ASHESI

ASHESI UNIVERSITY

SMART VEHICLE DETECTION TRAFFIC MANAGEMENT SYSTEM ENHANCED WITH IOT AND MACHINE LEARNING

CAPSTONE PROJECT

B.Sc. Computer Engineering

Edinam Kofi Klutse

2021

ASHESI UNIVERSITY

SMART VEHICLE DETECTION TRAFFIC MANAGEMENT SYSTEM ENHANCED WITH IOT AND MACHINE LEARNING

CAPSTONE PROJECT

Capstone Project submitted to the Department of Engineering, Ashesi University in partial fulfilment of the requirements for the award of Bachelor of Science degree in Computer Engineering.

Edinam Klutse

2021

DECLARATION

I hereby declare that this capstone is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature:

………………E.K.K…………………………………………………………………………
…

Candidate's Name:

Edinam Kofi Klutse……

Date:

22/04/2021………………………………………………………………………………
…………

I hereby declare that preparation and presentation of this capstone were supervised in accordance with the guidelines on supervision of capstone laid down by Ashesi University

Supervisor's Signature:

……………………………………………………………………………………………

Supervisor's Name:

……………………………………………………………………………………………

Date:

……………………………………………………………………………………………

Acknowledgements

To my supervisor, Mr. Francis Gatsi of the Ashesi University engineering department, this is to say I appreciate all the words of encouragement and direction you gave me to help carry out this project successfully. Additionally, my sincere gratitude goes out to the entire engineering faculty for guiding and giving me the adequate knowledge and skills during my four years in Ashesi to help me undertake this capstone project.

Abstract

Traffic congestion is a major thief of time of the citizens of Ghana especially in urban areas like Accra. Due to the high demand of transportation services in urban areas, which is continually increasing day in day out, the problem has become prevalent in the Ghanaian society.

The problem of traffic congestion affects the productivity of the working population negatively because of the time lost in traffic jams. Automated traffic lights and traffic instructors are doing their best to try and combat the traffic congestion however these measures are not dynamic enough to solve the problem.

Consequently, this problem serves as a motivation to find a better way to handle traffic congestion in urban areas; hence, this paper delves into the use of the internet of things approach coupled with computer vision and machine learning techniques to develop a smart traffic management system based on a vehicle detection algorithm.

Table of Contents

Chapter 1 : Introduction

1.1 Background

The rate at which science and technology are both affecting the transportation sector all over the world is very alarming. Every now and then there is a new vehicle produced with mind-blowing technology, and consumers keep on buying these vehicles to satisfy their needs. To put this into context, from 1976 to 2018, for every 20 years that passed by, the number of vehicles bought in the world has been doubled, moving from 342 million vehicles in 1976 to approximately 1.4 billion owned vehicles in 2018[1].

Evidently, the main resource being used by these vehicles are roads, and with the exponential increase in vehicle production and consumption, there is a resulting increase in pressure on roads. Consequently, engineers in charge of constructing roads are trying their best all over the world to meet the demand for more roads by bringing about innovative solutions like overhead bridges, tunnels, roundabouts etc. However, although road networks have expanded significantly in the 21st century, traffic congestion is still a major problem in the transportation sector. This problem is more prevalent in countries with a lot of urban areas, indicating that the increase in infrastructure alone is failing in combating traffic congestion [2].

In a country like Ghana, especially in an urban area like Accra, traffic congestion is causes significant delay in the daily activities of civilians. Workers and students lose valuable hours trying to pass by traffic light intersections that are congested. It turns out that these intersections with traffic lights are usually the focal point where most of the traffic arises from.

What one may realize at an intersection is that sometimes the traffic is only located at a particular junction, and after that junction, the road is free. This phenomenon is due to so many factors, including accidents, wrong driving skills, and most importantly, the timing sequence of traffic lights at intersections.

Traffic lights at intersections in Accra are configured to have predefined sequential waiting times at each junction; however, this builds up traffic in particular lanes when other lanes have less pressure on them. Some lanes may be used by priority vehicles like ambulances and military vehicles. Some lanes may have encountered an accident too. Essentially, there are so many factors that may cause traffic in a particular lane at an intersection whilst other lanes are operating smoothly.

In the event that uneven traffic congestion occurs at intersections, if traffic lights could gather some intelligence on road networks to calculate the timing of green lights on road demand-based algorithms using real-time data from Internet of Things devices and machine learning algorithms, there should be a significant decrease on the pressure on roads.

1.2 Motivation

As an engineer, problem-solving should be one's main priority when embarking on a project. This project brings an opportunity to engineers to combat a bottleneck that is causing road users to lose time and money. Consequently, I relish this as an opportunity to help improve the traffic light experience of pedestrians and drivers in society through the use of technological driven innovative engineering solutions.

1.3 Problem Definition

Traffic Jam all over the world is an issue that greatly affects the working population to a very large extent. Traffic jams caused by so many factors affect the productivity of workers and students who use the roads.

Consider someone who losses an opportunity because there is an accident on the road which prevents him/her from getting to a job interview on time or paying two hundred Ghana Cedis for an uber trip estimated to be fifty Ghana Cedis due to traffic jam, or even losing a family relative because an ambulance could not arrive in time due to traffic jam. These examples just put into perspective some of the negative implications traffic congestion could have on one's life in an urban area.

In Accra, the capital city of Ghana, the problems being caused by traffic jam are very evident in the day-to-day activities of civilians. During peak times, children are on the move from school, adults are moving from workplaces, hawkers are selling, delivery drivers/riders are making deliveries, taxis are completing trips, etc. A change in weather, an accident, or even a faulty traffic light can turn a day from a productive one into an unproductive one. With so many processes being used by the roads, one can imagine the ripple chain of effects that arise from traffic jams in Accra.

Predefined time-sequenced traffic lights have been the de facto at traffic intersections in Accra. This traditional traffic system allocates the same amount of time to roads regardless of the situation on the road, which causes heavily unbalanced traffic jam when there are casualties on some roads. In cases where there are emergencies, road priority should be shifted to favour emergency vehicles; unfortunately, this is not the case because the traditional traffic light system has little to no intelligence.

1.4 Proposed Problem Solution

The traditional traffic control technique is not smart enough; hence with the introduction of the Internet of Things and machine learning, devices can gain intelligence to behave as if they have a brain. The internet of things approach involves connecting sensors and microcontrollers to sense(listen) situations and control(act), respectively.

With the introduction of computer vision cameras and powerful microcontrollers, cameras can provide real-time visual data on traffic intersections which can be processed into meaningful information about a traffic situation by a microcontroller, consequently informing traffic light how to act accordingly.

Additionally, the system enables remote monitoring and control of traffic lights by a system administrator which would enable troubleshooting of traffic congestion via the web. Essentially the project aims to design a smart traffic control system by using intelligent traffic lights powered with an IoT network and computer vision-based machine learning algorithm for traffic light timing.

1.4 Objectives

- Development of Traffic Congestion Detection System: Developing of circuitry and code to detect traffic congestion at roads using cameras and a microcontroller. This hardware-software interface would be used to control traffic lights when the need arises.
- Development of various algorithms for traffic light timing.
- Development of a Traffic light monitoring system and control system: Development of circuitry and code to notify the administrator when there is a faulty traffic light.
- Development of API for an administrator to remotely control and troubleshoot traffic light problems.

1.5 Requirements

- Microcontroller Unit
- Cameras
- LEDs
- Cables
- Resistors

1.6 Summary

This chapter provides insight into problems caused by traffic congestion at traffic light intersections in urban areas, with a particular focus on a case study in Accra. Looking at the issue from a bird's eye view, this project proposes a method that would tackle the problem of traffic congestion in Accra by modifying the traffic lights from being predefined timers to being intelligent situation-based timers by coupling IoT with machine learning

Chapter 2 : Literature Review and Related Work

With the evolution of smart cities and The Internet of Things, engineers all over the world have been coming out with innovative solutions to tackle traffic congestion in urban areas. Consequently, numerous approaches with varying technological architectures have been developed in the last decade.

Throughout the course of this chapter, various technologies that aim at solving traffic congestion in urban areas would be discussed, compared, critiqued and analyzed in a bid to draw out valuable lessons that may positively affect the structure of this smart traffic management project.

Consequently, the next five subchapters would discuss the papers that are used for the literature review of the proposed smart traffic management system.

2.1 Next-generation intelligent traffic management system and analysis for smart cities

In 2017, Megha H.N and Goudar R.H proposed a solution to traffic congestion in urban cities with the use of data from video surveillance and digital sensors [3]. The main aim of the project was to use IoT methods to determine which civilians are disrupting the flow of traffic. The project revolves around an algorithm that works in the following way:

- Fix Cameras and sensors at traffic intersection to capture data
- Collect and divide camera data(visual) for sensor data(digital)
- Analyze digital data
- If there is traffic congestion spanning over 100 meters, create an alternative path for vehicles.

- If any traffic rule is violated, camera data is used to gather information on the offender(s)' vehicle and passed on to officials to act on the case.

The methodology in the paper is very well documented, the approach seems very intuitive, yet it feels like more meaning can be interpreted from the data being generated by the cameras, because so much data is generated by the sensors and cameras but there are no computer vision based machine learning algorithms for calculating traffic density.

The simulation tool used LabView is very good and shows a visual representation of the system architecture which may be helpful for simulation in this project.

The use of visual/camera data would be applicable to this project however the paper does not detail what tools they use to process the visual data. It is noticed that they use both motion sensors and cameras to capture data however for this Smart traffic management project, it is not necessary to use motion sensors as all the traffic density data required can be captured with the camera with the help of a microcontroller.

The next sub chapter would talk about how a similar project can be carried out by using machine learning algorithms which were not considered by Megha and Ghouda.

2.2. A Collaborative Reinforcement Learning Approach to Urban Traffic Control Optimization

In a bid to curb Urban traffic congestion, As'ad Salkham et all proposed A Collaborative Reinforcement Learning Approach to Urban Traffic Control Optimization in 2008. This approach deals with the design and implementation of a smart traffic system that aims at using information about the status of traffic at a junction to optimize traffic control in urban areas [4].

This method involved the use of an important concept in my project which is machine learning and predictive algorithms. However, cameras are not used to capture data with this approach although I intend to use cameras for data acquisition.

The system is deployed using a local adaptive round-robin phase switching model enhanced with collaborative reinforcement learning. The model works by using local adaptive round-robin phase switching-collaborative reinforcement learning controllers at each junction of interest. These controllers communicate with one another to learn appropriate traffic light timing based on the traffic data gathered from all the various junctions.

I believe that the use of numerous controllers for learning timing at one junction is unnecessary and expensive. A powerful micro controller unit coupled with IP cameras would do a better job since the cameras are less expensive than controllers.

This method was compared to the traditional fixed time traffic control and a saturation balancing algorithm after a simulation of traffic in Dublin. Vivid results show that the adaptive round-robin phase switching-collaborative reinforcement learning approach discussed in this paper lowers the average waiting time per vehicle by 57% when compared to the saturation balancing algorithm which shows very huge potential for this technology.

Although the paper is very organized and intricate, what stands out in the paper is how the simulation results are displayed visually. The graphs and tables tell a vivid story enough to conclude on the results of the research. The next sub chapter would talk about the use of different data acquisition methods for gathering traffic data in an IoT network.

2.3 The intelligent traffic control based on the Internet of Things

In 2015, Guanghua Fu and Zifen Yang proposed a solution to curb traffic congestion based on an adaptive genetic algorithm. Data acquisition for this project was facilitated by the use of Radio Frequency Identification (R.F.I.D), radar devices, and infrared [5].

The use of RFID, radar devices and infrared sensors make the network architecture more complicated as compared to cameras and digital sensors because data must be converted between digital and analog for some nodes in the system. This conversion makes the network architecture complicated with several ADC and DAC channels. That is why I believe using one type of digital sensor(cameras) would be better for gathering data on roads.

This data acquired by the sensors is processed using the adaptive genetic algorithm in order to reduce average waiting time at each intersection. This adaptive genetic algorithm works in 4 steps, namely:

- Chromosome encoding
- Fitness Function
- Genetic Operator
- Stop Condition

Simulation results show that as compared to the traditional genetic algorithm for traffic management in urban areas, this approach yields results which are significantly better than the traditional methods. The method involves very heavy processes like chromosome encoding. The idea of solving this problem by using an adaptive genetic algorithm is intelligent however it bears a lot of cost because not so many IoT processers would be powerful enough to carry out genetic algorithms.

The documentation and research work in this paper is very extensive, also the simulation results look impressive by proving the null hypothesis of the project. The research concluded that traffic congestion can be reduced with the use of IoT and an adaptive genetic algorithm.

With a lot of mathematical equations present in the paper to derive the adaptive genetic algorithm, the paper seems complex at first sight however with the use of some annotated diagrams for network architecture, it brings an interesting way of viewing the topology of the system which helps a lot in visualizing the design.

A good takeaway point from this paper is the use of a star network architecture to facilitate the communication between the sensors and the microcontroller. This topology ensures there is no loss of data due to its robustness. Consequently, the star network architecture may be adapted for the use of the smart traffic management system. The next subchapter talks about a project that delves into the idea of big data analytics unlike the previously discussed papers.

2.4 Internet of Things - Smart Traffic Management System for Smart Cities Using Big Data Analytics

At the 14th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP) in 2017, Abida Sharif et all proposed a smart traffic system with the use of low cost IoT sensors and big data analytics [6]. This approach requires sensors to be fixed every 500 meters on roads to acquire traffic data for real-time streaming using big data analytics coupled with a smart traffic system algorithm.

Real-time streaming of data is a desirable feature of my smart traffic management project, so I believe that the use of big data analytics to process sensor data in real time is a very efficient method. Numerous Internet of things applications such as the use of NoSQL schemas and edge processing nodes for real time processing of data would be applicable to my project as it increases speed and robustness of IoT networks

This approach, compared with traditional methods, also proved to be significantly better as per simulation results. The most important part of this paper is how well the authors described the process of big data analytics using diagrams and a short paragraph. The use of big data analytics for sensor data resonates with the concept of this project as I intend to implement some machine learning algorithms. The next chapter would discuss a paper which also goes into the use of big data analytics coupled with web application technology to solve a similar problem

2.5 A Traffic Management System to Improve Traffic Control

Edwin Adatsi proposed a traffic management system in 2018 which uses cameras to capture data, process the data with computer vision and machine learning techniques and control the traffic light based on the traffic density with the help of a web application.

I believe that the work done in this project is good however only an unsupervised learning method was used to carry out the vehicle detection techniques. I intend to explore both supervised and unsupervised machine learning techniques for processing the video data.

The use of the MEAN (Mongo DB, Express JS, Angular Js, Node Js) web technology stack to create a user-friendly web application that can monitor traffic density and control traffic lights is a superb idea. The steps involved in using this technology is well outlined and justified by its robustness in the IoT network. Consequently, the MEAN web application would be considered as a component of the smart traffic management system being developed in this project.

The next chapter outlines the key concepts and findings from the literature review that would affect the proposed smart traffic management system.

2.6 Key Findings

The literature review from the five papers above have shaped up the following ideas for the proposed traffic management system:

- The use of cameras to capture traffic data as used by Megha H.N and Goudar R.H in section 2.1
- The use of machine learning algorithms to learn traffic density data as discussed in section 2.2
- The use of a star network architecture discussed in section 2.3
- The use of big data analytics on sensor data as discussed in section 2.4
- The use of a MEAN full stack application to monitor and control a smart traffic intersection in section 2.5.

These key findings would serve as a guide to help come out with the requirements for the smart traffic management system which would be discussed in chapter 3.

Chapter 3 : Requirement Specifications

In a bid to clearly define the scope of this project, this chapter outlines the requirement specification that would serve as a guideline for the design, simulation, and implementation of a smart traffic monitoring system. The requirements would be divided into three parts, namely user requirements, system requirements, and non-functional requirements.

3.1 User Requirements

The user requirement section typically consists of the qualitative constraints needed to be met in order to make the system efficient and user friendly. After brainstorming the following user requirements have been derived for the project:

1. The traffic light system should have a very short waiting time.

2. Traffic light system should be safe for all road users

3. Traffic light system should have a low cost of maintenance and implementation

4. Traffic light system should be intelligent and autonomous

5. The traffic light system must be responsive to the system administrator's input.

3.2 System Requirements

The system requirement section details the technical specifications of the traffic light system that would make the various subsystems responsive and efficient for the users. The system requirements for the smart traffic management system are as follows:

1. The system should have highly accurate sensors and cameras at intersections to capture data on-road usage.

2. The system should have an effective means of communicating data collected between various sensors and subsystems, with high throughput.

3. The system should have a microcontroller capable of processing the data and carrying out machine learning algorithms on data in real-time

4. The system should have edge and fog nodes for processing some of the data.

5. The nodes in the system communicating should have end to end encryption.

3.3 Non-Functional Requirements

This section defines the desired system's attributes at a high level, in relation to what would make the system efficient and effective. The Non-Functional Requirements are stated below:

1. Subsystems should be interoperable with one another.
2. The entire Traffic System should be scalable.
3. The entire Traffic System should be reliable and redundant.
4. The entire traffic system should be easy to maintain.
5. The entire Traffic system should secure.

Summary

Going forward, the requirements specified in this chapter would be used as a guide or roadmap to iteratively design and build a safe, secured, smart traffic management system with high throughput and low cost.

Chapter 4 : Design and Implementation

4.1 Design

This section details the design choices made based on the requirements specified in chapter 3. In order to effectively and efficiently build a working prototype of a smart traffic management system, the various subsystems need to be clearly defined and designed based on the requirements needed for the whole system to function. The subsystems identified in this project are Vehicle detection Subsystem, Software Subsystem, and Electrical Subsystem.

4.1.1 Overall System Architecture

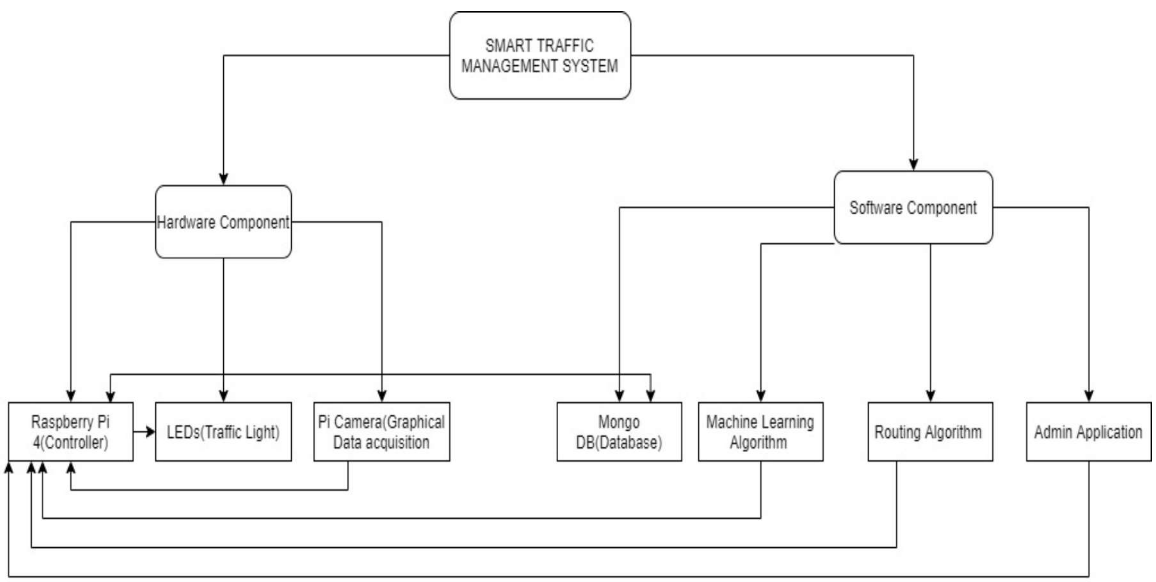


Figure 4-1System Architecture

The figure above depicts the birds-eye view of the smart traffic monitoring system using a simple block diagram. This figure does not necessarily show the subsystems because components in each subsystem overlap. However, this diagram does a good job in describing the individual components in the project and how they interact with one another. Mainly composed of hardware components and the software components which communicate with each other to yield results, the subsequent sections in this chapter would provide a broad overview on how the system is designed to function.

4.1.2 Vehicle detection Subsystem

The first subsystem to be discussed is the vehicle detection subsystem. This subsystem is the one responsible for sensing, processing and storing data concerning traffic density on the road. The vehicle detection system is made up of 4-12 Cameras for capturing video data and a computer/processor/microcontroller for edge processing of the video data using computer vision and machine learning algorithms. After processing, the microcontroller is also responsible for sending the data to the cloud.

4.1.2.1 Design Choice for Microcontroller

The design criteria for evaluating the microcontrollers suitable for this project are defined as:

- Quality: Processing power, Power Constraints, Scalability, and interoperability.
- Availability: How available the said device is on the Market
- Cost: Unit cost of buying said device

The three main devices considered for this purpose were the Raspberry Pi 4, The Arduino Mega, and Le Potato microcontroller units. The table below summarizes how well each device performs based on the selection criteria specified.

Camera	Quality	Availability	Cost

Raspberry Pi 4	Relatively High	Readily Available in Market	$45
Arduino Mega	Relatively low	Readily available in Market	$38
Le Potato	Relatively moderate	Not available in. Ghana	$35

Due to the **Raspberry Pi 4** being the only available choice on the Market with high peripheral and processing specifications, it was selected as the primary choice for this project despite being the most expensive device.

4.1.2.2 Design choice for camera sensor

The design criteria for evaluating the right camera sensor for the project are defined as:

- Quality: Camera range, interoperability, pixel quality
- Availability: How available the said device is on the Market
- Cost: Unit cost of buying said device

Camera	Quality	Availability	Cost
Raspberry Pi Camera Module v1	Fairly good	Readily available	$15

Raspberry Pi Camera Module v2	Good	Available outside the country	$30
Raspberry Pi NoIR camera module v2	Very Good	Available outside the country	$30
Waveshare Raspberry Pi Camera Module	Fairly Good	Out of stock	$40
Arducam Video module for Raspberry Pi	Fairly Good	Out of stock	$25

From the design criteria, it turns out that although the NoIR camera seems to be the best option. However due to shipping problems in Ghana, currently, the Raspberry Pi v1 camera module is the only accessible device on the list. Initially, it ranks as the 3rd best for this purpose among the five devices listed above. However, due to the availability constraint, this project would be carried out by using the Raspberry Pi v1 camera module as a compromise for capturing graphical data.

4.1.2.3 Design choice for machine learning and video processing language

The Python language is the obvious choice for video processing with the raspberry pi camera as it has extensive libraries that are very suitable for videography with the raspberry pi camera [8]. Additionally, the computer vision and Machine learning aspect of this project is suitable for python as it has libraries like **Open CV** and ImageAI.

For the purpose of video processing and training the vehicle detection model, Python's Open CV library would be used because of how well documented it is, making it is simple to use. Additionally, Open CV is an open-source computer vision and Machine Learning library; hence it is free for all to use for whatever purpose [9].

4.1.3 Software Subsystem

This subsystem is mainly comprised of the application that allows an administrator to monitor and control the traffic situation remotely. The application should have a login page for authentication and a dashboard for the authorized administrator to monitor and control the system when necessary. The application should have a database that it can read and write data to with little to no latency.

From the literature review and research done before going into the project, engineers who have tried to control hardware like raspberry pi would use either a web application or a mobile application. Going forward, this project would make use of a web application instead of a mobile application because of interoperability and ease of access that new web application technologies bring with respect to dealing with data. There are numerous free web development stacks for this purpose, such as the MEAN stack, the LAMP stack, the MEVN stack etc.

For the purpose of this project, the MEAN stack would be used to build the application because of its unanimous superiority over all the open-source full web stack technologies [7]. MEAN is a software stack made up of a layer of technologies (Mongo DB, Express, Angular, and Node Js) that is used for creating a full-stack web application. MEAN has many advantages over other web stacks, namely:

- Being able to use a single language throughout the deployment process

- Reusability of code
- Fast queries with MONGO DB NoSQL schema
- Easy to use Node JS and Angular

The MEAN stack uses:

- Angular for the front-end UI (Html and CSS components)
- Node Js for running server side back-end process out of the browser
- Express Js for routing of pages
- Mongo Db Atlas cloud for storing data in JSON like documents

These reasons justify why the MEAN web stack would be used for the software aspect of the smart traffic management system. The MEAN web application would interface with the raspberry pi by using MQTT publish-subscribe server technology with the help of Node Js due to the flexibility and interoperability of both technologies in IoT platforms [10].

4.1.4 Electrical Subsystem

For the purpose of modelling the electrical system, certain electrical components would be used to represent certain objects in real life. This system would be made up of the raspberry pi, led lights, a breadboard, cameras, resistors, and wires. The figure below shows how the system would be modelled for experimentation with the use of a simple schematic.

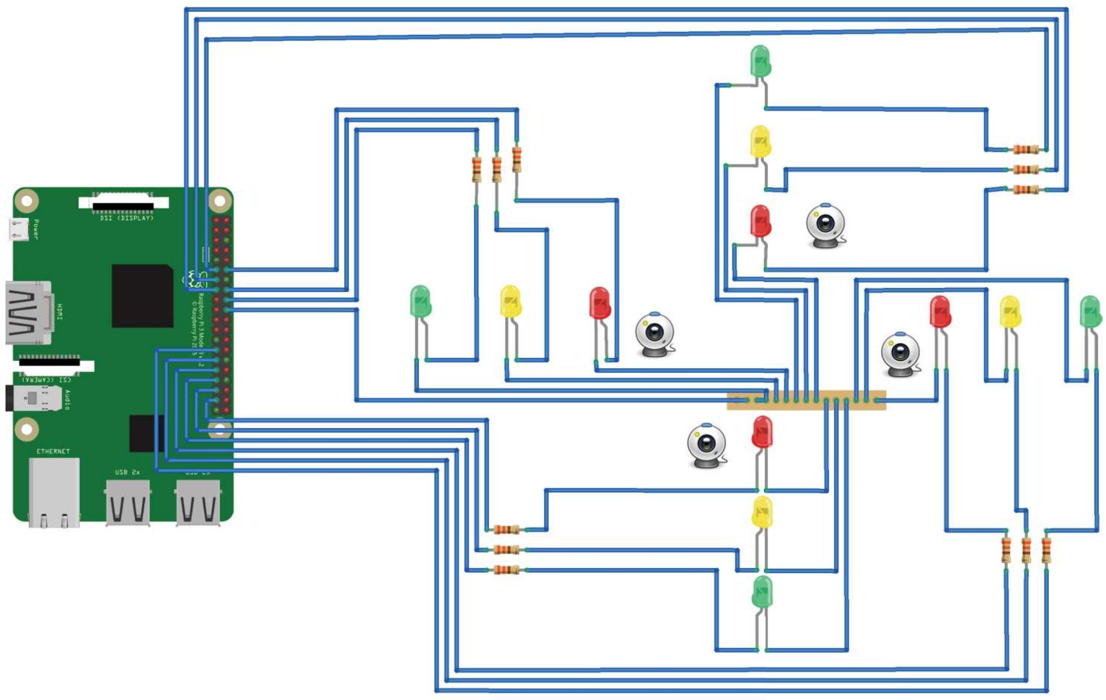


Figure 4-2Electrical Schematic

4.2 Implementation

4.2.1 Vehicle Detection Subsystem

With the help of Python's Open CV library, the main part of the project, i.e., Vehicle detection was accomplished using supervised and unsupervised machine learning methods. Essential computer vision concepts which constitute the steps for developing the computer vision and machine learning techniques are [11]:

- Frame Differencing: Since a video is a collection of frames, to be able detect moving objects in a video, the model should be able to detect the difference in each frame.
- Image Thresholding: This process involves binarizing gray scale frames which means that every set of pixels above a certain pixel intensity threshold can be detected as a certain object(vehicles) whilst every other set of frames is detected as something else.

- Drawing Contours: Using contours to identify certain shapes in specific areas of a frame which the same color intensity
- Image Dilation: Convolution of areas of images which seem to have same color intensity but maybe separated by space due to movement of frames

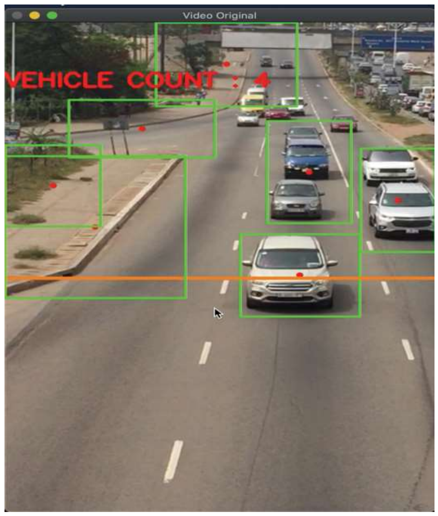
The supervised learning method involved training a deep learning model using two hundred pictures and videos of cars, motorbikes, pedestrians, and roads to be labelled as vehicle on non-vehicle based on the developed model. Essentially this means that cars and motorbikes are labelled as vehicles, whilst every other thing is labelled as a non-vehicle. Hence the model can detect cars and motorbikes moving as vehicles and use that to determine the traffic density.

The unsupervised learning method involved using a pre-trained HAAR Cascade Car face classifier XML file to detect the cars in each frame of a video. This method used a pre-trained classifier, implying that the features are not labelled as per the features of a typical Ghanaian road scene; hence more faces may be detected as vehicles once detected in the frames.

Both methods use a region of the video frame as the detection region. A line is drawn at a specific coordinate, which is the boundary for the car detection algorithm. Any detected vehicle that moves from one frame to another across the car detection boundary is added to the vehicle counter, which is recorded as the traffic density.

Videos taken from thirteen feet above the ground at the footbridges at Shiashie and Madina in Accra were used to test both algorithms. The vehicle detection boundary becomes orange when a new detected vehicle enters the boundary. The figure below shows a screenshot depicting how the vehicle detection system works. The results would be discussed in chapter 5.

Figure 4-3



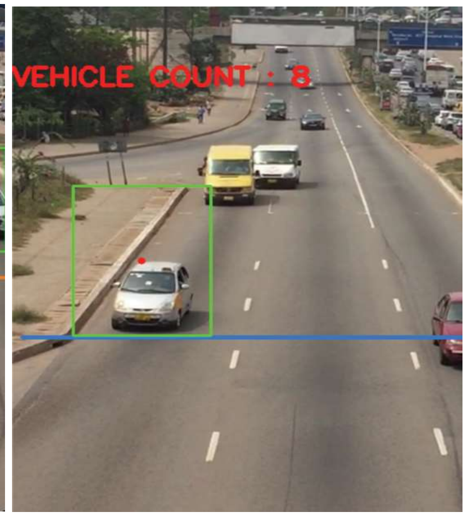


Figure 4-4

4.2.2 Software Subsystem

The software subsystem is built using a MEAN (Mongo DB, Express, Angular, Node Js) stack web application.

The application has a login page, and a dashboard. A user/admin logs in to the website to authenticate the log in process. Upon success, the user is allowed to view the traffic intersection dashboard which permits the user to control which algorithms the traffic lights should run. This is done by using the Node Js backend to send signals to the raspberry pi through an MQTT server.

Consider a four-way intersection in which only one lane can have a green light on at each time as seen in the figure below

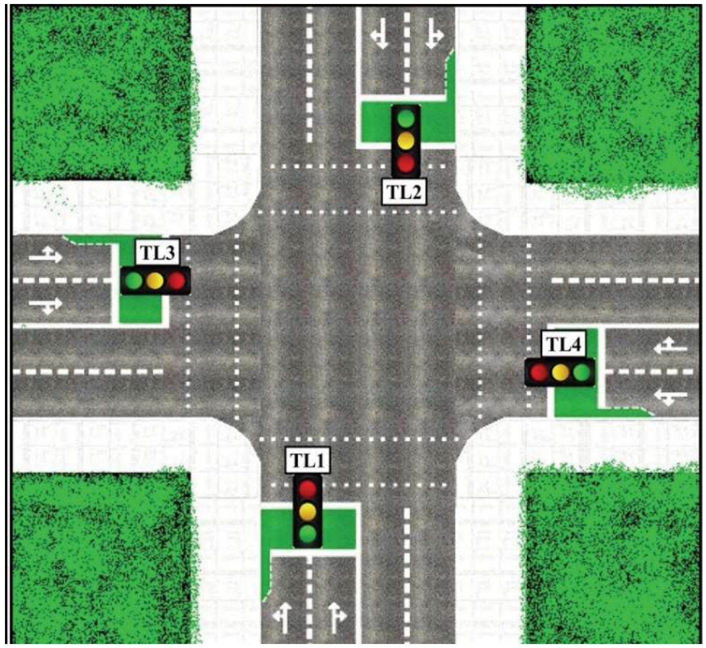


Figure 4-5

For the purpose of testing, the traffic light algorithms are written for such a traffic intersection in the figure above with the following assumptions:

- Only One Traffic Light can be green at a time
- The outflow of traffic from one lane does not flow into another lane
- Traffic lights are modelled as LEDS which can be controlled with GPIO pins

The traffic light intersection can be operated using either the standard round robin predefined timing algorithm that is used for regular unintelligent traffic lights in Accra or the smart traffic density-based timing algorithm developed for this project

The pseudocode for the two main algorithms written in python for the traffic lights to be controlled by the raspberry pi 4 are:

1. Standard Round Robin Predefined timing algorithm

All green lights, red lights, and amber are on for the same period of time for each lane in the intersection
Consider the length of period for which each light should be on to be denoted by

Rn = time on for red light n
An = time on for amber light n
Gn = time on for green light n
Where n is the traffic light number, example for TL1 n is 1
Then
$R1 = R2 = R3 = R4$
$A1 = A2 = A3 = A4$
$G1 = G2 = G3 = G4$

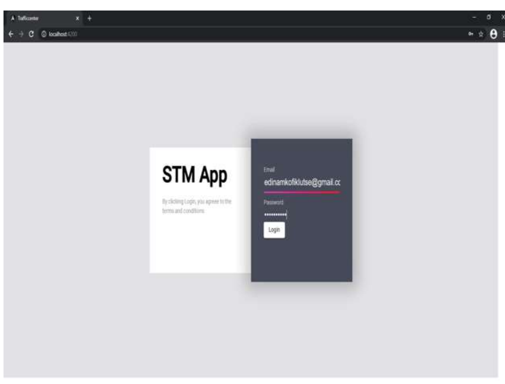
2. Smart traffic density-based timing algorithm

Red lights and amber lights are on for the same period of time for each lane in the intersection
The period for which green lights are on is calculated using the traffic density data based on a simple proportional mathematical formula.
Consider the length of period for which each light should be on to be denoted by
Rn = time on for red light n
An = time on for amber n
Gn = time on for green light n
Where n is the traffic light number, example for TL1 n is 1
Then
$R1 = R2 = R3 = R4$
$A1 = A2 = A3 = A4$

$$Gn = \frac{Traffic\ density\ of\ lane\ n}{Sum\ of\ traffic\ density\ of\ all\ lanes} \times Maximum\ Waiting\ Time\ for\ the\ intersection$$

Hence lanes with higher traffic density are on for a longer time

Figure 4-6



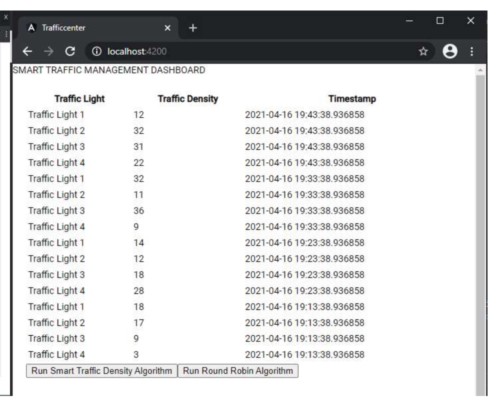


Figure 4-7

The figures above shows the GUI for the MEAN web application which was deployed and test. The test results would be further discussed in chapter 5

4.2.3 Electrical Subsystem

The electrical subsystem model is built using simple electronic components to model a smart traffic intersection. The Raspberry Pi 4 is used to control LEDS, which have been connected to the raspberry pi GPIO pins with the help of a breadboard and jumper cables as seen in figure.

The main aim of this subsystem is to model how the traffic lights would be connected using GPIO (general purpose input and output) ports and also to display how the routing algorithms would work for the traffic light. The figure below shows the model that was created to test the use of the Raspberry Pi 4 GPIO pins for controlling traffic lights modeled as LEDS.

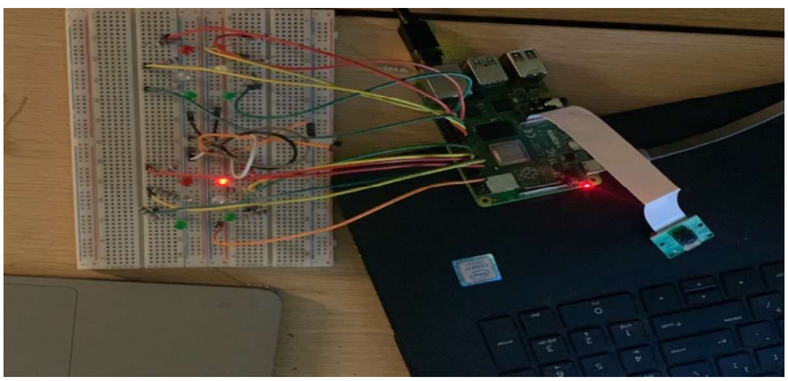


Figure 4-8

4.2.4 Implementation of Full System Architecture

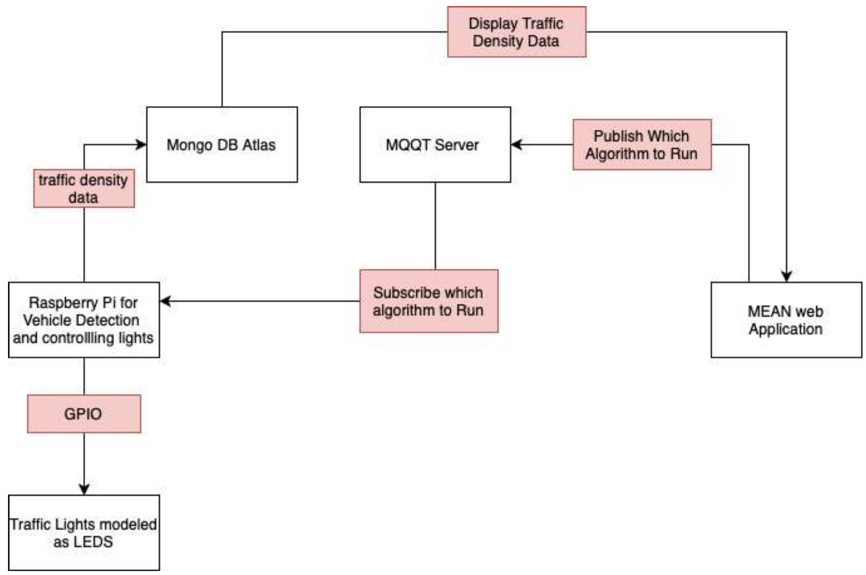


Figure 4-9

The entire smart traffic system is implemented based on the structure seen above. The Raspberry Pi is always running a python script that detects the number of vehicles on the road and sends the traffic density data to the Mongo DB atlas cloud in real time. The Raspberry Pi simultaneously controls the traffic lights using the GPIO pins. The traffic lights are timed based on which traffic light routing algorithm is being used.

By default, the raspberry Pi uses the round robin timing algorithm. However, in order for an administrator to control which timing algorithm the traffic lights should use, the raspberry Pi subscribes to a topic on an MQTT server which tells the raspberry Pi which algorithm to use. At the same time, the MEAN web application publishes to the same MQTT topic, which algorithm the administrator has instructed the raspberry Pi to use.

Essentially the Administrator talks to the raspberry pi by using the MQTT server. When the Administrator logs in, traffic density data from the Mongo DB atlas cloud is displayed in real time and the administrator can publish the desired algorithm to the MQQT server by clicking a button on the administrator dashboard.

4.3 Conclusion on Design and Implementation

This chapter sheds light on the design process for the various subsystems of the proposed smart traffic management system. Notable technologies like Computer Vision, Machine Learning, MQTT, MEAN web stack, etc. are used to contextualize the design and how the system should be implemented. The next chapter would show how the performance of the components of the system were evaluated.

Chapter 5 : Testing and Results

This chapter delves into how the different subsystems of the traffic monitoring system are tested to make sure that the system works. This section would use statistical concepts on the results obtained from testing to draw conclusions about how feasible the design is

5.1 Vehicle detection subsystem testing

In a bid to test the accuracy of the vehicle detection scripts developed and discussed in the previous chapter, three different categories of videos taken at the Accra-Shiashie highway were used to test the accuracy of the vehicle detection algorithms. Both the supervised and unsupervised Open CV machine learning vehicle detection python scripts were tested. Three different categories of videos were labelled as, fast-moving traffic, average speed moving traffic, and slowly moving traffic. Each category had five videos hence the accuracy test used an average of five data points for each category, and fifteen data points in total.

In order to test for the accuracy of both the supervised and unsupervised learning methods, visual inspection was used to count the actual number of vehicles in the video, then compare that figure to the number of vehicles detected by the scripts. The vehicle detection accuracy results for both methods are compared and statistically summarized in the next two sub-chapters.

5.1.1 Supervised Learning method results

Video Label	Mean Actual Number of Vehicles	Mean Number of vehicles recorded by script	Mean Accuracy
Fast Moving Traffic	24	19	79.16667%
Average Speed Moving Traffic	15	11	73.3333%
Slowly Moving Traffic	33	44	66.6667%

For the supervised model, the mean accuracy of the algorithm for all 15 data points was 73.0553% with a standard deviation of 5.1068. It turns out that the model is more accurate in detecting vehicles in faster videos.

The possible reason for this phenomenon would be that the deep learning model was trained using videos with faster moving traffic hence it is more familiar with that type of data. Also, another reason that could justify this is that some of the pixels detected as cars do not get dilated as much in faster-moving traffic as compared to slower moving traffic, because cars are not so close to each other in faster moving traffic.

To shed more light on why excessive image dilation could cause bad results; when contours detected by the computer vision algorithms are too close to each other, the image dilation function may turn two or more vehicles into one, hence increasing the false negatives. Implying that the fewer the cars, the better the performance of the model since less cars would be falsely convoluted by the image dilation function.

5.1.2 Unsupervised Learning Method results

Video Label	Average Actual Number of Vehicles	Average Number of vehicles recorded by script	Accuracy
Fast Moving Traffic	24	14	58.3333%
Average Speed Moving Traffic	15	9	60.0000%
Slowly Moving Traffic	33	29	87.8787%

For the unsupervised learning model, the mean accuracy of the vehicle detection algorithm was 68.7373% with a standard deviation of 16.5978, due to data being skewed by the 5 data points in the slowly moving traffic data category.

It turned out that the accuracy was low for fast-moving and average speed moving traffic because of lack of detections which could be visually spotted by the script.

This is because the data used to train the models are significantly different from the testing data. Unlike the supervised methods which used videos of roads to train and test the data, the unsupervised model uses pre trained classifiers which only know the features of cars, but not that of roads, pedestrians, motorbikes, sign boards, etc.

Less contours where drawn each time the algorithm was run on the videos for fast moving traffic and average speed moving traffic, hence there were a lot of visible false negatives for that category; however, for slowly moving traffic, the model detected a lot of vehicles which is because most of the objects detected in the frames as vehicles were not vehicles, hence there were a lot of false positives due to over dilation.

On average, the results conclude that both models can be used for detecting vehicles; however, the supervised learning model is more consistent and accurate. Nevertheless, due to lack of training data, the model is not yet accurate enough for a real traffic system which should have an accuracy of at least 90% [11] as compared to the 73.0553% accuracy obtained by the supervised learning model.

5.2 Software Subsystem (Routing Algorithm Testing)

Due to limitations, the routing algorithms could not be tested on the actual roads. However, in a bid to test the efficiency of the smart traffic routing algorithm as compared to that of the round robin algorithm, a python script was written to handle the test.

The python script modeled a traffic intersection as a list of queues. A queue is a data structure that works on the first in first out principle. Traffic light lanes also work on the same principle; hence each traffic light lane was modeled as a queue.

The following assumptions were made:

- A vehicle is an object in the queue
- A queue is a traffic lane in a four-way traffic intersection
- A list is a four-way traffic intersection made up of 4 queues

- The time taken for each vehicle to exit a lane is the same for all lanes i.e., cars are moving at the same pace
- The length of the queue is the traffic density of the said lane
- When a traffic light is red, the same number of vehicles are added to the queue for the corresponding lane
- When a traffic light is green, cars are removed from the queue for the corresponding lane at the same pace
- The routing algorithm determines how long each queue should have a green light on

The test was run on both algorithms by using the same set of queue data generated by the python script for consistency. The queue data mimics the initial traffic density data that were recorded by the vehicle detection algorithm. The queue data is generated dynamically to resemble real traffic density data by using python's random library. Twenty different sets of queue data were used for the test.

The parameter used to test the efficiency of the algorithms was the average traffic density for the list of queues/traffic intersection after running the algorithm on the list. This parameter was used because the algorithm's main aim is to reduce traffic density.

The average traffic density was calculated by averaging the length of each queue in the list after the algorithm had been run on the traffic intersection. The results of the test were collated in a csv file to be statistically analyzed using R studio's unpaired t test. The tables below summarize the results of the test.

	Smart Traffic Algorithm Average Traffic Density	Round Robin Algorithm Average Traffic Density
Mean	12.79	24.40

T-score	Df	P Value	Confidence Interval
-5.7649	36.02	1.43e-06	0.95

The alternative hypothesis of this test is that; With a lower average traffic density of 12.79, the smart traffic density algorithm can be said to be more effective as compared to the Round Robin algorithm. The p value of the test being 1.43e-06, indicates that the means are statistically different hence the alternative hypothesis is proven.

This result indicates that the smart traffic density-based algorithm reduces average traffic density better than the round robin algorithm being used by traffic lights in Accra based on the queue data sets generated by the python interpreter. Hence the smart traffic algorithm is better than the round robin algorithm.

5.3 Testing of Interoperability of Entire Traffic Management System

Finally, the communication between the various components that make up the subsystems and the whole IoT network is tested to ensure that the system is feasible and scalable.

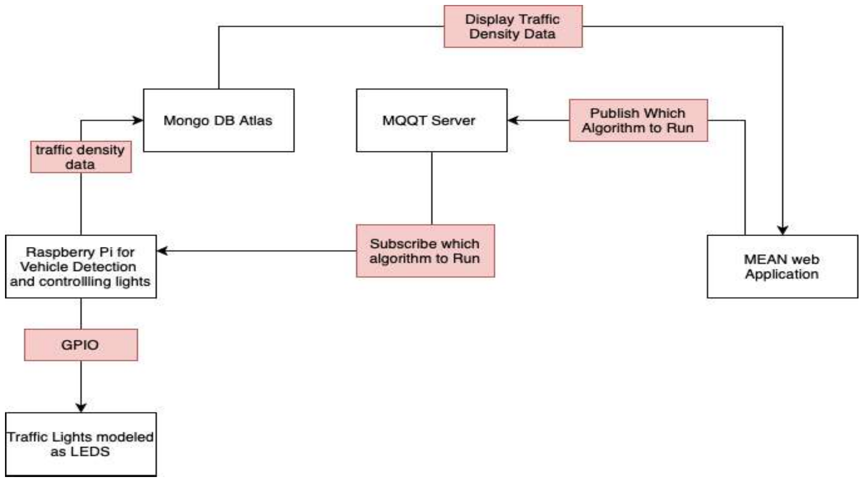


Figure 5-1

The system architecture seen in the figure above is tested to check the latency of the system and how interoperable the components of the system are with one another.

To test the communication between the Mongo DB Atlas database, the raspberry Pi and the Web Application, the raspberry pi terminal is used to send data to the Mongo DB atlas database twenty times for consistency. Consequently, a page is opened on the Mongo DB atlas portal to inspect how fast the data is being updated with the help of timestamps.

Simultaneously, the web application is logged in to the admin portal to check how fast the data is being updated by the node Js runtime. Before doing this test, the raspberry Pi and the Node Js runtime must be whitelisted by the Mongo DB atlas cloud. The timestamps shown on the test data being sent by the raspberry Pi is immediately received by the Mongo DB atlas with little to no latency in less than a second, indicating that the two-way connection of the database to the website and the raspberry pi is fast and efficient.

Essentially the timestamps generated by all 3 endpoints (raspberry pi, mongo db atlas, and node js runtime for the web application) are compared for all twenty sets of data published by the raspberry Pi to check for the network's latency and loss of data. The visual test shows that there is no loss of data, and the data travels through all three nodes in less than a second. The figures below show how visual inspection is used to check for loss of data and speed of communication.

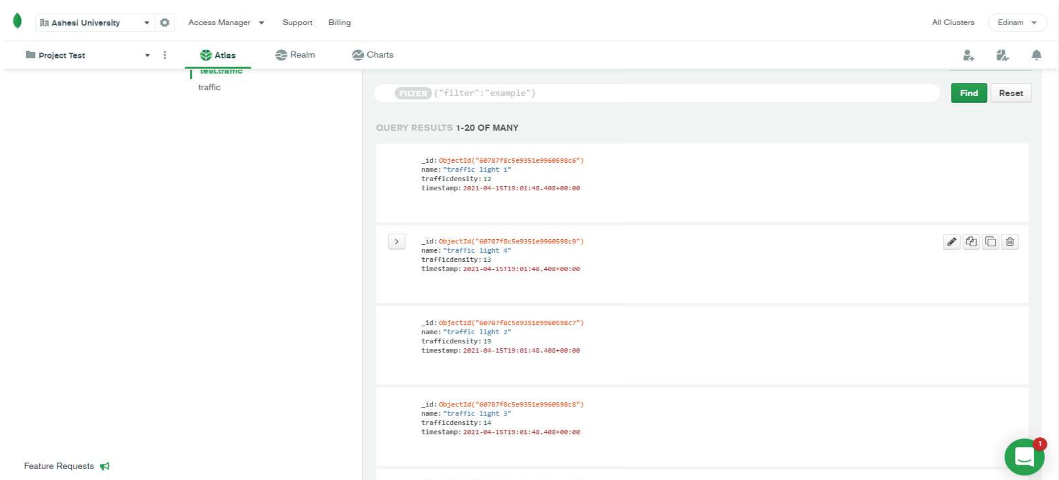


Figure 5-2

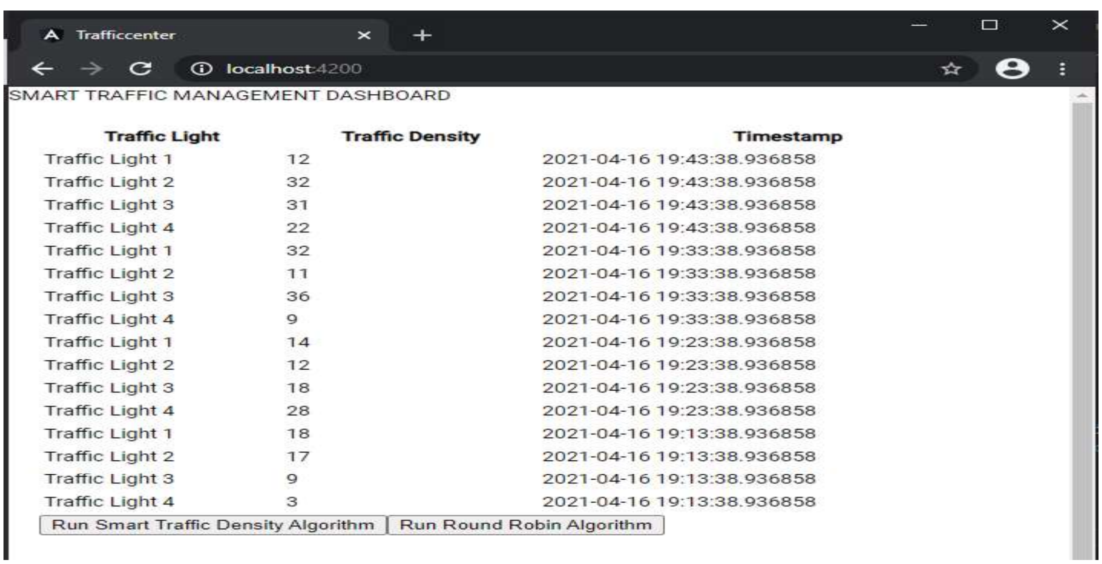


Figure 5-3

To test how the quick the admin web application is able to communicate with the raspberry pi by using the MQTT broker, the raspberry pi terminal is subscribed to a specific topic on the MQTT broker.

The code is written in such a way that as soon as the raspberry Pi receives a specific string of bytes "a" from the broker, it prints some traffic density data to the terminal and runs the smart traffic density algorithm which controls the LEDs connected to the GPIO pins. Alternatively, when the raspberry Pi receives another string of bytes, "b", it runs the round robin algorithm to control the LEDs connected to the GPIO pins.

Hence one button on the web application labeled "Run Smart Traffic Density Algorithm" publishes string "a" to the same topic that the raspberry Pi is subscribed to whilst the other button labelled "Run Round Robin Algorithm" Publishes string "b".

To test how quick the two-way MQQT connection is, the admin of the web application clicks on one of each button on the admin dashboard to publish data to a specific topic on the MQQT broker. It is noticed that immediately, the raspberry Pi terminal receives the data and the LEDs come on respectively with no noticeable delay in the system.

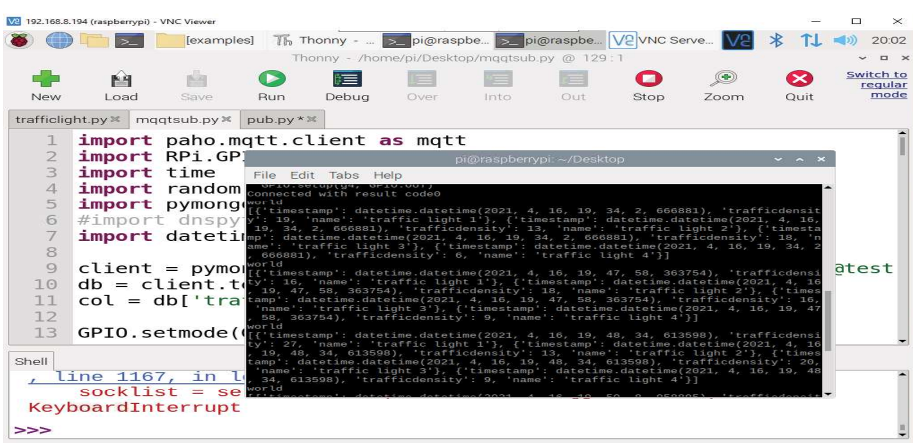


Figure 5-4

The test proves that the network architecture is fast, safe and secure. There is no loss of data and all the data entry points in network are interoperable with one another. This shows that the overall system architecture is very good hence can be scaled up and used in the real-world application of a smart traffic management system.

Chapter 6 : Conclusion and Future Works

This chapter presents a summary of the results of the project. The main aim of this chapter is to unfold the major findings, challenges and limitations that came out during the course of the project and how they may shape the thought process for better prototypes to be made in the future.

6.1 Summary

This capstone project delves into the use of upcoming technologies like the Internet of Things, machine learning, and computer vision to propose a solution to traffic congestion in urban areas in Ghana such as Accra. With the evolution of smart cities, more vehicles and more road networks, urban areas in Ghana would need smarter traffic management systems to increase productivity of the working population.

This method proposes that cameras be stationed at traffic intersections to help traffic lights make the right timing decisions based on the traffic density data captured by the cameras and processed by a micro controller. The project proves that this traffic density-based timing is more efficient than the current round robin pre-defined timing sequence used by traffic intersection in Ghana. The network architecture of the proposed solution is simple to implement and scalable.

6.2 Limitations

Certain bottlenecks were discovered on the way to implementing this project which called for compromises to be made.

- Lack of enough traffic surveillance data to train the machine learning models to perfection

- Complicated Computer Vision techniques like image dilation were hard to figure out
- Shipping parts to Ghana for experimentation was impossible for some devices like the NoiR v2 Raspberry Pi Camera
- The nodes in the network needed to be whitelisted by the database anytime a connection is to be made because the IP addresses keep changing
- Finding a place to mount the cameras and take good surveillance footage was a challenge

6.3 Future Works

This section indicates the improvements that can be made to make the proposed solution a more viable one. Certain improvements which can be made on this project are:

- Finding a better way of mounting cameras in order to take more stabilized footages
- Gathering more video data to train a better vehicle detection model
- Using a better camera than the raspberry pi camera in order to get better quality videos
- Improving the Web application to show a live stream of the traffic intersection being modeled
- Migrating the whole full stack web application to be run on cloud instead of a local host

6.4 Conclusion

To conclude this paper, the smart traffic management system designed and developed from this learning experience provides a lot of essential insights into the possibilities of changing the traffic management scene in urban areas of Accra.

The results obtained from testing the various subsystems of the proposed solutions indicate that the network architecture is scalable and feasible, however the vehicle detection model needs a lot of fine tuning to be able to be implemented in the real world.

This project can be deemed as a successful one due to the fact that the various subsystems were developed and tested successfully to unfold useful findings that would improve the work of researchers that may come across this paper.

References

[1] Dickson, I. (2020, January 08). Want to improve traffic congestion? Don't just build new roads. Retrieved October 05, 2020, from https://360.here.com/want-to-improvetraffic-congestion-dont-just-build-new-roads

[2] IEA. (2018, May 29). Global EV Outlook 2018 – Analysis. Retrieved October 05, 2020, from https://www.iea.org/reports/global-ev-outlook-2018

[3] R. H. Goudar and H. N. Megha, "Next generation intelligent traffic management system and analysis for smart cities," 2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon), Bangalore, 2017, pp. 999-1003, doi: 10.1109/SmartTechCon.2017.8358521.

[4] A. Salkham, R. Cunningham, A. Garg and V. Cahill, "A Collaborative Reinforcement Learning Approach to Urban Traffic Control Optimization," 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Sydney, NSW, 2008, pp. 560-566, doi: 10.1109/WIIAT.2008.88.

[5] G. Fu and Z. Yang, "The intelligent traffic control based on the Internet of Things," 2015 8th International Conference on Biomedical Engineering and Informatics (BMEI), Shenyang, 2015, pp. 614-618, doi: 10.1109/BMEI.2015.7401577.

[6] A. Sharif, J. Li, M. Khalil, R. Kumar, M. I. Sharif and A. Sharif, "Internet of things — smart traffic management system for smart cities using big data analytics," 2017 14th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), Chengdu, 2017, pp. 281-284, doi: 10.1109/ICCWAMTIP.2017.8301496.

[7] Adatsi, E. (2018). A TRAFFIC MANAGEMENT SYSTEM TO IMPROVE TRAFFIC CONTROL. *Ashesi University 2018 Capstone*.

[8] Fouladi, S., Riad, R., Shacklett, B., Zeng, W., & Sivaraman, A. (2017). Encoding, Fast and Slow: Low-Latency Video Processing Using Thousands of Tiny Threads. *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI '17)*.

[9] Kapur, S. (2017). *Computer vision with Python 3 image classification, object detection, video processing, and more*. Birmingham ; Mumbai: Packt.

[10] Chooruang, K., & Mangkalakeeree, P. (2016). Wireless heart rate monitoring system using mqtt. *Procedia Computer Science, 86*, 160-163. doi:10.1016/j.procs.2016.05.045

[11] Joshi, P. (2020, December 23). Vehicle detection in videos using OpenCV and Python. Retrieved April 21, 2021, from https://www.analyticsvidhya.com/blog/2020/04/vehicle-detection-opencv-python/