VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi, Karnataka- 590 018

Project Report

On

Smart Traffic Control System for Emergency Vehicles

Submitted in partial fulfillment of the requirements for the reward of the degree of

Bachelor of Engineering

in

Electronics & Communication

Submitted by

H P Jeevan	**1BI18EC051**
G Rohith	**1BI18EC045**
Emyl Varghese George	**1BI18EC044**

Under the Guidance of

Radha B L

Associate Professor, Dept. of ECE

BIT, Bengaluru

Department of Electronics & Communication Engineering

BANGALORE INSTITUTE OF TECHNOLOGY

K. R. Road, V.V Puram, Bengaluru- 560004

2021-2022

BANGALORE INSTITUTE OF TECHNOLOGY

K.R. Road, V. V Puram, Bengaluru -560004
Phone: 26613237/26615865, Fax: 22426796
www.bit-bangalore.edu.in

Department of Electronics and Communication Engineering

CERTIFICATE

Certified that the project work entitled **"Smart Traffic Control System for Emergency Vehicles"** by **H P JEEVAN (USN: 1BI18EC051), G ROHITH (USN: 1BI18EC045), and EMYL VARGHESE GEORGE (USN: 1BI18EC044)**, bonafide students of Bangalore Institute Of Technology in partial fulfilment for the award of Bachelor of Engineering in Electronics and Communication Engineering of the Visvesvaraya Technological University, Belagavi during the year 2021- 2022. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library. The project report has been approved as it satisfies the academic requirements in respect of Project Work prescribed for the said Degree.

Signature of Guide	Signature of HOD	Signature of Principal
RADHA B L Associate Professor, Dept. of ECE, BIT.	**Dr. HEMANTH KUMAR A.R** Professor & HOD, Dept. of ECE, BIT.	**Dr. ASWATH M.U** Principal, BIT.

External Viva

Name of the Examiner	**Signature with date**
1.	
2.	

ABSTRACT

The rise in population has increased the number of automobiles leading to a steep growth in traffic. Traffic congestion has become an inescapable condition in large and growing cities across the world. This makes it difficult for the emergency vehicle to pass through traffic during drastic situations.

The objective of this project is to detect the emergency vehicles and control the traffic system, hence aiding the emergency vehicles in reaching their destination on time. These emergency vehicles considered in this project include ambulances, fire engines, and police cars. The system designed can recognize sirens and adjust the traffic signals accordingly. The system proposed in this project is completely audio-based and uses concepts such as machine learning, filtering, and autocorrelation. The complete system is implemented using Raspberry Pi.

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. All the group members, in their own capacities have contributed in carrying out this project work.

We deeply express our sincere gratitude to our guide **RADHA B L**, Associate Professor, Department of Electronics and Communication Engineering, BIT for the guidance, support, motivation and regular source of encouragement extended to us during the execution of the project.

Our deepest thanks are owed to **Dr. HEMANTH KUMAR A.R.**, HOD, Department of Electronics andCommunication Engineering, BIT for providing us with constant support, guidance and valuable suggestions.

We would also like to thank **Dr. ASWATH M. U.**, Principal, BIT, for his encouragement and moral support.

We would also like to thank **Mr. GAHAN A.V. and Dr. MUKTHI S. L.,** Assistant Professor, B.E. Project Coordinator, Department. of Electronics and Communication Engineering, BIT and all other teaching and non-teaching staff of Electronics and Communication Engineering department who has directly or indirectly helped us in the completion of the Project Work.

We would also like to thank all the staff members of Department of Electronics and Communication Engineering for their constant support and encouragement whenever required.

H P JEEVAN - 1BI18EC051

G ROHITH - 1BI18EC045

EMYL VARGHESE GEORGE - 1BI18EC044

TABLE OF CONTENTS

CHAPTER	CONTENT	PAGE NUMBER
Chapter 1:	**INTRODUCTION**	**1**
	1.1 Overview	1
	1.2 Aim of the project	2
	1.3 Objectives	2
	1.4 Problem statement	2
Chapter 2:	**LITERATURE SURVEY**	**3**
Chapter 3:	**METHODOLOGY**	**5**
	3.1 Generation of dataset	5
	3.2 Neural network	5
	3.3 Spectrogram	6
	3.4 Direction of arrival estimation	7
	3.4.1 Delay estimation	9
	3.5 Filter design	9
	3.6 Traffic signal control	10
	3.7 Summary	10
Chapter 4:	**IMPLEMENTATION AND DESIGN**	**12**
	4.1 Generation of dataset	12
	4.2 Detection of siren	13
	4.3 Spectrogram	15
	4.4 Direction of arrival estimation	15
	4.4.1 Delay calculation	15
	4.4.2 Angle estimation	16
	4.5 Filter	17
	4.6 Components	19
	4.6.1 Hardware Components	19
	4.6.2 Softwares	20
	4.7 Circuit connections	21

Chapter 5: RESULTS **22**

5.1 Detection of siren 22

5.2 Testing on scaled model 22

5.2.1 Model on one-dimensional road 23

5.2.2 Model on one-dimensional road with the direction of arrival 24

5.2.3 Model on T-type traffic signal with the direction of arrival 25

Chapter 6: CONCLUSION **27**

6.1 Advantages 27

6.2 Limitations 27

6.3 Future scope 27

REFERENCES **28**

APPENDIX **30**

Appendix 1 – Pin diagram and Board specifications 30

Appendix 2 – Raspberry Pi 4 specifications 30

LIST OF FIGURES

FIGURE NUMBER	CONTENT	PAGE NUMBER
3.1	Convolutional neural network layers	6
3.2	Audio signal and spectrogram	7
3.3	Position of microphones and source	8
3.4	Procedure to calculate the delay	9
3.5	FSM of traffic system	10
3.6	Block diagram of the system built	11
4.1	Data acquisition and cleaning	13
4.2	CNN architecture	14
4.3	Training and validation accuracy and loss	14
4.4	Steps for spectrogram generation	15
4.5	Autocorrelation of the two signals	16
4.6	Possible x and y values for given AB' and x_B values	17
4.7	Filter design	18
4.8	Filter response	18
4.9	Components	19
4.10	Connections to Raspberry Pi	21
5.1	Scaled model of traffic system	23
5.2	Model used for the case of one-dimensional road	24
5.3	Model used for the case of the one-dimensional road with angle calculation	24
5.4	Model used for the case of T-type traffic signal with angle calculation	25
i	Raspberry Pi 4 model	30
ii	Pin diagram of Raspberry Pi 4	30

LIST OF TABLES

TABLE NUMBER	CONTENT	PAGE NUMBER
4.1	LED connections	21
5.1	Confusion matrix on test data	22
5.2	Results of the model on the one-dimensional road	23
5.3	Results of the model on the one-dimensional road with angle calculation	25
5.4	Results of the model on T-type traffic signal with angle calculation	25

Chapter 1

Introduction

1.1 Overview

Emergency vehicles play an important role in every life-threatening situation. In today's world, it is observed that there is a steady rise in traffic due to the increased use of personal vehicles. Traffic jam takes more than 20% of patient lives in an ambulance but when the patient's condition is very serious the percentage of patient death is increased. Similarly, fire trucks need to address fire accidents immediately but delays are caused due to traffic. The purpose of this project is to develop an automated solution based on siren detection to add to the currently existing traffic system that gives preference to emergency vehicles.

A siren is a special signal sounded by emergency service vehicles such as fire trucks, police cars, and ambulances. Sirens are essential for assisting the safe and rapid arrival of these vehicles to the scene of an emergency. They are signals whose range typically alternates between 820Hz to 1320Hz and 2470Hz to 3950Hz. These frequencies are detected in traffic and hence can be used to determine if a traffic signal needs to be changed to allow for the passage of emergency vehicles. To assist with the detection of sirens, it is important to understand the soundscape of traffic. This soundscape can be roughly separated into three sub-sources of sounds, including siren sounds, vehicle horns generated by ordinary vehicles, and noises.

This project has two main regions of interest: detection of emergency vehicles and determining the direction of arrival. For the detection of emergency vehicles, a convolutional neural network (CNN) is used. A prerequisite to using CNN is to convert the audio data to images which are done via the use of spectrograms. The entire CNN model is then dumped onto a Raspberry PI, which is a microprocessor capable of running complex neural networks.

The direction of arrival of the ambulance is determined using two microphones. A filter is used to suppress the noise and only the necessary frequencies are sent to a program that based on the angle of arrival of sound determines the direction of approach of emergency vehicle.

The two sections are combined to form the final model which is capable of controlling traffic signals if an emergency vehicle is detected. This is achieved by collecting signals from the microphones and then sent to the model. The model checks if the collected audio signal contains a siren and also determines the direction of the approach. Upon detection, the finite state machine present in the model will change the corresponding signals.

1.2 Aim of the Project

To design and implement a smart traffic control system for emergency vehicles.

1.3 Objectives

a) To make a low-cost system that can be implemented in actual traffic signals.
b) To make an efficient detection system for emergency vehicle detection.
c) To reduce the delay due to traffic for emergency vehicles.
d) To make an automated system that does not require constant maintenance.

1.4 Problem Statement

a) The current traffic signal scenario does not focus on helping the passage of emergency vehicles at the traffic signals.
b) The systems that presently exist for emergency vehicle detection are image-based which are not computationally effective.
c) Systems which make use of cloud computing or IOT to detect emergency vehicles are not feasible.
d) This project uses siren signals of the emergency vehicles which is computationally more efficient as compared to image based data.
e) The system designed in this project does not need to be connected to the Internet, therefore it can be implemented remotely

Chapter 2

Literature Survey

Convolutional Neural Network is used for processing images and CNN algorithm, You Only Look Once (YOLO) is used to determine if an ambulance is present in the captured image [1]. A camera module captures a short video clip which is converted into a series of images and these are sent to the YOLO model which determines if the image passed has a truck or not. If the image contains a truck, then the image is passed to the main program which checks and returns '1' if the truck is an ambulance. If an ambulance has been detected then the corresponding signal is turned to green for 15 seconds and then another video capture occurs for the detection cycle to continue. if an ambulance is no longer detected in the image, then the signal resumes its normal cycle. YOLO has been implemented to detect if an ambulance is present in the signal area in [2]. It then checks for the siren to see if the ambulance is in an emergency state. This sound detection process is done by the use of spectrograms. There is a database that contains the fingerprint hashed values of the siren signals. Only if the siren signal received matches the fingerprint in the database will the signal turn green. A new CNN-based ensemble model was proposed by Van-Thuan Tran and Wei-Ho Tsa, called SirenNet which determines if an ambulance is presently based on audio signals. The SirenNet contains 2 sub-streams: WaveNet which works on the raw waveform and MLNet which works with spectrograms in [3].

Ropashree V et al. proposed the use of an android app and GPS neo connected to ESP8266 to control traffic. When an emergency is in progress, the ambulance driver sends a notification via the app to the firebase with its location, and the congestion present on the roads is returned [4]. The system in [5] has three units: a crash detection unit, an ambulance unit, and the main server are proposed. The location of the emergency is sent to the main server which notifies the same coordinates to the ambulance. The traffic signals are controlled using Zigbee based on the location of the ambulance. Karthik B V et al. proposes a system using GPS, Node MCU, AWS, and raspberry pi. The GPS location of the ambulance is being constantly transmitted to the AWS cloud using the Message Query Telemetry Transport protocol. The cloud upon receiving the GPS location checks if the ambulance has crossed the first reference point and changes the signal to green. When the ambulance passes the second reference point then the traffic light is turned back to its normal sequential flow [6]. The pre-emption algorithm is used coupled with Timed Petri

Nets (TPN) which are a particular kind of bipartite directed graphs populated by places, transitions, and directed arcs between them, each of which can be associated with a deterministic firing time. The TPN allows for the pre-emption system to control the traffic based on two sensors: sensor-in and sensor-out. When the sensor-in signal is detected then the vehicles queued are allowed to pass and once the ambulance has passed the sensor-out signal is received and the traffic model returns to its previous state [7].

The complete implementation of the project requires the knowledge of direction of approaching emergency vehicles to control traffic signal. Carlos Fernández Scola and María Dolores Bolaños Ortega used two microphones which are used to determine the direction in which sound is arriving [8]. Prajoy Podder et al. conducted a study on the types of filters and the way they suppress noise. Chebyshev Filters which have rapid transitions from passband to stopband have been selected to use as the noise suppression filters [9].

Chapter 3

Methodology

3.1 Generation of Dataset

In this section, the generation of the audio dataset is presented. The major issues with collecting the audio dataset manually are obtaining a large number of audio samples and to get a diverse dataset. A simple solution to this problem is to create the audio signals by combining the individual sounds present in the environment. This process is known as Synthetic data generation.

The traffic signal environment mainly consists of noise emanating from different vehicles such as horns, exhaust noise, and engine noise. These noises are randomly combined with emergency vehicle sirens to mimic the real-world scenario.

3.2 Neural Network

An artificial neural network learning algorithm, or neural net, is a computational learning system that uses a network of functions to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. Neural networks work better for prediction-based systems than other machine learning algorithms such as linear regression models which use only input and output nodes to make predictions. These algorithms cannot be used for complex tasks like image classification. The neural network uses the hidden layers which makes its predictions more accurate and applicable to difficult tasks.

A Convolutional Neural Network (CNN) is a Deep Learning algorithm for processing data that has an input shape like a 2D or 3D matrix such as images. Convolutional neural networks use relatively little pre-processing compared to other image classification algorithms. This means that the network learns to optimize the filters (or kernels) through automated learning. This independence from prior knowledge and human intervention in feature extraction is a major advantage of using CNN.

The CNN consists of 3 types of layers where each layer performs a unique operation. The Convolution Layer is responsible for capturing features at higher and lower levels like edges, color, gradient orientation, etc. The pooling layer is responsible for reducing the spatial size and still keeping all the information intact. This is used to decrease the computational power required to process the data through dimensionality reduction. A Fully-Connected layer is an easier way of learning non-linear combinations of high-level features. These layers are shown in Figure 3.1.

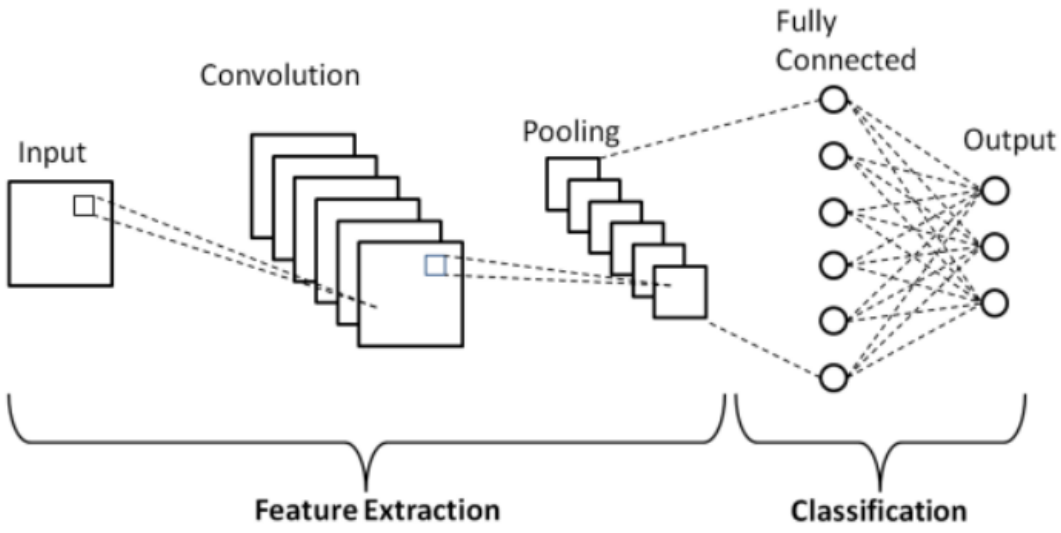


Fig. 3.1 Convolutional neural network layers

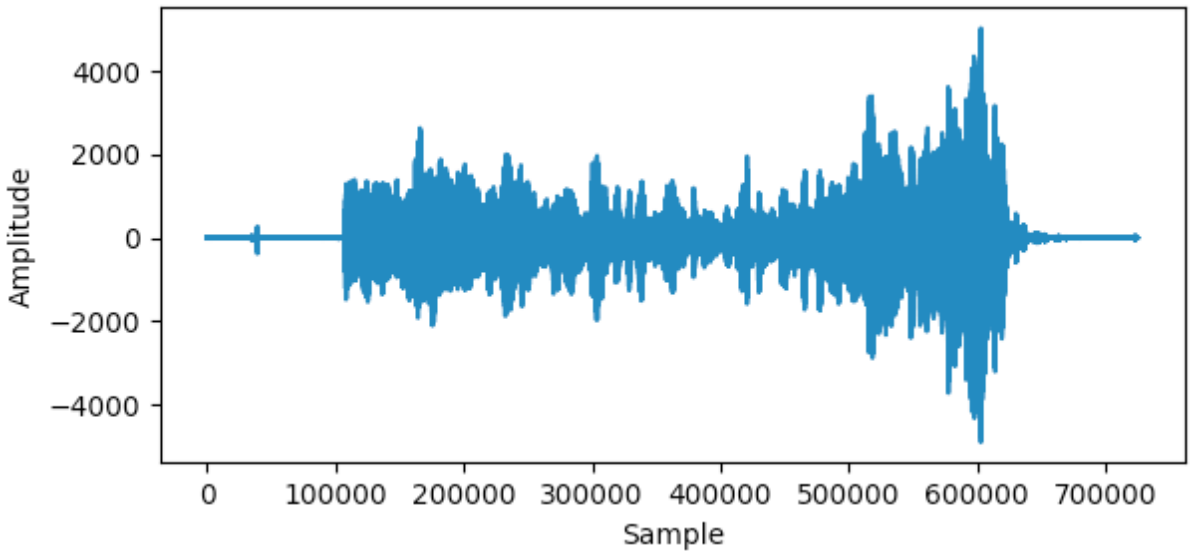
3.3 Spectrogram

A spectrogram is a visual representation of the spectrum of frequencies of a signal as it varies with time in a particular waveform. A spectrogram chops up the signal into smaller time segments and then applies the Fourier Transform to each segment. It then combines the Fourier Transforms for all those segments into a single plot.

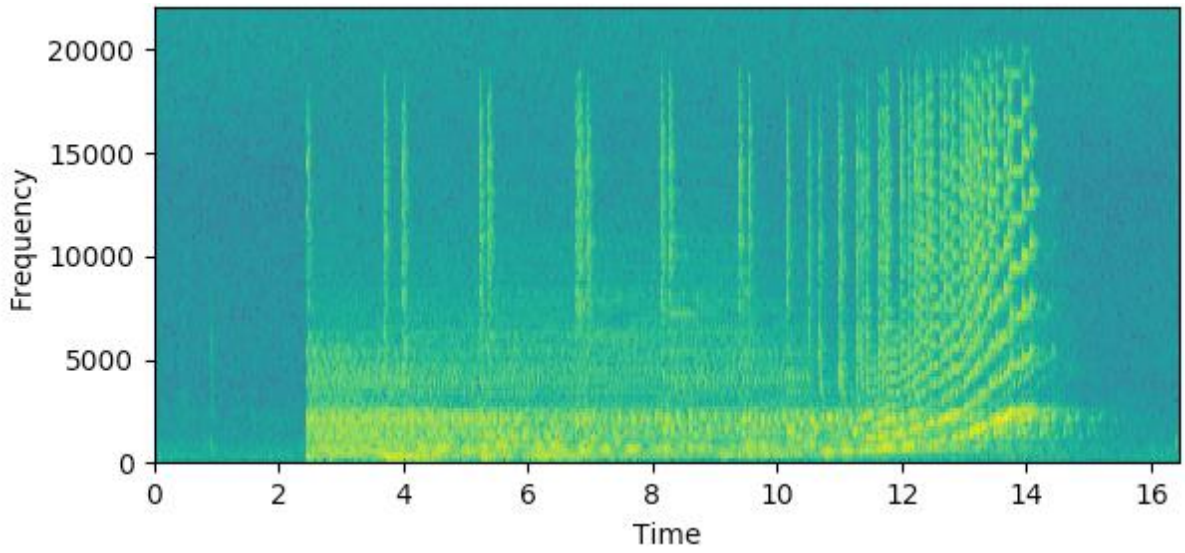
As spectrograms provide a better representation of the audio data, they are more suitable for neural networks. Spectrograms even help in data reduction and frequency extraction of the audio signal. As spectrograms are grayscale images they can be used as input to a convolutional neural network.

Figure 3.2(b) shows the spectrogram of the time domain audio signal depicted in Figure 3.2(a). The x-axis represents the time; the y-axis represents the frequency and the

color represents the amplitude of each frequency in a time interval. Spectrograms are commonly used to display frequencies of sound waves as recorded by microphones.



(a)



(b)

Fig. 3.2 (a) Audio signal and (b) Spectrogram

3.4 Direction Of Arrival Estimation

The direction of a sound source can be estimated as shown in the paper [8]. Two microphones are placed at positions A and B at a distance d apart from each other as shown

in Figure 3.3. Point M represents the source of audio and α' represents the angle between the audio source and the traffic junction. Let B' be the point at which the wavefront is the same as at Point B at any given time t on the line MA. This forms an isosceles triangle MBB' where MB = MB'.

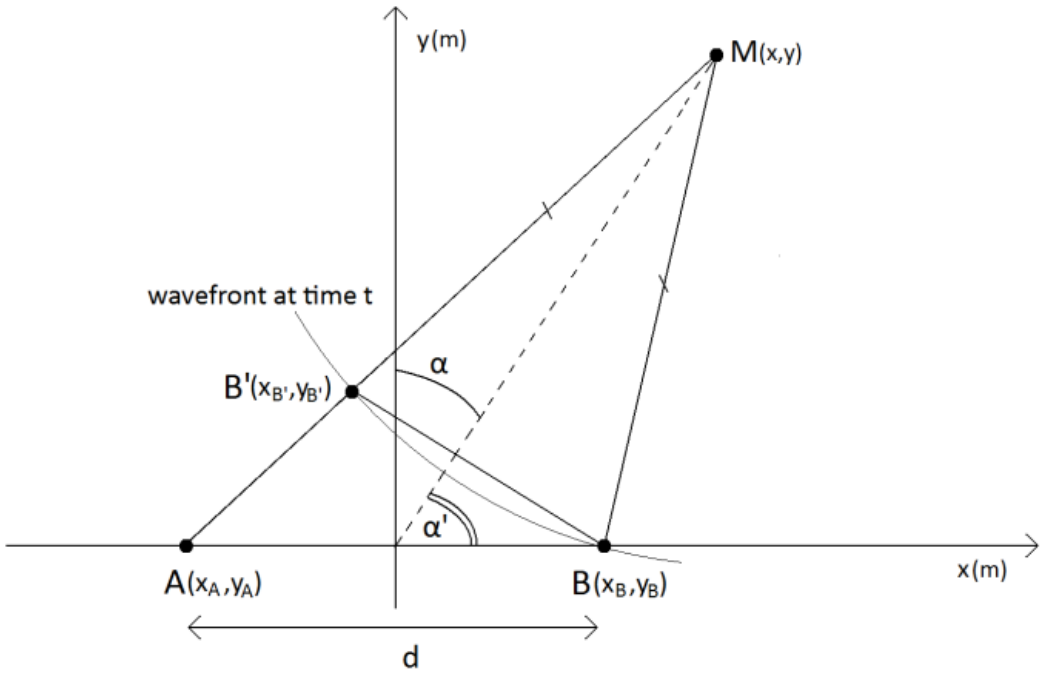


Fig. 3.3 Diagram of the position of microphones and source

The position of M can be described by the equation

$$y^2 = \frac{AB'^2}{4} - {x_B}^2 + x^2\left(\frac{4*{x_B}^2}{AB'^2} - 1\right) \quad (3.1)$$

In equation 3.1, the only variables are y and x. The value is always constant since it represents the position of the microphones, which can be seen as reference points. Moreover, even if the direction varies, the length of AB' remains unchanged. So the equation represents all the possible positions of M, given a certain delay. Considering that the signal travels at the speed of sound c and has a delay of τ, the distance AB' is given by

$$AB' = c * \tau \quad (3.2)$$

The angle α' can be calculated using the equation.

$$\alpha' = tan^{-1}\left(\frac{dy}{dx}\right) \quad (3.3)$$

3.4.1 Delay Estimation

Fig. 3.4 Procedure to calculate the delay

The algorithm that is used for delay calculation is shown in Figure 3.4. The simplest way to calculate time delay (τ) between the audio heard at positions A and B is by using auto-correlation. Autocorrelation is a mathematical representation of the degree of similarity between a given time series and a lagged version of itself over successive time intervals. The time delay can be calculated using the index of max similarity and sampling frequency using the following equation

$$\tau = \text{Lag\_index} / F_s \quad (3.4)$$

3.5 Filter Design

Filtering is a process that can either completely or partially suppress some of the frequencies in the signal. The signal received from the microphones generally has internal and external noise. The internal noise is mainly due to the electrical components. The external noise constitutes horns, engine sounds, etc present in the environment. Removing the noise from the siren is important to get the autocorrelation and direction of the siren precisely. Siren sounds contain only two bands of frequencies. Therefore, two bandpass filters can be used to get the two frequency bands. Two Chebyshev type-1 IIR bandpass filters can be used as filters. Chebyshev type-1 filters provide less passband ripple and order. Due to this the number of coefficients is reduced and the computation cost is minimized.

The most popular tool used for designing filters is the *Filter Designer* app of MATLAB. It is an interactive application that eases the work of filter design. It provides magnitude and phase response and can save the coefficients of the filter in the format of SOS (Second-order sections) and gain directly.

3.6 Traffic Signal Control

Traffic control can be seen as a finite state machine as shown in Figure 3.5. The traffic signal at any given time is present in one of the two states; Ready or Blocking state. In the ready state, the system will continuously check for sirens in the environment to know the presence of an emergency vehicle. If a siren is not detected then the system increases the counter. If a siren is detected the system will shift from the ready state to the blocking state. In the ready state, the traffic lights are controlled based on the value of the counter.

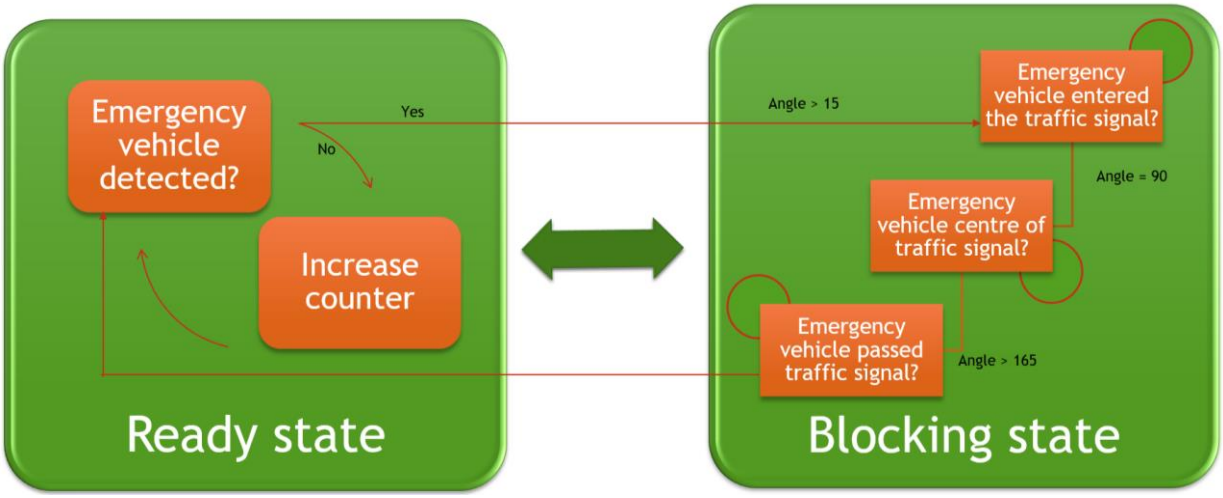


Fig. 3.5 FSM of traffic system

In the blocking state, the direction of the siren is used to switch between substates. Once the direction of the emergency vehicle is known the traffic lights are changed to let the emergency vehicle pass. We monitor the angle to check the position of the emergency vehicle in the traffic signal. The three sub-states represent the entry, center, and passing of the emergency vehicle. Once the emergency vehicle has passed the traffic signal area the traffic lights are switched and move to the ready state.

3.7 Summary

Figure 3.6 summarises the complete flow of the project. The first step is to get the siren which is collected by the microphones connected to the Raspberry Pi. This audio data is cleaned and then processed in two ways, one for detection of the emergency vehicle and the other for localization of the emergency vehicle. For detection of the siren, the audio

signal is converted to a spectrogram and a neural network model is developed as described in sections 3.3 and 3.2. For localization of the siren, the signal is filtered as described in section 3.6 and then the filtered signal is used to find the direction of the siren. The final step is to control the traffic signal using a state machine as described in section 3.7.

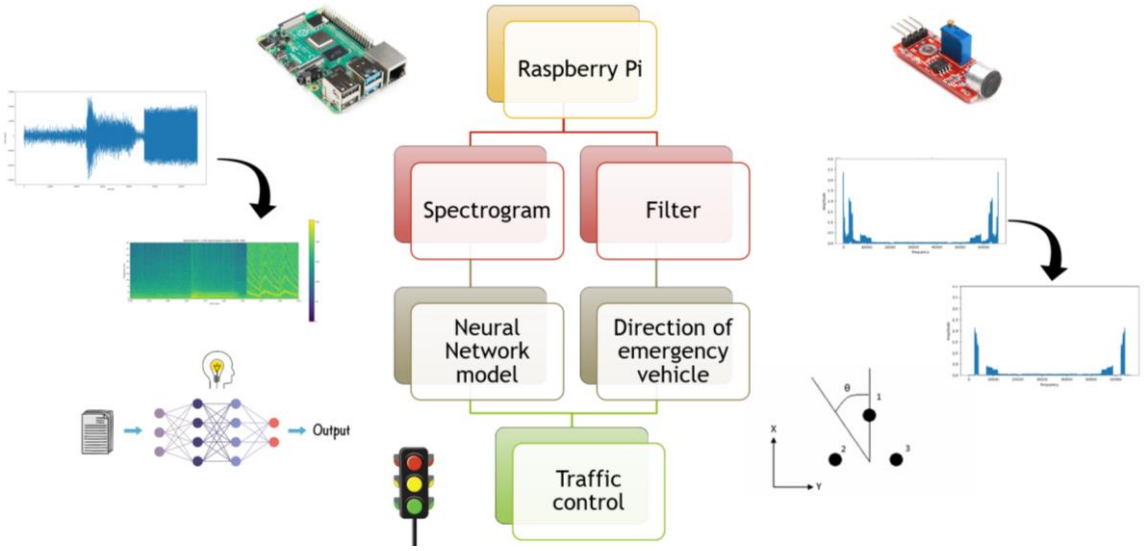


Fig. 3.6 Block diagram of the system built

Chapter 4

Implementation And Design

4.1 Generation of Dataset

The audio signals present in the traffic signal environment are collected. This included audio of car and motorcycle horns, noise from the engine, exhaust, and other random noise. Along with these audio samples, emergency vehicle sirens are collected which include two fire trucks, three ambulances, and three police van sirens.

The sampling frequency of all the audio signals is changed to 22050Hz. This sampling frequency is found using the Nyquist criteria as shown below:

$$F_{max} = 2450Hz$$

$$F_s \geq 2 * F_{max} \quad (4.1)$$

$$F_s \geq 4900Hz$$

A greater sampling frequency is taken than the minimum requirement to get higher precision while calculating the delay to find the direction of an emergency vehicle which is discussed in section 4.4.

To perform mathematical operations such as element-wise addition and multiplication on the audio signals, it is necessary to have the number of bits per sample be the same for all the audio signals. Therefore, we are setting the number of bits per sample to 16-bits (2-byte integer).

The audio signals are combined in six ways to create all the possible combinations that can be observed in the traffic signal environment.

a) Noise
b) Siren (emergency vehicle approaching the traffic signal) + noise
c) Siren + noise
d) Siren (emergency vehicle approaching the traffic signal) + noise + car horns
e) Siren + noise + car horns
f) Noise + car horns

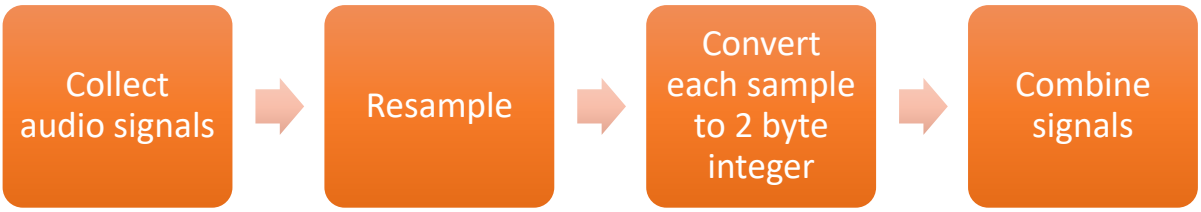


Fig. 4.1 Data acquisition and cleaning

The signals that are generated are divided into two classes i)with siren (signals 2, 3, 4, 5) and ii)without siren (signals 1, 6). Figure 4.1 shows the complete procedure of data acquisition and data cleaning of the audio signals. These signals are then clipped to random 3 seconds intervals. A total of 3000 signals are generated using the above method. Spectrograms of these generated signals are created as mentioned in section 4.3. These spectrograms are split into train, test, and validation datasets in the ratio of 75%, 10%, and 15% which contain both types of signals; with and without sirens.

4.2 Detection of Siren

The CNN architecture adopted to detect the siren sounds in the spectrogram is shown in Figure 4.2. The model contains four convolutional layers, three max-pooling layers of kernel size 3X3, and two fully connected layers.

ReLu activation function is used at each layer as it is faster to compute. Sigmoid activation is used for the last layer for binary classification. The Adam optimizer is used to train the neural network with beta1 and beta2 values of 0.9 and 0.999. Adam optimizer is chosen over other optimizers because it works well with noisy data and has a faster convergence rate. The learning rate is set to $4*10^{-3}$. Since the problem at hand is a binary classification problem (classify spectrograms with and without siren) Binary Cross-entropy is used as the loss function.

The model was trained for 300 epochs with a batch size of 32. The total number of parameters trained (weights and bias) is 7241. A total of 1950 spectrograms are used to train the model and 750 spectrograms are used to validate the trained model. The training and validation accuracies obtained are 98.8% and 97.2% respectively. Figure 4.3 shows the accuracy and loss at different epochs while training.

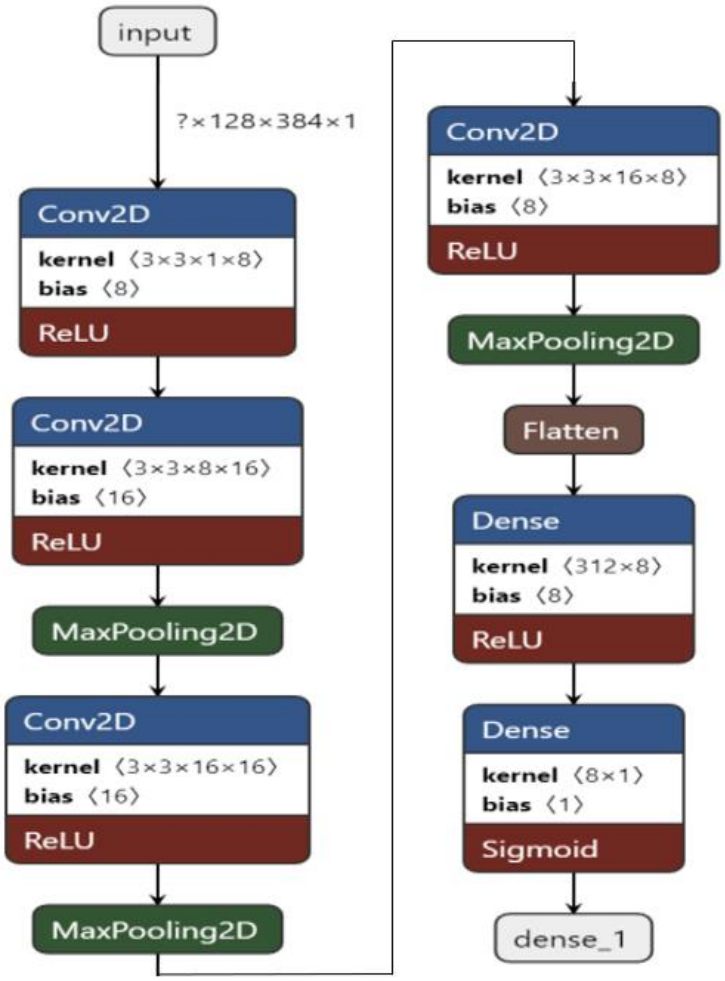
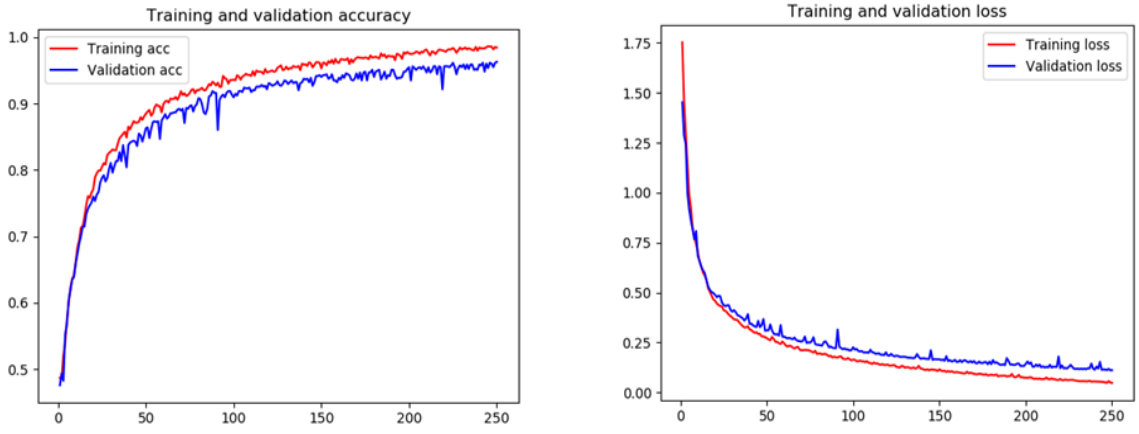


Fig. 4.2 CNN Architecture



(a) (b)

Fig. 4.3 Training and validation (a) accuracy and (b) loss

4.3 Spectrogram

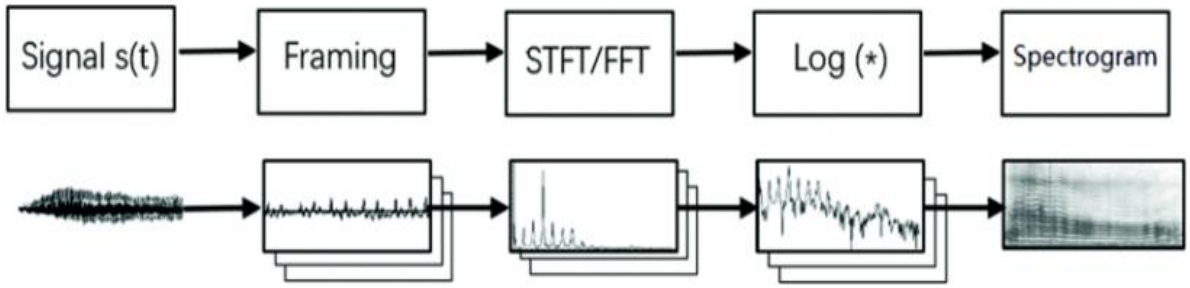


Fig. 4.4 Steps for spectrogram generation

Figure 4.4 shows the steps involved in the generation of a spectrogram. It takes in an audio signal as input and provides a spectrogram as the output. The steps are described in detail below:

1. Framing: The audio signal in the time domain is broken down into chunks of the same length. This is called frame length and for this project, the frame length is chosen to be 256.
2. FFT: A N-point FFT is performed on each of these frames to calculate the magnitude of the frequency spectrum.
3. Log (*): Element-wise log operation is applied to each of the FFTs calculated to get spectrums as given below

$$Y = 10 * \log_{10}(x) \quad (4.2)$$

   where, x = input FFT

4. Stacking: Each spectrum is stacked side by side to form the spectrogram.

4.4 Direction of Arrival Estimation

The process of finding the direction of the siren can be explained in 2 parts.

4.4.1 Delay Calculation

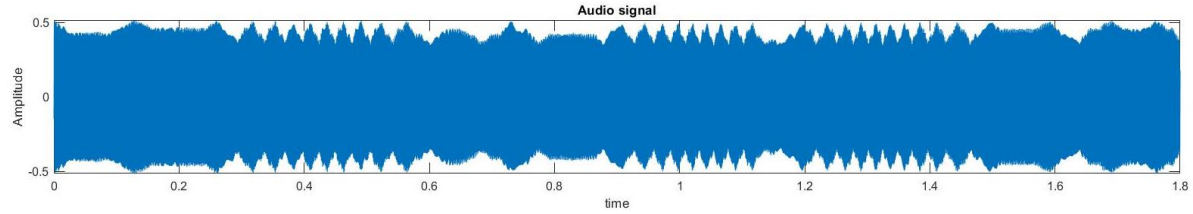
As mentioned in section 3.5.1, equation 3.4 can be used to calculate the delay between the 2 audio signals. For example, consider the audio signal and its time-shifted signal as shown

in Figure 4.5 (a) and (b). Figure 4.5(c) shows the autocorrelation of the two signals. The maximum value of similarity is seen at the lag of 2205. Delay is calculated as

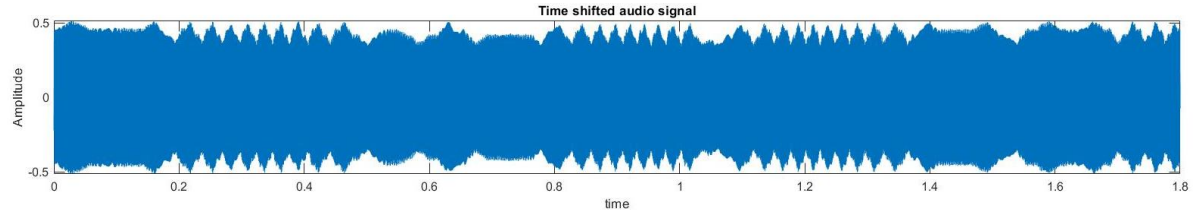
$$\tau = \text{Lag\_index} / F_s$$

$$\tau = 2205/22050 = 0.1 \text{ sec}$$

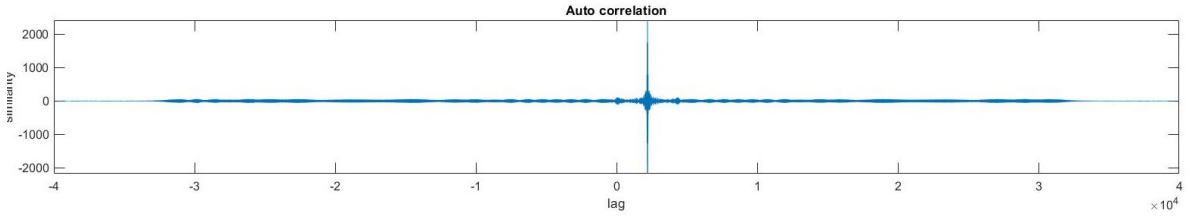
Therefore, we can conclude that the first signal is 0.1 seconds ahead of the second signal.



(a)



(b)



(c)

Fig. 4.5 a) Audio signal, b) time-shifted signal, and c) autocorrelation of the two signals

4.4.2 Angle Calculation

The direction of a siren or the angle between the emergency vehicle and traffic junction can be calculated using Equations 3.1 to 3.3. First, we need to define the parameters x_B and AB' mentioned in section 3.5. Let the distance between the two microphones be 10cm. Therefore, we get the value of x_B to be 5cm [$x_B = d/2$]. The delay is in the range of 40µs to 500µs. Using the time delay value and equation 3.2 we can calculate the value of AB'.

From equation 3.1 we can plot the possible sets of values of x, and y for the calculated AB' and x_B values. This is shown in Figure 4.6.

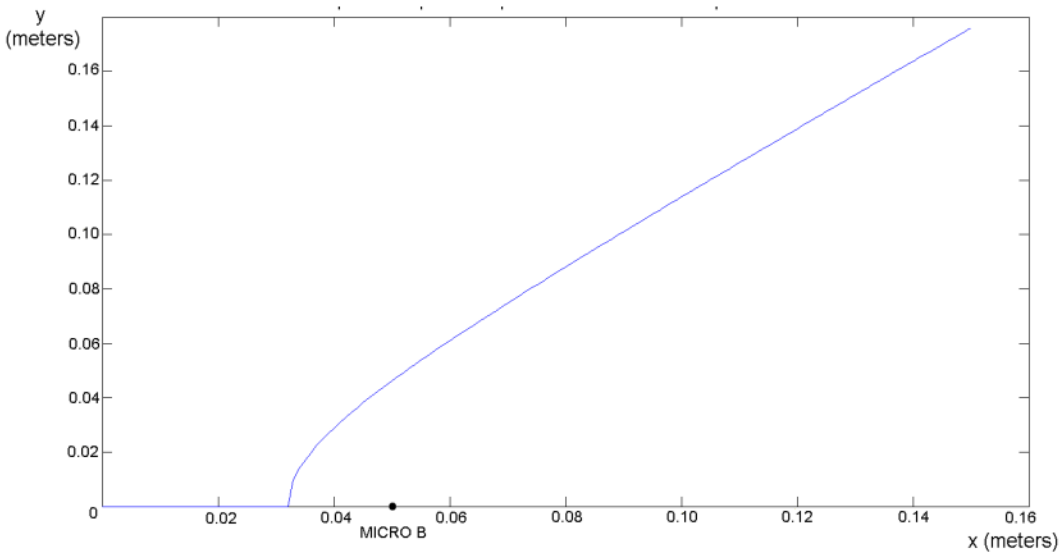


Fig. 4.6 Possible x and y values for given AB' and x_B values

The slope of the curve can be calculated using the equation

$$\frac{dy}{dx} = \frac{y_2 - y_1}{x_2 - x_1} \tag{4.2}$$

Let, $$\frac{dy}{dx} = 1$$

Then, using the equation 3.4 we can calculate the angle α' as

$$\alpha' = \tan^{-1}(1) = 45°$$

4.5 Filter

The time domain and the frequency domain representation of the siren signal corrupted with noise are shown in Figure 4.7(a) and (b). The frequency bands of interest are 820Hz to 1320Hz and 2470Hz to 3950Hz. It can be seen from Figure 4.7(b) that noise occupies frequency ranges that are outside the range of frequencies of sirens mentioned above.

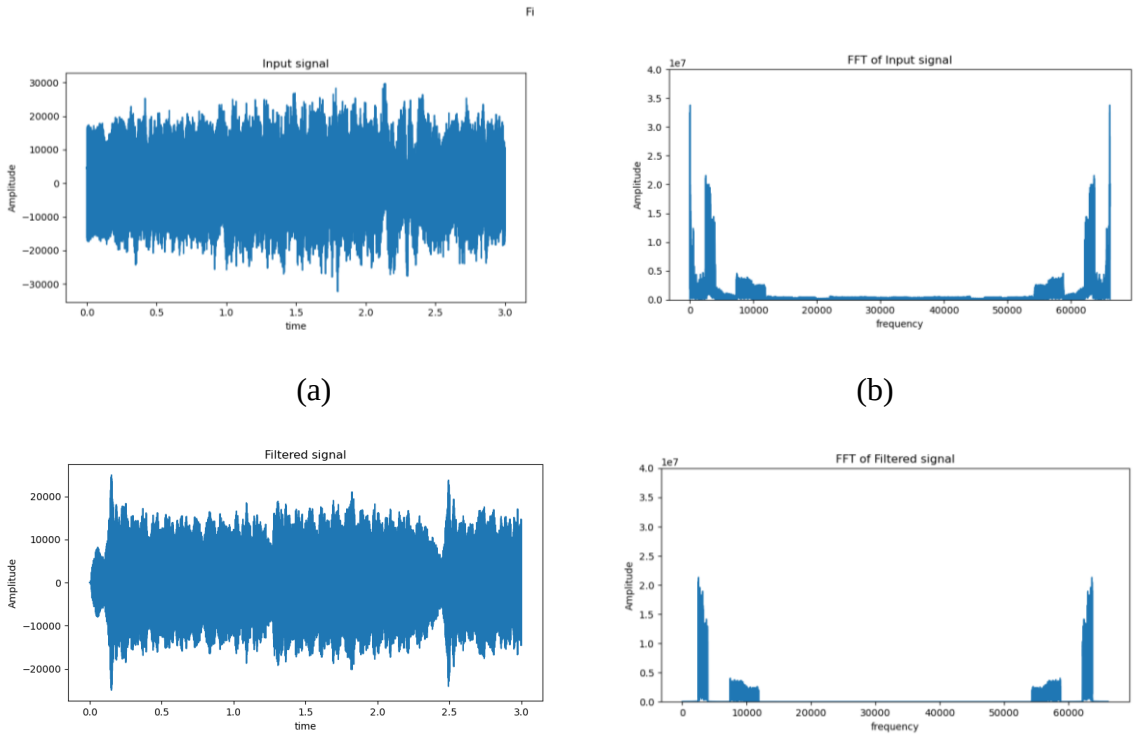


Fig. 4.7 Filter design: a) input signal b) FFT of the input signal c) filtered signal and d) FFT of the filtered signal

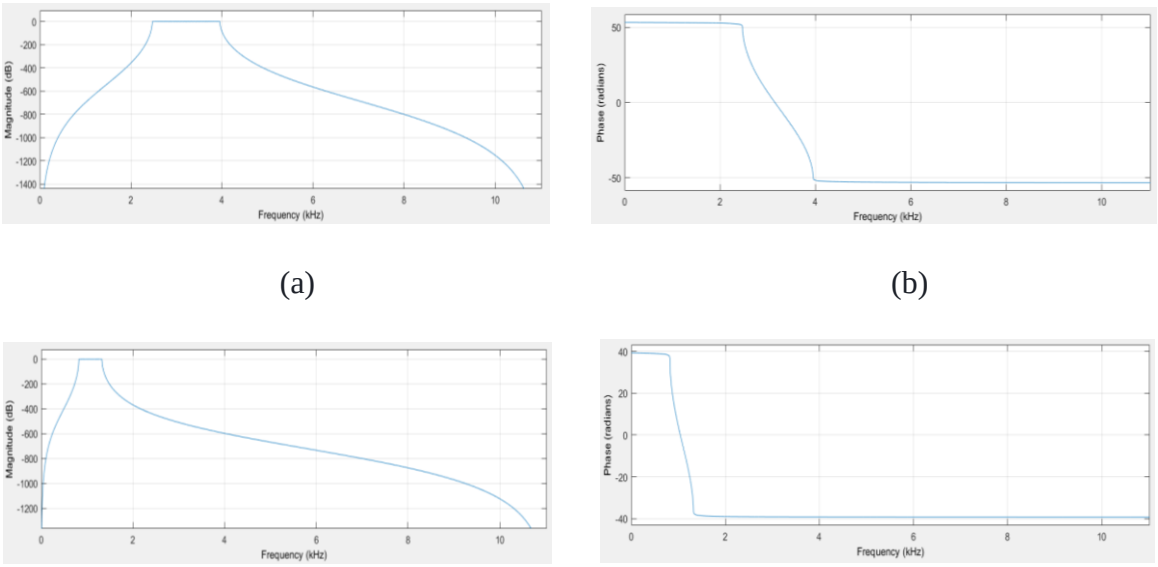


Fig. 4.8 Filter Response: a) Magnitude response of Bandpass filter I b) Phase response of Bandpass filter I c) Magnitude response of Bandpass filter II and d) Phase response of Bandpass filter II

Therefore, to remove the noise we have designed two Chebyshev Type-1 bandpass filters where the parameters for each of these bandpass filters are defined as follows:

- Bandpass filter I:

  F_{LP} = 820Hz, F_{LS} = 770Hz, F_{HP} = 1320Hz, F_{HS} = 1350Hz

- Bandpass filter II:

  F_{LP} = 2470Hz, F_{LS} = 2440Hz, F_{HP} = 3950Hz, F_{HS} = 4000Hz

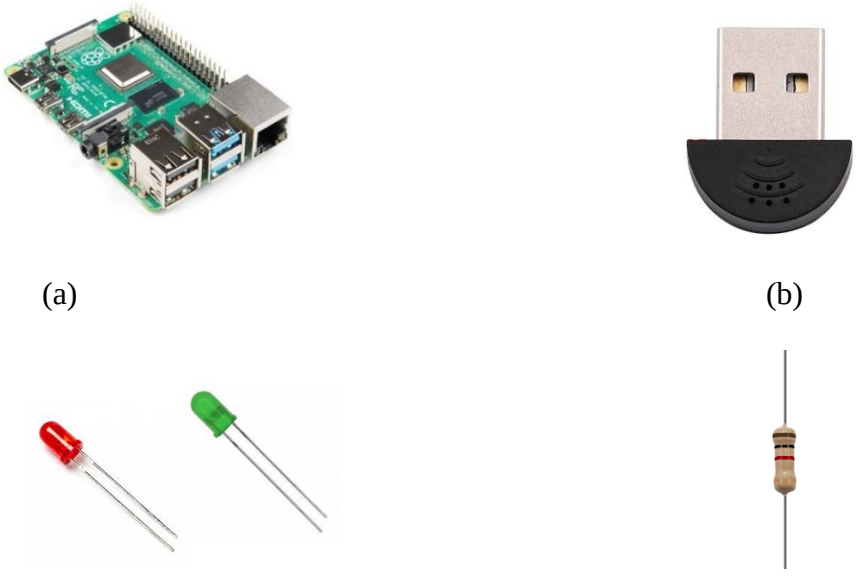
The time domain and the frequency domain representation of the filtered signal are shown in Figure 4.7(a) and (b). The magnitude and phase response of both the band pass filters are shown in Figure 4.8.

4.6 Components

4.6.1 Hardware Components

The hardware components used in this project are USB microphones, LEDs, resistors, and Raspberry Pi.

The USB microphone has been used to record the sirens. It has a sound sensitivity of 67dB and a frequency response of 100Hz to 16kHz which is suitable for this project since the frequencies of the sirens dealt with in this project lie in this range.



(a) (b)

(c) (d)

Fig. 4.9 Components a) Raspberry Pi 4 b) USB microphone c) LEDs and d) 1kΩ resistor

A total of eight LEDs have been used (four red LEDs and four green LEDs) which are used to mimic the traffic signals of the real world. The forward voltage rating of LEDs depends on the silicon elements and dopants used. The operating voltage of red LED ranges from 1.7 to 2.0 volts and for green LEDs in the range of 2.2 to 3.3 V when conducting. The LEDs have a current rating of 20mA.

For each of the LEDs, a resistor is used to limit the current flow to the LED. The resistance value considered is 1kΩ. This value has been chosen so that the current rating of the LEDs is not violated.

All the hardware components and sensors have been connected to the Raspberry Pi. The version used in this project is Raspberry Pi 4 Model B. It consists of a 64-bit quad-core Cortex-A72 processor and 4 gigabytes of RAM. It has 4 USB ports which have been used to connect the USB microphones. The total number of GPIO pins available is 40 out of which 28 pins can be used to connect the LEDs.

4.6.1 Softwares

The softwares that have been used in the design of the project are Python and MATLAB. Python is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. It consists of a wide range of libraries which can be used for different applications. In this project Python has been used to design the machine learning model used to detect the emergency vehicles. To filter noise present in the signals the Filter Designer app of MATLAB was used. MATLAB is a proprietary multi-paradigm programming language and numeric computing environment developed by MathWorks. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages.

To facilitate communication between the computer and Raspberry Pi PuTTy and VNC viewer have been used. These softwares establish communication using the SSH protocol. PuTTY is a free and open-source terminal emulator, serial console and network file transfer application. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. PuTTy allows for command line control of Raspberry Pi. For GUI based control of Raspberry Pi, VNC viewer has been used.

4.7 Circuit Connections

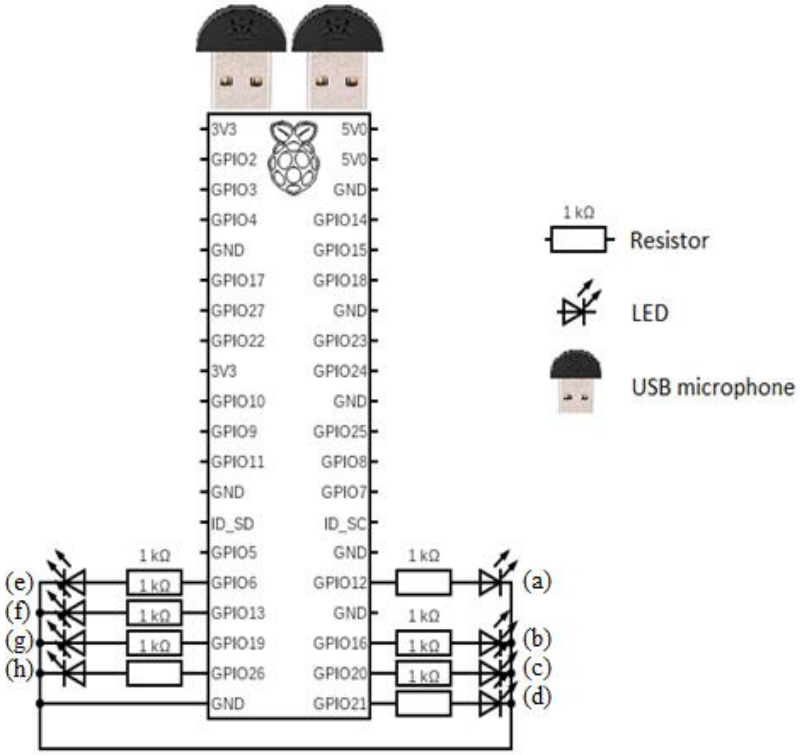


Fig. 4.10 Connections to Raspberry Pi

The connections made between the hardware components are shown in Figure 4.10. The two USB microphones have been connected to the USB 2.0 ports available on the Raspberry Pi. Each of these has been connected using USB – USB cables so that they can record sirens from different directions. The red LEDs have been connected to the GPIO pins of values 6, 13, 19, and 26 and the pins 12, 16, 20, and 21 have been used to connect the green LEDs. The resistors are placed in series with LEDs. All the LEDs are grounded by connecting all their negative terminals to a single ground pin on the Raspberry Pi.

Table 4.1 LED connection

Pin no.	Green LED no.	Pin no.	Red LED no.
12	(a)	6	(e)
16	(b)	13	(f)
20	(c)	19	(g)
21	(d)	26	(h)

Chapter 5

Results

5.1 Detection of Siren

The neural network model built was tested on both spectrograms and continuous audio signals received from the microphone. The model was first tested on 300 randomly generated spectrograms which consisted of 60% of signals with siren and 40% without siren sound. The confusion matrix is shown in table 5.1.

Table 5.1: Confusion matrix on test data

	Predicted NO	**Predicted YES**
Actual NO	119 (TN)	1 (FP)
Actual YES	8 (FN)	172 (TP)

The accuracy of this data can be calculated using the equation

$$\text{accuracy} = (TP + TN)/ (TP + TN + FP + FN) \quad (5.1)$$

$$\text{accuracy} = (172+119) / 300 = 0.97 \text{ or } 97\%$$

Ten different continuous audio signals each of 50 seconds were taken from the traffic environment for testing the neural net, out of which 6 signals contained a siren sound and 4 did not have a siren. When tested on these continuous audio signals the accuracy is found to be 96.2% .

5.2 Testing on Scaled Model

Three experimental tests are carried out on the hardware model built. The model built is a 1:50 scaled version of the traffic signal environment. The three experiments that are carried out are explained in the following subsections. The data extracted from these tests are the time taken to detect the siren, execution time, and success of the trial. From this data, the time taken for the emergency vehicle to enter and leave the traffic signal is calculated. The success of the trial can take three possible values. The value “1” means that the emergency vehicle was detected when present in the traffic signal region. The value “0.5” means that

the emergency vehicle was detected when entering the traffic but not while leaving the region. The value "0" means that the emergency vehicle was not detected when present in the traffic signal region. Figure 5.1 shows the setup of the test conducted.

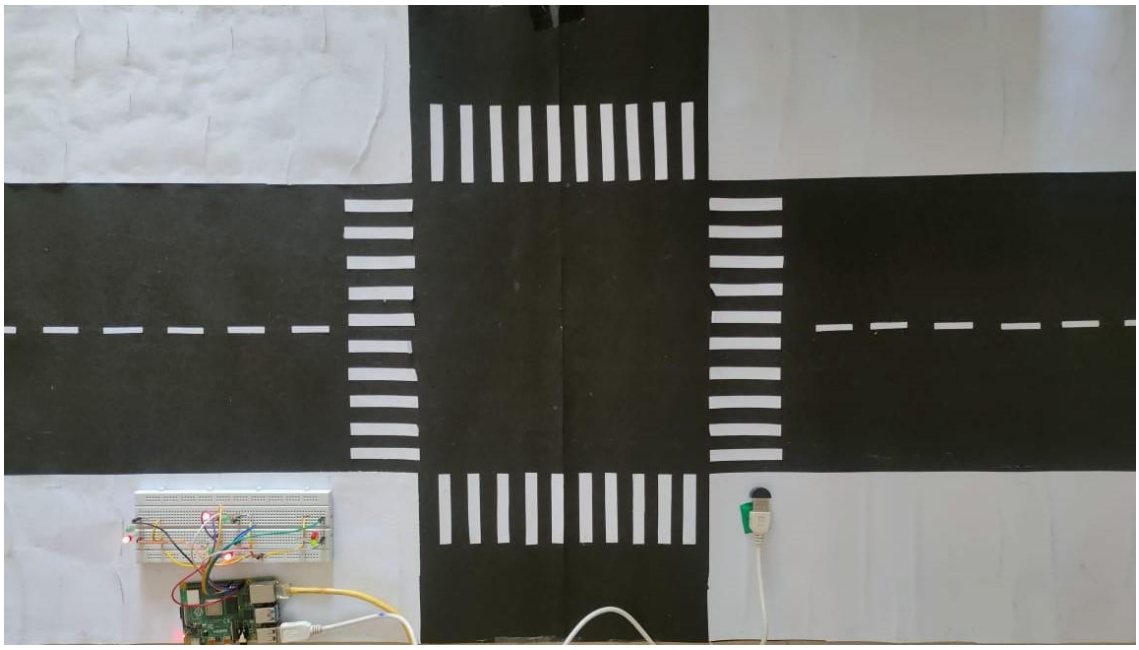


Fig. 5.1 Scaled model of traffic system

5.2.1 Model on One-Dimensional Road

The first experiment conducted consisted of only one road with one microphone as shown in Figure 5.2. The model was tested for 30 trials and the results are shown in Table 5.2.

Table 5.2: Results of the model on the one-dimensional road

	Detection time (sec)	**Execution time (sec)**	**Time for which emergency vehicle is at the traffic signal (sec)**	**Success rate (%)**
Average	29.32	57.42	28.11	88
Standard deviation	7.75	14.73	10.53	-

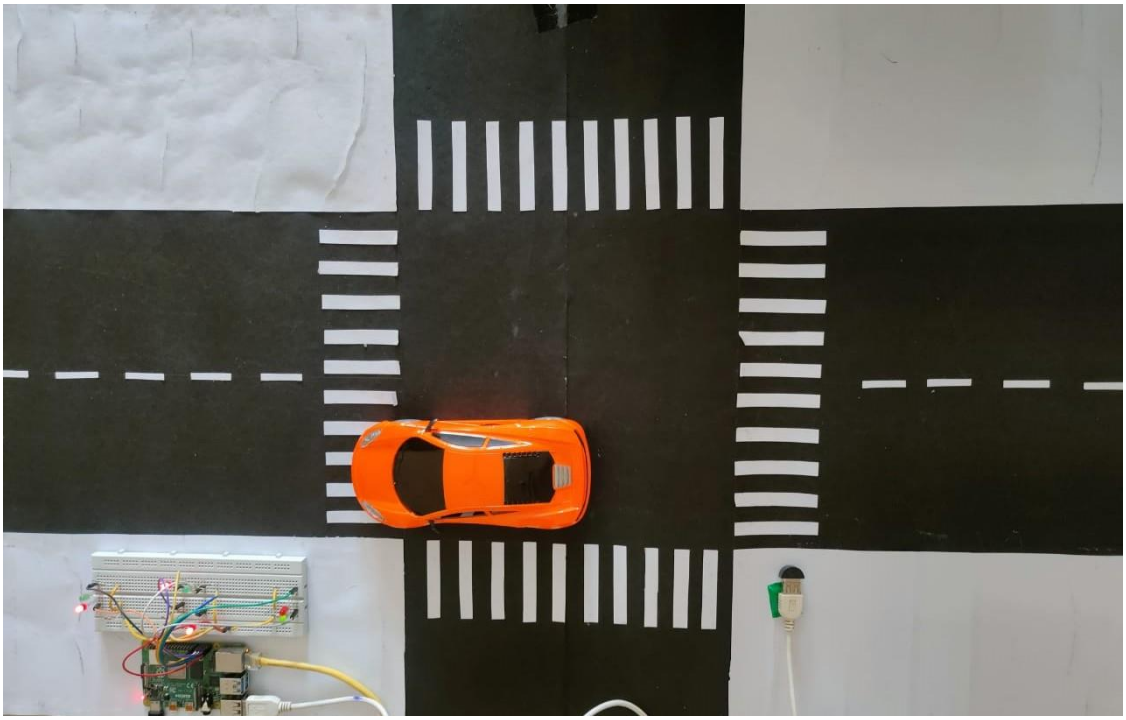


Fig. 5.2 Model used for the case of one-dimensional road

5.2.2 Model on One-Dimensional Road with the Direction of Arrival

The experiment conducted consisted of only one road with two microphones as shown in Figure 5.3. The model was tested for 30 trials and the results are shown in Table 5.3.

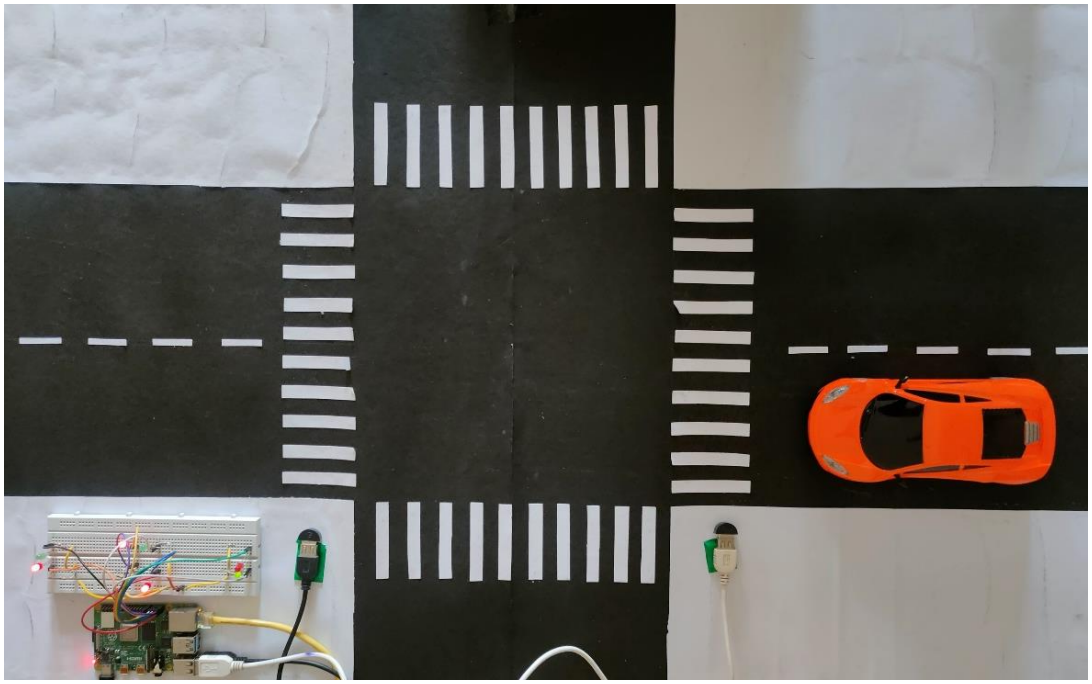


Fig. 5.3 Model used for the case of the one-dimensional road with angle

Table 5.3: Results of the model on the one-dimensional road with angle calculation

	Detection time (sec)	**Execution time (sec)**	**Time for which emergency vehicle is at the traffic signal (sec)**	**Success rate (%)**
Average	36.94	66.38	30.06	84
Standard deviation	8.84	13.43	11.61	-

5.2.3 Model on T-Type Traffic Signal with the Direction of Arrival

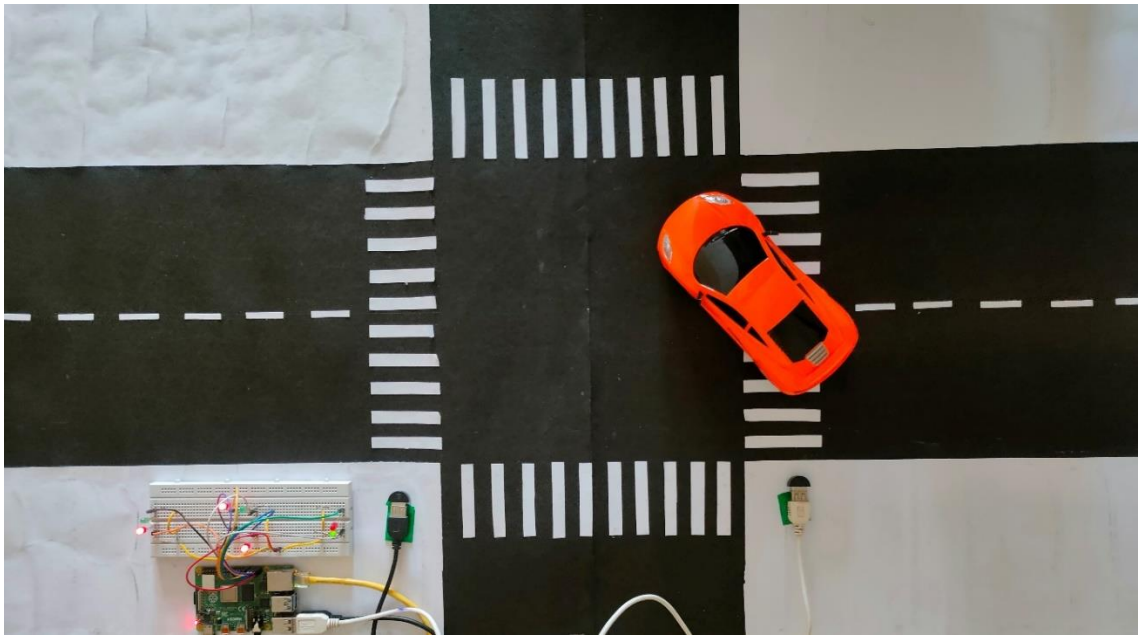


Fig. 5.4 Model used for the case of T-type traffic signal with angle calculation

Table 5.4: Results of the model on T-type traffic signal with angle calculation

	Detection time (sec)	**Execution time (sec)**	**Time for which emergency vehicle is at the traffic signal (sec)**	**Success rate (%)**
Average	40.78	67.82	30.12	84
Standard deviation	10.34	12.13	15.25	-

The experiment conducted consisted of a T-type traffic junction with two microphones as shown in Figure 5.4. The model was tested for 30 trials and the results are shown in Table 5.4.

Chapter 6

Conclusion

The objectives of this project were to develop a low-cost and efficient system to detect emergency vehicles and control the signals and these have been achieved. The model proposed in this project works with a success rate of 84%. The model is computationally efficient as it does not require any extra memory other than to load the model. This also allows for short and sparsely distributed maintenance times, which allow for the system to be in use for longer periods. The model in total costs about RS. 6000 to make.

6.1 Advantages

- The model does not require much initial set up.
- The model is not memory intensive.
- Requires low maintenance and downtimes.
- Due to using audio data and filters, the model can efficiently detect and determine the direction of emergency vehicles.

6.2 Limitations

- Long-range microphones are expensive and require proper setup.
- The system has been tuned to the specific range of siren frequency and may not run efficiently in other regions of the world. Therefore, a new neural network model has to be build for different region emergency vehicle.

6.3 Future Scope

- Implement the use of long-range microphones to improve the range of detection of the model.
- Implement the complete model on a real world traffic signal

References

[1] K Agrawal, M K Nigam, S Bhattacharya, and Sumathi G, “Ambulance detection using image processing and neural networks,” in Journal of Physics: Conference Series, 2115 012036, 2021.

[2] Ashwin Mohan, Anjitha R Nair, Anjali Anilkumar, Amal Suresh, and Anish A Aziz, “A Study on Smart Regulation of Emergency Vehicles,” in IJRASET, vol. 8, pp. 1010-1014, 2020.

[3] V. Tran and W. Tsai, "Acoustic-Based Emergency Vehicle Detection Using Convolutional Neural Networks," in IEEE Access, vol. 8, pp. 75702-75713, 2020.

[4] Roopashree V, Malavika D N, Nikitha Bai E, Suman A, Dr. Shashikala, “Traffic Congestion Detection and Alerting Ambulance using IoT,” in IJERT, vol. 09, 2020.

[5] Dr. Rajashree V. Biradar and Prashant, “Ambulance Rescue System with Traffic Control using IoT,” in IJESC, vol 9, pp. 21069-21072, 2019.

[6] Karthik B V, Manoj M, Rohit R Kowshik, Akash Aithal, and Dr. S. Kuzhalvai Mozhi, “Ambulance Detection and Traffic Control System,” in IRJET, vol. 6, pp. 1128-1134, 2019.

[7] Y. Huang, Y. Weng and M. Zhou, "Design of Traffic Safety Control Systems for Emergency Vehicle Preemption Using Timed Petri Nets," in IEEE Transactions on Intelligent Transportation Systems, vol. 16, pp. 2113-2120, 2015.

[8] Scola, Carlos Fernández and María Dolores Bolaños Ortega. “Direction of arrival estimation: A two microphones approach,” in Master Thesis, Blekinge Institute of Technology, 2010

[9] Podder, Prajoy & Hasan, Md. Mehedi & Islam, Md & Sayeed, Mursalin, “Design and Implementation of Butterworth, Chebyshev-I and Elliptic Filter for Speech Signal Analysis,” 2020

[10] Bernard Gold, Dan Ellis, and Nelson Morgan, Speech and Audio Signal Processing: Processing and Perception of Speech and Music, John Wiley and Sons, 2011, Accessed on: 25 May 2022 [Online].

[11] Gerard Blanchet and Maurice Charbit, Digital Signal and Image Processing using Matlab, John Wiley and sons, 2006, Accessed on 12 April 2022 [Online].

[12] MATLAB. Estimating Direction of Arrival with MATLAB. (Oct 11, 2017). Accessed: May 25, 2022. [Online Video]. Available: https://www.youtube.com/watch?v=Sz_4jJKjPIQ&t=926s

[13] DrBennett. Cross-Correlation: time-delay estimation and matched filtering. (Jul 31, 2020). Accessed: Jun 5, 2022. [Online Video]. Available: https://www.youtube.com/watch?v=TS8HHW-HlGA&t=260s

Appendix

Raspberry Pi 4: Pin Diagram and Board Specifications

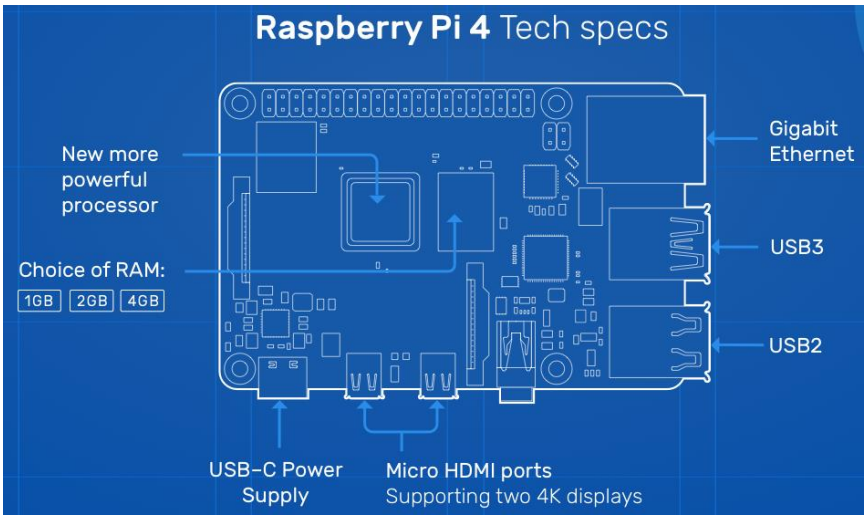


Fig. i) Raspberry Pi 4 model

PIN	NAME		NAME	PIN
01	3.3V DC Power		5V DC Power	02
03	GPIO02 (SDA1,I²C)		5V DC Power	04
05	GPIO03 (SDL1,I²C)		Ground	06
07	GPIO04 (GPCLK0)		GPIO14 (TXD0, UART)	08
09	Ground		GPIO15 (RXD0, UART)	10
11	GPIO17		GPIO18(PWM0)	12
13	GPIO27		Ground	14
15	GPIO22		GPIO23	16
17	3.3V DC Power		GPIO24	18
19	GPIO10 (SP10_MOSI)		Ground	20
21	GPIO09 (SP10_MISO)		GPIO25	22
23	GPIO11 (SP10_CLK)		GPIO08 (SPI0_CE0_N)	24
25	Ground		GPIO07 (SPI0_CE1_N)	26
27	GPIO00 (SDA0, I²C)		GPIO07 (SCL0, I²C)	28
29	GPIO05		Ground	30
31	GPIO06		GPIO12 (PWM0)	32
33	GPIO13 (PWM1)		Ground	34
35	GPIO19		GPIO16	36
37	GPIO26		GPIO20	38
39	Ground		GPIO21	40

Fig. ii) Pin diagram of Raspberry Pi 4

Raspberry Pi 4 Specifications

- Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- 1GB, 2GB, 4GB or 8GB LPDDR4-3200 SDRAM (depending on model)
- 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
- Gigabit Ethernet
- 2 USB 3.0 ports; 2 USB 2.0 ports.
- Raspberry Pi standard 40-pin GPIO header (fully backward compatible with previous boards)
- 2 × micro-HDMI ports (up to 4kp60 supported)
- 2-lane MIPI DSI display port
- 2-lane MIPI CSI camera port
- 4-pole stereo audio and composite video port
- H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode)
- OpenGL ES 3.1, Vulkan 1.0
- Micro-SD card slot for loading operating system and data storage
- 5V DC via USB-C connector (minimum 3A*)
- 5V DC via GPIO header (minimum 3A*)
- Power over Ethernet (PoE) enabled (requires separate PoE HAT)
- Operating temperature: 0 – 50 degrees C ambient