IoT Based Patrol Bot

Project Report submitted in the partial fulfilment

Of

Bachelor of Technology

In

Mechatronics Engineering

By

Adnan Amir H006

Aaryaman Chandgothia H012

Moksh Goel H021

Under the supervision of

Dr. Venkatesh Deshmukh

(HOD, MECHATRONICS, MPSTME)

Prof. Nirmal Thakur

(Asst. Prof. MECHATRONICS, MPSTME)

MUKESH PATEL SCHOOL OF TECHNOLOGY MANAGEMENT & ENGINEERING

Vile Parle (W), Mumbai-56

2022-23

<u>CERTIFICATE</u>

This is to certify that the project entitled **Patrol Bot**, has been done by **Adnan Amir H006 , Aaryaman Chandgothia H012, Moksh Goel H021** under my guidance and supervision & has been submitted in partial fulfilment of degree of Bachelor of Technology in Mechatronics of MPSTME, SVKM's NMIMS (Deemed-to-be University), Mumbai, India.

Name of the Guide

(Project Guide)

Examiner

Date

Place: Mumbai

Dr. Venkatesh Deshmukh

(HOD Mechatronics)

Dr. Alka Mahajan

Dean, MPSTME, SVKM's NMIMS

(Deemed-to-be University)

ACKNOWLEDGEMENT

For this Project, I am immensely thankful for the chance to work on this IoT Patrol Bot. This project came from a real life problem statement and working to challenge this problem is fulfillment on its own. Learning about the programming of processors, Hardware Implementation, working with IoT thingspeak and implementing all of that into one product was really helpful to both our skill set and to us personally.

Without the ongoing assistance of our mentor Dr. Venkatesh Deshmukh and Prof. Nirmal Thakur, who were there for us at every stage of the process to assist us with problem-solving or to generate creative ideas that would advance the research, We would not have been able to complete this project.

We would also like to thank NMIMS's MPSTME for providing us with all the resources and facilities that have made this project a success. We could not have done this without the organizations support and supervision.

Finally, we would like to thank all our fellow colleagues who helped us reach so far in our final project and helped us with troubleshooting, generating ideas or looking at problems from a different angle .

Adnan Amir H006,
B.Tech (MTRX)

Aaryaman Chandgothia H012,
B.Tech (MTRX)

Moksh Goel H021
B.Tech (MTRX)

ABSTRACT

In many regions of the world, security is a serious worry. The rise in thefts, from small thefts to full-fledged heists, has been reason for alarm. Many security measures have been put in place to counteract these issues. There are several degrees of security systems in the globe, ranging from alarm systems to CCTV cameras and human patrols. A CCTV camera system is the most prevalent of them. Recent advances in IoT have enabled remote access to these systems, but they have also significantly increased the cost.

The most basic and widely used CCTV camera systems include four cameras per room that stream a live feed to a PC that is monitored by a human guard. These are found in the majority of small and medium-sized businesses. Despite their low cost, many security measures fail to account for human mistake. On the night shift, the guard watching CCTV in underpaid workplaces and small stores frequently falls asleep. Despite having a security system in place, this leads to burglaries.

In this article, we cover the design and implementation of an auxiliary security system that works in conjunction with the usual CCTV configuration seen in most workplaces. Depending on the number of walls with windows that may be switched on at night when the workplace is unoccupied, the system comprises of 1-4 units of rover type robots. These Wi-Fi-enabled robots will patrol the room at night and use IoT principles to send an alert if an intruder is detected.

Table of Contents

List of the figures

List of Tables

Chapter 1

Introduction

Background of the project topic

This project was chosen to tackle the common problem of theft and burglary. We chose to look at this problem from a mechatronics standpoint and after analyzing the situation decided to build a rover that would tackle the aforementioned problem. A Patrol Bot would essentially monitor the entire room after hours and alert the required people if it senses a human presence.

Motivation and scope of the report

One anecdote of such a situation which also played a crucial role in the development of this system involves a robbery experienced by a relative of one of the authors. This relative who is the owner of a small retail store had valuable stock stolen despite having a CCTV Surveillance camera. The guard who was in charge of monitoring the camera feed had fallen asleep which led to this robbery.

Salient contribution

- The Patrol Bot designed is made to be small and stealthy , such that it attracts minuscule attention from intruders
- The Patrol Bot uses object detection through open CV to detect only humans
- Running Object detection and Navigation on different processors for smoother operation
- Adjustable Pi cam Angle to Capture all angles
- Night Vision No-IR Camera to ensure full vision even in the darkest rooms
- It employs IoT system to send alerts on different platforms so has a level of redundancy
- Random Exploration Algorithm for navigation system

Chapter 2

Literature survey

To prepare for the undertaking of this project, certain steps were followed to ascertain the proper flow of this project from the design phase to the implementation phase. One of the most important steps that were taken was the literature survey in which numerous papers were studied and listed below are our understandings of each of them

From [1], It helped us gain an insight into usage and programming the NodeMCU effectively to be used in an alert system which helped set up our base for the IOT domain of our project. Although we did not use the NodeMCU in the final project, this paper gave us insights into working with an ESP8266 module, which we did use.

[2] Talks About Various Parts required to make a line follower, its Sensor configuration and also Provides Good Code Structure. this paper gave as un understanding into positioning the sensors to reduce effect of ambient light and about the various states of motor while providing us with a good base to build our code on

[3] Helped us to map out our Integration of multiple sensors a and understand how to Use Wi-Fi to send data over to a aggregation point. It was also hugely beneficial for our design stage as it provides a Flowchart for the alarm system. Which was instrumental in understanding the working of the entire alert system

[4] suggested a new and efficient system instead of complex algorithms for motion detection . This paper also provided us with an innovative way for the movement algorithm for the patrol robot which gave us different options to program our patrol bot in case of any errors. This was the paper which initially gave us the idea of sending data over Wi-Fi to a smart phone and how to implement it successfully using Arduino and raspberry pi.

[5] Helped us get a deeper understanding of ultrasonic sensors and as to why they are the backbone of this project. This entire system uses the technique of echolocation which is based on ultrasonic wave radar, RF communication & Infra- Red technology which is separately controlled by different Arduino boards which controls the different sensors, motors & line following robot according to the desire. We took inspiration from this idea and implemented it by using an Arduino mega instead of a normal Arduino.

This robot works on the principle of Infra-Red Line Tracking. Which can be white or black line based on the programming. This paper showed us the different applications depending upon the sensors such as: Accurate Distance Measurement, Close Range Device, Level Detection, Vehicle Detection, Blind Person Support and Robotics Barrier

[6] Talks about viability of Thingspeak as an IOT server for small scale projects. And the t process of IOT integration over Wi-Fi . it also introduced us to the MTTQ which we implemented successfully. This paper helped us set our entire operation for the IOT protocol as it provided us with a Description on How to use Thingspeak and also the use of MQTT protocol to route to an app which we have implemented successfully. It further helped us in our implementation as it provided entire flowchart for sending data collected from sensor to getting alert on phone.

[7] And [8] talks about the motion detection technique to reduce noise creation. Interior space mapping, environment modelling, virtual reality applications, military settings, the mining industry, and graphic applications are all possible uses for it. It can be used to map an area with a diameter of 40 meters when mapping outside. Using the moving average filter, the mapping procedure was carried out as close to the real data as possible. These two papers give us the much needed post processing operations to map ultrasonic data to an occupancy grid.

[9] This paper established a method to supress vibrations by motors. While the motor used in the paper is an AC motor, the same reasoning also applies to a DC motor. This paper suggests using a capacitor to smoothen the vibrations thus reducing noise. We will experiment with the findings in this paper

[11] Presents two important things, firstly, it gives us insight on how to map distance error to rotation correction paving the way to derive the transfer function. Secondly, it tells us about an optimization method for the PID controller known as the genetic algorithm. It will help us get a better response from our controller

[12] Gives us insight into how to position the grideye sensor and narrows down the cases to consider while coding for different conditions in the grideye sensor. It seems like it is difficult to determine the number of intruders with this sensor but the presence of one can be detected. This paper led us to use AMG8833 sensor to implement thermal vision over a melexis IR array.

[13] presents an AC system which broadly does 3 things: Human detection, Body Temperature Isolation and AC control. We are concerned with the Human detection module. The paper presents a python script which gives us tips on how to implement human detection for this project.

Chapter 3

Problem Definition

Robotics is a discipline that is expanding very quickly and has clear effects on day-to-day living. Robotics' success is largely due to the fact that it eliminates the possibility of human mistakes in the system.

One anecdote of such a situation which also played a crucial role in the development of this system involves a robbery experienced by a relative of one of the authors. This relative who is the owner of a small retail store had valuable stock stolen despite having a CCTV Surveillance camera. The guard who was in charge of monitoring the camera feed had fallen asleep which led to this robbery.

Human mistakes have adverse effects on many systems. Nuclear energy systems, medical automation systems, and security systems are just a few examples.
Despite all the benefits of automating a system, there are numerous faults that cannot be accounted for, making a fully autonomous system extremely risky.

In light of the aforementioned considerations, The Patrol Bot is intended to function as a secondary security system and primarily as an alarm system that is used in combination with a primary surveillance system like CCTV. With moving robots that can patrol the designated area and offer continual surveillance, this security system is active as opposed to passive.

The bot also tackles the issue of unreliable electricity which often disables standard CCTV cameras. Though there exist cameras with battery backup, these do not long enough because they are also active during the day. This bot aims to alleviate the issue by only operating at night when robberies are most likely to happen.

Many systems exist to solve these problems but they are not accessible to common retailers. By keeping the cost near the cost of a standard CCTV setup, this addendum to security will definitely add a level of autonomy which is accessible and reliable.

Chapter 4

Methodology

This Patrol Bot was designed with a mechatronic approach in mind. Each system was designed concurrently and designed to integrate well with each system to prevent designing errors. Electrical and design aspects of the project were designed parallel and simultaneously and they closely depend on each other. Changing aspects of one system to compromise for the other system helped us develop an all rounded Robot that can perform efficiently on all systems

The methodology we followed for completing this project consisted of 4 main steps:

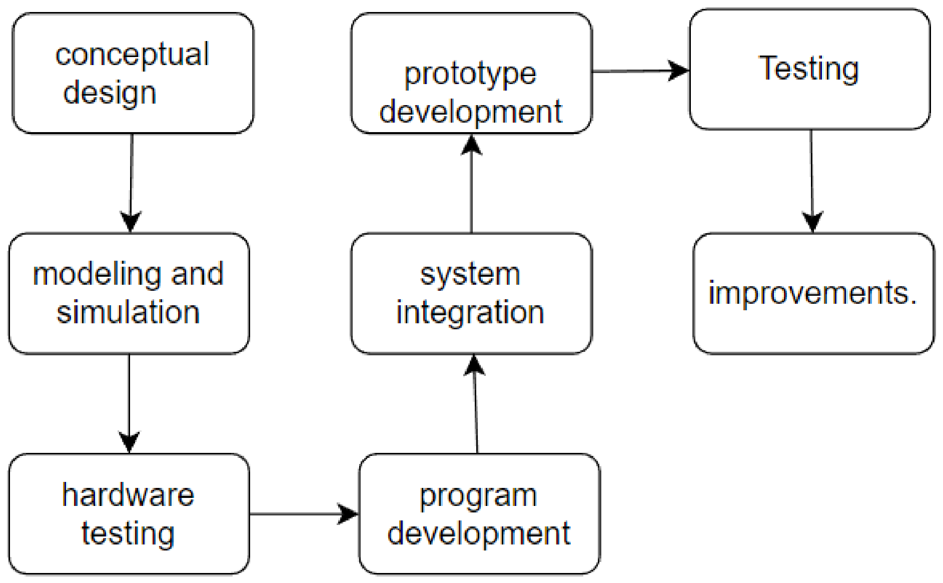


Fig 1. Methodology For Project

We first identified a problem statement and understood the outlying conditions and circumstances of it. We understood the various requirements needed to challenge this problem and then decided to approach it keeping an open mind to solutions and looking to tackle it with a mechatronics system. We performed extensive research and literature review to understand the scope of our project and challenges going forward as well as solutions to it. One adequate research was performed, we performed various tests to see the viability and robustness of our systems as well as understand their weaknesses. The final step was for us to implement the culmination of all of our efforts into a physical model.

Chapter 5

Mechatronic System Analysis

System Description:

The system consists of 1 bot per room monitoring the room and uploading alert data to a cloud IoT Platform known as ThingSpeak. The overall system can be broadly classified into three categories; the mechanical system, the electronic system and The IoT system.

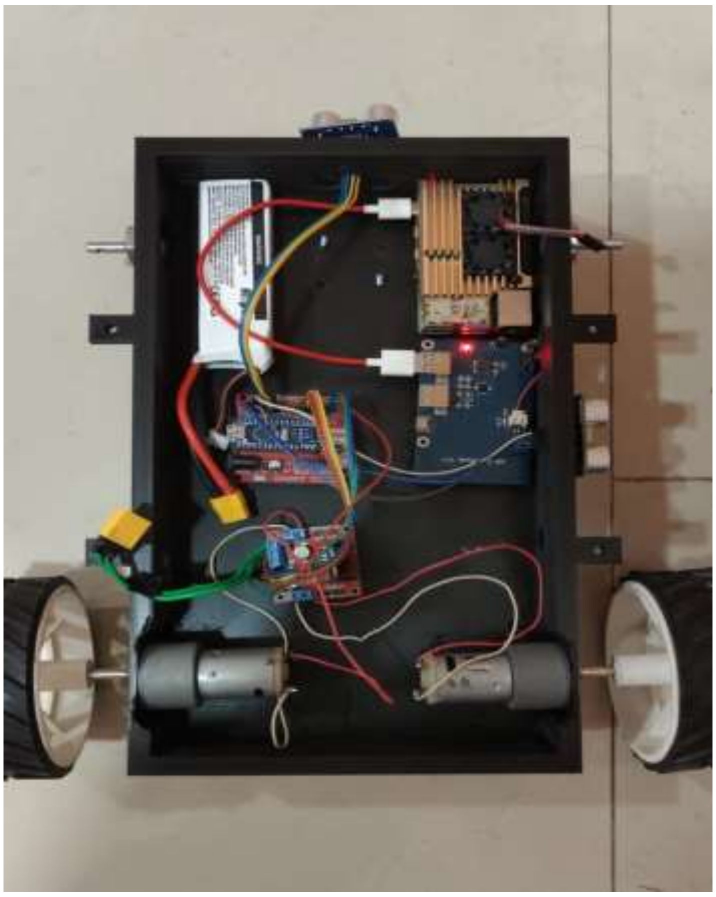


Fig 2. Final electronics

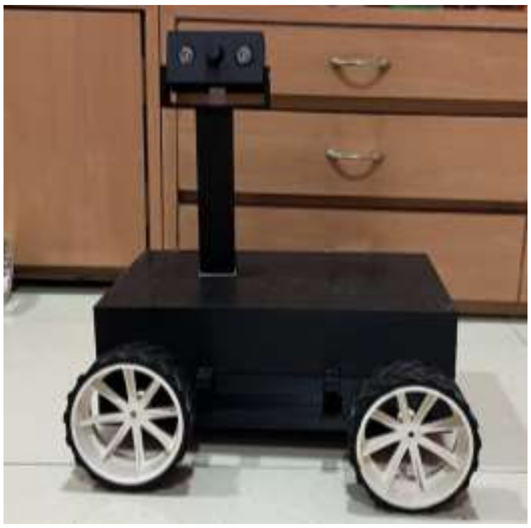


Fig 3. Final Hardware Model

Mechanical system:

The chassis of the bot is to be made 3D printed. This will be robust, clean, cost-effective and easily replaceable. It has also been printed using black filament to blend in with the darkness at night.

The model will house a motor driver, two microprocessor boards, a camera mount, battery pack and Ultrasonic sensors .Four plastic wheels have been attached to the motors mounted on the model as they are easily available and very cheap. The rims of the wheel have also been made black to make the bot inconspicuous.

There are slots made to charge the Battery Part and easily, there are also 2 slots directly available to the microprocessor board so it becomes easy to access and reprogram the controller with proper IoT Credentials and Wi-Fi SSID and password. The motor driver is put on top so that the heatsink can dissipate the heat effectively. All the components are housed inside to make it stealthier and less tamperable.

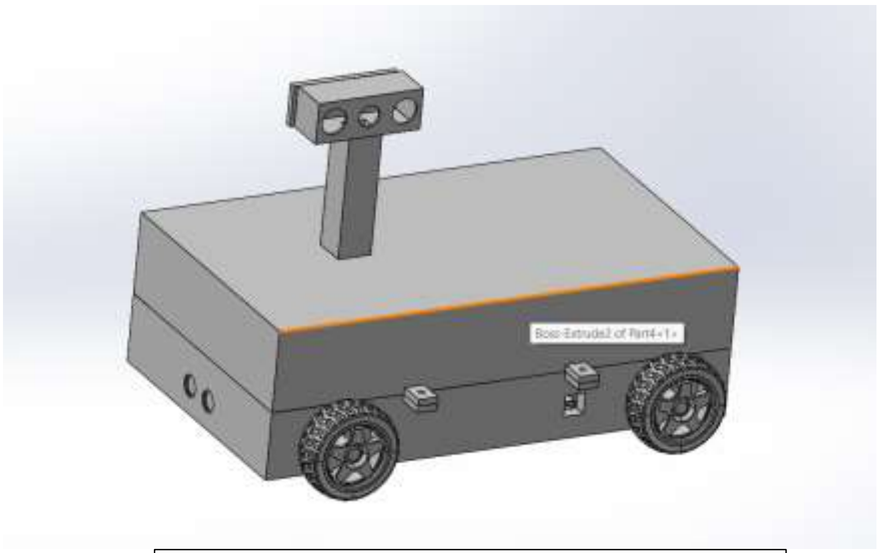


Fig 4. Solidworks Model

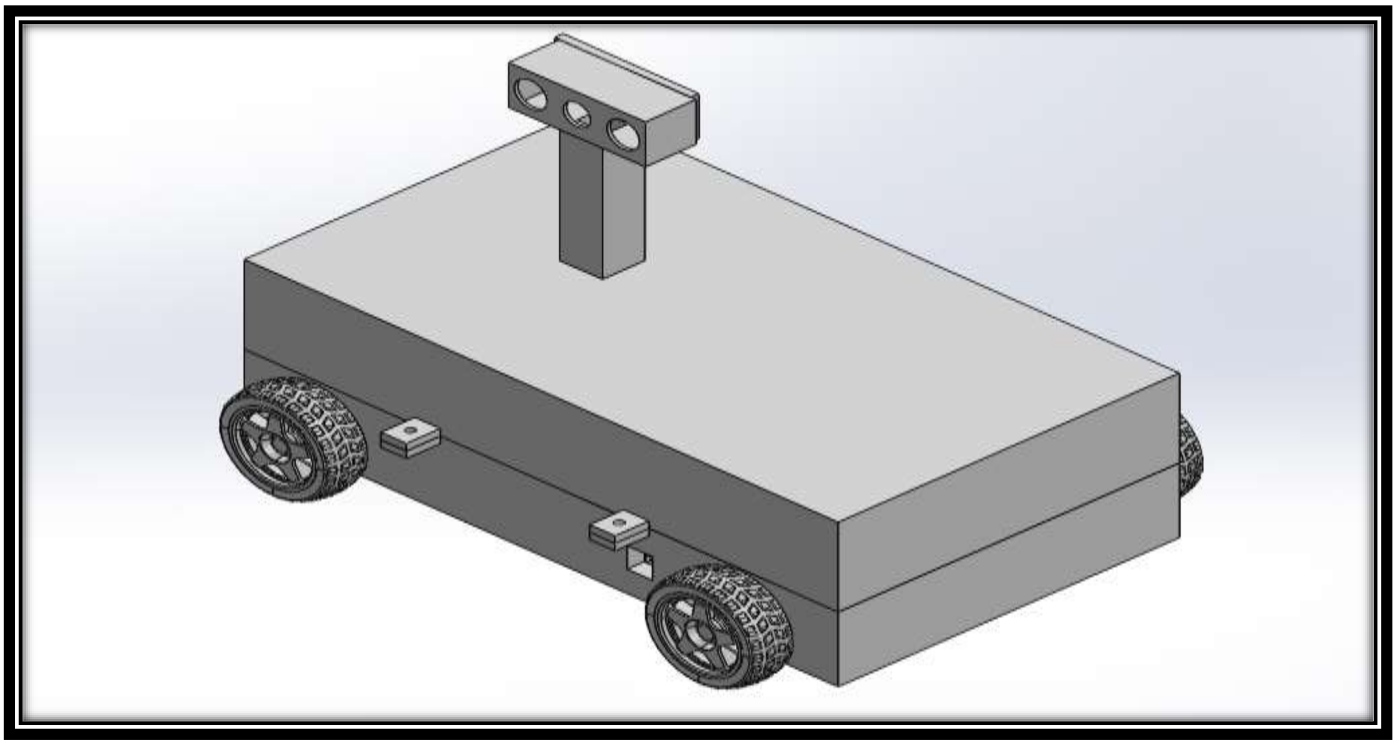


Fig 5. Solidworks Model isometric

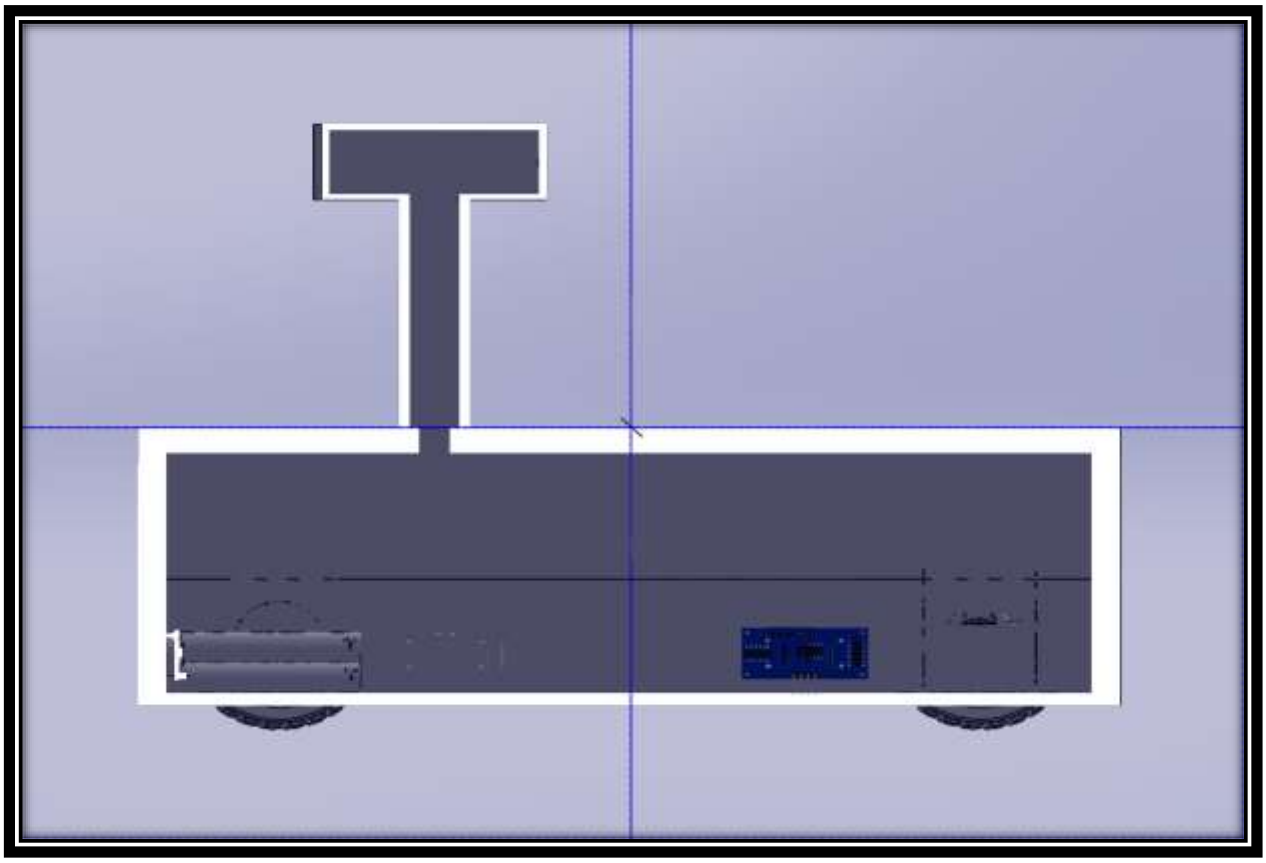


Fig 6. Solidworks Model Section View

Electronic System:

The main processor used by our project is The Raspberry Pi 4 Model B. It is the latest version of the low-cost Raspberry Pi computer made by the Raspberry Pi Foundation, a UK charity that aims to educate people in computing and create easier access to computing education.

In comparison to the Raspberry Pi 3 Model B+, the quad-core Raspberry Pi 4 Model B is both quicker and more powerful. The Pi 4's CPU, the board's primary processor, offers two to three times the performance of the Pi 3's processor in various evaluations.

The new board, unlike its predecessor, can play 4K video at 60 frames per second, enhancing the Pi's capabilities as a media centre. The Pi's various operating systems are currently working to support this hardware acceleration for H.265-encoded video, so this is more of a potential feature than a feature that is currently available. That said, not all video will play this smoothly.

With integrated Wi-Fi and Bluetooth, the Pi 4 offers wireless internet right out of the box as well. The most recent board can boot straight from a USB-attached hard drive or pen drive and will allow PXE booting from a network-attached file system after a future firmware upgrade. A network-attached disc may be used to share an OS image between computers and to update a Raspberry Pi from a distance.

We further use an Arduino board to run the navigation algorithm as running it on separate board’s increases the accuracy and response rate. We therefore run the object detection algorithm on Raspberry Pi module and Navigation on an Arduino Board. We can use a UNO or a Nano board (combined with an expansion board) for the same purpose.

The other components of the system include an L298N dual motor driver. We are only using one because it is a 2WD system and the front wheels are there for support. This keeps it cost effective, low power and provides efficiency in navigation (because there are less motors to control)

There are two ultrasonic sensors, one in the front for detecting when to turn and one on the side to get feedback for PID based wall following algorithm. These two effectively act as the cornerstone of the navigation system.

The battery will be a rechargeable Lithium Polymer which provides us with 11.1 V with close to 2200 mAh which is perfect for driving our Johnson motors. It is also better as it is densely packed so it can fit in the chassis

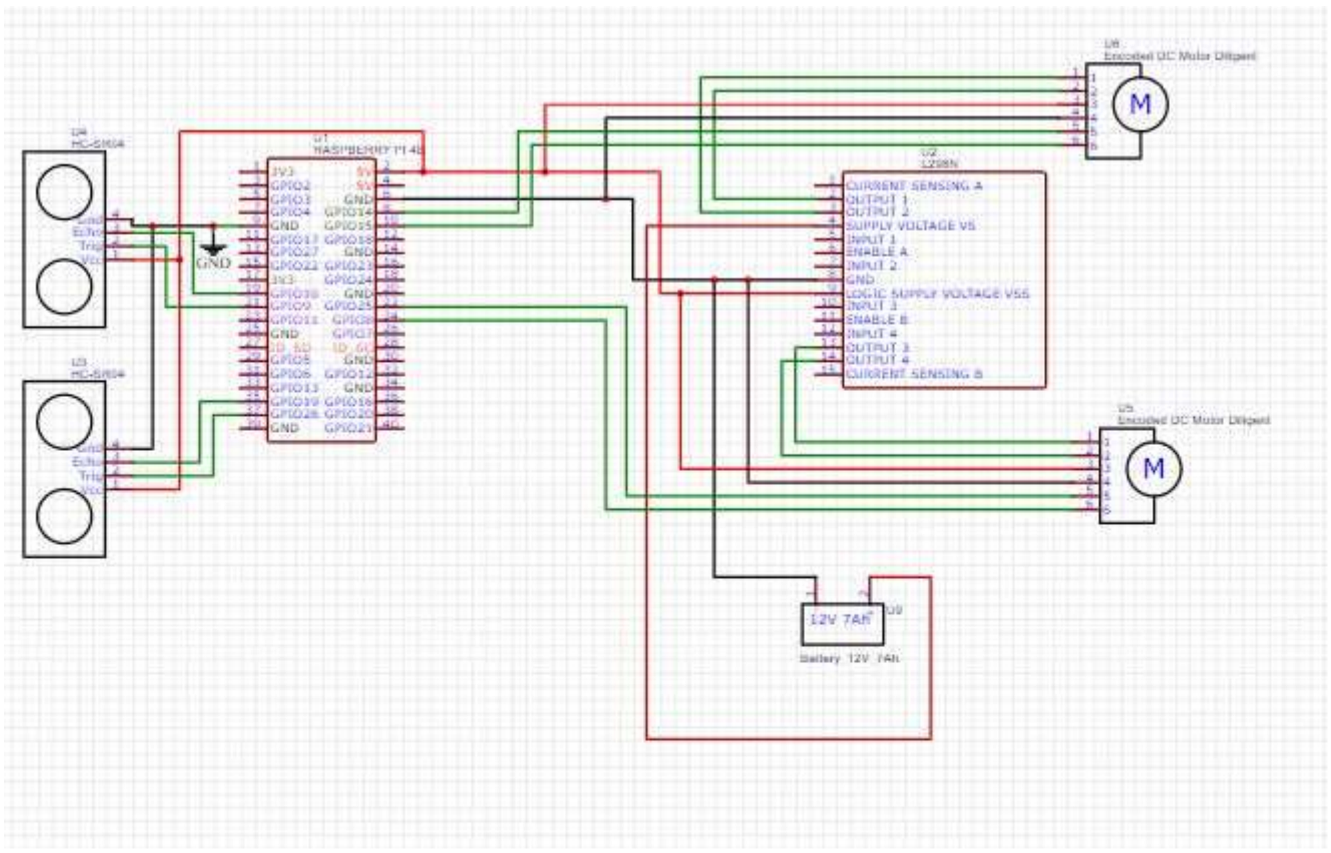


Fig 7. Circuit diagram

Detection system

Object detection consists of 3 important factors

1. The detection algorithm
2. The CNN
3. The Dataset

1) The Detection algorithm

We considered 2 algorithms that don’t use neural networks (HOG descriptor and Cassacade Classifier) and 2 that do use neural networks (Region-based Convolutional Neural Network (RCNN) and Single Shot MultiBox Detector (SSD)) as these are lightweight and reliable.We also considered using You only look once (YOLO) but it was too resource intensive with requirement of GPU. There is a way to run it on CPU as well but it did not perform well in terms of speed.

a) SSD (Single Shot MultiBox Detector) is a deep learning-based object detection algorithm that can detect objects in images in real-time. The algorithm uses a single feedforward convolutional neural network to generate object detection results. Here are the key steps in how the SSD algorithm works:

   a. Generating Feature Maps: The input image is passed through a series of convolutional layers to generate a set of feature maps at different scales. These feature maps encode information about the presence of objects at different positions and scales in the image.

   b. Generating Object Proposals: For each feature map, a set of default boxes (or anchor boxes) is defined at different scales and aspect ratios. Each default box is associated with a set of predicted offsets that are used to adjust the size and position of the box to better fit the objects in the image.

   c. Predicting Class Scores and Offsets: The feature maps are used to predict two sets of outputs for each default box: the class scores and the offsets. The class scores indicate the probability that the box contains an object of a particular class, while the offsets adjust the position and size of the box to more accurately fit the object.

d. Non-Maximum Suppression: The predicted boxes are filtered to remove duplicates and weak detections using non-maximum suppression (NMS). NMS removes overlapping boxes with low confidence scores and keeps only the highest-scoring boxes.

e. Output: The remaining boxes and their associated class scores are output as the final detection results.

The SSD algorithm is designed to detect objects at multiple scales and aspect ratios, which allows it to handle objects of different sizes and shapes. It is also efficient, as it generates object proposals and predicts class scores and offsets in a single pass through the neural network, making it much faster than other object detection algorithms that use multiple stages or region proposal networks.

b) Faster R-CNN (Region-based Convolutional Neural Network) is a deep learning-based object detection algorithm that uses a two-stage approach to detect objects in images. The algorithm is an improved version of the earlier R-CNN and Fast R-CNN algorithms and is capable of achieving state-of-the-art performance on a wide range of object detection tasks. Here are the key steps in how the Faster R-CNN algorithm works:

a. Generating Region Proposals: The input image is passed through a convolutional neural network (CNN) to generate a feature map. A region proposal network (RPN) is then applied to the feature map to generate a set of object proposals. The RPN is a small neural network that learns to predict a set of candidate object bounding boxes, along with their confidence scores, based on the features extracted from the input image.

b. Extracting Region Features: The proposed regions are extracted from the feature map and warped to a fixed size. These regions are then passed through a convolutional neural network (CNN) to generate a set of features that are used to classify the object and refine the bounding box.

c. Predicting Class Scores and Bounding Boxes: The features extracted from each region proposal are used to predict the class scores and bounding box offsets for each proposed object. The class scores indicate the probability that the region contains an object of a particular class, while the bounding box offsets adjust the position and size of the box to better fit the object.

d. Non-Maximum Suppression: The predicted boxes are filtered to remove duplicates and weak detections using non-maximum suppression (NMS). NMS removes overlapping boxes with low confidence scores and keeps only the highest-scoring boxes.

e. Output: The remaining boxes and their associated class scores are output as the final detection results.

The Faster R-CNN algorithm uses the RPN to generate region proposals, which allows it to share convolutional features between the proposal generation and detection stages. This makes it much faster and more efficient than earlier region-based algorithms, while also improving detection accuracy.

c) HOG (Histogram of Oriented Gradients) descriptor is a feature extraction method used in object detection tasks. It works by computing a descriptor that captures the gradient information of the image. Here are the key steps in how the HOG descriptor works:

a. Image Preprocessing: The input image is preprocessed by converting it to grayscale, and then applying contrast normalization to equalize the image intensity.

b. Computing Gradients: The image gradients are computed by convolving the image with a set of filters (such as Sobel filters) in the x and y directions. This results in two gradient images representing the magnitude and orientation of the gradients at each pixel location.

c. Generating Cells: The image is divided into small overlapping cells, each containing a fixed number of pixels. The gradient magnitude and orientation values within each cell are then used to compute a histogram of gradient orientations.

d. Block Normalization: The histograms of gradient orientations within each cell are combined to form a block. Each block contains a fixed number of cells and is used to capture the spatial information of the gradient features. The gradient histograms within each block are then normalized to reduce the effects of illumination changes and improve the robustness of the descriptor.

e. Concatenation: The normalized block histograms are concatenated to form the final feature vector, which is used for object detection.

The HOG descriptor captures the local texture and edge information of the image, which can be used to detect objects of interest. The algorithm is popular for object detection tasks, particularly in pedestrian detection, due to its effectiveness and efficiency. However, it does not take into account the spatial relationships between features, which can limit its performance in complex object detection scenarios.

d) Cascade Classifier is a machine learning-based object detection algorithm that uses a series of classifiers trained on increasingly complex features to detect objects in images. The algorithm is based on the Haar-like features, which are simple rectangular patterns that can be computed efficiently. Here are the key steps in how the Cascade Classifier algorithm works:

   a. Image Preprocessing: The input image is preprocessed by converting it to grayscale, and then applying contrast normalization to equalize the image intensity.

   b. Haar-like Features: The algorithm uses Haar-like features, which are simple rectangular patterns that can be computed efficiently. These features are used to capture local texture and edge information of the image.

   c. Training: The algorithm is trained using a set of positive and negative examples. The positive examples are images containing the object of interest, while the negative examples are images that do not contain the object. The algorithm learns a set of classifiers, each trained on increasingly complex features.

   d. Sliding Window: The image is divided into a set of overlapping windows, each containing a fixed number of pixels. The Haar-like features are computed within each window.

   e. Classification: The Haar-like features within each window are used to classify the window as containing the object of interest or not. The classification is performed using a series of classifiers trained on increasingly complex features. The classifiers are organized into a cascade, where each stage consists of a series of weak classifiers that can quickly reject windows that do not contain the object.

   f. Non-Maximum Suppression: The predicted boxes are filtered to remove duplicates and weak detections using non-maximum suppression (NMS). NMS removes overlapping boxes with low confidence scores and keeps only the highest-scoring boxes.

g. Output: The remaining boxes and their associated class scores are output as the final detection results.

The Cascade Classifier algorithm is fast and efficient, as it can quickly reject windows that do not contain the object. However, it may require more training data and tuning of parameters compared to other object detection algorithms. It is commonly used for real-time object detection in applications such as face detection and pedestrian detection.

1) Neural Network

Neural networks for object detection typically involve a type of architecture called a convolutional neural network (CNN).

A CNN consists of multiple layers, including convolutional layers, pooling layers, and fully connected layers. In the context of object detection, the input to the network is typically an image, and the output is a set of bounding boxes that indicate the location of objects within the image.

During the forward pass of the network, the input image is processed through a series of convolutional and pooling layers, which extract features at different scales and levels of abstraction. These features are then fed into fully connected layers, which output the final set of bounding boxes.

To train the network, a large dataset of labeled images is used to adjust the weights of the network such that it learns to accurately detect objects in new, unseen images. The training process typically involves minimizing a loss function that measures the difference between the predicted bounding boxes and the ground truth bounding boxes.

Once the network has been trained, it can be used to detect objects in new images by passing them through the network and using the output bounding boxes to identify the location and extent of the objects in the image.

We explored the use of 3 most common neural networks: ResNet, MobileNet, and Inception. They are deep neural network architectures that are widely used for image recognition tasks.

a. ResNet: ResNet (Residual Network) is a deep neural network architecture that introduced the concept of residual learning. It uses residual blocks to allow the network to learn residual mappings, which can help address the problem of vanishing gradients in very deep networks. ResNet50 and ResNet101 are variants of the original ResNet architecture that differ in the number of layers and complexity.

b. MobileNet: MobileNet is a deep neural network architecture designed for mobile and embedded devices. It uses depthwise separable convolutions, which separate the spatial and channel-wise convolutions to reduce the number of computations and parameters. This makes MobileNet more computationally efficient than other deep neural network architectures.

c. Inception: Inception is a deep neural network architecture that uses a combination of 1x1, 3x3, and 5x5 convolutions in parallel to capture features at different scales. It also uses pooling and concatenation operations to further improve the performance. Inception V3 and Inception V4 are variants of the original Inception architecture that differ in the number of layers and complexity.

2) Dataset

We used the COCO dataset, but also tried the AVA dataset. The COCO (Common Objects in Context) and AVA (Aesthetic Visual Analysis) datasets are two widely used datasets in computer vision and image recognition research.

1. COCO: COCO is a large-scale image recognition, segmentation, and captioning dataset. It contains more than 330,000 images of everyday scenes with common objects in their context. The images are annotated with object bounding boxes, object categories, and object segmentation masks. COCO also includes a captioning task, where each image is annotated with a set of captions describing the content of the image.

2. AVA: AVA is a large-scale dataset for aesthetic visual analysis, which aims to predict the aesthetic quality of images. It contains more than 250,000 images that are selected from the DPChallenge photography competition. Each image is annotated with a set of aesthetic ratings, which reflect the quality of the image as rated by human annotators. The annotations include elements such as the overall quality, composition, focus, and lighting of the image.

Configuration Comparisons

Detection Algorithm	Neural Network	Dataset	FPS (On PC)	Accuracy (On a scale of 1-10)
HOG Descriptor	None	None	25	1
Cassacade Classifier	None	Haar Casacade Fullbody dataset	6	3
RCNN	Inception	COCO	2	8
RCNN	ResNet	COCO	0.8	10
RCNN	ResNet	AVA	0.8	9
SSD	Inception	COCO	9	8
SSD	Mobilenet	COCO	16	6
SSD	Mobilenet	AVA	17	5

Table 1. Configuration Comparisions

Navigation system

The original aim of the navigation system was to mobilize the camera and stay as inconspicuous as possible. Thus, building upon the previously designed line following navigation, we wanted to remove the need for the initial setup of the line while maintaining the idea of staying inconspicuous from the sidelines. We devised a PID based wall following algorithm to stick to the walls and maintain smooth motion. However, we very soon realised that it required a lot of computation and did not deliver the best speed, so we figured that out as its limitation. Besides, the camera did not provide a very clear image thus creating blind spots.

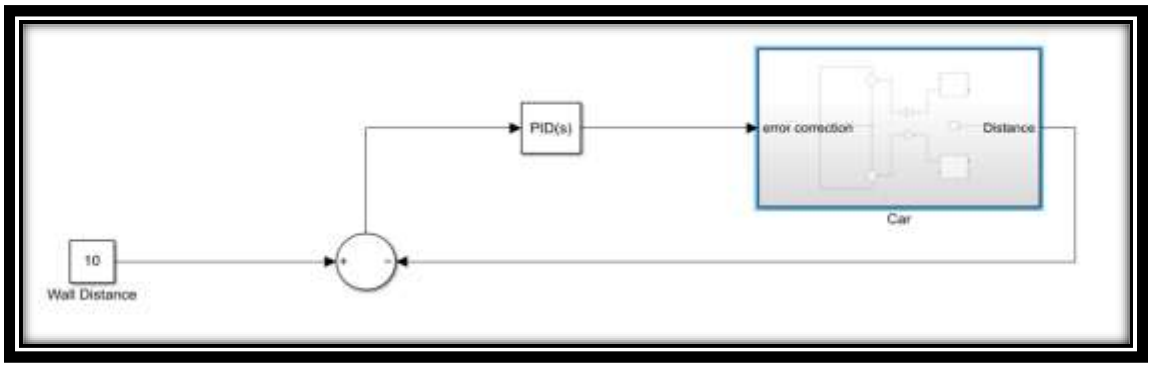
As the project progressed, we figured out that the more important thing about the navigation system is that it should cover as much ground as possible and be very responsive with low cost hardware. Thus, with our constraint of cost effectiveness barring us from high cost sensors like the LiDAR and high cost microcontrollers like the teensy 4.1, we decided that the random exploration algorithm is best suited to this project. The random exploration algorithm works as follows:

- The bot goes straight unless it detects a wall
- If there is a wall, it tends to turn right
- If however, the right sensor detects something, the bot turns left instead

This system minimizes the use of sensors while covering a lot of ground. The colour and build of the robot makes it inconspicuous. So, the navigation system only has to focus on giving the camera as many angles as possible thereby negating any blind spots. Moreover, the navigation system is completely isolated from the vision system, making it completely immune to software failures.

Overall, the system maintains a balance between speed, cost and effectiveness in patrolling.

The general block diagram of the original PID wall following system is shown below:



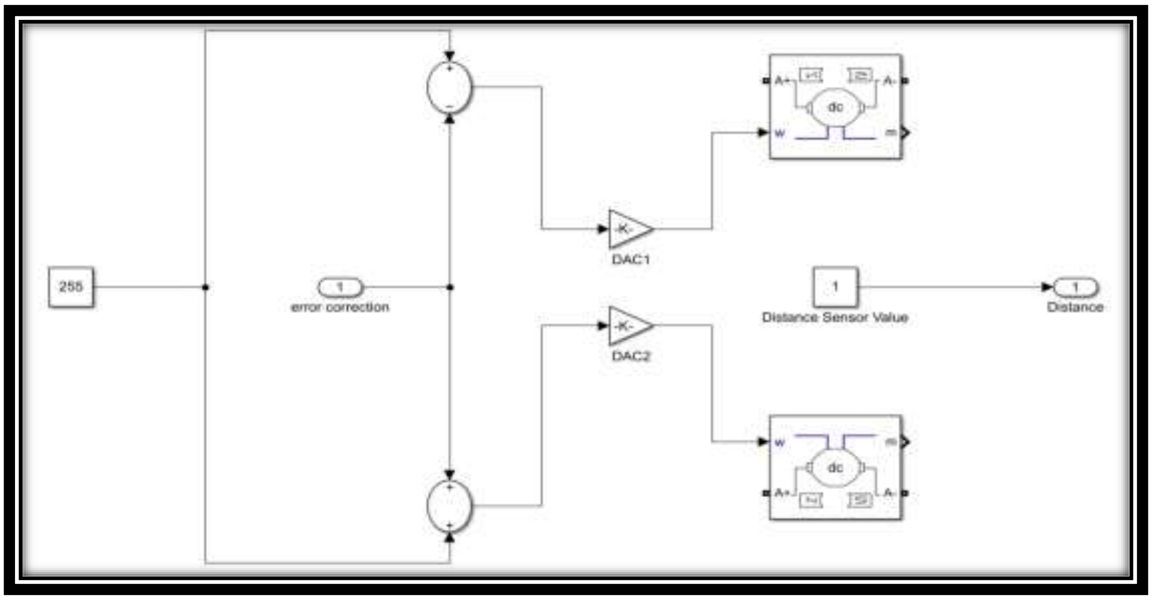


Fig 8. Wall following system block diagram

As observed above, the error in distance from the ultrasonic sensor is mapped to wheel RPM. This can be made better if we can derive the transfer function for this situation. Otherwise, experimental tuning the system can be done. An Alternate navigation method we are testing aims to solve one of the biggest problems in navigation today. Replacement of a Lidar by Ultrasonic sensor for mapping. If we can make this system reliable, it will allow us to test out many different navigation algorithms and even implement intruder avoidance (to add to the stealth aspect), all while keeping the bot around the same cost as CCTV camera system.

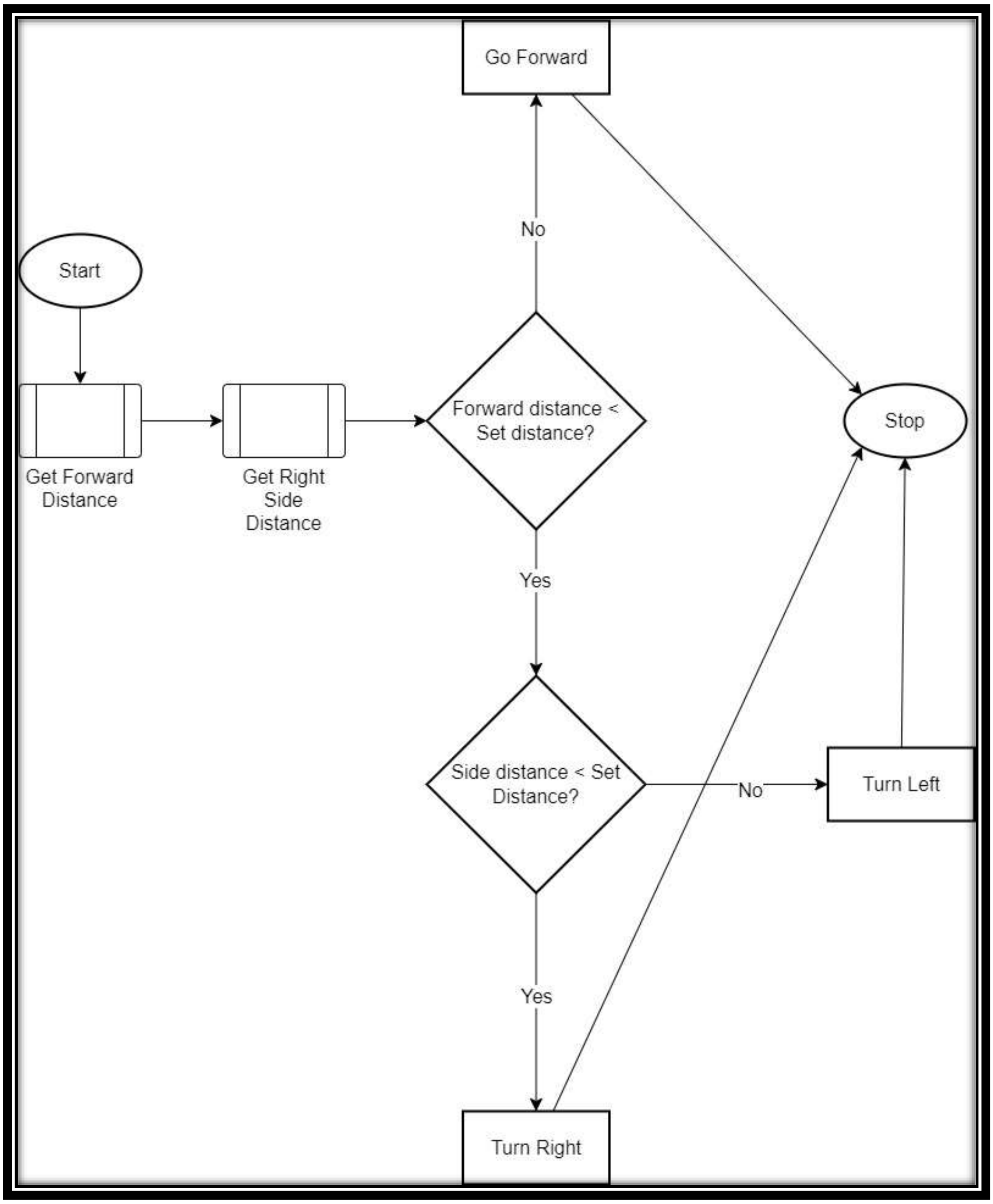


Fig 9. Navigation System

IoT System:

To implement the IoT system we have used ThingSpeak IoT Platform. ThingSpeak is a free and open-source software written in Ruby which allows users to communicate with internet-enabled devices. It facilitates data access, retrieval and logging of data by providing an API to both the devices and social network websites. ThingSpeak was originally launched by ioBridge in 2010 as a service in support of IoT applications.

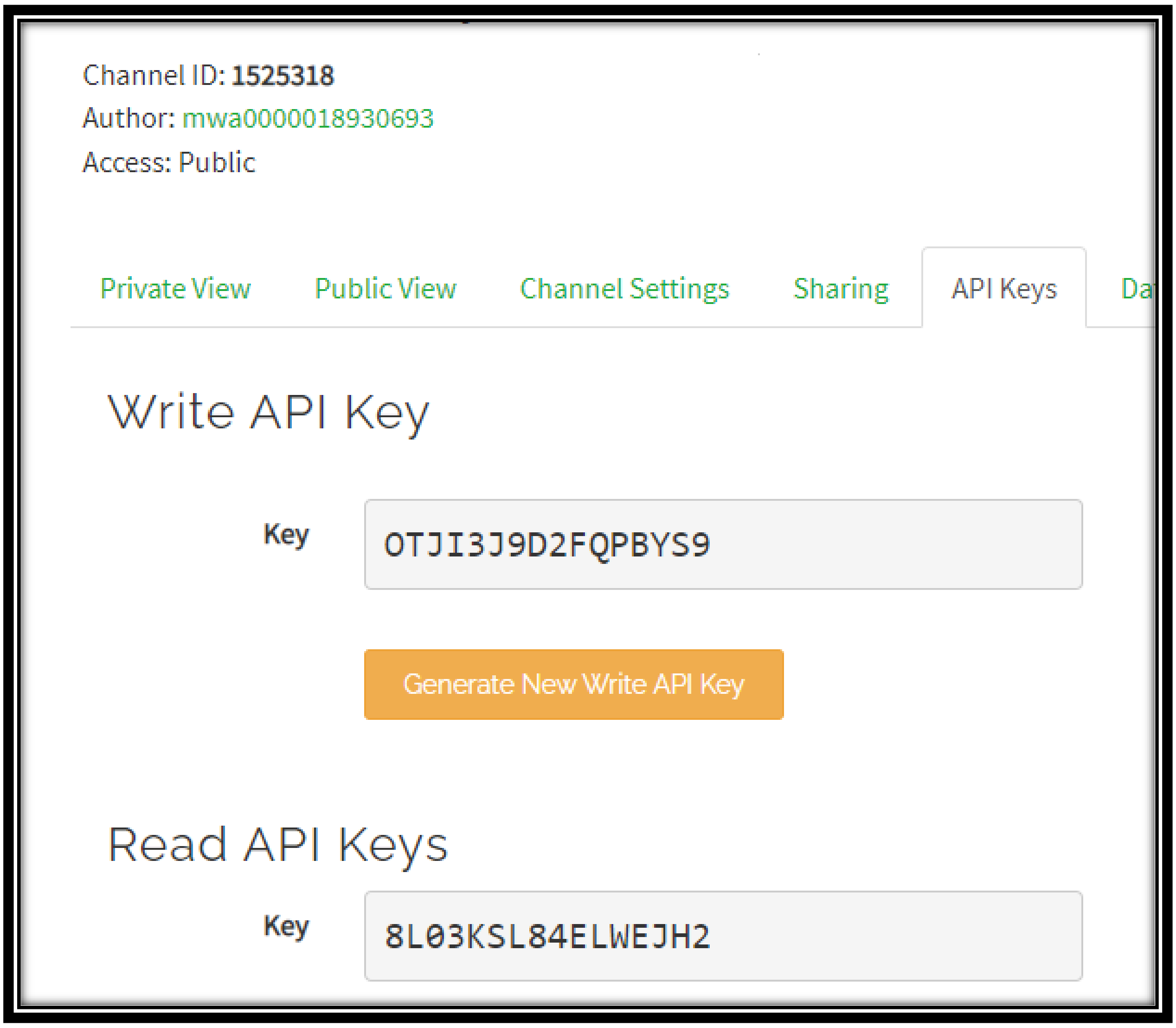
ThingSpeak has integrated support from the numerical computing software MATLAB from MathWorks, allowing ThingSpeak users to analyse and visualize uploaded data using MATLAB without requiring the purchase of a MATLAB license from MathWorks.

ThingSpeak has multiple apps which are used to set up the Alert System. The apps used in Thingspeak include React, Thingtweet, MATLAB Analysis and MQTT Communication. Each has been explained below

To Get Started with ThingSpeak, we first created a new public channel and added a field in that channel known as the EMW alert system. After the creation of the channel and field, we noted the write API key, since we would be writing a flag value to this channel from the SBC and we also noted the channel ID from the API key Section. The illustration of the same has been listed below. Although the channel ID is not used, it is required by the Thingspeak Arduino library for writing to a channel field.

Chapter 6

Software Description



Channel ID: **1525318**
Author: mwa0000018930693
Access: Public

Write API Key

Key OTJI3J9D2FQPBYS9

Read API Keys

Key 8L03KSL84ELWEJH2

Fig 10. Illustration of API keys

The flag value can be written to this channel by sending the following GET request to Port 80 (common internet port):

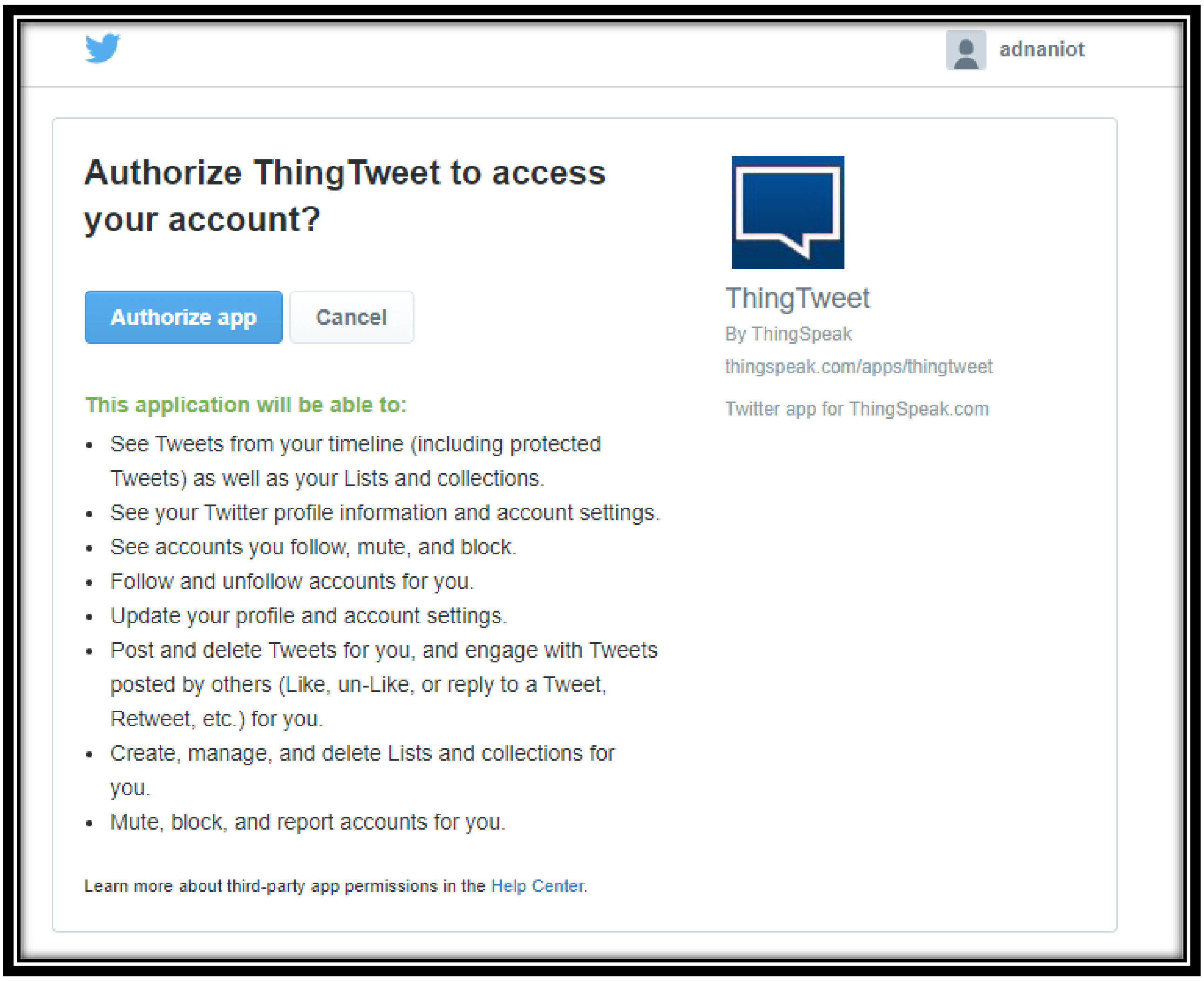
GET https://api.thingspeak.com/update?api_key="<write api key>"&field<number>=<value>

ThingSpeak allows sending of up to 3,000,000 messages per year on a free account in this way.

Once this is set up, we can send alerts using React App which sends a tweet via ThingTweet and email via MATLAB Analysis. We also use ThingSpeak MQTT to send an instantaneous desktop alert.

ThingTweet:

The ThingTweet app is used to link a Twitter account to the user's ThingSpeak account. The devices can then send alerts via Twitter using ThingTweet. Once we click on link account, we are redirected to a Twitter page where, upon logging in, we have to authorize ThingSpeak to send a Twitter alert. An illustration is provided below



adnaniot

Authorize ThingTweet to access your account?

Authorize app | Cancel

This application will be able to:

- See Tweets from your timeline (including protected Tweets) as well as your Lists and collections.
- See your Twitter profile information and account settings.
- See accounts you follow, mute, and block.
- Follow and unfollow accounts for you.
- Update your profile and account settings.
- Post and delete Tweets for you, and engage with Tweets posted by others (Like, un-Like, or reply to a Tweet, Retweet, etc.) for you.
- Create, manage, and delete Lists and collections for you.
- Mute, block, and report accounts for you.

Learn more about third-party app permissions in the Help Center.

ThingTweet
By ThingSpeak
thingspeak.com/apps/thingtweet

Twitter app for ThingSpeak.com

Fig 11. Thingtweet Authorization

Following this, an API key for twitter will be generated. This may be used in a POST request sent to port 80 as follows:

POST https://api.thingspeak.com/apps/thingtweet/1/statuses/update

api_key=<twitter API key>

status=<body of tweet>

However, this is done in this system via React App.

MATLAB Analysis:

This app allows the use of running MATLAB code and is used in conjunction with React app.

This requires Thingspeak Alert API Key which can be obtained from the main account page in ThingSpeak. It begins with “TAK”.

The following code was used to send an email to the google account used to register with ThingSpeak. The system employed uses a Java payload system to send the e-mail. Following code was used:

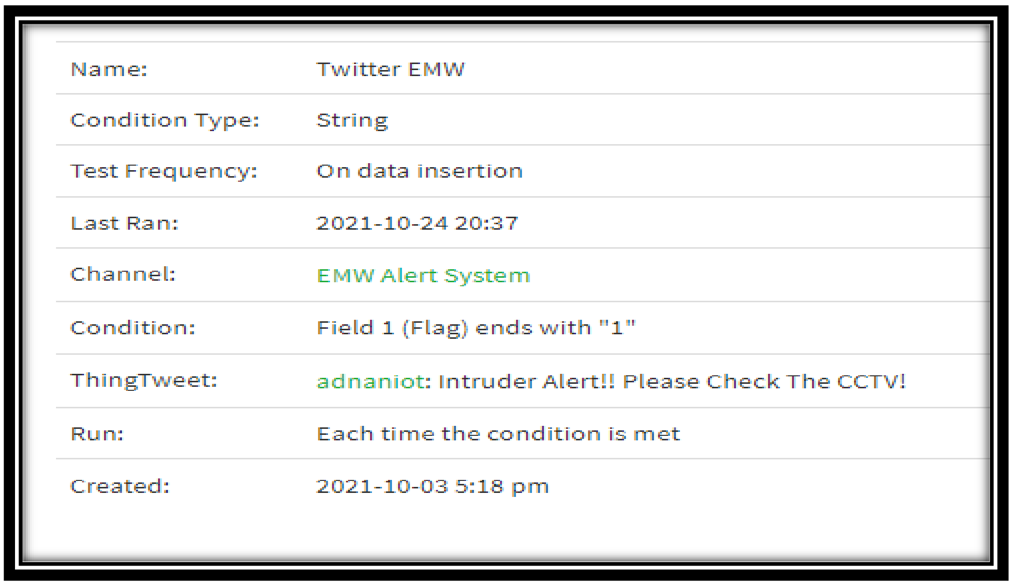
```
alert_body = 'Please Check CCTV';
alert_subject = 'Intruder Alert';
alert_api_key = 'TAKG5N7ZH1Q7TQ31Z3PUG';
alert_url= "https://api.thingspeak.com/alerts/send";
jsonmessage = sprintf(['{"subject": "%s", "body": "%s"}'],
alert_subject,alert_body);
options = weboptions("HeaderFields", {'Thingspeak-Alerts-API-Key',
alert_api_key; 'Content-Type','application/json'});
result = webwrite(alert_url, jsonmessage, options);
```

ThingSpeak allows sending of 800 emails per year on a free account.

React:

React works with ThingHTTP, ThingTweet, and MATLAB Analysis apps to perform actions when channel data meets a certain condition. In our case, React posts a tweet and an email to send an alert when the EMW alert field receives a flag value "1"

To send the tweet, we use the ThingTweet react in the following configuration



Name:	Twitter EMW
Condition Type:	String
Test Frequency:	On data insertion
Last Ran:	2021-10-24 20:37
Channel:	EMW Alert System
Condition:	Field 1 (Flag) ends with "1"
ThingTweet:	adnaniot: Intruder Alert!! Please Check The CCTV!
Run:	Each time the condition is met
Created:	2021-10-03 5:18 pm

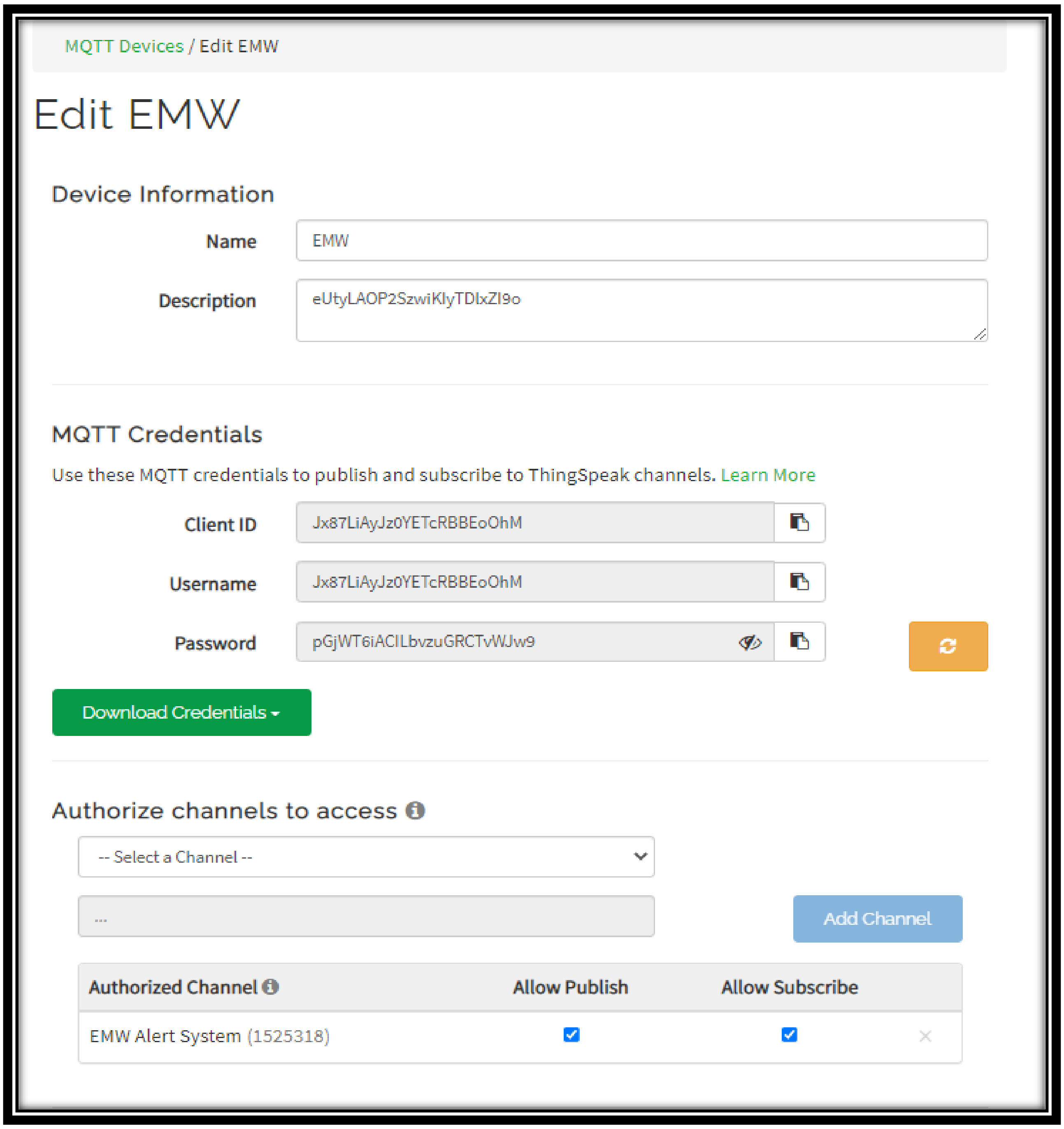
Fig 12. Thingtweet configuration

This sends a tweet once we receive a flag message. Similarly, we add a new react in which MATLAB Analysis is also run under the same configuration.

Desktop Notifications (MQTTX)

MQTT is a lightweight, publish-subscribe network protocol that transports messages between devices. The single most amazing advantage of MQTT protocol is that it is practically instantaneous since it does not require the Thingspeak Server's response which is normally required in the other two alerting methods.

ThingSpeak allows the addition of one MQTT device for free. We have added a device with the following configurations.


MQTT Devices / Edit EMW

Edit EMW

Device Information

Name	EMW
Description	eUtyLAOP2SzwiKlyTDIxZI9o

MQTT Credentials

Use these MQTT credentials to publish and subscribe to ThingSpeak channels. Learn More

Client ID	Jx87LiAyJz0YETcRBBEoOhM
Username	Jx87LiAyJz0YETcRBBEoOhM
Password	pGjWT6iACILbvzuGRCTvWJw9

Download Credentials

Authorize channels to access

-- Select a Channel --

...

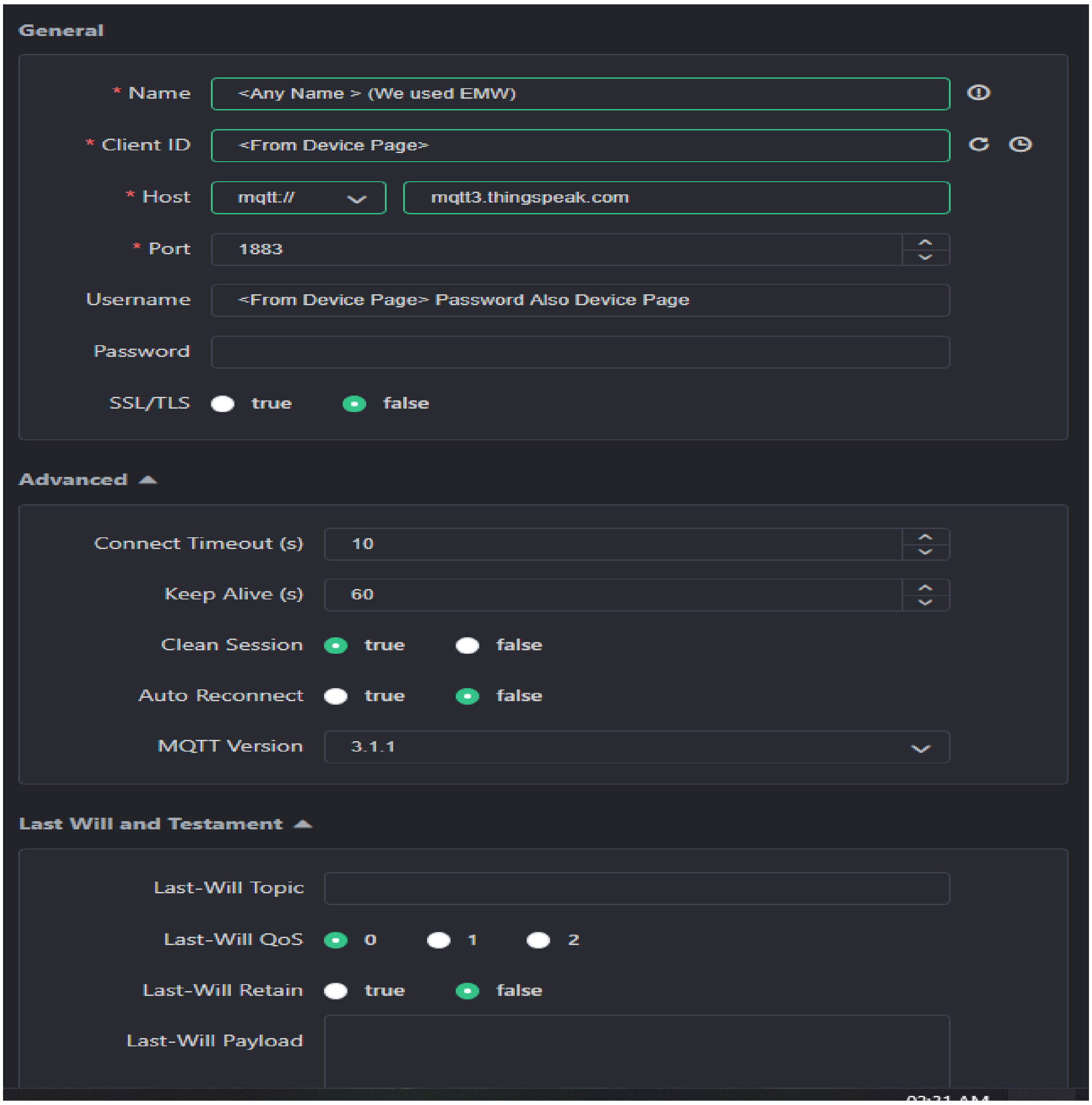
Add Channel

Authorized Channel	Allow Publish	Allow Subscribe
EMW Alert System (1525318)	✓	✓

Fig 13. Thingspeak MQTT configuration

Once the device is created, a client ID, username and password are generated. This allows desktop apps such as MQTTx to subscribe to the MQTT device on thingspeak which is publishing data from the authorised channel (the alert channel) continuously.

We have used MQTTx to push desktop notifications. To make MQTTx a subscriber, we first add thingspeak as a broker using the following configurations



General

* Name	<Any Name > (We used EMW)
* Client ID	<From Device Page>
* Host	mqtt:// mqtt3.thingspeak.com
* Port	1883
Username	<From Device Page> Password Also Device Page
Password	
SSL/TLS	true [✓ false]

Advanced

Connect Timeout (s)	10
Keep Alive (s)	60
Clean Session	[✓ true] false
Auto Reconnect	true [✓ false]
MQTT Version	3.1.1

Last Will and Testament

Last-Will Topic	
Last-Will QoS	[✓ 0] 1 2
Last-Will Retain	true [✓ false]
Last-Will Payload	

Fig 14. MQTTX subscriber configuration

Then we subscribe to a field by adding a new subscription with topic channels/<channelID>/subscribe/fields/field<field number>

This sets up the desktop notification.

We also are experimenting with Slack, Discord and Telegram APIs to send messages through those apps.

IFTTT

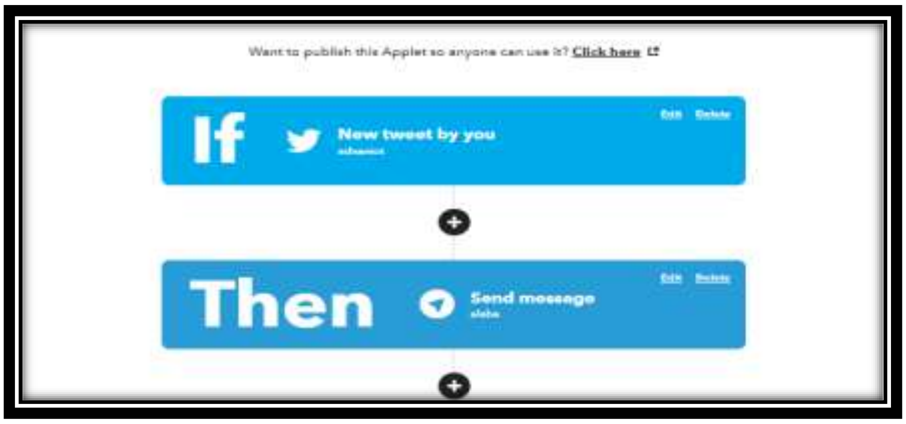
IFTTT, or "If This Then That," is an acronym. It is a web-based tool that enables users to design automated workflows, also referred to as applets, across various web services and apps. These applets use a change or event in one service to initiate a response in another. A trigger is an occurrence or modification in one service, such as getting an email or tweeting something new. A task that is automatically carried out in response to the trigger is referred to as an action. Examples of actions include issuing an alert or adding a new row to a spreadsheet.

An applet that for instance that would email you each time a new item is uploaded to your preferred twitter feed. IFTTT gives users a visual interface so they can make and manage applets. Customers have a large selection of pre-built applets to pick from, or they can design their own by choosing a trigger and an action from the options provided. Once an applet is formed, IFTTT regularly checks the trigger service for any modifications or new events, and whenever the trigger takes place, the accompanying action is carried out automatically.

IFTTT may be integrated with a huge selection of web programmes and services, including social media sites, office suites, smart home appliances, and more. With a straightforward visual interface, users can browse already-existing applets or design their own unique applets. IFTTT was created to streamline repetitive processes, make them simpler, and make workflows that are unique to each user's needs. Essentially, IFTTT automates processes between various web applications and services to expedite and simplify repetitive chores.

We have created three applets with different purposes

a) Integrate Twitter with telegram - To integrate Twitter with Telegram, you can create an IFTTT applet that sends a Telegram message every time a new tweet is posted by a specific Twitter account. You must link your Twitter and Telegram accounts in order to begin. Once both accounts are linked, you can build a new applet and set the action to "Send message" on Telegram and the trigger to "New tweet by selected person" on Twitter. You can enter the Twitter handle of the account you want to get notifications for in the "New tweet by specific user" trigger. You can define the Telegram conversation, the message format, and any other required details in the "Send message" action. These details can include the tweet content, the author name, and more. Once the applet is established, IFTTT will automatically send a message to the designated Telegram conversation each time a new tweet is made and continuously check the specified Twitter account for new tweets.

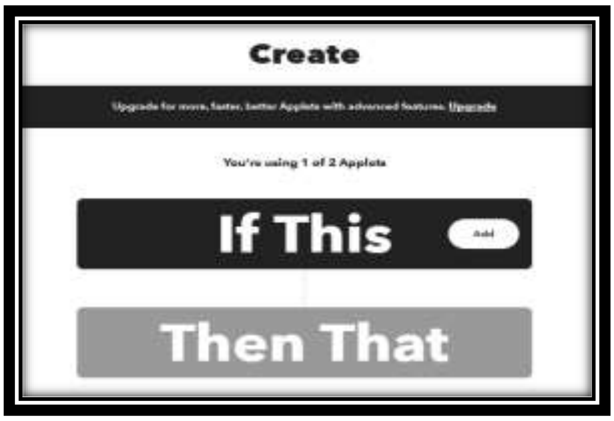


Want to publish this Applet so anyone can use it? Click here

Fig 15. Telegram Applet

Setting up the IFTTT to make an applet

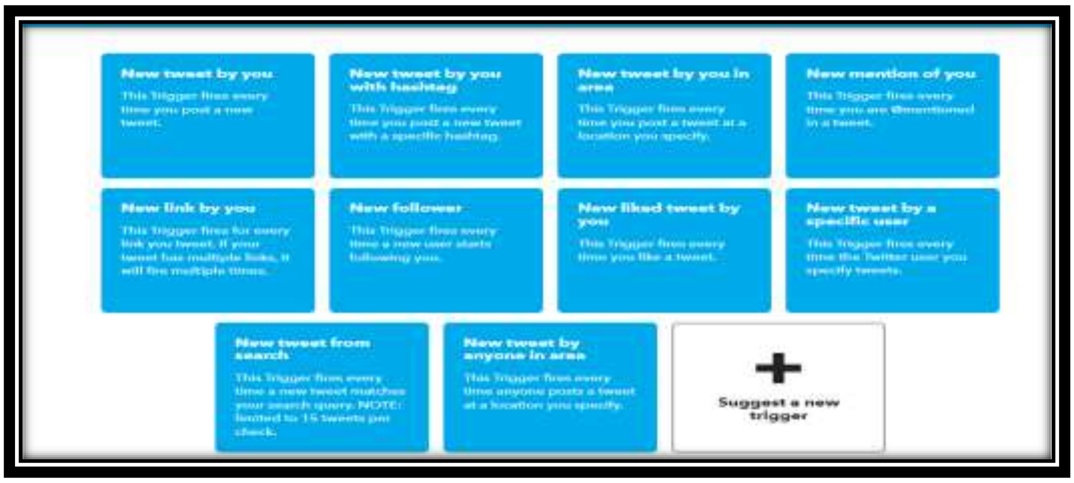
This is the screen that comes up when making an applet. We have to enter two main fields which are "IF this" which is the trigger cause, and "Then that" which is the event that occurs once triggered. We can choose from a wide range of triggers for each service. For example we need to use twitter as a trigger event.



Upgrade for more, faster, better Applets with advanced features. Upgrade

You're using 1 of 2 Applets

Fig 16. Creating applet



New tweet by you
This Trigger fires every time you post a new tweet.

New tweet by you with hashtag
This Trigger fires every time you post a new tweet with a specific hashtag.

New tweet by you in area
This Trigger fires every time you post a tweet at a location you specify.

New mention of you
This Trigger fires every time you are @mentioned in a tweet.

New link by you
This Trigger fires for every link you tweet. If your tweet has multiple links, it will fire multiple times.

New follower
This Trigger fires every time a new user starts following you.

New liked tweet by you
This Trigger fires every time you like a tweet.

New tweet by a specific user
This Trigger fires every time the Twitter user you specify tweets.

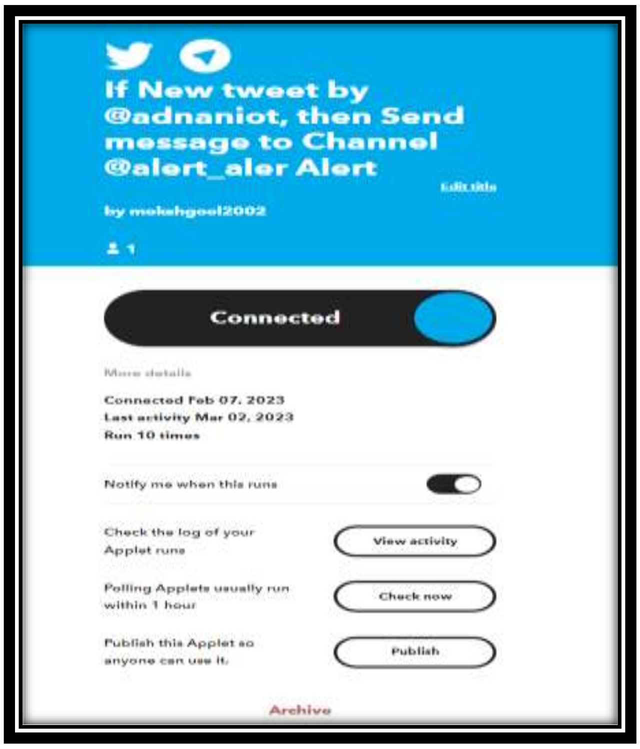
New tweet from search
This Trigger fires every time a new tweet matches your search query. NOTE: limited to 15 tweets per check.

New tweet by anyone in area
This Trigger fires every time anyone posts a tweet at a location you specify.

Suggest a new trigger

Fig 17. Twitter Triggers

Once we have set a twitter as a trigger alert we have to connect the account for which we receive the tweet as the trigger. We then select the cause event which is what will occur once the trigger happens. Here we use telegram alert as the cause event. We have to enter set fields such as the telegram account, whether we want the alert to go to a private channel or a chat with IFTTT bot. We can then customize the body of the text with certain fields. Here we are using only the text of the tweet, the author and the timestamp.



If New tweet by @adnaniot, then Send message to Channel @alert_aler Alert

Edit title

by mehehgoel2002

1

Connected

More details

Connected Feb 07, 2023
Last activity Mar 02, 2023
Run 10 times

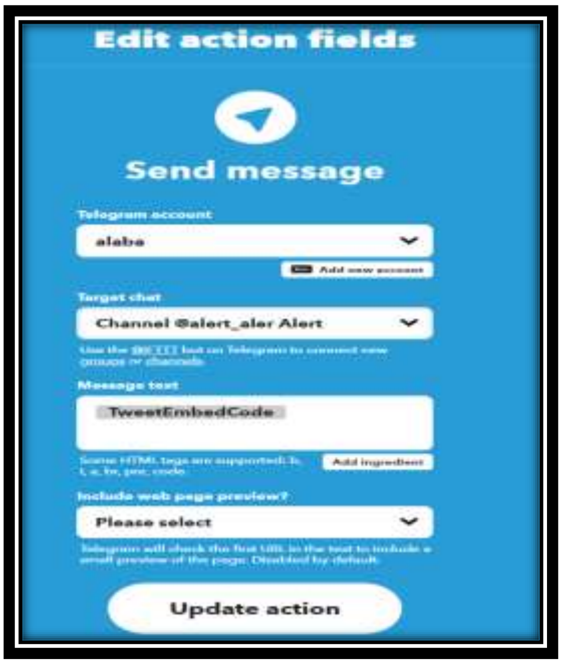
Notify me when this runs

Check the log of your Applet runs — View activity

Polling Applets usually run within 1 hour — Check now

Publish this Applet so anyone can use it. — Publish

Archive



Edit action fields

Send message

Telegram account: alaba

Add new account

Target chat: Channel @alert_aler Alert

Use the @IFTTT bot on Telegram to connect new groups or channels

Message text: TweetEmbedCode

Some HTML tags are supported: b, i, a, br, pre, code

Add ingredient

Include web page preview?: Please select

Telegram will check the first URL in the text to include a small preview of the page. Disabled by default.

Update action

Fig 19. Action Fields

Fig 18. Created Applet

Once Created the applet looks like this and contains certain fields – the switch shows whether the applet is connected or not. We can check the log of when the applet has run and if it succeeded or failed and for what reason it failed. We can even publish it so anyone on the internet can use this applet.

b) Integrate web hooks with VOIP calling - VoIP, or voice over internet protocol, is a technology that enables you to place calls online as opposed to over landlines or mobile networks. Despite what its name might imply, VoIP services today are far more versatile and can provide video calls, file transfers, group calls, and much more. It's also known as internet telephony or IP telephony. Computers, smartphones, tablets, dedicated VoIP phones, regular phones linked to an adaptor, and other internet-connected devices can all be used to make VoIP calls. IFTTT uses a trigger and cause action, with the trigger being set to each time the web hook is triggered. Once this is triggered IFTTT uses its own app to send a VOIP call to the device connected to the IFTTT account and upon answering the call it voices the text of the specified message we add.

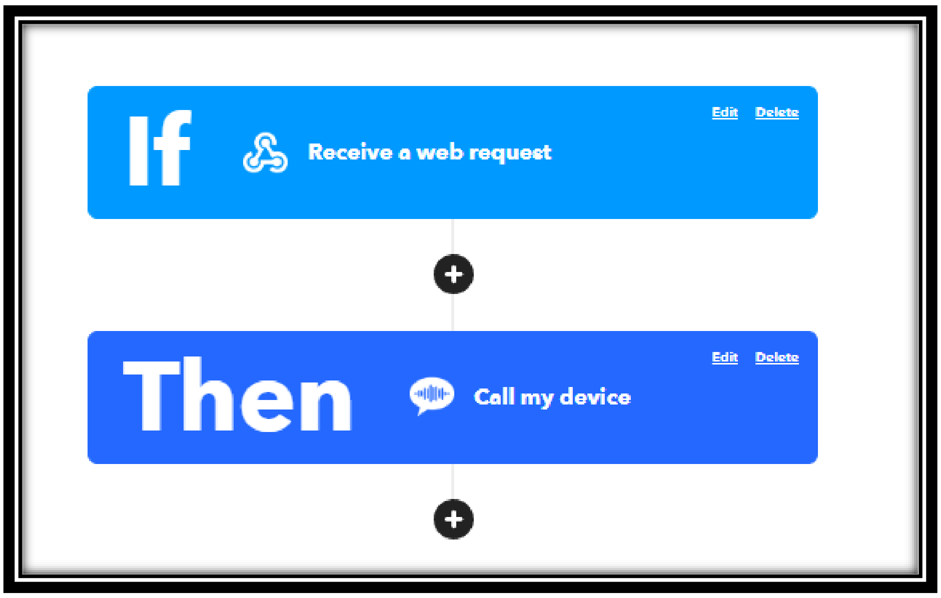
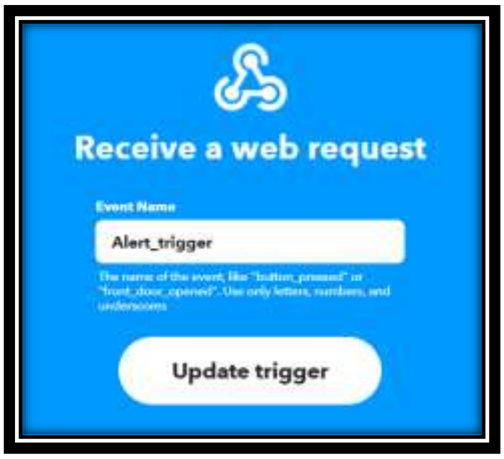


Fig 20. VOIP Applet

Setting up the IFTTT to make the applet

We first select the trigger action as trigger from a web hook, in this we specify the name for the event name. This event name is put into a custom URL web hook that must be triggered for this applet to run. This web hook will look like this:

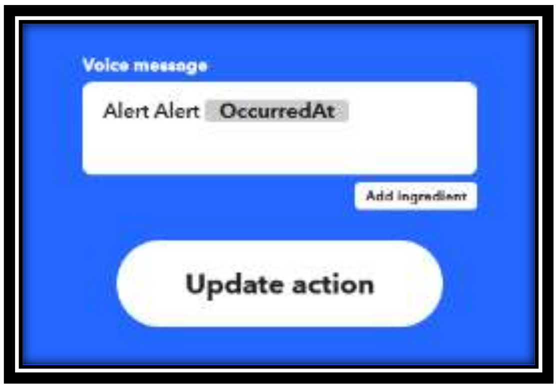
https://maker.ifttt.com/trigger/Alert_trigger/json/with/key/d9g-mOe4H2UAx8udz1ZVqGDaQp_Ws_S4tNr3evnEovk .



Receive a web request

Event Name: Alert_trigger

The name of the event, like "button_pressed" or "front_door_opened". Use only letters, numbers, and underscores



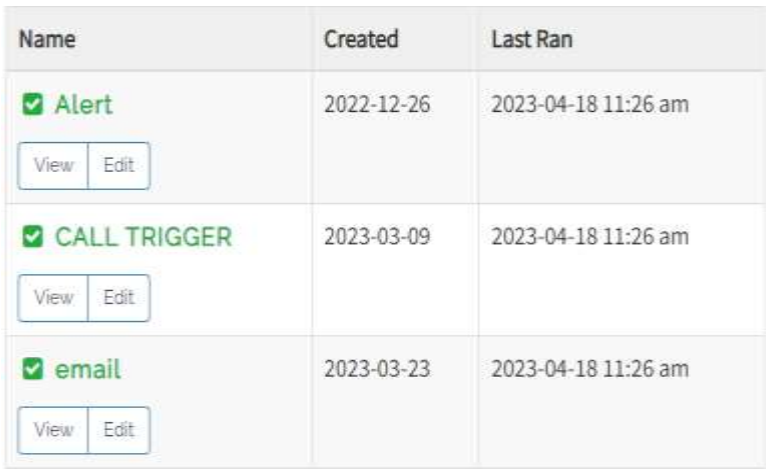
Voice message: Alert Alert OccurredAt

Fig 21. Event Name

Fig 22. Alert Message Customization

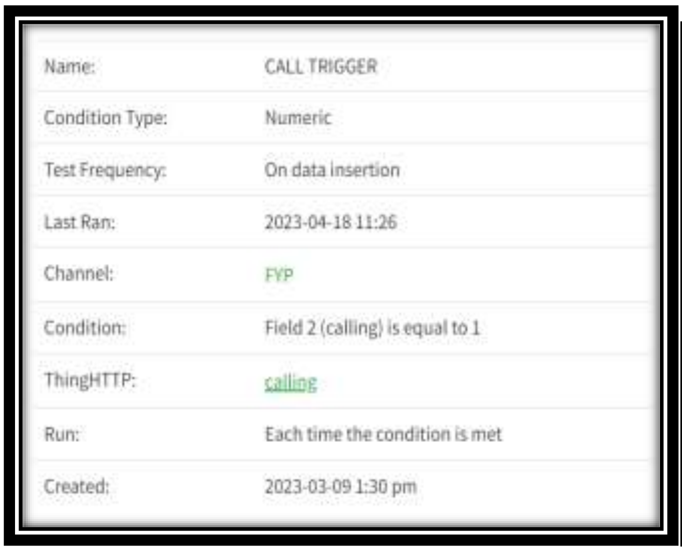
When this web hook is triggered it will execute the next part of the applet which is to send a voice-call to the specified number. We can customise the message we want to send and add some different ingredients such as the timestamp it occurred at as well. We simply need to code it into the message.

The bot reacts and sends a web hook on IFTTT for it to call, we need to set up a new react and Thing-http. The processor upon detection of any human will send a flag to our Thingspeak Channel, upon receiving flags the channel has setup reacts which will react to the incoming flags. We have setup so that upon receiving the flags the channel will use Thingtweet to send a Tweet and a ThingHTTP to send a web hook to our IFTTT applet.



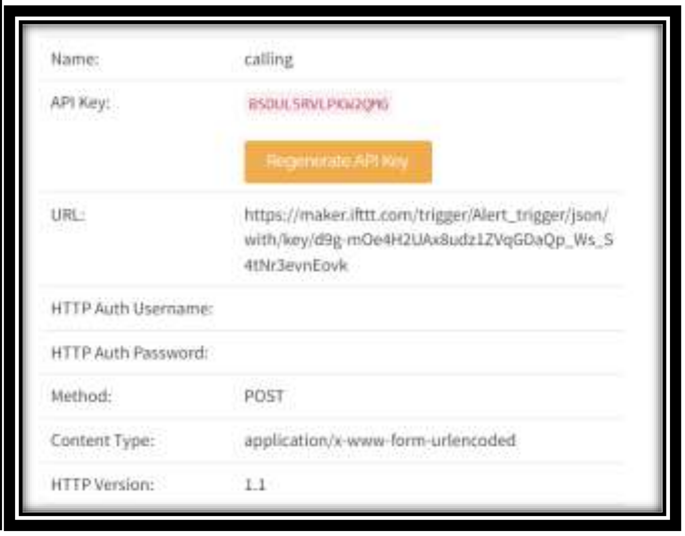
Name	Created	Last Ran
✓ Alert View Edit	2022-12-26	2023-04-18 11:26 am
✓ CALL TRIGGER View Edit	2023-03-09	2023-04-18 11:26 am
✓ email View Edit	2023-03-23	2023-04-18 11:26 am

Fig 23. Thingspeak Reacts



Name:	CALL TRIGGER
Condition Type:	Numeric
Test Frequency:	On data insertion
Last Ran:	2023-04-18 11:26
Channel:	FYP
Condition:	Field 2 (calling) is equal to 1
ThingHTTP:	calling
Run:	Each time the condition is met
Created:	2023-03-09 1:30 pm

Fig 24. Calling React



Name:	calling
API Key:	8SOULSRVLPKW2QM6
URL:	https://maker.ifttt.com/trigger/Alert_trigger/json/with/key/d9g-mOe4H2UAx8udz1ZVqGDaQp_Ws_S4tNr3evnEovk
HTTP Auth Username:	
HTTP Auth Password:	
Method:	POST
Content Type:	application/x-www-form-urlencoded
HTTP Version:	1.1

Fig 25. Calling ThingHTTP

a) Integrate Twitter with discord - To integrate Twitter with Discord, you can create an IFTTT applet that triggers a Discord message every time a new tweet is posted by a specific Twitter account. We then link our Twitter and Discord accounts in order to get started. Once both accounts are linked, you can build a new applet and set the action to "Post message" on Discord and the trigger to "New tweet by specific user" on Twitter. You can enter the Twitter handle of the account you want to get notifications for in the "New tweet by specific user" trigger. The Discord server, channel, and message format, which can include the tweet text, author name, and any other desired information, can all be specified in the "Post message" action.

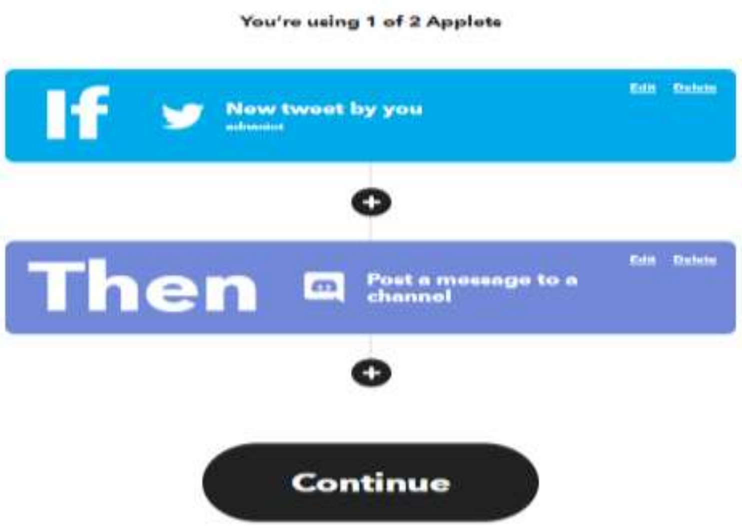
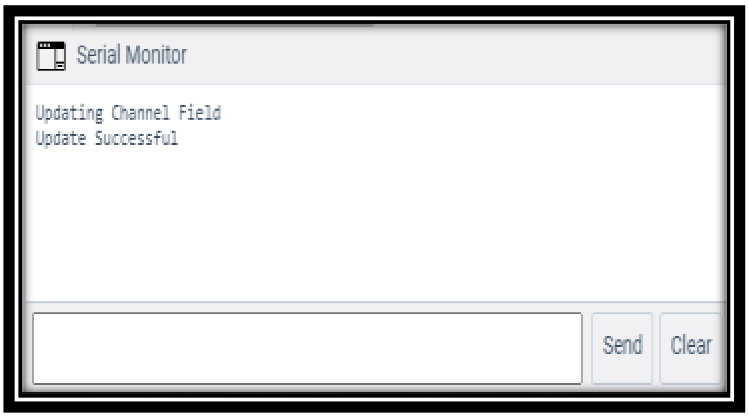


Fig 26. Discord Applet

Chapter 7

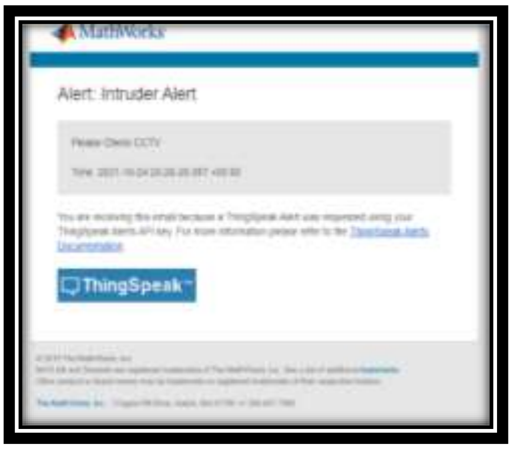
Testing and Results



```
Updating Channel Field
Update Successful
```

Sending Code Through serial communication . Affirming connection between processor and the laptop

Fig 27. Code sent from Microcontroller



Alert: Intruder Alert

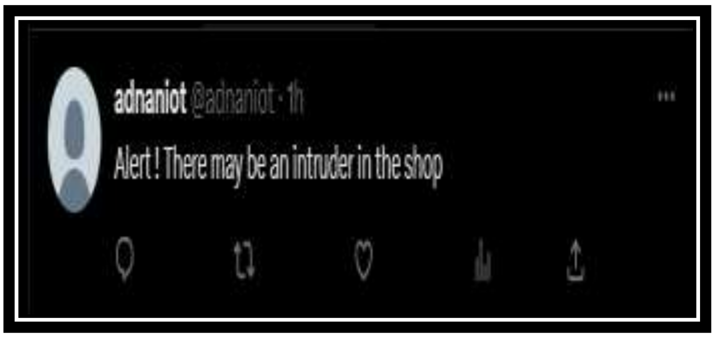
Please Check CCTV

Time: 2021-10-24 [illegible]

[illegible]

Sending Email Alerts using Matlab Analysis On Thingspeak Platform . We Setup ThingReact to react to a flag and send an email alert with a customised message

Fig 28. Email alert

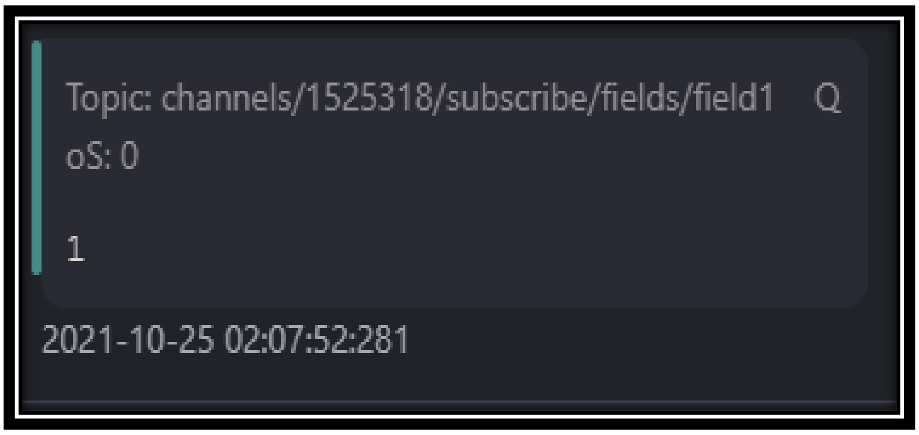


adnaniot @adnaniot · 1h

Alert ! There may be an intruder in the shop

Sending twitter Alerts using ThingTweet function . We Setup ThingTweet to react to a flag and send an email alert with a customised message.

Fig 29. Twitter alert



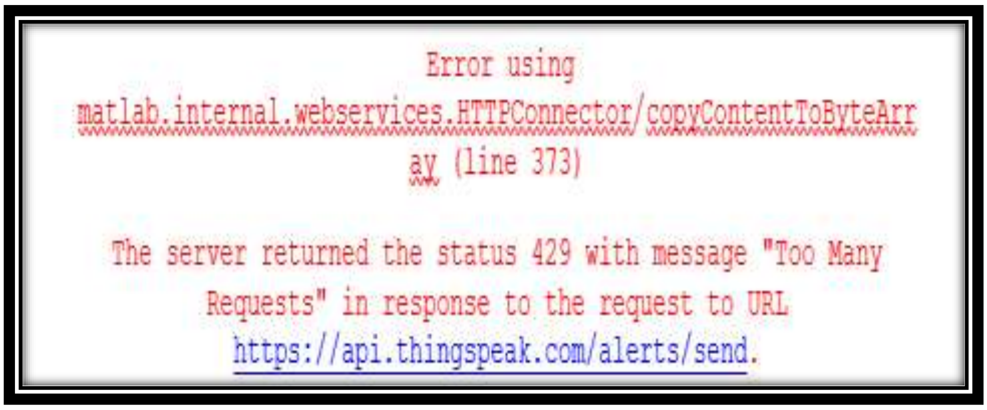
```
Topic: channels/1525318/subscribe/fields/field1    Q
oS: 0

1

2021-10-25 02:07:52:281
```

Sending MQTTX Alerts using MQTT Platform . This is much faster and requires low net connection

Fig 30. MQTT alert

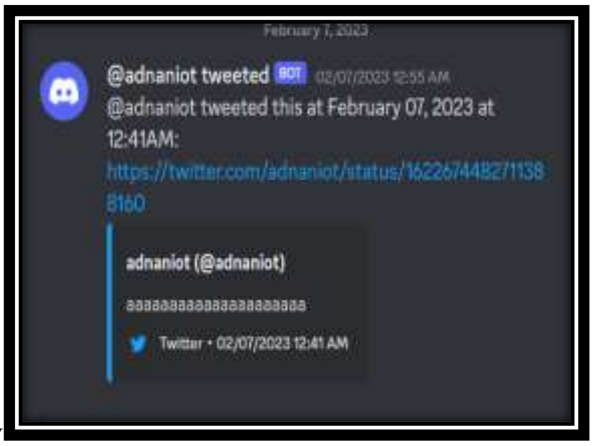


```
Error using
matlab.internal.webservices.HTTPConnector/copyContentToByteArr
ay (line 373)

The server returned the status 429 with message "Too Many
Requests" in response to the request to URL
https://api.thingspeak.com/alerts/send.
```

Too many emails within a set time can cause spam errors and will result in this error message

Fig 31. Email spam error



February 7, 2023

@adnaniot tweeted BOT 02/07/2023 12:55 AM

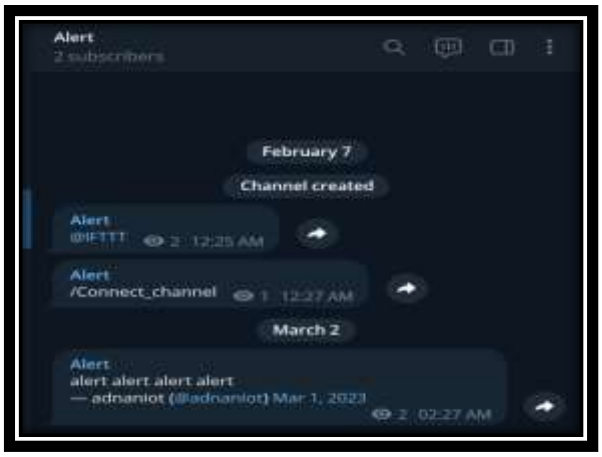
@adnaniot tweeted this at February 07, 2023 at 12:41AM:

https://twitter.com/adnaniot/status/16226744827l138 8160

adnaniot (@adnaniot)

aaaaaaaaaaaaaaaaaaaaa

Twitter • 02/07/2023 12:41 AM



Alert
2 subscribers

February 7

Channel created

Alert
@IFTTT 2 12:25 AM

Alert
/Connect_channel 1 12:27 AM

March 2

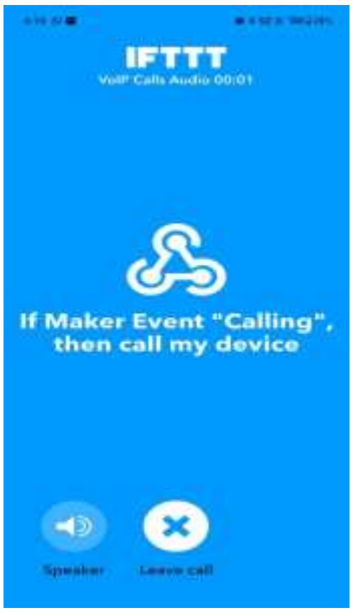
Alert
alert alert alert alert
— adnaniot (@adnaniot) Mar 1, 2023
2 02:27 AM

v

Fig 32. Discord Alert

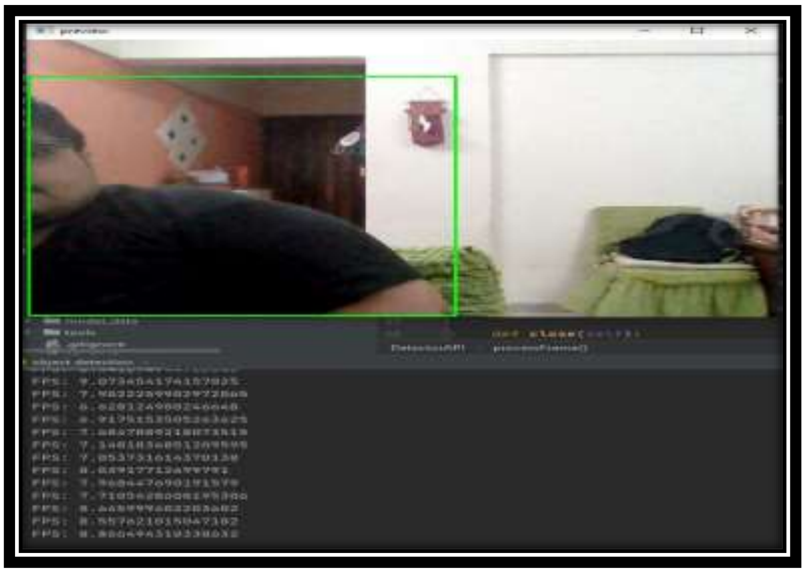
Fig 33. Telegram alert

Sending Discord and Telegram Alerts using a combination of React and IFTTT services . We Setup ThingReact to react to a flag and send another flag to IFTTT which posts alerts on respective channels. We here connect twitter with discord and Telegram.



This is one of our most important alerts, in this we send an alert using a voice call. This is most important as it sends a continuous buzz as opposed to as singular message or flag which might go unnoticed. We use Thing React, ThingHTTP and IFTTT for this alert. Thing React reacts to the flag and activates the ThingHTTP, ThingHTTP sends a flag on a custom URL web hook which then IFTTT uses to pass the call to the specified number

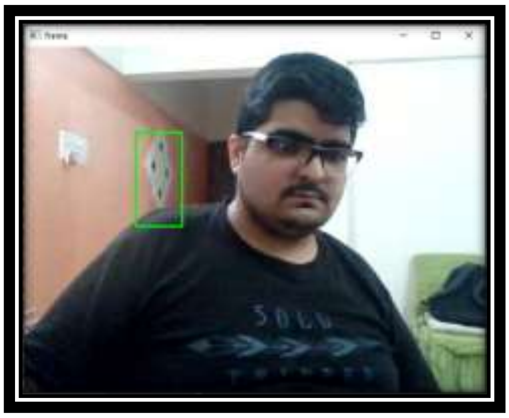
Fig 34. Call Alert



```
FPS: 7.073454174157025
FPS: 7.982223999829772865
FPS: 6.628124908246048
FPS: 6.917915350052263625
FPS: 7.886788921887351 3
FPS: 7.340183680120959 5
FPS: 7.053731614570138
FPS: 8.689277124999791
FPS: 7.960447090191579
FPS: 7.710942800081975300
FPS: 8.445999682283482
FPS: 8.557621015947182
FPS: 8.860494519338032
```

All different types of Neural Net that have different levels of speed and accuracy, we have used Inception v2 to provide us a decent balance of speed and accuracy. If the application area does not have a lot of deterring agents (Mannequins, Pictures of humans etc.), We can use Mobile net for more speed.

Fig 35. SSD InceptionV2



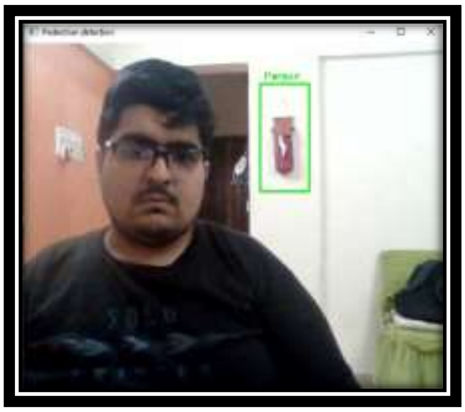


Fig 36. Cassacade classifier

Fig 37. Hog Descriptor

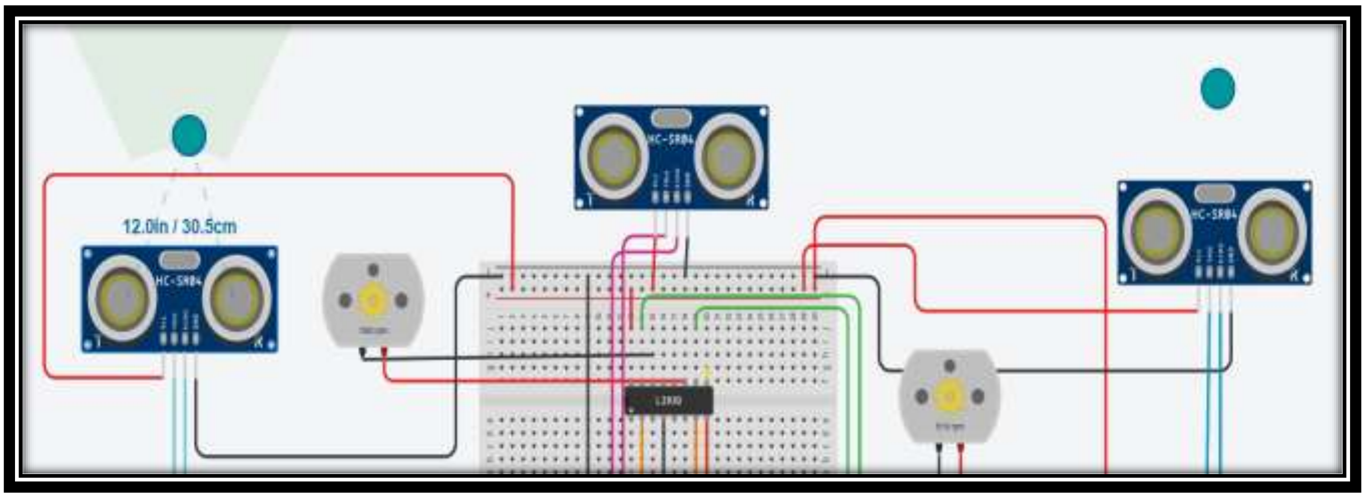


Fig 38. Navigation PID Simulation

Chapter 8

Advantages, Limitations and Applications

Advantages

- This Patrol Bot is an secondary system and not a primary system so it acts as a multilayer security blanket thus being one of a kind systems
- It runs on an independent power source thus power outages will not effect this security measure
- As it is autonomous it has negligible to no error as human error is removed
- The small size of this bot paired with the range of the ultrasonic sensors makes the robot almost undetectable to the intruder before setting of the alert system
- The alert system uses IOT mechanisms such as thingspeak, MQTT, Matlab Thingtweet and Thingreact to instantly send alerts making the reaction time almost instantaneous
- The multiple ways of sending alerts acts as a failsafe as if one service is down the other services will still go through

Limitations:

- The range of the Patrol Bot decreases as the distance from the furniture/objects decreases i.e. extruding parts of the room will decrease the overall range of the bot
- The battery life of the patrol robot is limited as it works on a rechargeable lithium polymer battery and therefore has to be charged before every patrol session and works on a limited timeline
- The alert system heavily relies on Internet service and Wi-Fi connection to send alerts across all its platforms and therefore becomes handicapped during connection issues
- If it is knocked of course by manual or natural phenomenon it will be immobilized until human intervention due to its natural code

Applications:

- In offices and building to aid the already existing security measures by adding an extra layer of security
- In huge area with many entrances as the alert from the bot will help to pin point the location of the intruder
- In small businesses as it provides lower cost of implementation
- With additions to movement and IOT domains it can be used in border patrol systems as well

Chapter 9

Conclusion and Future Scope

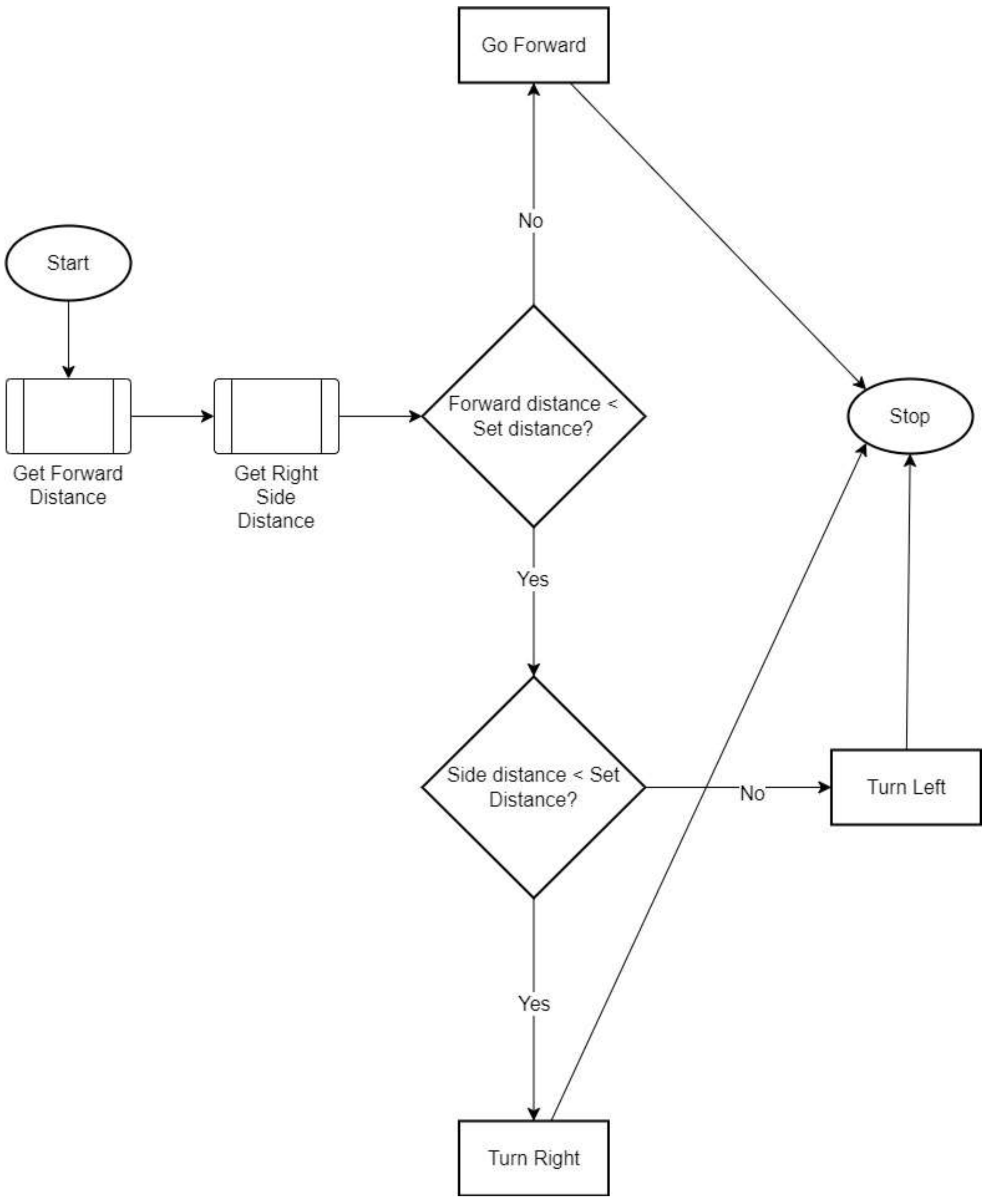
Conclusion

- This robot can be a very important addition to existing security systems which include mostly just a CCTV monitoring station. As there is a margin of human error in low Grade security systems such as commercial shops or Buildings,
- This Patrol Bot was designed to be a secondary cost effective security system which would integrate very easily with their own current standing Security systems and add a blanket of security.
- This project was modelled successfully on Proteus, Easy ADA and Solidworks and IOT system was implemented using ThingSpeak IOT platform which is again cost effective as it is open sourced.

Future Scope

- The navigational system can be further upgraded by the usage of Li-DAR technology which would map put the entire room and set a path for itself while avoiding obstacles through complex navigational paths
- We can also unify the alert system by making an autonomous app which pushes notifications on desktop as well as mobile platforms.
- Exploring use of Blynk IOT to streamline the alert system and notification as well as make a customized front panel for the bot
- To aid in the navigational process we can add mecanum wheels which will help the bot to turn and sharp angles in tight spaces further increasing its navigational abilities
- PID control for smooth motion makes the distance measurement algorithm smooth and less error prone

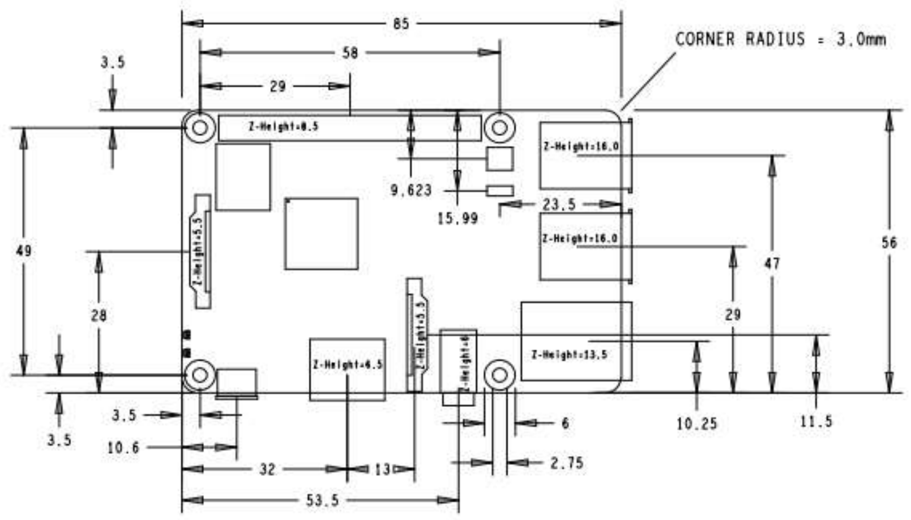
Appendix A: Soft Code Flowchart



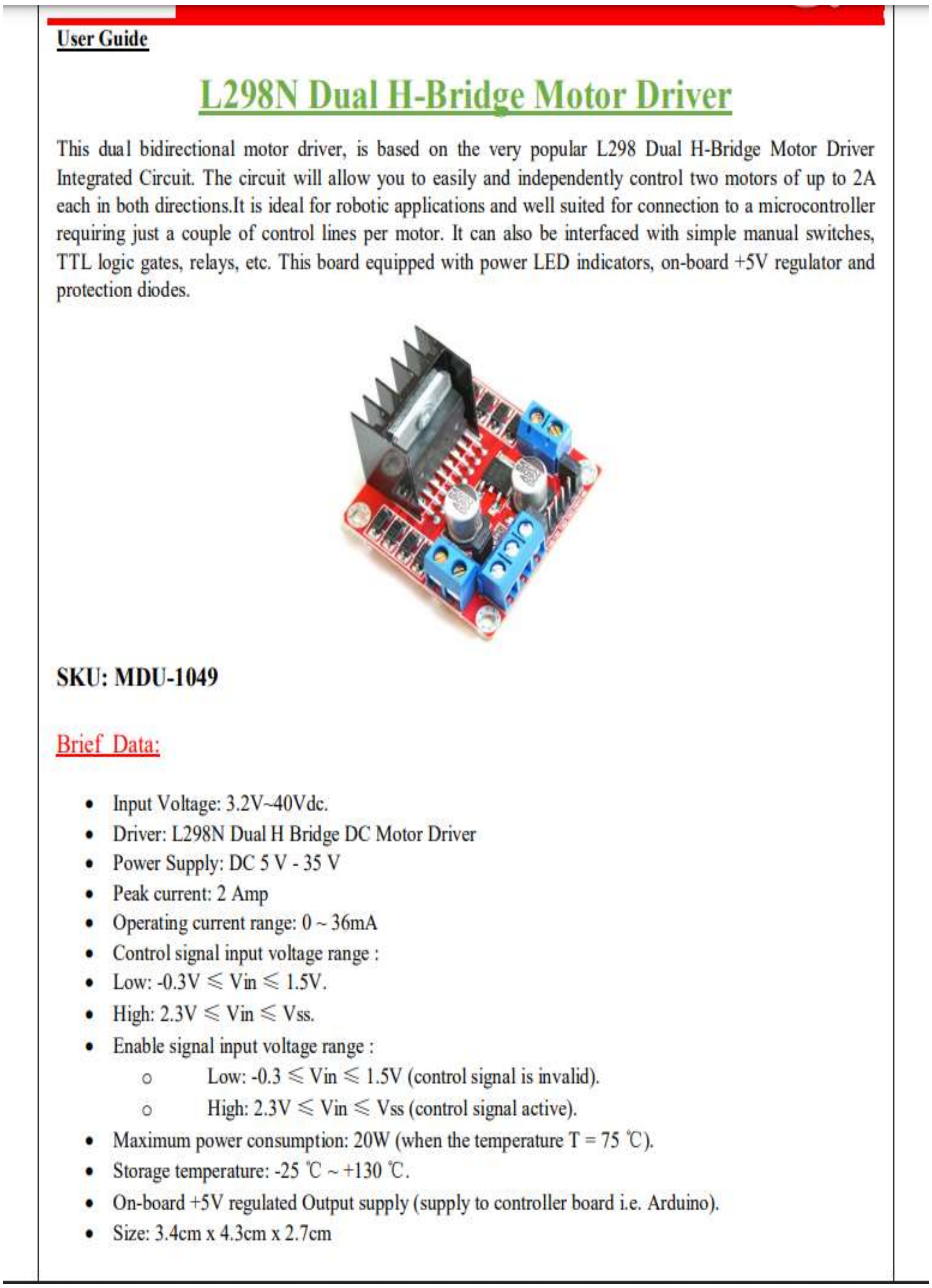
Appendix B: Data Sheets

Raspberry pi 3 model B+

Physical specifications



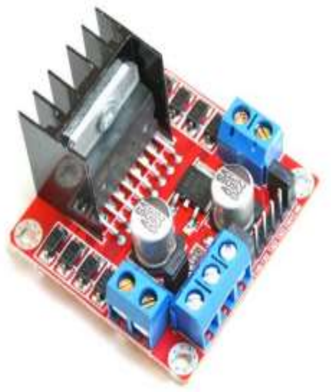
Processor:	Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz
Memory:	1GB LPDDR2 SDRAM
Connectivity:	■ 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE ■ Gigabit Ethernet over USB 2.0 (maximum throughput 300Mbps) ■ 4 × USB 2.0 ports
Access:	Extended 40-pin GPIO header
Video & sound:	■ 1 × full size HDMI ■ MIPI DSI display port ■ MIPI CSI camera port ■ 4 pole stereo output and composite video port
Multimedia:	H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics
SD card support:	Micro SD format for loading operating system and data storage
Input power:	■ 5V/2.5A DC via micro USB connector ■ 5V DC via GPIO header ■ Power over Ethernet (PoE)–enabled (requires separate PoE HAT)
Environment:	Operating temperature, 0–50°C
Compliance:	For a full list of local and regional product approvals, please visit www.raspberrypi.org/products/raspberry-pi-3-model-b+
Production lifetime:	The Raspberry Pi 3 Model B+ will remain in production until at least January 2023.



User Guide

L298N Dual H-Bridge Motor Driver

This dua l bidirectional motor driver, is based on the very popular L298 Dual H-Bridge Motor Driver Integrated Circuit. The circuit will allow you to easily and independently control two motors of up to 2A each in both directions.It is ideal for robotic applications and well suited for connection to a microcontroller requiring just a couple of control lines per motor. It can also be interfaced with simple manual switches, TTL logic gates, relays, etc. This board equipped with power LED indicators, on-board +5V regulator and protection diodes.

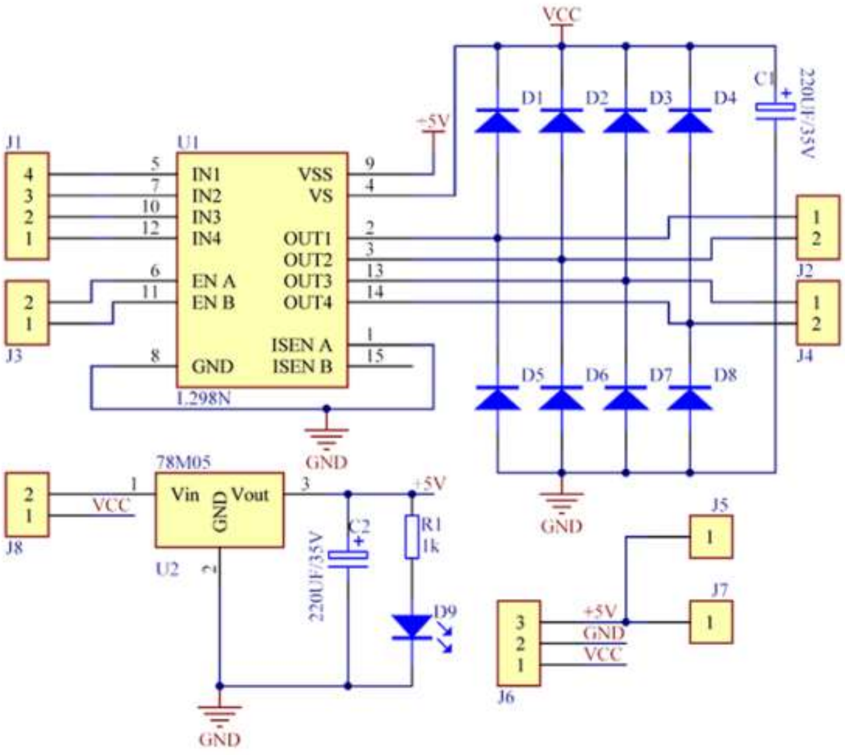


SKU: MDU-1049

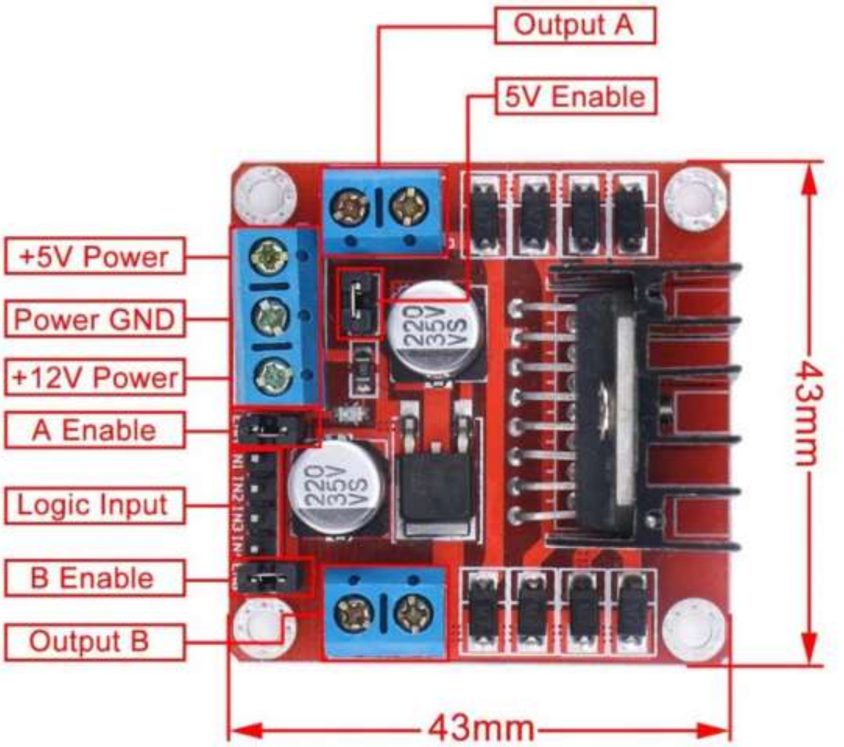
Brief Data:

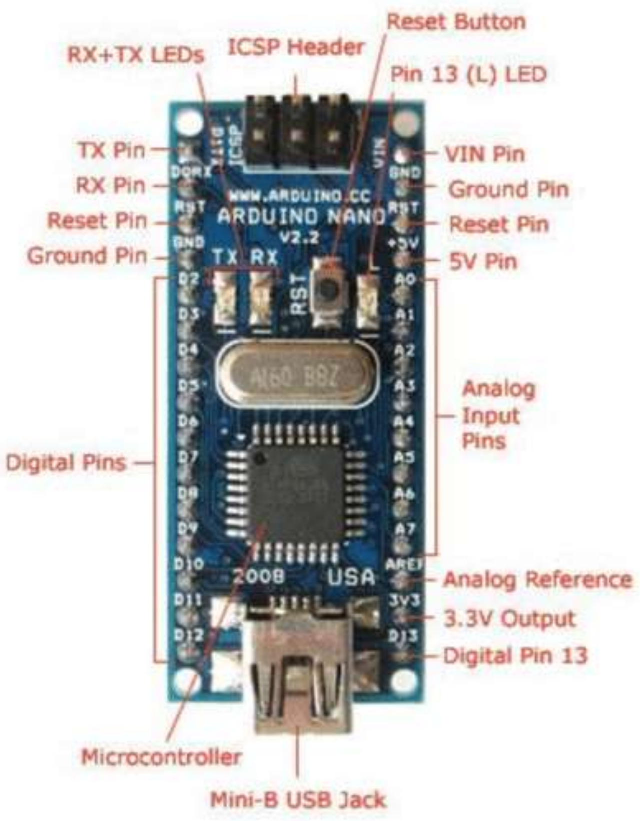
- Input Voltage: 3.2V~40Vdc.
- Driver: L298N Dual H Bridge DC Motor Driver
- Power Supply: DC 5 V - 35 V
- Peak current: 2 Amp
- Operating current range: 0 ~ 36mA
- Control signal input voltage range :
- Low: -0.3V ≤ Vin ≤ 1.5V.
- High: 2.3V ≤ Vin ≤ Vss.
- Enable signal input voltage range :
  - Low: -0.3 ≤ Vin ≤ 1.5V (control signal is invalid).
  - High: 2.3V ≤ Vin ≤ Vss (control signal active).
- Maximum power consumption: 20W (when the temperature T = 75 ℃).
- Storage temperature: -25 ℃ ~ +130 ℃.
- On-board +5V regulated Output supply (supply to controller board i.e. Arduino).
- Size: 3.4cm x 4.3cm x 2.7cm

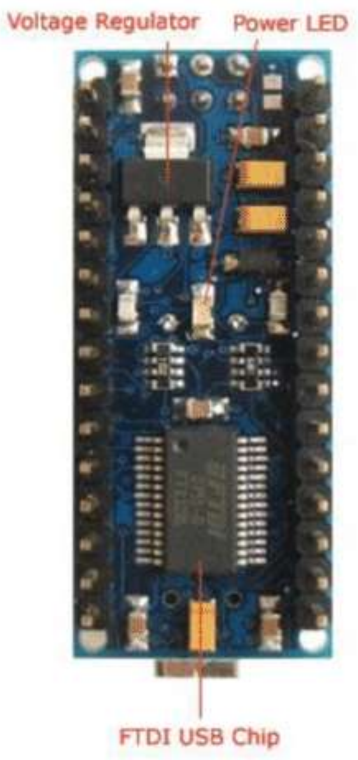
Schematic Diagram:



Board Dimension & Pins Function:







Schematic and Design

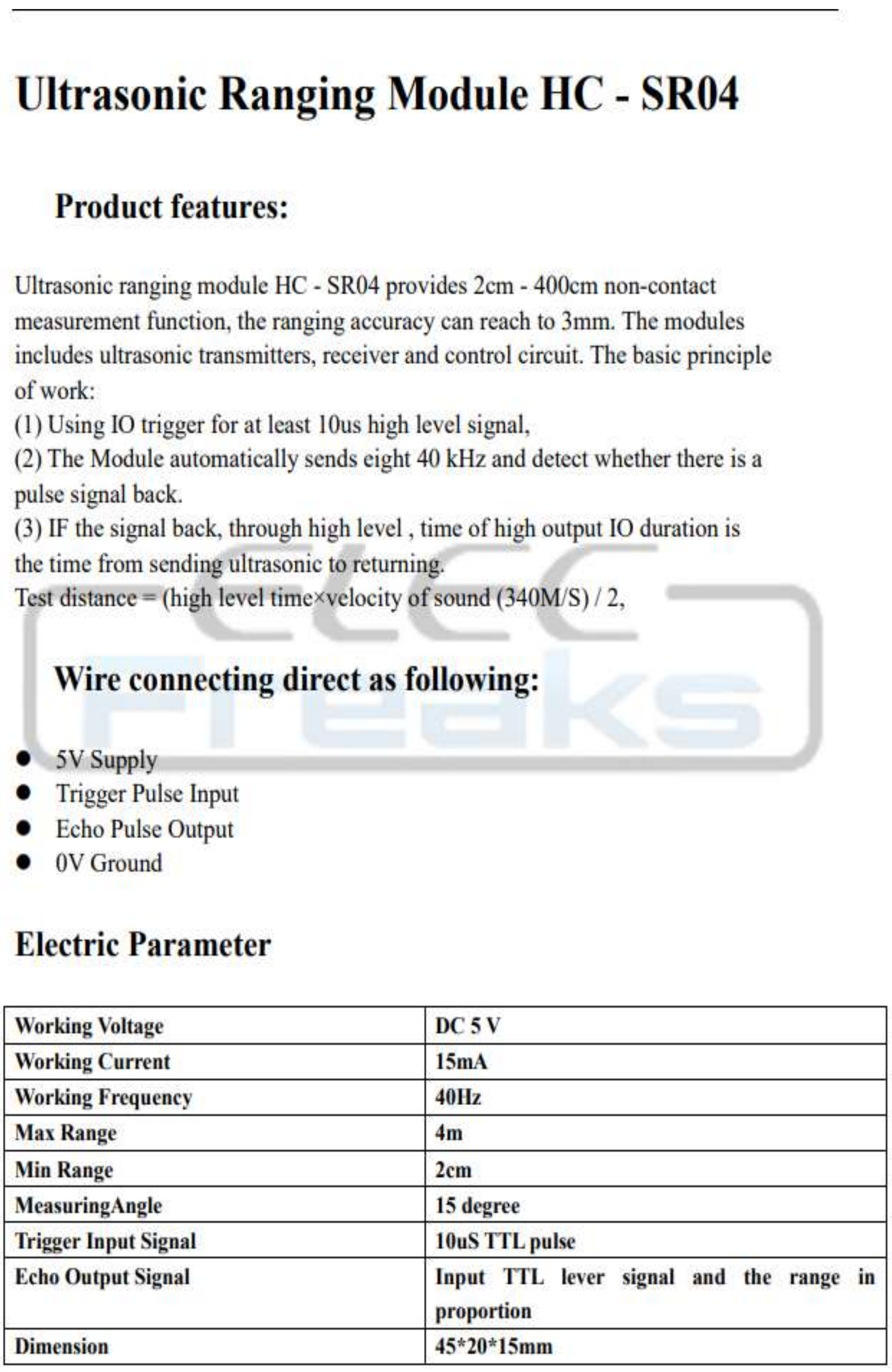
Arduino Nano 3.0 (ATmega328): schematic, Eagle files.
Arduino Nano 2.3 (ATmega168): manual (pdf), Eagle files. *Note:* since the free version of Eagle does not handle more than 2 layers, and this version of the Nano is 4 layers, it is published here unrouted, so users can open and use it in the free version of Eagle.

Specifications:

Microcontroller	Atmel ATmega168 or ATmega328
Operating Voltage (logic level)	5 V
Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8
DC Current per I/O Pin	40 mA
Flash Memory	16 KB (ATmega168) or 32 KB (ATmega328) of which 2 KB used by bootloader
SRAM	1 KB (ATmega168) or 2 KB (ATmega328)
EEPROM	512 bytes (ATmega168) or 1 KB (ATmega328)
Clock Speed	16 MHz
Dimensions	0.73" x 1.70"

Power:

The Arduino Nano can be powered via the Mini-B USB connection, 6-20V unregulated external power supply (pin 30), or 5V regulated external power supply (pin 27). The power source is automatically selected to the highest voltage source.



Ultrasonic Ranging Module HC - SR04

Product features:

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:
(1) Using IO trigger for at least 10us high level signal,
(2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
(3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.
Test distance = (high level time×velocity of sound (340M/S) / 2,

Wire connecting direct as following:

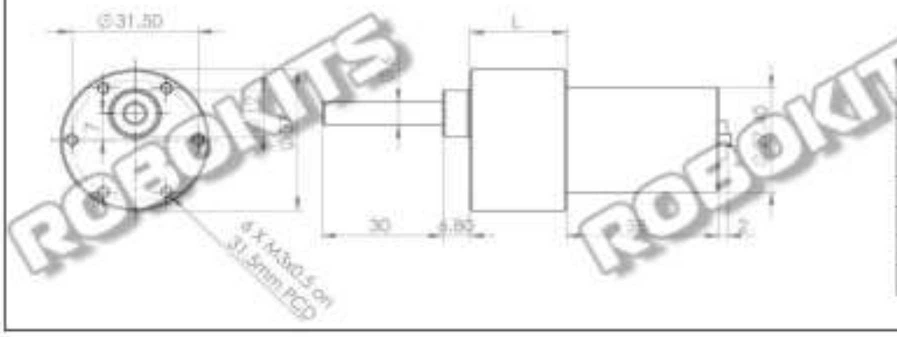
- 5V Supply
- Trigger Pulse Input
- Echo Pulse Output
- 0V Ground

Electric Parameter

Working Voltage	**DC 5 V**
Working Current	**15mA**
Working Frequency	**40Hz**
Max Range	**4m**
Min Range	**2cm**
MeasuringAngle	**15 degree**
Trigger Input Signal	**10uS TTL pulse**
Echo Output Signal	**Input TTL lever signal and the range in proportion**
Dimension	**45*20*15mm**

Model Number RKI-1145	Johnson Motor High Torque DC Geared 12V 200RPM - Grade A

Mechanical Drawing (Dimensions are in mm)



10RPM	L=28.5mm
30RPM	L=28.5mm
60RPM	L=26.5mm
100RPM	L=26.5mm
200RPM	L=25mm
300RPM	L=25mm
600RPM	L=22mm
900RPM	L=22mm

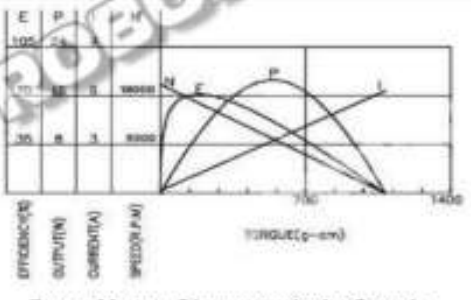
Technical specifications of Motor

Motor Type	High Torque Geared Motor
Operating voltage	12V DC
Motor Speed at Output Shaft (RPM)	200RPM
Stall Torque(Kgcm)	13.59Kgcm
Rated Torque(Kgcm)	5.2Kgcm
Gear Ratio	1 : 90
Weight	200gm

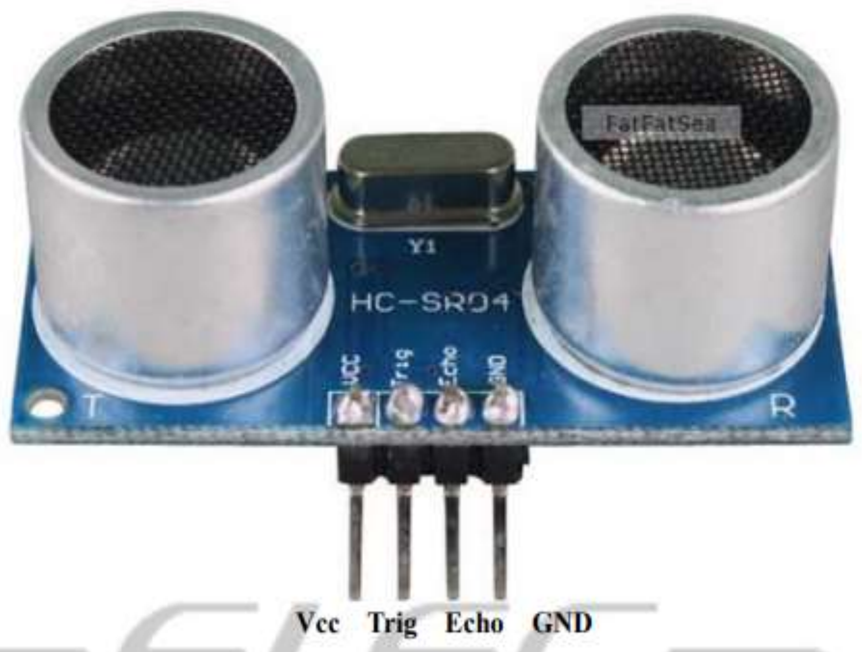
Base Motor Specifications

Nominal voltage(V)	12V
No-Load Speed (RPM)	18000
Rated Torque (gcm)	151
No-load current (A)	0.8A
Load current(A)	up to 7.5 A(Max)

RHINO

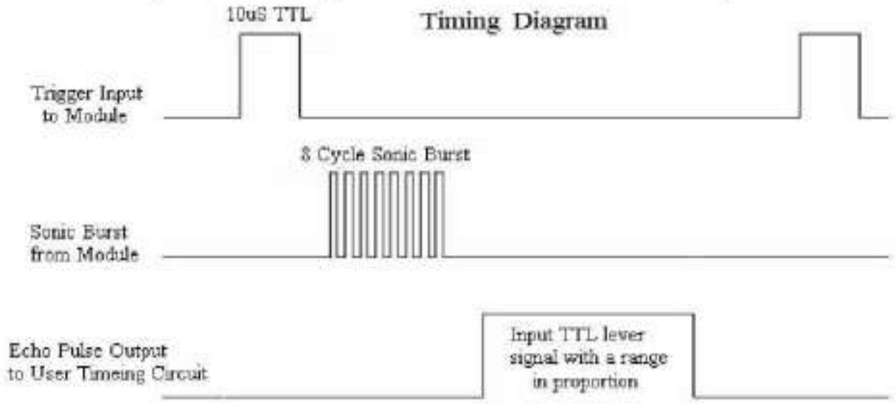


Base Motor Characteristic Graph



Timing diagram

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.


10uS TTL
Timing Diagram
Trigger Input to Module
8 Cycle Sonic Burst
Sonic Burst from Module
Echo Pulse Output to User Timeing Circuit
Input TTL lever signal with a range in proportion

Appendix C: List of Components

- Pi 4 Model B
- Arduino Nano
- HC-SR04 x32
- Wheels
- Johnson Motors
- L298N Motor Driver
- 12V rechargeable LiPO battery

Appendix D: List of Paper Presented and Published.

1. Patel, Neel and Tank, Darshan and Patel, Keyur and Adatkar, Rashmi and Thosar, Rohan, Vehicle Accident Alert System (April 8, 2019). 2nd International Conference on Advances in Science & Technology (ICAST) 2019 on 8th, 9th April 2019 by K J Somaiya Institute of Engineering & Information Technology, Mumbai, India,

2. Pakdaman, Mehran & Sanaatiyan, Mohammad Mehdi & Rezaei, Mahdi. (2010). A line follower robot from design to implementation: Technical issues and problems. 5 - 9. 10.1109/ICCAE.2010.5451881.

3. K. Muheden, E. Erdem and S. Vançin, "Design and implementation of the mobile fire alarm system using wireless sensor networks," 2016 IEEE 17th International Symposium on Computational Intelligence and Informatics (CINTI), 2016, pp. 000243-000246, doi: 10.1109/CINTI.2016.7846411.

4. Internet of Things Based Indoor Smart Surveillance and Monitoring System using Arduino and Raspberry Pi N C Sendhilkumar **et al** 2021 **J. Phys.: Conf. Ser.** 1964 062083

5. Kumar, Amit, et al. "Border Security System using Arduino & Ultrasonic Sensors." *International journal of scientific engineering and technology research* 6 (2017): 2489-2492.

6. Khaing, Khin Kyawt Kyawt. "Temperature and Humidity Monitoring and Control System with Thing Speak." *International Journal of Scientific Research and Engineering Development* (2019).

7. I. Abadi and N. El-Sheimy, "Manhattan World Constraint for Indoor Line-based Mapping Using Ultrasonic Scans," 2022 IEEE 12th International Conference on Indoor Positioning and Indoor Navigation (IPIN), 2022, pp. 1-8, doi: 10.1109/IPIN54987.2022.9918099.

8. P. S. B. Naga, P. J. Hari, R. Sinduja, S. Prathap and M. Ganesan, "Realization of SLAM and Object Detection using Ultrasonic Sensor and RGB-HD Camera," 2022 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET), 2022, pp. 167-171, doi: 10.1109/WiSPNET54241.2022.9767130.

9. J. Bayless, N. Kurihara, H. Sugimoto and A. Chiba, "Acoustic Noise Reduction of Switched Reluctance Motor With Reduced RMS Current and Enhanced Efficiency," in IEEE Transactions on Energy Conversion, vol. 31, no. 2, pp. 627-636, June 2016, doi: 10.1109/TEC.2015.2496968.

10. R. M. Pindoriya, A. K. Mishra, B. S. Rajpurohit and R. Kumar, "An Analysis of Vibration and Acoustic Noise of BLDC Motor Drive," 2018 IEEE Power & Energy Society General Meeting (PESGM), 2018, pp. 1-5, doi: 10.1109/PESGM.2018.8585750.

11. H. Suwoyo, Y. Tian, C. Deng and A. Adriansyah, "Improving a Wall-Following Robot Performance with a PID-Genetic Algorithm Controller," 2018 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), 2018, pp. 314-318, doi: 10.1109/EECSI.2018.8752907.

12. A. Géczy, R. De Jorge Melgar, A. Bonyár and G. Harsányi, "Passenger detection in cars with small form-factor IR sensors (Grid-eye)," 2020 IEEE 8th Electronics System-Integration Technology Conference (ESTC), 2020, pp. 1-6, doi: 10.1109/ESTC48849.2020.9229693.

13. Javed Mehedi Shamrat, F.M., Ghosh, P., Mahmud, I., Nobel, N.I., Dipu Sultan, M. (2021). An Intelligent Embedded AC Automation Model with Temperature Prediction and Human Detection. In: Hassanien, A.E., Bhattacharyya, S., Chakrabati, S., Bhattacharya, A., Dutta, S. (eds) Emerging Technologies in Data Mining and Information Security. Advances in Intelligent Systems and Computing, vol 1286. Springer, Singapore.

14. J. N. Amrutha and K. R. Rekha, “Night Vision Security Patrolling Robot Using Raspberry Pi”, IJRESM, vol. 3, no. 8, pp. 432–436, Aug. 2020.