Robotics Laboratory 2

ROBOTICS AND EMBEDDED SYSTEM

Seth Paul C. Babayson
BSECE-4A
College of Engineering
Samar State University

***Abstract*—This experiment involves building a robot that detects and avoids obstacles using an ultrasonic sensor and motors controlled by an Arduino. The robot's movement is based on distance readings from the sensor, with the Arduino adjusting motor actions accordingly. A Webots simulation was used to test the robot's behavior, where it successfully avoided obstacles by stopping or changing direction when the sensor detected objects within a 30 cm range. The results showed effective obstacle avoidance, and adjustments to the motor speed and sensor threshold improved performance. This project demonstrates how sensor and motor integration can enable robots to navigate autonomously.**

I. Rationale

This experiment introduces the basics of robotics and embedded systems. It focuses on how sensors and motors can work together using a microcontroller like Arduino. With technology quickly advancing, It is important to understand how these systems control movement and respond to their surroundings. This activity helps students learn how to control motors, detect, and obstacles, similar to how real robots work in the real world.

According to Siciliano and Khatib (2016) in Springer Handbook of Robotics, sensor-actuator integration and embedded processing are critical for building intelligent robotic systems capable of interacting with dynamic environments. These skills form the basis of more complex tasks like autonomous navigation, path planning, and human-robot interaction. Furthermore, by simulating and testing the system in Webots, students gain both hardware and software experience in a safe, controlled, and repeatable virtual environment.

II. Objectives

A. Show the ability to control a DC motor using Arduino and a motor driver, adjusting at least three motor speeds with PWM control.

This objective focuses on controlling a DC motor using Arduino and a motor driver. And will use PWM (Pulse Width Modulation) to adjust the motor to at least three different speeds, demonstrating precise motor control.

Identify applicable funding agency here. If none, delete this.

B. Accurately measure the distance detected by an ultrasonic sensor within a ±5 cm range for distances between 10 cm and 200 cm.

Ultrasonic Sensor will be utilized to measure distances accurately within ±5 cm range, for objects located between 10 cm and 200 cm. This introduces sensor integration, read, process or interpret data (converting the time it takes for the sound wave to return into a distance), and making decisions and conditions in the code to avoid obstacles.

C. Program the robot to respond to ultrasonic sensor data and move forward or avoid obstacles, achieving a minimum of 90% success rate in a simulated Webots environment.

To program the robot to move forward or avoid obstacles based on distance reading from the ultrasonic sensor. The robot should achieve at least 90% success rate in obstacle avoidance during simulation in the Webots environment.

III. Materials and Software

- Hardware:
  - Arduino Uno R3
  - 4x DC Motor
  - 1x Servo Motor
  - L298N Dual H-Bridge Motor Driver Module
  - HC-SR04 Ultrasonic Sensor
  - Breadboard and Jumper Wires
  - 3x 3.7V Li-ion Battery
- Software:
  - Arduino IDE
  - Webots Simulator

IV. Procedures

- Motor Driver Setup
  - Each motor driver can accommodate 2 DC motors, since the objective requires at least 3 DC motors to operate/run, I used 4 dc motors and 2 motor drivers. So, Connect 2 DC motors to a motor driver and also connect the other 2 DC motor to the another motor driver.
  - Connect the 4 IN pins of 2 motor drivers to the digital pins of the arduino uno for direction control. Also, connect the ENA and ENB to the PWM pins for speed control

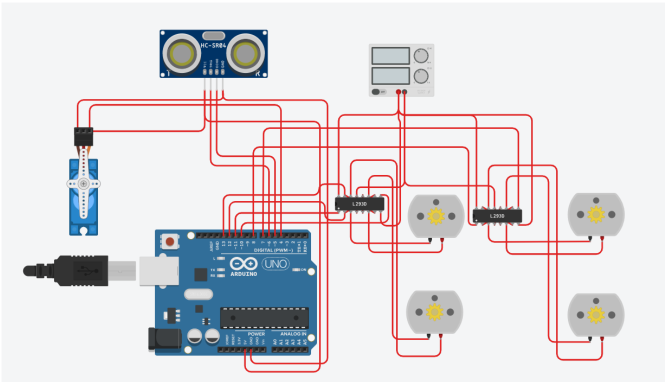

Fig. 1. Schematic Diagram Component Connections

TABLE I
MOTOR DRIVERS/ULTRASONIC SENSOR/SERVO MOTOR TO ARDUINO UNO CONNECTIONS

Components Pin Connections	
Motor Drivers/Ultrasonic Sensor/Servo Motor Pins	***Arduino UNO Pins***
IN2 & IN1	D13
IN4 & IN3	D12
ENA	D11
ENB	D10
IN1 & IN2	D8
IN3 & IN4	D7
TRIG Pin	D6
ECHO Pin	D5
DATA Pin(Servo)	D4

- Ultrasonic Sensor interface
  - Connect the VCC and GND of the HC-SR04 sensor.
  - Connect TRIG and ECHO pins to Arduino digital pins.
  - Implement the timing function to calculate distance based on wound wave travel.
- Servo Motor Setup:
  - Connect the servo motor to the Arduino Uno
- Arduino Programming
  - Declare all pins in the IDE, set pin mode of each pins as "OUTPUT" in void setup(), and in void loop(), all left wheels, IN (forward) are controlled by digital pin(D13) and IN (backward) pins are in digital pin(D8). Also, all right wheels, IN (forward) are controlled by digital pin(D12) and IN (backward) pins are conntrolled by digital pin(D7). For the EN pins, both ENAs and ENBs pin of 2 motor drivers are connected and also declared as D11 and D12 respectively in the .
  - TRIG and ECHO pin of the ultrasonic sensor are declared to digital pins (D6 and D7) of the arduino respectively. Setting the pin mode of the TRIG pin as "OUTPUT" and the ECHO pin as "INPUT" in the void setup().
  - and Data pin of the servo motor is declared to digital pin (D4) of the arduino. Setting the Data pin as "OUTPUT" in the void setup().

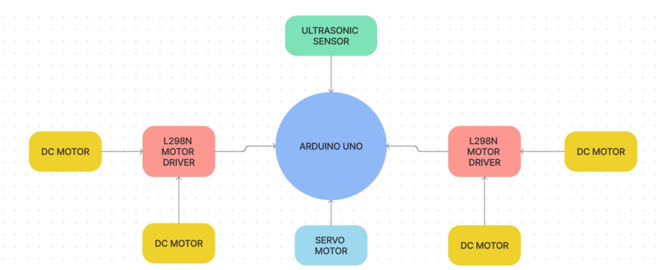

Fig. 2. Block Diagram of Connections of the Materials/Components

- Webots Simulation
  - Import the robot in Webots and use the simulated ultrasonic sensor and motors.
  - Upload the same logic programmed on Arduino into the Webots controller.
  - Run several trials and record the results of the simulation.

V. OBSERVATIONS AND DATA COLLECTION

In this section, data were observed and collected during the testing process.

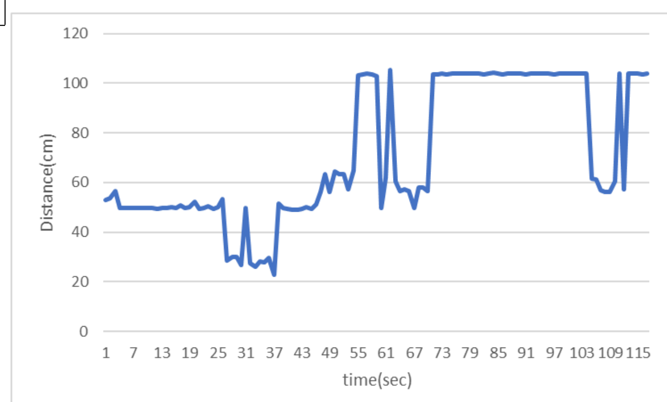

Fig. 3. 1st Trial of HC-SR04 Sensor Values (Actual)

As shown above, these are the sensor values or distance values in centimeters, where the nearest value was only above 20 cm and the farthest was below approximately 110cm. The purpose of the testing is to determine how far the ultrasonic sensor can detect to decide what threshold value must be declared for obstacle detection.

In Fig. 3, it shows that when the distance value approaches 200 cm, the sensor's readings are unstable. It fluctuates to approximately 1200 cm.
If the sudden fluctuations are removed, we can see the values clearly and closely within or near 200 cm. (See Fig. 4)

It shows (in Fig. 4) that the sensor can read and detect above 200 cm, but unstable. Within approximately 150-180

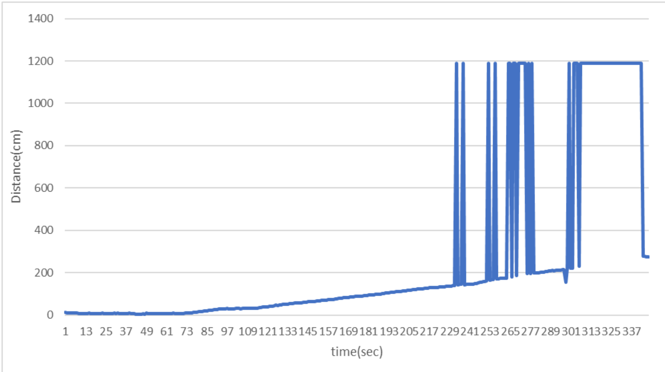

Fig. 4. Final Testing of HC-SR04 Sensor Values(Actual)

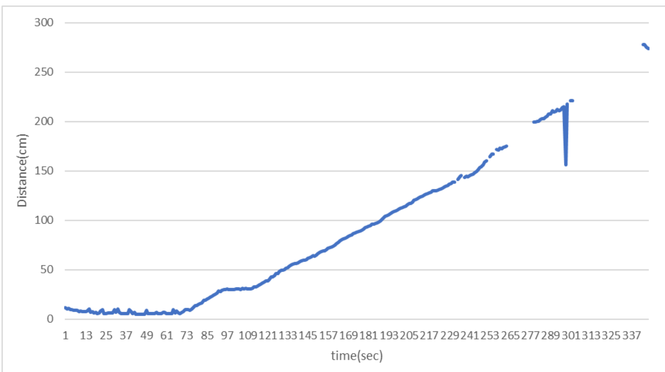

Fig. 5. Removed fluctuated wave

cm is the farthest distance value that can be detected accurately.

With that range, we can now set the threshold values for the obstacle detection. As for the threshold, values such as 10 cm, 30 cm, and 50 cm are individually declared and tested during the testing process. And it seems that the 30 cm threshold is sufficient. As observed in the 10 cm threshold, sometimes the robot is less responsive or has a delay in detecting and stopping the motors. In a 50 cm threshold, it is too sensitive in detecting obstacles, even when there is no obstacle in front.

Integrating the servo motor with the ultrasonic sensor, the servo motor angles from 0 to 180 degrees and back to 0 to detect not only in front, but also the side fronts. At first testing, the servo speed was assumed by 20 and the servo step was also assumed by 1. It was observed that the time it takes to complete the 180 degrees from 0 degrees was too slow because it only increments by 1 degree at a time. It would be better to increase the servo step to also increase the increment of the servo step. That's why the servo step was adjusted by 10. And the goal is to turn the servo from 0 to 180 degrees, and back from 180 to 0 degrees for around 1 to 2 seconds. (See calculation in Data Analysis)

Theoretically, the full sweep of the servo motor must be 0.72 seconds. However, considering the friction and the weight of the ultrasound sensor on top of the servo motor, it was recorded by the stopwatch, and its full sweep is approximately around 1.5 to 2 seconds.

The DC motor is controlled by the L298N motor driver in speed and direction. There are 3 movement states of the robot. It includes forward, backward, stop, and turning.

The forward function, DC motors have the same direction and speed of 130(PWM value) was declared in the program. The backward function has same functionality as forward, but it opposes the direction of the forward function, and the speed is decreased by 70(PWM value). The turn function has a different function compared to the two movement states mentioned. The wheels on the left side have a clockwise direction, and the wheels on the right side have a counterclockwise direction, but both sides have the same speed by 80(PWM value).

In the Webots simulation, a robot was tested to see how well it could avoid obstacles using two front sensors named `ps0` and `ps7`. These sensors measured the distance to objects in front of the robot. If either sensor gave a reading above 80, the robot would move backward for a short time and then turn to avoid the obstacle. The direction of the turn depended on which sensor had the higher value, turning away from the closer object.

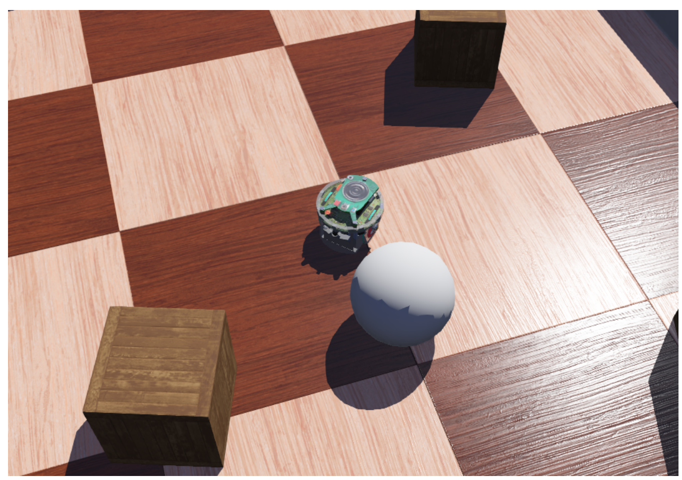

Fig. 6. Virtual Robot Simulation for obstacle detection and avoidance

During the simulation, the robot moved around a virtual environment with walls and obstacles. The sensor readings changed depending on how close the robot got to objects. When the path was clear, the readings were usually below 50. When an object was near, the readings went above 80, and the robot reacted by reversing and turning. This behavior happened multiple times, and the robot was able to avoid all obstacles without crashing.

VI. Data Analysis

Sweep time calculation of the servo motor.

$$\begin{aligned}
\Delta\theta &= 10^\circ \\
D &= 20\,\text{ms} \\
\theta_{\text{min}} &= 0^\circ, \quad \theta_{\text{max}} = 180^\circ \\
N &= \frac{\theta_{\text{max}} - \theta_{\text{min}}}{\Delta\theta} = \frac{180^\circ - 0^\circ}{10^\circ} = 18 \\
T_{\text{total}} &= 2 \cdot N \cdot D = 2 \cdot 18 \cdot 20\,\text{ms} = 720\,\text{ms} = 0.72\,\text{seconds}
\end{aligned}$$

$$\begin{aligned}
\text{where:} \quad D &= \text{Delay per step (ms)} \\
\Delta\theta &= \text{Servo step size (degrees)} \\
N &= \text{Number of steps} \\
T_{\text{total}} &= \text{Total back-and-forth sweep time}
\end{aligned}$$

The servo speed was adjusted by 20, and the step was incremented by 10, so it can rotate back and forth in a 180-degree range in 0.72 seconds or approximately 1 second.

Ultrasonic Sensor Distance Value for obstacle detection.

Time#	Distance (cm)	Action
1	24.13	Obstacle Detected
2	25.26	Obstacle Detected
3	26.12	Obstacle Detected
4	28.31	Obstacle Detected
5	28.02	Obstacle Detected
6	28.90	Obstacle Detected
7	29.70	Obstacle Detected
8	29.96	Obstacle Detected
9	30.48	Move Forward
10	30.06	Move Forward
11	29.79	Obstacle Detected
12	29.55	Obstacle Detected
13	29.53	Obstacle Detected
14	30.58	Move Forward
15	30.48	Move Forward
16	30.73	Move Forward
17	29.70	Obstacle Detected
18	30.99	Move Forward
19	30.48	Move Forward
20	31.25	Move Forward
21	30.48	Move Forward
22	30.58	Move Forward
23	30.30	Move Forward
24	31.08	Move Forward
25	32.53	Move Forward
26	32.55	Move Forward
27	33.32	Move Forward
28	34.71	Move Forward
29	35.40	Move Forward
30	36.58	Move Forward
31	37.27	Move Forward
32	38.42	Move Forward
33	38.60	Move Forward
34	40.13	Move Forward
35	42.31	Move Forward
36	42.72	Move Forward
37	44.30	Move Forward
38	46.00	Move Forward
39	45.81	Move Forward
40	47.80	Move Forward

TABLE II
Ultrasonic Sensor Distance Readings and Robot Response

The values above show the distance values of the ultrasonic sensor and the response of the robot. Since the threshold value was set to 30 cm, the robot must respond stop, backward, and turning when it detects less than or equal to 30 cm distance value.

To analyze the performance of the ultrasonic sensor in obstacle detection, we divided the raw distance data into two categories:

- **Obstacle Detected:** Distance $< 30\,\text{cm}$
- **Move Forward:** Distance $\geq 30\,\text{cm}$

We then calculated the mean distance for each category using the following formula:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

where:

- $\bar{x}$ is the mean (average) distance
- n is the number of distance readings
- x_i is the i-th distance reading

Obstacle Detected Group

$$\begin{aligned}
\text{Values: } & 24.13,\ 25.26,\ 26.12,\ 28.31,\ 28.02,\ 28.90,\ 29.70, \\
& 29.96,\ 30.48,\ 30.06,\ 29.79,\ 29.55,\ 29.53 \\
n &= 13 \\
\sum x_i &= 363.47\,\text{cm} \\
\bar{x}_{\text{obstacle}} &= \frac{363.47}{13} = 27.96\,\text{cm}
\end{aligned}$$

Move Forward Group

$$\begin{aligned}
\text{Values: } & 30.99,\ 30.48,\ 31.25,\ 30.48,\ 30.58,\ 30.30 \\
& 31.08,\ 30.58,\ 30.48,\ 30.73,\ 32.53,\ 30.48, \\
& 30.06,\ 32.55,\ 33.32,\ 34.71,\ 35.40,\ 36.58, \\
& 37.27,\ 38.42,\ 38.60,\ 40.13,\ 42.31,\ 42.72, \\
& 44.30,\ 46.00,\ 45.81,\ 47.80 \\
n &= 28 \\
\sum x_i &= 1034.45\,\text{cm} \\
\bar{x}_{\text{forward}} &= \frac{1034.45}{28} = 36.94\,\text{cm}
\end{aligned}$$

To analyze the performance of the ultrasonic sensor in detecting obstacles, the distance readings were divided into two movement states based on the robot's response: "Obstacle Detected" for distances less than or equal to 30 cm, and "Move Forward" for distances greater than 30 cm. For each movement states, the mean distance was calculated using the standard formula for the arithmetic mean representing each reading.

In the Obstacle Detected group, there were 13 recorded values partially with a total sum of 363.47 cm. This results in an average distance of approximately 27.96 cm. This indicates that the robot correctly recognized nearby obstacles when the measured distance was around this average, which is below the 30 cm threshold. On the other hand, the Move Forward group had 28 distance values with a total sum of 1034.45 cm. The calculated mean for this group is approximately 36.94 cm, showing that the robot generally continued moving forward when the obstacle was safely beyond 30 cm.

DC Motor Analysis

Motor Configuration

The robot uses 4 DC motors controlled by the 2 L298N motor drivers. Speed is adjusted using PWM (Pulse Width Modulation) signals via the analogWrite() function in Arduino connected to ENA and ENB of motor drivers. The PWM values used in this project are as follows:

- Forward motion: PWM = 130
- Backward motion: PWM = 70
- Turning: PWM = 80

PWM Duty Cycle Calculation

PWM in Arduino uses an 8-bit resolution, meaning the maximum value is 255. The duty cycle determines how much power is delivered to the motor. It is calculated using the formula:

$$\text{Duty Cycle (\%)} = \left(\frac{\text{PWM Value}}{255}\right) \times 100$$

Duty Cycle Values

$$\begin{aligned}
\text{Forward:} \quad & \left(\frac{130}{255}\right) \times 100 = 50.98\% \\
\text{Backward:} \quad & \left(\frac{70}{255}\right) \times 100 = 27.45\% \\
\text{Turning:} \quad & \left(\frac{80}{255}\right) \times 100 = 31.37\%
\end{aligned}$$

These duty cycle values represent the amount of time or percentage of the motor receiving power in each PWM cycle. The higher the duty cycle, the higher the speed. During forward motion, both motors receive approximately 51% power, providing higher speed. For other movement states such as backward and turning, the power is reduced to around 27% and 31%, respectively. This helps the robot slow down when avoiding obstacles or changing direction. These speed settings were chosen to balance how it responds and to stabilize the movements of the robot.

Webots Simulation Analysis

The sensor values were analyzed to check how well the robot detected obstacles. Any reading above 80 meant there was something in front of the robot. The average reading when an obstacle was detected was about 93.4 for `ps0` and 91.7 for `ps7`. This shows that the robot correctly identified nearby objects using the set limit of 80.

When no obstacle was in the way, both motors moved the robot forward at a speed of $0.3 \times 6.28 = 1.884$ radians per second. If an obstacle was detected, the robot moved backward for about 640 milliseconds and then turned either left or right for another 640 milliseconds. This process helped the robot avoid getting stuck. The results showed that the robot reacted correctly each time, and the program worked well in helping it move safely in the simulation.

VII. Discussion and interpretations

The main objective of this experiment was to develop a robot that avoids obstacles using an ultrasonic sensor for distance detection and DC motors for both direction and speed movement. A servo motor was also employed to sweep the ultrasonic sensor from $0°$ to $180°$, enhancing the field of view of the robot. The actual results demonstrate that the robot was able to detect obstacles within a certain distance threshold and respond by stopping, reversing, and turning to avoid collisions.

The ultrasonic sensor data was analyzed by grouping distance readings into two movement states: obstacle detected ($< 30\,\text{cm}$) and move forward ($\geq 30\,\text{cm}$). The mean distance in the obstacle detected state was calculated to be approximately $27.96\,\text{cm}$, while the forward state had a mean of $36.94\,\text{cm}$. These values support the chosen threshold of $30\,\text{cm}$. It confirms that the behavior of the robot aligns the expectations based on sensor response for decision-making.

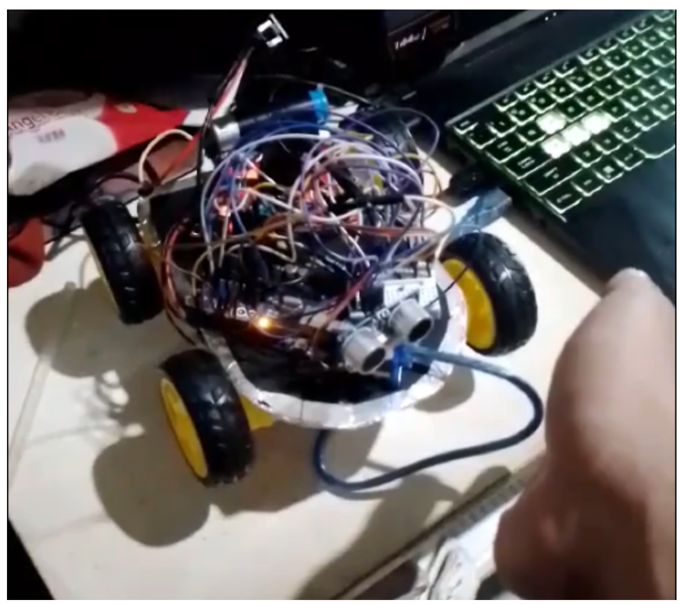

Fig. 7. Actual Testing of obstacle detection using Ultrasonic Sensor with continuous 180 degree scanning rotation of servo motor, and changing speed and direction of DC motors

However, some discrepancies were observed between expected and actual behaviors, such as inconsistent obstacle detection at the boundary of the threshold. Errors include:

- Inaccuracy or noise in ultrasonic sensor readings, especially on soft or angled surfaces.
- Servo motor jitter, or errors timing that may affect scanning accuracy.
- Variations in motor speed due to changes in battery voltage or friction.

The servo sweeping function was designed to run on every reading of ultrasonic sensor with declared value of step size, completing a full sweep every 0.72 seconds (or approximately 1.5 to 2 seconds in actual sweeping output) with a step size of $10°$ and a delay of $20\,\mathrm{ms}$ per step. This allowed for continuous scanning, but also introduced potential timing conflicts if the sweep and movement decisions were not well synchronized.

Motor movement was set through fixed PWM duty cycles on every movement state. While effective for basic motion, this method lacks feedback control, which limits precision and adaptability to terrain changes.

VIII. Conclusion

Overall, the experiment met its objectives, showing a functioning robot that avoids obstacles. The findings confirmed that using an ultrasonic sensor with an integrated moving servo for wider detection and supporting the logic for decision used for motion control was effective. The limitations include the precision of the sensor and obstacle considerations such as curved/angled walls or edges. For recommendations, further improvements could include speed feedback using encoders, or implementing PID controller for more accurate motion.

References

[1] (Siciliano, B., & Khatib, O. (Eds.). (2016). Springer Handbook of Robotics (2nd ed.). Springer.)

IX. Appendix

https://github.com/seth-paul/Elective-2-Robotics-Technology/tree/main/Laboratory%202