**Department of Electrical and Computer Engineering**

**North South University**



**Senior Design Project**

An Intelligent Traffic Management System for Multi-way Intersections Using Computer Vision Techniques

**Tousif Osman ID # 1220490042**

**ShahreenShahjahan Psyche ID # 1210494042**

**J.M.ShafiFerdous ID # 1030140042**

**Faculty Advisor:**

**Dr. Hasan Uz Zaman**

**Associate Professor**

**ECE Department**

**Spring, 2016**

**Letter of Transmittal**

April, 2016

To

Dr. Arshad M. Chowdhury

Associate Professor and Chairman,

Department of Electrical and Computer Engineering

North South University, Dhaka

**Subject: Submission of Capstone Project on “An Intelligent Traffic Management System for Multi-Way Intersections Using Computer Vision Techniques”**

Dear Sir,

With due respect, we would like to submit Our **Capstone Project Report** on “**An Intelligent Traffic Management System for Multi-Way Intersections Using Computer Vision Techniques”** as a part of our BSC program. The report deals with automated management of traffic system using computer vision and image processing. We tried our level best to make the report meaningful and informative.

The Capstone project was very much valuable to us as it helped us to gain experience from practical field. It was a great learning experience for us. We tried to the maximum competence to meet all the dimensions required from this report.

We will be highly obliged if you are kind enough to receive this report and provide your valuable judgment. It would be our immense pleasure if you find this report useful and informative to have an apparent perspective on the issue.

Sincerely Yours,

.........................................................

Tousif Osman

ECE Department

North South University, Bangladesh

........................................................

Shahreen Shahjahan Psyche

ECE Department

North South University, Bangladesh

.........................................................

J.M. Shafi Ferdous

ECE Department

North South University, Bangladesh

**Approval**

The capstone project entitled “**An Intelligent Traffic Management System for Multi-Way Intersections Using Computer Vision Techniques**” by Tousif Osman(ID#1220490042), Shahreen Shahjahan Psyche(ID#1210494042), J.M. Shafi Ferdous(ID#1030140042) is approved in partial fulfillment of the requirement of the Degree of Bachelor of Science in Computer Science and Engineering on April, 2016 and has been accepted as satisfactory.

**Supervisor:**

**Dr. Hasan Uz Zaman**

**Associate Professor**

Department of Electrical and Computer Engineering

North South University

Dhaka, Bangladesh.

**Student’s Declaration**

This is our truthful declaration that the **“Capstone Project Report”** we have prepared is not a copy of any **“Capstone Project Report”** previously made by any other team. We also express our honest confirmation in support of the fact that the said **“Capstone Project Report”** has neither been used before to fulfill any other course related purpose nor it will be submitted to any other team or authority in future.

.........................................................

Tousif Osman

ECE Department

North South University, Bangladesh

........................................................

Shahreen Shahjahan Psyche

ECE Department

North South University, Bangladesh

.........................................................

J.M. Shafi Ferdous

ECE Department

North South University, Bangladesh

**Acknowledgement**

First of all, we wish to express our gratitude to the Almighty for giving us the strength to perform our responsibilities and complete the report.

The capstone project program is very helpful to bridge the gap between the theoretical knowledge and real life experience as part of Bachelor of Science (BSc) program. This report has been designed to have a practical experience through the theoretical understanding.

We also acknowledge our profound sense of gratitude to all the teachers who have been instrumental for providing us the technical knowledge and moral support to complete the project with full understanding.

We would like to convey my gratitude to my faculty **Dr. Hasan Uz Zaman** for his stimulating inspiration, kind guidance, valuable suggestions, sagacious advice and kind co-operation throughout the period of word undertaken, which has been instrumented in the success of our project. At this level of understanding it is often difficult to understand the wide spectrum of knowledge without proper guidance and advice. His suggestions & guidance have made the report a good manner.

We thank our **friends and family** for their moral support to carve out this project and always offer their support.

We would also thank “Tech Shop BD”, “Suzuki Electronics”, and “Tools BB” for providing the necessary equipment at our doorsteps.

**Abstract**

This paper presents the design and implementation of an intelligent and automated traffic control system which takes advantage of computer vision and image processing techniques. This system should help reduce traffic jams and congestion at busy road intersections. It detects the number of vehicles on each road near a node where a node represents an intersection with traffic lights. Depending on the number of vehicles on a specific node and its surrounding nodes, this system assigns optimized amount of waiting time (red signal) and driving time (green signal). It is a fully automated system which replaces the conventional pre-determined fixed-time based traffic system with a dynamic traffic system that auto-adjusts according to the changing road conditions. The designed system should help solve traffic problems in busy cities to a great extent thus saving a significant amount of man-hours that gets lost waiting on jammed roads.

Table of Contents

[Chapter 01: Overview 5](#_Toc447020596)

[1.1 Introduction 6](#_Toc447020597)

[1.2 Traffic Management System with cause and effect of Traffic Jam 6](#_Toc447020598)

[1.3 Some Statistics Around the world regarding Traffic Jam 8](#_Toc447020599)

[1.4Proposed Solution 9](#_Toc447020600)

[1.5 Summary 10](#_Toc447020601)

[Chapter 02: Motivation 11](#_Toc447020602)

[2.1 Introduction 12](#_Toc447020603)

[2.2 Motivation towards Our Project 12](#_Toc447020604)

[2.3 Summary 12](#_Toc447020605)

[Chapter 03: Related Works 13](#_Toc447020606)

[3.1 Introduction 14](#_Toc447020607)

[3.1 Works related to our project 14](#_Toc447020608)

[3.4 Summary 15](#_Toc447020609)

[Chapter 04: Theory 16](#_Toc447020610)

[4.1 Introduction 17](#_Toc447020611)

[4.2 Optimal Time computation 17](#_Toc447020612)

[4.3 Car detection with Computer Vision 18](#_Toc447020613)

[4.4 Light intensity varying Image detection algorithm 21](#_Toc447020614)

[4.5 Summary 22](#_Toc447020615)

[Chapter 05: System Design 24](#_Toc447020616)

[5.1 Introduction 25](#_Toc447020617)

[5.2 Design Details 25](#_Toc447020618)

[5.3 Summary 29](#_Toc447020619)

[Chapter 06: Software Design 30](#_Toc447020620)

[6.1 Introduction 31](#_Toc447020621)

[6.2 Software design in details 31](#_Toc447020622)

[6.3 Summary 35](#_Toc447020623)

[Chapter 07: Components and Assembly or Setup 37](#_Toc447020624)

[7.1 Introduction 38](#_Toc447020625)

[7.2 Details about Components and the setup 38](#_Toc447020626)

[7.3 Arduino Yun 40](#_Toc447020627)

[7.4Arduino UNO 42](#_Toc447020628)

[7.5 CAM Shield 47](#_Toc447020629)

[7.6 Router 48](#_Toc447020630)

[7.7 MATLAB 49](#_Toc447020631)

[7.8 Summary 52](#_Toc447020632)

[Chapter 08: Operation of The System 54](#_Toc447020633)

[8.1 Introduction 55](#_Toc447020634)

[8.2 Fetching Images Data 55](#_Toc447020635)

[8.3 Receiving and Analysis of the Image on the Sever 55](#_Toc447020636)

[8.4 Sending Signal to the Traffic Lights 56](#_Toc447020637)

[8.5 Summary 57](#_Toc447020638)

[Chapter 09: Results and Discussion 59](#_Toc447020639)

[9.1 Introduction 60](#_Toc447020640)

[9.2 Simulation 60](#_Toc447020641)

[9.3 Results from Demo 64](#_Toc447020642)

[9.4 Summary 64](#_Toc447020643)

[Chapter 10: Future Work 65](#_Toc447020644)

[10.1 Introduction 66](#_Toc447020645)

[10.2 Future Work 66](#_Toc447020646)

[10.3 Summary 66](#_Toc447020647)

[Chapter 11: Costing 68](#_Toc447020648)

[11.1. Introduction: 69](#_Toc447020649)

[11.2. Arduino Uno, Ethernet Shield, Camera Shield 69](#_Toc447020650)

[11.3. Arduino Yun, Arduino Uno, Web camera 69](#_Toc447020651)

[11.4. Raspberry Pi, Web camera 69](#_Toc447020652)

[11.5. The Server 70](#_Toc447020653)

[11.6. Summary 70](#_Toc447020654)

[Chapter 12: Compliance with Standards 71](#_Toc447020655)

[12.1 Introduction 72](#_Toc447020656)

[12.2 Compliance with IEEE standard 72](#_Toc447020657)

[12.3 Compliance with US standard 72](#_Toc447020658)

[12.4 Compliance with European standard 73](#_Toc447020659)

[12.5 Summary 73](#_Toc447020660)

[Chapter 13: Design Impact 74](#_Toc447020661)

[13.1 Introduction 75](#_Toc447020662)

[13.2 Environmental Impact 75](#_Toc447020663)

[13.3 Economic Impact 75](#_Toc447020664)

[13.4 Social Impact 75](#_Toc447020665)

[13.5 Ethical Impact 76](#_Toc447020666)

[13.6 Manufacturability 76](#_Toc447020667)

[13.7 Sustainability 76](#_Toc447020668)

[13.8 Summary 76](#_Toc447020669)

[APPENDIX A 77](#_Toc447020670)

[MATLAB Codes 77](#_Toc447020671)

[A.1 Code of Image Read 78](#_Toc447020672)

[A.2 Code of Capturing Image 78](#_Toc447020673)

[A.3 Code of Counting Cars 78](#_Toc447020674)

[A.4 Code of Running The System 78](#_Toc447020675)

[APPENDIX B 80](#_Toc447020676)

[Nodes Codes **Error! Bookmark not defined.**](#_Toc447020677)

[B.1 Codes Related to Arduino 81](#_Toc447020678)

[B.2 Codes Related To Raspberry Pi 84](#_Toc447020679)

[APPENDIX C 88](#_Toc447020680)

[Technical Manual 88](#_Toc447020681)

# Chapter 01: Overview

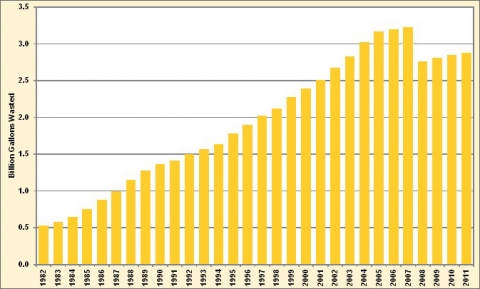
## 1.1 Introduction

“Traffic Jam” is a very common phenomenonin our daily lives. This is specially a huge problem in congested cities like Dhaka where traffic jam is actually one of the biggest social problems. A study has found that people lose their precious 8.15 million hours, 40 percent of which are working hours –in traffic jams [2]. In cities of developing countries, traffic is controlled in either of two ways, either the lights change at a pre-determined time intervals or the traffic police does the work manually. Both of them are wasteful processes. Sometimes in a junction we see that one side has lower congestion than the other side, but as the lights are changing at fixed time intervals, the jam on the road which has higher congestion keeps getting worse. On the other hand, most of the traffic polices are unaware of the situation at places away from him/her. It is also very hard to look at every side of a junction constantly by one person and decide correctly how to guide the traffic. That is why we are proposing a better solution to this problem. Using the power of Artificial Intelligence (AI), we can build a system which can decide which road needs to be cleared quickly and change the time intervals for signal change accordingly.

## 1.2 Traffic Management System with cause and effect of Traffic Jam

The traffic management systems of Bangladesh and the Indian Subcontinent are those that were adopted in the 1980s when the country’s population was only 87 million, and now is more than 160 million.More than 20 million people live in Dhaka, the capital city of Bangladesh. As a consequence, Dhaka city’s traffic congestion problem has grown to alarming proportions, and it is one of the most challenging issues. Studies show that on an average day, vehicles are stopped for about six hours in total for trains, traffic signals at intersections and just because of congestion, too many vehicles trying to move on too few roads. According to Abeda Sultana“New methods of traffic management are not being updated to accommodate the population growth, and as a result, the growing traffic jams are making the lives of normal people more difficult. The causes of traffic congestion in Dhaka are many. Starting from the city center, the skeleton, the structure and the lay-out of Dhaka’s roads are not well-planned and well-directed, and the population overwhelms the capacity of the road system. While the Bangladesh government mandates 25 percent of all roads to be paved, Dhaka has only 7.5 percent. Even more, 30 percent of the paved roads have additional obstructions to traffic: hawkers, salesmen, construction materials, waste containers and road-side stands that are on the pavement rather than on the shoulder of the road. As a result, vehicles have even fewer lanes to use. Dhaka has several types of vehicles: public buses, taxis, microbuses, private vehicles, motorcycles, rickshaws and more. The increasing population increases the demand for more vehicles on the streets, and the result is both motorized and non-motorized vehicles occupy the same streets at the same time. Rickshaws are a common reason for traffic jams. The number of rickshaws is very high and they don’t follow any traffic laws. However, the rickshaws cannot be eliminated because they are both inexpensive and the greenest vehicle on the street. But their slow speed creates chaos and congestion. The public transport system in Dhaka is not adequate or properly-routed. The three major bus stations, Sayedabad, Gabtoli and Mohakhali do not have sufficient capacity to accommodate all the buses coming in from other districts, nor enough buses to handle all the people who want to use the buses for transportation. As a result, mini-buses, microbuses and private vehicles that can only carry a few passengers at a time add to the traffic congestion. Limited parking arrangements are another major cause of excessive traffic in Dhaka. It has become a regular practice to park cars on roads because businesses and apartments do not construct designated parking areas. In addition to the sources of traffic congestion, the actual condition of the roads contributes to the transportation difficulties. In most cases roads are winding and twisting, which results in a larger number of intersections. Lack of proper maintenance means drivers are likely to stop or swerve unexpectedly to avoid bad sections of the road. During the rainy season, the situation becomes more critical because the roads are sometimes under water due to heavy rainfall”.

## 1.3 Some Statistics Around the world regarding Traffic Jam



**In 2011, 2.88 billion gallons of fuel were wasted due to traffic congestion**

**So the Yearly loss = 2.88 \* 2.318 billion USD**

**= 6.67 billion USD**

## 1.4Proposed Solution

Building an intelligent system that will observe the situation of the vehicles around a signal and it will give back a fixed timing to release a road depending on the nodes(traffic signals) around that specific node.

By managing the traffic signals automatically based on time management, will lower the congestion of vehicles and less man power will be needed to maintain the system and it will be cost effective.



## 1.5 Summary

This chapter gives us insights about how traffic jam is hampering our daily life, losses due to traffic jam, statistics related to traffic jam. We projected our thoughts upon how traffic system is being controlled in our country and worldwide. Lastly we have proposed our solution towards traffic jam and traffic management system.

# Chapter 02: Motivation

## 2.1 Introduction

We, the urban people who live in cities, are feeling the disastrous effects of traffic jam in our daily life. Being city dwellers, we thought that we should deal with this problem immediately**.**

## 2.2 Motivation towards Our Project

We intent to make a system that minimizes the waiting and maximizes the running time of traffic lights. The system is intended to identify number of vehicle on each traffic node with computer vision. All the traffic signals is connected to a central server. System considers roads leaving a traffic signal as outgoing edge and roads coming towards a traffic signal as incoming edge. While determining time for a single traffic node, system considers first Childs by default of nodes connected via incoming and outgoing edge to provide better time calculation. By considering number of waiting cars on each road connected to a traffic junction system computes minimal waiting and running time for a specific node.

## 2.3 Summary

This chapter gives the idea about the motivations towards our project which are to reduce traffic jam by replacing the static system with a dynamic system. This will help in increasing the running time and reducing the waiting time of vehicles.

# Chapter 03: Related Works

## 

## 3.1 Introduction

While working in this research project, we have encountered several research papers which have been conducted on traffic, analysis of traffic and optimization of traffic. Among these papers, two of them are closely related to our research. Both of them have used Artificial Intelligence to solve the problem.

## 3.1 Works related to our project

In one of them, research has been done using the methods of fuzzy logic to solve traffic congestion problem [3]. Fundamentally they have introduced a fuzzy controller system for an isolated four-way junction. The fuzzy controller system takes an input variable arrival which actually represents the number of cars present in the running road in a certain parameter and another input variable queue which represents the number of the cars in the waiting area. The output is basically the extension time of the running road. This total system is quite distinct from us but solves almost same problem but the main difference is they are working on a single isolated junction and calculating the extension time accordingly while we are actually considering other junctions which are directly connected with the certain junction. By doing this we are actually calculating the running time of a single node more precisely according to the neighborhood area. Another difference is, though they are working in a four junction way in their system they are actually considering two situations in their system- Traffic from the north &south and traffic from the east & west. They aren’t considering right and left turns but our designed system will do so. We are considering each road connected to a junction as a single entity and hence arbitrary number of roads can contour a traffic junction.

On the other hand, the second research [4] we are referring has used image analysis to solve the problem, which is more close to our research proposal. In this research they have worked on live video streams of roads. Their vehicle detection algorithm is very accurate as they have first identified whether the object is moving or not and after that they have used edge detection on those objects and after that they got the total number of vehicles. In night mode they have used headlights instead of vehicle itself and hence the system is very precise at night as well. Though this algorithm is very good in terms of vehicle counting accuracy but it comes with a good cost of processing power. To run the above algorithm in real time we need a significant amount of processing power to analyze the video feeds of an entire city.

## 3.4 Summary

Considering all the above discussions in mind we have only used pattern matching algorithm which is pretty straight forward and less heavier algorithm. We have analyzed single image for a car count in a given time which is more cost effective as we don’t want car count in every second for traffic optimization. As a result, the system is much optimized in terms of processing power and hence the system allows the server to handle more traffic. However, to active this car counting accuracy drops slightly but we can ignore the fact as we doesn’t need accurate number of vehicle to provide efficient and effective traffic management system.

# Chapter 04: Theory

## 4.1 Introduction

Primary areas of this research is to optimize time by using the vehicle number on nearby connected traffic signal, identify number of vehicles using computer vision, develop a light intensity varying mechanism to make the vision algorithm to work independent of sounding environment.

## 4.2 Optimal Time computation

Initially all node in the system have infinite waiting time and zero running time. System has a starting node predefined in the network. The system also has two thresholds for waiting and running time for a specific junction of nodes where nodes are connected by incoming and outgoing edges. For determining the waiting time for a specific node the system considers nodes connected to the junction with incoming edge and number of waiting cars on those nodes. The system sums up the number of waiting cars and makes a tentative ratio for each node. Finally, the system divides the threshold with the assigned ratio for each node connected in the junction and determines waiting time. For running time using the same algorithm for waiting time the system assigns running time for each node connected in the junction. In this case system considers incoming nodes instate of outgoing nodes.

Computation on a junction:

αW< waiting threshold >

αR< running threshold >

Sout =

Sin =

Tw = αW. (1)

TR = αR . (2)

. Where αW waiting time threshold and αR is the running time threshold. Sout represents sum of all waiting cars on the outgoing nodes and Cout represents number of waiting cars on each outgoing nodes. Again Sin represents sum of all waiting cars on the incoming nodes and Cin represents number of waiting cars on each incoming node. W is the number of waiting cars on the current node. Tw is the waiting time for the current node and TR is the running time for current node.

## 4.3 Car detection with Computer Vision

For car detection, the system takes images of free road on a specified time interval depending on the traffic pattern of the specific road. This suitable time interval is need to be predefined during the system setup. To identify car on a given time the system takes a snapshot of the road with vehicles and subtract it with the image of empty road. If absolute of subtracted is compared with a small threshold value pixels congaing cars is identified. Afterwards if we consider connected comments of the image individual cars on the image is identified. Finally, total number of cars on a given image can be identified by counting individual cars.

[d] = [Image without cars] – [Image with cars]

[η, c] = connectedComponents([d])

T =

Here square brackets represent matrix. η, represents number of connected components and c represents all the identified connected component as a matrix. We are considering “connectedComponents([d])” as a function that take matrix as input and cluster out the connected regions and returns a matrix with a one-degree greater detention containing each cluster.

We have collected images of road and analyses those images with our algorithm. Fig.1 is an image of empty road; Fig.2 is an image of same road with on the same position vehicles. Fig.3, is

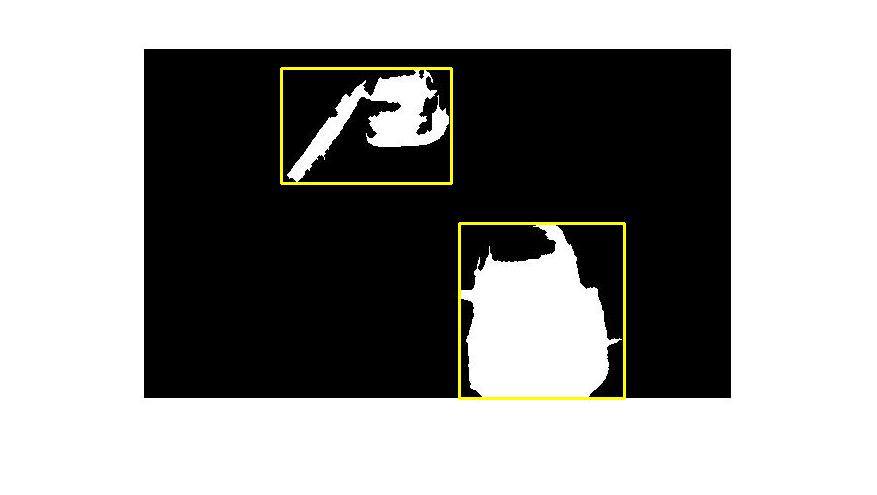
the resulting image after image analysis.



1. Image of empty road



1. Image of road with vehicles



1. Resulting image with idefied vehicles after image analyses

## 4.4 Light intensity varying Image detection algorithm

To identify vehicles, image of empty road is needed for our algorithm but it is not possible to achieve empty road image on every signal event. Instate of taking images on every signal event, image of empty road is taken on a specific interval, then adjusted and finally is used in the car computation. Signal nodes have light sensors. Measuring the presence of light of the original image and current time system will adjust the image of no vehicles.

In the low light condition where vehicle is barely visible, instead of identifying vehicles itself system identifies headlights of cars and count those and dividing the total number of lights by 2 gives a tentative measurement of vehicles on the traffic node

In the situation of light intensity change during the day time system takes small slice of the empty road image and take similar region from the new road image with vehicles and by subtracting these two we get the intensity change. Using this value we adjust the intensity of the subtracted image.

## 4.5 Summary

In this chapter we have discussed about the car detection algorithm that we have used in our system. It calculates the waiting time and running time of cars in a traffic junction. We have designed a car detection algorithm which takes empty image of a road and image of that road with vehicles. Through image processing it detects and then counts number of cars on a road. We have also designed a light intensity varying algorithm considering different natural conditions.

# Chapter 05: System Design

## 5.1 Introduction

There are two major components of the system. Small embedded device to control the traffic lights and capture images from the road. Another component to process images and performs time optimization centrally. We have considered the embedded device as traffic node and distant central component as server.

## 5.2 Design Details

We have designed the system such a way that the image processing is done on the server and the processing the image on the embedded devices will increase the cost of the node and hence will increase the cost of the entire system in a broad scale. Processing the image on the server makes the system cost efficient and at the same time this will allow the system to process more images simultaneously with a lesser amount of time. To build the network between server and the node we have used http protocol and network techniques as it has well defined communication protocols for images and other data as well. To reduce the power consumption of each node, instate of allowing direct wireless communication with the server the nodes in a junction is be connected with a single router and that router communicates with the server wirelessly or via wire.

A traffic node contains four different components. They are one camera, three signal lights, an Ethernet interface and a microcontroller and a junction has an additional component, a router. Our system counts car by analyzing images of street and to do so we need a component capture images and hence we need cameras. In our system design we have used VC0706 cameras. Cameras capture images depending on the instructions provided by the server. The camera in a node is connected to a microcontroller. The microcontroller acts as the single platform on which, the traffic node is designed. As microcontroller we prefer to use ATmega328P microcontroller which has 14 input-output pins. Using these pins, a camera is communicating with the microcontroller. Arduino UNO has this microcontroller integrated in it. So we have used Arduino UNO as microcontroller platform. We have chosen Arduino over other microcontrollers because we need a controller which can command the camera shield to capture an image and sent it to the server. We don’t need powerful microcontroller for our purpose. Hence Using Arduino is energy and cost effective. An Ethernet interface is used to connect the microcontroller with the internet through the router. For this purpose, we have used W5100 Ethernet Shield which is an Arduino compatible Ethernet module. The Router connects all of the nodes around a junction. It establishes a connection between the server and the traffic junction using http protocol. Signal lights of three different colors remains on or off during the time interval provided by the software, which is calculated using the Optimal Time Calculation algorithm. We are using LED lights as signal lights.

Each road connected to a traffic junction has its traffic controller. Each junction has a wireless transmitter to broadcast images to server. Server has a graph of the traffic system. Each listed node on the graph has their location info and associated MAC\IP address. Initially controller device is set to red light. Server software requests for live image to traffic controller on certain time division via its IP. On request traffic controller returns live image feed to the IP/PORT from which the request was sent. On receiving image server processes it compute number of vehicles and forward to traffic management section of the software which computes optimal time for the traffic. Software sets clock for the specific node and send red light signal to controller device. Traffic controller device will set the light indicated by the server and holds its state until further instruction from the server. After the computed time elapse

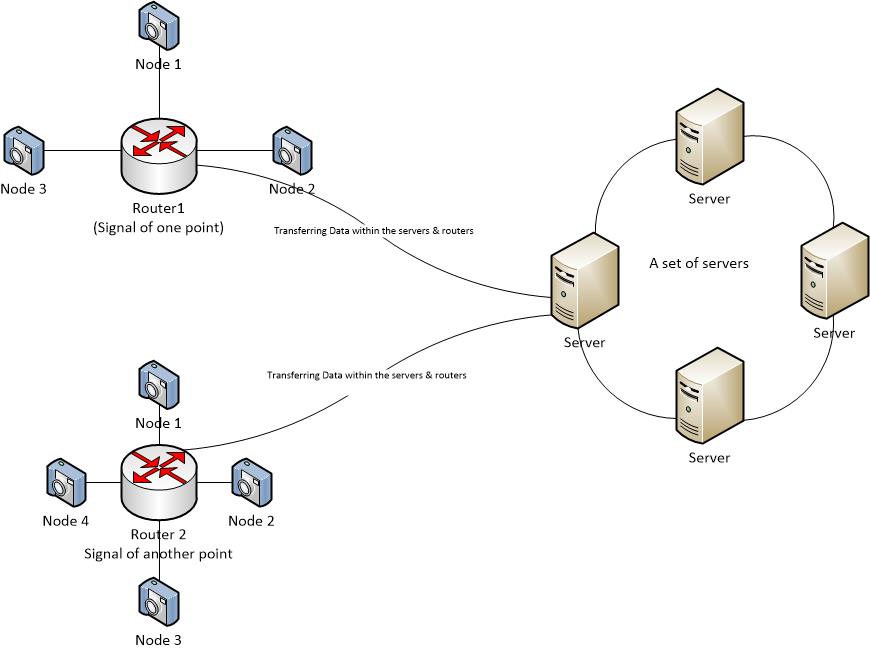
## 

server sends controller device to set green light. Fig.4 describes the system, its components and its actions.



1. System Design

All the traffic nodes are connected to a server. Server takes all the decision and commends the traffic nodes. Server has software to manage the system. The Server software is based on Java platform. We are using MATLAB for image processing.



1. Network Diagram

Server device is connected to the network and communicates with the traffic nodes on TCP/IP port. In the server all the traffic nodes need to be registered and mapped in its memory. Server will identify each node uniquely by their MAC address. To instruct a specific node server will use its MAC address to get the IP address from the data map and using that IP address server will communicate with the embedded device. When the system starts up, initially it commands all the connected nodes to light up red light. Upon calculating optimized waiting time and running time server can command specific node to perform specific task. Instead of using only one server grid of server we can used where every server is connected with its adjacent server and makes a server web. In Fig.5, we can see that instead of using any one server we are using a grid of servers.

## 5.3 Summary

In our system design, the embedded device is made using an arduino microcontroller as it’s base. On top of the microcontroller, we have an Ethernet shield and on top of those we have a camera shield. In a function four of these embedded device are connected altogether using a router. This router is connected with the server.

# Chapter 06: Software Design

## 6.1 Introduction

Two separate programs have been constructed to design the system. An embedded program has been written for the microcontroller. This program is written in C++. This software uses standard interfacing libraries to interface the components with the microcontroller.

## 6.2 Software design in details

Embedded software opens a socket in TCP/IP port and creates a socket server which listens on the port for instruction. A separate program runs the server. Server software is the heart of the system. It is designed to process the images and calculated the optimal time for several junction at the same time. It can serve multiple nodes at the same time on different IP ports.



1. Graph Representation of nodes

System is implemented with object oriented software design. Every traffic node is represented as an object in the software. Road to road connection is represented as directed edge. Three or four node combined together will form a junction.

For example, if there is a junction where 4 roads are connected then we can say that a single rode on the junction can go to 3 different road and it is represented by 3 different edge. So for 4 roads there are 12 directed edges. Every incoming node has at least 1 incoming edge and at least 1 outgoing edge. All the objects and edges are represented as a graph in the system. Fig. 3 describes the software representation of the system. Dashed Square in Fig. 6 represents a 4 road connecting junction.

The software has two sections. One Section is to retrieve image, process it drive vehicle count and optimized time computation section to compute waiting time for each signal controller. These two run on different threads.

Image of Empty Street is subtracted from the image from the input stream to reduce unnecessary information. Use pattern matching algorithm to identify all rectangle shaped patterns on the image. Rectangles between specific thresholds can be considered as vehicles. Count number of vehicles and send it to the optimize time calculation section of the software to compute waiting time.



1. Processing Software Diagram

Time calculation section seeks nearby node’s waiting vehicles for a given node and makes a ratio of cars. Default time slice is multiplied with the ratio and assigns time for the given node. This calculation will be further researched for further optimizing. Processing flow of the system is represented in Fig. 7. We can see that incoming image as input signal and each stage it passes through the system to acquire vehicle count to optimized time computation and finally provide light activation signal as output.

## 6.3 Summary

At first system will take the images of empty road around a junction where the image taking call will have done by the software. Then after fixed time division, system will take images of road with vehicles. Then using image processing algorithm which uses pattern matching to identify shapes, the software will identify square shaped object and then count the number of cars in roads. Then using the number of cars, the software will calculate optimal waiting and running time for each of the road around a traffic junction. Then the software will send signals for activating signal lights according to the optimally calculated time.

# Chapter 07: Components and Assembly or Setup

## 7.1 Introduction

We need a system where we can capture live image from the traffic signal and that can be sent to the server over a wireless transmission. In our research we have used the below components to construct the signal nodes:

• Arduino

• Ethernet Shield

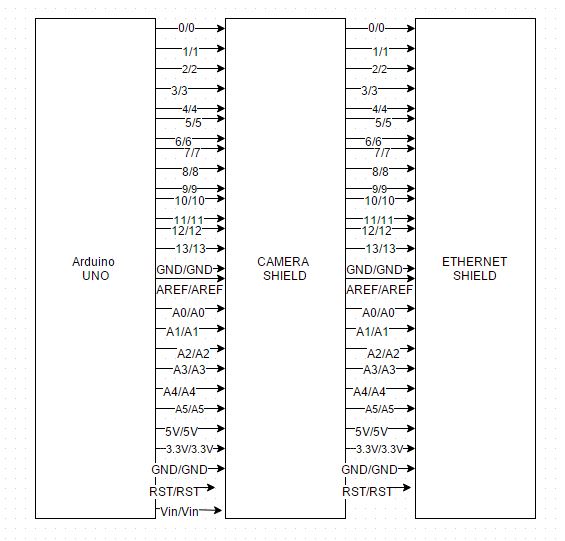
• Camera Shield

• Router

The connections between the devices are shown in Fig.8.

## 7.2 Details about Components and the setup

As the Arduino board and the components Ethernet board and the camera board all three of them are directly pluggable we have just stack them on top another and attached their pins. Manufacturers of these boards provide necessary library and schematics. This is one of the reasons for selecting these boards over others. Finally, three led lights have been attached on pin 5, 6, 7.



1. Components and assembly diagram

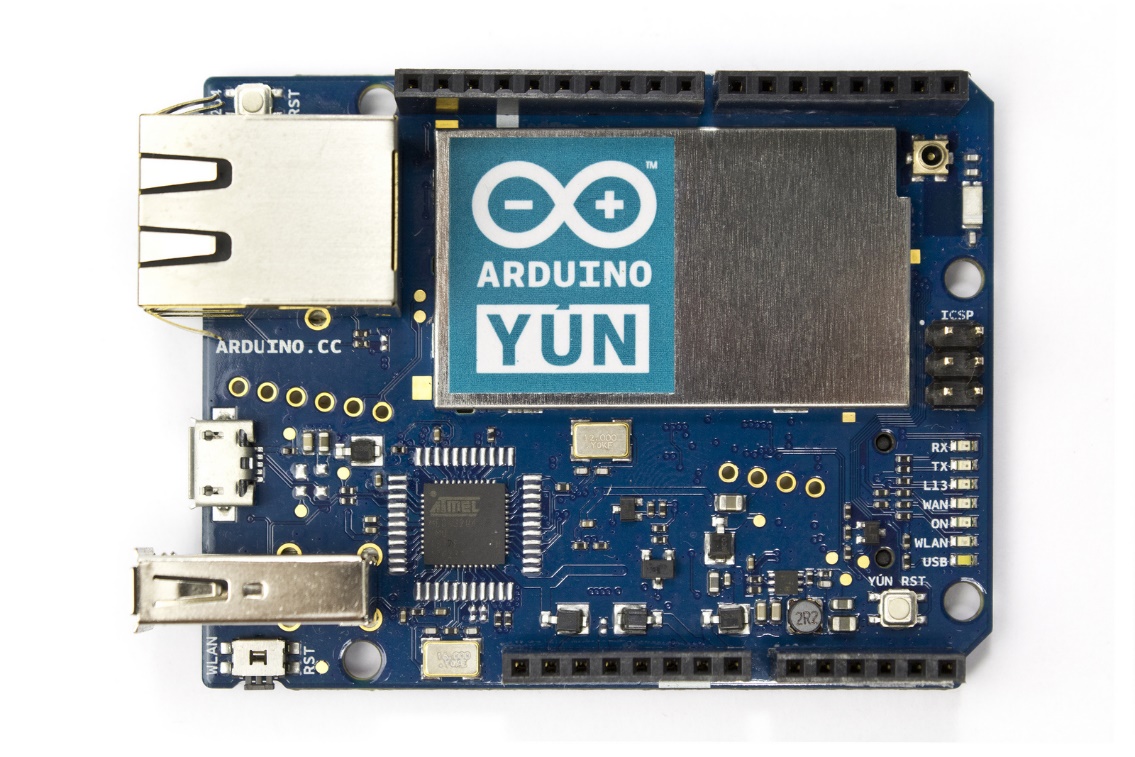
Ether shield has an onboard Ethernet port, what has been connected to a router. As in our research we have implemented two junction we have used two routers. We have connected nodes Ethernet port with associated router’s Ethernet port. As there can be only be 3 or 4 nodes connect to a single node a router having 4 Ethernet port is good enough for this purpose. To keep the development setup simple we build the system by using wireless routers and connect those routers to a wireless Ethernet network. Regarding these facts we have used TP-Link WR740N router which a 5 port Wi-Fi router. This router communicates with the server over Wi-Fi network.

We are using simple wired Ethernet lines from the nodes to the router of a junction. It’s because in a junction the nodes won’t be much far away from each other and we can easily connect it to the router and as we are using Ethernet cables to connect, image passing will be both cost effective and time saving.

Least but the major component of the system, Server, is connected with the system by its onboard Wi-Fi card and we have complete system where all the components are connected in the network.

## 7.3 Arduino Yun

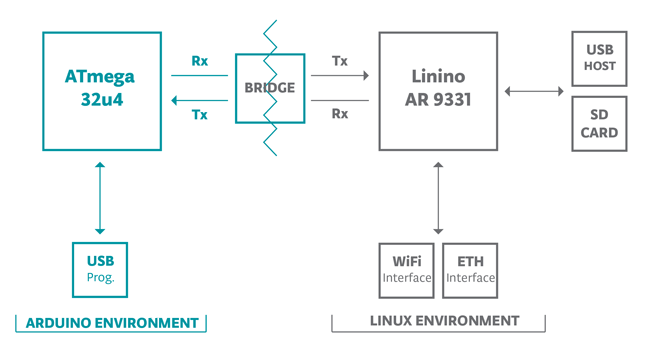
The ArduinoYún is a microcontroller board based on the [ATmega32u4](http://www.atmel.com/Images/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf) and the [Atheros AR9331](https://www.openhacks.com/uploadsproductos/ar9331_datasheet.pdf). The Atheros processor supports a Linux distribution based on OpenWrt named OpenWrt-Yun. The board has built-in Ethernet and WiFi support, a USB-A port, micro-SD card slot, 20 digital input/output pins (of which 7 can be used as PWM outputs and 12 as analog inputs), a 16 MHz crystal oscillator, a micro USB connection, an ICSP header, and a 3 reset buttons.



1. Arduino Yun

The Yún distinguishes itself from other Arduino boards in that it can communicate with the Linux distribution onboard, offering a powerful networked computer with the ease of Arduino. In addition to Linux commands like cURL, you can write your own shell and python scripts for robust interactions.

The Yún is similar to the Leonardo in that the ATmega32u4 has built-in USB communication, eliminating the need for a secondary processor. This allows the Yún to appear to a connected computer as a mouse and keyboard, in addition to a virtual (CDC) serial / COM port.



1. Arduino Yun Schematics

The Bridge library facilitates communication between the two processors, giving Arduino sketches the ability to run shell scripts, communicate with network interfaces, and receive information from the AR9331 processor. The USB host, network interfaces and SD card are not connected to the 32U4, but the AR9331, and the Bridge library also enables the Arduino to interface with those peripherals.

The memory on the AR9331 is not embedded inside the processor. The RAM and the storage memory are externally connected. The Yún has 64 MB of DDR2 RAM and 16 MB of flash memory. The flash memory is preloaded in factory with a Linux distribution based on OpenWrt called OpenWrt-Yun. You can change the content of the factory image, such as when you install a program or when you change a configuration file. You can return to the factory configuration by pressing the "WLAN RST" button for 30 seconds. The OpenWrt-Yun installation occupies around 9 MB of the 16 MB available of the internal flash memory. It is not possible to access the I/O pins of the Atheros AR9331. All I/O lines are tied to the 32U4.

Each of the 20 digital pins on the Uno can be used as an input or output, using pinMode(),digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms.

The Yún has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega32U4 provides a dedicated UART TTL (5V) serial communication. The 32U4 also allows for serial (CDC) communication over USB and appears as a virtual com port to software on the computer. The chip also acts as a full speed USB 2.0 device, using standard USB COM drivers. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB connection to the computer.

## 7.4Arduino UNO

The Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the Arduino index of boards.

The Uno can be programmed with the Arduino Software (IDE). Select "Arduino/Genuino Uno" from the Tools > Board menu (according to the microcontroller on your board). For details, see the reference and tutorials.

The ATmega328 on the Uno comes preprogrammed with a boot loader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

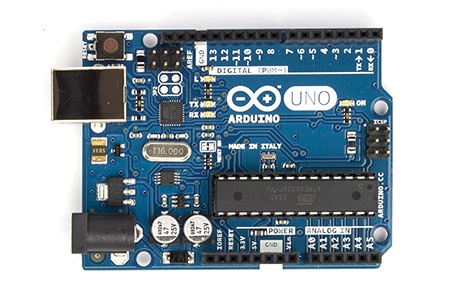
You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using Arduino ISP or similar; see these instructions for details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available in the Arduino repository. The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

* On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then rese ing the 8U2.

1. Arduino uno

* On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

The Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

The Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

The board can operate on an external supply from 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows: Vin. The input voltage to the Uno board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.

3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

GND. Ground pins.

IOREF. This pin on the Uno board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

The ATmega328 has 32 KB (with 0.5 KB occupied by the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Each of the 14 digital pins on the Uno can be used as an input or output, using pin Mode(),digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions: Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.

External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.

PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.

SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.

LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.  
There are a couple of other pins on the board:AREF. Reference voltage for the analog inputs. Used with analogReference().

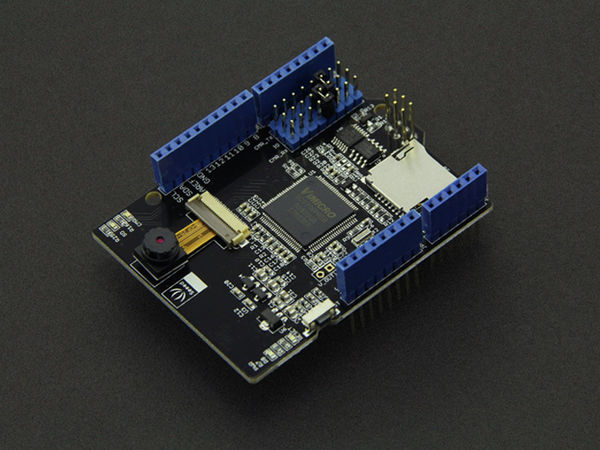
The Uno has a number of facilities for communicating with a computer, another Uno board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

## 7.5 CAM Shield

Camera Shield is the first shield board that support photograph, it is based on the solution of VC0706 + OV7725, the former is a high performance camera processor with enhanced image processing functions and it embedded the hard-wired JPEG codec, and the latter is a high-performance 1/4 inch, single-chip VGA camera and image processor in a small footprint package. We have written our library to control the Camera Shield via UART/SPI, which makes it possible to take picture easily by your Arduino board.



1. Seedunio Camera Shield

It is a first camera shield compatible with arduino which comes from Seeed, we use **VC0706 + OV7725** to achieve it. VC0706 is a high performance camera processor with enhanced image processing functions and it embedded the hard-wired JPEG codec.

## 7.6 Router

We have used TP-LINK TL-WR841N 300Mbps Wireless N Router in our design.

## C:\Users\green\Desktop\images.jpg

1. Router

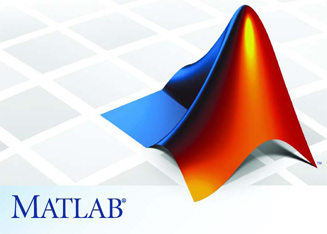
It’s300Mbps wireless speed is ideal for interruption sensitive applications like HD video streaming. Two antennas has greatly increase the wireless robustness and stability. It hasEasy wireless security encryption at a push of WPS button.IP based bandwidth control allows administrators to determine how much bandwidth is allotted to each PC.Compatible with IPv6 isthe more recent Internet Protocol version. TP-LINK Tether App allows quick installation and easy management using any mobile device.

## 7.7 MATLAB

**MATLAB** (**mat**rix **lab**oratory) is a multi-paradigm numerical computing environment and fourth-generation programming language. A proprietary programming language developed by MathWorks. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, Fortran and Python.

Although MATLAB is intended primarily for numerical computing, an optional toolbox uses the MuPad symbolic engine, allowing access to symbolic computing abilities. An additional package, Simulink, adds graphical multi-domain simulation and model-based design for dynamic and embedded systems.

Millions of engineers and scientists worldwide use MATLAB® to analyze and design the systems and products transforming our world. MATLAB is in automobile active safety systems, interplanetary spacecraft, health monitoring devices, smart power grids, and LTE cellular networks. It is used for machine learning, signal processing, image processing, computer vision, communications, computational finance, control design, robotics, and much more.



The MATLAB platform is optimized for solving engineering and scientific problems. The matrix-based MATLAB language is the world’s most natural way to express computational mathematics. Built-in graphics make it easy to visualize and gain insights from data. A vast library of prebuilt toolboxes lets you get started right away with algorithms essential to your domain. The desktop environment invites experimentation, exploration, and discovery. These MATLAB tools and capabilities are all rigorously tested and designed to work together.

## 7.8 Summary

This chapter includes details about the components that we have used in our setup and the details about the setup.

# Chapter 08: Operation of The System

## 8.1 Introduction

The system has to fetch the image data to the server for analysis. Server will call for the image data on time. Then the server will process the image data and count the number of vehicles. Then the server will assign optimal time and send signal to the LEDs in accordance with the optimal time.

## 8.2 Fetching Images Data

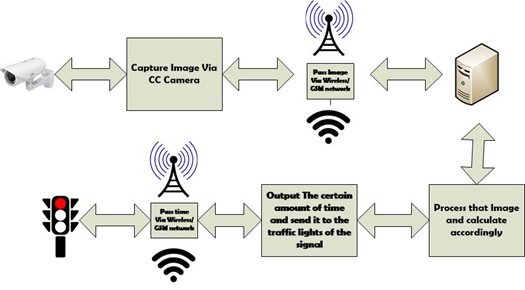
Each road connected to a traffic junction has its traffic controller(nodes). Each nodehas a camera to capture images and has access to wireless transmitter to broadcast images to the server. But to allow the server to commend the embaded device a predefined instruction is need. To acive this purpuse the host socket in node listens to requests and has a defined set of instruction as system commands. Node interprates /getImage request as capture image command. When this request is received Arduino controls the devices and capture images and sends that image back as a response to that request. Each node or traffic device has its own location information and MAC/IP adress through which the server can identify the particular node. Now when the server requests an image to any node like, "http://xxx.xxx.xxx.xxx/getImage" ,the node will respond to the server. Here we are using TCP/IP protocol to send the image. Here "xxx.xxx.xxx.xxx" the IP address of the node. The image is sent to the server via wireless connection from the wireless transmitter.

## 8.3 Receiving and Analysis of the Image on the Sever

This is the main part of our research work. We have explained briefly in the system design section how the image analysis process actually works. On the iterating process of calculating optimal time server will request a specific node to provide street image for the time calculation of that specific node. At the end of the request server will receive image of road and pass it to its computation section to process the image and compute optimal time. After computing optimal time server will provide light command to node.

## 8.4 Sending Signal to the Traffic Lights

Just like capture images to allow the server to light up a specific light on a specific node a set of instruction is required. Node software interprets /red, /grn, /yhl request as light command and up on receiving one of this request software will light up the respective signal light. Here /red request turns on red light, /grn turns on green light /yhl turns on yellow light.Initially server turns all the traffic lights to red. Now after getting the time period for a certain signal of a junction, the server sends signal to that particular node through TCP/IP connection. Server will command to turn the traffic light green from red like “http://xxx.xxx.xxx.xxx/grn” and server counts down the time period which has been calculated by the server. When thecountdown is done server commands the traffic light to turn red again.



1. Input Output signal

In this section, we have discussed about the operation of the system and in Fig. 9, we have shown communication the process for better understanding. The discussion was based on one junction but this process will be going on in every junction.

## 8.5 Summary

This chapter includes the discussion about how the server will send signal for different instructions to be done by different equipments. It also says about how the images will be served to the server for image processing, car counting and lighting the signals for proper amount of time.

# Chapter 09: Results and Discussion

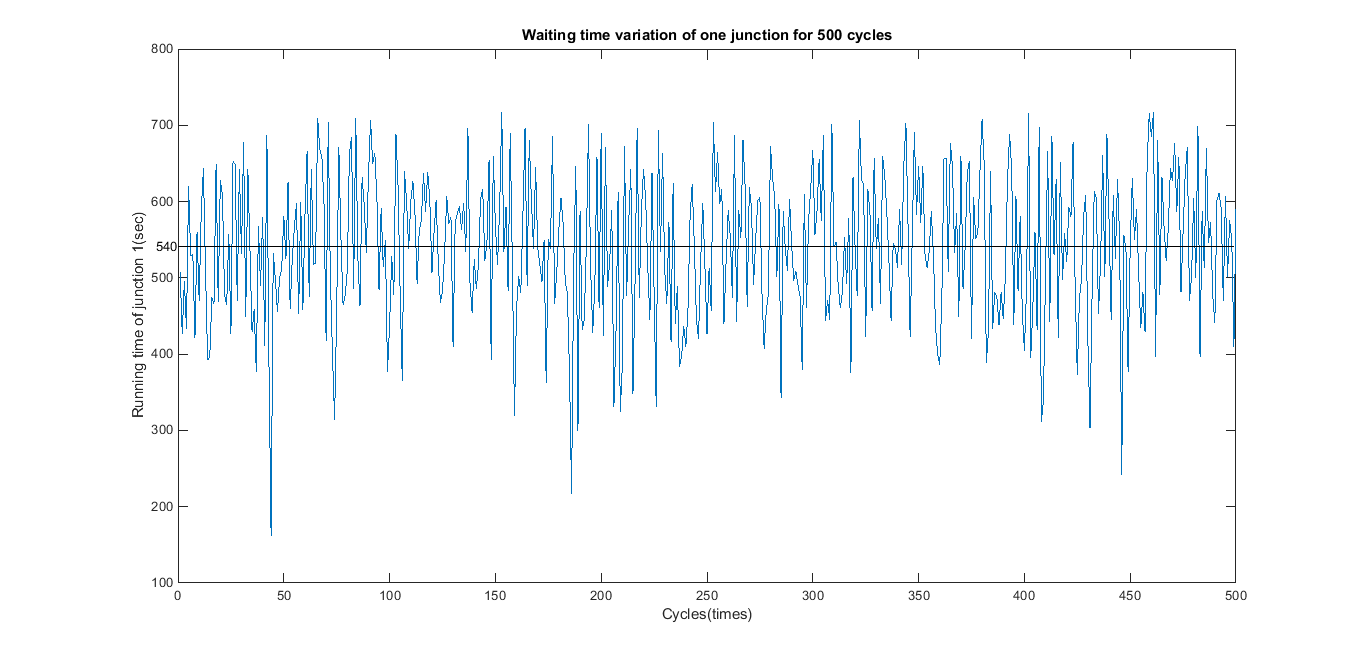
## 9.1 Introduction

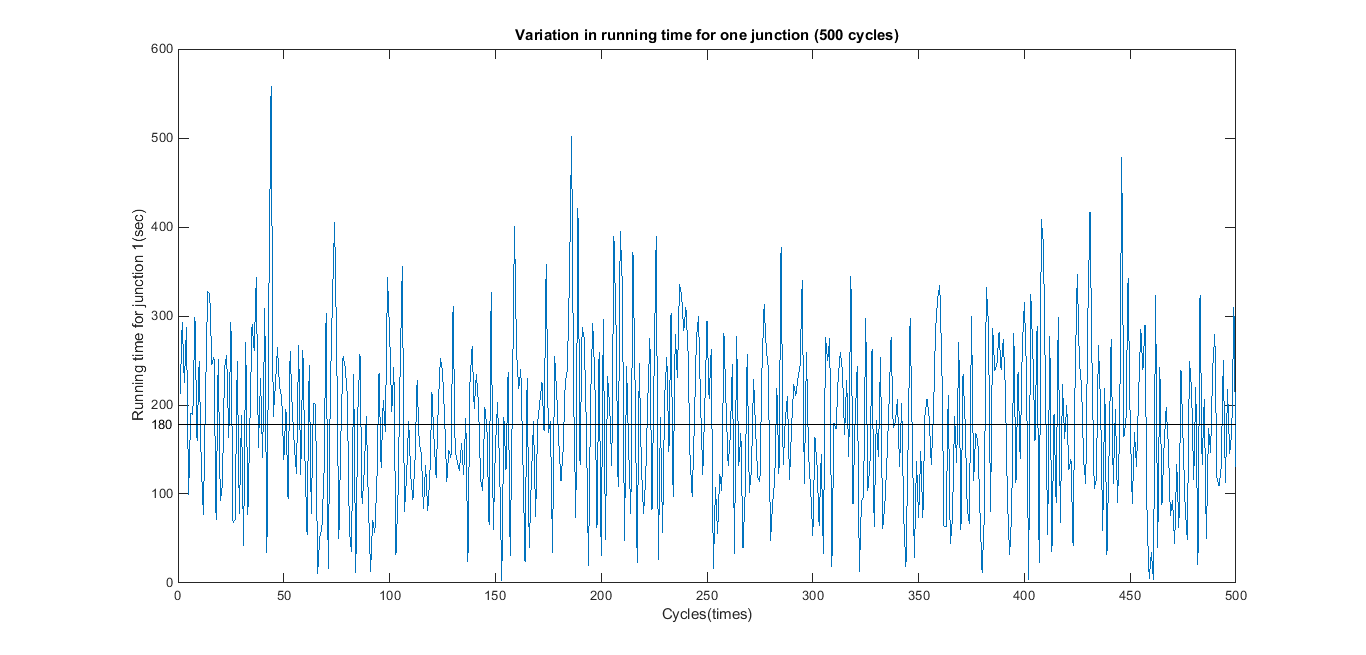
In this section we will discuss the outcome of our system. Results part are divided into two sections. The first one is the simulated results which has done in MATLAB. Another part is the result that we have got from our demo. Both of them are given below.

## 9.2 Simulation

This research is conducted in a replicated environment and in that environment significant evidence of time optimization was measured. Research shows us that for a fixed 180 seconds running time four road cross junction takes around 720 seconds to complete a cycle. The dynamic system also takes 720 seconds to complete a cycle by but waiting time of individual vehicle reduces according to the number of cars in each road of a junction.

Using randomly generated number of cars at the junctions a simple simulation has been run in MATLAB. Here in the Fig. 10, the variation of running time for one junction has been shown. The horizontal straight line represents the constant time 180s which would be the running time for a node of a junction if the running time was same for all. But if the technique that has been proposed here is used the time for running time won’t be the same each time. There would be variation in time. Same logic goes for the waiting time, Fig. 11. Waiting time won’t be the same all the time. But the black horizontal line represents the constant waiting time which is 540s for each cycle.

1. Visualization of running time



1. Visualization of waiting time

Table 1 represents the car numbers of the four roads of a junction which have been used as inputs and table 2 and table 3 represents the running time and waiting times of first 10 cycles.

1. Number of cars

|  |  |  |  |
| --- | --- | --- | --- |
| Road 1 | Road 2 | Road 3 | Road 4 |
| 43 | 96 | 73 | 59 |
| 55 | 71 | 1 | 79 |
| 93 | 1 | 83 | 77 |
| 100 | 23 | 92 | 65 |
| 11 | 27 | 77 | 81 |
| 11 | 47 | 22 | 93 |
| 33 | 86 | 26 | 88 |
| 19 | 76 | 4 | 65 |
| 57 | 38 | 22 | 80 |
| 15 | 49 | 2 | 19 |

1. Running times

|  |  |  |  |
| --- | --- | --- | --- |
| Running Time(Road1) | Running Time(Road2) | Running Time(Road3) | Running Time(Road4) |
| 114.24 | 255.06 | 193.95 | 156.75 |
| 192.23 | 248.16 | 3.50 | 276.12 |
| 263.62 | 2.83 | 235.28 | 218.27 |
| 257.14 | 59.14 | 236.57 | 167.14 |
| 40.41 | 99.18 | 282.86 | 297.55 |
| 45.78 | 195.61 | 91.56 | 387.05 |
| 101.97 | 265.75 | 80.34 | 271.93 |
| 83.41 | 333.66 | 17.56 | 285.37 |
| 208.32 | 138.88 | 80.41 | 292.39 |
| 127.06 | 415.06 | 16.94 | 160.94 |

1. WATING TIME

|  |  |  |  |
| --- | --- | --- | --- |
| **Waiting Time(road1)** | **Waiting Time(road2)** | **Waiting Time(road3)** | **Waiting Time(road4)** |
| 605.76 | 464.94 | 526.05 | 563.25 |
| 527.77 | 471.84 | 716.50 | 443.88 |
| 456.38 | 717.17 | 484.72 | 501.73 |
| 462.86 | 660.86 | 483.43 | 552.86 |
| 679.59 | 620.82 | 437.14 | 422.45 |
| 674.22 | 524.39 | 628.44 | 332.95 |
| 618.03 | 454.25 | 639.66 | 448.07 |
| 636.59 | 386.34 | 702.44 | 434.63 |
| 511.68 | 581.12 | 639.59 | 427.61 |
| 592.94 | 304.94 | 703.06 | 559.06 |

## 9.3 Results from Demo

In our demo we have ran our system on three roads. The analyzed pictures are given below.



1. Analyzed images in our demo

For this , we have got 12 cars in total(with noise) but the ratio is same as the noises are same in all of the pictures. We have given 60s for a cycle and the given time by the system is given below in the table:

Table IV: Outputs from the demo

|  |  |  |  |
| --- | --- | --- | --- |
| **Fields** | **Number of Cars** | **Running time** | **Waiting time** |
| Road1 | 4 | 20 | 40 |
| Road 2 | 5 | 25 | 35 |
| Road 3 | 3 | 15 | 45 |

## 9.4 Summary

In this section we have discussed the results and thus we have proved that our system is well functioned a gives output as expected.

# Chapter 10: Future Work

## 10.1 Introduction

In this chapter we will discuss about the future of our project and how our project can be extended further.

## 10.2 Future Work

Currently the system uses fixed or predefined thresholds depending on the road to measure number of vehicles. System will be enhanced with machine learning abilities so that system itself can identify those thresholds. Currently the system runs on a single server for an individual network. System can be developed such a way so that system can run on a grid of servers so that multiple networks can be supported under a single system. Currently the components on the road run on external power. Devices on the traffic nodes can be powered by solar cell to increase power efficiency.

## 10.3 Summary

In this section we have discussed about the future works of our project.

# Chapter 11: Costing

## 

## 11.1. Introduction:

In this section we will discuss about the costing of our system step by step. There are basically three set up for the nodes. Basically you can use any of the setup from the three set ups.

## 11.2. Arduino Uno, Ethernet Shield, Camera Shield

4 Arduino Uno = 845 \* 4 = 3380Tk

4 Ethernet shield = 872\*4 = 3488Tk

4 Camera Shield = 9892Tk

Wires & Leds = 30.80Tk

**In Total = 16790tk (approx)**

## 11.3. Arduino Yun, Arduino Uno, Web camera

4 webcam = 500\*4 = 2000tk

4 yun = 3476\*4 = 13904tk

4 Arduino Uno = 845 \* 4 = 3380Tk

Led and wires = 30.80tk

**Total Cost = 19314.8tk(approx)**

## 11.4. Raspberry Pi, Web camera

1 Raspberry Pi = 4554tk

4 Webcam = 2000tk

Wires and leds = 30.80tk

**Total Cost = 6584.8tk**

## 11.5. The Server

The costing of the server depends on how much powerful server you want and that actually depends on how much computation will be done on the server. The server can range from 1,00,000Tk to 4,00,00Tk.

## 11.6. Summary

In this section , the costing of our system has been discussed.

# Chapter 12: Compliance with Standards

## 12.1 Introduction

The designed system complied with all standards including IEEE, US, and European standards.

## 12.2 Compliance with IEEE standard

There are a few distinct guidelines put forward by IEEE Standards affiliation. The majority of them however are not material for our framework. For the course of our work, we did attempt to take after the IEEE 1233 standard and gave System Requirements Specification in expansive points of interest. We have done our system work wirelessly which complies withthe IEEE 802 rules by IEEE standards.

## 12.3 Compliance with US standard

The need for stronger compliance and regulations for electrical appliance is strongly needed in US. The most significant regulation in this context is the [Sarbanes–Oxley Act](https://en.wikipedia.org/wiki/Sarbanes%E2%80%93Oxley_Act) developed by two U.S. congressmen, Senator [Paul Sarbanes](https://en.wikipedia.org/wiki/Paul_Sarbanes) and Representative [Michael Oxley](https://en.wikipedia.org/wiki/Michael_Oxley) in 2002 which defined significantly tighter personal responsibility of corporate top management for the accuracy of reported financial statements.

The [Office of Foreign Assets Control (OFAC)](https://en.wikipedia.org/wiki/OFAC) is an agency of the United States Department of the Treasury under the auspices of the Under Secretary of the Treasury for Terrorism and Financial Intelligence. OFAC administers and enforces economic and trade sanctions based on U.S. foreign policy and national security goals against targeted foreign states, organizations, and individuals.

Compliance in the USA generally means compliancy with laws and regulations. These laws can have criminal or civil penalties or can be regulations. The definition of what constitutes an effective compliance plan has been elusive. Most authors, however, continue to cite the guidance provided by the [United States Sentencing Commission](https://en.wikipedia.org/wiki/United_States_Sentencing_Commission) in Chapter 8 of the Federal Sentencing Guidelines.

Therefore, this product has compliance with European standard.

## 12.4 Compliance with European standard

There is considerable regulation in the European Union, some of which is from EU legislation. Various areas are policed by different bodies, such as the FCA ([Financial Conduct Authority](https://en.wikipedia.org/wiki/Financial_Conduct_Authority)), [Environment Agency](https://en.wikipedia.org/wiki/Environment_Agency) and [Scottish Environment Protection Agency](https://en.wikipedia.org/wiki/Scottish_Environment_Protection_Agency), [Information Commissioner's Office](https://en.wikipedia.org/wiki/Information_Commissioner%27s_Office) and others.

Important compliance issues for all organizations large and small include the [Data Protection Act 1998](https://en.wikipedia.org/wiki/Data_Protection_Act_1998) and, for the public sector, [Freedom of Information Act 2000](https://en.wikipedia.org/wiki/Freedom_of_Information_Act_2000).

The UK Corporate Governance Code (formerly the Combined Code) is issued by the [Financial Reporting Council](https://en.wikipedia.org/wiki/Financial_Reporting_Council) (FRC) and sets out standards of good practice in relation to board leadership and effectiveness, remuneration, accountability and relations with shareholders. All companies with a Premium Listing of equity shares in the UK are required under the Listing Rules to report on how they have applied the Combined Code in their annual report and accounts (The Codes are therefore most similar to the US' Sarbanes-Oxley Act).

Therefore, this product has compliance with European standard.

## 12.5 Summary

In this chapter, we discuss about the different standards that the designed system meets.

# Chapter 13: Design Impact

## 13.1 Introduction

We have designed a system to reduce the waiting time in traffic to the minimum possible time but doing so we have created some environmental, economic, social impact. Moreover, our designs sustainability and manufacturability is also a very big issue. In this chapter these issues are discussed.

## 13.2 Environmental Impact

Every system we build has an environmental impact. Our system is also not an indifferent one. Our system has some environmental impact. As we are reducing the traffic jam to the least possible that’s why the emission of carbon will be reduced which will positively affect the environment.

## 13.3 Economic Impact

As traffic jam will be reduced and all the vehicles have to wait for the least possible time, the amount of fuel will be saved to a great extent. That’s why we will be able to save money and fuel which is good for our economy. Moreover, working people will not lose time just waiting in the traffic jam and they will be able to spend their time more on work which will also effect positively.

## 13.4 Social Impact

Our system will have also some social impact. Social impact means the impact that is for the well fare of the society. As we aill able to control traffic jam and will be able to reduce to its lowest possible time people will be saved from losing time in the jam. So they will be able to give their work and family more time and thus will lead a happy life.

## 13.5 Ethical Impact

Our system is free from ethical dilemma. It only saves people time by reducing the waiting time to the lowest possible time.

## 13.6 Manufacturability

Manufacturing of our system is easy and less complex. The parts of the system are easily available and it is easy to construct. When it will be manufactured commercially the cost will be reduced a lot.

## 13.7 Sustainability

Our system will use a private network to communicate between the server and the nodes so it will not be a vulnerable network. The protocol we have used the HTTP protocol which is also a very powerful protocol to break. Camera will be in well protected area. Thus our system will sustain for a long time with proper maintenance and it’s easy to maintain.

## 13.8 Summary

In this section all the design impacts related to our project has been discussed.

# APPENDIX A

# MATLAB Codes

## A.1 Code of Image Read

function result = read\_gray(filename)

temp = double(imread(filename));

if (size(temp, 3) == 1)

result = temp;

else

result = 0.3\*temp(:,:,1) + 0.59\*temp(:,:,2) + 0.11\*temp(:,:,3);

end

## A.2 Code of Capturing Image

function image = captureImage()

image = zeros(480,640,4);

parfor i = 1:4

image(:,:,i) = read\_gray(strcat(strcat('http://192.168.0.' ,int2str(i)),'/cap.jpg'));

%strcat(strcat('http://192.168.0.' ,int2str(i)),'/cap.jpg')

end

## A.3 Code of Counting Cars

function result = count\_cars(img,img1)

temp = img - img1 ;

temp = abs(temp);

temp = (temp>50);

[labels, number] = bwlabel(temp, 8);

bw = bwareaopen(labels,500);

bw = imfill(bw,'holes');

[B,L] = bwboundaries(bw,'noholes');

imshow(bw);

result = length(B);

## A.4 Code of Running The System

node\_car = zeros(1,4);

node\_img = captureImage();

node\_background = captureImage();

for i = 1:4

node\_car(1,i) = count\_cars(node\_background(:,:,i),node\_img(:,:,i));

end

total\_car = node\_car(1,1)+node\_car(1,2)+node\_car(1,3)+node\_car(1,4);

Total\_time = 60;

node\_ratio = zeros(1,4);

node\_time = zeros(1,4);

for i = 1:4

node\_ratio(1,i) = node\_car(1,i)/total\_car;

node\_time(1,i) = Total\_time \* node\_ratio(1,i);

end

for i = 1:4

urlread(strcat(strcat('http://192.168.0.' ,int2str(i)),'/grn'));

tic;

pause(node\_time(1,i));

toc;

urlread(strcat(strcat('http://192.168.0.' ,int2str(i)),'/red'));

end

# APPENDIX B

# Code of the Nodes

## B.1 Codes Related to Arduino

**This code is in c++.**

#include "SoftwareSerial.h"

#include <VC0706\_UART.h>

#include <SD.h>

#include <SPI.h>

#define SS\_SD 10

//use software serial

SoftwareSerial cameraconnection(2,3);//Rx, Tx

VC0706 cam = VC0706(&cameraconnection);

//use hardware serial

//VC0706 cam = VC0706(&Serial1);

void setup()

{

Serial.begin(9600);

Serial.println("VC0706 Camera Snapshot Test ...");

if (!SD.begin(SS\_SD)) {

Serial.println("SD Card init failed...");

return;

}

if(true == cameraInit()){

snapShot();

}else{

Serial.println("camera init error...");

}

}

void loop()

{

//nothing to do

}

bool cameraInit()

{

cam.begin(BaudRate\_19200);

char \*reply = cam.getVersion();

if (reply == 0) {

Serial.println("Failed to get version");

return false;

} else {

Serial.println("version:");

Serial.println("-----------------");

Serial.println(reply);

Serial.println("-----------------");

return true;

}

}

void snapShot()

{

Serial.println("Snap in 3 secs...");

delay(3000);

if (! cam.takePicture()){

Serial.println("Failed to snap!");

}else {

Serial.println("Picture taken!");

}

// Create an image with the name IMAGExx.JPG

char filename[13];

strcpy(filename, "IMAGE00.JPG");

for (int i = 0; i < 100; i++) {

filename[5] = '0' + i/10;

filename[6] = '0' + i%10;

// create if does not exist, do not open existing, write, sync after write

if (! SD.exists(filename)) {

break;

}

}

// Open the file for writing

File imgFile = SD.open(filename, FILE\_WRITE);

Serial.print("create ");

Serial.println(filename);

uint16\_t jpglen = cam.getFrameLength();

Serial.print("wait to fetch ");

Serial.print(jpglen, DEC);

Serial.println(" byte image ...");

int32\_t time = millis();

cam.getPicture(jpglen);

uint8\_t \*buffer;

while(jpglen != 0){

uint8\_t bytesToRead = min(32, jpglen);

buffer = cam.readPicture(bytesToRead);

imgFile.write(buffer, bytesToRead);

//Serial.print("Read "); Serial.print(bytesToRead, DEC); Serial.println(" bytes");

jpglen -= bytesToRead;

}

imgFile.close();

time = millis() - time;

Serial.println("Done!");

Serial.print("Took "); Serial.print(time); Serial.println(" ms");

cam.resumeVideo();

}

## B.2 Codes Related To Raspberry Pi

**This code is in python.**

import picamera

from time import sleep

import pygame

import random

WIDTH=1280

HEIGHT=1024

FONTSIZE=50

def quote():

options = ["Strange person",

"You should have smiled",

"you will break the camera",

"you need a hair cut",

"You should try SnapChat",

"Are you a model?",

"You've grown an inch",

"Have you shrunk?"

]

return random.choice(options)

# INIT CAMERA

camera = picamera.PiCamera()

camera.vflip = False

camera.hflip = False

camera.brightness = 60

# BUILD A SCREEN

pygame.init()

screen = pygame.display.set\_mode((WIDTH,HEIGHT))

black = pygame.Color(0, 0, 0)

textcol = pygame.Color(255, 255, 0)

screen.fill(black)

while True:

# TAKE A PHOTO

camera.start\_preview()

sleep(0.5)

camera.capture('image.gif', format='gif', resize=(WIDTH,HEIGHT))

screen.fill(black)

pygame.display.update()

camera.stop\_preview()

#READ IMAGE AND PUT ON SCREEN

img = pygame.image.load('image.gif')

screen.blit(img, (0, 0))

#OVERLAY CAPTIONS AS TEXT

text = quote()

font = pygame.font.Font('freesansbold.ttf', FONTSIZE)

font\_surf = font.render(text, True, textcol)

font\_rect = font\_surf.get\_rect()

font\_rect.left = 100

font\_rect.top = 100

screen.blit(font\_surf, font\_rect)

pygame.display.update()

# WAIT A BIT

sleep(3)

# CLOSE CLEANLY AND EXIT

pygame.quit()

# APPENDIX C

# Technical Manual

TABLE OF CONTENTS

C.1 System Overview with Functional Block Diagram 90

C.2 Theory of Operations 91

C.2.1 Sending Image of The Roads 91

C.2.2 Receiving and Analysis of the Image on the Sever 91

C.2.3 Sending Signal to the Traffic Lights 92

C.3 Installation 92

C.3.1 Installing The Nodes 92

C.3.2 Setting Up the server 93

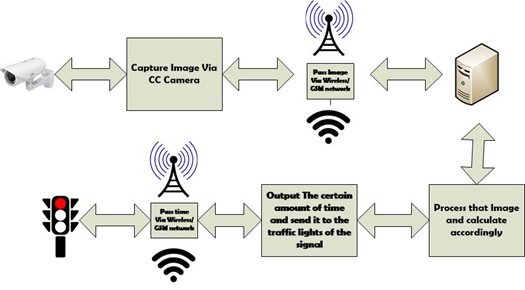
C.4 Schematics & System Architecture 94

C.5 Troubleshooting Guide 95

C.6 System Specifications 95

## C.1 System Overview with Functional Block Diagram

The system is divided into two sections. The nodes consisting of camera and signal lights and the server. The nodes mainly take picture of the roads and sends to the server. The server calculates the running time for each road by analyzing the images and again the server signals the nodes to light the traffic signals. Below a brief diagram has been given to show the system in brief.



1. Working diagram of the system

## C.2 Theory of Operations

### C.2.1 Sending Image of the Roads

Each road connected to a traffic junction has its traffic controller(nodes). Each nodehas a camera to capture images and has access to wireless transmitter to broadcast images to the server. But to allow the server to commend the embaded device a predefined instruction is need. To acive this purpuse the host socket in node listens to requests and has a defined set of instruction as system commands. Node interprates /getImage request as capture image command. When this request is received Arduino controls the devices and capture images and sends that image back as a response to that request. Each node or traffic device has its own location information and MAC/IP adress through which the server can identify the particular node. Now when the server requests an image to any node like, "http://xxx.xxx.xxx.xxx/getImage" ,the node will respond to the server. Here we are using TCP/IP protocol to send the image. Here "xxx.xxx.xxx.xxx" the IP address of the node. The image is sent to the server via wireless connection from the wireless transmitter.

### C.2.2 Receiving and Analysis of the Image on the Sever

This is the main part of our research work. We have explained briefly in the system design section how the image analysis process actually works. On the iterating process of calculating optimal time server will request a specific node to provide street image for the time calculation of that specific node. At the end of the request server will receive image of road and pass it to its computation section to process the image and compute optimal time. After computing optimal time server will provide light command to node.

### C.2.3 Sending Signal to the Traffic Lights

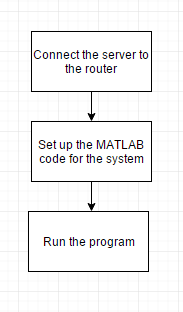
Just like capture images to allow the server to light up a specific light on a specific node a set of instruction is required. Node software interprets /red, /grn, /yhl request as light command and up on receiving one of this request software will light up the respective signal light. Here /red request turns on red light, /grn turns on green light /yhl turns on yellow light.Initially server turns all the traffic lights to red. Now after getting the time period for a certain signal of a junction, the server sends signal to that particular node through TCP/IP connection. Server will command to turn the traffic light green from red like “http://xxx.xxx.xxx.xxx/grn” and server counts down the time period which has been calculated by the server. When thecountdown is done server commands the traffic light to turn red again.

## C.3 Installation

### C.3.1 Installing The Nodes

### 

### C.3.2 Setting Up the server



1. Server Setup

## 

## C.4 Schematics & System Architecture

The whole system architecture has been given in the following image.



1. Schematics of the system

## C.5 Troubleshooting Guide

## 

1. Troubleshooting Guide

## C.6 System Specifications

To build the system the requirement of the system is –

1. Raspbrry Pi B+
2. Any CCTV camera for a good quality picture
3. A server that can run MATLAB

# REFERENCES

[1] Traffic Congestion. (n.d). Retrieved from https://en.wikipedia.org/wiki/Traffic\_congestion

[2] S. Khan, 'The Financialexpress-bd', Thefinancialexpress-bd.com, 2015. [Online]. Available: http://www.thefinancialexpress-bd.com/2014/07/17/45710/print. [Accessed: 22- Nov- 2015].

[3] 'INTELLIGENT TRAFFIC LIGHTSCONTROL BY FUZZY LOGIC', Malaysian Journal of Computer Science, vol. 9, no. 2, pp. 29-35, 1996.

[4] R. Cucchiara, M. Piccardi and P. Mello, 'Image Analysis and Rule-Based Reasoning for a Traffic Monitoring System', IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTAT ION SYSTEMS, vol. 1, no. 2, pp. 119-130, 2000.

# THANK YOU!