.



## KAYNAKÇA

<https://maker.robotistan.com/arduino-dersleri-19-hc-sr04-ultrasonik-mesafe-sensoru-kullanimi/>

<https://maker.robotistan.com/arduino-ile-basit-park-sensoru-yapimi/>

<https://maker.robotistan.com/kategori/arduino/>

<https://hayaletveyap.com/arduino-ile-ultrasonik-mesafe-sensoru-hc-sr04-kullanimi/>

<https://www.arduino-media.com/arduino-ve-hc-sr04-ultrasonik-sensor-ile-mesafe-olcumu/>