# A PROJECT ON

***CBOT (Autonomous Vacuum Cleaning Robot Using IOT)***

# Submitted by

***Adityakumar Dhruvkumar Shukla***

# FOR THE DEGREE OF

**MASTER OF INFORMATION TECHNOLOGY**

**(SEMESTER –IV)**

**UNDER THE GUIDANCE OF**

***[Prof. Akash Kamble]***

# B.K. BIRLA COLLEGE OF ARTS, SCIENCE & COMMERCE (AUTONOMOUS)

**KALYAN ACADEMIC YEAR**

**2019-2020**

**A PROJECT ON**

***“CBOT (Autonomous Vacuum Cleaning Robot Using Internet Of Things(IOT))”***

******

**SUBMITTED BY**

***Adityakumar Shukla***

**YEAR OF SUBMISSION [2019-2020]**

**PROJECTGUIDE**

***Prof. Akash Kamble***

**IN THE PARTIAL FULFILLMENT OF DEGREE OF**

**“MASTER OF INFORMATION TECHNOLOGY” SEMESTER-IV**

**B.K. BIRLA COLLEGE OF ARTS, SCIENCE & COMMERCE (AUTONOMOUS)**

**KALYAN**



# CERTIFICATE

This is to certify that Mr. Adityakumar Dhruvkumar Shukla (Roll No. ) has satisfactorily completed the project entitled “ CBOT (Autonomous Vacuum Cleaning Robert Using IOT ) ” for partial fulfillment of the M.Sc.(Information Technology) for the year2019-2020.

Date:

Place:

Internal Project Guide Signature of External Examiner Course Coordinator

# ACKNOWLEDGEMENT

I, Mr. /Ms. Aditya Shukla would take this opportunity to thank University of Mumbai for providing me an opportunity to study on WORKING OF “CBOT (Autonomous Vacuum Cleaning Robert Using IOT )” this has been an enormous learning experience.

I would like to acknowledge and thank people, who made this project work possible,

Prof. (Akash Kamble) who has been guiding force to me while doing this project and the teaching staff of my college, friends for providing their help as when required to complete this project. It is because of their valuable suggestions that this project has turned out to be a success.

# INDEX:

ABSTRACT

1. INTRODUCTION
2. AIM AND OVER VIEW OF THE DESIGN
3. MATERIALS AND METHODOLOG
   * ELECTRONICS
   * HARDWARE
   * DIGITALLY FIBRICATED PARTS
4. ATTACHING THE STEPPER MOTER

1. ELECTRONICS
2. VISION SYSTEM
3. SETTING UP THE CLOUD DATABASE
4. CREATING THE MOBILE APP
5. PROGRAMMING THE RASPBERRI PI
6. PROGRAMMING THE ARDUINO

CONCLUSION

REFERENCES

**ABSTRACT**

# A new service robot designed for cleaning tasks in home environment is introduced.

# System has three subsystems: electical, software and mechanical of which microcontroller, sensors (opponent and light)

# And motor are the electrical and mechanical subsystem respectively and the software subsystem is the brain of the robot.

# The cleaning robot uses a microcontroller to detect obstacles and manipulates its direction its direction as per the input.

# It is programmed to accept inputs to sense obstacles around it and controls the robot to avoid any collision.

# We are also created an android application to connect over the internet using raspberry pi and store its data over the cloud database for trace and monitoring purpose.

**1. INTRODUCTION:**

A **robotic vacuum cleaner**, often called a **roomba**, is an [autonomous robotic](https://en.wikipedia.org/wiki/Autonomous_robot) [vacuum cleaner](https://en.wikipedia.org/wiki/Vacuum_cleaner) which has intelligent programming and a limited vacuum [floor cleaning](https://en.wikipedia.org/wiki/Floor_cleaning) system. The original design included manual operation via [remote control](https://en.wikipedia.org/wiki/Remote_control) and a "self-drive" mode which allowed the machine to clean autonomously without human control. Some designs use spinning brushes to reach tight corners, and some include a number of cleaning features along with the vacuuming feature (mopping, [UV sterilization](https://en.wikipedia.org/wiki/UV_sterilization), etc.).

An advantage of using a robotic vacuum cleaner is how quiet it is compared to a regular vacuum cleaner.[[1]](https://en.wikipedia.org/wiki/Robotic_vacuum_cleaner#cite_note-1) Also, they are seen as more convenient to use because they can vacuum on their own. Robotic vacuums can be kept under beds or desks or in closets, whereas a regular vacuum cleaner requires a larger amount of space. However, a downfall to a robotic vacuum cleaner is that it takes an extended amount of time to vacuum an area due to its size. They are also relatively expensive.

**2. AIM AND OVER VIEW OF THE DESIGN**

To conceptualize the dust monitoring robot, we first designed it on paper and later on solid works . Here are some of its features:

The robot can be controlled via an app through the internet. This allows the user to connect to the robot from anywhere.

An onboard camera that live streams a video feed to the smartphone can help the user to maneuver around the house and interact with the pet.

An add-on treat bowl that can reward your pet remotely.

Digitally fabricated parts also used recycled parts of old cd-drive for customizing a robot .

A Raspberry Pi was used to connect to the internet as it features an onboard wifi mode.(here we are used a vmware for simulating a raspberry pi )

An Arduino was used along with a CNC shield to give commands to the stepper motors.

**3. MATERIALS ANDMETHODOLOGY:**

**ELECTRONICS:**

* **Arduino Uno**x 1
* **Raspberry Pi**(flashed with latest raspbian) x 1
* **CNC Shield**x 1
* **A4988 Stepper Motor Driver** x 2
* **Picamera**x 1
* **Ultrasonic Distance Sensor** x 1
* **11.1v Lipo Battery** x 1
* **NEMA 17 Stepper Motor** x 2
* **5v UBEC**x 1

**HARDWARE:**

* **Wheels** x 2 (the wheels that we used were 7cm in diameter)
* **Castor Wheels** x 2
* **M4 and M3 nuts and bolts**

**ArduinoUno:**

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega32P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog [input/output](https://en.wikipedia.org/wiki/Input/output) (I/O) pins that may be interfaced to various [expansion](https://en.wikipedia.org/wiki/Expansion_board) boards (shields) and other circuits.

**Arduino Uno** is a microcontroller board based on the ATmega328P ([datasheet](http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf)). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0),a USB connection, a power jack, an ICSP header and a reset button.Itcontains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. You can tinker with your Uno without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again."Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Unoboard andversion 1.0 of Arduino Software (IDE) were the reference versionsof Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the Arduino index ofboards.

The board has 14 Digital pins, 6 Analog pins, and programmable with the [Arduino](https://en.wikipedia.org/wiki/Arduino) IDE (Integrated Development Environment) via a type B [USB cable](https://en.wikipedia.org/wiki/USB_cable).

(probably in conjunction with a bread board / PCBs and some wire. They usually have black plasticheaders‟thatallowyoutojustplugawirerightintotheboard.TheArduinohasseveral

different kinds of pins, each of which is labeled on the board and used for different functions. GND (3): Short for „Ground‟. There are several GND pins on the Arduino, any of which can be used to ground your circuit. 5V (4) & 3.3V (5): As you might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off 5 or 3.3 volts. Analog (6): The area of pins under the „Analog In‟ label (A0 through A5 on the UNO) are Analog In pins. These pins can read the signal from an analog sensor (like a light sensor) and convert it into a digital value that we can read. Digital (7): Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED). PWM (8): You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins but can also be used for something called Pulse-Width Modulation (PWM). AREF (9): Stands for Analog Reference. Most of the time you can leave this pin alone. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins. RESET BUTTON The Arduino has a reset button (10). Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if your code doesn’t repeat, but you want to test it multiple times. POWER LED INDICATOR Just beneath and to the right of the word “UNO” on your circuit board, there’s a tiny LED next to the word „ON‟ (11). This LED should light up whenever you plug your Arduino into a power source. If this light doesn’t turn on, there’s a good chance something is wrong. Time to re-check yourcircuit!



### Fig.1 Arduino UNO

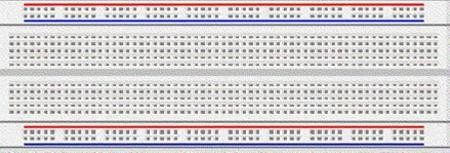


**USB Cable**

**Breadboard:**

A modern solderless breadboard consists of a perforated block of plastic with numerous tin-plated phosphor bronze or nickel silver alloy spring clips under the perforations. The clips are often called tie points or contact points. The number of tie points is often given in the specification of the breadboard. The spacing between the clips (lead pitch) is typically 0.1 in (2.54 mm). Integrated circuits (ICs) in dual in-line packages (DIPs) can be inserted to straddle the centerline of the block. Interconnecting wires and the leads of discrete components (such as capacitors, resistors, and inductors) can be inserted into the remaining free holes to complete the circuit. Where ICs are not used, discrete components and connecting wires may use any of the holes. A breadboard is utilized to build and test circuits expeditiously afore finalizing any circuit design. The breadboard has many apertures into which route components like ICs and resistors can be connected. A typical breadboard that includes top and bottom power distribution rails is shown below figure.

A breadboard is a construction base for [prototyping](https://en.wikipedia.org/wiki/Prototype) of [electronics](https://en.wikipedia.org/wiki/Electronic_circuit). Originally the word referred to a literal bread board, a polished piece of wood used for slicing bread. In the 1970s the solderless breadboard (a.k.a. plugboard, a terminal array board) became available and nowadays the term "breadboard" is commonly used to refer to these. Because the solderless breadboard does not require [soldering](https://en.wikipedia.org/wiki/Soldering), it is reusable. This makes it easy to use for creating temporary prototypes and experimenting with circuit design. For this reason, solderless breadboards are also popular with students and in technological education. Older breadboard types did not have this property. A [stripboard](https://en.wikipedia.org/wiki/Stripboard) ([Veroboard](https://en.wikipedia.org/wiki/Veroboard)) and similar prototyping [printed](https://en.wikipedia.org/wiki/Printed_circuit_board) circuit boards, which are used to build semi-permanent soldered prototypes or one-offs, cannot easily be reused. A variety of electronic systems may be prototyped by using breadboards, from small analog and digital circuits to complete [central processing units](https://en.wikipedia.org/wiki/Central_processing_unit)(CPUs).



### Bread Board

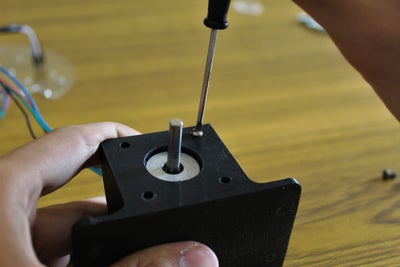
**JumperWires:**

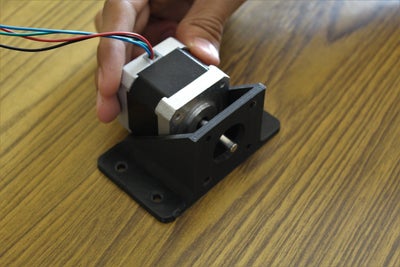
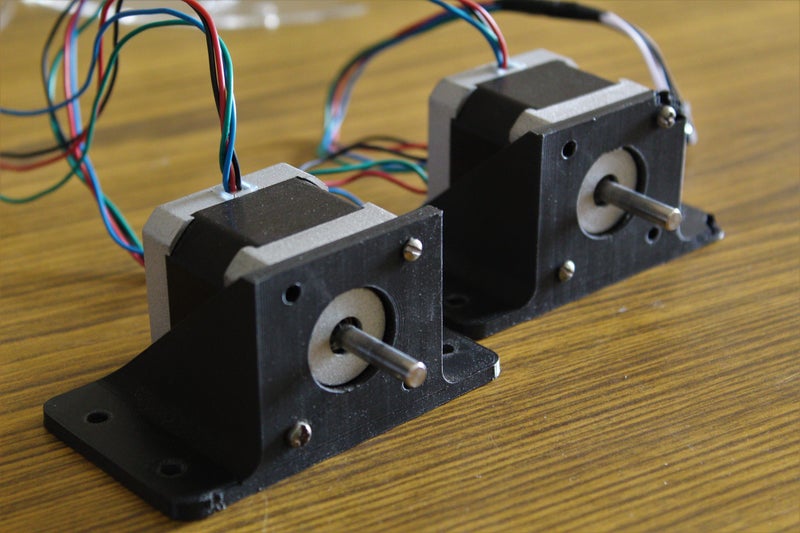
Jump wires are generally used to establish connectivity with bread board as shown in figure 5. Jumper wires typically come in three versions: male-to-male, male-to-female and female-to-female. The difference between each is in the end point of the wire. Male ends have a pin protruding and can plug into things, while female ends do not and are used to plug things into. Male-to-male jumper wires are the most common and what you likely will use mostoften.



1. **ATTACHING THE STEPPER MOTER**

Attaching the Stepper Motor Once all the parts are 3D printed, begin the assembly by mounting the stepper motor into the stepper holder. The stepper motor holder that we designed is meant for the NEMA 17 model (if one uses different steppers it will require a different mount). Pass the shaft of the motor through the hole and secure the motor in place with the mounting screws. Once done both motors should be securely held to the holders.

****

****

To mount the holders to the laser-cut bottom panel we used M4 bolts. Before securing them with the nuts, add the 3D printed chassis reinforcement strips and then fasten the nuts. The strips are used to distribute the load evenly on the acrylic panel.

Finally, pass the wires through the respective slots provided on the panel. Make sure to pull them all the way through to avoid them getting entangled in the wheels.

**A picture containing indoor, table, sitting, wooden

Description automatically generated**

**A picture containing camera, table

Description automatically generated**

**5 .ELECTRONICS**

Once the base of the chassis is fully assembled, it's time to mount the electronics onto the acrylic panel. We have made holes in the acrylic panel that align with the mounting holes of the Arduino and the Raspberry Pi. Using 3D printed standoffs we elevated the electronics slightly above the acrylic panels so that all the excess wiring can be tucked away neatly underneath. Mount the Arduino and the Raspberry Pi to their corresponding mounting locations using M3 nuts and bolts. Once the Arduino is fixed attach the CNC shield to the Arduino and connect the stepper wires in the following configuration.

* **Left stepper** to **CNC shield X-axis port**
* **Right stepper**to **CNC shield Y-axis port**

With the stepper motors attached, connect the Arduino to the Raspberry Pi using the Arduino's USB cable. Eventually the Raspberry Pi and Arduino are going to communicate through this cable.

**Note:** The front of the robot is the side with the Raspberry Pi

A circuit board

Description automatically generated

A circuit board

Description automatically generated A close up of a device

Description automatically generated

**6. VISION SYSTEM**

The primary environment input for our pet monitoring robot is vision. We decided to use the Picamera which is compatible with the Raspberry Pi to feed a live stream to the user via the internet. We also used an ultrasonic distance sensor to avoid obstacles when the robot is functioning autonomously. Both sensors attach onto a holder with the help of screws.

The Picamera slots into its designated port on the Raspberry Pi and connect the ultrasonic sensor in the following manner:

* Ultrasonic Sensor VCC to 5v rail on CNC shield
* Ultrasonic Sensor GND to GND rail on CNC shield
* Ultrasonic Sensor TRIG to X+ end stop pin on CNC shield
* Ultrasonic Sensor ECHO to Y+ end stop pin on CNC shield

**A hand holding a small camera

Description automatically generatedA picture containing sitting, window, computer, table

Description automatically generated**

**A lamp on a table

Description automatically generated**

**7. SETTING UP THE CLOUD DATABASE**

The first step is to create a database for the system so that you can communicate with the robot from your mobile app from anywhere in the world. Click on the following link ([Google firebase](https://firebase.google.com/?gclid=CjwKCAiA_f3uBRAmEiwAzPuaM1GaFxeXvUiDYIjjRp-MxQvvQw1f9BREosH2q7VJU2YqwVyRkA8gFxoCcmsQAvD_BwE)), which will lead you to the Firebase website (you will have to log in with your Google account). Click on the "Get Started" button which will take you over to the firebase console. Then create a new project by clicking on the "Add Project" button, fill in the requirements (name, details, etc) and complete by clicking on the "Create Project" button.

We just require Firebase's database tools, so select "**database**" from the menu on the left-hand side. Next click on the "**Create Database**" button, select the "**test mode**" option. Next set the database to a "**realtime database**" instead of the "**cloud firestore**" by clicking on the drop-down menu at the top. Select the "**rules**" tab and change the two "false" to "true", finally click on the "data" tab and copy the **database URL**, this will be required later on.

The last thing that you will need to do is to click on the gear icon next to the project overview, then on "project settings", then select the "service accounts" tab, finally click on "Database Secrets" and note down the security code of your database. With this step complete, you have successfully created your cloud database which can be accessed from your smartphone and from the Raspberry Pi. (Use the pictures attached above in case of any doubts, or just drop a question in the comment section)

**Graphical user interface, website

Description automatically generated**

**Graphical user interface, text, application, email

Description automatically generated**

**Graphical user interface, text, application, email

Description automatically generated**

**8. CREATING THE MOBILE APP**

The next part of the IoT system is the smartphone application. We decided to use MIT App Inventor to make our own customized app. To use the app that we created first open the following link ([MIT App Inventor](https://appinventor.mit.edu/)), which will lead you to their webpage. Next click on "create apps" towards the top of the screen, then log in with your Google account.

Create a new project just give its name and we can also import a project by given below

.Open the "projects" tab and click on "Import project (.aia) from my computer" next select the file that you just downloaded and click "ok". In the components window, scroll all the way down till you see "FirebaseDB1", click on it and modify the "Firebase Token", "FirebaseURL" to the values that you had kept a note of in the previous step. Once these steps are complete you are ready to download and install the app. You can download the app directly onto your phone by clicking on the "Build" tab and clicking on "App (provide QR code for .apk)" then scanning the QR code with your smartphone or clicking "App (save .apk to my computer)" you will download the apk file onto your computer which one can then shift to your smartphone.

**Graphical user interface

Description automatically generated**

**Graphical user interface, application

Description automatically generated**

**Qr code

Description automatically generated**

**9. PROGRAMMING THE RASPBERRI PI**

The Raspberry Pi is used for two primary reasons.

* It transmits a live video stream from the robot to a web server. This stream can be viewed by the user using the mobile app.
* It reads the updated commands on the firebase database and instructs the Arduino to perform the required tasks.

For setting up the Raspberry Pi to live-stream, a detailed tutorial already exists and can be found [here](https://elinux.org/RPi-Cam-Web-Interface). The instructions boil down to three simple commands. Switch on the Raspberry Pi and open the terminal and enter the following commands.

* git clone <https://github.com/silvanmelchior/RPi_Cam_Web_Interface.git>
* cd RPi\_Cam\_Web\_Interface
* ./install.sh

Once the installation is complete, restart the Pi and you should be able to access the stream by searching <http://your> Pi's IP address on any web browser.

With the live streaming set up, you will need to download and install certain libraries to be able to use the cloud database. Open up a terminal on your Pi and enter the following commands:

* sudo pip install requests==1.1.0
* sudo pip install python-firebase

Finally, download the python file attached below and save it on your Raspberry Pi. On the fourth line of the code change the COM port to the port the Arduino is connected to. Next, change the URL on line 8 to the firebase URL you had kept a note of earlier. Finally, run the program through the terminal. This program fetches the commands from the cloud database and relays it to the Arduino through the serial connection.

########################################CODE###############################

from firebase import firebase

import serial

ser = serial.Serial('/dev/ttyACM0', 9600)

firebase = firebase.FirebaseApplication('Firebase URL', None)

while True:

direction = firebase.get('/iot-pet-monitoring-system', 'data')

print direction

ser.write(str(direction))

**Graphical user interface, text, application, email

Description automatically generated**

1. **.PROGRAMMING THE ARDUINO**

The Arduino is used to interpret the commands from the Pi and instructs the actuators on the robot to perform the needed tasks. Download the Arduino code attached below and upload it onto the Arduino. Once the Arduino is programmed, connect it to one of the Pi's USB ports using the dedicated USB cable**.**

**Graphical user interface, text, application

Description automatically generated**

**Graphical user interface, text

Description automatically generated**

The robot will be powered off a 3 cell lipo battery. The battery terminals need to be split into two, where one goes directly to the CNC shield to power the motors, while the other gets connect to the 5v UBEC, which created a steady 5v power line which will be used to power the Raspberry Pi through the GPIO pins. The 5v from the UBEC is connected to the 5v pin of the Raspberry Pi and the GND from the UBEC is connected to the GND pin on the Pi.

**#################################CODE############################**

**#include <Servo.h>**

**#define enPin 8**

**#define lDir 5**

**#define lStep 2**

**#define rDir 6**

**#define rStep 3**

**int stepDelay = 700;**

**bool containerOpen = false;**

**Servo lid;**

**void step(bool dir, int dirPin, int stepPin){**

**digitalWrite(dirPin, dir);**

**digitalWrite(stepPin, HIGH);**

**delayMicroseconds(stepDelay);**

**digitalWrite(stepPin, LOW);**

**delayMicroseconds(stepDelay);**

**}**

**void front(int distance){**

**int steps = 300\*distance/23;**

**for (int i = 0; i<steps; i++){**

**step(true, lDir, lStep);**

**step(true, rDir, rStep);**

**}**

**}**

**void back(int distance){**

**int steps = 300\*distance/23;**

**for (int i = 0; i<steps; i++){**

**step(false, lDir, lStep);**

**step(false, rDir, rStep);**

**}**

**}**

**void left(int angle){**

**int steps = 300\*angle/23;**

**for (int i = 0; i<steps; i++){**

**step(true, lDir, lStep);**

**step(true, rDir, rStep);**

**}**

**}**

**void right(int angle){**

**int steps = 300\*angle/23;**

**for (int i = 0; i<steps; i++){**

**step(true, lDir, lStep);**

**step(true, rDir, rStep);**

**}**

**}**

**void openContainer(){**

**lid.write(100);**

**}**

**void closeContainer(){**

**lid.write(10);**

**}**

**void setup() {**

**// put your setup code here, to run once:**

**Serial.begin(9600);**

**lid.attach(9);**

**pinMode(enPin, OUTPUT);**

**pinMode(lStep, OUTPUT);**

**pinMode(lDir, OUTPUT);**

**pinMode(rStep, OUTPUT);**

**pinMode(rDir, OUTPUT);**

**digitalWrite(enPin, LOW); //pulled LOW to enable drivers**

**}**

**void loop() {**

**// put your main code here, to run repeatedly:**

**if (Serial.available() > 0){**

**char data = Serial.read();**

**if (data == '1') front(40);**

**else if (data == '2') left(5);**

**else if (data == '3') right(5);**

**else if (data == '4') back(40);**

**else if (data == '5') openContainer();**

**else if (data == '6') closeContainer();**

**}**

**}**

# CONCLUSIONS:

# Robots are an important component in Intelligent Environments

# Automate devices

# Provide physical services

# Robot Systems in these environments need particular capabilities

# Autonomous control systems

# Simple and natural human-robot interface

# Adaptive and learning capabilities

# Robots have to maintain safety during operation

# While a number of techniques to address these requirements exist, no functional, satisfactory solutions have yet been developed

# Only very simple robots for single tasks in intelligent environments exist

### **REFERENCES:**

<https://www.instructables.com/id/Arduino-and-Raspberry-Pi-Powered-Pet-Monitoring-Sy/#discuss>

<http://github.com>