Automation Challenge II

Automated Vacuum Cleaning Robot

Tharusha Lekamge - 190352M
Wenuka Somarathne - 190601D

Contents

Introduction

The main aim of this project is to build an automated vacuum cleaning robot that automatically detects the optimum path to travel and cleans an area when the floor plan is given.
The functional and non-functional requirements of our system are as follows.
The map is generated using a maze generator.

Functional Requirements

- Power delivery - The input voltage must be moderated before sending it to the microcontroller. Safety measures must be taken to protect the circuitry from the back EMF of the motors.
- Locomotion - The system must have all the necessary motors to move around.
- Obstacle detection - The system must be able to detect the surroundings and avoid obstacles while moving.
- The system must be able to return to the charging pod.

Additional Functional Requirements

- Cleaning - The robot must consist of a suction motor, garbage collector for the vacuum cleaner functionality.

Non-Functional Requirements

1. The system must consume low power. Because the robot can cover a larger area from a single charge.
2. Should not damage the surroundings while cleaning
3. Easy to set up and operate
4. Ease of maintaining the vacuum cleaning system
5. Low-cost production

The necessity of this project

Cleaning and sweeping is a recurring task. Therefore automating it will save a lot of time from a user of the product. And also, building this will give the developers an insight to robot automation and unknown terrain exploration. This system being a low-cost product makes this available to a larger consumer base.

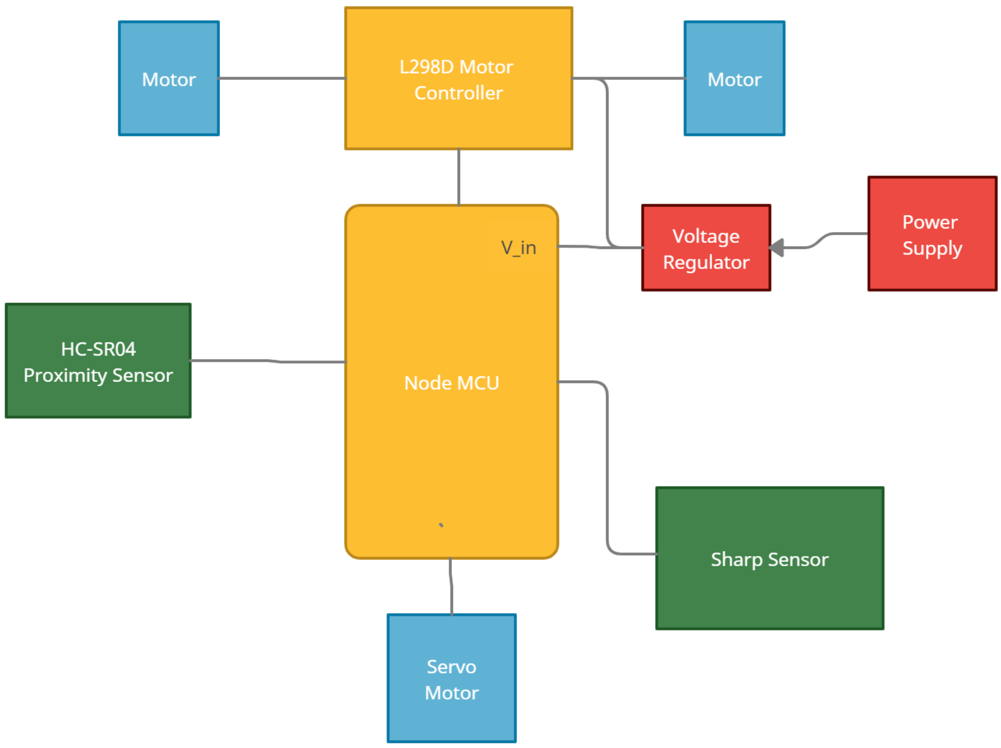
List of subsystems & components needed

- Power delivery
    a. Battery - Battery capacity to be estimated
    b. Charger - Depends on the battery
    c. Resistors - Depends on the power delivery system
    d. Voltage Regulator - To protect the microcontroller, sensors from voltage spikes from the motors
    e. Voltage Divider - Depends on the power delivery system
- Locomotion
    a. 2 wheel robot car chassis(2WD V8 Smart Car Chassis Detection Rate Tracking Remote Control Avoidance Robot – Scion Electronics) (LKR 1900)
        i. The system contains two gear motors. We need these to control the torque and to measure the distance travelled.
- Obstacle detection
    a. For mapping
        i. Sharp sensor * 1 - 3 input pins (LKR 2200)
    b. For fall or high ground detection
        i. Fall detection -(Ultrasonic HC-SR04)(LKR 1000)
    c. Servo Motor(sg90)
- Board
    a. Node MCU
- Cleaning
    a. High power motor
- Outer frame
- Logic Level Converter(IIC I2C Logic Level Converter Bi-Directional Module 5V to 3.3V – Scion Electronics - LKR 72.00)

Optional Components

- High Ground Detection
    - Contact key will be sufficient - 1 input pin
- Communication Module
    - Sync with the mobile application.
    -

Block Diagram of the system



Localisation Algorithm - Simplified

- *Feed a map of boundaries to the robot.*
  - Need a boundary generator app
  - Have to match the real-world boundary to the map
  - Using a grid with the unit length as the length of the robot.
  - It is better to use a square-shaped or circular robot to achieve this.
- *Have to provide a predetermined starting location and position.*
  - This is very important as the robot does not have a compass to find its initial facing direction.
  - Have to feed this information to the robot with the map of boundaries.
  - The start point can be taken as the charging point.
  - The location marked on the map **must** map to the real location as the robot will do calculations relative to that point.
- *The obstacles or precipices (Stair drop) inside the boundary are not necessarily fed in the initial map.*
  - The robot has to identify them by terrain exploration.
  - The robot must update the map after detection
- *The distance travelled can be found using an accelerometer or a wheel encoder.*
- *After the initial map is fed the robot must be placed in the predetermined location(Charging pod).*

- Initially, the IR sensor is rotated around and it will detect the obstacles. The map is then updated.
    - IR sensor is rotated by mounting it on a servo motor with a vertical axis.
    - This is because rotating the whole robot is more power-consuming.
    - The rotation happens clockwise and counterclockwise alternatively to prevent the wires from tangling.
- The robot determines an optimal path and traverses through that until it faces an obstacle. Again the IR sensor is rotated around to find obstacles. The map is updated. This process continues until it reaches the starting point.

Proposed Timeline

Task	**Week 1**	**Week 2**	**Week 3**	**Week 4**	**Week 5**	**Week 6**	**Week 7**	**Week 8**
Detailed System Specifications								
Learning ROS/ Other required								
Designing the Robot								
Virtual prototype								
Programming basic functionalities								
Testing phase 1								
Programming unknown terrain exploration								
Testing phase 2								
Cleaning module implementation								
Unit testing sub-systems								
Testing phase 3								
Deployment								
Documentation								

Estimated Budget

Component	**Price(LKR)**	**Remarks**
Node MCU ESP8266 - 12E	750	
2 Wheel Robot Smart Car Chassis Kit ● Car chassis: 22cm*12cm ● Wheel size: 6.5cm (Dia.) * 2.7cm (H) ● Geared motors ● Speed encoder ● Caster wheel ● Battery box	950	
SG90 9g Towerpro Servo 25cm	420	
Infrared Proximity Sensor Long Range SHARP GP2Y0A02YK0F 20-150cm + Cable	1680	
Ultrasonic Sensor HC-SR04	220	
L298N Motor Driver Module for Arduino (Normal)	410	
<Power delivery>	Est. 1000	
TOTAL	**5430**	