Self-Driving Bot
Project 1-1. Computer Science

Department of Advanced Computing Sciences
Maastricht University

Contents

1 Self-Driving Bot

In this project, "Self-Driving Bot," the task involves developing a robot that can move autonomously, follow lines, detect obstacles, and eventually solve complex navigation tasks such as mazes and obstacle tracks. The robot will also be controlled and monitored through a Java-based graphical interface.

The project integrates several concepts from both software and hardware domains. By the project's conclusion, you will have a deeper understanding of how hardware components can be programmed and controlled to perform complex tasks autonomously. This knowledge is vital in today's technological landscape, where automation and robotics are increasingly prevalent in various industries.

2 Project Description

This project entails the creation of a self-driving bot capable of autonomously navigating a track, solving mazes, performing manoeuvrers, and being controlled remotely. The project aims to enhance understanding of integrating hardware with software, sensor integration, movement algorithms, and the development of user interfaces for remote control and data visualization.

2.1 Background

Autonomous robots have become essential in modern technology, contributing significantly to fields such as industrial automation, medical robotics, and advanced research. The project introduces fundamental principles of robotics, focusing on movement algorithms, sensor integration, and user interface development. The self-driving bot will simulate real-world applications of autonomous navigation systems, such as those found in self-driving cars and automated guided vehicles (AGVs).

Historically, the development of autonomous systems has evolved from simple programmed instructions to complex algorithms capable of making real-time decisions based on sensor inputs. This evolution is driven by advancements in microcontroller technology, sensor accuracy, and algorithmic efficiency. Understanding these components and their integration is crucial for anyone aiming to delve into the field of robotics and automation.

2.1.1 Project Objectives

1. Wire a robot capable of basic movement.
2. Implement line-following functionality using infrared (IR) sensors.
3. Integrate an ultrasound sensor for obstacle detection and emergency stops.
4. Develop and improve a Java-based graphical user interface (GUI) for remote control and data visualization.
5. Enable the robot to solve mazes, race around a track, find itself after getting lost, and demonstrate special manoeuvrers.

2.1.2 Equipment

For this project, we will provide you the following equipment:

- **Pre-assembled Robot chassis** — base structure for motors, sensors, and components
- **Motors** — provide movement capabilities
- **Infrared (IR) sensors** — to detect and follow lines on the ground

- **Ultrasound distance sensor** — detects obstacles in front of the robot
- **Adafruit microcontroller** — the brain of the robot
- **Batteries and connectors** — power supply for components
- **Miscellaneous assembly tools** — wires, flathead screwdriver

The Motors, IR sensors, Ultrasound sensors, Adafruit, and batteries will already be mounted on the robot chassis. Additionally the main electrical wiring with the battery will already be set up. You will **only** be expected to wire the motors and sensors to the microcontroller.

2.2 Phases of the Project

2.2.1 Phase 1: Wiring and Basic Movement

This phase focuses on wiring the robot correctly, and achieving basic motion. The following tasks should be fulfilled. The minimal passing requirements are points 1, 2, 3 and 4:

1. **Wire the robot** and understand why the wiring works (or doesn't work).
2. **Make the robot move** using provided boilerplate code, ensure you implement the following motions:
   - Drive forward
   - Stop all motors
   - Drive backward
   - Drive left
   - Drive right
   - Turn on the spot (right)
   - Turn on the spot (left)
3. **Set up WiFi connection** using the provided code on canvas.
4. **Implement Java code** that controls the motion of the robot in multiple directions through terminal or a GUI.
5. **Research** methods to do line following in preparation of phase 2.

2.2.2 Phase 2: Line Following and Obstacle Detection

This phase introduces sensor-based movement and expands the GUI for real-time interaction. The following tasks should be fulfilled. The minimal passing requirements are point 1 and 3:

1. **Implement line following** using IR sensors and normal black tape. The robot should also be able to follow lines. (data preprocessing: what happens if the sensors give incorrect data, you need to deal with that)
2. **Emergency stop:** implement logic to stop the robot if the ultrasound sensor detects an obstacle ahead.
3. **Implement a GUI** with features such as:
   - Directional control
   - A plotted map of the line or path the robot follows
   - Display of useful statistics from the robot (e.g., sensor data, distance, performance metrics)

2.2.3 Phase 3: Maze Solving, Drawing, and Advanced Manoeuvrers

This is the most advanced phase, requiring autonomous problem-solving and creative tasks. The following tasks should be completed. The minimal passing requirements are points 1 and 4.

1. **Maze Solving:**
   - Store the path the robot has followed
   - Trace steps back when necessary
   - Implement an algorithm to solve the maze efficiently
2. **Race Track:**
   - Complete a track as fast as possible
   - Implement an algorithm that follows a line as fast as possible.
3. **Kidnapped Robot Problem:**
   - Place the robot away from the line.
   - Have it find it's way back to a line.
4. **Special manoeuvres:**
   - Implement at least **three special manoeuvres**
   - During evaluation, examiners will select three manoeuvres for your group to demonstrate

2.3 Final Demonstration

At the end of the project, each group will demonstrate their robot completing the following:

- Maze solving
- Race Track
- Lost Robot
- Three Special Manoeuvres

This final demonstration will showcase the robot's full range of capabilities, from fundamental wiring to advanced autonomous navigation and control.

2.4 Appendix

2.4.1 Special Manoeuvrers

The following manoeuvrers are part of the Dutch driving test and serve as inspiration for the project's advanced tasks. Examiners will select three for your robot to demonstrate.

1. Reversing in a Straight Line (Recht Achteruit Rijden)
2. Reversing Around a Corner (Bocht Achteruit Rijden)
3. Three-Point Turn (Keren Door Middel van Steken)
4. U-Turn (Halve Draai)
5. Parking in a Box Forward (Vooruit Inparkeren)
6. Emergency Stop

2.4.2 Car Usage

For this project we are loaning to you a robot car. We expect you to treat it with care.

General Responsibilities:

- You are responsible for the robot car for the duration of the project.
- Handle the car and all accessories carefully to avoid damage.
- Do not attempt to disassemble or modify parts of the car unless explicitly instructed.

Usage Guidelines:

- Use the robot car only for project-related activities.

- Operate it in safe environments (e.g., flat indoor surfaces or designated test areas).
- Avoid water, excessive dust, or environments that could damage electronics.

Care and Maintenance:

- Always switch off the car when not in use.
- Store it in a safe place to prevent falls or accidental damage.
- Regularly check that all components (wheels, sensors) are secure.

Return Policy:

- At the end of the project, you must return the robot car and all accessories in good condition.
- Any missing or damaged parts must be reported immediately.
- Failure to return the robot car may affect your course grade or result in replacement costs.