ASHESI UNIVERSITY

A TRAFFIC MANAGEMENT SYSTEM TO IMPROVE TRAFFIC CONTROL

CAPSTONE PROJECT

B.Sc. Computer Engineering

Edwin Adatsi

2019

A TRAFFIC MANAGEMENT SYSTEM TO IMPROVE TRAFFIC CONTROL

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.

Edwin Adatsi

April 2019

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:

……

Candidate's Name:

……

Date:

……

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, Dr. David Ebo Adjepon-Yamoah, whose direction, encouragement and academic advice helped me undertake this project. Also, my acknowledgement goes to the entire Engineering faculty for the knowledge acquired over the years of this bachelor's degree.

Abstract

Vehicles provide an efficient way of transportation across the cities of Greater Accra in Ghana. Technology has impacted the transportation sector through variety of modes such as applications for routing. An example of this is the google maps mobile application. However, due to bad road networks and a time sequenced traffic control system in Ghana, traffic congestion is a major problem during peak times of the day. In cases of emergencies where a route needs to be created for emergency vehicles, there is no control system in place to safely prioritize their movements at intersections.

In this project, a traffic management system was designed and built as a way of improving traffic flow by decreasing the waiting time experienced by users on the road. This system is aimed at providing a modular and efficient way of controlling traffic based on a condition-based algorithm. The system also enables an administrator to control the traffic junctions remotely to meet different emergency situations.

Table of Contents

Table of Figures

List of Tables

Chapter 1: Introduction

1.1 Background

The current world population of 7.6 billion is expected to reach 8.6 billion in 2030, 9.8 billion in 2050, and 11.2 billion in 2100 [1]. According to the United Nations [1], 83 million people are being added to the world's population every year, and the upward trend in population size is expected to continue even if fertility levels continue to decline. The world population is increasing at an exponential rate. This growth brings an increase in the number of cars using roads and other public amenities. The increase in the number of cars on the roads leads to traffic congestion due to several reasons which may include but not limited to, a poor road network, inefficient traffic control systems, and many others. Traffic jam or congestion has been a problem in urban areas all over the world. This problem is especially so in countries which have high population densities and a high number of road users in their cities [2]. Developing countries like Ghana face these challenges in urban areas as people migrate in masses for jobs. The automobile industry is also developing at a rapid rate, and new automobiles are continuously being pushed onto the market and roads. The high number of automobiles contribute to traffic congestion in urban areas. Traffic lights are designed to control the movement of vehicles and pedestrians at intersections to regulate the flow of traffic. The traffic light is required to take appropriate actions to determine the sequence of green light from an intersection, and its duration based on the condition or state of the intersection at that time [2].

If the traffic light sequence and its duration can be set based on the condition at the intersections, congestion will be reduced, and human activity of people in the city would be impacted effectively and efficiently [2]. Traffic systems which control traffic based on a timed sequence do not adequately handle traffic congestion well. This usually leads to long waiting

times at the traffic intersections. In an era of increasing technology and Internet of Things devices, traffic systems can be more complex, based on machine learning algorithms [2], [3], and other algorithms and control mechanisms to handle traffic on a situation-based level rather than a time-based sequence control.

1.2 Problem Definition

Traffic management is a key issue which can greatly affect productivity and can pose a health hazard in the country. Traffic congestions are a major problem space for a lot of countries and are more prevalent in developing countries whose infrastructure are underdeveloped. During peak times of the day, a greater population of the country would be on the move for various reasons; work, school, transportation of goods and services, amongst others. Having to be delayed for hours in traffic poses a threat to productivity: workers may get to work late; students may arrive at school late. The delivery of goods and services to customers are delayed. There is an entire ripple chain of effects caused by long waiting times in traffic. In harsh weather conditions such as a very sunny day or a heavy downpour, persons using the roads are exposed to different health hazards.

In Ghana's capital Accra, all the traffic lights which order the flow of traffic are sequenced based on fixed time control. The system does not pay attention to the conditions at intersections and or other connecting intersections. The traffic lights make decisions sequentially and are totally time-based. These do not control traffic effectively. In cases where there are emergencies which require routes to be created for emergency vehicles such as those of government personnel, medical, security, etc., there is no control system in place to safely prioritize their movement at intersections. In Ghana, traffic junctions do not have any remote controlling feature to alter traffic to suit changing conditions at road intersections. This concern establishes

the need to design and build a traffic management system to address the inefficiencies of the traffic system in Ghana. The aim of this project is to design and develop a traffic management system that will improve the waiting times in traffic situations and effectively enhance the creation of routes by a control station to better manage emergencies in the country at large. To achieve this aim, the objectives of the project are presented in the next section.

1.3 Objectives of the Project Work

- **Development of a car recognition system at traffic intersections**
    - This objective would guide the development of the hardware component of the traffic management system that will have cameras to determine the number of cars at each intersection.
    - Part of this hardware component would also control the traffic lights; determining when the light should change and for how long.
- **Development of a web control panel**
    - This objective would guide the design of the web application which the administrator would have access to control the different intersections in real-time.
    - The web panel would also have a model view of the intersections being monitored in real-time to see the traffic situations at the intersections in real-time.
- **Well integrated software and hardware system for the traffic control system**
    - This objective would ensure that the software and the hardware components of the traffic management system interface seamlessly.

1.4 Summary

Developing countries have been cited as countries with very high population growth rates [1]. Ghana falls within this bracket of countries. The increase in population coupled with the bad road network in Ghana, have rendered the current traffic systems in Ghana very ineffective during certain peak hours. This concern motivates the objectives of this project highlighted in the section above. The remaining chapters discuss the methodology, design, implementation, and testing of the traffic management system which was developed in this project.

Chapter 2: Literature Review and Related Work

There are countries who have made some advances with smart cities, solar-powered homes and facilities and intelligent traffic management systems [2] – [5]. Several approaches have been utilized to develop efficient traffic management systems which prevent traffic congestion and enable efficient traffic flow. This chapter highlights three approaches which have been used in building traffic systems; reinforcement learning algorithms, fuzzy control systems, and the Mobile Intelligent Traffic Control System (MITCS). Excerpts from these approaches have impacted the design of the traffic management system in this project.

2.1 Reinforcement Learning – Collaborative Algorithms

In Indonesia, [2] identified that the main problem in traffic control was the traffic light having to make decisions sequentially. Due to the high population density of the country, traffic congestion during rush hours is an everyday hustle. The paper [2] developed a machine learning approach to create a solution for the problem. It used Collaborative Q-Learning Algorithms to develop an intelligent traffic control system. After multiple simulations and experiments, the system proved to be very efficient having an average waiting time of 54.67 seconds. [2] discusses the representation of traffic in the real world in three different levels; an agent, a state and an action. An agent represents an intersection with four connected roads, with each road having two lanes with two directions. The state represents the traffic density in the four lanes connected to an intersection. The research categorizes the traffic density into percentages of different ranges. The action represents the lane which will get the green light on. The system described in the paper models a single intersection with reinforcement learning.

The best parameter obtained through experimentation for Collaborative Q-Learning Algorithms method is a learning rate value of 1, discount factor value of 0.8 and immediate

reward method is the results of reduction between waiting time before and waiting time in the current cycle.

2.2 Fuzzy Control System

Fuzzy Control makes use of heuristic information to construct nonlinear controllers for different control applications [7]. Regardless of the source of the heuristic control knowledge, fuzzy control provides a proper formalism for implementing ideas to achieve high-performance control. Fuzzy systems are based on fuzzy logic – a mathematical system that analyses analog input values that take on continuous values between 0 and 1 [7]. The research paper [4], proposed the use of fuzzy traffic lights controller at a complex traffic junction in the middle of Kuala Lumpur City, Malaysia. The proposed fuzzy traffic lights controller can communicate with neighbor junctions and manages phase sequences and phase lengths adaptively.

The research compares the fuzzy traffic lights controller with two other existing controllers in Malaysia; the preset-cycle time and the vehicle-actuated controllers. The design of the fuzzy traffic lights controller had two main features; reduction of the total delay time of waiting vehicles and the synchronization of the local traffic controller with its neighbors, such as controlling outgoing vehicles into neighboring traffic controllers. Under various traffic conditions, the fuzzy controllers were tested against the two existing controllers. The proposed fuzzy traffic lights controller [4] were designed with the capability to determine the capacity of each link, synchronize with neighbor traffic lights controllers, consist of multiple phase sequence, and the capability to observe traffic conditions continuously and re-evaluate at every sample and the capability to consider the pedestrian crossings. A case study using the complex junction in the middle of the city of Kuala Lumpur provided a real traffic scenario for more

robust tests. The research concluded that the fuzzy traffic lights controller can be practically designed and implemented with significant advantages.

Intelligent Traffic Systems (ITS) [8] also make use of fuzzy controllers. In [8], a new hybrid model is developed based on fuzzy controllers for traffic flow forecasting. This model is presented from a robustness point of view. Data obtained from traffic junctions may be tainted for several reasons; faulty sensors, different climate conditions, and accident occurrences on the road. The model developed on fuzzy controllers for short-term and long-term traffic flow forecasting at 15-minute intervals. Based on non-holiday data captured from sensors from highways in the U.S. state of Minnesota, the fuzzy controller I selects the model with the least forecasting error [8] which is used as the basis of prediction.

2.3 Mobile Intelligent Traffic Control System (MITCS)

The Mobile Intelligent Traffic Control System is a traffic control framework [5] which integrates micro-mechanical and electrical technologies with embedded systems, wireless transmission, image processing, and a solar power module. In [5], the system identifies closed-circuit televisions which monitor traffic approaching the intersection. The MITCS evaluates the traffic density and determines the control strategy for the immediate traffic condition. Figure 2.1 indicates that the controllers at each junction can identify traffic conditions and coordinate other intersections for better control.

The wireless access points on each intersection make communication possible between the intersections. Figure 2.2 shows that the MITCS system can integrate and interconnect several traffic groups for harmonious control in a large-scale network.

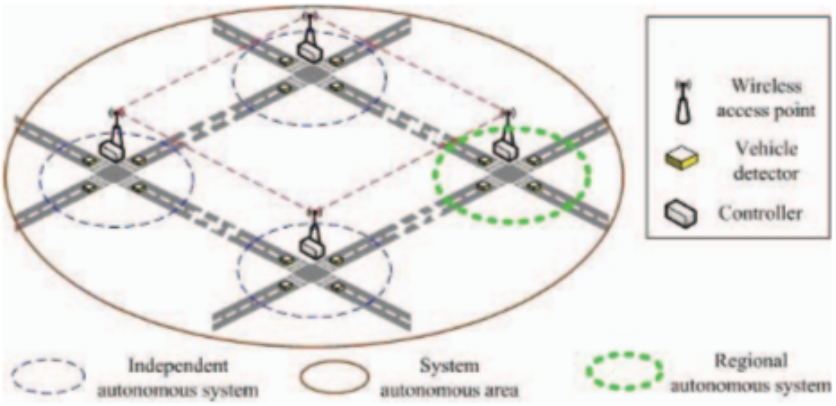

Figure 2.1 Autonomous group control [5]

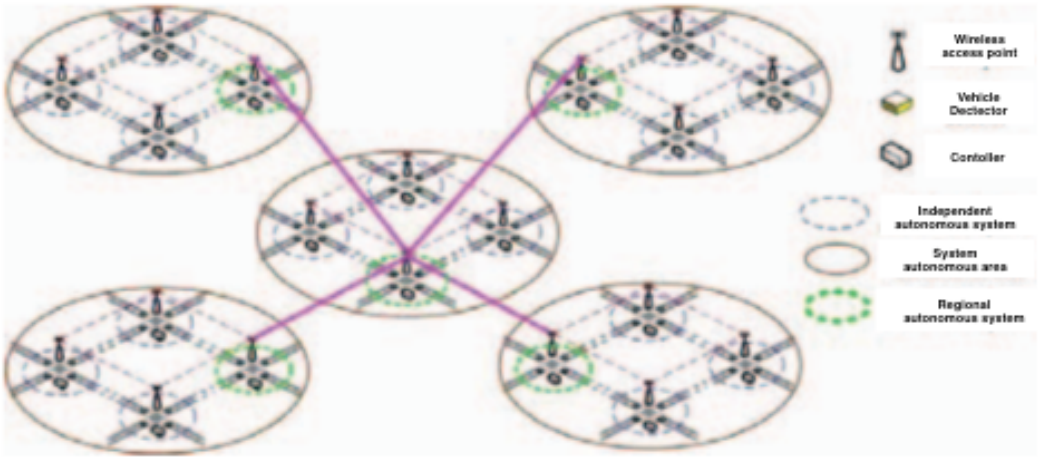

Figure 2.2 Autonomous area control [5]

The paper [5] proposes a comprehensive framework which interconnects closed-circuit television, changeable message sign, vehicle detector, and traffic lights to autonomously manage the traffic approaching an intersection.

2.3 Literature Review Impact on Project

The Mobile Intelligent Traffic Control System [5] informed the traffic management system development with regards to the software and hardware components of the system. The traffic management system (discussed into detail in Chapter 4) was designed based on the framework [5]. The model of 4 junction lanes and a lane state were concepts used in the development of the model used in the traffic management system.

Chapter 3: Requirements Specification

This chapter highlights the requirements documentation that informed the system design and testing. Through interviews and excerpts from Chapter 2, this chapter identifies and documents the user requirements, system/function requirements and non-functional requirements of the system.

3.1 Requirements Specification

This project has two major sufficiency requirements to be met; the user and the system requirements. To determine both requirements adequately, research was conducted in the form of interviews with experts in the Internet of Things (IoT) technology, software engineers, and some professionals with experience with traffic control systems. These interviews informed the user requirements and system requirements of the traffic management system design. A questionnaire (Appendix A) was used to gather data during the interviews. Research works and publications about traffic and street light management systems in smart cities [6] also informed the requirements for the system.

3.1.1 User Requirements

The user requirements of the traffic management system include a typical user's expectation of how the system is supposed to operate, and how it's expected to interact with the user. From the various research approaches mentioned, the following user requirements were identified. The user refers to the individual car owners who may spend time in traffic, the administrator of a traffic control center, and a pedestrian who may cross the road. The user expects the system to

1. Effect little or no waiting time in traffic situations

2. Have a low-cost implementation
3. Have a central control center
4. Be safe
5. Be self-sustaining and autonomous

3.1.2 System Requirements

The system requirements serve as the technical baseline upon which the user's requirements are met. Considering the user requirements identified, the system requirements identify the necessary technical features of the traffic management system needed to ensure a good synchronization between the system and the user. The system refers to the entirety of the traffic management system. The following make up the system requirements. The system must

1. Have sensor devices which collect traffic data at each junction. These devices must have high accuracies with low response times to enable efficient real-time data collection.
2. Recognize and identify the traffic situation from sensor inputs. With the identified traffic situation per intersection, informed decisions must be made to efficiently route traffic.
3. Have a web application with an efficient server-side environment which interacts with the database, sends traffic signals to intersections, and update the traffic junction model on the front-end.
4. Have a web application with an efficient front-end environment which is constantly being updated with the identified traffic situation per intersection.
5. Scalable and expandable to serve multiple intersections.
6. Enable the intersections to be able to communicate with each other.
7. Work independently of human interaction.

8. Enable administrator control of traffic junctions in cases of emergencies

3.1.3 Non-Functional Requirements

1. The system must be reliable
2. The system must provide accurate information
3. The system must be secure and require administrative credentials to log in
4. The system must provide up-to-date information

3.2 Scope of Project

This project will consist mainly of two parts: firstly, a web-based client side that would enable an administrator to log in and control traffic junctions remotely. The web application would run a traffic routing algorithm on the backend to send control signals to the traffic junctions. The web application would have a real-time view of the traffic intersections with traffic information. Secondly, a hardware traffic junction unit which would detect the number of cars on each lane of a junction and update a database for each junction.

Chapter 4: Design and Implementation

4.1 System Design

The proposed solution for the traffic management system factors in both the user requirements and system requirements identified. The traffic management system is made up of two core components; the hardware component and the software component. Each of these components has specific features tailored to fit the key functioning of the traffic management system.

4.1.1 System Design Architecture

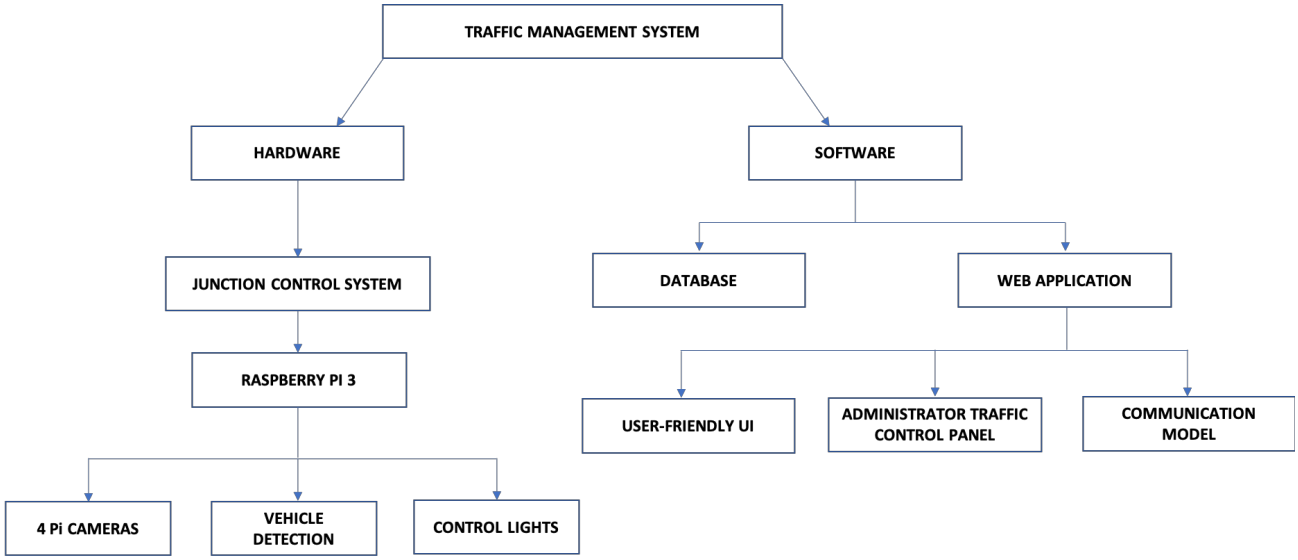

Figure 4.1 Traffic Management System Design Architecture

The system design architecture seen in Figure 4.1 identifies the two main components that make up the traffic management system. The hardware component makes up the interface which operates at a traffic intersection. The software component makes up the part of the system that will run the web application which processes traffic density data, creates a real-time modeled view of traffic intersections and instructs the hardware component on which traffic signals to give.

4.1.2 System Hardware Component

In the system, each intersection would have a hardware component. The hardware component comprises of the raspberry pi 3, which interfaces with four cameras and the traffic control lights for each interconnecting lanes. Figure 4.2 highlights the architectural view of the hardware component of a traffic junction unit. For each lane at the intersection, the camera is required to record short videos of each intersecting lane. The raspberry pi processes the videos from the camera and determines the traffic density on the said lane. The traffic density of a lane represents the number of cars on the lane. The traffic density data is sent to a database on the server where the software component of the system makes use of the data. The raspberry pi 3 receives instructions from the software component on what signal it should pass to the traffic control lights.

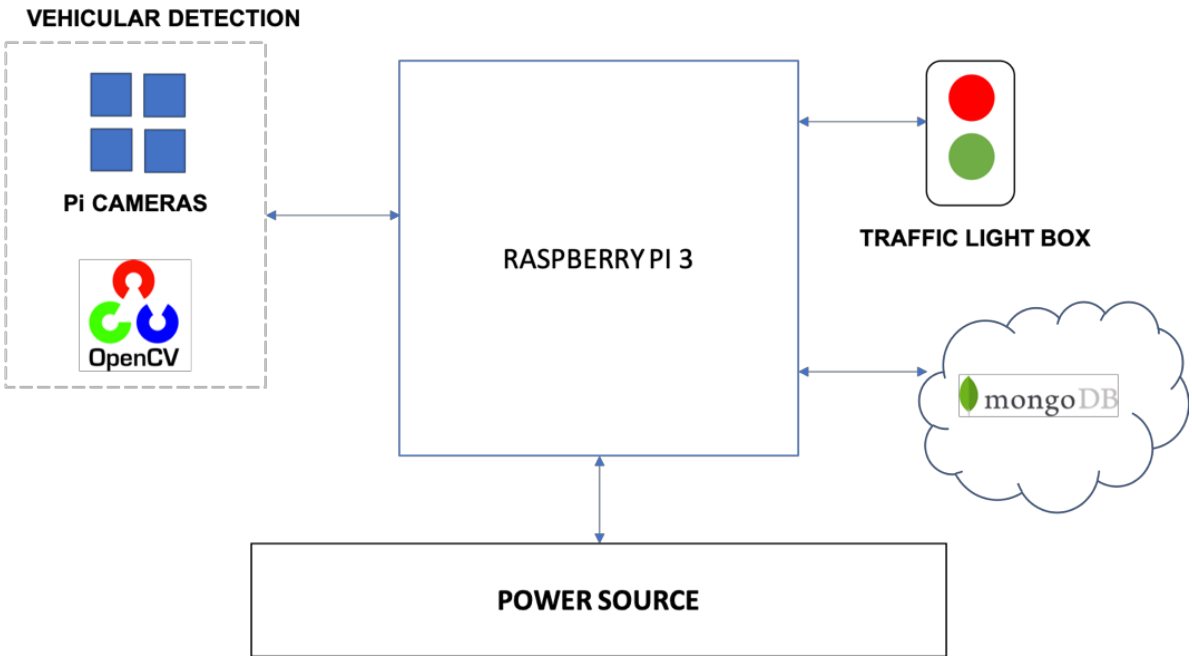

Figure 4.1 Hardware Architectural Component at a traffic intersection

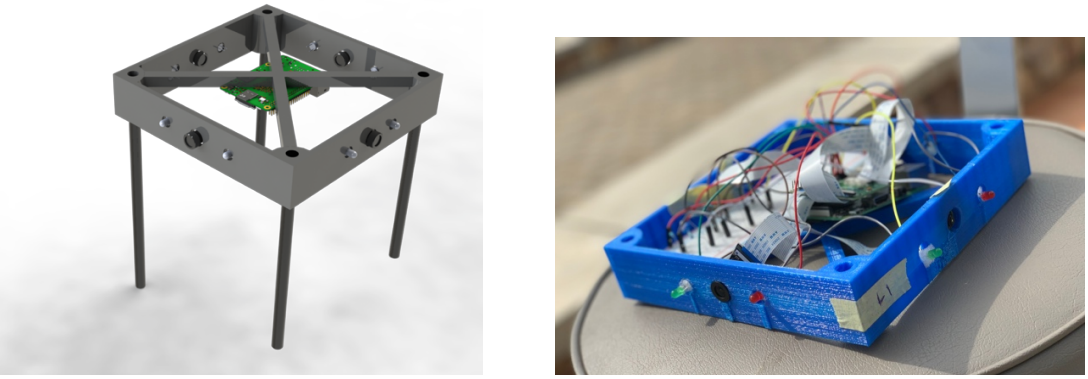

Figure 4.2 3D model (Left) and built model (right) of the traffic junction unit

4.1.2.1 Vehicular Detection

The raspberry pi 3 runs a vehicle detection script [10] with the OpenCV image processing library to detect cars from videos. The script makes use of a HAAR classifier which has been trained over 100 videos with moving cars [10]. The four pi cameras are set up on a 3D printed frame (seen in Figure 4.3). Each camera points in the direction of a traffic lane. With the aid of a camera multiplexer, the raspberry pi records three-second videos of each lane. The multiplexer allows only one recording to be taken at a time. These videos serve as inputs to the vehicle detection function. The vehicle detection script runs a while loops till the end of the video. For every frame of change identified by the classifier, a rectangular figure is drawn. Ideally, a rectangular drawing represents an object identified as a vehicle by the HAAR classifier. Thus, keeping count of the number of rectangles drawn throughout the video is used to represent the number of cars present in the video.

4.1.2.2 Traffic Control Indicators

The raspberry pi 3 controls the red and green indicator lights at the traffic junction. Red and green LED lights are used to represent the traffic indicator lights. A typical traffic indicator has the red,

amber and green lights. However, this project makes use of red and green indicator lights to simplify the junction control component.

4.1.3 System Software Component

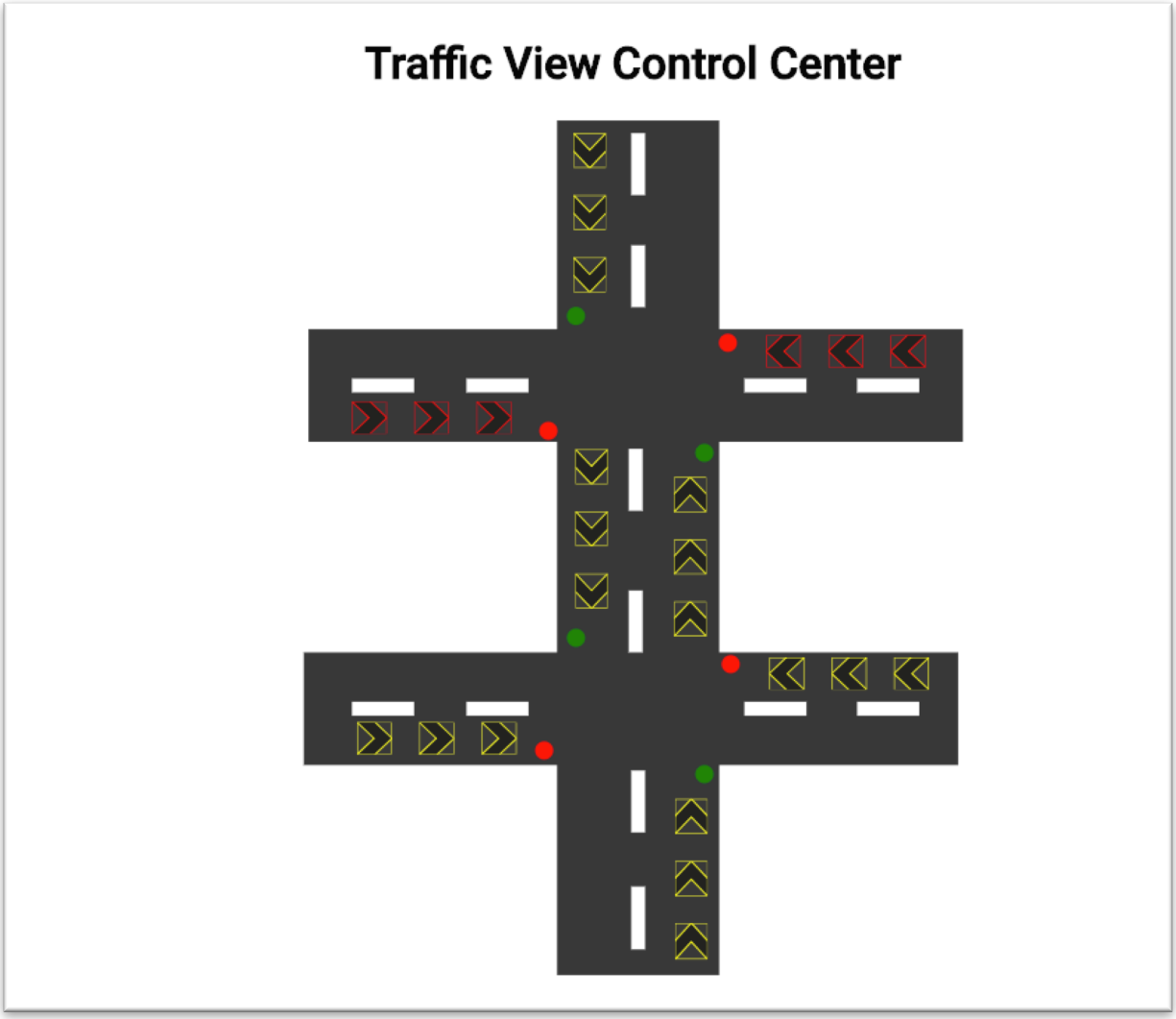

Figure 4.4 Two intersections with four connecting lanes

The software component of the project is a MEAN (MongoDB, Express, Angular 7, NodeJS) stack application. MongoDB is a non-relational database which stores data in documents with JSON data-like structured. MongoDB enables fast queries to be made to the database, making it an ideal option for the traffic view component interaction of the web

application. The traffic view component is discussed further in this chapter. [1]Express is a routing and middleware web framework that has minimal functionality of its own. However, Express enables efficient routing in the NodeJS backend. [2]NodeJS is a server-side JavaScript runtime that executes JavaScript code outside the browser. NodeJS and Express enable efficient handling of the backend processes of the web application. Angular is used for front-end development. Its modular component structure makes it ideal and efficient for building the web application interface.

The software component of the system is a web application which performs three major functions;

- It processes traffic density data received from the hardware component
- It creates a real-time modeled view of traffic intersections based on traffic density data (Figure 4.4)
- It instructs the hardware component on which traffic signals to give using the administrator control board seen in Figure 4.7

Figure 4.4 shows a real-time modeled view of traffic intersections based on the traffic density data received from the hardware component. The traffic view component (Figure 4.4) polls every 30 seconds to retrieve traffic data from the MongoDB documents and updates the color of the lanes accordingly. Each lane is colored based on the traffic density; yellow represents mild traffic and red represents dense traffic. The web application of the system

[1] https://expressjs.com/en/guide/using-middleware.html
[2] https://nodejs.org

enables two distinct modes of operation; the pre-emptive control mode (also known as the administrator control mode) and an auto-mode.

The pre-emptive control mode allows an administrator to manually take control of the system and control the traffic lights at the junction. This mode enables emergencies to be handled more effectively. It also allows routes for dignitaries and government officials to be planned and controlled more effectively.

The auto-mode employs the use of a traffic routing algorithm to process traffic density data and control the traffic lights at an intersection. The algorithm runs on the server environment of the web application and sends signals to the raspberry pi to control the lights at an intersection. There are two algorithms which will be run based on the traffic state; a timed-sequence algorithm and a greedy algorithm. The timed-sequence algorithm runs when there is little or no traffic at the intersections. The greedy algorithm is run when there is dense traffic at the intersections.

Figure 4.5 shows the architecture of the MEAN stack web application. The front-end or client side of the application is made up of the User Interface (UI) which has the modeled-view of the traffic junction and the administrator sign-in page. The server side of the web application is run by Node JS and makes use of MongoDB for database related operations. The server-side environment handles the administrator login authentication. It also runs the routing algorithm which sends control signals to the hardware component at an intersection.

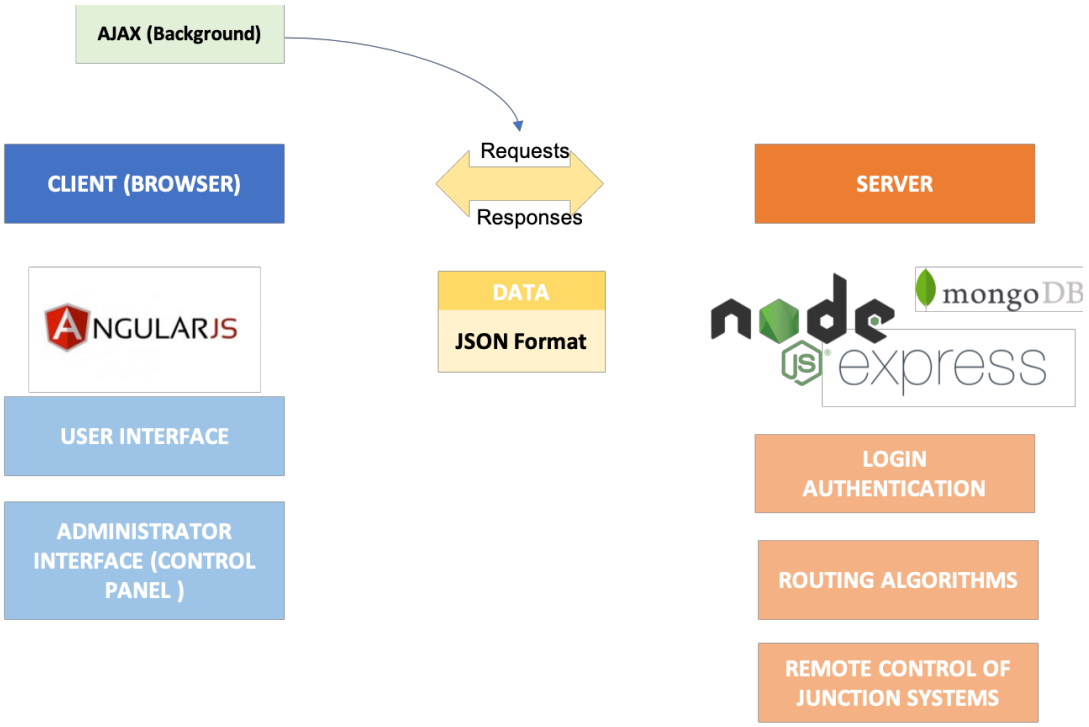

Figure 4.5 MEAN Stack web application architecture

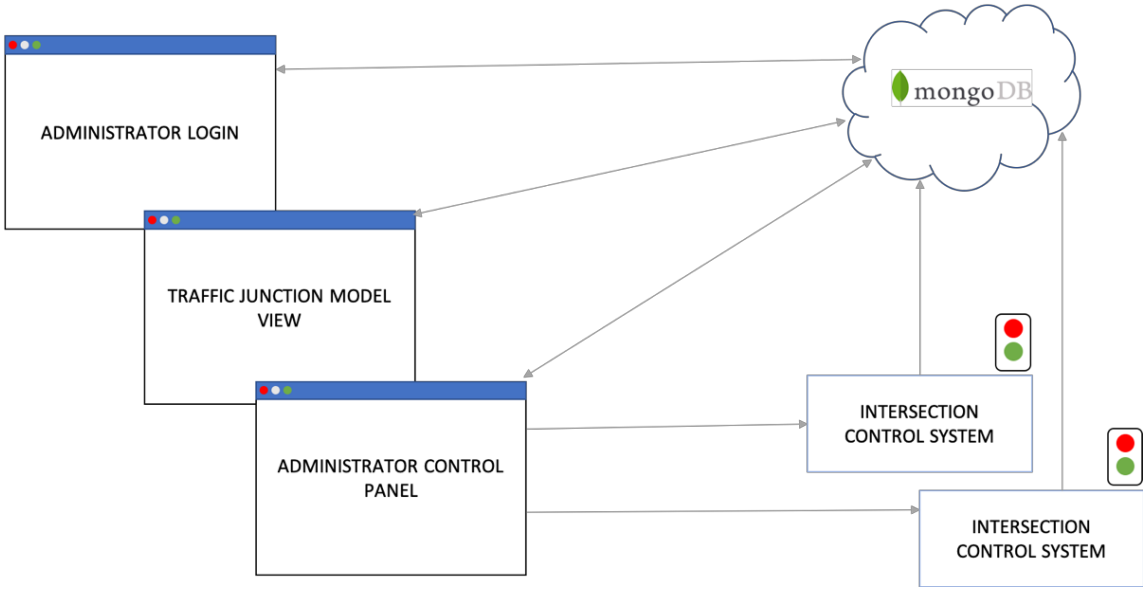

Figure 4.6 Web Application Interaction with Database and Intersection Control System

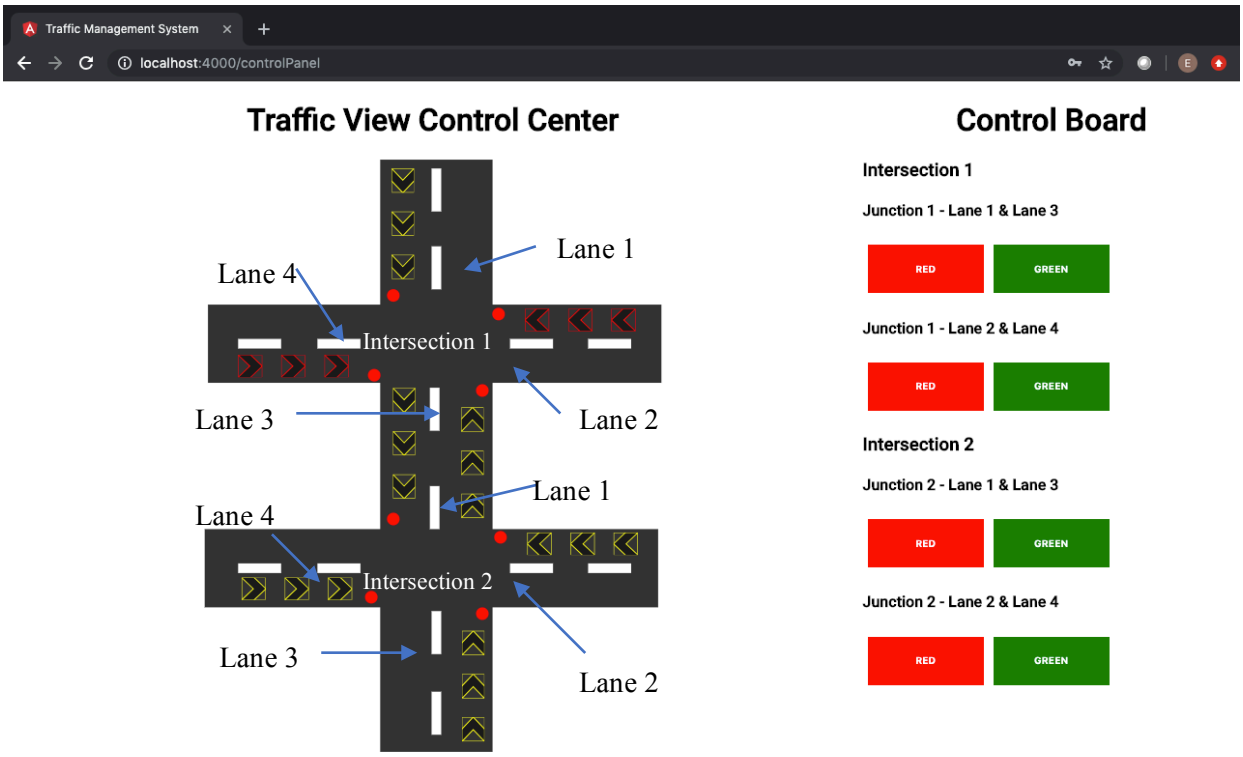

Figure 4.7 Administrator Control Page

Figure 4.6 shows how the different components of the web application interact with the database and the intersection control system. The administrator log-in component accesses the database and performs a login authentication to allow or restrict access to the administrator control panel. The log-in component interacts mainly with the database. The traffic junction model view in Figure 4.6 is seen in explicitly in Figure 4.4. This component renders the view of the traffic intersection in real-time to show the state of the traffic junction (yellow – mild traffic; red – dense traffic). The traffic junction model view interacts mainly with the MongoDB documents. Figure 4.7 represents the Administrator Control Panel page in Figure 4.6. The administrator control panel enables the system's administrator to manually control the traffic lights at an intersection. It interacts with the database and the intersection control system. The intersection control system (Figure 4.6) represents the traffic control unit (Figure 4.3) at a traffic

intersection. The intersection control system updates the MongoDB documents with traffic density data and receives control instructions from the web application.

4.1.3.1 Traffic Control Routing Algorithm

The traffic routing algorithm is based on a greedy and a round-robin algorithm and was built in Python 3 but is run in the Node JS server-side environment. The algorithm has a function that makes HTTP requests to the MongoDB Cloud to update and access traffic data at each junction. A function **HighTrafficLane**() was written to get the lanes with the greatest traffic density at both intersections given the database connection object as a parameter. The function **updateMongo**() was written to update the MongoDB documents given a collection (the intersection being referred), the lane and the new traffic density as parameters. The function **reduceTraffic**() takes the database connection object, collection, the lane and a number as parameters. The **reduceTraffic**() function deducts the number parameter from the traffic density of the specified lane of the junction collection. The function also sends a green signal to the specified lane on the traffic hardware junction component.

The routing algorithm makes use of the functions mentioned above to regulate traffic flow. The table below provides the pseudo code upon which the algorithm was developed.

Table 4.1 Pseudo Code of Traffic Routing Algorithm

```
                    ALGORITHM
1. Connect to MongoDB
2. Scan all the lanes at Intersection 1 and Intersection 2
3. Return the lane with the greatest traffic density for both
   intersections
```

4. Run the greedy algorithm which calls **reduceTraffic**() on the lanes with the greatest traffic density
5. Wait for 30 seconds then perform 2,3 and 4 again
6. Run the round robin algorithm which calls the **reduceTraffic**() function on all the lanes for both intersections

4.2 Design Components

In building the hardware component of the traffic management system, four main components were used. These include the raspberry pi 3, pi camera modules, camera multiplexer and LED lights.

Raspberry pi 3

Figure 4.8 Raspberry Pi 3[3]

The raspberry pi is a low-cost capable little device with 1.4 GHz 64-bit quad-core processor, dual-band wireless LAN, Bluetooth and faster Ethernet [9]. This device allows

[3] https://www.raspberrypi.org

connections to a standard sized keyboard as well as a mouse and a monitor. It mainly allows languages like Scratch and Python and is widely used all around the world. The raspberry pi 3 is used over the Arduino which is a micro-controller motherboard. This is because the Arduino is only capable of running one program at a time, repeatedly whereas the raspberry pi is a general-purpose computer which can run multiple programs at the same time. The traffic management system will require the intersection control system to interact dynamically with the web application, and this requires the processor in the intersection control system to run multiple programs.

Pi Camera Module

The Pi camera module is a raspberry pi accessory that enables the raspberry pi to capture images and videos.

In determining the accessory to use to aid in counting the number of cars on a lane at an intersection, two devices were considered; the Pi camera module and the ultrasonic level sensor. The Pi camera was used over the ultrasonic sensor.

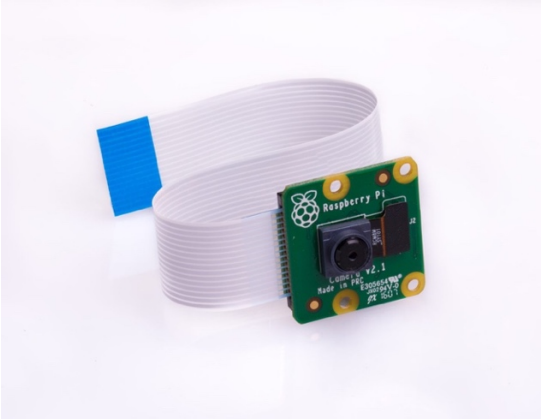

Figure 4.9 Raspberry pi 3[4] camera module

[4] https://www.raspberrypi.org

This is because the ultrasonic sensor would not be able to determine the number of cars in a queue on a lane. It would be able to determine just the first car in the queue. The Pi camera module however with the help of the OpenCV library can determine the number of cars in a queue on the lane.

ARDUCAM Camera Multiplexer

The ARDUCAM camera multiplexer makes use of the GPIO pins of raspberry pi 3 to allow four cameras to be connected to a single raspberry pi. Using pin configurations, the raspberry pi can switch between the cameras to activate and record videos or captured images. The camera multiplexer is by the junction control unit to switch between lanes and record videos of the traffic lanes.

Figure 4.10 The ARDUCAM Camera Multiplexer[5]

[5] http://www.arducam.com/multi-camera-adapter-module-raspberry-pi/

LED lights

In modeling the intersection control system, Light Emitting Diodes (LEDs) are used to represents the traffic lights indicators.

Figure 4.11 LED Lights[6]

Red and green lights are used to represent the red light and the green light of a traffic intersection respectively. The yellow light of a normal traffic light was exempted in modeling the traffic intersection for simplicity.

[6] https://activedark.com/2012/10/25/fun-facts-what-are-l-e-d-lights/

Chapter 5 – Testing & Results

From the requirements identified in section 3.1, the traffic management system has features which address several of the requirements.

The following features meet the user requirements identified in section 3.1.1.

- ✓ The system employs a traffic routing algorithm which runs on a server to control traffic junctions remotely.
- ✓ The system has a central control center where the administrator can control the traffic junctions and or determine which mode the system should run in.
- ✓ The system employs fault handling measures to ensure collisions are avoided in the controlling of the traffic lanes.
- ✓ The auto-mode of the system allows the traffic management system to operate autonomously and control traffic smartly based on data about the state of the traffic junction.

The following features meet the system and non-functional requirements identified in section 3.1.2 and 3.1.3 respectively.

- ✓ The Traffic Management System uses Pi cameras as its input sensors.
- ✓ The Traffic Management System uses the OpenCV image processing library to determine the number of cars per junction and updates a MongoDB collection with the data.
- ✓ The web application of the system has a traffic view component which is constantly updated in real-time to show the state of the traffic junctions.

- ✓ The web application provides a central point where all data is processed. New junctions can be added to the scope of the system at any time.
- ✓ The Traffic Management System does not allow the different junction control units to interact with each other. However, all the junction control systems are coordinated from the web application to control the traffic junctions either by the administrator or the traffic routing algorithm.
- ✓ The Traffic Management System can operate independently of human interaction when in the auto-mode.
- ✓ The Traffic Management System allows an administrator to control the junction systems remotely.

5.1 Unit Testing

The individual functions and control features of the application were tested with different inputs to ensure they functioned as required.

A user can only get access to the administrator control page after signing in. The sign in units of the application was tested with valid inputs and invalid inputs. The results for this test were a success as the web application only redirected for correct inputs. Upon refreshing the administrator control page (Figure 4.7), the user is redirected back to the home page. The control panel route is protected.

The traffic view component (Figure 4.4) is required to be updated every 30 seconds to identify changes on the lanes of the traffic junctions where red means traffic density greater than 10 and yellow means traffic density less than 10. To test this unit of the web application, the MongoDB documents were updated manually and the changes were observed in the traffic view component.

Clicking the control buttons to change the lights of the traffic junctions were also tested. For a junction, while green was on for lane 1 and lane 3 (Figure 4.7) the green command for lane 2 and 4 was clicked. A subsequent change to all lanes having green lights would have resulted in a fail of the system. The green command did not trigger.

A NodeJS server which runs on the raspberry pi 3 to receive control commands when the system is in administrator mode was also tested. The end-point testing software, Postman, was used to test the server. Wrong inputs and right inputs were issued with the Postman software. The server responded only to valid inputs for control of the indicator lights.

5.2 Component Testing

The component interfaces which are made up of several individual integrated units were tested to discover and fix bugs which occurred from some hardware and software components interactions. This section covers the tests conducted on the vehicular detection at traffic junctions and the efficiency of the traffic routing algorithm.

5.2.1 Vehicular Detection at Traffic Junctions

The traffic management system's efficiency largely relies on the data being provided on the status of the junction. The junction control systems make use of 4 pi cameras and the OpenCV library to detect vehicles. Five-second videos are taken for each traffic lane at a junction and the OpenCV library [10] makes use of motion detection to capture the number of cars present on a lane at a junction.

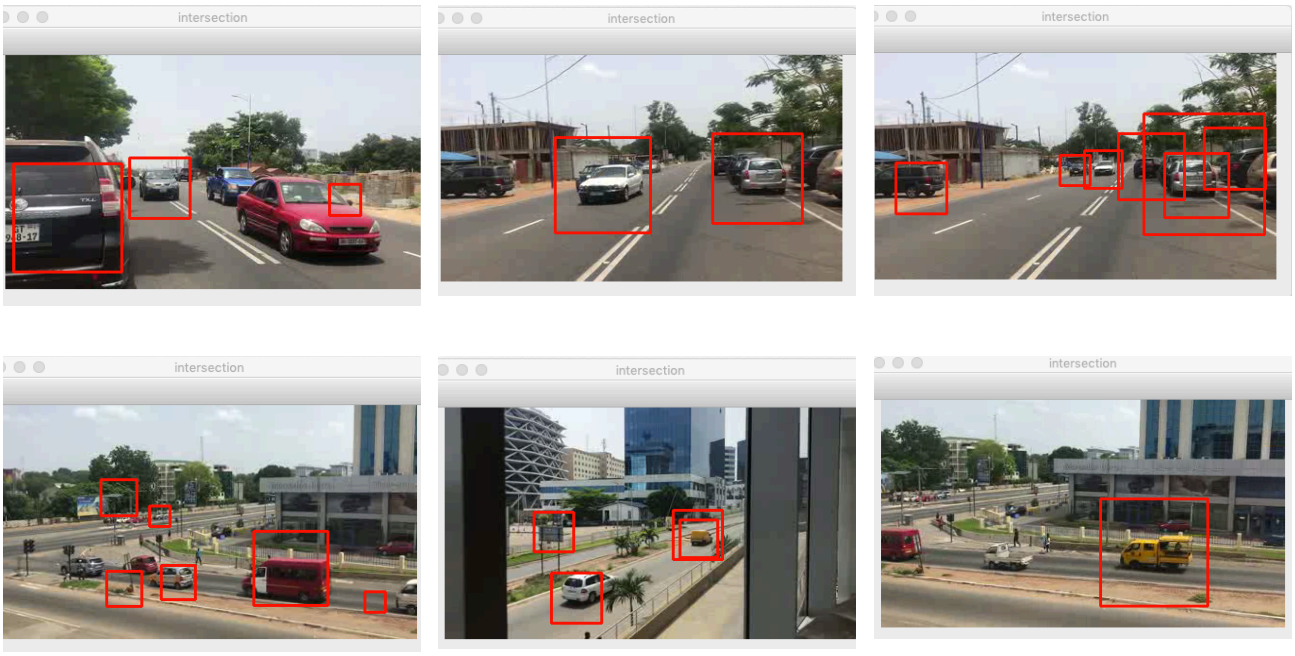

Figure 5.1 Screenshots of Vehicle Detection Script on recordings

To test the efficiency of the vehicle detection script and the OpenCV, short clips of videos were recorded at intersections at Airport City, East Legon and Lapaz in the Greater Accra region. These recordings were made at different times of the day. The short video clips were run through the vehicle detection script. Figure 5.1 shows the vehicle detection script running on some of the recordings. The number of cars seen in the video and the number of cars the script could identify were recorded in Table 5.1 below.

Table 5.2 Table of car count recorded by eye and vehicle detection script

Location	Time of Day	Number of Cars Seen in Recoding	Number of Cars Recorded by Script
East Legon	11:00 GMT	4	35
East Legon	11: 02 GMT	5	55
East Legon	11:04 GMT	3	103
East Legon	11:05 GMT	6	67
East Legon	11:10 GMT	3	201

Airport City	9:00 GMT	7	321
Airport City	9:02 GMT	6	102
Airport City	9:03 GMT	5	21
Airport City	9:05 GMT	4	231
Airport City	10:15 GMT	6	132
Lapaz	20:00 GMT	15	6
Lapaz	20:02 GMT	10	3
Lapaz	20:05 GMT	14	5
Lapaz	20:07 GMT	18	12
Lapaz	20:10 GMT	15	4

Table 5.1 records the number of vehicles identified by the vehicle detection script [10]. From the table, it can be observed that the number of cars observed at a lane in the video differs largely from the number of vehicles identified by the vehicle detection script. Several reasons account for the disparity.

The vehicle detection script makes use of a HAAR classifier to draw rectangles (seen in Figure 5.1) around objects looking like cars in the video. For a single video, it was observed that rectangles were drawn several times for each motion detected. Thus, a single vehicle can be counted several times until the video ends. This was the case observed in the recordings made at East Legon and Airport City.

For the recordings made at Lapaz, it is observed from the table that the vehicles identified by the script were less than the number of vehicles recorded from the videos. At

Lapaz, the videos were taken around 20:00 GMT. It was evening, and as a result, lighting was poor. Thus, the script was unable to detect motion properly from the videos.

These tests indicated that the vehicle detection script best works during the daytime and on a shorter length of videos. Thus, the raspberry pi 3 of the junction control component was configured to record three-second videos of each lane at a junction.

5.2.2 Traffic Routing Algorithm

To test the efficiency of the routing algorithm used in the Traffic Management System, the round robin algorithm which models how the traffic system currently operates in Accra is created as a benchmark. Given certain data in the MongoDB documents, both algorithms were run and their performance metric was measured by the time taken for each lane to reach zero traffic density. Table 5.2 shows the data recorded for each junction lane. The following assumptions were made regarding this test:

- ✓ There was no inflow of traffic per test, that is, the lanes were updated at the start of the algorithms and the time taken for each lane's traffic to reach zero were recorded.
- ✓ Traffic flow is bi-directional such that, opposite lanes would move in opposite directions at a time. Thus, when lane 1 (Figure 4.7) is green, lane 3 would also be green.
- ✓ For a period of 10 seconds, traffic of 5 cars can flow.
- ✓ Traffic outflow from a lane on one intersection does not flow into another lane of another junction.

From the results recorded in Appendix B and C, it can be observed that the routing algorithm used for automating the traffic flow for both junctions outperformed the time sequenced traffic control used in Ghana. Random data was generated to fill the MongoDB

documents of the traffic junctions and both the timed sequenced and traffic routing algorithm was run with the data. Test runs were conducted for each algorithm with random data per run.

For each run, it was observed that the traffic routing algorithm which makes use of both a greedy algorithm and a round robin algorithm performed better than the time sequenced algorithm (Round Robin). For each run, it was observed (Appendix B and C) that the traffic routing algorithm achieves fairness and shorter waiting times, that is, the time taken for the lanes to reach zero traffic density. Fairness here refers to all the traffic lanes spending relatively equal amounts of time (Appendix C). Waiting times of the traffic lanes were greatly improved to half the values observed from tests with the time sequenced algorithm (Appendix B).

5.3 System Testing

As a complete system, the administrator control mode is enabled when an administrator logs into the administrator control board (Figure 4.7). Upon login, all traffic lanes are set to red, awaiting a control signal from the administrator. When a signal doesn't come in 30 seconds, the system switches back to having the traffic routing algorithm control traffic. The system was tested with both the login instance and an instance with no signal over 30 seconds. The system responded appropriately. Control signals were sent from the administrator control board to the junction control units. The responses were fast and prompt. The delay between the control signal being sent and the junction control lights responding was unnoticeable.

The junction's traffic control system updates the MongoDB with traffic data with the aid of the raspberry pi 3 and the cameras. As mentioned in section 5.2.1, the data was consistently greater than the actual number of cars present at the traffic junctions. This brought rise to the real-time traffic view component (Figure 4.4) of the web application not being accurate and representative of the two traffic junctions.

Chapter 6 – Conclusion and Future Works

This chapter presents a summary of this project. It restates the goal of the project and provides a summary of what was implemented. This chapter also highlights some challenges which were faced, limitations of the project and future improvements which can be made to better this project.

6.1 Summary

This project led to the design and development of a Traffic Management System. The traffic management system is a step towards modernization. With the rise of smart cities and ever increasing Internet of Things systems, the Traffic Management System provides an efficient and modular way of managing traffic in the country. The system is modular and allows more junction units to be installed as and when the road network grows. Multiple control stations can be set across the country to manage smaller networks of junctions as well.

6.2 Limitations

As seen in the results there are a few short-comings mainly identified from the vehicular detection component of the system. There are multiple frame changes with regards to motion when the camera records videos. The continuous change in frames of the video because of motion resulted in a misrepresentation of the amount of traffic present at the junction.

The raspberry pi runs a NodeJS server which listens to a port for traffic control signals. Each time the raspberry pi connects to a new network, the IP address of the raspberry pi changes. This makes the initial configuration of the system cumbersome.

6.3 Future Works

This project accomplished its goal of building and implementing a real-time web application and traffic junction control units which make up the traffic management system collectively. While this system works quite well, further improvements can be made with regards to the way of detecting vehicles at a junction. Preceding intersections could have entry markers which would count the cars to provide the database with more accurate data. Alternatively, the HAAR classifier used in the vehicle detection script [10] could be trained on more models and on pictures to be able to identify the traffic density more effectively. The Google Maps API [11] can be implemented into the traffic management system to scale the map and plan routes better from the administer control board (Figure 4.7).

In conclusion, the traffic management system built provides a modular and efficient way for managing traffic. The auto-mode of the system presents an intelligent way of controlling traffic flow as compared to the timed sequence of traffic control. The administrator mode also provides a means of controlling the traffic intersection manually from a central location. Entirely, the traffic management system provides a modular way for scaling and increasing the traffic network to be managed.

References

[1] World population projected to reach 9.8 billion in 2050, and 11.2 billion in 2100 | UN DESA | United Nations Department of Economic and Social Affairs. (2017). Retrieved from https://www.un.org/development/desa/en/news/population/world-population-prospects-2017.html

[2] A. R. Rosyadi, T. A. B. Wirayuda and S. Al-Faraby, "Intelligent traffic light control using collaborative Q-Learning algorithms," *2016 4th International Conference on Information and Communication Technology (ICoICT)*, Bandung, 2016, pp. 1-6. doi: 10.1109/ICoICT.2016.7571925

[3] K. Mahata, "Queen bee generation method in hive and its application in sensor networks," *2012 International Conference on Collaboration Technologies and Systems (CTS)*, Denver, CO, 2012, pp. 609-612. doi: 10.1109/CTS.2012.6261114

[4] M. Khalid, See Chin Liang and R. Yusof, "Control of a complex traffic junction using fuzzy inference," *2004 5th Asian Control Conference (IEEE Cat. No.04EX904)*, Melbourne, Victoria, Australia, 2004, pp. 1544-1551 Vol.3.

[5] L. Lin, H. Huang, J. Lin and F. F. Young, "A new intelligent traffic control system for Taiwan," *2009 9th International Conference on Intelligent Transport Systems Telecommunications, (ITST)*, Lille, 2009, pp. 138-142. doi: 10.1109/ITST.2009.5399369

[6] Thomas Novak, Klaus Pollhammer, Heimo Zeilinger, Samer Schaat, "Intelligent streetlight management in a smart city", *Emerging Technology and Factory Automation (ETFA) 2014 IEEE*, pp. 1-8, 2014.

[7] Passino, K. M., Yurkovich, S., & Reinfrank, M. (1998). *Fuzzy control* (Vol. 42, pp. 15-21). Menlo Park, CA: Addison-wesley.

[8] Hosseini, S. M. Hadi & Shabanian, Mahdieh. (2018). A New Hybrid Intelligent Approach for Traffic Flow Forecasting based on Fuzzy Controllers. 10.1109/IECON.2018.8591398.

[9] What is a Raspberry Pi?. Retrieved from https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/

[10] Andrews, O. (2016). Vehicle Detection with Haar Cascades with OpenCV. GitHub. https://github.com/andrewssobral/vehicle_detection_haarcascades

[11] Google Maps Directions API. (2018). Google Developers. Retrieved 18 April 2018, from https://developers.google.com/maps/documentation/directions/intro

Appendix

Appendix A – Interview Questionnaire used for Interview Research

Intelligent Traffic Light System Requirements Interview Research

Name:

Department:

What do you look out for when designing intelligent systems?

What factors should be considered when designing an intelligent traffic system?

Top 3 requirements an intelligent traffic system should have ?

Appendix B – Time Sequence Algorithm Test Results

Run 1					
Junction 1	Initial Traffic Count	Time to Zero (seconds)	Junction 2	Initial Traffic Count	Time to Zero (seconds)
Lane 1	**30**	**112.18**	Lane 1	**50**	**176.07**
Lane 2	**40**	**145.98**	Lane 2	**67**	**222.79**
Lane 3	**33**	**130.54**	Lane 3	**45**	**160.12**
Lane 4	**43**	**160.12**	Lane 4	**32**	**130.54**
Run 2					
Junction 1	Initial Traffic Count	Time to Zero (seconds)	Junction 2	Initial Traffic Count	Time to Zero (seconds)
Lane 1	**63**	**204.99**	Lane 1	**42**	**145.18**
Lane 2	**70**	**215.43**	Lane 2	**55**	**175.51**
Lane 3	**32**	**115.11**	Lane 3	**22**	**84.57**
Lane 4	**63**	**204.99**	Lane 4	**11**	**55.48**
Run 3					
Junction 1	Initial Traffic Count	Time to Zero (seconds)	Junction 2	Initial Traffic Count	Time to Zero (seconds)
Lane 1	**33**	**136.91**	Lane 1	**34**	**136.91**
Lane 2	**142**	**476.94**	Lane 2	**12**	**71.03**
Lane 3	**130**	**430.24**	Lane 3	**21**	**101.59**
Lane 4	**23**	**101.59**	Lane 4	**32**	**136.92**
Run 4					
Junction 1	Initial Traffic Count	Time to Zero (seconds)	Junction 2	Initial Traffic Count	Time to Zero (seconds)

Lane 1	**32**	**153.69**	Lane 1	**321**	**1056.85**
Lane 2	**22**	**115.62**	Lane 2	**100**	**369.95**
Lane 3	**230**	**840.59**	Lane 3	**210**	**779.91**
Lane 4	**231**	**855.67**	Lane 4	**210**	**779.14**
Run 5					
Junction 1	Initial Traffic Count	Time to Zero (seconds)	Junction 2	Initial Traffic Count	Time to Zero (seconds)
Lane 1	**100**	**309.88**	Lane 1	**33**	**125.**
Lane 2	**32**	**125.15**	Lane 2	**44**	**155.10**
Lane 3	**80**	**255.42**	Lane 3	**32**	**125.15**
Lane 4	**29**	**110.55**	Lane 4	**31**	**125.15**

Appendix C – Traffic Routing Algorithm Test Results

Run 1					
Junction 1	Initial Traffic Count	Time to Zero (seconds)	Junction 2	Initial Traffic Count	Time to Zero (seconds)
Lane 1	**30**	**70.57**	Lane 1	**50**	**120.14**
Lane 2	**40**	**100.72**	Lane 2	**67**	**120.14**
Lane 3	**33**	**100.72**	Lane 3	**45**	**120.14**
Lane 4	**43**	**100.72**	Lane 4	**32**	**70.57**
Run 2					
Junction 1	Initial Traffic Count	Time to Zero (seconds)	Junction 2	Initial Traffic Count	Time to Zero (seconds)
Lane 1	**63**	**124.25**	Lane 1	**42**	**105.06**
Lane 2	**70**	**124.25**	Lane 2	**55**	**86.19**
Lane 3	**32**	**86.18**	Lane 3	**22**	**86.19**
Lane 4	**63**	**124.25**	Lane 4	**11**	**40.51**
Run 3					
Junction 1	Initial Traffic Count	Time to Zero (seconds)	Junction 2	Initial Traffic Count	Time to Zero (seconds)
Lane 1	**33**	**96.09**	Lane 1	**34**	**96.09**
Lane 2	**142**	**241.09**	Lane 2	**12**	**43.66**
Lane 3	**130**	**266.48**	Lane 3	**21**	**70.58**
Lane 4	**23**	**43.66**	Lane 4	**32**	**96.09**
Run 4					
Junction 1	Initial Traffic Count	Time to Zero (seconds)	Junction 2	Initial Traffic Count	Time to Zero (seconds)

Lane 1	**32**	**101.95**	Lane 1	**321**	**500.72**
Lane 2	**22**	**70.21**	Lane 2	**100**	**355.21**
Lane 3	**230**	**465.89**	Lane 3	**210**	**322.73**
Lane 4	**231**	**465.89**	Lane 4	**210**	**500.72**
Run 5					
Junction 1	Initial Traffic Count	Time to Zero (seconds)	Junction 2	Initial Traffic Count	Time to Zero (seconds)
Lane 1	**100**	**129.22**	Lane 1	**33**	**90.95**
Lane 2	**32**	**129.22**	Lane 2	**44**	**90.95**
Lane 3	**80**	**115.05**	Lane 3	**32**	**90.95**
Lane 4	**29**	**115.05**	Lane 4	**31**	**65.11**