IoT Based Smart Dustbin with Notifications

Manish Agarwal
180102040
IIT GUWAHATI
agarwal18@iitg.ac.in

***Abstract*—Smart Dustbins are a better alternative to the regular dustbins to improve the degrading condition of hygiene in various parts of the world. As normal dustbins require pressing of foot against its lever to open it for throwing garbage into it. Also, it needs to be monitored by a person to ensure it does not overflow. To avoid all such scenarios, we propose smart dustbin in which waste management is automated.Our system helps to manage the waste more efficiently and keeps the city clean and healthy.**

I. Introduction

The main objective of our project is to build an automated Dustbin. We use IR sensor to detect human foot tap. We use servos which helps in dustbin lid to open automatically upon detecting an object/ obstruction. We use NodeMCU for internet connectivity and microcontroller for the whole system control. In addition, we use ultrasonic sensor which will constantly keep track of garbage level in the bin. Micro-controller transmits reading data from sensor side along with its unique ID over the web to the server. Server will process this information and when level of the bin reaches the specified threshold limit municipal authority will be alerted over mobile app to take necessary step.

II. Implemented Attributes

Sensors are attached to dustbin which sends data to Arduino UNO which uses wired Serial Communication to send data to NodeMCU and also drives the actuators and LCD screen. Further the NodeMCU establishes an internet connection and sends data wirelessly to a mobile application.

A. Garbage Level Detection

To detect the garbage levels an ultrasonic sensor was placed close to the lid of the dustbin. It detects the distance of the object closest to it by sending a trigger signal using Arduino UNO. The distance measurements were then converted into garbage percent levels by mapping the distances to the dustbin's dimensions. A Green LED indicates garbage level is below the threshold whereas a Red LED indicates above threshold.

B. Opening the lid

To detect human presence we used an IR sensor module which has a HIGH output when IDLE and LOW output when an object is detected. So when a person comes near to the IR sensor on the dustbin it sends a LOW output signal and along with that the garbage percent level is taken into account. If the percent is less than the threshold and IR sensor output is LOW then the servo attached to the lid along with the shaft rotates and opens up the lid.

C. Displaying the garbage level

The data gathered from the ultrasonic sensor is processed and is displayed on an 16x2 LCD display in real time. Also this data is sent to the Node MCU though the serial communication channel. Node MCU is connected to a hotspot to wirelessly connect to the Blynk mobile application which also displays the garbage level in real time.

D. Alert notifications

The blynk mobile application displays the real time status of the dustbin and can be monitored from any location. The app gives a alert notification when the dustbin threshold is crossed.

III. Configuration Diagrams

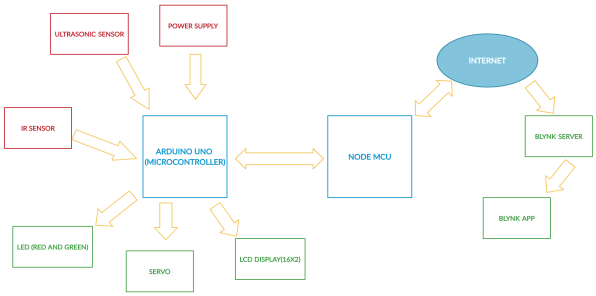

Fig. 1. Block Diagram

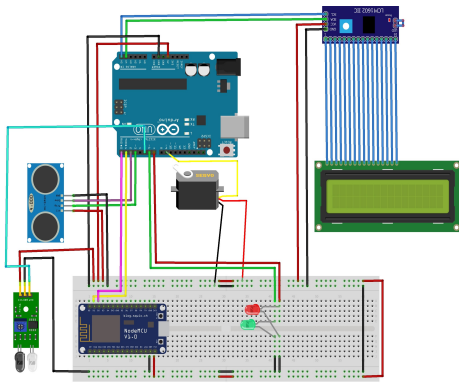

Fig. 2. Configuration Diagram for connections

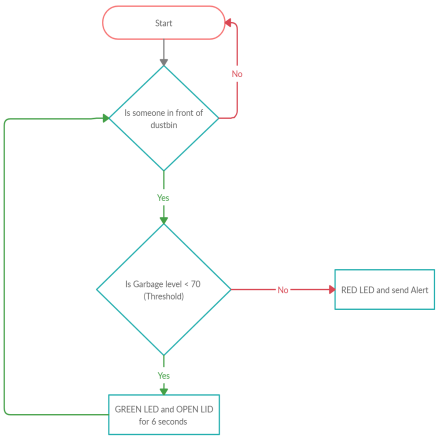

Fig. 3. Flow Diagram of Arduino Code

IV. Sample Outputs

Fig. 4. LCD display showing garbage level

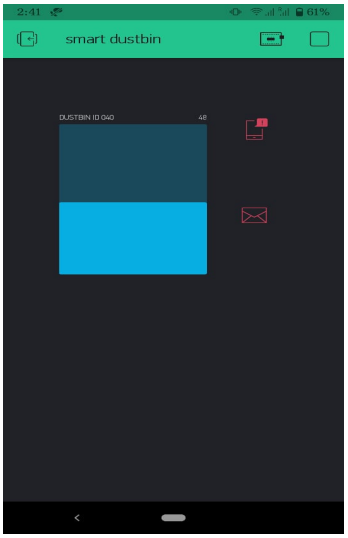

Fig. 5. App showing the live status

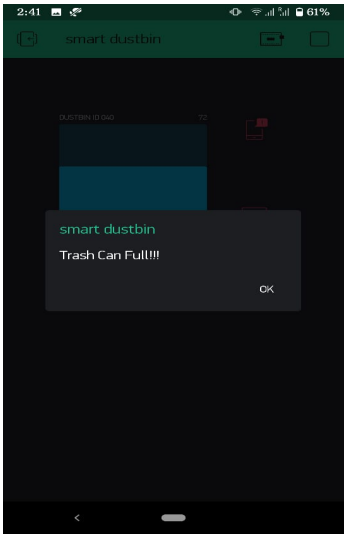

Fig. 6. App showing notification alert

V. Codes

A. *Code for Arduino UNO*

```
#include <Wire.h>
#include <Servo.h>
#include <LiquidCrystal_I2C.h>

/*
 attach pin D2 Arduino to pin Echo of HC-SR04
 attach pin D3 Arduino to pin Trig of HC-SR04
 attach pin D5 Arduino to out pin of infrared
     sensor module
 attach pin D9 Arduino to signal pin of servo
 */

#define echoPin 2
#define trigPin 3
#define IRout 5
#define servo 9
#define green 6
#define red 7

/*
 * LCD Screen with I2C module with Arduino UNO
 * LCD --> Arduino UNO
 * Vcc --> Vcc
 * GND --> GND
 * SDA --> Analog Pin 4
 * SCL --> Analog Pin 5
*/

LiquidCrystal_I2C lcd(0x27, 16, 2);
Servo myservo;

// defines variables
long duration; /*variable for the duration of
    sound wave travel*/
int distance; /* variable for the distance
    measurement*/
int percentage = 0, close_lid;
String msg;
```

```
void setup() {
  pinMode(trigPin, OUTPUT); /* Sets the trigPin
       as an OUTPUT */
  pinMode(echoPin, INPUT); /* Sets the echoPin
      as an INPUT */
  pinMode(IRout, INPUT);
  pinMode(green, OUTPUT);
  pinMode(red, OUTPUT);
  myservo.attach(servo);
  myservo.write(90);
  Serial.begin(9600);
  /* Serial Communication is starting with 9600
       of baudrate speed*/
  lcd.begin();
  lcd.backlight();
}

void loop() {
  close_lid = digitalRead(IRout);
  //myservo.write(90);
  if(close_lid == 0 && percentage < 70)
  {
    /*
    open lid for 6 seconds and then check the
        dustbin percentage
    Rotate servo 90 degrees to open the lid
    */

    myservo.write(0);
    delay(6000);
    //close the lid
    myservo.write(90);
    delay(1000);

    // Clears the trigPin condition
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    /* Reads the echoPin, returns the sound
        wave travel time in microseconds */
    duration = pulseIn(echoPin, HIGH);
    // Calculating the distance
    distance = duration * 0.034 / 2;
    percentage = map(distance, 0, 21, 100, 0);
  }
  else
  {
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    /* Sets the trigPin HIGH (ACTIVE) for 10
        microseconds */
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    /* Reads the echoPin, returns the sound
        wave travel time in microseconds */
    duration = pulseIn(echoPin, HIGH);
    // Calculating the distance
    distance = duration * 0.034 / 2;
    percentage = map(distance, 0, 21, 100, 0);
  }
  //Threshold after which dustbin won't open
  if(percentage >= 70)
  {
    digitalWrite(red, HIGH);
    digitalWrite(green, LOW);
  }
  else
  {
    digitalWrite(red, LOW);
    digitalWrite(green, HIGH);
  }

  /*
   * This Arduino UNO is connected to nodeMCU
       using the default pins as TX AND RX
   * This establishes serial communication
       between the two devices.
   * Arduino UNO <==> nodeMCU
   * Digital Pin 1(TX) => D1(RX)
   * Digital Pin 0(RX) <= D2(TX)
  */

  // Displays the percentage on the Serial
      Monitor
  Serial.println(percentage);
  msg = String(percentage, DEC);

  //Displays the percentage on the LCD screen
  lcd.clear();
  lcd.setCursor(2,0);
  lcd.print("DUSTBIN ID 040");
  lcd.setCursor(4,1);
  lcd.print(msg);
  lcd.print("%");
  lcd.print("FULL");

  delay(1000);
}
```

B. Code for NodeMCU

```
/*
NODEMCU code for the smart dustbin for serial
    communication with arduino and
sending the same to blynk app. This way we
    update the level display in the app as per
     the current
percentage occupancy of the dustbin.
THRESHOLD = 70% --> Above this level the lid
    won't open automatically until the Dusbin
    is emptied.

*/
#define BLYNK_PRINT Serial

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <SoftwareSerial.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "dWMyrz*********i53dB";

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "JioFiber-7FuAM";
char pass[] = "12345678";

BlynkTimer timer;
```

```
SoftwareSerial mySerial (D1, D2); //Rx, Tx
int percent;

BLYNK_READ(V5)
{
  // This command writes Dustbin percentage to
     Virtual Pin (5)
  Blynk.virtualWrite(V5, percent);
}

void setup()
{
  // Debug console
  Serial.begin(115200);
  mySerial.begin(9600);

  Blynk.begin(auth, ssid, pass);
  // You can also specify server:
  //Blynk.begin(auth, ssid, pass, "blynk-cloud.
     com", 80);
  //Blynk.begin(auth, ssid, pass, IPAddress
     (192,168,1,100), 8080);
}

void alert(int a)
{
  if(a>=70)
  {
    Blynk.notify("Trash Can Full!!!");
  }
}

void loop()
{
  //read value from arduino transmitted to
     nodeMCU
  String msg = mySerial.readStringUntil('\n');

  //convert that to integer
  percent = msg.toInt();

  //print on serial monitor
  Serial.println(percent);

  //run the blynk app
  Blynk.run();

  //send notification alert when dustbin level
     crosses the threshold value
  alert(percent);
  delay(1000);
}
```

VI. User Manual

IoT based Smart Dustbin with Notifications. This product can be build as per the following instructions.

A. Hardware Required

1) NodeMCU
2) Arduino UNO
3) IR sensor
4) Ultrasonic sensor
5) LCD Display
6) LED's (Green and Red)
7) Breadboard

B. Connections

- Stick the servo at the top of container just below the lid such that it opens the lid at 90 degrees rotation.
- Place IR sensor in front of the bin to detect humans close to bin.
- Place Ultrasonic sensor at the top of container facing the inside bottom of the bin.
- Now connect all these components with Arduino and NodeMCU using a breadboard as per the configuration diagram above.

C. Accessing the Mobile App

Download the Blynk Application from the app store. Create a new project named SmartBin and add notification widget and level V widget to it. Select hardware as NodeMCU and connection type as WiFi. Then upload the codes to Arduino and NodeMCU. And the product is now complete. You can now monitor the real time status of the dustbin from any location. If the dustbin gets filled the Municipalty receives a Notification and the lid won't open until the bin is emptied.

References

1) https://www.instructables.com/
2) http://docs.blynk.cc/
3) https://www.arduino.cc/en/main/docs