



دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر

گزارش نهایی پروژه‌ی درس آزمایشگاه سخت‌افزار

عنوان:

عصای هوشمند

نگارندهان:

محمد فراهانی، کسری عبدالله

استاد:

دکتر علیرضا اجلالی

دستیار آموزشی:

دکتر امین فصحتی

شهریور ۱۴۰۱

اللهُ أَكْبَرُ

فهرست مطالب

۸

۱ مقدمه

۱۰

۲ معماری سیستم

۱۰ ۱-۲ طراحی و پیاده‌سازی ساخت افزارها

۱۱ ۱-۱-۲ سنسور فاصله‌سنج HY-SRF05

۱۴ ۲-۱-۲ ماژول بلوتوث HC-05 Bluetooth Module

۱۷

۳ راهکار ناموفق با Raspberry pi

۲۶

۴ راهکار موفق با Arduino

۳۳

۵ آزمایش‌ها

۳۳ ۱-۵ آزمایش‌های Raspberry pi

۳۳ ۲-۵ آزمایش‌های سنسور فاصله‌سنج HY-SRF05

۳۴ ۱-۲-۵ آزمایش حداکثر فاصله قابل اندازه‌گیری

۳۵ ۲-۲-۵ آزمایش اندازه‌گیری در حالت نزدیک شدن مانع

۳۷ ۳-۲-۵ آزمایش اندازه‌گیری در فواصل مشخص

۳۸ ۴-۲-۵ آزمایش اندازه‌گیری با زاویه در حالت افقی

۴۲	۵-۲-۵ آزمایش اندازه‌گیری زاویه در حالت عمودی	۵-۲-۵
۴۲	۶-۲-۵ آزمایش در صورتی که یک مانع به سنسور چسبیده	۶-۲-۵
۴۴	۳-۵ آزمایش‌های اتصال سنسورها	۳-۵
۴۴	۱-۳-۵ آزمایش اتصال به وسیله سیم جامپر	۱-۳-۵
۴۵	۲-۳-۵ آزمایش اتصال از طریق پشت برد	۲-۳-۵
۴۶	۳-۳-۵ آزمایش اتصال به وسیله سیم مفتولی مسی	۳-۳-۵
۴۷	۴-۵ آزمایش منبع تغذیه و سنسورها و ماژول بلوتوث	۴-۵
۵۱	۶ هزینه	۶
۵۳	۷ بسته‌بندی	۷
۵۵	۸ اپلیکیشن اندروید	۸
۵۵	۱-۸ صفحات اپلیکیشن	۱-۸
۵۹	۹ پیشنهادات	۹
۶۰	۱۰ جمع‌بندی	۱۰

فهرست تصاویر

۱۰	معماری سیستم	۱-۲
۱۱	شیوه ای اتصال سنسور HY-SRF05 به Arduino	۲-۲
۱۳	سنسور اولتراسونیک HY-SRF05	۳-۲
۱۴	ماژول بلوتوث HC-05	۴-۲
۱۵	اتصال ماژول بلوتوث HC-05 به Arduino	۵-۲
۱۷	اتصال سنسور HY-SRF05 به Raspberry pi	۱-۳
۲۱	جدول پین های GPIO Extension Board	۲-۳
۲۳	مدار کاهش دهنده ولتاژ پین TRIG با استفاده از ۳ مقاومت	۳-۳
۲۶	مدار نهایی و اتصال قطعات به یکدیگر	۱-۴
۲۷	اتصال سنسور HY-SRF05 به Arduino	۲-۴
۳۴	اتصال سنسور HY-SRF05 به Arduino در عمل	۱-۵
۳۴	منع تغذیه ولتاژ و جریان آن	۲-۵
۳۵	حداکثر برد سنسور HY-SRF05 در عمل	۳-۵
۳۵	رابطه زمان اندازه گیری شده توسط سنسور HY-SRF05 با فاصله واقعی	۴-۵
۳۶	ابتدای متر	۵-۵

۳۷	۶-۵ انتهای متر
۳۸	۷-۵ رابطه‌ی فاصله‌ی واقعی با فاصله‌ی اعلام شده توسط سنسور
۳۹	HY-SRF05	۸-۵ تنظیم نقاله و متر و استفاده از آنها برای به دست آوردن میدان دید سنسور
۴۰	۹-۵ زاویه‌ی ۵ درجه
۴۰	۱۰-۵ زاویه‌ی ۱۰ درجه
۴۱	۱۱-۵ زاویه‌ی ۱۳ درجه
۴۱	۱۲-۵ زاویه‌ی ۱۴ درجه
۴۲	۱۳-۵ زاویه‌ی ۱۵ درجه
۴۳	۱۴-۵ شیوه‌ی بررسی فواصل کمتر از ۲ سانتی‌متر
۴۳	۱۵-۵ نتایج بررسی فواصل کمتر از ۲ سانتی‌متر
۴۷	۱۶-۵ حالت سنسور و ولتاژ متصل به قسمت عادی منبع تغذیه با ولتاژ ۵
۴۸	۱۷-۵ حالت سنسور و ولتاژ متصل به قسمت ۵ ولت ثابت
۴۹	۱۸-۵ حالت این که هر کدام به یک قسمت متفاوت منبع متصل شوند
۵۴	۱-۷ نقاط سنسورهای فاصله‌سنجی روی بدن. فلاش‌ها جهتی که سنسورها اندازه‌گیری می‌کنند را نشان می‌دهد.
۵۶	۱-۸ صفحه‌ی اول اپلیکیشن اندروید
۵۸	۲-۸ صفحه‌ی دوم اپلیکیشن اندروید

فهرست جداول

۱-۵ فواصل واقعی و فواصل اندازه‌گیری شده با سنسور HY-SRF05 ۵۰

۱-۶ لیست قطعات به همراه قیمت هر کدام. قیمت‌ها به واحد هزار تومان می‌باشند. ۵۲

فصل ۱

مقدمه

پروژه‌ی عصای هوشمند ابزاری برای جایگزین کردن عصای نابینایان پیاده‌سازی می‌کند. هدف ابزار ارائه شده این بوده است که در دو سطح به کاربر نابینا کمک کند تا به مقصد خود برسد. در سطح اول هدف این است که صرفا اجسام نزدیک کاربر شناسایی شوند. این همان کاری است که فرد نابینا با استفاده از عصای خود و تکان دادن آن با برخورد به اجسام متوجه حضور آنها می‌شود. در سطح دوم هدف آن بوده است که به فرد نابینا کمک کنیم تا در مسیری درست به سمت مقصد قرار بگیرد و دور یا نزدیک شدن او به مقصد به او گزارش شود.

این پروژه به طور کلی از دو بخش تشکیل شده است:

۱. بخش برد سخت‌افزاری و سنسورها: این بخش وظیفه دارد تا دوری و نزدیکی اجسام در جهات مختلف را با استفاده از سنسورهایی تشخیص دهد و به نوعی با ارسال داده، کاربر را از وجود این اجسام مطلع کند.

۲. بخش اپلیکیشن اندروید: این بخش اپلیکیشنی است که دو وظیفه دارد. وظیفه‌ی اول آن دریافت اطلاعات ارسالی از برد سخت‌افزاری و پردازش و هشدار به کاربر از طریق فرمان صوتی است. وظیفه‌ی دوم آن انتخاب مقصد و بررسی دور شدن از یا نزدیکی شدن به مقصد است و همچنین با توجهی به دور شدن یا نزدیک شدن کاربر، فرمانی صوتی به او می‌دهد و هنگامی که کاربر به مقصد رسید او را مطلع می‌کند.

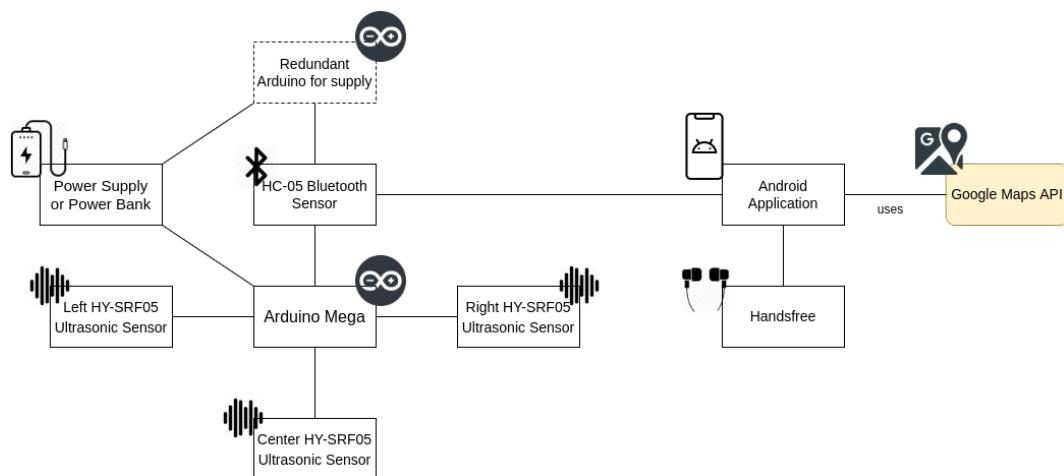
مزیت راهکار ارائه شده در این پروژه نسبت به یک عصای ساده در این نهفته است که اولاً با استفاده

از سنسورها می‌توانیم اجسام را در ارتفاعات مختلف تشخیص دهیم. این در حالی است که کاربر با استفاده از عصای ساده تنها می‌تواند اجسامی که پیش پایش هستند را تشخیص دهد و اجسامی که مثلاً در ارتفاع ۱ متری زمین باشند با محدوده‌ی رایج تکان دادن عصا قابل تشخیص نیستند. مزیت دوم این است که اساساً با استفاده از عصا، کاربر نمی‌تواند بررسی کند که در حال نزدیک شدن به مقصدش است یا دارد از آن دور می‌شود. با استفاده از ابزار ارائه شده کاربر می‌تواند از فردی کمک بگیرد تا مقصدش را برایش مشخص کند و شروع به حرکت به سمت آن کند و از دوری و نزدیکی اش به مقصد مطلع شود.

فصل ۲

معماری سیستم

در شکل ۱-۲ معماری سیستم را مشاهده می‌کنید.



شکل ۱-۲ : معماری سیستم

۱-۲ طراحی و پیاده‌سازی سخت افزارها

قسمت اصلی پروژه طراحی و بخش سخت افزاری آن است که در قسمت زیر لیستی از قطعات استفاده شده در پروژه آمده است و در ادامه برای برخی از آن‌ها توضیحاتی اضافه شده است:

• برد Raspberry Pi 3B

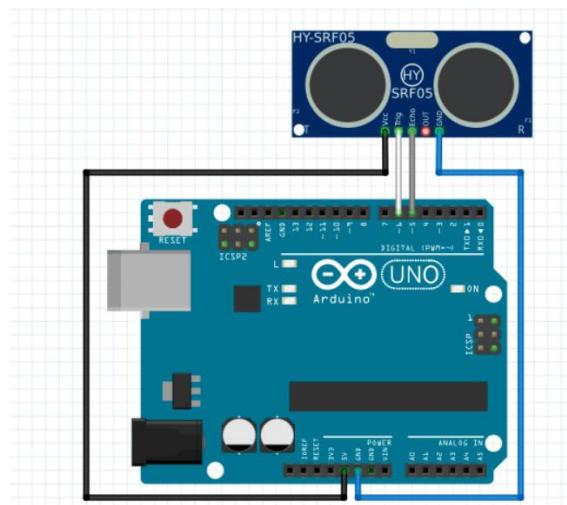
• برد Arduino Mega 2560

• سنسور فاصله‌سنج HY-SRF05

• ماژول بلوتوث HC-05 Bluetooth Module

۱-۱-۲ سنسور فاصله‌سنج HY-SRF05

این سنسور یک فرستنده و گیرنده مأ فوق صوت دارد و به وسیله اختلاف زمان بین فرستادن موج و دریافت بازتاب آن می‌توان فاصله تا مانع را به دست آورد. محدوده اندازه‌گیری فاصله ذکر شده برای این سنسور بین ۲ تا ۴۵۰ سانتی‌متر است و دقت آن $\frac{1}{3}$ سانتی‌متر بیان شده است. این قطعه را می‌توان به صورت آنچه در شکل ۲-۲ به Arduino متصل کرد.



شکل ۲-۲: شیوه‌ی اتصال سنسور HY-SRF05 به Arduino برای مشاهده فاصله‌ها می‌توان از کد زیر استفاده کرد.

```

1 const unsigned int TRIG_PIN=6;
2 const unsigned int ECHO_PIN=5;
3 const unsigned int BAUD_RATE=9600;
4
5 void distance_detector(unsigned int TRIG_PIN, unsigned
int ECHO_PIN){
```

```
6   digitalWrite(TRIG_PIN, LOW);
7   delayMicroseconds(2);
8   digitalWrite(TRIG_PIN, HIGH);
9   delayMicroseconds(10);
10  digitalWrite(TRIG_PIN, LOW);

11
12
13  const unsigned long duration= pulseIn(ECHO_PIN, HIGH);
14  int distance= duration/29/2;
15  if(duration==0){
16      Serial.println("Warning: no pulse from sensor");
17  }
18  else{
19      Serial.print(duration);
20      Serial.print(" s-sensor distance to nearest object:");
21      Serial.print(distance);
22      Serial.println(" cm");
23  }
24 }

25
26 void setup() {
27     pinMode(TRIG_PIN, OUTPUT);
28     pinMode(ECHO_PIN, INPUT);
29     Serial.begin(BAUD_RATE);
30 }

31
32 void loop() {
33     distance_detector(TRIG_PIN, ECHO_PIN);
34     delay(200);
35 }
```

در شکل ۲-۲ پایه‌های سنسور فاصله‌سنج HY-SRF05 مشخص شده‌اند.



شکل ۲-۳: سنسور اولتراسونیک HY-SRF05

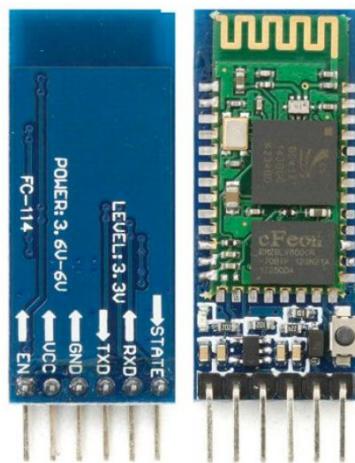
دو پایه‌ی VCC و GND برای اتصال به منبع تغذیه هستند. این سنسور برای کار کردن باید به ولتاژ بین ۴/۵ تا ۵/۵ متصل شود (طبق الگوهای مشاهده شده از دیگر سنسورهای فاصله‌سنج و همین‌طور آزمایش‌هایی که انجام شده برای این‌که به حداقل فاصله ۴۵۰ سانتی متر برسیم باید به یک منبع ولتاژ ثابت ۵/۵ ولت وصل باشد). پایه OUT برای این استفاده می‌شود که بین حالت یک پایه برای Trigger و Echo و دو پایه انتخاب کنیم. به این صورت که اگر این پایه به جایی متصل نباشد برای کار با سنسور باید از هر دو پایه TRIG و ECHO استفاده کنیم ولی اگر این پایه را به GND متصل کنیم می‌توانیم به جای دو پایه از یک پایه استفاده کنیم و یک پایه از Arduino را خالی نگه‌داریم.

پایه TRIG برای شروع عملیات اندازه‌گیری استفاده می‌شود به این صورت هربار که خواستیم یک اندازه‌گیری انجام دهیم یک پالس با حداقل زمان ۱۰ میکرو ثانیه به این پایه می‌دهیم.

پایه ECHO برای اندازه‌گیری زمان تا بازتاب موج ارسال شده است به این صورت که وقتی پالس به پایه Trig داده شد یک موج موفق صوت از سنسور ارسال می‌شود و پایه ECHO هم از LOW به HIGH می‌ورد و وقتی بازتاب موج توسط سنسور دریافت شد پایه ECHO از HIGH به LOW بر می‌گردد. با استفاده از طول بازی زمانی HIGH بودن این پایه و ضرب آن در سرعت صوت و تقسیم آن به ۲ (مسیر رفت و برگشت) می‌توان فاصله موانع از سنسور را به دست آورد.

۲-۱-۲ ماژول بلوتوث HC-05 Bluetooth Module

این ماژول برای ارتباط بیسیم بین دو دستگاه با استفاده از بلوتوث به صورت full duplex استفاده می‌شود. این ماژول با استفاده از USART با baud rate ۹۶۰۰ کار می‌کند و در مجموع ۶ پین، یک LED و یک push button دارد که در شکل ۴-۲ می‌توانید مشاهده کنید.



شکل ۲-۴: ماژول بلوتوث HC-05

در ادامه کارکرد هر کدام از این ۸ جزء توضیح داده خواهد شد.

۱. پین **STATE**: این پین به LED موجود در برد ماژول متصل است و از آن می‌توانیم به عنوان بازخوردی برای بررسی حالت ماژول استفاده کنیم. (مثلاً به یک LED دیگر متصلش کنیم)

۲. پین **RXD**: پین ورودی ماژول و دریافت داده از سایر قطعات به ماژول به صورت سریال است. هر داده‌ای که با استفاده از این پین دریافت شود با بلوتوث به صورت Broadcast منتشر می‌شود.

۳. پین **TXD**: پین خروجی ماژول و ارسال داده از ماژول به سایر بخش‌های یک مدار به صورت سریال است. هر داده‌ای که قرار باشد از ماژول دریافت شود از این پین است.

۴. پین **GND**: پین زمین ماژول است.

۵. پین **VCC**: برای روشن کردن ماژول استفاده می‌شود. به این پین باید ۵ ولت داده شود.

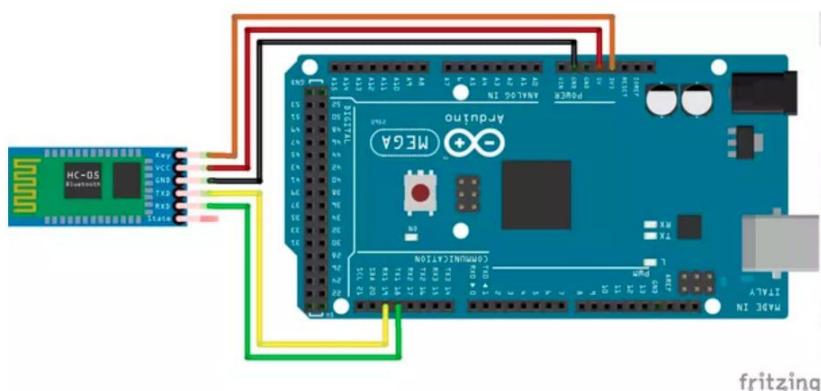
۶. پین **EN**: این پین برای مشخص کردن دو حالت موجود در این ماژول با نام‌های data mode و command mode استفاده می‌شود. به طور پیش‌فرض ماژول در حالت data mode قرار دارد

که برای تبادل اطلاعات با دستگاه‌های دیگر استفاده می‌شود. اگر به این پین LOW بدهیم در حالت data mode و اگر HIGH بدهیم در حالت command mode قرار می‌گیرد.

۷. **LED**: برای نشان دادن وضعیت مژول استفاده می‌شود. مثلا هنگامی که در حالت ارسال و دریافت داده است هر ۲ ثانیه یک بار روشن می‌شود. برای سایر حالات نیز به طوری دیگر چشمک می‌زند.

۸. **button Push**: می‌توان از آن برای تعیین وضعیت پین EN استفاده کرد و بین data mode و command mode سوئیچ کرد.

برای اتصال این سنسور به arduino mega می‌توان مطابق شکل ۲-۵ عمل کرد:



شکل ۲-۵: اتصال مژول بلوتوث HC-05 به Arduino مشاهده فاصله‌ها می‌توان از کد زیر استفاده کرد:

```

1 #include <SoftwareSerial.h>
2 SoftwareSerial MyBlue(19, 18); // RX | TX
3 int flag = 0;
4 int LED = 8;
5 void setup() {
6     Serial.begin(9600);
7     MyBlue.begin(9600);
8     pinMode(LED, OUTPUT);
9     Serial.println("Ready to connect\nDefault password is
10 1234 or 000");

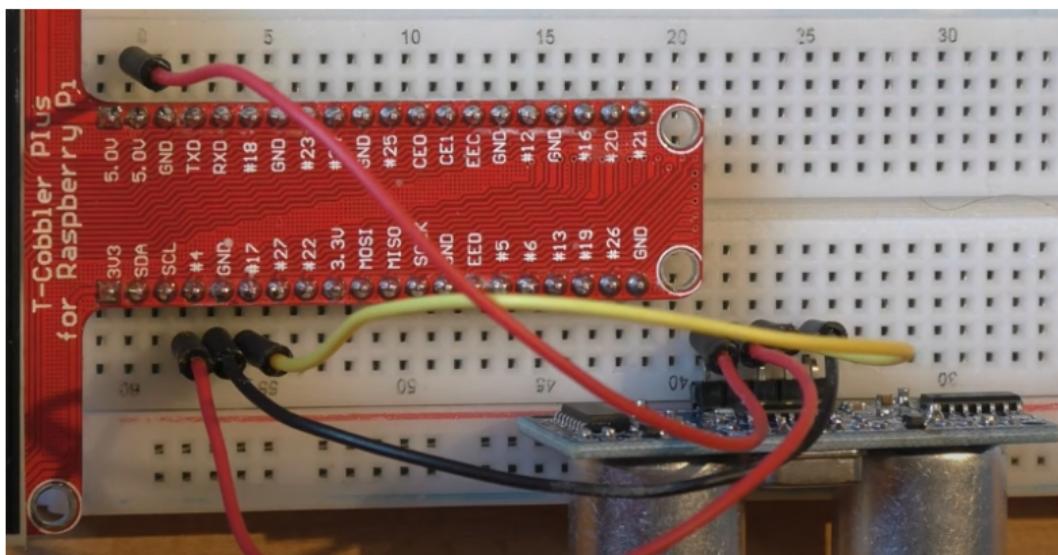
```

```
10 }
11 void loop() {
12     if (MyBlue.available())
13         flag = MyBlue.read();
14     if (flag == 1) {
15         digitalWrite(LED, HIGH);
16         Serial.println("LED On");
17     }
18     else if (flag == 0) {
19         digitalWrite(LED, HIGH);
20         Serial.println("LED Off");
21     }
22 }
```

فصل ۳

راهکار ناموفق با Raspberry pi

در ابتدا سعی کردیم پروژه را با استفاده از برد Raspberry pi اجرا کنیم. علت این بود که با استفاده از Raspberry pi هم قدرت پردازش بالاتری نسبت به Arduino داشتیم، هم نیازی به ماژول جدا برای بلوتوث نبود زیرا خود Raspberry pi دارای بلوتوث است. شیوه‌ی اتصال یک سنسور ultrasonic به Raspberry pi را در شکل ۱-۲ مشاهده می‌کنید.



شکل ۱-۳: اتصال سنسور HY-SRF05 به Raspberry pi

در این شکل پایه‌های VCC و GND سنسور به ترتیب به پین‌های VCC و GND در Raspberry pi متصل شده است. همچنین پین‌های TRIG و ECHO سنسور را به ترتیب به پین‌های GPIO

شماره ۴ و ۱۷ Raspberry pi متصل کردیم. کدی هم که برای تست استفاده کردیم به صورت زیر است:

```

1 import RPi.GPIO as GPIO
2
3 import time
4
5 trigger_pin = 4
6 echo_pin = 17
7 number_of_samples = 5
8 sample_sleep = 0.1
9 calibration1 = 30
10 calibration2 = 1750
11 time_out = 0.5
12
13 GPIO.setwarnings(False)
14 GPIO.setmode(GPIO.BCM)
15 GPIO.setup(trigger_pin, GPIO.OUT)
16 GPIO.setup(echo_pin, GPIO.IN, pull_up_down = GPIO.
    PUD_DOWN)
17
18 samples_list = []
19 stack = []
20
21 def timer_call(channel):
22     now = time.monotonic()
23     stack.append(now)
24
25 def trigger():
26     GPIO.output(trigger_pin, GPIO.HIGH)
27     time.sleep(0.0001)
28     GPIO.output(trigger_pin, GPIO.LOW)
```

```

29
30 def check_distance():
31     samples_list.clear()
32     while len(samples_list) < number_of_samples:
33         trigger()
34         while len(stack) < 2 :
35             start = time.monotonic()
36             while time.monotonic() < start + time_out:
37                 pass
38             trigger()
39
40     if len(stack)== 2:
41         samples_list.append(stack.pop() - stack.pop())
42
43     elif len(stack) > 2:
44         stack.clear()
45
46     time.sleep(sample_sleep)
47
48     return statistics.median(samples_list)*1000000*
49         calibration1/calibration2
50
51 GPIO.add_event_detect(echo_pin, GPIO.BOTH, callback =
52     timer_call)
53
54 for i in range(100):
55     print(round(check_distance(), 1))

```

حال به سراغ توضیح این کد می‌رویم هر چند ما در این پروژه از آن جواب نگرفتیم ولی در اینترنت مثال‌هایی وجود دارد که از Raspberry pi جواب گرفته‌اند.

```

1 import RPi.GPIO as GPIO
2 import time
3 import statistics

```

در این بخش کتابخانه‌های GPIO و time و statistics را به ترتیب برای کار کردن با پین‌های Raspberry pi و زمان و میانگین گرفتن اضافه می‌کنیم.

```

1 trigger_pin = 4
2 echo_pin = 17
3 number_of_samples = 5
4 sample_sleep = 0.1
5 calibration1 = 30
6 calibration2 = 1750
7 time_out = 0.5

```

در این بخش پین‌های ECHO و TRIG را مشخص می‌کنیم و سپس تعداد نمونه‌های گرفته شده برای یک اندازه‌گیری را ۵ تعیین کرده‌ایم. سپس فاصله زمانی بین اندازه‌گیری فاصله بین نمونه‌ها را ۰/۱ ثانیه تعريف کرده‌ایم و سپس زمان اندازه‌گیری شده برای فاصله ۳۰ سانتی‌متر مشخص شده است که ۱۷۵۰ میکروثانیه است (با استفاده از این دو عدد بقیه فاصله‌ها را به صورت نسبی به دست می‌آوریم).

```

1 GPIO.setwarnings(False)
2 GPIO.setmode(GPIO.BCM)

```

ابتدا مشخص کردیم که اخطارها را برای ما نمایش ندهد و سپس مشخص کردیم که از نامگذاری BCM برای پین‌ها استفاده شود که به صورت شکل‌شکل: جدول اکستنشن است. (ستون GPIO نه شماره پین عادی)

```

1 GPIO.setup(trigger_pin, GPIO.OUT)
2 GPIO.setup(echo_pin, GPIO.IN, pull_up_down = GPIO.
    PUD_DOWN)

```

در این بخش خروجی بودن پین TRIG و ورودی بودن پین ECHO مشخص شده همین‌طور نویزهای پین ECHO گرفته شده است.

```

1 samples_list = []

```

Name	wiringPi Pin	BCM GPIO		BCM GPIO	wiringPi Pin	Name
GPIO Extension Board						
3.3V	-	-	3V3	5V0	-	5V
SDA	8	R1:0/R2:2	SDA1	5V0	-	5V
SCL	9	R1:1/R2:3	SCL1	GND	-	0V
GPIO7	7	4	GPIO4	TXDO	14	TXD
GND	-	-	GND	RXDO	15	RXD
GPIO0	0	17	GPIO17	GPIO18	18	GPIO1
GPIO2	2	R1:21/R2:27	GPIO27	GND	-	0V
GPIO3	3	22	GPIO22	GPIO23	23	GPIO4
3.3v	-	-	3V3	GPIO24	24	GPIO5
MOSI	12	10	SPIMOSI	GND	-	0V
MISO	13	9	SPIMISO	GPIO25	25	GPIO6
SCLK	14	11	SPISCLK	SPICE0	8	CE0
0V	-	-	GND	SPICE1	7	CE1
ID_SDA	30	0	ID_SD	ID_SC	1	ID_SCL
GPIO21	21	5	GPIO5	GND	-	0V
GPIO22	22	6	GPIO6	GPIO12	12	GPIO26
GPIO23	23	13	GPIO13	GND	-	0V
GPIO24	24	19	GPIO19	GPIO16	16	GPIO27
GPIO25	25	26	GPIO26	GPIO20	20	GPIO28
GND	-	-	GND	GPIO21	21	GPIO29

شکل ۳-۲: جدول پین‌های GPIO Extension Board

```
2 stack = []
```

در این بخش دو لیست `stack` و `samples_list` برای نگهداری زمان هر نمونه و زمان تغییر پین Echo تعریف شده‌اند.

تابع `timer_call` هر وقت صدا زده شود زمان کنونی را در `stack` می‌ریزد. این تابع در موقع تغییر پین Echo صدا زده‌می‌شود. تابع `trigger` پالس لازم برای شروع کار سنسور را به پین `Trig` می‌دهد. تابع `check_distance` فاصله را اندازه‌گیری می‌کند در ابتدا تمام اعضای `samples_list` را حذف می‌کند و سپس وارد حلقه‌ای می‌شود که تا به تعداد تعیین شده نمونه نگیرد از آن خارج نمی‌شود. در حلقه تابع `trigger` صدا زده می‌شود تا اندازه‌گیری شروع شود و سپس منتظر می‌شود تا طول `stack` از ۲ بیشتر شود یعنی پین Echo از `HIGH` به `LOW` و بعد دوباره به `LOW` رفته است اگر طول `stack` از ۲ کمتر بود به مدت تعیین شده صبر می‌کند اگر از مدت تعیین شده `time_out` بیشتر شد دوباره `trigger` را صدا می‌زنیم که از اول اندازه‌گیری انجام شود. اگر طول صفحه دقیق ۲ تا شد اختلاف زمانی را به عنوان نمونه به `samples_list` اضافه می‌کنیم و اگر بیشتر از ۲ تا بود یعنی که مشکلی وجود داشه پس `stack` را پاک می‌کنیم و دوباره کار را تکرار می‌کنیم. در نهایت هم میانگین ۵ نمونه را در ۱۰۰۰۰۰ ضرب می‌کنیم تا به میکروثانیه تبدیل شود سپس بر `calibration2` که مدت زمان محاسبه شده برای ۳۰ سانتی‌متر تقسیم و در ۳۰ ضرب می‌کنیم.

```
1 GPIO.add_event_detect(echo_pin, GPIO.BOTH, callback =
    timer_call)
```

با این کار در سیستم تعریف می‌شود که هر وقت پین Echo از HIGH به LOW و یا از HIGH به LOW تغییر کرد در سیستم یک interrupt اتفاق می‌افتد که به وسیله interrupt تابع timer_call() اجرا می‌شود به این صورت دیگر نیاز به خواندن ECHO با روش polling نداریم و این کار به وسیله interrupt انجام می‌شود.

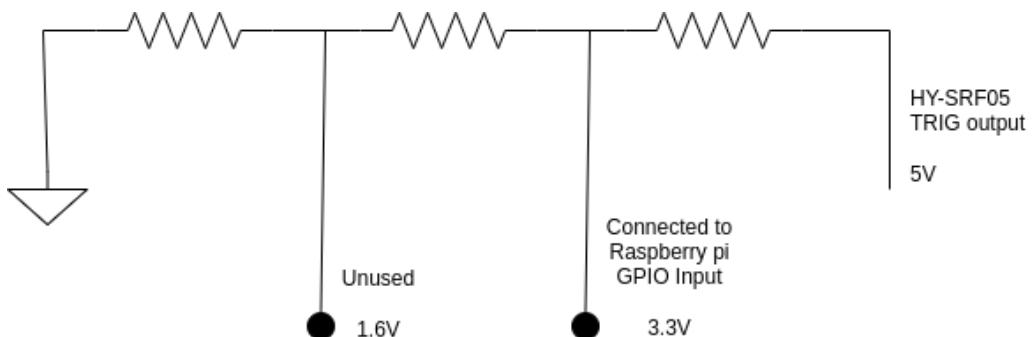
```
1 for i in range(100):
2     print(round(check_distance(), 1))
```

صد بار اندازه‌گیری فاصله انجام می‌شود و در واقع اصل کدی است که در حال اجرا است.

متاسفانه این روش جوابگو نبود و با شکست مواجه شد. نمونه‌ای از خروجی‌های این روش را در زیر مشاهده می‌کنید که تقریباً تمامی فواصل را برابر با ۵ سانتی‌متر نشان می‌دهد و در آزمایش‌ها هنگامی که دست را به سنسور می‌چسباندیم اعدادی بزرگ و عجیب مانند ۱۶۳ که در زیر مشاهده می‌کنید به دست می‌آمد که نشان از کار کرد غلط آن می‌داد.

```
1 5
2 5
3 5
4 5
5 6
6 5
7 5
8 163
9 163
10 162
11 163
12 5
13 5
14 5
15 5
```

علت این است که اگرچه پین VCC در Raspberry pi برابر با ۵ ولت است، اما پین‌های GPIO خروجی $\frac{2}{3}$ ولت می‌دهند و به عنوان ورودی هم $\frac{2}{3}$ ولت نیاز دارند. ابتدا تصور کردیم که خروجی سنسور چون ۵ ولت است و GPIO پین ECHO در Raspberry pi به $\frac{2}{3}$ ولت نیاز دارد نمی‌تواند زمان را به درستی اندازه بگیرد. به همین منظور از مدار شکل ۳-۳ استفاده کردیم تا ولتاژ خروجی ECHO سنسور را به $\frac{2}{3}$ ولت کاهش دهیم.



شکل ۳-۳: مدار کاهش‌دهنده ولتاژ پین TRIG با استفاده از ۳ مقاومت
این مدار به طور کلی از ۳ مقاومت تشکیل شده است که با یکدیگر سری هستند. اگر دو سر این ۳ مقاومت ۵ ولت باشد و مقاومت اول را آنی در نظر بگیریم که یک سرش به GND وصل است و مقاومت سوم هم آنی باشد که یک سرش به ۵ ولت، در این صورت بین مقاومت ۱ و ۲ ولتاژ $\frac{1}{66}$ و بین مقاومت ۲ و ۳ هم ولتاژ $\frac{3}{33}$ را خواهیم داشت.

با این تست سعی کردیم ولتاژ ورودی GPIO پین Raspberry pi را اصلاح کنیم ولی باز هم فاصله‌سنجی درست کار نکرد. علت این است که احتمالاً چون پین TRIG هم در سنسور به ۵ ولت نیاز دارد و از یکی از GPIO پین‌های Raspberry pi ورودی می‌گیرد و به آن $\frac{2}{3}$ ولت داده می‌شود نمی‌تواند پالس را به درستی بفرستد.

به این ترتیب به این نتیجه رسیدیم که با استفاده از Raspberry pi نمی‌توانیم از سنسور HY-SRF05 استفاده کنیم. با این حال چون همزمان در حال تست اتصال بلوتوث Raspberry pi به دستگاهی دیگر بودیم در ادامه به توضیح آن هم می‌پردازیم.

برای تست اتصال بلوتوث به Raspberry pi لازم بود که دو برنامه داشته باشیم. یکی از این‌ها بر روی Raspberry pi اجرا می‌شد و نقش server را داشت و دیگر بر روی لپتاپ اجرا می‌شد و نقش client را داشت. برقراری ارتباط بین دو دستگاه با استفاده از بلوتوث بسیار شبیه به سوکت‌های tcp است و به همین دلیل است که از دو جزء server و client تشکیل شده است. کد زیر مربوط به server

است که بر روی Raspberry pi اجرا می‌شود:

```

1 import bluetooth
2
3 def main():
4     server_sock=bluetooth.BluetoothSocket( bluetooth.
5         RFCOMM )
6     port = 1
7     server_sock.bind((" ",port))
8     server_sock.listen(1)
9     client_sock,address = server_sock.accept()
10    print("Accepted connection from ",address)
11    data = client_sock.recv(1024)
12    print(f"received [{data}]")
13    client_sock.close()
14    server_sock.close()
15
16 if __name__ == "__main__":
    main()
```

در این تست ما برای ارتباط client و server از پروتوكل RFCOMM استفاده کردیم که مانند پروتوكل tcp تعدادی پورت برای آن قابل تعريف است و server روی پورت شماره ۱ گوش می‌دهد و client به پورت شماره ۱ متصل می‌شود. در اینجا server بعد از برقراری اتصال پیامی از client دریافت می‌کند و آن را چاپ کرده و ارتباط را قطع می‌کند. حال به کد client توجه کنید:

```

1 import bluetooth
2
3 def main():
4     nearby_devices = bluetooth.discover_devices()
5     if not nearby_devices:
6         return
7     bd_addr = nearby_devices[0]
8     port = 1
```

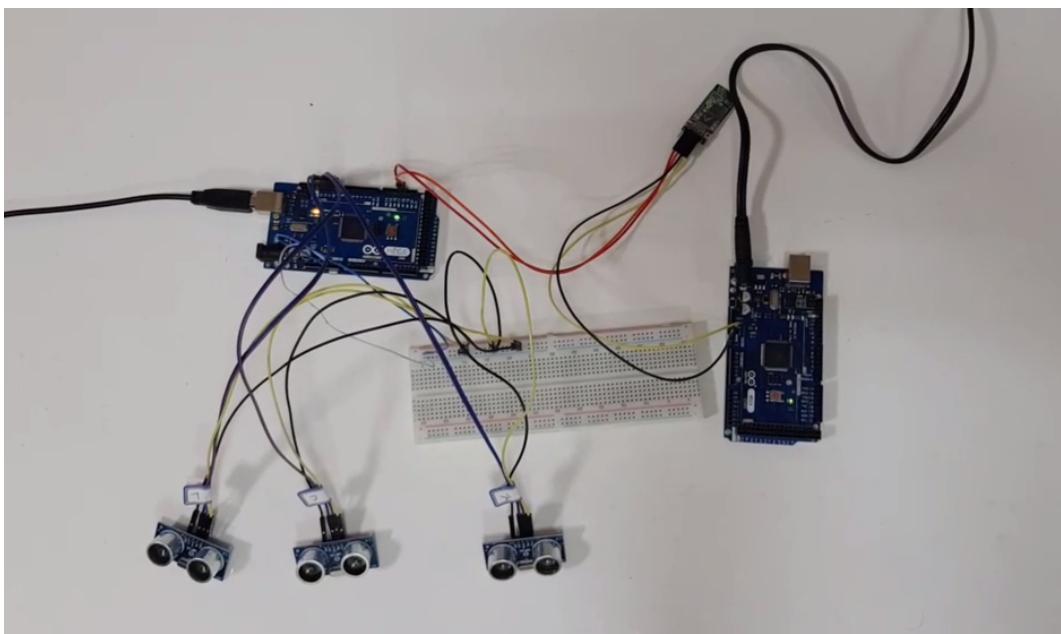
```
9  sock=bluetooth.BluetoothSocket( bluetooth.RFCOMM )  
10 sock.connect((bd_addr, port))  
11 sock.send("hello !!")  
12 sock.close()  
13  
14 if __name__ == "__main__":  
15     main()
```

در این کد ابتدا client به دنبال دستگاه‌های بلوتوث قابل کشف می‌گردد. در پروتکل بلوتوث یک دستگاه باید خود را در حالت discoverable قرار دهد تا بقیه بتوانند اطلاعات آن (MAC address) را دریافت کنند و اتصالی با آن برقرار کنند. پس از آن که یک دستگاه توسط client کشف شد به آن سوکت RFCOMM زده و داده‌ی hello!! را می‌فرستد.

فصل ۴

راهکار موفق با Arduino

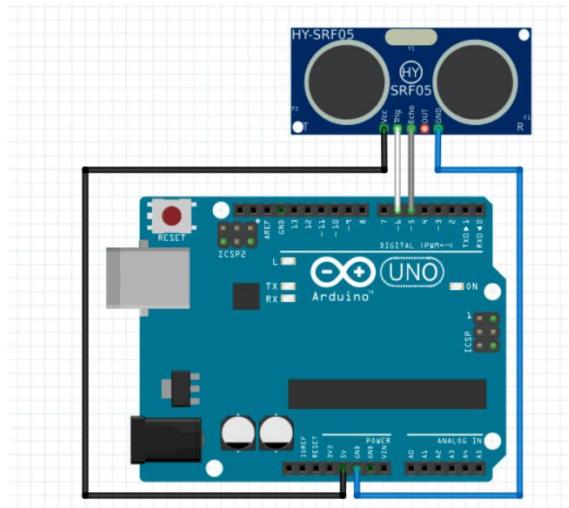
در این فصل به معرفی راهکاری که در نهایت برای اجرای پروژه استفاده کردیم می‌پردازیم. در شکل ۱-۴ می‌توانید ساخت افزار و اتصال نهایی قطعات را مشاهده کنید.



شکل ۱-۴ : مدار نهایی و اتصال قطعات به یکدیگر

پس از آنکه به دلیل ذکر شده در بخش قبل نتوانستیم با استفاده از Raspberry pi فاصله سنجی را انجام دهیم تصمیم گرفتیم با استفاده از Arduino پروژه را پیاده سازی کنیم. از آنجایی که برخلاف Raspberry pi ماژول بلوتوث ندارد از ماژول HC-05 استفاده کردیم. همچنین در ابتدا تصمیم

گرفتیم که فاصله‌سنجی را با استفاده از ۴ سنسور HY-SRF05 انجام دهیم. همانطور که در شکل ۲-۴ مشاهده می‌کنید اتصال یک سنسور HY-SRF05 به Arduino به این صورت است که GND و VCC در سنسور باید به ترتیب به GND و VCC در Arduino متصل شوند و پین‌های TRIG و ECHO هم باید به دو پین دیجیتال Arduino وصل شوند که به ترتیب input و output هستند.



شکل ۲-۴: اتصال سنسور HY-SRF05 به Arduino

برای اتصال هر ۴ سنسور به VCC موجود در Arduino از breadboard استفاده کردیم. اتصال این سنسورهای دیگر دقیقاً مانند شکل ۲-۴ بود و تنها تفاوت در پین‌های دیجیتال استفاده شده بود. با اتصال ۴ سنسور به چند مشکل خوردهیم که در ادامه به آنها می‌پردازیم.

مشکل اول مربوط به استفاده از سیم جامپر بود. هنگامی که با یک سیم نری به نری VCC را به breadboard منتقل می‌کردیم و با یک سیم جامپر دیگر که نری به مادگی بود VCC را به سنسور وصل می‌کردیم، سنسور نمی‌توانست فاصله‌سنجی کند. این در حالی بود که بدون استفاده از breadboard می‌توانستیم این کار را انجام دهیم. پس از بررسی‌های متعدد شیوه‌ی اتصال و استفاده از چند board دیگر متوجه شدیم که اشکال در استفاده از دو سیم جامپر است. هنگامی که VCC موجود در Arduino را به breadboard با استفاده از سیم مفتوحی منتقل کردیم سنسور روشن شد و توانستیم با آن فاصله‌سنجی را انجام دهیم.

مشکل دوم در تعداد بالای سنسورها نهفته بود. در آزمایش‌هایی که انجام دادیم Arduino توانست تا ۳ سنسور HY-SRF05 را همزمان روشن کند و فاصله‌سنجی را با آنها انجام دادیم ولی با اتصال سنسور چهارم دیگر هیچ یک از سنسورها روشن نبودند و مانند حالتی عمل می‌کردند که به کلی از مدار

جدا هستند. به نظر می‌رسد که Arduino توان لازم برای روشن کردن تمام ۴ سنسور را ندارد.

برای انجام پروژه از کد زیر استفاده کردیم و آن را بر روی Arduino بارگذاری کردیم:

```

1 #include <SoftwareSerial.h>
2 SoftwareSerial MyBlue(19, 18);
3
4 String msg;
5
6 const unsigned int TRIG_PIN1=13;
7 const unsigned int ECHO_PIN1=12;
8 const unsigned int TRIG_PIN2=11;
9 const unsigned int ECHO_PIN2=10;
10 const unsigned int TRIG_PIN3=9;
11 const unsigned int ECHO_PIN3=8;
12 const unsigned int BAUD_RATE=9600;
13
14 void distance_detector(unsigned int TRIG_PIN, unsigned
15   int ECHO_PIN, int sensor_num){
16   digitalWrite(TRIG_PIN, LOW);
17   delayMicroseconds(2);
18   digitalWrite(TRIG_PIN, HIGH);
19   delayMicroseconds(10);
20   digitalWrite(TRIG_PIN, LOW);
21   const unsigned long duration= pulseIn(ECHO_PIN, HIGH);
22   int distance= duration/29/2;
23   if(duration==0){
24     Serial.print("Warning: no pulse from sensor");
25     Serial.println(sensor_num);
26   }
27   else{
28     if (distance < 50) {

```

```

28     if  (sensor_num == 1){
29         MyBlue.write('L');
30         Serial.println("L");
31     }
32     if  (sensor_num == 2){
33         MyBlue.write('R');
34         Serial.println("R");
35     }
36     if  (sensor_num == 3){
37         MyBlue.write('C');
38         Serial.println("C");
39     }
40     MyBlue.write('\n');
41 }
42 Serial.print(duration);
43 Serial.print("s-sensor ");
44 Serial.print(sensor_num);
45 Serial.print(" distance to nearest object:");
46 Serial.print(distance);
47 Serial.println(" cm");
48 }
49 }

50
51 void setup() {
52     pinMode(TRIG_PIN1, OUTPUT);
53     pinMode(ECHO_PIN1, INPUT);
54     pinMode(TRIG_PIN2, OUTPUT);
55     pinMode(ECHO_PIN2, INPUT);
56     pinMode(TRIG_PIN3, OUTPUT);
57     pinMode(ECHO_PIN3, INPUT);
58     Serial.begin(BAUD_RATE);

```

```

59     MyBlue.begin(9600);
60 }
61
62 void loop() {
63     distance_detector(TRIG_PIN1, ECHO_PIN1, 1);
64     delay(200);
65     distance_detector(TRIG_PIN2, ECHO_PIN2, 2);
66     delay(200);
67     distance_detector(TRIG_PIN3, ECHO_PIN3, 3);
68     delay(200);
69 }
```

این کد مانند همان اندازه‌گیری با یک سنسور است با این تفاوت که برای سه سنسور نوشته شده است و اگر هر کدام فاصله‌ی کمتر از ۵۰ سانتی‌متر را اندازه‌گیری کرد به وسیله‌ی بلوتوث حرف متناظر با جهت خود را برای نرم‌افزار اندرویدی می‌فرستد و نرم‌افزار هم با توجه به این که کدام یک از حروف L و R و C را دریافت کرده‌است به ترتیب صدایی از طریق گوشی چپ، راست یا هر دو گوشی هندزفری پخش می‌کند.

حال با جزئیات بیشتر و قطعه قطعه مشخص می‌کنیم هر قسمت از این کد برای چه موضوعی است.

```

1 #include <SoftwareSerial.h>
2 SoftwareSerial MyBlue(19, 18);
3
4 String msg;
```

ابتدا کتابخانه SoftwareSerial.h را اضافه کردیم که در اصل کارکرد اصلی آن برای این است که بدون استفاده از پین‌های RX و TX بتوانیم ارتباط سریال برقرار کنیم (این کار به صورت نرم‌افزاری انجام می‌شود) ولی ما در اینجا برای راحت‌تر کردن کار برای ارتباط سریال توسط توابع این کتابخانه با ماژول بلوتوث از آن استفاده کردیم. سپس ماژول بلوتوث را با استفاده از این کتابخانه به صورت زیر تعریف کردیم.

```
1 SoftwareSerial(rxPin, txPin, inverse_logic);
```

در این خط، `inverse_logic` به صورت پیش فرض `false` است یعنی هر ۰ و ۱ همان معنی خود را دارد. در انتها یک متغیر `msg` تعریف شده است که در صورت لزوم بتوان با استفاده از آن پیام از طریق Bluetooth دریافت کرد.

```

1 const unsigned int TRIG_PIN1=13;
2 const unsigned int ECHO_PIN1=12;
3 const unsigned int TRIG_PIN2=11;
4 const unsigned int ECHO_PIN2=10;
5 const unsigned int TRIG_PIN3=9;
6 const unsigned int ECHO_PIN3=8;
7 const unsigned int BAUD_RATE=9600;
```

در این قطعه صرفاً پین های `TRIG` و `ECHO` برای سنسورها تعریف شده اند و همینطور نرخ ارسال اطلاعات هم تعریف شده است.

تابع `distance_detector` برای اندازه گیری فاصله استفاده شده است. ورودی های پین های `TRIG` و `ECHO` سنسور و همین طور یک عدد برای تشخیص این که این تابع برای کدام سنسور صدا زده شده است برای این تابع موجود است.

در ابتدای تابع یک پالس با بازه زمانی ۱۰ میکروثانیه به پین `Trig` داده شده است تا سنسور موج ماقوّصوت برای اندازه گیری بفرستد. سپس از تابع `pulseIn()` استفاده شده است که دو ورودی دارد که اولی یکی از پین های `ECHO` است و دومی مشخص می کند که پالس `High` یا `LOW` را تشخیص دهد. این تابع بر روی پین ورودی اول پالس مشخص شده را تشخیص می دهد و بازه زمانی آن را با واحد میکروثانیه برمی گرداند. برای تبدیل بازه زمانی به فاصله ابتدای نیاز داریم که با تقسیم بر 1000000 واحد را به ثانیه تبدیل کنیم و سپس در سرعت نور که در حدود 34100 سانتی متر بر ثانیه (341 متر بر ثانیه) است ضرب کنیم. در واقع می تونیم این کار را یکجا انجام دهیم و بازه زمانی را به صورت تقریبی بر 29 یعنی $34100 / 1000000$ تقسیم کنیم و در نهایت هم برای این که این بازه زمانی صرف مسیر رفت و برگشت شده است آن را بر 2 تقسیم کنیم. در بخش بعدی بررسی می کنیم که آیا بازه زمانی صفر بوده است یا خیر. اگر بازه زمانی صفر باشد یعنی سنسور مشکلی دارد (البته برای دقیق تر بودن این بررسی باید بینیم که طبق دیتا شیت سنسور بازه زمانی بین 100 میکروثانیه و 25 میلی ثانیه باشد). در این صورتی پیامی خروجی می دهیم تا متوجه این موضوع شویم. در غیر این صورت چک می کنیم اگر فاصله کمتر از 50 سانتی متر بود حرف متناظر با مکان سنسور را برای اپلیکیشن اندروید می فرستیم و

همین طور این حرف و فاصله و بازه زمانی برای مشاهده را خروجی می‌دهیم.

```

1 pinMode(TRIG_PIN1, OUTPUT);
2 pinMode(ECHO_PIN1, INPUT);
3 pinMode(TRIG_PIN2, OUTPUT);
4 pinMode(ECHO_PIN2, INPUT);
5 pinMode(TRIG_PIN3, OUTPUT);
6 pinMode(ECHO_PIN3, INPUT);
7 Serial.begin(BAUD_RATE);
8 MyBlue.begin(9600);

```

این تابع فقط یک بار در ابتدا در Arduino اجرا می‌شود که ما در آن پین‌های TRIG سنسورها را به عنوان خروجی و پین‌های ECHO را به عنوان ورودی تعریف کردیم همینطور نخ ارتباط و Serial هم ۹۶۰۰ قرار داده‌ایم.

```

1 distance_detector(TRIG_PIN1, ECHO_PIN1, 1);
2 delay(200);
3 distance_detector(TRIG_PIN2, ECHO_PIN2, 2);
4 delay(200);
5 distance_detector(TRIG_PIN3, ECHO_PIN3, 3);
6 delay(200);

```

این بخش اصلی کد است که همواره در Arduino در حال اجرا است که در آن ما به ترتیب با فاصله زمانی ۲۰۰ میلیثانیه فاصله موانع را اندازه‌گیری می‌کنیم.

فصل ۵

آزمایش‌ها

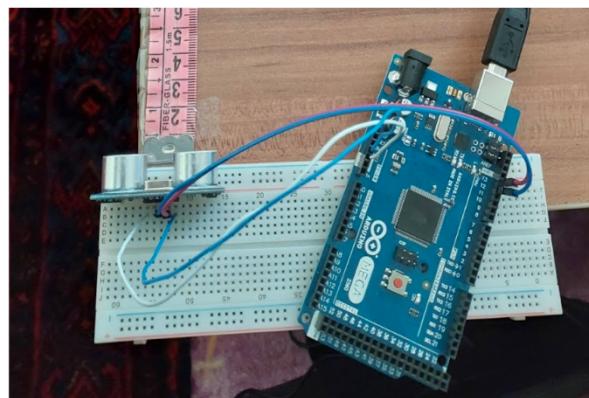
در این بخش آزمایش‌هایی را که در طول پروژه برای شناختن بهتر ویژگی‌های قطعات و یا رفع مشکلاتی که به آن برخوردیم شرح می‌دهیم.

۱-۵ آزمایش‌های Raspberry pi

آزمایش‌های مربوط به Raspberry pi در بخش راهکار ناموفق آورده شد و برای جلوگیری از تکرار، در این بخش دوباره آن را نمی‌آوریم.

۲-۵ آزمایش‌های سنسور فاصله‌سنج HY-SRF05

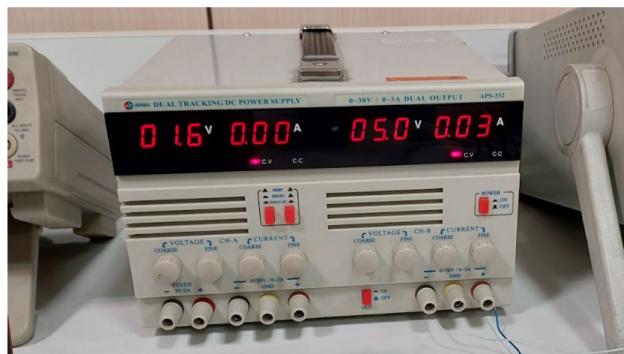
این آزمایش‌ها به منظور بررسی و اندازه‌گیری خصوصیات سنسور فاصله‌سنج هستند و در همه‌ی آن‌ها از قطعه کدی که در بخش راهکار به وسیله‌ی Arduino آورده‌یم استفاده شده است. و مدار هم به صورت شکل ۱-۵ بسته شده است.



شکل ۱-۵: اتصال سنسور HY-SRF05 به Arduino در عمل

۱-۲-۵ آزمایش حداکثر فاصله قابل اندازه‌گیری

در این آزمایش برای این‌که افت ولتاژ کمتری داشته باشیم VCC و GND سنسور را به منبع تغذیه آزمایشگاه که در شکل ۲-۵ آمده است استفاده می‌کنیم.



شکل ۲-۵: منبع تغذیه و ولتاژ و جریان آن

در اینجا از بخش ۵ ولت ثابت آن استفاده می‌کنیم و VCC را به قطب مثبت و GND را به قطب منفی متصل کرده و GND سنسور و Arduino هم به هم متصل می‌شوند. حال سنسور را به سمت دیوار می‌گیریم و آن را جابه‌جامی کنیم و به نتیجه شکل ۳-۵ می‌رسیم.

تحلیل: همان‌طور که مشاهده می‌کنیم سنسور تا فصل ۳۴۰ سانتی‌متر هم تشخیص می‌دهد پس همان‌طور که گفته شد احتمالاً حداکثر فاصله‌ای که سنسور تشخیص می‌دهد با ولتاژ ورودی آن رابطه مستقیم دارد و دلیل این‌که به هنگام اتصال به Arduino بیشتر از ۶۰ سانتی‌متر را تشخیص نمی‌دهد مربوط به افت ولتاژ ناشی از اتصال سنسور است.

```

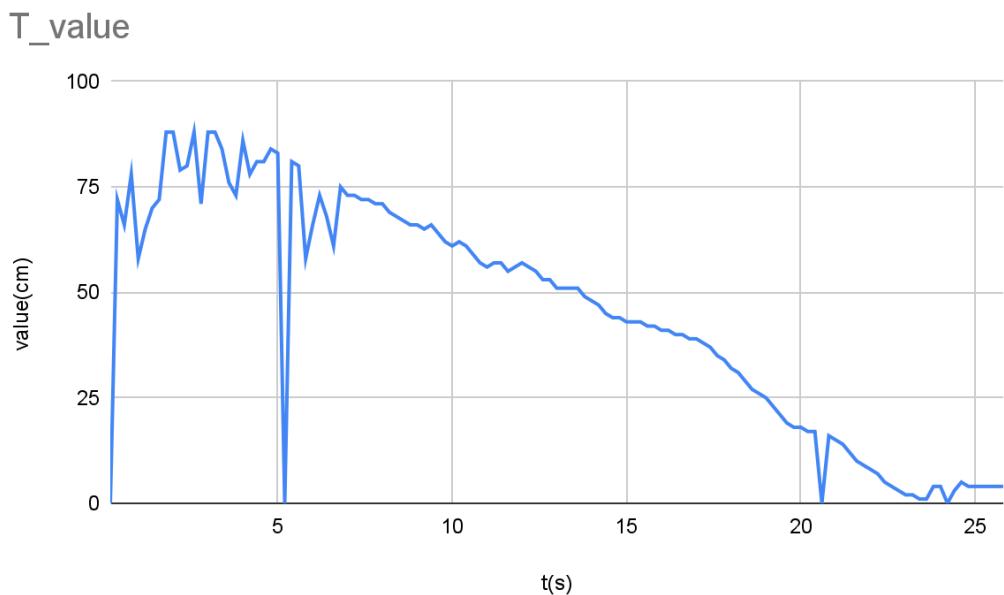
20008s-sensor 1 distance to nearest object:344 cm
20007s-sensor 1 distance to nearest object:344 cm
19955s-sensor 1 distance to nearest object:344 cm
19967s-sensor 1 distance to nearest object:344 cm
19988s-sensor 1 distance to nearest object:344 cm
19988s-sensor 1 distance to nearest object:344 cm
19937s-sensor 1 distance to nearest object:343 cm
13996s-sensor 1 distance to nearest object:241 cm
20048s-sensor 1 distance to nearest object:345 cm
19908s-sensor 1 distance to nearest object:343 cm
19932s-sensor 1 distance to nearest object:343 cm
20036s-sensor 1 distance to nearest object:345 cm
19937s-sensor 1 distance to nearest object:343 cm
19905s-sensor 1 distance to nearest object:343 cm
19909s-sensor 1 distance to nearest object:343 cm
19907s-sensor 1 distance to nearest object:343 cm
19925s-sensor 1 distance to nearest object:343 cm
19748s-sensor 1 distance to nearest object:340 cm
19964s-sensor 1 distance to nearest object:344 cm

```

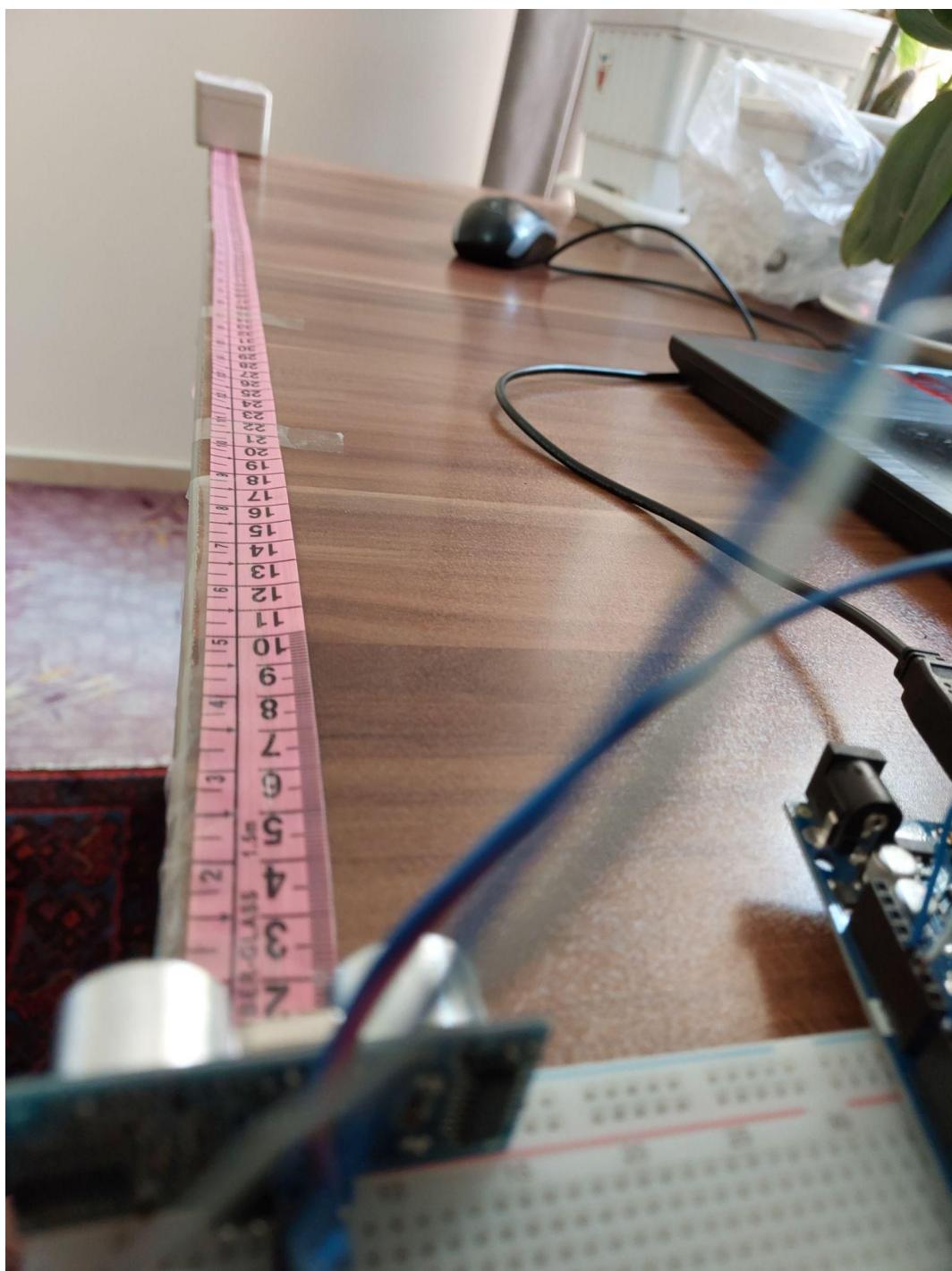
شکل ۳-۵: حداکثر برد سنسور HY-SRF05 در عمل

۲-۲-۵ آزمایش اندازه‌گیری در حالت نزدیک شدن مانع

در این آزمایش یک مانع را از فاصله به سنسور نزدیک می‌کنیم و فاصله‌ی اندازه‌گیری شده را نسبت به زمان بدست می‌آوریم. روی نمودار شکل ۴-۵ نمایش داده‌ایم.



شکل ۴-۵: رابطه‌ی زمان اندازه‌گیری شده توسط سنسور HY-SRF05 با فاصله‌ی واقعی همان‌طور که در اشکال ۵-۵ و ۶-۵ زیر مشاهده می‌کنیم از فاصله ۹۰ سانتی‌متری شروع کردیم و جسم را به سنسور نزدیک کردیم.



شکل ۵-۵: ابتدای متر



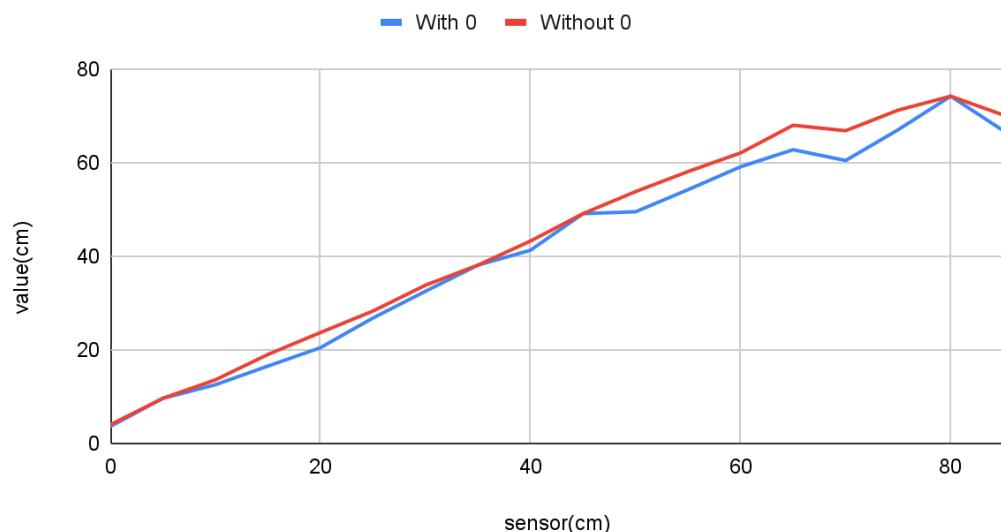
شکل ۵-۶: انتهای متر

۳-۲-۵ آزمایش اندازه‌گیری در فواصل مشخص

در این آزمایش ما در فاصله ۵ سانتی‌متری اندازه‌گیری را انجام می‌دهیم تا به نموداری از رابطه فاصله واقعی و فاصله گزارش شده برسیم. نتیجه‌ای که مشاهده می‌کنیم به صورت نمودار شکل ۷-۵ و

جدول ۱-۵ است.

value_sensor



شکل ۷-۵: رابطه‌ی فاصله‌ی واقعی با فاصله‌ی اعلام شده توسط سنسور

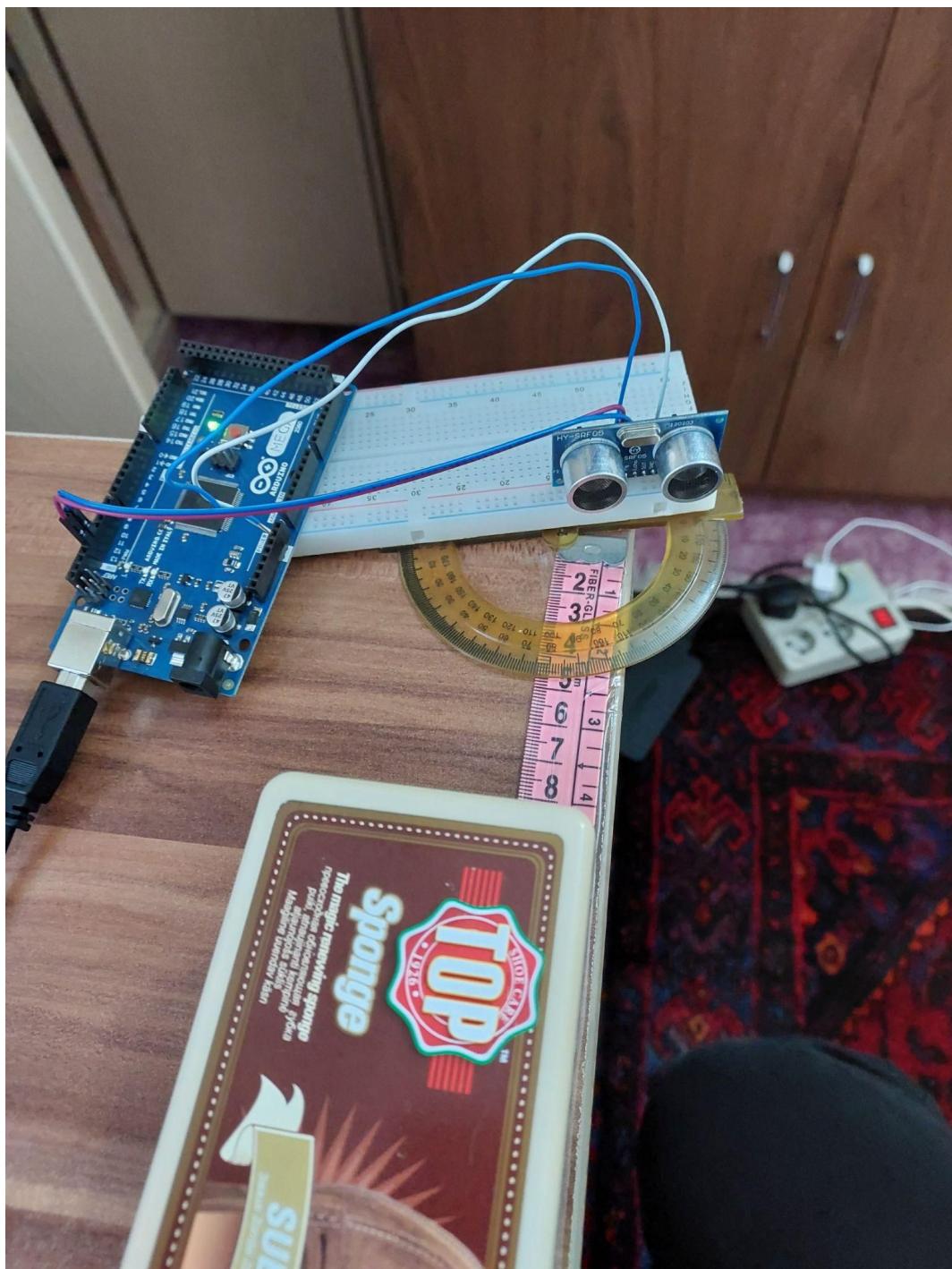
تحلیل: تقریباً ۲ سانتی‌متر خطا را داریم و بالای ۶۰ سانتی‌متر فاصله‌های اندازه‌گیری شده تقریباً بی معنی هستند.

۴-۲-۵ آزمایش اندازه‌گیری با زاویه در حالت افقی

در این آزمایش ما به دنبال محاسبه بیشینه زاویه‌ای هستیم که جسم می‌تواند نسبت به سنسور داشته باشد و همچنان اندازه‌گیری فاصله برای آن به درستی انجام شود. در شکل ۸-۵ شیوه‌ی اندازه‌گیری با استفاده از متر و نقاله را مشاهده می‌کنید.

نتیجه‌ای که بدست می‌آوریم در اشکال ۹-۵ و ۱۰-۵ و ۱۱-۵ و ۱۲-۵ و ۱۳-۵ است.

تحلیل: به طور تقریبی هر سنسور می‌تواند حداقل تا ۳۰ درجه را پوشش دهد. نتیجه‌ای که در آزمایش بدست آمده است نشان می‌دهد تقریباً با زاویه نزدیک ۱۵ درجه نسبت به یک طرف سنسور می‌توان تا فاصله ۵۰ سانتی‌متری را اندازه‌گیری کرد که اگر از دو طرف در نظر بگیریم ۳۰ درجه می‌شود.



شکل ۵-۸: تنظیم نقاله و متر و استفاده از آنها برای به دست آوردن میدان دید سنسور HY-SRF05

2966 s-sensor distance to nearest object:51 cm
 2942 s-sensor distance to nearest object:50 cm
 2942 s-sensor distance to nearest object:50 cm
 2945 s-sensor distance to nearest object:51 cm
 2945 s-sensor distance to nearest object:50 cm
 2966 s-sensor distance to nearest object:51 cm
 2942 s-sensor distance to nearest object:51 cm
 2918 s-sensor distance to nearest object:50 cm
 2942 s-sensor distance to nearest object:50 cm
 2923 s-sensor distance to nearest object:50 cm
 2942 s-sensor distance to nearest object:50 cm
 2967 s-sensor distance to nearest object:51 cm
 2948 s-sensor distance to nearest object:50 cm
 2966 s-sensor distance to nearest object:51 cm
 2942 s-sensor distance to nearest object:50 cm
 2942 s-sensor distance to nearest object:50 cm
 2942 s-sensor distance to nearest object:50 cm
 2918 s-sensor distance to nearest object:50 cm
 2918 s-sensor distance to nearest object:50 cm
 2942 s-sensor distance to nearest object:49 cm
 2942 s-sensor distance to nearest object:50 cm
 45 s-sensor distance to nearest object:0 cm
 3129 s-sensor distance to nearest object:53 cm
 3236 s-sensor distance to nearest object:55 cm
 3312 s-sensor distance to nearest object:57 cm
 3448 s-sensor distance to nearest object:59 cm
 3356 s-sensor distance to nearest object:57 cm
 3351 s-sensor distance to nearest object:57 cm
 3354 s-sensor distance to nearest object:57 cm
 3398 s-sensor distance to nearest object:58 cm
 3389 s-sensor distance to nearest object:58 cm
 3421 s-sensor distance to nearest object:59 cm
 3431 s-sensor distance to nearest object:59 cm
 3431 s-sensor distance to nearest object:59 cm
 3480 s-sensor distance to nearest object:60 cm
 3410 s-sensor distance to nearest object:58 cm
 3482 s-sensor distance to nearest object:60 cm
 45 s-sensor distance to nearest object:0 cm
 3434 s-sensor distance to nearest object:59 cm
 3434 s-sensor distance to nearest object:59 cm
 3434 s-sensor distance to nearest object:59 cm
 3404 s-sensor distance to nearest object:58 cm
 3411 s-sensor distance to nearest object:58 cm
 3459 s-sensor distance to nearest object:59 cm
 3435 s-sensor distance to nearest object:59 cm
 3457 s-sensor distance to nearest object:59 cm
 3479 s-sensor distance to nearest object:59 cm
 3428 s-sensor distance to nearest object:59 cm

شکل ۹-۵: زاویه‌ی ۵ درجه

2936 s-sensor distance to nearest object:50 cm
 2936 s-sensor distance to nearest object:50 cm
 2984 s-sensor distance to nearest object:51 cm
 2960 s-sensor distance to nearest object:51 cm
 2936 s-sensor distance to nearest object:50 cm
 2984 s-sensor distance to nearest object:51 cm
 46 s-sensor distance to nearest object:0 cm
 3026 s-sensor distance to nearest object:52 cm
 3119 s-sensor distance to nearest object:53 cm
 3140 s-sensor distance to nearest object:54 cm
 3201 s-sensor distance to nearest object:55 cm
 3299 s-sensor distance to nearest object:56 cm
 3410 s-sensor distance to nearest object:58 cm
 3387 s-sensor distance to nearest object:58 cm
 3455 s-sensor distance to nearest object:59 cm
 3455 s-sensor distance to nearest object:59 cm
 3560 s-sensor distance to nearest object:61 cm
 3584 s-sensor distance to nearest object:61 cm
 3519 s-sensor distance to nearest object:60 cm
 3593 s-sensor distance to nearest object:61 cm
 3542 s-sensor distance to nearest object:61 cm
 3542 s-sensor distance to nearest object:61 cm
 3491 s-sensor distance to nearest object:60 cm
 3543 s-sensor distance to nearest object:61 cm
 3494 s-sensor distance to nearest object:60 cm
 3489 s-sensor distance to nearest object:60 cm
 46 s-sensor distance to nearest object:0 cm
 3494 s-sensor distance to nearest object:60 cm
 3513 s-sensor distance to nearest object:60 cm
 3536 s-sensor distance to nearest object:60 cm
 3536 s-sensor distance to nearest object:60 cm
 3515 s-sensor distance to nearest object:60 cm
 3542 s-sensor distance to nearest object:61 cm
 3492 s-sensor distance to nearest object:60 cm
 3467 s-sensor distance to nearest object:59 cm
 3518 s-sensor distance to nearest object:60 cm
 3710 s-sensor distance to nearest object:63 cm
 3815 s-sensor distance to nearest object:65 cm
 3870 s-sensor distance to nearest object:66 cm
 3860 s-sensor distance to nearest object:66 cm
 3938 s-sensor distance to nearest object:67 cm
 4013 s-sensor distance to nearest object:69 cm
 4031 s-sensor distance to nearest object:69 cm
 4058 s-sensor distance to nearest object:69 cm
 4055 s-sensor distance to nearest object:69 cm
 46 s-sensor distance to nearest object:0 cm
 4037 s-sensor distance to nearest object:69 cm
 4014 s-sensor distance to nearest object:69 cm
 3987 s-sensor distance to nearest object:68 cm
 4038 s-sensor distance to nearest object:69 cm
 3900 s-sensor distance to nearest object:69 cm

شکل ۱۰-۵: زاویه‌ی ۱۰ درجه

```

1789 s-sensor distance to nearest object:30 cm
1732 s-sensor distance to nearest object:29 cm
1780 s-sensor distance to nearest object:30 cm
1887 s-sensor distance to nearest object:31 cm
1887 s-sensor distance to nearest object:30 cm
1788 s-sensor distance to nearest object:30 cm
1789 s-sensor distance to nearest object:30 cm
1783 s-sensor distance to nearest object:30 cm
1756 s-sensor distance to nearest object:30 cm
1887 s-sensor distance to nearest object:31 cm
42 s-sensor distance to nearest object:0 cm
1789 s-sensor distance to nearest object:30 cm
1756 s-sensor distance to nearest object:30 cm
43 s-sensor distance to nearest object:0 cm
1756 s-sensor distance to nearest object:30 cm
1887 s-sensor distance to nearest object:30 cm
1789 s-sensor distance to nearest object:30 cm
1783 s-sensor distance to nearest object:30 cm
1789 s-sensor distance to nearest object:30 cm
2020 s-sensor distance to nearest object:34 cm
42 s-sensor distance to nearest object:0 cm
2779 s-sensor distance to nearest object:47 cm
2932 s-sensor distance to nearest object:50 cm
3037 s-sensor distance to nearest object:52 cm
2959 s-sensor distance to nearest object:51 cm
2932 s-sensor distance to nearest object:50 cm
2958 s-sensor distance to nearest object:50 cm
3025 s-sensor distance to nearest object:52 cm
3110 s-sensor distance to nearest object:53 cm
3172 s-sensor distance to nearest object:54 cm
3106 s-sensor distance to nearest object:53 cm
2986 s-sensor distance to nearest object:51 cm
2914 s-sensor distance to nearest object:50 cm
2938 s-sensor distance to nearest object:50 cm
2986 s-sensor distance to nearest object:51 cm
3016 s-sensor distance to nearest object:51 cm
3016 s-sensor distance to nearest object:51 cm
3034 s-sensor distance to nearest object:52 cm
2959 s-sensor distance to nearest object:51 cm
2986 s-sensor distance to nearest object:51 cm
3034 s-sensor distance to nearest object:52 cm
3034 s-sensor distance to nearest object:52 cm
3242 s-sensor distance to nearest object:55 cm
3007 s-sensor distance to nearest object:51 cm
2983 s-sensor distance to nearest object:51 cm
2983 s-sensor distance to nearest object:cm
3034 s-sensor distance to nearest object:52 cm
2988 s-sensor distance to nearest object:51 cm
2962 s-sensor distance to nearest object:51 cm

```

شکل ۱۱-۵: زاویه‌ی ۱۳ درجه

```

4186 s-sensor distance to nearest object:72 cm
4444 s-sensor distance to nearest object:76 cm
4027 s-sensor distance to nearest object:69 cm
3925 s-sensor distance to nearest object:67 cm
3901 s-sensor distance to nearest object:67 cm
3923 s-sensor distance to nearest object:67 cm
4070 s-sensor distance to nearest object:70 cm
3241 s-sensor distance to nearest object:55 cm
4054 s-sensor distance to nearest object:69 cm
3193 s-sensor distance to nearest object:55 cm
2953 s-sensor distance to nearest object:50 cm
4210 s-sensor distance to nearest object:72 cm
3169 s-sensor distance to nearest object:54 cm
4093 s-sensor distance to nearest object:70 cm
4105 s-sensor distance to nearest object:70 cm
3874 s-sensor distance to nearest object:66 cm
3241 s-sensor distance to nearest object:55 cm
3166 s-sensor distance to nearest object:54 cm
3241 s-sensor distance to nearest object:55 cm
3196 s-sensor distance to nearest object:55 cm
3445 s-sensor distance to nearest object:59 cm
2908 s-sensor distance to nearest object:58 cm
3568 s-sensor distance to nearest object:61 cm
3619 s-sensor distance to nearest object:62 cm
3736 s-sensor distance to nearest object:64 cm
3574 s-sensor distance to nearest object:61 cm
3475 s-sensor distance to nearest object:59 cm
3814 s-sensor distance to nearest object:65 cm
3310 s-sensor distance to nearest object:57 cm
4030 s-sensor distance to nearest object:69 cm
3142 s-sensor distance to nearest object:54 cm
3566 s-sensor distance to nearest object:61 cm
2988 s-sensor distance to nearest object:50 cm
2857 s-sensor distance to nearest object:49 cm
43 s-sensor distance to nearest object:0 cm
3127 s-sensor distance to nearest object:53 cm
42 s-sensor distance to nearest object:0 cm
2962 s-sensor distance to nearest object:51 cm
3266 s-sensor distance to nearest object:56 cm
3421 s-sensor distance to nearest object:58 cm
3172 s-sensor distance to nearest object:54 cm
3130 s-sensor distance to nearest object:53 cm
3157 s-sensor distance to nearest object:54 cm
2930 s-sensor distance to nearest object:50 cm
2924 s-sensor distance to nearest object:50 cm

```

شکل ۱۲-۵: زاویه‌ی ۱۴ درجه

```

42 s-sensor distance to nearest object:0 cm
177 s-sensor distance to nearest object:3 cm
178 s-sensor distance to nearest object:3 cm
325 s-sensor distance to nearest object:5 cm
295 s-sensor distance to nearest object:5 cm
301 s-sensor distance to nearest object:5 cm
43 s-sensor distance to nearest object:0 cm
178 s-sensor distance to nearest object:3 cm
42 s-sensor distance to nearest object:0 cm
253 s-sensor distance to nearest object:4 cm
177 s-sensor distance to nearest object:3 cm
177 s-sensor distance to nearest object:3 cm
247 s-sensor distance to nearest object:4 cm
178 s-sensor distance to nearest object:3 cm
171 s-sensor distance to nearest object:2 cm
252 s-sensor distance to nearest object:4 cm
294 s-sensor distance to nearest object:5 cm
291 s-sensor distance to nearest object:5 cm
646 s-sensor distance to nearest object:11 cm
748 s-sensor distance to nearest object:12 cm
1285 s-sensor distance to nearest object:22 cm
768 s-sensor distance to nearest object:13 cm
2731 s-sensor distance to nearest object:47 cm
4927 s-sensor distance to nearest object:84 cm
3208 s-sensor distance to nearest object:55 cm
2876 s-sensor distance to nearest object:49 cm
4916 s-sensor distance to nearest object:84 cm
3647 s-sensor distance to nearest object:62 cm
4913 s-sensor distance to nearest object:84 cm
42 s-sensor distance to nearest object:0 cm
3527 s-sensor distance to nearest object:60 cm
3788 s-sensor distance to nearest object:65 cm
4955 s-sensor distance to nearest object:85 cm
3974 s-sensor distance to nearest object:68 cm
4624 s-sensor distance to nearest object:79 cm
4589 s-sensor distance to nearest object:79 cm
4622 s-sensor distance to nearest object:79 cm
5171 s-sensor distance to nearest object:89 cm
4616 s-sensor distance to nearest object:79 cm
4325 s-sensor distance to nearest object:74 cm
3955 s-sensor distance to nearest object:68 cm
4814 s-sensor distance to nearest object:83 cm

```

شکل ۱۳-۵: زاویه‌ی ۱۵ درجه

۵-۲-۵ آزمایش اندازه‌گیری زاویه در حالت عمودی

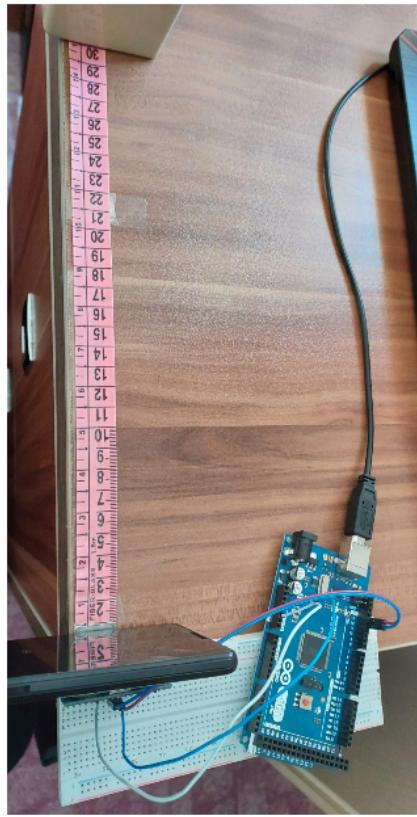
برای این آزمایش به طراحی و انجام آزمایش جدیدتری نیاز نداریم کافی است به آزمایش قبل مراجعه کنیم در همه این آزمایش‌ها مشاهده می‌شود که سنسور فقط جسم در محدوده جلوی خود را مشاهده می‌کند زیرا در غیر این صورت میز را به عنوان مانع تشخیص می‌داد و جواب‌ها کاملاً متفاوت با مقادیری که بدست آمده می‌شد.

تحلیل: از نظر عمودی سنسور فقط جلوی خود را به صورت خطی بدون زاویه خاصی مشاهده می‌کند.

۶-۲-۵ آزمایش در صورتی که یک مانع به سنسور چسبیده

در دیتاشیت آمده است که فواصل کمتر از ۲ سانتی‌متر قابل اندازه‌گیری نیست. در این آزمایش این مورد بررسی و رفتار سنسور در این مورد نشان داده می‌شود. مدار و موانع را به صورت شکل ۱۴-۵ می‌بندیم. نتیجه‌ای که مشاهده می‌کنیم به صورت شکل ۱۵-۵ است.

تحلیل: همان‌طور که مشاهده می‌کنیم در صورت نزدیکی بیش از حد جواب سنسور می‌شود بیشینه



شکل ۱۴-۵: شیوه‌ی بررسی فواید کمتر از ۲ سانتی‌متر

```

4785 s-sensor distance to nearest object:81 cm
4867 s-sensor distance to nearest object:83 cm
4308 s-sensor distance to nearest object:74 cm
4987 s-sensor distance to nearest object:85 cm
6137 s-sensor distance to nearest object:105 cm
5638 s-sensor distance to nearest object:97 cm
4680 s-sensor distance to nearest object:90 cm
42 s-sensor distance to nearest object:8 cm
3728 s-sensor distance to nearest object:64 cm
4068 s-sensor distance to nearest object:70 cm
4176 s-sensor distance to nearest object:72 cm
4185 s-sensor distance to nearest object:72 cm
4206 s-sensor distance to nearest object:72 cm
4146 s-sensor distance to nearest object:71 cm
4320 s-sensor distance to nearest object:74 cm
4060 s-sensor distance to nearest object:86 cm
5636 s-sensor distance to nearest object:87 cm
5201 s-sensor distance to nearest object:91 cm
5434 s-sensor distance to nearest object:93 cm
5290 s-sensor distance to nearest object:91 cm
5368 s-sensor distance to nearest object:92 cm
4 s-sensor distance to nearest object:0 cm
48 s-sensor distance to nearest object:0 cm
4 s-sensor distance to nearest object:0 cm
3734 s-sensor distance to nearest object:0 cm
3901 s-sensor distance to nearest object:67 cm
3668 s-sensor distance to nearest object:63 cm
4169 s-sensor distance to nearest object:71 cm
3747 s-sensor distance to nearest object:64 cm
3564 s-sensor distance to nearest object:61 cm
4914 s-sensor distance to nearest object:84 cm
5013 s-sensor distance to nearest object:86 cm
4254 s-sensor distance to nearest object:73 cm
5593 s-sensor distance to nearest object:96 cm
3941 s-sensor distance to nearest object:0 cm
5031 s-sensor distance to nearest object:86 cm
49 s-sensor distance to nearest object:0 cm
42 s-sensor distance to nearest object:0 cm

```

شکل ۱۵-۵: نتایج بررسی فواید کمتر از ۲ سانتی‌متر

مقداری که می‌تواند اندازه‌گیری کند و حتی مانع پشتی هم نمی‌بینند. البته در آزمایش‌های اندازه‌گیری در فواصل مشخص و اندازه‌گیری در حالت نزدیک شدن مانع مقادیری که در صورت نزدیک شدن زیاد مانع به سنسور به ما گزارش می‌شود اعداد کوچکی هستند. در نتیجه به جوابی که در همچنین حاتمی‌گیریم قابل اعتماد و پیش‌بینی نیست.

۳-۵ آزمایش‌های اتصال سنسورها

در این سری از آزمایش‌ها ما در تلاش هستیم تا VCC را از Arduino به BreadBoard منتقل کنیم زیرا Arduino تنها یک پین VCC که ۵ ولت است دارد پس برای این‌که تمام سنسورها و حتی ماژول بلوتوث را به این پین متصل کنیم نیاز داریم که این پین به BreadBoard منتقل شود و از آنجا به سنسورها پخش شود. همان‌طور که می‌بینیم در نهایت راه حل اتصال با سیم مفتولی است صرفاً دقت شود که اگر GND هم به BreadBoard منتقل می‌کنید از سیم مفتولی برای این کار هم استفاده کنید زیرا هر دو قرار است مثل هم مدار را جریان‌دهی کنند

۱-۳-۵ آزمایش اتصال به وسیله سیم جامپر

در این آزمایش این اتصال بین پین VCC و BreadBoard توسط سیم جامپر نری به نری انجام شده است. کد استفاده شده در این آزمایش همان کد استفاده شده برای آزمایش‌های سنسور فاصله سنج است. نتیجه همانی است که در بخش راهکار به وسیله‌ی Arduino توضیح دادیم.

تحلیل: دو نتیجه احتمالی می‌شود گرفت :

۱. VCC توان کافی برای روشن کردن سنسور را ندارد و با آوردن آن به BreadBoard مقداری از ولتاژ از دست می‌رود و سنسور کار نمی‌کند.

۲. اتصال VCC به BreadBoard مشکل دارد و به طور احتمالی این مشکل می‌تواند از سیم یا BreadBoard باشد. (مشکل BreadBoard با جابه‌جا کردن BreadBoard با یک نمونه دیگر ثابت شد که بی‌اساس است.)

برای این‌که متوجه شویم کدام یک از این مشکلات موجب این ایراد شده است دو آزمایش زیر طراحی

شده است.

۲-۳-۵ آزمایش اتصال از طریق پشت برد

برای آزمایش کردن مورد اول که توان کافی برای روشن کردن سنسور را ندارد از دو سنسور استفاده می‌کنیم و VCC یکی را با نری به مادگی به طور مستقیم به پین VCC برد Arduino متصل می‌کنیم و دیگری را از پشت برد Arduino با سیم نری به مادگی به قسمت لحیم شده پین VCC وصل می‌کنیم. برای این آزمایش از کد زیر استفاده شده است.

```

1 const unsigned int TRIG_PIN1=13;
2 const unsigned int ECHO_PIN1=12;
3 const unsigned int TRIG_PIN2=11;
4 const unsigned int ECHO_PIN2=10;
5 const unsigned int BAUD_RATE=9600;
6
7 void distance_detector(unsigned int TRIG_PIN, unsigned
8     int ECHO_PIN, int sensor_num){
9     digitalWrite(TRIG_PIN, LOW);
10    delayMicroseconds(2);
11    digitalWrite(TRIG_PIN, HIGH);
12    delayMicroseconds(10);
13    digitalWrite(TRIG_PIN, LOW);
14
15    const unsigned long duration= pulseIn(ECHO_PIN, HIGH);
16    int distance= duration/29/2;
17    if(duration==0){
18        Serial.print("Warning: no pulse from sensor");
19        Serial.println(sensor_num);
20    }
21    else{
22        Serial.print(duration);
23    }
24}
```

```

22   Serial.print("s-sensor ");
23   Serial.print(sensor_num);
24   Serial.print(" distance to nearest object:");
25   Serial.print(distance);
26   Serial.println(" cm");
27 }
28 }

29

30 void setup() {
31   pinMode(TRIG_PIN1, OUTPUT);
32   pinMode(ECHO_PIN1, INPUT);
33   pinMode(TRIG_PIN2, OUTPUT);
34   pinMode(ECHO_PIN2, INPUT);
35   Serial.begin(BAUD_RATE);
36 }

37

38 void loop() {
39   distance_detector(TRIG_PIN1, ECHO_PIN1, 1);
40   delay(200);
41   distance_detector(TRIG_PIN2, ECHO_PIN2, 2);
42   delay(200);
43 }
```

تحلیل: با اجرای این کد نتیجه‌ای که حاصل شد این بود که هر دو سنسور توانستند فاصله‌سنگی را انجام دهند و این موضوع ثابت می‌کند که Arduino توانایی روشن کردن حداقل ۲ سنسور را دارد.

۳-۳-۵ آزمایش اتصال به وسیله سیم مفتولی مسی

آزمایش را دقیقاً به همان صورت آزمایش اتصال به وسیله سیم جامپر انجام می‌دهیم فقط برای انتقال VCC از سیم مفتولی استفاده می‌کنیم. نتیجه آن بود که مانند حالت اتصال به پشت Arduino هر دو سنسور روشن شدند و با هر دو توانستیم فاصله‌سنگی کنیم.

تحلیل: این آزمایش نشان می‌دهد که مشکل از اتصالات بوده است. فقط این کار یک مشکل دیگر را به وجود آورد که آن ناپایدار بودن مدار است که به دلیل استفاده از سیم مفتوحی رخ داده است و با تکان خوردن ممکن است اتصالات از هم جدا شوند و فاصله ۰ را مشاهده کنیم.

۴-۵ آزمایش منبع تغذیه و سنسورها و ماژول بلوتوث

در این آزمایش حالت‌های مختلف اتصال VCC و GND به منبع تغذیه آزمایشگاهی و این‌که سنسورها و ماژول بلوتوث هر دو از یک منبع ولتاژ بگیرند بررسی شده است. همان‌طور که در شکل ۱۶-۵ مشاهده می‌کنید منبع روشن کردن مناسب سنسور برای اندازه‌گیری فصله ۵۰ که در این پروژه نیاز است را ندارد.

```

2527s-sensor 1 distance to nearest object:43 cm
3124s-sensor 2 distance to nearest object:53 cm
2987s-sensor 3 distance to nearest object:51 cm
3036s-sensor 1 distance to nearest object:52 cm
3392s-sensor 2 distance to nearest object:58 cm
2689s-sensor 3 distance to nearest object:46 cm
2727s-sensor 1 distance to nearest object:47 cm
2640s-sensor 2 distance to nearest object:45 cm
4513s-sensor 3 distance to nearest object:77 cm
2733s-sensor 1 distance to nearest object:47 cm
2924s-sensor 2 distance to nearest object:50 cm
3131s-sensor 3 distance to nearest object:53 cm
2691s-sensor 1 distance to nearest object:46 cm
3368s-sensor 2 distance to nearest object:58 cm
2990s-sensor 3 distance to nearest object:51 cm
3144s-sensor 1 distance to nearest object:54 cm
2745s-sensor 2 distance to nearest object:47 cm
2896s-sensor 3 distance to nearest object:49 cm
3020s-sensor 1 distance to nearest object:52 cm
3034s-sensor 2 distance to nearest object:52 cm
45s-sensor 3 distance to nearest object:0 cm
2842s-sensor 1 distance to nearest object:49 cm
2499s-sensor 2 distance to nearest object:42 cm
2900s-sensor 3 distance to nearest object:50 cm
2610s-sensor 1 distance to nearest object:45 cm
3394s-sensor 2 distance to nearest object:58 cm
2809s-sensor 3 distance to nearest object:48 cm
725s-sensor 1 distance to nearest object:12 cm
2252s-sensor 2 distance to nearest object:38 cm
1665s-sensor 3 distance to nearest object:28 cm
2568s-sensor 1 distance to nearest object:44 cm
2220s-sensor 2 distance to nearest object:38 cm
3005s-sensor 3 distance to nearest object:53 cm
2802s-sensor 1 distance to nearest object:48 cm
2479s-sensor 2 distance to nearest object:42 cm
3137s-sensor 3 distance to nearest object:54 cm
1602s-sensor 1 distance to nearest object:27 cm
2694s-sensor 2 distance to nearest object:46 cm
3038s-sensor 3 distance to nearest object:52 cm
4849s-sensor 1 distance to nearest object:83 cm
2825s-sensor 2 distance to nearest object:48 cm
3062s-sensor 3 distance to nearest object:52 cm
961s-sensor 1 distance to nearest object:16 cm
2935s-sensor 2 distance to nearest object:50 cm
3464s-sensor 3 distance to nearest object:59 cm
2895s-sensor 1 distance to nearest object:49 cm
2420s-sensor 2 distance to nearest object:41 cm
2782s-sensor 3 distance to nearest object:47 cm
2925s-sensor 1 distance to nearest object:50 cm
2854s-sensor 2 distance to nearest object:49 cm
3654s-sensor 3 distance to nearest object:63 cm
2910s-sensor 1 distance to nearest object:50 cm
2920s-sensor 2 distance to nearest object:50 cm
3275s-sensor 3 distance to nearest object:56 cm
2907s-sensor 1 distance to nearest object:50 cm
3264s-sensor 2 distance to nearest object:56 cm

```

شکل ۱۶-۵: حالت سنسور و ولتاژ متصل به قسمت عادی منبع تغذیه با ولتاژ ۵

هنگامی که سنسورها همگی به یک منبع ولتاژ وصل باشند نتایج به صورت شکل ۱۷-۵ خواهد شد. در این حالت سنسورها تقریباً در حد حالت متصل به Arduino خروجی می‌دهند ولی پایداری نداریم و ممکن است جواب‌های اشتباه هم بگیریم.

```

1028s-sensor 3 distance to nearest object:69 cm
4064s-sensor 1 distance to nearest object:70 cm
4052s-sensor 2 distance to nearest object:69 cm
4157s-sensor 3 distance to nearest object:71 cm
6096s-sensor 1 distance to nearest object:105 cm
3975s-sensor 2 distance to nearest object:68 cm
4271s-sensor 3 distance to nearest object:73 cm
3742s-sensor 1 distance to nearest object:64 cm
3566s-sensor 2 distance to nearest object:61 cm
3919s-sensor 3 distance to nearest object:67 cm
4994s-sensor 1 distance to nearest object:86 cm
3937s-sensor 2 distance to nearest object:67 cm
4382s-sensor 3 distance to nearest object:75 cm
4285s-sensor 1 distance to nearest object:73 cm
4157s-sensor 2 distance to nearest object:71 cm
3889s-sensor 3 distance to nearest object:67 cm
4722s-sensor 1 distance to nearest object:81 cm
4240s-sensor 2 distance to nearest object:73 cm
3950s-sensor 3 distance to nearest object:68 cm
4347s-sensor 1 distance to nearest object:74 cm
3288s-sensor 2 distance to nearest object:56 cm
4511s-sensor 3 distance to nearest object:77 cm
3554s-sensor 1 distance to nearest object:61 cm
4617s-sensor 2 distance to nearest object:79 cm
4237s-sensor 3 distance to nearest object:73 cm
3988s-sensor 1 distance to nearest object:68 cm
4133s-sensor 2 distance to nearest object:71 cm
4721s-sensor 3 distance to nearest object:81 cm
4222s-sensor 1 distance to nearest object:72 cm
4166s-sensor 2 distance to nearest object:71 cm
4457s-sensor 3 distance to nearest object:76 cm
4225s-sensor 1 distance to nearest object:72 cm
630s-sensor 2 distance to nearest object:10 cm
3435s-sensor 3 distance to nearest object:59 cm
3889s-sensor 1 distance to nearest object:67 cm
573s-sensor 2 distance to nearest object:9 cm
3911s-sensor 3 distance to nearest object:67 cm
4634s-sensor 1 distance to nearest object:79 cm
4022s-sensor 2 distance to nearest object:69 cm
4187s-sensor 3 distance to nearest object:72 cm
4377s-sensor 1 distance to nearest object:75 cm
3649s-sensor 2 distance to nearest object:62 cm
4372s-sensor 3 distance to nearest object:75 cm
4949s-sensor 1 distance to nearest object:85 cm
332s-sensor 2 distance to nearest object:5 cm
3959s-sensor 3 distance to nearest object:68 cm
2809s-sensor 1 distance to nearest object:48 cm
197s-sensor 2 distance to nearest object:3 cm
4231s-sensor 3 distance to nearest object:72 cm
3881s-sensor 1 distance to nearest object:66 cm
4226s-sensor 2 distance to nearest object:72 cm

```

شکل ۱۷-۵: حالت سنسور و ولتاژ متصل به قسمت ۵ ولت ثابت

حال اگر سنسور بلوتوث را به یک منبع و سنسورهای فاصله‌سنج را نیز به منبعی دیگر متصل کنیم نتایج به صورت شکل ۱۸-۵ خواهد شد. در این حالت سنسورها بهتر از حالت متصل به Arduino کار می‌کنند و پایدار هم هستند.

13744s-sensor 2 distance to nearest object:236 cm
13723s-sensor 3 distance to nearest object:236 cm
13612s-sensor 1 distance to nearest object:234 cm
13668s-sensor 2 distance to nearest object:235 cm
13647s-sensor 3 distance to nearest object:235 cm
13607s-sensor 1 distance to nearest object:234 cm
13618s-sensor 2 distance to nearest object:234 cm
13749s-sensor 3 distance to nearest object:237 cm
13687s-sensor 1 distance to nearest object:235 cm
13645s-sensor 2 distance to nearest object:235 cm
13674s-sensor 3 distance to nearest object:235 cm
13659s-sensor 1 distance to nearest object:235 cm
13644s-sensor 2 distance to nearest object:235 cm
13622s-sensor 3 distance to nearest object:234 cm
13582s-sensor 1 distance to nearest object:234 cm
13644s-sensor 2 distance to nearest object:235 cm
13647s-sensor 3 distance to nearest object:235 cm
13579s-sensor 1 distance to nearest object:234 cm
13668s-sensor 2 distance to nearest object:235 cm
13653s-sensor 3 distance to nearest object:235 cm
13612s-sensor 1 distance to nearest object:234 cm
13624s-sensor 2 distance to nearest object:234 cm
13653s-sensor 3 distance to nearest object:235 cm
13612s-sensor 1 distance to nearest object:234 cm
13624s-sensor 2 distance to nearest object:234 cm
13654s-sensor 3 distance to nearest object:235 cm
13639s-sensor 1 distance to nearest object:235 cm
13648s-sensor 2 distance to nearest object:235 cm
13653s-sensor 3 distance to nearest object:235 cm

شکل ۱۸-۵: حالت این که هر کدام به یک قسمت متفاوت منبع متصل شوند

مقدار واقعی	اندازه‌گیری سنسور بدون صفرها	اندازه‌گیری سنسور
۰	۳/۲۶	۳
۵	۴	۳/۶۳
۱۰	۹/۶۳	۹/۶۳
۱۵	۱۳/۵۸	۱۲/۵۴
۲۰	۱۹	۱۶/۵۲
۲۵	۲۳/۶۸	۲۰/۴۵
۳۰	۲۸/۲۷	۲۶/۷۹
۳۵	۳۳/۸	۳۲/۵
۴۰	۳۸/۰۹	۳۸/۰۹
۴۵	۴۳/۲۳	۴۱/۲۷
۵۰	۴۹/۰۸	۴۹/۰۸
۵۵	۵۳/۷۸	۴۹/۴۸
۶۰	۵۸/۰۷	۵۴/۲
۶۵	۶۲/۰۵	۵۹/۰۹
۷۰	۶۷/۹۵	۶۲/۷۳
۷۵	۶۶/۷۸	۶۰/۴۲
۸۰	۷۱/۱۹	۶۷
۸۵	۷۴/۱۶	۷۴/۱۵
۹۰	۷۰/۱۵	۶۶/۸۰

جدول ۱-۵: فواصل واقعی و فواصل اندازه‌گیری شده با سنسور HY-SRF05

فصل ۶

هزینه

در این فصل قیمت هر یک از قطعات استفاده شده به همراه تعدادشان و مجموع قیمت را در جدول ۱-۶ آورده‌یم.

نام ماظول	قیمت	تعداد
۱	۶۶۰	Arduino Mega
۱	۱۱۷	HC-05 (bluetooth module)
۳	۴۲	HY-SRF05 (ultrasonic sensor)
۱	۳۷	breadboard
۱	۳۶	سیم جامپر نری به نری (۴۰ تایی)
۱	۳۶	سیم جامپر نری به مادگی (۴۰ تایی)
۱	۸	سیم مفتول (۲ متر)
۱	-	Power bank
۱۰۲۱		مجموع

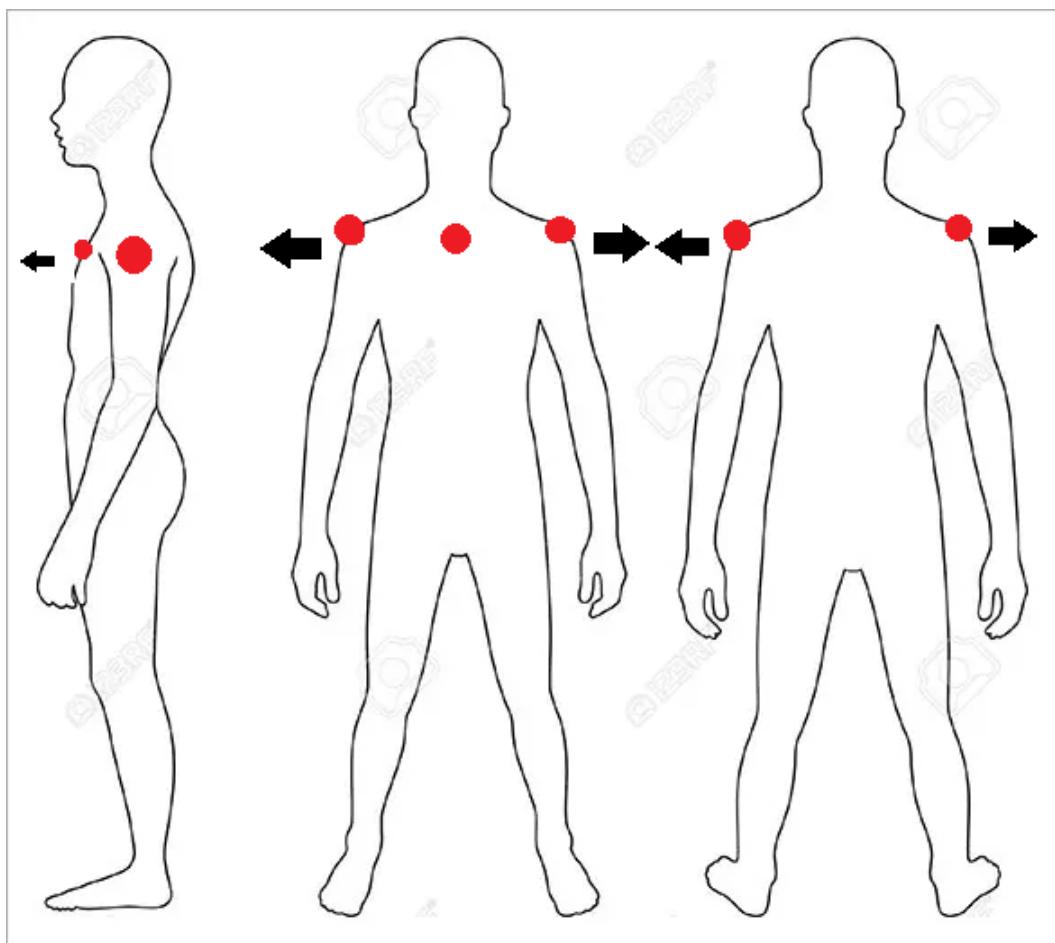
جدول ۱-۶: لیست قطعات به همراه قیمت هر کدام. قیمت‌ها به واحد هزار تومان می‌باشند.

فصل ۷

بسته‌بندی

برای این پروژه سه سنسور ultrasonic مشابه شکل ۱-۷ بر روی بدن قرار می‌گیرند. همچنین Arduino در حالی که در یک کیس قرار گرفته و Power Bank، به راحتی می‌تواند در دو جیب چپ و راست کاربر قرار بگیرند. تنها مسئله‌ای که باقی می‌ماند اتصال به وسیله‌ی سیم‌ها است. برای این کار کافی است از سیم مفتول بلند استفاده کنیم و سیم‌ها را به سنسورها و Arduino لحیم کنیم. در اندازه‌گیری انجام شده سیم‌های اتصال سنسورهای ultrasonic اگر ۸۰ سانتی‌متر باشند مناسب است. این مقدار از روی فاصله‌ی شانه‌ی چپ تا جیب راست شلوار به دست آمده است و در واقع بیشترین فاصله‌ی ممکن یک سنسور فاصله‌سنجی تا Arduino که در جیب قرار گرفته معیار ما بوده است.

برای حالت کامل که بتوان فاصله ۵۰ سانتی‌متری را تشخیص دهد حتی با فرض ساده کردن بدن به یک نقطه برای پوشش ۳۶۰ درجه‌ای حداقل نیاز به ۱۲ سنسور داریم و اگر بخواهیم جلوی پا و کمر هم رصد کنیم برای هر کدام ۱۲ تا سنسور دیگر هم اضافه می‌شود.



شکل ۱-۷: نقاط سنسورهای فاصله‌سنجی روی بدن. فلاش‌ها جهتی که سنسورها اندازه‌گیری می‌کنند را نشان می‌دهد.

فصل ۸

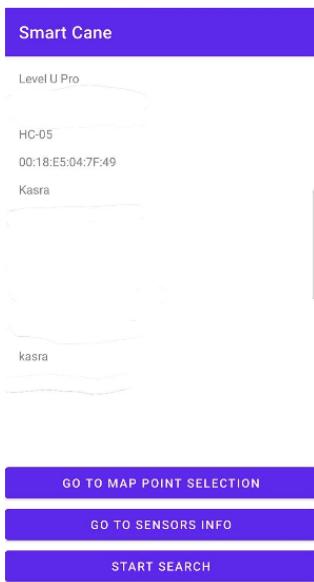
اپلیکیشن اندروید

همانطور که در ابتدا گفتیم، بخشی از پروژه مربوط به اپلیکیشن اندروید بود. وظیفه‌ی این اپلیکیشن این بود که داده‌های مربوط به تشخیص وجود جسم در جهات مختلف را دریافت کند و به کاربر هشدار صوتی بدهد. همچنین لازم بود که امکان انتخاب مقصدی از روی نقشه در این اپلیکیشن فراهم باشد و با شروع حرکت به سمت مقصد، فرمان‌های صوتی دیگری به کاربر داده شود که نشان دهد در حال نزدیک شدن به مسیر است یا خیر.

۱-۸ صفحات اپلیکیشن

اپلیکیشن اندروید طراحی شده از دو صفحه‌ی اصلی (activity) تشکیل شده است. اولین صفحه‌ی اپلیکیشن وظیفه دارد دستگاه‌های بلوتوثی که با گوشی pair شده‌اند را به کاربر نشان دهد تا کاربر مازول بلوتوث مربوط به Arduino را از بیشان انتخاب کند. با انتخاب این مازول، کاربر می‌تواند به صفحه‌ی دوم برود. در شکل ۱-۸ می‌توانید صفحه‌ی اول برنامه را مشاهده کنید.

صفحه‌ی دوم مربوط به مسیریابی است. در این صفحه نقشه‌ای به کاربر نشان داده می‌شود که می‌تواند از روی آن مقصد خود را انتخاب کند و با دکمه‌ای شروع به حرکت کند. این نقشه با استفاده از API google maps ساخته شده است. برای استفاده از این API‌ها لازم است پروژه‌ای در cloud platform google maps اساخته شود و در آن یک API key برای استفاده از سرویس Google Maps دریافت شود و در اپلیکیشن قرار داده شود. متاسفانه از آنجایی که کاربران ایرانی نمی‌توانند اطلاعات بانکی در Google



شکل ۱-۸: صفحه‌ی اول اپلیکیشن اندروید

وارد کنند امکان استفاده از اکثر سرویس‌های google را نیز ندارند و ما هم خودمان نتوانستیم API دریافت کنیم. اگرچه چنین مشکلی وجود داشت، اما در نهایت توانستیم به نوعی یک API key معتبر پیدا کنیم و برای تست اپلیکیشن و اجرای پروژه از آن استفاده کنیم.

هنگامی که کاربر شروع حرکت را با فشردن دکمه‌ای به اپلیکیشن اعلام می‌کند اتصال با ماژول بلوتوثی که کاربر پیش از این در صفحه‌ی قبل انتخاب کرده است برقرار می‌شود. هنگامی که ماژول بلوتوث کاراکترهای R و L و C را بفرستد اپلیکیشن با استفاده از API‌های Mediaplayer مخصوص اندروید بوقی را به ترتیب در گوش راست یا در گوش چپ یا در هر دو گوش پخش می‌کند. همچنین با استفاده از API‌های google maps می‌توانیم یک callback function تعريف کنیم که با تغییر مکان کاربر فراخوانی می‌شود. با استفاده از این callback function پیوسته فاصله‌ی اقلیدسی کاربر از مقصد را به صورت زیر حساب می‌کنیم.

$$distance = \sqrt{(x_{dst} - x_{user})^2 + (y_{dst} - y_{user})^2}$$

با محاسبه‌ی این فاصله بررسی می‌کنیم که آیا کاربر بیش از ۵ متر به مقصد نزدیک شده است یا از آن دور شده است. این ۵ متر دوری و نزدیکی نسبت به آخرین فاصله‌ای که از کاربر با مقصد ذخیره کردیم به دست می‌آید. اگر نزدیک شده بود بوقی موفقیت آمیز پخش می‌شود و اگر دور شده بود بوقی دارای هشدار پخش می‌شود و در هر دو صورت مکان قبلی به روزرسانی می‌شود. علاوه بر این دو مورد،

بررسی می‌شود که آیا کاربر به مقصد رسیده است یا خیر. این موضوع به این صورت بررسی می‌شود که فاصله‌ی کاربر با نقطه‌ای که علامت گذاری کرده است کمتر از ۲۰ متر شده است یا خیر. اگر کمتر از ۲۰ متر بود صوتی دیگر پخش می‌شود که پایان حرکت کاربر را نشان می‌دهد و بررسی فواصل به پایان می‌رسد تا این که باری دیگر، کاربر نقطه‌ای جدید به عنوان مقصد انتخاب کند و دوباره شروع کار را با زدن دکمه‌ی شروع به اپلیکیشن اعلام کند. برای تفهیم بهتر این روش به شبه کد زیر توجه کنید:

```

1 destination_location = // user selected point on map
2 prev_distance = 1000000000
3 function location_changed_callback(new_location){
4     distance = calculate_distance(destination_loction,
5         new_location)
6     if(prev_distance > distance + 5){ // got 5 meters
7         closer
8         play_sound(SUCCESS)
9         prev_distance = distance
10    }
11    else if (prev_distance < distance - 5){ // got 5
12        meters farther
13        play_sound(WARNING)
14        prev_distance = distance
15    }
16    if (distance < 20){
17        play_sound(REACHED_TO_DESTINATION)
18        stop_distance_change_monitoring_and_end()
19    }
20 }
```

در شکل ۲-۸ می‌توانید صفحه‌ی دوم برنامه را مشاهده کنید.



شکل ۸-۲: صفحه‌ی دوم اپلیکیشن اندروید

فصل ۹

پیشنهادات

۱. تحقیق درباره اتصال سنسور به pi Raspberry و انجام حداقلی پروژه با آن. (زیرا نمونه اش در اینترنت موجود است و باید مشکل مدلی که ما پیاده سازی کردیم را بیابید)
۲. تهیه منبع تغذیه برای سنسورها و پیاده سازی پروژه با آن.
۳. بررسی امکان پوشش بیشتر بدن با سنسور تا موضع در همه جهات تشخیص داده شوند. (بررسی توانی که مدار در این صورت دارد)
۴. تلاش برای پیاده سازی سنسورها بر روی بدن و رفع مشکل ناپایداری مدار.
۵. بهبود فرایند بررسی تغییرات فاصله از مقصد به طوری که از فاصله‌ی اقلیدسی استفاده نکنیم و به جای آن از API‌های مسیریابی استفاده کنیم و هنگام حرکت کاربر بررسی کنیم که در مسیر درست قرار دارد یا از آن منحرف شده است (به جای آنکه صرفاً فاصله‌ی دو نقطه را از یکدیگر حساب کنیم)
۶. هشدار با کم و زیاد کردن بلندی صدا نسبت به میزان نزدیکی یک مانع به فرد به جای آنکه صرفاً با قطع و وصل کردن صدا از وجود مانع فرد را مطلع کنیم. (مانند آنچه در هشدار صوتی دنده عقب ماشین‌ها می‌بینیم)

فصل ۱۰

جمع‌بندی

در این پروژه توانستیم با استفاده از Arduino و سنسورهای فاصله‌سنجی مافوق صوت و سنسور بلوتوث بخش سخت‌افزاری پروژه را پیاده‌سازی کنیم و به کاربر این امکان را دادیم که بتواند با استفاده از یک اپلیکیشن اندروید از این سخت‌افزار بهره بیرد.

مزیت این محصول همانطور که در ابتدا ذکر کردیم کمک به کاربر برای حرکت در جهت نزدیک شدن به مقصد و بررسی جهات مختلف بدون دخالت او است با این حال به دلیل برخی از محدودیت‌ها میدان دید صدرصد کافی برای یک شناسایی دقیق موضع فراهم نمی‌شود و جای بهبود دارد.

در این پروژه مهمترین چالش، تغذیه‌ی کل قطعات بود به نحوی که همگی هم روشن شوند و هم به میزانی به آنها توان برسد که بتوانند از حداکثر قابلیت خود استفاده کنند. این چالش را تا حدی توانستیم با روش‌هایی که در بخش‌های قبل ذکر کردیم برطرف کنیم. با این حال همانطور که در بخش پیشنهادات ذکر شد می‌توانیم با بهبود منبع تغذیه، به قابلیت‌های بیشتری مانند افزایش برد سنسورها و افزایش میدان دید با استفاده از افزایش تعداد سنسورها دست پیدا کنیم.

در نهایت معتقدیم این محصول را می‌توان به چشم یک prototype از محصول نهایی دید و در جهت توسعه‌ی بیشتر و تکمیل آن برای رسیدن به محصولی قابل عرضه به بازار کوشید.