A PROJECT REPORT

on

BatMobile – IOT Rover

Submitted by

Mr. Ganesh Umesh Tiwari

in partial fulfillment for the award of the degree

of

BACHELOR OF SCIENCE

in

COMPUTER SCIENCE

under the guidance of

Prof. Vipul Saluja

Department of Computer Science

R. D. & S.H. National College & S. W. A. Science College

Bandra, Mumbai – 400050.

(Sem VI)

(2018 – 2019)

DECLARATION

I, Mr. Ganesh Umesh Tiwari, hereby declare that the project entitled "**Batmobile - Mars Rover**" submitted in the partial fulfillment for the award of **Bachelor of Science** in **Computer Science** during the academic year **2018 – 2019** is my original work and the project has not formed the basis for the award of any degree, associateship, fellowship or any other similar titles.

Signature of the Student:

Place:

Date:

You don’t get another chance, life is no Nintendo game.

Marshall Mathers

ACKNOWLEDGEMENT

I have taken efforts in the project. However, it would not have been possible without the kind support and help of many individuals. I would like to extend my sincere thanks to all of them.

I wish to express my grateful thanks to the co-ordinator of Computer Science Department and project guide **Prof. Vipul Saluja** who gave us full support and valuable suggestions.

I express my gratitude to my teachers, **Prof. Akbar Khan**, **Prof. Kauser Shaikh, Prof. Nisha Yadav, Prof. Saima Qureshi, Prof. Madhura Batode** for giving valuable support and suggestions.

I would also like to express my thanks to **Mr. Nimish Sane, Mr. Prasad Ingale** my fellow colleagues, who were with me throughout the project and always supported and surprised me with there amazing ideas.

I also want to thank all other faculty members and non-teaching staff **Mr. Jagdev Verma, Mr. Rajaram Batwal, Mr. Rohit Pote and Mr. Chanduram Yadav** of our department for their support and peers for having stood by us to complete this project.

TABLE OF CONTENTS

Introduction

This project is entitled for the development of a fully functional mobile prototype “the Rover Station”, responsible for environmental data capture as Temperature, Humidity and Luminosity. The idea is in the future aggregate other functions to emulate what would be a Mars Rover emulator.

This project is built upon with help of growing technological advancements in the fields of Internet of things and Wi-Fi technology. This project utilizes Wireless LAN technology for connection between objects and MQTT protocol for transfer of data between the devices.

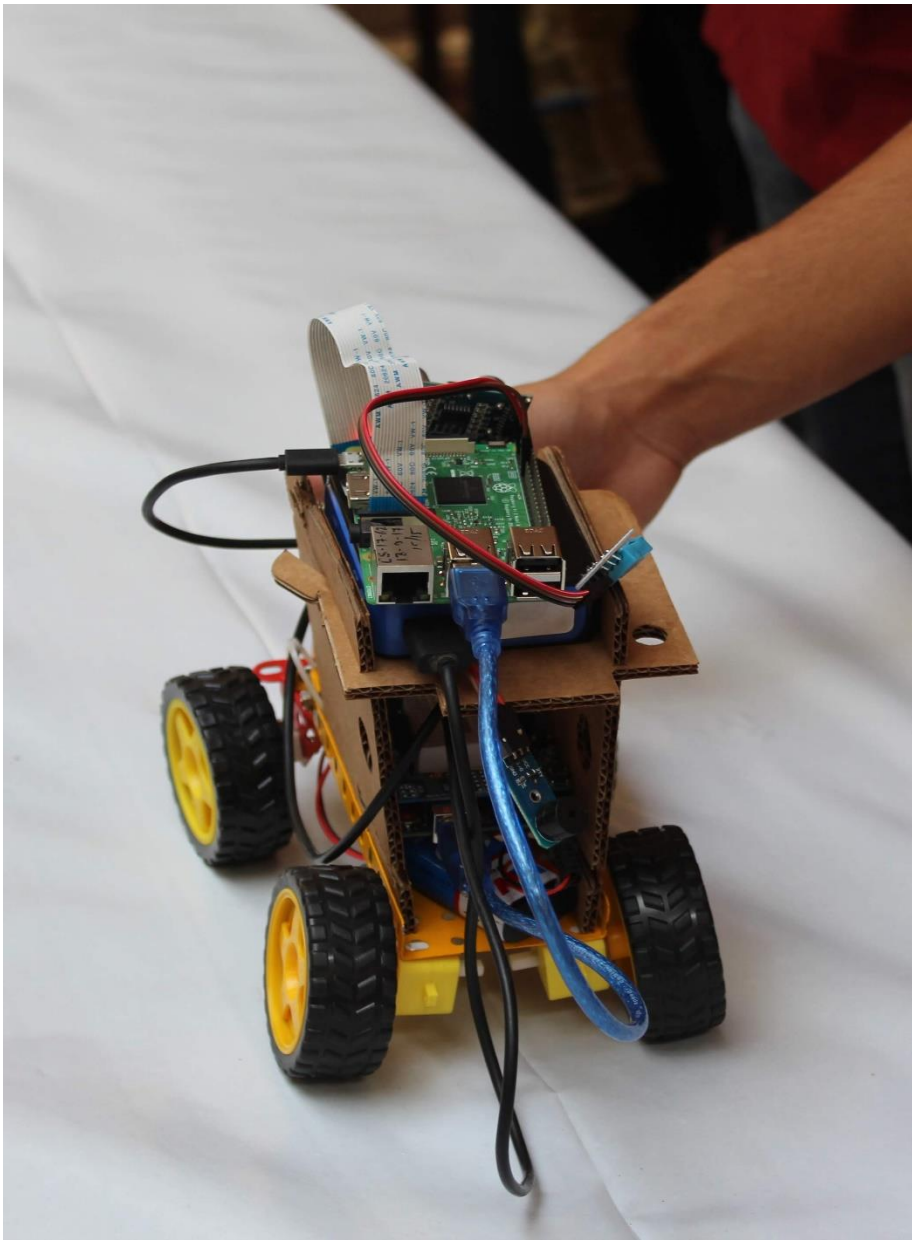


Figure 1 Earlier Beta Model Built by us.

Objective

This project was built with having certain ideas in mind, it can be used for some ideal situations like monitoring over things where humans can't reach or it is difficult to reach there. It can also be used as a harmless transport vehicle for small things to transport things from one place to another. It can be used to gather information about a place which is in remote location so that certain tasks can be achieved without actually getting a human intervention.

Purpose of the project

Purpose of this project was not actually about for making any discoveries or anything but it was for a Competition which was held in our college. We were excited to participate in the competition and to represent our college in front of other colleges and be able to see what others are building in their project. I was always eager to learn about Internet of Things and how different things if connected together and made available on internet can actually make a difference in the whole world.

Components Used in the project.

Raspberry Pi: -

The Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote teaching of basic computer science in schools and in developing countries.[5][6][7] The original model became far more popular than anticipated,[8] selling outside its target market for uses such as robotics. It does not include peripherals (such as keyboards and mice) and cases. However, some accessories have been included in several official and unofficial bundles.

Arduino: -

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

Jumper Cables: -

A jump wire (also known as jumper wire, or jumper) is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally or with other equipment or components, without soldering.

Raspberry Pi Camera Module: -

The Raspberry Pi Camera Module is a 5MP CMOS camera with a fixed focus lens that is capable of capturing still images as well as high definition video. Stills are captured at a resolution of 2592 x 1944, while video is supported at 1080p at 30 FPS, 720p at 60 FPS and 640x480 at 60 or 90 FPS. The camera is supported in the latest version of Raspbian, Raspberry Pi's preferred operating system.

Ultrasonic Sensor: -

As the name indicates, ultrasonic sensors measure distance by using ultrasonic waves. The sensor head emits an ultrasonic wave and receives the wave reflected back from the target. Ultrasonic Sensors measure the distance to the target by measuring the time between the emission and reception.

DHT Sensor: -

The DHT sensors are made of two parts, a capacitive humidity sensor and a thermistor. There is also a very basic chip inside that does some analog to digital conversion and spits out a digital signal with the temperature and humidity. The digital signal is fairly easy to read using any microcontroller.

Power Supply: -

Power banks can be defined as portable batteries that use circuitry to control any power in and power out. They can charge up using a USB charger when power is available, and then used to charge battery powered items like mobile phones and a host of other devices that would normally use a USB charger.

Lithium batteries are primary batteries that have metallic lithium as an anode. These types of batteries are also referred to as lithium-metal batteries.

They stand apart from other batteries in their high charge density (long life) and high cost per unit. Depending on the design and chemical compounds used, lithium cells can produce voltages from 1.5 V (comparable to a zinc–carbon or alkaline battery) to about 3.7 V. Disposable primary lithium batteries must be distinguished from secondary lithium-ion, lithium iron phosphate and lithium-polymer,[1] which are rechargeable batteries. Lithium is especially useful, because its ions can be arranged to move between the anode and the cathode, using an intercalated lithium compound as the cathode material but without using lithium metal as the anode material. Pure lithium will instantly react with water, or even moisture in the air; the lithium in lithium ion batteries is in a less reactive compound.

Lithium batteries are widely used in portable consumer electronic devices, and in electric vehicles ranging from full sized vehicles to radio controlled toys.

System Analysis

It is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components.

System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem-solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose.

The field of system analysis relates closely to requirements analysis or to operations research. It is also "an explicit formal inquiry carried out to help a decision maker identify a better course of action and make a better decision than she might otherwise have made."

The terms analysis and synthesis stem from Greek, meaning "to take apart" and "to put together," respectively. These terms are used in many scientific disciplines, from mathematics and logic to economics and psychology, to denote similar investigative procedures. Analysis is defined as "the procedure by which we break down an intellectual or substantial whole into parts," while synthesis means "the procedure by which we combine separate elements or components in order to form a coherent whole." [3] System analysis researchers apply methodology to the systems involved, forming an overall picture.

System analysis is used in every field where something is developed. Analysis can also be a series of components that perform organic functions together, such as system engineering. System engineering is an interdisciplinary field of engineering that focuses on how complex engineering projects should be designed and managed.

Advantages and Disadvantages of Arduino

Advantages: -

1- Ready to Use:

The biggest advantage of Arduino is its ready to use structure. As Arduino comes in a complete package form which includes the 5V regulator, a burner, an oscillator, a micro-controller, serial communication interface, LED and headers for the connections. You don't have to think about programmer connections for programming or any other interface. Just plug it into USB port of your computer and that's it. Your revolutionary idea is going to change the world after just few words of coding.

2- Examples of codes:

Another big advantage of Arduino is its library of examples present inside the software of Arduino. I'll explain this advantage using an example of voltage measurement. For example, if you want to measure voltage using ATmega8 micro-controller and want to display the output on computer screen then you have to go through the whole process. The process will start from learning the ADC's of micro-controller for measurement, went through the learning of serial communication for display and will end at USB - Serial converters.

3- Effortless functions:

During coding of Arduino, you will notice some functions which make the life so easy. Another advantage of Arduino is its automatic unit conversion capability. You can say that during debugging you don't have to worry about the unit’s conversions. Just use your all force on the main parts of your projects. You don't have to worry about side problems.

4- Large community:

There are many forums present on the internet in which people are talking about the Arduino. Engineers, hobbyists and professionals are making their projects through Arduino. You can easily find help about everything. Moreover, the Arduino website itself explains each and every functions of Arduino.

Disadvantages: -

1- Structure:

Yes, the structure of Arduino is its disadvantage as well. During building a project you have to make its size as small as possible. But with the big structures ofArduino we have to stick with big sized PCB's. If you are working on a small micro-controller like ATmega8 you can easily make your PCB as small as possible.

2- Easy to use:

In my opinion, if you started your journey of micro-controllers with Arduino then it will be very difficult for you to make the complex intelligent circuitries in future. The easy to use hardware/software of Arduino unable a person to learn the basics of many things likes Serial communication, ADC, I2C etc. This basically makes you think that microcontrollers are easy which is not the case.

Advantages and Disadvantages of Raspberry Pi

Advantages: -

- You can install a fully-fledged Operating System (e.g. Raspbian which is Linux-based or Windows IOT Core which is Windows based OS) and use it as a day to day computer
- The presence of GPIO (General Purpose Input Output pins) is what distinguishes a RPi from traditional computers. You can connect these pins to sensors and external components and interact with them programmatically using a language such as Python. This allows you to build and prototype Internet of Things devices that can sense the real world
- Newer models like the RPi has WIFI and Bluetooth built in. This allows you to take projects into wireless mode easily.

Disadvantages: -

- There is not any fuse protection on the Rpi, so if you connect pins incorrectly, you can damage the board
- It is not as fast in terms of CPU processing speed nor does it have as much memory as traditional PC or laptops. But for $35, it packs quite a punch
- There is no built in analog to digital conversion on the GPIO pins like there is in Arduino. So you need to have an ADC chip to work with analog signals when you need more accuracy

Circuit Diagram

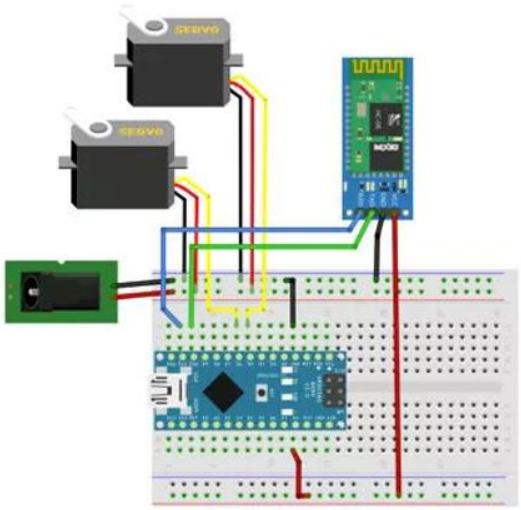


Figure 2 Arduino's connection with ultrasonic sensor and two servo motors

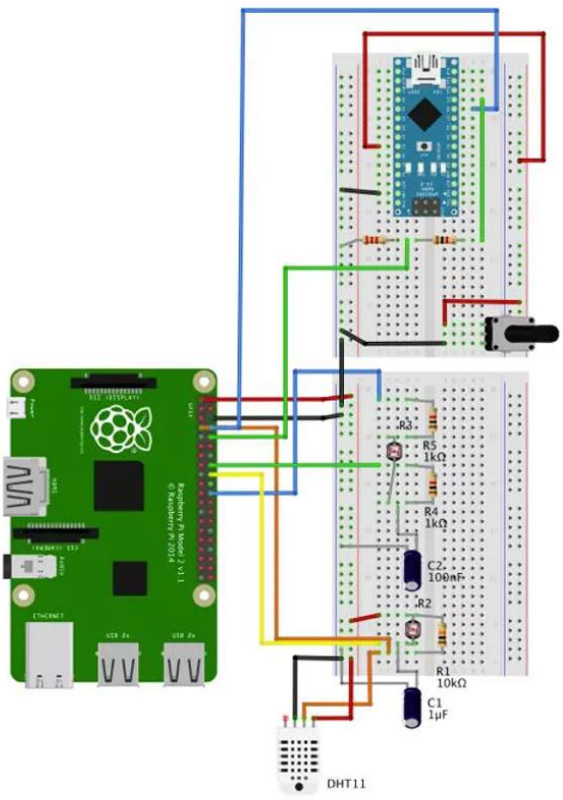


Figure 3 Raspberry Pi Connection with dht sensor

Design of the project

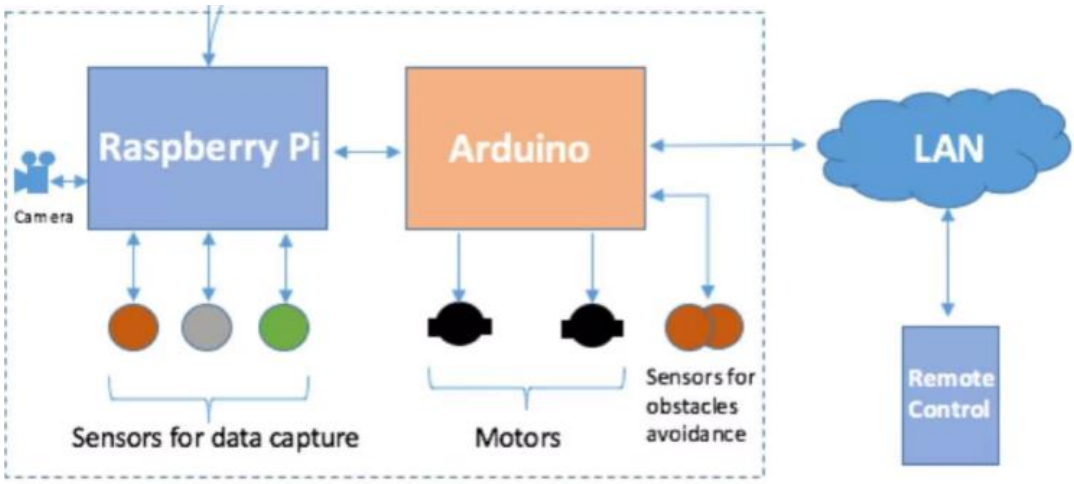


Figure 4 Design of the Whole Project

Applications of Raspberry Pi

The different applications of the raspberry pi model are

- Media steamer
- Tablet computer
- Home automation
- Internet radio
- Controlling robots
- Cosmic Computer
- Arcade machines
- Raspberry-pi based projects

Applications of Arduino Uno

Arduino is a micro controller board. The micro cotroller in popular arduino board has 20 I/O pins and Flash Program Memory of 32k (may be less based on different models)

You can do every thing that a 32k micro controller with 20 I/Os can do.

Some examples where I have used arduino are as below -

1. Traffic Light Count Down Timer
2. Parking Lot Counter
3. Weighing Machines
4. Medical Instrument
5. Emergency Light for Railways

Apart from these, there are lot of places where some thing equal to Arduino is being used (May be some one is also using Arduino there)

1. Window Aircon controller
2. Washing Machine
3. Microwave Over
4. Security Systems
5. CCTV Switchers

Requirement Analysis

Requirements Analysis is the process of defining the expectations of the users for an application that is to be built or modified. Requirements analysis involves all the tasks that are conducted to identify the needs of different stakeholders. Therefore requirements analysis means to analyze, document, validate and manage software or system requirements. High-quality requirements are documented, actionable, measurable, testable, traceable, helps to identify business opportunities, and are defined to a facilitate system design.

- **Requirements analysis process**
  The requirements analysis process involves the following steps

- **Eliciting requirements**
  The process of gathering requirements by communicating with the customers is known as eliciting requirements.

- **Analyzing requirements**
  This step helps to determine the quality of the requirements. It involves identifying whether the requirements are unclear, incomplete, ambiguous, and contradictory. These issues resolved before moving to the next step.

- **Requirements modelling**
  In Requirements modelling, the requirements are usually documented in different formats such as use cases, user stories, natural-language documents, or process specification.

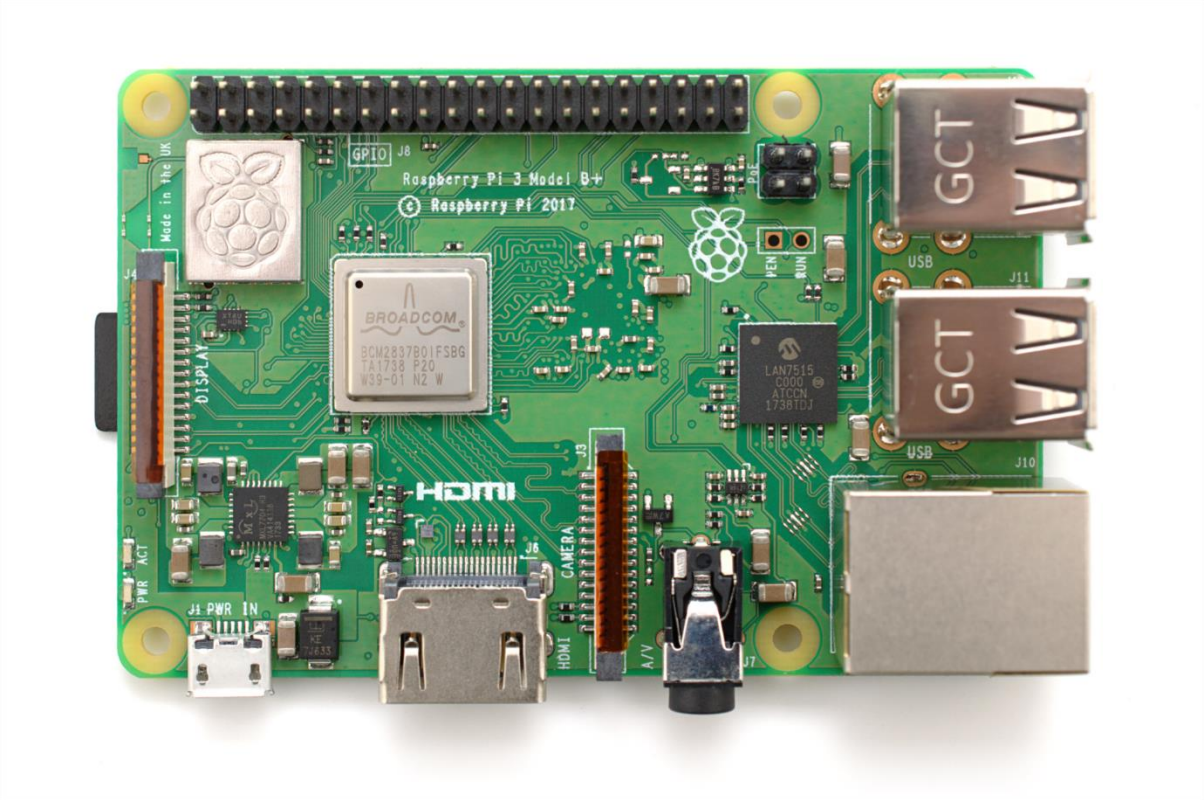
- **Review and retrospective**
  This step is conducted to reflect on the previous iterations of requirements gathering in a bid to make improvements in the process going forward.

After carrying out requirement analysis for our project we came out with a certain list of the things which I needed to make sure are present in order for this project to be in a working Condition. They are already mentioned in other places as well as in this documentation but I will list them here for convenience.

1. Raspberry Pi Model 3B
2. Arduino Nano
3. Jumper Cables
4. Power Supply (Power Bank as well as rechargeable Lithium Battery's)
5. Raspberry Camera Module
6. DHT11 Sensor

Raspberry Pi Model 3B

This model of Raspberry is specifically needed because it currently supports WIFI as well as all the things that are necessary for the project to be working properly. The latest Raspberry Pi 3 Model B+ has a faster 64-bit 1.4GHz quad core processor, 1GB of RAM, faster dual-band 802.11 b/g/n/ac wireless LAN, Bluetooth 4.2, and significantly faster 300Mbit/s ethernet.



Arduino Uno

Arduino Uno is necessary for this project as 2 servo motors are to be turned accordingly and this requires a lot of current and power which cannot be directly served through raspberry pi as it doesn't hold that much current and would probably blow up its circuit. The Arduino Uno is a small, complete, and breadboard-friendly board based on the ATmega328P (Arduino Uno 3.x). It has more or less the same functionality of the Arduino Duemilanove, but in a different package. It lacks only a DC power jack, and works with a Mini-B USB cable instead of a standard one.

Jumper Cables

Jumper cables are necessary for connecting different devices and equipments each other and it is very much necessary when connecting different iot devices each other. A jump wire is an electrical wire, or group of them in a cable, with a connector or pin at each end which is normally used to interconnect the components.

Power Supply

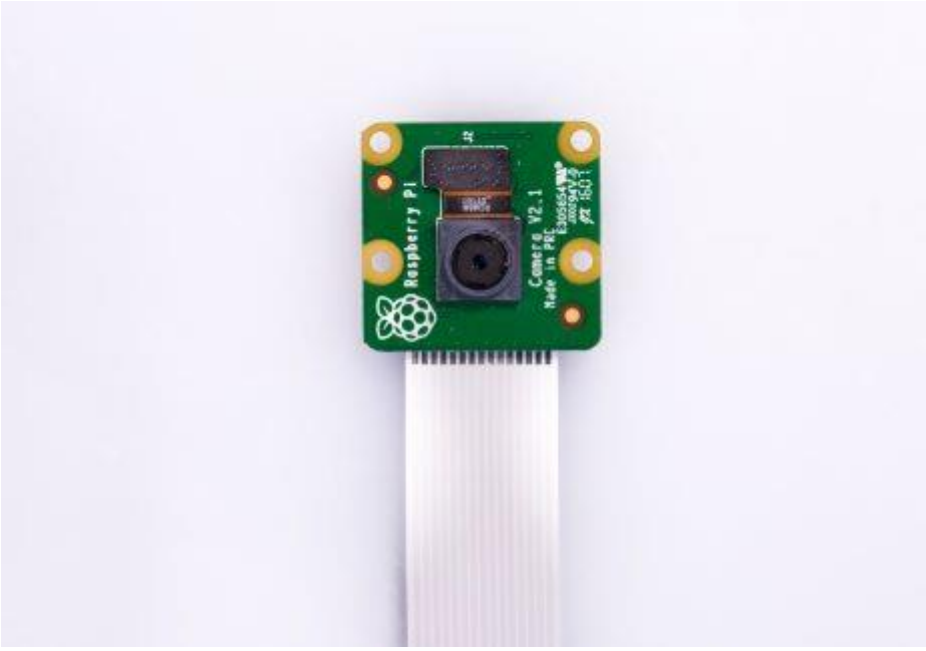
Power Supply is very much necessary for any iot device and it is an integral part of a sensor node. Power supply needed in this project will be a powerbank to power raspberry pi and two lithium cell batteries to power Arduino nano.

Raspberry Pi Camera Module

Raspberry Pi camera module is necessary for our project as it will be used for gaining live stream that will be routed to apps that will be controlling the rover. The Raspberry Pi Camera Module is a custom designed add-on for Raspberry Pi. It attaches to Raspberry Pi by way of one of the small sockets on the board upper surface. This interface uses the dedicated CSi interface, designed especially for interfacing to cameras.

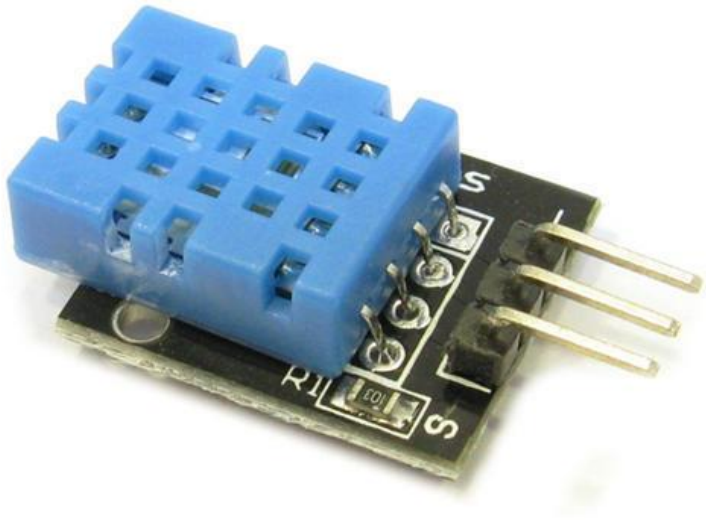
The board itself is tiny, at around 25mm x 20mm x 9mm. It also weighs just over 3g, making it perfect for mobile or other applications where size and weight are important. It connects to Raspberry Pi by way of a short ribbon cable.

The sensor itself has a native resolution of 5 megapixel, and has a fixed focus lens onboard. In terms of still images, the camera is capable of 2592 x 1944 pixel static images, and also supports 1080p30, 720p60 and 640x480p60/90 video.



DHT11 sensor

DHT11 sensor is necessary for our project as it is used to gain information about surrounding systems and is integral part of our project. The DHT11 sensor can either be purchased as a sensor or as a module. Either way, the performance of the sensor is same. The sensor will come as a 4-pin package out of which only three pins will be used whereas the module will come with three pins as shown above.
The only difference between the sensor and module is that the module will have a filtering capacitor and pull-up resistor inbuilt, and for the sensor, you have to use them externally if required.



Ultrasonic Sensor

Ultrasonic sensor is also a necessary part of our project as it is used to see if there is any obstacle present in the rover's way as it moves along the way and see whether it is near to hit any obstacle. ultrasonic sensors measure distance by using ultrasonic waves. The sensor head emits an ultrasonic wave and receives the wave reflected back from the target. Ultrasonic Sensors measure the distance to the target by measuring the time between the emission and reception.

System Installation and Processing

Installing Raspbian Operating System

Download the image

Official images for recommended operating systems are available to download from the Raspberry Pi website Downloads page.

Alternative distributions are available from third-party vendors.

If you're not using Etcher (see below), you'll need to unzip .zip downloads to get the image file (.img) to write to your SD card.

Note: the Raspbian with Raspberry Pi Desktop image contained in the ZIP archive is over 4GB in size and uses the ZIP64) format. To uncompress the archive, a unzip tool that supports ZIP64 is required. The following zip tools support ZIP64:

- 7-Zip (Windows)
- The Unarchiver (Mac)
- Unzip (Linux)

Writing an image to the SD card
Before you start, don't forget to check the SD card requirements.

You will need to use an image writing tool to install the image you have downloaded on your SD card.

Etcher is a graphical SD card writing tool that works on Mac OS, Linux and Windows, and is the easiest option for most users. Etcher also supports writing images directly from the zip file, without any unzipping required. To write your image with Etcher:

- Download Etcher and install it.
- Connect an SD card reader with the SD card inside.
- Open Etcher and select from your hard drive the Raspberry Pi .img or .zip file you wish to write to the SD card.
- Select the SD card you wish to write your image to.
- Review your selections and click 'Flash!' to begin writing data to the SD card.

Installing Arduino IDE for Raspberry Pi

The first step in programming an Arduino board with a Raspberry Pi is to install the Arduino IDE (integrated development environment) on your Raspberry Pi. This program checks code and loads it onto the Arduino. Install the latest version of Arduino IDE using apt:

```
sudo apt-get update && sudo apt-get upgrade
sudo apt-get install Arduino
```

Alternatively, open Chrome on your Raspberry Pi, head to magpi.cc/2tPw8ht, and click the Linux ARM link under 'Download the IDE'. Extract the file to your /opt directory , then open a Terminal and run the install.sh script to install.

```
cd Downloads/
tar -xf arduino-1.8.3-linuxarm.tar.xz
sudo mv arduino-1.8.3 /opt
sudo /opt/arduino-1.8.3/install.sh
```

You will find Arduino IDE under Menu > Programming. Open the app to start programming your Arduino board.

Installing MJPEG Streamer

To install Mjpeg Streamer it is necessary to have all the components that are installed on the system to be up to date.

```
sudo apt-get update
sudo apt-get upgrade -y
```

In order to make Mjpeg streamer to work properly on your system you should have

- Build essentials
- Libjpeg
- Imagemagick
- Libv4l-devl
- cmake

They can be installed by a single command

```
sudo apt-get install build-essential libjpeg8-dev imagemagick libv4l-dev cmake -y
```

Mjpeg Streamer is freely and openly available software developed on github and these are the necessary steps required to build the Mjpeg streamer directly from the source itself

```
cd /tmp
git clone https://github.com/jacksonliam/mjpg-streamer.git
cd mjpg-streamer/mjpg-streamer-experimental
make
sudo make install
```

Installing Mosquitto MQTT Broker

Installing mosquitto mqtt broker is a piece of cake as it is also a freely available software which is basically used in most of IOT projects.

```
sudo apt-get install mosquitto mosquitto-clients
```

This single line of code does all of the work and install mosquitto mqtt broker on raspberry pi

Installing python libraries of MQTT and PySerial on raspberry

Pip (pip) is a package management system used to install and manage software packages written in Python. Many packages can be found in the default source for packages and their dependencies — Python Package Index. Python 2.7.9 and later, and Python 3.4 and later include pip by default.

Pip is used to install all the packages and libraries in python.

MQTT can be installed as

```
pip install paho-mtt
```

PySerial can be installed as

```
pip install PySerial
```

PySerial can be installed from Conda:

```
conda install pyserial
```

Android Libraries for Application (Mqtt, Barcode Scanning, Simple Mjpeg Viewer)

In order to use a Support Library, you must modify your application's project's classpath dependencies within your development environment. You must perform this procedure for each Support Library you want to use.

To add a Support Library to your application project:

Include Google's Maven repository in your top-level build.gradle file.

```
allprojects {
    repositories {
        google()
        jcenter()

        // If you're using a version of Gradle lower than 4.1, you must
        // instead use:
        //
        // maven {
        //     url 'https://maven.google.com'
        // }
    }
}
```

Add the support library to the dependencies section. For example, to add the v4 core-utils library, add the following lines:

```
dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation"org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
    implementation 'com.android.support:appcompat-v7:28.0.0'
    implementation 'com.android.support.constraint:constraint-layout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:runner:1.0.2'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'

    //exoplayer
    implementation 'com.google.android.exoplayer:exoplayer:2.9.3'
    //anko commons
    implementation "org.jetbrains.anko:anko-commons:$anko_version"
    implementation 'com.github.niqdev:mjpeg-view:1.6.0'
    implementation 'com.android.volley:volley:1.1.1'
    implementation 'com.google.code.gson:gson:2.8.5'

    //mqtt necessary libraries
    implementation 'org.eclipse.paho:org.eclipse.paho.client.mqttv3:1.1.0'
    implementation 'org.eclipse.paho:org.eclipse.paho.android.service:1.1.1'
```

```
    //android network tools necessary to find rover on the same lan
    implementation 'com.github.stealthcopter:AndroidNetworkTools:0.4.3'

    //this is barcode scanner library necessary for scanning barcode of the rover
    implementation 'com.journeyapps:zxing-android-embedded:3.6.0'

    //this is the custom view we would be using to play the Mjpeg video streaming.
    implementation 'com.github.dydwo92:Android-Simple-MjpegViewer:0.0'
}
```

Gradle sync is a gradle task that looks through all of your dependencies listed in your build.gradle files and tries to download the specified version. It requires an internet connection because it is usually downloading these dependencies from a remote location. You can define what ports it uses by changing your gradle.properties.

So after adding all these repositories in the (app) build.gradle file please sync the project to download all the necessary repository and codes which will be necessary while building the application.

Programming Arduino: -

```
#include <Servo.h>

#define RIGHT 1
#define LEFT -1

Servo leftServo;
Servo rightServo;

const byte ledPin = 13;
const byte buttonPin = 9;

const byte power = 500;

int adj = 1;

//--------------------------------------------
void motorStop(int time =200)
{
  leftServo.writeMicroseconds(1500);
  rightServo.writeMicroseconds(1500);
  delay(time);
}

//--------------------------------------------
void motorForward()
{
  leftServo.writeMicroseconds(1500 - power);
  rightServo.writeMicroseconds(1500 + power*adj);
}

//--------------------------------------------
void motorFwTime (unsigned int time)
{
  motorForward();
  delay (time);
  motorStop();
}

//--------------------------------------------
void motorBackward()
{
  leftServo.writeMicroseconds(1500 + power);
  rightServo.writeMicroseconds(1500 - power);
}
```

```
//-----------------------------------------
void motorTurn(int direction, int time)
{
  leftServo.writeMicroseconds(1500 - power*direction);
  rightServo.writeMicroseconds(1500 - power*direction);
  delay (time);
  motorStop();
}

//-----------------------------------------
void setup()
{
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);

  leftServo.attach(6);
  rightServo.attach(5);

  while(digitalRead(buttonPin))
 {
 }

 motorTurn (LEFT, 500);
 motorTurn (RIGHT, 500);

}



void loop()
{

}
```

Programming Raspberry Pi: -

```
import Adafruit_DHT

# Set sensor type : Options are DHT11,DHT22 or AM2302
sensor=Adafruit_DHT.DHT11

# Set GPIO sensor is connected to
gpio=17


def getHumidAndTemp():
    # Use read_retry method. This will retry up to 15 times to
    # get a sensor reading (waiting 2 seconds between each retry).
    humidity, temperature = Adafruit_DHT.read_retry(sensor, gpio)

    # Reading the DHT11 is very sensitive to timings and occasionally
    # the Pi might fail to get a valid reading. So check if readings are valid.
    if humidity is not None and temperature is not None:
        print('Temp={0:0.1f}*C  Humidity={1:0.1f}%'.format(temperature, humidity))
        return {'temp':temperature,'humidity':humidity,'result':'passed'}
    else:
        return {'result':'failed'}
```

Figure 5 Programming Raspberry Pi for getting temperature from dht11 sensor

```
import paho.mqtt.client as paho
import threading
import time

def on_connect(client, userdata, flags, rc):
    print("connected: ", str(rc))

def on_message(client, obj, msg):
    print(" recieved " , str(msg.qos) ," " , str(msg.payload,encoding="utf-8"))

def on_publish(client, obj, mid):
    print("mid: " + str(mid))

def on_subscribe(client, obj, mid, granted_qos):
    print("Subscribed: " + str(mid) + " " + str(granted_qos))

def listen(client):
    client.subscribe("directions")
    client.loop_forever()

def publish(client,temp=None,humidity=None,obstacle=None):

    if(temp!=None):
        client.publish("temp",payload=temp,qos=1)
    if(humidity!=None):
        client.publish("humidity",payload=humidity,qos=1)
    if(obstacle!=None):
        client.publish("obstacle",payload=obstacle,qos=1)
    print("published the data")

def registerFunstions(client):
    client.on_connect = on_connect
    client.on_message = on_message
    client.on_subscribe = on_subscribe
    client.on_publish = on_publish


if __name__ == '__main__':
    client = paho.Client("rover")
    registerFunstions(client)
    client.connect("127.0.0.1",1883)

    publisher = paho.Client("publisher")
    registerFunstions(publisher)
    publisher.connect("127.0.0.1",1883)

    t1  = threading.Thread(target=listen,args=(client,))
    t2 = threading.Thread(target=publish,args=(publisher,),kwargs={"temp":32})
    t1.start()
    time.sleep(1)
    t2.start()
```

Figure 6 Programming Raspberry Pi for Paho MQTT

Programming Android App: -

MainActivity.kt

```
package `in`.webxstudio.batmobilecontroller

import android.content.Intent
import android.content.pm.PackageManager
import android.support.v7.app.AppCompatActivity
import android.os.Bundle
import android.support.v4.app.ActivityCompat
import android.support.v4.content.ContextCompat
import android.util.Log
import android.view.View
import android.widget.Toast
import com.google.zxing.integration.android.IntentIntegrator
import com.stealthcopter.networktools.SubnetDevices
import com.stealthcopter.networktools.subnet.Device
import kotlinx.android.synthetic.main.activity_main.*
import org.eclipse.paho.client.mqttv3.MqttClient
import org.jetbrains.anko.AnkoLogger
import org.jetbrains.anko.doAsync
import org.jetbrains.anko.uiThread
import java.util.ArrayList

class MainActivity : AppCompatActivity(), AnkoLogger
{
    var scannedMacAddress = ""
    val port = "12345"
    val scannerIntegrator = IntentIntegrator(this)
    val regex = "[0-9a-f]{2}([-:]?)[0-9a-f]{2}(\\\\1[0-9a-f]{2}){4}\$"
    val TAG="MainActivity"
    var ip_address = "192.168.0.102"
    val mqttPort = 1883
    val streamUrl = "http://$ip_address:8080/?action=stream"
    lateinit var mqttClient:MqttClient
    val controller = MqttController()

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        up_button.setOnClickListener {
            //move ahead
            controller.sendDirection("forward",mqttClient)
        }
```

```
        down_button.setOnClickListener {
            //backwards
            controller.sendDirection("backward",mqttClient)
        }

        right_button.setOnClickListener {
            //move right
            controller.sendDirection("right",mqttClient)
        }

        left_button.setOnClickListener {
            //move left
            controller.sendDirection("left",mqttClient)
        }



if(ContextCompat.checkSelfPermission(this,android.Manifest.permission.CAMERA)==Pa
ckageManager.PERMISSION_GRANTED){
            scannerIntegrator.setOrientationLocked(true)
            scannerIntegrator.initiateScan()
        }else{
            ActivityCompat.requestPermissions(this,
arrayOf(android.Manifest.permission.CAMERA),101)
        }
    }

    fun setupVideoPlayer(videoUrl:String){
        video_view.Start(videoUrl)
    }

    override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {
        super.onActivityResult(requestCode, resultCode, data)
        val result = IntentIntegrator.parseActivityResult(requestCode,resultCode,data)
        if(result!=null){
            if(result.contents ==null)
                Toast.makeText(this,"Cancelled",Toast.LENGTH_SHORT).show()
            else{
                scannedMacAddress = result.contents
                Log.d(TAG,"Scanned Mac Address $scannedMacAddress")
                findDevices()
            }
        }
    }

    var devicesList = ArrayList<Device>()
```

```
    fun findDevices(){
        doAsync {
            SubnetDevices.fromLocalAddress().findDevices(object :
SubnetDevices.OnSubnetDeviceFound{
                override fun onFinished(devicesFound: ArrayList<Device>?) {
                    devicesList = devicesFound!!
                    Log.d(TAG,"Search Completed")
                    uiThread {
                        setUpAccordingly()
                    }

                }

                override fun onDeviceFound(device: Device?) {
                    Log.i(TAG,"Device found ${device!!.ip} ${device!!.mac}")
                }
            })
        }
    }

    fun setUpAccordingly(){
        var deviceFound = false
        for(device in devicesList){
            if (device.mac != null){
                if (device.mac.toUpperCase() == scannedMacAddress){
                    deviceFound = true
                    ip_address = device.ip
                }
            }
        }
        if(deviceFound){
            Log.d(TAG,"Rover found successfully at $ip_address")
            Toast.makeText(applicationContext,"Rover Found starting
services",Toast.LENGTH_SHORT).show()
            progressBar.visibility = View.INVISIBLE
            Log.d(TAG,"Starting streaming service")
            setupVideoPlayer(streamUrl)
            mqttClient = controller.createController(ip_address)
            controller.getData(mqttClient,temp_text,humidity_text,obstacle_text)
        }
        else{
            Log.d(TAG,"Rover not found")
            Toast.makeText(applicationContext,"Rover not
found",Toast.LENGTH_SHORT).show()
            progressBar.visibility = View.INVISIBLE
        }
    }
```

```
    override fun onRequestPermissionsResult(requestCode: Int, permissions: Array<out
String>, grantResults: IntArray) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults)
        if (grantResults.isNotEmpty() &&
grantResults[0]==PackageManager.PERMISSION_GRANTED){
            IntentIntegrator(this).initiateScan()
        }else{
            Toast.makeText(this,"Camera Permission is necessary please allow to continue
using app",Toast.LENGTH_SHORT).show()
        }
    }

    fun createMqttConnection(){

    }

    override fun onResume() {
        super.onResume()

    }

    override fun onPause() {
        super.onPause()
    }
}
enum class Direction{
    Forward,Backward,Right,Left
}
```

MQTTController.kt

```
package `in`.webxstudio.batmobilecontroller

import android.util.Log
import android.widget.TextView
import org.eclipse.paho.client.mqttv3.*
import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence

val TAG = "MqttController"

class MqttController{
    lateinit var mqttClient: MqttClient

    val temperature = "temp"
```

```
    val humidity = "humidity"
    val obstacle = "obstacle"

    val directions = "directions"

    fun createController(serverUrl:String): MqttClient {
        Log.d(TAG,"ServerUrl is $serverUrl")
        mqttClient =
MqttClient("tcp://$serverUrl:1883","rover_client",MemoryPersistence())
        val options = MqttConnectOptions()
        options.isCleanSession = true
        Log.d(TAG,"Connecting to server $serverUrl")
        mqttClient.connect(options)
        Log.d(TAG,"Connected")
        return mqttClient
    }

    fun sendDirection(direction:String,client: MqttClient){
        client.publish("directions",direction.toByteArray(),1,true)
        Log.d(TAG,"Message Published")
    }

    fun getData(client: MqttClient,
                tempTextview:TextView,
                humidityTextView: TextView,
                obstacleTextView:TextView){
        client.subscribe(temperature)
        client.subscribe(humidity)
        client.subscribe(obstacle)
        client.setCallback(object : MqttCallback{
            override fun messageArrived(topic: String?, message: MqttMessage?) {
                Log.d(TAG,"Message received")
                when(topic){
                    temperature ->{
                        if (message!=null)
                            tempTextview.text = message.payload.toString()
                    }
                    humidity ->{
                        if (message!=null)
                            humidityTextView.text = message.payload.toString()
                    }
                    obstacle->{
                        if (message!=null)
                            obstacleTextView.text = message.payload.toString()
                    }
                }
            }
```

```
        override fun connectionLost(cause: Throwable?) {
            Log.e(TAG,"connection lost")
        }

        override fun deliveryComplete(token: IMqttDeliveryToken?) {
            Log.d(TAG,"Successfully delivered")
        }
    })
  }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_height="match_parent"
  android:layout_width="match_parent">

  <LinearLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:theme="@style/Theme.AppCompat.Light.NoActionBar"
    tools:context=".MainActivity">

    <Toolbar
      android:id="@+id/appbar"
      android:background="@color/colorAccent"
      android:layout_width="match_parent"
      android:layout_height="wrap_content">
      <TextView
        android:text="Controller"
        android:gravity="center"
        android:textSize="16sp"
        android:textColor="@android:color/white"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
    </Toolbar>

    <yjkim.mjpegviewer.MjpegView
      android:layout_below="@id/appbar"
      android:id="@+id/video_view"
```

```
    android:layout_width="match_parent"
    android:layout_height="300dp"/>

  <LinearLayout
    android:layout_below="@id/video_view"
    android:layout_margin="8dp"
    android:gravity="center"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <LinearLayout
      android:layout_alignParentBottom="false"
      android:orientation="vertical"
      android:layout_width="200dp"
      android:layout_height="wrap_content">
      <ImageButton
        android:id="@+id/up_button"
        android:background="@drawable/ic_arrow"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_gravity="center"
        android:focusable="true"
        android:defaultFocusHighlightEnabled="true"
      />

      <LinearLayout
        android:gravity="center"
        android:layout_width="match_parent"
        android:layout_height="50dp">

        <ImageButton
          android:id="@+id/left_button"
          android:rotation="270"
          android:background="@drawable/ic_arrow"
          android:layout_width="50dp"
          android:layout_height="50dp"
          android:layout_marginRight="25dp"/>

        <ImageButton
          android:id="@+id/right_button"
          android:layout_marginLeft="25dp"
          android:rotation="90"
          android:background="@drawable/ic_arrow"
          android:layout_width="50dp"
          android:layout_height="50dp"/>
      </LinearLayout>
```

```
    <ImageButton
        android:id="@+id/down_button"
        android:rotation="180"
        android:background="@drawable/ic_arrow"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_gravity="center"/>
  </LinearLayout>

</LinearLayout>

<LinearLayout
  android:orientation="vertical"
  android:gravity="center"
  android:layout_width="match_parent"
  android:layout_height="match_parent">

  <LinearLayout
   android:orientation="horizontal"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content">

   <TextView
          android:text="Obstacle Detected Ahead :"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"/>

                                        <TextView

   android:id="@+id/obstacle_text"

   android:layout_width="wrap_content"

   android:layout_height="wrap_content"/>

                                    </LinearLayout>

                                <LinearLayout
                                        android:orientation="horizontal"

   android:layout_width="wrap_content"

   android:layout_height="wrap_content">

                                <TextView

   android:text="Temperature Calculated :"
```

```
        android:layout_width="wrap_content"

        android:layout_height="wrap_content"/>


                                                <TextView

        android:id="@+id/temp_text"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"/>

                                            </LinearLayout>

                                        <LinearLayout
                                                android:orientation="horizontal"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content">

                                        <TextView
          android:text="Humidity Calculated :"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"/>

                                            <TextView

        android:id="@+id/humidity_text"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"/>
                                        </LinearLayout>
      </LinearLayout>

    </LinearLayout>
    <ProgressBar
      style="?android:attr/progressBarStyle"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:id="@+id/progressBar" android:layout_gravity="center"/>
</FrameLayout>
```

Conclusion

On Completing this project I was completely in awe about how can IOT change our working In day to day life and how can it benefit to this human race at a whole. It has applications in almost every field and it can increase by decreasing human intervention in unnecessary things and allowing humans to control their devices remotely can benefit at a whole new level.

IoT encourages the communication between devices, also famously known as Machine-to-Machine (M2M) communication. Because of this, the physical devices are able to stay connected and hence the total transparency is available with lesser inefficiencies and greater quality.

Due to physical objects getting connected and controlled digitally and centrally with wireless infrastructure, there is a large amount of automation and control in the workings. Without human intervention, the machines are able to communicate with each other leading to faster and timely output.

It is obvious that having more information helps making better decisions. Whether it is mundane decisions as needing to know what to buy at the grocery store or if your company has enough widgets and supplies, knowledge is power and more knowledge is better.

The second most obvious advantage of IoT is monitoring. Knowing the exact quantity of supplies or the air quality in your home, can further provide more information that could not have previously been collected easily. For instance, knowing that you are low on milk or printer ink could save you another trip to the store in the near future. Furthermore, monitoring the expiration of products can and will improve safety.

The biggest advantage of IoT is saving money. If the price of the tagging and monitoring equipment is less than the amount of money saved, then the Internet of Things will be very widely adopted. IoT fundamentally proves to be very helpful to people in their daily routines by making the appliances communicate to each other in an effective manner thereby saving and conserving energy and cost. Allowing the data to be communicated and shared between devices and then translating it into our required way, it makes our systems efficient.

Necessary Glossary

The internet of things, or IoT, is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.

A thing in the internet of things can be a person with a heart monitor implant, a farm animal with a biochip transponder, an automobile that has built-in sensors to alert the driver when tire pressure is low or any other natural or man-made object that can be assigned an IP address and is able to transfer data over a network.

Increasingly, organizations in a variety of industries are using IoT to operate more efficiently, better understand customers to deliver enhanced customer service, improve decision-making and increase the value of the business.

History of IoT

Kevin Ashton, co-founder of the Auto-ID Center at MIT, first mentioned the internet of things in a presentation he made to Procter & Gamble (P&G) in 1999. Wanting to bring radio frequency ID (RFID) to the attention of P&G's senior management, Ashton called his presentation "Internet of Things" to incorporate the cool new trend of 1999: the internet. MIT professor Neil Gershenfeld's book, When Things Start to Think, also appearing in 1999, didn't use the exact term but provided a clear vision of where IoT was headed.

IoT has evolved from the convergence of wireless technologies, microelectromechanical systems (MEMS), microservices and the internet. The convergence has helped tear down the silos between operational technology (OT) and information technology (IT), enabling unstructured machine-generated data to be analyzed for insights to drive improvements.

Although Ashton's was the first mention of the internet of things, the idea of connected devices has been around since the 1970s, under the monikers embedded internet and pervasive computing.

The first internet appliance, for example, was a Coke machine at Carnegie Mellon University in the early 1980s. Using the web, programmers could check the status of the machine and determine whether there would be a cold drink awaiting them, should they decide to make the trip to the machine.

IoT evolved from machine-to-machine (M2M) communication, i.e., machines connecting to each other via a network without human interaction. M2M refers to connecting a device to the cloud, managing it and collecting data.

Taking M2M to the next level, IoT is a sensor network of billions of smart devices that connect people, systems and other applications to collect and share data. As its foundation, M2M offers the connectivity that enables IoT.

The internet of things is also a natural extension of SCADA (supervisory control and data acquisition), a category of software application program for process control, the gathering of data in real time from remote locations to control equipment and conditions. SCADA systems include hardware and software components. The hardware gathers and feeds data into a computer that has SCADA software installed, where it is then processed and presented it in a timely manner. The evolution of SCADA is such that late-generation SCADA systems developed into first-generation IoT systems.

The concept of the IoT ecosystem, however, didn't really come into its own until the middle of 2010 when, in part, the government of China said it would make IoT a strategic priority in its five-year plan.

How IoT works

An IoT ecosystem consists of web-enabled smart devices that use embedded processors, sensors and communication hardware to collect, send and act on data they acquire from their environments. IoT devices share the sensor data they collect by connecting to an IoT gateway or other edge device where data is either sent to the cloud to be analyzed or analyzed locally. Sometimes, these devices communicate with other related devices and act on the information they get from one another. The devices do most of the work without human intervention, although people can interact with the devices -- for instance, to set them up, give them instructions or access the data.

The connectivity, networking and communication protocols used with these web-enabled devices largely depend on the specific IoT applications deployed .

IoT system example

Benefits of IoT

The internet of things offers a number of benefits to organizations, enabling them to: monitor their overall business processes; improve the customer experience; save time and money; enhance employee productivity; integrate and adapt business models; make better business decisions; and generate more revenue. IoT encourages companies to rethink the ways they approach their businesses, industries and markets and gives them the tools to improve their business strategies.

Consumer and enterprise IoT applications

There are numerous real-world applications of the internet of things, ranging from consumer IoT and enterprise IoT to manufacturing and industrial IoT (IIoT). IoT applications span numerous verticals, including automotive, telco, energy and more.

In the consumer segment, for example, smart homes that are equipped with smart thermostats, smart appliances and connected heating, lighting and electronic devices can be controlled remotely via computers, smartphones or other mobile devices.

Wearable devices with sensors and software can collect and analyze user data, sending messages to other technologies about the users with the aim of making users' lives easier and more comfortable. Wearable devices are also used for public safety -- for example, improving first responders' response times during emergencies by providing optimized routes to a location or by tracking construction workers' or firefighters' vital signs at life-threatening sites.

In healthcare, IoT offers many benefits, including the ability to monitor patients more closely to use the data that's generated and analyze it. Hospitals often use IoT systems to complete tasks such as inventory management, for both pharmaceuticals and medical instruments.

Future Enhancements

I haven't thought much about how can I make it more technologically advance but I know that this is not a best engineered work and it can be created in a better way and to do that I have some designing and crafting ideas like.

1. Better chassis using Fibre
2. Better power supply by using renewable energy sources
3. Adding cellular network in it. So that it will be able to send directly to cloud where it can be controlled via some remote channels
4. Building a better wheel so that it can work in some different terrains.

References

This project was never possible without help from people throughout my friend circle as well as college professors. They helped in every possible way and if this project is completed then many people have contributed for its success that's a fact but still It has intellectual properties taken from these resources.

1. Android Developers (developers.android.com)
2. Eclipse Foundation for paho Mqtt
3. Raspberry Pi Foundation
4. MJRoBot repository of MJRovai which has his implementation of rover documented.
5. LinuxWiki
6. AskUbuntu

Like Mr MJRovai I have made a github repo of code that is required for this project to work and for further reading about this project

The repository can be found at - https://github.com/gat786/controller