

YEDİTEPE UNIVERSITY

DEPARTMENT OF MECHANICAL ENGINEERING

ME456 MECHATRONICS WORKSHOP

**Ultrasonic Distance Measurement**

**Instructor:** Koray Kadir Şafak

**Group Id:** 5

**Group Members:**

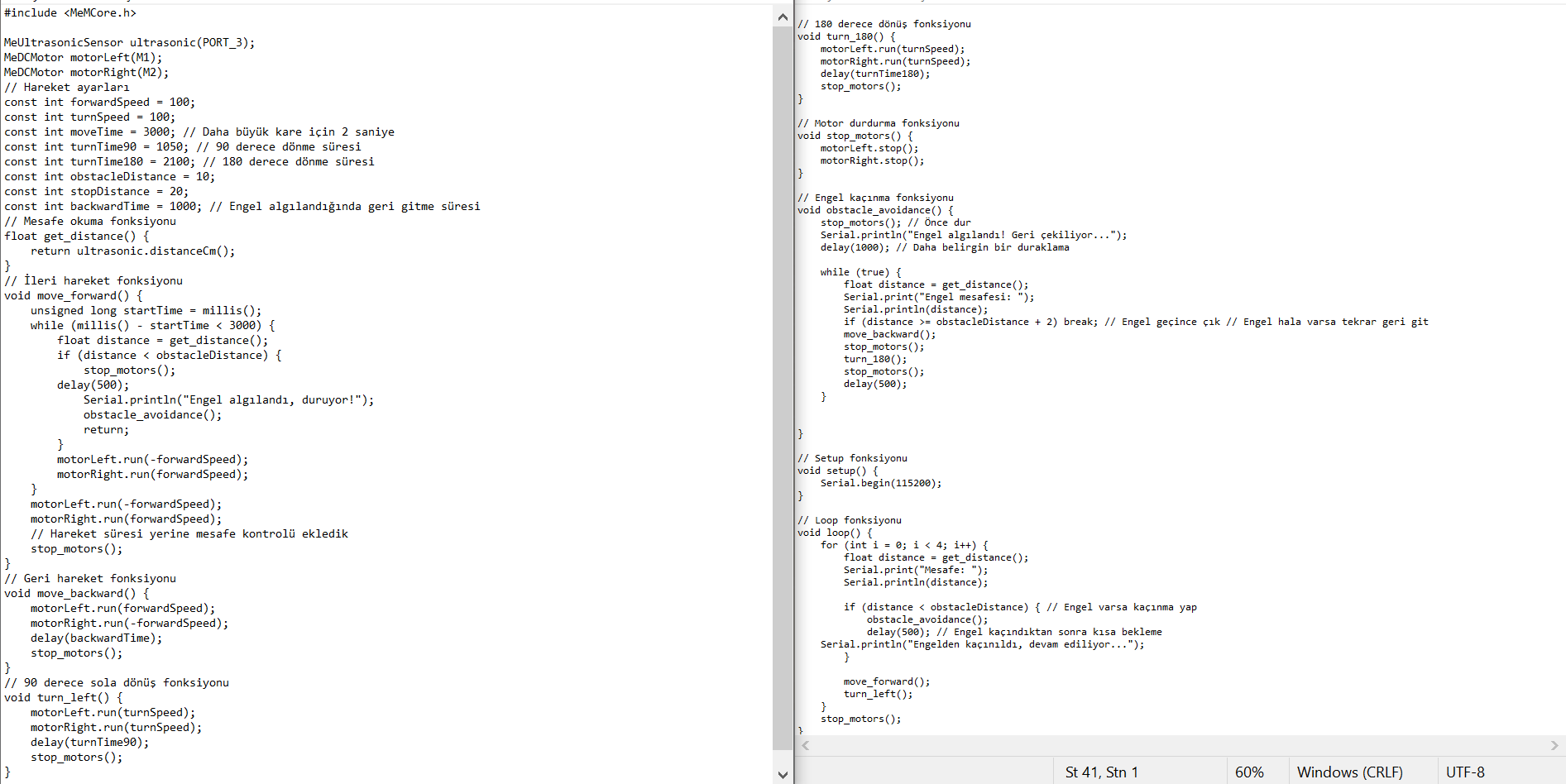
Ömer Dolaş 20200705001

Abdulkadir Salman 20200705027  
Abdullah Serdar Özdemir 20200705049  
Furkan Yılmaztekin 20180705040

# Introduction

The purpose of this experiment was to calibrate the HC-SR04 ultrasonic distance sensor and evaluate its accuracy by comparing measured values with actual distances. Additionally, a best-fit polynomial curve was derived to improve sensor readings, and the overall non-linearity error was assessed.

# Code



**Explanation of the Code** The Arduino code controls a sumo robot equipped with an HC-SR04 ultrasonic sensor and two DC motors. Below is an explanation of the key components:

* **Library Inclusions and Hardware Setup:**
  + #include <MeMCore.h>: This library provides control functions for Makeblock robots.
  + MeUltrasonicSensor ultrasonic(PORT\_3): Defines the ultrasonic sensor connected to port 3.
  + MeDCMotor motorLeft(M1), motorRight(M2): Defines the left and right motors.
* **Movement Control:**
  + move\_forward(): Moves the robot forward while continuously checking the distance from obstacles. If an obstacle is detected within 10 cm, the robot stops and calls the obstacle avoidance function.
  + move\_backward(): Moves the robot backward for a predefined duration.
  + turn\_left(): Turns the robot 90 degrees to the left.
  + turn\_180(): Turns the robot 180 degrees for obstacle avoidance.
  + stop\_motors(): Stops both motors.
* **Obstacle Avoidance:**
  + If an obstacle is detected within 10 cm, the robot moves backward and makes a 180-degree turn until it clears the obstacle.
* **Main Program Loop:**
  + The robot follows a square-shaped path.
  + At each step, the ultrasonic sensor checks for obstacles.
  + If an obstacle is detected, it executes the obstacle avoidance routine.
  + The process repeats for four iterations before stopping.

# Data

|  |  |  |  |
| --- | --- | --- | --- |
| **Actual (cm)** | **Sensor (cm)** | **Predicted (y = 0,975x + 0,8572)** | **Error (cm)** |
| 5 | 5,3 | 6,0247 | 0,7247 |
| 6 | 6,12 | 6,8242 | 0,7042 |
| 7 | 6,74 | 7,4287 | 0,6887 |
| 8 | 7,57 | 8,23795 | 0,66795 |
| 9 | 8,55 | 9,19345 | 0,64345 |
| 10 | 9,47 | 10,09045 | 0,62045 |
| 11 | 10,41 | 11,00695 | 0,59695 |
| 12 | 11,68 | 12,2452 | 0,5652 |
| 13 | 12,54 | 13,0837 | 0,5437 |
| 14 | 14,59 | 15,08245 | 0,49245 |
| 15 | 15,48 | 15,9502 | 0,4702 |
| 16 | 16,5 | 16,9447 | 0,4447 |
| 17 | 17,42 | 17,8417 | 0,4217 |
| 18 | 18,47 | 18,86545 | 0,39545 |
| 19 | 19,53 | 19,89895 | 0,36895 |
| 20 | 20,88 | 21,2152 | 0,3352 |
| 25 | 25,69 | 25,90495 | 0,21495 |
| 30 | 30,53 | 30,62395 | 0,09395 |
| 35 | 35,53 | 35,49895 | 0,03105 |
| 40 | 40,57 | 40,41295 | 0,15705 |
| 50 | 50,12 | 49,7242 | 0,3958 |
| 55 | 54,88 | 54,3652 | 0,5148 |
| 60 | 59,74 | 59,1037 | 0,6363 |
| 65 | 64,62 | 63,8617 | 0,7583 |
| 70 | 69,5 | 68,6197 | 0,8803 |
| 75 | 74,38 | 73,3777 | 1,0023 |
| 80 | 79,26 | 78,1357 | 1,1243 |
| 85 | 84,14 | 82,8937 | 1,2463 |
| 90 | 89,02 | 87,6517 | 1,3683 |
| 95 | 93,9 | 92,4097 | 1,4903 |
| 100 | 98,78 | 97,1677 | 1,6123 |
| 110 | 108,64 | 106,7812 | 1,8588 |
| 120 | 118,53 | 116,42395 | 2,10605 |
| 130 | 128,35 | 125,99845 | 2,35155 |
| 140 | 138,15 | 135,55345 | 2,59655 |
| 150 | 147,92 | 145,0792 | 2,8408 |
| 160 | 157,63 | 154,54645 | 3,08355 |
| 170 | 167,25 | 163,92595 | 3,32405 |
| 180 | 176,85 | 173,28595 | 3,56405 |
| 190 | 186,41 | 182,60695 | 3,80305 |
| 200 | 195,98 | 191,9377 | 4,0423 |
| 210 | 205,35 | 201,07345 | 4,27655 |
| 220 | 214,75 | 210,23845 | 4,51155 |
| 230 | 224,17 | 219,42295 | 4,74705 |
| 240 | 233,44 | 228,4612 | 4,9788 |
| 250 | 242,65 | 237,44095 | 5,20905 |

# Discussion

The code we wrote successfully enabled the robot to complete the given tasks. It allowed the robot to stop its movement upon detecting obstacles and perform a pre-defined obstacle avoidance maneuver. Additionally, the logic for the robot to follow a square path and turn at a specific angle at each corner was implemented within the code.

When evaluating the sensor accuracy, the R² value obtained for the best-fitting polynomial curve was 0.9999, indicating very high accuracy. However, the maximum non-linearity error of the sensor was calculated as 2.08362%, which suggests that small error margins exist, especially at longer distances.

Sensor accuracy directly impacted the robot’s performance. Due to measurement errors, the robot might misjudge distances, leading to slight deviations during obstacle avoidance and navigation. However, the low error rate did not result in a significant negative impact on overall performance.

To improve performance, the following improvements can be made:

* Advanced distance sensors (e.g., LiDAR or high-accuracy ultrasonic sensors) can be used for more precise measurements.
* An automated and controlled measurement system could be developed to minimize human-related measurement errors.
* Turn times can be optimized based on the surface type to ensure more accurate guidance on different surfaces.
* A system that validates sensor data with multiple sensors can be used to increase measurement accuracy.

Finally, the robot followed a square path by making approximately 90-degree left turns as required by the given tasks. The turning durations were adjusted proportionally to the surface conditions, ensuring accurate angular movement on different terrains. This allowed the robot to maintain a stable trajectory and successfully complete the designated path.