TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS

VEHICLE TRAFFIC MANAGEMENT AND ANALYSIS

By:
Nischal Maharjan (073 BEX 421)
Rashik Shrestha (073 BEX 432)
Sajil Awale (073 BEX 436)
Shrey Niraula (073 BEX 443)

A PROJECT WAS SUBMITTED TO THE DEPARTMENT OF
ELECTRONICS AND COMPUTER ENGINEERING AS MINOR PROJECT OF 6TH SEMESTER

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
LALITPUR, NEPAL

August, 2019

COPYRIGHT

ACKNOWLEDGEMENT

We would like to express our special thanks of gratitude to our Department (Department of electronics and computer) as well as our minor project Supervisors (Dr Ram Krishna Maharjan, Prof. Dr. Dinesh Kumar Sharma and Dr. Surendra Shrestha) who gave us a golden opportunity to do this wonderful project on Traffic Optimization from Koteshwor to Suryabinayak, which also helped us in doing a lot of Research and we came to know about so many new things that we really thankful to them.

We would like to appreciate Kamal Nepal sir for his kind and welcoming attitude as well as for the information, resource and transportation. We learnt the current status of Traffic in our junctions, how are they being handled, previous attempts to utilize traffic lights, how most of them became a failure and the budget government spends on these projects. We are grateful for suggestions, criticism and motivation our supervisors and professors provide because they made the project something more.

Finally, we would also like to thank our parents and friends who assisted and driven us for the completion of this project within the limited time frame.

ABSTRACT

Traffic Jams has been a major problem in Kathmandu valley especially at traffic junctions and cross-roads. This requires for an efficient traffic management system to be implemented at such places. But, traffic lights at most road intersections operate on a fixed timing schedule that leads to unnecessary delays, deadlock situations and moreover lead to higher fuel consumption. Our goal is to design a system which can detect the flow of traffic across those junctions, analyze those data and design an optimal traffic light sequence that can best counteract that situation.

The street lighting system is based upon the electronic controller that utilizes the traffic density survey data. Data was collected and analyzed at different busy junctions of Kathmandu Valley. Cameras were used to capture video of the specific junction at various time, which was then processed to get the vehicle flow of the corresponding time.

The data gives an insight into the number of vehicles entering the junction and the time required for them to cross it. This is helpful to calculate the stoppage time which was programmed into the system for optimized and efficient traffic management.

Contents

List of Figures

List of Tables

LIST OF SYMBOLS / ABBREVIATIONS

MCU	MicroContorller Unit
PCE	Passenger Car Equivalent
PCU	Passenger Car Unit
SCOOT	Split Cycle and Offset Optimization Technique
SORT	Simple Online Real Time Tracking
VISSIM	Verkehr In Städten - SIMulationsmodell
YOLO	You Only Look Once

1 Introduction

1.1 Traffic Light

Traffic lights, also known as traffic signals, traffic lamps, traffic semaphore, signal lights, stop lights, robots, and traffic control signals are signaling devices positioned at road intersections, pedestrian crossings, and other locations to control flows of traffic.

Various colors and their sequences convey various meanings. These interpretations may differ as per the rules of the country. A typical vertical traffic signal has three aspects, or lights, facing the oncoming traffic, red on top, yellow below, and green below that. Generally one aspect is illuminated at a time. In some cases, a fourth aspect, for a turn arrow for example, is below the three lights or aspects in more complicated road traffic intersections.

Meaning of various traffic signals in Nepal are as follows:

- Red means to stop the traffic. Bring your vehicle to a complete halt behind the stop line or cross walk. Wait until the light turns green.
- Amber means caution. If you have entered the intersection and the light turns to amber, move on very carefully.
- Green means go on. Go through the crossing carefully. You can turn in the direction of the arrow by giving indicator.

Moreover, flashing signals gives different meaning. A flashing red signal means you should come to a complete stop and move through the intersection where in it safe to do so. A flashing amber signal warns to drive with caution. Pedestrian signals help pedestrians cross intersections safely. If you face a steady red human figure, do not enter the road. If the signal starts flashing, cross the road quickly if you are already on the road. Stop, if you are about to join the road. Walk cautiously if you face a steady green human figure.

1.2 Traffic Congestion

Traffic congestion is when vehicles travel slower because there is too much traffic on roads. This makes trip times longer, and increases queuing. This is also known as a traffic jam. Congestion may result from a decrease in capacity, for example accidents on the road or roads being closed. Bad road layouts can also restrict capacity. Increased traffic, for example by many cars leaving a sports stadium at the same time, can also cause congestion.

Where congestion is common, for example because of commuting in big cities, several methods are used to relieve it. Cars may be banned in certain districts or certain times, or made to carry passengers or pay a fee, or people may use public transport, such as rapid transit, which travel independently of car traffic and are not affected by traffic jams.

1.3 Traffic congestion in Nepal

In Nepal, academicians and stakeholders have done some level of research to develop solution to tackle traffic congestion in the cities, mostly in the Kathmandu valley. However, no concrete solution has been found yet and the problem of traffic congestion has been a troublesome. Metropolitan Traffic Police Division, Kathmandu and Road Division under Government of Nepal have done a kind of traffic assessment on the cities of Kathmandu valley which has already been almost four to five years. Although civil engineering students need to do road assessments and traffic survey under their course, it has only been focused to be submitted as a requirement for the degree.

Figure 1.1 shows the traffic congestion occurred in Jadibuti Junction at evening time. The left portion of the road is completely empty and the right portion is completely full of vehicles. This was due to blockage at the junction in between. Despite there were three Traffic police officer working continuously to manage the traffic at the junction, there was congestion for more than 15 minutes. This situation needs a better technology and better synchronization policy rather than traditional method. Similarly, Figure 1.2 shows the deadlock situation, where none of the vehicles were being able to move forward and created traffic blockage for more than 10 minutes.

Figure 1.1: Traffic Congestion at Jadibuti Chowk

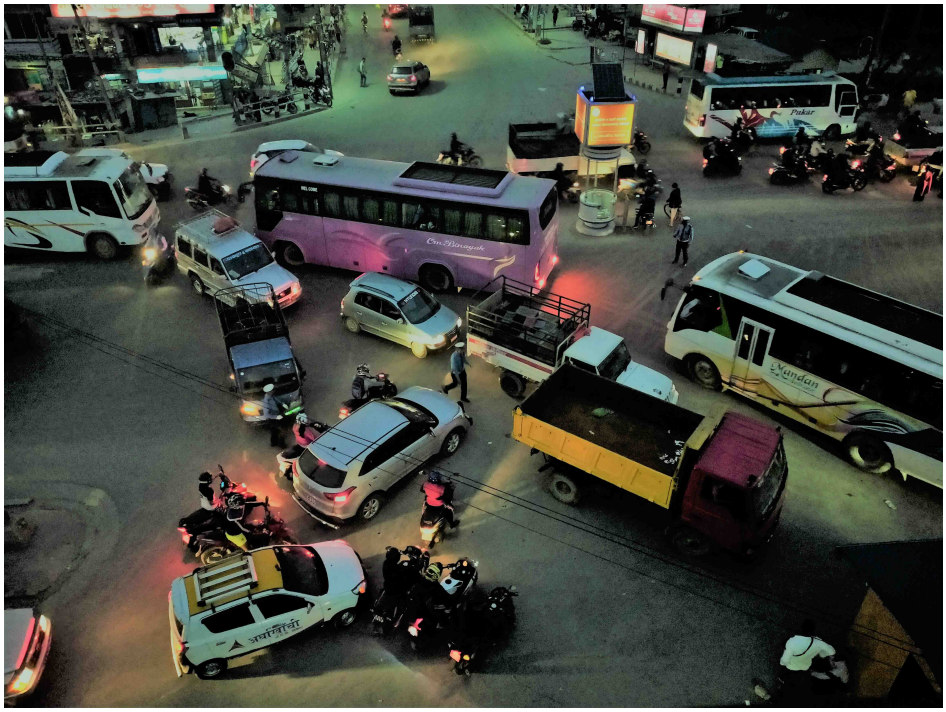

Figure 1.2: Deadlock situation observed at Jadibuti Chowk

1.4 Status of Traffic Control Stations

Traffic Lights controlling centers/boxes are available in lots of places in Nepal. These control station is must where there are traffic lights. These stations may used to function properly in order to control the traffic lights few years back. But currently, we couldn't find any working control stations. Most if them were rusted and have missing parts. The switches were totally jammed due to rust and remained unused for such a long time.

Figure 1.3 and Figure 1.4 shows the traffic light control centers currently being implemented in Gatthaghar, Bhaktapur. As we can see, these are full of rust and non in the functional condition. Control Stations we visited were of following areas:

- Suryabinayek
- Sallaghari
- Naya Thimi
- Gatthaghar
- Kausaltar
- Lokanthali
- Jadibuti
- Koteshwor

2 Objectives

This project was conducted with the primary motive of analyzing the Current Traffic Situation in certain junction of Kathmandu city and propose an immediate theoretical solution for this problem. Some objectives of this project are as follows:

- To study about the present condition of Traffic in certain junctions of Kathmandu (primarily Koteshwor and Jadibuti Junction)
- To collect the data such as vehicle density, phase timings, road width, speed limit etc. for these two junctions
- To create a mathematical model which can represent the real scenario of traffic flow in these junctions
- To generate a best possible traffic light sequence which can minimize the delay time and waiting time in these two junctions.
- To test the traffic flow in simulation software PVT Vissim and collect the data form the simulation.
- To design and fabricate the hardware which can operate ta traffic signals as per the timing sequence calculated by the algorithm.
- To make a software (web application) which can communicate with the hardware wirelessly and share data.

3 Literature review

At the international level, many research works have been carried out for the automatic detection of vehicles using DIP (Digital Image Processing) algorithms. Some such works include Image Processing Based Intelligent Traffic Controller.

In New York City, 7,660 (of a total of 12,460) signalized intersections are controlled by a central computer network and monitored by traffic management centers.

Similarly, in Toronto, 83% of its signals are controlled by the Main Traffic Signal System (MTSS). 15% also use the SCOOT (Split Cycle and Offset Optimization Technique), an adaptive signal control system.

4 Project Dissection

Project was divided into four phases as shown in Figure 4.1. Quick summary of these phases are as follows:

4.1 Data Collection

The starting of the project was data collection. Following things were done at the beginning of the project:

- Existing flow of vehicles per unit time (Vehicle Density)
- Dimensions of the road
- Number of junctions
- Existing Lane rules
- Existing rules for maximum and minimum speed limits

Initially, these data were collected using various techniques explained in the following part of this report.

4.2 Modeling

Considering all the collected data, now a mathematical model was prepared of the existing traffic situation. These data were manipulated and analyzed using various techniques and premade algorithms and a model was designed which represented the real life traffic situation. The modeling could not be exact due to various factors and limiting conditions. There is uncountable number of parameters that can affect the collected data. As, there is no way to consider all the real life parameters, so some major parameters such as Time of the day, working days/ holidays etc were taken to make our model.

4.3 Data analysis and Prediction

The collected data were used to generate the traffic sequences. A famous algorithm known as Webster method was used. This algorithm takes traffic flow in various roads as parameters and output the traffic sequence as the output. The data that collected was not sufficient for webster algorithm, so interpolation techniques was used to fill up ta missing data in gap. This may cause little deviation in output, but is still better than having no sequence at all.

4.4 Hardware Design and Fabrication

Hardware was designed to visualize the traffic sequence. Atmega32 microcontroller was used as the main brain of the hardware system. It receives data from esp8266, which is WiFi supportable Integtrated Circuit. Then process that data and shows the output via 3 traffic light display boards. Esp8266 communicates with atmega32 via UART protocol and sends parallel data to traffic light display boards.

4.5 Software Design

The software design part basically includes two major aspects: the development of the web application and the time sequence generation based on optimization. The python based coding was performed on the control server that generated the sequence. On the other side, the python flask based server was established integrating database with it. These two processes were threaded to access the common memory space that made interaction with them possible. There are still more software portion carried out to made this project possible.

4.6 Testing and Optimization

Generated traffic sequence was tested in Simulation tool VISSIM. VISSIM was expensive software, so student version was used for free with lots of limitations. The simulation results was then compared with the current data. Pretty better results were observed from simulation using the traffic sequence using Webster algorithm.

5 Data Collection

Data collection was done in using two approaches. The methods used for data collection were:

- Manual Recording
- Video analysis method

5.1 Manual recording

This might be the most tedious way of data collection. But, it was must for the project and should be done at the beginning. The data taken manually was to visualize and record the existing traffic sequence used my Traffic Police. Two junctions were targeted i.e Koteshwor and Jadibuti. Then, the traffic sequence was divided into three phases in each junction. Now, the opening and closing time of each junctions were recorded manually. Data are tabulated in Table 5.1

Table 5.1: Manually collected data of three phases

Phase	Koteshwor Chowk(duration)	Jadibuti Chowk(duration)
Phase 1	2:41 min	3:51 min
Phase 2	8:30 min	7:20 min
Phase 3	2:03 min	1:20 min
Phase 1	2:20 min	3:10 min
Phase 2	7:15 min	6:25 min
Phase 3	3:24 min	4:14 min
Phase 1	3:05 min	2:37 min
Phase 2	7:40 min	6:30 min
Phase 3	3:57 min	2:48 min
Phase 1	2:50 min	3:20 min
Phase 2	7:55 min	6:37 min
Phase 3	2:35 min	3:25 min
Phase 1	2:30 min	3:15 min
Phase 2	8:15 min	7:20 min
Phase 3	2:53 min	2:55 min

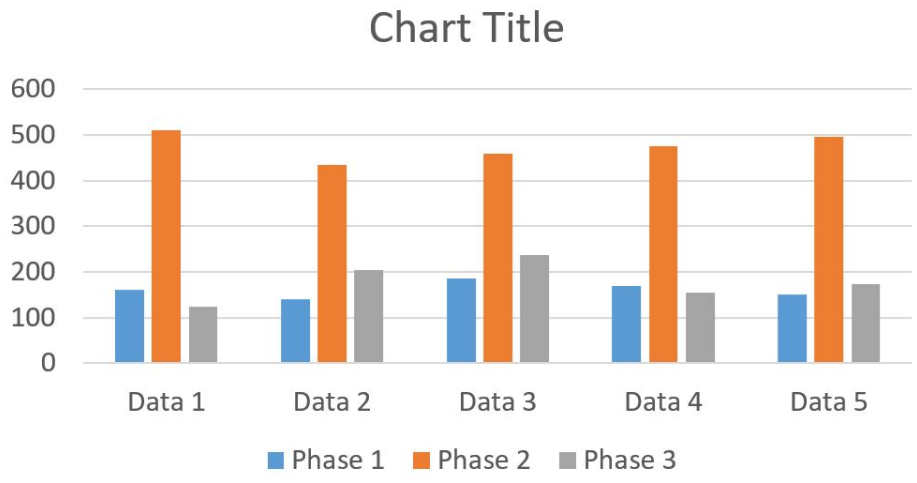

Figure 5.1: Phase bar graph

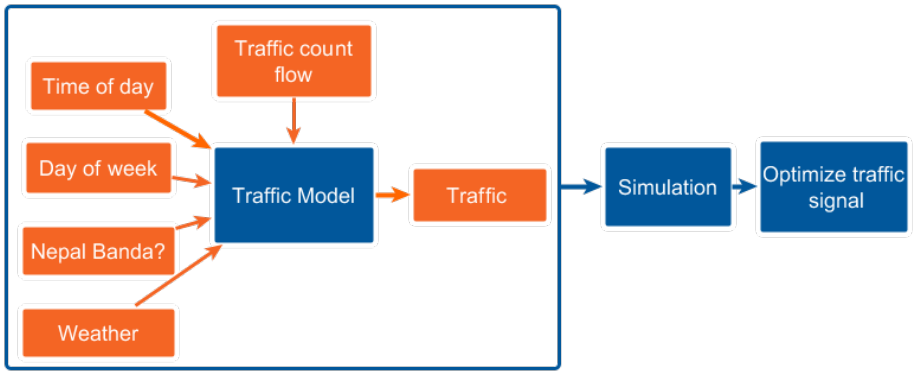

Figure 5.2: Model block diagram

5.2 Video Analysis Method

5.2.1 Method for counting vehicles

The overall data flow process of counting vehicles is shown in Figure 5.3. To count vehicles, we first need to detect them. This is a classic object detection problem, with many pertained models available for deep neural nets. It can also be done with image processing and counting the blobs. But deep neural nets (Convolution Neural Network) is a rapidly developing area, can be further enhanced easily and we can be optimistic that the computational power it requires to process every frame will dramatically be cheaper.

After detecting vehicles, we need to track them throughout the sequence of frames. This is necessary because you don't want to count the same vehicles multiple times but only once. Tracking gives you an identity of each vehicle so that we don't lose them in a sequence of video. SORT(simple online and realtime tracking) effective and quick algorithm is for 2D multiple object tracking in video sequences.

Finally, we need to count the unique vehicles passing across the road. We define an arbitrary line perpendicular to the direction of the road. We also draw a line from previous (frame) bounding box coordinates to new bounding box coordinates. If the two lines intersect them we can increment the vehicle count. This is illustrated in Figure 5.3.

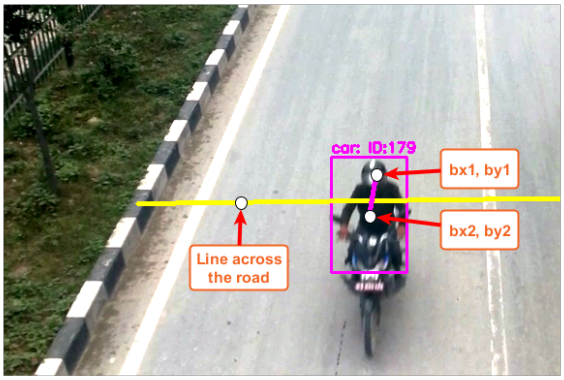

Figure 5.3: Increment vehicle count

Where,
bx1, by1 - previous bounding box coordinates
bx2, by2 - next bounding box coordinates
The steps involved in counting vehicles are listed below:

- bx1, by1 & bx2, by2 are center coordinates of bounding box of object with same obj ID (same obj) in past and current frame

- Draw line segment from bx1, by1 to bx2, by2
- If this line and line segment across the road intersect, then add to the vehicle counter

5.2.2 Output of Counting Vehicle application

Finally, the output of vehicle counting application is shown in Figure 5.4.

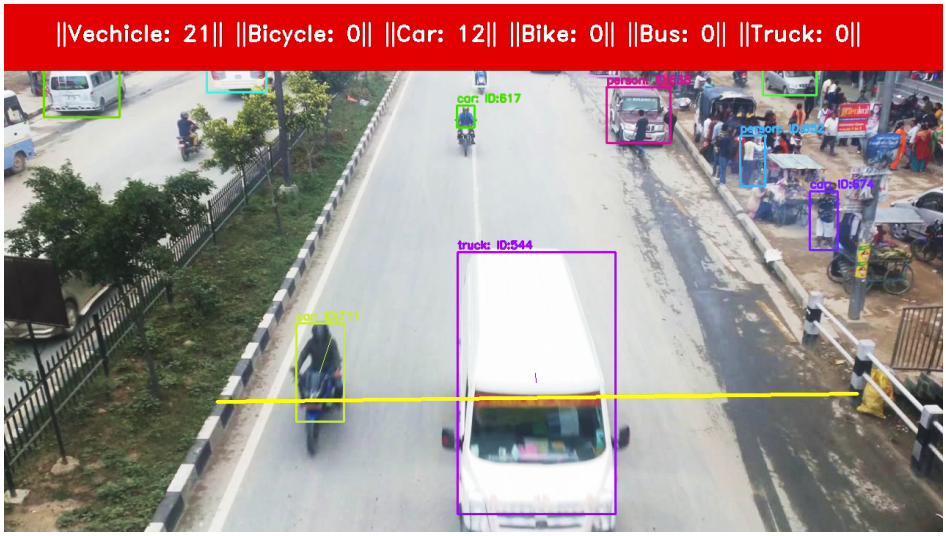

Figure 5.4: Vehicle count output

5.2.3 YOLO (You Only Look Once)

YOLO is a state-of-the-art, real-time object detection algorithm. It has three versions till date and few model configurations. Its accuracy and speed outmatches other models easily.

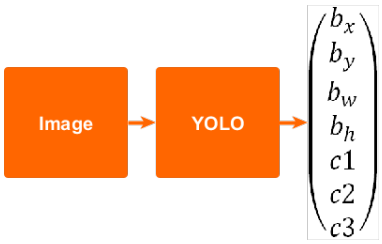

Figure 5.5: Bounding boxes

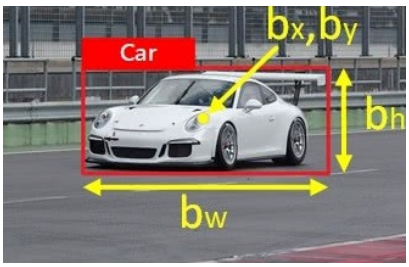

Figure 5.6: Input and Output of YOLO

Here,
bx, by = bounding box center coordinates
bw, bh = bounding box width and height
c1 = person, c2 = car, c3 = dog

YOLO divides the input image into an SxS grid. Each grid cell predicts only one object. Each grid cell predicts only one object. Each of the grid cell is fed in to the yolo neural network architecture. Standard yolo architecture has 109 convolutional layers. YOLO can make duplicate detections for the same object. To fix this, YOLO applies non-maximal suppression to remove duplicates with lower confidence. Also any two detections with the same class and high IOU (intersection over union area) than threshold is replaced by only one detection with higher confidence.

5.2.4 SORT (Simple Online Real Time Tracking)

It is an algorithm for 2D multiple object tracking in video which uses Deep Association Metric to identify same object and it is designed to track where past and current frame is available and finally it provides object identities on the fly.

SORT is a barebones implementation of a visual multiple object tracking framework based on rudimentary data association and state estimation techniques. While this minimalistic tracker doesn't handle occlusion or re-entering objects its purpose is to serve as a baseline and testbed for the development of future trackers.

5.2.5 Collection of data

For the proper optimized signal design, we need accurate vehicle flow rate (vehicles flowing per hour) of each traffic source. We selected Jadibuti, Kathmandu, Nepal chowk/intersection which is a 3 leg (T type) intersections shown in Figure 5.7. For each source, we collected data of traffic flow. This was achieved by collection of video footage of the junction, which was processed with YOLO to find number of vehicles passing (also the type of vehicles) of each source which was mentioned how it was done.

Figure 5.7: Jadibuti Chowk

Finally, the traffic flow volume (q) was computed in terms of PCU (Passenger car equivalent). Passenger Car Equivalent (PCE) or Passenger Car Unit (PCU) is a metric used in Transportation Engineering, to assess traffic-flow rate on a highway. A Passenger Car Equivalent is essentially the impact that a mode of transport has on traffic variables (such as headway, speed, density) compared to a single car. For example, typical values of PCE (or PCU) are:

- private car (including taxis or pick-up) = 1
- Motorcycle = 0.75
- bicycle = 0.5
- horse-drawn vehicle = 4
- bus, tractor, truck = 3

The data collection process is illustrated in Figure 5.8.

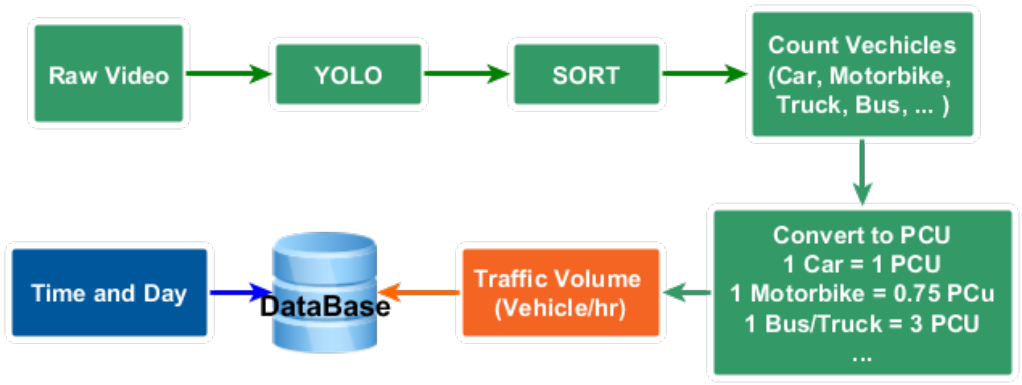

Figure 5.8: Flow chart of data collection

6 Data Visualization

Of collected traffic flow data from different traffic source in Jadibuti intersection, we constructed line graphs and bar graphs to better understand the flow distribution with and day of the week.

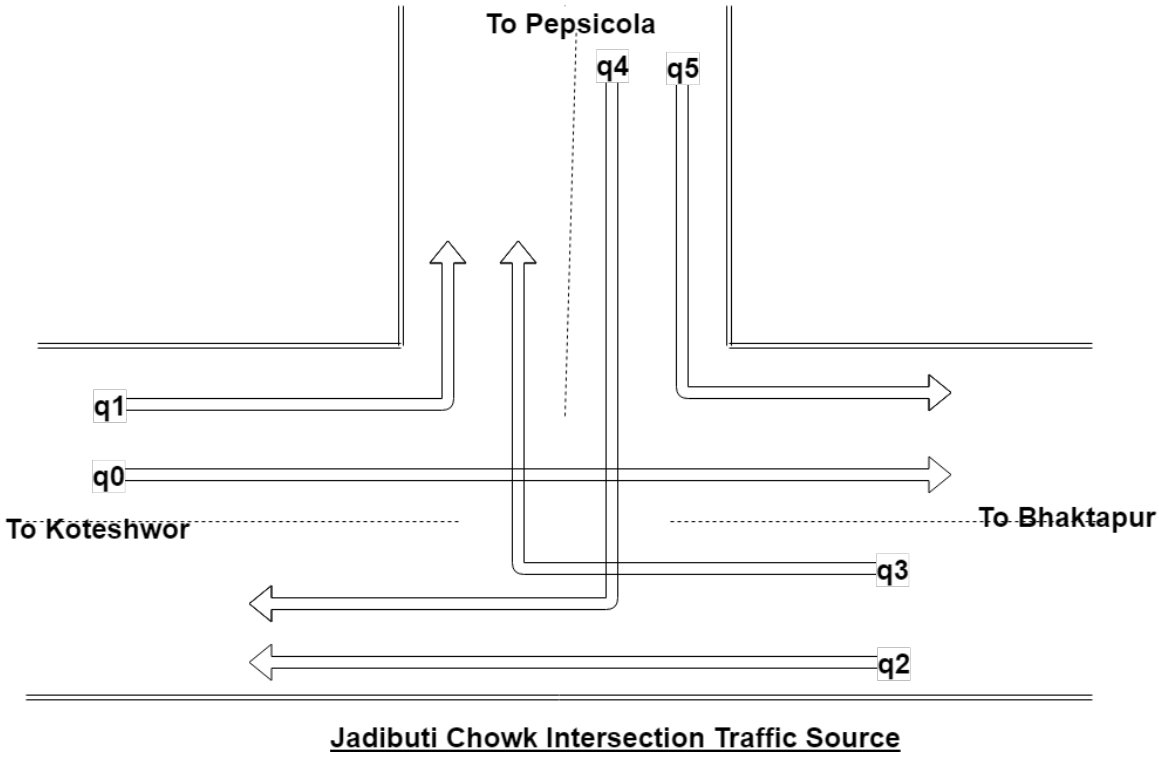

Figure 6.1: Jadibuti intersection Traffic Source

The Figure 6.1 shows that q0, q1, q2, q3, q4, q5 are different sources of traffic originating from one point and departing to another point in the jadibuti junction. We measured the flow for everyday of the week from 8 AM to 8 PM divided into 7 segments.

Following python Dictionary contains all the collected data by direct and indirect method:

```
 'Sunday' : {
        #qx = [qx at 08,qx at 10,qx at 12,qx at 14,qx at 16,qx at 18,qx at 20]
        'q0' : np.array([6200, 7380, 4240 , 4360, 5780, 8720, 4780]),
        'q1' : np.array([1240, 1476, 1278, 1150, 1352, 1744, 956,]),
        'q2' : np.array([7380, 11560, 6254, 6380, 6105, 9250, 5720]),
        'q3' : np.array([1150, 1204, 1520, 986, 950, 1120, 1004]),
        'q4' : np.array([7180, 10560, 2685, 6354, 6842, 7815, 5419]),
        'q5' : np.array([2504, 3450, 2510, 2154, 2160, 2538, 1370]),
    },

'Monday' : {
        #qx = [qx at 08,qx at 10,qx at 12,qx at 14,qx at 16,qx at 18,qx at 20]
        'q0' : np.array([6720, 7310, 4399 , 4517, 5218, 8459, 4258]),
        'q1' : np.array([1592, 1754, 1148, 1485, 1357, 1745, 758,]),
        'q2' : np.array([7489, 11750, 6478, 6348, 6940, 9745, 5912]),
        'q3' : np.array([1120, 1207, 1590, 782, 864, 1140, 1048]),
        'q4' : np.array([7485, 10468, 6247, 6458, 6715, 7915, 5248]),
        'q5' : np.array([2489, 3356, 2641, 2044, 2069, 2458, 1458]),
    },
'Tuesday' : {
```

```
          #qx = [qx at 08,qx at 10,qx at 12,qx at 14,qx at 16,qx at 18,qx at 20]
          'q0' : np.array([6452, 7280, 4258 , 4360, 5780, 8720, 4926]),
          'q1' : np.array([1354, 1587, 1347, 1162, 1452, 1678, 985,]),
          'q2' : np.array([7450, 11468, 6351, 6258, 6258, 9157, 5684]),
          'q3' : np.array([1248, 1350, 1468, 957, 982, 1240, 950]),
          'q4' : np.array([7458, 10278, 6248, 6254, 6845, 7805, 5490]),
          'q5' : np.array([2498, 3354, 2510, 2254, 2168, 2489, 1278]),
     },
 'Wednesday' : {
          #qx = [qx at 08,qx at 10,qx at 12,qx at 14,qx at 16,qx at 18,qx at 20]
          'q0' : np.array([6254, 7350, 4268 , 4390, 5685, 8690, 4851]),
          'q1' : np.array([1284, 1648, 1320, 1245, 1342, 1678, 967,]),
          'q2' : np.array([7456, 11428, 6351, 6247, 6246, 9280, 5480]),
          'q3' : np.array([1168, 1208, 1670, 947, 964, 1154, 1158]),
          'q4' : np.array([7190, 10548, 6238, 6380, 6749, 7749, 5647]),
          'q5' : np.array([2641, 3510, 2640, 2235, 2345, 2468, 1259]),
     },
'Thursday' : {
          #qx = [qx at 08,qx at 10,qx at 12,qx at 14,qx at 16,qx at 18,qx at 20]
          'q0' : np.array([6150, 7458, 4120 , 4315, 5987, 8790, 5948]),
          'q1' : np.array([1340, 1578, 1347, 1056, 1458, 1978, 789,]),
          'q2' : np.array([7458, 13475, 6158, 6158, 6125, 9475, 5848]),
          'q3' : np.array([1250, 1648, 1485, 895, 884, 1056, 1054]),
          'q4' : np.array([7458, 11547, 6328, 6254, 6489, 7848, 5684]),
          'q5' : np.array([2648, 3415, 2554, 2478, 2648, 2644, 1347]),
     },
'Friday' : {
          #qx = [qx at 08,qx at 10,qx at 12,qx at 14,qx at 16,qx at 18,qx at 20]
          'q0' : np.array([6354, 7285, 4352 , 4522, 5710, 8654, 4578]),
          'q1' : np.array([1348, 1375, 1345, 1345, 1257, 1786, 942,]),
          'q2' : np.array([7354, 11972, 6357, 6278, 6257, 9647, 5842]),
          'q3' : np.array([1250, 1354, 1650, 1062, 860, 1250, 1520]),
          'q4' : np.array([7458, 1156, 6485, 6235, 6945, 7658, 5485]),
          'q5' : np.array([2548, 3458, 2485, 2268, 2364, 2648, 1284]),
     },
 'Saturday' : {
          #qx = [qx at 08,qx at 10,qx at 12,qx at 14,qx at 16,qx at 18,qx at 20]
          'q0' : np.array([4200, 5380, 2240 , 2360, 3780, 6720, 2780]),
          'q1' : np.array([640, 876, 758, 740, 852, 924, 456,]),
          'q2' : np.array([2380, 7560, 4154, 3180, 4005, 5150, 3720]),
          'q3' : np.array([950, 1004, 860, 486, 350, 920, 752]),
          'q4' : np.array([5180, 6560, 4254, 4354, 3042, 4815, 3025]),
          'q5' : np.array([504, 1050, 510, 954, 960, 1038, 1070]),}
```

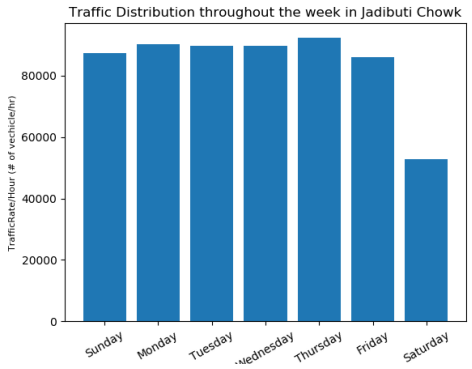

Figure 6.2: Traffic Distribution throughout the week in jadibuti chowk

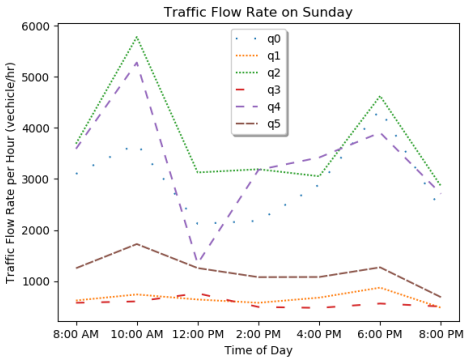

Figure 6.3: Traffic Flow Rate on Sunday

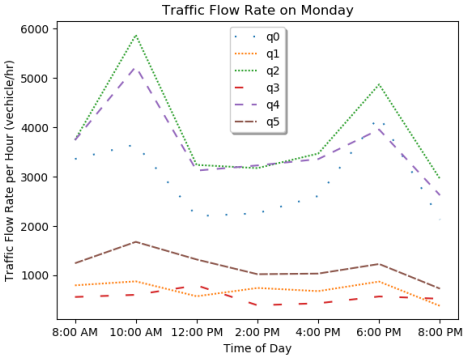

Figure 6.4: Traffic Flow Rate on Monday

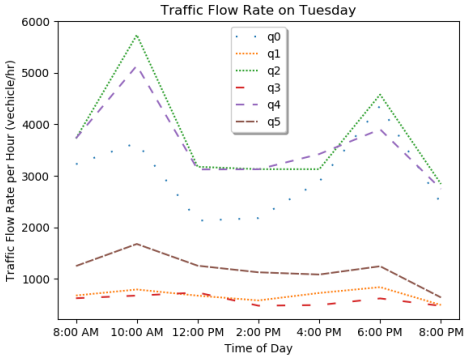

Figure 6.5: Traffic Flow Rate on Tuesday

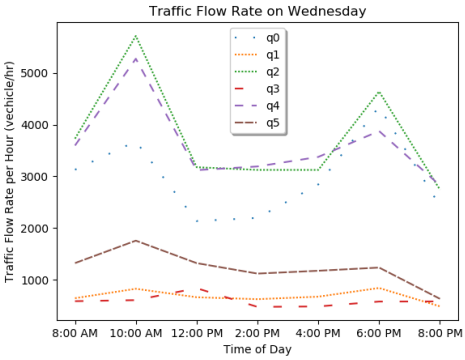

Figure 6.6: Traffic Flow Rate on Wednesday

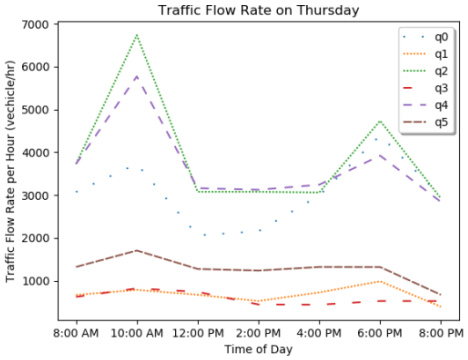

Figure 6.7: Traffic Flow Rate on Thursday

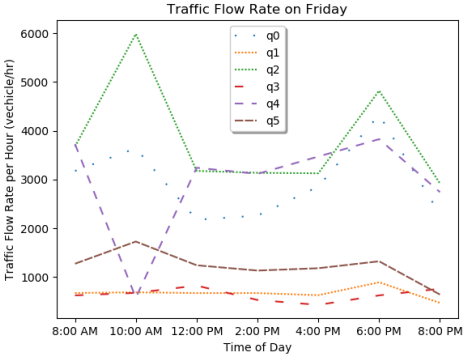

Figure 6.8: Traffic Flow Rate on Friday

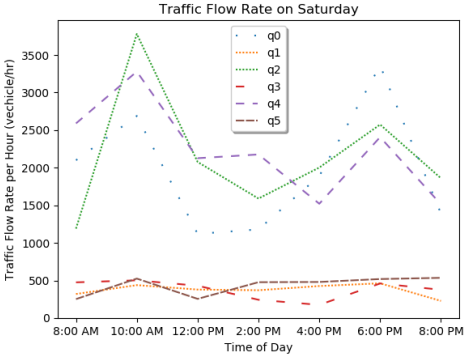

Figure 6.9: Traffic Flow Rate on Saturday

Figure 6.2 is data representation of traffic volume of all day for each day in the week from Sunday to Saturday. Figure 6.3, Figure 6.4, Figure 6.5, Figure 6.6, Figure 6.7, Figure 6.8 and Figure 6.9 are the traffic flow throughout the time of the day on each traffic/vehicles source from q0 to q5 staring from Sunday to Saturday respectively.

7 Hardware Design and Fabrication

7.1 Techniques used in designing Prototype

7.1.1 Connection with Server

The Prototype is connected with the server through the wireless medium via WiFi network. NodeMCU works as a bridge between hardware prototype and the Server. It receives the timing sequence processed and generated in the Server and transfers it to the At mega32 Micro controller via serial communication through UART Protocol.

7.1.2 Embedded Programming

Atmega32A is used as a microcontroller in our prototype and Embedded C Programming has been implemented for programming the microcontroller. It has four 8-bit output ports capable of driving various kind of loads in our case Leds and seven segment displays. The Microcontroller receives the timing sequence from NodeMCU and generates control signals to drive the Leds and seven segment timers.

7.1.3 Decoding

In order to make our prototype as efficient as possible decoding technique has been applied. 7448 decoder has been used in order to decode BCD codes into the input of the seven segment display decoder. Thus enabling us to control 7 data lines by the help of 4 data lines. Also the Binary coded Decimal is standard code and implementation of it made programming efficient and easy.

7.1.4 Multiplexing

Beside decoding the technique of multiplexing has been applied. Due to multiplexing a number of seven segment display and Leds are being controlled by minimum number of control pins as possible. The input lines of number of displays are connected to common shared data lines and select pins are used to select the display to be drived at a time. Multiplexing also helps in efficient power consumption by saving the amount of power being used .

7.2 Microcontroller based Prototype

The Prototype of the system for the control of traffic lights in the three way junction was designed which is a microcontroller based system. The designing of the circuits boards were carried out in the Kicad Software for PCB design. The Prototype was designed using the components that are readily available in the market and has good documentation. The prototype is constitute of three circuit boards resembling traffic light for each way. Beside these one main board which performs the process of controlling remaining circuit boards is present in our prototype. The kicad design view of the circuits boards used in our prototype is as below.

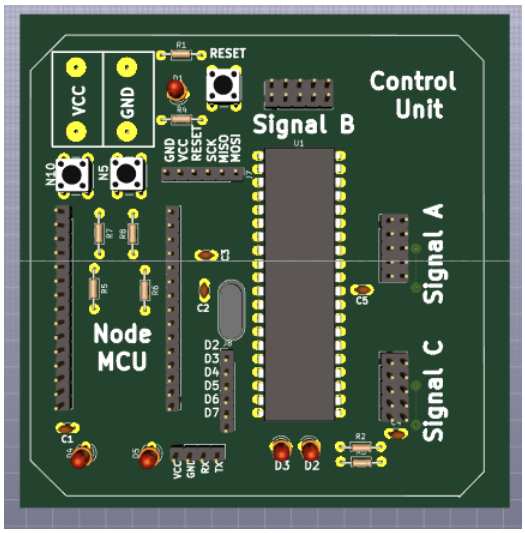

Figure 7.1: Kicad Design Controller Board

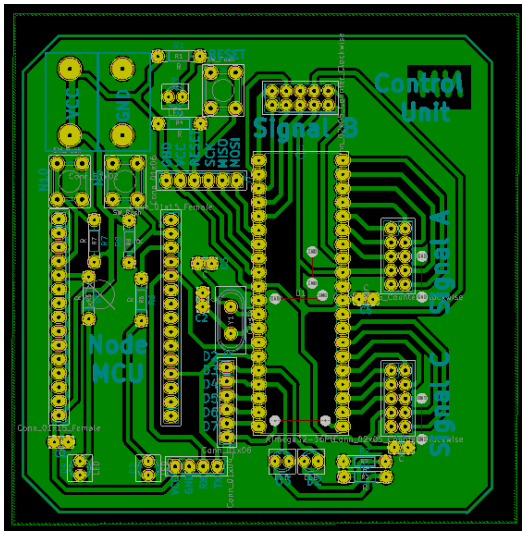

Figure 7.2: Kicad Design Controller Board

This is the circuit design of our control unit through which control signals are being transmitted to light boards via connectors and the circuits of traffic lights is as below.

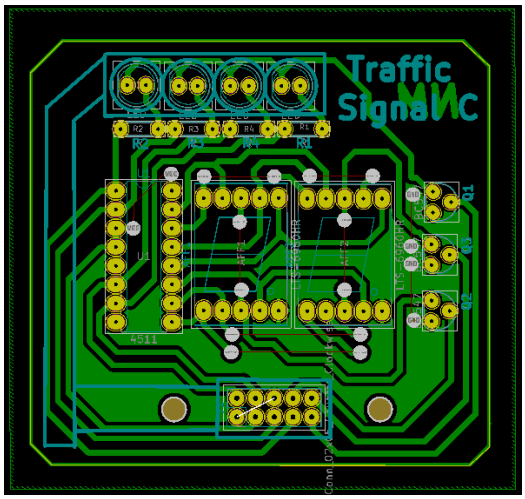

Figure 7.3: KiCad Design Display Board

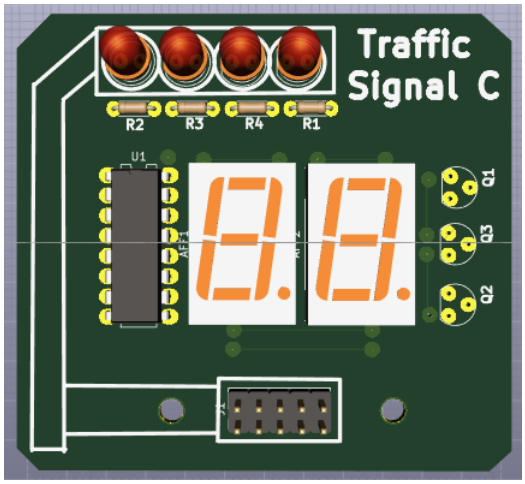

Figure 7.4: KiCad Design Display Board

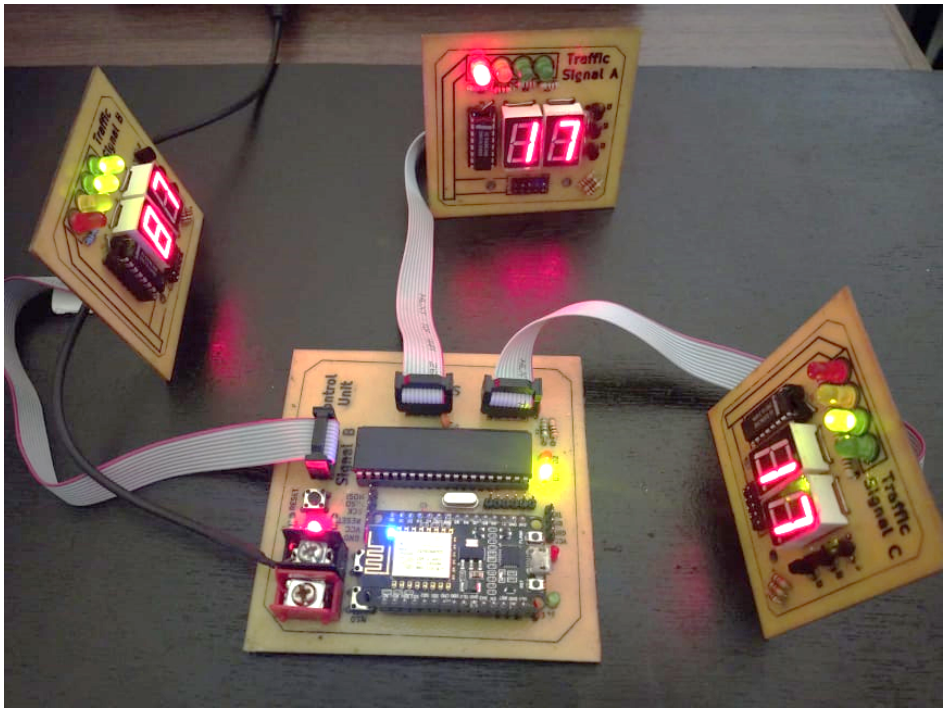

Figure 7.5: Hardware Prototype

8 Software Development

8.1 Reasons for Web Application

The main reason for creation of web application are:

1. Monitor Purpose
    - Dynamic traffic simulation based on time sequence.
    - Currently operating phase
    - Live Stream of junction
2. Control
    - Reset Phase in case of emergency

8.2 Main Features of Web application

The features available in web application are:

8.2.1 Only Authenticated Access

The access to web application is possible by only the authenticated user preventing access, modification, traffic hijackings. Email and Password needs to be entered to gain the access to web application as shown in Figure 8.1.

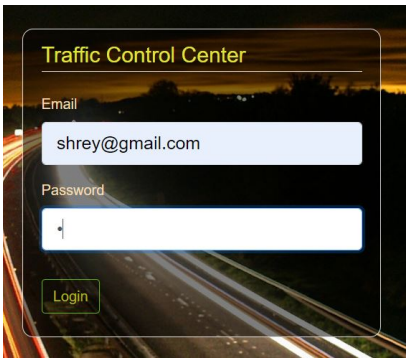

Figure 8.1: Login Page

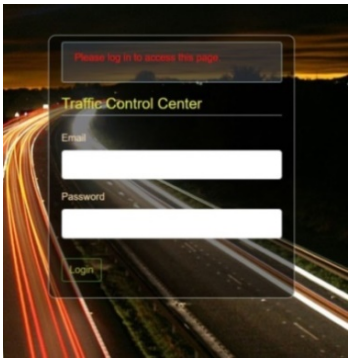

Figure 8.2: User Redirected to Login Page

The default page is set as login page and user gets redirected to login page when other pages are accessed without login. In Figure 8.2, user is redirected back giving alert in red color.

8.2.2 Database Integration

SqLite database is used to hold the authenticated user's information such as password, emails and other information.

8.2.3 Time Basis Simulation

Current operating phases are shown with the images. The accessible phase is shown by green arrow. Remaining time for current operating phase to terminate is shown by phase time counter. The 8.3 shows phase 1 being operating right now. The accessible way is shown by green arrow while red non-operating is shown by red arrow. Similarly, the conditions for phase 2 and phase 3 can be shown in similar manner in figure 8.4 and 8.5.

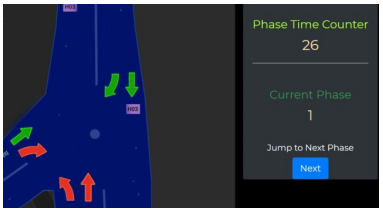

Figure 8.3: Phase 1

Figure 8.4: Phase 2

Figure 8.5: Phase 3

8.2.4 Multiple Junctions

Monitoring and control operations are incorporated for three junctions; Koteshwor, Jadibuti and Lokanthali. With user's choice, any of the junction can be visited by clicking the button which can be seen in Figure 8.6

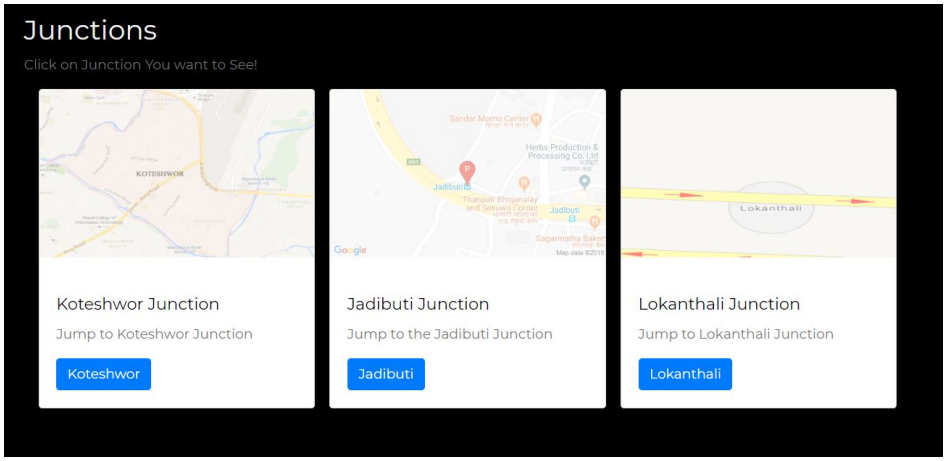

Figure 8.6: Junction Page

8.2.5 Phase Reset of Each Junction

Phase reset mechanism has been implemented for the case of emergency. The current phase state is recognized when next phase button is pressed. Then current phase is stopped and next phase is started which gets reflected in web application and even to hardware. The phase 2 will be started when next phase button in case of Figure 8.8 is pressed.

8.2.6 Charts of Traffic Density of Junctions

The each junction's average traffic volume is implemented in charts. The time basis average volume is shown by line chart whereas day basis peak volume is shown by bar graph. Line and bar graph with yellow color can be seen on Figure 8.7, where value will be displayed when hovered over chart in web application.

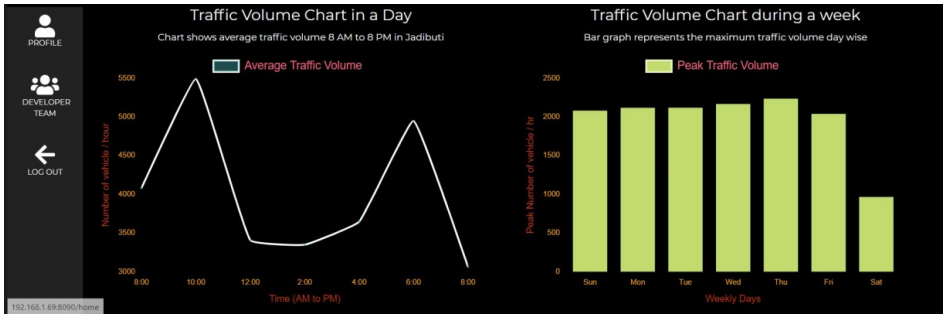

Figure 8.7: Data Visualization in Webapp

8.2.7 Live Streaming of Each Junction

The real time junction view can be livestreamed to web application with the help of IP camera that can be set at the each junction. Frame is captured through Ip camera by using python openCV library and sent to web application using iterator concept of python. With this continuous process, live video is made possible just like in motion-JPEG(mJPEG). The same page is livestreamed in Figure 8.8 as seen on right bottom corner.

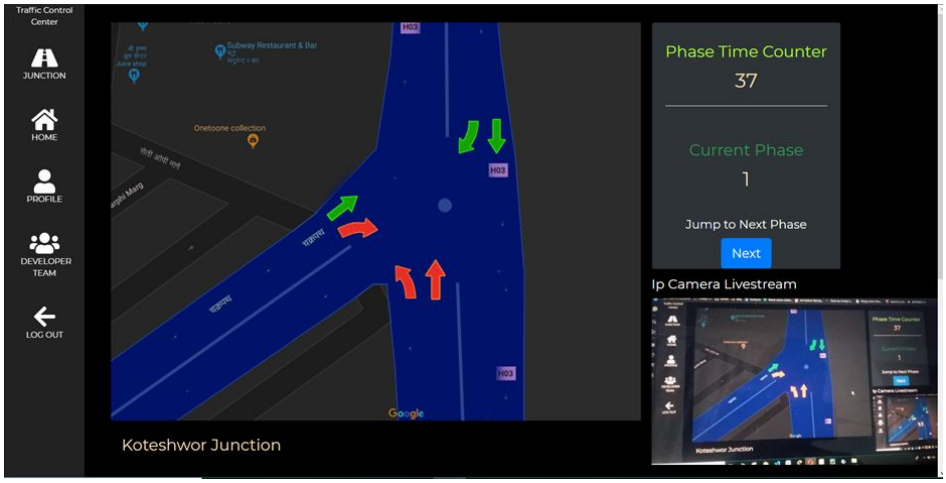

Figure 8.8: Livestream in Webpage

8.2.8 The Other Features

About page as seen in Figure 8.9, Our team member descriptions as shown in Figure 8.8.10 and contact method as in Figure 8.11 are some of other features. Other features such as Log out, Home page are also present in web application.

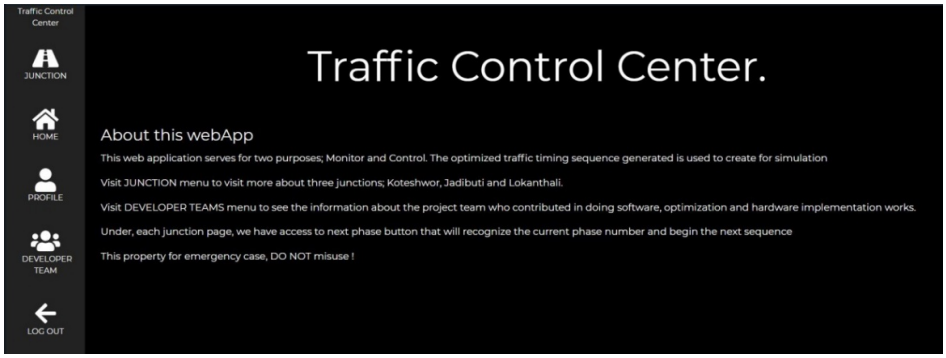

Figure 8.9: About Us

Figure 8.10: Team members

Figure 8.11: Contact Us

8.3 Basic Operation Mechanism

The block diagram as shown in fig 8.12 can be explained as:

8.3.1 Flask

Flask is used for backend purpose. Routing to specific URL is managed by Flask. Route related functions are defined in Routes.py. Specific HTML file gets rendered on webpage via these functions which gets executed when certain route is visited. The database is managed and integrated with the help of Flask where different information can be stored. These data in database can be passed to templates containing html files that get rendered on pages. Login form is modelled using class of python in Forms.py with each form field (such as password, email) being represented by member variable within that class. Each field gets associated with database. User's login control, password encryption are also handled by flask.

8.3.2 Database Integration

SQLAlchemy is a common database abstraction layer and Object Relational Mapper (ORM) that requires a little bit of configuration effort, which is being done by Flask. The database in flask is modelled with the help of the class such that each class member variable represents the column in database table as shown in Figure 8.13. The whole layout of such database is constructed in models.py file. The database is used for login system where entered email and password are verified with the existing one in database allowing access if matched. The email and password errors are also notified.

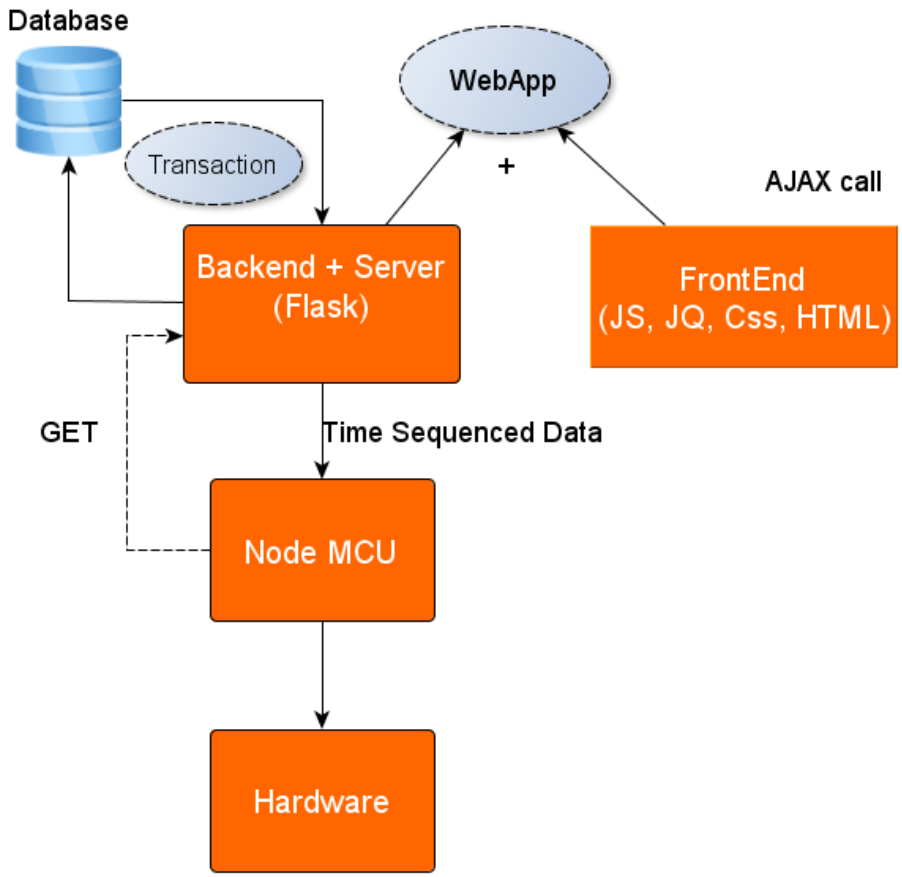

Figure 8.12: General Workflow in Webapp

```
class User(db.Model, UserMixin):
    id = db.Column(db.Integer, primary_key = True)
    username = db.Column(db.String(20), unique=True, nullable=False)
    email = db.Column(db.String(70), unique=False, nullable=False)
    password = db.Column(db.String(25), unique=False, nullable=False)
```

Figure 8.13: Database implementation by Python Class

8.3.3 Front-end Development

With the help of JavaScript, JQuery, basic HTML, CSS and Bootstrap, the frontend side for web application is done. Various JavaScript libraries such as including chart.js are incorporated in design of front look of the web application. The concept of AJAX is extensively used for asynchronous polling, which is technique to poll some content from the backend side without needing to refresh the webpage. I.e. portion of the page can be updated constantly without refreshing whole page. This concept is used in time basis simulation to update value at each second fetching the data from backend side. The phase reset mechanism is also the implementation of AJAX.

8.3.4 Time Sequence Generation and Threading

Time Sequence is generated on python files including the current phase state, on off state for Red, Yellow, Green and Side (RYGS) and time remaining for that phase that gets passed to flask application running parallel. This is made possible by using the multithreading concept that shares the memory between each py files. The four threads; flask thread, and other three timing generation threads are active and are run at the server which can be seen in Figure 8.14 and sharing the common data space.

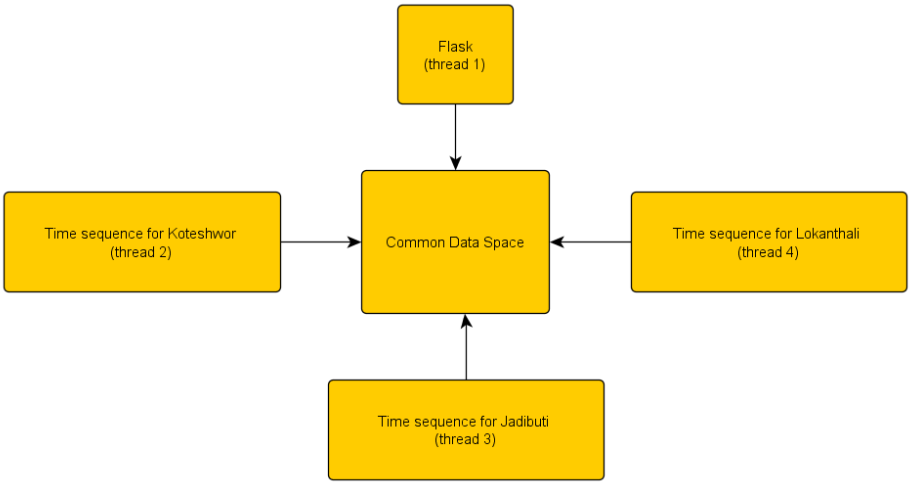

Figure 8.14: Mutithreading

8.3.5 Communication with NodeMCU and Hardware

Time generation sequence from server is passed to Traffic junction constructed in hardware. The IOT based approach of using the internet is chosen for communication. 'GET' request is sent constantly by NodeMCU (Wi-Fi-module) to the flask app. The time sequence then gets passed to the hardware implemented junction where the effect takes place. The phase reset in hardware is also possible after next phase button press. As shown in Figure 8.15, by constant GET request, necessary data (time sequence values) are obtained by hardware.

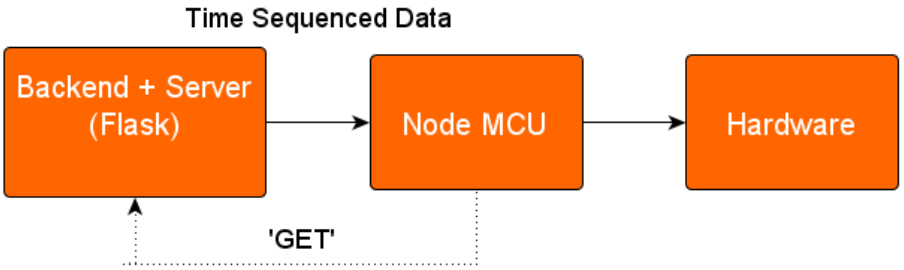

Figure 8.15: NodeMCU Communication

9 Traffic Optimization

Traffic Optimization are the methods by which time stopped in road traffic (particularly, at traffic signals) is reduced. Texas Transportation Institute estimates travel delays of between 17–55 hours of delay per person per year relating to congestion on the streets.Traffic device optimization hence becomes a significant aspect of operations. Several techniques exist to reduce delay of traffic.

The well-known Webster timing was proposed by Webster and Miller. This timing algorithm is designed to reduce the vehicle delay time and optimize the signal cycle. This algorithm provides a basis for research on the signal timing algorithm.

9.1 Webster's Algorithm for Traffic Optimization

Webster's algorithm can be used for Design of Traffic Signal at an intersection. This includes calculating cycle length of signal, green, red and yellow time of each phases. Formulas to calculate these parameters given by webster method are as follows.

9.1.1 Cycle length of signals

Webster has shown that for a wide range of practical conditions minimum intersection delay is obtained by the Equation 9.1.

$$C_0 = \frac{1.5L + 5}{1 - \sum_{i=1}^{\phi} Y_i} \tag{9.1}$$

Where,
C_0 = optimum cycle length (sec)
L = total lost time per cycle (sec)
Y_i = (sum of critical flow ratios)maximum value of the ratio of approach flow to saturation flow for all lane groups using phase i (i.e., q_{ij}/s_j)
ϕ = number of phases
q_{ij} = flow on lane groups having the right of way during phase i
s_j = saturation flow on lane group i

9.1.2 Total Lost time (L)

It gives the time lost in a cycle when no flow is allowed in the intersection. Equation 9.2 gives the lost time for phase i.

$$l_i = G_{ai} + \tau_i + G_{ei} \tag{9.2}$$

Where,
l_i = lost time for phase i
G_{ai} = actual green time for phase i (not included yellow time)
τ_i = yellow for phase i
G_{ei} = effective green time for phase i

Finally, equation 9.3 is the total lost time.

$$L = \sum_{i=1}^{\phi} l_i + R \tag{9.3}$$

Where,
L = total lost time
R = is the total all red time during the cycle

9.1.3 Allocation of Green times

In general, the total effective green time available per cycle is given by equation 9.4.

$$G_{te} = C - L = C - \sum_{i=1}^{\phi} l_i + R \tag{9.4}$$

Where,
C = actual cycle length (usually obtained by rounding of Co to the nearest five seconds
G_{te} = total effect green time per cycle

To obtain minimum total delay, the total effective green time should be distributed among the different phases in proportion to their Y values to obtain the effective green time for each phase given by equation 9.5.

$$G_{ei} = \frac{Y_i}{Y_1 + Y_2 + Y_3 + ... + Y_\phi} * G_{te} \tag{9.5}$$

And the actual green time for each phase is obtained from equation 9.6.

$$G_{a\phi} = G_{e\phi} + l_\phi - \tau_\phi \tag{9.6}$$

9.1.4 Minimum Green Times

The minimum green time can be determined by using the HCM expressions given in equations 9.7 and 9.8.

$$G_p = 3.2 + \frac{L}{S_p} + [2.7 * \frac{N_{ped}}{W_E}]; for W_E > 10ft(3.2m) \tag{9.7}$$

$$G_p = 3.2 + \frac{L}{S_p} + [2.7 * N_{ped}]; for W_E < 10ft(3.2m) \tag{9.8}$$

Where,
G_p = minimum green time (sec)
L = crosswalk length (ft, M)
S_p = average speed of pedestrian, usually taken as 4ft/sec (1.28M/s) assumed to present 15th percentile pedestrian walking speed
3.2 = pedestrian start up time
W_E = effective crosswalk width
N_{ped} = number of pedestrians crossing during an interval

9.1.5 Average delay

Average delay using the Webster's uniform delay formula (UD) given in Equation 9.9 or can found from the chart Figure 9.1.

$$UD = \frac{C(1(\frac{g}{c})^2)}{2(1 - (\frac{v}{s}))} \tag{9.9}$$

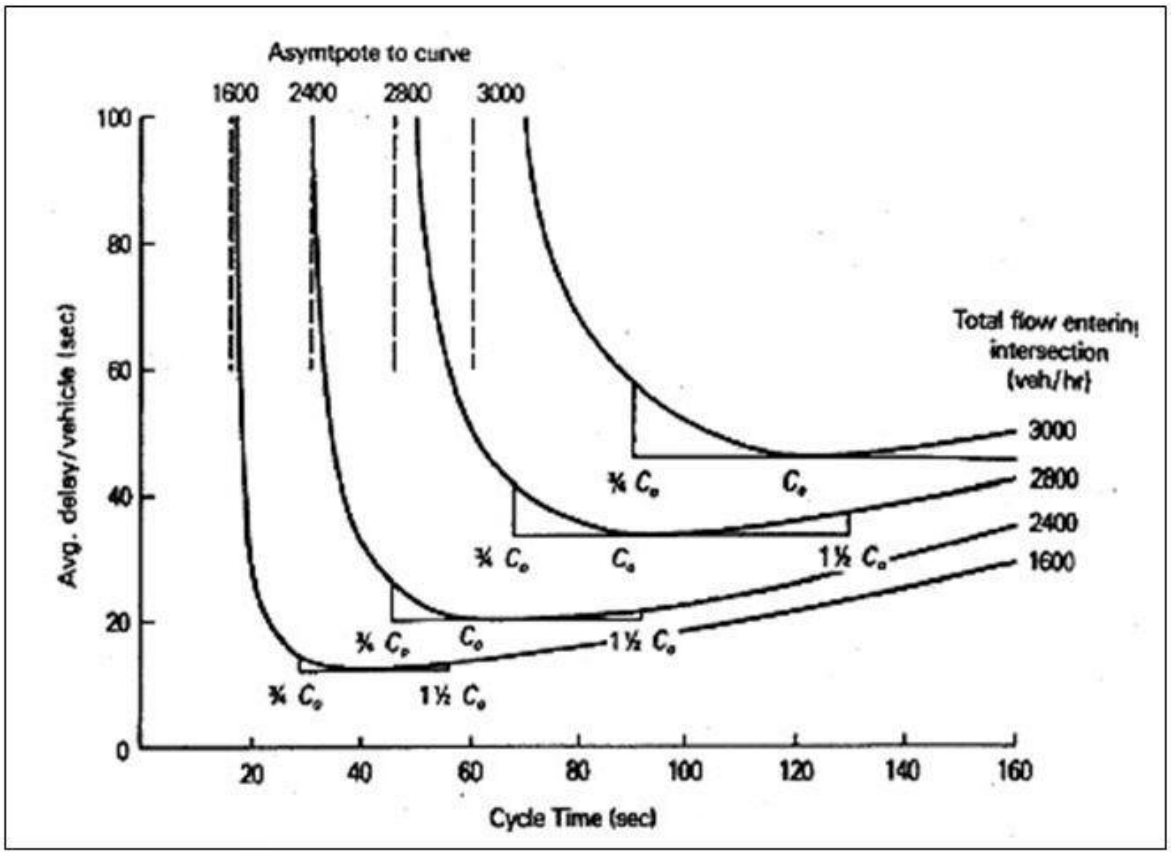

Figure 9.1: Average delay vs Cycle length in Webster

9.1.6 Signal Design Using Webster Algorithm

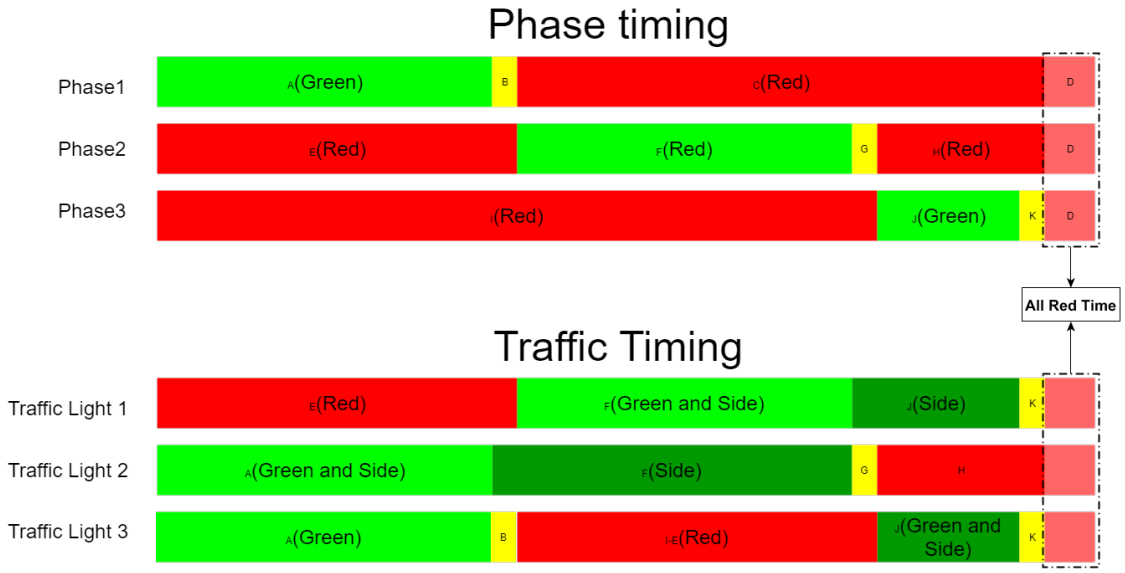

Figure 9.2: Phase Timing Design

Given we calculated cycle Length, green duration, yellow duration, red duration and all red duration from the formulas written above with collated values of traffic flows. We can construct Signal Phase

Diagram for better understand the traffic signal design. One example of such diagram is shown in Figure 9.2.

9.1.7 Implementation of Webster's Algorithm in the project

Webster's method can calculate and design signal timing based on the given traffic volumes of each traffic/vehicle source. In practise the traffic volume of any source varies drastically depending on the time of day, day of the week or even on special occasions. To acknowledge the issue, we can collect traffic volumes of each traffic source at various intervals of a day for every day of the week. And calculate and design the signal timing for the traffic flow, which corresponds to the current time, day of week or even on special occasions.

In the project, we divided a day from 8:00 AM to 8:00 PM into 7 divisions, taking traffic flow data at an interval of 2 hours each for 7 times for each day of the week.

The implementation of Webster algorithm is illustrated in Figure 9.3.

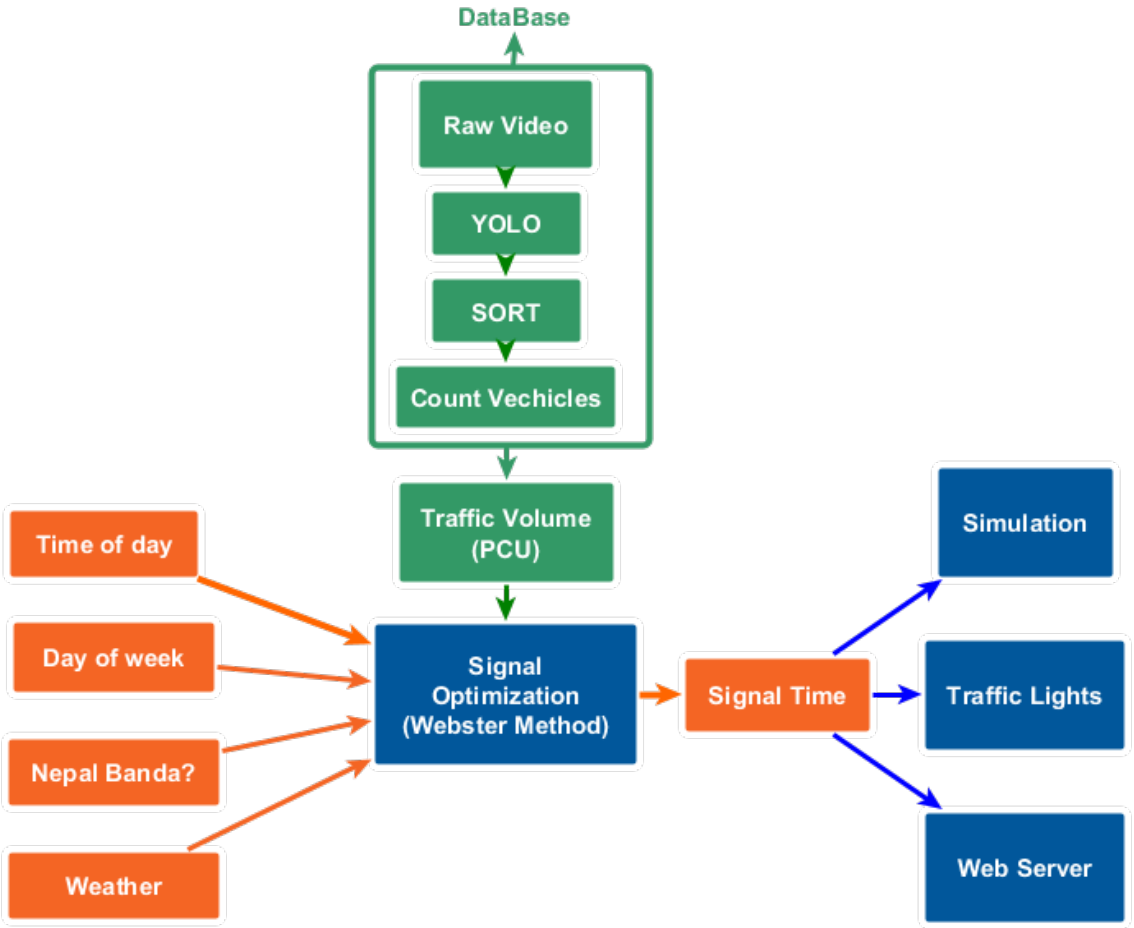

Figure 9.3: Implementation of Webster's Algorithm

10 Results and Simulation

10.1 Simultaion

For the purpose of testing the efficiency of timing sequence generated by the processing the data collected the simulation software PTV Vissim was used. PTV Vissim is a microscopic multimodal traffic flow simulation software package developed by PTV Planung Transport Verkehr AG in Karlsruhe, Germany. Multi-modal simulation describes the ability to simulate more than one type of traffic where all these types can interact mutually. In Vissim the various types of traffic can be simulated such as Vehicles(cars, buses, trucks), public transport, motorcycles, pedestrians etc. The scope of the application ranges from various issues of traffic engineering, signal timing to 3D visualization for illustrative purpose. Figure 10.1 shows the timing sequences given as input to PVT Vissim software. Here, respective color represents the signal color and length of the bar represents the timing of that phase.

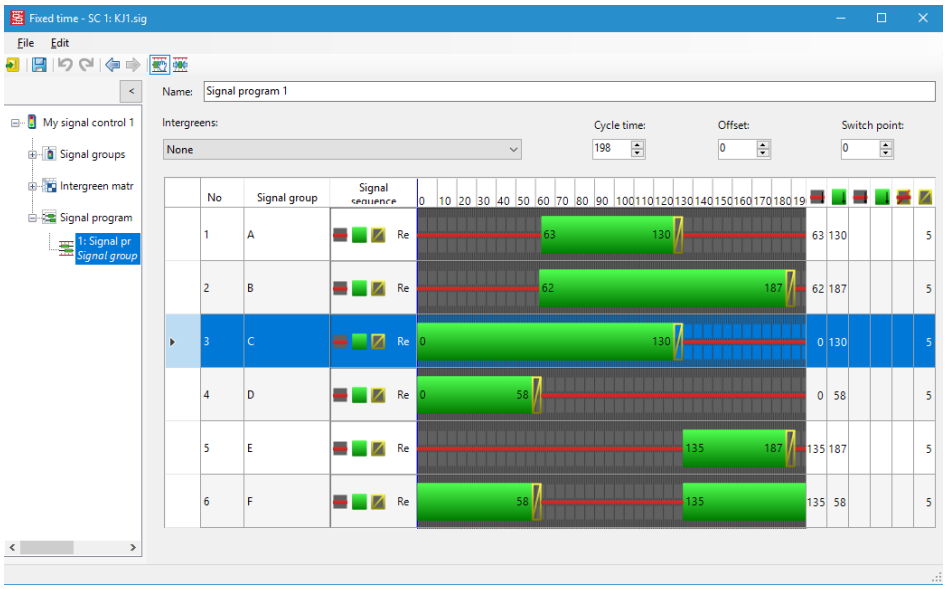

Figure 10.1: Timing Sequence Configuration

Figure 10.2 shows 3D model of three way junction. The amount of vehicle volume was configured in the setting according to the data collected. And the timing sequence for each phase for given volume calculated using Webster method was implemented in the Signal control of the VISSIM as shown in Figure 10.1. Thus the model was implemented in the software and the respective data were collected and observed.

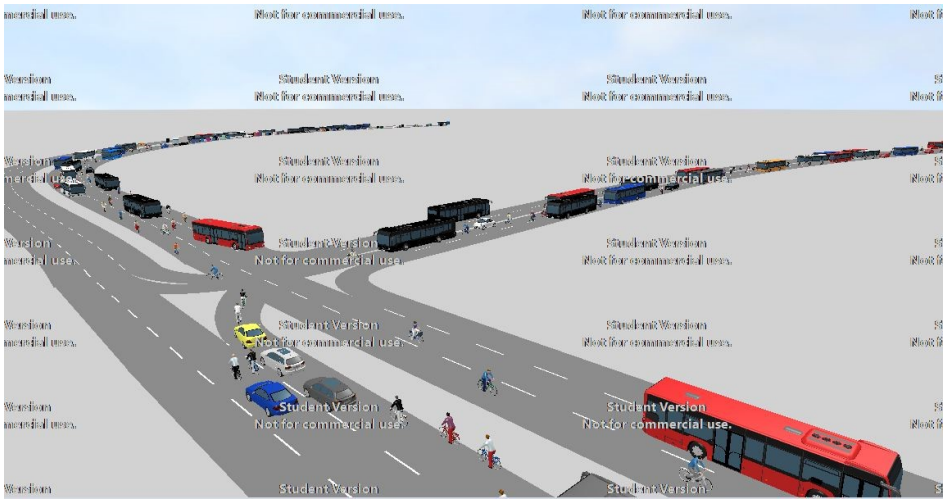

Figure 10.2: 3D visualization of Simulation of Jadibuti Juncction

10.2 Simulation Results

Figure 10.3, Figure 10.4 and Figure 10.5 shows the data collected during the simulation for each road connecting to the junction. The number of vehicles passing through junction and Queue length of traffic in each roads were considered as the parameters to define the efficiency of the model.

Start Page | Network Editor

Data Collection Measurements

Select layout... | <Single List>

Count: 3	Vehs(Current,Total,All)	Vehs(Current,Avg,All)	Vehs(Current,0,All)	Vehs(Current,100,All)	Vehs(Current,200,All)	Vehs(Current,300,All)	Vehs(Current,400,All)	Vehs(Current,500,All)
1	240	34	53	25	54	32	46	30
2	119	17	7	36	2	30	0	44
3	190	27	34	11	69	9	58	9

Figure 10.3: Vehicle count data collection

Start Page | Network Editor

Queue Counters

Select layout...

Count: 3	QLen(Current,Total)	QLen(Current,Avg)	QLen(Current,StdDev)	QLen(Current,Min)	QLen(Current,Max)	QLen(Current,0)	QLen(Current,100)	QLen(Current,200)	QLen(Current,300)
1	1964.14	327.36	219.80	0.00	501.53	0.00	94.34	451.31	461.15
2	1997.29	332.88	147.13	42.54	434.03	42.54	351.48	402.58	424.62
3	1997.97	332.99	198.44	9.12	511.46	9.12	183.88	486.48	450.51

Figure 10.4: Queue length data collection

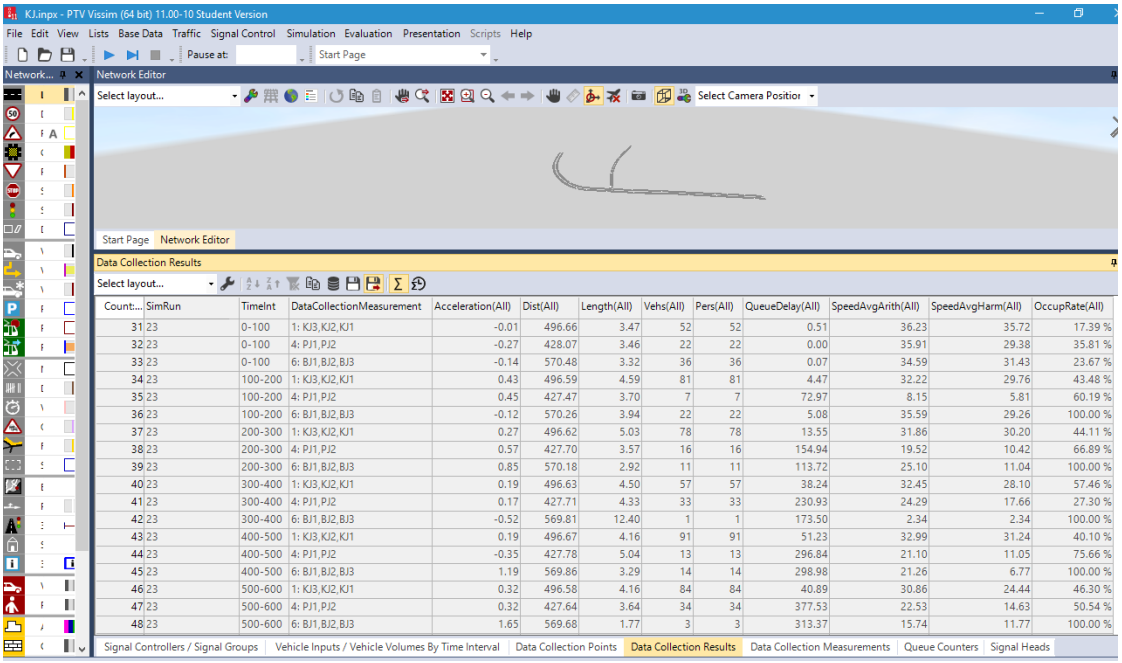

Figure 10.5: Data collection from VISSIM software

The results we achieved form the simulation results above was comparable ...

11 Limitations

A simple prototype for the control of traffic lights of the three way junction was designed in this project. In order to realize this system as a real life application a lot of things has to be improvised in this prototype. Some of the limitations of our prototype are as follows:

- Commercial grade electronic components was used in designing our prototype hence in case of real life application of system industrial grade components are to be implemented. The operating temperature range of commercial grade component is typically -20 to 60 degree Celsius whereas industrial grade components operate over range of -40 to 85 degree Celsius. Industrial grade components may offer various features such as improved signal sensitivity or greater tolerance to electrical noise.
- Since the traffic system is to be implemented in outdoor environment the system must be compatible with environments subject to wider ambient temperature variation, pressure difference etc. It should operate under various kind of environmental conditions such as rainy, windy condition and must be resistant to various environmental factor such as humidity, dust, lightning, rusting etc. This Prototype doesn't fulfill such requirement and is suitable in case of indoor application only.
- The connection between the system and control station server must be made more secure because the system security if is compromised and is altered may result in hazard situation of the traffic.
- Since the efficiency of system is fully based over the result of simulation, the real life application may still be one of the great challenge.

12 Conclusions

This project was not able to achieve any groundbreaking results. The performance shown by automated timing sequence in simulation was not better than the present status of the Traffic. But of course it can suggest a new way to tackle one of the biggest problem of Kathmandu City.

This project was limited by the lack of enough and reliable data, facility/freedom to collect data, sluggish and ignorance behavior of government workings, lack of enough guidance and sufficient time. But this can be further enhanced in higher level who have access to all the facilities and sufficient resources to achieve some good results in this context.

Managing the traffic lights to optimize the vehicle flow is not an easy job. This is due to the dependency of vehicle flow in uncountable and unaddressable number of parameters. The sequence generated might be the optimum sequence for the simulation, but might not be effective in real life scenario. Developed countries uses similar simulation tool to simulate their traffic flow and generate the sequence. They actually uses that sequence in real life which have shown good results too. But, in case of our country, there are lots of randomness in vehicle flow behavior such as non-lane system and unpredictable and selfish behaviors of drivers.

So, a better simulation tool is required to address all these parameters. Moreover, alternative algorithms can be developed instead of using premade algorithms, which might focus the vehicle flow behavior in countries like Nepal and other south Asian countries.