

**Addis Ababa University**

**Addis Ababa Institute of Technology**

**School of Electrical and Computer Engineering**

**Autonomous Vacuum Cleaner Robot**

**By: Biruk Berhanu**

**Advisors’ Name: Mr. Kinde Mekuria**

**Date of submission**: 29/1/2020

Contents

[Abstract 1](#_Toc31577809)

[Acknowledgment 2](#_Toc31577810)

[Acronym 3](#_Toc31577811)

[Chapter 1 4](#_Toc31577812)

[1.1 Introduction and Background 4](#_Toc31577813)

[1.2 Problem Statement 5](#_Toc31577814)

[1.3 Objective 6](#_Toc31577815)

[Chapter 2 7](#_Toc31577816)

[2.1 Literature Review 7](#_Toc31577817)

[Chapter 3 10](#_Toc31577818)

[3.1 Hardware Design 10](#_Toc31577819)

[3.1.1 Wheels and Chassis 12](#_Toc31577820)

[3.1.2 Cleaning 13](#_Toc31577821)

[3.1.3 Navigation 14](#_Toc31577822)

[3.1.4 Control sensors and microcontroller 15](#_Toc31577823)

[3.1.5 Power 15](#_Toc31577824)

[3.1.6 Physical Dimension 17](#_Toc31577825)

[3.1.7 Schematic Diagram 18](#_Toc31577826)

[3.2 Software Design 18](#_Toc31577827)

[3.2.1 Super loop Vs Real time operating system (RTOS) 18](#_Toc31577828)

[3.2.2 Code 18](#_Toc31577829)

[3.2.1 Arduino Code 18](#_Toc31577830)

[Chapter 4 19](#_Toc31577831)

[4.1 Result 19](#_Toc31577832)

[4.2 Conclusion 20](#_Toc31577833)

[Reference 21](#_Toc31577834)

[Appendix 22](#_Toc31577835)

[Appendix A Arduino Code 22](#_Toc31577836)

[Figure 2‑0‑1 The layout of the electrical system showing connection between components and Arduino ATmega2560 8](#_Toc31554665)

[Figure 2‑0‑2 Simple design of contact sensor 9](#_Toc31554666)

[Figure 3-0‑1 Solid work design exploded view of RoboClean 10](#_Toc31554667)

[Figure 3‑0‑2 solid work design 3D view of RoboClean 11](#_Toc31554668)

[Figure 3-0‑3 solid work design 3D view captured from bottom of RoboClean 11](#_Toc31554669)

[Figure 3-0‑4 Solid work design 3D exploded view 12](#_Toc31554670)

[Figure 3-0‑5 DC motor 12](#_Toc31554671)

[Figure 0‑6 Cleaning section of RoboClean 14](#_Toc31554672)

[Figure 3-0‑7 Power Bank 16](#_Toc31554673)

[Figure 3-0‑8 Inside the USB cable wire 16](#_Toc31554674)

[Figure 3-0‑9 Schematic Diagram for the circuit design 18](#_Toc31554675)

[Figure 4-0‑1 Proteus simulation of RoboClean 19](#_Toc31554676)

[Figure 4-0‑2RoboClean Controller Java app 20](#_Toc31554677)

[Figure4- 0‑3 Welcome and control activities of the Android app 20](#_Toc31554678)

# Abstract

We are in the age of the new era, which is the age of the digital world. Almost everything has become autonomous and accompanied by AI. I strongly argue not to do house chores manually since we have already passed the manual age. There is some progress in building an Autonomous vacuum cleaner robot to clean our houses merely without any help from people. But current Vacuum cleaner robots are somewhat expensive and inefficient. It is, therefore, a sine qua non to improve the technology of vacuum cleaning to reduce these deficiencies.

Here, I report the development of a compact and efficient vacuum cleaner robot for potential office and home use. The robot will be disk-shaped, equipped with vacuuming and cleaning technology and controlled by Arduino Mega microcontroller. It sucks dirt via a retractable dustbin on top of which a cooling fan is mounted, and two sweepers each driven by motor. The robot navigates via two motor shield controlled rear wheels and a front caster wheel which also governs its turning. Four ultrasonic sensors, placed at 90 apart, detect obstacles and subsequently help the robot navigate.

# Acknowledgment

I would like to express my special thanks of gratitude to my advisor Mr. Kinde Mekuria who gave me the opportunity to do this project on the topic Autonomous Vacuum Cleaner Robot, which also helped me in understanding a lot of ideas and approach. I come to know about so many new things I am really thankful to them.

Secondly, I would also like to thank my friends who helped me a lot in finalizing this project within the limited time frame.

# Acronym

SLAM-----Simultaneous Localization and Mapping

RoboClean ---- Robot Cleaner

DC --- Direct Current

# Chapter 1

## 1.1 Introduction and Background

A very notable household chore is floor cleaning which is oftenconsidered as unpleasant, difficult, awkward and boring. Inmost cases, cleaners are hired to do the task rather than thehousehold residents do it. The discomfort posed by this recurrent chore necessitated development of a vacuum cleaner thatcould assist human with such a task. A vacuum cleaner is anelectromechanical appliance commonly used for cleaningfloors, furniture, rugs and carpets by suction. An electricmotor inside the appliance turns a fan which creates a partialvacuum and causes outside air to rush into the evacuated space. This forces any dirt or dust near the nozzle into a baginside the machine or attached to the outside.

The robotic vacuum is mainly built from a circular piece of foam board. The robotic vacuum uses a rotating brush underneath the unit to vacuum a carpet as it passes over it. Two DC motors, aligned across the center axis of the robot, are used to accurately drive the robotic vacuum around a room. Because the body of the robot is circular and the DC are placed along the center axis, the robot can spin in place in any direction. One free-spinning chair wheel is located at the rear of the robot to keep it balanced.

The robot is programmed to sense the direction of a collision with an obstacle using an ultrasonic sensor. The robotic vacuums movement is based upon a random walk around a room, which theoretically will cover the entire area of a room given enough time. The robot is programmed to drive straight until an obstacle is hit. At that point, it will turn and continue driving straight until another obstacle is hit, and so on.

## 1.2 Problem Statement

Currently, in an urban area, most people spent their time doing their day to day jobs. In addition to their day to day jobs, they are also required to do their household chores. Naturally, household chores are boring and it would not be fair for a person to engage in such kind of task after a long working day. A vacuum cleaner robot will eliminate one of the tiresome chores, which is cleaning a house. Therefore, a lot of busy people would benefit from the autonomous vacuum cleaner.

Cleaning of these offices and buildings happen at dawn, which is a time when peoples start to crowd the place. During this time there is a smell of dust and detergent, which is very uncomfortable and unhealthy. However, most offices and buildings are not used at night. Therefore, if cleaning is made at the time when there are no people, for example starting from midnight, it would eliminate the above problem. The solution can be achieved using an autonomous vacuum cleaner robot.

Cleaning home has never been easy for older peoples and disables. They need a form of a robot that performs the task. Implementing the vacuum cleaner robot would help a lot with such kind of community.

Depending on the design target, robotic vacuum cleaners are appropriate for offices, hotels, hospitals, and homes. However, most cheap cleaners need a better cleaning pattern algorithm for efficient functioning while the smart ones are rather costly, and thus beyond the reach of most homes. These challenges were carefully considered while designing the robot vacuum cleaner described in this paper.

As a final project, I decided to design and build a robot capable of vacuuming the floor of a room or area without any human interaction other than just starting the unit. I realized the need for a cheap and convenient product that can be easily used to vacuum a room on its own, saving a person valuable time.

## 1.3 Objective

Generally making a robot requires a lot of parameters for consideration and it can have many scopes on how deep the robot is intended to do the required task. On this project, it is planned to have the following feature:

1. It will have two DC motor for its movement with the help of one steering wheel
2. It will have one motor to create a vacuum at the bottom of the robot
3. It will have an ultrasonic sensor for obstacle detection and navigation
4. It will have two DC motor to implement the sweeper
5. Two mode of operation: the first mode of operation is by using an android phone to monitor and navigate. The second mode of operation is without any control from the external system. This mode of operation is what makes the robot autonomous.

# Chapter 2

## 2.1 Literature Review

A robotic vacuum cleaner is an autonomous electronic device that is intelligently programed to clean a specific area through a vacuum cleaning assembly. Some of the available products can brush around sharp edges and corners while others include a number of additional features such as wet mopping and UV sterilization rather than vacuuming. Some of the available products are discussed below.

**A. *iRobot*:** In 2002, iRobot launched its first-floor vacuum cleaner robot named Roomba. Initially, iRobot decided to manufacture limited number of units but Roomba immediately became a huge consumer sensation. Due to its increased market demand, a series of following robots have been launched in the market:

**1. Roomba:**• Launch Date: 2002  
• Manufacturer: iRobot (American)  
• Type of Use: Dry Vacuum  
• Technology: IR, RF and auto-charging mechanism  
• Price: $500  
**2. Scooba**• Launch Date: 2005  
• Manufacturer: iRobot (American)  
• Type of Use: Wet Washing of Floor  
• Technology: IR with virtual wall accessories  
• Price: $500  
**3. Brava**• Launch Date: 2006  
• Manufacturer: iRobot, KITECH, Sony  
• Type of Use: Floor moping for hard surfaces/Dry clean  
• Technology: IR with virtual wall accessories for industrial cleaning  
• Price: $700  
***B.* NEATO Robotics**With the advent of robotic vacuum cleaners, many countries had started manufacturing robotic cleaners. China also started manufacturing these robots with more reliable technology and advanced features.  
**1. Neato XV-11**• **Launch Date:** 2010  
• **Manufacturer:** Neato-Robots XV series (California)/China  
• **Type of Use:** Vacuum Cleaning  
•**Technology:** Laser range finder technology, SLAM (Simultaneous localization and mapping) and auto-charging  
• Price: $399  
***C.* Dyson**In 2001, Dyson built a robot vacuum known as DC06 which was never released to the market due to its high price. In 2014, Dyson launched a new product named as Dyson 360 Eye which uses a different technology for path finding as compared to products manufactured by NEATO Robotics or iRobot.  
**1. EYE-360**• Launch Date: 2016  
• Manufacturer: Dyson (UK)  
• Type of Use: Vacuum Cleaning

Technology: It uses a 360-degree panoramic vision camera to monitor its environment in real time and a turbo brush for efficient cleaning along with an auto-charging mechanism  
(Benchmark in history of cleaning robots)  
• Price: $1000 (approx.)

There is a Journal, Development of a vacuum cleaner robot, written by T.B. Asafa, T.M. Afonja, E.A. Olaniyan, H.O. Alade in Mechanical Engineering Department, Ladoke Akintola University of Technology, P.M.B. 4000, Ogbomoso, Nigeria. The article was written in mechanical department. Their paper focuses on more on the mechanical part. Their overall design of the system looks like the figure below. They used four ultrasonic sensors for navigation.

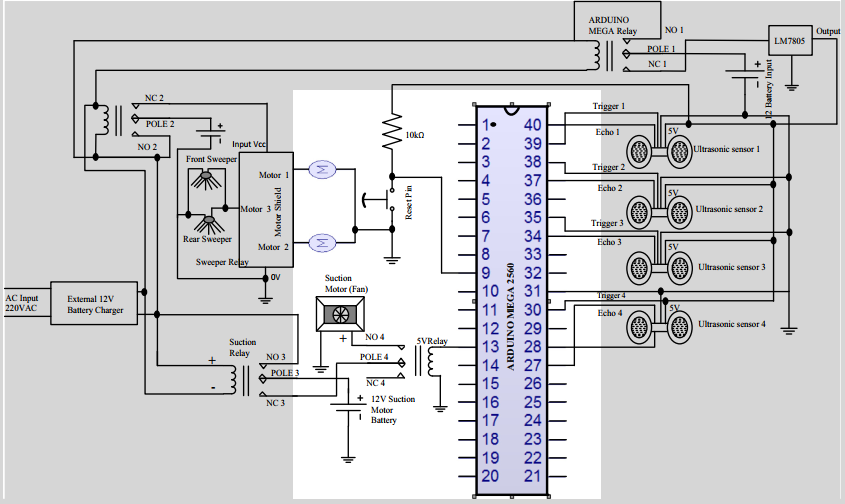


Figure 2‑0‑1 The layout of the electrical system showing connection between components and Arduino ATmega2560

But their design can be more optimized by using only one ultrasonic sensor for navigation with two contact sensor in the left and right side of the front side. The contacts are simple contact between ground and input pin. The contact will look like as the figure below. Using this method, we can reduce three ultrasonic sensors.



Figure 2‑0‑2 Simple design of contact sensor

In the article they use ultrasonic measurement in the four direction to map the environment. But in our project, I will use an industry standard algorithm for mapping the environment.

# Chapter 3

## 3.1 Hardware Design

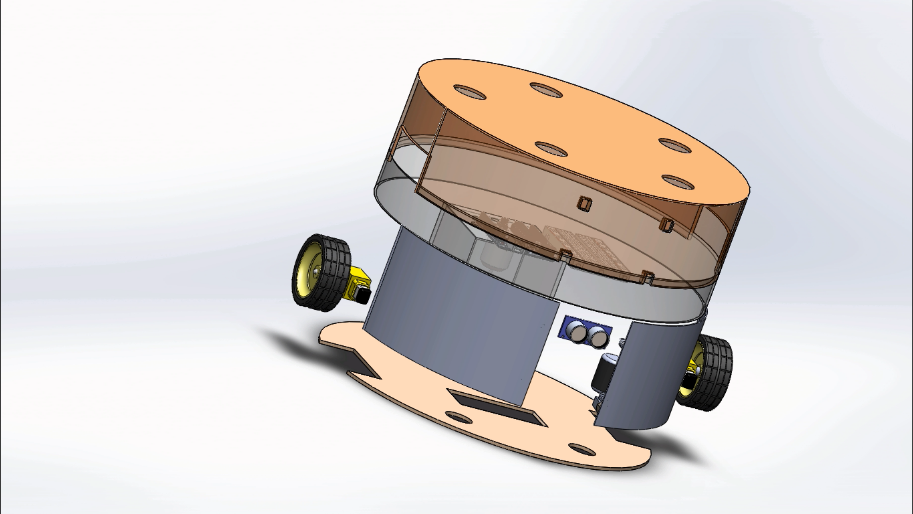
* Arduino Mega
* Two motors with gears and power wires. Motor with gears has 100-200 RPM (revolutions per minute). It is good to connect ceramic capacitor 1uF to motor power contacts - to reduce electric noise.
* Two wheels
* Battery set, or accumulator, or power-bank - on 5 volts
* Air sucker motor with the right structure of propeller
* Wires to connect to battery set. Or USB cable - to connect to power-bank with USB power connector.
* Cable with power connector for Arduino
* Wires for Arduino contacts (breadboard wires)
* Cardboard
* Glue or melting glue
* Piece of tin (e.g. from aluminum can)
* Plastic box - for dust-bin

Figure 3-0‑1 Solid work design exploded view of RoboClean

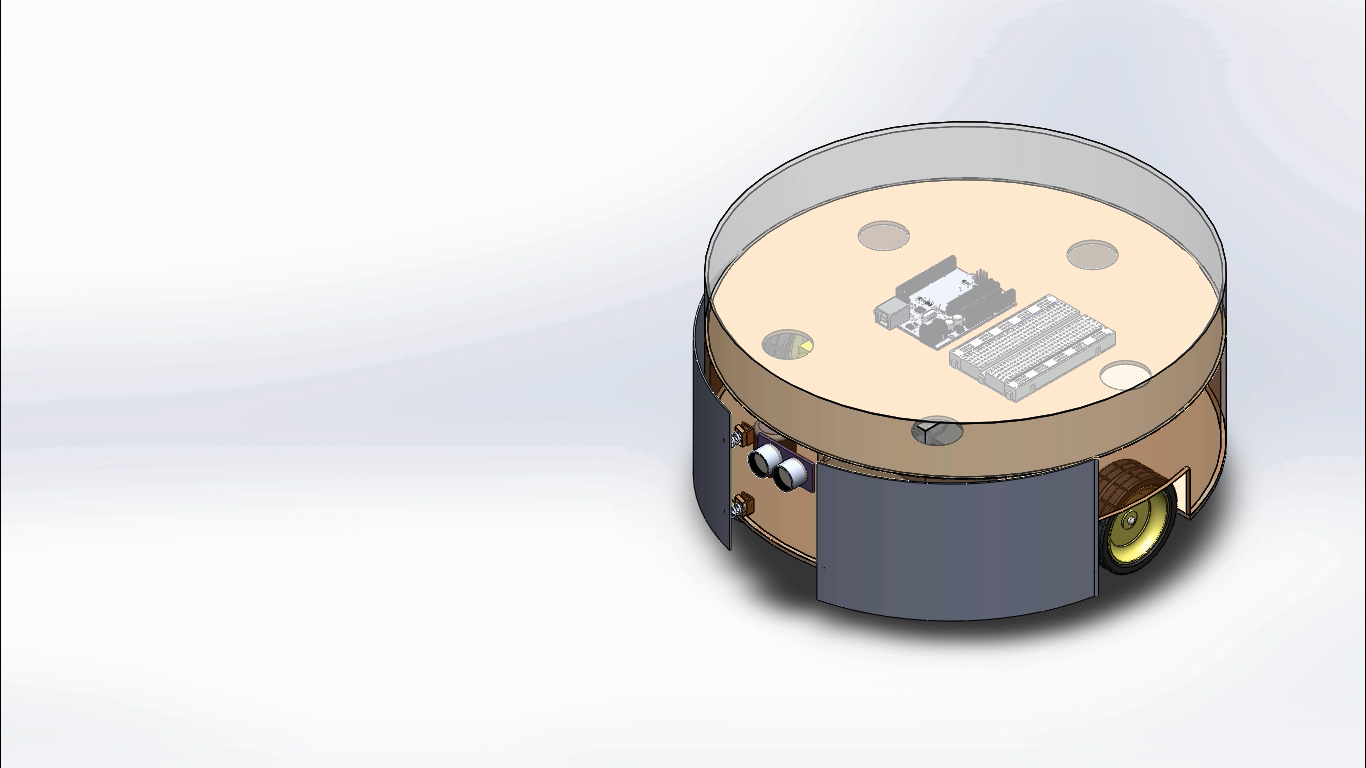


Figure 3‑0‑2 solid work design 3D view of RoboClean

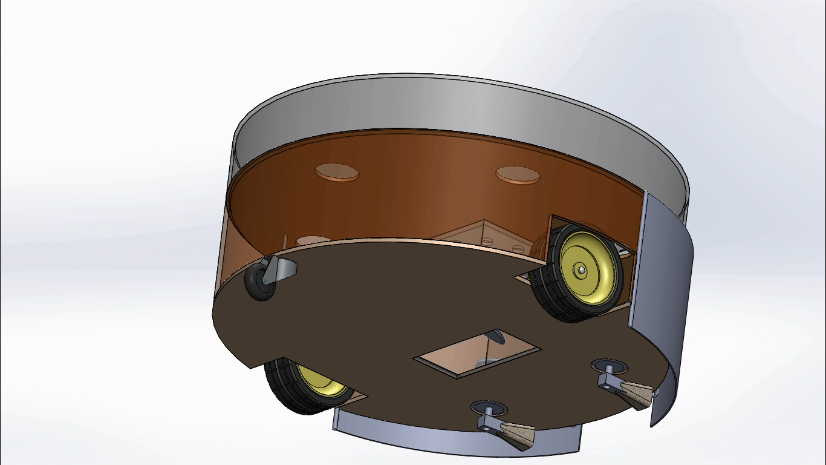


Figure 3-0‑3 solid work design 3D view captured from bottom of RoboClean

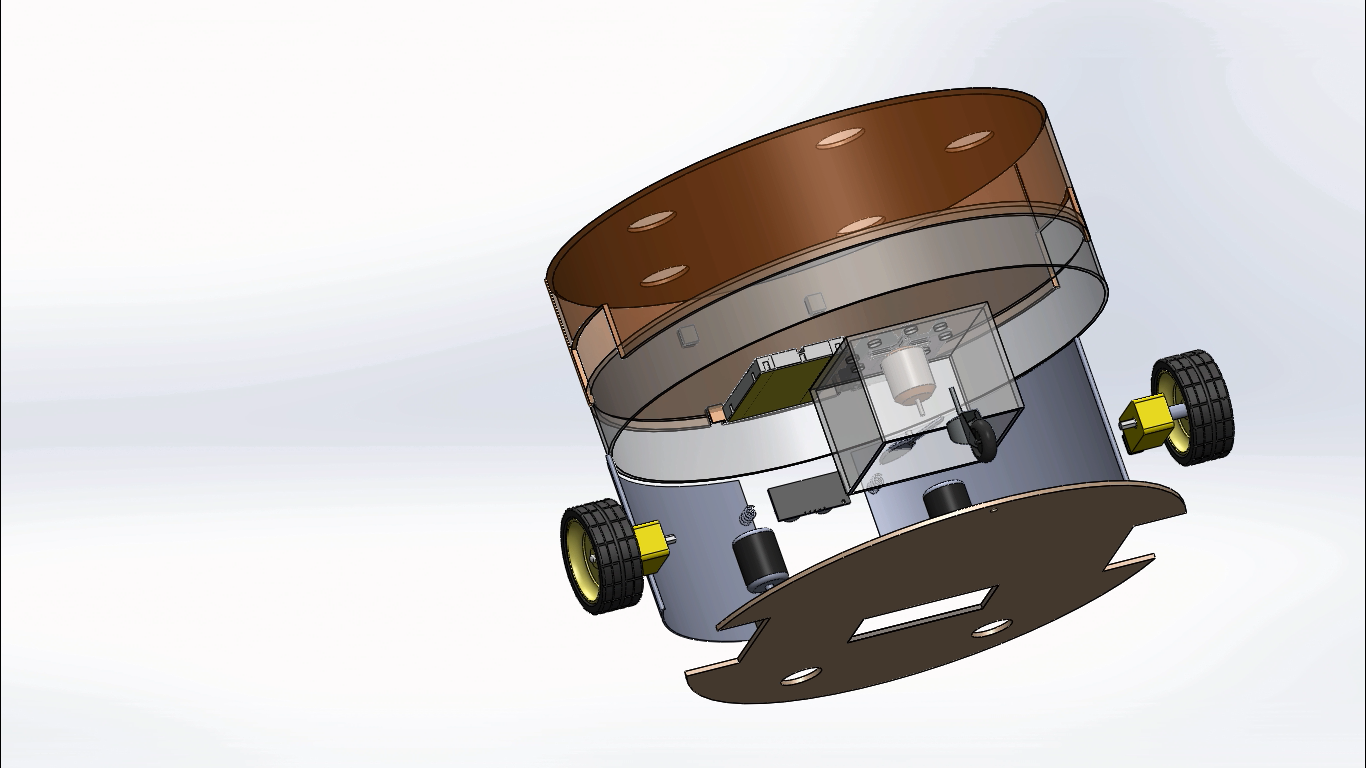


Figure 3-0‑4 Solid work design 3D exploded view

### 3.1.1 Wheels and Chassis

The Robot have three wheels in total. Two of the three wheels are near the center of the chassis. The third one will be a dummy wheel, which helps for balancing the mass load of the robot. The two-navigation wheel are driven using a DC motor. The specification of the DC moter listed below are from a datasheet from a manufacturer called Adafruit Industries.



Figure 3-0‑5 DC motor

**Description**

These are standard '130 size' DC hobby motors. They come with a wider operating range than most toy motors: from 4.5 to 9VDC instead of 1.5-4.5V. This range makes them perfect for controlling with an Adafruit Motor Shield, or with an Arduino where you are more likely to have 5 or 9V available than a high current 3V setting. They'll fit in most electronics that already have 130-size motors installed and there's two breadboard-friendly wires soldered on already for fast prototyping

**Technical details**

• Operating Temperature: -10°C ~ +60°C  
• Rated Voltage: 6.0VDC  
• Rated Load: 10 g\*cm  
• No-load Current: 70 mA max  
• No-load Speed: 9100 ±1800 rpm  
• Loaded Current: 250 mA max  
• Loaded Speed: 4500 ±1500 rpm  
• Starting Torque: 20 g\*cm  
• Starting Voltage: 2.0  
• Stall Current: 500mA max  
• Body Size: 27.5mm x 20mm x 15mm  
• Shaft Size: 8mm x 2mm diameter  
• Weight: 17.5 grams

**Analysis of Motor specification**

There are three different amounts of current needed by the motor, no load current, Loaded current, and stall current. Considering the motors will run in full loaded mode the current required for the motor will be 250mA.

From the specification we can see that the starting torque of the motor is 20 g\*cm. The robot is driven using two motor. Two motor, each having 20g\*cm torque, are expected to carry the whole mass around. The expected mass is calculated in the physical dimension section. The whole expected mass is around 700gram.

### 3.1.2 Cleaning

The robot is designed to be equipped with two rotating sweeper motors, each equipped with two brushes which spin the dirt into the vacuum region. The sweeper motor will be placed slightly away from the center axis. At the center of the robot there will be a vacuum region which will be created by the vacuum DC motor.

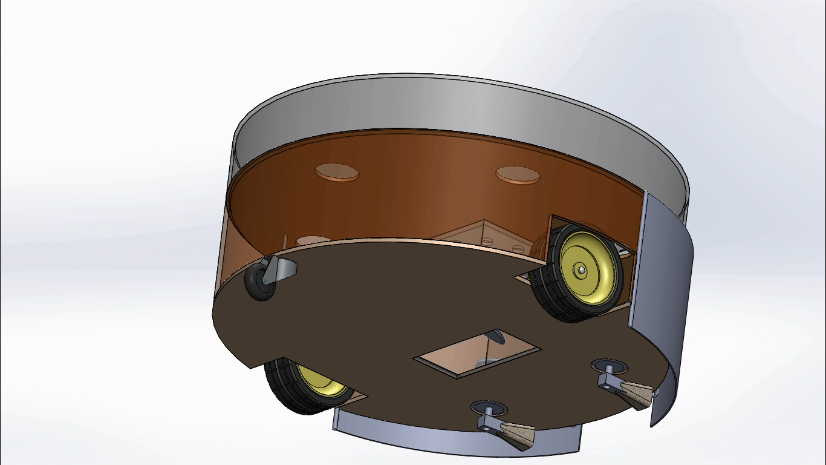


Figure 0‑6 Cleaning section of RoboClean

### 3.1.3 Navigation

This functionality is implemented in two ways. First by using android phone application to control the navigation. The second one is by using robot navigation algorithm. There are many algorithms used for robot navigation system. At first, I will use the first way to control the robot. After implementation of the first way I will pass to the second way on which I will implement the SLAM algorithm.

**Radom walk algorithm**

This algorithm is all about walking in a random direction within a bounded environment. Theoretically, the robot will cover the whole entire room after some time.

**Simultaneous Localization and Mapping (SLAM)**

The (SLAM) problem asks if it is possible for a mobile robot to be placed at an unknown location in an unknown environment and for the robot to incrementally build a consistent map of this environment while simultaneously determining its location within this map. A solution to the SLAM problem has been seen as a ‘holy grail’ for the mobile robotics community as it would provide the means to make a robot truly autonomous. The ‘solution’ of the SLAM problem has been one of the notable successes of the robotics community over the past decade. SLAM has been formulated and solved as a theoretical problem in a number of different forms. SLAM has also been implemented in a number of different domains from indoor robots, to outdoor, underwater and airborne systems. At a theoretical and conceptual level, SLAM can now be considered a solved problem. However, substantial issues remain in practically realizing more general SLAM solutions and notably in building and using perceptually rich maps as part of a SLAM algorithm

### 3.1.4 Control sensors and microcontroller

Sensors help robots perceive external environment, make decision and act accordingly. The robot will one ultrasonic sensor (HC-SRO4 model) which are inbuilt with sonar designed to determine distance to an object. This type of sensor offers excellent non-contact range detection (2– 400 cm) with high accuracy. Its operation is not affected by sunlight or black material, although acoustically soft materials like cloth can be difficult to detect. It comes with ultrasonic transmitter and receiver module.

Microcontrollers, in general, contain processor cores, memory, and programmable input/ output peripherals. They are used in automatically controlled products and devices, such as automobile engine control systems, remote controls, office machines, appliance and other embedded systems. In this work, I used Arduino microcontroller because it uses single board computer, making it popular in professional market. The Arduino is open-source with relatively cheap hardware. While Arduino microcontroller comes in different features depending on the complexity of use, Arduino mega is used for this work because it allows us to use more pins and more memory to retain codes. The Arduino Mega is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header and a reset button.

### 3.1.5 Power

All needed power to the components is supplied by a power bank(battery). No component needed power is supplied using the microcontroller. This helps to supply the necessary needed power for the components without loading the microcontroller.

|  |  |  |  |
| --- | --- | --- | --- |
| Components | Running time (%) | Number of components | Current drawn(mA) |
| Microcontroller | 100 | 1 | 50 |
| Wheel Motor | 100 | 2 | 2\*250 |
| Fan motor | 100 | 1 | 250 |
| Sweeper motor | 100 | 2 | 2\*50 |
| Ultrasonic sensor | 100 | 1 | 20 |
| Total |  |  | 920 |

**Power Source**



Figure 3-0‑7 Power Bank

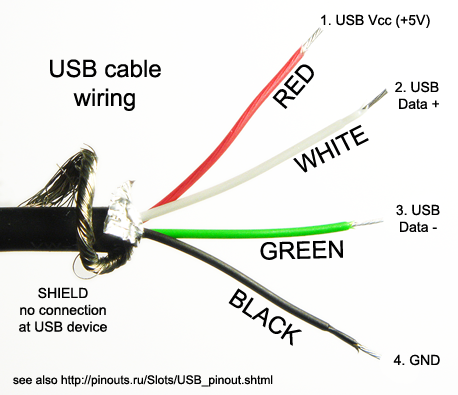


Figure 3-0‑8 Inside the USB cable wire

A power bank is the perfect power source for driving the motors and the Arduino. As we can see the above pictures, we can use normal USB cable in conjunction with a power bank to power the circuit.

**Battery specification:**

Nominal Voltage = 5 V.  
Nominal Capacity = 10,000mAh.

Total current consumption is 920 mA.  
Running time of the robot =10,000 mAh/ 920mA = 10.87hour

### 3.1.6 Physical Dimension

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Components | Diameter(cm) | Height (cm) | Area(cm2) | Volume(cm3) | Mass(g) |
| Dustbin | 10 | 3 | - | 235 | 10 |
| Chassis | 60 | - | - | - | 100 |
| Motor | - | - | - | - | 20\*5 |
| Battery | - | - | 70 | - | 250 |
| breadboard | - | - | 70 | - | 80 |
| Arduino |  |  | 50 | - | 40 |
| Additional body part | - | - | - | - | 100 |
| Total |  |  |  |  | ~700 |

### 3.1.7 Schematic Diagram

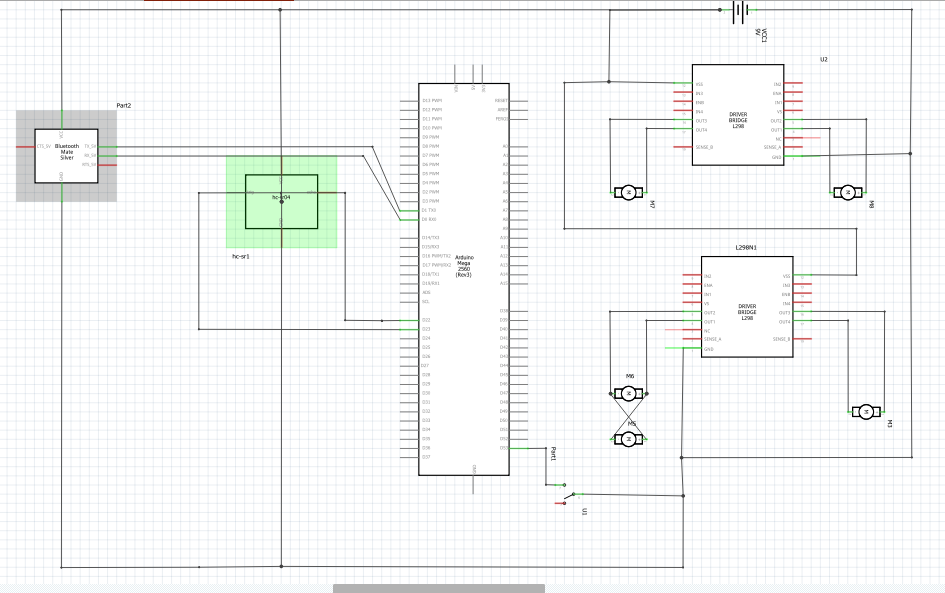


Figure 3-0‑9 Schematic Diagram for the circuit design

## 3.2 Software Design

### 3.2.1 Super loop Vs Real time operating system (RTOS)

There are two major ways to implement the robot code. The first and the obvious way is using a super loop approach. Super loops work well when there is no time critical task.  They can still be pretty responsive if you only test for one, or a few, conditions during each pass. If there are critical tasks that can't be handled by a simple interrupt driven queue, an RTOS is the way to go. Considering the behavior of the robot a simple super loop is enough.

### 3.2.2 Code

### 3.2.1 Arduino Code

*View Code At Appendix A.1*

# Chapter 4

## 4.1 Result

As stated in the proposal the Robot will be controlled in two mode. The first mode of operation is to control the robot using android application. The second mode of operation is controlling the robot is using an algorithm to control the robot autonomously. There were two type of algorithm stated in the proposal on how to navigate the robot. The first one is using random walk algorithm and the second one is using the known mapping algorithm called SLAM**.** This semester I was able to implement the random walk method. The simulation was implemented in Proteus simulation environment. Both autonomous mode and android app control mode implemented. There were also some new concepts and ideas are found when trying to find the proposal stated problem. For example, the way the code implemented can be using real time operating system or super loop method. The former would provide an accurate timing and controlling small tasks. But for small system a super loop approach can implement the intended tasks. In this project I used the super loop method to implement the code.

Figure show below are screenshot of the proteus simulation, the android app. Due to the difficulty of connecting android phone to proteus directly, I made a simple GUI Java app that communicate in the way the android app is intended to communicate.

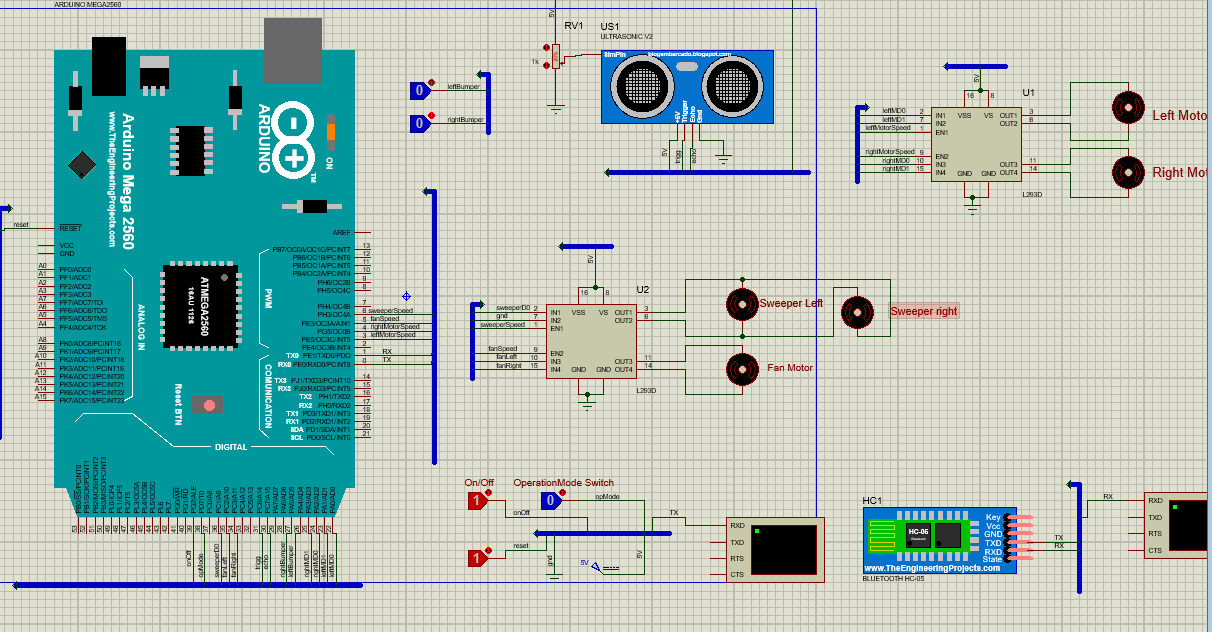


Figure 4-0‑1 Proteus simulation of RoboClean

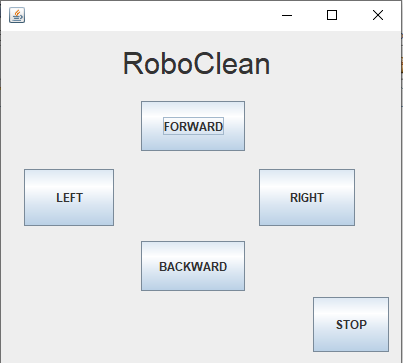


Figure 4-0‑2RoboClean Controller Java app

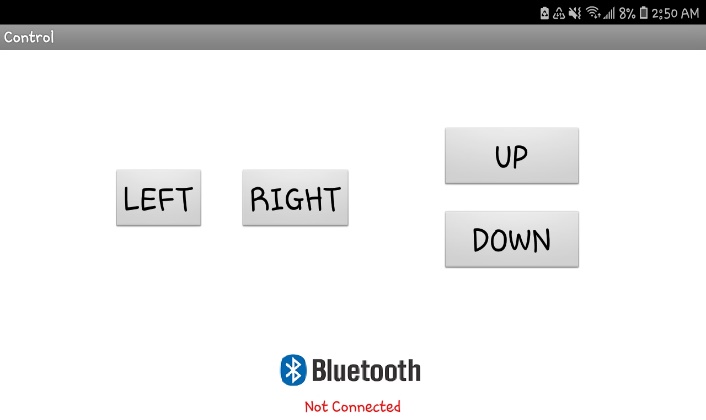
 

Figure4- 0‑3 Welcome and control activities of the Android app

## 4.2 Conclusion

This paper highlights a better and simpler approach in providing an overview of the design of an autonomous vacuum cleaner robot. This robot is designed to have all the features of a  
conventional vacuum cleaner. It can work automatically and manually. The robot can be made useful in large areas of coverage such as hospitals, industries, schools, etc. It can also be  
used where cleaning involving humans may be harmful.

# Reference

[1].T.B. Asafa, T.M. Afonja, E.A. Olaniyan, H.O. Alade, Development of a vacuum cleaner robot, 20 October 2016, Nigeria

[2]. R. Neato, ‘‘Company | Neato Robotics,” Neato Robotics, 2015 [Online]. Available: <http://www.neatorobotics.com/company/> Accessed 26 05 2015].

[3]N. Mitchel, The Lazy Person’s Guide to a Happy Home: Tips for People Who (Really) Hate Cleaning, 9 Jan 2016 [Online]. Available: http://www.apartmenttherapy.com/the-lazy-personsguide-to-a-happy-home-cleaning-tips-for-people-who-reallyhate-cleaning-197266 [Accessed 22 June 2016]

[4]. Abhishek Pandey, Anirudh Kaushik, Amit Kumar Jha, Girish Kapse, “A technological survey on autonomous and home cleaning robots”,International Journal of Scientific and  
Research Publications, Volume 4, Issue 4, April 2014

[5]. Neato, 'Neato Robotics | Smartest, Most Powerful, Best Robot Vacuum', 2015. [Online]. Available: http://www.neatorobotics .com/

[6]. Manreet Kaur, PreetiAbrol, “Design and Development of Floor Cleaner Robot”, International Journal of Computer Applications (0975 – 8887) Volume 97– No.19, July 2014  
[7]. Karthick.T, Ravikumar.A, Selvakumar.L, Viknesh.T, Parthiban.B and Gopinath. A, “Simple Autonomous cleaner Robot”, International Journal of Science, Engineering and Technology Research (IJSETR), Volume 4, Issue 5, May 2015 [5]. Cleaning and Household Robots: A Technology Survey by P. Fiorini, 2000, pp.23-65

## Appendix

### Appendix A Arduino Code

#include "SR04.h"

const int Baud = 4800; //UART baud rate

const long MinFrontDistance = 20; //Minimum allowed front distance

//controlling pin

const int pinOpMode = 38;

      int operationMode = 0; // 0 for autonomous mode and 1 for manual mode/Autonomous Mode

      int command;

const int pinOnOff = 39;

//right side

const int pinRightMotorDirection0 = 24;

const int pinRightMotorDirection1 = 25;

const int pinRightMotorSpeed = 4;

const int pinRightBumper = 27;

//left side

const int pinLeftMotorDirection0 = 22;

const int pinLeftMotorDirection1 = 23;

const int pinLeftMotorSpeed = 3;

const int pinLeftBumper = 26;

//ultrasonic sensor

const int pintrigg =31 ;

const int pinecho =30 ;

//sweeper and fan

const int pinSweeperMotorDirection0 = 36;

const int pinFanMotorDirection0 = 35;

const int pinFanMotorDirection1 = 34;

const int pinFanMotorSpeed = 5;

const int pinSweeperMotorSpeed = 6;

SR04 ultrasonic = SR04(pinecho,pintrigg);

long dis;

const int turnRightTimeout = 3;

const int turnLeftTimeout = 3;

int countDownWhileMovingToRight ;

int countDownWhileMovingToLeft ;

void setup() {

  Serial.begin(Baud);

  Serial.println("STARTING");

  initPins();

  bool opmode = digitalRead(pinOpMode);

  if(opmode = true)operationMode = 1;

  else operationMode = 0;

}

//Main loop

void loop() {

  bool onOff = digitalRead(pinOnOff);

  if(onOff == false){

      stopMotors();

      delay(1000); // make this value large for the purpose of consumption of

 power in power off mode

      operationMode = 1;

      return;

    }

  bool opmode = digitalRead(pinOpMode);

  if(opmode == true){

    if(operationMode == 1){

      Serial.println("heyyy");

           runRightMotorForward();

           runLeftMotorForward();

           delay(100);

           startSweeperAndFan();

       }

      Serial.println("opMode:High");

      verifyAndSetRightSide();

      verifyAndSetLeftSide();

      processRightSide();

      processLeftSide();

      delay(10);

      operationMode = 0;

    }else{

        if(operationMode == 0){

             Serial.println("opMode:low");

            stopMotors();

            stopSweeperAndFan();

          }

        operationMode = 1;

        if (Serial.available() > 0) {

        command = Serial.read();

        switch(command){

            case 'F':{

              startMotors();

              startSweeperAndFan();

              break;

            }

            case 'B':{

              runLeftRightMotorBackward();

              startSweeperAndFan();

              break;

            }

            case 'L':{

              turnLeft();

              startSweeperAndFan();

              break;

            }

            case 'R':{

              turnRight();

              startSweeperAndFan();

              break;

            }

            case 'S':{

              stopMotors();

              stopSweeperAndFan();

              break;

            }

          }

        }

    }

}

//---------------------------------------------------

void initPins(){

  pinMode(pinRightMotorDirection0, OUTPUT);

  pinMode(pinRightMotorDirection1, OUTPUT);

  pinMode(pinRightMotorSpeed, OUTPUT);

  pinMode(pinRightBumper, INPUT\_PULLUP);

  pinMode(pinLeftMotorDirection0, OUTPUT);

  pinMode(pinLeftMotorDirection1, OUTPUT);

  pinMode(pinLeftMotorSpeed, OUTPUT);

  pinMode(pinLeftBumper, INPUT\_PULLUP);

  pinMode(pinOpMode,INPUT\_PULLUP);

  pinMode(pinFanMotorSpeed,OUTPUT);

  pinMode(pinFanMotorDirection1,OUTPUT);

  pinMode(pinFanMotorDirection0,OUTPUT);

  pinMode(pinSweeperMotorDirection0,OUTPUT);

  pinMode(pinSweeperMotorSpeed,OUTPUT);

}

void startSweeperAndFan(){

  digitalWrite(pinSweeperMotorDirection0,HIGH);

  analogWrite(pinSweeperMotorSpeed,50);

  digitalWrite(pinFanMotorDirection0,HIGH);

  digitalWrite(pinFanMotorDirection1,LOW);

  analogWrite(pinFanMotorSpeed,255);

}

void stopSweeperAndFan(){

  analogWrite(pinSweeperMotorSpeed,0);

  analogWrite(pinFanMotorSpeed,0);

}

void startMotors(){

  runRightMotorForward();

  runLeftMotorForward();

}

void stopMotors(){

   analogWrite(pinLeftMotorSpeed, 0);

   analogWrite(pinRightMotorSpeed, 0);

}

void turnLeft(){

  runLeftMotorBackward();

  runRightMotorForward();

}

void turnRight(){

   runRightMotorBackward();

   runLeftMotorForward();

}

void runLeftRightMotorBackward(){

  stopMotors();

  delay(10);

  runLeftMotorBackward();

  runRightMotorBackward();

  delay(1000);

}

void runRightMotorForward(){

  runMotorForward(pinRightMotorDirection0,pinRightMotorDirection1, pinRightMotorSpeed);

}

void runLeftMotorForward(){

  runMotorForward(pinLeftMotorDirection0,pinLeftMotorDirection1, pinLeftMotorSpeed);

}

void runRightMotorBackward(){

  runMotorBackward(pinRightMotorDirection0,pinRightMotorDirection1, pinRightMotorSpeed);

}

void runLeftMotorBackward(){

  runMotorBackward(pinLeftMotorDirection0,pinLeftMotorDirection1, pinLeftMotorSpeed);

}

void runMotorForward(int pinMotorDirection0,int pinMotorDirection1, int pinMotorSpeed){

  digitalWrite(pinMotorDirection0, true); //set direction forward

  digitalWrite(pinMotorDirection1, false); //set direction forward

  analogWrite(pinMotorSpeed, 255);

}

void runMotorBackward(int pinMotorDirection0,int pinMotorDirection1, int pinMotorSpeed){

  digitalWrite(pinMotorDirection0, false); //set direction backward

  digitalWrite(pinMotorDirection1, true); //set direction backward

  analogWrite(pinMotorSpeed, 255);

}

void processRightSide(){

  if(countDownWhileMovingToRight <= 0)//checks if counter was NOT ran when bumper had been pressed

    return;

  //otherwise - counter is counting down (as a delay) while the right motor is moving backward

  countDownWhileMovingToRight--;//decrease the counter if it WAS ran when bumper had been pressed

  if(countDownWhileMovingToRight <= 0)//if the running counter got down to zero

    runRightMotorForward();//run the right motor forward

}

void processLeftSide(){

  if(countDownWhileMovingToLeft <= 0)

    return;

  countDownWhileMovingToLeft--;

  if(countDownWhileMovingToLeft <= 0)

    runLeftMotorForward();

}

void verifyAndSetRightSide(){

  if(checkBumperIsNotPressed(pinRightBumper) //checks if right bumper has NOT been pressed

  && measureDistance() > MinFrontDistance)//checks if the minimum allowed front distance is not reached

    {

    return;

}

    runLeftRightMotorBackward();

  if(checkCounterIsNotSet(countDownWhileMovingToRight))//if the counter is not yet counting down

      turnLeft();

  countDownWhileMovingToLeft = turnRightTimeout;//set the counter to maximum value to start it counting down

}

void verifyAndSetLeftSide(){

  if(checkBumperIsNotPressed(pinLeftBumper) //checks if left bumper has NOT been pressed

   && measureDistance() > MinFrontDistance)//checks if the minimum allowed front distance is not reached

    {

    return;}

    runLeftRightMotorBackward();

  if(checkCounterIsNotSet(countDownWhileMovingToLeft))//if the counter is not yet counting down

     turnRight();

  countDownWhileMovingToRight = turnLeftTimeout;//set the counter to maximum value to start it counting down

}

bool checkCounterIsNotSet(int counter){

  return counter <= 0;

}

bool checkBumperIsNotPressed(int pinBumper){

  return !digitalRead(pinBumper);

}

//-- sonar methods --//

long measureDistance(){

  long distance =  ultrasonic.DistanceAvg();

  Serial.print(distance);

  Serial.print("cm");

  Serial.println();

  return distance;