A TRAFFIC CONGESTION LEVEL MONITORING SYSTEM
IN CROWDED DEVELOPING WORLD CITIES

By

SSENYANGE JOSEPH
Reg. No: 2010/HD18/423U
BSc (CS) (Makerere University)
Department of Computer Science
College of Computing and Information Sciences, Makerere University
E-mail: jssenyange@gmail.com,
Tel: +256712600648

A Project Report Submitted to School of Graduate Studies in Partial Fulfillment of the
Requirements for the Award of the Degree of Master of Science in Computer Science of
Makerere University

OPTION: Computer Vision

Supervisors

Dr. John A. Quinn and Rose Nakibuule
Department of Computer Science
College of Computing and Information Sciences, Makerere University
jquinn@cit.ac.ug, rnakibuule@cit.ac.ug

July, 2014
DECLARATION

I Ssenyange Joseph do hereby declare that this Project Report is original and has not been published or submitted to this or any other University for any academic award.

Signed:......................... Date:.................................

Ssenyange Joseph
MSc Computer Science
Department of Computer Science
College of Computing and Information Sciences
APPROVAL

This Project Report has been submitted for examination with my approval as supervisor:

Signed:................................. Date:.................................

Dr. John Quinn
Department of Computer Science
College of Computing and Information Sciences
Makerere University-Uganda.
DEDICATION

This work is dedicated to my wife, classmates and lecturers who have been my constant source of inspiration. Their drive and support made possible the completion of this project.
ACKNOWLEDGEMENTS

I would like to extend my sincere gratitude towards all the people who have helped me complete this project. Their guidance, help and encouragement made it possible to make strides in this project.

I extend special thanks to my supervisors Dr. John Quinn and Rose Nakibuule whose valuable guidance and support contributed to my completion of this project.

I am grateful to College of Computing and Information Sciences for giving me this academic opportunity.

At last but not least gratitude goes to friends Frank Kilima, Keneth Bwire, Ochola Aloysius, and Innocent Kumurubuga who directly or indirectly helped me to complete this project.
Contents

DECLARATION i

APPROVAL ii

DEDICATION iii

ACKNOWLEDGEMENTS iv

LIST OF ACRONYMS ix

ABSTRACT x

1 Introduction 1

1.1 Background to the study .............................................. 1

1.2 Problem statement .................................................. 3

1.3 Objectives .......................................................... 3

1.3.1 General objective .................................................. 3

1.3.2 Specific objectives .................................................. 3

1.4 Scope of the study .................................................. 4

1.5 Significance of the Study ............................................ 4

2 Literature Review 5

2.1 Traffic congestion monitoring and tracking tools ................. 5

2.2 Traffic monitoring systems in developing countries ............. 7

2.3 Traffic monitoring systems in developed countries ............. 9

2.4 Related work ....................................................... 11
2.4.1 Image processing based research ........................................ 11
2.4.2 Acoustic array based research ........................................... 13
2.5 Research gap ...................................................................... 14

3 Methodology ........................................................................ 15

3.1 Introduction ...................................................................... 15
3.2 Tools and setup overview ..................................................... 15
  3.2.1 Monitoring Unit .............................................................. 15
  3.2.2 Mobile phone ............................................................... 16
  3.2.3 Application server ........................................................ 16
  3.2.4 Camcorder .................................................................. 16
3.3 Data Collection .................................................................. 16
  3.3.1 Mobile application ....................................................... 16
  3.3.2 Data collection using a camcorder .................................... 18
  3.3.3 Camera placement ........................................................ 18
3.4 Designing a congestion estimation algorithm ....................... 18
3.5 Evaluation of traffic congestion algorithm ............................ 18
3.6 Software development .......................................................... 19
  3.6.1 Web Application ............................................................ 19
  3.6.2 Data Storage ............................................................... 19
  3.6.3 Mobile Application ....................................................... 19

4 System Design ...................................................................... 20
4.1 Component overview ................................................. 20
4.2 Mobile client ......................................................... 22
  4.2.1 Image Capture Subsystem ................................. 22
  4.2.2 SMS command receiver application .................... 24
4.3 Mobile client web data API ....................................... 25
4.4 Message queuing system .......................................... 25
4.5 Congestion estimation client .................................... 26
  4.5.1 Background model construction .......................... 26
  4.5.2 Region of interest ........................................... 26
  4.5.3 Congestion estimation ....................................... 27
4.6 Data and information Store ...................................... 28
4.7 End-user web application ........................................ 30
  4.7.1 User front-end ............................................... 30
  4.7.2 Administrative section ...................................... 31
4.8 Phone reset client .................................................. 31

5 Results ........................................................................... 32
  5.1 Experiment Setup .................................................... 32
  5.2 Accuracy evaluation of the system in measuring traffic congestion levels. 36
5.3 User Interfaces ......................................................... 37
    5.3.1 Map Page ...................................................... 37

6 Conclusion and Recommendations .................................. 40
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Conclusion</td>
<td>40</td>
</tr>
<tr>
<td>6.2 Recommendations</td>
<td>40</td>
</tr>
<tr>
<td><strong>Bibliography</strong></td>
<td>41</td>
</tr>
<tr>
<td><strong>Appendix</strong></td>
<td>44</td>
</tr>
<tr>
<td>A System Installation Prerequisites</td>
<td>44</td>
</tr>
<tr>
<td>A.1 Server requirements</td>
<td>44</td>
</tr>
<tr>
<td>A.2 Android phone requirements</td>
<td>45</td>
</tr>
<tr>
<td>B System Installation</td>
<td>45</td>
</tr>
<tr>
<td>B.1 Server Configuration</td>
<td>45</td>
</tr>
<tr>
<td>B.1.1 Database setup</td>
<td>45</td>
</tr>
<tr>
<td>B.1.2 Gunicorn setup</td>
<td>45</td>
</tr>
<tr>
<td>B.1.3 Nginx setup</td>
<td>46</td>
</tr>
<tr>
<td>B.1.4 Website Configuration</td>
<td>48</td>
</tr>
<tr>
<td>B.1.5 Cronjob setup</td>
<td>48</td>
</tr>
<tr>
<td>B.1.6 RabbitMQ Task workers</td>
<td>48</td>
</tr>
<tr>
<td>B.1.6.1 SMS sender</td>
<td>48</td>
</tr>
<tr>
<td>B.1.6.2 Congestion estimation client</td>
<td>49</td>
</tr>
<tr>
<td>B.2 Android application setup</td>
<td>49</td>
</tr>
</tbody>
</table>
# LIST OF ACRONYMS

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>CAN</td>
<td>Controller Area Network</td>
</tr>
<tr>
<td>CCD</td>
<td>Change Coupled Device</td>
</tr>
<tr>
<td>CCTV</td>
<td>Closed Circuit Television</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>HMM</td>
<td>Hidden Markov Model</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>ITS</td>
<td>Intelligent Transportation Systems</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>RADAR</td>
<td>Radio Detection And Ranging</td>
</tr>
<tr>
<td>REST</td>
<td>Representational State Transfer</td>
</tr>
<tr>
<td>VIP</td>
<td>Video Image Processor</td>
</tr>
</tbody>
</table>
ABSTRACT

Many cities in developing countries like Kampala do not have modern intelligent traffic monitoring and management systems (ITS) like those deployed in developed countries because of the differences in financial and traffic conditions. Most of the cities in developed countries have good roads with clear lanes, relatively simple and orderly traffic flow patterns, homogeneous mix of vehicles, cars installed with special devices to work with ITS and are financially well to purchase and deploy such expensive systems.

Contrary, cities in developing countries like Kampala have poor roads which are bumpy, potholed and sometimes non-laned with disorderly and chaotic complex traffic conditions characterized by a lot of braking and sometimes honking. Most of these roads are narrow, poorly planned and constructed and are non-laned or with only two lanes. Traffic congestion on these roads is caused by a heterogeneous mix of objects ranging from large carriers, small vehicles, motor bikes, pedestrians, hawkers and sometimes animals.

With lack of real time information on traffic flows, traffic police officers fail to redirect traffic in order to avoid traffic jams, and travellers can not plan their journeys in advance in real time to avoid congested routes.

The project developed a low cost solution which is able to avail real time information on traffic flow to travellers via web interface that is accessible via portable devices like mobile phones with an internet connection. The main drawback of the monitoring solution is that it performs poorly in significant weather changes like heavy rain and night time.

To improve functionality of the system, future work should extend the system to make use of historical information to predict the traffic state based on day of the week and time period when there is no available fresh data from the traffic data input sources.
1 Introduction

1.1 Background to the study

Traffic congestion is the saturation of the road network capacity due to increased traffic volume or interruptions on the roads causing increase in travel time. While traffic congestion is being experienced in all biggest cities of the world, it is more severe and difficult to mitigate in cities of the developing world countries like Kampala. Despite efforts made by governments, city planners, researchers, network managers and private sectors, the problem continues to grow year after year [1], [4] and [23].

Researchers have outlined several factors contributing to traffic congestion and group them as recurrent and non-recurrent factors. According to [4], [10] and [23], recurrent factors act periodically on transportation systems and are always predictable. These are such as rapid increasing number of vehicles and residents in the cities, inadequate infrastructure, poor urban planning which is inappropriate for cars and inadequate traffic management resources such as traffic personnel and traffic lights especially at traffic junctions.

Several factors contribute to the inadequate growth of infrastructure in developing countries, some of which have been mentioned by [19] and [23] to be insufficient funds by governments, bureaucracy and lack of physical space for increased traffic volume. Slow expansion in road networks which does not commensurate with the rapid increase of population of both residents and vehicles in these cities is also another factor. In India, while the number of vehicles increased from 300,000 in 1951 to 30 million in 2004, which is a 100 times increase, the road network within the same period of time expanded from 0.4 million kilometers to 3.32 million kilometers, only an 8 times increase [1].

For non recurrent congestion triggered by unexpected and unplanned factors, it affects parts of transportation systems randomly and is difficult to be predicted and modeled in advance [4], [21]. It includes incidents such as weddings, political demonstrations and convoys of political and government leaders, traffic accidents, vehicle breakdowns, construction activities on or near the roads, unpredictable weather conditions and natural disasters, major sporting and entertainment events.
Traffic congestion has negative effects to commuters, environments and economy. According to [12], [13] and [20] traffic congestion causes unpredictable travel times to passengers and drivers going to or from work, meeting or other appointments. Secondly, it contributes to air pollution by adding significant amount of carbon emissions in the atmosphere. These emissions increase global warming, smog and increased respiratory problems in the communities.

Authors [8], [12], [20] and [23] state that traffic congestion excessively increases transportation cost and the costs of operating the vehicles particularly cost for fuel and maintenance of engine and other mechanical parts. The stop-start driving practiced by many drivers and additional time of driving caused by traffic congestion increase mechanical damage and wear and tear to engines, clutches, brakes and other parts of vehicles. It also causes early decrease of road surface lifetime due to heavy weight of vehicles traveling on them causing top surfaces to be deflected downwards [16] since there is no enough time for deflection correction to take place which necessitates the roads to be maintained earlier and irregularly than expected.

Although different measures have been proposed and implemented to mitigate traffic congestion, the success has been very little. Such measures are the use of traffic police, traffic lights (signals), volunteer traffic marshals, exclusive public transport lanes, reversible lanes, road pricing policy and employee parking cash out, use of regulations and traffic engineering such as parking restrictions, turn restrictions and loading/unloading restrictions. Also included are the use of speed limits where cars are not allowed to move less than certain speeds in some areas of the roads and deployment of Intelligent Transportation Systems (ITSs).

According to [13], [18], [19] and [20], although ITSs implemented in developed countries offer an effective solution in mitigating traffic congestion, they are inapplicable in developing countries like Uganda. This is because of their high cost, assumption of orderly traffic and poor road conditions in cities of developing countries. These ITSs assume constrained and clutter-free environments, presence of good and laned highways with cars moving uniformly and homogeneous mix of vehicles. This is not the case in developing countries which lack such well planned roadways, sufficient number of closed Circuit Television (CCTV) cameras and other sensors, and qualified personnel to process camera captured data.
With absence of efficient ITSs in developing countries results in lack of real time information on traffic flows to commuters and drivers, cities of developing countries need tailored and less expensive ITSs which are efficient in traffic conditions experienced in developing countries [8], [16], [18], [19] and [20].

1.2 Problem statement

Existing congestion monitoring technology is too expensive [17] for deployment in cities of developing countries and relies on inappropriate assumptions such as orderly traffic and good laned highways. With lack of real time information on traffic flow, traffic police officers fail to redirect traffic in order to avoid traffic jams. In addition to not prioritizing areas to dispatch traffic police officers, car drivers and passengers fail to plan their journeys in advance too.

1.3 Objectives

1.3.1 General objective

To develop a traffic monitoring system based on image processing and computer vision techniques that collects and processes traffic video frames in order to produce real time information on congestion levels to help prediction and monitoring of traffic congestion.

1.3.2 Specific objectives

The following are the specific objectives of the project:

(i) To collect data using a mobile application capturing road image sequences.

(ii) To formulate an algorithm for traffic congestion estimation using computer vision methods.

(iii) To design and implement a traffic monitoring system

(iv) To evaluate the accuracy of the system in measuring traffic congestion levels.
1.4 Scope of the study

The study is limited to the use of background subtraction techniques in determining traffic congestion level during day light clear weather conditions. The study will not be based on vehicle segmentation and object tracking.

1.5 Significance of the Study

(i) Real time information on traffic flow will help the police force to redirect traffic and prioritize areas to dispatch resources in order to manage traffic congestion.

(ii) The system’s real time information on traffic flows will help commuters to plan their journeys in advance by either choosing different routes or times to travel. This will reduce time and economy losses, air and noise pollution, road rage and transportation cost.

(iii) The project will be used by other researchers for reference.
2 Literature Review

2.1 Traffic congestion monitoring and tracking tools

There exist many technologies and tools used to control and monitor traffic congestion in urban areas in different cities in the world. The following are the most popular technologies with their merits and demerits [9], [14], [15], [17] and [22].

<table>
<thead>
<tr>
<th>Sensors</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inductive</td>
<td>Most commonly used sensors for ITS. Well established, time tested, understood and mature technology. Reliable when properly installed.</td>
<td>High installation and maintenance costs. Require lane closure for installation and maintenance. Sensitivity, reliability and usefulness depend on installation procedures. Provide limited coverage. Installed on good pavements which reduce pavement life. Require re-installation after road is repaired. Susceptible to damage by heavy vehicles, road repairs and utilities.</td>
</tr>
<tr>
<td>loop $500 - $800 [17]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ultrasonic</td>
<td>Installed over the roads. Installation and maintenance do not require lane closure. Widely used in Japan. Can collect data from multiple lanes.</td>
<td>Are usually affected by bad weather and temperature conditions.</td>
</tr>
<tr>
<td>$600 - $1,900 [17]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microwave</td>
<td>Mature technology widely used in Europe. Is not affected by inclement weather conditions. Can provides multiple lane data collection. Operate in day and night times. Installation and repair does not require lanes to be closed.</td>
<td>Can provide unwanted vehicle detection due to reception of sidelobe radiation. Can provide false vehicle detection due to multipath. Must ensure that the antennae beamwidth and transmitted waveform are suitable for applications. Cannot detect stopped or very slow moving vehicles unless installed with an auxiliary device. May have poor performance at intersection locations due to large number of vehicles.</td>
</tr>
<tr>
<td>RADAR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$700 - $3,300 [17]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Magnetic</td>
<td>Less susceptible to traffic stresses than inductive loops. Some of the models transmit data over wireless radio frequency link.</td>
<td>Their installations require pavement cut. Installation and maintenance require lane closure. Decrease the pavement life.</td>
</tr>
<tr>
<td>(Magneto meters)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$1,100 - $6,300 [17]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 2: Common used ITS Sensors

<table>
<thead>
<tr>
<th>Sensors</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magnetic</td>
<td>Less susceptible to traffic stresses than inductive loops. Useful where</td>
<td>High installations and maintenance costs. Cannot detect stopped or slow moving</td>
</tr>
<tr>
<td>(search coil</td>
<td>inductive loops are not feasible e.g under bridge decks. Installation of</td>
<td>vehicles. Have small detection zones. Installations of some models do require</td>
</tr>
<tr>
<td>Magnetometers)</td>
<td>some models do not require pavement cut.</td>
<td>pavement cuts.</td>
</tr>
<tr>
<td>$1,100 - $6,300 [17]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Acoustic</td>
<td>Installation do not require pavement cuts. Provide passive detection.</td>
<td>Affected by extreme cold weather conditions which degrade quality of data.</td>
</tr>
<tr>
<td>arrays</td>
<td>Provide day and night operations. Not affected by precipitation. Can</td>
<td>Have high purchase and installation costs. Some models do not work well with</td>
</tr>
<tr>
<td>$3,100 - $8,100 [17]</td>
<td>collect data from multiple lanes</td>
<td>slow moving vehicles in stop and go traffic.</td>
</tr>
<tr>
<td>Video</td>
<td>Better performance in clear conditions. Produce a variety of traffic</td>
<td>Can be affected by weather conditions, shadows and reflection from road</td>
</tr>
<tr>
<td>Image Processor</td>
<td>information than many other techniques. Installation and maintenance</td>
<td>surfaces. Their algorithms require big memory to process huge their data. Their</td>
</tr>
<tr>
<td>(VIP)</td>
<td>are easy, do not require lane closure as installed above roads. Can monitor</td>
<td>cameras and image processing devices are expensive. Have high energy</td>
</tr>
<tr>
<td>$5,000 - $26,000 [17]</td>
<td>multiple lanes and have a wide area of coverage. Operate in day and night</td>
<td>requirements and not appropriate in battery powered environment.</td>
</tr>
<tr>
<td></td>
<td>conditions.</td>
<td></td>
</tr>
</tbody>
</table>

Despite their disadvantages, VIPs outperform most of the traffic monitoring technologies, are very flexible and most of their disadvantages can be overcome by improving the design and installation of hardware and algorithms [13].

Figure 1: Traffic congestion detector in Germany (Image Source: Wikimedia)
2.2 Traffic monitoring systems in developing countries

Many cities in developing countries like Kampala do not have modern traffic monitoring and management systems (ITSs) like those deployed in developed countries because of the differences in financial and traffic conditions [16], [18], [19] and [20]. Most of the cities in developed countries have good roads with clear lanes, relatively simple and orderly traffic flow patterns, homogeneous mix of vehicles, cars installed with special devices to work with ITS and are financially well to purchase and deploy such expensive systems [13], [16], [18], [19], [20] and [22].

Contrary, cities in developing countries like Kampala have poor roads which are bumpy, potholed and sometimes non-laned with disorderly and chaotic complex traffic conditions characterized by a lot of braking and sometimes honking [16], [18], [19] and [20].

Most of these roads are narrow, poorly planned and constructed and are non-laned or with only two lanes. Traffic congestion on these roads is caused by a heterogeneous mix of objects ranging from large carriers, small vehicles, motor bikes, pedestrians, hawkers and sometimes animals. Worse enough, they have high rate of non-recurrent factors such as car break downs, traffic accidents, construction activities which proliferate traffic congestion. There is also little and sometimes no adherence to the right of way protocols and other road safety regulations.

Figure 2: A traffic congestion scene at Jinja Rd, Roundabout (Image source: monitor.co.ug)

Another big problem is the presence of heterogeneous mix of vehicles and other road users
which among others include buses, trucks, cars and vans, motorcycles, auto rickshaws, bicycles, carts, pedestrians, hawkers and sometimes animals. With such big differences in traffic conditions, it is difficult for the ITSs specially made for developed countries to work effectively in developing regions like Uganda. Also ITSs are too expensive for developing countries to purchase in large number and deploy them on cities’ roads and highways; and also require a lot of technical expertise which is not adequately available in developing countries [16], [18], [19] and [20].

It is stated by [19] that solutions for the generation and dissemination of information on flows and levels of traffic congestion for developing regions are urgently needed not because problems associated with traffic congestion are more severe in those regions but also because their patterns are fundamentally different from those in developed world. Mohan et al [18] states that because developing regions form the bulk of the world, solutions for information on flows and levels of traffic congestion are very important.

Traffic congestion in developing countries is largely managed by using traffic lights, traffic police officers (cops) and volunteer traffic marshals who control movements of vehicles mainly at traffic junctions. Other methods used are;

- **Grade separation**: Bridges are used to free vehicles from stopping for other crossing movements.

- **One way roads**: Using one way roads in some parts of the cities.

- **Separate road lanes**: Using separate road lanes for specific user groups such as pedestrians, buses and motorcycles.
• **Parking restrictions:** Discouraging the use of private vehicles by increasing cost for parking in the cities.

• **Park and ride facilities:** Restrict parking at a distance and then allow the use of public transport to reach to various destinations in the heart of the cities.

• **Road space rationing:** Prevent certain types of vehicles such as big trucks, tankers, auto rickshaws and motor bikes from driving at certain times of the day.

All these methods are ineffective in monitoring traffic congestion levels for various reasons. First, traffic lights and police officers are very few and inadequately distributed in cities causing most of the traffic junctions and other usually congested places to be unmanaged. Secondly, they do not generate and disseminate any real time information on congestion flows and congestion levels. This information helps commuters and drivers to plan their journeys in advance by selecting other options like changing time of traveling or take other optimal routes to avoid congestions.

Most of the cities in developing countries do not have ITSs not only because of the differences in traffic conditions with those in developed countries, but also they require expensive hardware such as sensors, CCTV cameras, servers and well trained personnel to operate them [13], [16], [18], [19], [20], [22] and [24].

### 2.3 Traffic monitoring systems in developed countries

Monitoring and management of traffic congestion in developed countries are done using a number of methods apart from those mentioned in Section 2.2.

• **Online shopping:** People buy goods and services from sellers over the Internet. Since online shopping is very vibrant in this part of the world, it greatly reduces the number of vehicles on the roads from people who were to drive in order to buy the goods or services. This in turn reduces traffic congestion in many cities.

• **Telecommuting:** A form of working in which employees do commute to their central work places and instead perform their work duties from home or other locations by
using mobile telecommunications technologies. This also reduces traffic congestion as many employees do not have to drive to get to their work places and back home.

- **Exclusive public transport lanes:** Lanes of certain roads are exclusively used by public transport.

- **Employee parking cash out programs:** Companies pay their employees for not driving their private cars. This encourages employees to use public transport.

- **Reversible lanes:** Certain sections of the highways operate in the opposite directions at different times of the day.

- **Road pricing:** Charging drivers some money for access onto certain roads.

- **The use of Intelligent Transportation Systems (ITSs):** Computer based applications which use different techniques to provide several traffic management services including real time information on traffic congestion flows and levels to various road users and authorities.

Modern ITSs which use a host of techniques with a number of sensors and detectors such as Doppler radar, inductive loop detectors, magnetic detectors, CCTV cameras, acoustic sensors, video and image sensors and bridge sensors are widely deployed in developed countries like United States, Japan and Europe [13], [18], [19] and [22]. These systems provide several traffic management services including generating and disseminating real time information on traffic congestion flows and levels and alleviate congestion [8], [16], [18] and [19].

![Figure 4: A traffic congestion scene in Berkeley, California (U.S.)](Image Source: Wikimedia)
ITSs are very efficient, expensive and best suite the traffic conditions of most of the cities in developed countries which have very good roads with clear lanes and homogeneous mix of vehicles [8], [13], [16], [18], [19] and [22]. Most of vehicles in these cities are installed with special devices such as GPS and Controller Area Network (CAN) bus which work with most of the ITSs for the collection and delivery of traffic information.

![Traffic police officer in Italy](Image Source: Wikimedia)

Figure 5: Traffic police officer in Italy (Image Source: Wikimedia)

According to [8], [13], [16], [18], [19] and [22], ITSs in developed countries are highly effective because they are installed on wide, well planned and constructed roadways with multiple dedicated lanes. Finally, most of these cities are well planned to provide quick and controlled movements of vehicles [23]. These roads have a homogeneous mix of vehicles which are the only biggest cause of traffic congestion. To enhance their performance most of these systems have the ability to track individual vehicles and some of them track special features such as vehicle number plate.

## 2.4 Related work

Several research works have been conducted on automated traffic monitoring and management systems for both developed and developing countries. These research works are based on two categories which are estimation of road geometry and tracking of objects (vehicles).

### 2.4.1 Image processing based research

In [22], a VIP based traffic monitoring system is proposed. The system assumes that individual vehicles can be tracked and therefore requiring segmentation of vehicles in each image frame. Such an approach has serious disadvantages as it may cause shadow detection
and requires sophisticated algorithms to remove occlusion. It also assumes that vehicles are the only moving objects being monitored and are constrained within specific lanes and roads.

Research on traffic congestion caused by different objects such as vehicles, pedestrians and cyclists was conducted by [2] and [3]. Like others, the research was based on segmentation and tracking of individual objects and does not provide classification of the types of objects being tracked.

Authors [6] and [5] researched on traffic congestion without segmentation of vehicle and object tracking using hidden markov model (HMM). Though the work provides classification of objects, it assumes that road conditions are not cluttered, which is not the case with the roads in the developing countries.

Another work on traffic congestion estimation is detailed in [16]. This work aims at using a VIP based urban traffic monitoring technique to determine the real-time traffic congestion estimation without vehicle classification. The proposed solution aims at determining two parameters in order to estimate the traffic congestion. These parameters are occupation rate and movement status of the vehicles. Though the solution is proposed to have high performance on a wide range of traffic conditions like light traffic, congestion and varying speeds, in a wide variety of lighting conditions like sunny, overcast, twilight, night and rainy, and operate in real-time, it has two major weaknesses. Firstly, it relies on expensive surveillance CCD cameras, and secondly it assumes that vehicles are the only moving objects being monitored and are constrained within specific lanes and roads.

Fei Zhu [7] proposes a video based traffic monitoring system that uses adaptive background subtraction method to predict the occurrence of traffic congestion. According to [16], video solutions based on background subtraction method are not effective in heavy traffic congestion and abrupt change of light as they fail to produce satisfying background model.

Glasl et al [11] proposes a traffic monitoring system based on embedded computer vision system to predict the occurrence of traffic congestion. Despite its strengths, the solution has serious weaknesses. It relies on expensive CCTV cameras and attains fairly low accuracy of about 74% of traffic prediction caused by factors such as resource constraint (memory and processing power) of the embedded computer system, occlusion and bounding box inaccu-
racies generated by the detection algorithms. Also the solution is based on object detection and tracking, and assumes that road conditions are not cluttered.

Another work [16] proposes a video based system that uses charge coupled devices cameras (CCDs) installed on poles or buildings close to city roads and highways. The system which is based on background subtraction, frame differencing and time spatial image detects the movement of vehicles and then uses this information to produce real time traffic congestion estimation for various road users. This system has several weaknesses. First, it has no concerns about traffic congestion conditions of cities of developing countries. The system is well built for city roads and highways of the developed countries. Secondly, it lacks a mechanism for disseminating information on traffic congestion flows and levels to road users.

2.4.2 Acoustic array based research

R. Sen et al [19] propose a low cost traffic monitoring and management system for cities of developing countries. The system comprises of a pair of road side acoustic sensors which captures vehicles’ honks as an input data. The captured data (vehicle’s honk) is then sent to a centralized server for processing in order to detect vehicles’ speed and classify the traffic state as either congested or free flowing. The analyzed information is then routed to road users over mobile phones.

Although the system is inexpensive and was meant for city roads and highways of developing countries, it has some shortcomings. First, it makes an assumption that honking is a drivers’ common practice in all developing countries as it is in India, which in not true. In many cities of developing countries like Uganda, drivers only honk in certain political, sports or chaotic events such as ”walk to work”, when the national football team, The Uganda Cranes plays in Kampala (or has played and worn outside Uganda) or under very chaotic traffic congestion. There is usually very little or no honking when there is calm or free flowing of traffic. Secondly, the system’s sensors will only capture vehicle’s honks when the vehicles are close enough to the sensors. Thirdly, the system will only be able to help road users who are at that particular moment driving on the roads equipped with sensors and can honk. It cannot help those who are at home or any other places with no sensors but would want
to get the state of the traffic congestion on certain roads in the city they want to drive on. Lastly, the system is not able to sense noise from opposite direction lanes of wide roads.

Another work [18] proposes an inexpensive and energy efficient ITS called Nericell tailored for cities in developing countries which uses various sensors namely accelerometer, GPS, microphone and GSM radio available on smart phones. These sensors are used to detect potholes, bumps and vehicles’ braking or honking and send the data to a centralized server for aggregation. Processed data (information) may then be sent to various road users to help them search for roads with little or no traffic congestion. Though the system is inexpensive and uses mobile phones which most of the drivers have, its major weakness is that the system’s data sources (bumps, potholes and honking) are not evenly distributed on city roads and highways. Some sections have many potholes and bumps than others and the honking is not common practice by drivers in all driving situations. This has a potential of generating inaccurate information for drivers.

2.5 Research gap

With the evidence gathered from literature review, it is clear that traffic ITSs deployed in cities of developed countries cannot have equally high performance and effectiveness when deployed in cities of developing countries [16], [18], [19] and [20]. This is due to differences in traffic conditions, urban texture, high price and technical expertise of the systems as explained in Sections 2.2 and 2.3. Because of these limitations, many cities in developing countries have not deployed such systems and instead they are still using old methods of monitoring and managing traffic. These methods do neither provide real time information on congestion flows and levels nor predict the occurrence of traffic congestion to help road users to prevent or avoid traffic congestion.

This project attempted to implement a cost effective traffic monitoring system based on image processing and computer vision techniques which is suitable for traffic conditions of developing countries. The system uses low cost android phones and makes use of background subtraction method to generate and deliver real time information about congestion levels to drivers, passengers and traffic police officers.
3 Methodology

3.1 Introduction

This chapter states the methods and tools that were used to achieve specified objectives at the various phases of this project. The project used implementation driven research methodology where components were built, tested, fixed and then integrated with an incremental model.

3.2 Tools and setup overview

The following tools were used to achieve objectives of the project.

3.2.1 Monitoring Unit

The monitoring unit comprised of a solar panel, a battery, phone charger and mobile phone. The solar panel charges the battery from which a phone charger is connected to the mobile phone. Charging the battery first allows for continual supply of power in case of cloud cover. The monitoring unit was mounted at the main gate of Makerere University.

Figure 6: Traffic Congestion Estimation: Monitoring unit (Image courtesy of AI-DEV group Makerere University)
3.2.2 Mobile phone

A Huawei IDEOS mobile phone running Android operating system was locked inside the monitoring unit with the camera positioned at a hole. The mobile phone had an application that was uploading image data captured at intervals to a remote application server.

3.2.3 Application server

Image data captured by the mobile application was uploaded to a remote server hosted on Amazon EC2 cloud application server. This server was setup to host the REST API servers, database, messaging system, image processing programs and the Front-end user view website.

3.2.4 Camcorder

A camcorder was used as an alternative means of collecting image data before and after the monitoring unit was ready to be installed at Bwaise - Northern Bypass roundabout.

3.3 Data Collection

Collection of image data was done via an android mobile application and a camcorder. The data collection phase using the mobile application was necessary in testing the stability of the components of system. Both data from the phone application and Camcorder was used during the tuning of the algorithm to estimate traffic congestion.

3.3.1 Mobile application

An Android mobile application was developed that captures image sequences at defined interval and uploads them to a remote server for traffic analysis. The application is installed on a mobile phone running an Android operating system and has a camera.

The application was installed on a mobile phone and then setup in a specially designed housing unit (see figure 6) by Makerere University Artificial Intelligence group. The unit was mounted at the Makerere main entrance gate. This allowed the opportunity to test and
fix the stability and performance of the application while still collecting data. The application was able to run trouble free collecting data from 8th October 2012 to 4th November 2012, a total of 28 days.

The phone failed on November 4th and no more data was collected after it rained and water clogged inside the unit leading to failure of the phone.

The data collected in the period the phone was operating amounted to more than 10gb. This data was very useful in the early phase of tuning of the traffic congestion algorithm. The data also gave more insight on why having to choose a good mount point for the camera is necessary for optimal traffic congestion estimation.

Considering the location of the mounting of the camera was for a test drive of the system, the data collected had objects in the lower lane of the road where occluded by ones in the upper lane of the road (see figure 7). This made it impossible to test features that required having good visibility of both lanes of the roads.

![Figure 7: Traffic Congestion Estimation: Sample images captured](image)

To avoid occlusion, the unit would have had to be moved to a different location, but the bureaucracy involved in this endeavor would take long and impede progress on the project. Continual data collection during development required using an alternative method, the Camcorder.
3.3.2 Data collection using a camcorder

Using a camcorder mounted on a tripod stand for stability, a steady video stream was captured for a period of time at Bwaise on the Northern Bypass. The video was later broken into frames/images using the FFmpeg tool. A program was written that would simulate a phone capturing and uploading images. This solution allowed continued development and improvement of the traffic congestion system when no data was streaming in from the android application.

3.3.3 Camera placement

The choice of camera placement has to be almost parallel to the moving traffic at a height where there is a good field of view at an angle of about 45°. The preferred location of placing the camera is not at junctions but at service roads that feed traffic to road junctions.

3.4 Designing a congestion estimation algorithm

The congestion estimation algorithm was based on background subtraction technique which is a process of extracting foreground objects in an input image. The background subtraction process involved constructing a model of the background and deciding which pixels don’t belong to the background class. To derive a background model, the median of each pixel intensity value is calculated in the set of images from the data collection.

When the mode of the background is subtracted from an image, the remainder are objects which have been introduced in the background. The proportion of these objects in the road they are covering is expressed as a congestion estimate.

3.5 Evaluation of traffic congestion algorithm

To evaluate the performance of the algorithm, three sets of images each with twenty images classified in the same set category was used. The sets were categorised as low, medium and high. For each image in its classified traffic congestion group, the system calculated
congestion level was calculated and saved. A proportion of test cases correctly classified was counted that gave an accuracy measure of performance of the algorithm.

3.6 Software development

The actual development of the traffic congestion monitoring system was carried out as explained below.

3.6.1 Web Application

The web site and data services where developed using Django web framework that allows for rapid development. The programming language used was python because of easy integration with OpenCV image processing software.

3.6.2 Data Storage

Information was stored in MySQL database whereas image files uploaded and intermediate processed images were stored on filesystem.

3.6.3 Mobile Application

The mobile application was developed using Java programming language with the Android software development kit provided by Google.
4 System Design

4.1 Component overview

The five major components of the system are the monitoring unit, web application, message broker, congestion estimation service, and data storage.

![Component diagram]

The monitoring unit component is responsible for collection of data which is served to the web application. The web application forwards metadata about received data to the message broker component which is responsible for distributing work to the next available congestion estimation service obligated to extract traffic parameters and persisting it to data storage. The web application can then display traffic status information from data storage.

The monitoring unit component comprises of a mobile phone client for data collection as a sub-component. The mobile phone contains the image capture and the phone reset process. The image capture process is responsible for capturing images and uploading it to the web application via a data service api. The phone reset process is responsible for receiving commands from the web application via SMS.

The web application component comprises of a mobile client web data API REST service and an end-user web site for administrative configuration and visual display of congestion on a map. The data service feeds metadata about the received data from mobile phones.
to the message broker component for further distributed processing. The message broker is implemented with RabbitMQ as the message queuing system used to decouple CPU intensive image processing congestion estimation component from the web application.

The congestion estimation service component is assigned work to process by the message broker. It comprises of a congestion estimation client for deriving of congestion estimation from uploaded image data from the mobile phone. The derived estimate is saved to Mysql data storage for later lookup from the website.

The data storage component comprises of Mysql and filesystem storage. Mysql is a data and information store for persistent storage of settings and congestion estimations whereas the filesystem is for storage of uploaded and intermediate files.

The cross functional flowchart below describes the flow of data between subsystems of the different components.

![Traffic Congestion Estimation: Data Collection And Flow](image)

**Figure 9: Traffic Congestion Estimation: Data Collection and Flow**
The webserver is setup with Nginx reverse proxy server to serve the map page, administration backend, and REST API services. For minimal setup all the three are run under the same backend application but with different network configurations. For real deployment that needs scalability, all the three would have to run on different hosts but masked by Nginx proxy server. The choice of Nginx as proxy server is due to its high concurrency, high performance and low memory usage compared to its big open source competitor apache.

4.2 Mobile client

The android mobile client is a java mobile application developed for collecting sequences of images. The application is further divided into two subsystems, the image capture and SMS command receiver application.

![Image Capture Subsystem](image)

Figure 10: Mobile client actors

4.2.1 Image Capture Subsystem

This is responsible for the capture of image sequences and uploading them to a remote server. This application is started automatically on phone boot or manually via the menu. It comprises of the following services that run in parallel:
1. **Settings refresh service:**

This service fetches settings via the settings web data JSON REST API on a configured interval. The settings which are managed remotely control the internal behavior of the application. The settings comprise of:

- **Service URL** Base url location where remote service endpoints are hosted
- **Image upload interval** Interval at which image sequences are uploaded to the server
- **Settings refresh interval** Interval at which new settings are fetched remotely
- **Picture Size** Dimension of images captured and uploaded
- **JPEG Quality** Quality of images captured.
- **Zoom level** Controls the camera zoom level
- **Frames to capture per upload** Size of the set of images to capture and upload
- **Frames Capture Interval** Interval between capturing images part of an upload set.

2. **Location refresh service:**

This service is used to get the GPS coordinate of the phone. Since the unit is housed in an enclosed unit, GPS locking tends to take time and mostly it fails to lock on satellites. The fallback is to enter the GPS coordinate manually for the phone via the administrative interface.

3. **Image capture service:**

This service captures image sets at a given interval, saves them to the phone file system and queues a request via Android messaging Intent system to the Upload service to send them to the server. The upload request consists of metadata not actual files with the following information:

- **Frames Capture Interval** Interval between capturing frames that was used.
- **List of files** A list of images captured with each image with the following information
  - **File Name** The path to the location where the image was saved
  - **Capture Date** The time image was captured
  - **Sequence Number** The sequence at which the image was captured
Queuing of the upload request enables the capture process to continue without being blocked by the network upload of the files. To address bandwidth concerns each phone can be configured separately to upload a cropped image which is a bounding box for zones which are the region of interest. A thumbnail that is to be displayed to the user is also uploaded. This reduces bandwidth requirements to $1/10^{th}$.

4. Upload Service:

This service is responsible for network upload of files and cropping of images if the phone is configured to upload only the bounding box of the region of interest. When the upload service receives an upload request from the image capture service, it performs a network upload of files to the server including the following parameters:

- **Phone IMEI** International Mobile Station Equipment Identity for device identification
- **Location** Current GPS location of the device if available
- **upload_roi** Signifies cropping of images was done
- **cropZoneX** X point location where cropping was started
- **cropZoneY** Y point location where cropping was started
- **cropZoneWidth** Width of the cropped zone
- **cropZoneHeight** Height of the cropped zone

If the network upload fails or is successful, the image files are deleted to free up space.

4.2.2 SMS command receiver application

When test running the image capture application with an Ideos Huawei phone, it was realized the data network stops after a few days of usage. The only way to fix it was to reboot. To avoid having to take down the unit every time uploads are not being received, the SMS command receiver application was developed that receives commands via the SMS channel to reboot the phone when no updates are received at the server.
4.3 Mobile client web data API

The settings and image upload API’s are the REST endpoints the android client uses to communicate with the server. Information sent to the server is via the HTTP POST method whereas information received from the server is encoded in JSON format. The settings endpoint is used by the phone to register to the server and also get the latest settings. The image upload API saves the files and queues the request via RabbitMQ for further processing to avoid executing resource intensive computer vision algorithms in a short HTTP request window. The API services are developed in the python language.

4.4 Message queuing system

RabbitMQ is used as the message broker between the image upload API and the congestion estimation clients. A persistent work queue ImageUpload created in RabbitMQ is used by the image upload API to publish new image sets uploaded. The messages are dispatched to subscribed applications (e.g. congestion estimate clients) to the work queue using a Round-robin algorithm. The reason for using task queues is to avoid doing complex computer vision resource-intensive task immediately in a short HTTP request window.

![Message queuing system diagram](image)

Figure 11: Message queuing system
4.5 Congestion estimation client

This is a task worker that subscribes to RabbitMQ to process any new uploads. When a new upload request is submitted to RabbitMQ, it’s routed to the next available congestion estimation program. On receiving and decoding the message, the application calculates the congestion estimate in each image for each road region of interest and saves the results in the MYSQL database. The final congestion estimation value displayed to the user is the average from the image set or the last image that was captured in the set depending on the configuration made in the administrative interface.

It uses OpenCV as the basis library for image processing algorithms with background subtraction technique employed in deriving a congestion metric from road image sequences. Background subtraction is a technique of extracting foreground objects in an input image. The background subtraction process involves constructing a model of the background and deciding which pixels don’t belong to the background class. These pixels combine to form foreground objects. The percentage of area covered by foreground objects in a region of interest which is the road in this context, gives an estimate of congestion.

4.5.1 Background model construction

Using some of the images captured at the location of interest, a gray scale background model of the scene is constructed by getting the median of points across all images. A median value of the intensity values in all images at each point is calculated. Figure 12 shows a background model constructed out of 272 images. The background model can only be constructed from images which are mostly having low congestion. Once constructed, the background model is saved and reused in subsequent operations to estimate congestion.

4.5.2 Region of interest

The region of interest is manually input by the user during configuration. Each region of interest maps to a road lane.
4.5.3 Congestion estimation

Given $B$ as background model (see figure 13(A) below), $S$ as image of scene (see figure 13(B) below) in which we need to estimate congestion, $(x, y)$ as position in image, and $R$ as region of interest, the following steps are used to get a congestion indicator:

(i) Convert $B$ and $S$ to grayscale as $B_g$ and $S_g$ respectively.

(ii) Extract the region of interest in $B_g$ and $S_g$ using a mask $M$ representing region of interest $R$ into $R_b$ and $R_s$ (see figure 13(D) below) respectively

$R_b = B_g \land M$

$R_s = S_g \land M$ where $\land$ denotes a bitwise and.

(iii) Find absolute difference between the background and scene as $A = |R_s - R_b|$ (see figure 13(E) below)

(iv) Apply binary thresholding on $A$ to get $T$ (see figure 13(F) below)
\[
T(x, y) = \begin{cases} 
  \text{maxVal} & \text{if } A(x, y) > \text{threshold} \\
  0 & \text{otherwise}
\end{cases}
\]

where threshold by default is 30.

(v) To remove holes and small objects in \( T \), a morphological closing is performed on \( T \) using a 7x7 convolution kernel \( K \), where \( K \) is filled with values of 1, giving an image matrix \( I \) (see figure 13(G) below).

\[
I = T \cdot K = (T \oplus K) \ominus K
\]

(vi) In the result binary image \( I \), we calculate the proportion \( C \) of white pixels in relation to the total number of pixels in the region of interest \( R \). \( C \) is the percentage of area covered by the foreground objects which is an indicator of congestion.

![Figure 13: Traffic Congestion Estimation: Algorithm Execution](image)

### 4.6 Data and information Store

MYSQL database is used to store the traffic congestion estimation values and settings for the server applications and phone whereas the filesystem is used for storage of images up-
loaded and their resized versions. The images are stored in folder paths with format phone id/Year/Month/Day.

The data schema in the MYSQL data is shown in figure 14 below. The tables in the schema have the following storage functions.

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>settings</td>
<td>Keeps application settings in key-value format</td>
</tr>
<tr>
<td>phonesharedsecret</td>
<td>Keeps shared keys used to generate hashes for validating requests</td>
</tr>
<tr>
<td>phone</td>
<td>Keeps phone meta data</td>
</tr>
<tr>
<td>phonestatus</td>
<td>Keeps data regarding last communication status between services and phone</td>
</tr>
<tr>
<td>phoneroadgeom</td>
<td>Keeps data describing each region of interest in phone’s field of view</td>
</tr>
<tr>
<td>trafficservicelevel</td>
<td>Keeps traffic service range categorization</td>
</tr>
<tr>
<td>locationtrafficserivelvel</td>
<td>Keeps estimation congestion levels for each region of interest</td>
</tr>
<tr>
<td>ltslresource</td>
<td>Keeps artifacts derived from processing of estimating traffic congestion</td>
</tr>
</tbody>
</table>
4.7 End-user web application

Django is the web framework used in development of the website using python programming language. The website has the user front-end and the administrative section. Access to the administrative section requires basic authentication.

4.7.1 User front-end

The front-end is a single page with a Map and two menus, Bookmarks and Watchlist. Bookmarks lists all the locations from which we are fed traffic congestion information. Watchlist is selective list of locations a user wants to monitor updates for on the map. The watch list
is stored in browser cookies. At a configured time, an AJAX call is made to the server and updated information is returned. This information is used to update the visual display of the traffic at source points on the map. On clicking a location on the map, detailed view is displayed. The user front-end page is viewable both on phone and desktop.

4.7.2 Administrative section

This section is for managing configurations which include the RabbitMQ connectivity options, android client settings, server side system settings, phone location and region of interest options. Access is restricted to this section using Basic Authentication built into the web hosting container.

4.8 Phone reset client

This is scheduled task that runs repeatedly at the set interval to send reboot command SMS to phones that have not uploaded any images in the last 15 minutes and a reboot command has not been sent to them in the last 45 minutes. This script is scheduled via the Linux crontab or windows task scheduler. The SMS messages are sent via Smsjaja.com API.
5 Results

5.1 Experiment Setup

The system was initially mounted at the Makerere University main gate on 28th September 2012 and at 01:14 PM it had started data collection. After a day on 29th September 2012 at 05:21 pm, the system stopped uploading images. The system had been initially configured to upload three images every two minutes and had managed to upload 2,469 images totaling to 305.52 Megabytes with a loss of about 2.37% of the data due to upload failures. Taking down the unit, it was discovered the phone data connection had stopped working. To avoid costly time of taking down units, an additional phone application was developed that receives SMS commands to reboot the phone. The sms messages are sent by the remote application if it does not receive any uploads within the last 15 minutes and no reboot command sent in last 45 minutes.

The second time the unit was re-installed after an update to the application, the data upload started from 8th October 2012 at 05:07 PM to 4th November 2012 07:54 AM. Data collection stopped after it rained and water clogged inside unit leading to failure of the phone.

<table>
<thead>
<tr>
<th>Start</th>
<th>End</th>
<th>Count</th>
<th>Bandwidth (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>28/9/2012 13:14</td>
<td>29/9/2012 17:21</td>
<td>2,469</td>
<td>305.52</td>
</tr>
</tbody>
</table>

Expected Count: 2,530
Time loss (%): (2,530 - 2,469)/3 = 20 * 2 = (40 / 1687)* 100 = 2.37%

Figure 15: Data collection statistics 28/9/2012 13:14 - 29/9/2012 17:21

<table>
<thead>
<tr>
<th>Start</th>
<th>End</th>
<th>Count</th>
<th>Bandwidth (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8/10/2012 17:07</td>
<td>4/11/2012 7:54</td>
<td>89,610</td>
<td>10.02</td>
</tr>
</tbody>
</table>

Expected Count: Not Available
Time loss (%): Not Available

Figure 16: Data collection statistics 8/10/2012 17:07 - 4/11/2012 7:54

In that period, the phone configuration settings were altered to upload between five and ten images every one to five minutes. The data collected totaled to 10.02 gigabytes with 89,610 images. Since the number of images uploaded and the upload interval where both altered, it
is not possible to give an accurate estimate of the expected data, but a logarithmic upload delays graph in Figure 17 is used to internalize the upload performance. The Y axis is the delay in minutes between uploads and the X axis is time.

![Graph showing estimates of upload delays](image)

The delay was calculated as the time in minutes between uploads rounded off to nearest integer. As expected most of the delay tends to two minutes since most of the time the frames capture interval was 120 seconds. Given their was no efficient real time monitoring and debugging of the application, we can only speculate the cause of the other upload delays to be one of the following:

(i) Some of the delay times can be attributed to the network stack of Android failing to establish a 3G data connection after long use until a full reboot of the phone is initiated every 15 minutes of upload delay and 45 minutes of last reboot command. This would apply to delays mainly within 10 to 70 minutes. These times will have to be lowered to eliminate the delays.

(ii) The MTN internet connection is sometimes slow and keeps disconnecting and thus breaking the upload sessions. Any upload that is not successfull is not retried and
files are deleted to save space. The big number of the frames to upload can also lead to delays for example on the 10th October 2012, the number of images where increased from 5 to 10 coupled with MTN internet intermittent failures, this increased the number of upload failures since upload sessions would break.

(iii) The MTN internet connection is down

(iv) The internet data bundle has been used up. During the second testing phase, data was loaded twice but the date this was made isn’t recorded so it can’t be linked to any of the delays in the graph above with 100% certainty. From the graph, the most evident delay is one which started on sunday 2012-10-28 13:27 until monday 2012-10-29 15:56. This is mostly due to data bundle running out and it had to be preloaded again.

To reduce on the bandwidth requirements, the mobile application was modified to upload only cropped images with only the region of interest. Given a region of interest $R$, $R = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$

The cropped region $C_R$ is

$C_R(min) = (Min(R_x), Min(R_y))$

$C_R(max) = (Max(R_x), Max(R_y))$

This reduced the bandwidth requirements by $1/10^{th}$. From the last data collection above, this would reduce to 1.002 gigabytes.

The choice of placement of the unit was for testing purpose and learning more of the challenges that will be faced when deployed in the live environment. The placement of the camera did not give a good view of the two lanes, leading to occlusion of the lower lane by the upper lane objects. In figure 18 below the RAMCO GAS vehicle in the upper lane obstructs part of a pickup and they may be other motorists that we can’t tell. In addition to objects in the road, a good view of the road is blocked by a tree, and sign post which lead to loss of data in region that they are obstructing.

The test unit mount location was good for testing the resilience of the system, but the location where the unit was mounted made the images captured not a good source for building a congestion estimate algorithm. To fine tune the congestion algorithm, addition data was collected using a camcorder in Bwaise at the Northern Bypass junction. This data
had good view of the lanes and less clutter from trees and sign posts as seen in figure 19. A simulation program was built that would upload the frames to allow building of the system to continue.
5.2 Accuracy evaluation of the system in measuring traffic congestion levels.

A steady video of the road was taken for set period of time and frames were extracted per second using ffmpeg command below.

```bash
ffmpeg.exe -i /path/to/video/file.mov -r 1 -sameq /path/to/output/image7-%9d.jpg
```

For each category HIGH, MEDIUM and LOW, twenty frames are assigned based on their traffic congestion level.

For each frame in its classified traffic congestion group, the system calculated congestion level is calculated and classified based on the table below.

<table>
<thead>
<tr>
<th>Classification</th>
<th>Minimum level (%)</th>
<th>Maximum level (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOW</td>
<td>0.00</td>
<td>50.09</td>
</tr>
<tr>
<td>MEDIUM</td>
<td>50.10</td>
<td>80.09</td>
</tr>
<tr>
<td>HIGH</td>
<td>80.10</td>
<td>100.00</td>
</tr>
</tbody>
</table>

The following are the results presented using confusion matrix.

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LOW</td>
</tr>
<tr>
<td>LOW</td>
<td>19</td>
</tr>
<tr>
<td>MEDIUM</td>
<td>0</td>
</tr>
<tr>
<td>HIGH</td>
<td>0</td>
</tr>
</tbody>
</table>

The wrongly classified traffic both in LOW and HIGH are attributed to the difficulty in choosing the ranges that correctly classify a percentage of congestion. Take for example figure 20 shows the image that was put in the LOW category but was classified as MEDIUM
based on the table above with the minimum and maximum congestion level. The image’s congestion level is estimated to be 63.11% but to the human, this can be considered as low traffic.

Figure 20: Low Congestion: Invalid classification

Given the wrong classifications, the system is able to correctly identify traffic with worst-case accuracy at \((18/20) \times 100 = 90\%\). To achieve this accuracy it is important when calibrating the region of interest it should only include the zone where vehicles pass and must exclude passenger lanes and outer and inner most shoulders of a lane.

5.3 User Interfaces

5.3.1 Map Page

Users can view the traffic congestion status on the map. The page is divided into three sections as shown in figure 21 below. The upper section has the menu, the middle section the map and the lower section the legend.

The menu has Bookmarks which lists all active locations with traffic updates and Watchlist
lists only those a user has chosen to monitor.

The map displays markers of locations where traffic information is collected. The markers are displayed in four colors.

<table>
<thead>
<tr>
<th>Color</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Black</td>
<td>No traffic updates in the last 15 minutes</td>
</tr>
<tr>
<td>Red</td>
<td>High traffic</td>
</tr>
<tr>
<td>Yellow</td>
<td>Medium traffic</td>
</tr>
<tr>
<td>Green</td>
<td>Low traffic</td>
</tr>
</tbody>
</table>

For a location where more than one lane is monitored, the marker is split into two with each half displaying its state color. Clicking on a marker will display more information about
that location like in figure 22. For each lane or region monitored, the time the last update
was made will be displayed, the current traffic level, a spark line graph showing the last
15 updates in the last hour and the average congestion in the last 30 minutes. The time
frames can be changed by the administrator. The last thumbnail taken at that location is
also displayed in the detailed view.

Figure 22: Location Detailed View
6 Conclusion and Recommendations

6.1 Conclusion

The main objective of the project was to develop a traffic monitoring system that uses image processing and computer vision techniques for crowded developing world cities. This goal was attained by developing an application acting as a sensor that runs on an android phone that captures images and uploads them to a remote server for further processing to extract traffic state parameters which are later visualized on Google maps.

In collaboration with Makerere University Artificial Intelligence Group, a prototype of a solar powered unit where a test run phone was installed was mounted at the gate. The application was able to run trouble free collecting data from 8th October 2012 to 4th November 2012, a total of 28 days after which the phone failed after heavy rains that clogged the unit. The prototype unit is in process of improvement to allow good saturation of air, lower temperatures and protection of the phone from water for optimal operation.

To expand on the functionality of the system, another module showing vehicle speeds is being developed by Rose Nakibuule. To validate the speeds derived from the uploaded image sets, another module was developed for GPS collection of data from GPS sensors on android phones. This GPS information is used to correlated the speed estimated from the speed module to that captured from GPS at the same time.

6.2 Recommendations

The system needs further improvement to work not only during day light but also during significant weather changes like heavy rain and night time to provide continuous supply of traffic information. When the system is deployed in a live environment and enough data is collected, the system may be extended to make use of historical information to predict the traffic state based on day of the week and time period when there is no available fresh data from the sensors (android phone). The system also needs bandwidth or data bundle monitoring to be re-stocked when its running low.
References


[17] Luz Elena Yañez Mimbela, Lawrence A Klein, and Vehicle Detector Clearinghouse, *Summary of vehicle detection and surveillance technologies used in intelligent transportation systems*, Vehicle Detector Clearinghouse, Southwest Technology Development Institute, New Mexico State University, 2003.


Appendix

A System Installation Prerequisites

Prior to installing the congestion system a number of prerequisites have to be satisfied on the server side and phone.

A.1 Server requirements

The preferred choice of operating system is Ubuntu Linux though the same can be done on windows. The minimum requirements for the server is 512MB RAM, 3GB Physical storage and 1GHz CPU. The following table lists software packages that have to be installed on the server.

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>System software</td>
<td></td>
</tr>
<tr>
<td>Python</td>
<td>Install python version &gt; 2.7 and &lt; 3.0</td>
</tr>
<tr>
<td>Erlang</td>
<td>Install Erlang runtime that is required by RabbitMQ</td>
</tr>
<tr>
<td>RabbitMQ</td>
<td>Download and install RabbitMQ from <a href="http://www.rabbitmq.com/install-debian.html">http://www.rabbitmq.com/install-debian.html</a></td>
</tr>
<tr>
<td>Django</td>
<td>Install Django web framework</td>
</tr>
<tr>
<td>Gunicorn</td>
<td>Python WSGI HTTP Server</td>
</tr>
<tr>
<td>Nginx</td>
<td>HTTP server and reverse proxy</td>
</tr>
<tr>
<td>Python packages</td>
<td></td>
</tr>
<tr>
<td>OpenCV</td>
<td>Install the python opencv package</td>
</tr>
<tr>
<td>Sqlalchemy</td>
<td>ORM framework</td>
</tr>
<tr>
<td>Geoalchemy</td>
<td>Extensions to SQLAlchemy to work with GPS points</td>
</tr>
<tr>
<td>MySQLdb</td>
<td>Python support for MYSQL database</td>
</tr>
<tr>
<td>Pika</td>
<td>Python implementation of the AMQP to connect to RabbitMQ</td>
</tr>
<tr>
<td>Jsonpickle</td>
<td>JSON serialization support</td>
</tr>
</tbody>
</table>
A.2 Android phone requirements

Applications running on the phone require minimum Android OS version 2.2 with support for SDK version 8 and above. The phone has to be rooted and superuser application and term.apk installed. The Android rooting is to allow privileged access to the Android subsystem to be able to reboot the phone from the application. You can unlock/root the phone using SuperOneClick program.

B System Installation

B.1 Server Configuration

Transfer the project folder to the server to location $ProjectFolder

B.1.1 Database setup

a) Login to MYSQL and create database with name traffic.

b) Use the mysql command to run the database script traffic-setup.sql in folder $ProjectFolder\database.

B.1.2 Gunicorn setup

Create a new executable script gunicorn_traffic_app.sh with the contents as below. Replace the $ProjectFolder and LOGFILE paths to valid locations

```bash
#!/bin/bash
set -e
LOGFILE=/var/log/gunicorn/django_app.log
LOGDIR=$(dirname $LOGFILE)
NUM_WORKERS=3 #recommended formula here is 1 + 2 * NUM_CORES

#we don’t want to run this as root..
USER=www-data
GROUP=www-data

cd $ProjectFolder
source bin/activate
```
export PYTHONPATH=$PYTHONPATH:$ProjectFolder/libs
cd webclient
test -d $LOGDIR || mkdir -p $LOGDIR
exec gunicorn_django -t 300 -w $NUM_WORKERS \
   --log-level=debug \ 
   --log-file=$LOGFILE 2>>$LOGFILE \ 
   --user=$USER --group=$GROUP

Configure the service to start on boot by creating a configuration file traffic_app.conf in /etc/init/ folder with contents as below. Ensure the paths match the locations on the system.

description "Django Traffic Application"
start on runlevel [2345]
stop on runlevel [06]
respawn
respawn limit 10 5
exec /mnt/images/traffic_app/run.sh

The web application will start automatically on boot or you can start it manually using the command service traffic_app start

B.1.3 Nginx setup

Create an htpasswd file in /etc/nginx.

Use the htpasswd command to add users to the file.

Create a site configuration file. Use the configuration below as template and for help see the nginx documentation at http://wiki.nginx.org/Configuration. Modify the configuration sample below to suit your setup.

server {
   listen 80;
   server_name tc.kutokea.com;

   access_log /opt/django/logs/nginx/vc_access.log;
   error_log /opt/django/logs/nginx/vc_error.log;

   # no security problem here, since / is alway passed to upstream
   root /opt/django/;
location /admin/static_files/ {
    alias /mnt/images/traffic_app/webclient/static/;
    autoindex on;
}

location /admin/ {
    auth_basic "Restricted";
    auth_basic_user_file htpasswd;
    proxy_pass_header Server;
    proxy_set_header Host $http_host;
    proxy_redirect off;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Scheme $scheme;
    proxy_connect_timeout 300;
    proxy_read_timeout 300;
    proxy_pass http://localhost:8000/admin/;
}

location /static/ {
    # if asset versioning is used
    if ($query_string) {
        expires max;
    }
    autoindex on;
    autoindex_localtime on;
}

location / {
    proxy_pass_header Server;
    proxy_set_header Host $http_host;
    proxy_redirect off;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Scheme $scheme;
    proxy_connect_timeout 300;
    proxy_read_timeout 300;
    proxy_pass http://localhost:8000;
}

# what to serve if upstream is not available or crashes
error_page 500 502 503 504 /media/50x.html;
}

B.1.4 Website Configuration

Access the website administrative section via http://[Hostname | IP address][:Port]/admin.
Enter the username and password configured under Nginx setup.
Under RabbitMq Settings enter the host, username and password to RabbitMQ. Under Common Settings and Global phone settings, complete all the fields.
NB: The Service Url under Global phone settings should be http://[Hostname | IP address][:Port]/client/m/v1.

B.1.5 Cronjob setup

Create a cron_tasks.sh with content as below. Replace the paths and variable $ProjectFolder accordingly

cd $ProjectFolder
export PYTHONPATH=$PYTHONPATH:$ProjectFolder/libs
/usr/bin/python $ProjectFolder/libs/common/CronJobs.py phone_status &>> /var/log/cron_task.log

Schedule the script above in crontab with entry as such below to run every five minutes.

*/5 * * * * root /path/to/cron_tasks.sh

B.1.6 RabbitMQ Task workers

B.1.6.1 SMS sender

It gets outgoing messages from the SMS Queue and sends them out via SMS jaja API.
Create a run_sms_tasks.sh with contents as below. Modify accordingly the paths.

cd $ProjectFolder
export PYTHONPATH=$PYTHONPATH:$ProjectFolder/libs
exec /usr/bin/python $ProjectFolder/libs/common/SmsService.py &>> /var/log/sms_task.log

Configure the task to start on boot by creating a configuration file sms_tasks.conf in /etc/init/ folder with contents as below. Ensure the paths match the locations on the system.
description "sms sending task"
start on runlevel [2345]
stop on runlevel [06]
respawn
respawn limit 10 5
exec $ProjectFolder/run_sms_tasks.sh

The task will start automatically on boot or you can start it manually using the command
service sms_tasks start

B.1.6.2 Congestion estimation client

It calculates the congestion estimation from uploaded images. Create a run_image_tasks.sh
with contents as below. Modify accordingly the paths.

cd $ProjectFolder
export PYTHONPATH=$PYTHONPATH:$ProjectFolder/libs
exec /usr/bin/python $ProjectFolder/libs/common/Tasks.py &>>
/var/log/congestion_task.log

Configure the task to start on boot by creating a configuration file image_tasks.conf in
/etc/init/ folder with contents as below. Ensure the paths match the locations on the
system.

description "Image processing task"
start on runlevel [2345]
stop on runlevel [06]
respawn
respawn limit 10 5
exec $ProjectFolder/run_image_tasks.sh

The task will start automatically on boot or you can start it manually using the command
service image_tasks start

B.2 Android application setup

Before installing the application, you have to enable installation of applications from Un-
known sources by checking Settings => Applications => Unknown sources, mute the
phone, and turn on mobile data and add the network Access Point Name (APN).
a) Transfer **PhotoCapture.apk** and **PhotoCaptureBridge.apk** to the phone using the data cable or bluetooth.

b) Go to location where you saved **PhotoCaptureBridge.apk**, open the apk to begin installation.

   i) Go to phone menu and open application **PhotoCaptureBridge**
   ii) Press the menu button and choose reboot
   iii) Go on to allow privilages
   iv) Phone should reboot. On start, repeat step (i) and (ii). The phone should not ask for permissions on the second reboot. If it does continue to step (iii) until no prompt for privileges is given.

c) Go to location where you saved **PhotoCapture.apk**, open the apk to begin installation.

   i) Go to phone menu and open application **PhotoCapture**
   ii) Press the menu button and choose reboot
   iii) Go on to allow privilages
   iv) Phone should reboot. On start, repeat step (i) and (ii). The phone should not ask for permissions on the second reboot. If it does continue to step (iii) until no prompt for privileges is given.

   v) Open the **PhotoCapture** => Go to preferences => Set the Base Service Url e.g. http://tc.kutokea.com/client/m/v1 and set network choice to mobile network or Wifi

   vi) Go to fetch settings => Enter identifier and shared secret. Choose refresh settings. The identifier and shared secret should be unique for every phone and can be configured from Shared Secrets section via the Administrative section of the site.