Lidar Based System for Elderly Monitoring in Assisted Living Homes

Bachelor project 2019: Malekzai Bashir Ahmad

Acknowledgement

I would like to thank all who contributed to this project with time and knowledge and experience. A special thanks goes to the following amazing people: Nikolai Frøvik my coordinator and one of my supervisors – for helping me all the time and working with me in doing all the experiments in this project, and for giving me ideas to think about and discussing it in philosophical way. I want to thank my main supervisor Knut Øvsthus for giving me this project, a great teacher that I always would remember, specially his patience and understanding.

I would also give my thanks to a good friend of mine Alejandro Lliso Cosin, an exchange student at HVL (høgskolen på Vestlandet), from the university of Valencia in Spain for helping me in making the server program of this project, an excellent programmer and a great guy.

And lastly it's been a long journey that had some ups and downs in term of learning, but a great experience and the most knowledgeable time I have ever had in my whole 3 years at the university.

Abstract

Elderly population is on the rise in Norway and developing countries. This sector of the population has many health issues due to age related problems, like decline in functional activity and injuries sustained due to fall, trauma or fractures. To minimize these kind of risks we have worked in this project on a distance sensor [LIDAR] that could detect a person’s movement in a distance from a 40 mm to 4 m. This type of distance sensor (LIDAR) can be installed in different parts of the home to detect the movement of the person.

Three experiments performed in this project to verify three parts in this sensor, the distance that the sensor can actually detects a person’s movement. The second one is the time delay between the server that collects the data from the sensor, and the last experiment is to find out the actual angle view of the sensor. All these three experiments performed because we wanted to know if there is a use for these kinds of distance sensors in measuring the movement of person.

From the results of the experiments, we found out that the sensor has better range of detection from 500mm (millimeters) or 50 cm distance to 3500mm or 350 cm (centimeters) distance. The time delay between the sensor and the server is approximately between 100ms (millisecond) and 120ms (millisecond) or 1/10 of a second, which is not that significant delay. In addition, I found out that the angle view or the field of view of the sensor is better at 2000mm or 2m (meters) distance. This means that the sensor angle view or field of view is more correct at 2000mm (millimeters) or 2m (meters) distance.

The analysis and the experiments in this project could be used as a tool for future use of the sensor. The product is at the beginning stages, so it is obvious that more work is needed in order to make the distance sensor more reliable for costumer use.

Table of Contents

1 Introduction

Elderly population is on the rise in Norway and developed countries. In 2002 there were 610000 Thousand elderly over 67 years old, and this will rise to 1.4 million in 2050. It is the fastest growing part of the population not only Norway but in the developed countries [1]. This sector of the population has many health issues due to age related problems, like, decline in functional activity and injuries sustained due to fall, trauma or fractures [2].

There has been a project that have been done between the Western University of Norway for Applied Sciences and the Warsaw University of Technology by the name of (RADCARE) that has worked on a care technology that could help the elderly at homes. This technology has been already used and tested in Norway [3].These new technologies will help the elderly population to live at home independently in the future. This could reduce the cost and reduce hospital visits

The modern monitoring system for elderly folks in homes helps with other health condition, like diabetes, heart failure or other health issues. Among these health issues falls are the biggest reason that elderly folk is hospitalized for or have a sudden death [4]. Fall has been the main reason for hospitalization for a long period; it is costly for both public sector and the private sector. To reduce these costs there has been a variety of different sensing technology in recent years that helps monitoring health condition in elderly living at home, for example wireless wristband that has a help button that alerts the staff in case of an emergency or GPS system that monitors the elderly motion at a care institution.

Sensing technologies can be categorized in two parts (wearable and non-wearable), wearable parts are those that are attach to the body or cloth, like smart watches and non-wearable are infrared sensors are cameras that are installed at homes, but there are some privacy issues regarding these systems. Privacy issues are wide range of issues, such as identity thefts of the concerned person, because the system that monitors these individuals are most of the time connected to a server that collects data, pictures, names and addresses of the individual.

Our focus in this project is to have a more reliable and effective sensing system that protect the individual’s privacy and give as accurate data regarding movement and health of the individual living at home. The data that we are collecting from the sensor will improve our understanding to monitor the elderly at home more effectively. On the other hand, it will reduce the risk of sudden death due to fall at elderly homes.

In this framework, the Western University of Norway faculty of Electrical Engineering has come up with a prototype of a Lidar sensor system that insures more reliable and effective sensing system in elderly homes.

1.1 Goal of the project

The goal of the project is to find out the actual distance measurement of the sensor, the time delay between the sensor and the server that collects data from the sensor, and lastly finding out the angle view or the field of view of the sensor. By angle view, we mean how much area does the sensor cover.

Collected data through the experiments are compared with the description of the sensor in the data sheet of the manufacturer. The data that we will collect from this project could benefit future work on this sensor.

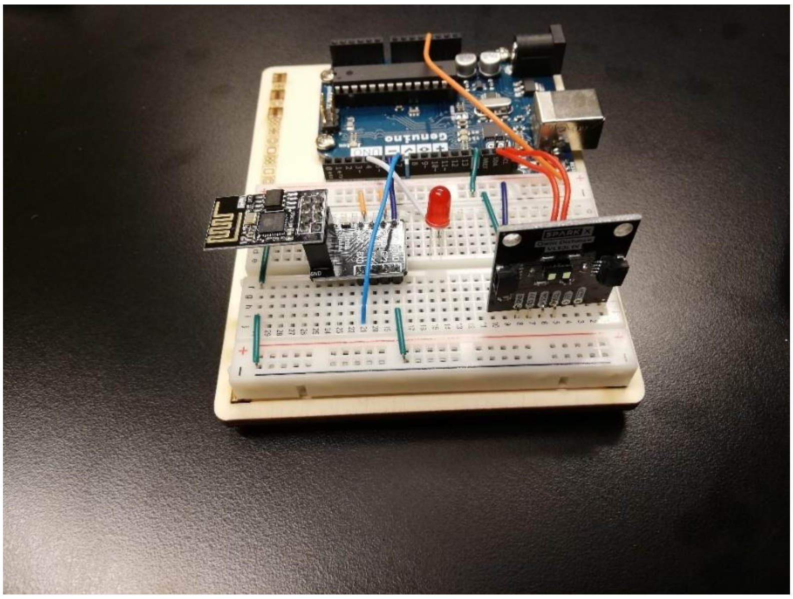


Figure 1 shows the actual sensor installed on an Arduino board with a Wi-Fi chip

2 Method

In this chapter, I will discuss some theory behind Time of flight principles and the distance sensor (VL53L1X) that we are using in this project.

2.1 Principle of Time of Flight (ToF)

Time of flight is a method for measuring distance between an object and a sensor. This measurement calculated by taking the difference between the time the sensor sending the signal to the object and its return to the sensor. The signal can be either a light or a sound.

Distance Sensors based on the principle of Time of flight measures the time that it takes for a light wave to travel from the emitter to the object and then back to the sensor.

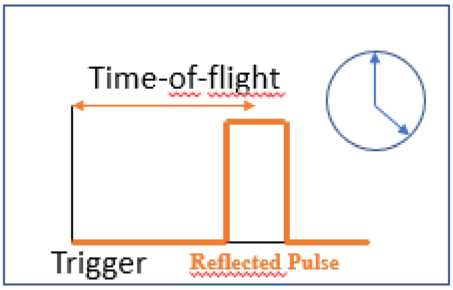


Figure 2 shows the principle of pulse reflected light from the object

We can measure this distance by this formula:

$$D = \frac{C*T}{2}$$

D = the distance from the sensor to the object

C= speed of light (light speed =$3x10^8 \ \frac{m}{s}$)

T = the time it takes for the pulse to come back

The time here is divided by 2 because the light must travel to the object and return to the sensor back.

2.2 Distance Sensor (VL53L1X) Overview

The information in this chapter is from the manufacturer website and data sheet [5].

According to the manufacturer own data sheet for the sensor, the range of the sensor is from 40mm to 4m. It uses a laser that emits light towards the object and then calculate the distance it took to reflect the light back to the sensor. [5]

According to the datasheet, the distance sensor has a frequency up to 50 Hz. The distance sensor (VL53L1X) is programmable through Arduino board. Although this sensor has a narrow field of view, but it has a faster update rate and longer range. Size of the distance sensor is 4.9 x 2.5 x 1.56m^3, and the operational voltage is between 2.6 V to 3.5 V, its infrared emitter light has 940 nm range. [5]

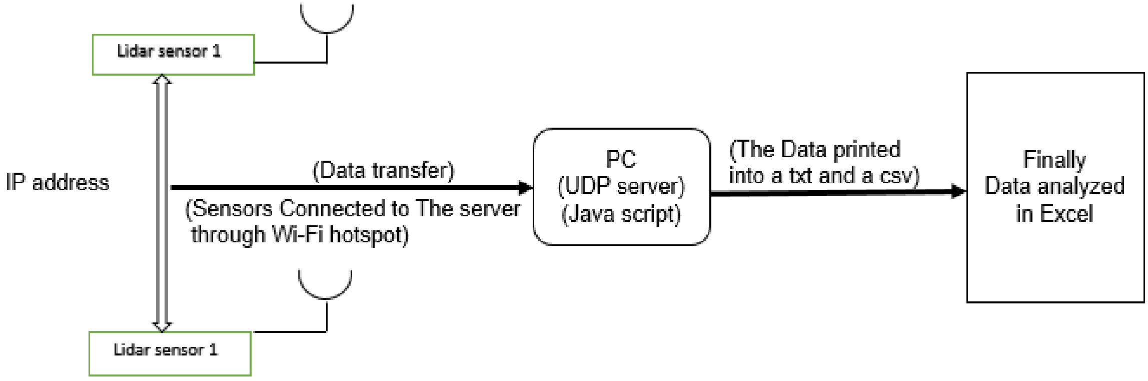
The distance sensor has three mode for distance measurements, short, medium and long. [5]

We can use the distance sensor (VL53L1X) in many different areas:

- can be used in drones
- Vending machines
- Smart homes
- Robots

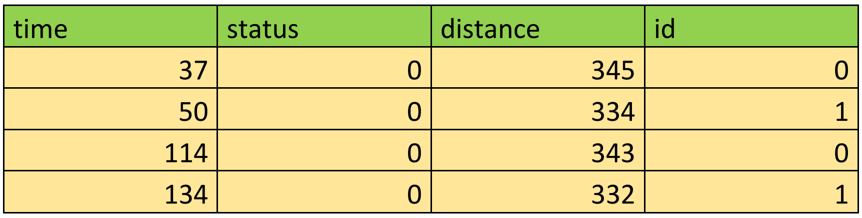
2.3 Physical Design (System design)

In the physical design, I will draw the physical output of the system and I will discuss the activity of each part in the system. Down in the block diagram, we can see the flow of the data from the sensors through the UDP server inside the PC, the last part is where data is printed in two formats a txt file and a csv file.


Lidar sensor 1
IP address
(Data transfer)
(Sensors Connected to The server through Wi-Fi hotspot)
PC
(UDP server)
(Java script)
(The Data printed into a txt and a csv)
Finally Data analyzed in Excel
Lidar sensor 1

Block diagram showing the system design of the project

Both of the sensors detect the movement of a person and then sends the data to the server. From the server the data is printed out in a separate file as a txt file or a csv file. At the end, the data is analyzed in Excel program. An example of the data printed in a csv format in Excel program shown as follow.



time	status	distance	id
37	0	345	0
50	0	334	1
114	0	343	0
134	0	332	1

Table 1 shows the data printed out of the server in a csv format inside Excel program

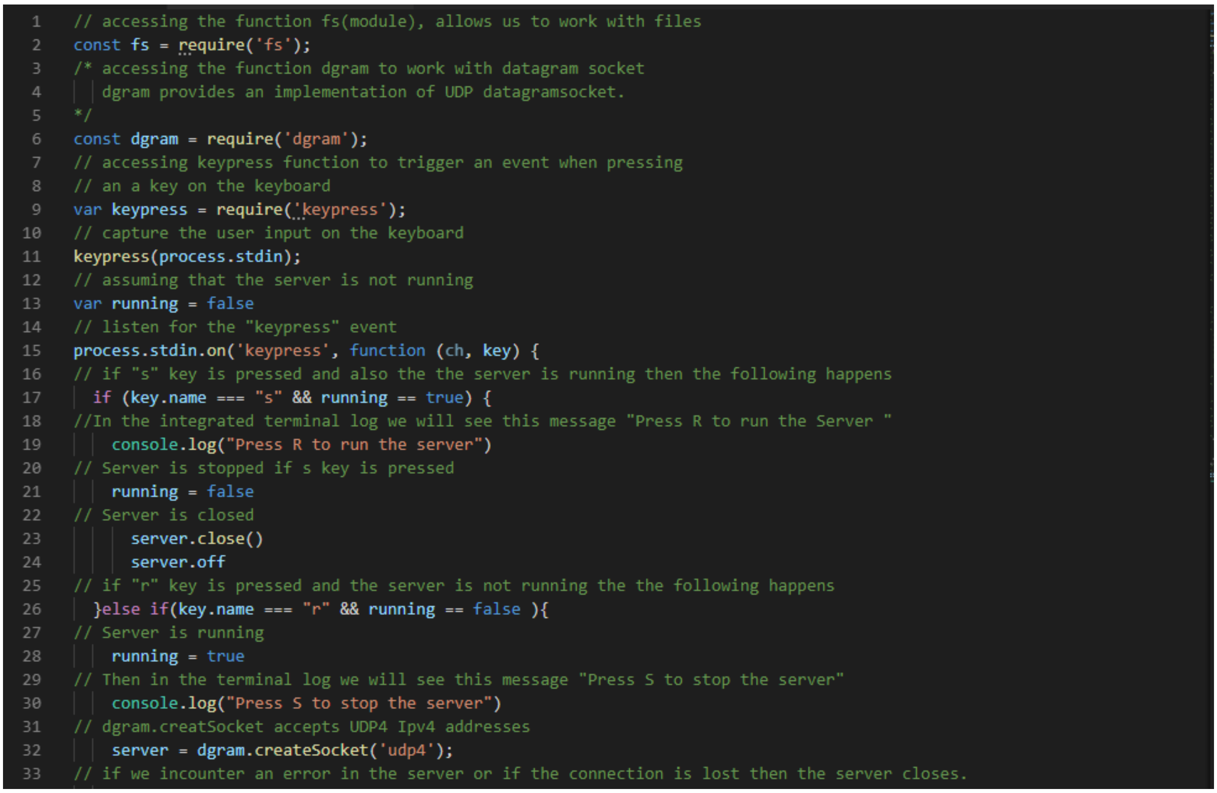
One thing to mention here is that the time is registered in millisecond (ms) unit, and status shows if the sensor is registering any data or if it is out of range. The distance data is registered in millimeter (mm) unit. I have divided both sensors in id format, and I named them sensor 1 as id 0 and sensor 2 as id 1.

2.4 Implementation (code in the server)

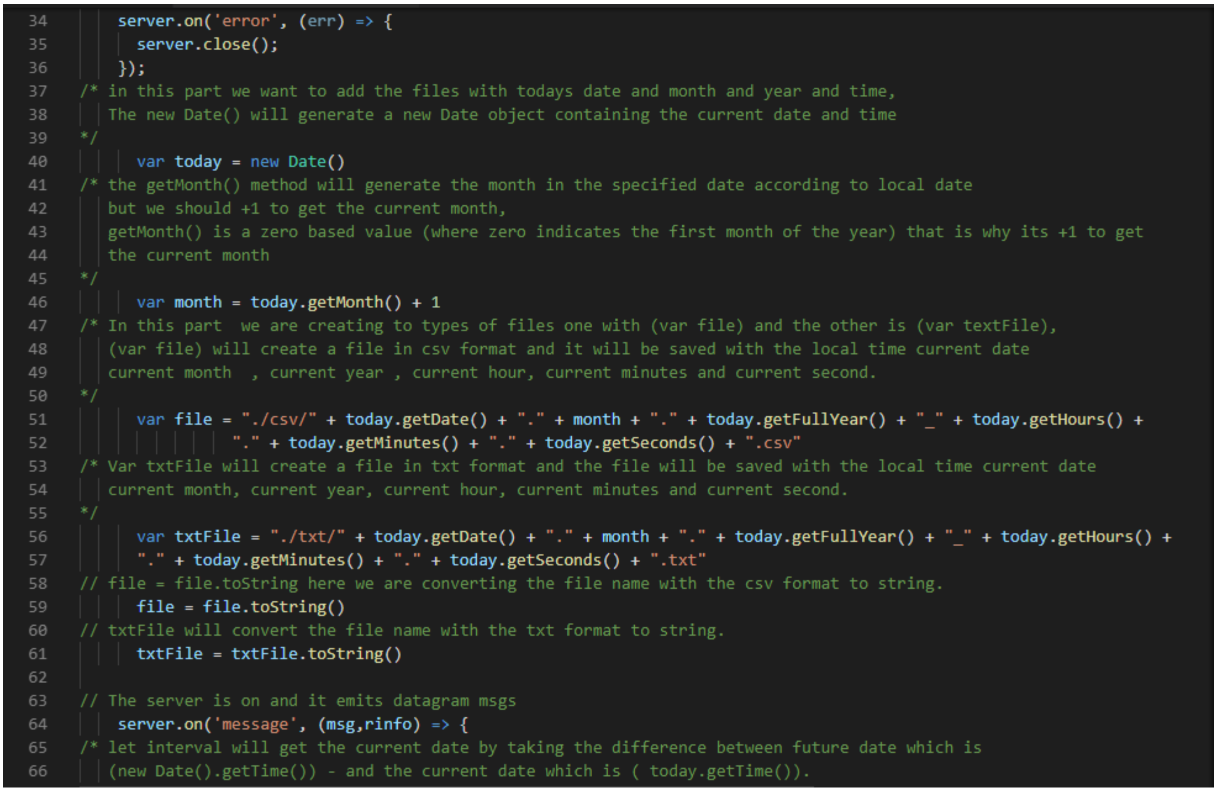
In this part, we made a server that could collect data from the sensor. I will print out the whole server program here and I will explain every section. I have made the server program in Java script, and I got help with making the server from Alejandro Lliso Cosin, an exchange student at HVL.

To make the Server program we used JavaScript platform for it, because it is easier to implement the server program. To write the code for the Server program I downloaded Visual studio code editor to write the JavaScript code.

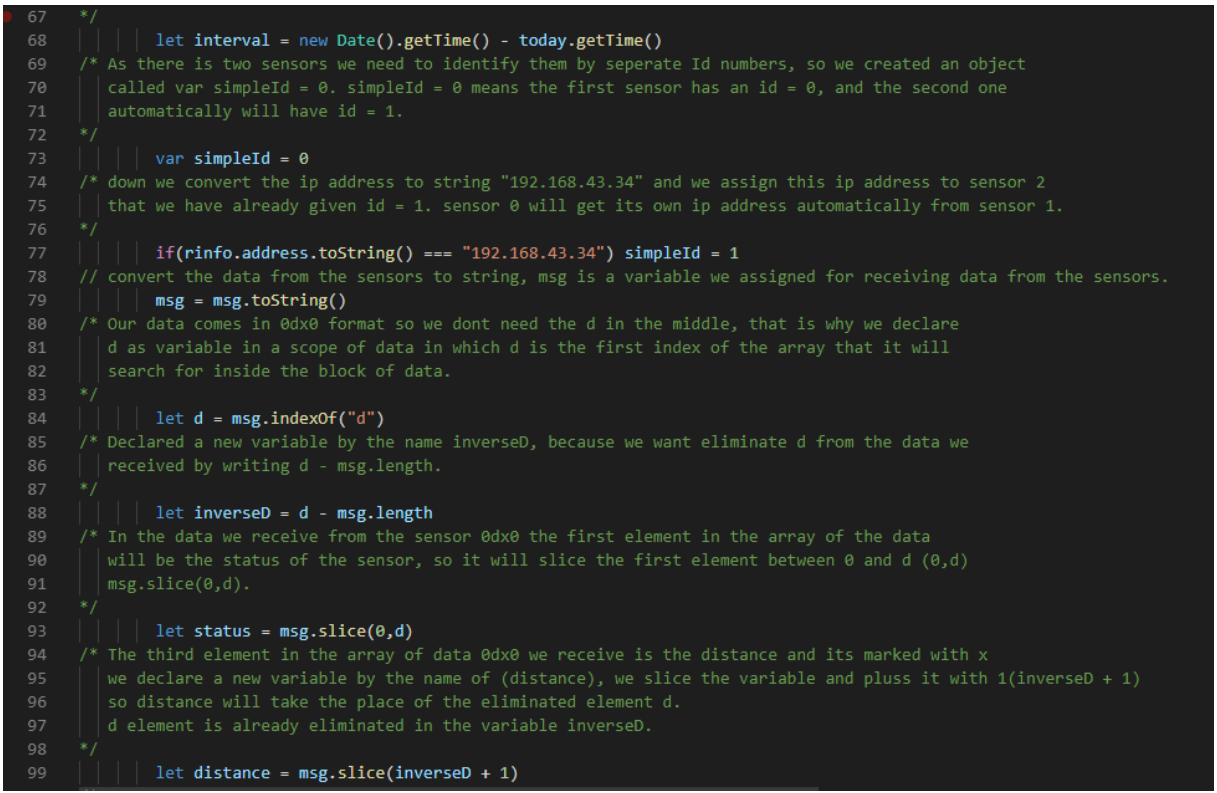
Down I will put the server code with an explanation comments on every line code inside the server. The comments are in green color.



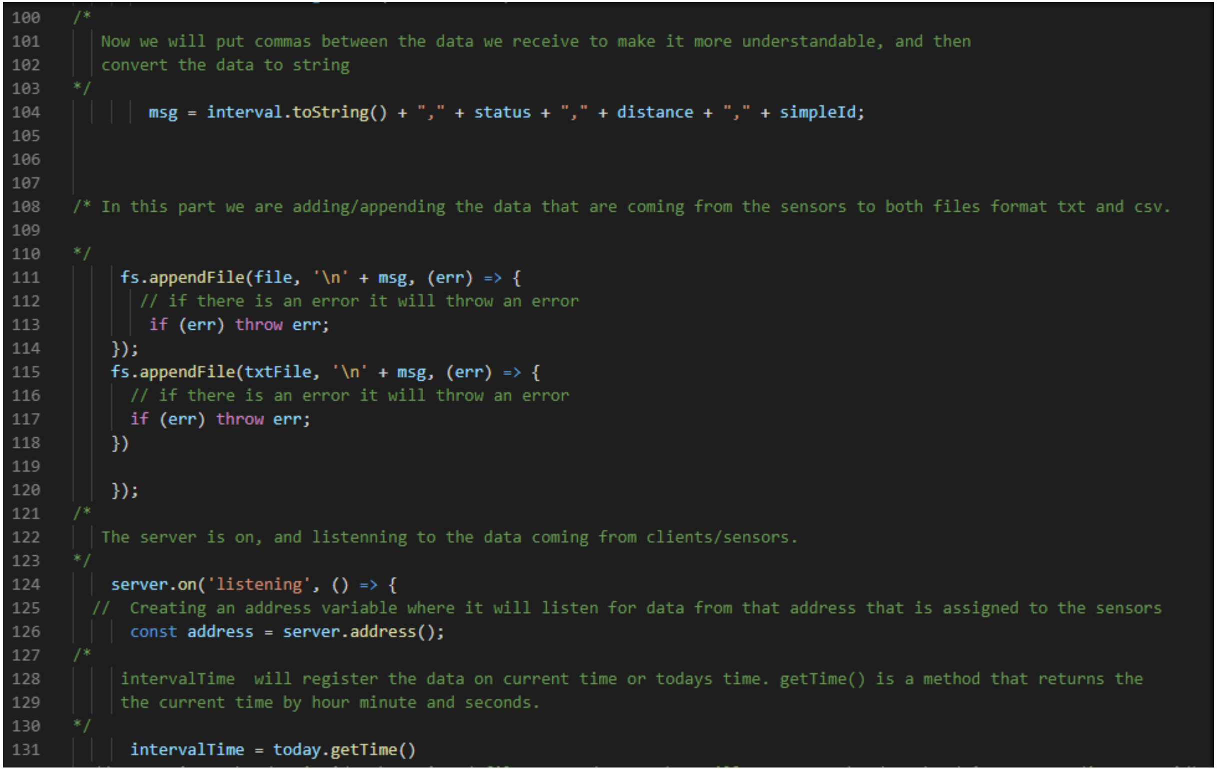
```
// accessing the function fs(module), allows us to work with files
const fs = require('fs');
/* accessing the function dgram to work with datagram socket
    dgram provides an implementation of UDP datagramsocket.
*/
const dgram = require('dgram');
// accessing keypress function to trigger an event when pressing
// an a key on the keyboard
var keypress = require('keypress');
// capture the user input on the keyboard
keypress(process.stdin);
// assuming that the server is not running
var running = false
// listen for the "keypress" event
process.stdin.on('keypress', function (ch, key) {
// if "s" key is pressed and also the the server is running then the following happens
  if (key.name === "s" && running == true) {
//In the integrated terminal log we will see this message "Press R to run the Server "
    console.log("Press R to run the server")
// Server is stopped if s key is pressed
    running = false
// Server is closed
      server.close()
      server.off
// if "r" key is pressed and the server is not running the the following happens
  }else if(key.name === "r" && running == false ){
// Server is running
    running = true
// Then in the terminal log we will see this message "Press S to stop the server"
    console.log("Press S to stop the server")
// dgram.creatSocket accepts UDP4 Ipv4 addresses
    server = dgram.createSocket('udp4');
// if we incounter an error in the server or if the connection is lost then the server closes.
```



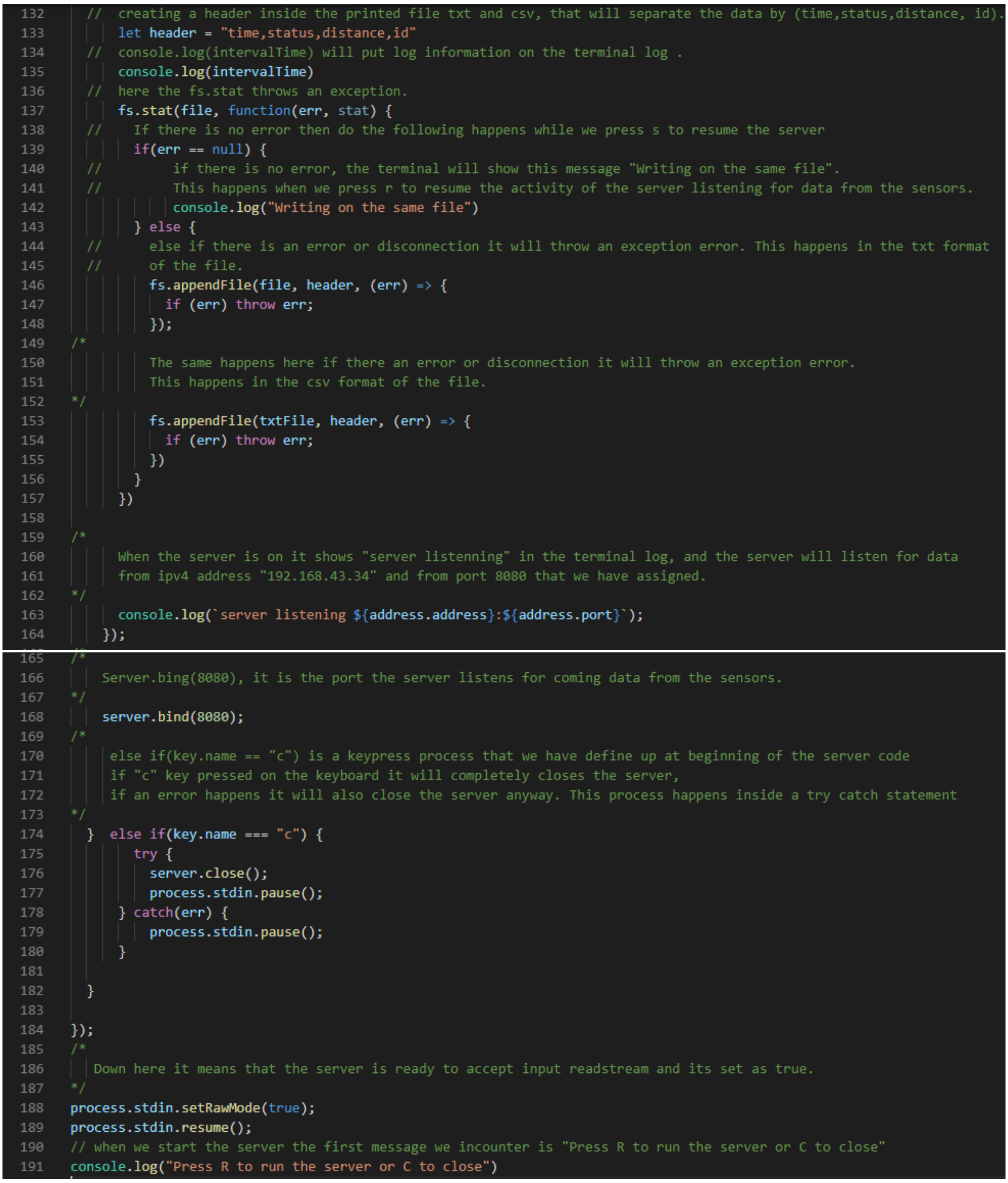
```
    server.on('error', (err) => {
      server.close();
    });
/* in this part we want to add the files with todays date and month and year and time,
    The new Date() will generate a new Date object containing the current date and time
*/
      var today = new Date()
/* the getMonth() method will generate the month in the specified date according to local date
    but we should +1 to get the current month,
    getMonth() is a zero based value (where zero indicates the first month of the year) that is why its +1 to get
    the current month
*/
      var month = today.getMonth() + 1
/* In this part  we are creating to types of files one with (var file) and the other is (var textFile),
    (var file) will create a file in csv format and it will be saved with the local time current date
    current month  , current year , current hour, current minutes and current second.
*/
      var file = "./csv/" + today.getDate() + "." + month + "." + today.getFullYear() + "_" + today.getHours() +
                "." + today.getMinutes() + "." + today.getSeconds() + ".csv"
/* Var txtFile will create a file in txt format and the file will be saved with the local time current date
    current month, current year, current hour, current minutes and current second.
*/
      var txtFile = "./txt/" + today.getDate() + "." + month + "." + today.getFullYear() + "_" + today.getHours() +
      "." + today.getMinutes() + "." + today.getSeconds() + ".txt"
// file = file.toString here we are converting the file name with the csv format to string.
      file = file.toString()
// txtFile will convert the file name with the txt format to string.
      txtFile = txtFile.toString()

// The server is on and it emits datagram msgs
    server.on('message', (msg,rinfo) => {
/* let interval will get the current date by taking the difference between future date which is
    (new Date().getTime()) - and the current date which is ( today.getTime()).
```



```
*/
        let interval = new Date().getTime() - today.getTime()
/* As there is two sensors we need to identify them by seperate Id numbers, so we created an object
  called var simpleId = 0. simpleId = 0 means the first sensor has an id = 0, and the second one
  automatically will have id = 1.
*/
        var simpleId = 0
/* down we convert the ip address to string "192.168.43.34" and we assign this ip address to sensor 2
  that we have already given id = 1. sensor 0 will get its own ip address automatically from sensor 1.
*/
        if(rinfo.address.toString() === "192.168.43.34") simpleId = 1
// convert the data from the sensors to string, msg is a variable we assigned for receiving data from the sensors.
        msg = msg.toString()
/* Our data comes in 0dx0 format so we dont need the d in the middle, that is why we declare
  d as variable in a scope of data in which d is the first index of the array that it will
  search for inside the block of data.
*/
        let d = msg.indexOf("d")
/* Declared a new variable by the name inverseD, because we want eliminate d from the data we
  received by writing d - msg.length.
*/
        let inverseD = d - msg.length
/* In the data we receive from the sensor 0dx0 the first element in the array of the data
  will be the status of the sensor, so it will slice the first element between 0 and d (0,d)
  msg.slice(0,d).
*/
        let status = msg.slice(0,d)
/* The third element in the array of data 0dx0 we receive is the distance and its marked with x
  we declare a new variable by the name of (distance), we slice the variable and pluss it with 1(inverseD + 1)
  so distance will take the place of the eliminated element d.
  d element is already eliminated in the variable inverseD.
*/
        let distance = msg.slice(inverseD + 1)
```



```
/*
  Now we will put commas between the data we receive to make it more understandable, and then
  convert the data to string
*/
        msg = interval.toString() + "," + status + "," + distance + "," + simpleId;



/* In this part we are adding/appending the data that are coming from the sensors to both files format txt and csv.

*/
      fs.appendFile(file, '\n' + msg, (err) => {
        // if there is an error it will throw an error
         if (err) throw err;
     });
     fs.appendFile(txtFile, '\n' + msg, (err) => {
       // if there is an error it will throw an error
       if (err) throw err;
     })

     });
/*
  The server is on, and listenning to the data coming from clients/sensors.
*/
     server.on('listening', () => {
  //  Creating an address variable where it will listen for data from that address that is assigned to the sensors
       const address = server.address();
/*
      intervalTime  will register the data on current time or todays time. getTime() is a method that returns the
      the current time by hour minute and seconds.
*/
       intervalTime = today.getTime()
```



```
132    //  creating a header inside the printed file txt and csv, that will separate the data by (time,status,distance, id).
133        let header = "time,status,distance,id"
134    //  console.log(intervalTime) will put log information on the terminal log .
135        console.log(intervalTime)
136    //  here the fs.stat throws an exception.
137        fs.stat(file, function(err, stat) {
138    //    If there is no error then do the following happens while we press s to resume the server
139          if(err == null) {
140    //         if there is no error, the terminal will show this message "Writing on the same file".
141    //         This happens when we press r to resume the activity of the server listening for data from the sensors.
142               console.log("Writing on the same file")
143          } else {
144    //       else if there is an error or disconnection it will throw an exception error. This happens in the txt format
145    //       of the file.
146            fs.appendFile(file, header, (err) => {
147              if (err) throw err;
148            });
149   /*
150            The same happens here if there an error or disconnection it will throw an exception error.
151            This happens in the csv format of the file.
152   */
153            fs.appendFile(txtFile, header, (err) => {
154              if (err) throw err;
155            })
156          }
157        })
158
159   /*
160        When the server is on it shows "server listenning" in the terminal log, and the server will listen for data
161        from ipv4 address "192.168.43.34" and from port 8080 that we have assigned.
162   */
163        console.log(`server listening ${address.address}:${address.port}`);
164      });
165   /*
166       Server.bing(8080), it is the port the server listens for coming data from the sensors.
167   */
168       server.bind(8080);
169   /*
170        else if(key.name == "c") is a keypress process that we have define up at beginning of the server code
171        if "c" key pressed on the keyboard it will completely closes the server,
172        if an error happens it will also close the server anyway. This process happens inside a try catch statement
173   */
174     }  else if(key.name === "c") {
175          try {
176            server.close();
177            process.stdin.pause();
178          } catch(err) {
179            process.stdin.pause();
180          }
181
182     }
183
184   });
185   /*
186      Down here it means that the server is ready to accept input readstream and its set as true.
187   */
188   process.stdin.setRawMode(true);
189   process.stdin.resume();
190   // when we start the server the first message we incounter is "Press R to run the server or C to close"
191   console.log("Press R to run the server or C to close")
```

Figure 3 shows the server code with explanation by green color comments.

Here i will explain some parts that need a little bit of explaination the other parts are already explained by comments in the code.

There are 191 line numbers in this code I will name the line number to be more specific which part I am referring to.

On line number 18 I have reffered to an integrated terminal log, the terminal log is located under the code, by clicking Terminal in visual studio code and click on New Terminal, Terminal log opens up.

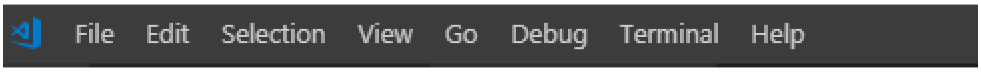
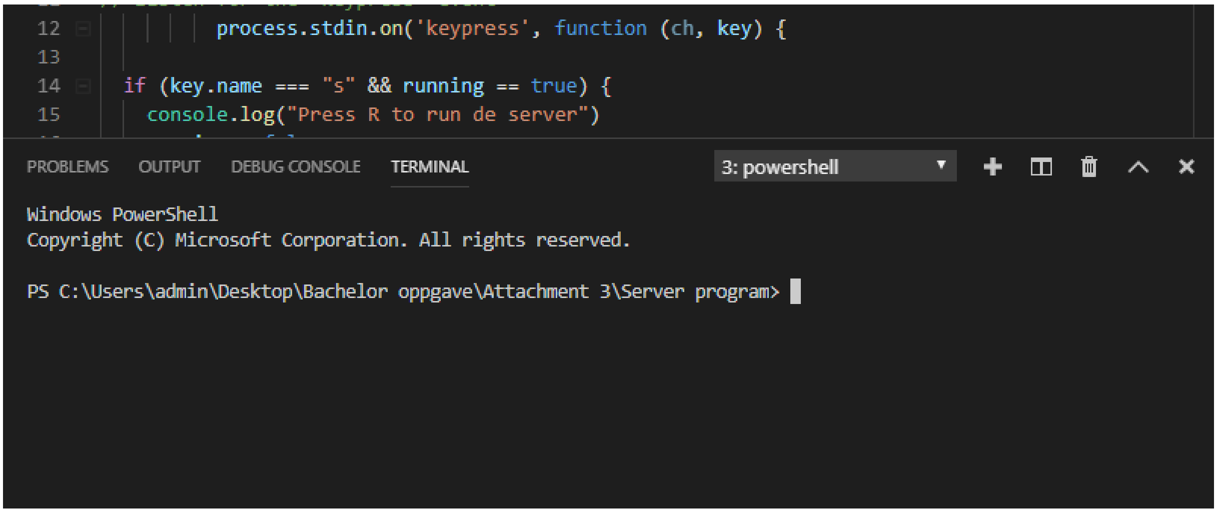


Figure 4 shows the upper bar (tab bar) in title area above editor in visual studio code

Click on Terminal then click on new Terminal, then Terminal logs opens up.

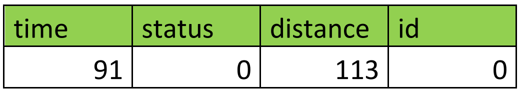


```
            process.stdin.on('keypress', function (ch, key) {

    if (key.name === "s" && running == true) {
      console.log("Press R to run de server")
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\admin\Desktop\Bachelor oppgave\Attachment 3\Server program>
```

Figure 5 showing the terminal log

On line 132 and 133 I mentioned making a header file, let header = “time,status, distance,id”, this is how it looks like inside the file after it prints the file inside an excel sheet:



time	status	distance	id
91	0	113	0

The header is made in by the server inside the file.

3 Experiments

We have done three experiments in this project to identify the accuracy of three parts in the sensor. The first part is the Accuracy in the distance measurement of the sensors. The second part is the time delay between the sensors and the server that get the data from the sensors. The last part is the Measurement of the field of view in the sensors.

By doing these experiments, we could find out deficiencies or flaws in the sensor, and maybe work out a way to overcome these problems. These experiments will show us the capacity of the sensor in measuring the movement of a person.

To do these experiments we choose the laboratory in the electrical department at the university. Because the room was big, and we had, a lot of space to use to cover a large range of area that the sensors were able to cover.
We also used the hallway in the fourth floor of the electrical engineering department to do the distance experiment.

The experiments done in cooperation with the lab engineer of the electrical department. In the next parts of this chapter, I will show the results and discuss the outcome of every experiment separately and I will give a conclusion at the last.

3.1 Experiment 1 (Distance measurement of the sensors)

In this experiment, we measured the distance accuracy of the sensors. We used two-distance sensor (VL53L1X) and a server, which takes those measurements, save them and then print those data in a separate file as text file and a csv file.

The sensors connected to the server on the computer through a Wi-Fi hotspot.
The Wi-Fi hotspot created on the mobile phone so that it can connect both the server and the sensors on the same Wi-Fi.

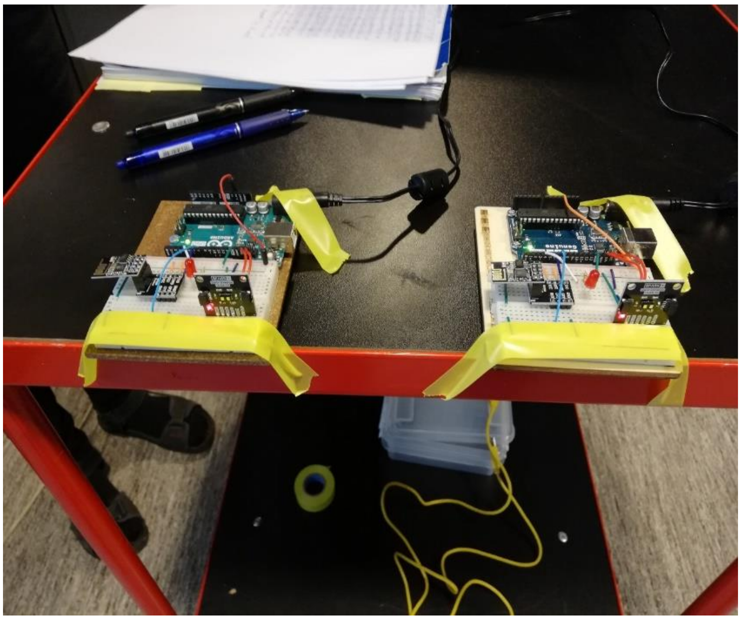
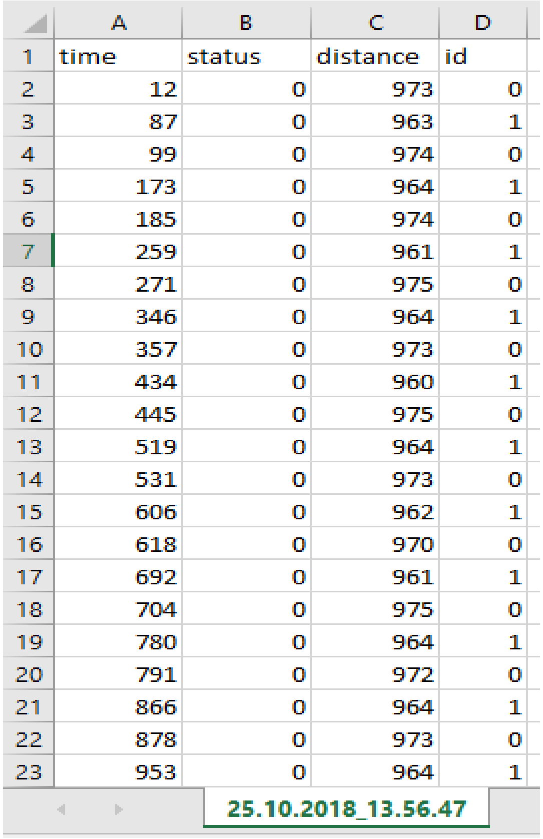


Figure 6 sensors attached to the trolley

We have divided the distance in the following manner: 20mm, 50mm, 100mm, 200mm, 300mm, 400mm, 500mm, 1000mm, 1500mm, 2000mm, 2500mm, 3000mm, 3500mm, 4000mm, 4500mm, 5000mm, 5500mm, and 6000mm. This kind of the division will help us to find out if there is any inaccuracy in the sensor in relations to distance measurement. We could also find the critical point, where the sensor has nearly accurate distance measurement compared to the actual distance.

The other thing with doing the experiment in this way is that we could find where the sensor is not able to register any data (distance).

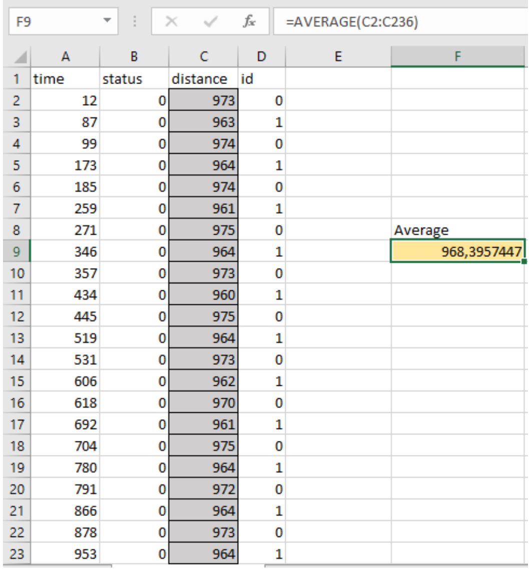
We took five rounds of measurements at same distances, to see the variation of data in every round. On every round, we placed the sensor on one distance and measured the data in 10-second time period, for example, on 100mm distance we waited for 10 seconds, then went to the next distance 200mm, and did the same. By doing it this way, we could take more data from the sensor. An Example of one measured data from the sensors at 1000mm distance in 10 seconds shown in the next page.



	A	B	C	D
1	time	status	distance	id
2	12	0	973	0
3	87	0	963	1
4	99	0	974	0
5	173	0	964	1
6	185	0	974	0
7	259	0	961	1
8	271	0	975	0
9	346	0	964	1
10	357	0	973	0
11	434	0	960	1
12	445	0	975	0
13	519	0	964	1
14	531	0	973	0
15	606	0	962	1
16	618	0	970	0
17	692	0	961	1
18	704	0	975	0
19	780	0	964	1
20	791	0	972	0
21	866	0	964	1
22	878	0	973	0
23	953	0	964	1

Figure 7 shows the collected data from 1000mm in 10-second time period, saved in excel file from the server.

In the next page at table 2, column round_1 with the following data (0, 0, 0, 101.3, 226.1, 343.7, 447.3, 968.4, 1483.04, 1988.8, 2493.1, 2986.9, 3471.9, 3927.7, 4355.3, 4647.9, 3950.7, 649.7) [mm]. I rewrote the data because I want to show how you can find for example 968.4 one average sample, 968.4mm can be found by opening the excel sheet and writing the following formula (=Average (select the distance column then enter)), then you will get 968.4mm.
An example of this presented like this:



F9 =AVERAGE(C2:C236)

	A	B	C	D	E	F
1	time	status	distance	id		
2	12	0	973	0		
3	87	0	963	1		
4	99	0	974	0		
5	173	0	964	1		
6	185	0	974	0		
7	259	0	961	1		
8	271	0	975	0		Average
9	346	0	964	1		968,3957447
10	357	0	973	0		
11	434	0	960	1		
12	445	0	975	0		
13	519	0	964	1		
14	531	0	973	0		
15	606	0	962	1		
16	618	0	970	0		
17	692	0	961	1		
18	704	0	975	0		
19	780	0	964	1		
20	791	0	972	0		
21	866	0	964	1		
22	878	0	973	0		
23	953	0	964	1		

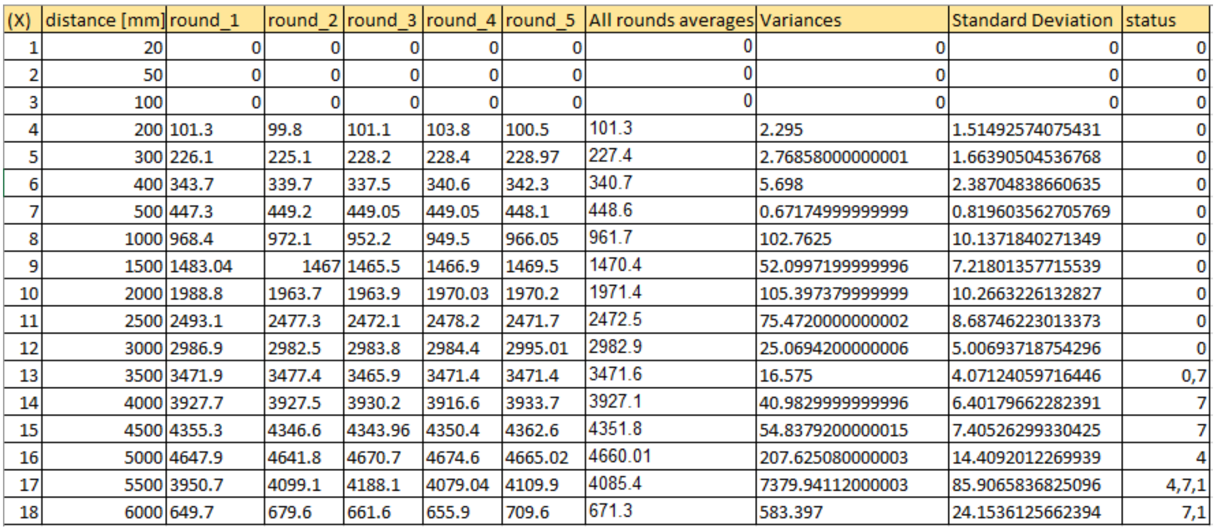
Figure 8 shows how to find the an average of every sample in every round inside table 2

Next, I will show the result we got from this experiment.

3.1.1 Results of the experiment (Distance)

In the result part I will show the results of the experiment in table 2 that contains the average of each sample of data in each round and the total all rounds average, and also the Variances and the standard deviation of the data. I have already explained the average of each sample at the previous page, but in this part, I will show how you can find the mean all or the all rounds average, variance, and standard deviation of the data.

I have used matlab to find out the all rounds averages, variance, and standard deviation. In table 2 the all rounds averages, variances and the standard deviation found through matlab code with the following formula I used down in figure 9.

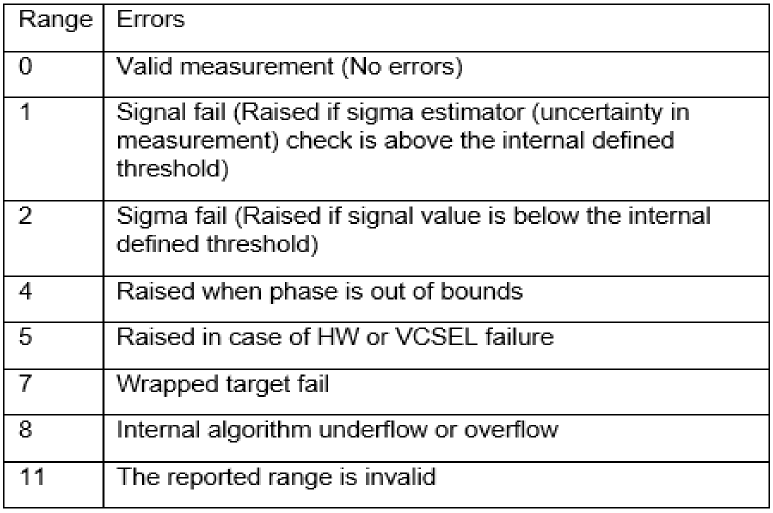


(X)	distance [mm]	round_1	round_2	round_3	round_4	round_5	All rounds averages	Variances	Standard Deviation	status
1	20	0	0	0	0	0	0	0	0	0
2	50	0	0	0	0	0	0	0	0	0
3	100	0	0	0	0	0	0	0	0	0
4	200	101.3	99.8	101.1	103.8	100.5	101.3	2.295	1.51492574075431	0
5	300	226.1	225.1	228.2	228.4	228.97	227.4	2.76858000000001	1.66390504536768	0
6	400	343.7	339.7	337.5	340.6	342.3	340.7	5.698	2.38704838660635	0
7	500	447.3	449.2	449.05	449.05	448.1	448.6	0.67174999999999	0.819603562705769	0
8	1000	968.4	972.1	952.2	949.5	966.05	961.7	102.7625	10.1371840271349	0
9	1500	1483.04	1467	1465.5	1466.9	1469.5	1470.4	52.0997199999996	7.21801357715539	0
10	2000	1988.8	1963.7	1963.9	1970.03	1970.2	1971.4	105.397379999999	10.2663226132827	0
11	2500	2493.1	2477.3	2472.1	2478.2	2471.7	2472.5	75.4720000000002	8.68746223013373	0
12	3000	2986.9	2982.5	2983.8	2984.4	2995.01	2982.9	25.0694200000006	5.00693718754296	0
13	3500	3471.9	3477.4	3465.9	3471.4	3471.4	3471.6	16.575	4.07124059716446	0,7
14	4000	3927.7	3927.5	3930.2	3916.6	3933.7	3927.1	40.9829999999996	6.40179662282391	7
15	4500	4355.3	4346.6	4343.96	4350.4	4362.6	4351.8	54.8379200000015	7.40526299330425	7
16	5000	4647.9	4641.8	4670.7	4674.6	4665.02	4660.01	207.625080000003	14.4092012269939	4
17	5500	3950.7	4099.1	4188.1	4079.04	4109.9	4085.4	7379.94112000003	85.9065836825096	4,7,1
18	6000	649.7	679.6	661.6	655.9	709.6	671.3	583.397	24.1536125662394	7,1

Table 2: The Average of every round with all rounds averages, variances and standard deviation of the experiment. All the data is in millimeter (mm).

Status of the sensor shows if the sensor is in a valid range to register the data. We can see from table 2 that the range of detection in the sensor above 4000mm will be invalid range of detection, because the distance sensor we have cannot register data over 4000, and it will show us the range is 7,4,1 which means it is an invalid range of detection.

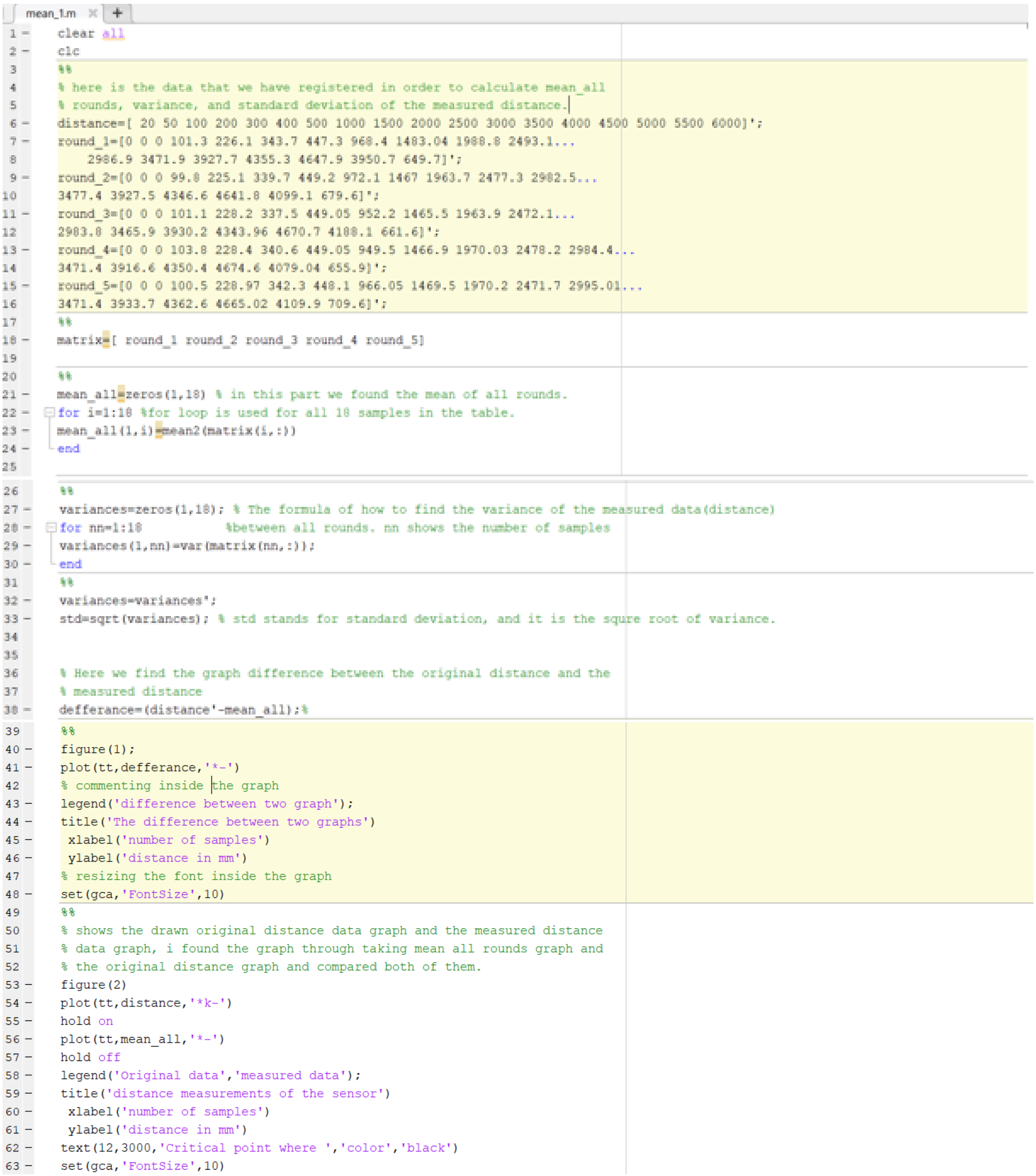
Range status explained in the website. I referred to the website in the reference list. [5]



Range	Errors
0	Valid measurement (No errors)
1	Signal fail (Raised if sigma estimator (uncertainty in measurement) check is above the internal defined threshold)
2	Sigma fail (Raised if signal value is below the internal defined threshold)
4	Raised when phase is out of bounds
5	Raised in case of HW or VCSEL failure
7	Wrapped target fail
8	Internal algorithm underflow or overflow
11	The reported range is invalid

Table 3 shows the range status of the sensor

In the next page, I have placed the matlab code I used to draw the graphs for the data in table 2, and I show how to find mean all or all round averages, variances, and standard deviation of the data.



```
1   clear all
2   clc
3   %%
4   % here is the data that we have registered in order to calculate mean_all
5   % rounds, variance, and standard deviation of the measured distance.
6   distance=[ 20 50 100 200 300 400 500 1000 1500 2000 2500 3000 3500 4000 4500 5000 5500 6000]';
7   round_1=[0 0 0 101.3 226.1 343.7 447.3 968.4 1483.04 1988.8 2493.1...
8       2986.9 3471.9 3927.7 4355.3 4647.9 3950.7 649.7]';
9   round_2=[0 0 0 99.8 225.1 339.7 449.2 972.1 1467 1963.7 2477.3 2982.5...
10  3477.4 3927.5 4346.6 4641.8 4099.1 679.6]';
11  round_3=[0 0 0 101.1 228.2 337.5 449.05 952.2 1465.5 1963.9 2472.1...
12  2983.8 3465.9 3930.2 4343.96 4670.7 4188.1 661.6]';
13  round_4=[0 0 0 103.8 228.4 340.6 449.05 949.5 1466.9 1970.03 2478.2 2984.4...
14  3471.4 3916.6 4350.4 4674.6 4079.04 655.9]';
15  round_5=[0 0 0 100.5 228.97 342.3 448.1 966.05 1469.5 1970.2 2471.7 2995.01...
16  3471.4 3933.7 4362.6 4665.02 4109.9 709.6]';
17  %%
18  matrix=[ round_1 round_2 round_3 round_4 round_5]
19
20  %%
21  mean_all=zeros(1,18) % in this part we found the mean of all rounds.
22  for i=1:18 %for loop is used for all 18 samples in the table.
23  mean_all(1,i)=mean2(matrix(i,:))
24  end
25
26  %%
27  variances=zeros(1,18); % The formula of how to find the variance of the measured data(distance)
28  for nn=1:18          %between all rounds. nn shows the number of samples
29  variances(1,nn)=var(matrix(nn,:));
30  end
31  %%
32  variances=variances';
33  std=sqrt(variances); % std stands for standard deviation, and it is the squre root of variance.
34
35
36  % Here we find the graph difference between the original distance and the
37  % measured distance
38  defferance=(distance'-mean_all);%
39  %%
40  figure(1);
41  plot(tt,defferance,'*-')
42  % commenting inside the graph
43  legend('difference between two graph');
44  title('The difference between two graphs')
45   xlabel('number of samples')
46   ylabel('distance in mm')
47  % resizing the font inside the graph
48  set(gca,'FontSize',10)
49  %%
50  % shows the drawn original distance data graph and the measured distance
51  % data graph, i found the graph through taking mean all rounds graph and
52  % the original distance graph and compared both of them.
53  figure(2)
54  plot(tt,distance,'*k-')
55  hold on
56  plot(tt,mean_all,'*-')
57  hold off
58  legend('Original data','measured data');
59  title('distance measurements of the sensor')
60   xlabel('number of samples')
61   ylabel('distance in mm')
62  text(12,3000,'Critical point where ','color','black')
63  set(gca,'FontSize',10)
```

Figure 9 shows how to find the all rounds averages or mean all, variance and standard deviation

I have explained the code in figure 9 by comments in green color

From line 3 to 16 with the yellow color is the set of data shown in table 2. From line number 21 to 24 is the process of finding the all mean or the all rounds average of the data. Line number 53 to 63 is the process of drawing the graph of both original distance [mm] data and the measured distance data (all round averages). Line number 40 to 48 I have drawn the difference between two graph, the original distance [mm] data and the measured distance data. The purple color line in figure 9 is the comment I wrote inside the graph.

In the discussion part of this experiment, I will show you the graphs that I drew from the formulas I used in matlab, and I will discuss the results.

3.1.2 Discussion (experiment 1)

In this part, I will explain the results I got from this experiment with analysis that I did and the results I found, I have drawn more graphs so the idea could be clearer to the reader.

I have placed the graphs that I drew from the formula I used in matlab. In figure 10 graph, we can see two lines of graphs, the black line is the original distance [mm] data and the blue line is the measured distance data or the all rounds averages [mm], from the graph, we can see that the sensor has a critical point at 2987mm compared to original distance at 3000mm. There is a minimum error of 13mm, which shows that the measured distance (all rounds averages) [mm] data is approximately near to the original distance [mm] data.

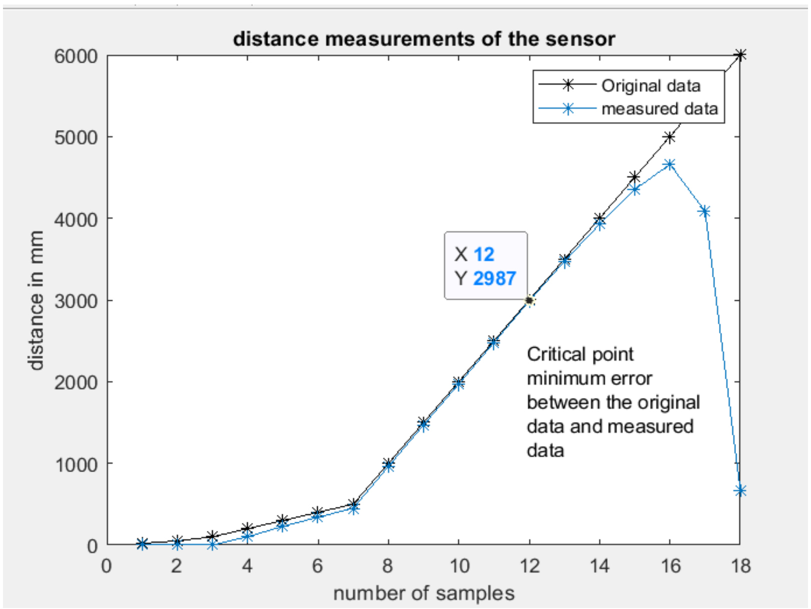


Figure 10 shows the original distance [mm] graph (black line) and the measured distance graph (all rounds averages [mm] [blue line])

I should mention here is that the black line graph is the data from the original distance [mm] in table 2, that I compared with the all rounds averages [mm] [blue line graph].

The important part is that the sensor is able to detect the object within the range of 500mm to 3500mm with an error of 50mm. To accept the maximum error of approximately 50mm, we can say that the range of detection is acceptable from 500mm to 3500mm (millimeters) as shown in figure11.

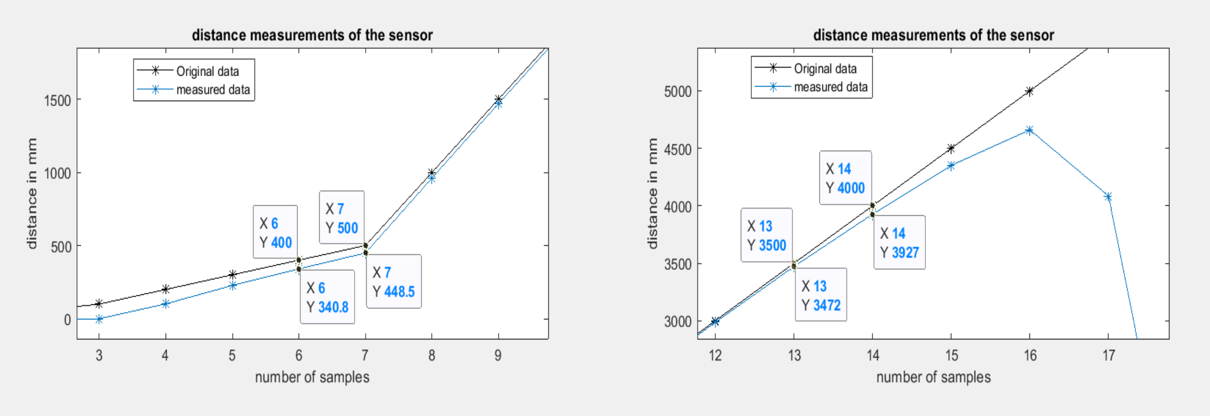


Figure 11 shows the minimum range and maximum range of the sensor.

In figure 12, I choose the range of detection in the sensor between 500mm and 3500mm. because both graphs blue and black looks near to each other with minimum error between each graphs.

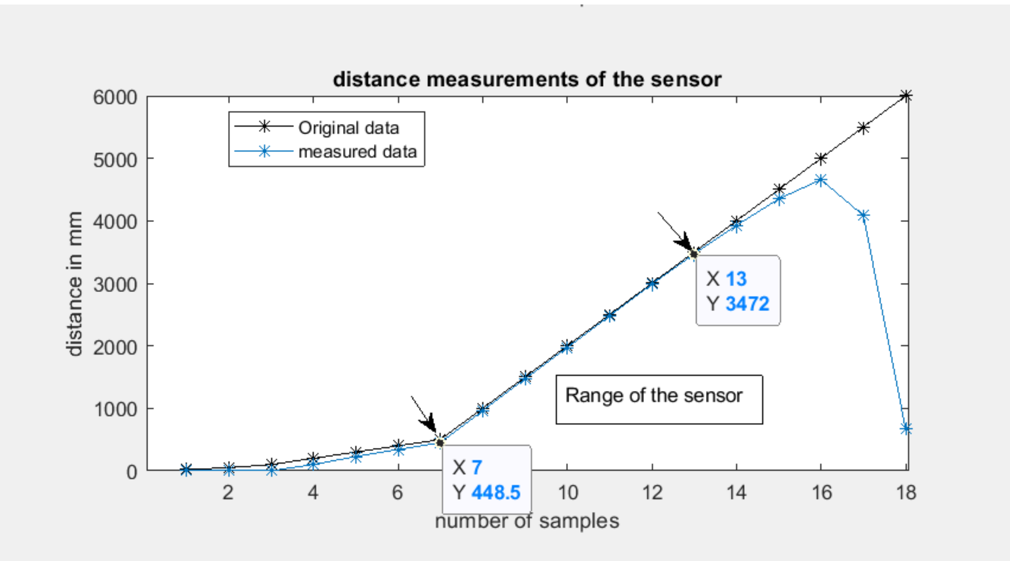


Figure 12 shows the range of detection between 448.5mm to 3472mm measured distance

Figure 13 shows the difference between original distance [mm] data and measured distance (all rounds averages [mm]) data for all samples. This can clearly shows that the sensor detection range reduces significantly after 3500mm distance, because the error graph exponentially increases after 4000mm, which is the maximum range of detection for the sensor.

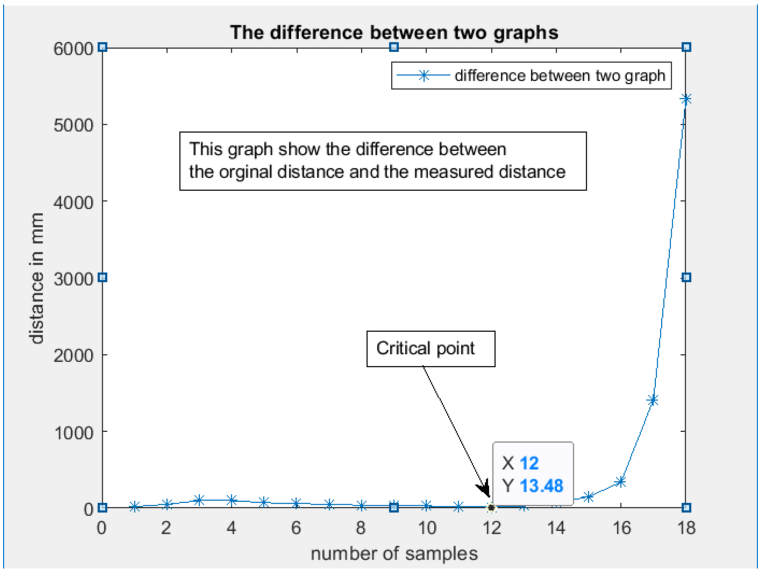


Figure 13 shows the graph difference between the two graphs from figure 10

3.2 Experiment 2 (Time delay measurements of the sensors)

In this experiment, we wanted to find the time delay between the sensors and the server, so we added an extra button in the code called an “x” button. We got a cardboard, I held the cardboard in front of the sensors, and lab engineer was controlling the server, by clicking on the “x” button while the server was running and collecting data, and simultaneously when he clicked the button, I lifted the cardboard from the sensors to see how fast the sensors will register the new distance data.

We called this time delay for Δt. The Δt is the time delay from which we remove the cardboard from the front of the sensors and we simultaneously click the button “x” on the keypad in the pc, and then calculate the time it took for the sensors to register the new data and send data to the server.

From figure 15, we can see that I was holding the cardboard and the lab engineer was ready to click on the x button on the keypad in the pc. In figure 14, I simultaneously with the x button pressed lifted the cardboard and we took a measurement of how fast the sensor sends the new distance of data to server.

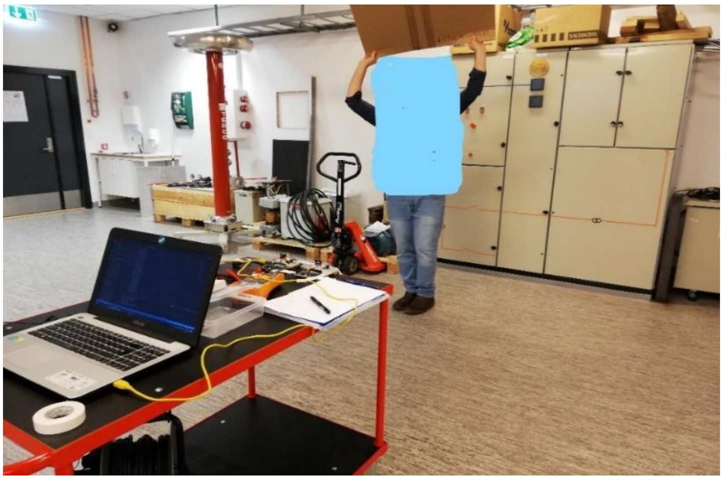


Figure 14 shows the time when I lift the cardboard from the of the sensor

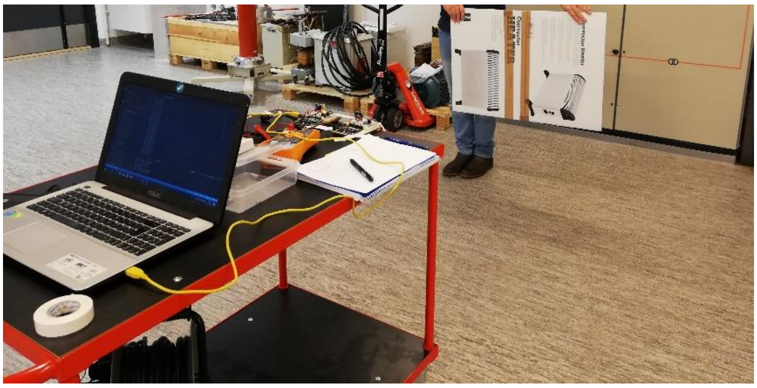


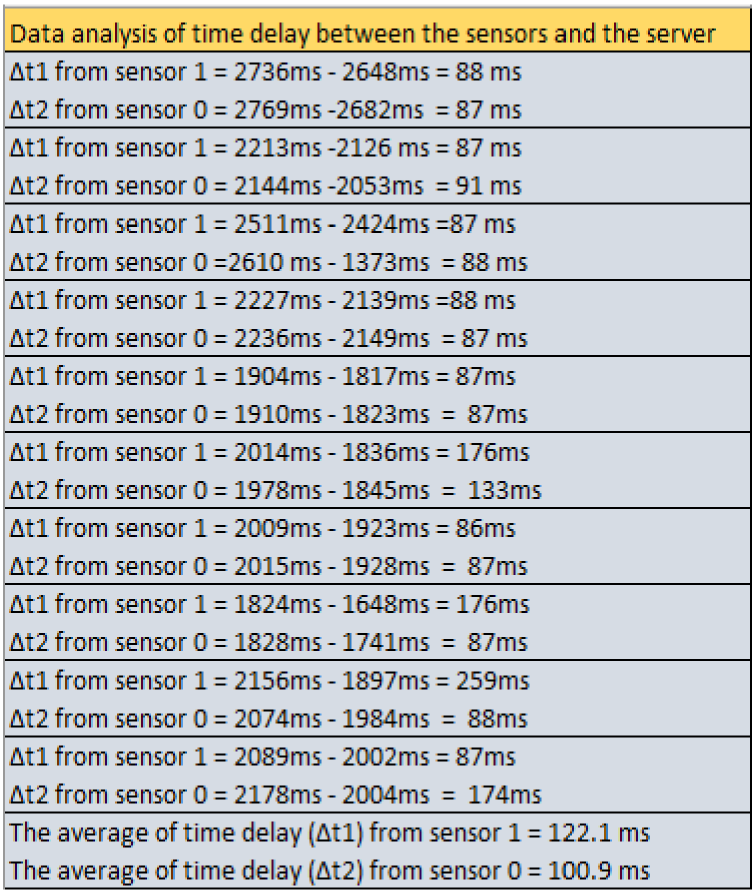
Figure15 shows the time when I was holding cardboard in front of the sensors.

In the next part of this experiment, I will show the result I took from this experiment.

3.2.1 Results of the experiment 2(Time delay)

We took 10 rounds of data for every time delay to have a clear picture of how long does the sensors takes to send the data to the server. Two sensors were used sensor 0, and sensor 1. The time delay is in millisecond (ms), because the sensors registers time delay in milliseconds (ms).

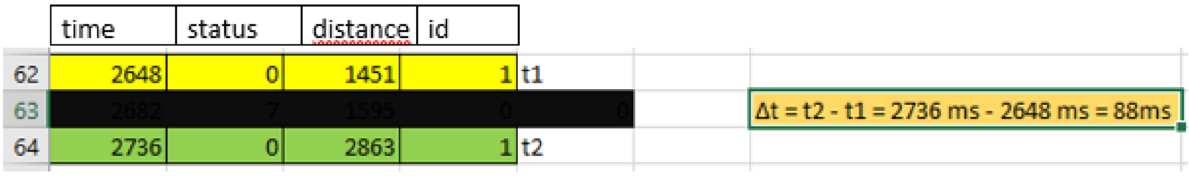
In table 4, I have 10 rounds of data that we collected from the experiment. I have also taken the average of the time delay between both the sensors and the server. The idea here is to find out how fast the server register the data from the sensors.



Data analysis of time delay between the sensors and the server
Δt1 from sensor 1 = 2736ms - 2648ms = 88 ms Δt2 from sensor 0 = 2769ms -2682ms = 87 ms
Δt1 from sensor 1 = 2213ms -2126 ms = 87 ms Δt2 from sensor 0 = 2144ms -2053ms = 91 ms
Δt1 from sensor 1 = 2511ms - 2424ms =87 ms Δt2 from sensor 0 =2610 ms - 1373ms = 88 ms
Δt1 from sensor 1 = 2227ms - 2139ms =88 ms Δt2 from sensor 0 = 2236ms - 2149ms = 87 ms
Δt1 from sensor 1 = 1904ms - 1817ms = 87ms Δt2 from sensor 0 = 1910ms - 1823ms = 87ms
Δt1 from sensor 1 = 2014ms - 1836ms = 176ms Δt2 from sensor 0 = 1978ms - 1845ms = 133ms
Δt1 from sensor 1 = 2009ms - 1923ms = 86ms Δt2 from sensor 0 = 2015ms - 1928ms = 87ms
Δt1 from sensor 1 = 1824ms - 1648ms = 176ms Δt2 from sensor 0 = 1828ms - 1741ms = 87ms
Δt1 from sensor 1 = 2156ms - 1897ms = 259ms Δt2 from sensor 0 = 2074ms - 1984ms = 88ms
Δt1 from sensor 1 = 2089ms - 2002ms = 87ms Δt2 from sensor 0 = 2178ms - 2004ms = 174ms
The average of time delay (Δt1) from sensor 1 = 122.1 ms The average of time delay (Δt2) from sensor 0 = 100.9 ms

Table 4 time delay Δt of the sensors

I can take one example from the data we have collected from the experiment. In this example, im going to find out the time delay between sensor 1 and the server.



	time	status	distance	id			
62	2648	0	1451	1	t1		
63	2682	7	1595	0	0		Δt = t2 - t1 = 2736 ms - 2648 ms = 88ms
64	2736	0	2863	1	t2		

Table 5 showing how to calculate the Δt, the time delay

In the next part, I will discuss the results of this experiment.

3.2.2 Discussion (experiment 2)

In table 4 we have the time delay of the sensors. At the last part of the table 4 I took the average time delay between the server and each sensor. The average time delay for sensor 1 is 122.1ms and the average time delay for sensor 0 is 100.9ms. This result is 1/10 of a Second. It means that it takes 122.1ms for sensor 1 and 100.9ms for sensor 0 to send the new distance to the server.

The server has a time delay of 122.1ms for sensor 1 and 100.1ms for sensor 0 to receive the data, is this a long-time delay?

Well to answer the question above we must think that what can we do in 122.1ms or 100.9ms (milliseconds), well a human eye can blink in 200ms, is this relevant to what we do here? No, here we are observing the movement of older people, so older people would not do much in 122.1ms or 100.9ms.

Older people movement is slower than younger people movement. Older people have some weaknesses related to sickness and diseases. Our results of 122.1ms for sensor 1 and 100.9ms would not affect what we want to observe in older people movement. We observe if they fall or something happens to them.

The results we got from this experiment shows us that there is significantly short time delay between the server and the sensors. This means that when there is a sudden change in movement a person the sensor immediately registers that and transfer it to the server with a time delay of approximately 1/10 of second or 100ms time period.

3.3 Experiment 3 (Field of view of the sensor)

In this experiment, we found the θ angle view or the field of view of the sensor.
We did this experiment at electrical department laboratory on the fourth floor of the engineering department in HVL (høgskolen på vestlandet). We first put three long line of plastic white tape on the ground to mark the distance of each meter in the ground. We did mark 3 meters on the ground with the white tape.

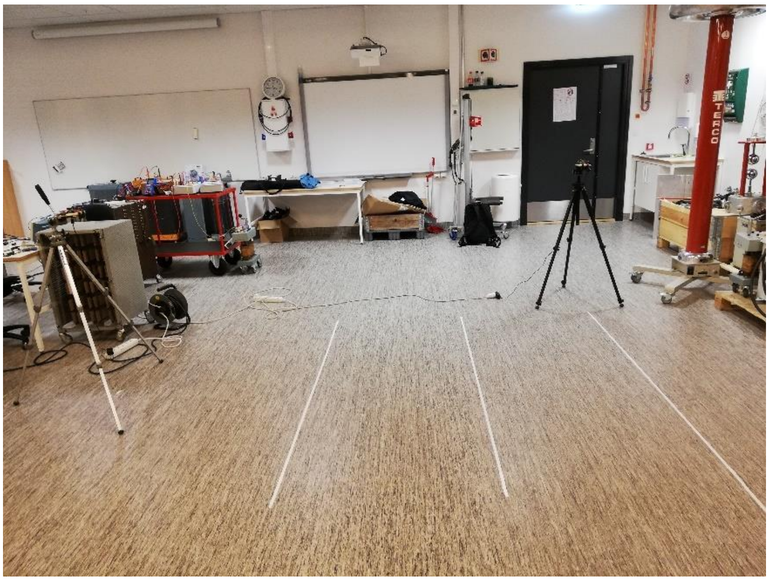


Figure 16: showing the white tape in the ground.

As we see in figure 16 sensor1 is on left side of the picture attached to a white tripod and sensor 0 is attached on a black tripod on the left of the picture. After that, I stood in front of sensor 0 and then I walked along the line of the white tape. We did this for 1 meter, 2 meters and 3 meters. I walked with a small cardboard on my side, first doing 10 rounds for each meter so it is 30 times for 3 meters, and then without the cardboard doing the same thing.

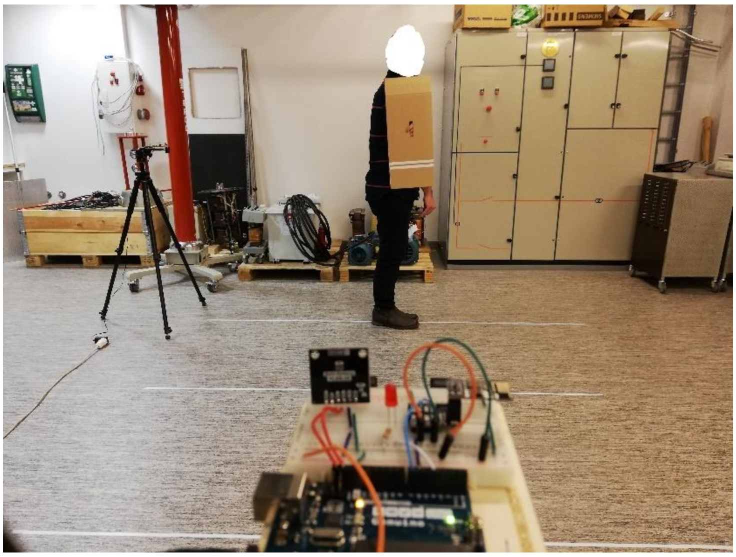


Figure 17 holding the cardboard and walking along the line in front of the sensors.

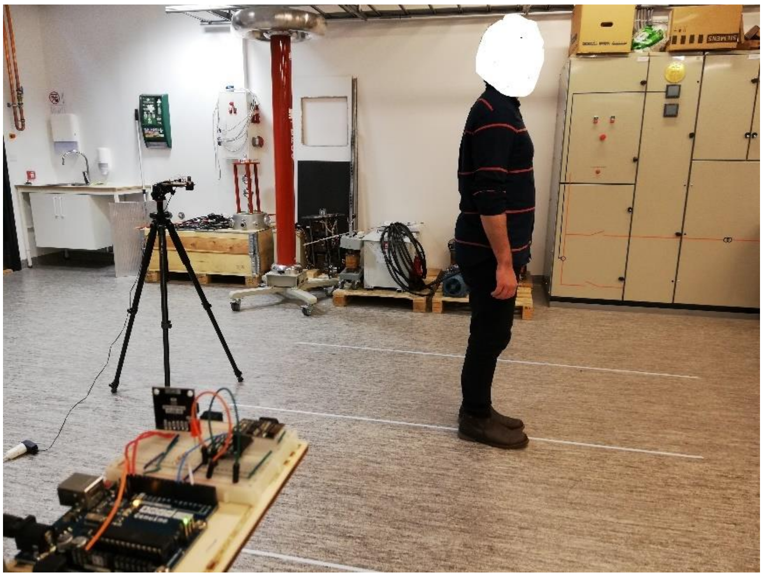


Figure 18: walking in front of the sensors without the cardboard

We can see in figure 18 sensor 0 is in my back and I am walking in front of sensor 1. The idea here is that sensor 0 will make a straight line in the graph and when I come in front of sensor 1 it will make parabola kind of graph. You can see the graph in figure 19 in the result section of this experiment.

In the next page, I will show and explain how to find the angle view of the sensor. I named the angle view of the sensor as θ angle,

3.3.1 Results of experiment 3 (Field of view)

In this part, I am showing the results I got from this experiment in table 6.
Table 6 shows the angle of view of the sensor done in two way. One by holding the cardboard [red] to see how accurate can the sensor register the θ angle view or the field of view, and the other without holding the cardboard [blue].

I can take one example of figure 19 and show how I found the θ angle (the field of view) of the sensor:

From figure 19, we can see that there is a θ angle, and this is the angle view of sensor 1 or the field of view of sensor 1. To find this θ angle view we used reverse tangent of the angle by finding B and D from graph.

The formula for the angle θ is: $\theta = 2 * \text{atan}\left(\frac{B/2}{D}\right)$

$B = 1470 - 800 = 678$, and D = 1940, these values are taken from the graph in figure 19. Now that we have both B and D, we put it in our formula:
find θ: $\theta = 2 * \text{atan}\left(\frac{B/2}{D}\right) = 2 * \text{atan}(\frac{678/2}{1940})$ = 19.8 º

As I mentioned earlier the field of view or the θ angle view of the sensor is between 15 º - 27 º [5], so this result is more realistic, and it corresponds with datasheet of the sensor [5].

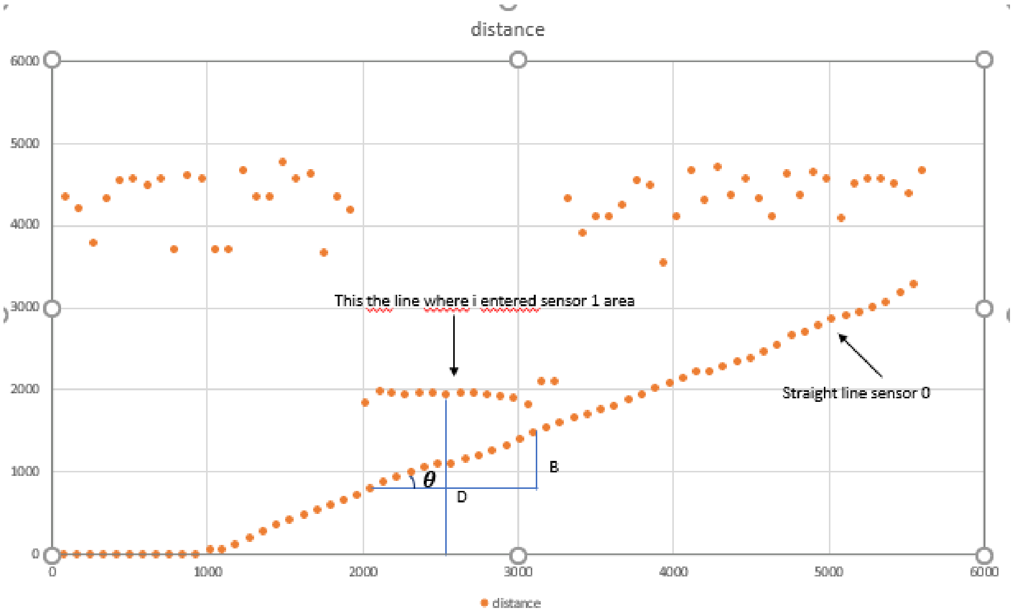
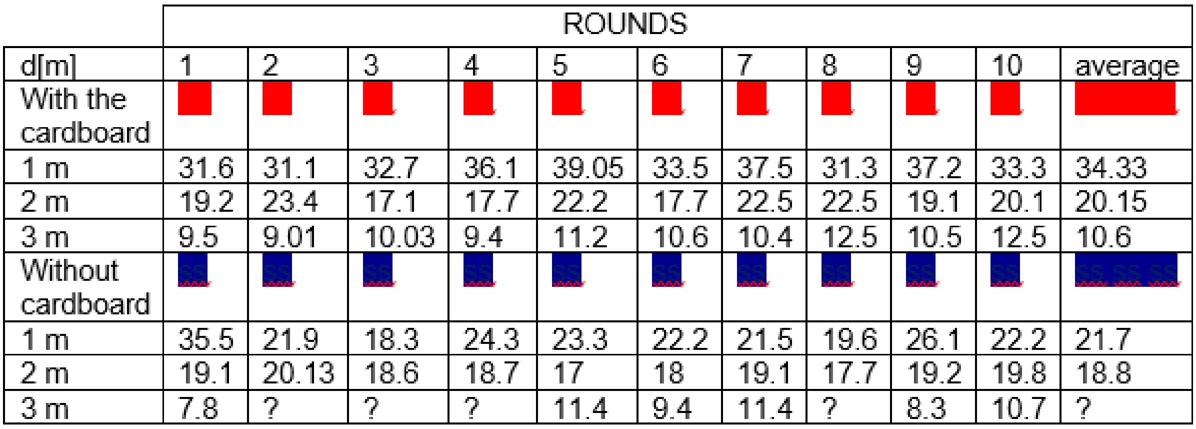


Figure 19: showing sensor 0 with a straight line and sensor 1 has a parabola kind of graph.

In table 6 I found all the θ angle view or the field of view of the sensor 1, and I also calculated their averages so it could be easier to understand the data.



	ROUNDS										
d[m]	1	2	3	4	5	6	7	8	9	10	average
With the cardboard	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥
1 m	31.6	31.1	32.7	36.1	39.05	33.5	37.5	31.3	37.2	33.3	34.33
2 m	19.2	23.4	17.1	17.7	22.2	17.7	22.5	22.5	19.1	20.1	20.15
3 m	9.5	9.01	10.03	9.4	11.2	10.6	10.4	12.5	10.5	12.5	10.6
Without cardboard	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
1 m	35.5	21.9	18.3	24.3	23.3	22.2	21.5	19.6	26.1	22.2	21.7
2 m	19.1	20.13	18.6	18.7	17	18	19.1	17.7	19.2	19.8	18.8
3 m	7.8	?	?	?	11.4	9.4	11.4	?	8.3	10.7	?

Table 6 showing the θ angle view or the field of view of the sensor 1. All the values are in º degrees.

3.3.2 Discussion (experiment 3)

In table 6 I have all the values with the average of each round in º degrees. In table 6 above, the first 3 set of data is the θ angle view or field of view of sensor 1 where I hold the cardboard [red]. The last 3 data set of data is the θ angle view or field of view of the sensor 1 is where I was standing in front of the sensor without holding the cardboard [blue].

If we see the data in 1 m [red] with the cardboard [red], we have a larger field of view or θ angle view, but in 2 m [red] the average field of view of sensor 1 is 20.15 º, we are inside the field of view written by the sensors datasheet and it is between 15º-27º degrees [5]. The question here is why it is showing larger field of view or θ angle view in 1 m distance with the cardboard that is because the sensor has a long-range mode detection and it has some difficulty registering data in short range mode detection. This is probably one reason I can come up with that could mean that the sensor is not that accurate in measuring θ angle view or the field of view below the 1 m distance.

In addition, when we look at 3 m [red] line distance with the cardboard we have an average field of view angle of 10.6º it is the below the threshold. I am not sure, why the result is lower here, maybe because the sensor could not get exact data at that position, or there was something else that have interfered with the measurements.

However, when we look at the 3-last data of º degrees in table 6 without the cardboard [blue], we see that there is a slight difference in 1 m distance, we can see the average θ angle or field of view of the sensor is 21.7º, which is inside the threshold written by the sensor's manufacturer data sheet.

However, in 2 m distance the average θ angle is 18.8º, this means that the data is a little bit comparable with the information provided in the datasheet of the sensor [5]. In 3 m line without the cardboard, I could not find the θ angle because the data were out of order. Sensor 1 could not detect me well and that is why it was difficult for me to find the angle. Down in figure 20 I am showing the data from 3 m line without the cardboard this will probably be clearer to see.

One thing to mention here is that when I was holding the cardboard, the sensors was sending the signal 90º or what we call normal standing angle on the cardboard. The other thing is that when I was standing alone without the cardboard the sensor signal was not 90º because my body is not flat, it is round, and so the sensors were sending the signal on all direction of the body.

The sensor field of view is more visible with flat object, other than round object like for example a person. That is why we used a cardboard to see how clear are the field of view or the θ angle view of the sensor is, and I also measured myself without the cardboard to see how the sensor register the θ angle view or the field of view of the sensor in round shape bodies like a person body.

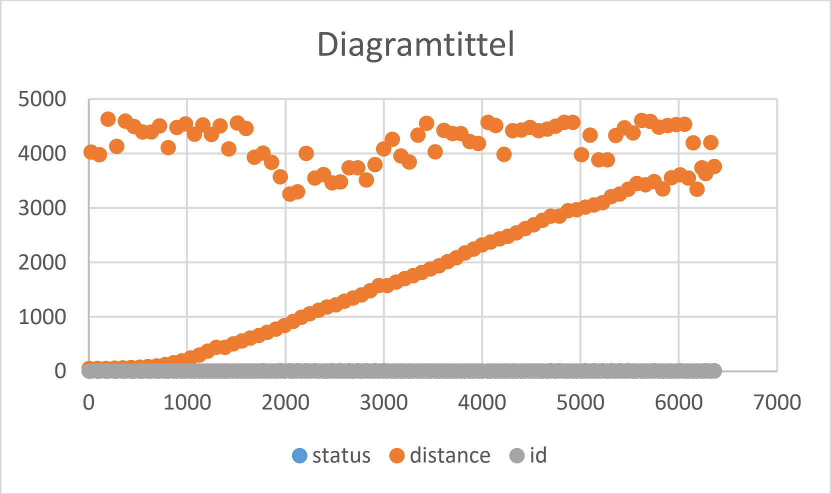


Figure 20 showing data from 3 m without holding the cardboard, the signal is out of order that is why it is hard to find θ angle.

In figure 21 the sensor in my opinion has a better θ angle view or field of view in 2 m distance, the data is clearer, and you get the result that is comparable with the sensor standard field of view written in the datasheet [5].

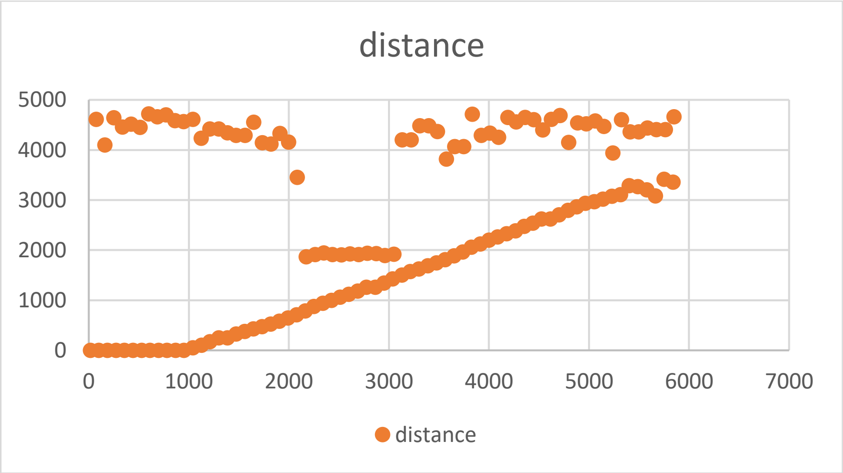


Figure 21 showing data from 2 m line without the cardboard.

4 Further work

The finish project is completed according to the plan. We have had 3 experiments that was done at HVL (Høgskolen på Vestlandet), these experiments have shown us a lot about the accuracy of the sensor in term of distance, time delay and field of view of the sensors.

I have done some analysis that could be used as a tool for future use of the sensor. Future application with this sensor relies on further experiments and cooperation between different fields, such us medical field for example.

Our product is at the beginning stages so it is obvious that more work is needed to develop it more reliable for example at future the sensor could be developed to a smaller more compact product, so that it would be easy for costumers or users to install and use it.

5 References

[1] K. GLAD, "Eldrebølgen slår lenger inn over Europa enn Norge," 7 April 2003. [Online]. Available: https://www.ssb.no/befolkning/artikler-og-publikasjoner/eldrebolgen-slaar-lenger-inn-over-europa-enn-norge. [Accessed 18 november 2018].

[2] G. Diraco, A. Leone and P. Siciliano, "A Radar-Based Smart Sensor for Unobtrusive Elderly Monitoring in Ambient Assisted Living Applications," *biosensors,* pp. 7,55, 24 November 2017.

[3] Høgskolen på Vestlandet, "hvl," Høgskolen på Vestlandet, 2018. [Online]. Available: https://www.hvl.no/prosjekt/473997/. [Accessed 14 Mars 2019].

[4] S. R.Lord, C. Sherrington and H. B. Menz, "Falls in older people," in *Falls in older people, Risk factor and strategies for prevention*, Cambridge, United Kingdom, University press, Cambridge, 2001, pp. 9-10.

[5] Sparkfun, "Sparkfun Start Something," Sparkfun Electronics, 2018. [Online]. Available: https://learn.sparkfun.com/tutorials/qwiic-distance-sensor-vl53l1x-hookup-guide. [Accessed 25 September 2018].

[6] Visual Studio Code, "Visual Studio Code," Microsoft, November 2018. [Online]. Available: https://code.visualstudio.com/download. [Accessed 1 Januar 2019].

[7] STMicroelectronics, "ST," STMicroelectronics, Februar 2018. [Online]. Available: https://cdn.sparkfun.com/assets/b/f/2/9/8/VL53L1X_Datasheet.pdf. [Accessed 1 Januar 2019].

Appendix A GANTT CHART

See the attachment 1 with Excel file Gantt Chart

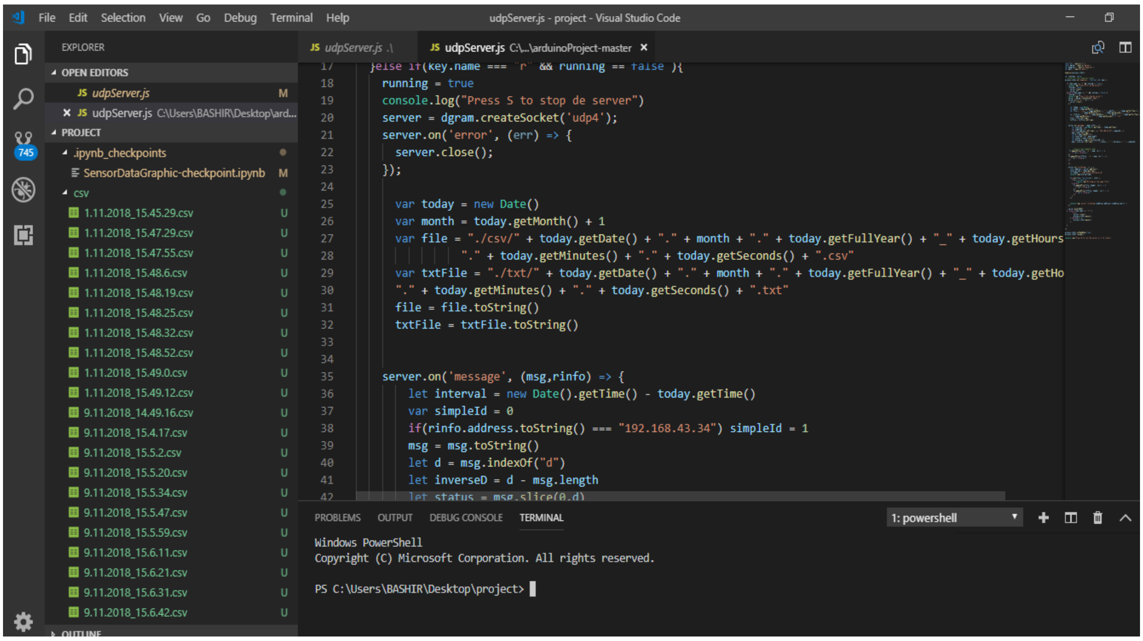
Appendix B DATA COLLECTED FROM THE EXPERIMENTS, SERVER PROGRAM, MATLAB CODE

Attachment 2 contains all the data collected in the project from all the experiments. Attachment 0 contains matlab code used in distance experiment. Attachment 3 is the code used in the server program.

Appendix C HOW TO RUN AND INSTALL THE SERVER PROGRAM JAVASCRIPT

To run the server, I have attached another file with this document(attachment 3), that includes all the library that allow the server to run and you can find the server itself. You must install visual studio code from the visual studio code site, https://code.visualstudio.com/download , that I already reffered to this site in the reference list.

When visual studio is installed in your PC, run the server that's in the file by clicking right click in run it as a visual studio code. When the server comes up on the left of the page at top bottom there is a picture of this 🗐 click on it in choose the folder that the server is located in that folder. The last part click at the top on the menu bar on Terminal button it looks like this:



```
}else if(key.name === "r" && running == false ){
  running = true
  console.log("Press S to stop de server")
  server = dgram.createSocket('udp4');
  server.on('error', (err) => {
    server.close();
  });

    var today = new Date()
    var month = today.getMonth() + 1
    var file = "./csv/" + today.getDate() + "." + month + "." + today.getFullYear() + "_" + today.getHours
              "." + today.getMinutes() + "." + today.getSeconds() + ".csv"
    var txtFile = "./txt/" + today.getDate() + "." + month + "." + today.getFullYear() + "_" + today.getHo
    "." + today.getMinutes() + "." + today.getSeconds() + ".txt"
    file = file.toString()
    txtFile = txtFile.toString()


  server.on('message', (msg,rinfo) => {
      let interval = new Date().getTime() - today.getTime()
      var simpleId = 0
      if(rinfo.address.toString() === "192.168.43.34") simpleId = 1
      msg = msg.toString()
      let d = msg.indexOf("d")
      let inverseD = d - msg.length
      let status = msg.slice(0,d)
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\BASHIR\Desktop\project>
```

You can see the Terminal button at the top click on it, and choose New Terminal, and you can see from the picture above that there is a Terminal that have opened down at the picture, when Terminal opens write the following:

```
node .\udpServer.js\
```

The red sentence above is the name of the server. After this, the server will run you will have data registered on the two folders called csv and txt on the same folder that the server is located.