Smart traffic control with ambulance detection

Varsha Srinivasan[1], Yazhini Priyadharshini Rajesh[1], S Yuvaraj[2] and M Manigandan[2]

[1] Student, Department of ECE, SRM Institute of Science and Technology, Chennai, India
[2] Assistant Professor (O.G), Department of ECE, SRM Institute of Science and Technology, Chennai, India

Corresponding Email: varshu.md@gmail.com, yazhini321@gmail.com, yuvasivasanthi@gmail.com, manigandan.m@ktr.srmuniv.ac.in

Abstract. The problem of urban traffic congestion is constantly spreading. The increase in traffic is due to the growing number of vehicles and the limited expansion of roads. We propose a system for reducing traffic congestion using image processing by detecting blobs and tracking them. The system will detect vehicles through images instead of using electronic sensors embedded in the pavement. We also plan to provide a suitable solution for emergency vehicles stuck in traffic to clear the route by using Bluetooth, thus assuring timely help to those in need.

1. Introduction

One of the many problems that the world faces with increased population and rapid growth in the number of vehicles is traffic congestion. In countries such as India, the rate of road expansion is just one-third the vehicular growth rate. Statistics show that the current annual growth of vehicles is around 11% while the annual road extension remains to be only around 4%. The effects of increased traffic congestion are many. Congestion hinders economic growth through delayed services, wastage of fuel and adversely affects the environment. Studies say that in one day, traffic congestion causes wastage of 2.5 lakhs of liters of non-renewable fuel.

Our project focuses on the severe impact caused by traffic congestion on the emergency vehicle transportation system. In places such as India and Thailand where the road width and length prove to be impossible to create a separate lane for emergency vehicles, it is difficult for ambulances to navigate through the traffic.

Existing ideas include controlling a traffic light using timers for each phase or employing electronic sensors to detect vehicles [1] and produce signal that cycles. Fuzzy Logic Networks [2] are also used in traffic systems but in such cases dispute over real-time traffic systems proves to be restraints even though the system is very quick. These systems however require the presence of traffic policemen during the hours of peak traffic. Also, above methods do not cater to the necessities of emergency vehicles such as ambulances with lives at stake.

The main objectives of our project are as follows:
• Calculate the vehicle density based on image processing techniques to detect vehicles.
• To regulate the traffic signal timings based on a threshold value of traffic.
• Implementing a safe and reliable method to ensure emergency vehicles can meander through gridlocked roads thus saving the seconds that count.

Advantages of using Blob Detection are exact vehicle detection is possible, easy to implement and the method is fast. Canny edge detection isn't used because while the method is suitable for balancing noise and preserving edges [3-4], it has a problem with detecting the exact location as the gaussian smoothing blurs the edges making it tough to detect the location of the vehicle. It also requires complex computation which is time consuming and not suitable for real time image processing.

2. Methodology

Our project consists of three phases.
1. Vehicle Detection and Counting
2. Ambulance Detection
3. Decisions Based on Data

2.1 Vehicle Detection and Counting

Following are the steps involved in this phase:
1. Camera Positioning
2. Image Subtraction
3. Blob Detection
4. Blob Analysis
5. Blob Tracking
6. Vehicle Counting
7. Data Processing

Procedure:
Step 1: Camera Positioning is a crucial step in detecting vehicles. The camera is to be positioned such that it covers the entire lane. For example, in the below videos we tested, the camera is approximately set at height of 25 feet and 23 feet respectively. For the software to detect the incoming vehicles, the camera is to be positioned at an optimum height i.e., approximately within 25 feet. The angle is chosen such that at a certain distance is covered and the threshold line is distinctly visible.

Base Angle = $7°$ and Rise $=7.62$

$$\text{Base} = \frac{7.62}{\tan(7)}$$

$$\text{Top angle} = (90° - 7°) = 83°$$

$$\text{Diagonal} = \frac{7.62}{\sin(7)}$$

Figure 1. Camera Calibration

Step 2: Image Subtraction is where the frames from the video are compared with one another to find the difference between them. The difference is then used to detect blobs.
Step 3: By using the image subtraction results, changes in the two frames are noted. If there are any changes, those are the moving blobs i.e., vehicles. The remaining is the background which are stationary and are excluded as they are not moving objects.

Step 4: Analysis of these blobs are required. If at all any pedestrians are caught in the video, they mustn't be considered as blobs. For this, the size of the blobs is taken into consideration. If they are too small, these objects are ignored.
Step 5: After they are analyzed, the vehicles are tracked. A threshold line is formed which is used to calculate the number of cars. The line is drawn at an optimum level next to the traffic signal. Only when the vehicle crosses the line, it is counted.
Step 6: Vehicle Counting keeps a track of the number of cars crossing the threshold line. The car count is important to estimate the density of the traffic.
Step 7: The car count is used to find if the density is high or low and is used to alter the timings of the traffic signal. A threshold is set which forms the maximum number of cars, say around 40, before which the traffic is too high.

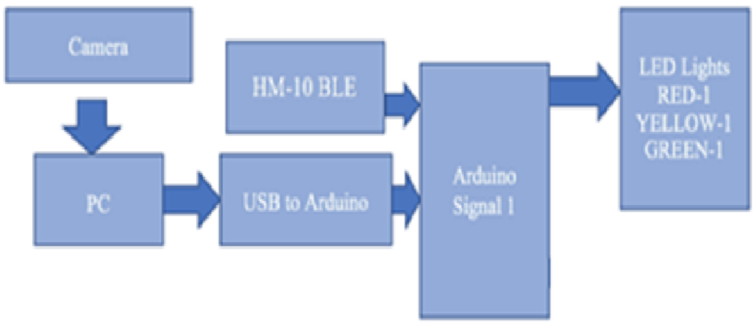


Figure 2. System Model Diagram

2.2 Ambulance Detection

Detecting the Ambulance: This phase involves a Bluetooth module and a phone whose Bluetooth is active. Once the ambulance is near the signal, the person driving the emergency vehicle can send a command to the Bluetooth module thereby guiding the traffic signal to change accordingly.
However, there is a security-accessibility trade-off in this system. To overcome this, we should make sure the code with which the ambulance can connect to a traffic signal is unique and is replaced every 24 hours. This would make the system more secure and reliable.

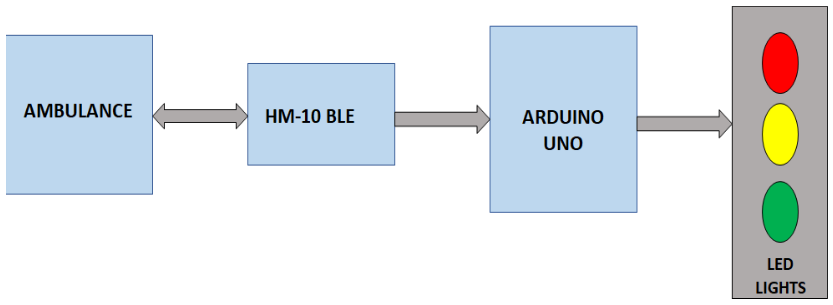


Figure 3. Ambulance Detection Model

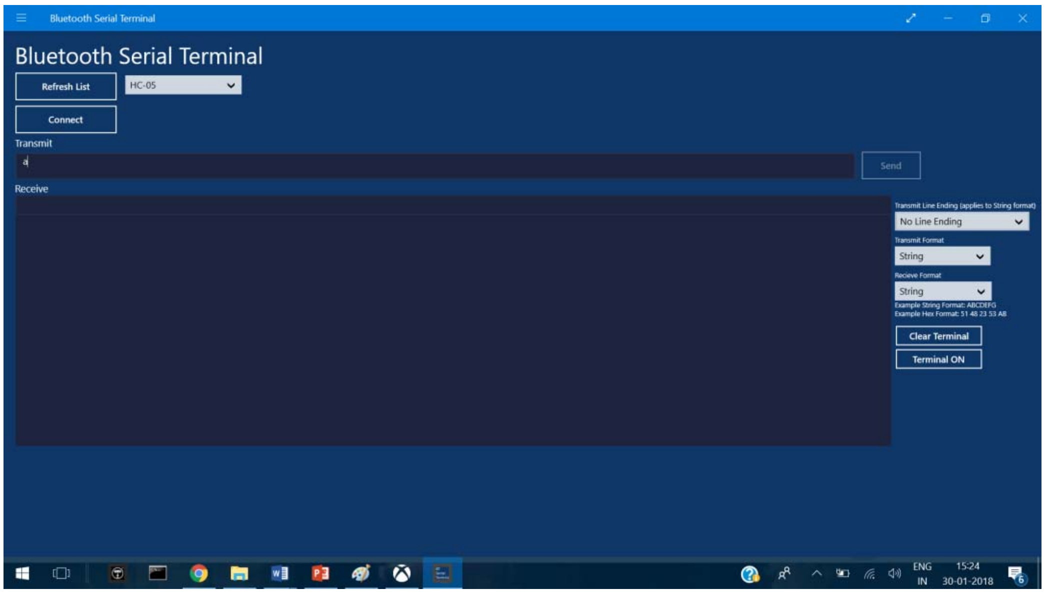
2.3 Decisions Based on Data

The processed information is passed onto an Arduino and decisions are made accordingly.

- If the car count exceeds the threshold level set (which changes at every signal depending on daily car count), the data given to the Arduino is ‘f’ i.e., full. Once the Arduino receives it, it changes the timings of the LEDs accordingly.
- If the car count remains to be less than the threshold level set for a pre-determined time (no data is passed onto the Arduino), it automatically sets the green signal.
- If an Ambulance is present and a code is received by the Arduino, for example, “switch”, it automatically switches on the green light. If the green light is already switched on, it delays the light for an extra 10 seconds.

3. Results obtained

We obtained the following results through our project. The Bluetooth interface used for commanding the Bluetooth module is as shown below:



Bluetooth Serial Terminal

HC-05

Transmit: a

Transmit Line Ending (applies to String format): No Line Ending

Transmit Format: String

Recieve Format: String

Example String Format: ABCDEFG
Example Hex Format: 51 48 23 53 AB

Figure 4. Bluetooth Interface from PC/Phone

The following images are the results obtained using OpenCV for processing a sample video file

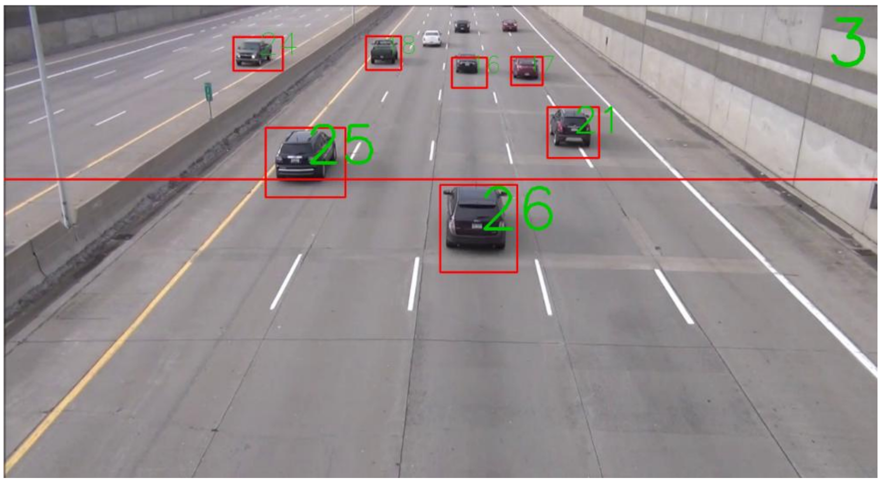


Figure 5. Car Count Displayed on Screen

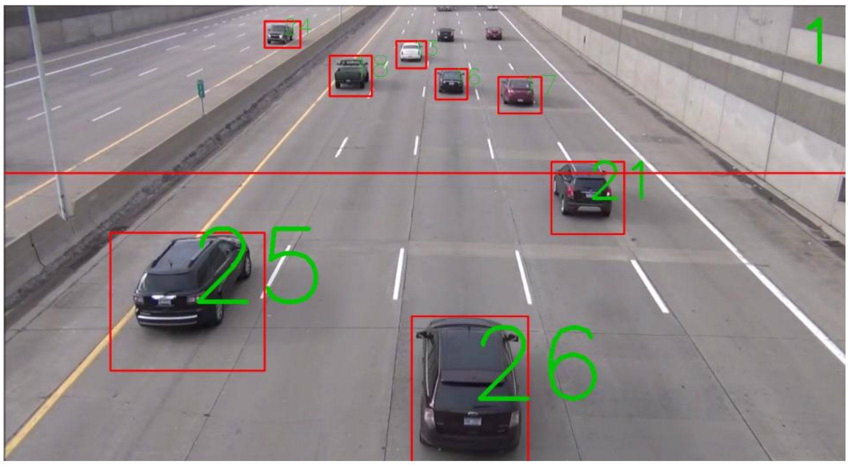


Figure 6. When the Count= 5, count initializes to zero

Table 1. Input and Output Analysis

Input	Output
Car Count remains less than threshold	If time elapsed for the red signal is more than 30 seconds, automatically turn green
Car count greater than threshold	Signal turn yellow followed by green for 20s
Ambulance sends out the command and signal is red.	Signal turns yellow followed by green for 30 seconds
Ambulance sends out command and signal is green.	Signal remains green for an extra 10 seconds.

The above-mentioned timings are chosen for the traffic in the sample video input file. However, for real-time traffic, by observing traffic patterns, we can calculate the time required.

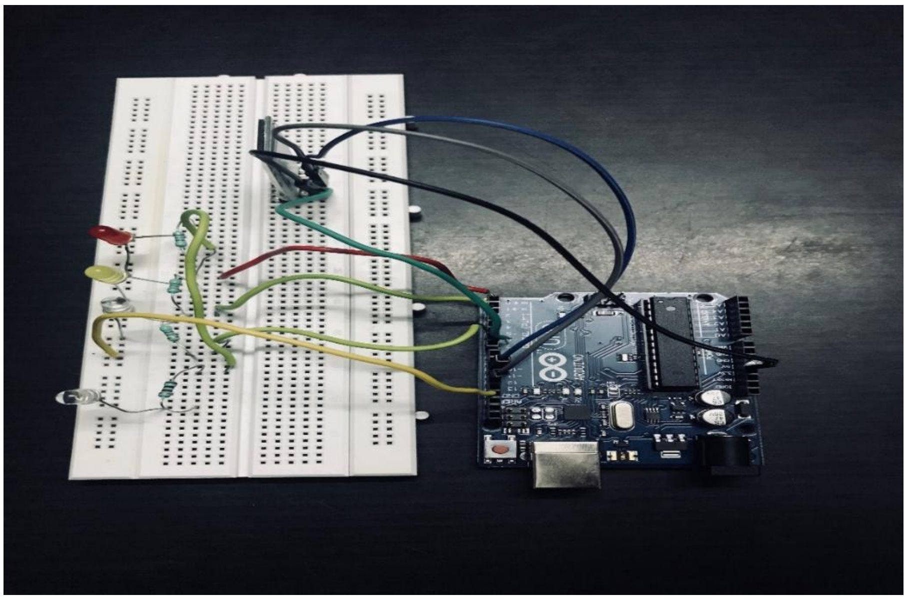


Figure 7. Traffic Lights Assembly

4. Conclusion

By following the above-mentioned steps and performing image processing, we were able to achieve the following results:

- Vehicle counting by choosing a suitable threshold count (for a sample video of a four-lane road and maximum traffic density 10 at a time, we chose the threshold to be 6) and altering the timing of the signals accordingly. When the traffic density exceeds the threshold, the duration of green light is extended by 20 seconds.
- When an ambulance is detected, and the signal is green, the timing is extended by 20 seconds and if the signal is red, the timing is extended for a longer duration of 40 seconds. This can be extended to any number of signals along the ambulance's path so that the emergency cases can be served immediately.
- The timing for the signals can be decided upon analyzing the traffic pattern in an area for a fixed amount of time and calculating the extended time for which the signal lights have to be switched on accordingly.

This idea can be implemented for a larger network by using encryption algorithms to ensure safety and stability of systems. Also, the extended time can be calculated by the system itself. By keeping records of traffic patterns and using an algorithm, the timing can be chosen according to the traffic patterns. Through this paper we have been able to present and implement a smart solution for emergency cases in traffic to give maximum preference to lives at stake.

5. References

[1] Patel R and Shah T 2017 *IJSETR.* **6,** 694-697
[2] Hawi R, Okeyo G and Kimwele M 2017 *Computing Conference* 450-460
[3] Uddin M, Das A and Taleb A 2015 *ICEEICT.* 21-23
[4] Srinivas P, Y L Malathilatha and Prasad M 2013 *IJCSIT.* **4,** 17-20
[5] Laganière R 2011 *OpenCV 2 Computer Vision Application Programming Cookbook* Pub. 106