L-Università ta' Malta
Faculty of Information & Communication Technology

A Smart Robot Car Project

Matthias Bartolo

B.Sc. It (Hons) Artificial Intelligence (Third Year)

Study Unit Code:	**ARI2202**
Study Unit:	**Robotics 1**
Lecturer:	**Dr Ingrid Galea**
Date:	**01 June 2024**
Link:	https://github.com/mbar0075/Robotics-1

Acknowledgements

I wish to express my sincere gratitude to Dr. Ingrid Galea, who, despite not being directly involved in supervising this project, generously provided me with the necessary support and resources, enabling me to meet the challenges and successfully complete it. Her proactive involvement greatly contributed to the project's success.

Contents

1 Introduction

1.1 Definition of “Robot”

Within popular culture, the term “robot” often evokes imagery of anthropomorphic features, such as robotic “arms” employed in welding tasks. The tendency to assign human-like traits to robots may find its roots in the origin of the term “robot”, coined by a Czech playwright in 1920. This term, derived from the Czech word “robota” meaning forced labour, suggests a historical association between robots and laborious tasks in his plays [1]. This inclination is in alignment with the definition provided by the Robot Institute of America, which widely acknowledges a robot as a “reprogrammable, multifunctional manipulator designed to move material, parts, tools, or specialized devices through variable programmed motions for the performance of a variety of tasks” [2]. The evolution from human-like mechanical constructs to a broader range of forms suitable for diverse tasks reflects the pragmatic realities of robotics. While inherently mechanical, robots need not adhere to anthropomorphic or even animalistic designs [1].

1.2 Components of a Robotic System

A robotic system’s three main components as illustrated in Figure 1.1, are **sensors**, **actuators**, and **controllers**. These components comprise the backbone of the system’s planning process, allowing it to interact with the environment [1]. Sensors act as the

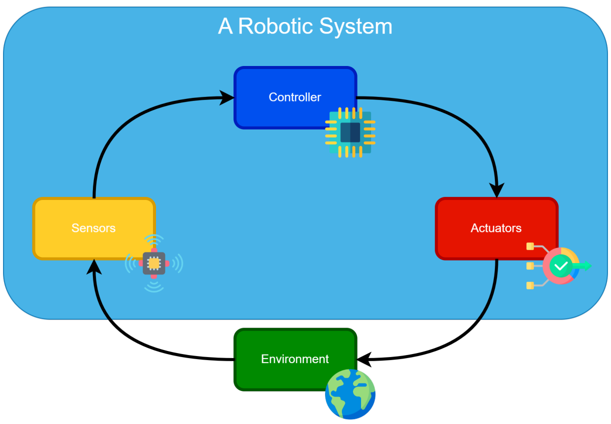

Figure 1.1 Components of a robotic system.

robot’s sensory organs, collecting data from the surroundings using a variety of techniques such as cameras, lidars, and proximity sensors. Actuators, on the other hand, execute physical actions depending on controller instructions. These actions might be as basic as spinning a wheel or as sophisticated as grasping an item. The controllers serve as the operation's brain, analysing sensor data, making judgements, and issuing orders to actuators to complete tasks. Together, these components interact dynamically with their surroundings, constantly collecting feedback from sensors, analysing information, and modifying actions accordingly [1].

1.3 Fundamental Laws of Robotics

Isaac Asimov outlined the ”Three Fundamental Laws of Robotics,” which serve as foundational ethical guidelines for the behaviour of robots. These guidelines pertain to:

1. A robot may not injure a human being or, through inaction, allow a human being to come to harm.
2. A robot must obey the orders given by human beings, except when such orders would conflict with the first law.
3. A robot must protect its own existence, as long as such protection does not conflict with the first or second law.

2 Robot Assembly and Basic Movement

2.1 Robot Description and Assembly

The smart robot car depicted in Figure 2.1 represents the assembled robotic platform utilised for this project. This robot consists of a four-wheeled motorised chassis, incorporating sensory components such as an ultrasonic (sonar) sensor and a camera module, both of which are mounted on servo motors to enable directional movement. The system is integrated on an Arduino Uno board with an IO Expansion Board, enhancing its connectivity and functionality. Additionally, the robot features a line tracking module, enabling preset modes including line-tracking, obstacle avoidance, and auto-follow functionalities. Additionally, the robot's capabilities, including real-time detection and measurement of orientation, rotation, and angular velocity, are enhanced by the inclusion of the MPU6050 gyro module. Power is supplied by a rechargeable lithium battery, ensuring sustained operation of the system.

Figure 2.1 ELEGOO Smart Robot Car Kit V4.0.

The assembly process commenced with the attachment of motors using the provided screws, ensuring each was securely fastened to the robot's bottom plate following the assembly instructions provided with the kit. Following this, the line-tracking module was affixed beneath the bottom plate and firmly secured. Subsequent steps involved the integration of chips into the system: first, the MPU6050 gyro chip was connected to the expansion board, followed by the Arduino Uno chip, all of which were affixed to the robot's top plate. The lithium battery was then securely mounted onto the top plate. Assembly of the camera module entailed connecting the camera to its bracket, while the ultrasonic sensor was connected to the servo module; both components were then joined and fixed onto the robot's top plate. Next, the various robot components were connected to the IO Expansion board using a set of wires, following the kit's instructions. Finally, the top and bottom plates were joined together using a set of six copper cylinders. To complete the assembly, the wheels were connected to the robot and firmly screwed onto the motors. Figure 2.2 illustrates the labelled components of the aforementioned smart robot car.

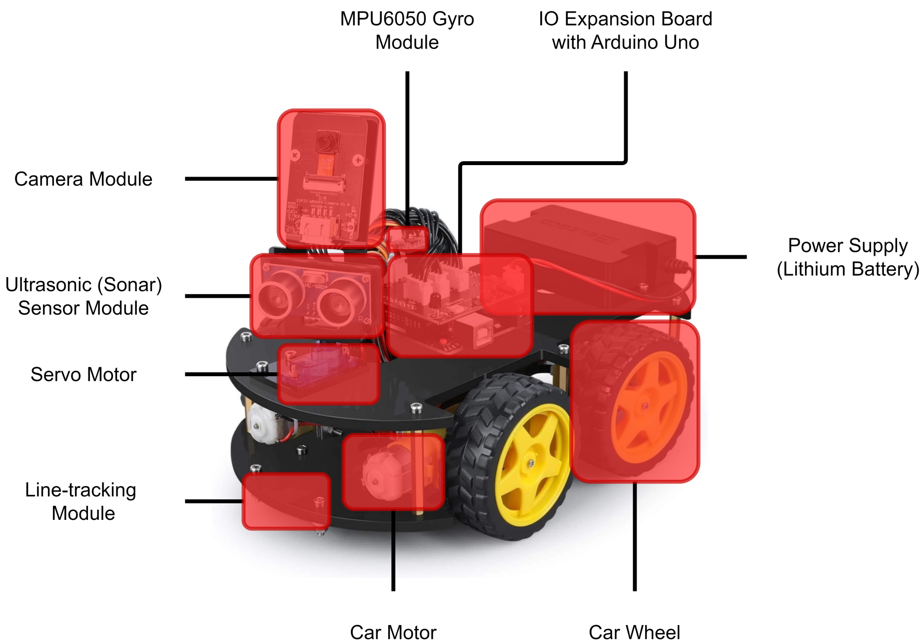

Figure 2.2 Labelled components of the ELEGOO Smart Robot Car Kit V4.0.

2.2 Software Installation

In the software installation process, the initial step involved installing the Arduino IDE, followed by downloading the system through the installation setup wizard. Subsequently, the specified zip libraries from the robot kit's package files were added to the system. These libraries, namely:

1. **FastLED**
2. **I2Cdev**
3. **IRremote**
4. **NewPing**
5. **Pitches**

were integrated into the system from the Sketch tab by selecting “Include Library” and then “Add .ZIP Library”. Upon the successful addition of these libraries, the robot was connected to the computer via the provided USB port in the kit. From the Arduino IDE, the current board was configured as “Arduino Uno” and connected to the port of “COM3”, that corresponded to the port to which the robot was attached.

2.3 Programming Basic Movement

To enable fundamental locomotion, a program was devised to command the robot’s motion: moving forward for 5 seconds, then backward for 5 seconds, before coming to a halt. This functionality was achieved through encoding and uploading specific commands to the Arduino Uno board. Within the `setup` function, critical components and configurations essential for robotic operation were initialised. This entailed establishing serial communication, defining pin assignments for motors, tracking sensors, and an ultrasonic sensor, and setting their respective modes. Moreover, the accelerometer and gyroscope components underwent calibration, while the I2C communication bus was initialised. Additionally, the servo mechanism was calibrated to orient the robot forward, with its current angle recorded for reference. Although the sensors and servo motor were not utilised in this step, it is generally well-known practice to define these in the `setup` function. Within the `loop` function, the robot executed a predefined sequence of actions. It began by retrieving its current orientation using the MPU6050 sensor, then propelled forward for 5 seconds via the `fwd` function. Following a brief pause, it reversed direction for 5 seconds using the `bwd` function, culminating in a 10-second stationary interval before repeating the cycle. Further details regarding the `fwd` function can be found in Listing 2.1 for additional clarity and reference.

```
void fwd(uint8_t right_motor_speed, uint8_t left_motor_speed)
{
  /* Function to move the robot forward.
   Parameters:
      right_motor_speed: Speed for the right pair of motors.
      left_motor_speed: Speed for the left pair of motors.

    Digital signals:
      HIGH: represents the digital signal of 1.
      LOW: represents the digital signal of 0.
  */
  // Setting the direction of the motors.
  digitalWrite(PIN_Motor_AIN_1, HIGH);
  digitalWrite(PIN_Motor_BIN_1, HIGH);
```

```

  // Setting the speed of the motors.
  analogWrite(PIN_Motor_PWMB, left_motor_speed);
  analogWrite(PIN_Motor_PWMA, right_motor_speed);

  // Enabling the motors.
  digitalWrite(PIN_Motor_STBY, HIGH);
}
```

Listing 2.1 Function encoding forward movement of the robot.

3 Ultrasonic Sensor and Obstacle Detection

3.1 Ultrasonic Sensor Analysis

In this experiment, the primary focus was dedicated to scrutinising the functionality of the Ultrasonic Sensor. In pursuit of this objective, a series of observations were recorded. The experimental setup involved investigating the sensor's response to dynamic stimuli, particularly by placing an object (which can be detected by the sensor) in front of the sensor and systematically varying its proximity to the robot in order to discern whether it is detected or not. Notably, the Ultrasonic Sensor demonstrated the capability of registering the furthest obstacle at a distance of 1183 centimetres, while discerning objects as close as 2 centimetres as illustrated in Figure 3.1.

```
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 704 Echo Pin: 704 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 706 Echo Pin: 706 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 468 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 704 Echo Pin: 704 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 704 Echo Pin: 703 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 470 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 703 Echo Pin: 703 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 471 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 698 Echo Pin: 697 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 702 Echo Pin: 701 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 702 Echo Pin: 701 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 702 Echo Pin: 701 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 703 Echo Pin: 702 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 702 Echo Pin: 702 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 470 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 704 Echo Pin: 704 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 704 Echo Pin: 704 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 471 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 704 Echo Pin: 703 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 470 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 703 Echo Pin: 703 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 703 Echo Pin: 703 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 702 Echo Pin: 701 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 704 Echo Pin: 703 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 704 Echo Pin: 704 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 468 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 703 Echo Pin: 703 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 469 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 703 Echo Pin: 703 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 470 cm
Sensor Angle: 115 Sensor FOV: 230 --- Trigger Pin: 703 Echo Pin: 703 --- Max Distance: 1183 cm Min Distance: 2 cm--- Distance: 471 cm
```

Figure 3.1 Serial Monitor.

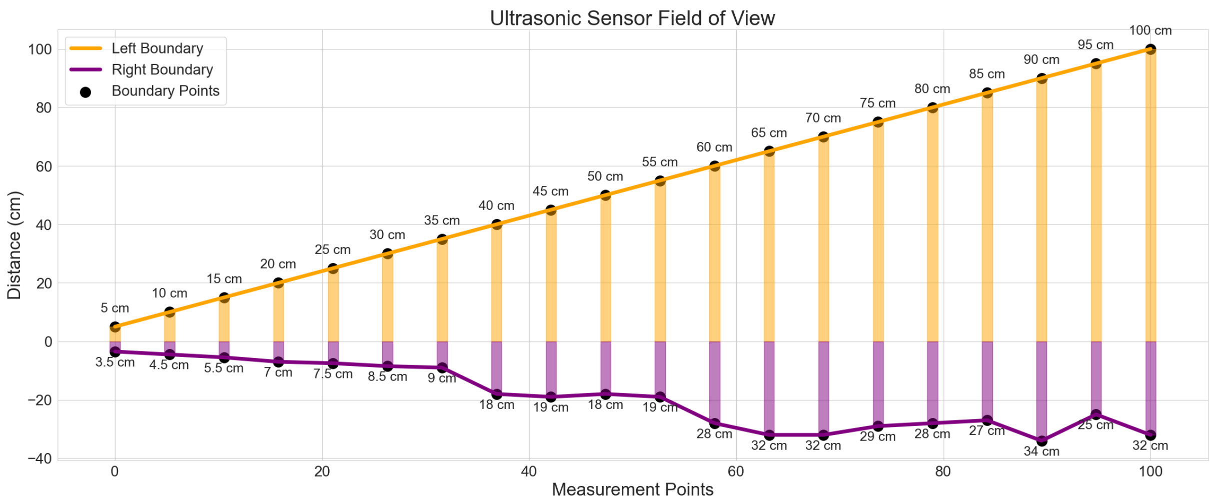

Figure 3.2 Ultrasonic (sonar) sensor field of view (FOV).

Additionally, Figure 3.2 represents the Ultrasonic Sensor's field of view (FOV), where the cone-shaped region illustrates the sensor's detection range and coverage, with areas not marked in the plot also representing blind spots.

3.2 Basic Obstacle Detection

This experiment encompassed the development of a rudimentary obstacle detection algorithm, wherein the robot was programmed to advance for a duration of 10 seconds but promptly halt upon detecting an obstacle within a 20-centimetre proximity of the Ultrasonic Sensor. The procedural workflow is depicted in Figure 3.3, wherein the initial image portrays the robot's forward movement towards the obstacle, while the subsequent image illustrates the robot's cessation upon detecting an obstacle within the prescribed 20-centimetre range. Furthermore, observations revealed inherent constraints regarding the positioning of the ultrasonic sensor on the robot. Notably, due to its elevated placement as recommended by the provided kit components, certain obstacles were not readily detected. This limitation stemmed from the necessity for both the transmitter and receiver to directly face the detected obstacle for optimal sensor functionality, highlighting the practical challenges inherent in deploying the robot within complex environments characterised by diminutive obstacles. It is essential to underscore that these limitations primarily originate from hardware constraints rather than programming deficiencies.

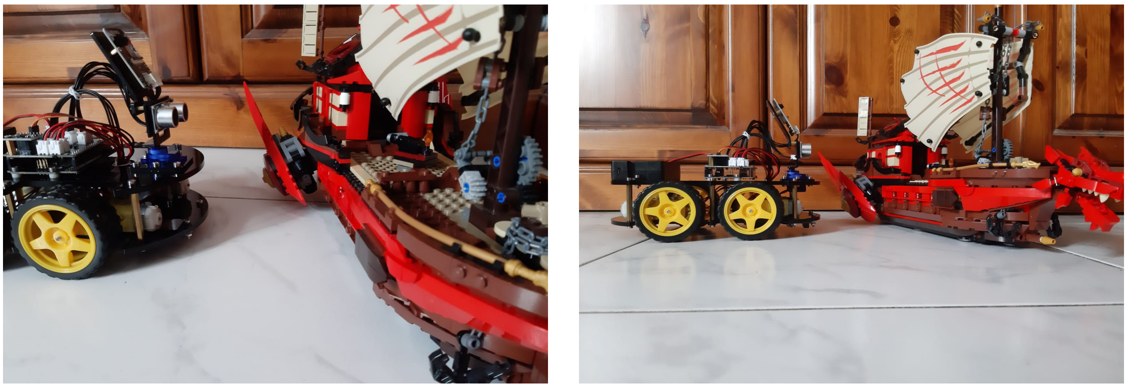

Figure 3.3 Obstacle Detection.

3.3 Basic Obstacle Avoidance Algorithm

This experiment progressed from the preceding phase of obstacle detection to the subsequent stage of evading the identified objects. The algorithm devised for obstacle avoidance drew inspiration from the pre-existing algorithm packaged with the robot kit, albeit with enhancements introducing a degree of randomness in robot movement.

This refined algorithm is detailed in Algorithm 1. Additionally, the implementation of obstacle avoidance is depicted in Figure 3.4.

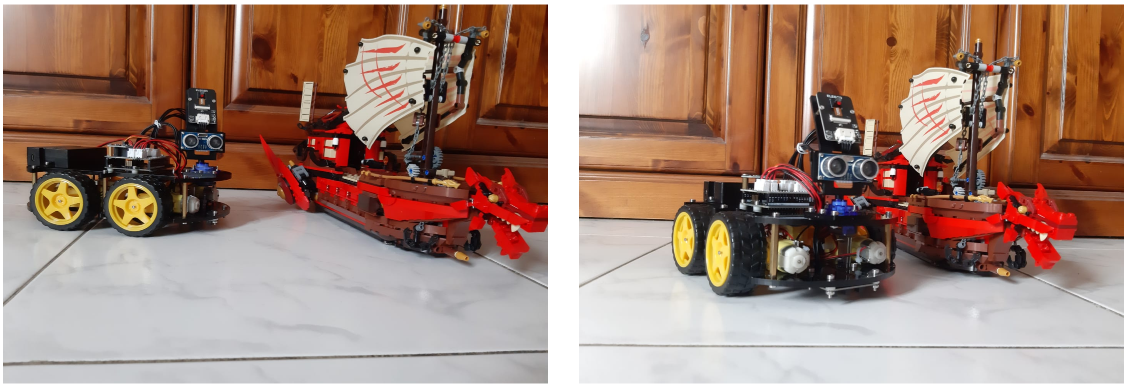

Figure 3.4 Obstacle Avoidance.

In the initial image, the ultrasonic sensor detects the obstacle, prompting the servo motor to tilt the sensor to the right. Subsequently, in the succeeding image, the robot manoeuvres in the direction of the tilted angle, effectively circumventing the obstacle and averting collision. Furthermore, it is notable that the challenge of sensor positioning manifests in certain scenarios, wherein diminutive obstacles fail to prompt the robot to halt, resulting in various collisions. Amidst the algorithm's development, a persistent challenge pertained to coordinating the movement of the servo motor and determining optimal angles. This hurdle was surmounted through observation of the chosen hyper-parameters by the ELEGOO kit, which demonstrated smooth and fluid movement as the robot adjusted its angle. Additionally, the current algorithm incorporates a slight reversal of the robot's movement before executing a turn, thereby ensuring a wide arc of rotation without collision. Moreover, a robot speed of 100 units was chosen for this avoidance manoeuvre, with the closest distance for obstacle detection set at 20 units to afford the robot ample time for halting, reversing, and evading the obstacle effectively.

Algorithm 1 Obstacle Avoidance.

```
procedure obstacle_avoidance
    if obstacle detected then
        Stop the robot
        for angles: 30, 90, 150 do
            Scan with servo at angle
            if obstacle still detected then
                Stop the robot
                if reached maximum angle then
                    Move backward and turn randomly
                    Break loop
                end if
            else
                Move backward and according to the angle
                Break loop
            end if
        end for
    else
        Move forward
    end if
end procedure
```

References

[1] R. Murphy, *Introduction to AI Robotics*, 2nd. London: The MIT Press, 2019.

[2] D. M. Considine and G. D. Considine, “Robot technology fundamentals,” in *Standard Handbook of Industrial Automation*, D. M. Considine and G. D. Considine, Eds. Boston, MA: Springer US, 1986, pp. 262–320, isbn: 978-1-4613-1963-4. doi: 10.1007/978-1-4613-1963-4_17.

Plagiarism Form

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Declaration

Plagiarism is defined as “the unacknowledged use, as one’s own work, of work of another person, whether or not such work has been published” (Regulations Governing Conduct at Examinations, 1997, Regulation 1 (viii), University of Malta).

I / ~~We~~* , the undersigned, declare that the [assignment / Assigned Practical Task report / Final Year Project report] submitted is my / ~~our~~* work, except where acknowledged and referenced.

I / ~~We~~* understand that the penalties for making a false declaration may include, but are not limited to, loss of marks; cancellation of examination results; enforced suspension of studies; or expulsion from the degree programme.

Work submitted without this signed declaration will not be corrected, and will be given zero marks.

* Delete as appropriate.

(N.B. If the assignment is meant to be submitted anonymously, please sign this form and submit it to the Departmental Officer separately from the assignment).

______________________	______________________
StudentName	Signature
______________________	______________________
StudentName	Signature
______________________	______________________
StudentName	Signature
Matthias Bartolo	
StudentName	Signature

ARI2202	A Smart Robot Car Project
CourseCode	Title of work submitted

01 June 2024
Date