International Journal of Engineering &Technology, 7 (3.12) (2018) 926 -932

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Efficient Traffic Management System

[1]Suraj Kumar G Shukla, [2]Aadithya Kandeth, [3]D.Sai Santhiya, [4]Kayalvizhi Jayavel

SRM Institute of Science & Technology, Kattankulathur, Chennai
**Corresponding Author Email: [1]surajkumar_gh@srmuniv.edu.in, [2]aadithyakandeth_sa@srmuniv.edu.in*
[3]saisanthiya.d@ktr.srmuniv.ac.in, [4]kayalvizhi.j@ktr.srmuniv.ac.in

Abstract

Traffic Management is a big issue which impacts us almost daily. Use of technology such as IoT and image processing can lead to a smooth traffic management system. The common reason for traffic congestion is due to the lack of an efficient traffic prioritization system. The internet of things is a network of devices. The embedded systems includes sensors, actuators, and electronics.With software and connectivity locally or over internet helps in transfer of data. Each of these devices is uniquely identifiable in the network and are highly interoperable. Image processing using OpenCV is a technique that is used to process an input image and obtain the traffic densities along various lanes in a junction. Existing traffic management solutions include using RFID tags on vehicles to obtain a vehicle count. This can also be done using ultrasonic sensors. The problem with these methods is that when implemented in a large scale, the cost of the entire system can be exponentially higher than an image processing approach as each vehicle will have to be fitted with an RFID tag. Hence, implementing this model at a largescale level, for example in a metropolitan city will be time consuming and expensive. This has led to the development of our algorithm, which uses image processing and IoT along with CCTV cameras. This system is efficient, as it uses CCTV cameras that are already present in traffic signals of most major metropolitan cities. Hence implementing the system at a largescale level is feasible. This algorithm also takes care of corner cases like heavy utility vehicles and motorbikes. It can also be used at night and during unfavorable weather conditions.The algorithm used detects the density of traffic as opposed to the count of vehicles by taking input images from a CCTV camera, comparing it with a sample image of an empty road and obtaining a match percentage. The traffic density can be found easily using this since it has a disproportionate relation with the match percentage. The traffic signals can be altered accordingly using the traffic density. The output is then sent to the ThingSpeak cloud where it can be analyzed

1. Introduction

The smart traffic management system aims at the automation of traditional traffic management system. The traditional traffic systems are working on fixed algorithms and requires human support. The smart traffic management system aims at regulating the signal in such a way that maximum traffic could be cleared off in a lesser amount of time.

To develop an efficient technique that can be used for traffic control using image processing and IOT. In such a system, the traffic density of lanes is calculated using image processing which is performed on images of lanes that are obtained from a digital camera source such as a surveillance camera. According to the traffic densities on all roads, our proposed model will allocate smartly the time period of green light for each road by assigning priorities based on the traffic density. Image processing can be used for the measurement of traffic density levels because an implementation through cameras are cheaper than large scale sensor based systems.

The algorithm being used will also work in corner cases including heavy vehicles like trucks and small vehicles like bikes since it matches purely based on traffic densities and not on the count of vehicles on the road.

A surveillance camera is used to implement the algorithm and give priority.

The system also has the capability to store data on a ThingSpeak cloud server which will help with data storage and further data analysis.

This system replaces existing traffic management solutions such as:

- .RFID tags on vehicles to count the number of vehicles and assign signals accordingly. Recognition of vehicles using RF IDs by installing the RF IDs on all the vehicles. RF IDs are very accurate and also require very less amount of energy for working. RF IDs are also easy to install but looking at the level of implementation using RF IDs is expensive as well as time taking process.The vehicular congestion is the biggest issue faced by the traffic authorities and it has been increasing day by day. The number of vehicles present on any road in a city is directly proportional to the population growth and hence both have been increasing at a great rate.
- Using Ultrasonic sensors to get a vehicle count.Ultrasonic sensors detect any object in the path.Each time an object is detected, the count of vehicles incremented.This method needs installation of sensors along the width of every road which proves to be costly when setting up in a large country like India.

These existing implementations are expensive whereas our proposed system makes use of existing CCTV cameras which are already set up in most metropolitan cities for surveillance.

2. Literature Survey

After thorough reading and researching various approaches to solve problems are :

- An Image Processing Based Approach for RealTime Road Traffic Applications Sensors at their current state are not enough to provide necessary traffic information to engineers and their setup

can create a disturbance in the ecosystem. Hence morphological edge detection is one of the best possible approaches to analyze traffic. This system then uses a background neural network to judge traffic parameters. Neural networks are faster and more accurate as compared to other systems when it comes to analyzing traffic.

- An Automated Vehicle Counting System for Traffic Surveillance This system uses a software comprising of two components: An embedded DSP software and a host PC software. The embedded DSP software is used to acquire a video image from a camera, counts the number of cars and transmits the results and real-time images to the PC software. The PC software works as a GUI through which the end user can access the image processing results. The algorithm used is designed to maintain efficient counting of vehicles over inconsistent illumination levels.
- Evaluation of how well the Edge Detection Techniques for Images in Spatial Domain Edges performs is of extreme importance for multiple image processing applications. Hence the edge detection technique used should be able to filter out noise and unnecessary data. This paper provides a comparison of various edge detection techniques. It has been concluded that the Canny edge detection algorithm performs better than other techniques in this regard.
- Rain can create a severe hamper on the edge detection process since the algorithm will also consider the kinetic motion of raindrops as an edge. Hence, the system has to either be able to filter out raindrops and only allow the required image to pass through or remove them using a post-processing technique. The best way to remove raindrop edges is using an NMF filter to remove low-frequency rain streaks and the canny edge detection algorithm which can remove high-frequency rain streaks.
- Intelligent crossroad traffic management and control system which monitor traffic by placing photoelectric sensors at a distance from traffic lights to monitor traffic moving towards it and calculate relative weight of each road.Priority is given to the road that is overcrowded.RFID system can be to open up traffic in emergency situations like the passing of presidents, ministries and ambulances vehicles.This System can be integrated with moving cameras and could be stored in database or cloud, which could be accessed via a web-based interface.
- Traffic Monitoring and Vehicle Tracking is done using Roadside Cameras using digital image processing.Calibration can be done automatically or manually.Initially, the camera is calibrated by obtaining road geometry then removal of shadow.Moving vehicles are segmented from the original input images.Using an a-P filter increases performance and helps in extracting all traffic parameters.
- Edges are boundaries of an image. Edge detection helps in removing useless data, noise, and frequencies from an image. Canny's edge detection algorithm performs better in most cases and is able to detect both strong and weak edges.Sobel method, Prewitt method, and Roberts method provide low-quality edge maps relative to the Canny's edge detection algorithm.
- High-Density Traffic Management is performed using a process called Image background subtraction, here both image processing and embedded systems are used in monitoring.Transport system can be equipped with sensors to communicate about the traffic condition.Video-Based monitoring of traffic can be used in traffic monitoring, surveillance, and helps understanding traffic pattern.The density of Vehicles on road is calculated by Background Subtraction of the reference image from the image obtained by the camera.We can also track, count the number of vehicles and speed detection is also done by taking a ratio of distance traveled by cars and frame rate to classify vehicles into heavy vehicles(trucks, bus) and low vehicle(cars, bikes). Our algorithm has a linear complexity and is consistent in the size of a video frame, frame rate and the density of detected traffic.

3. Problem Statement

The number of vehicles using roads in most metropolitan cities is increasing by a large factor every day. Inefficient management of traffic can lead to a wastage of time, increased pollution levels, wastage of fuel, cost of transportation and stress to drivers, etc.
Hence, a system has to be designed to avoid the above casualties thus preventing accidents, collisions, and traffic jams. In some situations, emergency vehicles like ambulances and fire-engines are not able to pass through because of heavy traffic congestion. This delay can be fatal and lead to a loss of lives. There are a number of factors contributing to the increased congestion in traffic in the previous years; a few of them are listed below.

1) The slow speeds of freight vehicles.

2) There is a 9.08% annual increase in freight volume and 10.76% increase in other vehicles, but the increase of roads is only by 4.01%

3) The extra waiting time for signals even when no congestion exists.

With so many factors contributing to the increase in congestion, it is important to think of new and efficient methods of clearing congestions. The traditional signaling system is unable to produce a significant effect in curbing congestion of vehicular traffic.

4. Proposed System

Fig. 1. depicts the architecture of the proposed system. The steps involved in implementing the proposed system are given as follows:
Step 1: Video Acquisition
Step 2: Split video into frames
Step 3: Image Pre-Processing

Image Pre-Processing Involves
Step 1.Image Resizing
Step 2. Grayscaling
Step 3. Canny Edge Detection
Step 4. Template matching.
Step 5. Calculating the percentage of match of images.
Based on the results of the template match, store the count of cars in each frame.Adjust the signal light according to the number of cars detected.The data is sent to the ThingSpeak cloud dynamically.

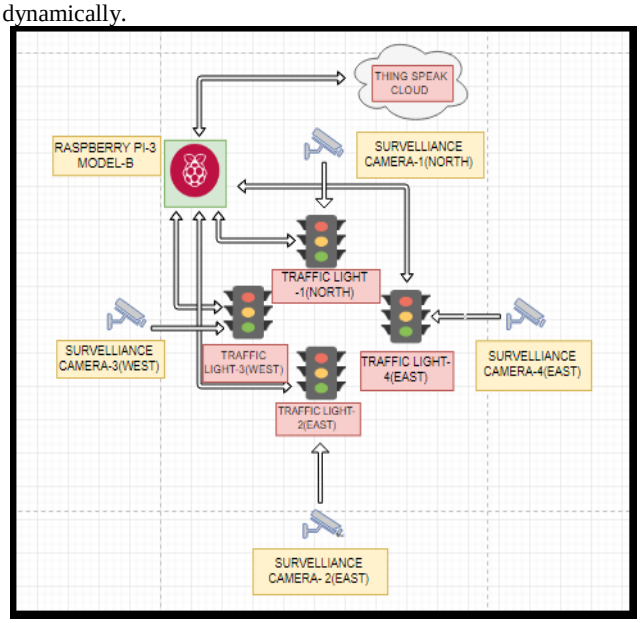

Fig. 1: Architecture of the proposed system

5. Image Pre-Processing

A. Image Acquisition

Image acquisition refers to the process of acquiring the image from a suitable source. In this project, the source is a camera(preferably CCTV, to avoid additional setup). The image is obtained by converting a video into multiple frames and processing each frame at a time. The image resolution can vary from 720*480p to 1280*720p based on the camera used.

B. Image Resizing

The acquired image is scaled down from the original resolution into 400*300p. This increases the frame rate of the image acquisition and hence helps the system avoid latency issues. Image scaling brings a level of consistency to the images acquired by converting input from all camera sources into a single resolution. It reduces the number of edges detected by ignoring minute details. Image resizing is primarily required when the system's processor cannot handle a large density of pixels at high speeds. Based on the algorithm that is used, image scaling can produce highly differentiated results as compared to a non-resized image.
new height = Previous height/Previous width*new width
new width = Previous width/Previous height*new height .

C. RGB to Gray-Scale

A grayscale image is one where each pixel depicts only its intensity. Hence the colors of each pixel are restricted to shades of gray ranging from white to black. The image is converted from RGB to grayscale during the acquisition process. Grayscaling refers to the process of converting an image from RGB into black and white. Multiple functions in OpenCV can work only if a grayscale image is passed as a parameter. Canny Edge detection also uses grayscale images. The edge detection algorithm returns white for all detected edges and black for the rest of the image.
OpenCV uses the following formula to convert a pixel from RGB to grayscale:
$Y \leftarrow 0.299R+0.587G+0.114B$
R, G, B denote the values of the red, green and blue colors of a single pixel in an image.

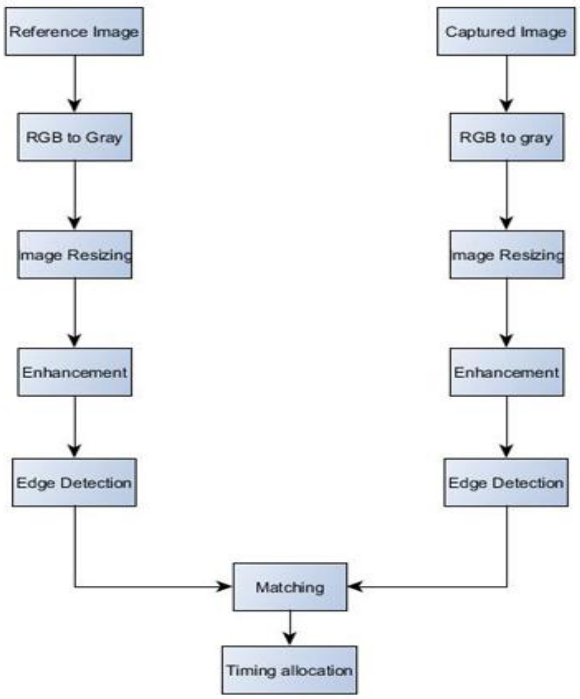

Fig. 2: Flow Chart of image processing

D. Canny Edge Detection

We choose it since Canny edge detection has optimal quality, is Simple and Single edge point response.
The Canny edge detection algorithm follows a set of basic steps:

Step 1: input image with a Gaussian filter to remove noise and unwanted details and texture.
Step 2: Calculate the gradient magnitude and angle images.
Step 3: A non-maxima based suppression is applied to the gradient magnitude image.
Step 4: Hysteresis

E. Template Matching

Template matching is a technique in image processing which is used to find the regions of an image that match (are similar) a template image (patch).It is a brute force method where we compare two images pixel by pixel along the height and width of an image. Steps involved in this are :
Step 1. Obtain source and template image.
Step 2. Sliding: Move the template image pixel by pixel in both x and y directions.
Step 3. Calculate a metric at each location
Step 4. Using this metric, deduce how good or a bad of a match is formed between the images.
Step 5. The brightest location indicates the best matches

F. Calculating Match Percentage

Step 1: The match_percent function is used to calculate the match percentage of two input images.
Step 2: It takes the reference image and the camera input as parameters. Usually, an image of the empty road with no vehicles is taken as a reference image.
Step 3: Image captured by the camera, which will contain vehicles on the road at the current instance is taken as the path image.
Step 4: Both the images are converted to grayscale and the resolution is reduced to 400*300.
Step 5: Grayscale ensures that an image is in black/white and no other color. The dimensions 400*300 ensures that both images are similar in terms of their heights and widths.
Step 6: Canny edge detection algorithm is used to detect edges in both images. The number of edges in the reference image is represented by white pixels in the image.
Step 7: In the reference image, the edge pixels are found bt calculating the number of white pixels(dots).
Step 8: In the path image, the number of white pixels is found by counting all white pixels excluding the ones present in the reference image.
Step 9: The match percentage is found by calculating the ratio between the white pixels in the reference image and the sum of the number of white pixels of cars in the sample image and the total number of white pixels in the reference image. This ratio is then multiplied by 100 to obtain the final match percentage.
Step 10: Return the match percentage
Match % = (_a)/(_a +_b) *100
_a = edges part of sample image and not part of reference image .
_b = Total edges in reference image
The algorithm being used will also work in corner cases where we have a single heavy vehicle like trucks and other single small vehicles like bikes or hatchback cars, priority should be given to the smaller ones since its logical that cars or bikes take lesser time and they occupy less space.The match is purely based on traffic densities and not on the count of vehicles on the road.
We obtain the density of all the sides.A surveillance camera is used to implement the algorithm and give priority.The system also has the capability to store data on a ThingSpeak cloud server which will help with data storage and further data analysis.

In Fig. 2. shows how the entire big picture works for a simple scenario consisting of two lanes.Each of the lanes is monitored by a surveillance camera.

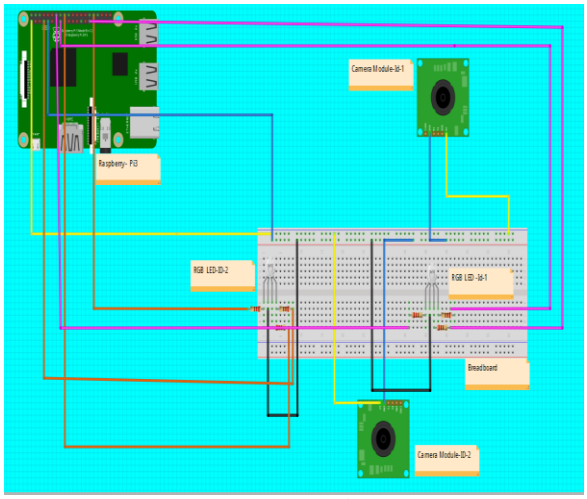

Fig. 3: Circuit Diagram for two lanes

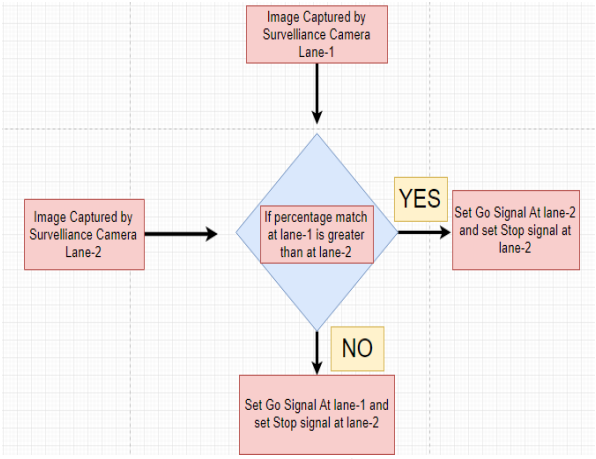

Fig. 4: Block Diagram for two lanes

In the above setup, Raspberry Pi is a system on chipboard.RGB Led is used to demonstrate traffic signal. The camera module is powered and tethered to pi. Camera module 1 acquires an image of first lane and Camera module 2 occupies image of the second lane.Image processing is done images and priority to the signals are decided.The most populous lane is given first priority.

6. Software Requirement

A. Open CV

OpenCV (*Open Source Computer Vision*) is a python supported library of multiple pre-defined functions that are aimed primarily at real-time image processing. It was developed initially by Intel followed by further support from Willow Garage and then Itseez (which was later acquired by Intel). The library free for use under the open-source BSD license. It is also cross-platform and supports multiple deep learning frameworks including Torch/Py Torch, Google Tensor Flow and Caffe. Open CV can run on the following desktop operating systems: Linux, macOS, Windows and other open source BSD systems like FreeBSD, Net BSD, Open BSD. Open CV is also available on the following mobile operating systems: iOS, Android, Maemo, BlackBerry 10. OpenCV is available for an end user through their official releases which can be found on SourceForge or from platforms like GitHub where one can obtain the latest versions. OpenCV uses CMake which is an open source, compiler independent build process management software.

B. Python

Python is a high-level interpreted programming language used for general-purpose programming. Python's design philosophy puts a heavy emphasis on code readability. It uses whitespace and indentation to provide a structure to the program. It enables a simple and clear programming methodology for both small and large-scale applications.
Python has an automatic memory management system and is considered to be a dynamic type system. It includes and supports multiple object-oriented, imperative, functional and procedural programming paradigms.
Python interpreters can work on multiple operating systems. CPython, the reference implementation of Python, is an open source software which is constantly updated by community based development. CPython is maintained by a non-profit community called the Python Software Foundation.

C. ThingSpeak Cloud

ThingSpeak is an open source API and application that is used for storing and retrieving data from things connected in an network using the HTTP protocol. A ThingSpeak cloud implementation enables functions like logging from sensors, real-time location tracking applications, and a social network of things with status updates. It provides aggregation, visualization and analyzing of live data streams in the cloud. ThingSpeak also provides methods to visualize the data posted by your sensors in real-time. ThingSpeak can also perform online data analysis and data processing as and when it comes in.
IoT protocols simplify the process of sending data to the ThingSpeak cloud from device sensors. Sensor data is visualized in real-time. Data can easily be aggregated into the system on-demand from the user. MATLAB can be used to make sense of your IoT data analytics . ThingSpeak also supports communication of data and third-party external services like any social media that exposes its API.
The following services are provided by the ThingSpeak cloud which makes it suitable :

- Data is sent privately to the cloud. Sensors are present everywhere. They can be found in homes, smartphones, city infrastructure, automobiles, industrial equipment and multiple other sources. Sensors can detect, measure and transmit data of external variables like temperature,humidity and pressure etc received from sensor. This data is transmitted to the controller either in the form of a numerical value or an electrical signal.
- Analyze and visualize data in the cloud with MATLAB. Storing data in the cloud is the easiest way to access, process, display and analyse data. Multiple online analytical tools provide facilities to explore and visualize data. Using this tools,we can identify trends in data. Data can be visualized in the form of plots, charts, and gauges.
- Trigger a reaction. Acting on data can range from being simple to highly complex. It could be something as simple as activating an alarm when the temperature being measured goes above 80° F. Intricate actions such as controlling a motor when the water level in a tank drops below a fixed limit and can also be used in any complex systems. Another application can be found in using remote controlled devices.

7. Hardware Requirements

A. Raspberry Pi-B Model-B

The Raspberry Pi is a system on chip developed by the Raspberry Pi Foundation. Its primary goal is to promote learning of basics of computer science in schools. Here, we use a Raspberry Pi 3B+ model controller for controlling the traffic lights according to the data collected from the surveillance cameras.

B. Pi - Camera

The Raspberry Pi Camera Module v2 has 8 mega-pixel resolution and can be attached to Raspberry Pi microcontroller . Its lens is focused and fixed.
The 8 mega-pixel sensor is capable of producing static images at a resolution of 3280x2464 and also supports 1080p, 720p and 640x480p video at 30/60/90 frames per second.Its a tiny board, and dimensions are 25mm x 23mm x 9mm. It weighs 3g and can be connected to the Raspberry Pi using a short length ribbon cable. We use it in our system as a prototype of a Surveillance Camera.

C. Power Adapter

Raspberry Pi is a single board small computer which can be powered using this power adapter which a normal phone charger.

D. LED - Lights

Light emitting diode (LED) are the small lights which require low energy to light up are here used to resemble the traffic signals to demonstrate the real-time regulation of signals according to vehicle data from the microcontroller.

E. Data Cables

Data cables are used to transfer the data from the Surveillance camera to the Raspberry Pi.The camera is tethered to the pi allowing it to host a webcam server on port:8080.

8. Experimental setup

Initially, GPIO pins of Rpi are connected to the Red, yellow, and green color LED lights, which indicate stop, start and go signal of a single traffic light.We have 4 traffic signal for 4 individual lanes at the intersection of roads.
These are connected Raspberry Pi - Model-B to pin:-
signal 1 ->LED is connected GPIO PIN Number 2,5,37
signal 2 ->LED is connected GPIO PIN Number 15,13,11
signal 3 ->LED is connected GPIO PIN Number 26,24,22
signal 4 ->LED is connected GPIO PIN Number 40,38,36
All signal are connected are grounded.
.There are 4 Surveillance cameras connected via USB cable to Rpi3 and tethered to it.It can also be connected to wifi instead of tethering.The camera captures an image of a single lane and sends it to the pi for processing and calculation.

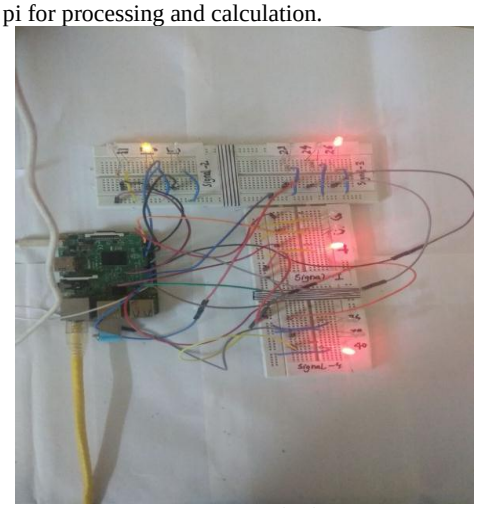

Fig. 5: Implementation for four lanes.

Surveillance camera host webcam service locally.We have taken reference image as an image of an empty road.A python script is written to find out the percentage of match between the image rendered from the surveillance camera to the reference image.

Here we first take the reference image and second path image and resize it an equivalent height of 400h and 300w.We perform grayscale and canny edge detection. We perform template matching over a region of image and count matching pixels. By following the above method we get the percentage match for all the 4 lanes. We can easily prioritize the lane with a large density of vehicles and operate the LED signals accordingly.

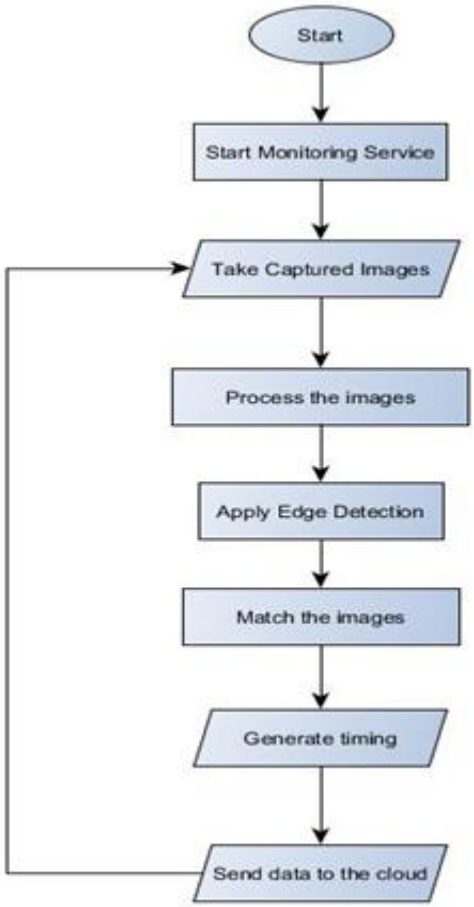

Fig. 6: Flowchart of working of Raspberry Pi.

Data obtained by percentage of match is sent to cloud.The python script uses write key generated by cloud to perform a connection. Http request and using POST method send values of feild1,feild2,feild3,feild4,feild5,feild6 to the thingspeak cloud server.Cloud Server listens to the request and stores it.Thingspeak provides us with Matlab tools to perform analytics, draw graphs and also covert data to JSON, CSV, and XML, any format. After obtaining match percentage we make the priority of all available signals. Flowchart show depicts the entire process.
Figure.5(a) is our implementation. We built a model of road intersection using thermocol and applied the same algorithm on to it by keeping different numbers of toy cars on different lanes. We got better results as we expected. Also, we analyzed the plotted traffic by uploading traffic density values onto the ThingSpeak cloud.
Some advantages are :
The following services are provided by the ThingSpeak cloud which makes it suitable :

- Installation Cost of the system**:** Compared to other systems, the setup cost is relatively low because we obtain a live feed from a series of surveillance cameras that are present at traffic junctions. Most major cities with traffic congestion issues already have surveillance cameras installed and ready to use.
- Use of Data analytics**:** The data obtained from traffic lights has a huge potential for usage as datasets in data analytics. This data can have multiple applications such as data visualization. This data can then be used for future construction and city development. For example, roads with higher traffic densities should be rerouted to

side roads which will reduce the congestion. Data analytics can also be used to detect the response time delay taken by emergency vehicles to reach the destination due to traffic congestion at the signals.

- Save fuel**:** This system reduces the fuel wastage in general by reducing the traffic congestion at signals. Since the system can efficiently reduce traffic congestion, the wait time of vehicles at these signals are also reduced and hence fuel is conserved.
- Maintenance Cost**:** The maintenance cost is restricted to the maintenance of the physical components such as the surveillance cameras and the Raspberry Pi as compared to other systems which uses a much greater number of hardware devices such as sensors. Using our system in a large scale will lead to a large saving on maintenance costs.
- Distance between Vehicles**:** Vehicles always remain at a distance to the vehicles directly in front of them. In a hardcoded system that uses sensors to obtain a vehicle count, small vehicles like bikes that fit in these spaces are considered as full-sized vehicles. On the other hand, our system checks for traffic density and not a vehicle count, hence corner cases like bikes don't create an issue.

9. Issues Faced by System

A. Night-Vision

Standard cameras cannot process images during the night. Since there are no external sources of light unlike daytime, where the light of the sun provides enough light for the camera to record videos. An alternative to this problem is to use a night vision camera which works on the principle of infrared light. IR lights turn on when the external light level is low. IR light is not visible to the naked eye so it floods an area in light that only the camera can see. Another advantage of IR light is that it is usually monochrome hence the conversion to grayscale is a lot less expensive than daylight imagery. Also, most CCTV cameras and security cameras that are set up in traffic signals come inbuilt with IR lights.

B. Weather Issues

Rain can create a severe hamper on the edge detection process since the algorithm will also consider the kinetic motion of raindrops as an edge. Hence, the system has to either be able to filter out raindrops and only allow the required image to pass through or remove them using a post-processing technique. The best way to remove raindrop edges is using an NMF filter to remove low-frequency rain streaks and the canny edge detection algorithm which can remove high-frequency rain streaks.

10. Output

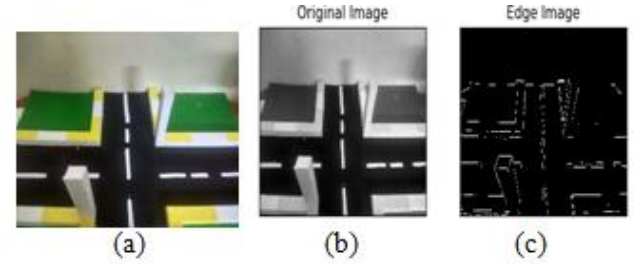

Fig. 5: (a) Reference image of an empty road (b) Gray Scale image of the reference image (c)Edge detection algorithm on the reference image

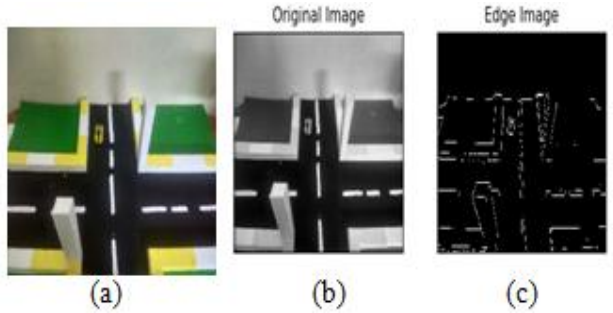

Fig. 6: (a) Sample-1 image (b) Gray Scale image of Sample-1 (c)Edge detection algorithm on Sample-1

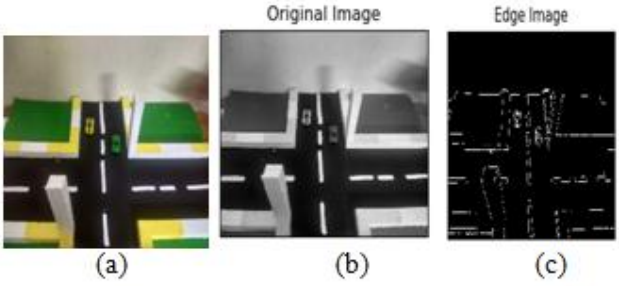

Fig. 7: (a) Sample-2 image (b) Gray Scale image of Sample-2 (c)Edge detection algorithm on Sample-2

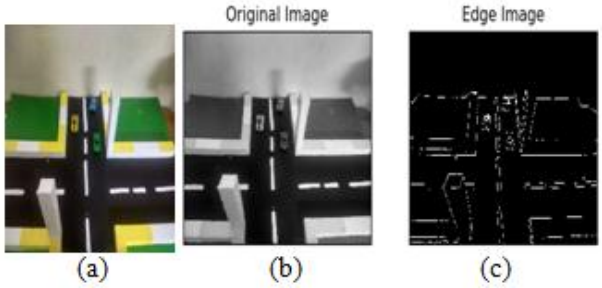

Fig. 8: (a) Sample-3 image (b) Gray Scale image of Sample-3 (c)Edge detection algorithm on Sample-3

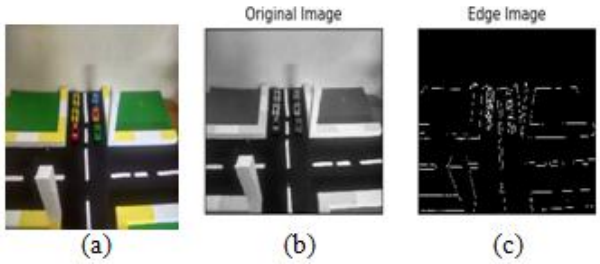

Fig. 9: (a)Sample-4 image (b)Gray Scale image of Sample4 (c)Edge detection algorithm on Sample-4

Table 1. The ratio of whites in each lane

Images (Sample images at 4 different lanes)	Ratio	Percentage	Signal Priority
SAMPLE -1	0.5160	51.60	4^{th}
SAMPLE-2	0.5091	50.91	3^{rd}
SAMPLE-3	0.5042	50.42	2^{ND}
SAMPLE-4	0.4889	48.89	1^{ST}

a. Ratio of whites in each images

Table 2. Match Percentage of whites

Images (Sample images at four different lanes)	No of whites in Reference image	No of whites part of path image but not in the reference image	Ratio
Sample -1	3215	3428	0.5160
Sample-2	3305	3428	0.5091
Sample-3	3371	3428	0.5042
Sample-4	3672	3428	0.4889

b. Match percentage of whites in each image

Fig. 5(a)depicts the reference image.Image of an empty road is taken as the reference image.Fig. 5(c)depicts the image after applying Canny edge detection we get 3428 white counts in the reference image.If we assume all lanes are designed same ways then we assume this image as a reference for all sample images of different lane obtained by a surveillance camera.
Fig. 6(a).depicts sample image obtained Camera-1 of lane-1.Count of only whites for 1 car is 3215 and excluding rest.Fig. 7(a). depicts sample image obtained Camera-2 of lane 2.Count of only whites for 2 cars is 3305 and excluding rest.Fig. 8(a). depicts sample image obtained Camera-3 of lane 3.Count of only whites for 3 vehicles is 3371 and excluding rest.Fig. 9(a). depicts sample image obtained Camera-4 of lane 4. Count of only whites for 6 cars is 3672 and excluding rest
From Table. II.it is very clear that:-
Priority is given 1st to sample-4 at Signal-4
Priority is given 2nd to sample-3 at Signal-3
Priority is given 3rd to sample-2 at Signal-2
Priority is given 4th to sample-1 at Signal-1

11. Future work

1. We can apply machine learning or template matching algorithms to detect emergency vehicles like ambulances and fire engines to let them pass as soon as possible. Providing priority to such vehicles which will save a lot of lives and property.
2.We can also enable the system to detect number plates of all the vehicles and extract the registration number of vehicles. So that, if needed it can help the Police and other authorities to track criminals by identifying the number plate of the vehicle in which criminal is escaping.
3.In future obtaining match percentage data of individual signals, we can perform detection of a special vehicle like an ambulance, data analysis on data stored in the cloud.And come up with an approach to that it gets green signal for all traffic light in its path.

References

[1] Brinda.R. B.Namratha Venkatesh Murth , B. M. Ramya , Faculty Advisor: Dr. Vijaya Prakash A M4" Edge Detection (Image Processing) Smart Traffic Control" international journal of innovative research in electrical, electronics, instrumentation and control engineering vol. 3, issue 11, november 2015
[2] Taqi Tahmid and Eklas Hossain, "Density Based Smart Traffic Control System Using Canny Edge Detection Algorithm for Congregating Traffic Information" 2017 3rd International Conference on Electrical Information and Communication Technology (EICT), 7-9 December 2017, Khulna, Bangladesh
[3] Arjun Raja and R.Swarnalatha, "Smart traffic signal by digital image processing using PCA algorithm and edge detection techniques" Journal of Engineering and Applied Sciences 12(22): 6076-6082, 2017
[4] Ahmed S. Salama, Bahaa K. Saleh , and Mohamad M. Eassa, "Intelligent Cross Road Traffic Management System (ICRTMS)," Eighth ACIS Int. Conf. On Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, Qingdao, August 2007, pp. 819- 822.
[5] Vikramaditya Dangi, Amol Parab, Kshitij Pawar & S.S Rathod," Image Processing Based Intelligent Traffic Controller" Undergraduate Academic Research Journal (UARJ), ISSN : 2278 – 1129, Volume-1, Issue-1, 2012
[6] Yao-Jan Wu, Feng-Li Lian, and Tang-Hsien Chang, "Traffic Monitoring and Vehicle Tracking using Roadside Cameras" 2006 IEEE International Conference on Systems, Man, and Cybernetics October 8-11, 2006, Taipei, Taiwan.
[7] Zhang Jinglei, Liu Zhengguang, "A Vision-Based Road Surveillance System Using Improved Background Subtraction and Region Growing Approach," Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing
[8] Siyal MY, B S Choudhary, A K Rajput, "An Image Processing Based Approach for RealTime Road Traffic Applications", Manuscript received May 30, 2005.
[9] Kunfeng Wang, Zhenjiang Li, Qingming Yao, Wuling Huang, and Fei-Yue Wang, Fellow, IEEE, "An Automated Vehicle Counting System for Traffic Surveillance", Manuscript received June 11, 2007. This work was supported in part by the MOST Grant 2006CB705500, NNSFC Grants 60334020, 60621001, CAS Grant 2F05N01, 2006 CAS Presidential Special Scholarship Grant, and SDAS Research Grant 2005006.
[10] Mamta Juneja , Parvinder Singh Sandhu "Performance Evaluation of Edge Detection Techniques for Images in Spatial Domain" International Journal of Computer Theory and Engineering, Vol. 1, No. 5, December, 2009 1793-8201